High-Dimensional Classification Models with Applications to Email Targeting

Degree project in Engineering Physics, second level (SF299X)

Anders Pettersson
anderpet@kth.se

May 26, 2015

Supervisor: Tatjana Pavlenko
Examinator: Boualem Djehiche
Company: Spotify with supervisor Jens Larsson

Dept. of Mathematics
Royal Institute of Technology

Abstract

Email communication is valuable for any modern company, since it offers an easy mean for spreading important information or advertising new products, features or offers and much more. To be able to identify which customers that would be interested in certain information would make it possible to significantly improve a company's email communication and as such avoiding that customers start ignoring messages and creating unnecessary badwill. This thesis focuses on trying to target customers by applying statistical learning methods to historical data provided by the music streaming company Spotify.

An important aspect was the high-dimensionality of the data, creating certain demands on the applied methods. A binary classification model was created, where the target was whether a customer will open the email or not. Two approaches were used for trying to target the costumers, *logistic regression*, both with and without regularization, and *random forest classifier*, for their ability to handle the high-dimensionality of the data. Performance accuracy of the suggested models were then evaluated on both a training set and a test set using statistical validation methods, such as cross-validation, ROC curves and lift charts.

The models were studied under both large-sample and high-dimensional scenarios. The high-dimensional scenario represents when the number of observations, N, is of the same order as the number of features, p and the large sample scenario represents when $N \gg p$. Lasso-based variable selection was performed for both these scenarios, to study the informative value of the features.

This study demonstrates that it is possible to greatly improve the opening rate of emails by targeting users, even in the high dimensional scenario. The results show that increasing the amount of training data over a thousand fold will only improve the performance marginally. Rather efficient customer targeting can be achieved by using a few highly informative variables selected by the Lasso regularization.

Keywords:
Statistical learning, logistic regression, random forest classifier, customer relationship management, customer targeting.

Högdimensionella klassificeringsmetoder med tillämpning på målgruppsinrikting för e-mejl

Sammanfattning

Företag kan använda e-mejl för att på ett enkelt sätt sprida viktig information, göra reklam för nya produkter eller erbjudanden och mycket mer, men för många e-mejl kan göra att kunder slutar intressera sig för innehållet, genererar badwill och omöjliggöra framtida kommunikation. Att kunna urskilja vilka kunder som är intresserade av det specifika innehållet skulle vara en möjlighet att signifikant förbättra ett företags användning av e-mejl som kommunikationskanal. Denna studie fokuserar på att urskilja kunder med hjälp av statistisk inlärning applicerad på historisk data tillhandahållen av musikstreaming-företaget Spotify.

En binärklassificeringsmodell valdes, där responsvariabeln beskrev huruvida kunden öppnade e-mejlet eller inte. Två olika metoder användes för att försöka identifiera de kunder som troligtvis skulle öppna e-mejlen, *logistisk regression*, både med och utan regularisering, samt *random forest klassificerare*, tack vare deras förmåga att hantera högdimensionella data. Metoderna blev sedan utvärderade på både ett träningsset och ett testset, med hjälp av flera olika statistiska valideringsmetoder så som korsvalidering och ROC kurvor.

Modellerna studerades under både scenarios med stora stickprov och högdimensionella data. Där scenarion med högdimensionella data representeras av att antalet observationer, N, är av liknande storlek som antalet förklarande variabler, p, och scenarion med stora stickprov representeras av att $N \gg p$. Lasso-baserad variabelselektion utfördes för båda dessa scenarion för att studera informationsvärdet av förklaringsvariablerna.

Denna studie visar att det är möjligt att signifikant förbättra öppningsfrekvensen av e-mejl genom att selektera kunder, även när man endast använder små mängder av data. Resultaten visar att en enorm ökning i antalet tränings-observationer endast kommer förbättra modellernas förmåga att urskilja kunder marginellt.

Nyckelord:

Statistisk inlärning, logistisk regression, random forest klassificerare, kundrelationshantering, kundinriktning.

Acknowledgements

First and foremost I would like to thank my supervisors Tatjana Pavlenko and Jens Larsson and express my gratitude for all their help, interesting suggestions and constructive criticism. Their guidance and support has been of utter importance for my progress and development during this time that marks the end of my five years as a civil engineering student at KTH.

I would also like to thank Spotify for the fantastic opportunity of performing my Master's Thesis under their care in a very inspiring and developing environment, where I have gotten the chance to use and refine my skills, while at the same time learning new things. Special thanks to my fellow master's students Erik Aalto and Matteo Poletti, their supervisors Anders Arpteg and Boxun Zhang and to everyone else in the Analytics team for their support, discussions and all the fun times.

Since this might be the end of my time at KTH I would also like to thank all of my friends that I have had the luck to gotten to know over the years and I look forward to the time we will spend together in the future, may it be continuing our discussions about boxes, trees in the forest or chewing gum, or something else.

Lastly I thank my fiancée Sofie for her endless support during this time and in the future.

Contents

List of Figures

List of Tables

1 Introduction

To get an overview of the thesis the introduction first describes the problem background, section 1.1, and discusses the problem at hand, section 1.2, before moving on to defining the objective, section 1.3. Lastly the disposition of the whole thesis is presented in section 1.4.

1.1 Problem Background

Email communication is important for many companies, since it is a way to advertise new products or features, spread information, perform marketing campaigns and much more. While it can be a powerful mean of communication it can also be a double-edged sword, since too many, to the customer, irrelevant emails will generate badwill and even make future communication through email impossible. As such it can be very valuable for a company to refine their usage of email communication by carefully selecting which customers to target with specific information.

Spotify, a global music streaming company, has over 60 million active users world wide, [23], and sends a lot of emails to their users every day. The emails are of different types such as notifications, offers and newsletters. Understandably Spotify wishes to utilize their usage of this mean of communication as well as they can, which makes the aim of this thesis to further improve their email communication. This thesis will be performed by using data provided by them and it will constitute a foundation for further research.

1.2 Problem Discussion

Targeting customers is closely connected to a subject called *customer relationship management*, abbreviated *CRM*. CRM is essential for companies and in this age of *Big Data* it is possible to extract and store enormous amounts of information about each customer, which together with data mining techniques create powerful methods for improving customer relations, through for example effective marketing and targeted advertising etc.. A number of papers and books have been written solely interested in these kinds of problems. *Application of data mining techniques in customer relationship management: A literature review and classification* Ngai, Xiu and Chau, [17], presents a good overview of the different subjects within the field and classifies a number of relevant papers by their subject and which methods that were used. By trying to fit this thesis into the same framework, it will be easier to relate to earlier work within the field. In [17] they present four categories of CRM:

1. Customer Identification

2. Customer Attraction
3. Customer Retention
4. Customer Development,

which all contain some subgroups of problems. This thesis mainly falls under *customer identification* and a subgroup called *target customer analysis*. Some earlier studies in this field are *An LTV model and customer segmentation based on customer value: a case study on the wireless telecommunication industry* by Hwang, Jung and Suh, 2004 [9], *Constructing a multi-valued and multi-labeled decision tree* by Chen, Hsu and Chou, 2003 [5] and *Targeting customers via discovery knowledge for the insurance industry* C-H. Wu, Kao, Su and C-C. Wu, 2005 [26].

In *Targeting customers via discovery knowledge for the insurance industry*, [26], C-H. Wu, Kao, Su and C-C. Wu uses data mining techniques to improve the sales of an insurance company. The method used was *ID3* for its performance level and comprehensibility, which is important to be able to more generally explain which customers to target. In the study they worked closely with the company that provided the data and created general rules for which customers to target, which is very important. To only state that they should target whichever the model recommends is usually not enough, since formalized rules can be used for creating future strategies and marketing etc., an aspect that was well treated in this study.

Hwang, Jung and Suh discuss *customer lifetime value* (LTV) in [9] and how it can be used to categorise customers. They tested three different models, *decision trees*, *artificial neural networks* and *logistic regression*, which all preformed well in the given setting.

Constructing a multi-valued and multi-labeled decision tree, [5], introduces possibilities to work with multi-valued targets and multi-labeled features, in other ways than with for example *dummy variables*, since in some settings such modifications are not desirable.

These earlier studies show that a lot can be gained by applying these techniques, because they enable the company to be able to select existing or new customers which seem to have a higher interest in buying their products and target them. This relates closely to the problem at hand where we are trying to select users to email and they create a solid foundation for this thesis.

There are several different data mining methods that are applicable in these kinds of situations, for example *response modeling*, [1, pp. 96–97], or *uplift modeling*, [20], but they are suited for slightly different scenarios, depending

on the objective and the available data. In *Data mining techniques :* [sic] *for marketing, sales, and customer relationship management*, [1], Berry and Linoff provide guidelines for anyone going to conduct data mining in connection to CRM. According to them any data mining process can be separated into four points [1, p. 43]:

1. Identifying the problem
2. Transforming data into information
3. Taking action
4. Measuring the outcome

and this will act as a guideline for this work. Berry and Linoff also introduce a methodology to avoid mainly two common mistakes, categorized as [1, p. 44]:

- Learning things that are not true.
- Learning things that are true, but not useful.

Berry and Linoff consider learning things that are not true to be the most severe of the mistakes, [1, p. 44], which is very understandable, since the risks are larger compared to learning something that is true but not useful, but still none of the scenarios are desirable for a company. Learning things that are not true can be caused by several different reasons and Berry and Linoff mention mainly three causes [1, pp. 45–48]:

- Patterns may not represent any underlying rule.
- The model set may not reflect the relevant population.
- Data may be at the wrong level of detail.

Lots of examples of these scenarios are given in [1], but an example for the first point could be *overfitting*, while for the second point it could be caused by some kind of bias in the dataset and the last point could happen if you have data for each day but instead study the changes over each month.

Learning things that are true but not useful is not as dangerous as its counterpart, but still important to avoid. Berry and Linoff gives two examples of this [1, pp. 49–50]:

- Learning things that are already known.
- Learning things that can't be used.

No further discussion of these cases will be presented, but Berry and Linoff gives an important reflection that not everything that appears useless is useless [1, p. 49].

The methodology that Berry and Linoff recommend, as a mean to try to avoid these mistakes, can be summarized by eleven steps to follow when preforming a data mining task in a CRM setting [1, pp. 54–55]:

1. Translate the business problem into a data mining problem.
2. Select appropriate data.
3. Get to know the data.
4. Create a model set.
5. Fix problems with the data.
6. Transform data to bring information to the surface.
7. Build models.
8. Asses models.
9. Deploy models.
10. Assess results.
11. Begin again.

This methodology will be applied in this thesis, because it introduces a solid and well thought through framework for problems such as the one at hand. Now is a natural time to transition to defining and formalizing the objective of the thesis.

1.3 Objective

For a company the most interesting question would be which users should be targeted to maximize the company's profit, but since this study is supposed to work as a foundation for further analysis the objective has instead been chosen to try to nurse email communication. For example that the users seem to appreciate the emails sent to them. How to measure this appreciation is not entirely clear or for that matter simple, but some relevant information that should reflect users interest in and appreciation of the emails will be available. For example it will be possible to see if a user has opened the email, which shows one level of appreciation, and then the next level is whether the user clicked on a link in the email, which indicates that this user has been targeted correctly. There might even be possibilities of

a negative response, such as if the user has clicked on the unsubscribe link. The objective can be formalized as follows:

> *Objective*: to increase the positive response rate for emails by targeting users with the help of statistical learning methods and historical data.

Since large amounts of data are available, a secondary objective is to study the effects of using a smaller amount of data and compare the performance levels. This question is interesting since computations on large amounts of data in many settings take a large amount of time, which might be deemed unnecessary if there is no significant improvement compared to using a more limited dataset. This objective can be formalized as follows:

> *Secondary objective*: define and quantitatively measure differences in performance levels on the original problem when using a smaller amount of training data.

1.4 Disposition

Since the plan is to follow the methodology and the four points of a data mining process which Berry and Linoff suggest, the first part will be about formalizing the business problem as a data mining problem and introducing some notation, sections 2.1 and 2.2, and then continuing on to selecting and getting to know the data, sections 2.3 to 2.6. When the data has been selected it becomes possible to study interesting models which can fulfil the objective and discuss their appropriateness, advantages and disadvantages, section 3, and, perhaps most importantly, how to evaluate the models, section 4. The models are then tested, by first transforming the data in an appropriate fashion. In section 6 the results are discussed and the performances of the methods are evaluated. Lastly the conclusions and suggestions for future work are presented in section 7.

2 Methodology

The methodology is based on the insights from Berry and Linoff, [1], as described in the introduction. The first step is to translate the business problem into a data mining problem, which is discussed in section 2.1. Section 2.2 introduces a mathematical notation and formulation based on the earlier discussion. Secondly data is selected, studied and discussed in sections 2.3 and 2.4. When some familiarity with the data has been achieved it becomes possible to create a *model set* which can be used for the specified problem, see section 2.5. Lastly *data preprocessing* is the topic in section 2.6.

2.1 Transforming the Business Problem into a Data Mining Problem

The translation of the business problem is essential, since a bad translation will lead to bad or irrelevant results. The translation needs to be closely connected to the objective, which in this case means that there is a need to translate user appreciation of email into something that is measurable in a data mining setting. A simple initial model would be to measure appreciation by studying whether a user opens the email or not. This model can then be extended to include different levels of appreciation, for example if the user clicks a link in the email then that indicates that the user is interested in the content of the email, whereas if the email is reported as spam it indicates a negative response. This leads to the conclusion that some kind of classification model is appropriate.

There is also the question of what kinds of emails one should look at, perhaps notification emails should always be sent regardless and perhaps it is more interesting to study emails containing offers or information about new features.

Berry and Linoff ask two important questions, which makes the work relate to the specified or sometimes unspecified goal of the company [1, pp. 57–58]:

- How will the results be used?
- How will the results be delivered?

In this case the results will be used to decrease the number of irrelevant emails sent to the users, but also work as a foundation for future work and studies, such as using the data to figure out information that can be interesting for the user and much more.

It is desirable that the results can be delivered in easily interpretable rules, that can be used for directed marketing and creating business strategies. For example individuals over fifty tend to be more interested in emails then the general user. These kinds of rules are useful in a different and perhaps a more important way for a company than the information to target these customers because the model says so [1, pp. 58–60]. Berry and Linoff illustrates this with the question "who is the yoghurt lover?", referring to a supermarket setting where the task focused on finding out which customers were interested in buying yoghurt [1, p. 59]. The yoghurt lover is not someone described by a high score by the model, instead perhaps described as a middle-age woman living in the city. This information is useful to the company which can then focus on advertising in areas where there live a lot of middle-age women.

A reasonable model for achieving this is to study the opening rates of the emails, which can be formalized as a classification problem, where the response is whether an email was opened or not. With such a model it should be possible to decrease the number of emails that will not be opened and hopefully increase the number of opened emails. The model is more closely defined in the next section.

