Universitetet i Stavanger

DET TEKNISK-NATURVITENSKAPELIGE FAKULTET

MASTEROPPGAVE

Studieprogram/spesialisering: Applied Data Science	Vårsemesteret, 2021.. Åpen / Konfidensiell
Forfatter: Markus Antonius Marszalek & Tord Are Strømsnes	
Fagansvarlig: Aksel Hiorth Veileder(e): Kjartan Kloster Osmundsen	
Tittel på masteroppgaven: Engelsk tittel: The application of machine learning models for appointment prediction at a psychiatric clinic	
Studiepoeng: 30	
Emneord: Machine learning Data mining Helse Vest Prediction model No-show	Sidetall: ……76…… + vedlegg/annet: ……18…… Stavanger, ……14.06.21…….. dato/år

The application of machine learning models for appointment prediction at a psychiatric clinic

Master Thesis in Applied Data Science
by
Markus Antonius Marszalek
&
Tord Are Strømsnes

A collaboration between Helse Vest RHF &
Faculty of Science and Technology
University of Stavanger

June 2021

Acknowledgements

First of all we would like to thank all the professors for the last few years at UiS for being accommodating due to our situation as full time teachers with busy and rigid schedules. They have been understanding, flexible and pragmatic with the courses and the work requirements, especially after the pandemic hit.

For the thesis we would like to thank our supervisor Aksel Hiorth. We would like to thank for the assistance and clarifications we needed underway, and for his experienced eye and feedback on the thesis as a whole.

At Helse Vest RHF we would like to thank Kjartan Kloster Osmundsen for coming onboard as our co-supervisor, providing much needed expertise on the field and data we were working on. Thank you for your thorough and critical feedback, it really helped.

We would also like to extend our gratitude to the entire team at Helse Vest RHF for taking us on as students, and providing us with this opportunity at all. They have provided us with a lot of information and made sure we got off to a good start. They have given us a lot attention throughout the project, ensuring that we were on the right track at all times, while also pushing us further. We really enjoyed presenting our results, although at times it could be scary as well.

Fredrik Haugland deserves our thanks for reading and correcting our thesis providing us with an external opinion of our work.

Markus would like to apologize to his wife Trine for once again being in this situation, though he thanks her for being so encouraging and forgiving. Markus' efforts here is dedicated to his newborn son, Emil.

Tord would like to apologize to Markus for putting him in this situation.

For this thesis we acknowledge:
Helse Vest RFK for providing anonymous patient records from a psychiatric clinic, and laptops with access to their computer systems to work on.

Markus Antonius Marszalek and Tord Are Strømsnes
Stavanger, June 2021

Contents

Chapter 1

Introduction

1.1 Background:

Healthcare services is a crucial part of modern societies, where it's main role is to provide the necessary healthcare for the community, assisting those in need of medical attention. However, it is also an important establishment as a work-provider, providing work at all levels within the community. In the recent years, the COVID-19 pandemic has highlighted a lot of the challenges in running healthcare services as they have been put to the test. The gained attention to healthcare has increased the discussion and interest in how healthcare sectors are in terms of quality, cost, and efficiency. A challenge that has always been present in healthcare services are patients not showing up for their appointments, *no-shows*, and late cancellations. Patient no-shows adds to the healthcare clinics costs, because this is time lost that the trained medical workers could potentially have used to treat other patients instead.

The thesis project was proposed by *Helse Vest Regionalt helseforetak* (RHF). Helse Vest RHF has the overall responsibility for the healthcare in the counties of western Norway and they provide the healthcare for about a million residents (Helse Vest RHF, 2017). It consists of several larger hospitals and numerous smaller clinics both in rural and urban areas. Helse Vest RHF has annually 230.000 planned patient-healthcare-contacts, appointments which are cancelled either due to no-shows or cancellations ahead of time. There are many factors which contribute such an outcome, and there are several strategies that attempts to reduce no-shows. All patient interactions lead to large amounts of data surrounding the appointments and the patients. The gathered data is useful for statistical analysis about the patients and the healthcare-providers and its services. It can assist in further understanding the interaction of patients and the day-by-day operation of clinics, and the possible areas of further improvements. Statistics is useful when working with the data analysis, but there are often limitations and challenges with traditional statistics, and in these cases modern machine learning can outperform the more traditional methods. In the last few decades machine learning has gained momentum and traction as technology and algorithms have improved. Classical statistics has traditionally been limited to few dozen input variables and tractability of computations (Bzdok et al., 2018). Modern machine learning can be used to generate knowledge (and predictions), where statistics previously has focused on describing data and testing hypotheses (Soria-Comas and Domingo-Ferrer, 2016). Pursuing the goal of improving the services from healthcare-providers, machine learning can be applied to the

large amount of data available in order to generate more knowledge about the patients and appointments.

1.2 Motivation:

The motivation for the thesis is to investigate ways of identifying which patients are at high risk of not showing based on the analysis and prediction of various methods of statistics and machine learning (ML). With ML algorithms and large quantities of data, one can statistically analyze some of the tendencies and investigate which features/attributes that are the most significant of no-show patients. Based on this, it is possible to develop models that can be used by healthcare-providers to forecast and predict likely no-show candidates. It has been done in the past in a few papers with different approaches and strategies, some of which are summarized in Dantas et al. (2018). Ideally a machine learning model would be reliable enough in its prediction for health-providers to apply follow-up routines to the patients who are likely to not show up. A strategy could be in the form of an additional reminder via text or phone calls, or some other form of encouragement or incentive practice depending on the patient group. Reducing no-shows will reduce downtime, and will thereby improve waitlines and costs. This could raise the overall quality and experience of the service for both healthcare providers and the patients. According to estimates made by Helse Vest RHF for a specific clinic, they estimated possible cost reduction of around NOK 615.000,- when using a sufficiently accurate model for predicting no-shows. The numbers are based on a wait line cut of 50% with a model with 70% accuracy for predicting no-shows.

Another important part of this thesis is the challenges around gathering patient records and the restrictions surrounding it. There is as previously mentioned a lot of information which is gathered as data in these appointments. The analysis done on the data needs to be in line with general data protection regulation (GDPR) as a lot of the information is personal data. That means a lot of the information surrounding the clinic and the patients is either blurred or generalized to achieve anonymity. Another important part of the GDPR is that the patients also have the right to openness and transparency of the applications of their data. They have the right to know how their personal information is being used in the model. This means a model using their data cannot be a “black box”. A model must be explainable to be able to justify the actions based on the data driven models’ prediction.

The data we worked with is real data supplied by Helse Vest RHF for a psychiatric clinic for adults. According to Mitchell and Selmes (2007) patients with mental health problems miss about double the amount of appointments compared to other medicinal fields. This was based on data in the United Kingdom but this tendency is also seen elsewhere. A psychiatric clinic for our project is thereby of particularly of interest, as the data is based on a patient group with higher rates of no-shows. The provided information about the clinic has been anonymized and limited in the amount of features in order to reduce the risk of inference, linkability and re-identification of the people involved. The supplied data was originally for the years 2016-2018 in the early development, but at later stages it was expanded to also include 2019 as well to serve as a test set. most information about the clinic is limited to that which is seen in the data.

1.3 Objective:

In the thesis we will investigate the gathered data itself and the circumstances surrounding it. With the data we need to address challenges such dependencies, anonymity, sample size, and imbalance, all at once. Dealing with these challenges will be important as they are a vital step in creating a decent machine learning model. The main emphasize will be on finding a model that can predict no-shows with high enough performance at the selected Helse Vest RHF psychiatric clinic. We will investigate the best model types and the best features to pair them with, in order to handle these challenges. We also need to find decent ways of measuring performance and evaluating models as it is a part of the challenges in the underlying circumstances of the data, i.e. the imbalance in the dataset between people showing to appointments and those who do not. We need to ensure a model that is simple and transparent enough for medical workers to able to understand it at some level, so they can explain it to patients. This will ensure that the patient's GDPR rights are fulfilled. We will also consider and discuss some of the challenges surrounding the deployment and application of a model in practice. All of these goals are mostly summarized in the following objective:

> Investigate relevant features, explore potential machine learning models, and look into the practical challenges of the application of an appointment prediction model at a psychiatric clinic.

Chapter 2

Theory

2.1 Personal Data Protection and anonymity

The EU General Data Protection Regulation (GDPR) was implemented in May 2018, and in Norway later the same year in July. The goal was to restrict and regulate how personal data is handled and processed. It sets strict requirements for how information is gathered and used, but also underlines the responsibilities around the security of personal data. Businesses must request permission for data gathering and have to disclose the personal information gathered if requested by the individual. It gives private individuals more control of the data being gathered, and transparency around the application of it. This has been necessary with the increased growth of big data application which allows further knowledge gathering with data mining based on all the data being generated by patient interaction and transactions.

An important aspect of the GDPR is how one handles the security aspect. Anonymized data are exempt from the obligations of the GDPR, as part of recital 26, which implies that the principle of data protection should not apply to anonymous information. For information to be anonymous, the information is not just de-identified, but the information is anonymized enough so that individuals can not be re-identified by any reasonable means (Goldsteen et al., 2020). There are several ways of achieving anonymity such as k-anonymity, and e-differential privacy, both with their pros, cons and caveats, some of which is discussed in Soria-Comas and Domingo-Ferrer (2016). Information about individual is at risk amongst others due to quasi-identifiers and linkabillity across datasets. Quasi-identifiers are variables like; zip-code, date of birth, and sex, which together can be used as a tuple to identify an individual. A selection of such attributes can be enough to identify someone although it is not an explicit unique identifier like a social security number. This was among other shown by Samarati and Sweeney (1998) with an example re-identifying anonymous medical data with the help of voter-lists. It shows how large enough tuples identifiers of information can identify individuals. When records are sorted into smaller sets based on similarities, they are also at risk of linkabillity if you have other independent resources which can be used to identify individuals. This puts some records at risk of re-identification if the anonymization process and algorithm is not thorough enough. With increase of big data and data mining, another challenge has been membership inference and attribute inference attacks where certain sensitive features may be inferred (Goldsteen et al., 2020). This all highlights the im-

portance of anonymity of the data being handled, and of the obligations present. The effects on privacy methods for anonymizing personal data on machine learning can result in poor accuracy. Performance changes have been compared on different types of machine learning models after k-anonymization by (Wimmer and Powell, 2014). The K in K-anonymization is a measure of to what degree the data has been anonymized. If the K is 10, it means we only can identify down to groups of minimum 10 people. The data used in this thesis has undergone anonymization before being handed over, which if efficient enough is exempt of GDPR regulations. However, to ensure that the information is safe, and to reduce the risk of any security breaches of personal information; all the data was kept on machines that are a part of Helse Vest RHF data systems.

GDPR is especially important when looking at the applications which is based on gathered data about a subject, and where decisions are made because of the predictions. Article 22 from the GDPR states (EU, 2018):

> *The data subject shall have the right not to be subject to a decision based solely on automated processing, including profiling, which produces legal effects concerning him or her or similarly significantly affects him or her.*

It points to an important ethical aspect, where one should not be judged with severe repercussions and consequences solely based on an automated process. This is one of the more challenging aspects of the practical applications of a prediction model. Thereby the possible applications of the final model is somewhat restricted and limited to what has been pre-approved by the legal team at Helse Vest. That is why the model currently is planned for application in predicting no-shows to trigger additional follow-up.

2.2 Definition of a no-show and errors

When predicting no-shows it is necessary to set some of the definitions. We break down the problem into a bi-classification for either *no-show* or *show*. Late cancellations are sometimes in practice as inconvenient as no-shows. The threshold for late-cancellations which are considered as no-show based on the clinic practices is chosen based on clinic practices (elaborated in later chapter about method). Similarly to the setup of Huang and Hanauer (2014), a basic definition for the classification of appointment i is given as prediction $\hat{M}_i$, where:

$$\hat{M}_i = \begin{cases} 0, \text{ Show predicted} \\ 1, \text{ No-show predicted} \end{cases} \tag{2.1}$$

The way prediction is made differs from model to model, but generally it will be the one with the highest probability. In our project, we define no-show as 1, and show as 0. The reason for the switch is to emphasize what prediction the model is attempting, which is identifying the no-show candidates. This means based on the actual observed outcome of the appointment M, we can define an error as:

$$E_i = \begin{cases} 0, \ \hat{M}_i = M_i \\ 1, \ \hat{M}_i \neq M_i \end{cases} \tag{2.2}$$

This gives us the possible errors in the predictions, but there is an important distinction in the type of errors. Type I error (also called a false positive) is when a patient is classified as a no-show, while being show. Type II error is a false negative, which is a no-show being classified as a show. This distinction might seem arbitrary, however, (Prati et al., 2009, p. 360) describes an example in medical diagnosis where most patients, but a small percentage have cancer. Here it is important to classify those with cancer correctly because if a cancerous patient is classified as healthy it could have lethal consequences. A healthy patient classified as sick will just result in a more extensive check-up before being declared healthy. To define the type of errors, N_{show} is set to be total number of actual show patients in the training set, and $N_{no-show}$ as the actual no-show patient population (Huang and Hanauer, 2014). The errors are thereby defined as;

$$\text{Type I error: } \frac{\Sigma E_i^{show}}{N_{show}} \tag{2.3}$$

$$\text{Type II error: } \frac{\Sigma E_i^{no-show}}{N_{no-show}}. \tag{2.4}$$

Where minimizing type II is the type of error that should be prioritized when optimizing the model. Although the potential cost of these errors can be defined and weighted.

2.3 The Bias-variance tradeoff

When determining which model performs best we want to find a model with an error as small as possible. The mean squared error is often used as a measure for error when working with machine learning algorithms, but for this problem it might not be the best solution. The mean squared error takes the true value, subtracts the predicted value and squares it, but since this problem is of a classification sort, there are more intuitive ways of explaining the error that occurs. More ways have been suggested to describe this relation for classification problems (Domingos, 2000).

When describing the error of a model there are often two terms that are used to explain it all; Bias and variance. It is imperative to understand these terms when understanding how a model performs. In figure 2.1 you can see a visualization of how bias and variance work. Bias means that center of our predictions has shifted from the optimal target. We want the bias to be as low as possible, this means that the chance of hitting our target is better and the likelihood of favoring one class over another goes down. When the bias increases it means we might get a skewed prediction. The variance is measure of how collective the predictions are. High variance means that small changes in our data might lead the prediction to end in another class or we can say that precision is lower.

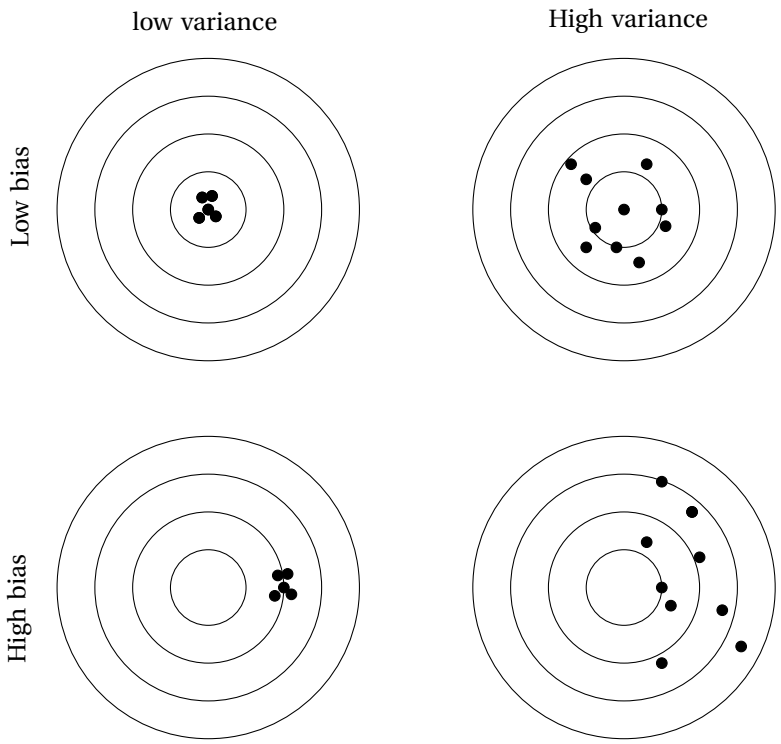


Figure 2.1: An example illustration of the bias-variance tradeoff

The predicted error has an additive relation between the bias and variance. We will do as Domingos (2000) and define a *loss function* $L(t, y)$, which measures the cost of predicting y when the true value is t. For our classification problem a zero-one loss function could be proposed ($L(t, y) = 0$ if $y = t$ and $L(t, y) = 1$ otherwise). We then define the expected loss of our model as $E_{D,t}[L(t, y)]$. D is here the training set, which can be divided up in multiple training sets. For some loss functions, like our one-zero loss function this decomposition of $E_{D,t}[L(t, y)]$ holds:

$$\begin{aligned} &E_{D,t}[L(t, y)] \\ &\qquad = c_1 E_t[L(t, y_*) + L(y_*, y_m) + c_2 E_D[L(y_m, y)]] \\ &\qquad = c_1 N(x) + B(x) + c_2 V(x) \end{aligned} \tag{2.5}$$

Where $V(x)$ is the variance, $N(x)$ is the noise in our data and $B(x)$ is the bias on our model. The y_* represents the optimal prediction, y_m represents the main prediction and y is all the predictions for the different training sets.The noise is impossible to eliminate, so it is often not taken into the equation, but it is important to note that it is a factor. The point is to reduce both the variance and the bias of our model, but in reality it is hard to reduce one without increasing the other. This is what is meant with the *bias-variance tradeoff* and the optimal point for this tradeoff varies from application to application (Domingos, 2000).

What this mean is that the bias of the model can be defined as 1 if the prediction does

not agree with the true label and zero otherwise.

$$Bias = \begin{cases} 1, \text{ if } y \neq t \\ 0, \text{ otherwise} \end{cases} \tag{2.6}$$

We can also set up a definition for the variance in this case which can be seen as the probability of predicted label not matching the main prediction.

$$Variance = P(y_* \neq t) \tag{2.7}$$

In practice, when working with machine learning and artificial intelligence (AI), this is important to take into consideration. High variance in our model could often translate to overfitting. If the model performs better on the training data, than it does on the test data, this could indicate high variance. The model could in this case have learned the training data too well, and will now follow the noise in the training data which is not necessarily present in the test data. What we mean by this is that the model has not learned the general pattern for the data, but the exact pattern of the data in the training set. On the other hand, if we try to make our model too general, it might not learn the correct patterns and maybe show prejudice towards some classes. This is referred to as an undertrained or underfitted model.

2.4 Feature selection

In machine learning there are often big datasets with a lot of features. When many features are present we sometimes want to limit the amount of features used for creating the model. This might seem counter intuitive because we usually would like as much information as possible when training a model, but there is actually a couple of reason why this, in some cases, is a good idea (Garreta and Moncecchi, 2013):

- For some methods, for example decision trees, we reduce number of features used to refine the model at each step. When a model does this it is possible that irrelevant features suggests correlation where there really is none. Some features might also be highly correlated to each other and give little new information. The model might also be prone to overfitting.
- The other reason is that a large amount of features could greatly increase the computation time without making the model any better. A model based on fewer features might also make it more generalizable, which means the model might perform better on a wider variety of problems. An example could be if Helse Vest RFH would like to extend the model to be used on other clinics as well.

The idea is to remove features that might be of less value to the model. This compresses the data, but good feature selection can also contribute to improved learning accuracy, avoid overfitting, reduced learning time and to simplify learning results (Langley et al., 1994; Langley, 1996; Zhao et al., 2010; Saeys et al., 2007). Compared to other algorithms, such as compression or projection (Principal componenent analysis), feature selection does not alter the original data, but chooses a subset from the data, and this could also increase interpretability for a domain expert (Saeys et al., 2007).

Normally more features means decreased classification error, but in cases where data is sparse, more features can cause increase in variance and further entail increased error (Hua et al., 2004). For the problem in this thesis, there are not enormous amounts of data, but there are not too many features present either. This will therefore not be a notable issue in this case, but if the problem was to scale with data that incorporate more features, it could be something to consider.

The library Scikit-learn comes with a high variety of methods to be used for feature selection, from ExtraTreesClassifier to varianceThreshold. To figure out what works best it has to be tested on the data. A manual approach could also be considered. For example could a correlation plot between the features be explored and some decisions could be made from this, but with many features it could be hard to get an accurate overview.

Another thing to consider is the presence of features that are not available in the prediction moment. If a variable is not available at the time of prediction, it is not useful when training the model either.

Zhao et al. (2010) arguments that two key concepts are important for feature selection: Relevance and redundancy. A features relevance is due to a couple of reasons, one of them is about the feature being strongly related to the class and the other is that the feature forms a subset with other features and this subset is strongly related to the class. Redundancy is about the features property related to other features. If a feature provide similar prediction power as other features it might become redundant.

For our thesis there are too many feature selection methods to test them all and compare them across the multiple models trained. Some of the feature selection methods are more popular or widely used than others and we will look into a couple of them, more specifically the ANOVA method and the extra trees classifier. This is what Saeys et al. (2007) calls a filter and an embedded method. There are other variants of these and we have an entire third category he classifies as wrappers that will not be investigated in this thesis, both because these methods can be computationally costly and they are at risk of overfitting data, which is not commendable when working with ensemble models (Saeys et al., 2007).

2.4.1 ANOVA

A popular way of finding the best possible features for a dataset is the ANOVA, or ***An****alysis* ***o****f* ***Va****riance*, method. ANOVA is a univariate feature selector (Drotár et al., 2015). This means that each feature is examined individually to find how their relationship with the target value is. This make the interpretability of the feature selection fairly simple and some studies show that they do not necessarily underperform in contest with more complex solution (Haury et al., 2011; Saeys et al., 2007).

Being a univariate feature selection we have the advantages that it is fast, scalable and independent of the classifier, but it also ignores interactions with the classifier and feature dependencies (Saeys et al., 2007).

2.4.2 Extra Trees Classifier

The Extra Trees Classifier, or extremely randomized trees, is a tree-based ensemble method for classification (Geurts et al., 2006). For our thesis we will use this method for feature selection. The method is similar to the Random Forest in that it is based on building trees,

but for the Extra Trees Classifier all tree models are made from the entirety of the sample-size and attribute, and cut-point choice are highly or entirely randomized. Making just one tree in such a fashion often results in high variance, but make a lot of them and you can lower the variance without increasing bias to much.

When using this classifier for feature selection, the algorithm can measure something called the *Gini*-impurity. This is usually used to measure how often a randomly chosen element would be classified incorrectly, but instead we can use this as a measure for feature importance, rank the features in our dataset and choose only the features that gives us the lowest probability of wrongful classification. In figure 2.2 you see that each node in the tree is assigned a Gini-value and each of these splits are done based upon one of the features in the dataset. Then you can average over all the trees and choose the features the Gini-values implies will give the lowest error rate.

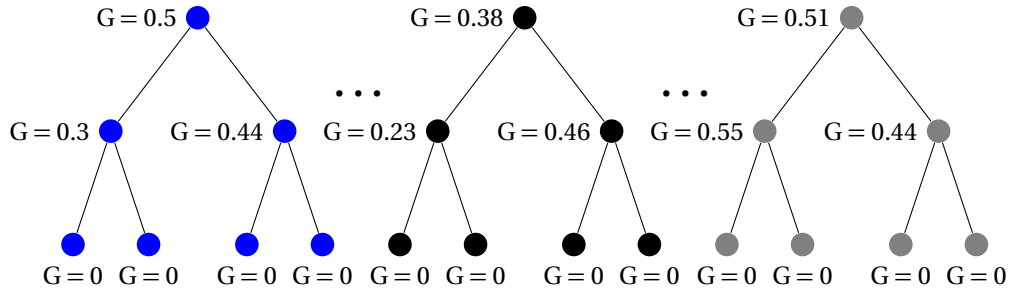


Figure 2.2: The extra trees classifier draws up multiple trees and the Gini-impurity is calculated for each feature in the trees. From there the features with the highest Gini-values are chosen for the further exploration. The G is short for Gini. The trees do not have to be symmetrical as in the figure, but when a leaf node has G-value 0 it stops splitting.

2.5 Imbalanced data

In machine learning we are often provided with the "curse of imbalanced data", where many of our datapoints belong to the same class (Lemaître et al., 2017). What often happens in these cases is that the model will prioritize the class with most samples, because it tends to skew towards higher accuracy, but we might be more interested in finding patterns for the smaller classes (Prati et al., 2009). To handle this imbalance some effort have been made and a couple of alternatives for dealing with the problem has been procured: (i) Change performance metrics, (ii) random undersampling of majority class, (iii) random oversampling of minority class, (iv) change algorithm or (v) make synthetic samples of the data (Boyle, 2019). A combination of these could also be a possibility. There are no single best alternative to use, so some experimentation is required to figure out what is best for your project (Brownlee, 2020b).

Dealing with this problem we are looking into the `imbalanced-learn` library, which is an open-source python toolbox made for this exact problem (Lemaître et al., 2017). This toolbox is also fully compatible with *scikit-learn* and therefore compatible with the `sklearn`-library

2.5.1 Performance metrics

Often we speak of the accuracy of a model, but this might not always be the best method for evaluating performance of our model. The basis for most of the evaluation metrics is the confusion matrix.

- True negative (TN): Values that are correctly classified to 0
- True positive (TP): Values that are correctly classified to 1
- False negative (FN): Values that are really 1, but classified to 0
- False positive (FP): Values that are really 0, but classified to 1

Table 2.1: A simple 2x2 confusion matrix

	Predicted: 0	**Predicted: 1**
Actual: 0	True negative (TN)	False postive (FP)
Actual: 1	False negative (FN)	True postive (TP)

According to Ghoneim (2019) accuracy is defined as the ratio of correctly labeled subject to the whole pool of subjects.

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.8}$$

When we work with imbalanced dataset, the accuracy becomes practically useless (Thai-Nghe et al., 2010). This is because a model prioritizing accuracy could potentially classify almost everything to majority class, while the interesting thing is to figure out what characterizes the data in the minority class.

