UNIVERSITY OF GOTHENBURG
SCHOOL OF BUSINESS, ECONOMICS AND LAW

Predicting hotel cancellations using machine learning

Enok Gartvall & Oscar Skånhagen

September 2021

Bachelor Thesis in Statistics
15HP

Thesis advisor:

Alexander Herbertsson

School of Business, Economics and Law,
Department of Economics

Contact:

gusgarten@student.gu.se & gusskanos@student.gu.se

Abstract

Room cancellations is a big challenge for the hotel industry since the number of guest affects the whole operational setup. The purpose of the thesis is to predict hotel cancellations using machine learning and analyse which factors have the most influence. Broadly speaking, machine learning can be summarized as an interdisciplinary science for using computers to solve a given problem by finding patterns and learning from existing data. Machine learning involves theory from among others probability, statistics, optimization, algorithms and computer science. The problem of predicting cancellations is a binary classification problem, as the two possible outcomes are cancellation or non-cancellation. Classification in statistics is the process of determining what class a given input data belongs to, in other words predicting a qualitative outcome variable. Data was provided by a hotel in the Gothenburg area and the machine learning algorithms used in the thesis were Random Forest, XGBoost and Logit. Random Forest and XGBoost are tree-based models, which creates decision trees in order to make predictions and in a classification problem these are referred to as classification trees. The aim for a classification tree is to determine a qualitative outcome variable by making step-wise binary splits, where the different outcomes are denoted as classes. The logit model, or logistic regression, is a form of binary regression which is used as a reference model in this thesis. Our main findings indicate that Random Forest is the best performing model onthe hotel data with an accuracy close to 80%. Leadtime, which is a numeric variable that represent the days between when the hotel reservation was made and day of arrival, was the most influential variable in the Random Forest model. Adding weather data marginally improved the accuracy of predicting hotel cancellations, for all models.

Acknowledgements

We would like to express our gratitude towards our supervisor Alexander Herbertsson, for his dedicated guidance and feedback throughout this process.

Contents

1 Introduction

In this section, the topic of the thesis is introduced. The background, purpose, research questions and the outline for this thesis will also be presented.

1.1 Background and problem discussion

Cancellations are one of the greatest challenges facing the hotel industry around the world, especially if they occur close to the date of stay. Cancelled hotel reservations that cannot be replaced with another guest mean not only financial losses for the hotel but also contribute to other problems. Since the number of guests is what determines the operational setup for a hotel, an inaccurate forecast can lead to over- and understaffing of personal, insufficient supplies etc.

Predicting customer cancellation behavior dates back to 1966, when the aviation industry was the first to use it structurally with the intention of increasing revenues by mapping out customer behavior (Chiang et al. 2007). Various industries have adopted this strategy throughout the years, such as car rentals, casinos, and the hotel industry (Kimes & Wirtz 2003). The problem of predicting hotel cancellations by using the behaviour of customers have been around since 1980 (Uysal & O'Leary 1986).

Although costumer cancellations are a significant problem for the hotel industry and the issue have been discussed for the last 40 years, there is still relatively little research on the subject, in particular for the studies which apply machine learning. Antonio, de Almeida & Nunes (2019) is describing it as "*poorly represented in the scientific literature*".

Antonio, de Almeida & Nunes (2017) further discuss the differences in results between studies and they argue that the results can be assumed to vary depending on hotel-type, e.g. resort, business etc. As most of the studies are conducted in different countries the conclusions cannot be generalized for the whole industry but are specific to the hotel in question. Furthermore, the models created for a specific hotel are expected to have low predictive power when used in another context.

1.2 Purpose and research questions for this thesis

The purpose of this thesis is to predict cancellations for a specific hotel in Gothenburg with the use of machine learning. Furthermore, it aims to quantify the underlying factors of the cancellations. More specific, we will address the following research questions:

Research questions

- *Q1 : Can cancellations for hotel guests be predicted with the use of machine-learning, based on the given data?*
- *Q2 : What factors are most influential when predicting cancellations?*
- *Q3 : Can external data in the form of weather improve predictions for cancellations?*

1.3 Scope and boundaries of thesis

This thesis will analyze whether hotel cancellations can be predicted with the help of machine learning and which variables that have the greatest impact. The thesis will not take into account any time aspects, which may be interesting for a hotel, for example predicting bookings two weeks in advance. The collected data and the applied statistical models are tested for one hotel, which means that the conclusions of the thesis can not be generalized.

The observations of the study exclude data affected by COVID-19, as they are not considered to be representative for future behaviour of hotel customers.

1.4 Outline

The rest of this thesis is organized as follows. In the second section the theoretical background is presented. Section three and four describe the previous research and methodology. Section five presents the data. The empirical results of the study and the analysis are presented in section six and seven.

2 Theoretical background

This section aims to briefly summarize some of the general statistical concepts with the purpose of introducing the reader to the statistical framework which this study relies on.

2.1 Classification

Classification in statistics is the process of determining what class a given input data belongs to, in other words predicting a qualitative outcome variable. The qualitative outcome variable can be binary or multiclass (James et al. 2013). A typical binary classification problem is for example determining if an individual will be able to pay back their loan i.e., default or not. An example of a multiclass classification problem is to predict the outcome of a football match, where the outcome can be a win, a loss or a draw. In this thesis we are working with a binary classification problem because a hotel

reservation is either cancelled or not cancelled. If a model predicts the correct class of an observation the predicted class will be equal to the true class. If the predicted class does not correspond to the true class a false classification have been made. This can be expressed as following:

$$I(y_i = \hat{y}_i) \text{ or } I(y_i \neq \hat{y}_i) \tag{1}$$

where y_i is the correct class for the ith observation and $\hat{y}_i$ is the predicted class for the ith observation. Furthermore, $I(\mathrm{A})$ is an indicator function that will be one if event A happens and zero otherwise. The proportion of misclassifications is called the error rate and is calculated by:

$$\frac{1}{N}\sum_{n=1}^{N} I(y_i \neq \hat{y}_i) \tag{2}$$

where N represents the sample size. A prediction model that is trying to solve a binary classification problem can make two types of misclassifications: a false positive and a false negative (Fawcett 2006). The following example will be used to explain the meaning of false positive and false negative. Assume a model that is trying to predict if a person will have a cardiac disease. If the model predicts that an individual is going to suffer from cardiac disease but in reality the person will not, the model predicted a false positive. On the contrary, if the model predicted that the individual is healthy i.e not suffer from cardiac disease, but in reality will suffer from cardiac disease, the model returned a false negative. Both situations are examples of misclassifications but highlight the difference between the errors. False positives and false negatives will almost always occur in a predicting model but the importance will differ given the context. In the cardiac example above, it is more severe to classify a sick person as healthy rather then a healthy person as sick i.e, it is worse to make a false negative. A false positive can also be referred to as a type 1 error and a false negative is refereed to as a type 2 error.

Evaluating how well a model performs by analysing classification rate, false positives and false negatives is only interesting if the model is tested on data it has not seen before. This will be explained in more detail in Subsection 2.3.

The false positive ratio is calculated by taking the number of false positives and divide it with the number of false positives plus the number of true negatives. Referring to the example above it would be dividing the individuals who are falsely classified as sick with the number of people that are healthy. The false negative ratio is calculated by taking the number of false negatives and divide it with the number of false negatives plus the number of true positives. Referring to the example above it would be dividing the individuals

who are falsely classified as healthy with all the people that suffer from cardiac disease (Fawcett 2006). Let FP be the number of observation falsely classified as positive, FN be the number of observation falsely classified negative, TN be the number of observation correctly classified as negative and TP be the number of observation correctly classified as positive. The false positive rate and the false negative rate are then given by

$$\text{False postive rate} = \frac{FP}{(FP+TN)} \qquad \text{False negative rate} = \frac{FN}{(FN+TP)}. \tag{3}$$

To calculate the total accuracy of the binary classification model one simply divides the correct number of classifications with all classifications, that is

$$\text{Accuracy} = \frac{TP+TN}{(FN+FP+TP+TN)}. \tag{4}$$

A common way to summarize the performance of a model that is trying to solve a binary classification problem is by using a confusion matrix that is displayed in *Figure 1*. The diagonal from top left to lower right in *Figure 1* displays the correct values, which means that the predictive values correspond to the true values. The diagonal from lower left to top right shows the false- positives and negatives (Fawcett 2006).

		Predicted class	
		Positive	**Negative**
True class	**Positive**	TP	FN
	Negative	FP	TN

Figure 1: Illustration of Confusion Matrix

2.2 ROC-curve & AUC

Receiver operating characteristic also known as the ROC-curve is a widely used tool to visualize the possible trade-off between type-1 and type-2 errors, i.e false positive and false negative, for a classification model. This metric shows the change in the types of error with respect to the threshold T of the classifier (James et al. 2013). The model used for a binary classification problem will output a score or probability for a given observation to belong to class 1, which we call P, and an observation is classified as 1 in the case of $P > T$, otherwise 0.

The threshold, T, is thus the minimum value of P required to be assigned to class 1. The most intuitive threshold for models which output a probability, like logistic regression, is therefore 0.5, so that each observation is assigned to the most likely class. However, for example lowering the threshold to 0.4 will result in that more observations will be assigned to class 1, which then increases the total amount of correct prediction for class 1, but will in turn decrease the amount of correct predictions for class 0. Changing the threshold of the classifier therefore corresponds to changing the criteria for when the model assigns an observation to either class. An example of a ROC-curve is displayed in *Figure 2*, taken from Wikipedia and created by cmglee (2018), where the blue, green and orange lines represents hypothetical ROC-curves.

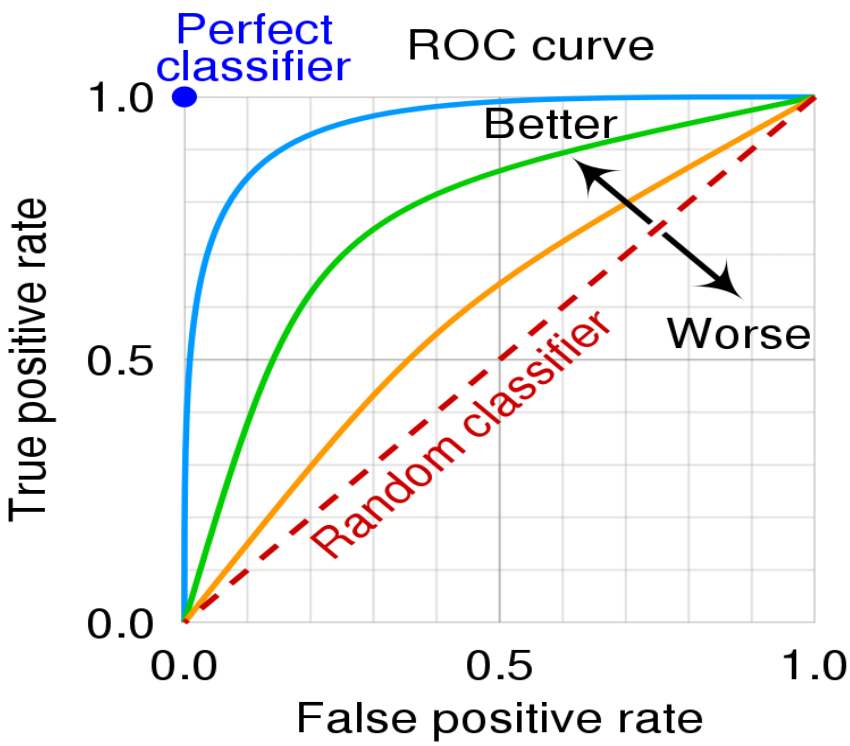

Figure 2: Illustration over ROC Curve by cmglee (2018)

Along the curves in *Figure 2* we find all the possible values of the thresholds between 0

and 1 and as these possible values are continuous, so is the curve. The threshold decrease as the curve goes from the lower left corner to the upper right. In the lower left corner we have a threshold of 1, meaning that we assign all observations to class 0. Therefore we have 0% true positives and 0% false positives as no observations are predicted as "positive". For the opposite alternative, in the upper right corner, the threshold is 0, meaning that all observations are classified as 1 or "positive". This results in that we will have 100 % true positive rate, but also that we will have 100 % false positive rate. Moving between these two extremes will thus be a way of balancing between increasing the true positive rate, while keeping the false positive rate down. This trade-off is what is visualized with the ROC-curve (James et al. 2013).

For a given level of threshold the ROC-curve illustrates the corresponding classification-rate for each class. When making predictions, the target is to find the optimal threshold that results in a desirable relationship between type-1 and type-2 errors. For a broader discussion about this trade-off between errors see the *cardiac* example described in Subsection 2.1. However, as this optimal threshold can be different depending on the model and data, a general measure to compare models is *area under the curve*, or AUC (James et al. 2013). As an optimal ROC-curve tangents the y-axis and the upper line in *Figure 2*, visualized by the blue dot, a AUC of 1 is therefore optimal. For comparison, we expect a classifier that has no predictive power, e.g. random assignments, to have an AUC of 0.5 and in *Figure 2* this corresponds to the red dashed diagonal line.

