Institute of Parallel and Distributed Systems

University of Stuttgart
Universitätsstraße 38
D–70569 Stuttgart

Masterarbeit

ACP Dashboard: an interactive visualization tool for selecting Analytics Configurations in an industrial setting

Volga Y.

Course of Study:	Informatik
Examiner:	Prof. Dr.-Ing. Bernhard Mitschang
Supervisor:	Alejandro Gabriel Villanueva Zacarias, M.Sc.
Commenced:	12. June 2017
Completed:	12. December 2017
CR-Classification:	I.2.5, I.5.2

Abstract

The production process on a factory can be described by big amount of data. It is used to optimize the production process, reduce number of failures and control material waste. For this, data is processed, analyzed and classified using the analysis techniques — text classification algorithms. Thus there should be an approach that supports choice of algorithms on both, technical and management levels. We propose a tool called Analytics Configuration Performance Dashboard which facilitates process of algorithm configurations comparison. It is based on a meta-learning approach. Additionally, we introduce three business metrics on which algorithms are compared, they map onto machine learning algorithm evaluation metrics and help to assess algorithms from industry perspective. Moreover, we develop a visualization in order to provide clear representation of the data. Clustering is used to define groups of algorithms that have common performance in business metrics. We conclude with evaluation of the proposed approach and techniques, which were chosen for its implementation.

Contents

List of Abbreviations

ACP Algorithm Configuration Profile. 21

AJAX Asynchronous JavaScript And XML. 44

CSS Cascading Style Sheets. 62

DOM Document Object Model. 62

GA Genetic Algorithm. 10

HTML Hypertext Markup Language. 48

HTTP Hypertext Transfer Protocol. 48

ML Machine Learning. 9

OS Operating System. 59

QCD Quality, Cost, Delivery. 21

REST Representational State Transfer. 59

TTFB Time to First Byte. 62

URL Uniform Resource Locator. 59

1 Introduction

This section guides the reader through the main idea of this work. We motivate the topic discussed in a thesis (section 1.1) and give reasons why it is interesting from industry perspective. Next, follows a short discussion on previous researches of this topic (section 1.2). After that we define problems (section 1.3) that we encountered while examining related work. And based on that problems we set goals that we aim to achieve with our solution (section 1.4). Finally, we present the structure of the thesis (see section 1.5).

1.1 Motivation

Let us look at the application scenario of Machine Learning (ML) algorithms in an industrial setting. A typical product life cycle consist of multiple phases from the acquisition phase to the utilization and recycle phase [AG98]. Data is generated in each phase of the cycle to describe the product's status. For example: the acquisition phase success can be defined from the data about resources that were used to make a product and logs of construction process; the utilization phase success can be implied by product reviews written by end users or error reports. For that reason, manufacturers want to use ML to get insights of how business is running and which patterns it has. This helps to enhance business strategy and amend flaws.

We consider a scenario where data comes as unstructured text that might contain spelling, grammatic mistakes or shortened words. For example, a worker examines finished product on flaws and writes his observations as free text. Observations are unstructured data for a classification task. Each observation can be assigned one or more keywords or key phrases describing its context. After the products are classified based on keywords and key phrases, manufacturer can define what caused the flaws. The list of application scenarios only in industrial setting can grow further. The advantages of using ML instead of manual classification by domain experts are — very good effectiveness together with accuracy and also possibility to port this approach to different domains [Seb01].

To bring data to the same shape and classify it, we need to use ML techniques. At this step the user should decide which algorithm to use. Algorithm comparison requires a certain approach, due to diversity and complexity of the data. The method that allows to compare algorithms can make the decision-making process easier, or on the contrary, more complicated. Moreover, academic and industry perspective usually have different vision on algorithm performance. They are also concerned about disparate problems that appear during algorithm comparison. In our scenario, an approach should take into account points of interest for both, academic and industry perspective.

1.2 Current State of Art

There exist several approaches for comparing different solutions that resolve the same problem.

The most common one — statistical comparison of the algorithms (we also call it Naïve). It takes two paths: either to compare the performance metrics of the algorithms in a pairwise manner, or to run statistical tests in order to identify differences between the algorithms (discussed in section 2.1).

Another group of approaches is based on innovative generative design technique — Genetic Algorithm (GA) (see section 2.2). In GAs, the user is given a solution set (possible solutions) to the given problem. These solutions then undergo recombination and mutation, producing new children. The process is repeated over various generations, until the GA reaches the optimum. This approach usually requires certain constrains in order to yield suitable solutions and eliminate those, which do not fulfill user requirements.

Finally, some researchers use metadata of the algorithms for their comparison (see Section 2.3). The different from the statistical comparison is that in metadata approach, data about algorithm performance and execution, as well as metadata about data sets is compared. We found this approach faster than the statistical and genetic algorithm approach, as the authors did not need to run each and every configuration of algorithms.

1.3 Problem

The focus of this thesis is to develop an approach for comparing classification ML algorithms. It is inspired by and is a part of a profile framework described in [Vil17]. This approach should help to define which ML algorithm performs better, from the industry perspective, according to certain user-defined constraints. We create a solution, which allows managers to choose algorithms, and do not have to turn to knowledge of experts in ML. Since in our case solutions to the academic perspective's problem serve as a framework for industry perspective solution.

1.3.1 Academic Perspective

There are several points that should be taken into account, for example:

- The number of possible algorithm configurations. Multiple combinations of parameters and hyper parameters make number of suitable classification algorithms grow. It is unpractical to compare algorithms pairwise, as it requires a considerable amount of time and resources for algorithm execution.
- Data quality, cleanliness and completeness. Characteristics of data, on which classification algorithm were applied, should also be considered during comparison. Performance of the algorithms strongly depends on data — the same algorithm can give different performance results when applied to different datasets.
- Representation of results. Performance results are usually presented in a text form or tabular form, which makes them hard to perceive. It is not a problem, when only one metric should be taken into account. It is better to use different data representation in scenarios when the algorithms should be compared on multiple metrics. The growing number of algorithms only aggravates the situation.

The complex approach should be designed for dealing with the above-mentioned points. It should result in an accurate, unbiased and clear in representation comparison procedure for classification algorithms. We need to define ways how to efficiently compare algorithms with different configurations, take into account dependency of the performance on data. Last but not least, we should focus on giving a clear representation of results even when there are many algorithms to compare.

1.3.2 Industry Perspective

We can compare algorithm's performance using ML evaluation metrics that already exist and successfully used by machine learning experts [For03]. Such metrics as precision, mean squared error or f1-score, point to flaws and strengths of the algorithms. At the same time, they do not give information about how difference in ML evaluation metrics values affects important business key performance indicators of the company (also referred to as business metrics — see section 3.1). Execution cost, operating revenue, time to service values remain undefined for each algorithm. We consider two main issues which appear from industry perspective:

- Business evaluation metrics. User should be able to see which benefits could bring one classification algorithm or the other, and to know whether the chosen algorithm fits to the business strategy of the company. ML evaluation metrics could not act as decision factors from industry perspective, thus we need a mapping between ML and business metrics.
- Suggest the best option. In addition to that, it is important to give the user an opportunity to navigate through multiple alternatives. This means suggesting which algorithm configuration combinations lead to the desirable business metric values. We can imply that algorithms with common configurations may perform in a same way, so it will be possible to group them together. Such groupings (or clusters) can

explicitly show which algorithms to use, and how to configure them in order to achieve specific values in business metrics.

In addition, here, as well as in academic perspective subsection, it is important to present the business performance values in a human-friendly manner.

1.4 Goals and Questions

Our goal is to implement an interactive visualization tool that extends the concept of Automated Text Classification Configurations Performance Cube [Vil17] into an Analytics Configuration Performance Dashboard (ACP Dashboard). This tool should facilitate the conveyance of technical aspects of several analytical solutions from the industry perspective, as well as show the impact of certain configurations of algorithms on business indicators (KPI). For this, the implementation should provide information about 1) configurations that are currently available and their performance in terms of business metrics, 2) the factors that affect the efficiency of the algorithm and, 3) the corresponding potential compromises. Thus we aim to answer five main questions that show us direction for solving this problem.

1. Is there a way to compare algorithms which have multiple parameter configurations efficiently and quickly?
2. Which combinations of ML evaluation metrics for performance assessment can be mapped to the business metrics?
3. How to ensure that representation of the information would be readable and perceivable by human, especially when number of data points grows?
4. How to represent comparison results, so that the patterns in comparison data could be seen immediately?
5. Is it possible to identify and group algorithms which have common effects on business metrics?

The above-mentioned questions address problems from both, academic and industry perspective. We believe that by answering those questions we can develop a system which will add to the already known comparison approaches.

1.5 Structure of the Thesis

In the thesis we motivate a need to discover new ways for comparing classification algorithms and making it easier from business perspective. This thesis is structured as follows. In chapter 2 we point to the previous work done on this topic. In chapter 3 we discuss fundamental notions which helped to develop our work. In chapter 4 we introduce our approach which is aimed to solve the problems described in the chapter 1. In chapter 5 we present an ACP Dashboard and guide the reader through the implementation process. There we also describe results achieved with our approach. In chapter 6 chapter we conduct evaluation of our thesis, and compare expectations with actual achievements. We also discuss future work that can enhance our solution. In chapter 7 summarize presented approach.

2 Related Work

The problem of comparing ML algorithms comes every time when there is a need to decide which algorithm to choose for solving classification task. Scientists constantly discover effective ways to analyze and compare algorithms. Section 2.1 presents the most common approach for algorithm comparison — statistical approach. We discuss recent work that was done in that direction and emphasize difference from our approach. In section 2.2 we discuss GA and how it is used to provide solution space for a particular problem. We use the idea of GA for deciding which algorithm configurations is useful to compare. Another researches use metadata for the faster, but not so precise, comparison of the algorithms (introduced in section 2.3). We enhance this approach in our thesis, in order to develop a tool that facilitates process of algorithm's comparison.

2.1 Naive Approaches and Statistical Comparison

Problem of defining the best optimal solution from the solution space appeared to be not a trivial task. Some researchers used approach, which main idea was to execute machine learning algorithm on a various (but predefined) sets of data and compare values of such ML metrics as accuracy, time, precision, recall, etc..

Dogan and Tanrikulu [DT13] presented a comparative analysis of fourteen classification algorithms. They compared the algorithms on such metrics as accuracy, speed and robustness. Each of the algorithms was run on ten different preprocessed datasets. It was mentioned that some algorithms perform better, when continuous variables are binned to the intervals, which make continuous data discrete. However that does not change the behavior of the algorithms that performed well before binning. Authors conclude that preprocessing of the data significantly improves the results of algorithms performance. Also they claim that success rate of the algorithm depends on the dataset and its attributes. Authors proved that values of the metrics which describe efficiency of the algorithm, depend on datasets, since there are datasets which are easier to classify. The main disadvantage of this approach is that in their experiment authors compare algorithms on three ML metrics separately. This does not give them idea which algorithms in general (taking into account all possible metrics) perform better. Also authors presented a regression model to compare algorithms, however it is linear and thus must give an inaccurate representation of the situation and correlation between algorithm tuning and values of metrics.

George Forman [For03] in his *"An Extensive Empirical Study of Feature Selection Metrics for Text Classification"* discussed metrics which are used to compare classification algorithms.