Thai-Nghe et al. (2010) further goes on to define precision as the ratio of correctly classified as 1 to all who have been labeled 1.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{2.9}$$

Recall is the ratio of correctly labeled 1 to all who are actually label 1.

$$\text{Recall} = \frac{TP}{TP + FN} \tag{2.10}$$

F1-score is a value calculated by combining precision and the recall.

$$\text{F1-Score} = \frac{2 \cdot \text{Recall} \cdot \text{Precision}}{\text{Recall} + \text{Precision}} \tag{2.11}$$

Specificity-metric is also mentioned as an important score for the performance of our model, but in reality it is actually the same as the recall value, but for the other label. So if the target is 0 and 1 you could switch them and calculate recall and get the same value. We calculate how many is classified correctly as true negative and divide on all who are actually true negative.

$$\text{Specificity} = \frac{TN}{TN + FP} \tag{2.12}$$

Combining more of these values can give a more accurate picture on how our model actually performs when predicting classes. Accuracy is a great measure when the data is symmetric or distributed homogeneously, but when data is imbalanced we can get a better picture using F1-score.

There might also be cases where there is acceptable that we classify something as false negatives, but not as false positives or vice versa. Then recall or specificity might give us better indications on how our model performs.

If what we want is to be confident that one class is classified correctly, then precision might give us the best measure for our data. In our case the recall performance might be especially important because we get a measure for how good our model is to classify the minority class.

2.5.2 ROC-curve

We often use specificity (False positive rate) and recall (True positive rate) in combination to make a ROC-curve (Receiver Operating Characteristic). Figure 2.3 shows an example. Indication of a good model is if the graph goes high up in the left corner.

The area under the ROC-Curve is called AUC (Area Under Curve). The closer this value is to 1, the closer we are to the perfect model. This would mean that all positive values are classified correctly and no negative values are classified as positive (Chawla, 2009). Still, if classifying positive correctly is more important than classifying false correctly, or vice versa, we might want to look at the specificity and recall individually and find what works best for our particular problem.

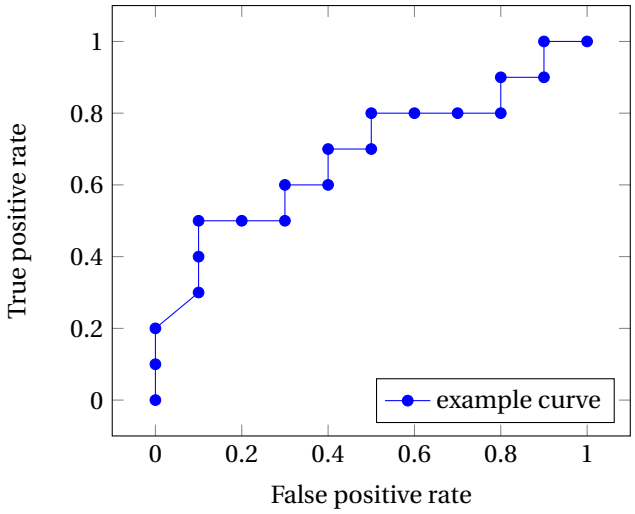


Figure 2.3: ROC curve, the area under the curve is often refered to as AUC (area under curve). The higher this value is, the better.

When deciding on our model we look to the ROC-curve and see that we would like to find a value for predictions that prioritizes true positive rate(TPR) over true negative rate(TNR). To

do this we introduce the G-mean metric (Thai-Nghe et al., 2010). The formula for calculating this is:

$$\text{G-Mean} = \sqrt{\text{TPR} \cdot \text{TNR}} = \sqrt{\text{TPR} \cdot (1 - \text{FPR})} \tag{2.13}$$

Where FPR is the false positive rate. The G-mean is a probability threshold value for the cutoff between classes. This metric tries to optimize the accuracy of each class while keeping them balanced.

2.5.3 Sampling strategies

When dealing with imbalanced datasets over- and undersampling has been introduced as possible solutions to the problem (Guo and Viktor, 2004; Weiss and Provost, 2003; Chawla, 2009). We also look into the use of Synthetic Minority Over-Sampling TEchnique (SMOTE), which is a way of making synthetic samples of the minority class based on what data already exists in this class (Mishra, 2017).

There are different ways of implementing these techniques. For our project we will look into the use of Random over- and undersampling strategies. Undersampling is a pretty simple way of handling imbalanced data, but it does not come without its drawbacks (Lemaître et al., 2017; Mishra, 2017). How do we know what data to remove from our dataset? We could potentially remove some datapoints that contains a lot of useful information for our model. The data could be processed manually, but this could lead to bias from the editor. Therefore we look into the random undersampler from the python `imblearn`-library. This library provides four different ways of dealing with the sampling problem: (i) under-sampling, (ii) over-sampling, (iii) a combination of both and (iv) ensemble learning (Lemaître et al., 2017).

Oversampling in turn can lead to replication of the same data over and over and lead to some weird results where certain data could be more influential to the model than what is the actual case .A combination of these methods is certainly also a possibility, but it is not easy to know what combination or what ratio of over and undersampling will give the best result for a particular problem.

Another thing to consider when sampling like this is that the balance between the classes will be different from the training set to the test set. This might cause some problems since most classifiers are built on the assumption that our test data is drawn from an equal distribution as the training data (Khalilia et al., 2011).

2.5.4 Changing the algorithm

One way of dealing with imbalanced data can be to explore different models. Some models are less suited to dealing with the problems of imbalanced data than others. Some models assumes well balanced classes and they can therefore tend to prioritize accuracy and end up neglecting the minority class, but in some cases the minority class could be the interesting part of the problem.

Knowing what algorithm to use for training the data is not always straightforward. There are so many alternatives to choose from: linear regression, logistic regression, naive bayes, decision tree classifier, random forest classifier, support vector machine and more. Some are more suited than others for dealing with the imbalance problem and some can be modified to work better with imbalanced data than they originally do.

One of the more popular choices when working with imbalanced data is the random forest classifier due to it being able to have a high degree of adaptability in the bias variance trade-off, so that we often get more of the minor classes correctly classified when making the trees a little deeper. Still, we have to be wary of not overfitting the classes. Combined with some of the sampling techniques it could also be interesting to see how a naive bayes classifier or xgboost could perform.

2.5.5 Create synthetic samples of data

Another interesting approach is SMOTE from the python library imblearn (we use this library also for over- and undersampling methods). This method oversamples the minority class by synthetically generating new data points based on the original data. What is important to remember when oversampling, with SMOTE as well, is to split the data into training and test-set before applying the method (Mishra, 2017). This is done to minimize the probability of the same data being present both in test and training set.

Over the years, different variations of the SMOTE-method has been developed. SMOTE itself works by selecting samples that are close to each other in the feature space (Brownlee, 2020b). Drawing a line between these samples (interpolate) you can take out a new sample along this line. Then you have made a syntethic sample that should be close to existing data in the specific class.

Other extensions to SMOTE could for example be Borderline-SMOTE and Borderline Oversampling. These two has the idea that we need more samples in the borderline between the minority class and the majority class, because this is where the samples that are difficult to classify exist (Brownlee, 2020b; Han et al., 2005). These and other methods exist, but it would be outside the scope of this thesis to test them all.

2.6 Model selection

Choosing a model is a particularly important part of the project. To find out which model will perform best there is some trying and failing involved, but we can also rely somewhat on what has been done before and what experiences others have had with the different machine learning algorithms. Least squares is a method not very well suited for a large multidimensional problem with imbalanced data. It is a method that draws a straight line or plane between the classes in a problem. This makes it a method that have problems with overlapping classes and can also be sensitive to outliers. On the other side we have Random Forest Classifier which have been proven to perform better on imbalanced data (Ghoneim, 2019; Khalilia et al., 2011). This might be because a decision tree often have high variance and can find small nuances in high granularity data, while averaging or voting over multiple decision trees could lower variance without increasing the bias significantly.

Another practical thing about random forest classifiers is that they are built on the idea of decision trees and therefore the interpretability is high compared to for example an artificial neural network (ANN). You can see how the logic transverses down a tree, but in for example a neural network it is not always as easy to understand how the decisions are made. It is frequently referred to as the black-box nature of neural networks and makes it hard for us humans to explain why the model makes a prediction, why some features are favored and how we can improve upon the model (Fan et al., 2020). In our thesis where we are dealing

with patients at a healthcare clinic, an ensemble tree method might be favorable because it is easier to explain and visualize, both for the practitioner and the patient.

2.6.1 Naive Bayes Classifier

There are a lot of different variations on the Bayes classifier and over the years they have found to perform well on different sorts of problems (Lewis, 1998). On some problems it can perform very well, but more complex problems with bigger training sets over the last year have seen some more complex models like ANN perform better. The reason for this is multifaceted, but in short ANN is a highly adaptable algorithm and on big datasets with many features it can find more subtle patterns in the data than Naive Bayes. Still we can find respectable results using the Naive Bayes method, especially for smaller datasets (Lewis, 1998).

By classifier we mean that the data is assigned to a class. If you look at figure 2.4 you can see a simple two-class problem with a defined decision border based on the Gaussian Naive Bayes classifier. If a new data point is introduced to this set, it will be assigned to a class based on what side of the decision border it is.

The Bayes network classifier is based upon the Bayes theorem, which for a multiclass purpose can be formulated as:

$$P(c_k|\mathbf{x}) = \frac{P(c_k) \times \prod_{j=1}^{d} P(x_j|c_k)}{P(\mathbf{x})} \tag{2.14}$$

Where c_k is the class and $\mathbf{x}$ is the entity being classified and x_j is the value of the j-th attribute. We want the posterior, $P(c_k|\mathbf{x})$, to be as high as possible. The feature vector will be assigned to the class with highest posterior. This form of of classification has as goal to minimize the number of errors and this perspective the name *naive Bayes classifier* come from.

Naive Bayes has some key features (Webb, 2010):

- training time is linear with respect to number of data points and number of features. Classification time is only affected by number of features. In other words, the naive Bayes is very efficient.
- The bias-variance trade-off often skews toward high bias and low variance.
- When new training data is available the model can be updated, so the naive Bayes will improve over time
- Posterior probabilities are directly computed
- Uses all attributes when predicting and is therefore insensitive to noise
- Also robust when classifying examples with missing values.

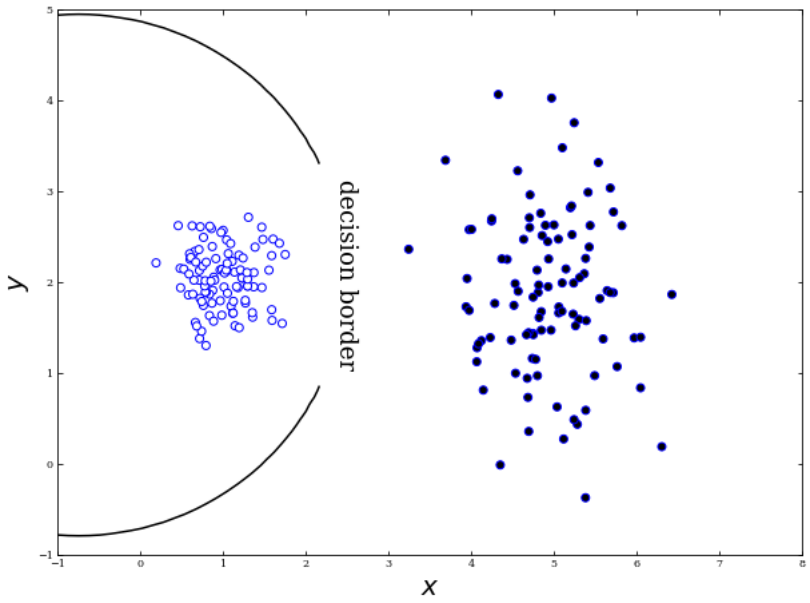


Figure 2.4: In this figure you see a simple example of the thought behind Naive Bayes. You can calculate the probabilities for the outcomes and draw a decision boundary between the classes. Everything on one side will be classified as one class and on the other side it will be classified as the other class. For n-dimensional problems it gets harder to visualize, but the idea is still the same.

Since the Naive Bayes Classifier is based on the Bayes theorem, which in itself is a pretty simple formula it has the advantage of being a model that is simple to explain, but for a multidimensional problem, like in this thesis, it could be hard to visualize.

2.6.2 Random Forest Classifier

Decision trees has long been a popular choice for making predictions, but it comes with some drawbacks. Often they come with low bias and high variance, which means that small changes to training data can make very different trees. A popular choice addressing this is to make many different trees and aggregate the results.

There are a couple of methods that are popular when constructing the different trees. One of the methods are boosting (Schapire et al., 1998) and another one is bagging (Breiman, 1996). Boosting is a method where each successive tree is given weight. At a given step the training examples that are classfied correctly will be given lowered weight while those classified wrong will have their weights increased (Liaw et al., 2002). In doing so, the idea is that we can combine many weaker classifier into a stronger one.

The other alternative, bagging (bootstrap aggregating), is where each tree is made independent from each other. Each of the trees is made independently by taking a bootstrap sample from the training data. Then a simple majority vote among the trees is taken for prediction. In figure 2.5 you can see what bootstrapping in itself is. If you have a dataset you

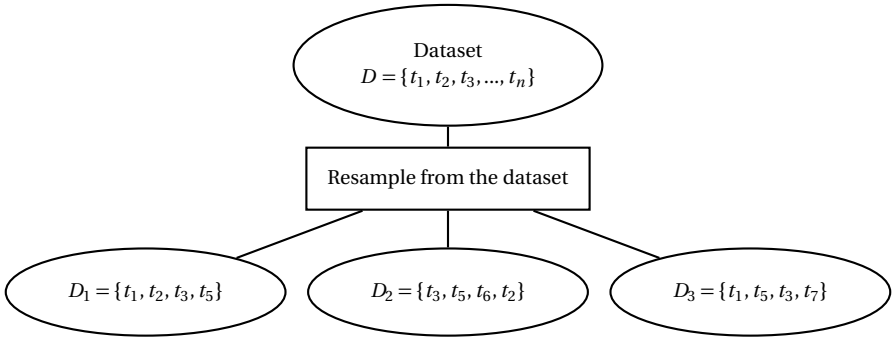


Figure 2.5: Example of bootstrapping. A randomized sample with reversal is subtracted from the dataset to create subsamples

can sample rows from that dataset without taking the data out and make multiple datasets from the first one.

This idea has been developed further into what we now know as Random Forest classifiers (Breiman, 2001). The difference here is that each node is split on the best variable among a subset of the features at that node. This gives an extra dimension of randomness to the making of the trees and it turns out this makes the classifier robust against overfitting (Breiman, 2001). A visualization of the Random Forest can be seen in figure 2.6.

Liaw et al. (2002) describes the algorithm for making a random forest as follows:

1. draw n bootstrap samples from the training data.

2. For each of the bootstrap samples grow a tree where you find the best split among a subset of the features. The splitting criterion can be defined in different way, but we will explore the use of entropy (Shih, 1999) and gini impurity (Khalilia et al., 2011).

3. predict by aggregating over the n trees in the forest.

Liaw et al. (2002) also have some notes for practical use. For example: The number of trees necessary for good performance is dependent on how many features is used to train the model. The numbers of trees needed increases when number of features increases.

Another thing is that we might adjust the threshold for majority vote when dealing with imbalanced data, which in our problem might be relevant to have in mind.

When making our model many hyperparameters for the random forest classifier has to be adjusted and tried. From number of estimators (number of trees), splitting criterion, depth of the tree and more. To find the best combination we will look into the use of a gridsearch which will be explained in section 2.7.

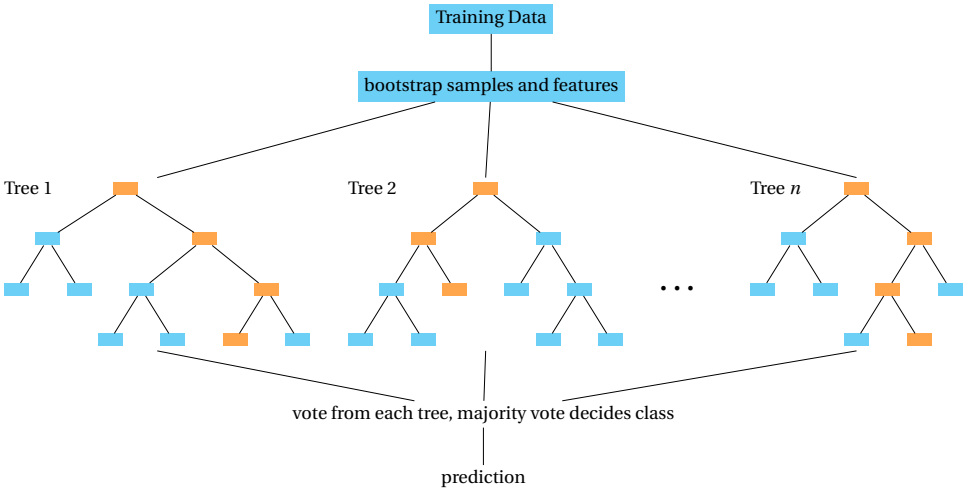


Figure 2.6: An example of a random forest. Many trees have been assembled and a vote between them is taken for which class the data belongs to. The orange color is the path through each tree for a single data entry. The end leaf contains the vote for which class it belongs to.

2.6.3 Tree boosting with XGBoost

As stated before, gradient boosting is a method where combining multiple weak learners into a better learner. A typical such weak learner is the decision tree. XGBoost stands for extreme gradient boosting and is a scalable machine learning method that has proven to perform at a state-of-the-art level for both classification and regression problems (Chen and Guestrin, 2016). For our project we will be using the python library `xgboost` that provide all we need and is integrated well with the `scikit-learn` package that we use for the other models.

A decision tree is, as explained in chapter 2.6.2, a classifier with high variance and low bias, but there are also different versions of the decision tree. In the xgboost package they actually have multiple variants of the splitting criteria for the decision trees. In the library we have the greedy algorithm, approximate algorithm, weighted quantile sketch and the sparsity aware split finding (Chen and Guestrin, 2016). For small datasets like the one in our problem the greedy algorithm will be chosen and for bigger datasets where we expect to find a lot of null- and missing values the sparsity aware method can save a lot of computation time. This is why the method is highly scalable and may be one of the better approaches to our problem. In a problem like this where the dataset will increase in size over time and as more data is available for training, the better our model might perform. There is also a possibility for weighting the data when using xgboost. We could either give weight to selected features or to more recent samples.

In the same way as the Random Forest Classifier the xgboost algorithm has high interpretability, because we can write out and show the trees that give our prediction and from there show patients how the model makes its decision. An example of how the model is trained is shown in 2.7. From the first tree misclassification errors is punished, so that the

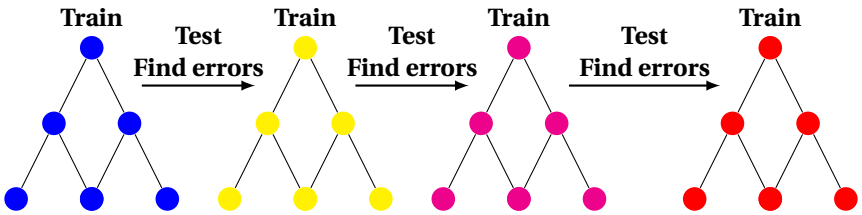


Figure 2.7: The point of boosting is creating new trees based on the last one. If you have cost function, the next tree will try to minimize errors based on this cost function.

next tree that is based on the first tree will hopefully perform a little better. When this approach is repeated enough the classifier should perform better.

2.7 Grid Search/ Cross validation

Many of the machine learning algorithms have some hyperparameters (will just call them parameters from here) that can be tuned. These make som changes to how the algorithms makes its decision and will affect the results to some degree. Therefore we have to figure out what valaues these parameters need to have for best performance in our model. Testing all possibilites is not realistic because it can be an infinite number of possibilities, so we have to do this for a limited set of possibilities.

A nice feature of the `scikit-learn` framework is the `GridSearchCV` class that can do some of the work for us. This method does cross validation and tests combinations of specified parameters to figure out the best possible fit. For more parameters we want to test, complexity of the search will increase and the time to figure out the best fit will increase exponentially (Garreta and Moncecchi, 2013).

K-fold cross validation is often used when determining the best model (Anguita et al., 2012; Jung, 2018). The idea of the method is fairly simple. By splitting the data into k-subsets we can use each of these subsets as the "target"-data and use all the other data to train a model with different parameters and then estimate the error in the model. An effort to visualize this has been made in figure 2.8. We see for each cross validation the red rectangle represents the temporary test set and all the white ones are the training data for each step of the cross validation.

This procedure will be executed for each subset and the parameters which gives the model with the best estimation, will be used to train the final model with all the data in our training set. How big the k should be there is no exact answer for, but if unsure it is often usual to use $k = 5$ or $k = 10$ (Jung, 2018).

Another regular method is the leave-one-out-cross-validation (LOOCV), where you use all data except from one to train a model and test upon that single point. A drawback with this is that it greatly increases computational time and there is no guarantee of better performance on our model.

Figure 2.8: Cross validation. The red rectangles represent the dataset that is used for testing the model (temporary test set) when trained on all the other four white ones on that line, based on the results for these the hyperparameters for the model is set. Then the model is trained on all the training set. The test set is used when all parameters are decided after the cross validation

2.8 Concept Drift

There is generally a challenge with machine learning models that are deployed where over time the model performance decays. The reduced performance of the model is the result of changes in the underlying data over time. The changes in the data shifts the model or the patterns of the target variable (concept), and can in turn change the outcome of the prediction of a target. This phenomenon is known as concept drift (Widmer and Kubat 1996). Gathered data will generally drift as there are changes that occur over time to the distribution of the data for the different features of the observed events. The temporal changes are sometimes inevitable and can be due to factors out of your control. The changes in the data might change properties or patterns of the target variable which a model is trained to look for, as the target variables properties evolve. It is hard to predict what the underlying changes in the data will be, and small changes in the data can propagate and shift the model in unpredictable ways. Concept drift is illustrated in the simplified example of an arbitrary classification model in the figure 2.9, where we see that the boundary of two different classes can shift over time because of changes in the distribution of the data. They are still similar, but as seen regions are now overlapping between the old and new observations.

As a result, patterns in the features for new observations might not align with the patterns in the predictions of a model trained on past data. New data might shift the models' concept or target variable when incorporated into a new model. This means new data is incorrectly predicted for a while and with a new model based on new data, old data based on the old concept is thereby likely to be misclassified. Both of these are scenarios that could reduce the performances of these models. The Netflix movie recommendation competition (known as Netflix Prize) is a practical example discussed by Žliobaitė et al. (2016). Some

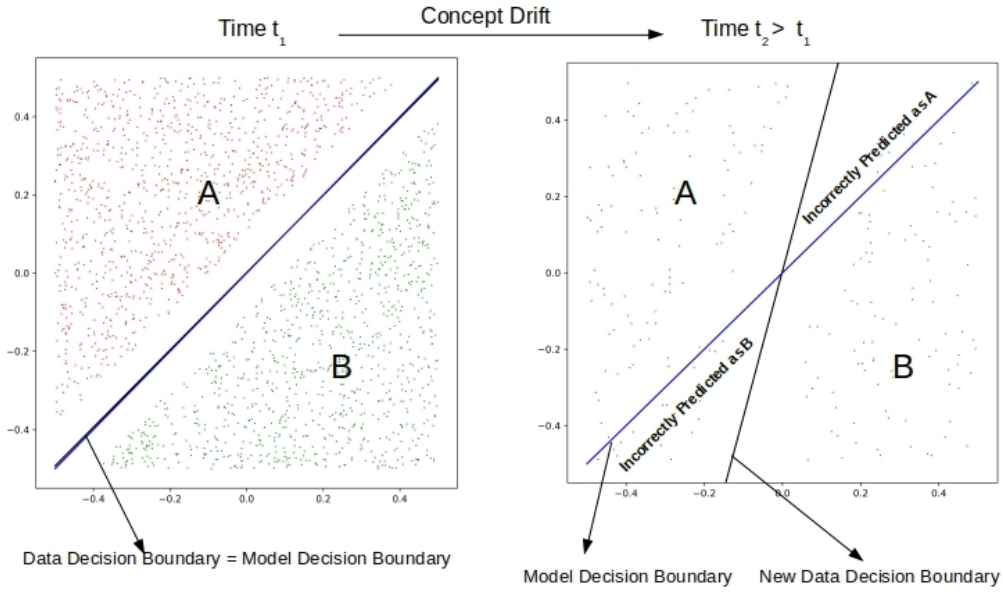


Figure 2.9: An illustration (Chilakapati, 2019) of how the region of two classes can shift due to changes in the distribution.

of the algorithms from that competition revealed the importance of seasonal patterns in product perception and popularity (IE. Holiday themed movies), but also user preferences that change over time independently (IE. Superhero movie fatigue). Concept drift can be observed in different ways, it is summarized and illustrated in figure 2.10. The changes to the concept or drift in the model can be; Sudden, Gradual, Incremental, or Reocurring. The colours indicate different classification for the target (concept), where we see sometimes everything can suddenly change after a certain time, or it can be a change over time, or something periodic.

An ideal model would be able to notice the shifts in the concept despite them manifesting differently and being unpredictable. The model would also have to be robust, not too susceptible to unforeseen changes in the data, while also being able to discern shifts from general noise in the data to actual drifts. It can be difficult to develop the perfect model capable of fulfilling all these criterions, as the interactions with all the drift and changes to the data and concept is a complicated interaction. There are algorithms developed to assist in detecting drift, for example the early drift detection method (EDDM) as described in Dong et al. (2018).