2.3 Cross-Validation

When making predictions, there is a important difference between the number of misclassifications for the training data compared to the test data. The misclassification rate when evaluated on the training data is referred to as "training error" and when evaluated on the test data it is referred to as "test error". Training a model on a specific data-set can result in accurate predictions within the data-set, but might not show the same predictive power when applied to new data. This creates problems as the intention of creating a model is often to apply it to new or not yet existing data. A way of solving this problem is to artificially create test data by dividing the available data into subsets and exclude these parts when training the model. This approach is referred to as cross-validation. There are multiple different techniques of sub-setting, where two of the most used are "Validation set approach" and "K-fold Cross-Validation" (James et al. 2013).

Validation set approach

The validation set approach is the easiest and most straightforward way of approaching the problem. It involves randomly dividing the available data into two sets, a training

set and a test or "validation" set. The model is only allowed to use the training data to create a model. The model is then evaluated on the test data, in order to simulate the case of applying the model to a new "case" (James et al. 2013).

K-fold Cross-Validation

K-fold Cross-Validation divides the data into k equally sized subsets, also referred to as "folds". Each of the k folds are then used as the validation set, while the model trains on the observations of the remaining $k-1$ folds. This results in a total of k estimations that are evaluated. K-fold Cross-Validation is considered to be a more robust technique compared to the validation set approach, as it decreases the influence of the random split by creating more subsets and realizing more evaluations (James et al. 2013).

2.4 Machine Learning

The terminology machine learning was coined in the year 1959 by Arthur Samuel who was at the time working for IBM. He programmed a computer to play checkers and demonstrated that in the end the computer was able to play the game better in comparison to the human who wrote the code (Samuel 1959). Since then the field of machine learning have grown exponentially. Fradkov (2020) argues that the *gold rush* in ML started at the beginning of the 21st century and was driven by three factors that amplified each other. The first one was *big data*, more data was stored then ever before and ML became a necessity to make sense of all the information. The second one was the technological development. Computers were able to better process big amount of data which both reduced the cost and time. The last factor was the breakthroughs made in the scientific field of ML and the attention put on the research area.

Today machine learning algorithms are used in many different industries. The finance industry use it when making credit decision and detecting fraud, big social media companies like Youtube and Facebook use recommendation algorithms for advertisement, the healthcare industry use ML to diagnose patients with image analysis. These are just a few example of how ML is used today in different industries (Brown 2021).

There is no universal definition of machine learning. Stanford University defines it as "Machine learning is the science of getting computers to act without being explicitly programmed." (Ng 2021). Ethem Alpaydin, the author of the book *Introduction to Machine Learning* defines it as "Programming computers to optimize a performance criterion using example data or past experience" (Alpaydin 2020). There are more definitions of ML but it can be summarized as a computer trying to solve a given problem by finding patterns and learning from existing data.

As machine learning is based on statistics, it is difficult to draw the exact difference between them. Their purpose can be discerned, ML primarily strives to make accurate predictions while statistics try to find inference within population (Stewart 2019).

3 Previous empirical studies

This section aims to briefly summarize a part of the recent research that have been conducted within the field of predicting hotel cancellations.

3.1 Andriawan et al. (2020)

Andriawan et al. (2020) studied how machine learning can be used to predict which individuals will cancel their hotel booking in order to optimize the number of bookings made by the hotel. The study uses a method of process called CRISP-DM to answer their questions. CRISP-DM is a method that focus on trying to understand the data in depth, in order to take correct decisions further down the process and is widely used in the field of data science and data mining (Piatetsky-Shapiro 2014). The data used in the CRISP-DM model is from two different sources: resort hotel in Algarve, Portugal and a central hotel in Lisbon, Portugal. Both data-sets, however, have the same structure with 31 different variables, where each observation is a hotel reservation.

The study used 4 different tree-based models that were cross-validated with 10 folds and the best performing one was Random Forest, with the result of 87% accuracy. Random Forest will be explained more in detail in Subsection 4.2. The variable that had the biggest influence for the Random Forest algorithm was lead-time, which is defined as the number of days between day of booking and arrival date. The conclusion of the article is that machine learning can be used as an important tool to minimize loss of income for hotels.

3.2 Antonio, de Almeida & Nunes (2019)

Antonio, de Almeida & Nunes (2019) performed a study in collaboration with two hotels in Portugal in order to create an automatized system which predicts cancellations using machine learning and tree-based models. The resulting automatized model was trained daily on all active reservations at the hotels. This allows the model to adapt itself and its predictions continuously in time, which is significant when it comes to hotel reservations.

The system builds a new model every day, and tunes the hyper-parameters. The result is then compared with the last seven days, and changes the settings if the model performs better, otherwise the settings are left unchanged.

The results from all models are saved in a database and are evaluated, where correct classifications are rewarded and incorrect classifications are punished. Type 1 errors have a higher cost i.e larger punishment than other types of misclassifications. All this information is then used for the construction of new models.

The automatized model is based on the so called XGBoost algorithm, which will be discussed more in depth in Subsection 4.3, and resulted in over 84% correct classifications. The validation was done through the validation set approach. The system made it possible to identify potential cancellations, and contact these individuals in order to prevent a cancellation by e.g. offering free breakfast. The decrease in cancellations caused by the algorithm resulted in 39 million euro savings for the hotels per year.

3.3 Sánchez-Medina et al. (2020)

Sánchez-Medina et al. (2020) focused in their research on using machine learning and a reduced number of variables, to predict the probability for cancellations within the hotel industry. The purpose of using a low number of explanatory variables was to enable comparisons with other models and the use of the model with different data. As a result, the variables were also limited to those most hotels can be assumed to receive at a booking.

The data used in this study is from a hotel in Spain, Gran Canaria, and uses a number of different models, where the best one results in an accuracy of 98%. The remaining models does not exhibit the same predictive power, and have accuracies of around 80 %. The validation was done through the validation set approach.

The results are considered to be good in relation to previous literature in the field. Especially considering the fact that the study only uses 13 explanatory variables, which is well below the average for similar studies.

3.4 Antonio, de Almeida & Nunes (2017)

Antonio, de Almeida & Nunes (2017), have used data from four different hotels in Algarve, Portugal in order to predict cancellations. The purpose of the research was to demonstrate the economic value of being able to accurately predict cancellations.

The study uses a few different tree-based models, but also other types of algorithms. The results are based on the mean over the four hotels and they show that the tree-based models perform the best in terms of accuracy, showing an accuracy of approximately 95%. The validation was done through the validation set approach. From the results of the study, it can also be seen that the model performance varies between the hotels. There in no model that clearly generates the best results across the board. The same

pattern could be seen for which variables that were the most influential. Land of origin was for example the variable with the largest influence for two of the hotels, while it was ranked further down for the other two hotels.

3.5 Validity of empirical studies

It is worth mentioning that some of the chosen previous empirical studies have an approach from the view of the tourism industry rather than from the statistical standpoint. This means that the studies are more focused on applying already existing statistical models rather then developing new ones, thus not contributing to the development in the field of statistics. However, for this thesis, they are still chosen on the basis of the results and the level of statistical methods applied. This thesis makes an effort in applying these state of the art models with focus on the statistical method rather than the application within the tourism industry.

4 Methodology

Three different models will be used in this thesis to predict hotel cancellations and these are Logit, Random Forest and XGBoost. The previous empirical studies discussed in Subsection 3.1 - 3.4 have shown that tree-based models perform very well in predicting cancellations. Therefore the tree-based model chosen for this thesis are Random Forest and XGBoost. The Logit model, which is a simpler model should be seen as benchmark to verify the hypothesis that tree-based models perform better.

4.1 Logistic Regression

Logistic regression is based on the idea that, for a binary outcome, assign each observation a conditional probability as to whether the observation belongs to a specific category, meaning the probability depends on the value of the observation for the explanatory variables. Since the outcome of logistic regression is a probability, the defined set of values are $[0, 1]$. As a result, the form is adapted to meet the criterion of only being defined for the set of values $[0, 1]$ (James et al. 2013).
Let Y be a random variable typically representing the dependent variable, let $p(X) = P[Y = 1|X]$ and let $X = (X_1, X_2, \cdots, X_i)$ be the independent variables. Then the main idea in logistic regression is to model $p(X)$ as;

$$p(X) = \frac{e^{\beta_0+\beta_1 X_1 \cdots + \beta_i X_i}}{1 + e^{\beta_0+\beta_1 X_1 \cdots + \beta_i X_i}}. \tag{5}$$

Rewriting model (5) as follows can help make the results of logistic regression easier to interpret;

$$\log\left(\frac{p(X)}{1-p(X)}\right) = \beta_0 + \beta_1 X_1 \cdots + \beta_i X_i. \tag{6}$$

The left-hand side of *Equation* (6) is usually referred to as the "log-odds", and visualizes how a unit increase of X_i results in changing the "log-odds" by β_i. Where the "log odds" is the logarithmic ratio between the probability that the observation belongs to the category defined as 1, divided by the probability that the observation belongs to the category defined as 0. The estimation of the parameters, $(\beta_0, \beta_1, \cdots, \beta_i)$, is generally done using the maximum likelihood method (James et al. 2013). Maximum likelihood is based on the intuitive idea of estimating the parameters $(\beta_0, \beta_1, \cdots, \beta_i)$ so that the resulting $p(X)$ is, depending on the correct response value for the observation, as close as possible to 1 and 0 respectively. The estimates based on the maximum likelihood function can be expressed by the equation for the likelihood function;

$$\mathcal{L}(\beta_0, \beta_1, \cdots \beta_i) = \prod_{i:y_i=1} p(x_i) \prod_{i':y_i'=0} (1 - p(x_{i'})) \tag{7}$$

where $y_i = 1$ notates the observations belonging to class 1, $y_i' = 0$ notates the observations belonging to class 0 and x_i, $x_{i'}$ is the vector of inputs for the observations for respective class. The estimates for the parameters, $(\beta_0, \beta_1, \cdots \beta_i)$, are thus selected based on what maximizes the likelihood function, in *Equation* (7).

4.2 Random Forest

Random Forest is a tree-based algorithm that can be applied on both regression and classification problems. The algorithm was first created by Tin Kam Ho in 1995 (Ho 1995) and was further developed by Leo Breiman (Breiman 2001). To understand the Random Forest algorithm, one must first understand the concepts of classification trees, Bootstrap and Bagging, which will be presented below.

4.2.1 Classification trees

Tree-based methods can both be used for regression and classifications problem. A major advantage of tree-based methods is that the algorithm behind it is very intuitive, which makes the tree that is being built easy to understand and interpret (James et al. 2013).

The aim for a classification tree is to determine a qualitative outcome variable by making step-wise binary splits, where the different outcomes are denoted as classes. The algorithm always starts at the root node performing the first split on the entire dataset. The following splits in the tree down to the final binary split are called decision nodes. The

”branches” in the end of the classification tree are called leaf nodes. Since there is just a subset of data left after performing a split, the splits depend on each other. In the leaf nodes the classification tree predicts the observations to the most commonly occurring class (James et al. 2013). The tree performs the splits based on a specific splitting criterion. The most common split criteria are classification error rate and Gini index, these criteria will be explained more in detail later in this section. In each binary split the tree chooses one of the explanatory variables that contains the most information about the independent variable based on the split criteria. This process is repeated until that an additional split in the classification tree does not contribute to an increase in predictive power or if there is a predefined limit to how deep the tree can grow.

Figure 3 is an illustration of a classification tree with the aim to determine if an observation is a worker or student. The tree has two explanatory variables at its disposal, which are *income* and *age*.

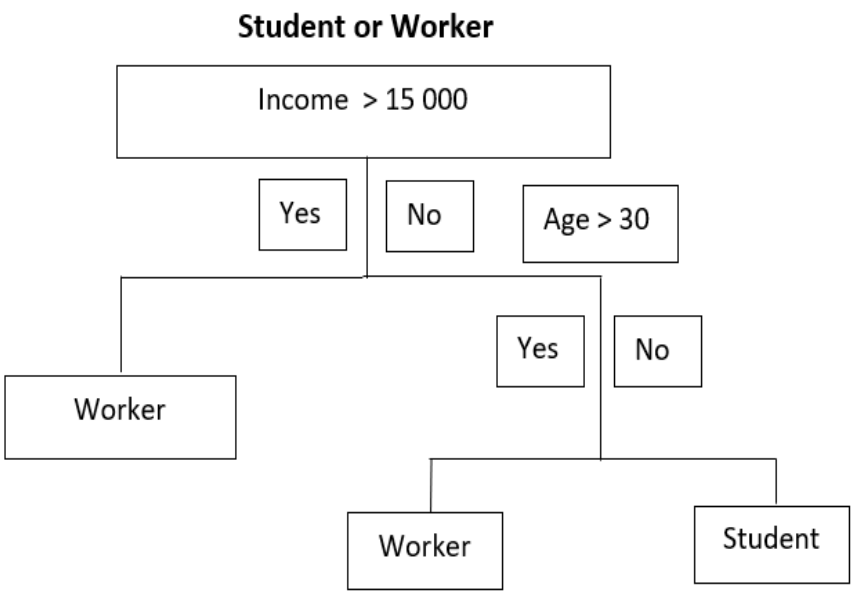

Figure 3: Example of a decision tree.

