Bacheloroppgave

NTNU
Norges teknisk-naturvitenskapelige universitet
Fakultet for økonomi
NTNU Handelshøyskolen

Marcus Hjertaas
Henrik Krantz Knudsen
Jakob Lindstrøm
Joakim Sælemyr

Identifying the Best Machine Learning Model for Predicting Bank Term Deposits: An Empirical Study Using Public, Post Financial Crisis Data

Bacheloroppgave i Økonomi og administrasjon
Veileder: Denis Mike Becker
April 2023

NTNU
Kunnskap for en bedre verden

Marcus Hjertaas
Henrik Krantz Knudsen
Jakob Lindstrøm
Joakim Sælemyr

Identifying the Best Machine Learning Model for Predicting Bank Term Deposits: An Empirical Study Using Public, Post Financial Crisis Data

Bacheloroppgave i Økonomi og administrasjon
Veileder: Denis Mike Becker
April 2023

Norges teknisk-naturvitenskapelige universitet
Fakultet for økonomi
NTNU Handelshøyskolen

NTNU
Kunnskap for en bedre verden

2/1/2023

Identifying the Best Machine Learning Model for Predicting Bank Term Deposits:

An Empirical Study Using Public, Post Financial Crisis Data

Bachelor thesis for Business Administration

Supervisor: Denis Becker

Norwegian University of Science and Technology

Faculty of Economics and Management- NTNU Business School

Marcus Hjertaas
Henrik Krantz Knudsen
Jakob Lindstrøm
Joakim Sælemyr

Preface

This bachelor thesis is the culmination of our three years of study at NTNU Business School. In the course of our research, we have gained a comprehensive understanding of various machine learning models and their applications in predicting long-term bank deposits from the public. The journey have been an engaging and challenging experience, which has enriched our knowledge significantly.

We would like to express our gratitude to our supervisor, Denis Becker, for his excellent collaboration and exceptional service throughout the project. His constant support, active involvement and easy accessibility have been highly appreciated.

The content of this thesis is the responsibility of the authors.

Executive summary

The purpose of this bachelor thesis is to find the best machine learning method to accurately predict bank term deposits from the public. We are viewing the issue from a business perspective, also taking real-world implications into account.

The data which the models are applied to provides a series of different information from a sample of Portuguese individuals, including whether or not these individuals have made committed long-term deposits into the bank. The data have gone through a set of configurations and resampling techniques. Retrieval of the data was in the wake of the Great Financial Crisis of 2008.

Four machine learning models were chosen for prediction, they are all based on acknowledged statistical classification methods, these are: binomial logistic regression, decision tree classifier, artificial neural networks, and support vector machine. These methods vary in complexity and have different sets of advantages and disadvantages. They will be applied to face the prediction-challenges in the data. It’s not possible to combine one method to cover another method's disadvantages. However, by using several methods it is believed that one can find the model that best faces the specific challenges provided by the data.

The best model is based on a comprehensive assessment, including two criterions. The first criterion is based on the prediction rate and the model’s ability at predicting actual possible customers. The second criterion concerns the model's training data and its amount of resampling interference.

The models are configured to give predictions for the conditions during the period of data-retrieval and are therefore constrained to these conditions. According to our findings, the support vector machine model trained on the undersampled data is the most favorable model. The model showed both great prediction accuracy and managed to classify the minority class precisely.

Sammendrag

Formålet med denne bacheloroppgaven er å finne den beste maskinlæringsmodellen for å nøyaktig predikere bankinnskudd fra offentligheten. Oppgaven blir sett på fra et bedriftsperspektiv, hvor en også tar hensyn til reelle faktorer som gjorde seg gjeldende i den aktuelle tidsperioden for datainnhentingen.

Dataen som blir benyttet inneholder en mengde ulik informasjon fra et utvalg av portugisiske individer, deriblant hvorvidt disse individene har gjennomført et forpliktende langsiktig bankinnskudd eller ikke. Tallmaterialet har gjennomgått en rekke variabel-konfigurasjoner og data strukturelle transformasjoner. Observasjonene er hentet i kjølvannet av finanskrisen i 2008.

Fire maskinlæringsmodeller ble anvendt for prediksjon. De er alle basert på anerkjente statistiske klassifikasjons modeller, disse er: binomial logistisk regresjon, beslutningstreklassifisering, kunstige nevrale nettverk og støtte vektor maskiner. Modellene varierer i kompleksitet, og har ulike fordeler og ulemper. Disse vil bli brukt for løse prediksjons-utfordringer som dataen inneholder. Det er ikke mulig å kombinere metodene for å kompensere for deres individuelle ulemper. Derimot, ved bruk av flere metoder kan det undersøkes hvilken modell som best imøtekommer de spesifikke utfordringene som dataen byr på.

Den beste modellen blir valgt basert på en helhetlig vurdering. Det første kriteriet baserer seg på prediksjons-nøyaktighet og modellens evne til å klassifisere potensielle kunder. Det andre kriteriet baserer seg på treningsdataen, og graden av transformasjon i datastrukturen.

Modellen er konfigurert for å predikere data gitt de gjeldende forholdene i datainnhentings perioden, derfor er modellene begrenset til disse forutsetningene. Våre funn tilsier at en modell basert på støtte vektor maskiner, trent på transformert data i henhold til randomisert under-utvelgelse av data, er den mest nøyaktige modellen. Modellen viser god evne til prediksjons-nøyaktighet og klassifikasjon av minoritetsklassen.

Table of Contents

Figure list

Table list

1. Introduction to the theme

A bank term deposit is an arrangement for banks to secure a stable and predictable way of knowing ahead of time how much they are able to lend at any given time. A term deposit is a deposit account where the money is locked up for a pre-set period of time. While a shorter duration results in a lower interest rate, a longer maturity yields a higher interest. Maturity ranges from one month to a few years (Chen, 2022, paragraph 5). The bank's purpose is to reinvest the money from these term deposit accounts in other financial projects that produce a higher rate of return. The bank could also lend the money to other clients, thus receiving a higher interest from borrowers. This discrepancy between what the bank rewards its customers for term deposits and the rate it charges its borrowers is called *net interest margin* (Bloomenthal, 2022, paragraph 7). If invested wisely, the margin associated with term deposits can make them a lucrative option for banks, potentially yielding a significant return.

There are two basic ways for businesses to market their goods and/or services: through mass marketing campaigns that target the whole public without discrimination, or relationship marketing that target a segmented group. (Framnes et al., 2021)
In practice, relationship marketing focuses more on personalized measurements in response to each consumer's wants and needs, in contrast to mass marketing techniques. Mass marketing can be defined as the uniform use of competition funding to bigger market groups (Framnes et al., 2021). Relationship marketing targets specific sections, treating every customer as a separate segment. Traditionally speaking it’s impractical and expensive, tailoring communications, products, prices, and distribution to each customer can yield long-term benefits. Relationship marketing is conducted with the intention of boosting consumer happiness and brand loyalty. (Framnes et al., 2021). We thereby came up with this research question:

“Which classification model can accurately predict bank term deposits from the public?”