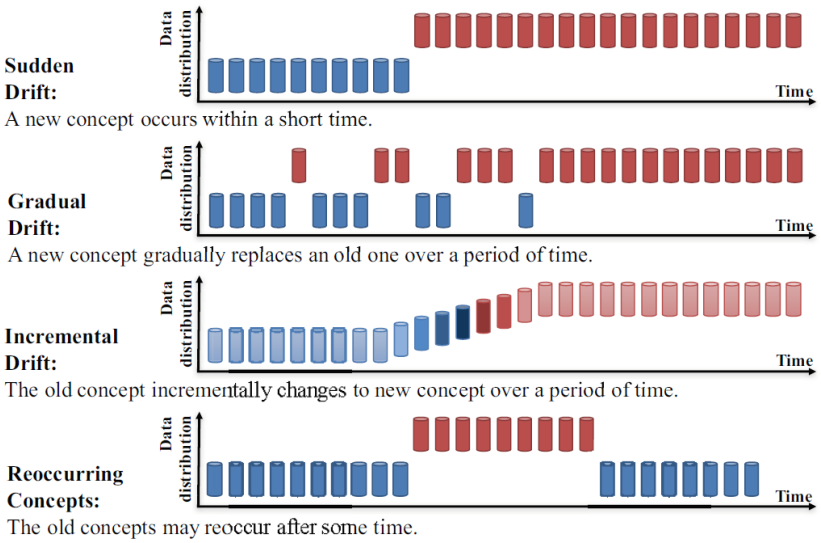


Figure 2.10: A illustration of different types of concept drift (Dong et al., 2018).

2.9 Summary of theory

From the legislation around the data to the underlying theory about building machine learning models, this chapter attempts to give a light introduction to many of the aspects surrounding this project. For the project three different models will be investigated to see how they perform on the data provided by Helse Vest. In order to do this, one must understand the circumstances surrounding the data, define the basic terms and definitions within this field, and understand how and which type of model to develop.

Before creating the models, the code will extract the most significant features using one of two feature selection techniques, ANOVA or Extra Trees Classifier. After this, one of three different sampling methods (Random under-/over-sampling or SMOTE) will be used to deal with the imbalance found in the dataset.

The Gaussian Naive Bayes, Random Forest Classifier and the xgboost-model are the three models that are explored. The two tree based models have the advantage of interpretability with printable representations of the trees used for prediction. The Gaussian Naive Bayes model is based on probability and could therefore be transparent in its own way.

When the model has been trained, the last step is to look at the performance of the model. This could be measured using many of the metrics described in chapter 2.5.1. However, there is not a single performance metric that alone can decide which model is best.

Chapter 3

Method

3.1 CRISP-DM process

For this thesis, the work process followed the *cross-industry standard process for data mining* (CRISP-DM or CRISP for short). The generic CRISP-DM process model is useful for planning, communication within and outside the project team, and documentation (Wirth and Hipp, 2000). As part of this process, the project was divided into to the main phases outlined below:

1. Business Understanding
2. Data Understanding
3. Data Preparation
4. Modelling
5. Evaluation
6. Deployment

These steps are not strictly chronological, and the work often goes back and forth between phases based on feedback as illustrated in figure 3.1. This workflow also serves as an overview and documentation of the workflow and process we had throughout the work. For this project we mainly dealt with the 5 first steps up until deployment.

Initially, a lot of time was spent working with the *Business Understanding*, which is understanding the context or the problem you are trying to solve. In order to understand more about the field of no-shows in medical work, we started by reviewing past literature and talking with those at Helse Vest RHF. After spending some time reading the literature and reviewing different theories, the patient data was supplied. Most of the first step has been covered in the previous chapters, the introduction and theory.

The *Data Understanding* step relies on delving into the data, where statistics and visualizations were used to explore the traits of the data. The aim of this step is to gain some insight into the data, in light of the problem (business understanding). It can shed light on how the distribution of the data is, and the possible strength and shortcomings of the input

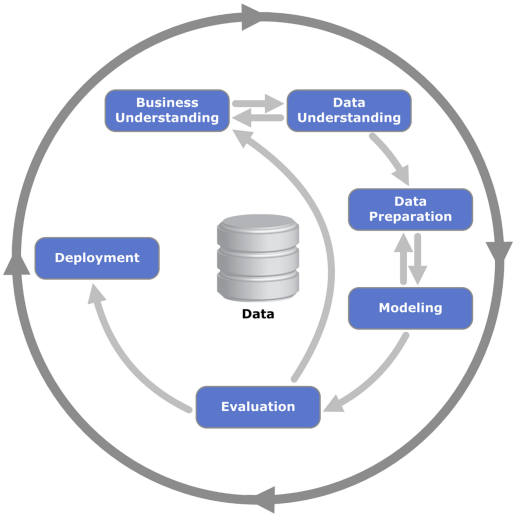


Figure 3.1: A diagram by Jensen (2012) showing the relationship between the different phases of CRISP-DM and illustrates the recursive nature of a data mining project.

for the model. This step helps in possibly identifying some of the more important variables and features for the model. Some of the analysis performed on the data is described and shown as part of the results chapter. We also wanted to identify some of the general traits of no-show candidates, and pre-emptively estimate which features are more relevant.

After reviewing the data, we moved on to the *Data preparation* step, referred to as pre-processing, which is more closely described in the following sections in this chapter. Some of the methods applied in this step was based on the initial analysis, but also on the model evaluation (which is shown as the arrows going both ways between the latter 3 steps in figure 3.1).

The objective of the thesis is mainly related to the *Evaluation* step. Although it might be the final goal of the thesis to have a working model, a continuous evaluation is ongoing throughout this process in light of the problem and the other steps. We attempt to find some of the strongest models in the evaluation step. The final step of *Deployment* is outside the main scope of the thesis, although some of the potential challenges of this phase is discussed in the discussion and future work.

3.2 Dataset

The data is from a clinic that offers psychiatric healthcare-services for adults. Not much information about this facility was not disclosed to us students in order to protect the identity of the individuals. Most information was thereby on a need-to-know basis, and generally limited. The received data was not completely raw, it was processed by the IT systems and a group at Helse Vest RHF. Most noticeably some information had been dropped, blurred and/or generalized in order to achieve k=5 order of K-anonymity. Decisions based on data about ethnicity, sex or religion need to be cleared in legislation due to dangers of discrim-

ination. Helse Vest told us this information was therefore dropped from this project. The specific methods of handling the raw data and the anonymization of it, was a process which was not disclosed to us. The anonymization step is necessary to reduce information which could serve as quasi-identifiers and potential linkability. The provided rows of records were often referred to as "contacts", which are appointments at the investigated clinic. Due to the anonymization and generalization each contacts for the same patients can not be connected or grouped. Each contact consists of information about the appointment, the attending patient, and other variables surrounding the interaction. For each of these features there are several variables, which can be categorical or numerical. The most important variable is the registered outcome of the appointment, which has the following codes in the data:

- 305 - When a patient did not show up.
- 308 - A cancellation by the patient
- 314 - The appointment was held as intended.

As a measure of the outcome of the appointment, it was for a while directly used as the target variable but this is changed and discussed in a later section. Some of the gathered information about the patient is about their characteristics such as their age or their travel distance to the clinic. Both of these are examples of generalized features. The patients age for instance was grouped in classes or bins, were one class was for instance $20 \leq age < 30$. The features about the appointment, can be for instance the time of year (although not what year), month, week, and weekday. Some of the features were calculated features, based on other features or on historical values. A full list of all these features, with some further explanation is provided in appendix B.

All in all we have around 69.000 patient-healthcare-contacts, with roughly 46.000 is the train set, and 23.000 in the test set. For the train set the distribution of the attendance is as seen in figure 3.2. As seen in the figure, the data is very imbalanced, especially when only no-show and a sub-set of the cancellations are of interest. This imbalance is one of the main challenges in the modelling, where different sampling techniques are tried and tested as a possible workaround.

Another important element of the dataset is the validity and quality of it. The dataset revealed some suspicious inconsistencies with contradicting feature values such as those who had immediate appointments, either not showing up/cancelling ahead of time or misreporting as being immediate. These can be due to possible human errors, where either wrong code for an ended appointment or other values being registered incorrectly. There are also intended misreporting, as there are fines for patients not showing up (around 1500 NOK), which is meant as an incentive for patients to show up. Sometimes no-shows and late cancellations were given some lee-way or mercy and registered as regular show, which has resulted in some of the contradicting records according to Helse Vest RHF.

3.2.1 Assumption of independence

There are some assumptions that must be made when modelling with this type of data. One of them is just like with most forms of statistics, we must assume the data is independent to a certain degree. All the appointments or records are anonymized, with the intention of not

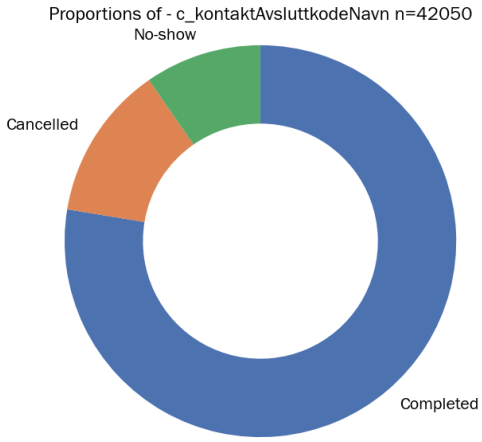


Figure 3.2: The distribution of the different outcomes in training data for the years 2016-2018.

being able to tell which record belongs to a specific patient. We know that the observations are not independent, as some are for the same patient, but also that some of the features surrounding the patient is also dependent. There will be some dependencies between the features, but also between the records. Some of the features that are dependent are the ones that refer to appointment or patient (abbreviated in Norwegian as "perHenv" or "perPas") in their feature name. These features are calculated feature, combined across appointments and other features, which means that they are dependent. With the methods applied we believe the features that are dependent, will still lead to better model performance as it gives more information about the target. We will compare the results with and without some of these features to investigate the assumptions closer.

3.3 Computer setup and Python libraries

For the entirety of the thesis, most of the programming was done on provided Dell-laptops, which was integrated with Helse Vests computer systems. The controlled workspace within the system, was limited and with quite strict terms of access outside the regular features of the operating system. It also meant that other than using pre-existing programs, new programs had to be installed through their ICT-group among others. This was a security requirement in order to have access to the confidential patient data. The laptops used had the following specifications:

- Operating system: Windows 10 - 64-bit version
- Processor: Intel(R) Core(TM) i5- 6300U CPU @2.4 GHz
- RAM: 8 GB

For the programming we used Anaconda navigator for handling the environment, with Python (3.8.5) as the programming language in Jupyter Notebook (6.1.4). Some of the main

libraries used:

- Pandas
- Numpy
- Matplotlib
- Seaborn
- imblearn
- SKLearn
- And others.

These libraries were used to handle the data in data frames, for the different operations of arrays and matrices, for plotting and also for the modelling with machine learning algorithms. The list of all imported libraries is also found in the provided code in the jupyter notebook for the project. This set the framework for most of the work around the data pre-processing, visualization, and modelling.

3.4 Data pre-processing

As mentioned in the previous sections, the data had already been processed and filtered to a certain extent for reasons such as anonymity and discrimination. This does not mean that the data was processed completely, as for the specific application changes still had to be made. Some of the records for the appointments had missing values, and some of the features that were initially classed or categorical were changed. Throughout the "cleaning" process, we noticed that for some appointments there were missing age of the patient, what kind of medical reference and the distance. Initially we developed codes which assigned the mean, and used linear models to estimate the distance. However because there were quite few records with these issues, and because other connected variables were missing for possible inference (due to anonymization), these were dropped (less than 20) in total.

Classed features such as distance to the clinic and patient age was to be replaced, to use it as a numerical value in the models. As part of the pre-processing the interval values were replaced with the middle value of the upper and lower bounds of the class. For instance for distance the value set to the interval 50 km $\leq$ distance $<$ 100 km was replaced with the value 75 km. The highest valued class was just set to the lower bound as it did not have an upper bound.

While modelling we came across some features that can not be used as predictors in the model. The reason is that some of the variables used as features are time dependent, meaning that they might not be available at the time of the prediction. These are variables other than the previously mentioned appointment outcome result, which are not available until after or shortly before the appointment. One of those variables was the variable for *"the time (in days) between cancellation and appointment"*. If there are no cancellations, it is registered as -1, otherwise it is how many days ahead of time it was cancelled. These variables are not available at the time of prediction, but are also an indirect indicator of the outcome. Another similar problem is the feature "ReminderSent"-feature. This is a feature

which just tracks whether a text message has been sent. This is one of the measures that has been introduced in the past to attempt to reduce no-shows. The problem is that this is registered two days prior to the appointment, which means this is also an indirect measure of the possible outcome of an appointment as it is at least not canceled up until then. This leaks information about the possible outcome of the appointment. That means that the feature also had to be dropped as it was indirect information about the outcome based on information unavailable at the time of prediction.

Some of the appointments where references of category 4, which means immediate help, these had to be dropped as they do not fall into type of data we are working with for this model. This lead to a significant loss of data, and went down to around 42.000 appointments/contacts in the training data. A very small amount of the appointments were also patients that registered were as dead, so their outcome was not much of mystery and thereby dropped.

3.4.1 Target feature

We initially used the outcome code (305,308 or 314) as the target feature or variable we wanted to predict (a three-class prediction). However shortly into the process we decided on moving away from a three classification problem (No-show, cancelled or completed), and instead addressed it as a binary classification problem. In previous sections it has been mentioned that in practice late cancellations will have some of the same consequences as a plain no-show. Throughout the project the threshold for the amount of days required to be *too late* has been adjusted to investigate some differences in the models and distributions. The main threshold we operated with was 3 days (although also compared with 1), as 3 days before was when the reminders were sent and where the clinic potentially passed on slots to other patients. Thereby knowing 3 days before, one could implement other measures in an attempt to reduce the likelihood of a no-show. Similarly to the standard "show"/"no-show" we change the definition slightly for the new re-purposing. In the new target feature anyone that showed up or cancelled ahead of time (before the set threshold) is classified as non-followup candidate (show or early cancellation), at times referred to as *pass* and with the value 0. Late cancellation or no-show, meaning a *follow-up* candidate is set to 1. Equation 3.1 shows the redefined prediction-variable

$$\hat{M}_i = \begin{cases} 0, & \text{Pass (Show or early cancellation)} \\ 1, & \text{Follow-up} \end{cases} \tag{3.1}$$

The errors is defined the same way as previously only with the new definition for these two classes. For the rest of the thesis no-show and follow-up will at times be used interchangeably, amongst other due to the literature and old habits. Our final analysis and predictions models are mainly concerned about whether candidates are in the "Follow-up" or "Pass" category.

3.5 Visualization of data

After cleaning up the data and adjusting some of the features we could begin visualizing the different features to explore the data. It helped with understanding the data, and find-

ing unique and interesting relationships that the models did not explicitly reveal. The data was visualized in different ways by using *matplotlib*, as well as *seaborn*. Seaborn combines *pandas* dataframes and matplotlib plotting in a practical and efficient way. This meant we could produce wedge plots, bar plots, joint-distribution plots and more. An issue that was revealed with the visualization was that the scale, and the distribution of some feature were quite spread. For instance the duration of appointments (a feature) normally had a duration of around hours, however sometimes they also last for days (although rarely). It is thereby hard to plot all the data at once, and requires some processing. A handful of the features were capped at the 99th percentile as an upper limit, as some of the plots generated were hard to read otherwise. Another challenge is the granularity of the data as a result of the generalization. It led to some less useful scatter plots in a grid-formation as some of the features were discrete valued due to generalization, as seen in figure 3.3 (left). In order to create more useful plots we applied seaborn's *kernel-density-estimation* (KDE) method when plotting the joint-distribution to show where observations were concentrated the most. An example of what it looks like is seen with age and month in the right plot of the figure. This method estimates the distribution centred at different areas of high concentration using a Gaussian kernel. This is seen as contour lines, but where the plots on the sides show the same topology but just for one of the features at the time. These contours show where the concentrations of the different outcomes (based on colours) can be seen. A higher density of contour lines means a higher gradient of the distribution, and thereby a higher concentration. In the scatter plot to the left in figure it is impossible to tell the concentration based on just a handful marked points due to repeated discrete values. The contour plot is thereby used to illustrate density in order to investigate possible areas of interests in feature-pairs. It can potentially help in identifying unique isolated groups at risk of no-show, or for picking features that can be useful in separating no-show candidates from the rest.

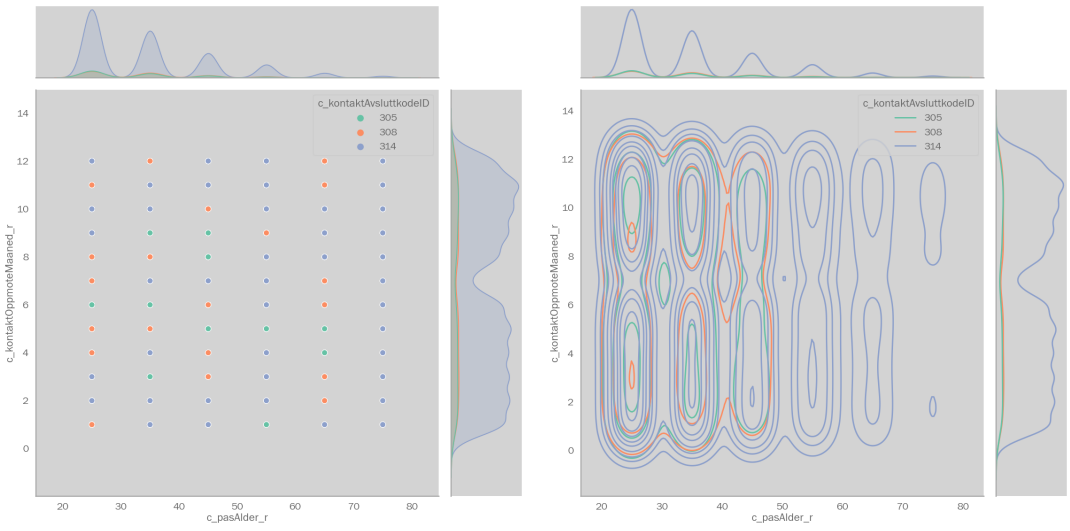


Figure 3.3: Two plots of the same two discrete valued features, "Month" and "Age" (group). Left is a scatter plot, and right is KDE-plot with contours for the densities of the different outcomes. 305: No-show, 308: Cancelled, 314: Completed.

3.6 Train/test split

Splitting data is an essential part of machine learning. We split the data into test and train data, so that we have some data that can be used to train our model and some data we can test our model on. We want these two datasets to be independent from each other, so they are separated to avoid potential data leakage. To achieve this split we use a function from the `sklearn`-library as shown in listing 3.1. When doing this we have to decide what ratio of our data is supposed to be training and test data. Usually this ratio is somewhere between 20%-35% test data, and respectively 65% - 80% training data. The benefit of having less data in the test-set means that we have more data to train our model, and it probably will get better, but we have less data to test it on and can not be as sure on our results. There is no right or wrong to where we put the split, so we find a limit that we think represents the data in a good way both for the training and test data (Brownlee, 2020c). The first models we created was based on this method and much of the code and initial thoughts were centered around this issue.

Because of the anonymization we have no idea of knowing if some of the datapoints is a recurring patient or a new one, so in this way we would not know in our model if there is some leakage between the datasets. What impact this will have on our final model is not easy to say as well. It could lead to a more generalized model, but it could also incur some relations that we would not know in the moment of prediction.

Later in the project we got two updated datasets that were already divided into training and test data based on year. This was probably more representative for the actual problem since the trained model will always use previously recorded data to make assumption for the future.

```
1  import numpy as np
2  from sklearn.model_selection import train_test_split
3
4  x_train, x_test, y_train, y_test = train_test_split(model_df, target,
     test_size = 0.3, random_state = 42)
5
```

Listing 3.1: Implementation of test/train data split

3.7 Sampling

When dealing with an imbalanced dataset as in our task, a common way of dealing with the *curse of imbalance* is applying sampling methods. To figure out the best way of sampling, different methods are applied and compared. Many sampling methods have been developed through the years and to find out what works for our specific problem a few has to be tested. For our problem we use the RandomOverSampler, RandomUnderSampler and SMOTE from the `imblearn`-library.

```
1  from imblearn.under_sampling import RandomUnderSampler
2  from imblearn.over_sampling import RandomOverSampler
3  from imblearn.over_sampling import SMOTE
4
5  def sampling_strategy(x_train, y_train, samp_strat = "
     RandomUnderSampler", ratio = 0.5):
6    if samp_strat == "RandomUnderSampler":
```

```
      sampler = RandomUnderSampler(ratio)
    elif samp_strat == "RandomOverSampler":
      sampler = RandomOverSampler(ratio)
    elif samp_strat == "SMOTE":
      sampler = SMOTE(ratio)

    x_train, y_train = sampler.fit_resample(x_train, y_train)
    return x_train, y_train
```

Listing 3.2: Implementation of sampling strategies

Before actually using the code it is important that the split of training and test data has been done before sampling. This is to avoid possible data leakage between the two datasets. The idea behind each of the sampling methods is pretty simple, but they all come with their pros and cons:

- **RandomUnderSampler:** Undersampling of the majority class. The imbalance is addressed, but we get rid of much data in the majority class. By randomly picking out data we could remove some data which could be influential to the performance of our model.
- **RandomOverSampler:** Oversampling resamples from the minority class to make more of the data in the minority class. This means we do not have to remove any data, but we could give some data more weight when training the model and we could maybe experience some bias in our predictions towards data that actually should not be as important.
- **SMOTE:** Making synthetic samples of the data. More data is generated through an algorithm to try and make more data points for the minority class based on the data that is already in this class. Some of the same drawbacks as with oversampling can often be found with this method.

An important part of this is deciding the ratio-parameter. If the ratio is zero it means we keep the original balance between the classes, but if we change this to one we will have sampled to a point where there is an equal number of data points in each class. Often a ratio of 1 could give a better recall value, but it might end up hurting the overall accuracy of the model. This is also something that has to be tried and tested to figure out what works.

When training the different models we found that there could be big differences in how the model performed based on what sampling method were used and what the ratio between major and minor class is. Often we found that the model performed better when using the Random undersampling method compared with the other ones. Although not always. Therefore we decided that in the result we would present a model that used this form of sampling with a ratio of about 0.67.

3.8 Selecting features

In our dataset there are 63 features to begin with, but not all of these are eligible for training our model. A couple of them is different ways of representing our target value, a couple is not present in the prediction moment and some of them correlate highly with each other and may cause overfitting in our model. To prevent this we look at our data to figure out

what features will give new information to our model and what features will not. Another thing that comes in to play here is that a model trained on less features has a considerably less computational cost and it often generalizes better.

We started this process by looking at a correlation plot to see what values correlates well with our target and what correlates highly with each other, but with around 60 features this amounted to a lot of combinations between the variables and to find the best combination of features this way could lead to bias from the interpreter. So instead we look at the methods explained in chapter 2.4.1 and 2.4.2 with the ANOVA and Extra Trees Classifier. We actually looked at a couple more, like a linear support vector classifier, variance-threshold and multiple scoring-functions in the `SelectKBest`-method, but decided it would take to much time to look at them all.

ANOVA is a feature in the `SelectKBest`-method which we focused on. `SelectKBest` is a method in the `sklearn`-library. As described in chapter 2.4.1 it makes an analysis of the variance and chooses the k-values with minimum variance. The k-value we decide ourselves and is the number of features chosen for further exploration and to train our final model. Knowing what the best number for this variable is case-dependent and we kind of just have to try and see what works. A higher value might give our model more data to train on, but it might also make our model very problem-specific. Say we wanted our model to also work outside the psychiatric outpatient clinic, then our model might be more generalizable and relevant for a wider range of branches. If our training data grows with time, it might also help considerably on the time it takes to train the model without necessarily reducing performance.

Using all features or too many can also harm our model performance if the data contributes with false patterns that our model give to much weight and selecting too few features might cause our model to be biased toward a class.

The other method we investigated for our project was the `ExtraTreesClassifier`. As described in chapter 2.4.2 this method chooses the number of features used for further investigation. For our problem we often ended up with a little over 30 features for the different runs we tried. The method is tree based and the trees are very randomized, so we can end up with different features each time we run it. We can decrease this variance by increasing the numbers of estimators used when selecting features, but this will also increase the run-time.

We ran the model with different values for k when we used the ANOVA method, but even with a range between 10 and 40 with a 5 increment we could not find much evidence that it performed better or worse than the extra trees classifier. The models that are shown in the results are therefore based on the extra trees classifier.

```
from sklearn.feature_selection import SelectKBest
from sklearn.feature_selection import f_classif
from sklearn.feature_selection import ExtraTreesClasssifier
feature_selector = "ExtraTreesClassifier"

if feature_selector == "ANOVA":
  #f_classif is the ANOVA method
  k_value = 10
  sel = SelectKBest(f_classif, k = k_value)
  sel.fit(x_train, y_train)
elif feature_selector == "ExtraTreesClassifier":
  n_estimators = 100
  sel = ExtraTreesClassifier(n_estimators = n_estimators)
```

```
    sel.fit(x_train, y_train)
    sel = SelectFromModel(sel, prefit = True)

  # make the new x_train-set
  X_new = sel.transform(x_train)

  #get the names of columns so we can adjust test-set.
  #x_train is a pandas dataframe

  names = list(x_train.columns[sel.get_support(indices =True)])

```

Listing 3.3: Implementation of ANOVA and ExtraTreesClassifier

Also in this example (listing 3.3) we make a variable with the names of the chosen columns, so we can adjust the test-data for predictions as well.

3.9 Selecting model

There are a lot of models we can try for our project, but it is hard to know what works best for any specific problem. In the start of our project we looked into different models like support vector classifiers, mlpclassifier (ANN), Least squares and more, but to narrow our project a little we ended up focusing on just tree different models. Based on our own experience and some of what has been read (Huang and Li, 2011; Lewis, 1998; Schapire et al., 1998; Breiman, 1996; Chen and Guestrin, 2016) we found that Naive Bayes, Random Forest and xgboost are often referred to as models that works good with small sample sets, are highly interpretable and the tree models are also easy to visualize for patients of the clinic. From a performance standpoint some of other models could be interesting to investigate as well, but we found that more of them could be harder to explain for a patient.

After deciding on what models to focus on, there is the issue of implementing them in our code. There are some things we have to consider before training the model and have it ready for evaluation.

