DEGREE PROJECT IN COMPUTER SCIENCE AND ENGINEERING,
SPECIALISING IN ICT INNOVATION,
SECOND CYCLE, 30 CREDITS
STOCKHOLM, SWEDEN 2020

Comparing Machine Learning Algorithms and Feature Selection Techniques to Predict Undesired Behavior in Business Processes and Study of AutoML Frameworks

Anushka Garg

KTH ROYAL INSTITUTE OF TECHNOLOGY
ELECTRICAL ENGINEERING AND COMPUTER SCIENCE

Author

Anushka Garg
Master in ICT Innovation - Cloud Computing and Services
School of Electrical Engineering and Computer Science
Stockholm, Sweden
October, 2020

Examiner

Amir Payberah
Stockholm, Sweden

Supervisor

Sina Sheikholeslami
Stockholm, Sweden

Host Company

Agilon Analytics

Stockholm, Sweden

Abstract

In recent years, the scope of Machine Learning algorithms and its techniques are taking up a notch in every industry (for example, recommendation systems, user behavior analytics, financial applications and many more). In practice, they play an important role in utilizing the power of the vast data we currently generate on a daily basis in our digital world.

In this study, we present a comprehensive comparison of different supervised Machine Learning algorithms and feature selection techniques to build a best predictive model as an output. Thus, this predictive model helps companies predict unwanted behavior in their business processes. In addition, we have researched for the automation of all the steps involved (from understanding data to implementing models) in the complete Machine Learning Pipeline, also known as AutoML, and provide a comprehensive survey of the various frameworks introduced in this domain. These frameworks were introduced to solve the problem of CASH (combined algorithm selection and Hyper-parameter optimization), which is basically automation of various pipelines involved in the process of building a Machine Learning predictive model.

Keywords

Machine Learning, AutoML frameworks, Predictive model, Business process management, CASH.

Sammanfattning

Under de senaste åren har omfattningen av maskininlärnings algoritmer och tekniker tagit ett steg i alla branscher (till exempel rekommendationssystem, beteendeanalyser av användare, finansiella applikationer och många fler). I praktiken spelar de en viktig roll för att utnyttja kraften av den enorma mängd data vi för närvarande genererar dagligen i vår digitala värld.

I den här studien presenterar vi en omfattande jämförelse av olika övervakade maskininlärnings algoritmer och funktionsvalstekniker för att bygga en bästa förutsägbar modell som en utgång. Således hjälper denna förutsägbara modell företag att förutsäga oönskat beteende i sina affärsprocesser. Dessutom har vi undersökt automatiseringen av alla inblandade steg (från att förstå data till implementeringsmodeller) i den fullständiga maskininlärning rörledningen, även känd som AutoML, och tillhandahåller en omfattande undersökning av de olika ramarna som introducerats i denna domän. Dessa ramar introducerades för att lösa problemet med CASH (kombinerat algoritmval och optimering av Hyper-parameter), vilket i grunden är automatisering av olika rörledningar som är inblandade i processen att bygga en förutsägbar modell för maskininlärning.

Nyckelord

Machine Learning, AutoML frameworks, Predictive model, Business process management, CASH.

Acknowledgements

I would like to express my sincere gratitude to my examiner Prof. Amir Payberah and my supervisor Sina Sheikholeslami for their guidance and support. The completion of this thesis would not have been possible without their valuable guidance. I would also like to thank KTH and Agilon Analytics for providing me with an opportunity to learn and conduct this research.

This thesis marks the end of my degree programme as master's in ICT innovation with a focus on Cloud computing and services at KTH, Royal Institute of Technology in Stockholm, Sweden and Technische Universität Berlin in Berlin, Germany.

Finally, a special thanks goes to my family, friends, and brother Mohit Garg who supported me throughout the journey and kept me motivated.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In recent years, Machine Learning [1] has conquered the tech industry. From the 1990's spam filter to the branding of deep learning or automated Machine Learning. The enthusiasm for Machine Learning applications has kept on increasing, decades after decades. Today, it is at the heart of much of the magic in today's high-tech products, speech recognition, personalised recommendations, personal assistant, and beating the world champion at the game of Go [2].

"The difficulty of beginning will be nothing to the difficulty of knowing how to stop."

— Agatha Christie, Murder in Mesopotamia (1936)

The concept of Machine Learning started years ago which has become an unstoppable learning since then. It is the science (and art) of programming computers so they can learn from data [2]. Machine Learning is the foundation of many recent developments and innovations. This extraordinary success has led to a lot of research activities that aim to develop more accurate and better algorithms and models for many more applications. As a result, all over the world researchers are engaged in an iterative process of improving the performance of the new Machine Learning algorithms. This iterative process results in ending up with many trials where each trial is different from the other one with regards to the configuration of the model, Hyper-parameters, and subsets of features among the other things.

Feature selection and Hyper-parameter tuning are both good examples of the Machine Learning research experiments in order to receive a better performance of the model.

Solving the problem of CASH (combined algorithm selection and Hyper-parameter optimization) [3] is also a Machine Learning advancement, which resulted in great success.

1.1 Problem

A lot of work has already been performed in the area of predictive process monitoring and more specifically in prediction of undesired behavior. The thesis focuses on finding out a solution for the company to help predicting the undesired behavior for multiple response/target variables. There is a predictive model but to improve its performance some tuning is required like implementation of optimization of Hyper-parameters of the algorithm and to choose a subset of the best features from the whole feature space. There are many feature selection techniques present but the thesis focuses on finding out the best one out of them, which gives a feature subset that results in giving the highest accuracy of the model.

Also, building a Machine Learning predictive model involves a lot of complex steps/pipelines. For example, data understanding, selection of algorithms, optimizing Hyper-parameters, model deployment, etc. The automation of all these steps is known as AutoML, which can also be referred as a solution to CASH (combined algorithm selection and Hyper-parameter optimization) problem. Various AutoML frameworks are introduced in recent years to solve the CASH problem. But not all of these have the same technique for Hyper-parameter optimization or a same set of Machine Learning algorithms in the search space. Therefore, another major focus of the thesis is to understand the pros and cons of these AutoML systems. To be more specific, to understand, which optimization techniques are used by them, their strategy for exploring and traversing the search space, configuration parameters of each one of them and the results on a given dataset.

1.2 Research goals

The thesis focuses on the following research goals:

- **Finding the most suitable supervised Machine Learning algorithm with optimized Hyper-parameters to make a predictive model for**

multiple target variables.
One of the main parts of the thesis is to compare different supervised Machine Learning algorithms and to find the best one out of them, to make a most efficient predictive model for multiple target values. So that the company can use the model to predict the undesired activity occurrence in the business process and also to provide quantified improvement suggestions, which will help the company to take actions in advance that can result in growth of the business. For example- identifying orders at risk of being delayed or customer invoices unlikely to be paid, etc. (for a dataset of product delivery companies).

- **Compare different AutoML platforms for solving CASH (combined algorithm selection and Hyper-parameter optimization) problem.**
  CASH [3] is a well known problem which is about automating the Machine Learning pipeline. Basically AutoML - to answer questions like, which Machine Learning algorithm and what parameterization of it, etc.? There are many frameworks present to automate the process of CASH. In this thesis, some of these frameworks are studied in detail and have drawn a detailed comparison amongst them.

- **Compare different techniques for feature selection and see how they impact on the accuracy of the models.**
  Feature selection is one of the most important preprocessing steps to generate a highly accurate predictive model. There are various feature selection techniques present, some of them are compared here on the basis of evaluation metric (accuracy of the model). The best one amongst these is then used to build the predictive model and to provide feature importance for the company dataset.

1.3 Objectives

Agilon Analytics [4], is a startup who:

- Visualizes process flows and finds deviations and bottlenecks.
- See their impact on key metrics.
- Provide quantified improvement suggestions.
- Get actionable insights on workflow improvements.

They visualize business processes to make them transparent, fast, and cost effective. The objective of this master thesis which is sponsored by Agilon, is to apply Machine Learning in their business process by making predictions of an undesired activity in business process cases with good results based on the chosen evaluation metric. To achieve this objective, supervised Machine Learning algorithms for the given dataset are investigated considering the challenge of the imbalanced dataset after applying the data preprocessing and the feature engineering part. The results of this study will be integrated into Agilon.

1.4 Methodology

The thesis work has been carried out by following the empirical method, as various Machine Learning algorithms are tested on a real–world business process dataset provided by Agilon. The proposed research goals include finding the suitable supervised Machine Learning algorithm and feature selection technique amongst various based on the evaluation metric. Therefore, the “quantitative” data analysis method is applied to measure the performance of different Machine Learning algorithms and feature selection techniques in Chapter 3.

1.5 Ethics and Sustainability

As the dataset used in this thesis is not publicly available, one of the main concerns is data security. In this thesis, the author has focused on keeping the company’s terms and conditions by keeping their user’s identities secure, generally by using anonymized user and aggregated data for all the compared studies, results or other business goals. Therefore, there is nothing unethical in this study.

In regards to sustainability, one can make a business process more efficient by detecting potential undesired behavior in advance, which will save time and resources both. Also, the author’s concern to use the sensitive information as less as possible made her understand how the different subset of features of datasets affects the performance of a Machine Learning model which leads to the elimination of features that are not so crucial for the model. Hence, use what is really required.

1.6 Outline

The thesis is organized as follows: First, relevant theories and background necessary for understanding the problem of the thesis are discussed in Chapter 2. Then, the thesis implementation is elaborated in Chapter 3. The results and discussion which explains how the solution satisfies the goals of this project are covered in Chapter 4. Finally, in Chapter 5 potential future work alongside the conclusion is presented.

Chapter 2

Background

2.1 Machine Learning

In Machine Learning, an algorithm is fed by the input and output data, through which a program is created which can generate powerful insights that can be used to predict future outcomes and perform several kinds of complex decision making. Machine Learning gives us an opportunity to get out of previous old rule based systems like.

If $(a = b)$ *then do* c

"Traditionally, software engineering combined human created rules with data to create answers to a problem. Instead, Machine Learning uses data and answers to discover the rules behind a problem"

- Chollet, 2017

In today's world, Machine Learning is being used in the development of various things. It is so prevalent that even we use it a couple of times in a day unknowingly, like predictions, image recognition's, speech recognition's and many more. To learn all these a Machine has to go through a learning process, they have to try different algorithms, rules and learn from their performance. That is why, it is called Machine Learning [5].

Machine Learning relies on some formatted inputs and it provides a result based on the task. The input format is specific to the type of Machine Learning technique used and also to the specific algorithm. This specific representation of input data is known as features or predictors.

Categories of Machine Learning Algorithms:

- **Supervised Learning** - It can be considered as the analogy to human learning under the guidance of a teacher.

  A analogical example of Machine Learning process is - we learn from teacher's teachings (training) $\rightarrow$ then we are tested to analyze our knowledge (testing) $\rightarrow$, which decide our destiny (evaluation) $\rightarrow$ this test is based on some threshold to pass (baseline) $\rightarrow$ scores/results determine whether we have to learn more or can move to next level (deployment). The words mentioned in brackets are the common Machine Learning professional terminology.

  The above example is exactly what happens in supervised Machine Learning. In this approach we are given labelled input (called features/predictors) and outputs (called target/response attribute), and the goal is to learn the mappings (rules) between them. The main objective is to learn and identify the patterns that can accurately predict the outcome for the new independent data.

- **Unsupervised Learning** - This can be considered as the analogy to human learning from experience. In this approach there is no target attribute i.e., there is no label output to aim for. The main objective is to identify patterns by deducing structures and relations of the input features in the provided dataset. It can be used to discover rules that collectively define a group such as partitioning (clustering) [6]. A simple example can be sorting different colour coins into separate piles. Just by looking at their features i.e. colour they can be clustered into their correct groups [5] [1].

- **Semi supervised Learning** - Semi-supervised learning is a combination of supervised and unsupervised learning approaches. It deals with partially labelled data. The learning process is not closely supervised with target outputs for each and every input, but it also doesn't let the algorithm do its own individual thing and provides no form of feedback. Semi-supervised learning acts as the intermediate. One of the most common examples of this approach can be some photo hosting services such as google photos [5].

- **Reinforcement Learning** - This approach is the one where the Machine learns by itself using rewards, does not use any labels as explained in the above learning techniques. The learning system (a.k.a. agent) has to get the maximum rewards over time by learning the best strategy called policy. It observes the environment, chooses and performs actions and gets rewards in return. There are positive rewards for the best policy and negative rewards for penalties. The main key in the reinforcement learning is its reward motivated nature [5] [1].

2.1.1 Traditional Machine Learning models

This thesis is majorly focused on supervised Machine Learning. Following are some classifiers algorithms:

- **Naive Bayes classifier** - Naive Bayes is the simplest Bayesian classifier and is a probabilistic model. As the name suggest, it is based on *Bayes theorem*

  $$P(A|B) = \frac{P(A)P(B|A)}{P(B)} \tag{2.1}$$

  The classifier is based on Bayes assumption of features being independent, i.e. the presence of one feature does not affect the other. For example : Assume a training set (x^i, y^i) for n samples, where each x^i is a d dimensional feature vector and each y^i is the class label where superscript is used to index training examples {i = 1,...,n}. In general, y is a discrete variable that falls into exactly one of K possible classes $\{C_k\}$ for $\{k \in 1, ..., K\}$, and the features of {x1,...,xd} can be any discrete or continuous attributes [7].

  According to the above mentioned Naive assumption of features being independent of each other. This implies that the probability of any instance belonging to class with features {x1,...,xd} corresponds to the probability of class $\{C_k\}$ multiplied by the probability of the existence of each feature in this class [8].

  $$P(ck|x1, ..., xd) = \frac{P(Ck)P(x1, ..., xd|Ck)}{P(x1, ..., xd)} = \frac{P(Ck)\prod_{j=1}^{d} P(xj|Ck)}{\prod_{j=1}^{d} P(xj)} \tag{2.2}$$

  $$P(ck|x1, ..., xd) \propto P(Ck)\prod_{j=1}^{d} P(xj|Ck) \tag{2.3}$$

  Naive Bayes is basic, fast and easy to execute, although feature independence seems to be a disadvantage for the classifier. Though it competes well amongst the other classifiers. It is usually used for recommendation systems, spam filtering etc.

- **Logistic Regression classifier** Logistic Regression (a.k.a. Logit Regression) is quite similar to linear regression model calculates the weighted sum of the input features including the bias term but rather than displaying the result directly, it shows the logistic of the result. Logistic Regression is commonly used to estimate the probability that an instance belongs to a particular class, which makes it suitable for binary classification tasks. If the estimated probability is more than 50% then the model belongs to that class and predicts 1 (positive) and otherwise it does not, then the model predicts 0 (negative). In this model, input features are first normalized to lie between 1 and 0.

$$\hat{p} = h_\theta(x) = (x^T\theta) \tag{2.4}$$

Where σ is the sigmoid function (defined in the below equation), displaying numbers between 1 and 0 and h$_\theta$(x) being the estimated model probability.

$$\sigma(t) = \frac{1}{1 + exp^{-t}} \tag{2.5}$$

So in logistic regression classifier if x$^T\theta$ positive the model predicts 1 otherwise 0 when it is negative.