Among them: accuracy, f-measure, recall and precision. The main point of his work was to consider existing ML metrics which will describe performance of the algorithm and conduct a study on feature selection. The author's goal is to find whether values of metrics are influenced by the number of selected for classification features. The study was performed considering a two-class classification problem, thus it does not deal with multiclass classification, which happens more often in industry. Also the datasets used for comparison are limited, and not all configurations of the algorithms are considered.

Kostiantis [Kot07] in his work gives an exhaustive comparison of the classification algorithms. He describes the main concepts and possible implementation of the algorithms, as well as specific tuning issues of the algorithms. The author shows a table where he compares algorithms, described in the paper, on multiple criteria, such as accuracy, speed, tolerance to missing values, redundant or irrelevant attributes. In the paper the usage of the algorithms is discussed along with the drawbacks and positive sides of each of seven groups of classification algorithms. Comparison is presented in a text and tabular manner and gives general performance overview for families of the classification algorithms.

Saaty [Saa08] presents The Analytic Hierarchy Process as a mean for comparing different solutions. The idea of the process is — to decompose the goal of the decision into a hierarchy. First author defines objectives to the lowest level, which will be also a set of alternatives. After that he compares parent node to all the leaf nodes and based on that comparison assign weights for parent and its children. This is done to every nod till the bottom of hierarchy is reached. The comparison of two possible solutions is based on the ranking given by the experts. This will result in a huge hierarchy with many levels and growth of comparison matrices. The Analytic Hierarchy Process is a relevant approach for comparing solutions with little alternatives, however there are some issues if one wants to use it to compare machine learning algorithms. For example, only pairwise comparison is possible, and also knowledge of the experts is needed to construct the ranking, which is something we want to avoid. Author decomposes the goal, revealing many parameters and hyper-parameters in each ML algorithm. We take the same way and also decompose algorithm settings in order to find which configurations influence performance.

Sze at al. [Sze+16] compare ML algorithms based on their performance on four different metrics: accuracy, energy consumption, throughput and cost, which are relevant for embedded ML. Authors express the belief that accuracy of the ML algorithm should be measured on large datasets, since only then we can get realistic results of the performance of the algorithms. They use publicly available datasets — such as ImageNet for testing. In the end, authors execute each algorithm on same datasets and compare metrics in order to find the most efficient solution. The main issue of this work is that eventually, only one ML metric is used to compare algorithms — accuracy.

Another scientist [Die98] in his work described statistical tests as a mean to compare algorithms. In total there were conducted five tests, two of which showed high probability of incorrectly detecting differences between algorithms. The other three however, based on cross-validation, did not have the issue of incorrect detection of the differences and were discovered to be more powerful in defining differences between algorithms. All tests consider accuracy (probability of predicting correct class) as the only metric on which machine learning algorithms for classification are compared. Moreover, tests are used to perform a pairwise comparison on algorithms which were executed on the same data. Comparing multiple algorithms at the same time seems impossible with statistical tests, as well as comparing the algorithms which were run on different data. The author also concluded that statistical tests cannot answer the question, which algorithm is the most suitable for a given task. The information which can be derived from the tests only describes the performance of the algorithms, but does not answer directly the aforementioned question.

[Dem06], in his article also used statistical test as a mean to compare machine learning algorithms. He performs the Wilcoxon signed rank test and the Friedman test for comparing two algorithms. Demsar addresses the issue with comparing more than two algorithms at once, and also comparing algorithms that were run on the different datasets. He constructs a setup with 40 industrial datasets and runs them on several common classification algorithms (C4.5, Naive Bayes) with their variations. His study shows that non-parametric tests are more suitable for comparing multiple algorithms. Results, however meaningful for a machine learning expert, do not map immediately onto business field, while showing only differences between algorithms. There is no explanation of what those differences may mean for business user.

The Garcia and Herrera [GHE08] expanded the study of Demsar and focused on statistical tests which can compare n times n classifiers. They performed all pairwise comparisons on five classifiers which were run on thirty data sets. The results of applying statistical test on these algorithms were rankings based on test accuracy by using a 10-fold cross-validation.

2.2 Genetic Algorithms for Solution Space Definition

Another approach for comparing machine learning algorithms, or more general — finding solutions, is to apply the GAs.

Gerber at al. [Jas+12] are using GA as an optimization technique for making the process of decision making easier. This paper is from architecture domain and focuses on a problem how to provide reasonable and suitable solutions on the early stages of design process. GA helps to manage large number of variables and provides a list of optimum solutions, which become a solution space. There is an opportunity to expand list of design solutions via adjusting various specifications and settings. New solutions can be fitted then to user's expectations. The Beagle tool developed by authors aims to provide designers with various design solutions created using GA. Although the tool was in development stage, it helped designers to work quicker. The use of GA may be, however, time consuming, especially when there are many configurations to consider [BI16]. Same approach used Lohan at el. [LDA17] and Lin & Lin-Chien [LL13].

Brander at al. worked on the Generplore model — which is a way to generate, explore and expand design concept. The key idea of it — is its preventive structure. Meaning that some promising sketches of design would be the base for further exploration and expansion in order to get satisfactory solution. The solutions derived from the Generplore, although considered (already) to be optimal for given criteria, are not the optimum end solutions. They are used as a base design solutions for further enhancements [BID14]. Liu et al. [LGL05] also used GA for constructing a multi-agent design system, which user to complete a certain design task. The genetic algorithm lies in the base of the agent and performs actions of inheriting, crossover and selecting alternatives from a solution space that fit user's requirements and constraints. William et al. [Wil05] took the same approach for designing antennas and fraud detection. The only difference is that int William's solution, intermediate results, created by GA are constantly evaluated by human and less relevant get erased. Previously stated works on genetic algorithms however did not consider this technique as a possible approach for comparing machine learning algorithms.

2.3 Meta-learning Approaches

Meta-learning is another approach for comparing algorithms and deciding, which algorithm is the best for a given problem.

Pavel B. Brazdil and Carlos Soares [Bra03] use a metadata about datasets, and performance of the algorithms on those datasets. The metaknowledge data is put through the k-Nearest Neighbor algorithm, which builds a (meta)model which can be used afterwards to predict performance of this algorithm on new problems. The author talks about three different points which help to define which algorithm better than another: 1. Using the ratios of the success rate (introduced in the paper). 2. Checking how the algorithms are ranked on different datasets. 3. Count with how many datasets one algorithm worked significantly better. The authors introduce a framework which creates a ranking of classification algorithms. The ranking is based on accuracy and time, as opposed to approach of this master thesis, where number of metrics on which algorithms are compared is minimum three. The ranking is presented to the user, so that he can decide which algorithm is better to apply for a particular task. In general the study offers more efficient rankings for the algorithms based on metadata than previous approaches.

There were also studies on comparison of ML algorithms based on error correlation. In his paper Alexandrous Kaluosis [KGH04] aims to find correlation between ML algorithms based on their performance on the same (one) dataset. He compares algorithms based on [newly] defined by himself performance metrics: relative performance (ranking of the algorithms, given by the number of points that each algorithm scored). The error correlation metric is created to find relations between algorithms. Author tries to find correlations between error rates of the algorithms in order to discover correlation between algorithms. The limitation of the paper lies in only one attribute used by authors to compare the algorithms (error correlation — defined in paper). However they claim that the metric number can be exchanged on user's demand.

Previous study of Aha [Aha92] also tries to find correlation between ML algorithms performance. The idea which author develops lies in finding rules that describe specific parameters of data. As author states himself, specific data parameters lead to particular performance metrics results. Aha concentrates mostly on features of data that cause differences in performance. The reason of difference is, however not considered in this research. The author also does not look for similarities between algorithms or correlation between those similarities.

Recent paper on text classification configurations [ZKM17] also touches the problem of defining the best suitable algorithm configuration for a text classification problem. Author considers that following configurations can be combined: a way to define feature set, a way to reduce feature set, and the algorithm settings itself. Based on this information, author created fourty Automated Text Classification Configurations and compared them. Results of comparison are used to define the best configuration. The performance of the algorithms is assessed by accuracy and time metrics. Performance values are shown in table and also plotted on a 3D scatter plot (called ATCC Cube). This enables comparing multiple configurations of algorithms at a time. We take the idea presented in the paper and develop it, by transforming the cube into an interactive web-application. Also we try to extend number of ML metrics which are used to compare algorithm performance and map them to the business metrics.

3 Background

In this section we provide information of basic notions which were used in order to build our solution. We start from business metrics and criteria that assess success of a business (section 3.1). After that we introduce Algorithm Configuration Profile (ACP) and its structure (section 3.2). We proceed to discussion on machine learning evaluation metrics and clustering techniques (section 3.3). Finally, we present various visualization approaches for multidimensional data (section 3.4).

3.1 Business Metrics and Balanced Scorecard

3.1.1 Success Criteria for IT Project Management

Quality, Cost, Delivery (QCD) — is the management approach for ruling the business. It is used to evaluate the various components of the production process. QCD gives feedback that helps managers make logical decisions in business strategy. Feedback is presented in form of facts and figures which make it easier for industrial companies to define priorities in their future goals. QCD proposes a method of evaluating business processes, which is applicable to simple and complex business processes [Woe10].

Quality. Quality is the ability of a product or service to satisfy and exceed customer expectations. The quality objectives are determined by customer requirements. It is considered to be one of the most important measures of business, because bad quality often leads to business failure. Effectiveness of the production process, consisting of workers, mechanisms and materials defines quality [TDL11].

We list several dimensions that are part of quality metric [GS14]:

- performance — describes operating characteristics of a product,
- conformance — shows how the product meets customer's expectations,
- aesthetics — assesses product's appearance, usually gives a subjective assessment,
- special features — additional, extra features of a product or service that can increase customer satisfaction,
- durability — how long the product can serve to the user, before it has to be replaced,
- reliability — refers to the time which product can survive without being repaired,

- serviceability — "Serviceability is defined by speed, courtesy, competence and ease of repair." [Gar87]

Cost Cost — is the amount of money that a company has to spend to design and produce a product or a service. The greatest cost in most commercial organizations is the cost of production [SSS12]. Production is directly responsible when it comes to monitoring and reducing production costs. We consider following types of production costs [SSS12]:

- raw materials,
- direct labor,
- expenses for taxes on property, insurance of buildings, renting of equipment etc..

3.1.2 Balanced Scorecard

The Balanced Scorecards (BSC) serve as performance measurement systems, first introduced in 1992. They are especially useful during the decision-making process. BSC helps organizations to enhance the process of monitoring company's operations, they make the strategy of the company transparent and defined. The BSC are designed to help in improving different business functions, both internal and external (design of the products, production, delivery, quality assessment, logistic, marketing, business strategies and many more) [BS07].

Balanced Scorecards are called balanced, because they allow to achieve business goals with three dominant constituents satisfied — customers, shareholders and employees. They aim to keep balance "between short term and long term objectives, between financial and non-financial measures, between lagging and leading indicators, and between internal and external performance perspectives" [BS07].

Companies successfully use BSC as a basis for defining a company's strategic system. With BSC managers and chief executive officers (CEOs) can align their business to new strategies. Based on BSC, business representatives can reduce the cost of operations, at the same time increase revenue, quality and values of the products or services [MDT99].