J.A. Choi and K.Lim have written an article which investigates and classifies different machine learning techniques that are used in business in order to enhance targeted online

advertising. In this study, twenty-three machine learning-based online targeted advertising strategies were identified and classified largely into two categories, user-centric and content-centric approaches. This paper also identifies an under-examined area, algorithm-based detection of click frauds, to illustrate how machine learning approaches can be integrated to preserve the viability of online advertising. (Choi et al., 2020)

2. Method

2.1 The Bias-Variance Tradeoff

The bias-variance tradeoff represents a classical belief in machine learning theory. This relationship is the concept of minimizing both overfitting and underfitting in order to achieve the best results for both train and test data. This concept applies to most machine learning models, however, it is more significant in some models than others. Underfitting refers to a model with too few parameters to adjust to, meaning the classifications are highly irregular, implying high variance. Often an underfit model will perform poorly on both training and testing data. Opposite of this is overfitting, implicating that the model fits the training data too well. In other words, the model is biased and will, by extension, perform significantly worse on the test data (Belkin et al., 2019).

Optimally the model should accurately predict new data points, which means the bias-variance tradeoff is of high relevance when adjusting the models to provide more precise results. It is worth mentioning that when utilizing interpolation some neural networks and decision trees have been able to break free of this tradeoff and receive extremely accurate results for both training and testing data (Belkin et al., 2019). This, however, applies to more complex models than the ones used in this thesis.

2.2 Accuracy paradox

The raw dataset contained a large imbalance in relation to the amount of 0s and 1s. The imbalance is addressed in more detail later, however, it is evident that the "Accuracy

Paradox" applies to this data, and by extension this thesis. Classification accuracy, or the proportion of accurate predictions to total predictions, is a metric used to assess the effectiveness of a classification model. This can be visualized by a confusion matrix for a two-class problem, which presents the results obtained by the classification models:

		Predicted	
		Positive	Negative
True	Positive	TP	FN
	Negative	FP	TN

(Branco et al., 2015, paragraph 6)

Accuracy, in this case, can be calculated as:

$$Accuracy = \frac{TP+TN}{TP+FN+TN+FP}.$$

In the formula, the number of *True Positives* (TP) and *True Negatives* (TN) are divided by the total sum of true classifiers plus the *False Negative* (FN) and the *False Positive* (FP) (Branco et al., 2015). The objective is to determine whether a customer will make a term deposit (1) or not (0). An explanation of the predicted/true matrix can be found below:

- True Positives occur when the model indicates 1 while in reality being 1.
- True Negatives occur when the model indicates 0 while in reality being 0.
- False Negatives occur when the model indicates 0 while in reality being 1.
- False Positives occur when the model indicates 1 while in reality being 0.

The models would ideally only predict True Positives and True Negatives, but the dataset used for this thesis is heavily imbalanced. The dataset contains nearly 40,000 instances of the majority class (0) and only 5,000 instances of the minority class (1). As a result, the models seemingly predict with high accuracy, however, in reality they simply predict every result as being of the majority class. This is, however, misleading; it conveys the information that the model performs excellently when in reality the model fails at a necessary part of this thesis

objective. The paradox relates heavily to the bias-variance tradeoff, as a more accurate prediction of the minority results in more False Positives. By extension, the model predicts that a customer will make a term deposit, as a result, the bank attempts to target this customer, even though they would not make a commitment. This could lead to a potential loss of revenue - an ineffective use of marketing resources. On the other hand, if this increases the chance of True Positives, this will make up for the number of False Positives, resulting in a higher chance of targeting genuine customers that would sign up for a bank term deposit. As a result, the bank will ultimately end up with a greater chance of yielding a higher net interest margin.

2.3 Statistical models

2.3.1 Binomial Logistic Regression

Binomial logistic regression is a popular statistical method within machine learning (Downey, 2014). Its field of application is regression and classification. The method requires one explained variable and one or more explanatory variables. A logistic regression applies the explanatory variables to predict the possibilities for whether the target variable is true or false.

The target variable of a binomial logistic regression must be binary, for example: 1/0, or true/false. The binomial logistic regression can be either univariate or multiple, depending on the number of explanatory variables. There are no requirements for the explanatory variables - they can be either discrete or continuous.

The method builds a regression model where the target variable is analyzed with its corresponding explanatory variables. The model then creates coefficients that are multiplied by the value of their corresponding variables. The sum of these products is the possibility of the target variable being true.

The regression model is based on the logistic function, also known as the sigmoid function. This function has the ability of giving y-values exclusively within the range of 0 and 1. The function is therefore suitable for the purpose of generating possibilities for a binary variable.

The y-value of the function defines the possibility of the target variable, and it's common to use 0.5 as the threshold for deciding whether the output is true or false. However, the threshold value can be decided in accordance with the nature and requirements of the project. It’s important to be aware of the precision/recall tradeoff (Gordon and Kochen, 1989). The tradeoff illustrates that an increase in the threshold will give high precision and low recall, and vice versa if the threshold is reduced.

Binomial logistic regression is a suitable tool for classification. The model is relatively easy to apply compared to other machine-learning classifiers, thus allowing for a precision/recall tradeoff adjusted by qualitative factors.

2.3.2 Decision Tree Classifier

A decision tree classifier is a supervised machine learning algorithm built to create decision boundaries that separate data points and assign them a class. Differing from many other classifying models, decision trees do not predict a data points class, rather it chooses where to assign it. The structure is similar to how humans make decisions. It approaches classification using rules, defining a threshold and distributing true or false values to data points based on this. However, what makes the decision tree differ from human decisions lies in its machine learning. Whereas humans would have simple, yet considered, thresholds when classifying the data, a decision tree mathematically acquires the optimal decision thresholds.

The mathematics will be further explained in the next paragraph, however, in order to truly grasp the decision tree model it is necessary to understand the model’s structure, more specifically the concept of nodes. The model contains three types of nodes, nodes being the entity that contains the threshold or if-statements (Swain P. H. & Hauska H, 1977). The root node is singular and found at the base of the model, all the data points pass through and are split according to its criteria. Following it are internal nodes, similarly containing an if-statement classifying the data points. Lastly there are leaf nodes, these nodes do not contain any if-statements or thresholds, yet are ultimately what classifies the new data. As most machine learning models, a decision tree is created using training data. The leaf nodes classify new data based on what the majority of its training observations are classified as. The objective of a decision tree is to create pure leaf nodes, whereas all data points are of the same class. However, as explained later on, this is not always optimal and with impure

leaves, where classifications are mixed, new data is assigned by majority based on the training majority.

Moving over to mathematics, the decision tree classifier uses impurity or loss variables, the most common variables used for decision tree classifiers are entropy, gini impurity and logistic loss. This study applies gini impurity because it is the default for sklearn's decision tree classifier as well as being fairly uncomplicated. Gini impurity is used to decide the optimal split form for a root node and the following internal nodes (Nembrini et al., 2018). When applying gini impurity you want to minimize its value in each node, lower impurity means higher homogeneity and a pure leaf node would receive a gini impurity zero. Gini impurity is in simple terms a measure of variance across classes (Nembrini et al., 2018). The Gini impurity-index is calculated as follows:

$$Gini = 1 - \sum_{i=1}^{C} (p_i)^2$$

$Gini$ = The Gini impurity

p_i = The probability of a given class

i = The given class

C = The total number of classes

Using a simple example of green and red points, the calculation is done by dividing the number of green data points in a node by the total number of observations in said node. This results in pi, the value is then multiplied by itself and the process is repeated for the red points and both values are subtracted from one. This returns a gini impurity for a single node.