- **Decision Tree classifier** - Decision trees are all purpose Machine Learning algorithms that can execute both classification and regression tasks. They make decisions or predict the target class using basic decision rules. The main objective of decision trees is to try to recursively split the training data so as to maximize the discrimination among the different classes over different nodes [7]. The algorithm uses the input feature set to split the training dataset into two subsets and constructs a binary tree so that the information gain can be maximized at each split. CART (Classification And Regression Tree), an implementation of decision tree is used by Scikit learn which implements both classification and regression. An example of a decision tree (built using the scikit-learn library in python [9]) is shown in (Figure 2.1.1). This example was built on the Iris flower dataset, which is publicly available [10]. As shown in this figure, the analysis of this classifier is simple and it interprets a set of rules leading to the defined target class [8].

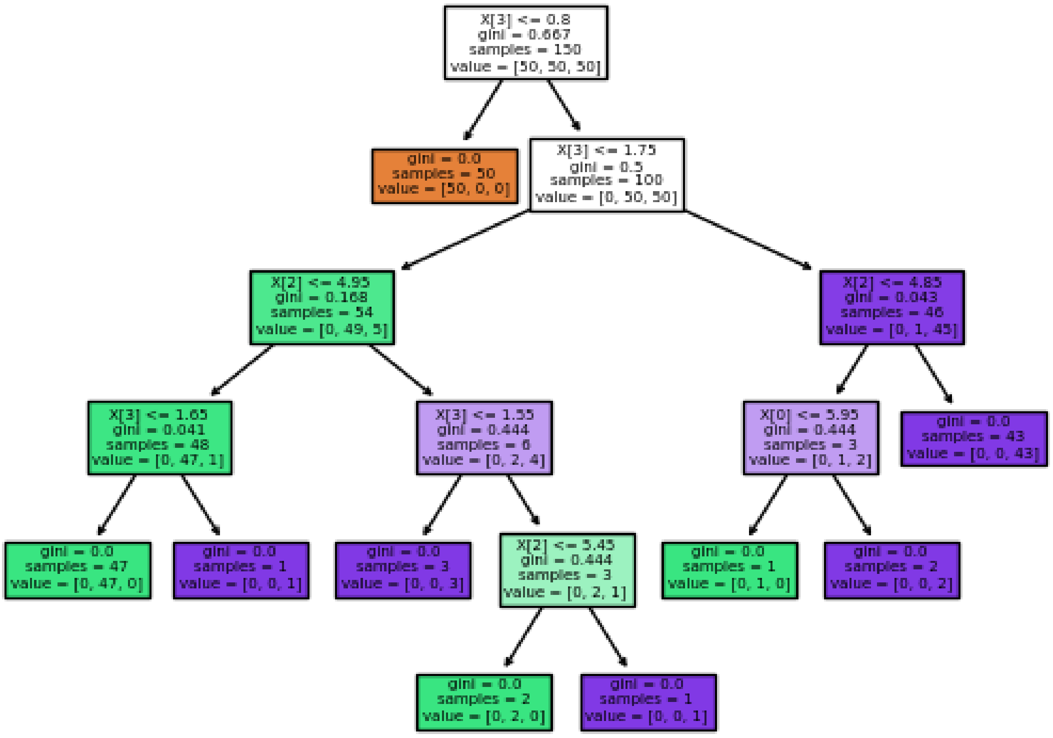


Figure 2.1.1: Decision Tree Classifier [11]

- **Random Forest classifier** - Aggregation of predictions from different groups of predictors (classifiers) could lead to better results. This is the basic idea of ensemble learning - to combine different classifiers of the same type. Random Forest [12] is a type of ensemble method that combines various decision trees to enhance the results. Firstly, the algorithm trains n different decision trees classifiers, each on random subsets of the training set. After that a best feature is selected amongst the random set of features to split (which introduces extra randomness [2]) and build a decision tree accordingly. Later these n different trees give n different predictions for each instance (class probabilities in case of classification). At last, final value prediction for each instance, is the class with highest votes. Random Forest is one of the most powerful Machine Learning algorithms present today [2].

- **K-Nearest Neighbors classifier** - KNeighbor classifier supports classification and regression tasks and is also suitable for multi label classification. A system which can output multiple binary tags is known as a multi label classification system [2]. The algorithm uses proximity as a proxy for likeness. The basic idea of this algorithm is to classify an instance based on the majority vote of its neighbors, with the instance being assigned to the class, which is most common among its k nearest neighbors where k is a positive integer specified by the user. It is a type of instance-based learning or non-generalizing learning. That is it does not try to build a general internal model, but simply stores instances of the training data. Classification is performed from a simple majority vote of the k nearest neighbors of each point [13].

- **SVM classifier** - SVM (Support vector Machine) is also used for classification and regression tasks. The basic idea of SVM is to use linear conditions to separate the two classes from one another as efficiently as possible [7]. SVM algorithm tries to find a hyper-plane in an N-dimensional space, where N is the number of features that uniquely classifies the data points [14]. This hyper-plane acts as the decision boundary between classes of data points in order to classify them. Our aim is to find a plane with maximum margin, so that the future data points can be classified more accurately. An example of a SVM (built using the scikit-learn library in python [15]) is shown in Figure 2.1.2. This example was built on the Iris flower dataset, which is publicly available [10]. As shown in this figure, the analysis of this classifier is simple and it creates hyper-planes between classes leading to the classification of the data points [8].

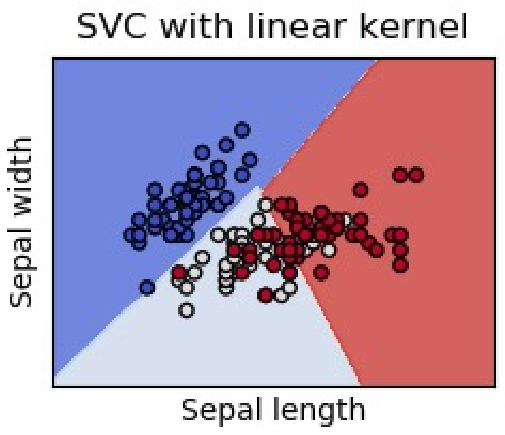


Figure 2.1.2: SVM Classifier [16]

2.1.2 Automated Machine Learning

Nowadays, the exigency of Machine Learning functionality has become a necessity in more and more application areas such as technology, science and society and these Machine Learning applications are already on an urge of becoming part of everyday's life. As it's hard for Machine Learning experts to be a constant support, there comes an urgent requirement in regards with tools that are easy to use. Therefore, AutoML aims to ease the workload and need of Machine Learning experts by automating the whole process of building Machine Learning models.

An AutoML system asks the user to provide the training data as input and it provides back an optimized model as output. AutoML automates all the regularly used steps including data preprocessing, feature preprocessing, selection of model, training and evaluation of a model and optimization of Hyper-parameters to a great extent. This has given rise to the field of automated Machine Learning in the recent years [17] [18].

2.2 Hyper-parameter Tuning

Hyper-parameters are the parameters that changes the performance of the classifier. Classifiers cannot learn Hyper-parameters on their own and therefore should be defined before starting the training process. There are several search methods to find the best Hyper-parameters for each classifier. Few most common and important ones are described below:

- **Grid search**- It is the most basic and traditional way of performing Hyper-parameter optimization. It performs a thorough search within a specific subset of Hyper-parameters. A model is created for each possible combinations of all the provided Hyper-parameter values, evaluating each model and selecting the architecture, which maximizes the target result [19]. Basically, as the name suggests, it creates a grid of configurations and evaluates all of them to find the best set of Hyper-parameters.

- **Random Grid search**- In this method of Hyper-parameter optimization, the values of each Hyper-parameter are chosen randomly independently of all others. This technique is used when the search space is larger. Also, it is straight forward to implement and to parallelize.

- **Bayesian Optimization** The above two techniques performs individual experiments to build models with different Hyper-parameter values and then record the performance of the model for each. They are easy to parallelize as their performance is isolated. However, because of this isolation, the information from one experiment can not be used to improve the next experiment. Bayesian optimization belongs to a class of sequential model-based optimization (SMBO) algorithms that improve the sampling method of the next experiment using the results of the previous iteration. Basically, it uses past evaluation results to select the next Hyper-parameter values of to evaluate. Due to this reason, this technique requires less number of iterations to get to the optimal set of Hyper-parameter values.

2.3 Automated Machine Learning Frameworks for CASH problem

In this digital era, there has been a continual and infinite increment in the amount of data. It has been acknowledged that it is tough for the present number of data experts to tackle the challenges, which occur due to this huge increment in the amount of data. Thus, there was a significant requirement for automating the process of building efficient and good Machine Learning models [17].

In the past couple of years, various techniques and frameworks have been introduced to address the challenges. To be specific, to tackle the challenge of automating the process of CASH (Combined Algorithm Selection and Hyper-parameter optimization) [3] in the Machine Learning domain. The main goal of all these techniques and frameworks is to decrease the work efforts of humans in the loop of building an efficient model while taking the place of a non-expert domain user and playing the role of a Machine Learning expert.

AutoML for CASH problem

Definition *(CASH)* Let $A = \{$A(1),..., A(R)$\}$ be a set of algorithms, and let the Hyper-parameters of each algorithm $A^{(j)}$ have domain $\Lambda^{(j)}$. Further, let $Dtrain = \{$(x1, y1),..., (xn, yn)$\}$ be a training set which is split into K cross-validation folds $\{D^1_{valid}, ..., D^K_{valid}\}$ $and\{D^1_{train}, ..., D^K_{train}\}$ such that $\{D^i_{train} = D_{train} \backslash D^i_{valid}\ \}$ for $\{i = 1, ..., K\ \}$. Finally, let $\{L(A^j_\lambda D^i_{train}, D^i_{valid})\}$ denote the loss that algorithm $A^{(j)}$ achieves on D^i_{valid} when trained on D^i_{train} with Hyper-parameters λ. Then, the Combined Algorithm Selection and Hyper-parameter optimization (CASH) problem is to find the joint algorithm and Hyper-parameter setting that minimizes this loss [3] [18] [20]:

$$A^*, \lambda_* \in \underset{A^{(j)} \in A, \lambda \in \Lambda^{(j)}}{\arg\min} \frac{1}{K} \sum_{i=1}^{K} L(A^j_\lambda D^i_{train}, D^i_{valid}) \tag{2.6}$$

In general, time budget is one of the constraints for CASH optimization techniques. The main purpose of the optimization algorithm is to tune a selected (chosen) Machine Learning algorithm that can perform efficiently, i.e., algorithm which can achieve best performance in terms of evaluation metric defined by the user (for example, accuracy, F1 score, recall, precision, etc.) in the time budget defined by the user for the whole search process [18] as shown in the Figure 2.3.1.

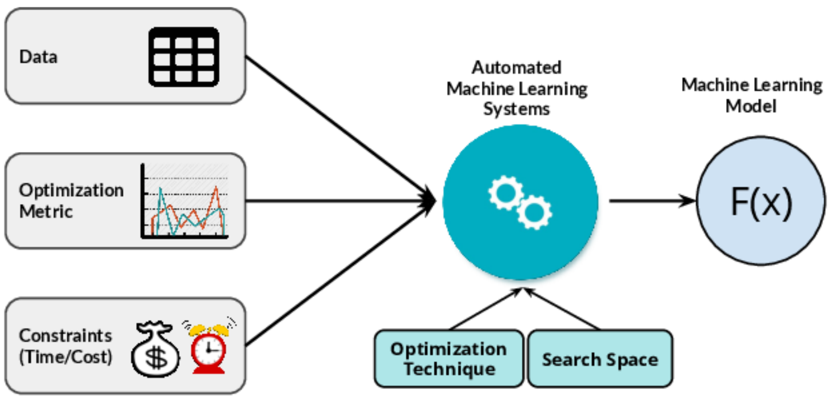


Figure 2.3.1: Workflow of AutoML process [21]

In this research, we present a comprehensive survey of the various frameworks that have been implemented to tackle the CASH problem [18].

- **Auto Sklearn** has been built around Scikit-Learn [22], a commonly used Machine Learning library. Auto-Sklearn provides supervised Machine Learning algorithms. It searches for the best learning algorithm for a dataset and optimizes its Hyper-parameters [23]. It also introduced the idea of meta-learning to speed up the optimization process. With the introduction to meta learning, it also implements ensemble learning in order to give an ensemble of best created pipelines, instead of just returning the best one. It tackles the challenge of CASH by freeing the Machine Learning expert user from algorithm selection and Hyper-parameter tuning.

- **Auto WEKA** was the first system to tackle the CASH problem by using the Machine Learning framework WEKA and is considered as the discoverer Machine Learning automation framework [24]. It was implemented in Java and applies Bayesian optimization. Main aim is to achieve significantly improved performance in the Machine Learning models by helping non experts to find the best learning algorithm and hyper parameters settings regarding their applications.

- **TPOT** The TPOT (Tree-Based Pipeline Optimization Tool) framework is one of the solutions, which has also been built on top of Scikit-Learn and it's own regressor and classifier methods [25]. It is an open source software, implemented in python. TPOT is based on genetic programming, it explores various possible pipelines of learning algorithms and feature engineering. Then, finds the best one out of them to fit the data.

- **H2o AutoML** H2O is an open source and distributed in-memory Machine Learning platform developed by H2O.ai [26] It supports Java, R and Python. It gives users a functionality to automate the process of building a large number of Machine Learning models and finding out the best pipeline within user specified time limits.

2.4 Feature Selection Techniques

In the whole Machine Learning process, feature selection is one of the most important steps to achieve improved results. A feature in case of a data set simply means a column. Machine Learning works on simple rules - if we pass in garbage values we will get only garbage values as output [27]. In a dataset, it is not necessary that every feature has a positive impact on the output variable. If we add these irrelevant features in the model, it will just make the performance of the model worse. This has given a boom to the necessity of implementing feature selection. Feature selection also helps to reduce over fitting and increases the accuracy of the model if the right subset of features is chosen.

There are various techniques and methodologies present to take out the best feature subset from the available feature space to increase the performance and efficiency of the model [27].

- **Filter Methods** As the name suggests, filter methods are the one in which we filter and keep only the selected subset of the relevant features. This method is commonly used as a preprocessing step. The relevant features are selected on the basis of their scores, achieved in multiple statistical tests for their relationship with the outcome variable. This selection of the features is completely independent of Machine Learning algorithms [27]. (Figure 2.4.1)

  Examples - chi-square, fclassif, Pearson's Correlation, etc.

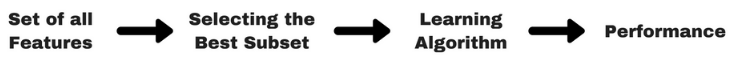


Figure 2.4.1: Workflow of filter methods [27]

- **Wrapper Methods** As mentioned before, in filter methods the feature selection is independent of any Machine Learning algorithm whereas wrapper methods need one Machine Learning algorithm to work on and then consider the performance of the model as the evaluation criteria to select the best feature subset. Wrapper methods select the best feature subset by considering it as a search problem where various subsets are prepared, assessed and compared to other subsets of features. Then these subset of features are evaluated by a predictive model and have given a score based on model accuracy. (Figure 2.4.2)

Examples - Forward Selection, Backward Selection, Recursive feature elimination, etc.