2.2 Mathematical Formulation

The mathematical formulation will be very important for transforming the business problem into a data mining problem and there are several formulations depending on what you what to measure with the model. Since a classification model has been chosen we can introduce some helpful notation. First let us denote the number of emails sent with N. Each email sent will be seen as an *observation*, note that the formalisation is not for each user but instead for each email. This can be called a *prospect*, where a prospect is represented by an unique email and a non-unique user, so there can be several emails per user. For a classification problem it is important to identify the *target variable*, denoted Y, which in this case can be expressed as a *binary variable*, $Y \in \{0, 1\}$, where $Y = 1$ represents that the email was opened and $Y = 0$ represents that the email was not opened. In the dataset, denoted D, there will be a number of positive observations and a number of negative observations, let us denote them N_P and N_N respectively. The *features* of each observation is denoted $\mathbf{X}$ and represents the traits of the user who received the email and the properties of the email, such as if it is a notification or a newsletter etc.. $\mathbf{X}$ can be written $\mathbf{X} = (X_1, X_2, \ldots, X_p)$, where p connotes the number of features and X_1 connotes the first feature and similarly for the other features. The different features can take different forms, some can be *categorical*, for example gender can be divided into "male", "female" and "other", while perhaps a feature such as the age of

the user can be *numerical* and some are binary variables. A specific set of feature values for X is represented by x and similarly for the target variable a specific value is denoted y. To summarise; the following notation has been introduced:

- D: the dataset.
- N: the number of observations in the dataset.
- Y: the target variable, binary.
- N_P: the number of positive instances in D.
- N_N: the number of negative instances in D.
- $X = (X_1, X_2, \ldots, X_p)$: the features, can be in the form of categoricals, numericals or binaries.
- p: the number of features.
- x: specific value of the features X.
- y: specific value of the target variable Y.

In the future it might be interesting to change the target variable to include more categories and as such creating a more advanced model. Such a change will greatly affect the possibilities and appropriateness of using different learning methods.

2.3 Selecting the Data

In many settings the data that can be used is scarce because of limited resources and lack of observations, but in this case a subset of the data needs to be selected because the sheer amount of raw data, a typical example of *Big Data*, at least in the sense of a large number of observations. Preferably data would be selected in such a way that it is possible to see when the email was sent and when it was opened, if it was opened, but it might be unclear how many days it is necessary to study to be sure that the email was not opened. A reasonable time frame would be a week or two. If the email has not been read during this time it probably has lost some or all of its relevance anyway. What features which could be interesting is something that the model preferably should answer on its own, as such the selection of features is done in a very including way, to be able to feed the model with as much information as possible.

Depending on how the models will be evaluated different datasets might be needed. Let us think back on the question raised in section 2.1: “how will

the results be used?", probably the results will be used to predict users to email in the future. Perhaps the model is based on currently available data, which dates back a week or two, but is supposed to be used on users today, because of this it can be interesting to have two datasets, which are based on emails sent with a few weeks apart. This is important to be able to evaluate and validate the model properly.

2.4 Feature Study

Studying the behaviour and forms of the features is important, for example to validate assumption made about the features. If a feature is supposed to represent a count of something, then it should not contain negative values for example. Usually features have some kind of description to describe what the feature represents and a incorrectly described feature, which actually represents something else, could be devastating for solving the problem.

To avoid such problems and other similar problems the features were studied in several different ways, such as looking at the distribution of the features by using histograms and investigating the range of the values a feature could take. A complete description of the features was also created to reduce the risk of misinterpreting the meanings of the features.

2.5 Creating a Model Set

Berry and Linoff describe the *model set* as the dataset that is to be used in the modeling process. The model set contains data for training, validation and evaluation and is usually formed by merging data from several sources to create *customer signatures*, a form of customer description [1, p. 68].

2.5.1 Customer Signatures

A customer signature represents all that is know about the customer and is usually represented by a row in the dataset [1, p. 68]. Creating customer signatures is an iterative processes, [1, p. 588], which is initialized by identifying what the "customer" represents, it could be anything, but in this case the simplest choice would be that a customer signature represents a prospect and the available information about the email and the user receiving the email. When the customer has been identified it becomes possible to add information about or features describing the customer, usually this process is repeated and tested, by adding and removing features or creating new features from existing ones, more on this subject in the preprocessing section 2.6.

The choice of target variable is essential for the customer signature, since depending on the target different information might be interesting to study and include in the model. In the analysis information that might have an impact on whether a user will open the email or not will be the focus when creating the customer signature.

2.5.2 Balancing the Model Set

Depending on the choice of target variable the sample of users might be heavily skewed for one type of action, for example with email it is more common for a user to ignore the email than opening it and perhaps only around 1% will actually click on the link in the email. This is a common problem within data mining since the positive examples are neglected when negative examples constitutes a massive majority, but there are a few remedies.

One alternative that has been suggested is to increase the weight on the positive examples and by that evening out the impact from positive and negative examples [1, pp. 68–69]. Another alternative is to balance the number of positive and negative examples by using a random sample from the negative examples of a comparable size to the number of positive examples, suggested by Berry and Linoff [1, pp. 68–69] and by Ling and Li [15]. In [15] Ling and Li were able to improve the performance of the model by applying the balancing method. Mainly the method of weighting the observations by their relative frequency will be used.

2.5.3 Creating a Model Set Suitable for Prediction

Preferably the model should be as stable over time as possible and this is achieved by including data over long time periods, when accessible, but for this setting where users open emails the impact of different seasons are expected to be low and will not be regarded as a factor. Instead focus will be on trying to predict users who will open emails in the future by studying how the users have been acting in the recent past.

To do this the model set will be partitioned into a *training set*, a *validation set* and a *test set*, where the training set and the validation set are from the same time frame, as a way of measuring internal validity, and where the test set is supposedly from a time frame in the future compared to the training and validation set, perhaps a week forth. This method is recommended by Berry and Linoff [1, pp. 70–72].

2.6 Data Preprocessing

Data preprocessing is very important to be able to use the data at all and of course to get good results. Fixing problems with the data, such as treating *missing values* or finding strange abnormalities is essential. Then the data can be transformed in different ways to bring information to the surface [1, pp. 74–77]. Many researchers have studied this specific question, for example Kotsiantis et al. addresses the issue of data preprocessing, in particular for *supervised learning* problems, such as the problem at hand, in [11]. The preprocessing is also dependent on the model of choice, some models can not handle certain types of data or are bad at handling such data types. For example regression models are sensitive to *multicollinearity*, where features are strongly correlated [12, pp. 29–30], and cannot handle categorical variables where there is no distance measure, while *random forest* does not suffer from these problems, but instead might have trouble with numericals and the performance can be increased by *binning* the numerical variables [1, p. 551].

2.6.1 Treating Features with Low Variance

In some datasets it is possible to find features which only contain one unique value, these features are as such useless for data mining, since they contain no information which can be used to distinguish the target variable [1, p. 544]. Such features were removed from the dataset.

Another case is when a feature contains almost only one value and then it is not as clear how the feature should be handled. A general rule of thumb is that if 95% to 99% of the values are the same the feature by itself is negligible, but it might be able to provide useful information in combination with other features [1, p. 546]. Whether the feature contains useful information or not is highly dependent on what the features represents and therefore it is important to carefully study and understand such features before dismissing them. Berry and Linoff discuss this in greater detail [1, pp. 544–546].

2.6.2 Missing Values

Missing values can for some features be a problem, for example in numericals such as 'age', but it is not really a problem for categorical features since a missing value can be seen as a category itself and provide some information about the user.

For numericals the missing values were substituted by the median of the numerical values, which is an easy correction but not without its flaws, since it changes the distribution of the feature. Another possible method

would be to use the other features and compare to the observations without missing values to try to model the missing values and replace them with the predicted values of the mode, but this method also has its problems [1, p. 591].

2.6.3 Dummy Variables

For some methods categorical features are problematic and a simple solution to this is to use dummy variables, where categorical features are split into several binary variables, one for each category. It is important to remove at least one of the dummies for one of the categories, otherwise it might cause problems depending on the model, and then the category represented by the removed dummy acts as a benchmark [10, pp. 85–86]. It is also possible to remove several dummies representing categories with few instances, since they have a low information value in accordance with the discussion above in section 2.6.1.

2.6.4 Normalization

It is common to normalize the numericals so they range from zero to one, which preserves the distribution but changes the values of the features. This affects for example logistic regression, and might improve the results, but not random forest, since for random forest only the order is considered, which is unchanged by normalization [1, p. 550]. An alternative would be standardization, where the numbers are transformed into the number of standard deviations from the mean [1, p. 551]. For the following analysis normalization will be used.

2.6.5 Binning

Some models have problems with handling numerical features and the values of such features can instead be grouped into buckets or bins that represents a range of values and as such converting the numerical feature into a categorical one. There are two types of binning equal-width and equal-height binning, where equal-width binning transforms the values into ranges with a fixed width. Equal-height binning on the other hand bins the values in such a way that each bin has the same height and as such creating a uniformly distributed variable [1, p. 551].

2.6.6 Creating New Features

It can also be interesting to create new features from the existing ones, perhaps to find non-linear relations or interactions terms. This can be done in several ways, but for numericals it is common to create ratios or proportions of different kinds. For example for houses it might be interesting to have a

feature which represents the price per square meter instead of just the price of the house and the house area [1, p. 569]. It is also possible to use logical operations to create new features [11].

Creating new features can be seen a method to bring information to the surface, that otherwise would be hidden [1, pp. 74–76]. For example a feature that may contain the date of birth of a person is in it original form quite useless, since there will be almost unique values for each observation, but if combined with the date of today you can create a new feature representing the age of the costumer, which can be very valuable for the analysis.

2.6.7 Stratified Sampling

Stratified sampling is a sampling method for picking out a sub-sample from a population with the same internal properties between mutually exclusive *strata* [21, p. 100], or in this setting classes. It will be used for studying the question of whether using a large amount of data will have a significant impact on the performance level. If we denote the whole population $U = \{1, \ldots, i, \ldots, N\}$ it can be divided into H sub-populations or strata, depending on the properties of each element. Let us denote them $U_1, \ldots, U_h, \ldots, U_H$, where $U_h = \{i : i \text{ belongs to stratum } h\}$ in accordance with the notation in [21, p. 101]. In the case of using a binary classifier the population can be split into two strata, one for the positive class and one for the negative class. Stratified sampling is done by selecting a sample s_h from U_h with the help of some kind of sampling design, where the selection in each stratum is independent of each other [21, p. 101]. The final sub-sample, denoted s, is created by the union of the individual samples from each strata [21, p. 101]:

$$s = s_1 \cup s_2 \cup \cdots \cup s_H \tag{1}$$

The sampling in each strata is done with *proportional allocation* as described in [21, p. 107]. Let us denote the sizes of the strata with $N_1, \ldots, N_h, \ldots, N_H$ and the expected sizes of the specific samples from the strata with $n_1, \ldots, n_h, \ldots, n_H$. The proportional allocation can then be defined as [21, p. 107]:

$$n_h = n\frac{N_h}{N} \tag{2}$$

where n is defined as the total sample size, expressed by [21, p. 106]:

$$n = \sum_{h=1}^{H} n_h. \tag{3}$$

The proportional allocation strategy is optimal in the cases where the standard deviation in each separate stratum is the same [21, p. 107], which is the case for a setting with strata created from binary classes. In each strata all observations are sampled uniformly.

3 Choice of Statistical Learning Methods

Two different methods were chosen for testing: logistic regression and random forest classifier and they will be discussed in greater detail below.

3.1 Logistic Regression

Logistic regression, abbreviated LR, is a widely used binary classification method and is a kind of substitution for *linear regression*, since linear regression is not suited for classification [10, pp. 129–130]. First logistic regression is described in greater detail in section 3.1.1 and then some of the advantages and disadvantages with the method are discussed, section 3.1.2.

3.1.1 Description

Logistic regression models the probability that an observation belongs to a certain class, given its features, $P(Y|X)$. This probability can be seen as a *score* on the observation. A high probability represents a high score and a low probability represents a low score. In the case of a binary classifier, $y \in \{0, 1\}$, the probability is modelled by a *sigmoid function* [27]:

$$P(Y = 1|X = x, w) = \frac{1}{1 + e^{-w^T x}} \tag{4}$$

for the positive class and

$$P(Y = 0|X = x, w) = \frac{1}{1 + e^{w^T x}} \tag{5}$$

for the negative class, where w is a vector of *weights* for each variable in X and an *intercept*. Otherwise the notation is the one introduced in section 2.2, x represents a set of values of the features. An observation with $P(Y = 1|x, w) > t$ is classified as belonging to class 1 for a given threshold $t \in [0, 1]$, commonly $t = 0.5$. Figure 1 displays a binary classification example, with a one-dimensional feature space, where the logistic regression model has been fitted to the available data. In the figure it is possible to observe the typical sigmoid shape of the probability $P(Y = 1|X = x, w)$ and see that for negative X far to the left $P(Y = 1|X = x, w)$ is close to zero, since the negative class is clearly dominating the area, while it is the opposite far to the right. In the middle, where the observations from the different classes are both present $P(Y = 1|X = x, w)$ increases quite rapidly.

To find the weights w it is common to use the approach *maximum likelihood estimation*. Given our binary classed data $\{x_i, y_i\}_{i=1}^N,\ x_i \in \mathbb{R}^p,\ y_i \in \{0, 1\}$ the *likelihood*, $L(w)$, is defined as:

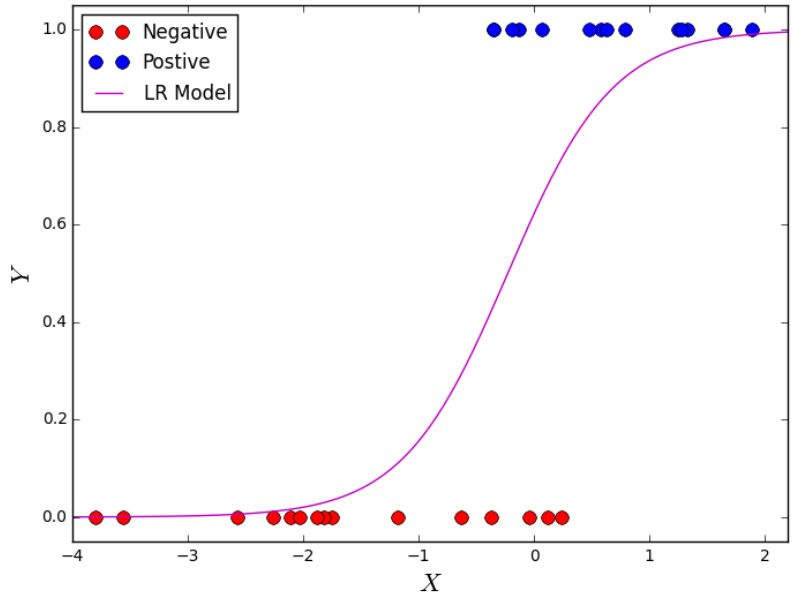

Figure 1: A one-dimensional binary classification example with randomly generated data. The blue circles and the red circles represents the observations belonging to the positive class and the negative class respectively. The magenta coloured line represents the probability of belonging to the positive class given by the logistic regression model, $P(Y = 1|X = x, w)$.

$$L(w) = \prod_{i=1}^{N} P(Y = y_i|X = x_i, w). \tag{6}$$

Finding the maximum of the likelihood is the same as finding the minimum of the negative log-likelihood [10, p. 133], since the logarithm is a strictly increasing function. The *negative log likelihood* is defined by [10, p. 133]:

$$l(w) = -\sum_{i=1}^{N} \log P(Y = y_i|X = x_i, w). \tag{7}$$

Since finding the optimum of equation 7 leads to non-linear equations, this is usually done with the help of iterative methods [8, p. 33]. The Python library scikit-learn, [18], uses *stochastic gradient descent*, or *SGD*, for solving this optimization problem, as described in section 3.1.3.