The next step would be calculating the total gini impurity for an if-statement or threshold. This total value is the weighted average for the impurity of its leaves, meaning the model calculates the probability of a point being in a leaf and multiplying that with the leaf's gini impurity. This process is repeated for all variables in an attempt to locate the split with the lowest gini impurity to be the root node. This split warrants new gini values and the calculations are repeated to find the next optimal split based on the root node's separation. For a numeric or continuous variable the mathematical procedure is similar. The decision tree

creates various thresholds between the variable values and calculates their individual gini impurity in order to apply the lowest one.

Overfitting was previously touched upon, this term is highly relevant for decision tree's as the classification model has a reputation of overfitting its tree. This is due to the model's target of constructing only pure leaf nodes. In actuality the optimal decision tree is as small as possible, maintaining high accuracy on new data and sacrificing the pure leaf nodes. The two most notable methods used for reaching the optimal tree is pruning and cross validation. The method of pruning used in this thesis is cost complexity pruning, often abbreviated to ccp, this method allows for adjustments in the decision tree's alpha value and, by extension, optimize it for unseen data by removing unneeded nodes that overfit the model (The Pennsylvania State University, 2023). The alpha value regulates the size of the decision tree. Using cross validation the model can regulate and test for an optimal alpha value or set boundaries to the amount of observations that can be assigned to leaf nodes.

The decision tree's biggest advantage is the model's ability to clearly visualize its decisions and reasoning for classifying subjects. This can be extremely helpful as most classification and regression models present a probability or classification result without any clarification making interpretation impossible for anyone without sufficient knowledge about the model. A decision tree however presents a neat and understandable visualization most would understand. In a practical case like this thesis, such an explanation could prove valuable.

2.3.3 Artificial Neural Network

Artificial Neural Networks, from here on abbreviated as ANNs, are computing systems that are inspired by the biological neural structures constituted in the brain. ANNs consist of interconnected processing units that are referred to as neurons. The neurons within the interconnected system transmit and process information. The intricate structure of these neurons within the ANNs enables them to learn, make predictions and recognize patterns. In terms of usage, the possibilities are endless and neural networks are used in speech recognition, natural language processing, financial forecasting and much more. ANNs date back to the 1940s and have seen significant improvements since, and are today regarded as one of the most powerful and versatile tools in the field of artificial intelligence (Maind and Wankar, 2014).

In assessment of the building blocks for ANNs, the composition of the layers in which the neurons themselves are linked is a vital component. The distinctions in regards to the layers are: the first layer where data is received is referred to as the input layer, the last layer where the result is produced is referred to as the output layer and everything in between is referred to as the hidden layers. The hidden layers can be of different types, this includes fully connected-, recurrent- or convolutional layers. Recurrent layers are used for processing sequential data by capturing time-related dependencies stored through memory maintenance of precursory data iterations (Zhang et al., 1997). Convolutional layers are designed to process images and signals by harnessing correlations in spatial and temporal data structures between adjacent data points, thus procuring comprehension and knowledge of local features. In a fully connected network, each neuron in each layer is connected to each neuron in the previous layer (Albawu et al., 2016)

Another crucial building block of ANNs are the neurons themselves. Neurons in ANNs are the primary processing units that receive inputs from other neurons or the network's external environment. The neurons then generate an output by processing that input. For each neuron, the weights and biases are key parameters that determine the outputs that the neurons return. The weights and biases are repeatedly optimized and adjusted during training to ameliorate the performance of any given task persistently (Siddique & Tokhi, 2001).

The weights of a neuron embody the strength of the connection between the inputs and outputs, with each neuron having an allocated weight. The output of each neuron is a weighted sum of the inputs and is passed through an activation function. The initial weights are arbitrary and are from then on refined through a process called backpropagation. In backpropagation, the gradient of a network’s error is computed with respect to the weights and from then on updated using gradient descent (Siddique & Tokhi, 2001).

The biases represent a constant input that is solely reliant on itself and is independent of the actual input, thus allowing the neuron to adjust its output autonomously. In the same way weights are optimized, biases are refined through backpropagation during training. In a simplistic sense biases are weights that are always multiplied by one (Siddique & Tokhi, 2001).

The process of computing each output of each neuron can be portrayed mathematically as follows (Sarhan & Helatat, 2007):

$$y = f(w_1 \cdot x_1 + w_2 \cdot x_2 + \ldots + w_n \cdot x_n + b)$$

y = The output of the neuron

f = The activation function

x = Each input

w = The weight associated with each input

b = The bias term

n = The number of inputs to the neuron

The formula demonstrates how each neuron computes its output through weights and biases and an activation function. Although weights and biases are optimized using backpropagation that relies on the calculation of the partial derivatives in a loss function, it is not that simple to calculate the optimal activation functions. An activation function does not have a direct gradient with respect to the loss function, thus making the same calculation obsolete. The choice of activation function is therefore based on prior knowledge, empirical experimentation and other heuristics. There are a lot of activation functions and all of them will not be addressed. In this thesis three activation functions will be addressed and discussed, these are: Sigmoid, ReLU and Leaky ReLU (Sharma et al., 2020).

For binary decisions in ANNs, a sigmoid function is a commonly used activation function that mathematically maps a value between 0 and 1. While the sigmoid function is useful and is therefore widely used, it has some liabilities - namely the vanishing gradient problem. The vanishing gradient problem implies that when the outputs are close to 1 or 0 the gradient is too small for the network to adjust its weights and biases properly. The Sigmoid function can be expressed as follows (Sharma et al., 2020):

$$Sigmoid: f(x) = \frac{1}{1 + e^{-x}}$$

Due to the issues regarding the vanishing gradient problem, the ReLU activation function has gained popularity as a viable alternative. The ReLU returns either 0 or any value up to 1. The ReLU function has a constant gradient and is efficient in terms of computational usage. While being useful and popular ReLU also has issues. One issue is that a neuron can effectively die and no longer contribute to the network. This occurs when the output is always zero for certain inputs. The leaky ReLU activation function avoids this by not letting all

values under zero simply be zero, but instead having a slight slope in the negative range. This in turn modifies the outputs to also return slight negative values contrary to just zeros. The functions are expressed as follows (Sharma et al., 2020):

$$ReLU: f(x) = max(0, x)$$
$$Leaky\ ReLU: f(x) = max(0.1x, x)$$

There are a lot more functions that could be considered and the ones mentioned could be discussed to a greater extent. Nonetheless, this thesis limits itself to these activation functions and any further exploration of these activation functions will deviate from the primary objective of this thesis.

The biggest advantage of ANNs are the dynamic learning abilities constituted in the building blocks of the network. These building blocks are continuously optimized to recognize complex patterns that even experts have difficulties perceiving. The highly adaptable nature of ANNs make them useful in a whole array of research fields and is today regarded as one of the most useful tools in machine learning and artificial intelligence.

2.3.4 Support Vector Machine

The Support Vector Machine, or abbreviated as SVM, is a supervised machine learning algorithm used for classification and regression analysis. This thesis is working with a classification issue, and therefore it will perform a discriminative classification. SVM will identify a line or curve (in two dimensions) or hyperplane (in multiple dimensions) that separates the classes from one another. As opposed to generative classification models like Naive Bayes Classification (Vanderplas, 2016, paragraph 4).