In the root node the tree uses income to perform its first binary split, if the individual earns more then 15000 the observation is classified as a worker. The second binary split is performed in the decision node, the tree uses the variable age to determine if the observation is a worker or a student. If the individual is older than 30, the observation is classified as a worker otherwise it is classified as a student. Since the tree is not deep i.e. does not

perform many splits, misclassifications will occur in the leaf nodes. Some students work extra and earn more than 15000 a month and will be wrongly classified as a worker. At the same time the are workers under the age of 30 that earn less than 15000 that will be misclassified as a student. However the two binary splits, in the specific order, with the specific values, given the split criteria, were made to minimize the misclassifications in the leaf nodes (given that the tree was only allowed to make two splits).

There are different criteria for the divisions in the internal nodes, for example classification error rate and Gini index. The simplest criteria is the classification error rate, which means that each split in the classification tree is made to maximize the proportion of the majority class k in the leaf nodes. The majority class refers to the class that is most commonly accruing in the leaf nodes. Let $\hat{p}_{nk}$ represent the proportion of observations in node n that belongs to class k. The classification error rate is then the number of observations that do not belong to the majority class divided by all observations in the leaf nodes, given by

$$\text{Classification error rate} = 1 - max(\hat{p}_{nk}). \tag{8}$$

Referring back to *Figure 3*, n would be one of the three leaf nodes where the classes K would be *Worker* (say $k = 1$) or *Student* (say $k = 0$) and $max(\hat{p}_{nk})$ would be the proportion of observations belonging to the majority class. However, it turns out that the classification error rate is not preferable as it is not sufficiently sensitive when building tree-models (Hastie et al.,2013).

In comparison to the *classification error rate* Gini index strives to maximize the homogeneity in the leaf nodes. Homogeneity refers to that a large proportion of the observations in a node belong to one of the classes, this is also known as node purity. The Gini criteria tries to minimize the variance between the K classes in each leaf node, in other words maximize that one class in each of the nodes becomes as dominant as possible (Muchai & Odongo 2014). So, the Gini index is the total variance over the classes K_j and is given by

$$Gini = \sum_{k=1}^{K} \hat{p}_{nk}(1 - \hat{p}_{nk}) \tag{9}$$

$w$here $\hat{p}_{nk}$ is defined as above. Note that Equation (9) shows that if $\hat{p}_{nk}$ is close to zero or close to one, the Gini index will be very small. This means that the binary split in the node resulted in a strong majority being assigned to one of the classes. In other words, the node has a high purity (James et al. 2013).

Figure 4 will be used to demonstrate the calculation of Gini index in the nodes. Referring back to *Figure 3*, when the classification tree made its second split the individuals that were over 30 years old were assigned to the class *Worker*. Assume that in the left leaf node there were in total 14 observations, where two observations was of the class *Students* i.e two misclassifications. Also assume that in the right leaf node there were in total 22 observations, where two observations was of the class *Worker* i.e two misclassifications. The Gini index for the left leaf node and the right leaf node are following:

$$\text{Gini index (Left node)} = \underbrace{(\frac{12}{14}(1-\frac{12}{14}))}_{k=1} + \underbrace{(\frac{2}{14}(1-\frac{2}{14}))}_{k=0} = 0.245$$

$$\text{Gini index (Right node)} = \underbrace{(\frac{2}{22}(1-\frac{2}{22}))}_{k=1} + \underbrace{(\frac{20}{22}(1-\frac{20}{22}))}_{k=0} = 0.165$$

The weighted Gini index after the split is following:

$$\text{Gini index (Weighted)} = 0.245 * \frac{14}{36} + 0.165 * \frac{22}{36} = 0.1825$$

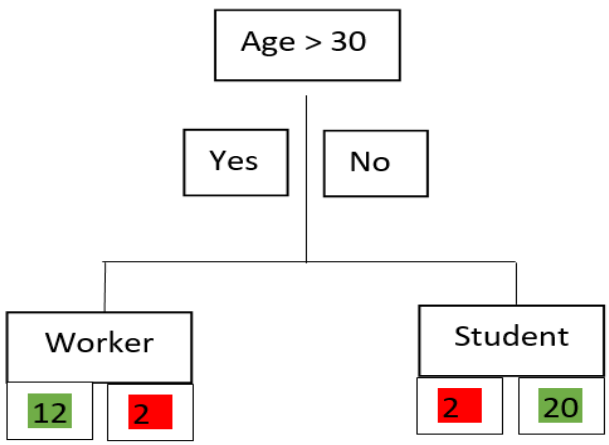

Figure 4: Example of a decision tree split.

For a fixed split point, the classification tree is iterating through all the feasible explanatory variables and all possible thresholds to find the variable and threshold that yields the smallest weighted Gini index. In the example in *Figure 4*, it means that explanatory variable *Age* with threshold of 30 yielded the lowest weighted Gini index and was therefore optimal for the second split in the tree.

4.2.2 Bootstrap och Bagging

The idea behind Bootstrap is to simulate new dataset by sampling from an original dataset. The Bootstrap method can be used to estimate errors, calculate confidence intervals, estimate bias etc. (Efron & Tibshirani 1994). There are many types of Bootstrap methods, for example bayesian Bootstrap, parametric Bootstrap and nonparametric Bootstrap. This thesis is going to focus on the nonparametric Bootstrap, which works as follows:

Assume the following original dataset with the explanatory variable $X = x_1, x_2..., x_n$ and the independent variable $Y = y_1, y_2..., y_n$. Sample at random from the original dataset $[Y, X]$, with replacement n times. Call these X_z, Y_z, where $z = 1, 2, 3.., Z$. Repeat until you have Z simulated dataset. Train and fit a classification tree $\hat{f}_z(x)$ for every dataset z simulated by the nonparametric Bootstrap method. Calculate the mean of the aggregate value for each observation x for all the classification trees $\hat{f}_1(x), \hat{f}_2(x)...\hat{f}_Z(x)$, where $\hat{f}(x_i)$ is the average value of the ith observation x for all the fitted classification trees and Z represent the number of trees, that is

$$\hat{f}(x_i) = \frac{1}{Z}\sum_{i=1}^{Z} \hat{f}_z(x_i). \tag{10}$$

In classification problems, the Bagging method uses majority voting to determine which class an observation belongs to (James et al. 2013). Majority voting works as follows: if we have a binary outcome variable $Y \in$ [0,1] the same observation x can be assigned to different classes in the different trees due to the fact that the trees are created from different datasets. The observation x will then be assign to the class it was assigned the most time to, this can be expressed as: $Y = 1$ if $\hat{f}(x) > 0.5$ and $Y = 0$ otherwise.

Referring to *Figure 3*, let the class *Worker* = 1 and *Student* = 0. Assume that three different classifications trees $\hat{f}_1(x)$,$\hat{f}_2(x)$,$\hat{f}_3(x)$ were created from the nonparametric Bootstrap method. The same observation x_1 is assigned to the class *Worker* in the trees $\hat{f}_1$ and $\hat{f}_2$ but in $\hat{f}_3$ the observation is assigned to the class *Student*. Then $\hat{f}(x_i)$ in *Equation* (10) will be equal to:

$$\hat{f}(x_1) = \frac{1}{3}(1 + 1 + 0) = \frac{2}{3}. \tag{11}$$

Because $\hat{f}(x_1) > 0.5$, the observation x_1 was assigned to the class *Worker* most times, the majority vote determine that the observation x_1 is a *Worker*. Hence, the core meaning of Bagging is to evaluate the aggregated result created by the Bootstrap method (James et al. 2013).

To assign an observation to a class by majority voting in a binary classification problem is equivalent to having a threshold of $T = 0.5$, further discussion about thresholds can be find in Subsection 2.2. Referring back to *Equation* (11), the threshold must be greater then $\frac{2}{3}$ for the observation to be assigned to the class *Student*.

There is an easy way to evaluate how well the Bagging method performs, this can be done by using so called *out of bag* observation. When one is simulating new datasets with the Bootstrap method, around one third of the observations will not be part of the new simulated data. If the *out of bag* observations are used to evaluate the Bagging method the result becomes credible because the models are evaluated on data on which they have not been trained on, which means that *out of bag* observations are equivalent to test data (James et al. 2013). This is a useful feature to the Bagging method and is similar to the idea of the cross validation technique.

Figure 5 shows an example of the nonparametric Bootstrap method. The original data has four observations of variable X and Y. To generate new datasets the method randomly samples four observations from the original data. Since the Bootstrap is randomly sampling with replacement, the same observation can occur more then one time in the simulated datasets.

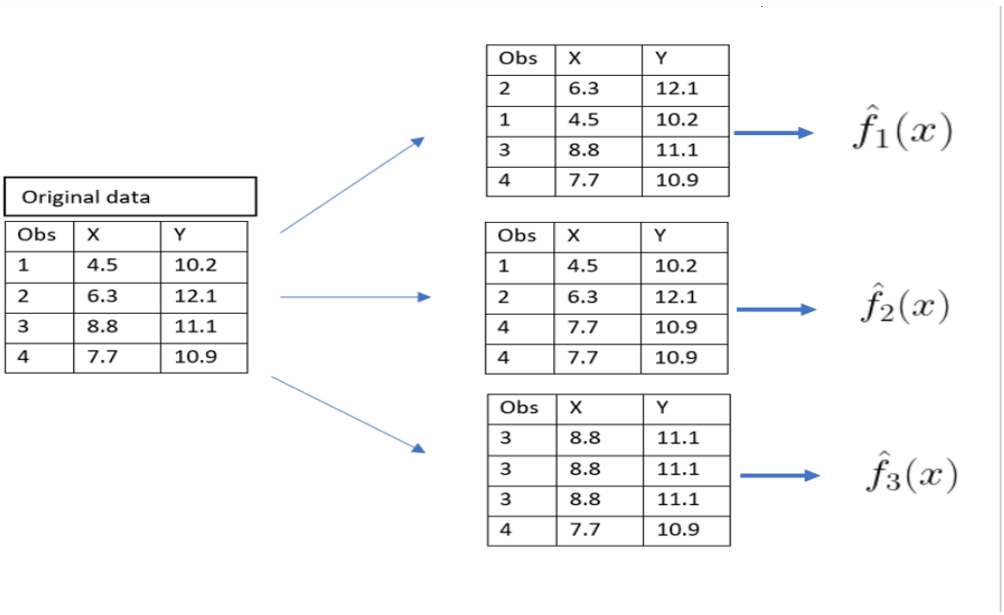

Figure 5: Example of a Bootstrap process with three samples. When used on classification trees, each Bootstrap sample will produce a classification estimator $\hat{f}_i(x)$ in the Bagging method.

4.2.3 Random Forest algorithm

The difference between the Bagging method and the Random Forest algorithm is that in Random Forest, the classification trees can only use a predetermined number m of randomly selected explanatory variable when performing the splits (James et al. 2013). Hastie et al. (2009) presents the Random Forest algorithm as follows:

---

1. For $b = 1, 2, 3, \ldots, B$, repeat the following steps,

   - Generate new datasets from the original data by nonparametric Bootstrap method.
   - For every new dataset, fit a classification-tree T_b.
   - At each binary split in the tree, a predetermined number m of explanatory variables are randomly selected for the tree to choose from. The algorithm picks the explanatory variable and the split point that maximize the homogeneity in the nodes by the Gini criteria (Assuming that the split criteria used is Gini), where $\hat{p}_{nk}$ represent the proportion of observations in node n that belongs to class k.

$$Gini = \sum_{k=1}^{K} \hat{p}_{nk}(1 - \hat{p}_{nk})$$

3. Collect all the fitted trees $\{T_b\}_{b=1}^{B}$

4. For a given input x

   - Let $\hat{C}_b(x)$ be the predicted class of the bth tree for observation x.
   - The Random Forest algorithm uses majority voting to determine which class an observation assigns to. The predicted class for input x is then:

$$\hat{C}(x) = \text{majority vote}\{\hat{C}_b(x)\}_{b=1}^{B}$$

There are several parameters in the Random Forest algorithm that can be calibrated to increase the performance (Géron 2019), three of them are going to be presented here. The first one is how many trees B that will be created, the second one is how many features m the algorithm can use in every split and the last one is how deep the trees are allowed to grow. By tuning these parameters through k-fold cross validation, a better accuracy can be achieved.

The random component m in the Random Forest algorithm makes it superior to the Bagging method. This is because the trees become uncorrelated. This may sound insignificant but makes a big difference. Assuming that the data set used to fit the classification trees

have a very strong explanatory variable and all the features could be used in every split. Feature is a synonym to explanatory variable, both terms will be used throughout the rest of this thesis. Although the trees are created from different datasets, they will all start to split in the rote node based on the dominant feature and in the end be very similar to each other. Random Forest solves this issue by only giving the trees a random subset of all feature in every split. This results in some trees not being able to perform their split at the root node based on the dominant feature and instead forced to take another explanatory variable (James et al. 2013).