Performance measures are to provide the information on whether the chosen operations meet customer expectations and strategy objectives. It points out whether there is a necessity in improving certain areas which do not correspond to the requirements of a manager. The performance metrics of the BSC are focused around four strategic objectives: financial perspective, customer perspective, internal business perspective, innovation and learning perspective.

Since the BSC is a basic technique for measuring performance of the company, introduced a few decades ago, we used it to identify which metrics are potentially interesting for management. From this we defined which metrics are meaningful to show on the Analytics

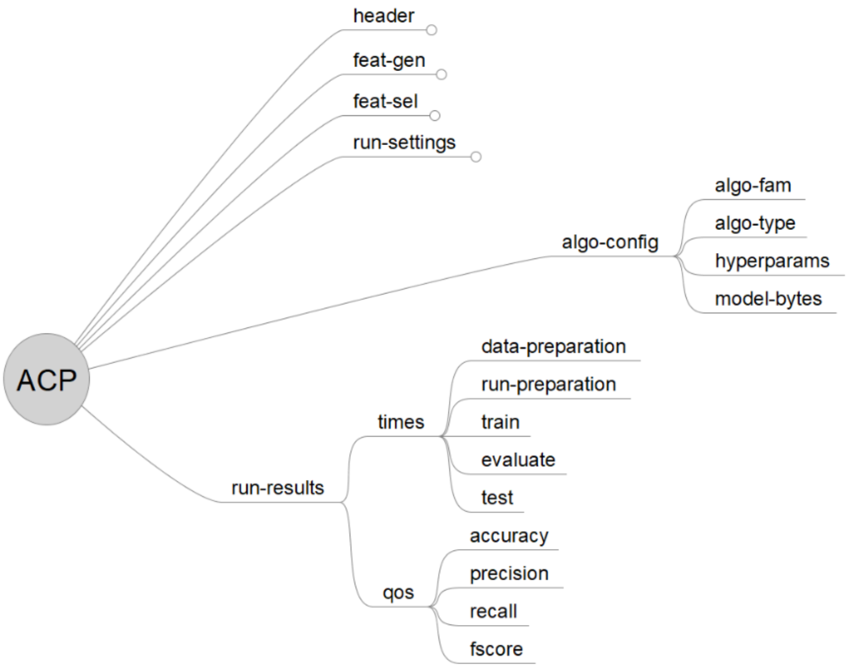


Figure 3.1: Mind map of the Analytics Configuration Profile [Vil17]

Configuration Profile Dashboard (Chapter 4). The successful usage of the BSC among companies [KN93], ensures that Business metrics for evaluating and comparing classification algorithms that we proposed, would be meaningful from business perspective. Hence it bridges the gap between mathematical metrics and business metrics.

3.2 Algorithm Configuration Profile

Let us describe information provided in the input metadata profiles, as well as underline which values we are using in our approach.

Analytics Configuration Profile (ACP) — contains metadata about performance of the algorithm. Before the model of the algorithm is actually built, there are two more stages that algorithm goes through. This stages are feature generation and feature selection — it is, basically, generating and selecting features that will be a basis for building the classification model. Based on those features, classification of new elements will happen. The structure of the profile presented on the **figure 3.1**. Here we explain some of the nodes of the ACP — in particular those that we will use further in our solution as a source of data.

The*feat-gen (feature generation)* node includes names of continuous and text features of the dataset, as well as their representation, and additionally weight scheme and description of the preprocessing pipeline for the text-features generation.

The *feat-sel (feature selection)* node gives a metadata about selection methods for continuous, discrete and text features. For each of the categories of features we are given the name of the method, parameters with which the method was executed, list of selected features and number of selected features.

Node *run-settings* provide data about settings which are applied before the algorithm is executed. This includes number in which iteration this particular ACP was executed (is it the first iteration, second or so on). Run settings also provides us with the information on how data was split: ratio of test set and type of split; cross-validation parameter (k-fold) and size of the training ant test sets in bytes.

The *algo-config* node gives information about family of the classification algorithm which was applied, type of the algorithm and also its hyper parameters. Those are settings of the algorithm. There is also information about how much disk storage the model takes, provided in bytes.

In the *run-results* node there is data about time and ML performance metrics of the algorithm. From the child node *time* you can get information about time which was needed to: prepare data, create data, to train the classification model, evaluate new instances and to test the model. In the *qos* child node ML metrics which correspond to the performance of the algorithm are provided: accuracy, precision, recall and f-score.

3.3 Machine Learning

Machine Learning techniques are a good help when it comes to finding patterns, establishing connections and relationships within data features. It helps to hand over computational difficulties to the machine, leaving to the human the process of evaluation and analysis. In supervised learning, data has labels which are used by algorithm to identify and classify new incoming data. In the unsupervised learning no labels are given, because the goal of unsupervised learning is to actually find those labels. One of the goals of my thesis is to help humans to choose which algorithm to use effectively and in correspondence to their requirements. The most common way to evaluate classifier — compute its accuracy. But there are of course other metrics that should be taking into account when evaluating the algorithms.

3.3.1 Machine Learning Evaluation Metrics

Accuracy is the most intuitive evaluation of algorithm effectiveness and quality. It describes the ratio of correctly predicted observations to the total amount of observations [SW10]. The common misconception is — the higher accuracy is, the better is the model. That is true only when the data sets are symmetrical, on other words if amount of false positive and false negatives is almost the same. Therefore, when assessing the algorithm it is also important to look at other evaluation metrics.

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

Where,

TN — number of negative examples, that were labeled correctly;

FP — number of negative examples, that were labeled as positives.

FN — number of positive examples, that were labeled as negatives;

TP — number of positive examples, that were labeled correctly.

Precision — is the ratio of positive observations predicted correctly to the overall amount predicted positive observations. High precision means that false positive rate is low [Pow11].

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

Where,

FP — number of negative examples, that were labeled as positives;

TP — number of positive examples, that were labeled correctly.

Recall shows the ratio of correctly predicted positive observations to the number of positive observations in a dataset [Pow11].

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

Where,

FN — number of positive examples, that were labeled as negatives;

TP — number of positive examples, that were labeled correctly.

F1-score is the weighted average of precision and recall. It takes into account both false positives and false negatives, it is recommended to use it when the ratio of false positives and false negatives is very different [For03].

$$F1 - score = \frac{2 \cdot (Recall \cdot Precision)}{(Recall + Precision)} \tag{3.4}$$

Loss functions for classifications represent the price paid for the inaccuracy of predictions in classification problems. The confidence of the prediction is measured in range from [0;1], and then the correct predictions are rewarded, and incorrect — punished according to the confidence of the prediction [Ros+03] [She05].

Area under ROC curve — measures performance of a binary classification problems [McC89]. ROC analysis helps to select possible optimal models and discard those which are not optimal, regardless of (and prior to) the class distribution. We do not consider this metric, because in our scenario we have a multi-class and multi-label classification problem, whereas ROC curve is designed to assess binary classification performance.

Accuracy paradox The accuracy paradox for predictive analytics says that predictive models with lower level of accuracy can yield better predictions than models with higher accuracy.

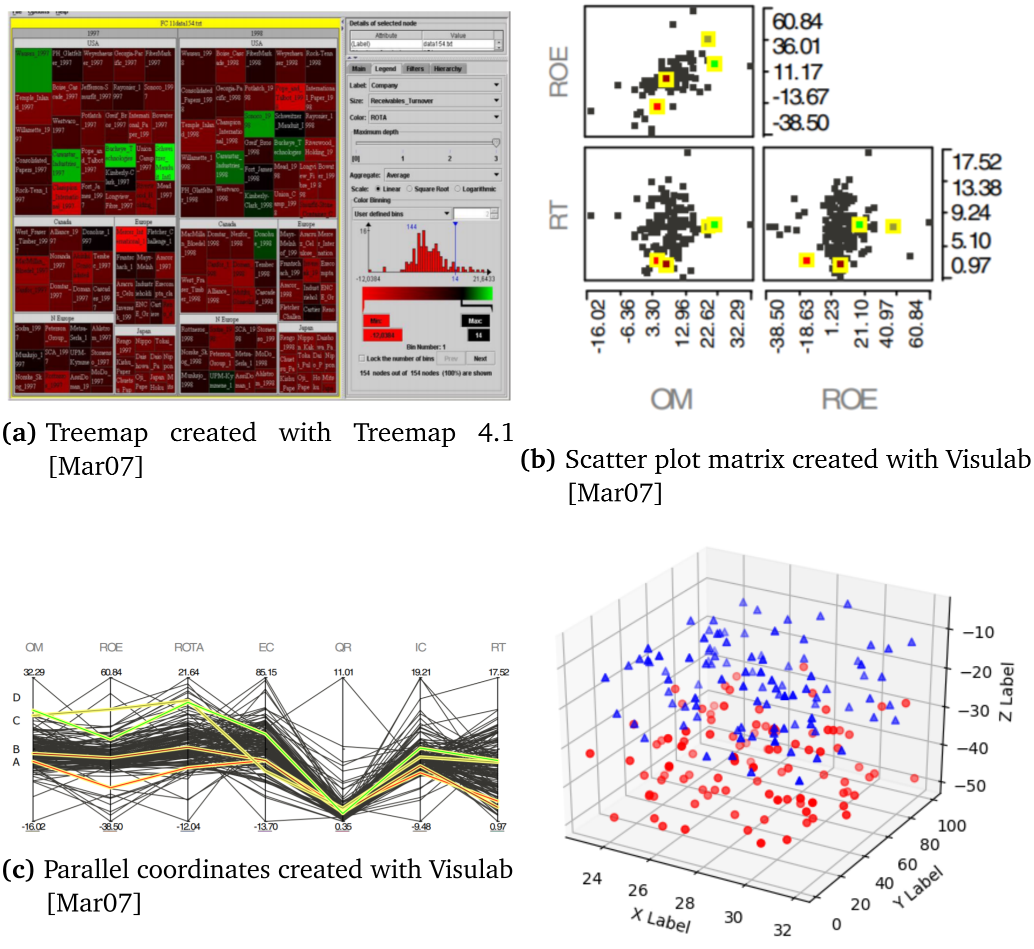
Let us assume that we have predictive model for detecting an insurance fraud [CT08]. Cases which are defined as high-risk by the model, will be investigated. The insurance company evaluates the performance of the model on a sample data set with 10,000 claims. It is already known beforehand, which out of 10,000 are fraudulent. The definition of accuracy, is shown below (see formula 3.4). Assume that our model predicted having $TN = 9,700, TP = 100, FP = 150, FN = 50$. In this case, according to the formula, the accuracy of the model will be $(9,700 + 100)/(9,700 + 150 + 50 + 100) = 98.0\%$. Now change the model and make it predict that there is "no fraud" ($TN = 9,850, TP = 0, FP = 0, FN = 150$), the accuracy value will become 98.5%. Although the model is bad by its idea (because it is always predicts "no fraud"), it has better accuracy than more correct model.

According to this we can say that when TN is less than FN, then accuracy will always increase when we tune the model to always output “negative” category. Conversely, if TP is less then FP, the same will happen when we change our rule to always output “positive”.