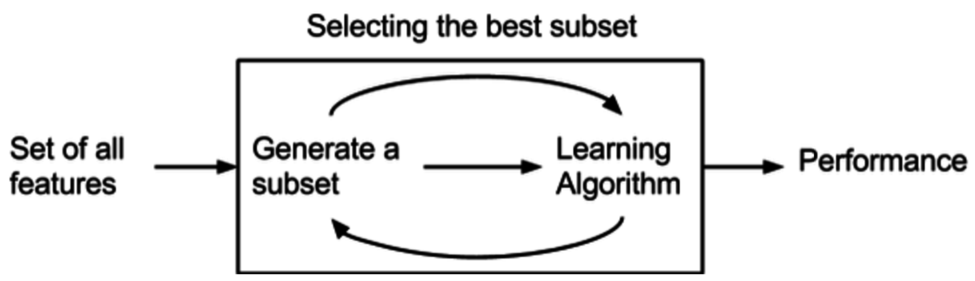


Figure 2.4.2: Workflow of wrapper methods [27]

- **Embedded Methods** These methods are the combination of filter and wrapper methods. Embedded methods are implemented by algorithms, which have their own built-in feature selection method [27]. They pick out the subset of features, which contribute in making the most accurate model while the model is being created. Embedded methods can be called as iterative in a manner that carefully looks into each iteration of the training process of the model and then make a subset of those features that are most important by checking their contribution to the training of the model for a specific iteration. (Figure 2.4.3)

  Example - Random forest (tree feature importance), LASSO and RIDGE regression, etc.

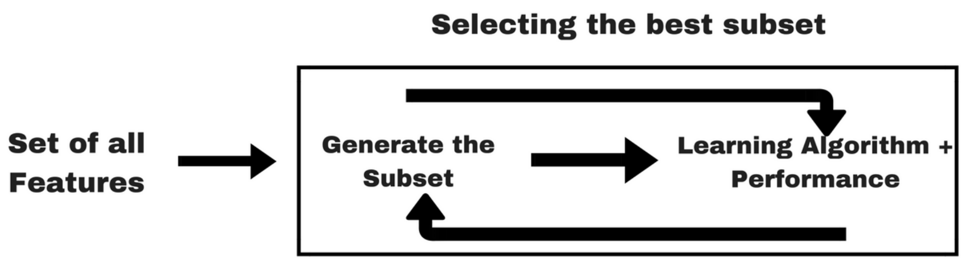


Figure 2.4.3: Workflow of embedded methods [27]

- **Correlation matrix** The term correlation can be defined as how two variables are related to each other. In common usage it is used to determine the linear dependency between the two variables. Correlation matrix can be used as one of the feature selection techniques as features with high correlation have almost the same effect on the dependent variable because they are linearly dependent on each other. So, we can drop one of the two features as they have high correlation between themselves [28].

- **LOFO importance** Leave one feature out (LOFO) importance calculates the importance's of a set of features based on a evaluation metric of choice, by iteratively removing each feature from the set, and evaluating the performance of the model [29]. LOFO first calculates the performance of the model using all the features available. Then it takes out one feature at every iteration and then evaluates the model's performance on the validation set. Based on that, it reports the scores of each feature importance. LOFO is also model agnostic, so it can work for any model.

2.5 Business Process Case

Agilon Analytics [4] visualizes process flows and finds deviations and bottlenecks. They get actionable insights on workflow movements and help in increasing business value. They want to optimize their business model and improvise the solutions given by them to their customers. They want to classify likely outcomes of business transactions based on event log data and provide quantified improvement suggestions by predicting using supervised Machine Learning. For example, identifying orders which are at risk of being delayed or customer invoices unlikely to be paid.

Their business process consists of various cases in which each one of the cases has a set of events occurring in a timely order. Each event is a set of features, e.g. order type, customer number/name, supplier number/name and many more with the timestamp of that event. In order to do process mining, a minimal initial data set is required – the event log. This event log contains the Activities (event names), stores the Timestamp of every event, and the Case ID (a business process case identifier) [30]. (See table 2.5.1)

This Event log table can be used to extract information on business process cases. For example, case–id (1....8.11) has a sequence of activities with three process steps, i.e. process step (1) is Registrerad, process step (2) is Frisläppt, and process step (3) is Ankomst/avgång. This implies that each business process case can be shown as a sequence of events (circles) with their corresponding features as shown in the Figure 2.5.1 [8].

Table 2.5.1: Business case event log

Case	Event	Timestamp
1....8.11	Registrerad	2018-01-19 00:00:00
1....8.11	Frisläppt	2018-02-06 00:00:00
1....8.11	Ankomst/avgång	2018-03-01 00:00:00
1....1.3	Orderdatum	2018-02-13 23:59:59
1....1.3	Önskad leverans	2018-04-05 00:00:00
1....1.3	Bekräftad leverans	2018-04-05 00:00:00

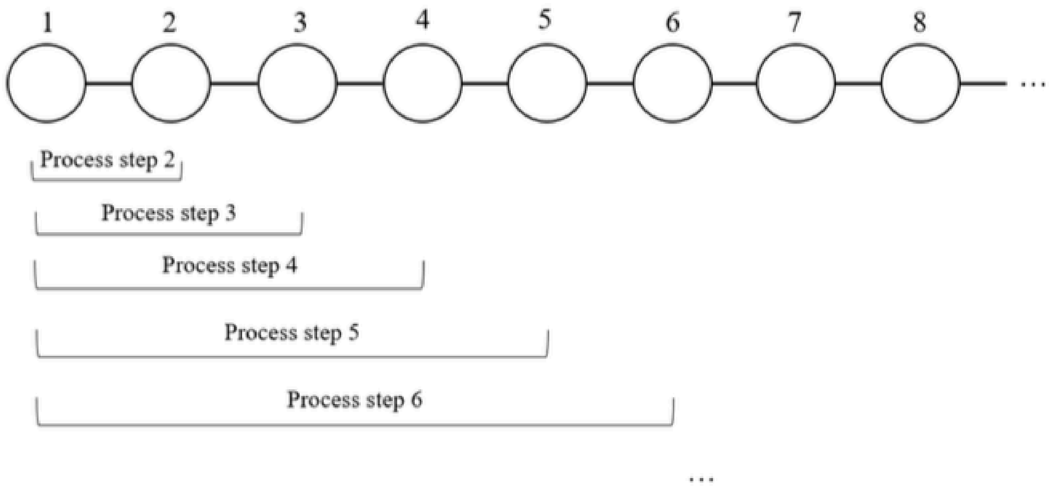


Figure 2.5.1: Business Case Process [8]

The occurrence of new events and corresponding attributes keep taking place as a business process case proceeds.

Chapter 3

Methods

The goal of this chapter is to present the implementations and methodology used to finalize on the most efficient data model with best features subset to be integrated with the Agilon platform. Along with this, this chapter also focuses on the implementations of the AutoML frameworks which are tested on the company dataset. Evaluation Metrics used in the study are also described under this chapter. Metrics which are defined are used to evaluate the quality of the models, implemented during the study.

3.1 Methodology

One of the focus of this research is to choose suitable supervised Machine Learning algorithm, so quantitative data analysis is chosen to measure and evaluate the performance of different algorithms discussed in chapter 2, e.g. Naive Bayes, Decision Tree, Random Forest, SVM, etc. based on the evaluation metric which are described later in this chapter. Finally, the research question has been answered by choosing a suitable classifier at each process step. Exploratory approach is where a large parameter space is explored to get to grips with the problem, without a defined hypothesis [8]. Here the exploratory approach is chosen to successfully complete the first research goal mentioned in chapter 1 under section 1.2, since various supervised Machine Learning algorithms are explored with a large parameter space [8]. The applied methodology in this study for comparing different supervised Machine Learning algorithms is shown in Figure 3.1.1.

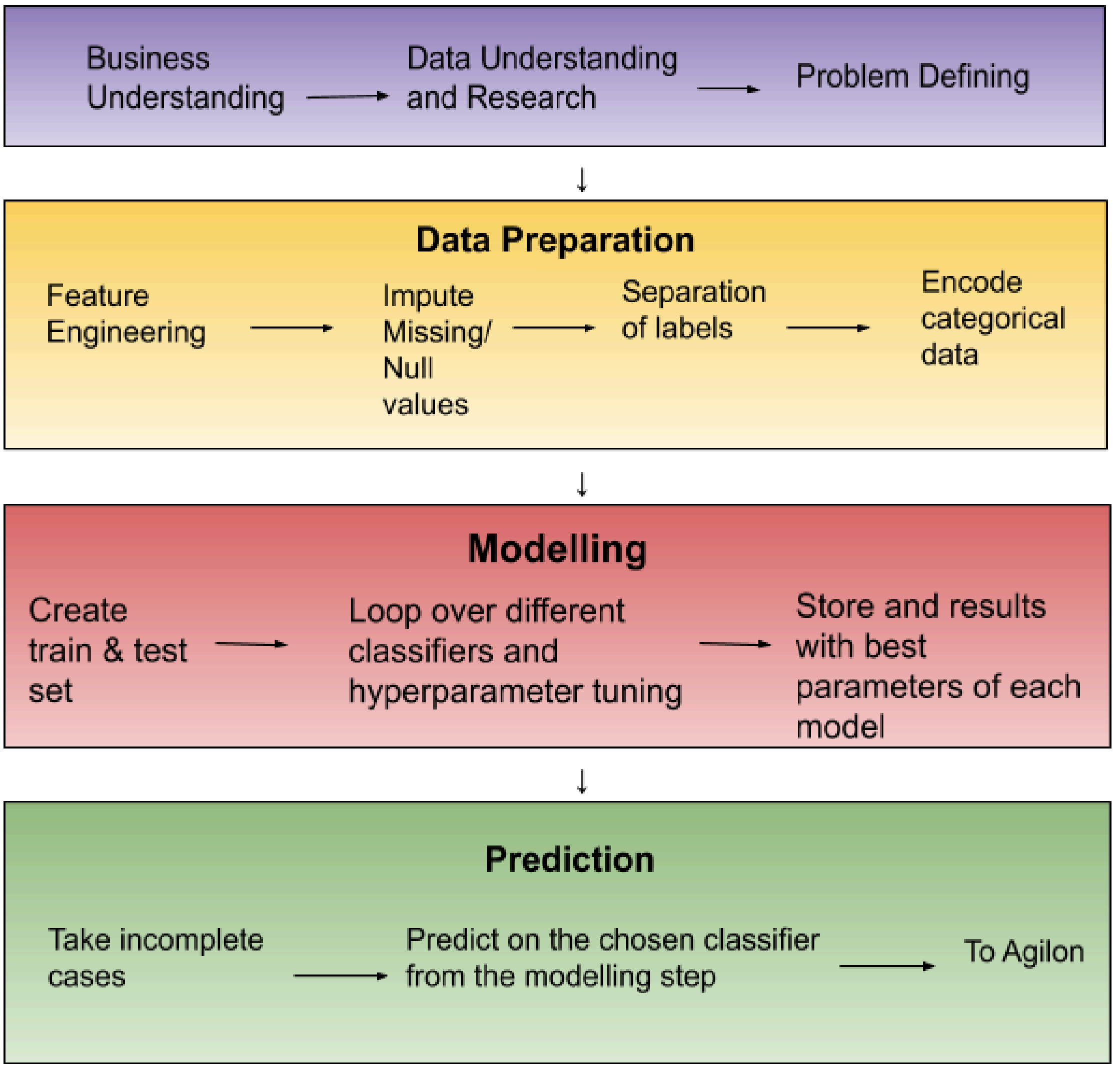

Business Understanding
Data Understanding and Research
Problem Defining
Data Preparation
Feature Engineering
Impute Missing/ Null values
Separation of labels
Encode categorical data
Modelling
Create train & test set
Loop over different classifiers and hyperparameter tuning
Store and results with best parameters of each model
Prediction
Take incomplete cases
Predict on the chosen classifier from the modelling step
To Agilon

Figure 3.1.1: Methodology applied

3.2 Dataset

The actual customer data set provided by Agilon for a P2P process over a period of one year (2018) is used. This dataset contains over 20,000 rows as instances and 256 columns as predictors. This dataset is used to predict or classify four labels so that it is a multi-label classification. In this job, a separate data table is used as the prediction target table. A brief overview of the entity dataset can be found in the table 3.2.1, which shows some of the 256 events (feature). In the entity dataset, each instance is identified by a unique identifier called a case. The prediction record is displayed in the table 3.2.2, which has labels in the target column. As shown, in table 3.2.2, there are four target labels to make predictions on. For data protection reasons, the full case IDs are hidden in this report, according to the company.

Table 3.2.1: Feature dataset

Case	**Events**
1....8.7	Ordertyp
1....8.6	Kundnummer
1....8.3	Leverantorsnr
1....8.2	Ordersumma
1....8.1	Leveranspostnr
1....7.1	Leveranspostort
1....5.3	Leveransland
....	....

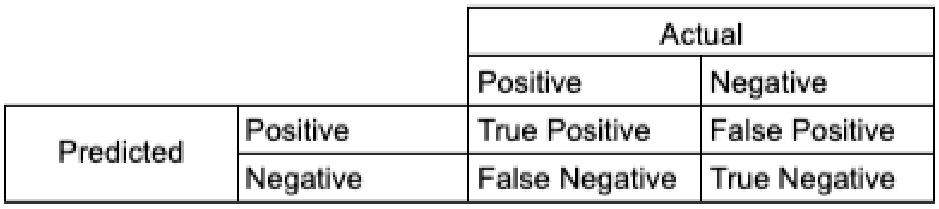
Table 3.2.2: prediction dataset

Case	**Targets**
1....8.7	KPI Sourcing
1....8.6	SLA - Orderbekräftelse
1....8.3	SLA - Ej Omplanerad
1....8.2	SLA - Ledtidsmål

3.3 Evaluation Metrics

Evaluation metrics calculate the quality of Machine Learning models. There are various metrics present to evaluate classifiers as:

- **Confusion Matrix** Confusion matrix is a N*N matrix, where N is the number of classes predicted. It is also known as error matrix, which provides insights on the performance of the predictive model which is defined as predicted classes over actual classes. It details about which classes are being predicted correctly or incorrectly. (Figure 3.3.1)



		Actual	
		Positive	Negative
Predicted	Positive	True Positive	False Positive
	Negative	False Negative	True Negative

Figure 3.3.1: Confusion Matrix

- True positive (TP)- Instances which are positive, and is predicted positive.
- False negative (FN)- Instances which are positive, but are predicted negative.
- False positive (FP)- Instances which are negative, but are predicted positive.
- True negative (TN)- Instances which are negative, and are predicted as negative.