For logistic regression it is also common to study the log-odds or logit, which is linearly described by the features. For convenience we will denote

$P(Y = 1|x, w)$ as $\pi(x)$, we omit the w. Then by rearranging the terms in equation (4) we can express:

$$\frac{\pi(x)}{1 - \pi(x)} = e^{w^T x}. \tag{8}$$

The logit is described by the logarithm of this expression [10, p. 132], which gives:

$$\log\left(\frac{\pi(x)}{1 - \pi(x)}\right) = w^T x. \tag{9}$$

From this expression it is clear that the logit is linearly described by the features. This means that changing a feature in x by one unit will increase the logit by the appropriate weight. It is also clear that the logit can take any value in $\mathbb{R}$, depending on the range of $w^T x$ [8, p. 6].

It is easy to extend logistic regression to be able to find non-linear decision boundaries, instead of just linear decision boundaries, which is normally the case. This is done by adding polynomial terms of the features to the model [10, p.184–185]. An example of this for a single predictor, i.e. $p = 1$, would be:

$$\log\left(\frac{P(Y = 1|X = x_1, w)}{1 - P(Y = 1|X = x_1, w)}\right) = w_0 + w_1 x_1 + w_2 x_1^2 + w_3 x_1^3 \tag{10}$$

which leads to a cubic logistic regression model, which can give a decision boundary formed as a third degree polynomial.

Logistic regression does not make the same basic assumptions as linear regression, such as *linearity* or *homoscedasticity*. One of the assumptions is the binomial assumption, which is an assumption on the distribution of the *error*, denoted ε [19]. The error term comes from the following expression of the target variable:

$$y = \pi(x) + \varepsilon. \tag{11}$$

Since $y \in \{0, 1\}$ is binary ε is also limited to only two possible values, $\varepsilon = 1 - \pi(x)$ if $y = 1$ with probability $\pi(x)$ or $\varepsilon = -\pi(x)$ if $y = 0$ with probability $1 - \pi(x)$ [8, p. 7]. This gives that the mean of the error is 0 and that the variance is $\pi(x)(1 - \pi(x))$, which gives a binomial distribution with $\pi(x)$ as the success probability [8, p. 7]. The binomial assumption is the only assumption that should be verified according to Peng et al. in [19]. A method for testing this assumption is a normal *Z-test*, but the assumption is usually robust if the observations in the dataset are independent [19].

For logistic regression different options of *regularization* are available and are usually used when the number of features p is comparable to the sample size N and is especially common in machine learning [14]. Regularization is a mean to push coefficients towards zero, which can both reduce over fitting and improve the performance of the model, by reducing the variance [10, pp. 214–215] [14]. It can also be used for variable selection [10, pp. 214–215]. With regularization an extra term is added to the negative log-likelihood and for the L_2-norm the negative log-likelihood is given by [27]:

$$l^{L_2}(w) = -C \sum_{i=1}^{N} \log P(Y = y_i | X = x_i, w) + \frac{1}{2} w^T w \tag{12}$$

where $C > 0$ is a *penalty parameter*. For the L_1-norm the formula is changed to:

$$l^{L_1}(w) = -C \sum_{i=1}^{N} \log P(Y = y_i | X = x_i, w) + \frac{1}{2} \sum_{j=1}^{p} |w_j| \tag{13}$$

Decreasing C puts more weight on the regularization term, resulting in a larger penalization on the weights, while increasing C reduces the impact of the regularization term. These different variants of regularization have different properties and are usually referred to as "ridge regression" for the L_2-norm, [13] , and "the lasso" for the L_1-norm [14]. This is the formed used in the Python library scikit-learn [18].

Equivalently equations (12) and (13) can be expressed on more well known forms, only changed by a positive constant and as such with the same minimum. The negative log-likelihood then takes the following forms, as seen in [13] for the ridge regularization:

$$l^{L_2}(w) = -\sum_{i=1}^{N} \log P(Y = y_i | X = x_i, w) + \lambda w^T w \tag{14}$$

and in [14] for the lasso:

$$l^{L_1}(w) = -\sum_{i=1}^{N} \log P(Y = y_i | X = x_i, w) + \lambda \sum_{j=1}^{p} |w_j| \tag{15}$$

where $\lambda > 0$ is also a positive penalty parameter, connected to C as $\lambda = \frac{1}{2C}$. Equations (15) and (14) can actually be seen as the *Lagrangian* of the following optimization problem [14]:

$$\min_{w} -\sum_{i=1}^{N} \log P(Y = y_i | X = x_i, w)$$
$$\text{subject to } \|w\|_j \leq K$$

where $j = \{1, 2\}$ represents the respective norm and K is a constant, connected to λ so that for each λ there is a K for which the optimization problem and the original problem give the same solution.

Figure 2 shows the connection between the weights for the different types of norms, were it can be seen that for the L_1-norm the weights are located on a diamond, while they are located on a circle, or sphere in higher dimensions, for the L_2-norm. The geometric figures can also be seen as constraint regions for the resulting optimization problems for the different norms, as seen in [10, pp. 221–223]. In figure 6.7 in [10, p. 222] there is a geometric explanation to why the lasso results in variable selection and it is because the solutions that are usually found are the edges on the diamond, where some feature weights are zero.

The choice of the regularization parameter is an interesting question and is highly dependent on the available data. One method is by empirically study the effects of changing the parameter and especially in this case the *area under the ROC curve* over *k-fold cross validation* will be used as a measure for determining the optimal regularization parameter. This is because the area under the ROC curve have specific traits that are valuable in this setting, more on that in section 4.2.

3.1.2 Advantages and Disadvantages

Traditionally logistic regression has been widely used and studied and is as such a well known and well understood method, making it a natural choice for testing and comparison.

Logistic regression is very convenient to use for a company since it directly models the probability for the observation to belong to a certain class, $P(Y|X, w)$, a score that can be used for making marketing decisions. It also produces an indication on how important the features are, a significant weight indicates that the impact of the specific feature is non-negligible, while a non-significant weight represents a small or negligible impact.

Regularization can often improve the performance of the model, but it comes at an computational cost and should as such be motivated before usage. One difference between the regularizations is that the ridge, L_2, regularization

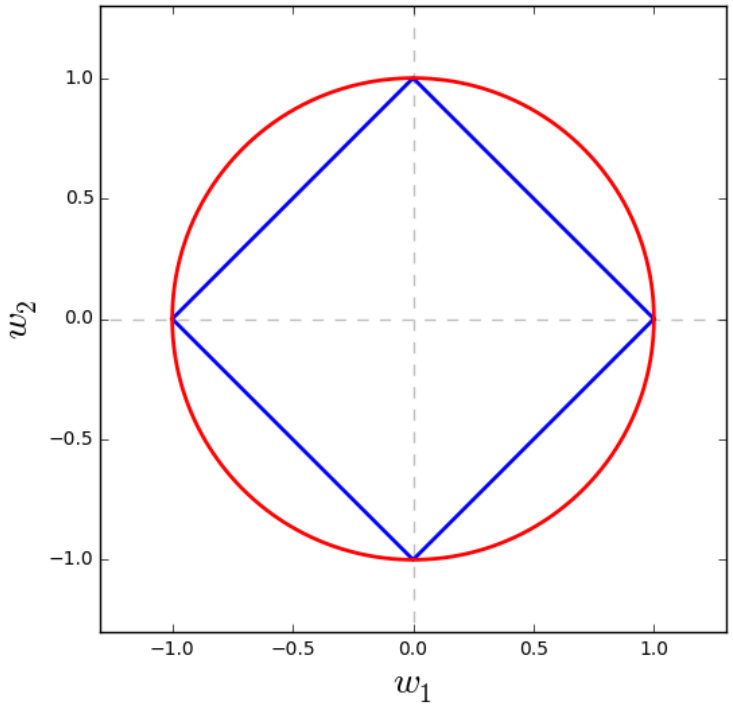

Figure 2: Image displaying the relation between the weights in 2 dimensions for the L_1-norm, shown in blue, and the L_2-norm, shown in red, when the sum of the weights adds to 1.

pushes small weights closer to 0, but does not put them to exactly 0 [10, pp. 215–216], while the lasso, L_1, regularization sets feature weights to 0, which as mentioned before is called feature selection or variable selection [10, pp. 219–220], [14]. Variable selection can also be used for data processing and to decrease the number of features, which is especially important in settings where N and p are of comparable sizes. This could be a strong reason to use regularization.

Whether the lasso or ridge regression will perform the best is hard to know beforehand. They have similar qualities, but are specialized for different settings. For example in a scenario where the target is dependent, perhaps to a small extent, on all features lasso will incorrectly discard some features while ridge will not. Similarly for a scenario where the target is not dependent on certain features, ridge regression will not be able to ignore these features entirely, as discussed in [10, pp. 223–224].

In *Machine Learning: A Probabilistic Perspective*, [16], Kevin P. Murphy recommends L_2-regularization for logistic regression even when large amounts of data are available [16, p. 254]. For example if the data is linearly separable the maximum likelihood estimation, in the case of logistic regression without regularization, will result in that $\|w\| \to \infty$, which corresponds to an infinitely steep sigmoid function [16, pp. 254-255]. Such a solution puts the maximal amount of probability mass on the training data, which results in that the solution probably will not generalize well [16, p. 255]. Murphy recommends L_2-regularization as a remedy to this problem [16, p. 255].

For logistic regression to be able to handle categorical variables it is necessary that there is some kind of distance measure between the categories, if that is not the case it becomes necessary to somehow transform the categorical variables into a form that can be handled by the regression. This can be done with the help of dummy variables, which can greatly increase the number of features. Compared to decision trees which can handle categorical variables without such transformations and as such preserve the number of necessary features. An increase in the number of features can cause problems especially for smaller sample sizes.

In the case of classification problems with more than two response classes logistic regression is not as widely used [10, p. 137]. Which makes it less of a natural choice for comparison in such a setting.

Because of these features and flaws with logistic regression the different regularizations and different choices the regularization parameters will be studied and compared to normal logistic regression without regularization.

3.1.3 Stochastic Gradient Descent

The following description of SGD is based on Léon Bottou's work in *Large-scale machine learning with stochastic gradient descent*, [3]. Let us first introduce some notation. Let z connote an observation with features and the target variable (x, y), note that y is not necessary representing classes any more, but can represents the target in a regression model. We then define a *loss function* $l(\hat{y}, y)$, where $\hat{y}$ represents the predicted y value. Note that l should not be confused with negative log-likelihood defined above, but the negative log-likelihood is an example of the loss function for logistic regression. $l(\hat{y}, y)$ measures the cost of predicting $\hat{y}$ when the correct value is y. Then we limit ourselves to a family $\mathcal{F}$ of functions that are parametrized by a weight vector w, $f_w(x)$. The sigmoid function in equation (4) is an example of such a family of functions. The objective is to find the function $f \in \mathcal{F}$ that minimizes the average loss, over the examples, $Q(z, w) = l(f_w(x), y)$ [3]. Given a set of observations $\{z_1, z_2, \ldots, z_N\}$ the *empirical risk* can be

expressed:

$$E_n(f) = \frac{1}{N}\sum_{i=1}^{N} l(f_w(x_i), y_i). \tag{16}$$

Usually one has to settle for the empirical risk since the *expected risk*:

$$E(f) = \int l(f(x), y)\ dP(z) \tag{17}$$

which would be desirable to calculate, is not available because of the unknown distribution $dP(z)$, which represents the true underlying distribution. According to Bottou the minimization of the empirical risk instead of the expected risk is justified by the results in [24] as long as the chosen family of functions $\mathcal{F}$ is sufficiently restrictive [3].

Regular *gradient decent*, or *GD*, is an alternative for minimizing the empirical risk, where the weights w are updated iteratively with the help of the gradient of $E_n(f)$ [3]:

$$w_{t+1} = w_t - \gamma \frac{1}{N}\sum_{i=1}^{N} \nabla_w Q(z_i, w_t) \tag{18}$$

where γ is a *gain* which has to be chosen carefully and $\nabla_w = [\frac{d}{dw_1}, \ldots, \frac{d}{dw_p}]$. GD achieves *linear convergence* under sufficient assumptions of regularity and if the initial estimate w_0 is not too far away from the optimum and the gain γ is small enough [3].

SGD is a simplification of GD, where instead of evaluating the gradient of $E_n(f)$ one estimates the gradient with the help of a single randomly selected observation z_t [3]:

$$w_{t+1} = w_t - \gamma_t \nabla_w Q(z_t, w_t). \tag{19}$$

This leads to a stochastic process $\{w_t, t = 1, \ldots\}$, which depends on the randomly selected observations and the idea is that equation (19) will behave as equation (18), since (18) is the expectation of (19).

Convergence for SGD is usually achieved under certain conditions of the decreasing gains γ_t. The conditions are $\sum_t \gamma_t^2 < \infty$ and $\sum_t \gamma_t = \infty$ [3].

SGD is especially powerful when performing calculations on large amounts of data because of its efficiency, both in performance and computational time [3]. A disadvantage is the gain parameter γ_t that might need to be tuned to achieve quick convergence.

3.2 Random Forest Classifier

The random forest classifier is based on the basic decision tree and was introduced by Leo Breiman [4] and has since then been used in many different applications. As before the method is first described more closely and then some pros and cons are discussed.

3.2.1 Description

Random forest, abbreviated RF, is a form of bagged decision trees, but the trees have been slightly decorrelated [10, p. 320]. Random forest can be used for both regression and classification, but we are mainly interested in the random forest classifier. Let us start by describing the decision tree.

Decision trees are widely used for both classification and prediction and one major reason is its easy interpretation, since splits in decision trees can be represented as rules [1, p. 165]. Berry and Linoff gives an intuitive description of the decision tree by referring to the famous Swedish botanist and zoologist Carl Linnaeus:

> "A decision tree is a structure that can be used to divide up a large collection of records into successively smaller sets of records by applying a sequence of simple decision rules. With each successive division, the members of the resulting sets become more and more similar to one another. The familiar division of living things into kingdoms, phyla, classes, orders, families, genera, and species, invented by the Swedish botanist Carl Linnaeus in the 1730s, provides a good example. Within the animal kingdom, a particular animal is assigned to the phylum chordata if it has a spinal cord. Additional characteristics are used to further subdivide the chordates into the birds, mammals, reptiles, and so on. These classes are further subdivided until, at the lowest level in the taxonomy, members of the same species are not only morphologically similar, they are capable of breeding and producing fertile offspring." [1, p. 166]

This is the essential idea of a decision tree, to divide the groups into smaller subgroups by studying features of the group. Decision trees are as such non-parametric and make no assumption on the distribution of the data [7]. Figure 3 gives an example of a decision tree for whether a person should go out and play tennis or not.