When the line can be used in 2-dimensional data it is called linear SVM. If the model can't separate the classes linearly, then it uses nonlinear SVM. In some cases there are several possible dividing lines that can perfectly separate the observed classes. Therefore, the algorithm uses a maximum margin estimator. The maximum margin estimator locates the support vectors, and maximizes the threshold separating the classes. The parallel support vectors are based on the two extreme data points that represent their separate classes and are nearest to one another. (Liu Q et al., 2008). See the Appendix and figure 1.1 for a

visualization. The Support Vector Machine's mathematical logic for maximizing the margin is based on advanced vector calculations. Mainly, the concept is built on the dot-product of two vectors. This thesis won't explore the mathematics of this concept in order to keep the explanation understandable.

The SVM and maximum margin estimation has insensitivity to all data points besides the support vectors. It benefits from this, disregarding the outliers far from the critical separation line. A drawback of this is the possibility of the observation centers being close to each other and the data hard to separate. The bias-variance tradeoff described above applies here for choosing a threshold. "*C*" is the parameter which describes how soft or hard the threshold will be, referring to whether or not it allows data points to reside inside of the margin area. Often the misclassification or error in the model can be represented as: .

$$SVM\ Error = Margin\ Error + Classification\ Error$$

(Saini, 2021, paragraph 22)

The "C" parameter is shown by the last term, "Classification Error". Increasing "C" means that the model focuses less on "Margin Error", creating a harder margin, and points cannot be observed in it. For a lower value of "C", the margin is softer, and is expanded to allow some points to lie in it. This makes the correct "C" value critical for a SVM to properly classify the observations. It's common to use a form of cross-validation to find the optimal parameter value. With respect to this thesis, it is ideal with a softer margin allowing for some misclassifications (Saini, 2021, paragraph 20). GridSearchCV tests all potential combinations of a given dictionary's values using the Cross-Validation method, then examines the model for each one. Determining the accuracy for each set of hyperparameters and selecting the one that performs the best. The use of GridSearchCV allow for optimization of both the "C" and γ (Gamma) variable.

γ (Gamma) is a hyperparameter which controls how much weight is given to data points that are located a specific distance from the Maximum Margin Hyperplane. Nearby points will be taken into account if the Gamma value is high. Furthermore, distant points will be a relevant factor if the Gamma value is low. Intuitively leading to the next SVM subject, utilizing kernel functions.

By using kernel functions, SVM can simplify the classification problem by transforming data from a lower-dimensional space to a higher-dimensional space. In this new space, the relationship between each pair of points is calculated using various vectors that assume the points are in a higher dimension, without explicitly considering the transformation. In layman's terms the kernel provides the dot-product of the vectors in a higher dimensional space. This means the kernel helps decide the classification of a new observation by taking the transformation of a sample vector and dotting it with another vector sample. The optimization problem of maximizing the margin only depends on the dot product of pairs of samples. This math can be summarized with the function:

$$K(x_i, x_j) = \phi(x_i) \cdot \phi(x_j)$$

$\phi(x) =$ The transformation of vector x
x_i or x_j represents two different vector samples.

These kernel functions accept inputs in the original lower dimensional space and return the dot product of the transformed vectors in the higher dimensional space. This approach reduces the complexity of the classification problem and enables the algorithm to work more efficiently, which is often referred to as the *kernel trick*. (Vanderplas, 2016, paragraph 12). In the end, this results in the classes being linearly separable, and by extension can fit a Maximum Margin Hyperplane, allowing the model to make predictions on new observations.

There are several different types of kernel functions within the SVM, the linear kernel and polynomial kernel being two of the most prominent. However, the model used in this thesis relies on the radial basis function kernel, abbreviated RBF. This kernel is common in SVM models, due to its high efficiency in solving non-linear SVM classifications and its similarities to the gaussian distribution. Most realistic classification tasks involve data that is not linearly separable. The RBF kernel finds decision boundaries in infinite dimensions. It behaves similarly to a Weighted Nearest Neighbor model. In other words, the closest observations have a lot of influence on how to classify the new observation and similarly observations further away have little influence on the classification. The RBF Kernel determines each observation's influence on new observations using this formula, producing what is called a high dimensional relationship:

$$e^{-\gamma(a-b)^2}$$

The formula represents the influence an observation a has on b. This influence is measured by squaring the distance and multiplying by the parameter gamma, as explained above, thus scaling the influence. Using this method, the further the data points are from each other the less influence they provide on new observations. The gamma parameter is important to avoid overfitting, this often occurs with higher gamma values.

The RBF kernel bases itself on the polynomial kernels parameters:

$$(a * b + r)^d$$

Defining $r = 0$ and $d = \infty$, leaves the dot product with coordinates for an infinite number of dimensions. In essence this is how the RBF calculates the maximum margin, the actual mathematics is far more expansive yet unnecessary to expand upon in this thesis.

3. Data

3.1 Data collection

The dataset used in this thesis is collected from the website *kaggle.com*.
Kaggle serves as a platform for data scientists and machine learning experts to collaborate and work together on data science projects. Users can access and exchange datasets, develop and test models in a web-based data science environment, as well as engage with other professionals in data science competitions (Uslu A, 2022, paragraph 1).
Prior to the cleaning and transformation of the variables, the dataset contained 45 211 rows and 17 columns. One benefit of using Kaggle datasets is that they frequently come with ready-to-use data. Particular focus should be given to data cleaning before analysis. It's not always necessary to handle null values and other incorrect values. The data handling and cleaning had already been completed for us (Rathi P, 2021).

3.2 Data introduction

There are many strategies to reach customers in the market in order to draw them in and convince them to make a bank term deposit. The research in this thesis is based on a Portuguese marketing effort relating to the subscription of bank term deposits. The thesis' objective is finding the best statistical model that locates potential clients willing to sign up

for a bank term deposit. The major application of such a model is to improve campaign efficiency by identifying the key success factors. This includes assisting in better resource management (human effort, number of contacts, time, etc.), and choosing a high quality and affordable group of potential customers. The outcome would be large and predictable revenue for the banks.

The information was gathered from a Portuguese bank that conducted targeted marketing efforts using its own contact center. The primary marketing channel was telephone, with a human agent acting as the interlocutor, while there were occasionally supplemental uses of the Internet's online banking channel (e.g. by showing information to specific targeted client). Additionally, each campaign was controlled in an integrated manner, and the output for every channel was combined. The dataset was gathered for 17 campaigns, totaling 79354 encounters, that took place between May 2008 and November 2010. An appealing long-term deposit application with competitive interest rates was made available during these telephone advertisements (Moro et al., 2012).

3.3 Understanding the macroeconomic and financial situation

As described above, the data is collected in the aftermath of The Great Financial Crisis of 2008. One of the areas that was heavily impacted by the crisis was the banking sector, where many of the largest and most well-established financial institutions faced significant challenges. The loss of deposits from the banking system is one of the most important and catastrophic effects of banking crises. Bank failures cause a systemic decline in deposits in the absence of deposit insurance or any other program aimed at restoring confidence in a country’s banking system. This is true not only because failed institutions lose their deposits (temporarily or permanently), but also because people may lose faith in the banking system as a whole. The public consequently decide to shift their portfolios of liquid assets away from the banking industry and toward a more basic method of saving. Understanding the macroeconomic and financial situation in the world during the collection of this data, is quite important in order to interpret the results the machine learning models produce.
Before analyzing the dataset with different types of statistical models, introducing the data’s context provides information to better interpret the results.