4.2.4 Feature importance - Random Forest

Feature importance gives information about how important each explanatory variable is to determine which class an observation belongs to in the Random Forest algorithm. The calculation depends on which split criterion the algorithm has used. If the Random Forest has used the Gini criteria then the feature importance for each variable is calculated by the decrease in Gini index *Equation* (9), in each split performed with the variable X_i, averaged across all B trees. Feature importance is then ordering the variables that minimize the variance in the leaf nodes (James et al. 2013). A high number indicates a high importance and a low number indicates a low importance. Feature importance do not indicate if a variable has a negative or positive impact on the outcome variable, only the ranking of the variable according to the above ordering method. The feature importance can through normalisation sum up to one, which is presented in *Figure 6*.

To illustrate the meaning of feature importance the following example will be used. Assume a Random Forest algorithm that tries to solve a binary classification problem if a student will pass an exam, where pass equals one and not pass equals zero. Also assume that the only features available are timed studied, number of lectures and coffee consumed. The feature importance for this example is displayed in *Figure 6* where the variable *time studied* contains the most information in determining if a student pass an exam or not. *Number of lectures* is the second most important feature and *Coffee consumed* contains no or very little information in determining if a student pass an exam or not. An important detail with *Figure 6* is that it does not show that the more a student studies, the greater the chance of passing the exam. It only shows that the feature *time studied* is the most important feature.

When analysing the feature importance of the Random Forest algorithm that uses the split criteria Gini index one has to take under consideration that high cardinal variables can be overestimated. High cardinal means that a variable has many unique values, for example income or height of a person and therefore can split in many different ways. This

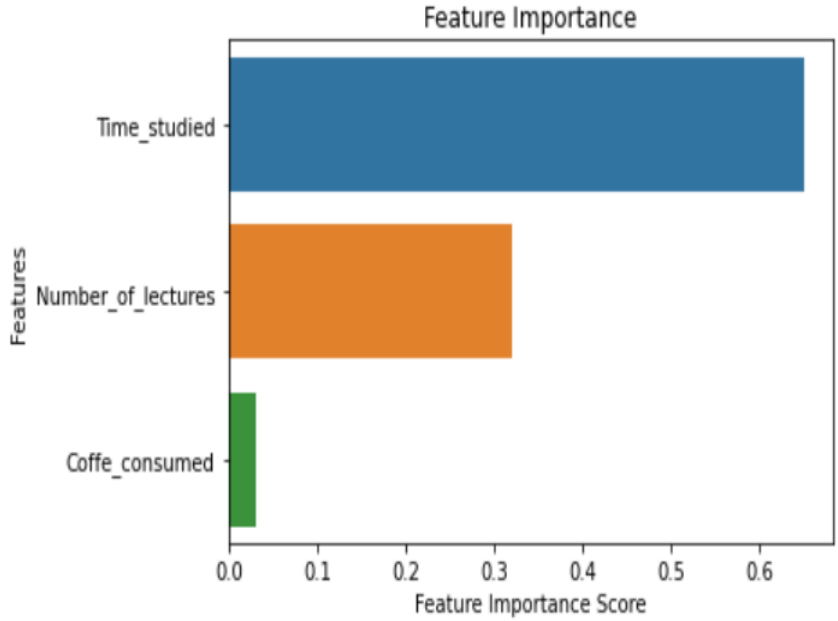

Figure 6: Example of feature importance.

can result in that a high cardinal variable by chance predicts the outcome variable well, which is referred to as an multiple testing problem (Scheidel 2018).

4.3 XGBoost

XGBoost stands for Xtreme Gradent Boosting and is an algorithm developed by Tianqi Chen (Chen & Guestrin 2016). The algorithm is an application of gradient boosting with decision trees as the weak learner and using Newton´s Method (Nielsen 2016). To understand the concept of XGBoost, this thesis first introduces the general concepts of Boosting and Newton´s method.

4.3.1 Boosting for tree-models

Boosting works similarly to earlier described Bagging in Subsection 4.2.2, but instead of creating the trees independently from the Bootstrapped data-set, the trees are built based on the information from the previously built trees. The idea of Boosting is based upon slow learning with small trees, where you fit the model step-by-step. The difference between the observed value and the estimated value, also called residual, r_i, is used to form the next tree. Hence, the residual r_i, obtained from the previous iteration, number i, is used to create the existing iteration. Each new tree created at each iteration is multiplied

with a scalar lambda, denoted by λ, that decides the weight of the new tree added. The Boosting method increases the fit of the model slowly, something that tends to generate good results (James et al. 2013).

James et al. (2013) defines boosting for the general case in these three steps, as follows;

---

1. Set $\hat{f}(x) = 0$ and $r_i = y_i$ for all i in the training set, where $\hat{f}(x)$ is the fitted function, r_i is the residual and y_i is the outcome for observation i.
2. For $m = 1, 2, ..., M$, repeat:

   a) Fit a tree $\hat{f}^m$ with d splits ($d + 1$ terminal nodes) to the training data, using the set of independent variables X and r_i as the dependent variable.
   b) Update $\hat{f}$ by adding in a shrunken version of the new tree,

$$\hat{f}(x) \leftarrow \hat{f}(x) + \lambda \hat{f}^m(x) \tag{12}$$

   where λ represents a shrinkage parameter that decides the learning rate, and can be set to 1 in the simplest case, e.g. no shrinkage.

c) Update the residuals,

$$r_i \leftarrow r_i - \lambda \hat{f}^m(x_i) \tag{13}$$

3. Output the Boosted model,

$$\hat{f}(x) = \sum_{m=1}^{M} \lambda \hat{f}^m(x) \tag{14}$$

so that the fitted function $\hat{f}(x)$ combines all the trees $\hat{f}^1 \dots f^M$ from the M steps, where $\hat{f}(x)$ then gives a weighted average probability for each observation based on all the built trees.

From *Equation* (13) we note that, the model fits every tree based on the current residual rather then using the response-variable and is therefore slowly improving by sequentially decreasing the residuals. The *"speed"* of the learning is decided through the size of each tree added e.g. number of splits, noted as d and also through λ, where a lower value of λ decreases the learning rate (Friedman 2001*a*).

4.3.2 Evaluation methods

As earlier mentioned, Boosting works by sequentially adapting the fitted function $\hat{f}(x)$ and through that improve the fit of the model. However, changing the evaluation parameters that fit the model results in different Boosting algorithms.

If we define the general process of updating the model $f(x)$ between iteration $(m-1)$ and iteration m as $f^{(m-1)}(x) \rightarrow f^{(m)}(x)$, then the change for the model at every iteration m can then be described as;

$$f^{(m)}(x) = f^{(m-1)}(x) + f_m(x) \tag{15}$$

where $f_m(x)$ is the step, or addition for the specific iteration. Two commonly used ways to decide this step, $f_m(x)$, at each iteration is either through Gradient Decent or Newton´s Method (Nielsen 2016). In this section we will only discuss Newton´s Method as it is the method used in XGBoost.

Newton´s Method

Newton's Method is a gradient-based method, with the purpose to update the model so that $r_m \leq r_{m-1}$, meaning the residuals at iteration m is smaller or equal to the previous residuals at iteration $(m-1)$. Hence, $r_M \leq \min r_i$ for $i = 1, 2 \ldots, (M-1)$, implying that each of the residuals from the $M-1$ first iterations must be larger or equal than the residual of the last iteration.

The Gradient of a function is best described as the slope of the function, and is in the case of a function with only one variable the same as the derivative of the function. The direction of the gradient is then, as for the derivative, pointing at the steepest change for the function. So, for a problem of finding a minimum, each iteration takes a step in the opposite way of the direction of the gradient (Nielsen 2016). Taking a easy example of the application of Newton's Method for finding square roots can be as follows;

To find the square root, x, of a number z, so that $x^2 = z$. Define the function $f(x)$ as;

$$f(x) = x^2 - z \tag{16}$$

which means that solving $f(x) = 0$ is the same as solving for x in the equation $x^2 = z$.

Rewriting with the notation from *Equation* (15) then gives;

$$x^{(m)} = x^{(m-1)} - \frac{f^{(m-1)}(x)}{f'^{(m-1)}(x)} \tag{17}$$

where $\frac{f^{(m-1)}(x)}{f'^{(m-1)}(x)}$ corresponds to $f_m(x)$ in *Equation* (15) and $f'^{(m-1)}(x) = 2x$ for this example.

For further intuition, assume that $z = 10$ and the initial value of x at iteration 0 is 2 e.g. $x^{(0)} = 2$

The following iterations is then;

$x^{(1)} = 2 - \frac{2^2-10}{4} = \frac{7}{2}$

$x^{(2)} = \frac{7}{2} - \frac{\frac{7}{2}^2-10}{7} = \frac{7}{2} - \frac{2{,}25}{7} = 3 + \frac{5}{28}$

$x^{(3)} = 3 + \frac{5}{28} - \frac{(3+\frac{5}{28})^2-10}{6+\frac{5}{14}} = 3 + \frac{5}{28} - \frac{81}{4984} = 3 + \frac{809}{4984} \approx 3,1623194$

$\sqrt{10} = 3,16227766$, already at iteration 3 we have a value that corresponds to the correct value of $\sqrt{10}$ to the third decimal. The accuracy will continue to increase with the number of iterations.

4.3.3 XGboost Algorithm

XGBoost is, as discussed in Subsection 4.3, an algorithm that uses Newton Boosting. Based on the scope and academic level of this thesis, the specifics of how the Newton boosting application for XGBoost differs from general Newton Boosting are omitted.

The notation used in this section is one of a risk minimization problem as proposed by Nielsen (2016), where $L(Y, f(X))$ is a differentiable loss function with the goal of minimizing the number of misclassifications given the function $f(X)$ and the dependent variable Y, where we remind the reader that $f(X)$ is generally an arbitrary function, but in our case an boosted tree function. Next, define $R(f)$ as;

$$R(f) = E[L(Y, f(X))] \tag{18}$$

where minimizing the function $R(f)$ is equivalent to minimizing;

$$R(f(x)) = E[L(Y, f(x))|X = x] \tag{19}$$

for all x in the domain of outcomes for X.
Nielsen (2016) defines the process of Newton Boosting in a general way with the following steps (see also pp. 39-41 in Nielsen (2016);

---

1. Initialize $\hat{f}^0(x) = \hat{f}_0(x) = \hat{\theta}_0 = \underset{\theta}{\operatorname{argmin}} \sum_{i=1}^{m} L(y_i, \theta)$, where y_i is the dependent variable and θ is the vector of coefficients for the current iteration e.g. the current estimate $f(x_i)$;
2. For $m = 1, 2, .., M$ do;

3. $\hat{g}_m(x_i) = \left[\frac{\partial L(y_i, f(x_i))}{\partial f(x_i)}\right]_{f(x)=\hat{f}^{(m-1)}(x)}$;

4. $\hat{h}_m(x_i) = \left[\frac{\partial^2 L(y_i, f(x_i))}{\partial f(x_i)^2}\right]_{f(x)=\hat{f}^{(m-1)}(x)}$;

5. $\hat{\phi}_m = \underset{\phi \in \Phi}{\text{argmin}} \sum_{i=1}^{n} \frac{1}{2}\hat{h}_m(x_i)[(-\frac{\hat{g}_m(x_i)}{\hat{h}_m(x_i)}) - \phi(x_i)]^2$, where $\hat{\phi}_m$ is a basis function learned at iteration m by using the the base learner $\phi(x_i)$ from the constrained set of basis functions, Φ, for example a tree, with the purpose of minimizing mean squared error at iteration m;

6. $\hat{f}_m(x) = \lambda\hat{\phi}_m(x)$;

7. $\hat{f}^{(m)}(x) = f^{(m-1)}(x) + \hat{f}_m(x)$;

8. Output :$\hat{f}^{(M)}(x) = \sum_{m=0}^{M} \hat{f}_m(x)$

At iteration m, the estimate of $f^{(m-1)}$ is evaluated in order to update the estimate to $f^{(m)}$, where the direction of the largest decrease or descent in, for this case, risk is given by the negative gradient;

$$\begin{aligned} -g_m(x) &= -\left[\frac{\partial R(f(x))}{\partial f(x)}\right]_{f(x)=f^{(m-1)}(x)} \\ &= -\left[\frac{\partial E[L(Y, f(x))|X = x]}{\partial f(x)}\right]_{f(x)=f^{(m-1)}(x)} \\ &= -E\left[\frac{\partial L(Y, f(x))}{\partial f(x)}|X = x\right]_{f(x)=f^{(m-1)}(x)} . \end{aligned} \tag{20}$$

.