• **Accuracy** It is well suited for binary classification and is defined as the proportion of correctly predicted instances among the total number of instances examined.

$$Accuracy = \frac{TP + TN}{TP + FN + FP + FN} \tag{3.1}$$

• **Precision** As the name suggests, precision analyse the preciseness about the target value. It is defined as the number of correctly predicted instances over the total number of positive predicted instances. It tells us about, how much proportion of predicted positives is truly positive.

$$Precision = \frac{TP}{TP + FP} \tag{3.2}$$

• **Recall** It is also know as sensitivity. Recall is defined as the proportion of actual positives, which are correctly classified.

$$Recall = \frac{TP}{TP + FN} \tag{3.3}$$

• **F1-score** It combines both precision and recall into one metrics. It is defined as the harmonic mean of the precision and recall and is a number between 0 and 1.

$$F1 = 2 * \frac{precision * recall}{precision + recall} \tag{3.4}$$

- **AUC ROC** The area under the receiver operating curve (AUC ROC) is primarily used as a visualization technique to show the performance of a classification model at different thresholds. AUC ROC indicates how well a model differentiates between positive and negative classes. ROC is a probability curve and AUC shows the degree of separability. To draw the ROC curve, we need to calculate the true positive rate (TPR) and false positive rate (FPR) on different threshold values. (Figure 3.3.2)

$$TPR = \frac{TP}{TP + FN} \tag{3.5}$$

$$FPR = \frac{FP}{FP + TN} \tag{3.6}$$

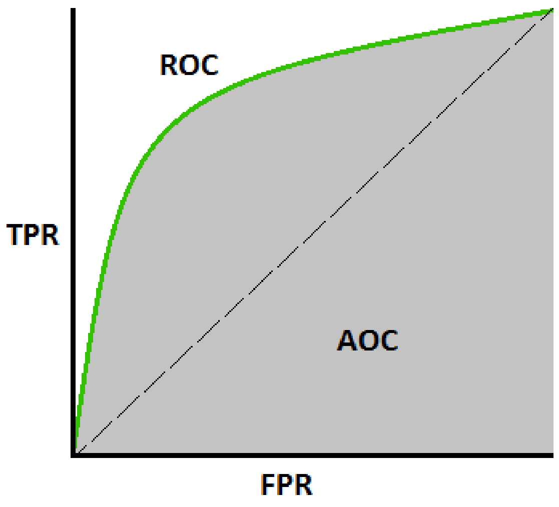


Figure 3.3.2: AUC-ROC Curve [31]

Higher the AUC, the better the model is in predicting the actual classes. AUC ROC having values around 1 or larger than 0.5 is considered to be a good classifier [31].

3.4 Machine Learning Models

As shown in 3.1.1, after defining a problem we start with data preparation step which includes:

Data Preparation Step

- Feature Engineering - Feature engineering plays an important role in Machine Learning applications. In this phase, business domain knowledge identifies important characteristics. That results in removing some features. For example, the case feature column is removed because it is the unique identifier.
- Imputation of missing values - Supervised Machine Learning algorithms could not process datasets with missing values because they receive an input array in which all values are numeric [32]. Therefore, there should not be any missing values in the input record. There are several ways to enter missing values, for example, with mean/median/constant. For numeric attributes with missing values, the mean of the existing values is used to enter the missing values and for categorical attributes, the constant option is used, which enters ”N” in the missing values.
- Encoding categorical data - Machine does not understand text, it only understand numbers. Most Machine Learning algorithms, require encoding of categorical data to numerical data in order to make predictions. In fact, an algorithm’s performance can also vary depending on which technique is used for categorical encoding. There are different approaches to handle categorical encoding:
  1. **One-hot Encoding-** For each categorical attribute/feature, the number of unique values is calculated as N, then N new binary attributes are added and the base attribute is removed. Basically, the categorical attribute is expanded to N new binary functions [8]. For example, shown in the following tables: there is the ”Fruit” attribute which has three values (”apple”, ”banana” and ”orange”). One-hot encoding this attribute, removes the ”Fruit” and instead adds three new attributes (”the Fruit is apple”, ”the Fruit is banana” and ”the Fruit is orange”) with 1 (true) and 0 (false) as corresponding values.
  2. **Label Encoding-** In this approach, each and every unique categorical feature value is encoded with a unique number. That means, if there are

Table 3.4.1: Categorical attribute

Case id	Fruit	Quantity
1	apple	2
2	banana	3
3	orange	5

Table 3.4.2: One-hot encoded attribute

Case id	Fruit is apple	Fruit is banana	Fruit is orange	Quantity
1	1	0	0	2
2	0	1	0	3
3	0	0	1	5

N categorical values in a feature then they will be mapped to the number from 1 to N as shown in the tables below. This approach is useful when there values are in order, so that the numerical encoded values can be compared.

Table 3.4.3: Label Encoded attribute

Case id	Fruit	Quantity
1	1	2
2	2	3
3	3	5

There are many more approaches to do categorical encoding. In this study, One-hot encoding is used.

Modelling step

- Train and Test sets spilt- Dataset is spilt into training and test set for cross validation and for inspecting the performance of the model. In this study, the ratio of train/test set is 70/30%.
- Machine Learning Classifiers (Algorithms)- Due to the scope of this study, only supervised Machine Learning algorithms are the focus of this research, only those algorithms, also known as classifiers, are taken into account. From various

classifiers, chosen classifiers are explained in section 2.1.1.

- Hyper-parameter Tuning- Here, different sets of hyper–parameters for each classifier are chosen. Based on the explanation in Section 2.2, Grid search and Random search are two most common methods used for Hyper-parameter optimization. Random search technique is used in this study.

  1. **Dummy classifier**- No Hyper–parameters
  2. **Naive Bayes classifier**- No Hyper–parameters
  3. **Logistic Regression classifier**
     - Penalty: [‘l1’]
     - C - param range: [0.001,0.01,0.1,1,10,100]
     - Solver: [’lbfgs’, ’liblinear’]
  4. **Decision Tree**-
     - The information gain criterion: [‘gini’,‘entropy’]
     - The maximum depth: [1,2,3,4,5,6,7,8,9,10,11,12]
     - The minimum samples leaf: [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
  5. **Random Forest**
     - The information gain criterion: [‘gini’,‘entropy’]
     - The maximum depth: [1,2,3,4,5,6,7,8,9,10,11,12]
     - The number of estimators: [10, 20, 30, 40, 50, 60, 70, 80, 90, 100]
     - Bootstrap (sample instances with replacement): [True,False]
  6. **SVM classifier**
     - C range : [1, 10, 100, 1000]
     - Gamma range : [‘scale’, ‘auto’]
     - Kernel : [‘linear’, ‘rbf’]
  7. **K-Neighbors classifier**
     - n neighbors : [2,4,5,10]

  – p value: [1,2]
  – leaf size : [3,5,7,9]
  – weights : ['uniform', 'distance']

Prediction step

Finally coming to the last step, which is to apply predictions on the on going business process. An appropriate (best performed) classifier is used from the modelling step to predict the company's current business. The latest results are finally being used by Agilon.

3.5 Comparison of AutoML Frameworks used for solving CASH problem

This section introduces the evaluated AutoML frameworks. The frameworks were selected based on their popularity, namely the number of citations and GitHub stars. In addition, each and every framework must use different techniques and be open source. First, the implementation of CASH algorithms is presented and analyzed, which is followed by the discussing the framework for creating complete ML pipelines.

CASH Algorithms

Some popular implementations of methods for algorithm selection and Hyper-parameter optimization are discussed here in this section:

- **Dummy**- To evaluate the effectiveness of various CASH algorithms, dummy classifier is added which uses stratified sampling to create random predictions. Scikit learn implementation with default Hyper-parameters values is used for the dummy classifier.
- **Grid Search**- Grid search is the most basic technique used for HPO as discussed in section 2.2. In this study, the existing GridSearchCV implementation from scikit-learn is used. GridSearchCV is by definition limited to HPO for a fixed algorithm. To extend this implementation for algorithm selection, a separate GridSearchCV instance is created for each available Machine Learning algorithm [20].
- **Genetic Algorithms**- The genetic algorithm is a classic fate-based evolutionary

algorithm (inspired by the process of natural selection), based on Darwin's theory of evolution. The main idea of genetic based optimization techniques is simply to apply various genetic operations to a population of configurations. For example, cross-breeding uses two parent chromosomes (configurations) and their genetic information combines to generate new offspring (child configuration). More specifically, two parent configurations are intersected at the same intersection point. The parts to the right of this point are then swapped between the two parent chromosomes. This contributes to two new offspring. A mutation randomly selects a chromosome and mutates one or more of its parameters, resulting in an entirely new chromosome [18].

- **SMAC**- Another CASH solver for selecting Hyper-parameter is SMAC [33], which stands for sequential model-based algorithm configuration. It is based on sequential model-based optimization. This was the first framework introduced explicitly supporting categorical variables. It also has an interesting feature of stopping configuration evaluation after a certain time. After initialization with the default configuration, or randomly if the default configuration is not available, the SMBO cycle repeats for a specific number of iterations or a specific time budget. The performance of all previous configuration runs is modeled using random forest regression, which contains ten regression trees that are trained via bootstrap-ping and the results are averaged. The Hyper-parameters remain the default for each tree. The choice of these meta Hyper-parameters is no longer justified. Candidate configurations are generated by searching locally in the configurations tested so far. In addition, new configurations are randomly selected from the entire configuration area [20].

AutoML Frameworks

As discussed in section 2.3, AutoML frameworks were selected on the basis of their popularity, namely the number of citations and GitHub stars. Each framework presented is able to build a complete Machine Learning process. All frameworks were implemented by specifying a hard time limit of 1 hour.

- **Auto Skleran** - A brief introduction of the tool, which is used for building classification and regression pipelines is described in section 2.3.
  - CASH Solver - SMAC

  - Fixed structure[1] - Yes
  - Ensemble learning - Yes, as mentioned in the description of the framework. It implements ensemble to give an ensemble of best created pipelines, instead of just the best one.
  - Categorical support - Manual encoding of categorical features is required.
  - Supports Multi-label classification.

- **Auto WEKA** - Auto Weka uses WEKA framework and applies Bayesian optimization for for both algorithm selection and hyper-parameter optimization. There is also an extension of Auto Weka namely Auto MEKA, which focuses on multi-label classification problem.
  - CASH Solver - SMAC and Tree-structured parzen estimator (TPE). (Default optimization algorithm is SMAC)
  - Fixed structure - Yes
  - Ensemble learning - Yes, well performing set of tuned classifiers are returned.
  - Categorical support - Manual encoding of categorical features is required.
  - Supports Single-label classification and regression.
- **TPOT** - As discussed earlier in section 2.3, this framework is used for tuning and creating arbitrary regression and classification pipelines. It constructs flexible pipelines and select an algorithm in each pipeline stage.
  - CASH Solver - Genetic Programming
  - Fixed structure - No, it has a variable structure.
  - Ensemble learning - No, it returns a single best performing pipeline.
  - Categorical support - No, integer encoding is required before passing the data.
- **H2O AutoML** - It is a Machine Learning framework similar to Scikit Learn but it does not uses Scikit-Learn as its training framework. H2O includes an automatic

[1]All the pipelines have a fixed structure: Firstly, a fixed set of data cleaning steps after that optional scaling, then an optional preprocessing and lastly mandatory modeling algorithm.

Machine Learning module that uses its own algorithms to build a pipeline [34].

- CASH Solver - Grid Search
- Fixed structure - Yes
- Ensemble learning - Yes, gives an ensemble model.
- Categorical support - Yes, supports categorical features

3.6 Impact of feature selection technique on accuracy of models

As introduced in section 2.4, it is not always necessary to have better results with more data features. Major focus should be on having relevant features, not on number of features. There are sometimes many irrelevant features that does not result towards any improvement in the model predictions. Also, in many cases there are redundant features, which in presence with others can slow down the learning process and also can result in over-fitting. Following are the implemented methods for feature selection. All the methods are implemented on a dataset, which was provided by the company that has a binary classification target.

- **Filter methods**

  As explained in section 2.4, filter methods does filters out the features before the starting the learning process. Here in this study all techniques describe below are used from Sklearn feature selection library.

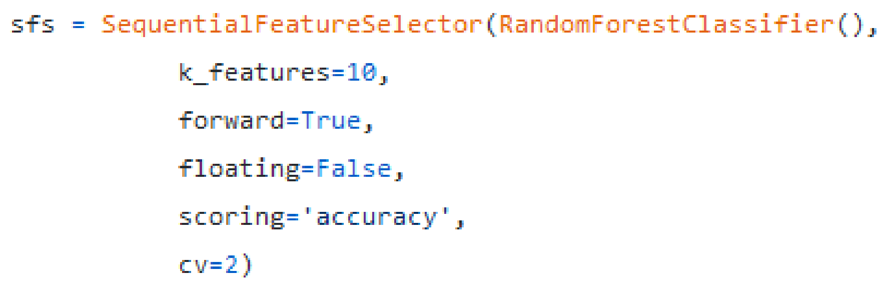
  1. Chi-square - It is statistical method[2], which access each and every feature independently. Chi-square score is majorly used to evaluate the categorical features. It works with non negative values and binary targets only [35].
  2. ANNOVA - It stands for Analysis of Variance and is uni-variate test. It is also similar to the previous one i.e. measure dependency of two variables. It also requires binary target [35]. F_ classif function is used for classification tasks.

- **Wrapper methods**

[2]Evaluation of whether the variable is important in order to discriminate against the target.

As introduced in section 2.4, unlike filter methods, wrapper methods needs a Machine Learning algorithm to work. Following are the techniques used for implementing wrapper methods. mlxtend is the library used for every technique describe below.

1. Forward Selection - It is an iterative approach, where first step is to evaluate each and every feature individually, and then pick the one feature that results in the best performance. In the second step, it take the selected feature in the first and evaluates all feasible combinations of the with the remaining features and retains the pair that results in giving the best performance. And thereafter, this loop continues by adding one feature at a time in each iteration until the preset criterion is reached.

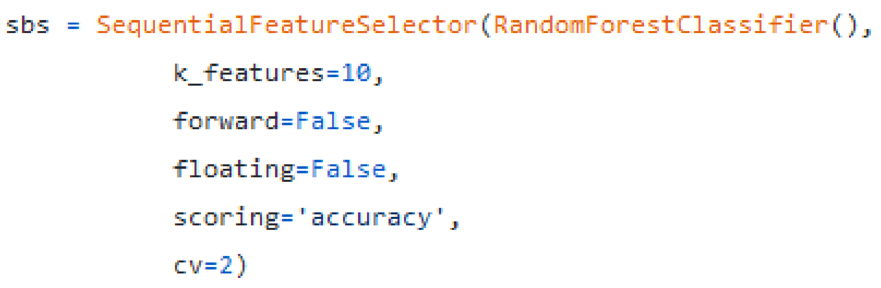


```
sfs = SequentialFeatureSelector(RandomForestClassifier(),
           k_features=10,
           forward=True,
           floating=False,
           scoring='accuracy',
           cv=2)
```

Figure 3.6.1: Forward selection parameters

Some of the parameters are shown in the Figure 3.6.1. Where, **k features** is the preset criteria, which is set to be 10 in this case (maximum limit for features to reach during iteration). **Forward** is the type of sequential technique, in this case it is forward step so it is set to true. **Floating** is to use an alternative of forward step selection, which is false in this case. **Scoring** is defined as the evaluation metric to test the model. **CV** specifies the number of folds for cross validation.

2. Backward Selection - Unlikely from the previous technique, we start with evaluating the model with all the features present in the dataset. And as the second step, at each iteration this technique, removes one feature at a time that results in producing the best performance of an algorithm. This step is repeated until the preset criteria is reached.

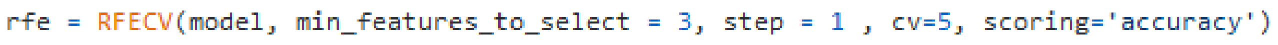


```
sbs = SequentialFeatureSelector(RandomForestClassifier(),
           k_features=10,
           forward=False,
           floating=False,
           scoring='accuracy',
           cv=2)
```

Figure 3.6.2: Backward selection parameters

As shown in the Figure 3.6.2. Where, **k features** is the preset criteria, which is set to be 10 in this case (represents the total number of features).**Forward** is the type of sequential technique, in this case it is forward step so it is set to false (as this is backward step technique). **Floating** is to use an alternative of backward step selection, which is false in this case. **Scoring** and **CV** are same as the previous one.