3.3.2 Clustering techniques

There exist multiple clustering algorithms, such as k-means, nearest neighbor, spectral clustering, mean-shift [Jai+99]. Among them there are parametric and non-parametric methods. For our approach we need a clustering algorithm that will require minimum input from the user, and this should be a non-parametric technique. Non-parametric techniques, unlike parametric, calculate how many clusters there must be in the data on their own. This is beneficial for us, because we do not want the user of ACP dashboard to input something else, except the ACPs.

The mean shift is a nonparametric clustering technique for determining the maximum of the density function. Imagine that there is a set of points in a two-dimensional space, and a circle or radius *r* — which is a kernel. mean shift algorithms shifts its kernel to a high density region, until it converges [Che95] [CM02]. Mean shift vector defines shift on each step. The vector points toward the direction where density is increased at most. On every step kernel moves to the mean of the points within area of maximum points density. The choice of the kernel defines method that calculates this mean. Algorithm converges when there is no direction at which kernel can accommodate more points inside itself.



(a) Treemap created with Treemap 4.1 [Mar07]

(b) Scatter plot matrix created with Visulab [Mar07]

(c) Parallel coordinates created with Visulab [Mar07]

(d) 3D scatter plot created with matplotlib version 2.1.0 [mat]

Figure 3.2: Multidimensional data visualization

3.4 Multidimensional Data Visualization

Visual analytics offers varieties of methods to represent data and still, it remains non-trivial and complex problem to visualize big volumes of multidimensional data. It is difficult because there is no exact notion about how many data user can perceive at once and the adjective "clear, understandable visualization" remains subjective. Before the user gets a comprehensible visualization, it may require to perform different permutations, such as dimensionality reduction, visualization techniques combination. However, there are principles and general ideas, which can be used while designing a visualization in order to help user's in perceiving a big amount data.

We compare such visualization techniques as scatter plot matrices, parallel coordinate plots and treemaps, since they appear most often in the literature [Ete+16] [Mar07]. They all, however, have significant limitations that question if those techniques could deliver "clear, understandable visualizations" [Tat+11][PWR04][Leh+12][Joh92].

Parallel coordinate plots (see **Fig.3.2c**), are good when the dataset does not exceeds more than one thousand rows and number of dimension is not more than dozen [Tat+11]. It does require additional learning from user, still it is intuitive enough. Together with interaction — for example brushing and linking, it becomes even more powerful as allows to combine multiple views of the same data. At the same time when amount of data gets bigger — the visualization becomes cluttered and it is then harder to understand correlations between points. One of the limitations of the parallel coordinate plots is that one can compare only two adjacent dimensions at a time, the ordering of dimensions should be specified beforehand. Parallel coordinate plots are well suited for displaying numerical data, but displaying categorical data with the parallel coordinate plots results in visualizations with lines concentrated in a few points of the dimension.

Scatter plot matrices (see **Fig.3.2b**) are easy to understand, first of all because they look similar to scatter plots which are known to everyone who have studied math at school. At the same time they are not efficient in terms of space consumption, if the number of dimensions goes over ten, the visualization already becomes too big and hard to perceive at a first glance immediately [Tat+11]. Scatter plot matrices designed to compare only two dimensions at a time. Nevertheless, scatter plot matrices can represent relatively big amount of data (compared to parallel coordinate plots), also scatter plot matrices are good instrument for finding clusters in data, because grouping of the points could be immediately seen on the visualization.

The *treemaps* is a modern visualization technique which is also sometimes used for displaying multidimensional data (see **Fig.3.2a**). They are hard to perceive and require additional learning for the user to be able to use them for data comparison [JS92]. With big amount of data, as any visualization it will clutter, however it might be, that readability of this visualization will stay high longer than of scatter plot matrix or parallel coordinate plots. This visualization is not suitable for categorical data, also it is hard to determine clusters using treemaps. However treemaps are good to apply in scenarios when the hierarchy information has to be visualized.

A *3D scatter plot* has all benefits of a scatter plot matrices (as it is an enhancement of a 2D scatter plot), it is simple to implement, intuitive and does not require an extra learning from user (see **Fig.3.2d**). It is well suitable for continuous data — which is in our case, values from business metrics. A 3D scatter plot is a visualization technique that is used to plot data point on three dimensions (X, Y, Z) at a time and aims to show dependency between dimension values. It is able to represent the same amount of data as scatter plot matrix and also facilitates pattern recognition and cluster definition.

4 Conceptual Solution

After careful analysis of a problem field and the results of predecessors we found out that previous solutions have some limitations (discussed in Related Work —– chapter 2) and could not be directly applied to the industrial setting problem (see chapter 1). Some of the approaches have obstructions in either number of performance metrics to be compared at once, visual representation, or — lack of mapping between business and ML evaluation metrics. This chapter describes our approach to the problem defined in chapter 1. We justify the metadata approach in section 4.1. Then we define business metrics on which algorithms will be compared. They are constructed from the ML metrics, which are extracted form the ACPs (see section 4.2). Next, based on the business metric values, that characterize ACPs we create a visualization. The goal of the visualization is — to make process of comparison easier and illustrative (see section 4.3). And last, but not least we describe a technique for defining common parameters of the algorithms 4.4).

4.1 Metadata Approach

We found the idea of using metadata of the classification machine learning algorithms as the most effective and beneficial, compared to other approaches (see chapter 3). In terms of time and data that is needed for comparison of the algorithms, the metadata approach outperforms the others. This approach addresses a question of dealing with the multiple combinations of the algorithm parameters. It requires a metadata of performance of the algorithm: algorithm settings, parameters and hyper parameters, machine learning algorithm performance values on a specific data set. Usually metadata is either taken from previous studies, or is constructed by execution of the algorithm.

For the metadata approach we do not need to execute multiple configurations of the algorithms. The idea is to take existing ACPs and use the metadata contained in them in order to find differences between algorithms. This allows us to omit execution of all possible combinations of the algorithms on certain datasets, thus saving time of execution and its cost. Of course, the drawback of this approach is that we might not have an ACP for the desired algorithm configuration. In that case running the algorithm with particular settings in order to get it's performance metadata will be inevitable.

We find it useful to present metadata of the algorithm in a structured way. It makes process of parsing and processing data easier. Such structure is suggested by Alejandro Villanueva, [Vil17] — which is a .json dictionary file, called *Algorithm Configuration Profile*. ACP provides metadata of the text classification algorithms, which were executed on industrial datasets (see section 3.2).

The metadata from the profiles is used to construct business metrics (see section 4.2), that in turn serve as basis for visualization and clustering (see sections 4.3, 4.4). We pick a subset of metadata values to conduct further analysis on the algorithm, as not all of the values can be used in our further calculations. In particular we are interested have ML performance metrics, and times from run-results. As those are values that contribute to the business metrics (see section 4.2). Other values, such as *cont_ft_repres, txt_ft_repr, cont_meth_name, algo_fam, algo_type* etc., contribute to the cluster names (see section 4.4).

Another important point of our approach — we compare algorithms that were executed ideally on the same (or similar) dataset. It is important because performance of the algorithm depends not only on its type and configurations, but also on the data set. If the data set is clean, complete, does not have NULL values, for instance, then we expect better performance than if the data set has a lot of NULL values, duplicates.

Using the pure metadata approach, though, does not give solutions for the problem of mapping ML evaluation metrics to business ones, or representing result in comprehensible visualization. Thus we applied some enhancements on the metadata approach, which allow us to solve the problems we have set in the introduction (section 1.3). To sum up, we say that the metadata approach is fast — if there are ACPs for desired algorithm configurations, then the only thing to do is — to compare those ACPs. The ACP provides metadata in structured representation which makes it easier to use this metadata for further analysis.

4.2 Business Metrics

The criteria for choosing the algorithm, is still defined by the business user from industry, so we have to ensure that metrics which are used for comparing algorithms will be useful from industry perspective. The comparisons of the machine learning algorithms is mostly done on values of the ML performance metrics, such as: accuracy, mean squared error, f1-score (for more details see chapter 3.3). Machine learning algorithm evaluation metrics (also ML metrics), however, do not clarify value of the classification algorithms from the industry perspective. Even experts who are supposed to be connected to the ML field do not find purely ML explanations useful [Cho+17]. This makes us think about how to present the machine learning metrics in form of business metrics.

The question is – which business metric can assess algorithm performance? We have to think about objectives that are most important for business. The literature research shows us that from the industry perspective, the benefit of a certain project or product can be evaluated by three indicators: quality, cost and time (or delivery if we talk about product). This notions were introduced almost fifty years ago [Ols71] and proven to be the most clear indicators of business success [Sta88][Atk99] [GS00][Bow+12]. Because of this mindset of people on the management, we decided to bring ML metrics into these particular terms

— cost, time, quality. Moreover, we also based out decision on the notion of Balanced Score Card (section 3.1) and our investigation of important business success criteria.

For this reason we design three business metrics: *Quality of Algorithm, Execution Time* and *Execution Cost* which can evaluate classification algorithms. Below we discuss in details how each of the business metrics is constructed and what purpose it has.

4.2.1 Quality of Algorithm

Gupta and Sushil mention several dimensions that model quality (see section 3.1). We decided that only *Performance* can be expressed by given ACPs data. Other dimensions are either irrelevant for algorithm assessment (aesthetic; durability – depends more on hardware than on the algorithm itself; serviceability — algorithm cannot break, thus does not require service; conformance — there are no such standards to which ML algorithms can correspond), or we do not have enough data in ACPs to express them (features, perceived quality, reliability). *Performance* dimension describes how good is product at performing functions it was designed for. In case of classification algorithm, the performance dimension is supposed to tell how accurate and precise predictions were.

There are accuracy, f1-score, precision and recall values in an ACP that indicate quality of the classification algorithm and can potentially be constituents of the *Quality of Algorithm* business metric. Accuracy, precision and recall measure different aspects of algorithm performance that is why it makes sense to use them both in the business metric formula. Moreover, because f1-score already contains precision and recall in itself, we do not include those values separately in the business metric formula (see section 3.3).

The *Quality of Algorithm* metric gives user notion of how accurate and precise results of classification are. It is calculated by the following formula:

$$Quality\ of\ Algorithm = \frac{(accuracy + f1score)}{2} \tag{4.1}$$

Where,

accuracy — indicates fraction of correct predictions out of all predictions that were made by the classification algorithm;

f1score — is a measure of algorithm accuracy, expressed in a balanced mean of precision and recall.

Although the accuracy itself is pretty straightforward and intuitive measure, it is not enough to evaluate the quality of algorithm's performance. There are situations when despite the high accuracy value, predictive model still can be useless (see chapter 3 on accuracy paradox). To deal with this case, the precision and recall metrics are used to indicate how many of the actual positive predictions were defined by the classifier, and how many of the predicted positives were actually positive. Accuracy and f1-score in an ideal case should aim to value one, range of values for these ML metrics is [0;1]. We sum the accuracy and f1-score values and normalize them by dividing by two. This will give us a *Quality* *of Algorithm* value, which depends on the accuracy, precision and recall (represented by f1-score).

4.2.2 Execution Time

Bhagwat in his paper introduced *customer query time* which shows time that a company need to provide customer with desired response [BS07]. We slightly modify this definition so that it fits our needs: we measure time that algorithm needs for performing data preparation and classification procedure on a given dataset and call this measurement *Execution Time*. Because in case when we are speaking of an algorithm, we want to know how long the algorithm has to run in order to yield a result.