Furthermore, $h_m(x)$ is the hessian at iteration m, and is calculated as;

$$\begin{aligned} h_m(x) &= \left[\frac{\partial^2 R(f(x))}{\partial f(x)^2}\right]_{f(x)=f^{(m-1)}(x)} \\ &= \left[\frac{\partial^2 E[L(Y, f(x))|X = x]}{\partial f(x)^2}\right]_{f(x)=f^{(m-1)}(x)} \\ &= E\left[\frac{\partial^2 (L(Y, f(x))|X = x)}{\partial f(x)^2}\right]_{f(x)=f^{(m-1)}(x)} \end{aligned} \tag{21}$$

In step five in the algorithm, $\hat{\phi}_m$, is to be seen as a learning parameter for the step direction, but a constrained one, where the constrain is a restriction on possible sets of functions, which then works to prevent over-fitting of the data. In the case of XGBoost, it would be a tree-related learning algorithm and the constraint is typically related to the

form of the tree (Nielsen 2016).

Step six in the XGBoost algorithm uses λ, *shrinkage* or also often referred to as *learning rate* which corresponds to the λ in *Equation* (12)-(14). Learning rate was first introduced by Friedman (2001*b*), and works as a shrinkage method which determines the speed of learning for the model.

The steps 2-5 in the XGBoost algorithm together form the change taken at iteration m, which is seen in step 6. And the output is then the summed changes or "steps" at each iteration, seen in step 8.

4.3.4 Feature importance - XGBoost

Calculating the importance of the included explanatory variables in a model, or here described as feature importance, can be a important tool in understanding the model. There exists different approaches to calculate this feature importance depending on the purpose. For a general illustrating example on the use of feature importance, see Subsection 4.2.4. Abu-Rmileh (2019), however, states that the use of so called *"Gain"* to evaluate feature importance is the most relevant way to analyze the relative importance of features for XGBoost. Gain is measuring the improvement in accuracy of the model when including a new feature on a given branch. It can be describe as if we have a existing branch of a tree, excluding variable x, which results in some amount of misclassifications. The increase in accuracy, or decrease in misclassifications, by adding a new split with variable x is then the gain. The feature importance is visualized as relative percentage gain, so that all features get a *feature importance score* which together sum to one.

5 Data

The data used in this thesis was obtained from a hotel based in Gothenburg, Sweden and includes bookings and associated information between 2016-01-01 and 2020-12-31. In addition, data including temperature and precipitation was collected for the days within the range. This infromation regarding weather was collected from SMHI, Sweden's meteorological and hydrological institute and was merged with date of arrival for each observation (SMHI 2021). A full description of the included variables is listed in the appendix in *Table 9*.

Since the time-frame of the data includes the years 2016 to 2020 some of the observations are affected by COVID-19. During the pandemic, 2020-2021, costumer behavior have changed significantly. Therefore, the observation for year 2020 are excluded from the data. If the observations would not be excluded the models would be trained with an

anomaly. Training data with a large proportion of deviating observations can disrupt the general trends and patterns and the models would perform worse on future data.
The original given data-set contained 1 896 579 observations and 22 possible variables for each observations, where each observation corresponds to a booking. Excluding the observations for the year 2020 resulted in a data-set of 1 638 735 observations as can be seen in *Table* 1. Adding the data over the weather, temperature and precipitation, resulted in a data-set with the dimensions of 1 638 735 x 24.

Data	Removed Observations	Remaining Observations
Total Observations	0	1 896 579
Removed based on year 2020	-257844	1 638 735

Table 1: Overview of the data from a hotel in Gothenburg, used in this thesis.

A number of variables were adjusted or re-categorized, either for reasons of practical matter, e.g. convert categories into dummies, or for reasons of inadequate data. Variables with categories that contained less than 1 % of the observations were merged, either with the most similar subcategory, or if that was not possible there were instead created as a new subcategory called "other". This merging of some data was done to avoid inaccurate conclusions, as this otherwise could result in the model assigning the categories in question disproportional large importance to whether a booking is cancelled or not. Some variables were completely excluded. Most of the variables were excluded based on redundant information, e.g. "Date of booking" is not needed as the information is contained in the variables "Arrival- Year, -Month and -Day". The variable "Number of adults" was also excluded based on the fact that the booking-system of the hotel had an approach of setting this value to 0 if a booking had been cancelled, which makes it unusable.

6 Results

In this section, the results of the statistical methods applied on our data will be presented. The results will be broken down for each single model including and excluding weather data. The process of obtaining the result in this thesis is a combination of validation set approach and cross validation presented in Subsection 2.3 and is summarized in *Figure 7* below. More specific, the original data is randomly split to a training dataset and a testing dataset which are equal in size.

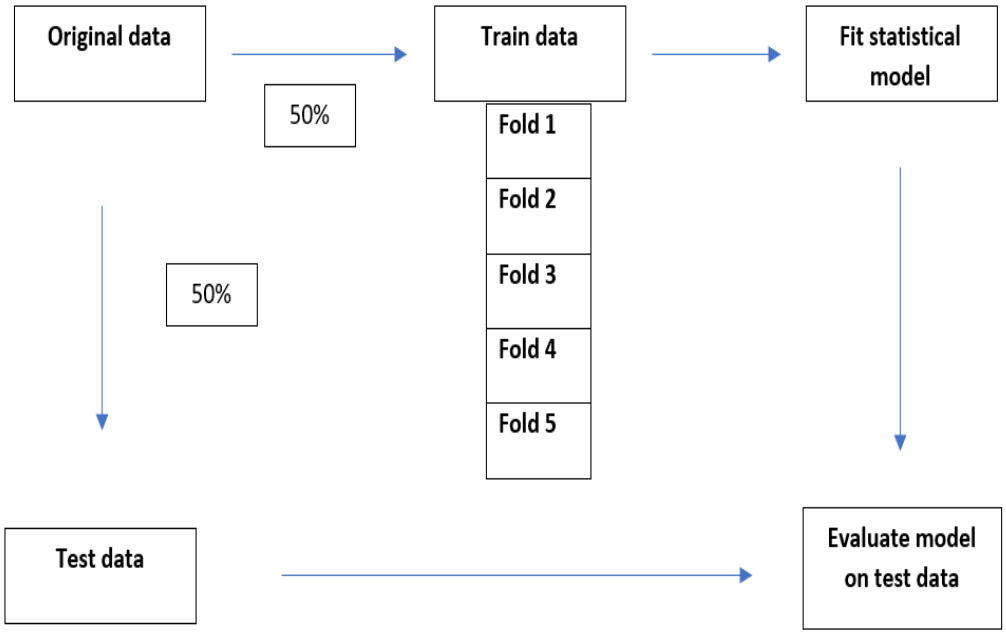

Figure 7: The process of obtaining the result for the thesis.

Each model is trained and fitted on the training dataset. To optimize each model, the model is tuned by combining different values of the hyperparameters with the cross-validation method. The hyperparameters tuned in Random Forest are number of trees, max depth and how many features that are used in every split. The hyperparameters tuned in XGBoost are max depth, min split loss and learning rate. The Logit method has no hyperparameters to tune and therefore skips this part of the process. After the optimal model has been selected and trained it is evaluated on the test dataset, which is the most important part of the process because the model is evaluated on data that it has not seen before, which gives a credible result.

All computations are made in the programming language *Python*, were the package *Scikit-learn* has been used, which means that the algorithms was not programmed from scratch (Pedregosa et al. 2011).

6.1 Logistic Regression

In this subsection we present some results when using logistic regression as a model for predicting booking cancellations on our hotel data. The two confusion matrices, in *Table 2* & *Table 3*, displays the result of the Logistic regression evaluated on test-data, with and without including weather-data. The rows are the actual classes and the columns are the predicted classes, where 1 represents a cancellation and 0 a non-cancellation. Here, Logistic regression uses the threshold 0.5 to determine the predicted class for the observation. The optimal value of the threshold varies depending on the model and the data used, as further discussed in Subsection 2.2. Therefore AUC is a better metric to evaluate and compare models.

Class	0	1
0	557 934	36 905
1	163 846	60 683

Table 2: Confusion matrix for Logistic Regression excluding weather

- Total accuracy of the model is 75.5%.
- Proportion of bookings which have been wrongly classified as cancellations (*False positive rate*) is 6.20%.
- Proportion of cancellations which have been wrongly classified as bookings (*False negative rate*) is 72.97%.

Class	0	1
0	559 461	35 932
1	163 880	60 095

Table 3: Confusion matrix for Logistic Regression including weather

- Total accuracy of the model is 75.6%.
- Proportion of bookings which have been wrongly classified as cancellations (*False positive rate*) is 6.03%.
- Proportion of cancellations which have been wrongly classified as bookings (*False negative rate*) is 73.17%.

Figure 8 & *Figure 9* display the ROC-curve for the Logit model applied to test-data from the hotel data presented in *Table 1*, *Section 5*, with weather data included and excluded. The AUC for both Logit models are the same and equals 0.6. We here remark that ROC-curve for Logistic Regression are created via the condition $\hat{p}(X) > T$, where the threshold T ranges between 0 and 1 and $\hat{p}(X)$ is constructed as in Subsection 4.1 and X is a vector representing the independent variables.

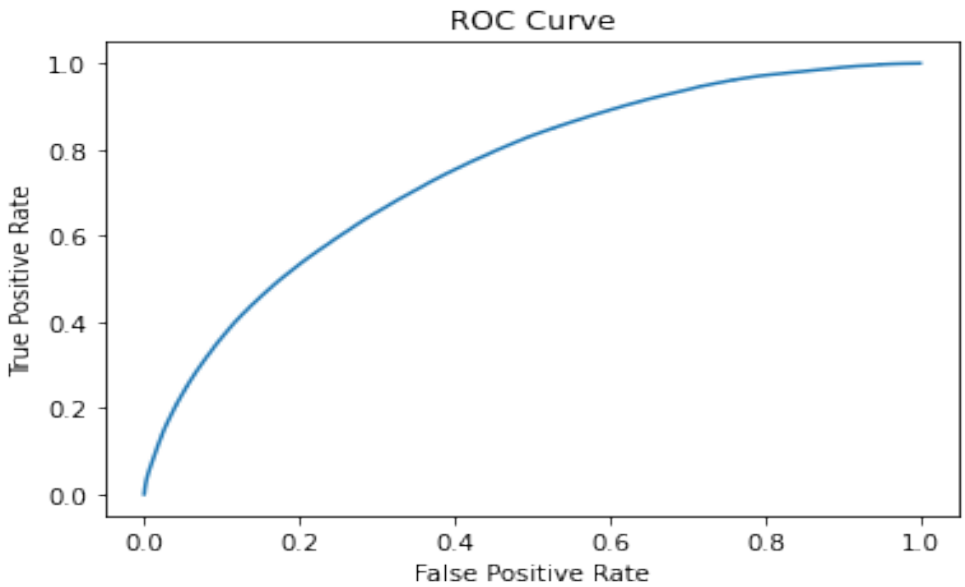

Figure 8: Receiver operating characteristic Logit, AUC = 0.6. (weather excluded)

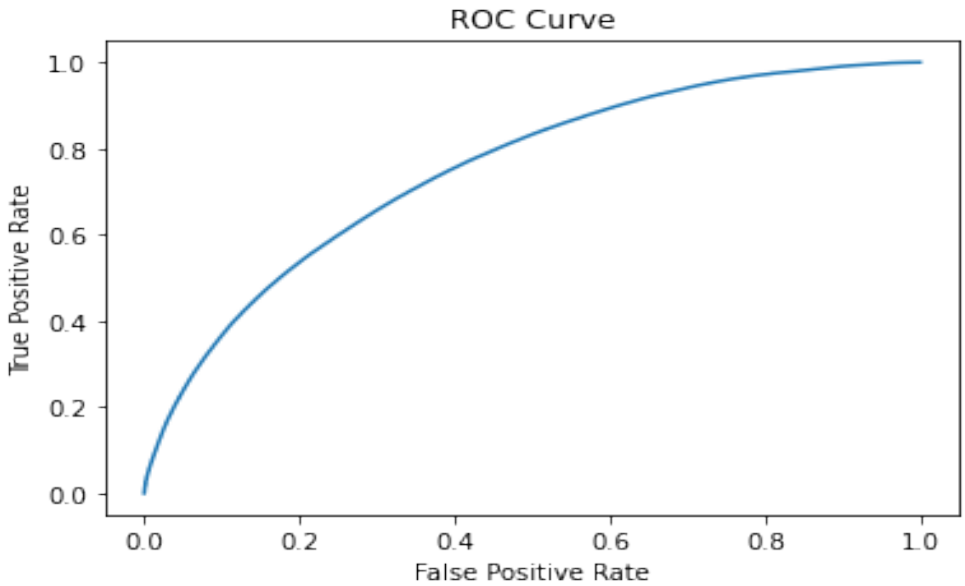

Figure 9: Receiver operating characteristic Logit, AUC = 0.6 (weather included)

6.2 Random Forest

In this subsection we present some results when using Random Forest as a model for predicting booking cancellations on our hotel data. The two confusion matrices, in *Table 4* & *Table 5*, displays the result of the Random Forest model evaluated on test-data, with and without including weather-data. The rows are the actual classes and the columns are the predicted classes, where 1 represents a cancellation and 0 a non-cancellation. The Random Forest model uses majority voting to determine the predicted class for the observation, which in this binary case corresponds to using a threshold of 0.5. The optimal value of the threshold varies depending on the model and the data used, as further discussed in Subsection 2.2. Therefore AUC is a better metric to evaluate and compare models.