When choosing which feature to study different criteria can be used, such as the *Gini index* or the *cross-entropy*. Both give a measure on the quality of the split [10, p. 312]. The Gini index is defined by the equation:

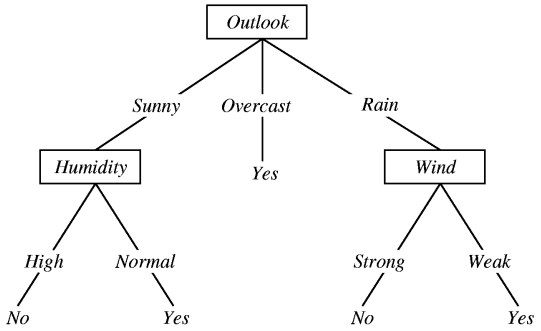

Figure 3: An example of a decision tree of whether to play tennis or not. The image and example was taken from [22].

$$G = \sum_{k=1}^{K} \hat{p}_{mk}(1 - \hat{p}_{mk}) \tag{20}$$

where K is the number of classes in the target variable and $\hat{p}_{mk}$ is the proportion of training observations from the class k in the mth set [10, p. 312]. Similarly the cross-entropy is expressed by:

$$D = -\sum_{k=1}^{K} \hat{p}_{mk} \log \hat{p}_{mk} \tag{21}$$

The tree is then built by splitting until further splitting is not possible, allowed or necessary and the remaining subsets are called leafs. The observation is classified as the majority class in the specific leaf it falls in. There are several possible specifications for stopping that can be made to the tree, for example maximum depth or a lower bound on the number of observations in a branch [1, p. 175]. After the tree has been built it is possible to prune the tree. Pruning is a method for stabilizing the tree, making it less vulnerable to noise in the training data [1, p. 184]. By pruning the tree leaves and other nodes are combined to a single leaf, usually done in such a way that the classification error is minimized on a validation set [1, pp. 189–190]. There are several different methods for pruning a tree, for example C5 pruning, pessimistic pruning and stability-based pruning [1, pp. 190–191], but these will not be discussed further.

Decision trees can be tinkered with in many ways to improve performance, but let us focus on when you combine several decision trees into a classifier or more specifically the random forest classifier. Random forest is different from the ensemble method of bagged decision trees, since random forest introduces an improvement that decorrelates the trees [10, p. 320], which in turn raises the variance. This decorrelation is done by limiting the number

of features each decision tree in the random forest is allowed to consider, meaning that each decision tree can only split on a random subset of features, but as with bagging the tree is trained on *bootstrapped* samples from the training data [10, p. 320].

As with logistic regression it is desirable to be able to rank or score each observation with the probability of that observation belonging to a certain class. For the random forest this probability will be estimated by the mean of the probability estimations of each tree in the forest and were each tree estimates the probability by taking the fraction of the samples of the same class of the observation in the particular leaf that the observation ends up in. This is called *relative class frequency* and is suggested by Boström in [2], were this method for estimating the probabilities clearly out preforms other alternatives, such as *average vote*, *Laplace estimate* and *the m-estimate*, on most of the studied datasets in terms of *accuracy*, *area under the ROC curve* and *Brier score*.

3.2.2 Advantages and Disadvantages

Decision trees in themselves have some desired properties as described in [10, p. 315], for example being non-parametric, how easy it is to explain how decision trees works and to display and interpret the results. It is also possible to then translate these results into rules [1, p. 165], which, as discussed before, can be very useful for a company. Decision trees are also said to mimic human decision-making to a greater extent than most algorithms [10, p. 315]. Logistic regression suffered from the problem of being unable to handle categorical variable without distance measures, decision trees are instead perfect for handling such variables, which can be done without the need of dummy variables, thus preserving the number of features [10, p. 315].

A problem with a single decision tree is that in many settings it is unable to perform well and random forest is a method to improve the accuracy [10, p. 316], but some of the advantages with the decision tree decreases, for example the interpretability of the result and the ability to create generalized rules.

In Breiman's original paper he introduces several strengths with random forest, such as accuracy comparable and sometimes better than *Adaboost*, which at the time was not generally achieved by similar decision tree ensemble methods [4]. Outliers and noise are problematic for some methods, but random forest is comparably robust and good at handling such problems [4]. Compared to bagging or boosting random forest is faster, while it is also easy to parallelize [4].

4 Evaluation Theory

Evaluating the models is essential and there are several tools available depending on the setting of the problem. This section discusses some methods for measuring the appropriateness of the models. First let us introduce some necessary additional notation for when a model is applied on a dataset:

- TP: the number of *true positives*, given by the model.
- FP: the number of *false positives*, given by the model.
- TN: the number of *true negatives*, given by the model.
- FN: the number of *false negatives*, given by the model.

FP means that the observation is classified as positive, but it is actually negative and vice versa for FN. By the definition of N_P and N_N from section 2.2 the following equations can be deduced:

$$TP + FN = N_P \tag{22}$$

and

$$TN + FP = N_N. \tag{23}$$

With this notation it becomes possible to introduce some measures for evaluating the models.

4.1 Precision, Recall and Accuracy

Precision and accuracy are often used to measure the performance of binary classifiers and are defined as follows [25]:

$$\text{Precision} = \frac{TP}{TP + FP} \tag{24}$$

and

$$\text{Accuracy} = \frac{TP + TN}{N_P + N_N} \tag{25}$$

Precision represents the ratio between the number of true positives and the total number of predicted positives and measures how well the model is able to distinguish between positive and negative examples. Accuracy measures the models ability to correctly identify the examples. A third measure is *recall* and it is defined by:

$$\text{Recall} = \frac{TP}{N_P} \tag{26}$$

for the positive class and for the negative class just replace TP with TN and N_P with N_N. The recall of a model is connected to the precision of the model. A high recall but low precision is not an indication of a successful model, but instead of a model that predicts that most observations are positive. The reversed with high precision and low recall is probably not desirable either, since then model is able to accurately classify some positive examples but misses a large portion of all the existing positive examples in the dataset. Instead a well performing model is usually characterised by both high precision and high recall.

4.2 ROC Curves

ROC, *Receiver Operating Characteristics*, *curves* have been used for evaluating models for many decades [1, p. 98]. First let us assume that a model has scored all of the test observations with a probability $P(Y = 1|X = x)$ and that if $P(Y = 1|X) > t$ for an observation then the observation is classified as positive, where t is the classification threshold of the model. Let us also denote the predicted class $\hat{Y}$ and define it as $\hat{Y} = \mathbb{1}_{\{P(Y=1|X)>t\}}$, where $\mathbb{1}$ is the indicator function. The ROC curve is defined by a parametric definition of the *false positive rate* and *true positive rate*:

$$FP\text{-rate}(t) = \frac{FP(t)}{N_N} = \frac{\sum_{j=1}^{N_N} \mathbb{1}_{\{P(Y=1|X=x_j)>t\}}}{N_N}, \tag{27}$$

where the sum is over the negative observations and

$$TP\text{-rate}(t) = \frac{TP(t)}{N_P} = \frac{\sum_{i=1}^{N_P} \mathbb{1}_{\{P(Y=1|X=x_i)>t\}}}{N_P}, \tag{28}$$

where similarly the sum is over the positive observations in the dataset. The false positive rate and the true positive rate have probabilistic interpretations that can be expressed as $FP\text{-rate}(t) = P_t(\hat{Y} = 1|Y = 0)$ and the $TP\text{-rate}(t) = P_t(\hat{Y} = 1|Y = 1)$ [6, p. 8]. This means for example in the case of the TP-rate the probability that a positive example is classified as positive by the model.

The x-axis of the ROC curve is defined by the FP-rate [25]:

$$x\text{-axis} = FP\text{-rate}(t) \tag{29}$$

and the y-axis by the TP-rate [25]:

$$y\text{-axis} = TP\text{-rate}(t). \tag{30}$$

See figure 4 for a typical example of a ROC curve. In the figure the origin, the point $(0, 0)$, represents the threshold where no observations are classified as positive and as such the false positive rate and the true positive rate are both zero. Then when t is decreased the curve takes shape and at the point $(1, 1)$ all observations are classified as positive and as such both the false positive rate and the true positive rate are 100%. A feature of the ROC curve is that it is independent of the ratio between N_P and N_N, which is usually desirable when the ratio varies [25].

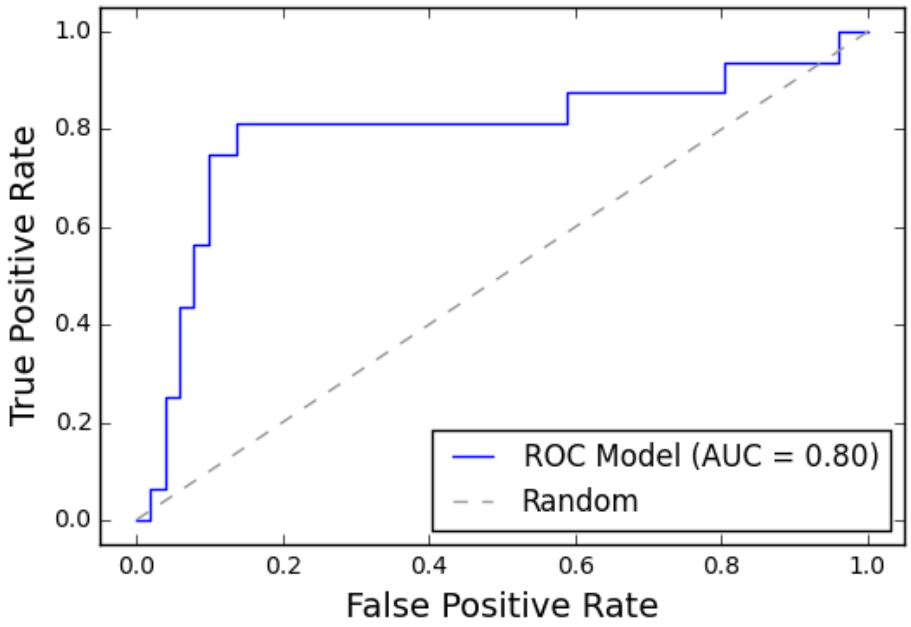

Figure 4: An example of a ROC curve, where the blue line indicates the performance of the model and the dotted grey line represents a random classifier. The area under the ROC curve, AUC, is given in the legend.

The ROC curve is useful by itself but it is also interesting to measure the area under the ROC curve or AUC. The AUC is defined by [25]:

$$\text{AUC} = \int_0^1 \frac{TP}{N_P}\, d\frac{FP}{N_N} = \frac{1}{N_P N_N} \int_0^{N_N} TP\; dFP. \tag{31}$$

In words equation (31) gives a method for calculating the area under the ROC curve, where the interpretation of $d\frac{FP}{N_N}$ is a small step on the x-axis. By inserting equations (27) and (28) we obtain:

$$\text{AUC} = \int_0^1 \frac{\sum_{i=1}^{N_P} \mathbb{1}_{\{P(Y=1|X=x_i)>t\}}}{N_P} \, d\left(\frac{\sum_{j=1}^{N_N} \mathbb{1}_{\{P(Y=1|X=x_j)>t\}}}{N_N} \right) =$$

$$= \left\{ \frac{d\left(\frac{1}{N_N} \sum_{j=1}^{N_N} \mathbb{1}_{\{P(Y=1|X=x_j)>t\}} \right)}{dt} = \frac{1}{N_N} \sum_{j=1}^{N_N} \delta_{\{P(Y=1|X=x_j)=t\}} \right\} =$$

$$= \frac{1}{N_P N_N} \int_0^1 \sum_{i=1}^{N_P} \mathbb{1}_{\{P(Y=1|X=x_i)>t\}} \sum_{j=1}^{N_N} \delta_{\{P(Y=1|X=x_j)=t\}} \, dt = \{unique$$

$$probability \ scores\} = \frac{1}{N_P N_N} \sum_{j=1}^{N_N} \sum_{i=1}^{N_P} \mathbb{1}_{\{P(Y=1|X=x_i)>P(Y=1|X=x_j)\}},$$

where δ is the delta function. As described by Vuk and Curk in [25] the probability that a random positive observation in the test set, Z_P, has a higher assigned score by the model $\mathcal{M}$ than a random negative example, Z_N, is calculated by, if all the scores are unique, for each negative example calculating the number of positive examples with a higher score, sum it together and divide by $N_P N_N$ or:

$$P(Z_P^{score} > Z_N^{score} \mid \mathcal{M}) = \frac{1}{N_P N_N} \sum_{j=1}^{N_N} \sum_{i=1}^{N_P} \mathbb{1}_{\{P(Y=1|X=x_i)>P(Y=1|X=x_j)\}}, \tag{32}$$

where the *score* superscript represents the score of the observation given by the model. By comparison we see that:

$$\text{AUC} = P(Z_P^{score} > Z_N^{score} \mid \mathcal{M}). \tag{33}$$

For a perfect classifier this probability is equal to 1 and for the worst kind of classifier, which is just randomly guessing, this probability is equal to 0.5 [1, p. 99].

If the scores are not unique, meaning that negative and positive observations both have the same score, equation (33) still holds true if the points with equal score are connected by the lower sides of a right-angled triangle [25]. There are alternatives to this method, such as if several observations have the same score you include all of the observations and move diagonally in the diagram, this method will be used when necessary. With this approach the AUC gets a new interpretation which is very similar to the old interpretation [25]:

$$\text{AUC} = P(Z_P^{score} > Z_N^{score} \mid M) + \frac{1}{2}P(Z_P^{score} = Z_N^{score} \mid \mathcal{M}) \tag{34}$$

The only modification is a term for describing the case where a random positive example and a random negative example have the same score given by the model.

From the ROC curve it is possible to find the classification threshold t which maximizes the accuracy of the classifier. From equation (25) we obtain:

$$\text{Accuracy} = \frac{TP(t) + TN(t)}{N_P + N_N} = \frac{TP\text{-rate}(t)N_P + (1 - FP\text{-rate}(t))N_N}{N_P + N_N}, \tag{35}$$

which means that the threshold that maximizes the accuracy is easily obtained from the ROC curve. This threshold is then useful when studying the precision and the recall of the classifier.