3.4 Introduction to the variables

In this section of the thesis we will introduce and explain all the different variables used in our assignment, both the target variable and the explanatory variables. We are primarily dealing with three types of variables, binary string, categorical and numeric variables. All binary string variables have been converted to a [0:1] format where 1=yes and 0=no. Our categorical variables have been transformed, where each category has its own binary variable, using the formatting explained above. The numerical values remain unchanged. Below we present the list of variables and explanation of their measurement:

"y" or Deposit	A binary variable and this thesis' target variable. It indicates if the client subscribed to a long-term bank deposit or not.
Age	A numeric variable, scaling from the value of 18 to 95 years old.
Job	A categorical variable collecting the customers job situation. Split in to the following binary variable categories; Admin, Unknown, Unemployed, Management, Housemaid, Entrepreneur, Student, Blue-Collar, Self-Employed, Retired, Technician, Services
Martial	A categorical variable collecting the customers marital status. Split in to the following binary variable categories; Married, Divorced, Single
Education	A categorical variable collecting the customers education level. Split in to the following binary variable categories; Primary, Secondary, Tertiary, Unknown
Default	A binary variable indicates if the client has credit in default or not.
Balance	A numeric value measuring how much the client has in average, annual balance. The currency ranges from -8019 euros up to 102 127 euros.
Housing	A binary value and indicates if the client has a house loan or not.
Loan	A binary value and indicates if the client have a personal loan or not.
Contact	A categorical variable labeling the type of communication with the client. Split into the following binary variable categories; Unknown, Telephone and Cellular

Day	A numerical variable measuring the last contact with the client, ranging from the first day in the month up to 31 days.
Duration	A numerical variable measuring the length of the call, expressed as a number, and measured in seconds. Ranges from 0 up to 4918 seconds.
Campaign	A numerical variable measuring the number of contacts performed during this campaign and for this client. The value ranges from 1 up to 63.
Pdays	A numerical variable measuring the number of days passed after the client was last contacted from a previous campaign. The value ranges from -1 to 871. (-1 means the client was not previously contacted.)
Previous	A numerical variable that represents the number of contacts performed before this campaign and for this client, it ranges from 0 to 275.
Poutcome	A categorical variable that represents the outcome of the previous marketing campaign. Split into the following binary variable categories; Success, Other, Failure, Unknown

Table 2: Table of variables.

4. Data Preparation

4.1 Handling categorical and string variables

Several of the data's explanatory variables are categorical or have a binary-string value, as we've shown above. They all share the attribute of being an "object" datatype. It is necessary to transform these object-values into integers so they can be used in machine learning models. We applied two methods to solve this problem: "get_dummies()" from the pandas library and "LabelEncoder" from the sklearn library. This resulted in several explanatory variables labeled "dummy variables", and these have 0 and 1 as their value. The dummy variable is a variable used in a regression model using qualitative predictor variables to get an estimator. The task at hand is to find the optimal solution for converting datatype variables into integers, and at the same time avoiding the Dummy Variable trap. This trap occurs when two or more dummy variables created by one-hot encoding are highly correlated. This means that one variable can be predicted from the others, making it difficult to interpret the

predicted coefficient variables (Karabiber F, 2020, paragraph 1). This can be demonstrated quite easily, consider the target variable “Deposit” with three values:

$$Deposit = [1, 0, 1]$$

Using the “get_dummies” function on this example would result in:

$$Deposit_{yes} = [1, 0, 1] \; and \; Deposit_{no} = [0, 1, 0]$$

This is resulting in two multicollinear dummy variables. Therefore, all binary-string variables have been formatted using “LabelEncoder”, resulting in:

$$Deposit = [Yes, No, Yes] \Rightarrow Deposit = [1, 0, 1]$$

With categorical variables the “LabelEncoder " would result in several numerical values. This would confuse the models, believing a higher value is worth more. Therefore, by applying the “get_dummies” function to all categorical variables and enabling “drop_first” the model avoids multicollinearity and generates several new binary variables relating to the variables previous string categories. With data handling have we mostly used the pandas library with additional functions from scikit-learn. In creating the regression models the packages scikit-learn were used for all but the neural network model, where the Tenser Flow framework and Keras packages were utilized. After the data handling the final dataset contained 45 211 rows and 43 columns.

4.2 Dataset resampling

Due to the imbalance in the variable Deposit as discussed in chapter 2.2 Accuracy Paradox, all the machine learning methods will be applied on four different datasets. There are three training sets and one universal test set. The original dataset was split into a train and test set, with a ratio of 80% and 20% respectively. The test set is universal and will be used to test the models which have been built up by the three following training sets: original, undersampled and oversampled (Shelke et al., 2017). The test set contains 9,043 unique observations.

The train set created above is derived from the original data and will undergo no further changes. This set has 36,168 unique observations and will be the base for the creation of the next two train sets. The undersampled train set contains all instances of the minority class from the original train set and an equal amount of random majority class observations. This set contains a total of 8,490 unique observations. The third and final train set is the oversampled one. It contains all instances of the majority class from the original train set and

has an equal amount of 1s. Since there are a lot more 0s than 1s, the 1s had to be replicated about 7.2 times to get a balanced amount of 0s and 1s. This set has 63,846 observations, where all the 0s are unique, but the 1s are replicated several times.

There are several ways of resampling datasets to adjust for imbalanced observations. The two methods chosen above are commonly used and acknowledged. Both methods have the intention of dealing with the problem of imbalance, and they are fairly easy to apply. There are several other ways of resampling which might have the ability to generate better results. However, the random sampling methods were chosen due to their ability to handle imbalanced datasets and their low grade of complexity.

5. Empirical results and discussion

5.1 Ranking criteria's

The statistical methods perform differently compared to each other, and even more so with the different types of datasets. Therefore, it is of interest to set some criteria's which makes it easier to evaluate how each model performs on the different datasets.
First the inference will be considered, this is the general model score on the test set. Then the ratio of FP/(FP+TP) will be considered, the lower the ratio the better. This ratio provides the amount of incorrect prediction of potential customers; potential customer loss, further referred to as PCL-value. From a business point of view it's important to detect potential customers, and avoid classifying them as non-potential.

The second criteria refer to the hierarchical structure of the train sets. In essence, how to prioritize the different datasets given all else being equal. The order is as follows, starting with the highest priority: original, and undersampled & oversampled. The reason for this order is due to the configuration of the datasets. The original train set represents the actual reality and is therefore prioritized higher than the two other sets. The undersampled and oversampled set contains both pros and cons which are evaluated to neutralize each other, and those sets are therefore ranked equally.

One of the strengths for the undersampled set is that it only contains unique values, however, it has fewer observations than the test set. On the other hand, the oversampled set has approximately the opposite characteristics of the undersampled set. It has a lot of observations, but contains many replicated observations of the original minority class. When weighing the benefits and disadvantages up against each other, it's considered that the resampling methods are equally good for the purpose of predicting the binary variable "Deposit".