Confusion matrix	0	1
0	528 010	66 829
1	108 169	116 360

Table 4: Confusion matrix for Random Forest excluding weather

- Total accuracy of the model is 78.6%.
- Proportion of bookings which have been wrongly classified as cancellations (*False positive rate*) is 11.2%.
- Proportion of cancellations which have been wrongly classified as bookings (*False negative rate*) is 48.2%

Confusion matrix	0	1
0	533 939	61 454
1	107 613	116 362

Table 5: Confusion matrix for Random Forest including weather

- Total accuracy of the model is 79.4%
- Proportion of bookings which have been wrongly classified as cancellations (*False positive rate*) is 10.3%.
- Proportion of cancellations which have been wrongly classified as bookings (*False negative rate*) is 48%.

Figure 10 & *Figure 11* displays the ROC-curve for the Random Forest-model applied to test-data from the hotel data presented in *Table 1*, with weather data included and excluded. The AUC for Random Forest excluding weather data equals 0.7 and including weather data equals 0.71. We here remark that ROC-curve for Random Forest are created via the condition $\hat{C}(x) > T$ where the threshold T ranges between 0 and 1 and $\hat{C}(x)$ is constructed as in Subsection 4.2.3 and x is a vector representing the independent variables.

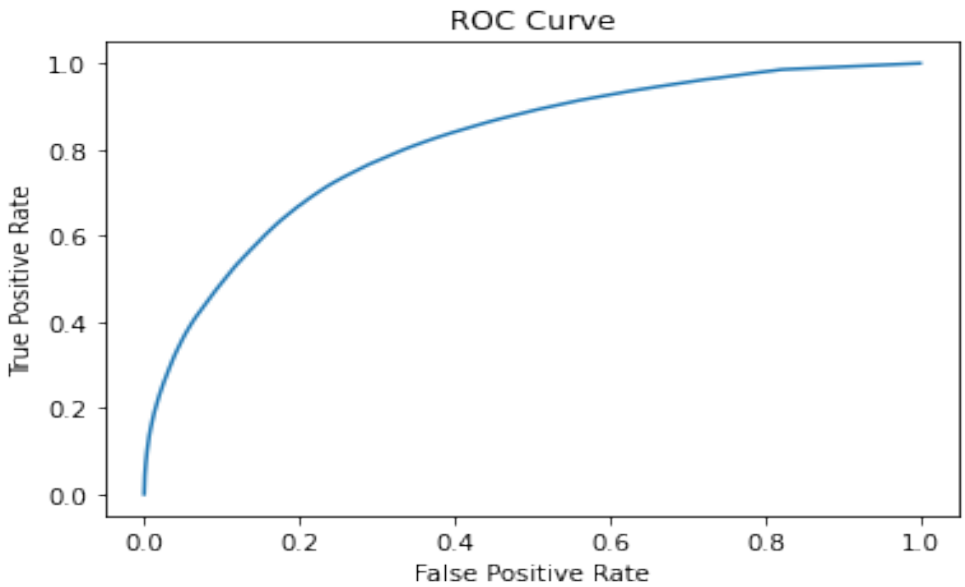

Figure 10: Receiver operating characteristic Random Forest, AUC=0.70 (excluding weather)

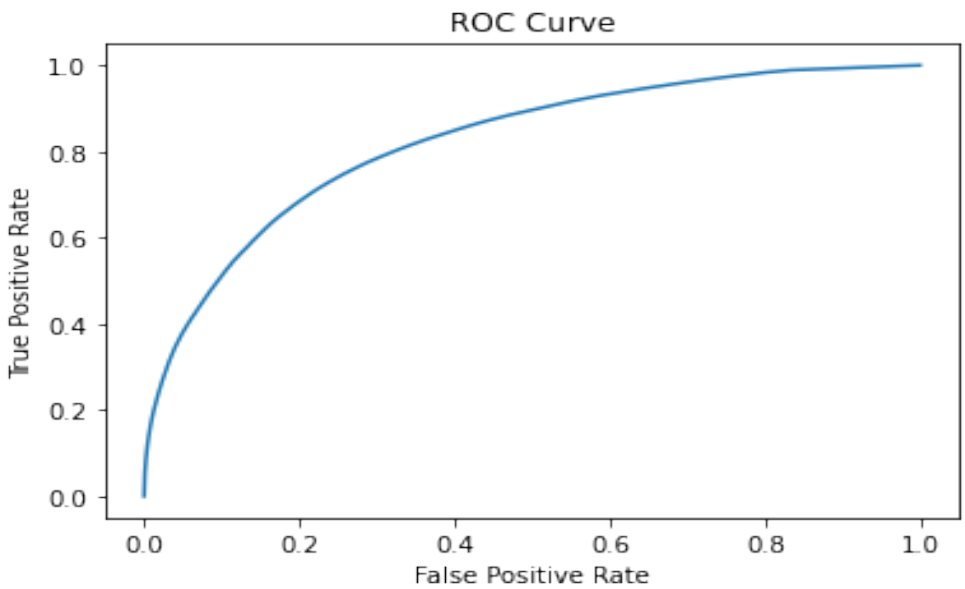

Figure 11: Receiver operating characteristic Random Forest, AUC=0.71 (including weather)

Figure 12 & *Figure 13* display the top 10 feature importance for the Random Forest model based on the Gini criteria, including and excluding the weather data. As mentioned in Subsection 4.2.4, feature importance do not indicate if a feature have a negative or positive impact on the outcome variable, it only show how much information each variable contains in order to determine which class an observation belongs to. In *Figure 12* Leadtime is by far the most dominant variable, with the rest of the top 10 variables having approximately the same score. The inclusion of weather data in *Figure 13* does not change Leadtimes dominant position as the most important feature, but both temperature and precipitation stands out and rank second and third respectively.

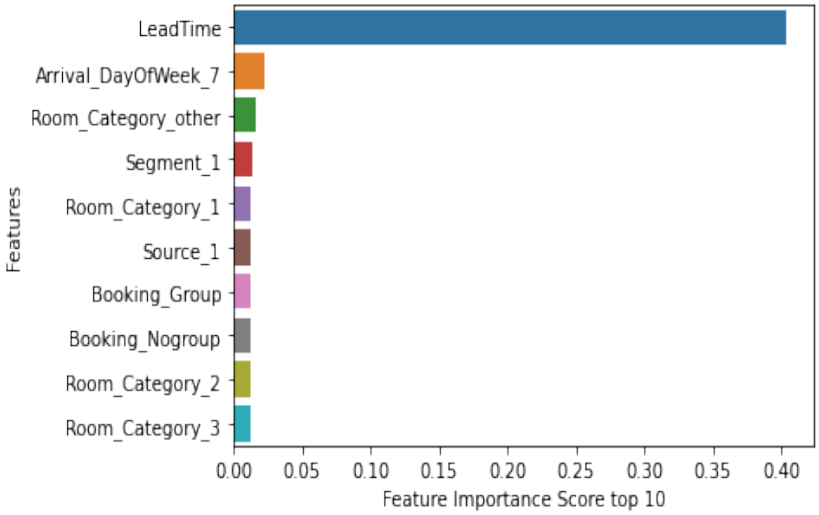

Figure 12: Feature importance Random Forest (excluding weather)

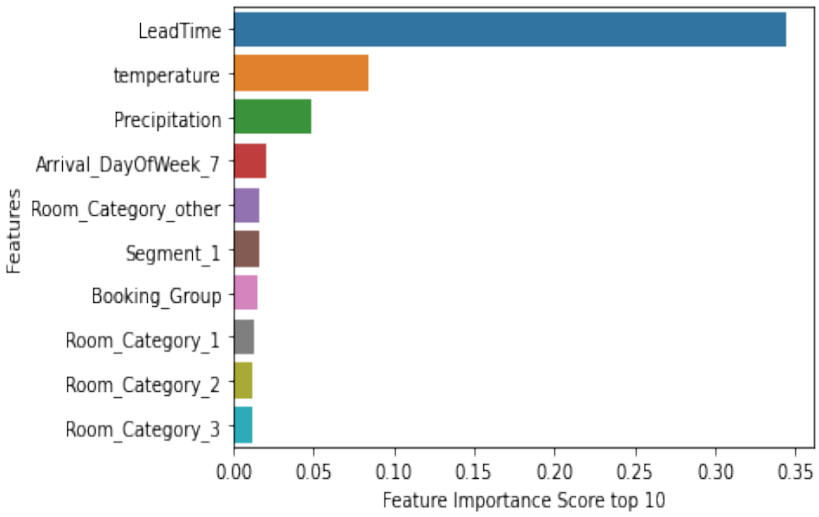

Figure 13: Feature importance Random Forest (including weather)

6.3 XGBoost

In this subsection we present some results when using XGBoost as a model for predicting booking cancellations on our hotel data. The two confusion matrices, in *Table 6* & *Table 7*, displays the result of the XGBoost model evaluated on test-data, with and without including weather-data. The rows are the actual classes and the columns are the predicted classes, where 1 represents a cancellation and 0 a non-cancellation. Here, the XGBoost model uses the threshold 0.5 to determine the predicted class for the observation. The optimal value of the threshold varies depending on the model and the data used, as further discussed in Subsection 2.2. Therefore AUC is a better metric to evaluate and compare models.

Confusion matrix	0	1
0	559 390	35 449
1	145 127	79 402

Table 6: Confusion matrix for XGBoost excluding weather

- Total accuracy of the model is 77.9%.
- Proportion of bookings which have been wrongly classified as cancellations (*False positive rate*) is 5.96%.
- Proportion of cancellations which have been wrongly classified as bookings (*False negative rate*) is 64.6%.

Confusion matrix	0	1
0	560 229	35 164
1	143 790	80 185

Table 7: Confusion matrix for XGBoost including weather

- Total accuracy of the model is 78.1%.
- Proportion of bookings which have been wrongly classified as cancellations (*False positive rate*) is 5.9%.
- Proportion of cancellations which have been wrongly classified as bookings (*False negative rate*) is 64.2%.

Figure 14 & *Figure 15* display the ROC-curve for the XGBoost model applied to test-data from the hotel data presented in *Table 1*, with weather data included and excluded. The AUC for both models are the same and equals 0.65. We here remark that ROC-curve for XGBoost are created via the condition $\hat{f}^{(M)}(x) > T$ where the threshold T ranges between 0 and 1 and $\hat{f}^{(M)}(x)$ is constructed as in Subsection 4.3.3 and x is a vector representing the independent variables.

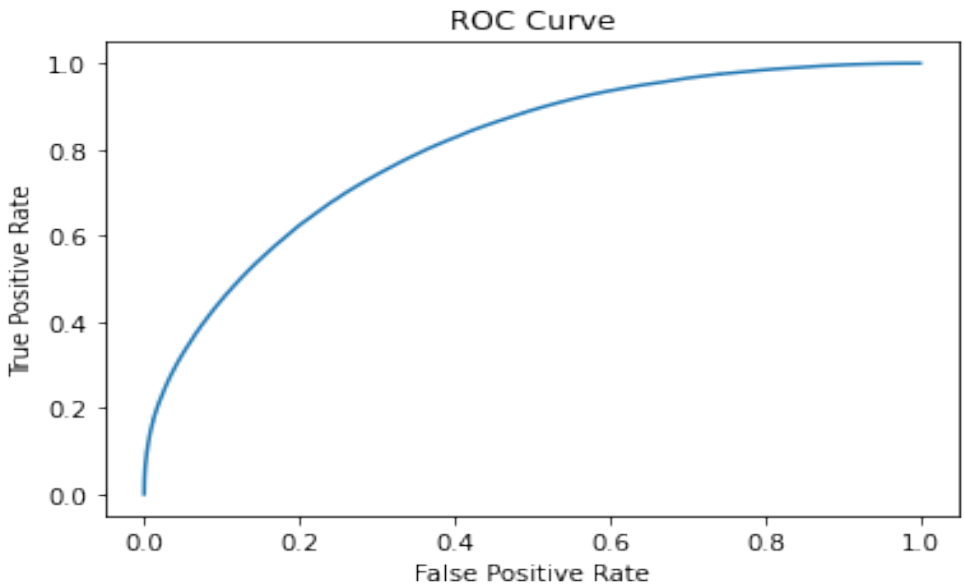

Figure 14: Receiver operating characteristic XGBoost, AUC=0.65 (excluding weather)

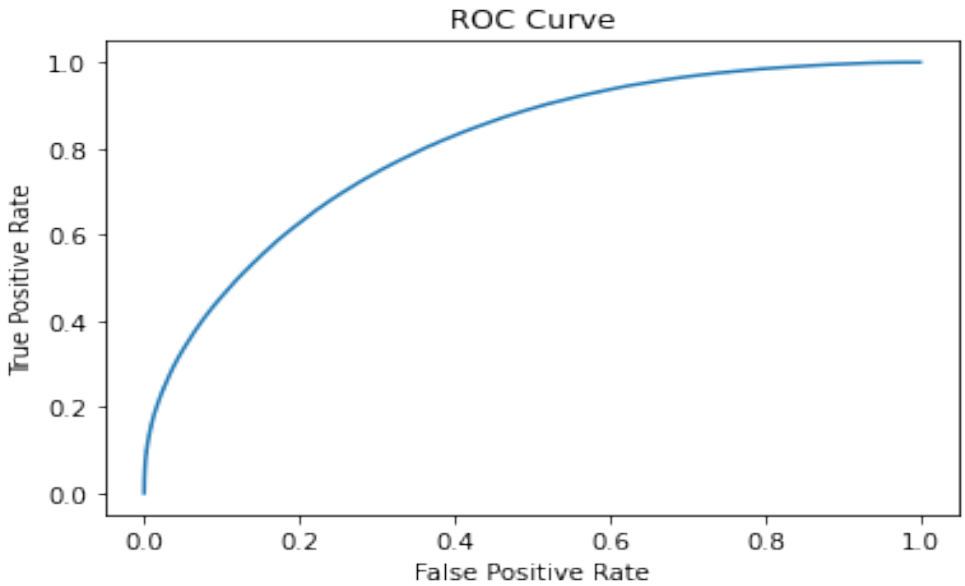

Figure 15: Receiver operating characteristic XGBoost, AUC=0.65 (including weather)

Figure 16 & *Figure 17* show the top 10 feature importance for the XGBoost model based on Gain, including and excluding the weather data, as described in Subsection 4.3.3. In *Figure 16* *Segment_1*, which is a categorical variable that describes the purpose of travel for example leisure or work, ranks the highest, followed by *Booking_No_Group* and *Room_category_4*. *Booking_No_Group* means that the observation does not belong to a group booking and *Room_category_4* is a certain category of room. With the inclusion of weather data in *Figure 17*, *Segment_1* and *Booking_No_Group* is still the two features with the highest score, but here followed by *Room_category_other*. We can also note that none of the weather variables are present in *Figure 17*.