4.3 Lift Charts and Cumulative Captured Response

Another common evaluation tool is lift charts, which are closely connected to the *cumulative captured response*, C_R, [1, pp. 81–84]. Captured response is the portion of positive examples that you would reach out to, y-axis, if you only contact a portion of the total population, x-axis, if the observations have been ordered according to the scores given by the model [1, p. 82], see the left plot in figure 5. Let us define the number of positive examples in the sub sample as $\tilde{N}_P$, then the captured response can be expressed as $C_R = \frac{\tilde{N}_P}{N_P}$, and the proportion contacted as P_C. *Lift*, L, is defined as the portion of positive examples in the sample divided by the overall response rate, so if a sample contains 30% positive examples and the dataset contains around 5% positive examples overall then the lift is equal to 6, $30/5 = 6$ [1, p. 82]. If we denote the size of the sub sample as $\tilde{N}$ the lift can be expressed by the following equation:

$$L = \frac{\frac{\tilde{N}_P}{\tilde{N}}}{\frac{N_P}{N_P+N_N}} \tag{36}$$

The connection to the captured response curve becomes more apparent if the terms are rearranged:

$$L = \frac{\frac{\tilde{N}_P}{N_P}}{\frac{\tilde{N}}{N_P+N_N}} = \frac{C_R}{P_C} \tag{37}$$

The lift is as such the division of the y-values and x-values on the captured response curve. See the right plot in figure 5 for a typical example of a lift

chart.

The lift chart is a bit problematic, for example even if there is a huge lift it might not mean that the model is desirable to use, perhaps if the sample where the lift is large is so small that it is not usable from a marketing perspective [1, p. 82]. A lift chart can not evaluate the usefulness of the model by itself, but it is useful for comparing different models to each other [1, p. 83]. The captured response curve is perhaps more useful and most importantly easy to understand and interpretate, since it is easy to for example see how many of the positive examples you reach out to if you only contact 50% of the total population.

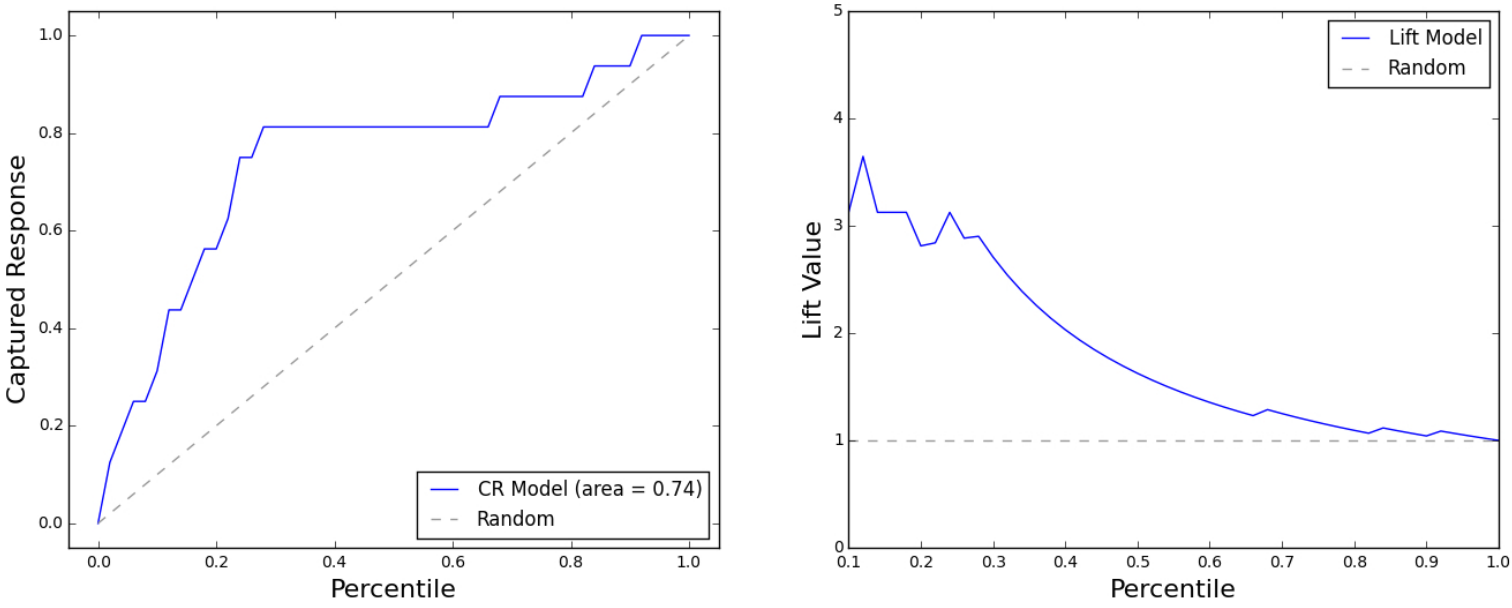

Figure 5: `Left:` Captured response plot, the blue line represents the performance of the model and the dotted grey line represents a random classifier. `Right:` The lift chart given by the captured response curve to the left.

4.4 k-Fold Cross-Validation

Cross-validation can be used for testing the performance of most statistical learning methods and is as such often used in machine learning. In k-fold cross-validation the dataset is randomly divided, with the help of stratified sampling, into k roughly equal groups or folds and then the first fold is put away as a validation set and the rest of the folds, $k-1$ of them, are used for training. This is repeated k times in total, such that each fold is left out once, and for each iteration the evaluations methods discussed above can be

used [10, p. 181].

k-fold cross-validation introduces a balance between bias and variance, it has smaller bias than using a separate validation set, but a smaller variance than for *Leave-One-Out Cross-Validation*, *LOOCV*, which is the same as k-fold cross-validation when $k = N$ [10, pp. 181–183]. This usually makes k-fold cross-validation balanced between too high bias and too high variance [10, p. 184]. It is also computationally efficient compared to LOOCV [10, p. 181], since usually k is chosen around 10, which makes the number of repetitions considerably smaller than for LOOCV, especially in this particular case where $N \gg 10$.

5 Experimental Study

In this section the data used for the analysis is described, section 5.1, and then the performance of the different methods is presented under the results section 5.2.

5.1 Data

The data was selected as described in the methodology section, 2, and resulted in one dataset from a time frame which we can call the "present" and denote it D_1 and another dataset from another time frame, called the "future", let us denote this dataset D_2. Each observation represents a prospect, or an email and a user. The positive class represents that the email was opened and the negative class represents that the email was not opened.

Dataset	N	N_P	N_N	N_P/N_N	p
D_1	194493	22619	171874	0.132	113
D_1^s	142	17	125	0.136	113
D_2	183597	18666	164931	0.113	113

Table 1: Properties of the data, where N is the number of observations, N_P is the number of observations belonging to the positive class, N_P is the number of observations belonging to the negative class and p the number of features.

D_1 acts as a training set, but some analysis is performed on this dataset in order to check the validity of the models. As seen in Table 1 D_1 contains $N = 194493$ observations with $N_P = 22619$ positive examples and $N_N = 171874$ negative examples. The number of features was equal to $p = 113$.

D_2 acts as a test set and the models performances will mainly be assessed by their results on this dataset. D_2 contains $N = 183597$ observations with $N_P = 18666$ positive examples, $N_N = 164931$ negative examples and $p = 113$ features.

The datasets have been chosen in such a way that it will reflect reality, where the "present", dataset D_1, is known and it is desirable to target users in the "future", dataset D_2. Since the training will be performed with data from D_1 and in D_1 N is very large compared to p we consider the D_1 as a population and not as a sample. Instead of training on the whole population training will be performed on a sample of initially 142 observations, since then the number of training points is similar to the number of features

when preforming 5-fold cross validation, this dataset is denoted D_1^s and was sampled from D_1 with the help of stratified sampling as described in 2.6.7.

5.2 Results

Below the results are presented without discussion. The logistic regression models and random forest were trained on data from D_1 and the performance was evaluated with the help of k-fold cross validation, ROC-curves and lift charts. The analysis was performed on both data from D_1 and D_2.

In some scenarios it can be interesting to send emails to a small sample of the population before sending it to a larger targeted part of the population. It is also more efficient to use a smaller training set, especially if no significant improvement in the performance can be gained. Therefore a smaller subset of data where $N \approx p$ was considered first, namely the dataset D_1^s. We call this setting *high dimensional case* or *HDC*. For comparison we also interest ourselves in the case where $N \gg p$, denoted *LSS* or *large sample scenario*.

5.2.1 Parameter Study

First a parameter study was carried out for the high dimensional case with the dataset D_1^s and 5-fold cross validation, where the AUC was used as the measure of performance. The classes were weighted by their relative frequency, because of the skewness between the classes in the data. Figure 6 shows how AUC varies with C for regularized logistic regression. Maximum AUC seems to be achieved for low C, as such $C = 1$ for both L_1-regularization and L_2-regularization was chosen for the rest of the analysis. For random forest Gini index and cross-entropy as splitting criterion were compared, while varying the max depth on each tree, figure 7, and the number of trees, figure 8. For the rest of the analysis no max depth will be applied, $n = 100$ and Gini index as the splitting criterion was chosen.

5.2.2 ROC Curves

Secondly the ROC curves were created with the help of k-fold cross-validation, where $k = 5$. Figures 9–11 display the ROC curves for logistic regression, with and without regularization, performed on each fold and the average ROC curves over all folds. Similarly figure 12 shows the ROC curves for random forest. The mean ROC curves are compared in figure 13.

Another test was how well the methods preformed on the complete training dataset, D_1, when scaled down to a sample size similar to the number of

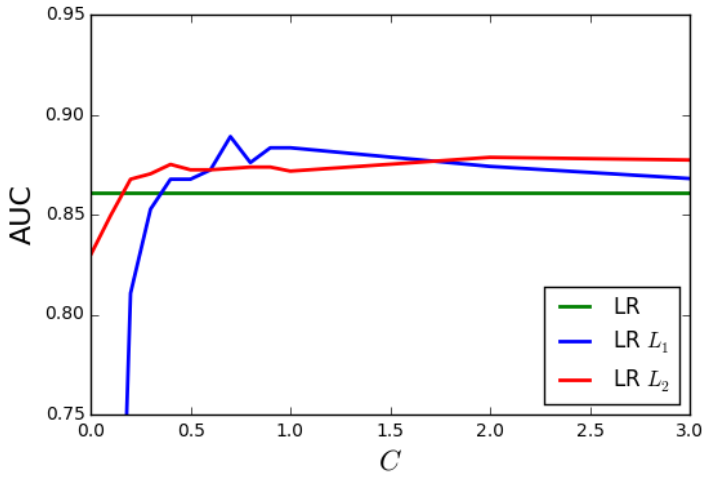

Figure 6: `HDC:` The mean AUC, over 5-fold cross validation, for LR with regularization as a function of the penalty parameter C, when $N \approx p$, dataset D_1^s. LR without regularization is not dependent on C, but included as a reference.

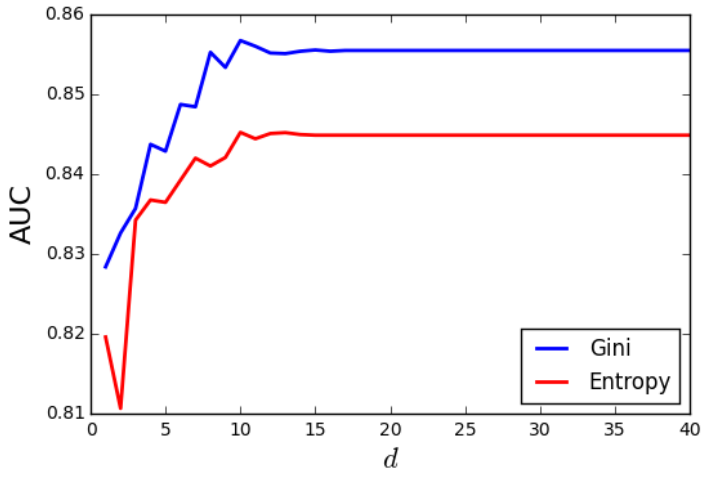

Figure 7: `HDC:` The mean AUC, over 5-fold cross validation, for RF as a function of the max depth d, when $N \approx p$, dataset D_1^s. For each d the mean AUC over the cross validation was averaged over 10 different initial random seeds. The number of trees n was set to 100.

features, as such a stratified sample of $N = 114$ observations was used for training and then it was tested on the remaining dataset. This was repeated 100 times and the average result can be found in figure 14.

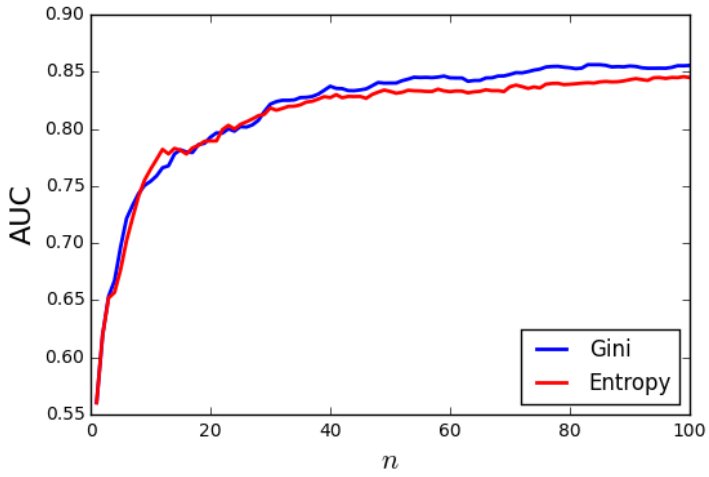

Figure 8: `HDC`: The mean AUC, over 5-fold cross validation, for RF as a function of the number of trees n, when $N \approx p$, dataset D_1^s. For each n the mean AUC over the cross validation was averaged over 10 different initial random seeds. No max depth was applied.

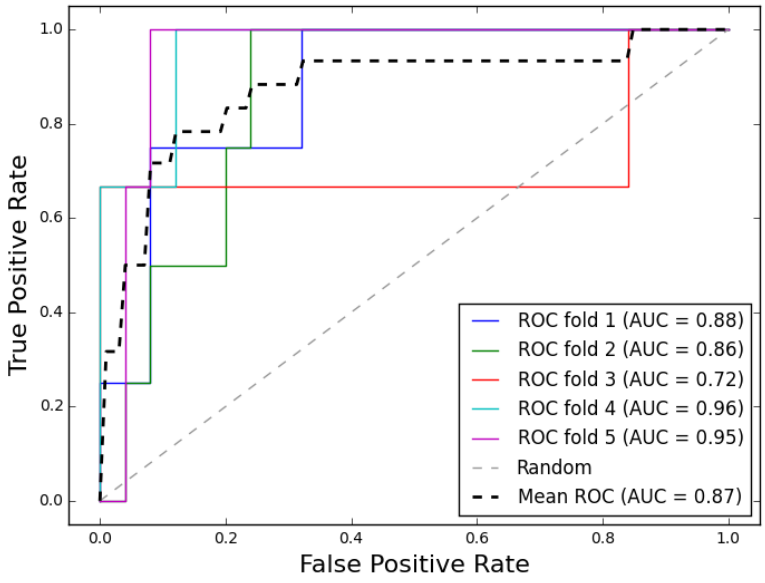

Figure 9: `HDC`: ROC curves for each fold over 5-fold cross validation using LR without regularization, when $N \approx p$, dataset D_1^s. The AUC is given in the legend.