First of all we want to scale our data, so that columns with inconsistent sizes on the values are not weighted different for each model. For this we use the StandardScaler from `sklearn` which gives the data a mean of zero and a standard deviation equal to one (Hale, 2019). The Naive Bayes and our tree models are not the models most known for being highly affected by the different features having different scale to the values. Still, by using a standard scaler we do not change the nature of the data (Brownlee, 2020a). The algorithms behind the models often have an easier time with scaled data and the code itself can more easily be used to implement other models as well if we at a later point would like to investigate for example a mlpclassifier.

Each of the models have their own set of hyperparameters that can be tuned for that specific model. Some are considered more important than other, but all can more or less influence how the final model will perform. For the Random Forest Classifier and the xgboost-models number of estimators can have a lot to say for how the final model will perform. Having a small number of estimators can make a model that evaluates both good and horrible. This is because there are few trees to combine for a final production and it is more up to chance if the trees combined will represent the data. Having more estimators will decrease the chance , but it will also increase complexity and time to make the final model. To find

what the value of each of these parameters should be we have to test a lot of different combinations of them. In the `sklearn`-library we have the `GridsearchCV`-method that can test many of these combination, but it is a time consuming effort, so we limit the number of parameters tested. Each new parameter that we want to test will increase the time it takes to test them all exponentially.

3.9.1 Pipeline

When implementing the standard scaler, our model and gridsearch with cross validation, there is a possibility of data leakage (Zhao, 2019). We do not want the standard scaler for temporary test set in the cross validation (see figure 2.8) to bee influenced by the temporary training set. Therefore we implement a pipeline. With this we can do a temporary scaler for each of the temporary training sets while doing the gridsearch.

3.9.2 GridsearchCV

Because each of our models have different parameters there have to be set a different parameter grid for each of the models when doing a gridsearch. A parameter grid is all the hyperparameters we want to test for our model and then the gridsearch-method tests the combination of them all to find what yields best performance in our model. We can in this part also choose what class to prioritize for our model. Here we have gone for the option "recall" to make our parameters prioritize classifying the follow-up-patients correctly.

The implementation is similar for the other three models, but the xgboost-model is not from the sklearn-library and the Gaussian Naive Bayes has no hyperparameters other than the prior, but this is adjusted based on the dataset, so we will not change the value.

```
#models
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
import xgboost as xgb

#choosing model parameters
from sklearn.preprocessing import StandardScaler
from sklearn.pipeline import Pipeline
from sklearn.model_selection import GridSearchCV

if model == "RandomForestClassifier":
  pipe = Pipeline([('scaler', StandardScaler()),
          ('clf', RandomForestClassifier())])
  param_grid = {
    'clf_n_estimators': [200, 400, 600, 800, 1000],
    'clf_max_features': ['auto', 'sqrt'],
    'clf_max_depth':  [4,5,6],
    'clf_criterion':  ['gini', 'entropy'],
    'clf_class_weight': [{0:w} for w in [1,2,3,4]],
    'clf_min_samples_split': [10, 20, 40, 50],
    'clf_min_samples_leaf': [4,8,12]
  }
elif model == "GaussianNB":
  pipe = Pipeline([('scaler', StandardScaler()),
  ('clf', GaussianNB())])
  param_grid = {}
elif model == "xgboost":
```

```
    pipe = Pipeline([('scaler', StandardScaler()),
    ('clf', xgb.XGBClassifier(eval_metric = "logloss", scale_pos_weight =
     10))])
    param_grid = {
      'clf__max_depth':     [4, 5, 6, 7],
      'clf__n_estimators':    [10, 12, 20, 40],
      'clf__gamma':       [0.01, 0.1, 0.2],
      'clf__min_child_weight':[0.1, 0.3]
    }

  clf = GridSearchCV(estimator = pipe, param_grid = param_grid, scoring =
     'recall', cv = 5)
  clf.fit(x_train, y_train)
```

Listing 3.4: Implementation of a pipeline and gridsearchCV

Here we can easily change, remove or add more hyperparameters to test. We tried to keep the numbers to a minimum, so we could run multiple tries with different sampling strategies, features and models without it being too time consuming. The PC we worked with had its limitations as well, so with a more powerful PC it would have been easier to test with more parameters.

3.10 Evaluating model

When evaluating the model there are multiple aspects we would like to look at. As described in chapter 2.5.1 we have different metrics that are used for evaluation of the model performance. When our trained model makes it predictions, it does so based on a probability value of what class show and follow-ups are a part of. The cutoff or threshold for this probability is based on the G-mean value described in formula 2.13. This value could also be set to some fixed value. The Threshold is used for predictions so we have a possibility of setting up a confusion matrix and evaluate our model. In practice we would more likely use the probability value for a prediction to set up different kind of measures to prevent follow-ups from happening. For example set a 50% value for sending a text message or a 70% value for calling the patient.

```
from sklearn.metrics import roc_curve

# prediction probabilities
predict_p = clf.predict_proba(x_test)[:,1]

#false and true positive rates of test data
fpr, tpr, thresholds = roc_curve(y_test, predict_p)
gmeans = np.sqrt(tpr*(1-fpr))
index = np.argmax(gmeans)

#predictions and optimal thresholds
y_pred = np.where(predict_p>thresholds[index], 1 ,0)
```

Listing 3.5: Using G-mean to find threshold and use it for predicitons

After predictions are done we would like to see the performance of our model. We use the methods from the `sklearn`-library to get a confusion matrix and an accuracy-, precision-, recall- and F1-score for our model.

The point of all these measures is to get an accurate overview of the model performance and to see what works in regards to our problem.

```
from sklearn.metrics import confusion_matrix,
  precision_recall_fscore_support, accuracy_score

print(confusion_matrix(y_test, y_pred))
print(accuracy_score(y_test, y_pred))
print("precision: ", precision_recall_fscore_support(y_test, y_pred)
  [0])
print("recall: ", precision_recall_fscore_support(y_test, y_pred)[1])
print("f_score: ", precision_recall_fscore_support(y_test, y_pred)[2])
```

Listing 3.6: sklearn.metrics provides all the scores for our model

In our problem we want to make a model that predicts the follow-ups correctly. So the recall-value ends up being very important in this case. Still, our model will not be very good if all shows are predicted as follow-ups as well. Because of the nature of the data and the imbalance in the dataset many shows get predicted as follow-ups when we train it with prioritized recall value. The F1-score gives a value that should represent the balance between the precision- and recall-scoring. When presenting the results we also show the ROC-plots. It can show how the false and true positive rate will be affected for what threshold value we choose

Having all the values as close to one would be the optimal solution, but in practice this would probably mean we have done something wrong, because this is close to impossible. Human behavior is hard to predict and in the end, the idea is to strike a balance between the best possible care for the patient and the cost for the healthcare system.

3.11 Summary of method

The CRISP process has been used as the outline of our workflow throughout the project. At the start of the project relevant articles provided by Helse Vest was read to understand what no-shows in healthcare practice actually meant (Dantas et al., 2018; Molfenter, 2013), and to understand the business side of the problem. From there, the project moved on to preparing and understanding the data, so that it could be visualized to understand relationships of the features and outcomes. In light of this a new target feature was defined, for the outcome follow-up and pass, instead of the standard no-show and show.

When data was understood and processed, the models could be built and evaluated, but this was not a linear process. After a model was built and evaluated, the process often returned to the start and was evaluated in light of the first stages of CRISP (see figure 3.1). The work continued in circle of the first five stages of CRISP throughout the project. This continued until there was no more time to optimize and experiment with the models, when we had to write up the results.

Chapter 4

Result

Before going into the main results of the thesis, the prediction models, we take a look at what the data reveals about the relationship between appointments and patients. In the following subsections, the training data (data from 2016-2018) is investigated with different forms of visualization to try and see why some of the features are relevant for the outcome of an appointment. All in all, there are about 63 features as mentioned earlier, which means that we do not have a chance to illustrate them all at once. We qualitatively investigate for points of interests by looking at the different features' distribution and density in one and two dimensions as part of our initial analysis. The focus narrowed down to the features which often appeared in the feature selection in the modeling stage. This might seem out of order, but understanding the data and the development of the models was a continuous and recursive process that went back and forth throughout the project. The analysis is not necessarily representative of how all models approach the data, as models have different strategies and underlying algorithms. The goal with the initial analysis is to shed light on some possible and unique distinctions observed within the data, which might be relevant for some of the models. The analysis starts with a look at the original feature for the outcome, No-show/Cancelled/Completed.

4.1 Qualitative analysis of the training data

4.1.1 Analysis for no-show, cancelled and completed

One dimension relative ratio distribution

If we have a look at the distribution of no-shows (305) versus the cancelled and completed appointments (308+314) we see that there is a significant drop in July as seen in figure 4.1. At first glance it might seem that July is significant, but the drop is seen in both categories, and it is simply due to reduced activity over the summer break. This points out the weakness of using the raw frequency when trying to investigate relationships between appointment outcomes and features.

That is why we look at the relative ratio distribution of these subsets where all counts are divided by the total count within the subset. That means we have taken the amount of

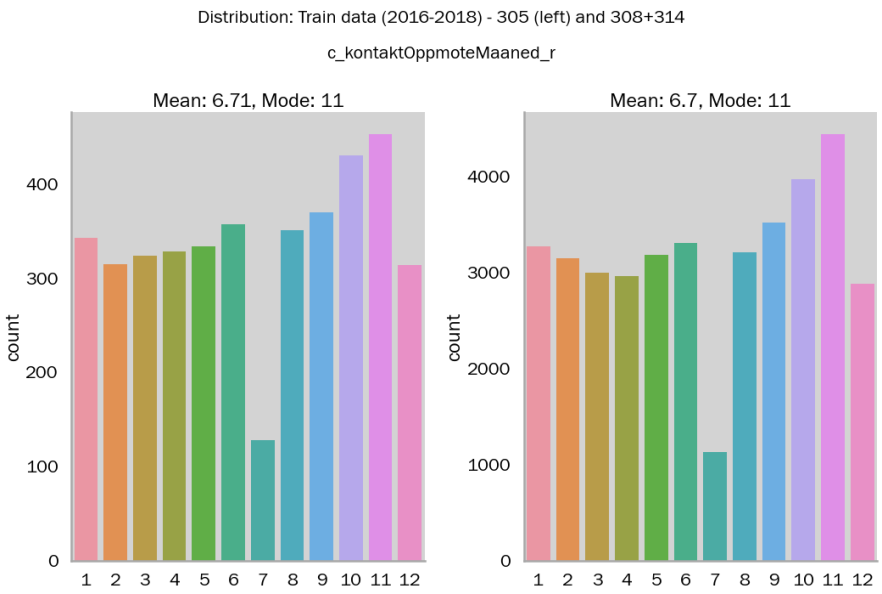


Figure 4.1: The distribution of no-shows vs cancelled and completed, over the different months. At first glance july seems unique.

appointments for the subset 305 in month 1 (January) and divided by the total of appointments in month 1. The formula for the ratio of no-shows in January can be expressed as:

$$R_{\text{No-Show} \wedge \text{January}} = \frac{N_{\text{No-shows in January}}}{N_{\text{Appointments in January}}} \tag{4.1}$$

Figure 4.2 shows the same distribution as previously, but where each column is the relative ratio within each subgroup. The right plot is the same calculation, but where its the sum of 308 and 314 divided by the total in each month. We see that pair-wise the same columns of each plot add up to a total of 1 as they are complementary events, and represent the entire sample space for appointments in January.

Other than the relative frequency being calculated as a ratio within the subset, we also provide a baseline with the marked diamonds. This shows the ratio between the selected subset and the total count. The purpose of the baseline is just to to know what the size of the selected subset is in regards to the total. It can help determine whether deviations are significant or just likely outliers. Some bars might have a large relative ratio, but it might just be due to a low amount of observations within that subset. Thereby if a bar has a large ratio compared to the rest while the baseline marker is closer to 0, it is considered an outlier as the ratio might just be due to a small sample size. This is well illustrated in figure 4.3, and can be seen with all the ratios above 15 hours in the time of day of appointments. We see that the distribution shows that there is a higher relative frequency for appointments around 16 and 17, however note that the markers put the appointment count quite close to 0. These appointments can either be removed, or accumulated and combined into a separate bin.

The reason the data is visualized as these ratios, is that it is a useful way of displaying the

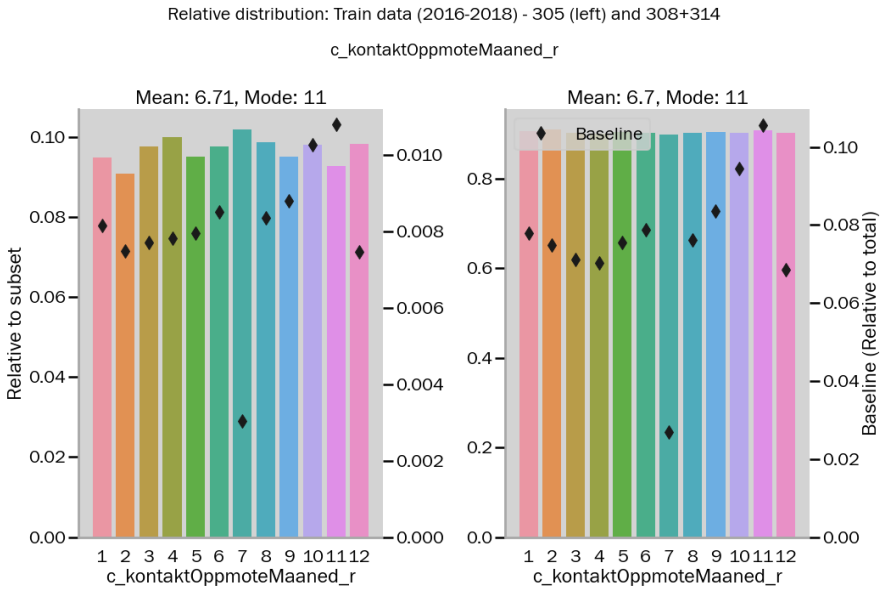


Figure 4.2: The relative ratio distribution of no-shows and the rest over different months. The plot shows that raw distribution is not as significant if we take into consideration the amount of appointments in each month. Note July especially, compared with last figure.

data as the subgroups of these features are at times quite imbalanced as well. These plots reveal some possible interesting characteristics of the appointments in the training data. If we look at the monthly distribution there can be some indication of seasonal variation, but the differences are not that large when taking the scale into consideration. It can be significant when combined with other features in the later models. Another feature where we can see the need for the relative ratio, is the age feature as seen in figure 4.4. In the figure we see that a significant group of the appointments are younger people in the 20 to 29 years group (marked as 20). More than two thirds of the appointments are with patients that are under 40 years. If we compare the frequency distributions in figure 4.5, it is no surprise that both highest on attending and not attending is the major class of "20".

That is why we introduce the relative scaling again, to show that age is significant other than just being the major group (although it is the major group for a reason). In figure 4.6 we see the distribution with relative scaling, so that we look at the proportions of attendance relative to size of the subset. In this figure, we can see that age has a connection with attendance, the relative frequencies decreases with increasing age.

This suggests that the younger patients needs more closer attention as they are at higher risk of no-shows. Not all the dependencies are as clear as the distribution of the age feature, figure 4.3 shows that appointments closer to noon have a higher no-show ratio.

Another feature which showed something interesting was the feature for patient no-show history, as seen in figure 4.7. This indicates that the patients with a past history of no-shows is important in future predictions. In the plot we see that the higher the no-show rate, the higher the ratio of no-show for each bin. We see that the right most bars in both

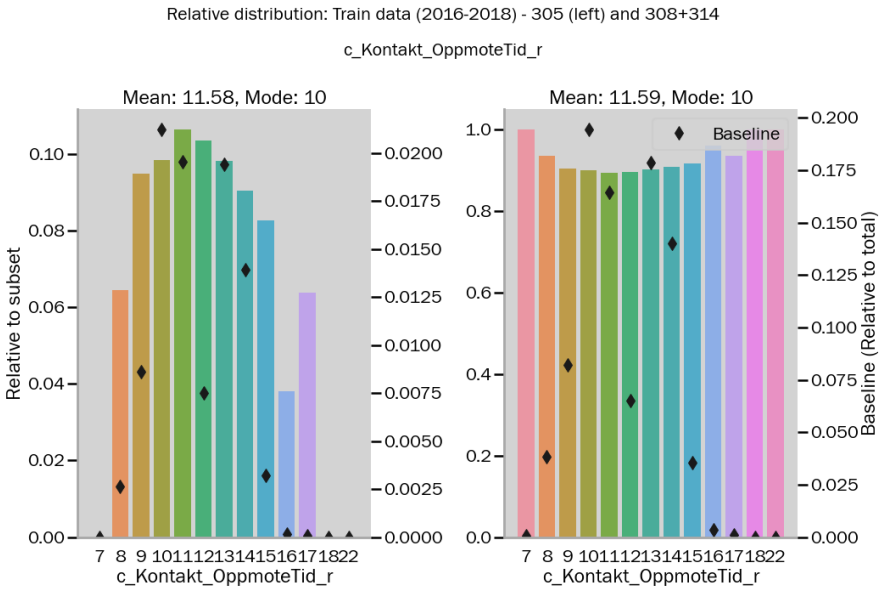


Figure 4.3: The relative ratio distribution of no-show and the rests as subsets for the time of day of the appointments. The plot shows that appointments closer to noon have a larger fraction of cancellation

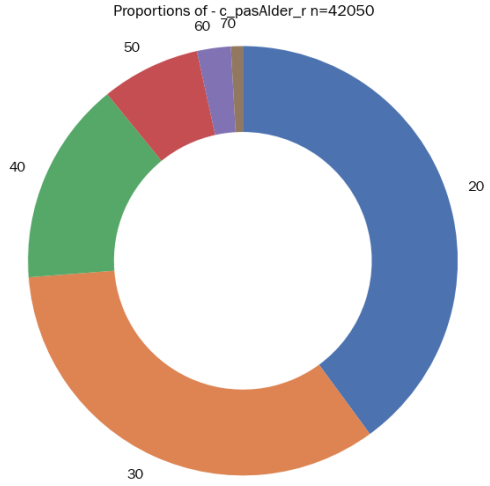


Figure 4.4: The distribution of the age plots, here the 70+ group also includes everyone above (a small number).

plots can be considered outliers (especially the one in the right plot). We can see that by enlarge the mean of the no-show candidates is much larger than for the cancelled and completed patients.

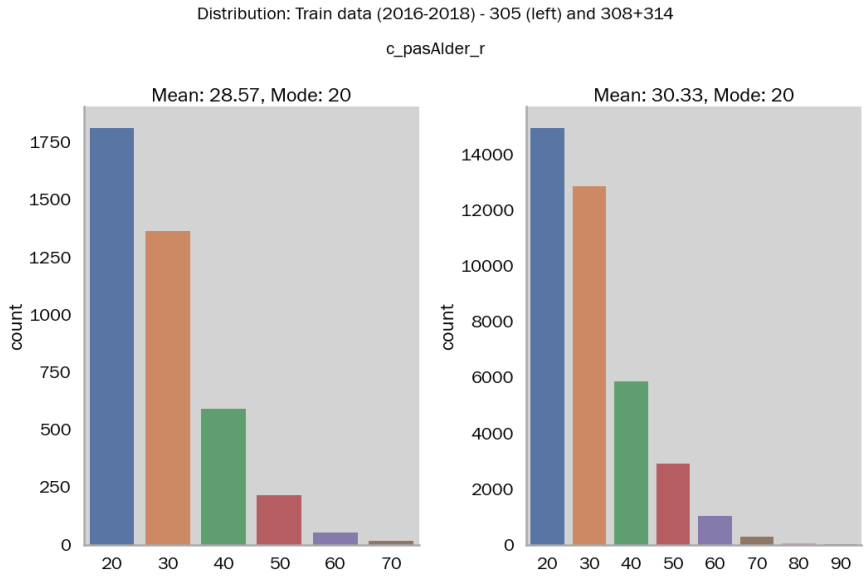


Figure 4.5: A bar plot with the raw frequency for the different age groups for no-show and the rest.

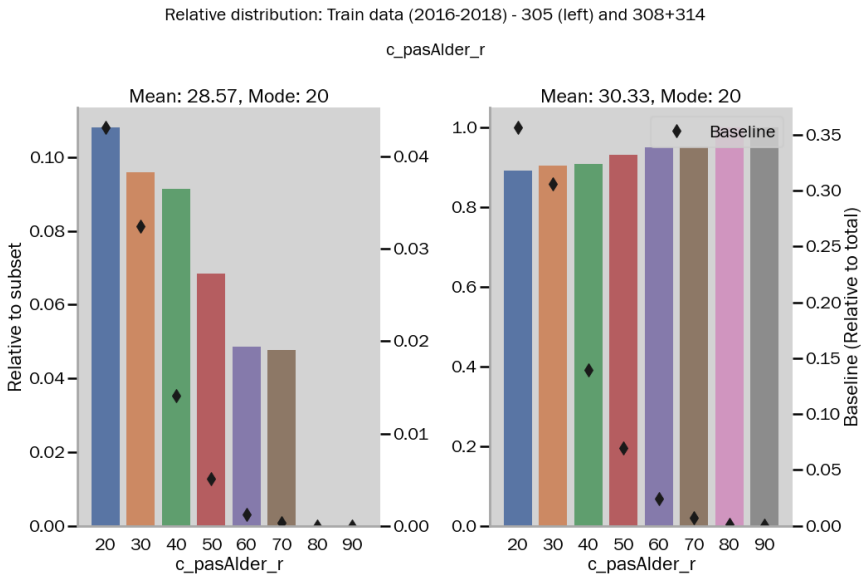


Figure 4.6: The relative ratio distribution for the different age groups for no-show and the rest. The plot shows that the fraction of no-show is "inverse proportionally" when taking the amount of appointments within each subset into account.

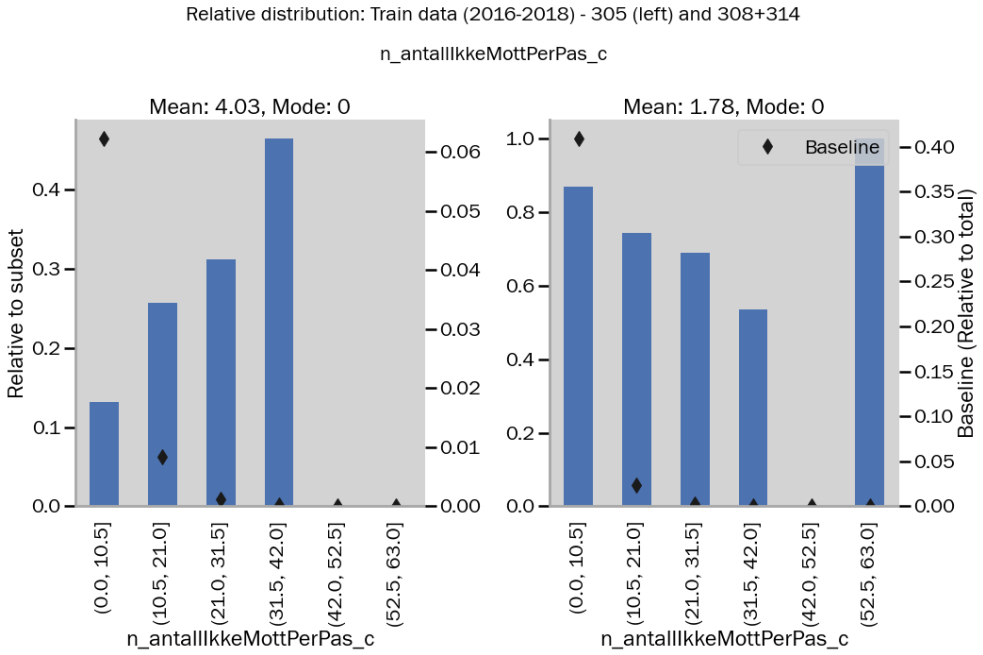


Figure 4.7: The relative ratio distribution for the past no-show counts for the patient for the subset of no-show and the rest. The bar plot to the left indicates that the fraction of no-shows increases for patient with higher past no-shows.

The analysis up until now has been done for the individual features in one dimension. The goal has been to discover features of interest for the outcome. Next the analysis move on to representing some of these features with plots in two dimensions to see possible relationships between features.

Two dimensional kernel density estimates

Not all features revealed anything of interest on their own in the one dimensional visualization. Some relationships emerge in two dimensions, when viewed as joint distributions of different features. In figure 3.3, it was shown that the granularity of the data causes some challenges with the visualization. By creating KDE-plots it is easier to see where, and how, the distributions are concentrated in two dimensions. It is not necessarily the true distribution (especially with the discrete valued features), but it is an estimation of the distribution around the values. With the contours of the KDE-plots, you can make the areas of higher concentrations and see overlapping regions between the different outcomes.

In figure 4.8 we see the joint distribution for prior no-show ratio (n_andelIkkeMottPerPAs_c) and age in an interesting KDE-plot. The figure confirms some of the observations we made about these features in one-dimension. We see that there are fewer older people that have no-show, but also that they have lower no-show ratio over all. However since this is density we need to be careful with some of the assumptions

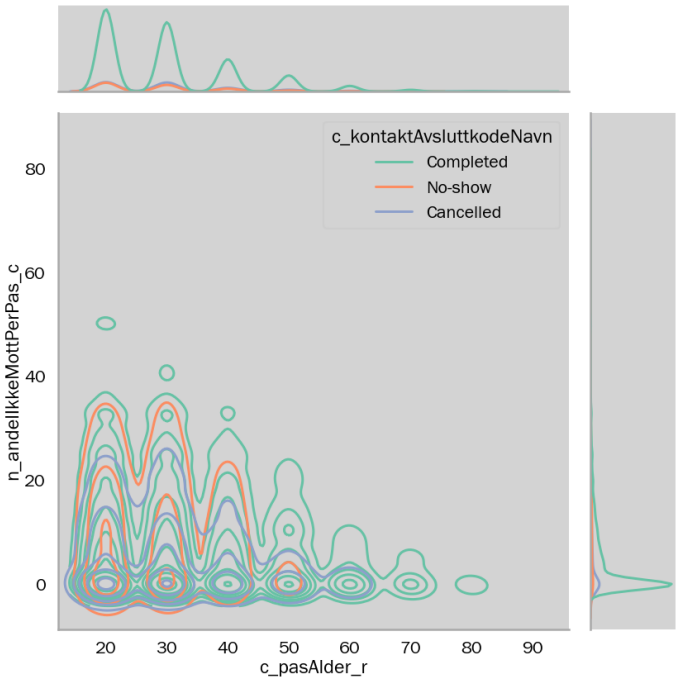


Figure 4.8: The features Age vs "Ratio of no-show per patient" in a joint distribution (KDE).

and conclusions we make, as the age group is very uneven (there are many more people under 40 than above). That is why we looked at the relative ratio in one-dimension for the age among others in the previous section. Due to the KDE-plot being based on the raw frequency, not all these observations are necessarily significant. That is why for the next figures the outcomes of attended appointments are dropped as the scale of densities of regular appointments overshadows the others due to the imbalance. In figure 4.8 we see some of the same tendencies as some of the earlier one-dimensional plots has implied. In the next figures we have removed "Completed" appointments, to have a closer look on the boundaries between no-shows and cancellations.