3. Recursive feature elimination - As the name suggests, it is the iterative feature elimination method. Firstly, train a model with all the available features in the dataset and then rank the features based on the performance of the model, evaluated by the chosen metric by any feature importance method (like tree based feature selection). Then, eliminate least important feature and re-train the model and measure the performance of the resulting model based on the previously chosen metric. Then compare the results by a pre-decided threshold to see the significance of that feature on the rest of the model. This step is repeated until all features are evaluated.



```
rfe = RFECV(model, min_features_to_select = 3, step = 1 , cv=5, scoring='accuracy')
```

Figure 3.6.3: RFE Feature selection parameters

As shown in the Figure 3.6.3. Where, **min feature to select** as suggested by the name, tells the minimum features to select.**Step** is the number by which the features should be removed. **Scoring** and **CV** are same as explained in the forward selection.

- **Embedded methods**

  As explained in the section 2.4, these methods does feature selection while training the model itself i.e., during the time of creating the Machine Learning algorithm. Below, some of the embedded techniques are explained.

1. Feature importance (Random Forest) - Random Forest is just an ensemble of decision trees, where predictions of all the trees are combined to achieve a final prediction. Every node in the decision trees acts as a condition of one feature. The measure based on which optimal condition is chosen is known as an impurity [36], which is Gini impurity/information gain (entropy) in case of classification. Thus, in the formation of the tree, the importance of the feature is measured as the reduction of the node impurity weighted in the tree. The higher the value, the more important the function. Hence, feature importance is derived by the purity of every node (set) [36]. In this study, this method is used from Sklearn library.

2. Lasso - Also called as ”L1” regularization. Lasso stands for Least Absolute Shrinkage and Selection Operator, which has the ability to shrink some of the coefficients to zero. That means, if the feature is irrelevant, lasso can penalize and make it’s coefficient 0. This means, some of the features will be zero when predicting the target variable. So as these features won’t be included in the final predictions of the model, they can be removed. Therefore, giving a reduced set of features for the final model. This is also used from Sklearn library during this study.

- **Correlation matrix** - Correlation matrix is used by filter methods and is usually done by using Pearson’s correlation.

  1. Pearson’s Correlation - The correlation is what one variable depends on the other and is very beneficial property. If a variable is highly correlated to another that variable can be used to predict another. Therefore, we see collinearity of features with the targets to predict the target variable efficiently. Pearson’s Correlation is commonly used to outline the robustness of the relationship between two variables, which can vary in between 1 and -1:

     – Positive correlation: The value of the Pearson’s Correlation is 1. Variables are directly dependent on each other.

     – Negative correlation: The value of the Pearson’s Correlation is -1.

Variables are inversely dependent on each other.

  - Linear correlation: The value of the Pearson’s Correlation is 0. Variables are not dependent on each other.

- **LOFO importance** - LOFO is implemented by installing ”LOFO-importance” and model used is Random Forest Classifier.

- **Permutation importance** - The permutation feature importance is defined to be the decrease in a model score when a single feature value is randomly shuffled [37]. This technique works after the model is trained. Permutation importance answers the question that how much the accuracy of the predictive model varies, if we randomly shuffle one feature of the validation data while balancing the other features and the target in position [38].

Chapter 4

Results and Discussion

In this chapter, we present the results and compare the performance of the models, feature selection techniques and frameworks.

4.1 Predictive model for multiple response target

In this study, various classifiers are experimented after optimizing their hyperparameters. Results of evaluated classifiers based on the chosen evaluated metrics - Accuracy, F1 Score, Precision and Recall is shown in the below Table 4.1.1.

Table 4.1.1: Classifiers evaluation

Classifier	**Accuracy**	**F1 Score**	**Precision**	**Recall**
Dummy Classifier	0.70155	0.93409	0.87854	1
Naive Bayes Classifier	0.2278	0.20345	0.50345	0.51833
Logistic Regression Classifier	0.96761	0.49176	0.48402	0.49976
Decision Tree Classifier	0.89965	0.97933	0.98709	0.97219
SVM (SVC) Classifier	0.96805	0.49188	0.48402	0.5
K Nearest Neighbor Classifier	0.999872	0.99997	0.99997	0.99997
Random Forest Classifier	0.99987	0.99997	0.99994	1

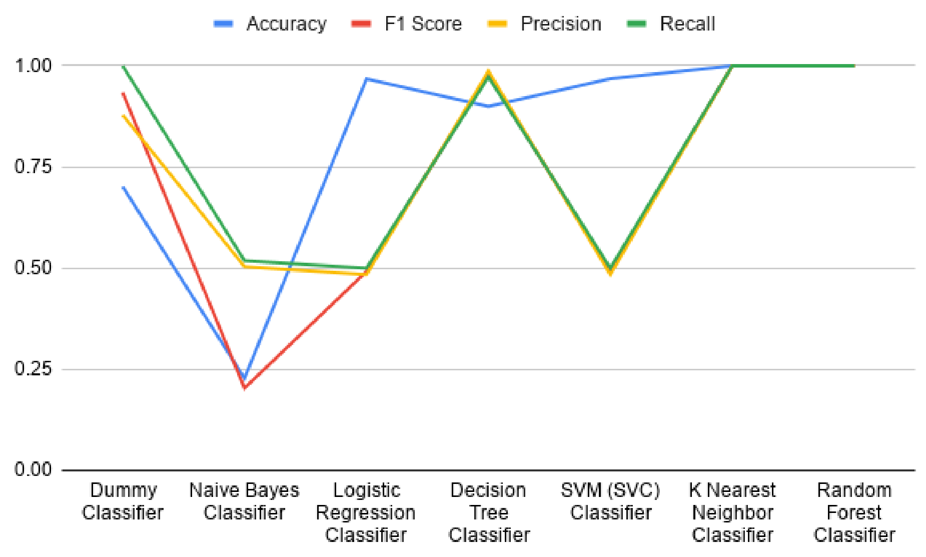


Figure 4.1.1: Evaluation results for all classifiers

From the results and the graph shown in Figure 4.1.1, it is clear that top two classifiers were Random Forest classifier and K Nearest Neighbor classifier. Now to pick one from these two, we evaluated them again using AUC/ROC curve shown in Figure 4.1.2. As it can be seen in the graph, both classifiers have almost equal performance but Random Forest has slightly better performance than K Nearest Neighbors, so Random Forest is considered as the best classifier.

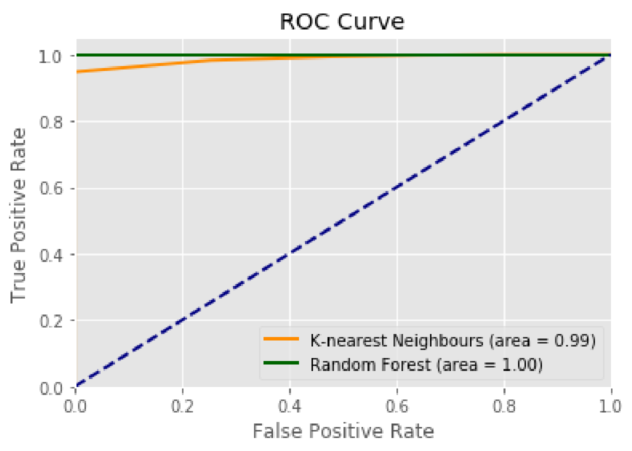


Figure 4.1.2: AUC/ROC curve for top two classifiers

A sample of final results is shown in table 4.1.2, with the predictions for all four labels as well as their prediction probability in % for every case id.

Table 4.1.2: Results for all labels with predict probability %

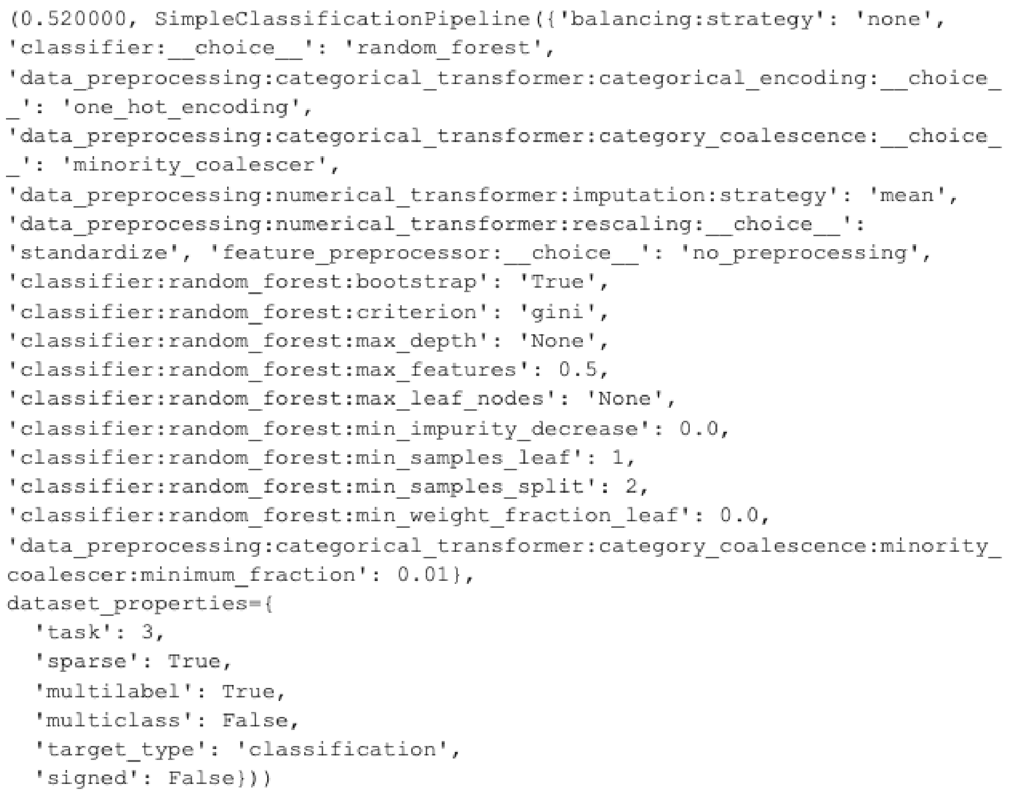
Case	KPI-Sourcing	SLA-Ej Omplanerad	SLA-Orderbekräftelse	SLA-Ledtidsmål	KPI-Sourcing predict prob	SLA-Ej Omplanerad predict prob	SLA-Orderbekräftelse predict prob	SLA-Ledtidsmål predict prob
1....8.7	1	1	1	1	100%	100%	100%	87%
1....5.3	1	0	0	1	94%	99%	53%	100%
1....9.4	1	1	1	0	100%	100%	100%	51%
1....3.1	1	1	1	0	100%	100%	89%	65%
1....4.4	1	1	1	1	100%	100%	100%	56%
1....3.6	1	1	1	1	100%	100%	100%	99%

4.2 Discussion on compared AutoML frameworks

In this study, various AutoML frameworks have been studied. Results of evaluated AutoML frameworks are presented below, which includes their best pipeline or classifiers and their performance results.

Auto Sklearn -

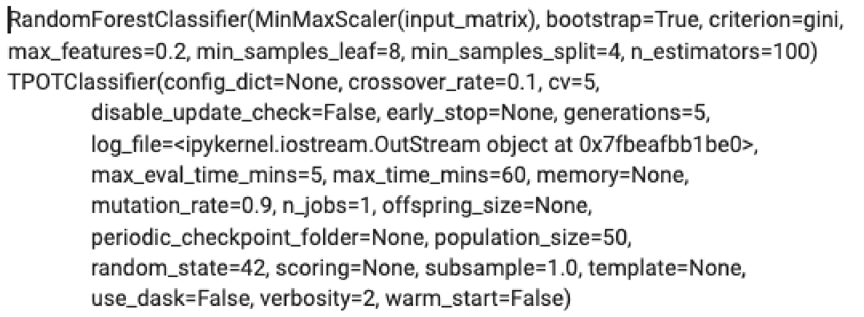
As discusses before, Auto-Sklearn gives ensemble of model as result. The first classifier in the ensemble is Random Forest with the properties shown in Figure 4.2.1. The full ensemble of models is presented in Appendix A. Auto-Sklearn was tested on the dataset provided by the company, in which it gave excellent performance with accuracy score - 0.99751 and R2 score - 0.99010. Having accuracy of 99% can be considered as being a very good model.



```
(0.520000, SimpleClassificationPipeline({'balancing:strategy': 'none',
'classifier:__choice__': 'random_forest',
'data_preprocessing:categorical_transformer:categorical_encoding:__choice_
_': 'one_hot_encoding',
'data_preprocessing:categorical_transformer:category_coalescence:__choice_
_': 'minority_coalescer',
'data_preprocessing:numerical_transformer:imputation:strategy': 'mean',
'data_preprocessing:numerical_transformer:rescaling:__choice__':
'standardize', 'feature_preprocessor:__choice__': 'no_preprocessing',
'classifier:random_forest:bootstrap': 'True',
'classifier:random_forest:criterion': 'gini',
'classifier:random_forest:max_depth': 'None',
'classifier:random_forest:max_features': 0.5,
'classifier:random_forest:max_leaf_nodes': 'None',
'classifier:random_forest:min_impurity_decrease': 0.0,
'classifier:random_forest:min_samples_leaf': 1,
'classifier:random_forest:min_samples_split': 2,
'classifier:random_forest:min_weight_fraction_leaf': 0.0,
'data_preprocessing:categorical_transformer:category_coalescence:minority_
coalescer:minimum_fraction': 0.01},
dataset_properties={
  'task': 3,
  'sparse': True,
  'multilabel': True,
  'multiclass': False,
  'target_type': 'classification',
  'signed': False}))
```

Figure 4.2.1: Auto Sklearn best pipeline

TPOT - As compared to Auto-Sklearn, TPOT does not give ensemble of model, it gives the best fitted classifier for the dataset. After running TPOT for 60 minutes, the best fitted classifier resulted was Random Forest with the properties shown in Figure 4.2.2. TPOT scores a little less than Auto-Sklearn with regards to the performance giving accuracy score of 0.97671.

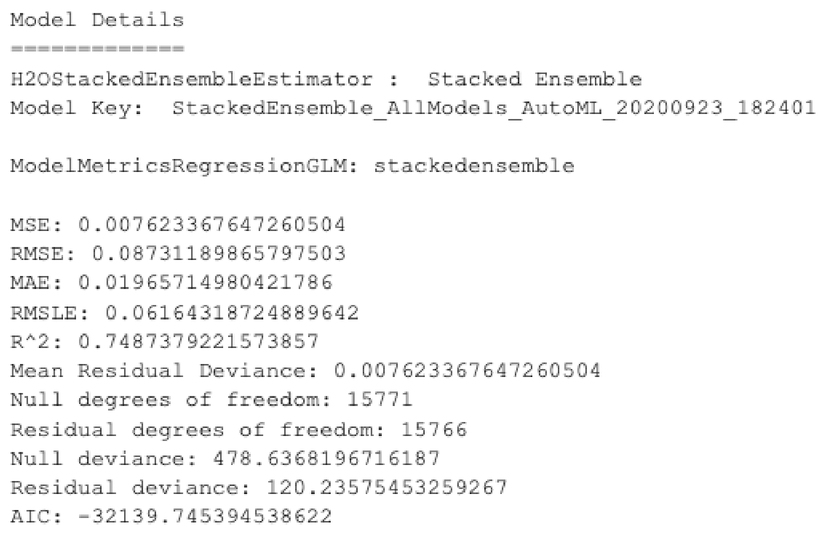


```
RandomForestClassifier(MinMaxScaler(input_matrix), bootstrap=True, criterion=gini,
max_features=0.2, min_samples_leaf=8, min_samples_split=4, n_estimators=100)
TPOTClassifier(config_dict=None, crossover_rate=0.1, cv=5,
        disable_update_check=False, early_stop=None, generations=5,
        log_file=<ipykernel.iostream.OutStream object at 0x7fbeafbb1be0>,
        max_eval_time_mins=5, max_time_mins=60, memory=None,
        mutation_rate=0.9, n_jobs=1, offspring_size=None,
        periodic_checkpoint_folder=None, population_size=50,
        random_state=42, scoring=None, subsample=1.0, template=None,
        use_dask=False, verbosity=2, warm_start=False)
```

Figure 4.2.2: TPOT best pipeline

H2o.ai - As discussed before H2O uses its own algorithms to build a pipeline. So the resulted best fitted classifier for H2O was H2OStackedEnsembleEstimator : Stacked

Ensemble with the properties shown in Figure 4.2.3. H2O also shows quite a low performance in comparison to Auto-Sklearn, with a R2 score of 0.74873.