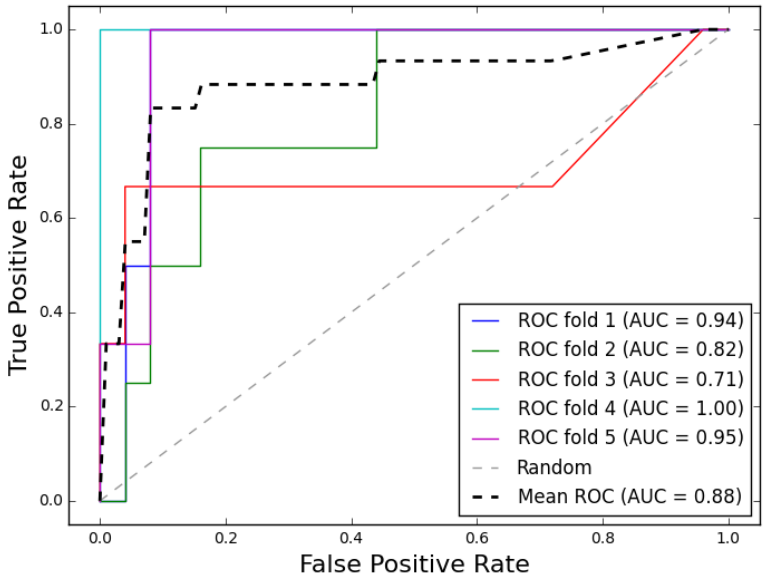

Figure 10: `HDC`: ROC curves for each fold over 5-fold cross validation using LR with L_1-regularization and $C = 1$, when $N \approx p$, dataset D_1^s.

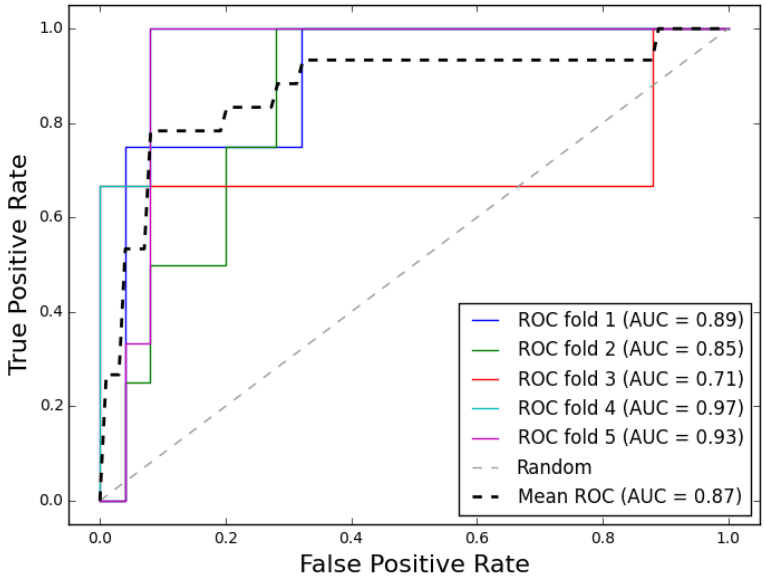

Figure 11: `HDC`: ROC curves for each fold over 5-fold cross validation using LR with L_2-regularization and $C = 1$, when $N \approx p$, dataset D_1^s.

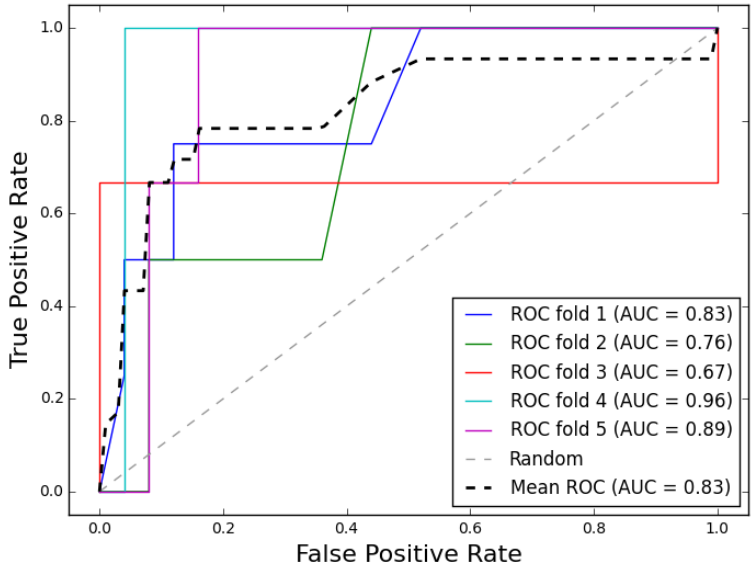

Figure 12: `HDC:` ROC curves for each fold over 5-fold cross validation using random forest with Gini index as splitting criterion, $n = 100$ trees and no max depth, when $N \approx p$, dataset D_1^s.

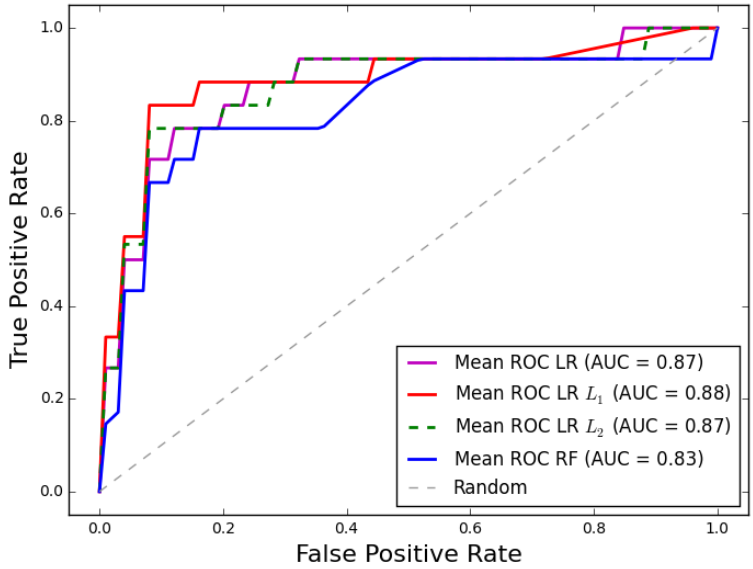

Figure 13: `HDC:` The mean ROC curves, over 5-fold cross validation, for the models as seen in figures 9-12.

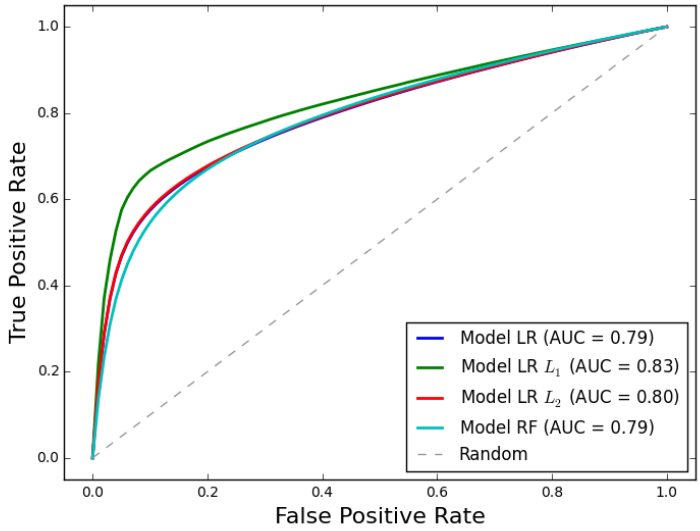

Figure 14: `HDC:` The mean ROC curves, over 100 repetitions, for the different models trained on a sampled dataset from D_1 of size 114 and tested on the remaining 194379 observations in D_1.

Then a test was preformed with training on a sampled dataset of size 114 and then tested on the test set D_2, for 100 different samples, see figure 15. For comparison figure 16 displays the resulting ROC curves when training on the whole training set D_1 and testing on the test set D_2.

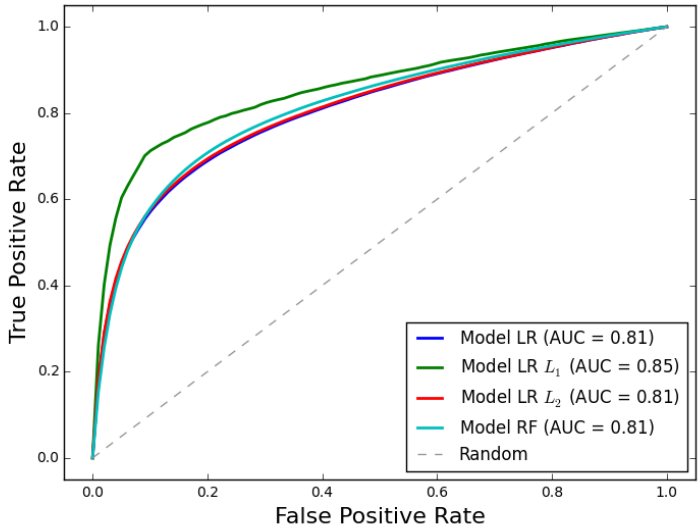

Figure 15: `HDC:` The mean ROC curves, over 100 repetitions, for the different models trained on a sampled dataset from D_1 of size 114 and tested on the test set D_2.

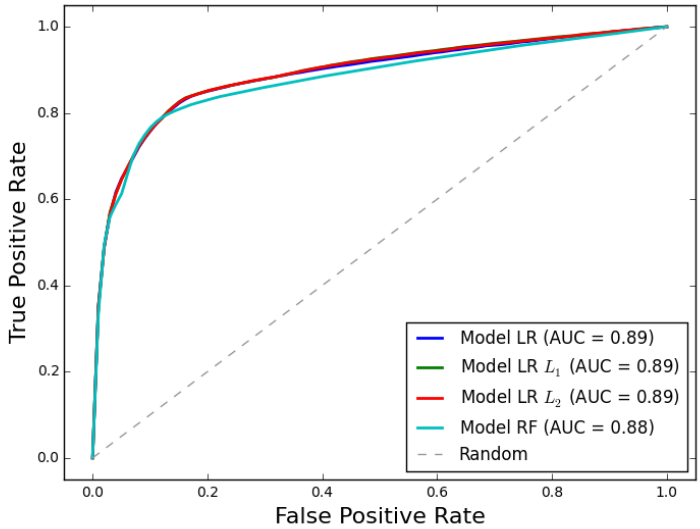

Figure 16: `LSS:` The ROC curves for the different models trained on the training data D_1 and tested on the test set D_2.

5.2.3 Cumulative Captured Response and Lift

In figure 17 the mean captured response and lift for the models are found, performed with 5-fold cross validation on D_1^s.

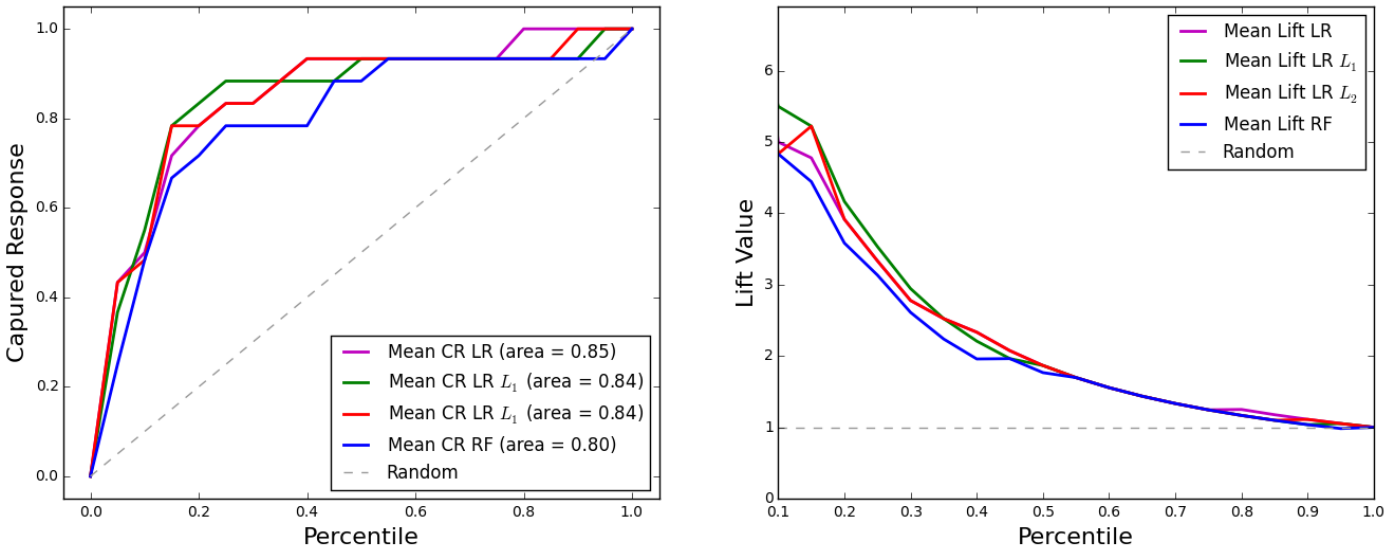

Figure 17: `HDC: Left:` Captured response in average over k-fold cross validation, with $k = 5$, for the different models and a random classifier, when $N \approx p$, dataset D_1^s. `Right:` The lift charts given by the captured response curves to the left.

Figure 18 displays the mean captured response and lift when trained on a sampled dataset from D_1 of size 114 and tested on the test set D_2, averaged over 100 repetitions. For comparison the captured response and lift when trained on the whole training set D_1 and tested on the test set D_2 is shown in figure 19.

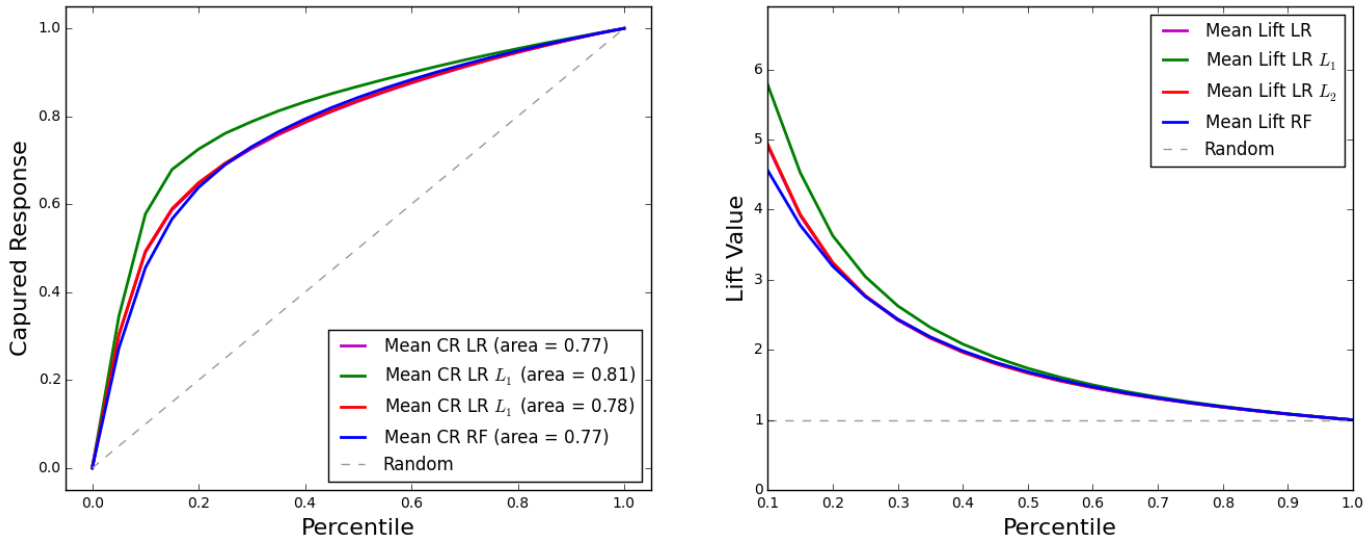

Figure 18: `HDC: Left:` The average captured response when trained on a sampled dataset from D_1 of size 114 and tested on the test set D_2, repeated 100 times. `Right:` The lift charts given by the captured response curves to the left.