The reason to drop completed appointments, is that these at times overshadows the outcomes we are the most interested in. A point of interest can also be what seperates a patient that cancels appointments and a patient that does not show up. In figure 4.9 we see that patients with a high ratio of no-shows are likely to not show up for their appointments. This can be seen by the green region above the densest area in the figure. The same was also observed for the days feature (not shown). In figure 4.10 we see the same with patients that have a high ratio of no-show, do not show up, and that patients with high ratio of cancellation tend to cancel. This observation is however a bit of a tautology, as the feature is dependent on past outcomes. What we can tell by the three figures shown in this subsection is that past history of patients seems to be significant for no-show.

It is hard to find clear boundaries between these two categories, so in the next subsection we will look closer at plots for our target feature, candidates to "Follow-up" and "Pass".

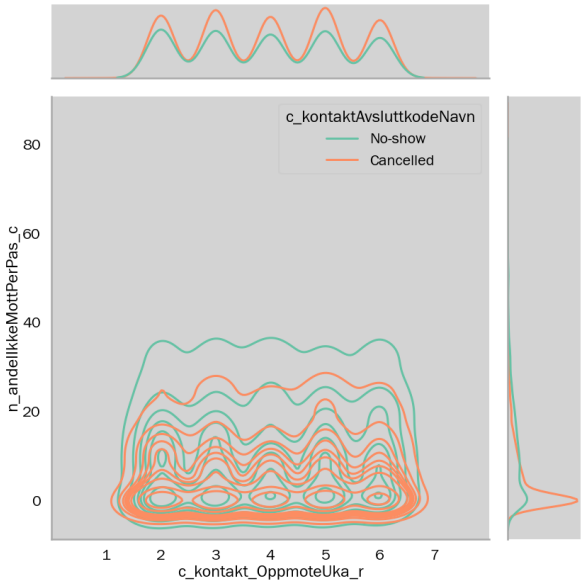


Figure 4.9: The features "Day of the week" vs "Ratio of no-show per patient" in a joint distribution (KDE).

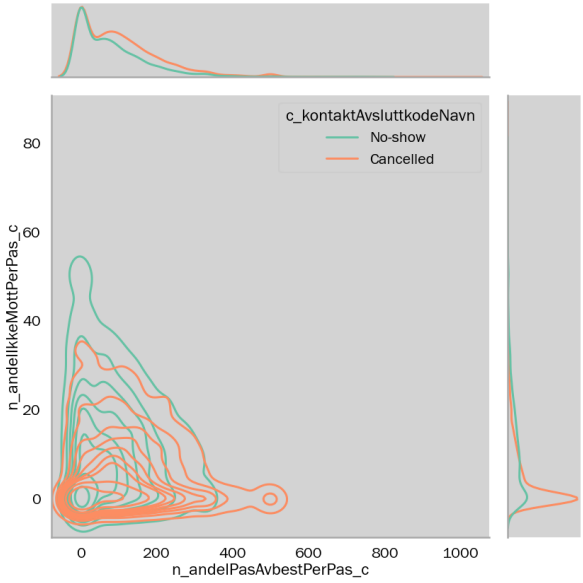


Figure 4.10: The features "Ratio of cancelled per patient" vs "Ratio of no-show per patient" in a joint distribution (KDE).

4.1.2 Analysis of Follow-up and Pass

One dimension relative ratio distribution

As part of the pre-processing, very late cancellations and no-show outcomes were combined as possible "Follow-up" candidates. Currently the threshold for a "too late"-cancellation is set at 3 days before the appointment due to existing practices at the clinic (although other threshold was also investigated). In this sub-section we will not revisit all the different plots, but we will highlight some of most notable observations made in the visualization of the new target category.

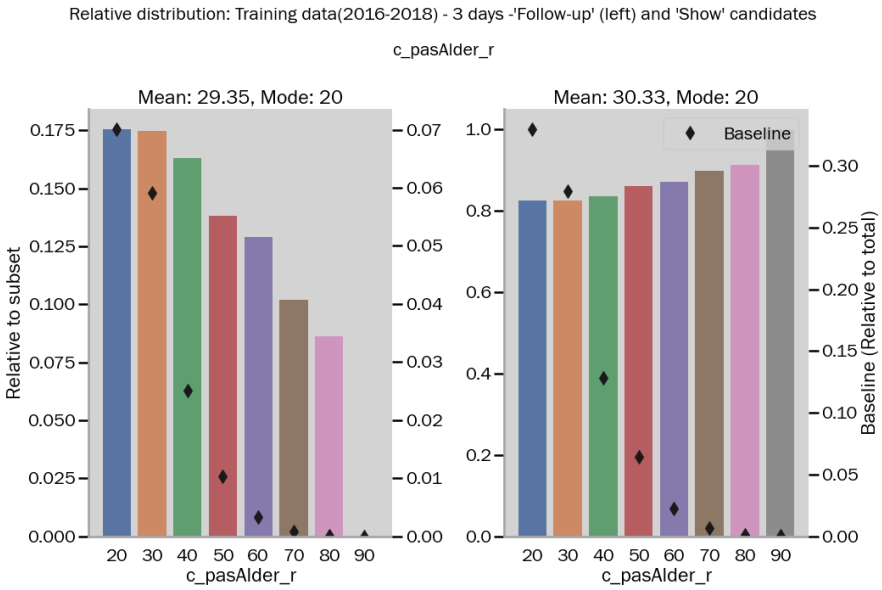


Figure 4.11: The relative ratio distribution for the different age groups for follow-up (left) and the rest (right) 3 days before.

We see that the distribution is slightly changed as some "cancelled" appointments migrate over to the "follow-up" category. The same change is also observed in the past no-show feature in figure 4.12. This means that when some of the cancelled patients has migrated over, we might see some changes in the distribution and boundaries of the features.

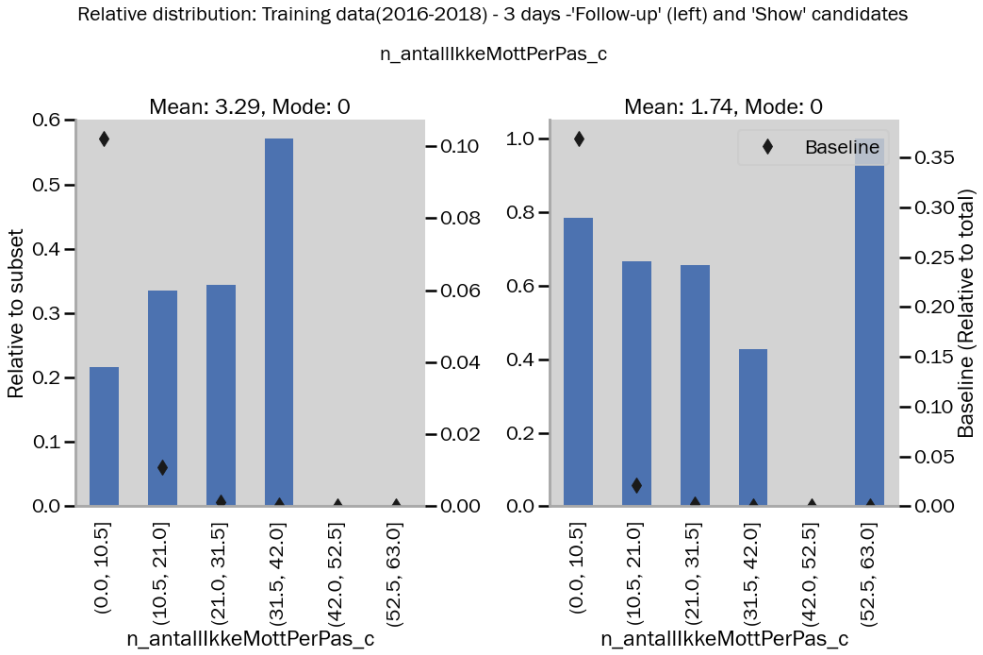


Figure 4.12: The relative ratio distribution for "past no-show counts" for follow-up (left) and the rest (right) 3 days before.

Two dimensional kernel density estimates

For the two-dimensional KDE of the target feature, we decided it was more appropriate to have an approach similar to that of the modelling. An issue which has been highlighted throughout this thesis, also in this section, has been the imbalance of the classes. This has been discussed in terms of the algorithms for the machine learning as part of the sampling before implementing different models. One of the approaches which has been used is the "Random undersampler", with the help of *imblearn-library* in python. We decided that it was reasonable to have the same approach in the analysis of the target feature, in order to look at the data in a similar fashion as the model would do. For the following plots we have applied a random under sampler with a ratio of 0.7, and a random seed with the value 2021. This means the analysis of features is subjective just for this seed of random samples, so there will be some differences with each sampling. The hope is to see some points of interests that can be investigated further with other seeds if needed. Other than to simulate an approach similar of the modelling, we have this approach because the majority class so often drowns the minor class. When the majority class is undersampled, we hope the density plots can show some more interesting characteristics with the joint distributions.

In figure 4.13 the advantage of undersampling of the majority class is clearly seen. There is a region of only follow-up which can be isolated, which enforces what we see in previous plots of the "ratio of past no-show" being an important feature. It is quite clear when compared with figure 4.8, where we see the same for no-show but where the majority class com-

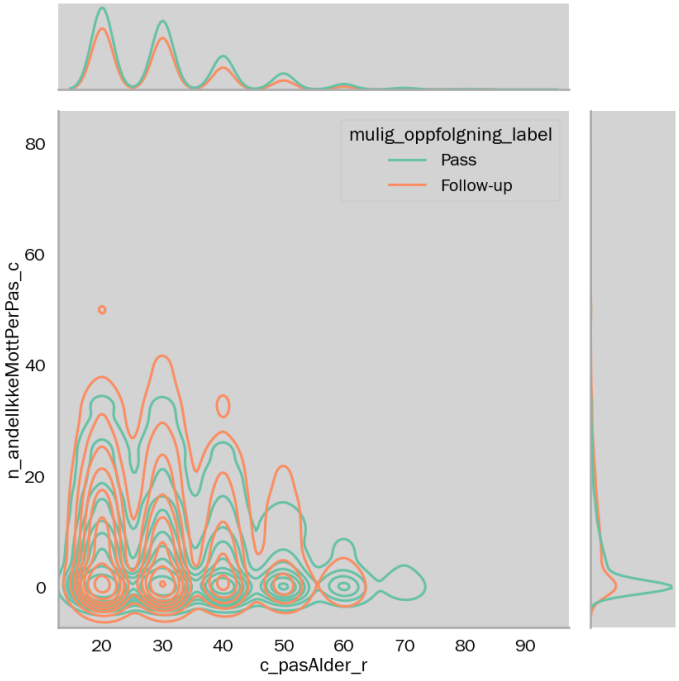


Figure 4.13: The features "age" vs "Ratio of no-show per patient" in a joint distribution (KDE) for the under sampled target feature.

pletely engulfs it. We also see some of the same observations in figure 4.14 with "week day" and "ratio of no-show", which also is similar to figure 4.9 (which had to be shown without the majority class). There are two figures that revealed some new and interesting relationships than what has been seen prior to the undersampling, figures 4.15 and 4.16. In both figures we see there are islands and regions that are almost completely isolated which is useful when trying to identify important groups of follow-up candidates both for the models and potentially for the clinic.

The plots were controlled for other random seeds as well, and the same patterns emerged in the those sets too. A sample is provided for the reader in appendix A. Although these distributions are not explicitly used in creating the models themselves, they are useful in understanding their significance. We now understand why some of these features have been seen again and again in the models as they find some of these unique regions. These regions and areas might seem irrelevant or marginal, however in light of the imbalance these are the type of patterns that models often look for. The rest is caught within the boundaries where they are well mixed and where they are not as easily discerned.

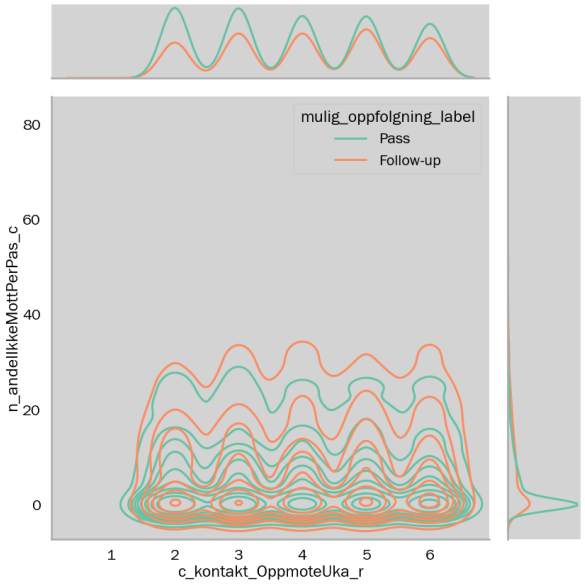


Figure 4.14: The features "Day of week" vs "Ratio of no-show per patient" in a joint distribution (KDE) for the under sampled target feature.

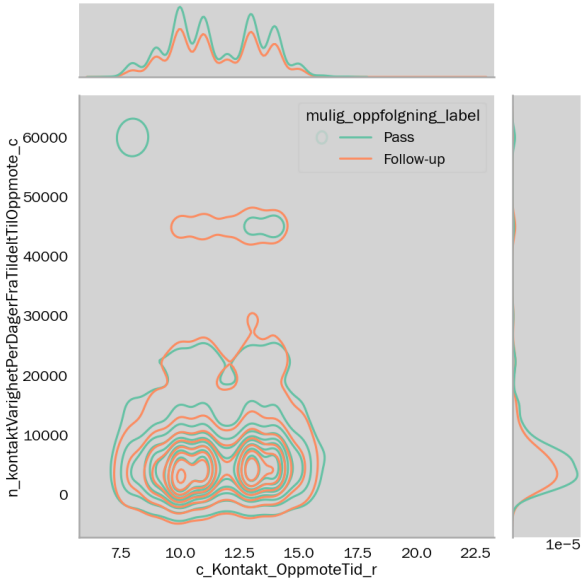


Figure 4.15: KDE plot of target for features: "time of day" vs "Duration in min of planned appointments divided by days between appointment allocation and first appointment".

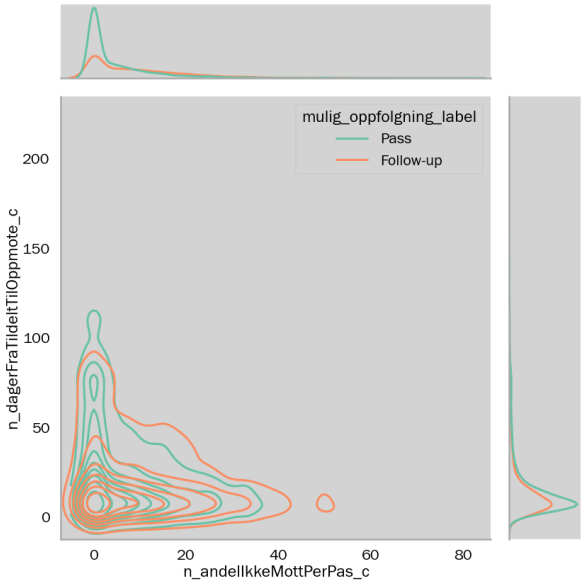


Figure 4.16: KDE plot of target for features: "Ratio of no-show per patient" vs "Days from allocation and to appointment".

4.2 Models

After training a myriad of models, with different sampling methods, classifiers, hyperparameters, feature selectors and some with and without certain features available we got a lot of different results. We wanted to look at some of these results and compare the performance, strengths and weaknesses of the different models.

Going into specific details on all of the different models is not realizable as there are endless possibilities. Therefore some choices were made on what samples to display here. After running several models, we discovered that models are less prone to overfitting, with a good recall-score and overall performance when using random undersampling. It seems to help when dealing with the imbalance, and the Extra Trees Classifier when selecting features. When testing multiple models we ran them with other types of sampling methods, but as described in chapter 3.7, we will show models based solely on undersampling.

The three models used in this project performed differently for the different metrics described in chapter 2.5.1. In this section there is a small summary of the strengths and weaknesses of each model with and without dependent data.

4.2.1 Independent models

As described in chapter 3.2.1 we have an assumption of independence in our data, but some of our data is not really independent. So we wanted to see how our model would perform without these feature. We removed all features which contained the phrases “PerPas” or “PerHen” as these indicated features that were based on other appointments and ran the

same code as for the other models. A complete list of these features can be seen in appendix C.

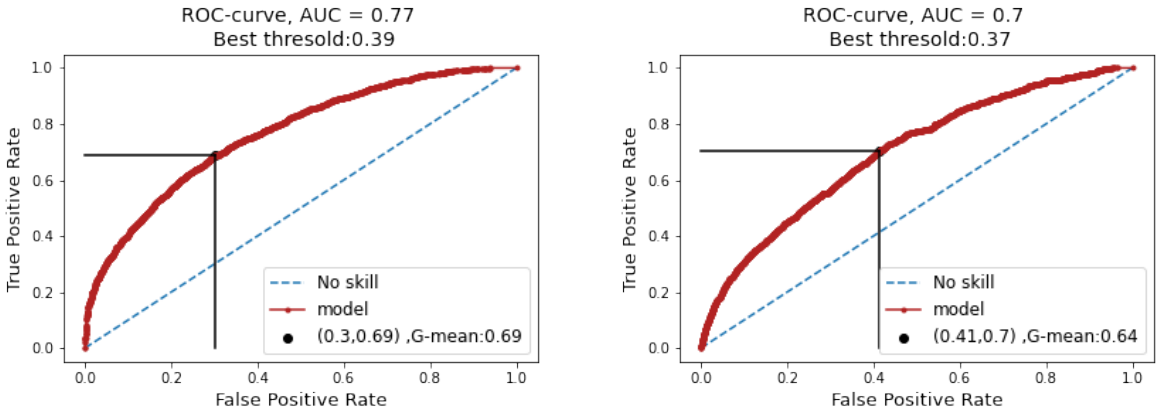
For these particular models we have used the following parameters

- Sampling Strategy (ratio): RandomUndersampler (0.67)
- feature selector: ExtraTreesClassifier

Random Forest Classifier

When assessing the model, the first thing we do is check the confusion matrix for the training and test data with accuracy and recall-score. The confusion matrix is made from the predictions of the data, and our predictions are based on the G-mean value (see equation 2.13). This is found when checking the true positive vs. the false positive values in a ROC-plot.

We do the same procedure for the training data. If there is to much of difference between the training and test data we probably have some sort of over- or underfitting in our model.



(a) Training data with the Random Forest Classifier

(b) Test data with the Random Forest Classifier

Figure 4.17: Comparing ROC plots for train and test set. The best threshold value shows the probability cutoff between those classified as pass and those classified as follow-up candidates

There are some signs of overfitting since there is a higher degree of accuracy in the training data than in the test data. As you can see from the top of the ROC-plot, the best threshold value based on the G-mean is represented and the predictions are based on this value. If the value is larger, the data is predicted as a follow-up and if it gets smaller it is predicted as a pass. Beneath you have a couple of lists that show the metrics for the training and the test data. The recall and the F1-score has two values in a list. The reason is that the model give the score for both classes. The first value in the list shows the score for the pass-class and the second one shows the value for the follow-up class.

Table 4.1: Confusion matrix for test and training data with the Random Forest Classifier

	Training Data	Predicted	
		Pass	Follow-up
Actual	Pass	2430	1041
	Follow-up	731	1595

	Test Data	Predicted	
		Pass	Follow-up
Actual	Pass	12088	8494
	Follow-up	268	636

Training data:

- Accuracy: 0.694
- Recall: [0.700 0.686]
- F1-score: [0.756 0.686]

Test data :

- Accuracy: 0.592
- Recall: [0.587 0.704]
- F1-score: [0.804 0.157]

The second F1-score value is much lower on the test set. As the model is trained on recall this kind of makes sense, because more of the patients that meet for their appointment is predicted as not showing and this could hurt the precision (see formula 2.9).

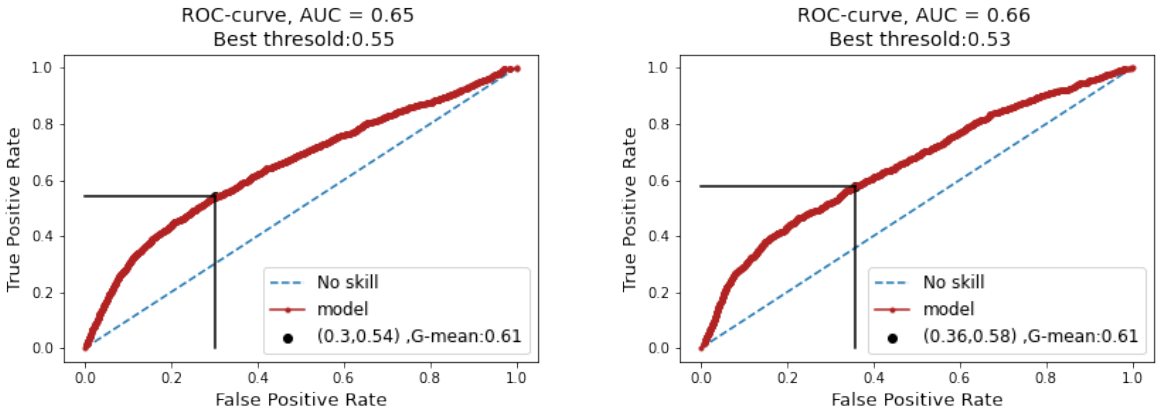
Since this model is based on the independent variables we do not have a lot of features and after the ExtraTreesClassifier and the RandomForestclassifier has run, the final model is only based on eight different features. The built-in "feature importance"-method from RandomForestClassifier ranks the three most important features in descending order as:

1. n_dagerFraTildeltTilOppmote_c
2. n_ventetid_c
3. c_kontakt_OppmoteUka_r

The first value describes the amount of time from the appointment was given till it was held, the second one is the time from a referral has been assessed by specialist health service and the last one is what day of the week the appointment is set.

Gaussian Naive Bayes

We want to present the result in the same way as we did for the Random Forest for the Gaussian Naive Bayes (GNB) as well. One important difference to have in mind for the Random Forest is that the "feature importance"-method does not exist for the GNB, but we have another one called the "permutation importance"-method which works in a similar fashion. For further reading on the difference between these methods check out Scikit-learn (2021).



(a) Train data

(b) Test data

Figure 4.18: Comparing ROC plots for the Gaussian Naive Bayes train and test set.

The AUC-value is lower for the GNB than it was for the random forest classifier, but when you compare the test results for these first two models you see the accuracy increase somewhat for the GNB-model. Even so the recall value for the follow-up class has gone down considerably, and we see the same pattern with the F1-score where it decreases considerably down for the test data. This happens with all the models.

Table 4.2: Confusion matrix for test and training data

	Training Data	Predicted	
		Pass	Follow-up
Actual	Pass	2424	1047
	Follow-up	1071	1255

	Test Data	Predicted	
		Pass	Follow-up
Actual	Pass	13235	7347
	Follow-up	381	523

Training data:

- Accuracy: 0.635
- Recall: [0.698 0.540]
- F1-score: [0.695 0.542]

Test data :

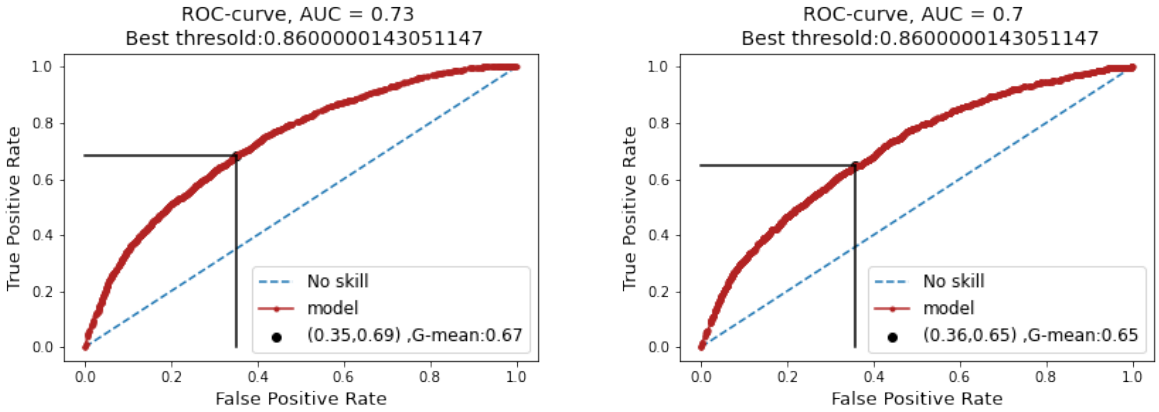
- Accuracy: 0.640
- Recall: [0.643 0.579]
- F1-score: [0.774 0.119]

The most important features for the GaussianNB-model is listed below. You can see that the second feature is age, which is an attribute of the patient, but the first and third feature are related to the appointment and logistic at the healthcare clinic.

1. c_kontaktOppmoteMaaned_r
2. c_pasAlder_r
3. c_kontaktOppmoteUka_r

xgboost

The xgboost model performs more stable than the other two models when not all features are present. We see that there are no big fluctuations between the scores in the training and test set. This could mean there is less chance of overfitting in this model. At least with the parameters used to train this model (check listing 3.4).