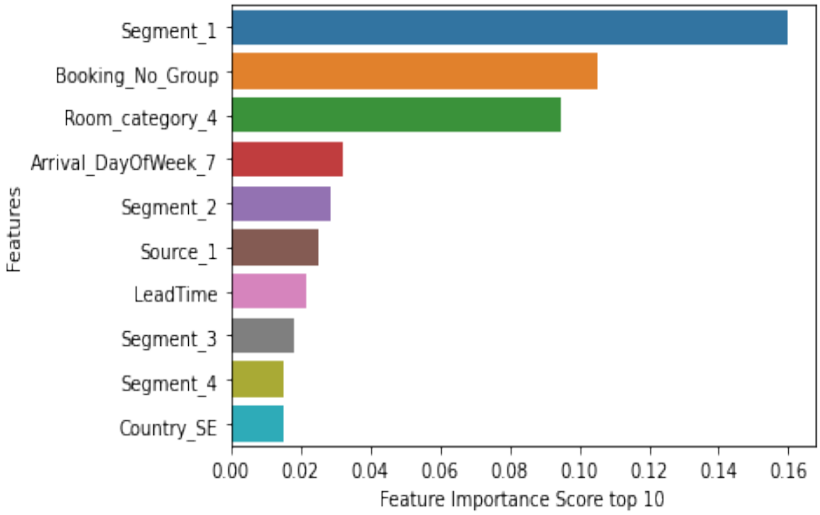

Figure 16: Feature importance XGBoost (excluding weather)

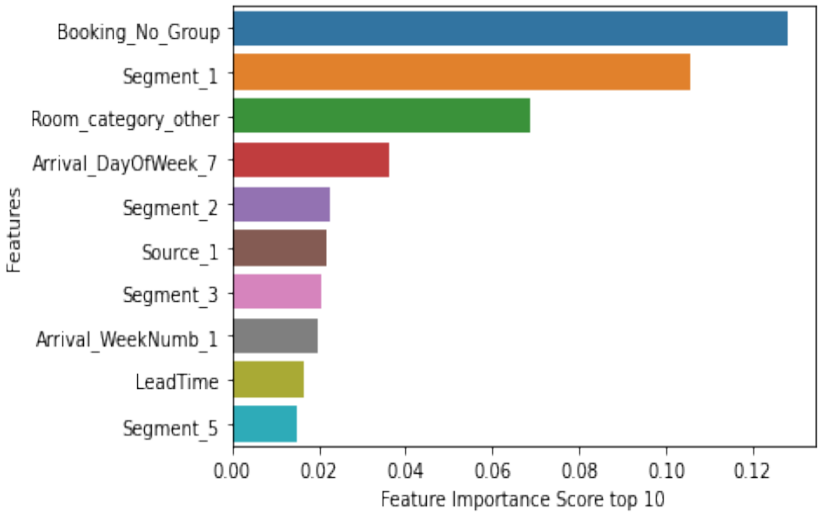

Figure 17: Feature importance XGBoost (including weather)

6.4 Summary

Model*	Accuracy *(%)*	FP-Rate *(%)*	FN-Rate *(%)*	AUC *(%)*
Random Forest	78,6	11,2	48,2	0,70
XGBoost	77,9	5,96	64,6	0,65
Logistic	75,5	6,2	72,97	0,60
Random Forest W	79,4	10,3	48	0,71
XGBoost W	78,1	5,9	64,2	0,65
Logistic W	75,6	6,03	73,17	0,60

Table 8: Summary of model performances on the hotel data.

Table 8 shows a summary of the results for the three models with and without weather, where W in the last three rows indicates that weather is included in the model. Random Forest outperforms Logistic Regression and XGBoost in regards to accuracy, with and without weather data. Furthermore, all models show higher False-negative rates compared to False-positive rates. Note that the FP-rate and FN-rate depends on the threshold, which in this case is 0.5. Therefore, a more robust way of comparing the performance between the models is the AUC-score. The AUC-score shows that the Random Forest model is the best performing model, independent of the inclusion of weather data.

Below in *Figure 18* and *Figure 19* the ROC-curves for all models are displayed with weather data included and excluded. Note, these curves are the same as represented in the earlier Subsections 6.1 - 6.3, but here visualized in the same graph. We can clearly see that all curves in *Figure 18* and *Figure 19* are above the dotted lines, and therefore performs better than chance, as discussed in Subsection 2.2. Furthermore, *Figure 18* and *Figure 19* displays that the Random Forest model performs best for most of the threshold values, but not all. Logit is the worst performing model for all threshold values.

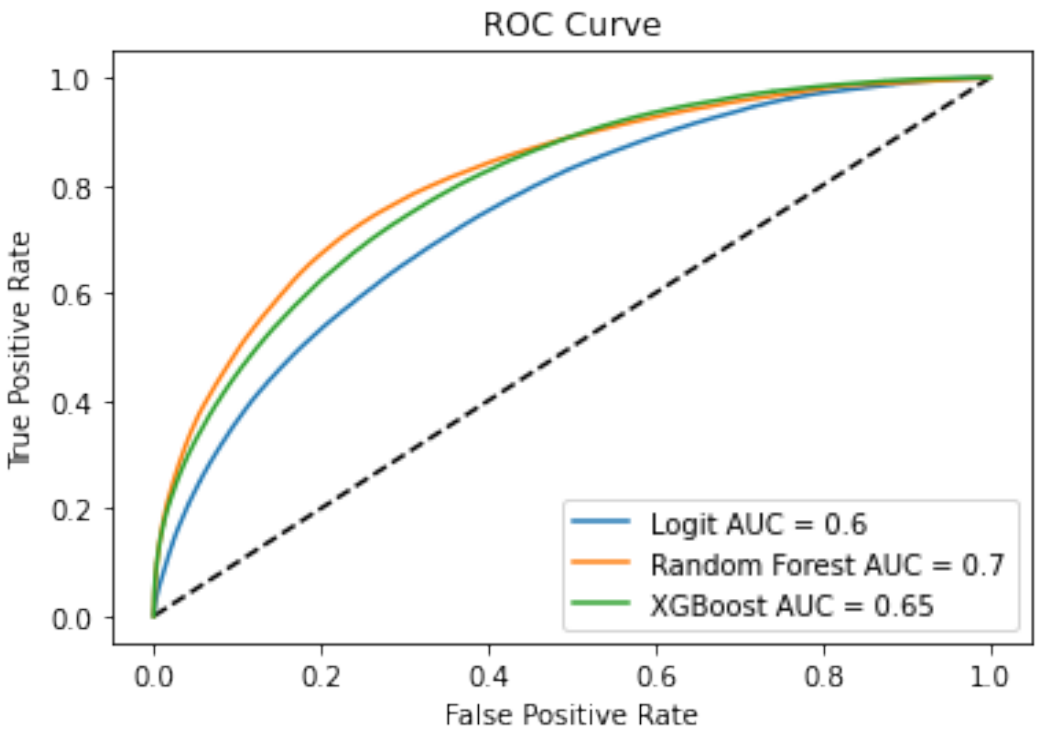

Figure 18: ROC-curves of all models with weather data excluded.

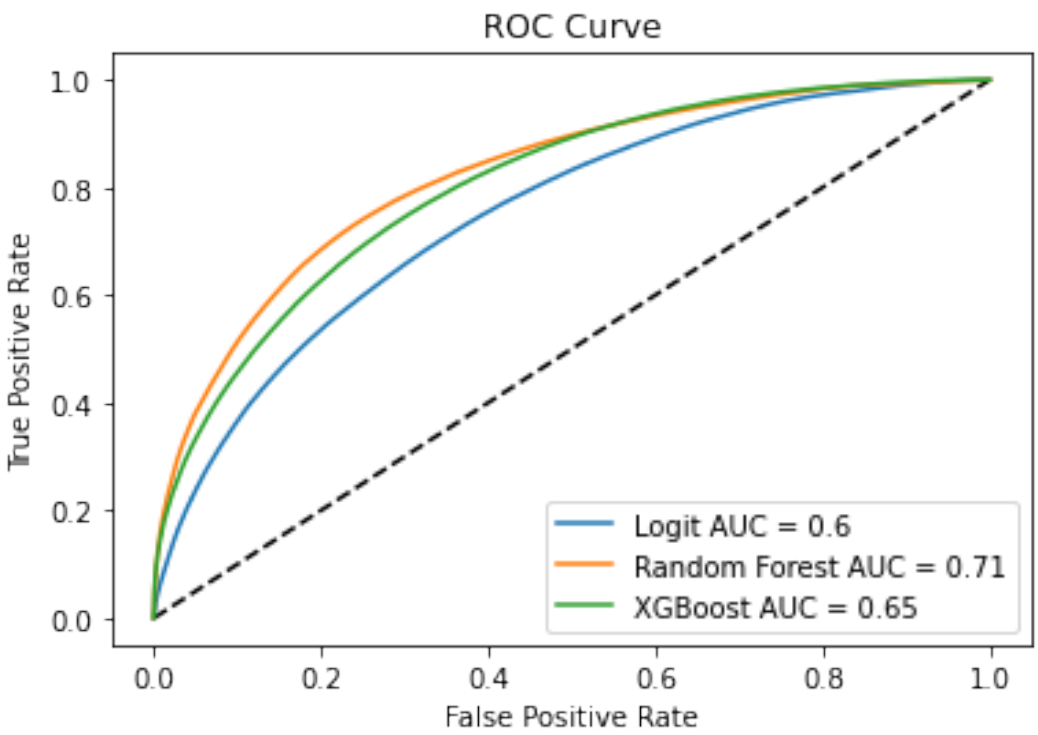

Figure 19: ROC-curves of all models with weather data included.

7 Analysis

7.1 Research question 1

Can cancellations for hotel guests be predicted with the use of machine-learning, based on the given data?

The results presented from this study indicate that machine learning can predict hotel cancellations based on the given data from a hotel in Gothenburg, Sweden. After excluding the observations for year 2020, the proportion of non cancellations in the data-set is 72.6%. If one would assume no cancellation this would yield an accuracy of 72.6%. Since all models have a higher accuracy then 72.6% this suggest that using statistical tools and machine-learning algorithms enables prediction over cancellations for hotel guests. Furthermore, the two models Random Forest and XGBoost outperform the logistic regression in regards to accuracy and AUC. This supports the idea that machine-learning, and especially tree-based models are not only applicable but a good choice for predicting cancellations within the given data.

From the results in Section 6 we see that all models have higher false negative rates compared to the false positive rates, which indicates that it might be harder to identify cancellations as compared to identify non-cancellations. However, as discussed in Subsection 2.2, changing the threshold for the predictions can change this relationship between false positive and false negatives. Based on this, the AUC is a more relevant criteria than false positive and false negative rate when evaluating and comparing the models.
By evaluating the models based on accuracy and AUC, it is clear that Random Forest is the model that yields the highest score, followed by XGBoost and with Logistic regression in last place. So, for this data-set and these three choices of models, both AUC and accuracy indicates that tree-based models are suitable for predicting cancellations.

Once again it is worth mentioning that the results of this study is only applicable to the hotel in question. The suitability of machine-learning as a tool for general problems on hotel-cancellations are thus not evaluated in this study. However, as the explanatory variables used in the models can be assumed to be available for most hotels, this supports the idea of that machine-learning models are applicable for predicting cancellations within the hotel-industry.