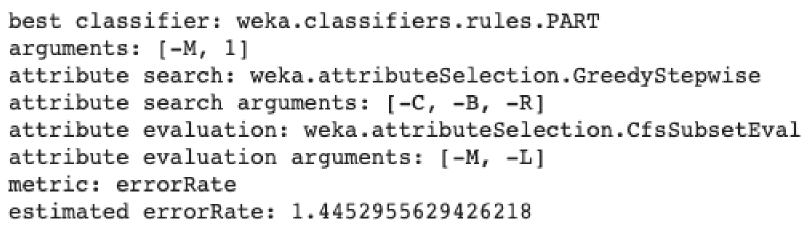
```
Model Details
=============
H2OStackedEnsembleEstimator :  Stacked Ensemble
Model Key:  StackedEnsemble_AllModels_AutoML_20200923_182401

ModelMetricsRegressionGLM: stackedensemble

MSE: 0.007623367647260504
RMSE: 0.08731189865797503
MAE: 0.01965714980421786
RMSLE: 0.06164318724889642
R^2: 0.7487379221573857
Mean Residual Deviance: 0.007623367647260504
Null degrees of freedom: 15771
Residual degrees of freedom: 15766
Null deviance: 478.6368196716187
Residual deviance: 120.23575453259267
AIC: -32139.745394538622
```

Figure 4.2.3: H20 best classifier

Auto-Weka - Auto-Weka is also one of the most popular framework for AutoML. Auto-Weka is used for binary classification in this study and was ran for hard time limit with 120 seconds. As a result, the best fitted classifier was weka.classifiers.rules.PART with the properties shown in Figure 4.2.4. In regards with performance, Auto-Weka gave magnificent results with the accuracy score of 0.99985.



```
best classifier: weka.classifiers.rules.PART
arguments: [-M, 1]
attribute search: weka.attributeSelection.GreedyStepwise
attribute search arguments: [-C, -B, -R]
attribute evaluation: weka.attributeSelection.CfsSubsetEval
attribute evaluation arguments: [-M, -L]
metric: errorRate
estimated errorRate: 1.4452955629426218
```

Figure 4.2.4: Auto-Weka best classifier

4.3 Feature selection techniques

As explained before, chosen evaluation metric for feature selection techniques is ”accuracy” and the models used are top two classifiers, which are ”K Nearest Neighbors classifier” and ”Random Forest”. The dataset contains more than 10k instances and around 200 features.

Results of various feature selection techniques is shown in the table below 4.3.1. Before any feature selection, the accuracy of the K neighbors classifier is 0.932 and of Random Forest classifier is 0.998.

Table 4.3.1: Feature Selection techniques for top two classifiers

Technique	**K-Neighbors**	**Random Forest**
chi2	0.983	0.997
ANNOVA (f-Classif)	0.989	0.989
Forward selection	0.998	0.999
Backward selection	0.989	0.999
LOFO importance	0.988	0.988
Tree feature importance	0.943	1
Lasso	0.983	0.997
Permutation Importance	0.931	1

As can be seen from the table 4.3.1 and the graph 4.3.1, that although all the techniques gives similar results but tree feature importance and permutation importance gives the best performance based on accuracy.

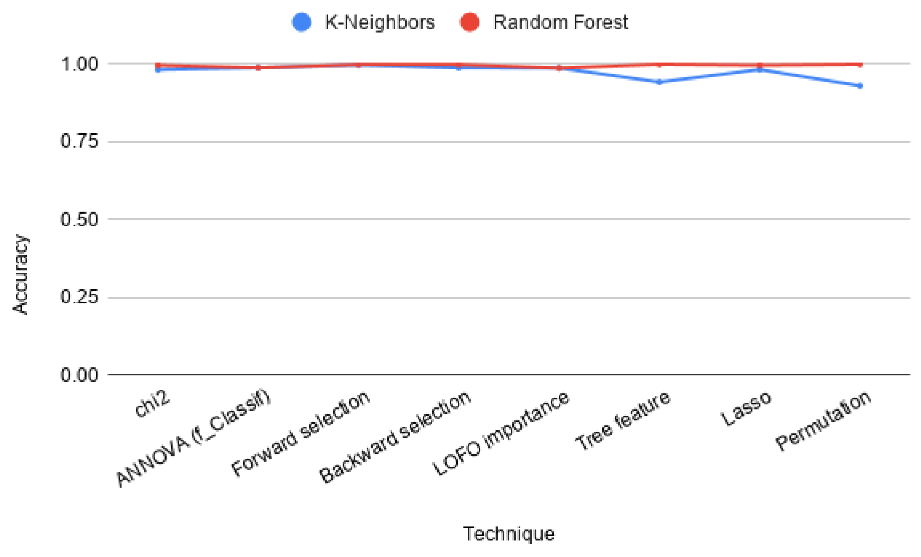


Figure 4.3.1: Feature Selection Techniques w.r.t. accuracy for top two classifiers

So, now below are the figures from individual feature importance for each and every target separately using permutation importance feature selection technique. In the

Figures 4.3.2, 4.3.3, 4.3.4 and 4.3.5 a sample of the feature importance is shown. It shows the features that have positive as well as, which negative impact on the prediction of that target. The upper most ones shows the positive impact on target, which can be considered as the most important features for the increase in performance of the model. But, the negative one means that removing those features can actually increase the performance of the model.