(a) Train data with the xgboost-model

(b) Test data with the xgboost-model

Figure 4.19: Comparing ROC plots for the xgboost. A bug in the numpy round-function caused the threshold value to show more decimals than intended

Table 4.3: Confusion matrix for test and training data with the xgboost

	Training Data	Predicted	
		Pass	Follow-up
Actual	Pass	2255	1216
	Follow-up	739	1587

	Test Data	Predicted	
		Pass	Follow-up
Actual	Pass	13230	7352
	Follow-up	319	585

As in the other models we still get a lot of patients that actually show up predicted as candidates to follow up. This is less important for the model than classifying the ones who does not show correctly. Still, if there is too many it will still end up being a lot of work for the healthcare service.

Training data:

- Accuracy: 0.663
- Recall: [0.650 0.682]
- F1-score: [0.698 0.619]

Test data :

- Accuracy: 0.643
- Recall: [0.643 0.647]
- F1-score: [0.775 0.132]

For the xgboost we got that the three most important features was described as:

1. n_dagerFraTildeltTilOppmote_c
2. n_dagerTilTimeGitt_c
3. n_kontaktVarighet

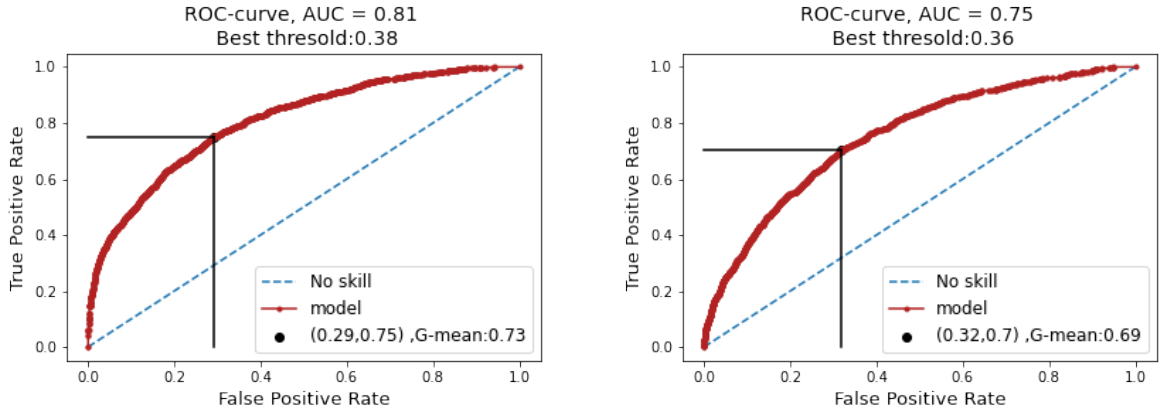
We can see that both the xgboost model and the random forest has both set the feature n_dagerFraTildeltTilOppmote_c as the most influential one for classification in these independent models. Both the Random Forest and the Naive Bayes model uses the feature that explains which day of the week the appointment is scheduled as one of the more influential attributes for the model.

4.2.2 Dependent models

Previously we removed the dependent variables to see if the model worked in a different manner without these variables and because some of them might not be available at the time of prediction. We now put them back in the mix and look at the models which will contain some history for the patient.

RandomForest

The Random forest is the first we will look at with all features involved. As you can see from figure 4.20 the ROC-plot now is more drawn up towards the left corner. A bit more for the training data, which means there could be some overfitting.



(a) Train data with the Random Forest Classifier

(b) Test data with the Random Forest Classifier

Figure 4.20: Comparing ROC plots for Random Forest Classifier train and test set

The confusion matrix with the accuracy and recall score now tells almost the same story about a model which is better at predicting on the training set than on the test set. Still, we can see that values have increased significantly from the models with fewer features. The recall value is actually slightly lower for the follow-up-class, but the accuracy is much higher. Which means we have a lot fewer people that are classified wrongly as follow-ups.

Table 4.4: Confusion matrix for test and training data

	Training Data	Predicted	
		Pass	Follow-up
Actual	Pass	2462	1009
	Follow-up	583	1743

	Test Data	Predicted	
		Pass	Follow-up
Actual	Pass	14007	6575
	Follow-up	269	635

Training data:

- Accuracy: 0.725
- Recall: [0.709 0.749]
- F1-score: [0.756 0.686]

Test data :

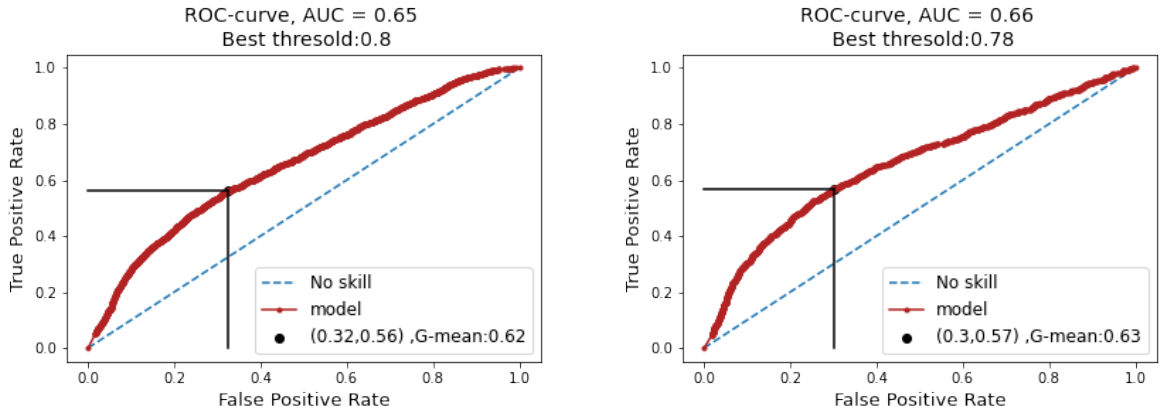
- Accuracy: 0.681
- Recall: [0.680 0.702]
- F1-score: [0.804 0.157]

We see that all of the features which are classified as important for the model has something to do about the waiting time for the appointment.

1. n_kontaktVarighetPerDagerFraTildeltTilOppmote_c
2. n_dagerFraTildeltTilOppmoteIPeriodenPerPas_c
3. n_dagerFraTildeltTilOppmote_c

Gaussian Naive Bayes

In the Gaussian model we can see that it still struggles somewhat compared to the other methods. The ROC curve leans more closely towards the "no-skill"-line and the results presented in this subchapter shows that the performance is overall poorer. The accuracy on the test data is about the same as some of the other models, but the recall value for the follow-up-patients is much lower.



(a) Train data with Gaussian Naive Bayes

(b) Test data with Gaussian Naive Bayes

Figure 4.21: Comparing ROC plots for GaussianNB train and test set

Table 4.5: Confusion matrix for test and training data

	Training Data	Predicted	
		Pass	Follow-up
Actual	Pass	2345	1126
	Follow-up	1021	1305

	Test Data	Predicted	
		Pass	Follow-up
Actual	Pass	14373	6209
	Follow-up	390	514

Training data:

- Accuracy: 0.630
- Recall: [0.676 0.561]
- F1-score: [0.686 0.549]

Test data :

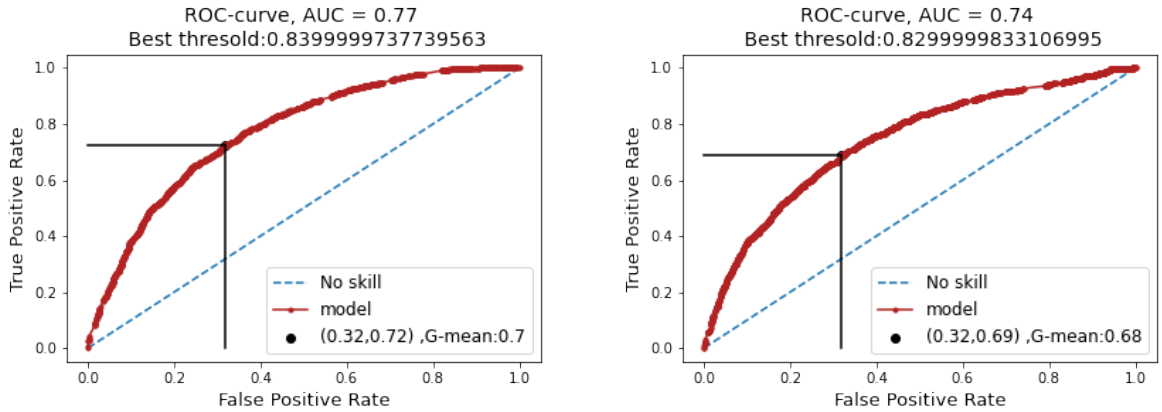
- Accuracy: 0.681
- Recall: [0.698 0.569]
- F1-score: [0.813 0.135]

With all features available the Gaussian Naive Bayes still uses the appointment day of the week and month as one of the three most important features together with age.

1. c_kontaktOppmoteMaaned_r
2. c_pasAlder_r
3. c_kontakt_OppmoteUka_r

Xgboost

This last xgboost model has increased its accuracy in the test set with about 0.04 which is pretty significant. It means that about 800 people less in this model is classified wrongly to the follow-up class and we still get 36 more classified correctly. The accuracy is about the same as the random forest with all features involved, but the the recall value for the follow-up is a bit less.



(a) Train data with the xgboost

(b) Test data with the xgboost

Figure 4.22: Comparing ROC plots for xgboost train and test set. As with the independent models there is a bug in the numpy library that does not round the value for the threshold.

Table 4.6: Confusion matrix for test and training data

	Training Data	Predicted	
		Pass	Follow-up
Actual	Pass	2368	1103
	Follow-up	642	1684

	Test Data	Predicted	
		Pass	Follow-up
Actual	Pass	14030	6552
	Follow-up	283	621

Training data:

- Accuracy: 0.699
- Recall: [0.682 0.724]
- F1-score: [0.731 0.659]

Test data :

- Accuracy: 0.682
- Recall: [0.682 0.687]
- F1-score: [0.804 0.154]

In the same way as the random forest the feature "n_kontaktVarighetPerDagerFraTildeltTilOppmote_c" is classified as the most important one for prediction, but there are two other features featured on the next two places. All of these features were not available when we trained the first xgboost-model.

1. n_kontaktVarighetPerDagerFraTildeltTilOppmote_c
2. n_dagerSidenSisteUtfortEpisodeOver7DagerPerPas_c
3. n_andelPasAvbestPerHenv_c

4.3 Summary of results

In this chapter the features that showed promise when training the models was highlighted. Some of them yields interesting plots where you can see that the patterns of the follow-up class diverges from the ones we expect to show (pass) for appointments. The models also shows some unique behavior where the Gaussian Naive Bayes seems to prioritize features differently than the tree based models.

Table 4.7: This table shows the scores of all the different models for the test data. They have not been trained on the test data and therefore these metrics are often the ones used to compare the models

Independent	**Accuracy**	**Recall**	**F1-score**
Random Forest	0.592	[0.587, 0.704]	[0.804, 0.157]
Gaussian NB	0.640	[0.643, 0.579]	[0.774, 0.119]
Xgboost	0.643	[0.643, 0.647]	[0.775, 0.132]
Dependent	**Accuracy**	**Recall**	**F1-score**
Random Forest	0.681	[0.680, 0.0.702]	[0.804, 0.157]
Gaussian NB	0.681	[0.698, 0.569]	[0.813, 0.135]
Xgboost	0.682	[0.682, 0.687]	[0.804, 0.154]

An important part of the results was to compare the dependent models with the independent models. Overall the performance metrics shows that the models that uses the historic features (thereby dependent) of the patient when training the models perform significantly better (see table 4.7). Even though the accuracy scores are quite similar for the dependent models, we can see that the recall score (amount of correctly classified follow-up patients) goes in disfavor when using the Gaussian Naive Bayes model.

Chapter 5

Discussion

For this thesis, we have looked into the data provided by Helse Vest RHF and tried to figure out what characterizes a patient who does not show up to their scheduled appointment. We have used some more traditional methods, where the data has been been looked at directly and we have set up different forms of visualization, like bar plots, sector diagrams and KDE-plots to see if we could spot some discrepancies or points of interests. We also set up three prediction models, applying some of the more acknowledged machine learning algorithms in Random Forest, Gaussian Naive Bayes (GNB) and xgboost.

5.1 Discussion of model

While the Random Forest and the xgboost are both based upon an ensemble tree model, they have some key differences described in chapter 2.6. The Naive Bayes is fundamentally different in the way that it sets up the model and as we see in the compared models, it diverges a bit from the other models in what features it categorizes as more important.

When looking at the models (see table 4.7), the best ones seldom reach an accuracy higher than about 70 %. This might not seem very high, but human behavior is a fickle thing and is not very easy to predict. Even though the data fed into the model is identical, it does not mean that people are, and the outcome of the appointment might in practice end with two different results. It is especially relevant for psychiatric and addiction patients which have a higher no-show rate than patients from other areas of medicine (Molfenter, 2013).

5.1.1 Optimal model

Discussing and deciding on an optimal model is not as straightforward as looking at the accuracy and saying that one is superior to the other. We have shown multiple models, with different performance metrics, which has their pros and cons regarding interpretability.

If you look at the independent models with GNB we see that the accuracy score for this model can compete with the other two models, but the recall score shows that there a lot more of the type II error (see chapter 2.2) than for the other models. This score also is lower for the GNB trained with all features, but for this model also the accuracy is lower than the

other models. There could be variations for the setup of the parameters, but with our setup it seems that when more features and data are available, the more complex tree models have an advantage for this specific problem. With that said, you could argue for a trial with GNB if the model were to be scaled to include other clinics as well. In that case a model has to be more generalizable across clinics, and a model based on less or different variables might perform better with the GNB.

The Random Forest does seem to get the highest recall value in both the independent and dependent model, but with hyperparameters used for this model it seems that both the independent model (see table 4.1) and the dependent model (see table 4.4) has a small tendency of overfitting the data. This is seen by the accuracy being a bit higher for the training data than it is with the test data. This could probably be improved with smaller trees and more samples in split or in leaf nodes, but training a model with many parameters in grid search, to then check performance in the test data, is a time consuming effort.

Apart from that the Random Forest model with all features available performed very well with 0.681 accuracy and a lower type II error present than the other models.

The last model applied is the xgboost model with all features available and this model seems to perform more stable between the training and test data. It matches the other models on accuracy and is not far behind the Random Forest on classifying no-shows correctly. Between all these model it is therefore very difficult to conclude which model performs best, or which would be best suited for implementation in the healthcare clinic.

The balance between accuracy and recall in this project is a difficult task to judge. You would like to get a high as possible recall score without it worsening the accuracy score, but for these types of models it is often a balance. If you want to increase the models ability to predict the non showing patients, it will often times also get worse at predicting the patients that do show. It is a give and take scenario, where you need to strike a balance on what to prioritize.

If the model is implemented into daily use, a balance will be easier to find when applied to a real life situation to see how the cost and workload is affected. Another thing to consider is whether the implementation and resources invested, is experienced as an improvement for the patient. Will the patient experience the predictions and actions based on the model as an incentive to show up for their appointment, or as an invasion of their privacy? There are many possible pitfalls one would like to avoid in such a scenario.

In general we do see that with all the features available the models performs better, and that coincides with what was presented in chapter 2.4. When we started the project we were also given some research from Helse Vest that suggested that having a patients history was crucial for good performance in the model (Dantas et al., 2018; Molfenter, 2013). The data that was taken out from the first models where mostly specific to the patient history, so this could substantiate the notion that it is important for accurate predictions.

5.1.2 Optimal features

In the results we started by looking at some of the most relevant features manually by visualizing them in different ways. The plots revealed some interesting patterns and tendencies in the data. Some of the features that were highlighted was the time of the appointment, patient age, and patient history. A nice characteristic of the models is the ability to say what features are more important for predictions, a summary can be seen in table 5.1. We see some of the type of features found in the analysis coincides with the top features in the

models. Not all of the features are represented, but in some cases it can just be due to a different variant of the same feature (i.e. ratio vs count).

One thing that we found interesting when looking through the output of our models is that many of the features ranked as most important for predictions were not dependent on the patient. The GNB-model in both the dependent and independent case, listed the features for month and day of the week as two of the three most important features for prediction. This is interesting because it does not necessarily have anything to do with the patient, but it might give some incentive to set patients prone to not showing up with appointments in months and on days where they would be more likely to show. The age feature is dependent upon the patient, but this is one of the features that has been grouped because of anonymization. This could mean a GNB-model could perform better on the data prior to discretization.

Both the Random Forest and the xgboost models characterizes the feature *n_kontaktVarighetPerDagerFraTildeltTilOppmote_c* as the most important feature for prediction when all features are available. This feature tells something about the planned duration of all contacts to be held with the patient. For the xgboost the two other features are more patient specific, but for the Random Forest model the most important features says something about the scheduled time for attendance for the patient. This is interesting, since as stated in the last chapter, patient history is crucial for good performance.

Table 5.1: The most important features for the models when predicting outcomes.

RandomForest	• n_kontaktVarighetPerDagerFraTildeltTilOppmote_c • n_dagerFraTildeltTilOppmoteIPeriodenPerPas_c • n_dagerFraTildeltTilOppmote_c
GNB	• c_kontaktOppmoteMaaned_r • c_pasAlder_r • c_kontakt_OppmoteUka_r
Xgboost	• n_kontaktVarighetPerDagerFra TildeltTilOppmote_c • n_dagerSidenSisteUtfort EpisodeOver7DagerPerPas_c • n_andelPasAvbestPerHenv_c

It is important however to note that when looking at past literature about no-show, that different countries have different approaches and circumstances surrounding their healthcare services. The patients, the culture and setup of healthcare services (public/private) vary, and thereby the results and conclusion in some of those papers may not be the transferable to Norway. The data itself is sometimes very different in what features are gathered.

What features that are allowed is also very different in Norway as we have strong ethical rules, and we will discuss this in section 5.2.2. The granularity in the data caused by the anonymization makes the machine learning harder as well. If we had access to more accurate values for the age and distance to clinic, or possibly if the patient live in an area with decent public transport, we could possibly improve the predictions. The point is that given the circumstances, the provided features are the ones found as most important, there could however be better features available if allowed through legislation.

Hyperparameters

The models are trained as they go through a gridsearch with cross validation, and many different parameters are tested in combination on the training data to find best fit for the model. It is pretty straight forward, but for the Random Forest and the xgboost it can be a time consuming effort and with all the different models that have been tested with different feature selectors and sampling strategies it has taken some time to single out the models we show in this thesis.

The hyperparameters for the models are chosen manually when put through the GridSearchCV-method, which is explained in further detail in chapter 3.9.2. Many more parameters could be tested at this stage. For example the number of estimators for Random Forest could be set even higher to possibly reduce randomness and variation in the tress or we could add more depth to the trees to find more subtle patterns, but this could potentially cause the model to increase variance. One could further experiment with these parameters to find better models.

The Gaussian Naive Bayes does not have as many possibilities for hyperparameter-tuning. The only thing that may be worth changing is the prior, but that is if you have a good estimate for the prior, else it uses the dataset and count the classes for the prior values. This could be looked into, and for the models we used with undersampling it could be a good idea to change the prior, because it might think the balance between classes is more symmetric than what is the actual case, but this will be for someone to check in the future.

5.1.3 Model transparency

One of the things we originally set out to do with these models, was to develop a model that was not a complete black-box model. We wanted to find models that are somewhat explainable both for patients and healthcare workers. Now, when comparing the different models, we made sure that the most important features was given as outputs, so that it is easier to explain to patients why actions due to model prediction has been executed. Healthcare workers can point to the features generated from the models as part of the reasoning for the intervention. This is the same across all three models, which means that they all have this option. However, if we want a more detailed explanation and more transparency it is clear that random forest and xgboost is much better suited as one can refer to the trees as seen in figure 5.1 to explain why a patient was predicted as a follow-up candidate.

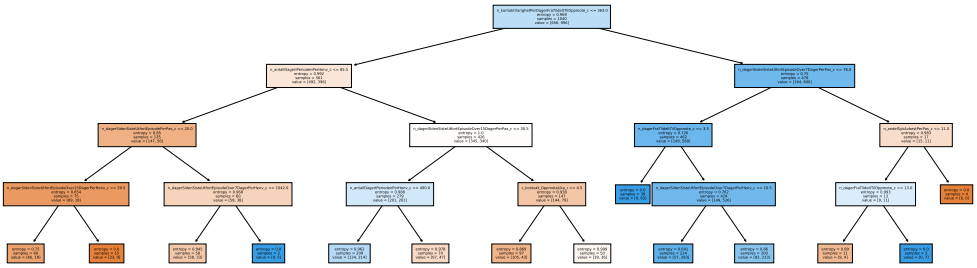


Figure 5.1: The Random Forest Classifier can print out the trees in the models, like this one. Right here the scale is small, so it is hard to make out the different features and conditions on each step, but it makes the model easy to interpret.

5.1.4 Independency assumption

At the start of our project we were led to believe that each row in the dataset could be seen as independent from each other, even though the same patient could be represented multiple times. It was not until much later that we discovered and were informed that many of the features and rows are both dependent and based upon earlier data from patients. Still, there was an agreement to continue the work as if each row were independent, because this is an underlying assumption made already by Helse Vest in their ongoing project. Even though we are making the same inaccurate assumption of independence, the models that contain the dependent features outperforms the independent ones. In most cases the features are historic and they supply more information for the prediction even though they are dependent. They are available at the moment of prediction, so they can be used although they are based on previous rows.

One thing that is important to remember here, especially since we do not have insight about the clinic and information about which data that belongs to which patient, there could be single patients that have a lot of rows in the dataset assigned to them. If this is the case, and if that actual patient behaves in a way that is representative for the user base, a single patient might actually affect how the model performs.

5.1.5 Computational Power

One of the largest obstacles in this thesis has been the hardware provided. Because of the strict protection of the data for the patients we had to work on the computer provided by Helse Vest. It was not the most powerful laptops and the preprocessing, analytics, plotting, training the model and producing outputs took a lot time. After looking through the models, tweaking them and running the code multiple times, time went by quickly and some point we had to stop experimenting, and focus our efforts on wrapping it up. At the end of the project there are still multiple things we would have liked to explore further which is not possible due to the time constraints and the limited hardware power. Some of these explorations are summarized in the "Future Work" section.

5.2 Practical application and challenges

5.2.1 Anonymity

As stated earlier in the thesis the provided patient data was partly pre-processed. The id of the patients had been stripped, and some features were generalized such as the age and distance from the hospital. The features like where they live, their sex, or other features that could be used for re-identification had been plucked from the dataset. All to ensure that it is anonymized to a k-anonymization of k=5, so that sensitive information is untied to specific patients and GDPR is fulfilled. This is important for the privacy of the patients and that ensures protection for potential misuse of the data if they were to go astray. As a consequence of anonymization, we had to address how to handle the grouped data, and discuss how the loss of granularity and features affects the models.

There has been extensive processes as part of the pre-processing in dealing with the generalized and somewhat incomplete data. However, we do not know if the way it was generalized as part of the anonymization is the optimal way for maintaining the performance, as it drops due to anonymization as shown Wimmer and Powell (2014). For instance the decision to divide the intervals for the age of patients could be reconsidered to better represent stages of life and their effects on appointment outcome. Maybe 20 to 27 years and 28 to 35 are better groups to capture these nuances? A point that was made as part of the initial analysis was also the uneven sizes of the subgroups for the different ages. Possibly the groups or bin-size could be made so that each subset was more uniform with the help of adjusted intervals.

The distance being similar, and location being undisclosed was also a challenge. There are differences for rural and urban areas according to Molfenter (2013). For the modelling knowing if the location or the distance is within an area of decent cover of transport service could be useful. Is it within a city or in a rural area with longer distances? Maybe one of those locations are more likely to have residents with their own vehicles and therefore easier access to the clinic. The purpose of this section is to briefly highlight that there are some aspects of the objectives' context that we did not have control over, and to discuss what impact some of the anonymization could have had on the final result. Without access to the prior data, it is not possible to investigate this further.

5.2.2 Ethics and discrimination

Some information about the patient will not be available to use for a project like this at all, even though they could improve the performance of the model. Other than ensuring GDPR rights, anonymization also has an ethical aspect to it. As part of the anonymization, features in the data which can lead to discrimination are dropped. A feature that could give a performance boost for a model like this, is the household income for the patient, but due to Norwegian law this was not prescribed in this project (see chapter 2.1). It could be possible to use, but there would be a long process to see how it would work on a legal level. At the same time it could be hard to procure this data about the patient. The reason is that one should not be judged solely on an automated process, especially where some groups can lead to more frequent target prediction (follow-up) based on things they themselves cannot control. The ethnicity of a patient could give a more accurate model, but this would be based on a irrational prejudice and would be considered discrimination if used for predictions.

The fact is that in a society certain ethnic groups have been discriminated to some degree, so some of these groups could have certain disadvantages, like weaker socio-economics. Thereby ethnicity, sex and religion is not part of the features that are taken into consideration. If a model were to incorporate this kind of patient information, there would be a risk of built-in discrimination bias in the prediction. This kind of prejudice and built-in discrimination in machine learning has been described in the paper by Pappada and Pauli (2018), where it shows it is important to detect and eliminate these both from an ethical and legal point of view. Thereby most information in the data provided is related to very basic patient information, or details around the appointments and past history. This reduces the risk of discrimination in any of the models based on these features.

5.2.3 Concept drift and practical application

An important stage of CRISP-DM (see figure 3.1) is the deployment stage. The deployment stage has not been discussed in great detail up until now as the main focus of the thesis has been the prior stages about the data and the development of the model itself. After working on the latter stages, we wanted to devote some attention to discuss some of the challenges we expect to see in the future with the deployment of the model (or similar models) considering the current plans for application. We decided to look into the field of "concept drift" and how it can affect the application of models such as this.