When evaluating the performance of the models, the two criteria's above will be used for ranking. However, an overall assessment, taking the criteria's and potential other factors into account, will decide the final ranking of the models. The table below shows the results and key figures from each model on the different datasets.

Model	Data Sample Type	Train Scores	Test Scores	Possible Customer Loss (PCL-value)	Bias (Train - Test)
Logistic Regression	Original	90.30%	90.00%	64.77%	0.3%
	Undersampled	82.76%	84.62%	17.89%	-1.86%
	Oversampled	83.10%	83.95%	17.70%	-0.85%
Decision Tree Classifier	Original	90.56%	90.67%	50.25%	-0.11%
	Undersampled	85.41%	83.03%	14.21%	2.38%
	Oversampled	99.98%	87.72%	55.65%	12.26%
Support Vector Machine	Original	92.10%	91.67%	57.62%	0.43%
	Undersampled	85.79%	70.22%	2.83%	15.57%
	Oversampled	89.84%	72.89%	9.50%	16.95%
Artificial Neural Network	Original	91.40%	90.95%	47.72%	0.45%
	Undersampled	86.81%	79.29%	7.29%	7.52%
	Oversampled	89.17%	77.02%	5.11%	12.15%

Table 2: Table of results.

5.2 Binomial Logistic Regression

The binomial logistic regression has proven to be an efficient statistical method to predict the variable “Deposit”. The three different models have all given results above 80% on both the train- and test set. An interesting observation is that all the models have a little bias, and two of them actually have negative biases.

The model on the original dataset has the best general score, followed by the undersampled- and oversampled set. The model for the original set has a high PCL-value in contrast to the other sets which have significantly lower values. Taking this information into consideration and evaluating the structure of the datasets, the assumed best model given the criteria’s is the model for the undersampled dataset. The reason why this model is favored over the original dataset lies in the PCL-value. The difference in the PCL-value is so high that it is assumed to outweigh the difference in test score and the data structure. Furthermore, the undersampled-model is considered to be better than the oversampled-model due to the small positive difference in the general score.

The favored binomial logistic regression model is the one fitted for the undersampled dataset. The model generates a test score of 84.62% and a PCL-value of 17.89%.

5.3 Decision Tree Classifier

This paragraph takes a closer look at the results for the decision tree classifier. Generally the model produced good results only generating scores above 80%. The results and reflection are only on the pruned or optimized models. As mentioned, the decision trees have a pattern of producing an overfit model and therefore have higher bias. In a real world scenario the unpruned results are mostly irrelevant, they provide some indication and result, yet hardly the optimal solution, therefore they are excluded.

The decision tree has provided several interesting results. If you purely look at the general test scores, the original data outperforms the others by some margin. The oversampled model

follows, leaving the undersampled model with the lowest score. Looking only at the PCL-values there are varying scores. The undersampled model performs well at classifying the minority class. The other two models perform poorly with this metric, this is expected for the original model. However, the oversampled model performed worse than expected. This discrepancy could be due to selection fault, or it is due to high sensitivity to pattern change in the model itself. Intuitively when building a decision tree more similar data points will be favorable. When building a decision tree model using replicated data points, the classifier will quickly pick up on the benefits of adapting to the given pattern and correctly classify the data. This gives an extremely accurate, yet highly overfitted model for the training data.

Given the criteria's, the most favorable decision tree classifier is the one fitted for the undersampled dataset. The model produces a general score of 83,03%, a PCL-value at 14.21% and has a relatively low bias. These results demonstrate that the model is well-rounded for classifying the objective.

5.4 Artificial Neural Network

In this section, we will take a closer look at the results of the neural network classifier model. Overall, the model produced results that were satisfactory for the original data. The model was, however, lacking for the resampled data. The implemented model had three layers that were fully connected. The input layer and hidden layer both applied the leaky ReLU activation function. The output layer applied the sigmoid activation function. The number of neurons in each layer was determined by trial and error. The model contains 32 neurons in the input layer, 16 in the hidden layer, and only one in the output layer - due to it being a binary classification task.

The results for the original data were the best in terms of overall accuracy. The model had a training accuracy of 91.40% and was not significantly overfitted, whereas the model on test data had an accuracy of 90.95%. However, the PCL-value was subpar, being 47.72%. For the resampled datasets the accuracy results were substantially worse whereas the model had a test accuracy of 79.29% and 77.02% for the undersampled- and oversampled data respectively. The bias was also high for both resampled, with 7.52% for the undersampled data and 12.15% for the oversampled data. This is an indication that the model overfits its training data, thus decreasing the accuracy on the test data. The PCL-values for the resampled data

were better, attaining scores of 7.29% and 5.11% for the under- and oversampled data respectively.

In terms of accuracy, the ANN model did perform well on the original data and not very well on the resampled data. The ANNs hyperparameters were tested and tweaked based on the original data. Certain tweaks that would have favored either one of the resampled data could perhaps increase the models' performance on these. Nonetheless, the redundancies or loss of information seen in the over- and undersampling respectively would have led to an overall decrease in the models' performance regardless. In light of the overall performance we have to address capability from a business perspective. From this perspective, the oversampled ANN model produced results that had low levels of false discovery.

5.5 Support Vector Machine

The table above illustrates how the performance of each of the SVM models and how the outcomes on the key figures differed. With the training data, all models achieved scores above 85%. On the other hand, the test score and PCL- values were varying when predicting the test data.

The model fitted to the original dataset performs the best with respect to the test scores. This model had an overall score of 91.67%. The oversampled model attains the second best performance with 72.89%, and the undersampled model has the worst score with 70.22%. An interesting point is the small difference in test score between the undersampled and oversampled model. Nonetheless, looking at the PCL-value, the original model performed the poorest. This follows the similar tendencies as the other original models, which have been highly trained to anticipate the majority class, and is therefore sensitive to predicting the minority class. The original model can be discarded because the goal is to improve the model’s ability to predict the minority class. There is very little difference in the outcomes and key metrics between the undersampled and oversampled models. Overall, the model fitted to the undersampled data set should be regarded as the optimal model for this application.

The undersampled model achieved a test score of 70,22%, a PCL-value of 2,83% and has a reasonable level of bias. The hyperparameters for this model were C=5 and Gamma=0.01.

6. Conclusion

The purpose of this thesis has been to identify the best machine learning models to predict bank term deposits from the public while taking real-world implications into account. The data was collected in the aftermath of the Great Financial Crisis of 2008 and includes an array of data from the Portuguese public. The data went through some configuration for optimal model performance in terms of predicting the dependent variable “Deposit”. Furthermore, the data underwent various resampling techniques to address an issue related to imbalance.

Four machine learning models have undergone evaluation during this thesis, these are: binomial logistic regression, decision tree classifier, artificial neural networks and support vector machines. The models have been selected based on their ability to handle various statistical classification tasks. The models vary in complexity and usage with them having advantages and disadvantages which have been discussed throughout the thesis. By applying different models and various data to each model, this thesis sought to find the combination that properly handled specific challenges linked to the data, thus managing to predict “Deposit”.