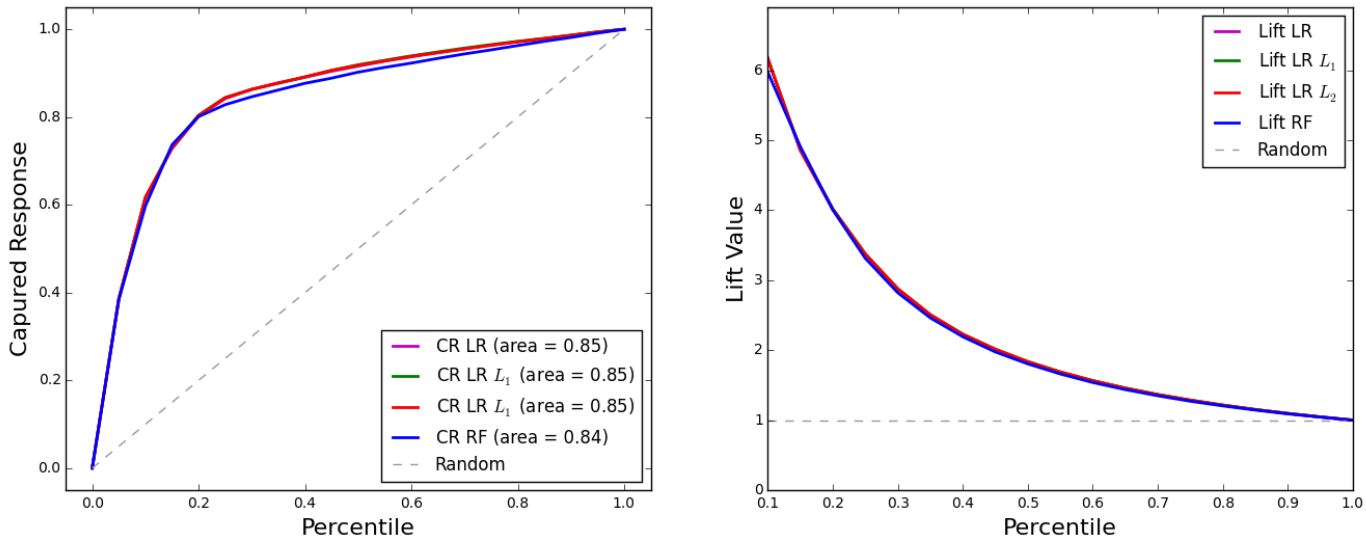

Figure 19: `LSS: Left:` The captured response when trained on the training set D_1 and tested on the test set D_2. `Right:` The lift charts given by the captured response curves to the left.

5.2.4 Varying the Amount of Training Data

The effects on the AUC of varying the size of the sample of the training data to train the model on is displayed in figure 20. 5-fold cross validation was used on D_1 and then the training points were sampled from the non-test folds with the help of stratified sampling and tested on the remaining test fold. This was repeated 10 times and the average AUC was calculated.

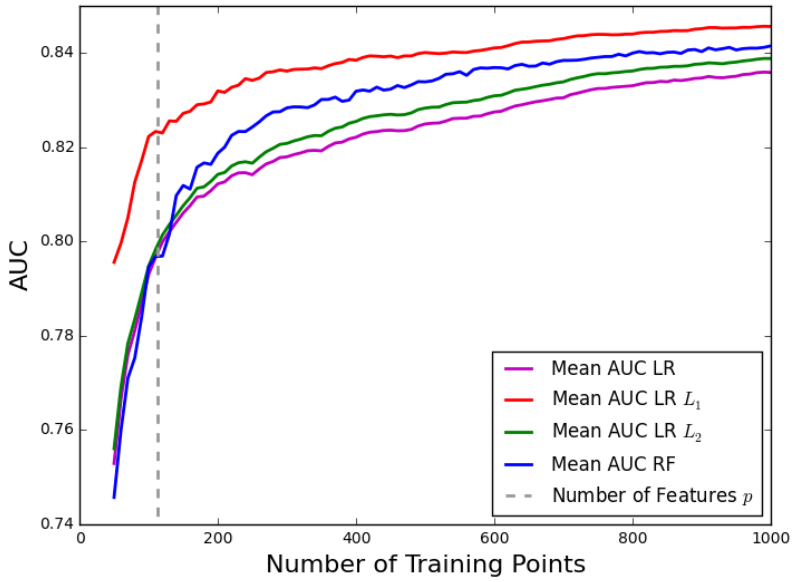

Figure 20: The mean AUC, over 10 repetitions, for the different models when a specific number of the training points were used. For each number of training points 5-fold cross validation was used on D_1 and then the training points were sampled from the non-test folds with the help of stratified sampling and tested on the remaining test fold.

Similarly the effects of varying the number of training points on the performance on the test set D_2 was evaluated and can be found in figure 21. For each number of training points the AUC was averaged over the result for 100 different samples from the training set D_1.

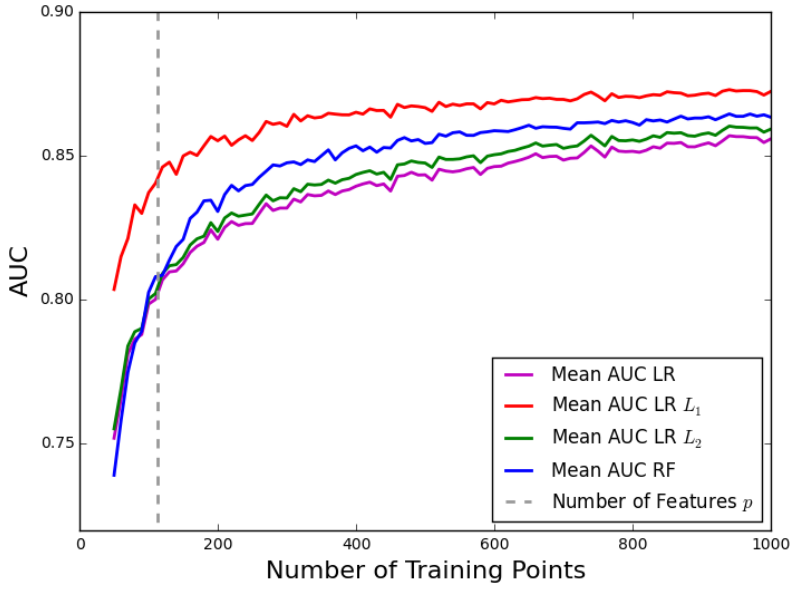

Figure 21: The mean AUC, over 100 repetitions, for the different models when a specific number of the training points were used. The training points were sampled from D_1 with stratified sampling and then the models were tested on the test set D_2.

5.2.5 Precision and Recall

The precision and recall of the models are presented in pair of tables, where the first table displays the result of training on a sampled dataset from D_1 of size 114, the high dimensional case, and testing on the test set D_2, averaged over 100 repetitions. The second table displays the result of training on the whole dataset D_1 and testing on the test set D_2, the large sample scenario. For logistic regression without regularization see table 2 and 3. For logistic regression with L_1-regularization see table 4 and 5. For logistic regression with L_2-regularization see table 6 and 7. For random forest see table 8 and 9. The classification threshold t was chosen independently for each repetition and trial as the t that gives the optimal accuracy according to the ROC curve.

Class	Precision	Recall	Support
0	0.93	0.98	164931
1	0.62	0.32	18666
Avg/Total:	0.90	0.91	183597

Table 2: `HDC:` Table displaying the average precision and recall, over 100 repetitions, for LR, without regularization, trained on a subset of D_1 of size 114 and tested on the test set D_2. The different classes were weighted accordingly to their support. Support represents the number of occurrences of the specific class in the dataset. The average was weighted with the classes' respective support.

Class	Precision	Recall	Support
0	0.95	0.98	164931
1	0.72	0.51	18666
Avg/Total:	0.92	0.93	183597

Table 3: `LSS:` Table displaying the precision and the recall for LR, without regularization, trained on the whole training set D_1 and tested on the test set D_2. The different classes were weighted accordingly to their support. Support represents the number of occurrences of the specific class in the dataset. The average was weighted with the classes' respective support.

Class	Precision	Recall	Support
0	0.94	0.97	164931
1	0.67	0.49	18666
Avg/Total:	0.92	0.92	183597

Table 4: `HDC:` Table displaying the average precision and recall, over 100 repetitions, for LR, with L_1-regularization, trained on a subset of D_1 of size 114 and tested on the test set D_2. The different classes were weighted accordingly to their support. Support represents the number of occurrences of the specific class in the dataset. The average was weighted with the classes' respective support.

Class	Precision	Recall	Support
0	0.95	0.98	164931
1	0.73	0.51	18666
Avg/Total:	0.92	0.93	183597

Table 5: `LSS:` Table displaying the precision and the recall for LR, with L_1-regularization, trained on the whole training set D_1 and tested on the test set D_2. The different classes were weighted accordingly to their support. Support represents the number of occurrences of the specific class in the dataset. The average was weighted with the classes' respective support.

Class	Precision	Recall	Support
0	0.93	0.98	164931
1	0.63	0.32	18666
Avg/Total:	0.90	0.91	183597

Table 6: `HDC:` Table displaying the average precision and recall, over 100 repetitions, for LR, with L_2-regularization, trained on a subset of D_1 of size 114 and tested on the test set D_2. The different classes were weighted accordingly to their support. Support represents the number of occurrences of the specific class in the dataset. The average was weighted with the classes' respective support.

Class	Precision	Recall	Support
0	0.95	0.98	164931
1	0.73	0.50	18666
Avg/Total:	0.92	0.93	183597

Table 7: `LSS:` Table displaying the precision and the recall for LR, with L_2-regularization, trained on the whole training set D_1 and tested on the test set D_2. The different classes were weighted accordingly to their support. Support represents the number of occurrences of the specific class in the dataset. The average was weighted with the classes' respective support.

Class	Precision	Recall	Support
0	0.92	0.98	164931
1	0.61	0.22	18666
Avg/Total:	0.89	0.91	183597

Table 8: `HDC:` Table displaying the average precision and recall, over 100 repetitions, for RF, with 100 trees. Trained on a subset of D_1 of size 114 and tested on the test set D_2. The different classes were weighted accordingly to their support. Support represents the number of occurrences of the specific class in the dataset. The average was weighted with the classes' respective support.

Class	Precision	Recall	Support
0	0.95	0.98	164931
1	0.72	0.52	18666
Avg/Total:	0.92	0.93	183597

Table 9: `LSS:` Table displaying the precision and the recall for RF, with 100 trees. Trained on the whole training set D_1 and tested on the test set D_2. The different classes were weighted accordingly to their support. Support represents the number of occurrences of the specific class in the dataset. The average was weighted with the classes' respective support.

5.2.6 Lasso Based Variable Selection

A further analysis was performed with a subset of features selected with the help of L_1-regularized logistic regression, because of its variable selection properties. Initially a subset of 114 observations was selected from D_1 with all of its features, 113, and then variable selection was performed by training the L_1-regularized model on the data and keeping the variables with non-zero coefficients. Then the models were trained on the subset with the remaining features and tested on the test set D_2. This was repeated 100 times and for each repetition around 8 features remained after the variable selection. The resulting ROC curves can be found in figure 22. A similar analysis was performed with all of the training data D_1, where first variable selection was performed resulting in 87 informative features and the model was trained on the training data and tested on the test set D_2, see figure 23 for the resulting ROC Curves.

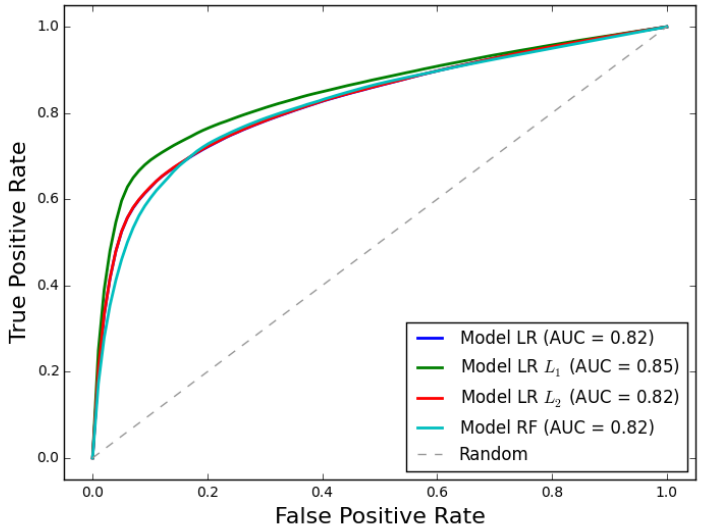

Figure 22: The mean ROC curves for the different models trained on a sampled dataset from D_1 of size 114 and tested on the test set D_2, repeated 100 times and performed with variable selection by using L_1-logistic regression. On average 8 out of 113 features remained after the selection.

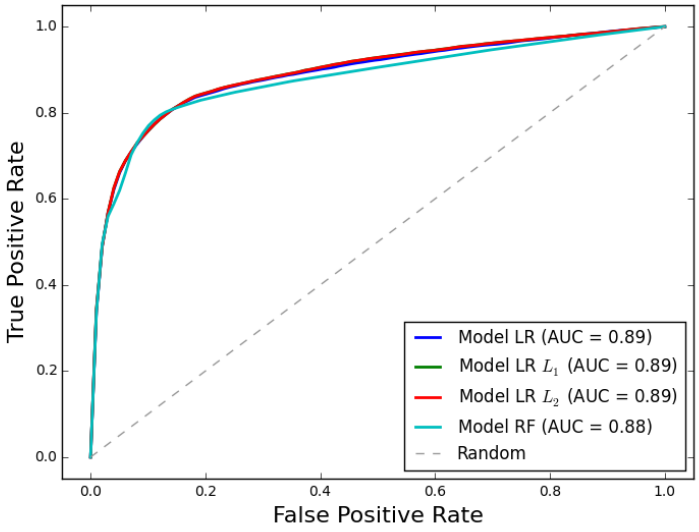

Figure 23: The ROC curves for the different models trained on the training data D_1 and tested on the test set D_2. Performed with variable selection with the help of L_1-logistic regression. 87 out of 113 features remained after the selection.