The ACP profile has five different values in "times" node. The *Execution Time* metric should be constructed out of those values. We omit using data-preparation and run-preparation times, as they are applicable only for profiles that were created in first iteration. Those profiles which were created in other iterations (second, third, fourth, etc.) do not need data- and run-preparation time to be considered. To make all ACPs equal for comparison, we do not include those values to the formula.

$$Execution\ time = train + evaluate + test \tag{4.2}$$

Where,

train — time needed to build and train classification model;

evaluate — time needed to predict classes using classification model;

test — time needed to compare predicted classes with actual values (results in accuracy, precision, recall metrics).

Instead we use time for learning data and training the model (train), time which algorithm spend to perform cross-validation of results (evaluate), and finally time for comparing predicted values to actual values (test) — this is also a step where accuracy, precision and recall are calculated. The train, evaluate and test phases are mandatory for every ACP, regardless in which iteration they are created. This is why we include these times to the *Execution time* business metric's formula.

4.2.3 Execution Cost

The cost is meant to measure cost of resources that are needed to produce and deliver product to the customer [Dom15]. In our scenario such resource is a hardware that business needs in order to run the algorithm. Last years a tendency to outsource resources, or move business to the cloud has risen and it is expected that this phenomena will evolve [MNSS09][Mar+11][GSW12]. As we know, when using cloud services, user "rents" particular hardware configuration and pays some fixed price for every hour of usage. This is why our cost metric will include time, which algorithm requires to run and cost per hour for renting a hardware. We emphasize here that in order to make comparison even — the ACPs have to be run on the same hardware (so that their running time is measured in the same conditions).

We call business metric *Execution Cost* (also *Cost*), because we measure how much money managers have to spend for executing the algorithm and running it on a particular cloud hardware. Therefore, the cost metric strongly depends on execution time and cost of a rented hardware, and calculated by the following formula:

$$Execution\ Cost = (train + evaluate + test) \cdot Resource\ Cost\ per\ Hour \tag{4.3}$$

Where,

train — time needed to build and train classification model;

evaluate — time needed to perform cross-validation on results;

test — time needed to compare predicted classes with actual values (results in accuracy, precision, recall metrics);

Resource Cost per Hour — cost per hour of renting resource on a cloud service.

The *Resource Cost per Hour* in ideal case should be cost of a same hardware configuration that was used to compute ACPs — because execution time was computed for algorithms ran on a specific hardware configuration. Obviously, hardware configuration influences execution time, and with different hardware settings algorithm may perform faster or slower. The ACPs we compare in Thesis were created on the hardware with following configuration: 16 vCPUs with frequency 2299.998 MHz, 50 GB of RAM, 120 Gb of hard drive, OS: Ubuntu 14.04 trusty, x86_64 architecture. This can possibly correspond to a m5.4xlarge with 16 VPU, 64 Gb or RAM and Elastic Block Store[1] from Amazon Elastic Compute Cloud web service[2].

4.3 Visualization

Imagine you are given data which is represented in a text or tabular form. Your task is — to explore it and derive meaningful conclusions from your explorations. In case you do it manually, you will only succeed if data does not exceeds certain limits. However, when amount of data is so big that human mind fails to perceive it all at once, it becomes difficult to extract relevant information from data [Tat+11]. Thus, there is a strong need for methods that help user to find patterns in data in a faster and more efficient way.

Visual Analytics is a branch in a field of information visualization. It aims to provide user with tools which can aid in perceiving and exploring big volumes of multidimensional data. It helps the user quickly find patterns in data, immediately see its behavior and recognize

[1] https://aws.amazon.com/ebs/pricing/

[2] https://aws.amazon.com/ec2/pricing/on-demand/

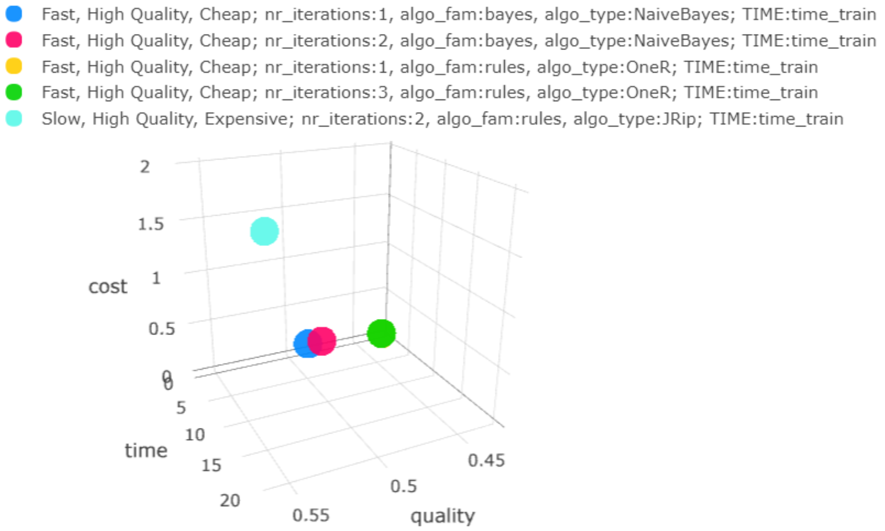


Figure 4.1: 3D scatter plot with cluster names and colored clusters

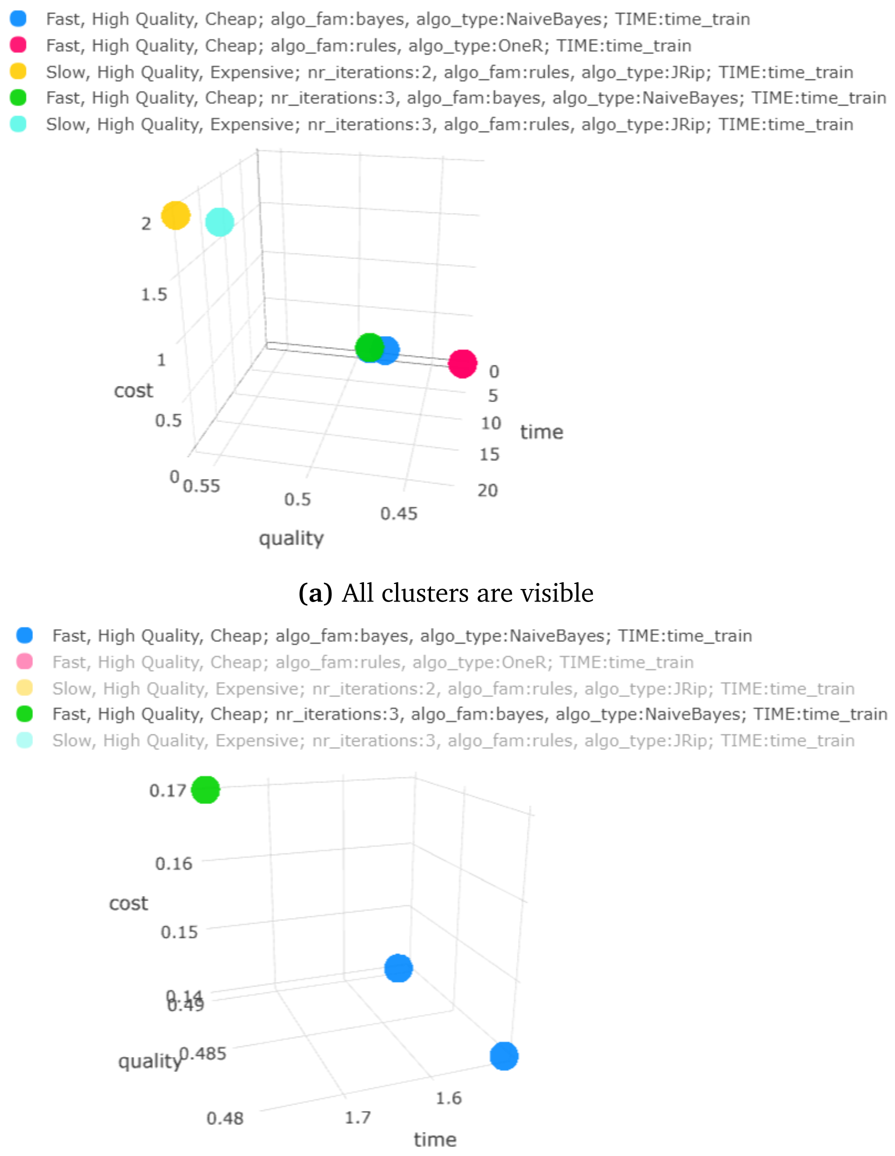
tendencies [Kei+08]. Visual representation of data, is, by default, easier to perceive than, for instance, tabular or text representation.

The *values* of the *business metrics* (see section 4.2) are data we want to visualize and represent to the user. It is continuous in its nature, have different ranges for each dimensions — from [0;1] to [0;∞+]. Distribution of data is significantly higher for *Execution Time* and *Execution Cost* metrics. That is why we also need a visualization that will scale good on dimensions, no matter if they have the same or completely different range of values.

Our idea is to implement an *Analytics Configuration Performance (ACP) Dashboard* [Vil17]. It is a tool for exploring, analyzing and comparing ACP performance. It consist of a visual part and control part (see section 5.2 for control part details). The visual part is built as a 3D scatter plot — also cube (see **figure 4.1**). The three business metrics (discussed in details in section 4.2) serve as three dimensions for a cube. Each metric represents one dimension.

3D scatter plot significantly overcomes other visualization techniques (discussed in section 3.4) and suits well to our scenario. Of course, static 3D scatter plot by itself does not give much opportunity to discover patterns and dependencies between data, find common points and behavioral specialties. That is why we enhance static 3D scatter plot by adding interaction to it. Interaction allows us, first of all, to overcome cluttering issue. Such options as — zoom in and out, rotate the cube and hide certain data points, give opportunity to examine the cube at the any angle. Even when some points are covering the other — simple change in orientation makes the view more clear. Moreover, if it does not help, user may blind out certain groups of points in order to see those, which are invisible (see **figure 4.2**).

On a 3D scatter plot, naturally only three dimensions can be shown at once and thus, compared. Interaction works as a solution to this problem as well. We enhance the 3D



(a) All clusters are visible

(b) Only blue and green clusters are visible

Figure 4.2: Demonstration of hiding clusters

scatter plot with a drop-down list that allows to substitute one dimension with another. This gives the user a new view on data, allows to chose which three dimensions to compare to each other (on detailed discussion about 3D scatter plot capabilities and its implementation check section 5.2).

4.4 Clustering

The visualization allows user position data points, which represent performance of the ACPs, on the cube and ensures easier comparison. To catch the behavior of data, patterns that appear, even interactive visualization is not enough. It could be, of course, done manually, by rotating the visualization and looking at it from different angles. The process becomes harder when some data points are so similar, that user requires additional actions (like zoom in and rotate) in order to see all the data points, or when the number of ACPs grows. Besides, manual processing of the data always implies errors caused by the human factor, which is undesirable, especially in an industrial scenario.