Upon analyzing the empirical results it was found that the best-performing classifier was the support vector machine model trained on the undersampled data. This model inferred training data to the test data with 70.22% accuracy. Additionally, the possible customer loss was at 2.83%, making any false discoveries rare. Considering the criteria for picking out the optimal model, the SVM model trained on the undersampled data best fulfills most of the desired criteria. Despite a lower accuracy score the model successfully classified the minority class, making it a desirable model for the purpose of this thesis. In the end, the model selection process came down to prioritizing the ability to detect all possible customers, sacrificing certain outreach to uninterested customers. It was important for us that our selected model had the ability to identify most of the minority class, ultimately this became the most important criteria. The selected SVM model has a relatively large bias, this however, is a relatively minor inconvenience and had little impact on model choice. If we attempt to

speculate into future changes in the macroeconomic context, the SVM model provides a good fit. As the trust in banks continues to grow, after the economic crisis, the threshold for signing up to a term deposit is likely to soften. This could lead to more possible customers with similar, but slightly altered variable configurations. Having a possibility to adjust the margin in the future could lead to correctly classifying these new minority members without having to drastically alter the model.

It is important to mention that the conditions applied to this data might not still hold true today. However, the overall generalization of implementing different resampling techniques and machine learning models to accurately predict a binary classification task has been the overall focus of this thesis. The knowledge which can be procured throughout this thesis can therefore be utilized for a multitude of classification tasks.

References:

Belkin M., Hsu D., Ma S., & Mandal S. (2019) *Reconciling modern machine-learning practice and the classical bias-variance trade-off*. Proceeding of the National Academy of Science Volume 116, Issue 32, p. 15849-15854. Available at: https://www.pnas.org/doi/epdf/10.1073/pnas.1903070116 (Retrieved: 07.02.2023)

Bloomenthal, A (2022). *Net interest margin*. Available at: https://www.investopedia.com/terms/n/netinterestmargin.asp (Retrieved: 01.02.2023)

Branco P , Torgo L & Ribeiro R P (2015). *A Survey of Predictive Modelling under Imbalanced Distributions*.Available at: https://arxiv.org/pdf/1505.01658.pdf. (Retrieved: 14.02.2023)

Chen, J.(2022) *Term deposit*. Available at: https://www.investopedia.com/terms/t/termdeposit.asp (Retrieved: 01.02.2023)

Choi, J-A and Limb,K (2020). *Identifying machine learning techniques for classification of target advertising*. Available at: https://reader.elsevier.com/reader/sd/pii/S2405959520301090?token=DBAD53D32F8964E43FAB9B6B6A8298F87082E7F9FBEBD79A6520E9189EFDA0F6588AE1F02F12507065E273A1BD7ABA26&originRegion=eu-west-1&originCreation=20230306161426 (Retreived: 06.03.2022)

Downey, A.B (2014) *Think stats*. Version 2.2. Needham: Green Tea Press. Available at: Think Stats (greenteapress.com) (Retrieved 22.02.2023)

Gordon M. and Kochen M. (1989) Recall-Precision Trade-Off: A Derivation, *Journal of the American Society for Information Science*. 40(3), p. 145-151. Available at: Recall-precision trade-off: A derivation (wiley.com) (Retrieved: 22.02.2023)

Karabiber, F (2020) *Dummy Variable Trap*. Available at: https://www.learndatasci.com/glossary/dummy-variable-trap/. (Retrieved: 15.02.2023)

Liu, Q., He, Q., Shi, Z. (2008). Extreme Support Vector Machine Classifier. In: Washio, T., Suzuki, E., Ting, K.M., Inokuchi, A. (eds) Advances in Knowledge Discovery and Data Mining. PAKDD 2008. Lecture Notes in Computer Science(), vol 5012. Springer, Berlin, Heidelberg. https://doi.org/10.1007/978-3-540-68125-0_21 https://link.springer.com/chapter/10.1007/978-3-540-68125-0_21

Maind S.B., Wankar, P. (2014) Research Paper on Basic Artificial Neural Network, *International Journal on Recent Innovation Trends in Computing and Communication*, 2(1), p. 96-100, Available at: Research_Paper_on_Basic_of_Artificial_Neural_Network-libre.pdf (d1wqtxts1xzle7.cloudfront.net) (Retrieved: 22.02.2023)

Moro S, Laureano R & Cortez P (2012). *Using Data Mining for Bank Direct Marketing: An Application of the Crisp-DM methodology*. Available at: https://core.ac.uk/download/pdf/55616194.pdf. (Retrieved: 02.02.2023)

Nembrini S. & König I. R. & Wright M. N. (2018) *The revival of the Gini importance?* Bioinformatics, Volume 34, p. 3711-3718. Available at: https://doi.org/10.1093/bioinformatics/bty373 (Retrieved: 22.02.2023)

Rathi P (2021). *Banking Dataset- Marketing Targets*. Available at: https://www.kaggle.com/datasets/prakharrathi25/banking-dataset-marketing-targets (Retrieved: 01.02.2023)

RM and Mass marketing: Runar Framnes, Håvard Huse, Arve Pettersen & Hans Mathias Thjømøe (2021), Markedsføringsledelse, Universitetsforlaget, 10. utgave

S. Moro, P. Cortez and P. Rita. A Data-Driven Approach to Predict the Success of Bank Telemarketing. Decision Support Systems, Elsevier, 62:22-31, June 2014. Available at: https://archive.ics.uci.edu/ml/datasets/Bank+Marketing# (Retrieved: 26.02.20223)

Saini, A (2021). *Support Vector Machine (SVM): A Complete Guide for Beginners*. Available at: https://www.analyticsvidhya.com/blog/2021/10/support-vector-machinessvm-a-complete-guide-for-beginners/ (Retrieved: 15.02.2023)

Sharma, S., Sharma, S. and Athaiya A. (2020) Activation functions in neural networks, *International Journal of Engineering Applied Sciences and Technology*, 4(12), p. 310-316. Available at: https://www.ijeast.com/papers/310-316,Tesma412,IJEAST.pdf (Retrieved: 22.02.2023)

Shelke, M.S, Deshmukh, P.R and Shandilya, V.K (2017) A review on Imbalanced Data Handling Using Undersampling and Oversampling Technique, *International Journal of Recent Trends in Engineering & Research*. 03(4), p. 444-449. Available at: a-review-on-imbalanced-data-handling-using-undersampling-and-oversampling-technique.pdf (archive.org) (Retrieved: 22.02.2023)

Swain P. H. & Hauska H.(1977) The decision tree classifier: Design and potential. *IEEE Transactions on Geoscience Electronics*, vol. 15, no. 3, pp. 142-147. Available at: https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=6498972 (Retrieved: 22.02.2023)

The Pennsylvania state university (2023) *Applied Data Mining and Statistical Learning*. Available at: https://online.stat.psu.edu/stat508/lesson/11/11.8/11.8.2 (Retrieved 11.02.2023)

Uslu C (2022). *What is Kaggle?* Available at: https://www.datacamp.com/blog/what-is-kaggle (Retrieved: 05.02.2023)

VanderPlas J. (2016) *The Python Data Science Handbook.* O'Reilly Media. Available at: Python Data Science Handbook | Python Data Science Handbook (jakevdp.github.io) (Retrieved: 26.02.2023)

S. Albawi, T. A. Mohammed and S. Al-Zawi, (2017) Understanding of a convolutional neural network, International Conference on Engineering and Technology (ICET). Available at: https://ieeexplore.ieee.org/document/8308186 (Retrieved 22.02.2023)