7.2 Research question 2

What factors are most influential when predicting cancellations?

From the results of the study one can see that the variables in feature importance differ between Random Forest and XGBoost which are two different tree-based models.

Random Forest is created through the Bootstrap and Baggning method and the feature importance is determined by the Gini criteria. XGBoost uses the Boosting method and the feature importance is based on *Gain*. Therefore it is not surprising that the variables differ and one cannot directly compare the feature importance between the models but instead analyze which factors are important for each model. As described in Subsection 4.2.4 and 4.3.3, feature importance does only indicate the importance of each feature and not if it has a negative or positive effect on the outcome variable. The below discussion as to how the variables affect hotel cancellations should therefore be seen as speculative.

The results of the study show that leadtime is the variable that contains the most information to predict hotel cancellations in the Random Forest model. Leadtime is a numeric variable and represent the days between when the hotel reservation is made and the day of arrival. It is intuitive why leadtime contains a lot of information. If an individual makes a hotel booking with the day of arrival well in advance, the person also has many opportunities to cancel the reservation. Another aspect may be that it is an easier decision to cancel an event if it occurs far in the future in comparison if it takes place sooner. At the same time a reservation booked in a short notice i.e low leadtime may also have a higher risk of cancellation due to sudden change of plans.

The most influential features in the XGBoost algorithm are *Segment 1* and *Booking NO Group*. Segment is a categorical variable that describes the purpose of travel for example leisure or work and *Booking NO Group* means that the observation does not belong to a group booking. One can assume that there is a difference in cancellation rate between guests who are traveling for different reasons, where work related bookings can be associated with lower risk of cancellation and non work-related bookings have a higher risk of cancellation. The same reasoning can be applied for group bookings, where reservations that belong to a group may have a smaller chance for cancellations compared to independet bookings.

Other variables that had a relatively large importance in both Random Forest and XGBoost were Sundays and different room categories. It is difficult to understand why especially Sundays would contain more information than other weekdays in determining cancellations. However, different kinds of hotel rooms could be correlated with external events and therefore help the model to predict. If an individual books a specific type of hotel room, for example a suite, it is reasonable to assume this person is there for a specific reason, either to celebrate something or attending a specific event. If the assumption is correct, the individual is less likely to cancel the hotel reservation because he/she is also attending an external event.

As described in Subsection 4.2.4, the importance of high cardinal values, i.e variables with a large number of unique values, have a risk of being overestimated in the Random Forest model. Leadtime, which is a high cardinal variable, should therefore be interpreted with caution. Having said that, Leadtime has a high risk of being overestimated in the result for the Random Forest model but since it is so dominant our hypothesis is that it still is the most influential variable. Previous studies of hotel cancellations presented in Subsection 3.1-3.4 have shown that leadtime is the most influential variable in some of their models, which supports our hypothesis.

7.3 Research question 3

Can external data in the form of weather improve predictions for cancellations?

Based on the results of this study, the inclusion of weather-data marginally increases predictive power of the models, which can be interpreted as support of research question three. However, there are certain aspects that needs reconsideration when interpreting the result. First and foremost, the weather-data used in this study over precipitation and temperature are realized values as compared to using weather-forecasts. This implies that it is not possible to recreate the model and actually use it for predictive purposes as the used weather data would not be available at the time of the prediction. Furthermore, if the decision to cancel a booking for a individual is affected by weather, then this decision would be based on the weather-forecast at the time.
So, for realized weather to have predictive power over cancellations we rely on that it is positively correlated with weather-forecasts. Since weather is difficult, or close to impossible to predict with accuracy far in the future, this correlation exists most likely only close to the date of stay. Hence, our use of weather can be assumed to decrease in predictive power for cancellations long before the date of stay.

From the results of feature importance, both temperature and precipitation rank high for the Random Forest model. Especially temperature results in a high feature importance score, which would suggest that the features have a high contribution for accurately predicting cancellations. However, there is reason to suspect that some of the importance of the feature temperature stems from other factors. This is mainly based on that the busiest periods of the hotel, with the most customers and the highest share of cancellations are during December, January, June, July and August, which are also the months where we observe weather extremes. The fact that cancellation rates are higher for months which traditionally have high and low temperatures implies that part of the observed importance of temperature might originate just from that it correlates with the more active periods of the hotel rather than temperature being decisive itself. Furthermore, as discussed in

Subsection 4.2.4, high cardinal variables as temperature have a tendency to be overestimated when using the Gini index. The fact that the weather-variables does not appear in the top 10 for XGBoost, which does not have the cardinal problem, can be seen as further support for that the real feature importance is smaller than what the results show. This does not mean that the temperature alone have no predictive power, but suggests that the effect might be over-estimated by the feature importance.

To summarize, the result support that weather data can improve predictions of cancellations, but that especially the results from the Random Forest model is likely to overestimate the effect. However, using forecasts instead of realized weather is likely to improve the quality of the models.

7.4 Conclusions

- Cancellation of hotel reservations can be predicted with the use of machine-learning for a specific hotel.
- Leadtime, that is the number of days between the booking and the date of arrival, is the most influential variable when predicting cancellations with the Random Forest algorithm. The variables that had highest importance in the XGBoost model were *Segment 1* and *Booking NO Group*.
- External weather data containing precipitation and temperature improves the result for the models marginally.

8 Future research

This thesis studies if it is possible to predict hotel cancellations with the use of machine learning and which variables that have the greatest significance in the models. To gain a more nuance understanding about the subject the following topics for future research are presented.

The time aspect is discarded in this thesis but would be interesting to investigate. As mentioned earlier in the thesis, it is cancellations that take place close to the day of arrival that create problems for hotels. A future study could analyse how well machine learning algorithms can predict cancellations that occur within e.g., 14 days in the future.

The results of this thesis indicates that external weather data in form of precipitation and temperature improves the results of the models. However, as the thesis use historical weather data in realized values, a future study could be based on weather forecasts, since the decision to cancel a reservation can be assumed to be based on the current forecast.

Another interesting topic of future research could be to estimate the economical effect on the hotel after implementing a machine-learning prediction system. This future research will probably be in the field of business but can highlight the monetary value of a statistical solution.

References

Abu-Rmileh, A. (2019), 'The multiple faces of 'feature importance' in xgboost'. Last visited 2021-12-15.
URL: *https://towardsdatascience.com/be-careful-when-interpreting-your-features-importance-in-xgboost-6e16132588e7*

Alpaydin, E. (2020), *Introduction to machine learning*, MIT press.

Andriawan, Z. A., Purnama, S. R., Darmawan, A. S., Wibowo, A., Sugiharto, A., Wijayanto, F. et al. (2020), Prediction of hotel booking cancellation using crisp-dm, *in* '2020 4th International Conference on Informatics and Computational Sciences (ICICoS)', IEEE, pp. 1–6.

Antonio, N., de Almeida, A. & Nunes, L. (2017), 'Predicting hotel booking cancellations to decrease uncertainty and increase revenue', *Tourism & Management Studies* **13**(2), 25–39.

Antonio, N., de Almeida, A. & Nunes, L. (2019), 'Big data in hotel revenue management: Exploring cancellation drivers to gain insights into booking cancellation behavior', *Cornell Hospitality Quarterly* **60**(4), 298–319.

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

Brown, S. (2021), 'Machine learning, explained'. Last visited 2021-11-30.
URL: *https://mitsloan.mit.edu/ideas-made-to-matter/machine-learning-explained*

Chen, T. & Guestrin, C. (2016), Xgboost: A scalable tree boosting system, *in* 'Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining', KDD '16, Association for Computing Machinery, New York, NY, USA, p. 785–794.
URL: *https://doi.org/10.1145/2939672.2939785*

Chiang, W.-C., Chen, J. C. & Xu, X. (2007), 'An overview of research on revenue management: current issues and future research', *International journal of revenue management* **1**(1), 97–128.

cmglee (2018), 'Roc-draft-xkcd-style.svg'. Last visited 2022-01-03.
URL: *https://commons.wikimedia.org/w/index.php?curid=109730045*

Efron, B. & Tibshirani, R. J. (1994), *An introduction to the bootstrap*, CRC press.

Fawcett, T. (2006), 'An introduction to roc analysis', *Pattern Recognition Letters* **27**(8), 861–874. ROC Analysis in Pattern Recognition.

Fradkov, A. L. (2020), 'Early history of machine learning', *IFAC-PapersOnLine* **53**(2), 1385–1390. 21st IFAC World Congress.

Friedman, J. H. (2001*a*), 'Greedy function approximation: A gradient boosting machine.', *The Annals of Statistics* **29**(5), 1189 – 1232.
URL: *https://doi.org/10.1214/aos/1013203451*

Friedman, J. H. (2001*b*), 'Greedy function approximation: a gradient boosting machine', *Annals of statistics* pp. 1189–1232.

Géron, A. (2019), *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: Concepts, tools, and techniques to build intelligent systems*, O'Reilly Media.

Hastie, T., Tibshirani, R. & Friedman, J. (2009), *The Elements of Statistical Learning: Data Mining, Inference, and Prediction*, Springer series in statistics, Springer.

Ho, T. K. (1995), Random decision forests, *in* 'Proceedings of 3rd international conference on document analysis and recognition', Vol. 1, IEEE, pp. 278–282.

James, G., Witten, D., Hastie, T. & Tibshirani, R. (2013), *An introduction to statistical learning*, Vol. 12, Springer.

Kimes, S. E. & Wirtz, J. (2003), 'Has revenue management become acceptable? findings from an international study on the perceived fairness of rate fences', *Journal of service research* **6**(2), 125–135.

Muchai, E. & Odongo, L. (2014), 'Comparison of crisp and fuzzy classification trees using gini index impurity measure on simulated data', *European Scientific Journal* **10**(18).

Ng, A. (2021), 'What is machine learning?'. Last visited 2021-11-28.
URL: *https://www.coursera.org/lecture/machine-learning/what-is-machine-learning-Ujm7v*

Nielsen, D. (2016), Tree boosting with xgboost-why does xgboost win" every" machine learning competition?, Master's thesis, NTNU.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M. & Duchesnay, E. (2011), 'Scikit-learn: Machine learning in Python', *Journal of Machine Learning Research* **12**, 2825–2830.

Piatetsky-Shapiro, G. (2014), 'Kdnuggets methodology poll'. Last visited 2021-11-11.
URL: *https://www.kdnuggets.com/polls/2014/analytics-data-mining-data-science-methodology.html*

Samuel, A. L. (1959), 'Some studies in machine learning using the game of checkers', *IBM Journal of Research and Development* **3**(3), 210–229.

Sánchez-Medina, A. J., Eleazar, C. et al. (2020), 'Using machine learning and big data for efficient forecasting of hotel booking cancellations', *International Journal of Hospitality Management* **89**, 102546.

Scheidel, C. (2018), 'Understanding bias in rf variable importance metrics'. Last visited 2021-12-29.
URL: *https://blog.methodsconsultants.com/posts/be-aware-of-bias-in-rf-variable-importance-metrics/*

SMHI (2021), 'Ladda ner meteorologiska observationer'. Last visited 2021-09-30.
URL: *https://www.smhi.se/data/meteorologi/ladda-ner-meteorologiska-observationer*

Stewart, M. (2019), 'The actual difference between statistics and machine learning'. Last visited 2021-11-15.
URL: *https://towardsdatascience.com/the-actual-difference-between-statistics-and-machine-learning-64b49f07ea3*

Uysal, M. & O'Leary, J. T. (1986), 'A canonical analysis of international tourism demand', *Annals of Tourism Research* **13**(4), 651–655.

9 Appendix

Names	Explanations	Range
Hotell	Part of the hotel.	1/0
Bokat Datum	Date for booking.	2016-01-01 - 2020-12-31
Vistelsedatum	Date of arrival.	2016-01-01 – 2020-12-31
Arrival-WeekNumb	Week of arrival.	1-52
Arrival-Year	Year of arrival.	2016-2020
Arrival-MOnth	Month of arrival.	1-12
Arrival-DayOfWeek	Day of arrival.	1-7
IndGrpBooking	Group-booking or not.	Group/No Group
Source	Source of booking.	7 categories.
Guests.CountryKorrad	Country of birth for guest.	185 countries.
EmailYN	Email exists or not.	Y/N
PhoneYN	Phonenumber exists or not.	Y/N
Guests.MailingList	Subscribed to mail-list.	Y/N
Contract	Booked via contract.	Y/N
Room-Category	Type of booked room.	22 st.
Segment	Purpose of travel/segment of company.	16 st.
InternetKampanjKod	Discount-code and what type.	0-50 %
Vuxna	Number of Adults.	0-5 (Inadequate)
Barn	Number of children.	0-4 (One obs. 77)
LeadTime	Number of days between booking and arrival.	1-726
WeekendORWeekday	Weekend or Weekday.	Weekday/Weekend
is-canceled	Cancelled or not.	1/0
Temp	Mean temperature per day.	-11.8 - 27.4
Regn	Precipitation per day.	0 - 43.7

Table 9: Summary of variables