Concept drift in follow-up predictions

The reason we bring concept drift into the discussion is that we want to prepare the future development of a model like this at Helse Vest about some of the coming obstacles. In the theory we explained the concept of concept drift. The discussion is to point out likely challenges to come, although it is mostly speculation based on assumptions and what we have read about it in relevant literature. We do not have the possibility to test it in practice until it is deployed, but it is important to plan ahead for the possible pitfalls. Currently the plan for the application of the predictions of the no-show model (or follow-up in our case), is to initiate some sort of intervention. This is possibly just an additional text or a phone call to candidates that have high probability of not showing up a few days prior to the appointment. The intention of this intervention is to try and reduce these no-shows for the good of the patient, but also for the benefits of reducing costs and wait times for others in turn as well.

The problem with the current planned application is that it is very susceptible to concept drift in the model as with the current practice it intervenes **before** an outcome is observed. A candidate prior to the appointment can bear all the traits and features of a candidate for follow-up. If the intervention takes place prior (and debatably even after) to the outcome, candidates' behaviour might alter. The observation is then a subjective observation due to the intervention. This means a candidate with features resembling a typical follow-up (or no-show), can be classified as a regular show as a result of an intervention. If the interventions are effective enough and continues over time, the model performance will decay as the characteristics or patterns of a follow-up candidate evolves. It is important to reiterate that this is all based on assumptions, like assuming the intervention has an effect, and there are more dimensions to it as well. The intervention could only have short term effect on

one appointment for instance, but it could also depend on what type of intervention is applied. Either way, it raises some questions around whether the application of this type of a model should change in order to be more sustainable over time. A different application could reduce the rate of decay due to the interventions. In literature, some of the no-shows predictions studies have the goal of developing a model with predictions of no-show to perform double-booking (see Huang and Hanauer (2014)). Applying a model this way will not directly affect the target variable through its practices as it is a response to the outcome of an appointment rather than aiming at changing a patients behaviour and decisions. This approach could be more sustainable based on what we have seen in the literature surrounding concept drift. It will not completely remove concept drift, but the data will not drift directly because of healthcare intervention on patients. The double booking approach is currently not allowed in Norway due to article 22 in the GDPR as mentioned in the theory.

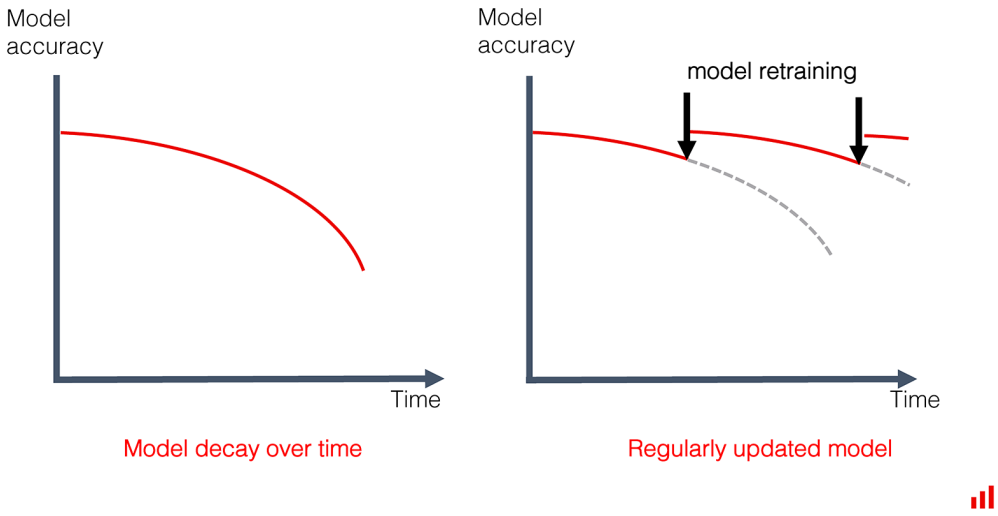


Figure 5.2: An illustration of the decay in models in terms of performance, and with regular maintenance and retraining (Samuylova, 2020).

Probably the most common and simplest way of handling concept drift is simply by retraining models as their performance drop below a certain threshold (see figure 5.2). It can either be done periodically as manual learning where you manually re-develop the model (like the process in this thesis). Or by having an automated system that can continuously evaluate and monitor the performance, that re-trains the model as soon as it drops too low. In all these cases the frequency and threshold for re-develop or re-train must be considered. It is important to note that in the CRISP-DM process, the circle can go on indefinitely between the data, model, evaluation, and deployment. With the need of maintenance, there is more work past the initial deployment as well. The process of maintaining and updating models is something that must continue indefinitely, and as mentioned there are different ways of re-training the models, either manually or automated. Part of this solution should also be to introduce a new feature for the interventions in the model so that it can be taken

into account when trying to find new patterns for the targeted variable. This also means that the data might have to be divided into pre- and post-model-intervention data as the context will be different for the target after the model deployment. Concept drift is a known field within machine learning, where this section barely scratches the surface in an attempt to bring attention to the phenomenon.

Concept drift due to the COVID-19 pandemic

Another challenge we see with the current models, is that we predict a change in the data due to the COVID-19 pandemic. We do not know how the clinic has adapted it routines, but assuming there have been changes in the year 2020, the data will probably also be different. We do not know if the clinic for instance introduced digital options, or reduced the daily capacity but we assume changes will be seen in the data for both 2020 and 2021 due to the pandemic. This will change the landscape in the data, meaning that depending on how big changes there has been in these years and whether new routines have been introduced, the data can paint a different picture of typical follow-up candidates. Most of this, however, is again speculation as we have no insight in specifically how the clinic has dealt with the pandemic, and the challenges there. We just assume the pattern might have changed or drifted, meaning that the post-pandemic model most likely will have to be trained more similar to the pre-pandemic data, unless there has been new practice changes, like online meetings or such.

5.2.4 Possible application and practices

In light of the described context and challenges around applications of these models we would like to discuss some of the possibilities around the way a model is used. As highlighted in the last sections, double booking has the incentive of possibly performing better in terms of concept drift, however, it is not currently legal. So how can we use the predictions of our follow-up candidates? The prediction of the model was already coined as follow-up, which means that there is an intention for reactively doing something with the candidates that are either no-show or cancelling late. A discussion should be had around what way the intervention or patient follow-up routine should be. It could as previously mentioned, just be a reminder, however if you want to change behavior for the better, you likely need something that is not as easy to ignore. There are many possibilities here, but it could for instance be that patients actively have to confirm their appointments, so that they are further obligated to meet up. In a study by Molfenter (2013), behaviour engagement strategies was one of the more efficient approaches, which consisted of contingency management and motivational interviewing. The largest challenge is to find approaches that outweighs the cost-benefit and which are not too complicated to implement.

Our data did not have a profile for individuals, but as seen in some papers (i.e. (Huang and Hanauer, 2014)) and also in our results that patients prior history is important. In the model itself, this can be improved on by having possibly more features around this. It is important to distinct patients who have a one time slip up, such as a forgotten appointment, and those who have recurrent and systematic absence. This should be taken into consideration when deciding what way to follow up a patient, where a long time no-shower should be taken in for a more thorough follow-up intervention.

As mentioned in last section, the recent year has shown new and creative ways of offering services remotely and digitally due to the pandemic, so there could be some possibilities here as well. With the models, several of the predictions come with a "probability", where the type of follow-up routines could be moderated by the probability of a follow-up. These applications rely on reactively changing the outcome of an appointment, so the underlying risk of deteriorating the model over time is present. Other than hoping on improving the no-show rates, and reducing wait lines and costs, there are also other possible applications for the model. We believe that having access to a model like this can be useful for the day to day planing of the people offering these healthcare services. Having a way of forecasting which appointments are at risk of not happening, could prove useful for short-term planning for the healthcare workers involved. This means that healthcare possibly could plan ahead for a more efficient work day, i.e. when to do paper work. Using the model in this way, does not directly affect the patient, and would thereby not add to the concept drift directly. It could also be something that is done in parallel with follow-up either way. There are in other words, many possibilities for the application of the model itself. Which applications to use has to be experimented on in practice, and reviewed over time to see their possible effects.

5.3 Future work

In the future more and more models like this will be used to give patients a better healthcare offer. Hopefully without increasing the workload of the healthcare workers, or in a best case scenario actually decreasing it and at the same time cutting costs. Before getting there, multiple aspects of such a model has to be assessed. The legals of launching such a model is massive, because every aspect of a patients privacy and rights has to be considered. Then there is the actual practice of how it is to be used. What incentives could you potentially implement for a patient to increase probability of them showing up without it being to invasive and how do you check if the model actually performs. As discussed in the chapter about concept drift, this is not as straight forward as it seems because of the fact that if you go in and affect the outcome of a target value the performance of the model might decrease.

There has been some literature (Goldsteen et al., 2020) that optimizes the anonymization process with the help of machine learning, where it can find the optimal anonymization based on the performance of the applied model. This could be interesting to investigate further, however it requires access to the un-anonymized patient data, where it is anonymized as part of the model development. It is something that was outside our scope, as we do not have access to the prior data. It could however be something for Helse Vest RHF to take into their further development.

These aspects are more on the practical use of the model, but there are multiple things that could be looked into directly related to our work as well. In this thesis we focused only on three type of models that have been trained and tested. There are many more and it could be interesting to investigate mplregressor (ANN) or maybe logistic regression models, to compare how these models perform.

There are also other things to investigate, like try out other ratios on the sampling methods or maybe just try other sampling methods. BorderlineSMOTE for example tries in the same way as SMOTE to synthesize new data based on the old data, but only in the region that it classifies as just between two classes because this is the place where it is harder to

distinguish classes. There could also be worth investigating other feature selecting methods or maybe just try the Extra Trees Classifier with other hyperparameters and see if other features would be selected.

One thing that should be done, which was a little bit late into the project before we figured out, is to implement the feature selector as part of the pipeline in sklearn. This should not make an impact on performance, but could streamline the process of training the model and maybe lead to a better method for testing what value k should get in the `SelectKBest`-method.

In the end, we could also look into better methods of testing hyperparameter options for the model. In our project we used the `GridsearchCV`-method, but the `RandomizedSearchCV`-method could possibly make it easier to test a wider range of hyperparameters without increasing training time. It works by testing random selection of the grid that has been set up and choose the best model.

Chapter 6

Conclusions

6.1 The project

The objective of this master thesis has been to investigate and explore how machine learning can be applied in appointment predictions (no-shows) at a psychiatric clinic. The project was approached with a CRISP workflow (see figure 3.1), the industry standard for data mining. Thereby the first stage of this project was studying up on relevant theory that could help us understand the business aspect of a healthcare service, and how machine learning has been used before. The studies we read were not performed in Norway and have therefore very different circumstances, with other rules, laws and regulations to follow. Still, it gave us some ideas on how to proceed with our project, and it was also useful for the data understanding and preparation (CRISP).

With the processed data, we explored the data by visualizing it with different types of plots to investigate and search for relevant features for the prediction model. The initial steps lead to a better understanding of the context, and the issues surrounding patient no-shows. In light of the challenges in the project, we found that a more practical approach to the target variable was to redefine the standard target of *no-show* to a *follow-up* candidate instead. Follow-up candidates are any patients that do not show up on the day, or that cancel less than 3 days beforehand. The purpose was to adjust it for the intended practical applications (forecasting 3 days in advance), but it also helped out with the imbalance of the different outcomes.

With the theoretical background and most of the preparation of the data finished, the project moved onto the modelling. Other than finding the right type of model, optimization and fine tuning of each model was a big part of the evaluation stage of this project. In many ways this was work that could have continued indefinitely. As addressed throughout this thesis, finding the optimal or best model for this kind of problem is a challenge in itself. There are no obsolete performance metrics; Some models score better on accuracy, while others score better on recall or precision. There is also a limit due to the very nature of human behaviour to how well a model like this can perform at all. That means several metrics have to be taken into consideration when looking for the right model together with its practical advantages. Openness and transparency of the model was sought after as GDPR and Helse Vest requires a model that is easy to interpret, both for the patient and the healthcare workers. This is one of the aspects that also has to be taken into consideration when picking

a model. We found that in most cases that both the xgboost and the Random Forest Classifier performed somewhat better than Gaussian Naive Bayes. Xgboost and Random Forest also have the possibility to print out their underlying decision trees. That means these models can help with the transparency of these processes, where healthcare workers can show the trees to patients and explain it's predictions if requested. All these models could however show which features that made the most significant impact for the decision.

As mentioned earlier, the art of predicting human behaviour is a difficult one. When we went into the project we had a preconception that a patient's individual properties would be vital for the prediction outcomes, and in some cases they are. However many of the features prioritized by the models were not necessarily patient specific, but more of a logistical type. Like what day of the week or month the patient appointment was scheduled, or the duration from scheduled appointment till it was held. Even so, we can see that the models trained in this project still have some skill at forecasting what patient will show, and not show. A deep dive in the theory suggested that making profiles with history of the patient was vital for good performance in our models, but we have seen that these are not the only features important for prediction. Either way this means that it can be worthwhile investigating which scheduling times are at risk for certain patient groups. We also had some discussion about the application and future deployment of these models, where we see some challenges to come related to the data for the next few years. It is hard to make any specific suggestions in light of this without any insight into the new data, or the changes at the clinic itself.

One of the most exciting steps was when we stumbled onto the area of *concept drift* when looking into the deployment stage. We had little knowledge about it beforehand, and it was exciting to read about it and implementing it as part of our discussion. We learned that these models will need maintenance to prevent degradation of the future model and it's predictions. Some suggestions about what to do to avoid or reduce it has been discussed, but it cannot be tested before the model is deployed.

6.2 What have we personally learned?

Working on this thesis has been fruitful in many ways. We had a project with an exciting problem, and with an opportunity to apply a lot of the knowledge attained throughout the last few years of our masters degree. It has been exciting to work on *real data*, and on a *real case* within a field we initially knew little about, no-show predictions in health care. It has been motivating to work on a project proposed by Helse Vest RHF, knowing that the work invested here could potentially be useful for their ongoing project. The master thesis has been an excellent way of experiencing the CRISP workflow, and seeing first hand how the different stages are intertwined. Past projects in past courses have had somewhat idealized and simplified cases, where as in this thesis we got to experience it in practice and at a greater extent. Generally we hope our discoveries, discussions and methods will be of interest and use for Helse Vest RHF.

Throughout the thesis we have had meetings with Helse Vest with our latest findings. It has been vital for our work, where we have presented our progress for feedback and to clarify questions we have had along the way. Having to work with a business, communicating the ongoing work, the plan and their expectations has been major part of the experiences attained. It has been very insightful, but also challenging and frustrating at times.

Working on this project has required us to refine a lot of our existing knowledge and pro-

gramming skills. We had to address various challenges such as working with an imbalanced and anonymized data set. Trying to work around these challenges in the visualization and the model development has been a decent learning curve that we feel we managed to overcome. Both of us have experiences with previous masters, but co-writing a masters thesis has been a completely new experience. It has been a challenge overall surrounding the hardware and workspace requirement on Helse Vest's computer systems. Having to sync up our coding and writing was also a big part of the challenge. However, cooperating on a project like this has been absolutely priceless as well. We have had an excellent opportunity for bouncing off ideas and observations we have made throughout the thesis. There has been a lot problems to solve, and having a partner to discuss possible solutions with has been very useful. Although we did not always agree, we always managed to find a compromise. It all allowed us to extend much further and thoroughly at each stage of the thesis. In order to make it work with two of us co-writing and solving theses issues, we had to plan ahead and delegate different roles and responsibilities throughout the project. Overall we are pleased with how well the cooperation has worked with each other and Helse Vest.

The project gave us experience in having to solve a problem for an arbitrary business and field we had little knowledge about beforehand. The overall process of attaining necessary understanding, processing the data, developing models and evaluating them, was filled with many obstacles to overcome. We believe this is the most important experience overall, as with most projects within applied data science it is all about bridging machine learning with the unknown.

Appendix A

Plots with undersampled data

In this section you will find a sample of random seeds for the of some of the same selected KDE-plots for the "Follow-up" target value for other random undersamples. All of these used the same settings as discussed in the results, imblearn's random undersampler with a ratio of 0.7. We see that some of these observations are consistent for with each random under sample, and that the selected plots in the results are not just cherry picked.

Plots with undersampled data

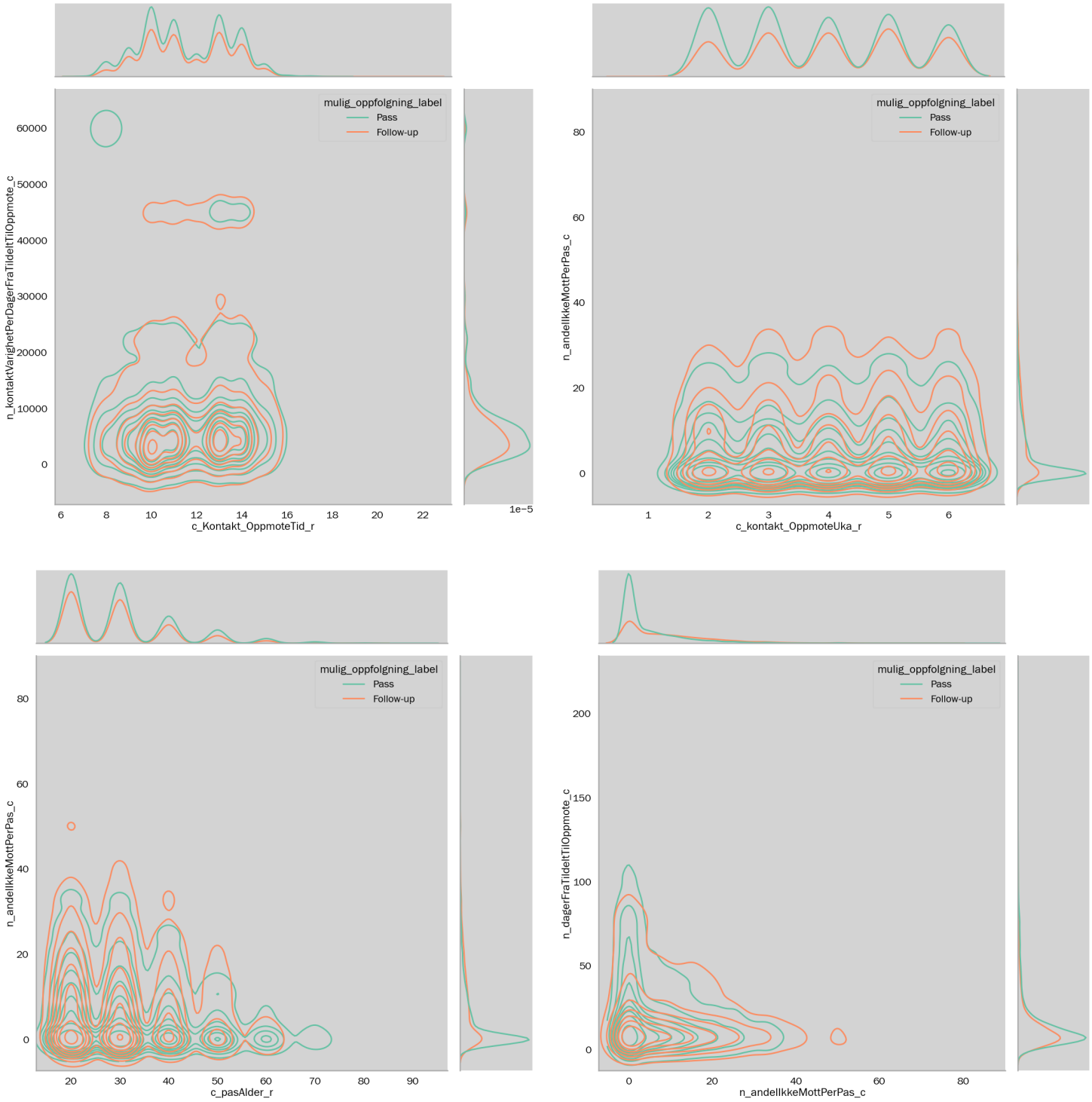


Figure A.1: Example KDE-plots for random under sampling with seed 687.

Plots with undersampled data

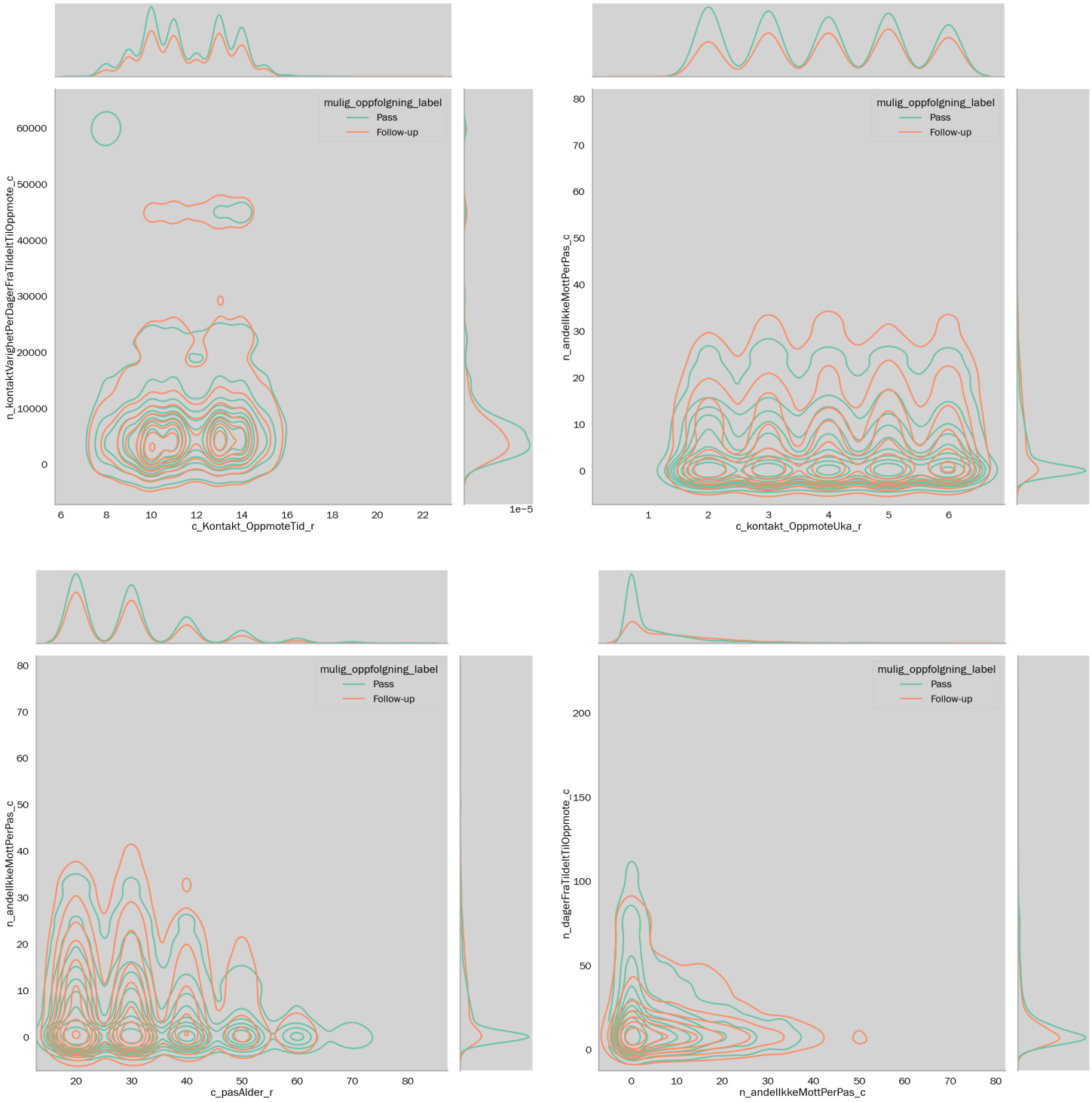


Figure A.2: Example KDE-plots for random under sampling with seed 1281.

Plots with undersampled data

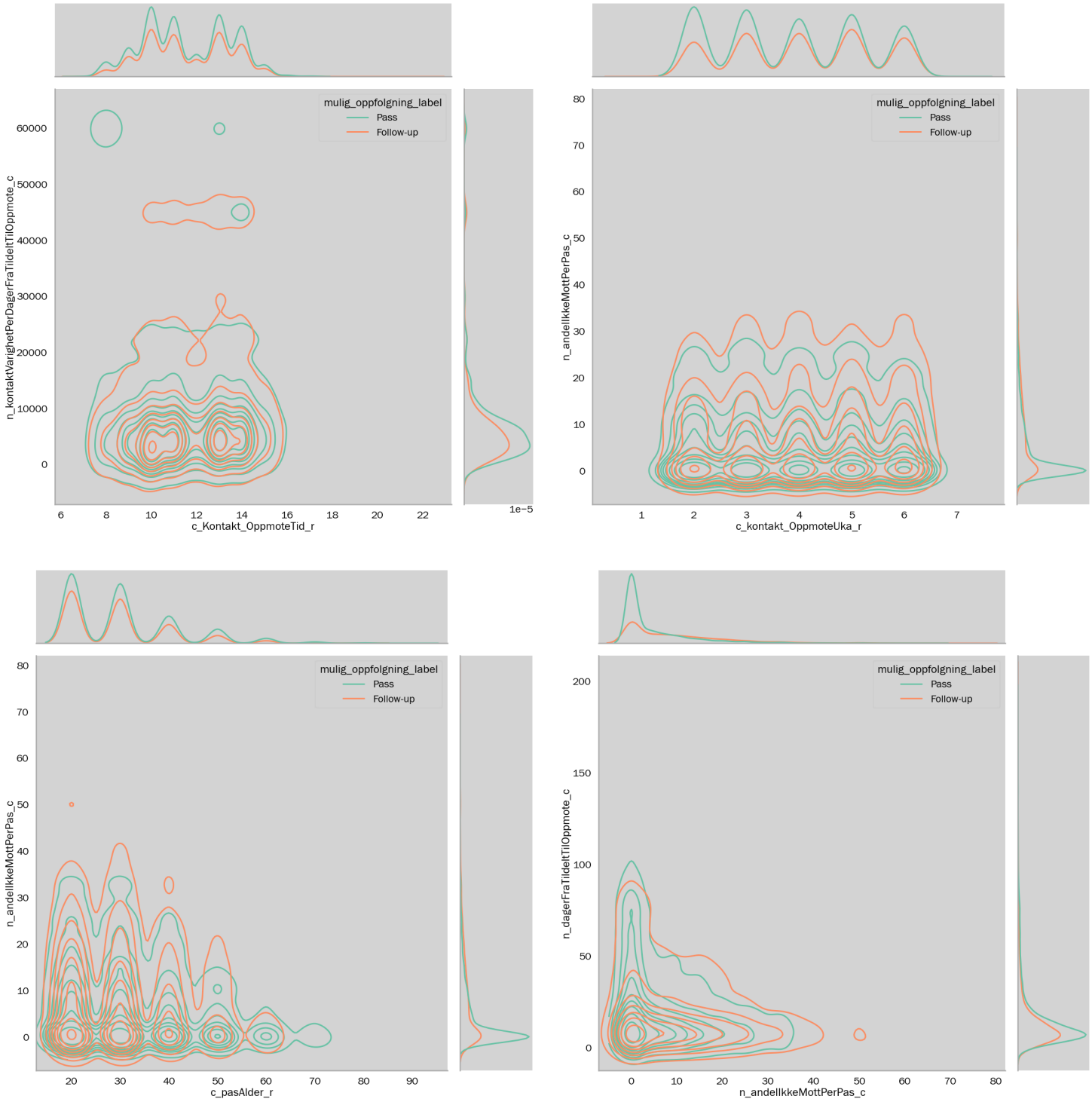


Figure A.3: Example KDE-plots for random under sampling with seed 8464.