P. Zhang, J. Xue, C. Lan, W. Zeng, Z. Gao and N. Zheng (1997) Adding Attentiveness to Neurons in Recurrent Neural Networks | Transactions on Image Processing. Available at: https://ieeexplore.ieee.org/abstract/document/8822600 (Retrieved 22.02.2023)

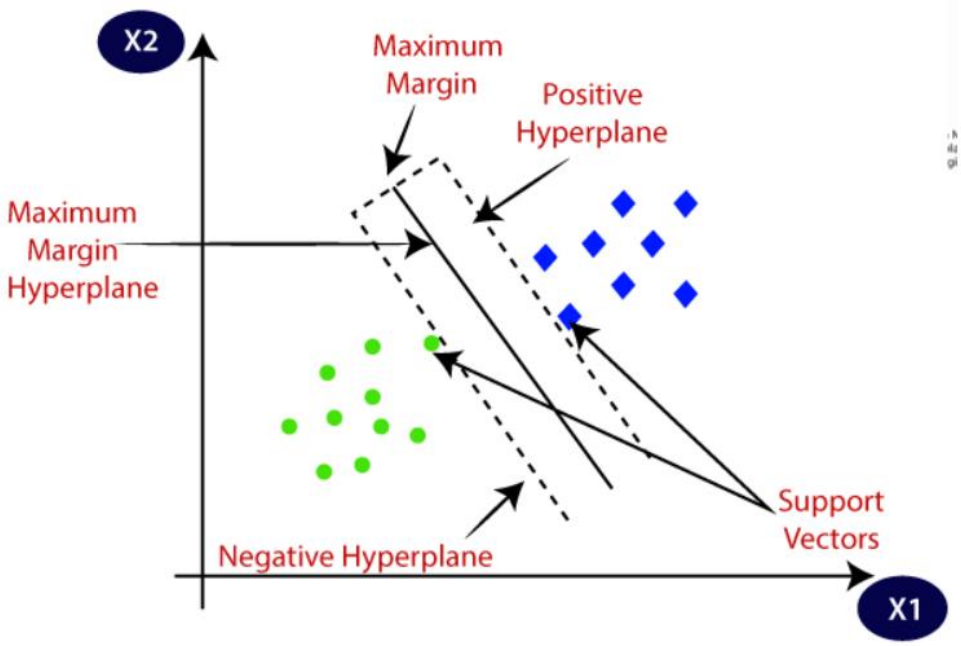
M. N. H. Siddique and M. O. Tokh (2001) Training neural networks: backpropagation vs. genetic algorithms, International Joint Conference on Neural Networks. Available at: https://www.semanticscholar.org/paper/Training-neural-networks%3A-backpropagation-vs.-Siddique-Tokhi/8137de8c31366904bf8206fd6c1db8c2477eb1b7 (Retrieved 22.02.2023)

Ahmad M. Sarhan, and Omar I. Al Helalat (2007) Arabic Character Recognition using Artificial Neural Networks and Statistical Analysis. Available at:

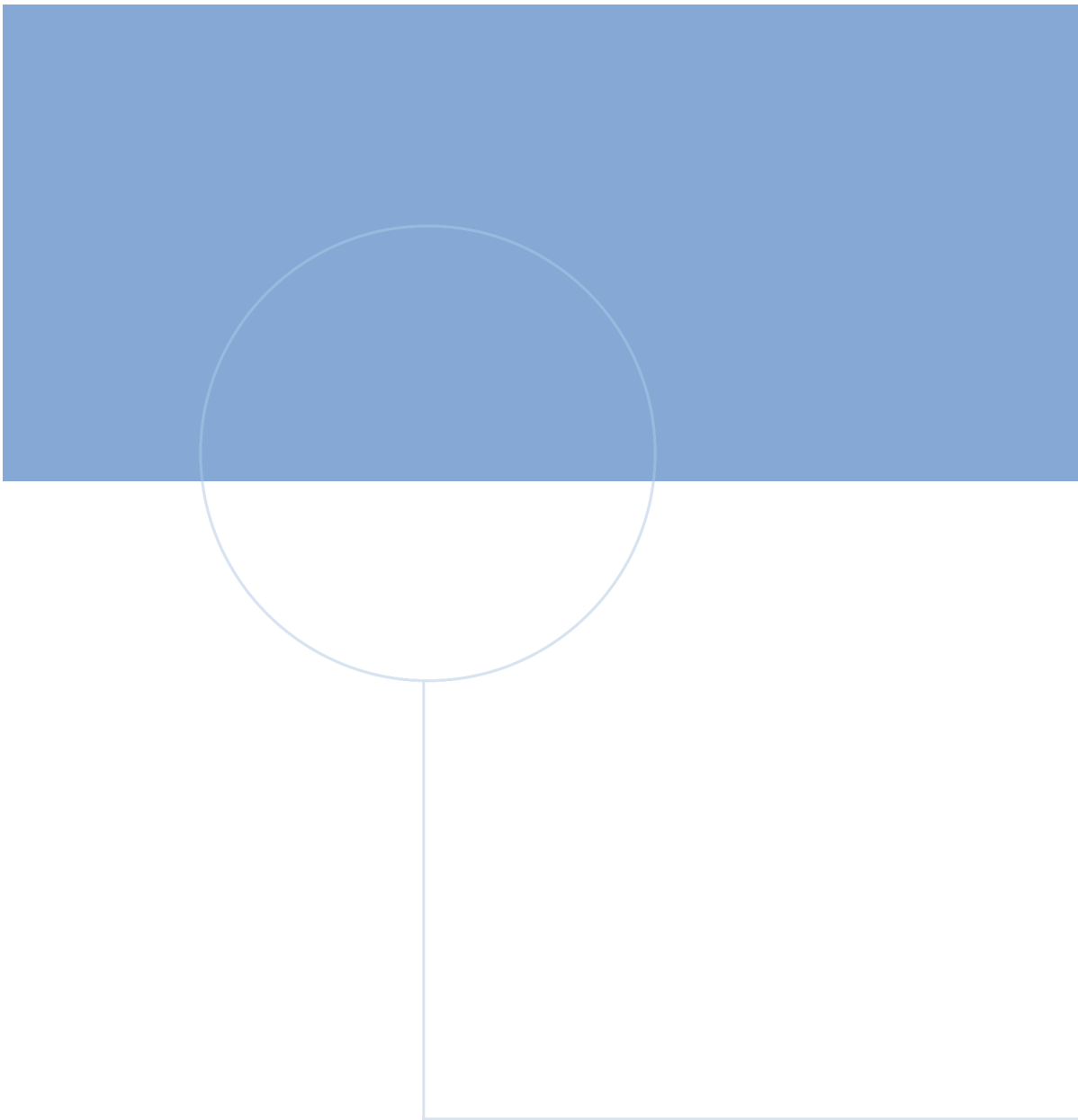
https://citeseerx.ist.psu.edu/document?repid=rep1&type=pdf&doi=f8f61977c9d5071ead8f2d876ead61004f31eb6d (Retrieved 22.02.2023)

1. Appendix

- **Figure 1.1**



Used to visualize the Support Vector Mahine, and how it works.



NTNU

Kunnskap for en bedre verden