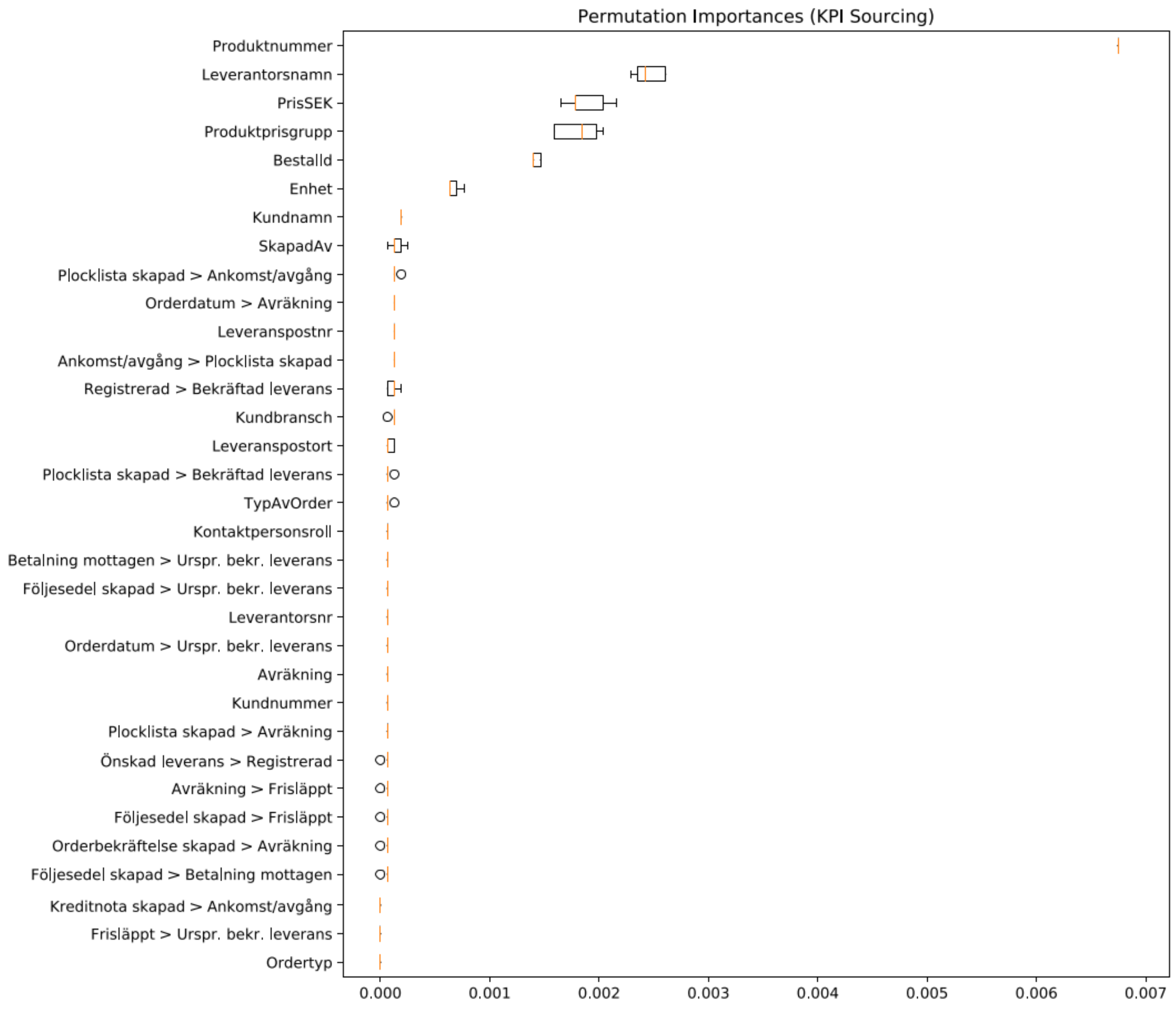


Figure 4.3.2: Sample of Feature Importance Result for KPI sourcing

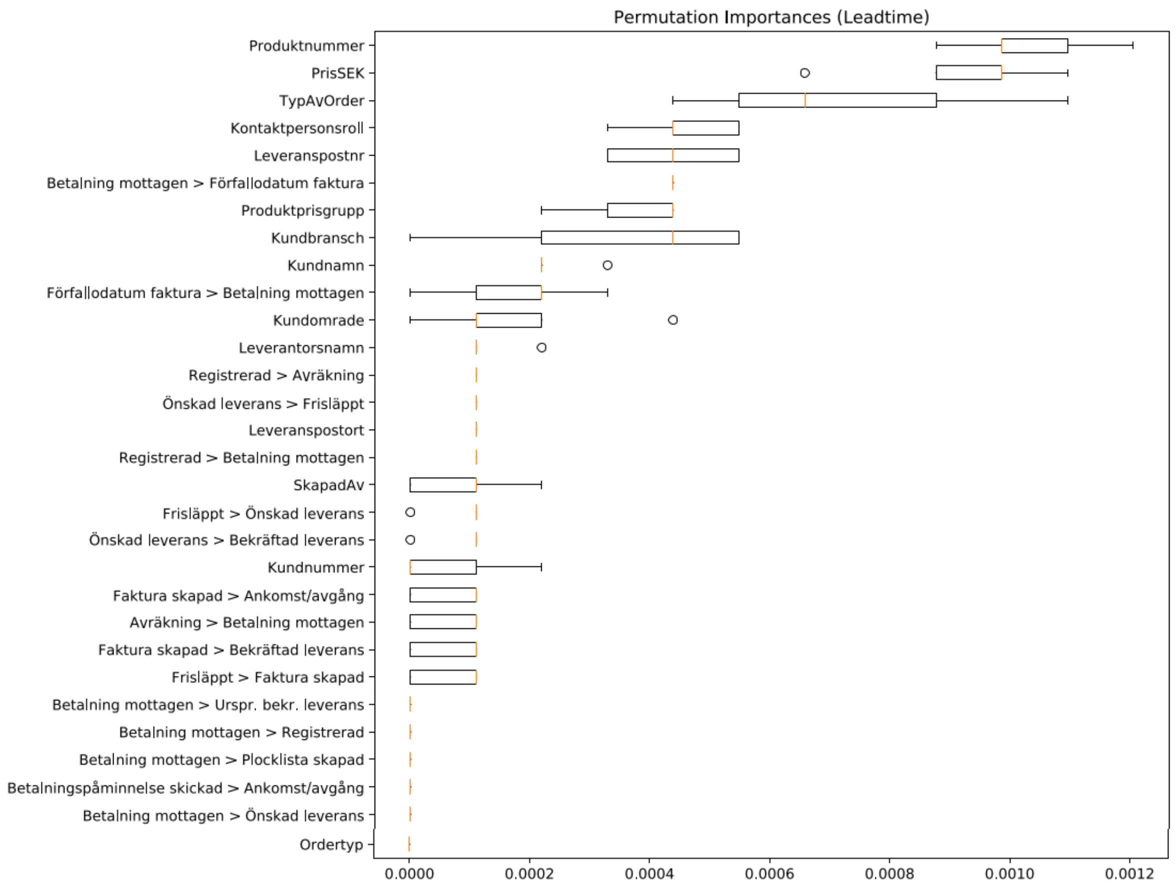


Figure 4.3.3: Sample of Feature Importance Result for SLA - Ledtidsmål

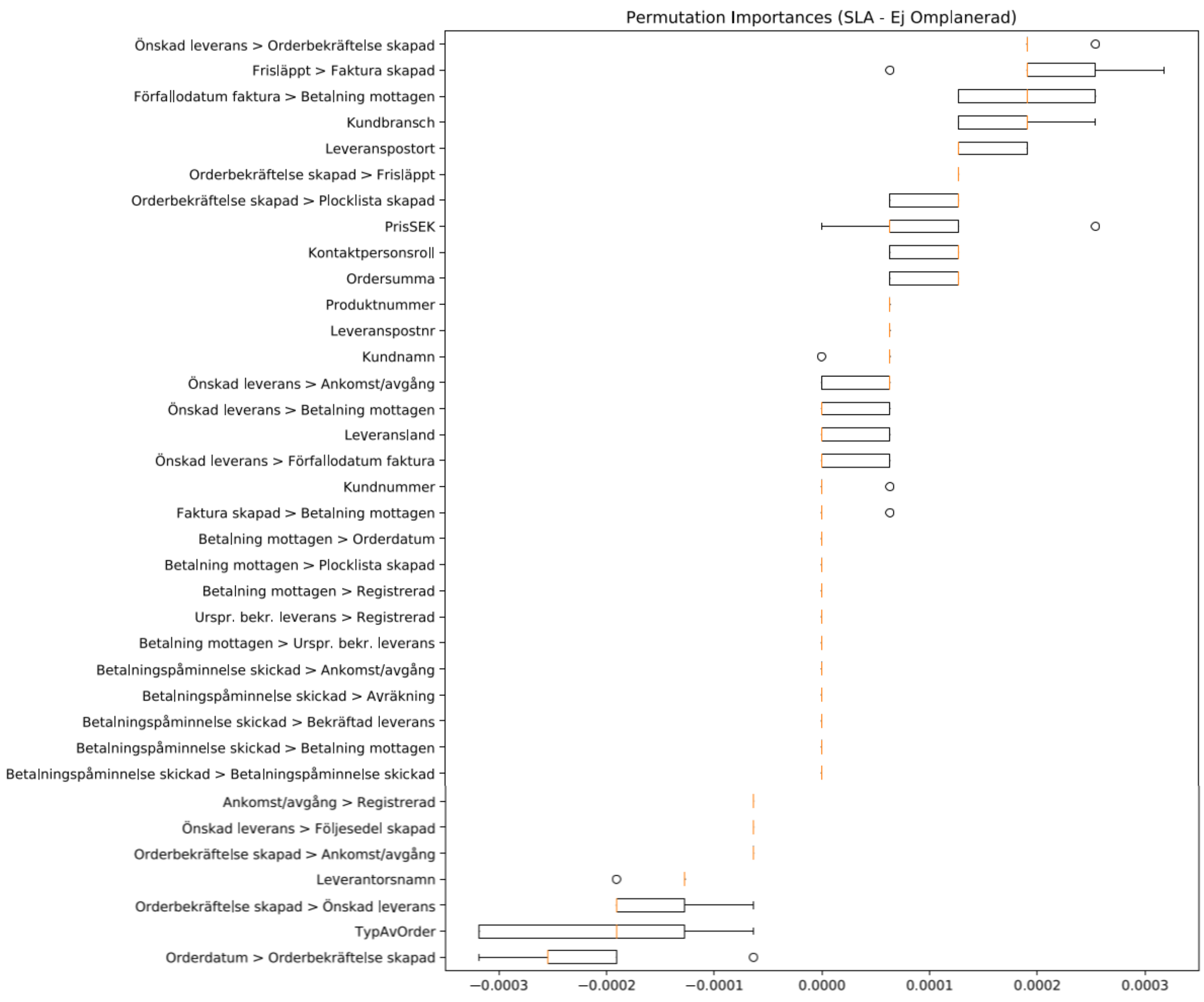


Figure 4.3.4: Sample of Feature Importance Result for SLA - Ej Omplanerad

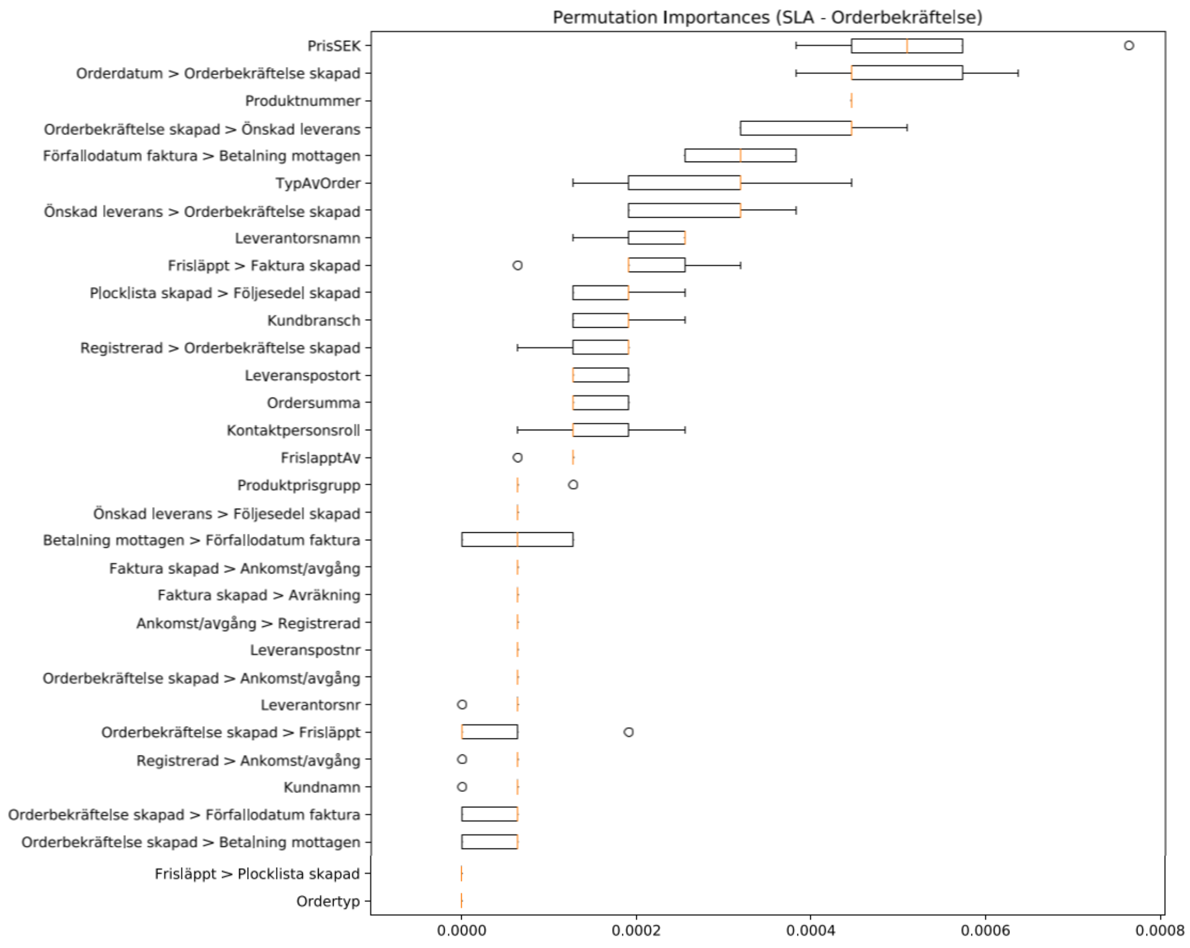


Figure 4.3.5: Sample of Feature Importance Result for SLA - Orderbekräftelse

Chapter 5

Conclusion and Future work

This chapter contains the overall conclusion and future work of this study.

5.1 Conclusion

Machine Learning has become one of the main, focused engines of the present era. In this thesis, we discussed three areas related to Machine Learning. Firstly, using Machine Learning for the predictive analysis. Secondly, to study a part of the production pipeline of a Machine Learning model namely feature selection techniques for better performance results. And the last one, to study various AutoML platforms, which solves the problem of various pipeline involved in building a Machine Learning model.

This study shows the combination of making manual production pipelines for building a Machine Learning model. Like, we showed stages in the study - Cleaning data, Hyper-parameter optimization, feature selection techniques, Algorithm selection and training the model. And after that we have discussed about the frameworks present which automate these pipelines, thus solving the problem of CASH (combined algorithm selection and Hyper-parameter optimization).

As we can see, for predictive analysis, Random Forest was the final model chosen to get the most out of the model in terms of performance. And to show, which features have the greatest influence on prediction, we used permutation importance as a feature selection technique.

5.2 Future Directions

By the completion of this study, in this section we aim to expose possible continuations of this work. We can illuminate new horizons for promising projects in the field of future monitoring of the business process and also in the area of automated Machine Learning specifically in AutoML frameworks.

- **Evaluating more classifiers:**

  This study evaluates different Machine Learning algorithms. Therefore, perhaps the next step is to explore the potential of more classifications or even methods such as deep neural networks, given their limitations.

- **Generation of features:**

  The work of this study explains the importance of feature engineering, some that always offer an opportunity for development or improving results and efficiency of the models. Hence next step may be to enrich the dataset by adding more features instead of reducing the number of features.

- **Scalability:**

  This study discuss about various centralized frameworks for automating the solutions for the CASH problem (e.g., Auto-Weka, Auto-Sklearn, etc.). Their main limitation is that they are tightly linked to a Machine Learning library (like Weka, scikit-learn, R) that can only run on a single node, making them not applicable to large amounts of data. Therefore, next steps can be the study of distributed Machine Learning platforms (e.g. SystemML, Spark MLib, etc.) to tackle the challenge of automatically creating and optimizing Machine Learning models on a huge datasets [18].

- **User friendly AutoML frameworks:**

  The frameworks studied in this study are not that user friendly, they all need some kind of technical knowledge for their implementation, which is not well suited for layman users and domain experts (e.g., physicians, accountants) who commonly have limited technical skills. Providing or building an interactive and light-weight web interfaces for such frameworks can be one of the future work in regards with Automated Machine Learning.

- **Multi Class - Multi Output classification:**

  As this study discusses the classification of multiple labels, more support is needed in this area. There are only a very small number of frameworks that support multi label / output classification. Therefore, we can provide more support for classification with multiple classes - as well as for multi output as the next steps.

Bibliography

[1] Pant, Ayush. “Introduction to Machine Learning for Beginners”. In: (2019). URL: https://towardsdatascience.com/introduction-to-machine-learning-for-beginners-eed6024fdb08.

[2] Géronen, Aurélien. *Hands-On Machine Learning With Scikit-Learn, Keras, And Tensorflow*. O’Reilly Media, 2019. ISBN: 9781492032649.

[3] Feurer, Matthias, Klein, Aaron, Eggensperger, Katharina, Springenberg, Jost, Blum, Manuel, and Hutter, Frank. “Efficient and Robust Automated Machine Learning”. In: *Advances in Neural Information Processing Systems 28*. Ed. by C. Cortes, N. D. Lawrence, D. D. Lee, M. Sugiyama, and R. Garnett. Curran Associates, Inc., 2015, pp. 2962–2970. URL: http://papers.nips.cc/paper/5872-efficient-and-robust-automated-machine-learning.pdf.

[4] *Agilon Process Intelligence - Agilon*. hhttps://www.agilon.io/.

[5] Edwards, Gavin. “Machine Learning | An Introduction”. In: (2018). URL: https://towardsdatascience.com/machine-learning-an-introduction-23b84d51e6d0.

[6] Andreas C. Müller, Sarah Guidoen. *Introduction to Machine Learning with Python: A Guide for Data Scientists*. O’Reilly Media, 2016, p. 392.

[7] Aggarwal, Charu C. *Data Classification: Algorithms and Applications*. 1st. Chapman Hall/CRC, 2014. ISBN: 1466586745.

[8] Sotudeh, Hadi. “Predicting undesired business process behavior using supervised machine learning”. MA thesis. KTH, School of Electrical Engineering and Computer Science (EECS), 2018, p. 54.

[9] *Decision Trees — Scikit-Learn 0.22.2 Documentation*. URL: https://scikit-learn.org/stable/modules/tree.html.

[10] Dua, Dheeru and Graff, Casey. *UCI Machine Learning Repository: Iris Data Set*. URL: `https://archive.ics.uci.edu/ml/datasets/iris`.

[11] *Plot the decision surface of a decision tree on the iris dataset*. URL: `https://scikit-learn.org/stable/auto_examples/tree/plot_iris_dtc.html`.

[12] Breiman, Leo. “Random Forests”. In: *Machine Learning*. Vol. 45. 2001, pp. 5–32. URL: `https://doi.org/10.1023/A:1010933404324`.

[13] *Nearest Neighbors — Scikit-Learn 0.22.2 Documentation*. URL: `https://scikit-learn.org/stable/modules/neighbors.html`.

[14] Gandhi, ARohith. “Support Vector Machine — Introduction to Machine Learning Algorithms”. In: (2018). URL: `https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47`.

[15] *Support Vector Machines — Scikit-Learn 0.22.2 Documentation*. URL: `https://scikit-learn.org/stable/modules/svm.html`.

[16] *Support Vector Machines*. URL: `https://scikit-learn.org/stable/modules/svm.html`.

[17] Das,
Sibanjan and Cakmak, Umit Mert. *Hands-On Automated Machine Learning: A Beginner’s Guide to Building Automated Machine Learning Systems Using AutoML and Python*. Packt Publishing, 2018. ISBN: 1788629892.

[18] El Shawi, Radwa, Maher, Mohamed, and Sakr, Sherif. “Automated Machine Learning: State-of-The-Art and Open Challenges”. In: *CoRR* abs / 1906.02287 (2019). URL: `http://arxiv.org/abs/1906.02287`.

[19] *Hyperparameter tuning for machine learning models*. URL: `https://www.jeremyjordan.me/hyperparameter-tuning/`.

[20] Zöller, Marc-Andre and Huber, Marco. “Benchmark and Survey of Automated Machine Learning Frameworks”. In: (2020). URL: `http://arxiv.org/abs/1904.12054`.

[21] *Automated Machine Learning: State-of-The-Art and Open Challenges*. URL: `https://deeplearn.org/arxiv/79212/automated-machine-learning:-state-of-the-art-and-open-challenges`.

[22] *Scikit-Learn: Machine Learning In Python — Scikit-Learn 0.23.0 Documentation.* URL: `https://scikit-learn.org/stable/`.

[23] *Auto-Sklearn.* URL: `https://www.automl.org/automl/auto-sklearn/`.

[24] Kotthoff, Lars, Thornton, Chris, Hoos, Holger, Hutter, Frank, and Leyton-Brown, Kevin. "Auto-WEKA: Automatic Model Selection and Hyperparameter Optimization in WEKA". In: 2019, pp. 81–95. ISBN: 978-3-030-05317-8.

[25] Olson Randal S.and Moore, Jason H. "TPOT: A Tree-Based Pipeline Optimization Tool for Automating Machine Learning". In: *Automated Machine Learning: Methods, Systems, Challenges.* Ed. by Frank Hutter, Lars Kotthoff, and Joaquin Vanschoren. Springer International Publishing, 2019, pp. 151–160. ISBN: 978-3-030-05318-5. URL: `https://doi.org/10.1007/978-3-030-05318-5_8`.

[26] *H2O - Open Source Leader In AI And ML.* URL: `https://www.h2o.ai/products/h2o/#overview`.

[27] Kaushik, Saurav. "Feature Selection Methods | Machine Learning". In: (2016). URL: `https://www.analyticsvidhya.com/blog/2016/12/introduction-to-feature-selection-methods-with-an-example-or-how-to-select-the-right-variables/`.

[28] *Feature Selection — Correlation And P-Value.* URL: `https://towardsdatascience.com/feature-selection-correlation-and-p-value-da8921bfb3cf`.

[29] *Lofo-Importance.* URL: `https://github.com/aerdem4/lofo-importance`.

[30] *What Is Process Mining: Nearly Everything You Need To Know.* `https://processgold.com/process-mining-everything-you-need-to-know/`.

[31] Narkhede, Sarang. "Understanding AUC - ROC Curve". In: (2018). URL: `https://towardsdatascience.com/understanding-auc-roc-curve-68b2303cc9c5`.

[32] *Imputation of missing values.* URL: `https://scikit-learn.org/stable/modules/impute.html`.

[33] Hutter, F., Hoos, H. H., and Leyton-Brown, K. "Sequential Model-Based Optimization for General Algorithm Configuration". In: *Proc. of LION-5.* 2011, pp. 507–523.

[34] Balaji, Adithya and Allen, Alexander. "Benchmarking Automatic Machine Learning Frameworks". In: (2018). URL: https://arxiv.org/pdf/1808.06492.pdf.

[35] *Hands-on with Feature Selection Techniques: Filter Methods*. URL: https://heartbeat.fritz.ai/hands-on-with-feature-selection-techniques-filter-methods-f248e0436ce5.

[36] *Hands-on with Feature Selection Techniques: Embedded Methods*. URL: https://heartbeat.fritz.ai/hands-on-with-feature-selection-techniques-embedded-methods-84747e814dab.

[37] *Permutation feature importance*. URL: https://scikit-learn.org/stable/modules/permutation_importance.html.