Appendix B

Data variables and description

This appendix covers a more detailed list for the variables in the data set provided by Helse Vest RHF. The data was provided as CSV-files and we also got an excel file with information for the different variables/features. The description is for both the training and test set. The variable names are descriptive explanations in Norwegian, followed by a column with more in depth description of the variable in English. Not all of these description are complete as they did not unveil all the information about all the variables. The column a the of datatype has the following types:

- Bool - Boolean, binary values.
- Int - Numeric value, sometimes categorical
- Char - String value, categorical features.

The column "Available" is meant to be "Available at the time of prediction". Not all variables can be used in the model as they are estimated or gathered close before an appointment or after. Some of the values in "Available" are set to "Predicted" which means it is the target features (to be predicted). Some description fields are marked ????, where Helse Vest could not disclose the full explanation of the feature.

Data variables and description

Variable name	Variable description	Datatype	Available
b_erNyPasient_c	New patient (1 = YES, 0 = NO) For this and other variables, "new" means new in the context of the time period of the dataset. Thus, a patient is regarded as a new patient if the scheduled appointment is the first appointment in the dataset belonging to this spesific patient.	Bool	Yes
b_erNyPasientIHenv_c	New medical referral? (1 = YES, 0 = NO) The medical referral is regarded as new if the scheduled appointment is the first appointment in the dataset related to this spesific referral.	Bool	Yes
b_kontaktErDirekteTime_c	The appointment time is set directly (1 = YES) or a tentative time slot is used before a final appointment time is confirmed (0 = NO). In both cases, the patient is only informed about the final appointment time.	Bool	Yes
b_kontaktOnskerPaaminning	Has the patient of the scheduled appointment given written consent to recieve SMS reminders for upcoming appointments (0 = No registered answer, 1 = YES, 2 = NO)	Int	Yes
b_kontaktPaaminingSendt	Has an SMS reminder related to the scheduled appointment been sent to the patient? (1= YES, 0 = NO)	Bool	No
c_henvFagomraade	The medical field of the scheduled appointment's referral. Pseudonomized, i.e. the numbers can not be used to find the spesific field names, just used to separate between the different fields.	Int	Yes
c_henvType	Referral type (1= Assessment , 2 = Treatment, 3 = Check-up)	Int	Yes
c_kontakt_OppmoteTid_r	Scheduled appointment time, hour	Int	Yes
c_kontakt_OppmoteUka_r	Scheduled appointment time, weekday (1 = Monday, 2 = Tuesday, 3 = Wednesday, 4 = Thursday, 5 = Friday, 6 = Saturday, 7 = Sunday)	Int	Yes
c_kontaktAvsluttkodeID	Appointment exit code, numeric (305 = No-show, 308 = Canceled by patient, 314 = Appointment conducted)	Int	Predicted
c_kontaktAvsluttkodeNavn	Appointment exit code, text	Char	Predicted
c_kontaktOmsorgsNivaa	Care level of the scheduled appointment (1 = Residential, 2 = Day treatmeant, 3 = Outpatient clinic)	Int	Yes

Data variables and description

c_kontaktOppmoteMaaned_r	Scheduled appointment time, month (1 = January, 2 = February, 3 = March, 4 = April, 5 = May, 6 = June, 7 = July, 8 = August, 9 = September, 10 = October, 11 = November, 12 = December)	Int	Yes
c_kontaktType	Appointment type (1 = Assessment , 2 = Treatment, 3 = Check-up)	Int	Yes
c_pasAlder_r	The patient's age group. Each age group consists of 10 years, e.g. [20-29)	Int	Yes
c_sistePLKontaktAvsluttKode _c	Exit code for the previous appointment, i.e. the patient's latest appointment that precedes the scheduled appointment in question. Missing value if this is the patient's first appointment in the dataset. See "DIM_c_sistePLKontakt AvslutKode_c.xlsx"	Int	Yes
n_andel_PasAvbestPerPas Over7Dager_c	Historic cancellation ratio for the patient of the scheduled appointment, where the cancellation finds place 7 days or more before the appointment time. The patient's number of previous appointments with c_kontaktAvsluttkodeID = 308 (registered at least 7 days before the appointments), divided by the patient's total number of previous appointments (which is present in the dataset).	Int	Yes
n_andelIkkeMottPerPas_c	Historic no-show ratio for the patient of the scheduled appointment. The patient's number of previous appointments with c_kontaktAvsluttkodeID = 305, divided by the patient's total number of previous appointments (which is present in the dataset).	Int	Yes
n_andelPasAvbestPerHenv_c	Historic cancellation ratio for the referral of the scheduled appointment. The number of previous appointments related to the referral in question with c_kontaktAvsluttkodeID = 308, divided by the total number of previous appointments related to this referral (which is present in the dataset).	Int	Yes
n_andelPasAvbestPerPas_c	Historic cancellation ratio for the patient of the scheduled appointment. The patient's number of previous appointments with c_kontaktAvsluttkodeID = 308, divided by the patient's total number of previous appointments (which is present in the dataset).	Int	Yes

n_andelPasAvbestPerPas3Dager_c	Historic cancellation ratio for the patient of the scheduled appointment, where the cancellation finds place 3 days or less before the appointment time. The patient's number of previous appointments with c_kontaktAvsluttkodeID = 308 (registered at most 3 days before the appointments), divided by the patient's total number of previous appointments (which is present in the dataset).	Int	Yes
n_andelPasAvbestPerPas7Dager_c	Historic cancellation ratio for the patient of the scheduled appointment, where the cancellation finds place 7 days or less before the appointment time. The patient's number of previous appointments with c_kontaktAvsluttkodeID = 308 (registered at most 7 days before the appointments), divided by the patient's total number of previous appointments (which is present in the dataset).	Int	Yes
n_andelSykAvbestPerHenv_c	Historic hospital cancellation ratio for the referral of the scheduled appointment. The number of previous appointments related to the referral in question that have been cancelled by the hospital (not by the patient), divided by the total number of previous appointments related to this referral (which is present in the dataset).	Int	Yes
n_andelSykAvbestPerPas_c	Historic hospital cancellation ratio for the patient of the scheduled appointment. The number of the patient's previous appointments that have been cancelled by the hospital (not by the patient), divided by the patient's total number of previous appointments (which is present in the dataset).	Int	Yes
n_andelUtfortElekEpisoder IPeriodenPerHenv_c	Number of historic elective appointments per day, for the referral related to the scheduled appointment. The number of elective appoitments related to the referral in question present in the dataset, divided by the dataset's day count.	Int	Yes
n_andelUtfortElekEpisoderIPeriodenPerPas_c	Number of historic elective appointments per day, for the patient of the scheduled appointment. The patient's number of elective appoitments present in the dataset, divided by the dataset's day count.	Int	Yes

n_andelUtfortOHEpisoderIPeriodenPerHenv_c	Number of historic urgent appointments per day, for the patient of the scheduled appointment. The patient's number of urgent appoitments present in the dataset, divided by the dataset's day count.	Int	Yes
n_andelUtfortOHEpisoder IPeriodenPerPas_c	Number of historic urgent appointments per day, for the patient of the scheduled appointment. The patient's number of urgent appoitments present in the dataset, divided by the dataset's day count.	Int	Yes
n_antallDagerIPerioden PerHenv_c	???????	Int	Yes
n_antallDagerIPerioden PerPas_c	???????	Int	Yes
n_antallIkkeMottPerPas_c	Historic no-show count for the patient of the scheduled appointment. The patient's number of previous appointments with c_kontaktAvsluttkodeID = 305 (which is present in the dataset).	Int	Yes
n_antallKontakterPerHenv_c	Number of historic appointments for the referral of the scheduled appointment in question.	Int	Yes
n_antallKontakterPerPas_c	Number of historic appointments for the patient of the scheduled appointment in question.	Int	Yes
n_antallPasAvbestPerHenv_c	Historic cancellation count for the referral of the scheduled appointment. The number of previous appointments related to the referral in question with c_kontaktAvsluttkodeID = 308 (which is present in the dataset).	Int	Yes
n_antallPasAvbestPerPas_c	Historic cancellation count for the patient of the scheduled appointment. The patient's number of previous appointments with c_kontaktAvsluttkodeID = 308 (which is present in the dataset).	Int	Yes
n_antallPasAvbestPerPas3Dager_c	Historic cancellation count for the patient of the scheduled appointment, where the cancellation finds place 3 days or less before the appointment time. The patient's number of previous appointments with c_kontaktAvsluttkodeID = 308 (registered at most 3 days before the appointments).	Int	Yes
n_antallPasAvbestPerPas7Dager_c	Historic cancellation count for the patient of the scheduled appointment, where the cancellation finds place 7 days or less before the appointment time. The patient's number of previous appointments with c_kontaktAvsluttkodeID = 308 (registered at most 7 days before the appointments).	Int	Yes

n_antallPasAvbestPerPasOver7Dager_c	Historic cancellation count for the patient of the scheduled appointment, where the cancellation finds place 7 days or more before the appointment time. The patient's number of previous appointments with c_kontaktAvsluttkodeID = 308 (registered at least 7 days before the appointments).	Int	Yes
n_antallSykAvbestPerHenv_c	Historic hospital cancellation count for the referral of the scheduled appointment. The number of previous appointments related to the referral in question that have been cancelled by the hospital (not by the patient).	Int	Yes
n_antallSykAvbestPerPas_c	Historic hospital cancellation count for the patient of the scheduled appointment. The number of the patient's previous appointments that have been cancelled by the hospital (not by the patient).	Int	Yes
n_antallUtfortElekEpisoderIPeriodenPerHenv_c	Number of historic elective appointments, for the referral of the scheduled appointment.	Int	Yes
n_antallUtfortElekEpisoderIPeriodenPerPas_c	Number of historic elective appointments, for the patient of the scheduled appointment.	Int	Yes
n_antallUtfortOHEpisoderIPeriodenPerHenv_c	Number of historic urgent appointments, for the referral of the scheduled appointment.	Int	Yes
n_antallUtfortOHEpisoderIPeriodenPerPas_c	Number of historic urgent appointments, for the patient of the scheduled appointment.	Int	Yes
n_avstandKomSyk	Distance between the patient's municipality and the hospital's municipality, given in kilometers.	Int	Yes
n_dagerFraAvbestTilOppmote_c	If the appointment was canceled, this variable describes at what point in time the cancellation found place, measures in number of days before the scheduled appointment time.	Int	No
n_dagerFraTideltTilOppmote_c	How early was the patient informed of the scheduled appointment? Given in number of days prior to the scheduled appointment time.	Int	Yes
n_dagerFraTildeltTil OppmoteIPeriodenPer-Henv_c	???????	Int	Yes
n_dagerFraTildeltTil OppmoteIPeriodenPer-Pas_c	???????	Int	Yes
n_dagerSidenSisteUtfort EpisodeOver15DagerPer Henv_c	Number of days between the referrals's last conducted appointment (c_kontaktAvsluttkodeID =314) and the scheduled appointment in question. This "last appointment" must have been conducted at least 15 days before the scheduled appointment (so it is not necessarily the most recent appointment)	Int	Maybe

Data variables and description

n_dagerSidenSisteUtfort EpisodeOver15DagerPer Pas_c	Number of days between the patient's last conducted appointment (c_kontaktAvsluttkodeID =314) and the scheduled appointment in question. This "last appointment" must have been conducted at least 15 days before the scheduled appointment (so it is not necessarily the most recent appointment)	Int	Maybe
n_dagerSidenSisteUtfort EpisodeOver7DagerPer Henv_c	Number of days between the referrals's last conducted appointment (c_kontaktAvsluttkodeID =314) and the scheduled appointment in question. This "last appointment" must have been conducted at least 7 days before the scheduled appointment (so it is not necessarily the most recent appointment)	Int	Maybe
n_dagerSidenSisteUtfort EpisodeOver7DagerPerPas_c	Number of days between the patient's last conducted appointment (c_kontaktAvsluttkodeID =314) and the scheduled appointment in question. This "last appointment" must have been conducted at least 7 days before the scheduled appointment (so it is not necessarily the most recent appointment)	Int	Maybe
n_dagerSidenSisteUtfort EpisodePerHenv_c	Number of days between the last conducted appointment related to the referral in question (c_kontaktAvsluttkodeID =314) and the scheduled appointment in question.	Int	Maybe
n_dagerSidenSisteUtfort EpisodePerPas_c	Number of days between the patient's last conducted appointment (c_kontaktAvsluttkodeID=314) and the scheduled appointment in question.	Int	Maybe
n_dagerTilTimeGitt_c	Number of days between the patient's last appointment and the scheduled appointment in question?	Int	Yes
n_kjoretidKomSyk	Driving time between the patient's municipality and the hospital's municipality, given in hours.	Int	Yes
n_kontaktVarighet	Planned duration of the scheduled appointment, given in minutes.	Int	Yes
n_kontaktVarighetPerDager FraTildeltTilOppmote_c	???????	Int	Yes
n_kontaktVarighetPerDager IPerioden_c	???????	Int	Yes
n_venteTid_c	Patient "wait time" for the referral of the scheduled appointment. The time from the referral was set in motion till the referral has been assessed by the specialist health service.	Int	Maybe

Appendix C

Dependent model variables

The following features are considered dependent features:

- n_antallUtforteOHEpisoderIPeriodenPerPas_c
- n_antallUtforteOHEpisoderIPeriodenPerHenv_c
- n_antallUtforteElekEpisoderIPeriodenPerPas_c
- n_antallUtforteElekEpisoderIPeriodenPerHenv_c
- n_andelUtforteOHEpisoderIPeriodenPerPas_c
- n_andelUtforteOHEpisoderIPeriodenPerHenv_c
- n_dagerFraTildeltTilOppmoteIPeriodenPerHenv_c
- n_kontaktVarighetPerDagerIPerioden_c
- n_kontaktVarighetPerDagerFraTildeltTilOppmote_c
- n_andelUtforteElekEpisoderIPeriodenPerPas_c
- n_andelUtforteElekEpisoderIPeriodenPerHenv_c
- n_dagerSidenSisteUtfortEpisodePerPas_c
- n_dagerSidenSisteUtfortEpisodeOver15DagerPerPas_c
- n_dagerSidenSisteUtfortEpisodeOver7DagerPerPas_c
- n_dagerSidenSisteUtfortEpisodePerHenv_c
- n_dagerSidenSisteUtfortEpisodeOver15DagerPerHenv_c
- n_dagerSidenSisteUtfortEpisodeOver7DagerPerHenv_c
- n_dagerFraTildeltTilOppmoteIPeriodenPerPas_c
- n_antallSykAvbestPerPas_c

Dependent model variables

- n_antallSykAvbestPerHenv_c
- n_antallPasAvbestPerPas_c
- n_antallPasAvbestPerHenv_c
- n_antallKontakterPerPas_c
- n_antallKontakterPerHenv_c
- n_antallPasAvbestPerPas3Dager_c
- n_antallPasAvbestPerPas7Dager_c
- n_antallPasAvbestPerPasOver7Dager_c
- n_antallIkkeMottPerPas_c
- n_andelSykAvbestPerPas_c
- n_andelSykAvbestPerHenv_c
- n_andelPasAvbestPerPas_c
- n_andelPasAvbestPerHenv_c
- n_andelPasAvbestPerPas3Dager_c
- n_andelPasAvbestPerPas7Dager_c
- n_andel_PasAvbestPerPasOver7Dager_c
- n_andelIkkeMottPerPas_c

Bibliography

Anguita, D., Ghelardoni, L., Ghio, A., Oneto, L., and Ridella, S. (2012). The'k'in k-fold cross validation. In *ESANN*, pages 441–446.

Boyle, T. (2019). Dealing with imbalanced data. `https://towardsdatascience.com/methods-for-dealing-with-imbalanced-data-5b761be45a18`. Accessed: 2021-03-01.

Breiman, L. (1996). Bagging predictors. *Machine learning*, 24(2):123–140.

Breiman, L. (2001). Random forests. *Machine learning*, 45(1):5–32.

Brownlee, J. (2020a). How to use standardscaler and minmaxscaler transforms in python. `https://machinelearningmastery.com/standardscaler-and-minmaxscaler-transforms-in-python/`. Accessed: 2021-06-02.

Brownlee, J. (2020b). Tour of data sampling methods for imbalanced classification. `https://machinelearningmastery.com/data-sampling-methods-for-imbalanced-classification/`. Accessed: 2021-03-01.

Brownlee, J. (2020c). Train-test split for evaluating machine learning algorithms. `https://machinelearningmastery.com/train-test-split-for-evaluating-machine-learning-algorithms/`. Accessed: 2021-04-02.

Bzdok, D., Altman, N., and Krzywinski, M. (2018). Points of significance: statistics versus machine learning.

Chawla, N. V. (2009). Data mining for imbalanced datasets: An overview. *Data mining and knowledge discovery handbook*, pages 875–886.

Chen, T. and Guestrin, C. (2016). Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*, pages 785–794.

Chilakapati, A. (2019). Concept drift. [Online; accessed 28.04.2021].

Dantas, L. F., Fleck, J. L., Oliveira, F. L. C., and Hamacher, S. (2018). No-shows in appointment scheduling–a systematic literature review. *Health Policy*, 122(4):412–421.

Domingos, P. (2000). A unified bias-variance decomposition for zero-one and squared loss. *AAAI/IAAI*, 2000:564–569.

Dong, F., Lu, J., Zhang, G., and Li, K. (2018). Active fuzzy weighting ensemble for dealing with concept drift. *International Journal of Computational Intelligence Systems*.

Drotár, P., Gazda, J., and Smékal, Z. (2015). An experimental comparison of feature selection methods on two-class biomedical datasets. *Computers in biology and medicine*, 66:1–10.

EU (2018). 2018 reform of eu gdpr - article 22.

Fan, F., Xiong, J., Li, M., and Wang, G. (2020). On interpretability of artificial neural networks: A survey. *arXiv preprint arXiv:2001.02522*.

Garreta, R. and Moncecchi, G. (2013). *Learning scikit-learn: machine learning in python*. Packt Publishing Ltd.

Geurts, P., Ernst, D., and Wehenkel, L. (2006). Extremely randomized trees. *Machine learning*, 63(1):3–42.

Ghoneim, S. (2019). Accuracy, recall, precision, f-score and specificity, which to optimize on? `https://towardsdatascience.com/accuracy-recall-precision-f-score-specificity-which-to-optimize-on-867d3f11124`. Accessed: 2021-02-17.

Goldsteen, A., Ezov, G., Shmelkin, R., Moffie, M., and Farkash, A. (2020). Anonymizing machine learning models. *arXiv preprint arXiv:2007.13086*.

Guo, H. and Viktor, H. (2004). Learning from imbalanced data sets with boosting and data generation: The databoost-im approach. *SIGKDD Explorations*, 6:30–39.

Hale, J. (2019). Scale, standardize, or normalize with scikit-learn. [Online; accessed 02.05.2021].

Han, H., Wang, W.-Y., and Mao, B.-H. (2005). Borderline-smote: a new over-sampling method in imbalanced data sets learning. In *International conference on intelligent computing*, pages 878–887. Springer.

Haury, A.-C., Gestraud, P., and Vert, J.-P. (2011). The influence of feature selection methods on accuracy, stability and interpretability of molecular signatures. *PloS one*, 6(12):e28210.

Helse Vest RHF (2017). Helse vest rhf - about us.

Hua, J., Xiong, Z., Lowey, J., Suh, E., and Dougherty, E. R. (2004). Optimal number of features as a function of sample size for various classification rules. *Bioinformatics*, 21(8):1509–1515.

Huang, Y. and Hanauer, D. A. (2014). Patient no-show predictive model development using multiple data sources for an effective overbooking approach. *Applied clinical informatics*, 5(3):836.

Huang, Y. and Li, L. (2011). Naive bayes classification algorithm based on small sample set. In *2011 IEEE International conference on cloud computing and intelligence systems*, pages 34–39. IEEE.

Jensen, K. (2012). Crisp-dm process diagram. [Online; accessed 15.03.2021].

Jung, Y. (2018). Multiple predicting k-fold cross-validation for model selection. *Journal of Nonparametric Statistics*, 30(1):197–215.

Khalilia, M., Chakraborty, S., and Popescu, M. (2011). Predicting disease risks from highly imbalanced data using random forest. *BMC medical informatics and decision making*, 11(1):1–13.

Langley, P. (1996). *Elements of machine learning*. Morgan Kaufmann.

Langley, P. et al. (1994). Selection of relevant features in machine learning. In *Proceedings of the AAAI Fall symposium on relevance*, volume 184, pages 245–271.

Lemaître, G., Nogueira, F., and Aridas, C. K. (2017). Imbalanced-learn: A python toolbox to tackle the curse of imbalanced datasets in machine learning. *The Journal of Machine Learning Research*, 18(1):559–563.

Lewis, D. D. (1998). Naive (bayes) at forty: The independence assumption in information retrieval. In *European conference on machine learning*, pages 4–15. Springer.

Liaw, A., Wiener, M., et al. (2002). Classification and regression by randomforest. *R news*, 2(3):18–22.

Mishra, S. (2017). Handling imbalanced data: Smote vs. random undersampling. *International Research Journal of Engineering and Technology (IRJET)*, 4(8).

Mitchell, A. J. and Selmes, T. (2007). Why don't patients attend their appointments? maintaining engagement with psychiatric services. *Advances in Psychiatric Treatment*, 13(6):423–434.

Molfenter, T. (2013). Reducing appointment no-shows: going from theory to practice. *Substance use & misuse*, 48(9):743–749.

Pappada, R. and Pauli, F. (2018). Discrimination in machine learning algorithms.

Prati, R. C., Batista, G. E., and Monard, M. C. (2009). Data mining with imbalanced class distributions: concepts and methods. In *IICAI*, pages 359–376.

Saeys, Y., Inza, I., and Larranaga, P. (2007). A review of feature selection techniques in bioinformatics. *bioinformatics*, 23(19):2507–2517.

Samarati, P. and Sweeney, L. (1998). Generalizing data to provide anonymity when disclosing information. In *PODS*, volume 98, pages 275487–275508. Citeseer.

Samuylova, E. (2020). Model decay and updating. [Online; accessed 28.04.2021].

Schapire, R. E., Freund, Y., Bartlett, P., Lee, W. S., et al. (1998). Boosting the margin: A new explanation for the effectiveness of voting methods. *Annals of statistics*, 26(5):1651–1686.

Scikit-learn (2021). Permutation importance vs random forest feature importance (mdi). `https://scikit-learn.org/stable/auto_examples/inspection/plot_permutation_importance.html#sphx-glr-auto-examples-inspection-plot-permutation-importance-py`. Accessed: 2021-05-15.

Shih, Y.-S. (1999). Families of splitting criteria for classification trees. *Statistics and Computing*, 9(4):309–315.

Soria-Comas, J. and Domingo-Ferrer, J. (2016). Big data privacy: challenges to privacy principles and models. *Data Science and Engineering*, 1(1):21–28.

Thai-Nghe, N., Nghi, D., and Schmidt-Thieme, L. (2010). Learning optimal threshold on resampling data to deal with class imbalance. In *Proc. IEEE RIVF International Conference on Computing and Telecommunication Technologies*, pages 71–76.

Webb, G. I. (2010). Naïve bayes. *Encyclopedia of machine learning*, 15:713–714.

Weiss, G. and Provost, F. (2003). Learning when training data are costly: The effect of class distribution on tree induction. *J. Artif. Intell. Res. (JAIR)*, 19:315–354.

Wimmer, H. and Powell, L. (2014). A comparison of the effects of k-anonymity on machine learning algorithms. In *Proceedings of the Conference for Information Systems Applied Research ISSN*, volume 2167, page 1508.

Wirth, R. and Hipp, J. (2000). Crisp-dm: Towards a standard process model for data mining. In *Proceedings of the 4th international conference on the practical applications of knowledge discovery and data mining*, volume 1. Springer-Verlag London, UK.

Zhao, K. (2019). Pre-process data with pipeline to prevent data leakage during cross-validation. `https://towardsdatascience.com/pre-process-data-with-pipeline-to-prevent-data-leakage-during-cross-validation-e3442cca7fdc`. Accessed: 2021-05-02.

Zhao, Z., Morstatter, F., Sharma, S., Alelyani, S., Anand, A., and Liu, H. (2010). Advancing feature selection research. *ASU feature selection repository*, pages 1–28.

Žliobaitė, I., Pechenizkiy, M., and Gama, J. (2016). An overview of concept drift applications. *Big data analysis: new algorithms for a new society*, pages 91–114.