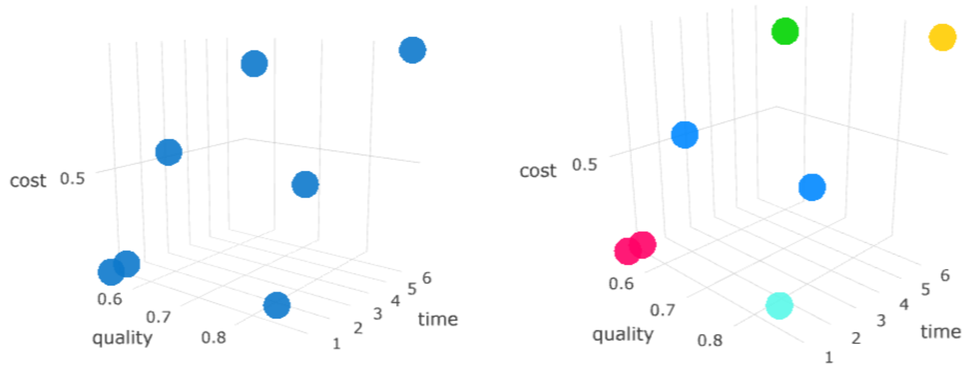
Clustering is a machine learning technique (for more details see chapter 3), which we use to define and group the ACPs, which values in parameters and hyper parameters have similar effect. Clustering algorithm significantly reduces time for defining and grouping ACPs which perform in a same way, according to the business metrics, which we defined above (see section 4.2).

There are, of course, other ways for defining similarities between the algorithms. For example, it could be possible to run the statistical analysis tests on each of the algorithm and based on that, recognize similarities or differences between algorithms [Die98]. This process will take time, as the dataset, on which algorithms are run, could be big and the execution time of the algorithms could differ. From the industrial perspective it is unpractical, especially when resources and time are limited.

We use mean shift clustering technique (read more in chapter 3) for finding ACP groups which are similar in their business metric values. We favored this technique among others, because it is non-parametric. This means that the user does not have to specify into how many clusters the clustering algorithm should group the dataset. The mean shift algorithm calculates number of clusters on its own, unlike k-means or spectral clustering. This could be also seen as a drawback, because when user has an opportunity to specify number of clusters, there is a chance to discover better number of clusters, than the algorithm suggests. At the same time, it requires additional time — to try and test different number of clusters.

4.4.1 Cluster formation

Input for clustering is a set of data points. Each data point represents a certain ACP via business metric values. To be more specific —- data point contains three values — one



(a) Clustering without color-coding

(b) Clustering with color-coding

Figure 4.3: Improvement in recognizing clusters with color-coding

for each business metric. Those values are calculated using formulas that map machine learning evaluation metrics and business metrics (see section 4.2).

Metrics values influence clustering. The mean shift algorithm is applied on a set of data points. Business metrics become attributes on which clustering happens. It depends on the effect of the attributes, which in its turn, depends on diversity of metrics values. First data is clustered based on *Execution Time* metric, as this metric appears to have highest influence on clustering. Then on *Execution Cost*, and finally on *Quality of Algorithm*. It is so, because *Execution Time* and *Execution Cost* have higher deviation, than the third metric. The *Quality of Algorithm* can only have deviation in range from [0;1] (by definition, see section 4.2) that is why its effect is the smallest.

Clustreing results and color-coding. Clustering algorithm results are — labels for the data points and a number of clusters. Each data point gets a label that indicates to which cluster it belongs. Resulting data set contains not only business metric values for each data point, but also a number of a cluster to which data point is assigned. Then, each cluster gets a separate color. This means that every data point is also colored to the corresponding color of the cluster. The color-coding helps immediately distinguish clusters on the cube and provides user with clear boundaries of a cluster (see **figure 4.3**). Colors save time when situation is dubious and it is not possible to define to which cluster the point belongs by simply looking at visualization. This process is, however, invisible to the user. In the end he only sees final 3D scatter plot with colored clusters on it.

Re-clustering. Clustering happens each time when a new ACP is added. In other words, a new ACP is not assigned to the cluster that already exists, but mean shift algorithm runs again and re-clusters data. This means that completely new clusters are formed, taking into account new ACP. It is necessary because user can input a profile which could be

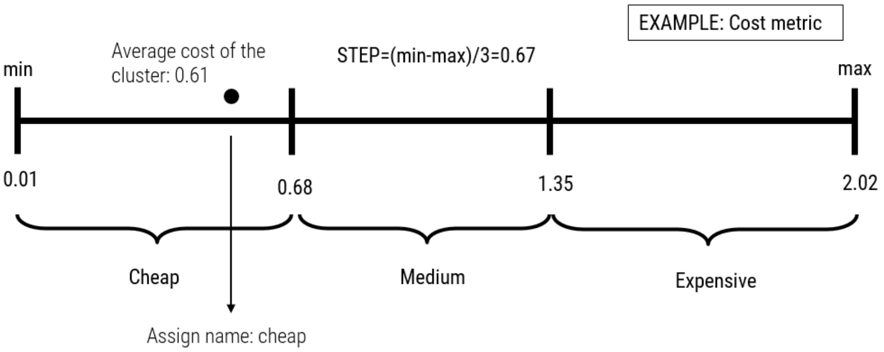


Figure 4.4: Procedure for assigning name for cluster from bins. Example on a cost metric

so different in its performance from the other ACPs, that it will require a new cluster. Moreover, the goal of clustering is not do define to which cluster a new point could belong. It is in finding which data points are so similar that they are able to form a cluster.

Minimum data points for clustering. At the beginning, when ACPs are given, user adds data points one by one. When is it time to start clustering? Which number of data points is sufficient for meaningful result? There is no rule that explicitly tells how many data points should be given to start clustering. However, the general suggestion is to have at least 2^n data points, where n — is a number of clustering variables (also attributes) [For84][Dol02]. This, however, is not a general suggestion and should be adjusted for each scenario separately. We start clustering when number of data points becomes 2^{n-1}. In our case, depending on the order of the ACPs, meaningful clusters can appear already with four data points (see discussion in chapter 5).

4.4.2 Cluster naming

The cluster name is a key detail that tells user what is common between ACPs that form a single cluster. It consists of three parts: bin names, common ACP attributes and their configurations, metric that influences clustering the most. Let us look in detail at every part of naming.

Bin names. We assume that on each dimension ACPs in cluster can perform either good, bad or on average. Such conditional division is intuitive and easy to understand. To create bins and find to which bin cluster falls, we perform following steps (see the schema of bin naming procedure on **figure 4.4**):

1. Find the difference between max and min values for a metric (f.e. cost).
2. Divide this difference value by three. This will be a step for three bins “Expensive, medium, cheap” for the metric — cost.

3. Compute average value on the dimension for each cluster. This average value will fit into one of the bins.
4. Assign cluster name according to its bin.
5. Do the same for other metrics (dimensions).

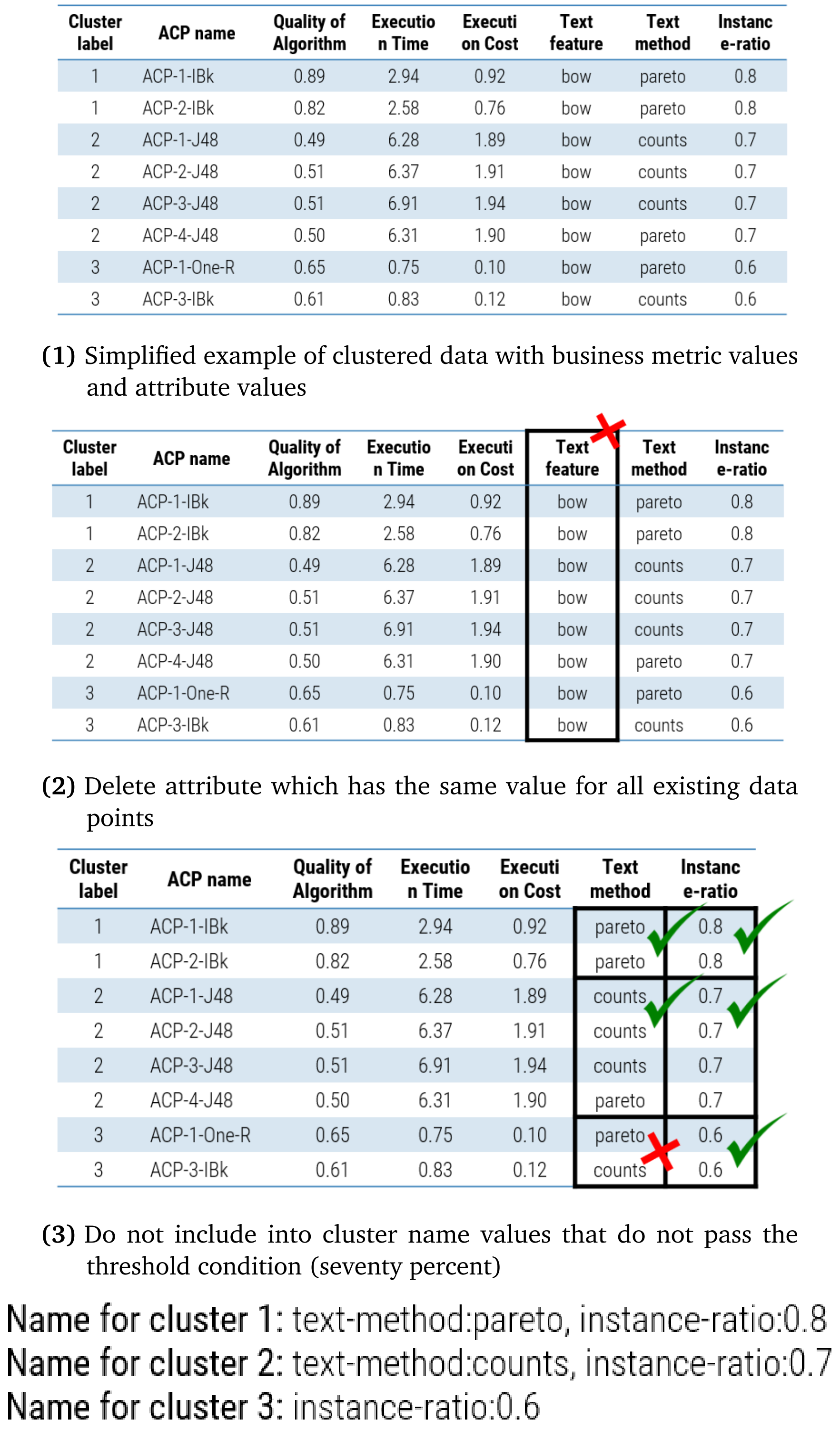
Example names of clusters after applying bin function: "*Fast, High Quality, Cheap*"; "*Slow, Medium Quality, Expensive*".

Common ACP attributes. Name of a cluster also depends on the number of common attributes that ACPs in that cluster have. When similar ACPs form a cluster, the name of a cluster will include all the configurations that were common among the ACPs of that cluster. At the same time, if different ACPs put into the same cluster, we cannot expect names to have many configurations. This is why the re-clustering after adding a new ACP is necessary. If we try to fit new ACP to the existing cluster, we might not find commong configurations. This will lead to a meaningless names for clusters. For constructing names for clusters out of common attributes we take following steps (see **figure 4.5**):

1. Delete parameters that have the same value in all data points – regardless to which cluster they belong. With this we ensure that we get rid of the parameters that could not influence performance results;
2. Check separately for each cluster, which parameters have the same value for all data points in a single cluster. This could be also seen as finding intersection between values for separate parameters;
3. If there exist a parameter which has the same value for seventy percent of data points in the cluster, we add parameters name and value to the name of the cluster. Seventy percent ensures us that we include in the name settings that share majority of ACPs.