6 Discussion

This section discusses the results presented in the previous section and tries to formulate useful insights for future work.

6.1 Parameter Study

The effects of changing the penalty parameter C for regularized logistic, which are observed in figure 6, are quite expected, since what is seen is that for C approaching 0 the model is not able to perform well and that is reasonable because a small C puts more weight on the regularization term which does not take the data in consideration, as can be seen in equations (12) and (13). When increasing C the performance improved to an extent, then no further improvement could be observed. Since the first term in equations (12) and (13) becomes larger with the number of training points while there is no change in the second term C was chosen as 1. The effects of this are mainly questionable when varying the number of training points as in section 5.2.4 and figure 20, since then the first term increases, but C stays the same while arguably C should decrease to balance out the increase in the first term.

For random forest the max depth and the number of trees were varied and for the high dimensional case there did not seem to be any reason to enforce any max depth when studying the results under cross-validation. Increasing the number of trees seemed to have a positive impact on the performance and as such it was chosen to 100 for performance and computational reasons. Gini index seemed to perform slightly better than the cross-entropy as splitting criterion and was chosen for that reason.

6.2 ROC Curves

In the high dimensional case when the ROC curve analysis is performed for the different models, with 5-fold cross validation on the sampled training set D_1^s, we see that the logistic regression models seem to perform slightly better than random forest. This is clearly seen in figure 13 and at the same time we observe no significant difference between the different logistic regression models. Instead we seem to observe a difference between the models when training on a sample from the training set D_1 with $N \approx p$ and testing on the reaming points in D_1, figure 14. Logistic regression with L_1-regularization seems to perform better than all of the other models, which might be reasonable since the model will discard features that probably are useless for the classification, while the other models do not perform this variable selection. Interestingly random forest performs at a similar level as the logistic regression and logistic regression with L_2-regularization even though it seemed to

perform worse in the cross-validation setting before, meaning that the random forest classifier found a solution that seem to generalize well compared to its results on the cross-validation task.

When training on a small amount of data and testing on the separate test set D_2, figure 15, the result is similar to the one seen in figure 14. L_1-regularized logistic regression performs better than the other models and achieve a very good result of an AUC around 0.85, which as discussed in section 4.2 can be interpreted as the probability of the model assigning a higher score to a random positive example than a random negative example. This means that the model should be able to perform well in a real world setting where we train on currently available data to try to predict user behaviour in the future even though it was only trained on 114 observations! To compare the analysis was performed by training on all of the available training data D_1, the large sample scenario, and tested on the test set D_2, figure 16. In the figure we see that all the models seem to perform comparably well, with no model outperforming the others. With an AUC around 0.89 the models trained on all of the data performs slightly better but the increase is quite small when regarding that the amount of training data used is over a 1000 times larger!

6.3 Cumulative Captured Response and Lift

As seen in figure 17 the models seem to be able to perform well under cross validation, with random forest performing slightly worse than the other models. When tested on the separate test set D_2 and trained on a small amount of data L_1-regularized logistic regression preforms a bit better than the other models, just as could be seen in the ROC curves, figure 15. It seems possible to capture about 80% of the positive responses by contacting 30% of the population, when using the L_1-regularized logistic regression model. When training on the whole training data, figure 19, the different models perform equally well, but if we compare with the case of only training on a few observations we see that the result only slightly improves, the same as seen in the ROC curve analysis. Increasing the amount of training data a 1000 times increases the performance of the models marginally.

6.4 Varying Amount of Training Data

Figure 20 displays how the AUC increases when the number of training points increases and as can be observed in the figure we see that most of the gain in performance of increasing the number of training points is achieved when the number of training points is smaller than the number of features p. The difference for L_1-regularized logistic regression is especially clear where the slope of the curve changes abruptly around the line where the

number of training points is equal to the number of features. For the other models the change of the slope is not as rapid by it clearly decreases with increasing number of training points. This seems to agree with the earlier analysis that the gain of using larger amounts of data only improves the results marginally. The results when testing on the separate test set, figure 21, are very similar, with L_1 logistic regression performing better than the other models.

6.5 Precision and Recall

Precision and recall on the test set was studied for the different models and trained on different sizes of training data. Generally the precision and recall seem to be lower for the models when trained on small amounts of data as expected, but especially logistic regression with L_1-regularization seem to perform comparably well when trained on only small amounts of data, table 4. On the other hand random forest in the high dimensional case, table 8, has a comparable precision compared to the other models, but a slightly lower recall. Though the recall could be increased at the expense of lowering the precision. Even though random forest in the high dimensional case seems to perform poorly from the perspective of precision and recall it performs comparably well in the sense of AUC and lift as seen in the figures 15 and 18.

6.6 Effects of Variable Selection

An analysis with variable selection was performed since L_1-regularized logistic regression performs well even with small amounts of data. In general the variable selection resulted in a decrease in the number of features, p, with more than 90%, when working on a dataset of size 114. The difference in performance to case with no variable selection is best studied by comparing figure 15 and 22. As can be seen there is almost no difference and if there is one then it slightly favours the models trained with variable selection. The interpretation of this is that there seems to be a few features which are able to describe the properties of the data especially well, since excluding such a large portion of the features does not impact the results.

Variable selection was also performed when using all of the training data and then 87 features were kept. The resulting ROC curve after variable selection, figure 23, is very similar to the one seen without variable selection, figure 16. This increases our confidence that some of the features are non-informative for the classification. Since the difference in performance is small compared to when trained on a small amount of data and a small amount of features the thought that some particular features contain most of the valuable information is strengthened.

6.7 HDC Compared to LSS

Logistic regression with L_1-regularization performs better than the other models in the high dimensional case and this might be because of its variable selection properties and that it recognises that some of the available features bring no information of value for the classification, which seem quite likely. Because what constitutes a user that opens emails is not really clear and to imagine how for example gender or listening patterns affects the users likelihood to open an email is not easy. This makes it quite expected that some features that are given some kind of impact by the other models are actually only finding noise in the training data and will not generalize as well as a result.

In the large sample scenario all the models perform equally well, which is different from the high dimensional case. This is most likely because of the huge amounts of data that cancels out most of the regularization effects for logistic regression, but why random forest have the same level of performance is a bit harder to explain, since some difference is reasonably expected. Perhaps the large amounts of training data causes the methods to find approximately the same solutions.

The performance difference when comparing the different cases are marginal compared to the massive increase in data available in the large sample scenario. Naturally the performance increases with more data available, but also the computational time to train the model increases and as seen in figure 21 the gain in AUC quickly decreases when the number of training points is increased. These different cases have different applications, while LSS can be used for trying to gain accuracy in the model and can be used in special cases when high performance is especially important, HDC on the other hand shows that perhaps small tests can be performed before sending emails to a larger amount of users and that a small test size is sufficient.

7 Conclusions and Future Work

Below conclusion that can be made from the study is presented, section 7.1, and recommendations for both how to use the result and future work, section 7.2.

7.1 Conclusions

By following the methodology presented by Berry and Linoff, as described in section 1.2, and avoiding crucial mistakes such as learning things that are not true or learning thing that are true but not useful it is possible to present some useful conclusions with confidence. The objective was ultimately to increase the positive response rate for emails sent by Spotify and there seem to be great possibilities for such an improvement, by targeting user with the help of historical data.

A large increase in the positive response rate can be achieved, for example by targeting the top 20% of the users ordered by their score. Then as seen in the lift chart of figure 19 the lift is around 400% or a 300% increase in the response rate! Converted to specific numbers it means that instead of sending emails to over $180,000$ and receiving around $18,500$ positive responses, which gives a response rate around 10%, we can achieve a response rate around 40% by targeting $36,000$ users, with around $14,500$ positive responses. Similar results can be achieved even if only a small amount of training data is used as seen in figure 18. The ROC curves also indicates that successful targeting can be achieved, with an AUC of around 0.85 on the test set.

The second question was to investigate how using a small amount of training data, the high dimensional case, would compare to using large amounts of data, the large sample scenario. As seen in the results for all of the evaluation methods the gain of increasing the number of observations from close to the number of features, $N \approx p$, to a number more than a factor 1000 more is quite marginal. The difference is especially small for the L_1-regularized logistic regression method, which performs better than the other methods in the high dimensional case.

Because Lasso logistic regression performed particularly well and because of its variable selection properties the effects of Lasso based variable selection was studied. In the high dimensional case the result was a massive reduction in the number of features, which was reduced by more than 90%, while the predictive power was unchanged. Similarly a large reduction in the number of features was achieved in the large sample scenario, without any loss in performance. This indicates that most of the predictive power was contained

in only a few features and that there is a great sparsity in the classification information.

7.2 Future Work

This thesis only scrapes what we believe to be the surface of possible improvements of email communication and there seem to be endless opportunities for further increase in performance.

An initial step could be to include more information about the emails form and content, especially when trying to improve the click rate, the number of users that click on the links in the email, and perhaps use more information about the users specifically.

The analysis could also be extended to include negative responses such as spam reports or users that unsubscribe, possibly by trying a multi-class classification model, arguably random forest classifier. It would also be interesting to try other models and methods, even in the original setting.

Performing experiments could tell more about the performance of using targeting and eventually also lead to the deployment of such a targeting system. Several different types of experiments would be interesting to perform, but the most urgent one would be a simple test of sending an email to a targeted group compared to sending it to a randomly selected group. After deployment it will be interesting to see the long term effects of targeting users, hopefully it will increase the response rates but perhaps the types of users that will be targeted will change over time.

Ultimately a company is interested in creating revenue and as such it will be more interesting to try to determine the effects of email targeting on the profit made or the amount of premium users in Spotify's case, instead of only studying the response rate. This is a very interesting problem, not only because it is important but also since it is not entirely clear how such a analysis would be performed, since finding a direct connection between possible higher email response rates and premium conversion perhaps is not possible.

In the scenario studied in this thesis the contents of the emails have already been determined and as such users have been targeted from the perspective of finding users interested in this particular information, but it would perhaps be possible to turn the tables and find a need for specific information amongst users. Then instead of asking "which users should we send this information to?" we ask "what information are each user interested in receiving?".

References

[1] Michael J.A. Berry and Gordon Linoff. *Data mining techniques : for marketing, sales, and customer relationship management*. Wiley Publishing, Inc., second edition, 2004.

[2] Henrik Bostrom. Estimating class probabilities in random forests. In *Machine Learning and Applications, 2007. ICMLA 2007. Sixth International Conference on*, pages 211–216. IEEE, 2007.

[3] Léon Bottou. Large-scale machine learning with stochastic gradient descent. In *Proceedings of COMPSTAT'2010*, pages 177–186. Springer, 2010.

[4] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[5] Yen-Liang Chen, Chang-Ling Hsu, and Shih-Chieh Chou. Constructing a multi-valued and multi-labeled decision tree. *Expert Systems with Applications*, 25(2):199 – 209, 2003.

[6] J. P. Egan. *Signal detection theory and ROC analysis*. Series in Cognition and Perception. Academic Press, 1975.

[7] M.A. Friedl and C.E. Brodley. Decision tree classification of land cover from remotely sensed data. *Remote Sensing of Environment*, 61(3):399 – 409, 1997.

[8] David W Hosmer Jr and Stanley Lemeshow. *Applied logistic regression*. John Wiley & Sons, 2004.

[9] Hyunseok Hwang, Taesoo Jung, and Euiho Suh. An {LTV} model and customer segmentation based on customer value: a case study on the wireless telecommunication industry. *Expert Systems with Applications*, 26(2):181 – 188, 2004.

[10] Gareth James, Daniela Witten, Trevor Hastie, and Robert Tibshirani. *An introduction to statistical learning*. Springer, 2013.

[11] SB Kotsiantis, D Kanellopoulos, and PE Pintelas. Data preprocessing for supervised leaning. *International Journal of Computer Science*, 1(2):111–117, 2006.

[12] Harald Lang. *Topics on Applied Mathematical Statistics*. KTH Teknikvetenskap, version 0.97, 2013.

[13] Saskia Le Cessie and Johannes C Van Houwelingen. Ridge estimators in logistic regression. *Applied statistics*, 41(1):191–201, 1992.

[14] Su-In Lee, Honglak Lee, Pieter Abbeel, and Andrew Y Ng. Efficient L_1 regularized logistic regression. In *Proceedings of the National Conference on Artificial Intelligence*, volume 21, page 401. Menlo Park, CA; Cambridge, MA; London; AAAI Press; MIT Press; 1999, 2006.

[15] Charles X Ling and Chenghui Li. Data mining for direct marketing: Problems and solutions. In *KDD*, volume 98, pages 73–79, 1998.

[16] Kevin P Murphy. *Machine learning: a probabilistic perspective*. MIT press, 2012.

[17] E.W.T. Ngai, Li Xiu, and D.C.K. Chau. Application of data mining techniques in customer relationship management: A literature review and classification. *Expert Systems with Applications*, 36(2, Part 2):2592 – 2602, 2009.

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

[19] Chao-Ying Joanne Peng, Kuk Lida Lee, and Gary M Ingersoll. An introduction to logistic regression analysis and reporting. *The Journal of Educational Research*, 96(1):3–14, 2002.

[20] Piotr Rzepakowski and Szymon Jaroszewicz. Decision trees for uplift modeling with single and multiple treatments. *Knowledge and Information Systems*, 32(2):303–327, 2012.

[21] Carl-Erik Särndal, Bengt Swensson, and Jan Wretman. *Model assisted survey sampling*. Springer Science & Business Media, 2003.

[22] Tom Mitchell Machine Learning slides Ch 3. Available from: `http://www.cs.cmu.edu/afs/cs.cmu.edu/project/theo-20/www/mlbook/ch3.pdf` [cited 15:th of April 2015].

[23] Spotify user data. Available from: `https://press.spotify.com/se/information/` [cited 30:th of January 2015].

[24] Vladimir N Vapnik and A Ya Chervonenkis. On the uniform convergence of relative frequencies of events to their probabilities. *Theory of Probability & Its Applications*, 16(2):264–280, 1971.

[25] Miha Vuk and Tomaz Curk. Roc curve, lift chart and calibration plot. *Metodoloski zvezki*, 3(1):89–108, 2006.

[26] Chien-Hsing Wu, Shu-Chen Kao, Yann-Yean Su, and Chuan-Chun Wu. Targeting customers via discovery knowledge for the insurance industry. *Expert Systems with Applications*, 29(2):291 – 299, 2005.

[27] Hsiang-Fu Yu, Fang-Lan Huang, and Chih-Jen Lin. Dual coordinate descent methods for logistic regression and maximum entropy models. *Machine Learning*, 85(1-2):41–75, 2011.