[38] *Permutation Importance*. URL: https://medium.com/@vivek_skywalker/permutation-importance-a1df5010fa99.

Appendix - Contents

Appendix A

First Appendix

Result models of Auto Sklearn

```
[(0.520000, SimpleClassificationPipeline(
{'balancing:strategy': 'none',
'classifier:__choice__': 'random_forest',
'data_preprocessing:categorical_transformer:
categorical_encoding:__choice__': 'one_hot_encoding',
'data_preprocessing:categorical_transformer:
category_coalescence:__choice__': 'minority_coalescer',
'data_preprocessing:numerical_transformer:imputation:
strategy': 'mean',
'data_preprocessing:numerical_transformer:
rescaling:__choice__': 'standardize',
'feature_preprocessor:__choice__': 'no_preprocessing',
'classifier:random_forest:bootstrap': 'True',
'classifier:random_forest:criterion': 'gini',
'classifier:random_forest:max_depth': 'None',
'classifier:random_forest:max_features': 0.5,
'classifier:random_forest:max_leaf_nodes': 'None',
'classifier:random_forest:min_impurity_decrease': 0.0,
'classifier:random_forest:min_samples_leaf': 1,
'classifier:random_forest:min_samples_split': 2,
'classifier:random_forest:min_weight_fraction_leaf': 0.0,
'data_preprocessing:categorical_transformer:
category_coalescence:minority_coalescer:
minimum_fraction': 0.01},
dataset_properties={
  'task': 3,
  'sparse': True,
  'multilabel': True,
  'multiclass': False,
  'target_type': 'classification',
  'signed': False})),

(0.140000, SimpleClassificationPipeline(
{'balancing:strategy': 'weighting',
'classifier:__choice__': 'bernoulli_nb',
'data_preprocessing:categorical_transformer:
categorical_encoding:__choice__': 'no_encoding',
'data_preprocessing:categorical_transformer:
category_coalescence:__choice__': 'no_coalescense',
'data_preprocessing:numerical_transformer:imputation:strategy':
'most_frequent',
'data_preprocessing:numerical_transformer:
rescaling:__choice__': 'minmax',
'feature_preprocessor:__choice__':
'extra_trees_preproc_for_classification',
'classifier:bernoulli_nb:alpha': 8.25565461859145,
'classifier:bernoulli_nb:fit_prior': 'False',
'feature_preprocessor:
extra_trees_preproc_for_classification:bootstrap': 'True',
'feature_preprocessor:
extra_trees_preproc_for_classification:
criterion': 'gini',
'feature_preprocessor:
extra_trees_preproc_for_classification:max_depth': 'None',
'feature_preprocessor:
```

```
extra_trees_preproc_for_classification:max_features':
0.7706131584628054,
'feature_preprocessor:extra_trees_preproc_for_classification:
max_leaf_nodes': 'None',
'feature_preprocessor:extra_trees_preproc_for_classification:
min_impurity_decrease': 0.0,
'feature_preprocessor:extra_trees_preproc_for_classification:
min_samples_leaf': 17,
'feature_preprocessor:extra_trees_preproc_for_classification:
min_samples_split': 6,
'feature_preprocessor:extra_trees_preproc_for_classification:
min_weight_fraction_leaf': 0.0,
'feature_preprocessor:extra_trees_preproc_for_classification:
n_estimators': 100},
dataset_properties={
  'task': 3,
  'sparse': True,
  'multilabel': True,
  'multiclass': False,
  'target_type': 'classification',
  'signed': False})),

(0.080000, SimpleClassificationPipeline(
{'balancing:strategy': 'none',
'classifier:__choice__': 'k_nearest_neighbors',
'data_preprocessing:categorical_transformer:
categorical_encoding:__choice__': 'no_encoding',
'data_preprocessing:categorical_transformer:
category_coalescence:__choice__': 'minority_coalescer',
'data_preprocessing:numerical_transformer:imputation:strategy': 'mean',
'data_preprocessing:numerical_transformer:rescaling:__choice__':
'standardize', 'feature_preprocessor:__choice__': 'no_preprocessing',
'classifier:k_nearest_neighbors:n_neighbors': 1,
'classifier:k_nearest_neighbors:p': 2,
'classifier:k_nearest_neighbors:weights': 'uniform',
'data_preprocessing:categorical_transformer:
category_coalescence:minority_coalescer:minimum_fraction':
0.020258747699768806},
dataset_properties={
  'task': 3,
  'sparse': True,
  'multilabel': True,
  'multiclass': False,
  'target_type': 'classification',
  'signed': False})),

(0.060000, SimpleClassificationPipeline(
{'balancing:strategy': 'weighting',
'classifier:__choice__': 'liblinear_svc',
'data_preprocessing:categorical_transformer:
categorical_encoding:__choice__': 'one_hot_encoding',
'data_preprocessing:categorical_transformer:
category_coalescence:__choice__': 'no_coalescense',
'data_preprocessing:numerical_transformer:
imputation:strategy': 'median',
'data_preprocessing:numerical_transformer:
rescaling:__choice__': 'minmax',
'feature_preprocessor:__choice__': 'no_preprocessing',
'classifier:liblinear_svc:C': 1046.0348490788947,
'classifier:liblinear_svc:dual': 'False',
'classifier:liblinear_svc:fit_intercept': 'True',
'classifier:liblinear_svc:intercept_scaling': 1,
'classifier:liblinear_svc:loss': 'squared_hinge',
'classifier:liblinear_svc:multi_class': 'ovr',
'classifier:liblinear_svc:penalty': 'l2',
'classifier:liblinear_svc:tol':
0.0022764089677808903},
dataset_properties={
   'task': 3,
   'sparse': True,
   'multilabel': True,
   'multiclass': False,
   'target_type': 'classification',
   'signed': False})),

(0.060000, SimpleClassificationPipeline(
{'balancing:strategy': 'weighting',
'classifier:__choice__': 'passive_aggressive',
'data_preprocessing:categorical_transformer:
categorical_encoding:__choice__': 'no_encoding',
'data_preprocessing:categorical_transformer:
category_coalescence:__choice__': 'no_coalescense',
'data_preprocessing:numerical_transformer:
imputation:strategy': 'median',
'data_preprocessing:numerical_transformer:
rescaling:__choice__': 'none',
'feature_preprocessor:__choice__': 'random_trees_embedding',
'classifier:passive_aggressive:C': 0.0048687691040752885,
'classifier:passive_aggressive:average': 'True',
'classifier:passive_aggressive:fit_intercept': 'True',
'classifier:passive_aggressive:loss': 'hinge',
'classifier:passive_aggressive:tol': 0.0003591338920367281,
'feature_preprocessor:random_trees_embedding:bootstrap': 'True',
'feature_preprocessor:random_trees_embedding:max_depth': 7,
'feature_preprocessor:random_trees_embedding:max_leaf_nodes': 'None',
'feature_preprocessor:random_trees_embedding:min_samples_leaf': 3,
'feature_preprocessor:random_trees_embedding:min_samples_split': 19,
'feature_preprocessor:random_trees_embedding:
min_weight_fraction_leaf':1.0,
'feature_preprocessor:random_trees_embedding:n_estimators': 57},
dataset_properties={
   'task': 3,
   'sparse': True,
   'multilabel': True,
   'multiclass': False,
   'target_type': 'classification',
   'signed': False})),

(0.040000, SimpleClassificationPipeline(
{'balancing:strategy': 'weighting',
'classifier:__choice__': 'random_forest',
'data_preprocessing:categorical_transformer:
categorical_encoding:__choice__': 'no_encoding',
'data_preprocessing:categorical_transformer:
category_coalescence:__choice__': 'minority_coalescer',
'data_preprocessing:numerical_transformer:imputation:strategy':
'most_frequent',
'data_preprocessing:numerical_transformer:rescaling:__choice__':
'robust_scaler', 'feature_preprocessor:__choice__':
'extra_trees_preproc_for_classification',
'classifier:random_forest:bootstrap': 'False',
'classifier:random_forest:criterion': 'gini',
'classifier:random_forest:max_depth': 'None',
'classifier:random_forest:max_features': 0.4679570535467069,
'classifier:random_forest:max_leaf_nodes': 'None',
'classifier:random_forest:min_impurity_decrease': 0.0,
'classifier:random_forest:min_samples_leaf': 20,
'classifier:random_forest:min_samples_split': 20,
'classifier:random_forest:min_weight_fraction_leaf': 0.0,
'data_preprocessing:categorical_transformer:
category_coalescence:minority_coalescer:minimum_fraction':
0.010170223576999557,
'data_preprocessing:numerical_transformer:
rescaling:robust_scaler:q_max': 0.7172327458143168,
'data_preprocessing:numerical_transformer:
rescaling:robust_scaler:q_min': 0.25269243962434246,
'feature_preprocessor:
extra_trees_preproc_for_classification:bootstrap': 'False',
'feature_preprocessor:
extra_trees_preproc_for_classification:criterion': 'entropy',
'feature_preprocessor:
extra_trees_preproc_for_classification:max_depth': 'None',
'feature_preprocessor:
extra_trees_preproc_for_classification:
max_features': 0.864383179195224, 'feature_preprocessor:
extra_trees_preproc_for_classification:
max_leaf_nodes': 'None', 'feature_preprocessor:
extra_trees_preproc_for_classification:
min_impurity_decrease': 0.0, 'feature_preprocessor:
extra_trees_preproc_for_classification:
min_samples_leaf': 19, 'feature_preprocessor:
extra_trees_preproc_for_classification:
min_samples_split': 2, 'feature_preprocessor:
extra_trees_preproc_for_classification:
min_weight_fraction_leaf': 0.0, 'feature_preprocessor:
extra_trees_preproc_for_classification:
n_estimators': 100},
dataset_properties={
  'task': 3,
  'sparse': True,
  'multilabel': True,
  'multiclass': False,
  'target_type': 'classification',
  'signed': False})),

(0.040000, SimpleClassificationPipeline(
{'balancing:strategy': 'none',
'classifier:__choice__': 'passive_aggressive',
'data_preprocessing:categorical_transformer:
categorical_encoding:__choice__': 'one_hot_encoding',
'data_preprocessing:categorical_transformer:
category_coalescence:__choice__': 'minority_coalescer',
'data_preprocessing:numerical_transformer:imputation:
strategy': 'median',
'data_preprocessing:numerical_transformer:rescaling:__choice__':
'standardize', 'feature_preprocessor:__choice__':
'extra_trees_preproc_for_classification',
'classifier:passive_aggressive:C': 0.460578315916177l5,
'classifier:passive_aggressive:average': 'False',
'classifier:passive_aggressive:fit_intercept': 'True',
'classifier:passive_aggressive:loss': 'hinge',
'classifier:passive_aggressive:tol': 0.04557857428827514,
'data_preprocessing:categorical_transformer:
category_coalescence:minority_coalescer:minimum_fraction':
0.00027457445401600137,
'feature_preprocessor:extra_trees_preproc_for_classification:
bootstrap':'True',
'feature_preprocessor:extra_trees_preproc_for_classification:
criterion': 'gini',
'feature_preprocessor:extra_trees_preproc_for_classification:
max_depth': 'None',
'feature_preprocessor:extra_trees_preproc_for_classification:
max_features': 0.48190346970486964,
'feature_preprocessor:extra_trees_preproc_for_classification:
max_leaf_nodes': 'None',
'feature_preprocessor:extra_trees_preproc_for_classification:
min_impurity_decrease': 0.0,
'feature_preprocessor:extra_trees_preproc_for_classification:
min_samples_leaf': 17,
'feature_preprocessor:extra_trees_preproc_for_classification:
min_samples_split': 18,
'feature_preprocessor:extra_trees_preproc_for_classification:
min_weight_fraction_leaf': 0.0,
'feature_preprocessor:extra_trees_preproc_for_classification:
```

```
n_estimators ': 100},
dataset_properties={
  'task ': 3,
  'sparse ': True,
  'multilabel ': True,
  'multiclass ': False,
  'target_type ': 'classification ',
  'signed ': False})),

(0.040000, SimpleClassificationPipeline(
{'balancing:strategy ': 'weighting ',
'classifier:__choice__ ': 'passive_aggressive ',
'data_preprocessing:categorical_transformer:
categorical_encoding:__choice__ ': 'one_hot_encoding ',
'data_preprocessing:categorical_transformer:
category_coalescence:__choice__ ': 'no_coalescense ',
'data_preprocessing:numerical_transformer:
imputation:strategy ': 'median ',
'data_preprocessing:numerical_transformer:rescaling:__choice__ ':
'standardize ', 'feature_preprocessor:__choice__ ':
'extra_trees_preproc_for_classification ',
'classifier:passive_aggressive:C': 7.599213541504855e-05,
'classifier:passive_aggressive:average ': 'False ',
'classifier:passive_aggressive:fit_intercept ': 'True ',
'classifier:passive_aggressive:loss ': 'squared_hinge ',
'classifier:passive_aggressive:tol ': 0.06784385222631517,
'feature_preprocessor:extra_trees_preproc_for_classification:
bootstrap ': 'True ',
'feature_preprocessor:extra_trees_preproc_for_classification:
criterion ': 'gini ',
'feature_preprocessor:extra_trees_preproc_for_classification:
max_depth ':'None ',
'feature_preprocessor:extra_trees_preproc_for_classification:
max_features ': 0.5061717386466865,
'feature_preprocessor:extra_trees_preproc_for_classification:
max_leaf_nodes': 'None',
'feature_preprocessor:extra_trees_preproc_for_classification:
min_impurity_decrease': 0.0,
'feature_preprocessor:extra_trees_preproc_for_classification:
min_samples_leaf': 15,
'feature_preprocessor:extra_trees_preproc_for_classification:
min_samples_split': 2,
'feature_preprocessor:extra_trees_preproc_for_classification:
min_weight_fraction_leaf': 0.0,
'feature_preprocessor:extra_trees_preproc_for_classification:
n_estimators': 100},
dataset_properties={
  'task': 3,
  'sparse': True,
  'multilabel': True,
  'multiclass': False,
  'target_type': 'classification',
  'signed': False})),

(0.020000, SimpleClassificationPipeline(
{'balancing:strategy': 'weighting',
'classifier:__choice__': 'liblinear_svc',
'data_preprocessing:categorical_transformer:
categorical_encoding:__choice__': 'no_encoding',
'data_preprocessing:categorical_transformer:
category_coalescence:__choice__': 'minority_coalescer',
'data_preprocessing:numerical_transformer:imputation:strategy':
'most_frequent', 'data_preprocessing:numerical_transformer:
rescaling:__choice__': 'minmax', 'feature_preprocessor:__choice__':
'no_preprocessing',
'classifier:liblinear_svc:C': 21971.955867679357,
'classifier:liblinear_svc:dual': 'False',
'classifier:liblinear_svc:fit_intercept': 'True',
'classifier:liblinear_svc:intercept_scaling': 1,
'classifier:liblinear_svc:loss': 'squared_hinge',
'classifier:liblinear_svc:multi_class': 'ovr',
'classifier:liblinear_svc:penalty': 'l2',
'classifier:liblinear_svc:tol': 0.04708388731111063,
'data_preprocessing:categorical_transformer:
category_coalescence:minority_coalescer:minimum_fraction':
0.0822885300060565},
dataset_properties={
  'task': 3,
  'sparse': True,
  'multilabel': True,
  'multiclass': False,
  'target_type': 'classification',
  'signed': False})),
]
```