Typical example of such names could be: "*algo_fam:bayes, algo_type:NaiveBayes, split:stratified wor, test-ratio:0.4001*"; "*cont_ft_repres:as-is month number, txt_ft_repr:bow, algo_fam:trees, algo_type:J48*".

The most influencing metric. After all, we also take into account business metrics and their influence on the clustering. Empirical study shows that the metric that has greater variance of values influences clustering the most. We find such metric, by extracting minimum value of the metric (over all data points) from its maximum value. The name of the metric is added to the cluster name. The next step – is to understand which component of the business metric, contributes to its value the most. We do it by simply finding the component which has the biggest value. For example, in *Execution Time* metric, the constituent *Time to Train* contributes the most to the metric value. The component is then also included to the cluster name.



Cluster label	ACP name	Quality of Algorithm	Execution Time	Execution Cost	Text feature	Text method	Instance-ratio
1	ACP-1-IBk	0.89	2.94	0.92	bow	pareto	0.8
1	ACP-2-IBk	0.82	2.58	0.76	bow	pareto	0.8
2	ACP-1-J48	0.49	6.28	1.89	bow	counts	0.7
2	ACP-2-J48	0.51	6.37	1.91	bow	counts	0.7
2	ACP-3-J48	0.51	6.91	1.94	bow	counts	0.7
2	ACP-4-J48	0.50	6.31	1.90	bow	pareto	0.7
3	ACP-1-One-R	0.65	0.75	0.10	bow	pareto	0.6
3	ACP-3-IBk	0.61	0.83	0.12	bow	counts	0.6

(1) Simplified example of clustered data with business metric values and attribute values

Cluster label	ACP name	Quality of Algorithm	Execution Time	Execution Cost	Text feature ✗	Text method	Instance-ratio
1	ACP-1-IBk	0.89	2.94	0.92	bow	pareto	0.8
1	ACP-2-IBk	0.82	2.58	0.76	bow	pareto	0.8
2	ACP-1-J48	0.49	6.28	1.89	bow	counts	0.7
2	ACP-2-J48	0.51	6.37	1.91	bow	counts	0.7
2	ACP-3-J48	0.51	6.91	1.94	bow	counts	0.7
2	ACP-4-J48	0.50	6.31	1.90	bow	pareto	0.7
3	ACP-1-One-R	0.65	0.75	0.10	bow	pareto	0.6
3	ACP-3-IBk	0.61	0.83	0.12	bow	counts	0.6

(2) Delete attribute which has the same value for all existing data points

Cluster label	ACP name	Quality of Algorithm	Execution Time	Execution Cost	Text method	Instance-ratio
1	ACP-1-IBk	0.89	2.94	0.92	pareto ✓	0.8 ✓
1	ACP-2-IBk	0.82	2.58	0.76	pareto	0.8
2	ACP-1-J48	0.49	6.28	1.89	counts ✓	0.7 ✓
2	ACP-2-J48	0.51	6.37	1.91	counts	0.7
2	ACP-3-J48	0.51	6.91	1.94	counts	0.7
2	ACP-4-J48	0.50	6.31	1.90	pareto	0.7
3	ACP-1-One-R	0.65	0.75	0.10	pareto ✗	0.6 ✓
3	ACP-3-IBk	0.61	0.83	0.12	counts	0.6

(3) Do not include into cluster name values that do not pass the threshold condition (seventy percent)

Name for cluster 1: text-method:pareto, instance-ratio:0.8
Name for cluster 2: text-method:counts, instance-ratio:0.7
Name for cluster 3: instance-ratio:0.6

(4) Final names for clusters

Figure 4.5: Example for cluster naming based on common ACP attributes

In the end "the most influencing metric" part of the name can look like: "*TIME:time_train*"; "*QUALITY:accuracy*"; "*COST:time_evaluate*" (for the full picture of a 3D scatter plot with clusters and full names check **figure 4.1.**)

To sum up, clustering defines ACPs that are similar on three dimensions — *Execution Time*, *Execution Cost* and *Quality of Algorithm*. Moreover it specifies which configurations have high or low quality, are cheap or expensive, are quick or slow. They serve as a pointer for set of configurations that will satisfy user requirements or bring user to the solution space.

5 Implementation and Results

This chapter describes the components and design choices involved in implementing the conceptual solution discussed in chapter 4. In section 5.1 we present general architectural solution of our implementation. In section 5.2 we discuss how the client side is constructed. In section 5.3. we deliberate on server logic and justify technology choice. In section 5.4. we introduce typical use cases of the developed tool.

5.1 Basic Logic and Functions of the Implementation

It was decided to design the ACP Dashboard as a Web-application, because once deployed, it is immediately reachable from any contemporary device which has a connection to the internet. While designing the prototype we stick to the 2-tier architecture of web-application designing patterns (see **figure 5.1**). The logical and data part are on the side of a server, and the visual representation — on a client part. Thus, server is responsible for the following functions:

- Communicate with the client (browser);
- Parse uploaded by the user ACPs;
- Store user data in a .csv format;
- Convert mathematical metrics to business metrics;
- Perform clustering on the input data and;
- Send the result of the clustering as data points for 3D scatter plot to the client side.

The client, on the other side, has to carry the following functions:

- Send user's ACPs to the server;
- Get clustered data points from the server;
- Visualize data points, received from server, on a 3D scatter plot;
- Enable possibility to explore the data.

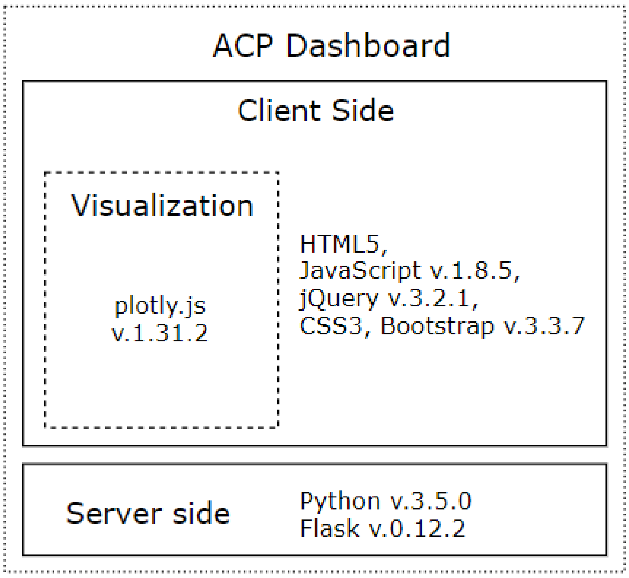


Figure 5.1: Architecture of the implemented solution

5.2 Client Side

The client side is built with typical technologies for front-end development: HTML 5, CSS3 and JavaScript (v.1.8.5).

- HTML — hypertext markup language for building the web-document layout;
- CSS — cascading style sheets for defining presentation of the elements of the web-document;
- JavaScript — script language for assigning functionality to the html elements, which makes the web-page interactive.

On top of that we use the open-source framework *bootstrap v.3.3.7* [1] — for styling following elements of our HTML page: buttons, drop-down lists, tables, and text elements. The bootstrap framework saves time, because we do not have to program the appearance of the elements of the web-page manually. This library already contains various CSS, HTML and even JavaScript templates for most of the objects of the DOM (Document Object Model).

The *jQuery v.3.2.1*[2] library helps to navigate, search, and handle elements of the web-document. It provides terms which make it possible to avoid long expressions of the JavaScript language. In our prototype we also use the jQuery notation to send and receive Asynchronous JavaScript And XML (AJAX) requests, establishing communication between the client and the server.

[1] https://getbootstrap.com

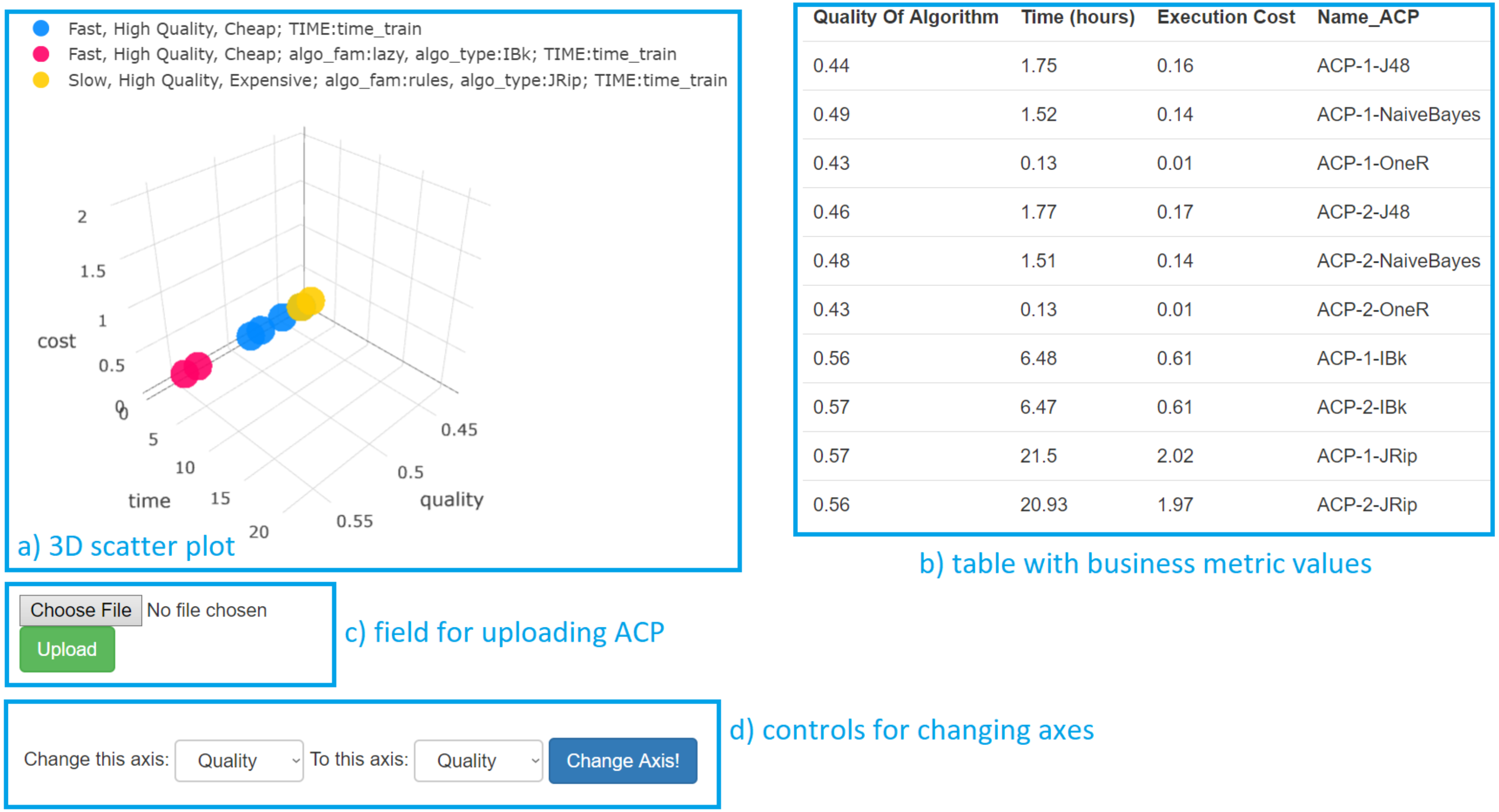
[2] https://jquery.com

The *plotly.js v.1.31.2*[3] library is one of the core technologies which we use for designing our prototype. It is based on the D3.js library for visualizing data with the help of HTML, CSS and SVG (Scalable Vector Graphics). It provides us with the interactive 3D scatter plot. Besides that, plotly allows such functionality as: plot rotation and zooming, coloring the points, hiding certain group of points, scaling axes range according to the data point values.

Both HTML page and JavaScript script are created by the server side (see section 5.3). The HTML page connects to the above-mentioned libraries (bootstrap, jQuery, plotly) via the `<head></head>` tag, this enables access to the libraries and its functions. The tag `<body></body>` contains different sub-tags, which correspond to the elements on a web page. The web page of our prototype consists of four main parts:

- 3D scatter plot (**figure 5.2-a**);
- Table with business metric values (**figure 5.2-b**);
- Field for uploading ACPs as a file in .json format (**figure 5.2-c**);
- A set of drop-down lists and a button to change the axes of the 3D scatter plot (**figure 5.2-d**).

[3] https://plot.ly/javascript/


● Fast, High Quality, Cheap; TIME:time_train
● Fast, High Quality, Cheap; algo_fam:lazy, algo_type:IBk; TIME:time_train
● Slow, High Quality, Expensive; algo_fam:rules, algo_type:JRip; TIME:time_train
cost
2
1.5
1
0.5
0
0
5
10
15
20
time
0.45
0.5
0.55
quality
Choose File
No file chosen
Upload
Change Axis!

a) 3D scatter plot

Quality Of Algorithm	Time (hours)	Execution Cost	Name_ACP
0.44	1.75	0.16	ACP-1-J48
0.49	1.52	0.14	ACP-1-NaiveBayes
0.43	0.13	0.01	ACP-1-OneR
0.46	1.77	0.17	ACP-2-J48
0.48	1.51	0.14	ACP-2-NaiveBayes
0.43	0.13	0.01	ACP-2-OneR
0.56	6.48	0.61	ACP-1-IBk
0.57	6.47	0.61	ACP-2-IBk
0.57	21.5	2.02	ACP-1-JRip
0.56	20.93	1.97	ACP-2-JRip

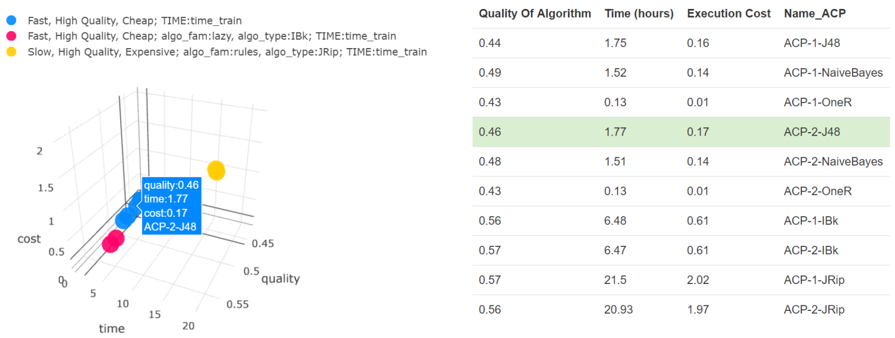
b) table with business metric values

c) field for uploading ACP

Change this axis: Quality To this axis: Quality

d) controls for changing axes

Figure 5.2: ACP Dashboard interface


● Fast, High Quality, Cheap; TIME:time_train
● Fast, High Quality, Cheap; algo_fam:lazy, algo_type:IBk; TIME:time_train
● Slow, High Quality, Expensive; algo_fam:rules, algo_type:JRip; TIME:time_train
quality:0.46
time:1.77
cost:0.17
ACP-2-J48
cost
0
0.5
1
1.5
2
time
0
5
10
15
20
quality
0.45
0.5
0.55

Quality Of Algorithm	Time (hours)	Execution Cost	Name_ACP
0.44	1.75	0.16	ACP-1-J48
0.49	1.52	0.14	ACP-1-NaiveBayes
0.43	0.13	0.01	ACP-1-OneR
0.46	1.77	0.17	ACP-2-J48
0.48	1.51	0.14	ACP-2-NaiveBayes
0.43	0.13	0.01	ACP-2-OneR
0.56	6.48	0.61	ACP-1-IBk
0.57	6.47	0.61	ACP-2-IBk
0.57	21.5	2.02	ACP-1-JRip
0.56	20.93	1.97	ACP-2-JRip

Figure 5.3: Demonstration of highlighted row

Scatter plot visualizes ACP data points. The point's position is based on its value in each of the business metrics introduced in chapter 4 (*Quality of Algorithm*, *Execution Time* and *Execution Cost*). The color of the point depends on the cluster, to which the algorithm was assigned. Each metric is used as a dimension. The code for creating the scatter plot and data on it is produced on the server side (section 5.3). As said before — it is possible to change view on the cube by rotating it, zooming in and out.

For each data point there is a corresponding row in a table (figure 5.2b). The row consists of four columns: quality of algorithm, time(hours), Execution cost and name of the ACP. When the user hovers on the data point on the cube, one row in the table gets highlighted. This indicates the row which corresponds to the data point, which was hovered (see figure 5.3)

The file-browse field (**figure 5.2-c**) allows user to upload his ACPs to the server. The files should be in .json format, uploading file that does not have ACP structure (see section 3.2) will lead to an error. After the file is uploaded, the new point on the 3D scatter plot appears. It corresponds to the recently uploaded profile. After the button "Upload" is clicked, server receives the request from client to catch the data — new profile in our case. Further processing of the files takes place on the server side. It parses the .json file and produces a data point with business metrics (see section 5.3).

Finally, the drop down list and a button (**figure 5.2-d**) are designed to change one dimension of a cube to another. One of the advantages of our prototype is the possibility to interchange dimension. Changing of dimension helps to discover new patterns, also it enables comparison of data points on a new combination of three dimensions. The user can chose which dimension should be substituted and change it to another dimension by clicking the button. The script behind the button will trigger the action that will rebuild the cube, and appearance of points will change according to the new dimension.

5.3 Server Side

Flask[4] — is a python web-service framework. Which forces us to a specific technological choice — python programming language. However, it is supported by an active community, well documented and is constantly developed and updated. Authors claim it to be extensible. Which means that even if at the beginning you create simple two-layered application (presentation and application layer), you can easily extend it by adding a database layer. Moreover, you can find multiple extensions (such as cache support, HTML-builder, user session management, mail, etc.) in the *Flask Extensions Registry*, or develop an extension on your own.

We used python v.3.5.0 programming language to implement the server side. The server itself was provided by flask (v.0.12.2), a web framework for creating or client-server applications. Installation of flask only is sufficient for creating an application. No extra tools or extensions (such as Hypertext Transfer Protocol (HTTP) or HTTP libraries) are needed. It already contains functionality for database access, http requests, ajax processing. The source code of the application consists of three python files:

- `main.py` — main file that should be executed in order to start the flask application; contains functions for generating the html page of the web-application;
- `extra_functions.py` — contains functions for generating JavaScript file, computing business metrics and writing the algorithm configurations to the .csv file;
- `clustering_functions.py` — contains functions for clustering, defining name of clusters.

After the flask-application is started, it can be reached via the localhost. At first, if no data points (ACPs) were given previously, the Hypertext Markup Language (HTML) page shows empty 3D scatter plot and empty table with business metric values.

When a new ACP is added via the input fileupload buttons added via file-browse fields (figure5.2-c), the client side sends the request to the server side. The request is processed by the function `my_form_post()` from the `main.py` file. Flask reaches uploaded files via request method. The content of the uploaded ACP (algorithm configurations) is saved on the storage space of a server in the folder "`UPLOAD_FOLDER`". We decided to save this functionality because further on there could be a need to check which files were uploaded. Module *os* is used to save .json file — ACPs on the disk space of a server.

We parse uploaded .json file with the python library `json`. It creates a dictionary object out of the .json file. The dictionary object is easy to traverse, we have immediate access to the values of ACPs. From dictionary object of an ACP we extract the following values:

- Feature generation:

    Continuous features: Representation (of Time and Score);

[4] http://flask.pocoo.org/docs/0.12/license/

Text features: Representation and weight scheme;

- Feature selection:

  Continuous method: Name and parameters;

  Discrete method: Name and quantiles;

  Text method: Name and instance-ratio;

- Algorithm configuration:

  Algorithm family and algorithm type;

- Run settings:

  Number of iteration;

- Run results:

  Times: Data preparation, run preparation, train, evaluate, test;

  Math metrics: Accuracy, precision, recall, f1-score.

After the values from ACP are extracted, we compute business metrics values (see the formulas of the business metrics in section 4.2). We form the file `ACP_configurations.csv` and put extracted values and computed business metrics there. In this way we efficiently store metadata about the ACP profiles, as well as the business metrics data. The `ACP_configurations.csv` file is a convenient input for the clustering functions, as it could be easily transformed into a pandas data frame.

We used python library `pandas` (version 0.21.0) for data manipulation and analysis. Pandas data frame is a tabular data structure with labeled rows and columns. It is convenient, because all manipulations can be done with SQL-like functions (filtering, adding rows or columns, merging, dropping columns etc.). Also pandas data frame serves as a perfect input for clustering functions.

The python library `sklearn` (version 0.19.1) provides us with clustering functions. In our implementation we use a mean shift clustering (see section 3.3.2), because it is a non-parametric clustering algorithm. This means that user does not have to specify number of expected clusters in advance. The mean-shift clustering requires a data set as the only input. It returns clusters and number of clusters when done.

The `numpy` library (version 1.13.3) — is a scientific computing package. We use it for transforming some of the columns of a pandas data frame to the python list data structure, also for calculating average, min and max values in lists.

We transform information from the `ACP_configurations.csv` into a pandas data frame. We perform clustering on the data frame fields, but first specify which columns should be taking into account when forming the cluster. In our prototype those columns are: quality of service, time to service, cost. We cluster the ACPs based on their business metrics values, because we want to know which algorithm configurations performed in a same way.

After all the calculations are done, server side generates HTML page and JavaScript. The business metric values serve as x,y,z coordinates for the data points (ACPs) on the cube. The clustering result serve for coloring data points. Data points which belong to the same cluster will be colored in the same color. Finally, names of the clusters serve as names of the colored points. Generated HTML page is then sent to the client (browser) and represented to the user. The whole process of reading the ACPs calculating business metrics and computing clusters does not take more than one second. So the user does not fill the sophisticated process which takes place behind the data visualization.

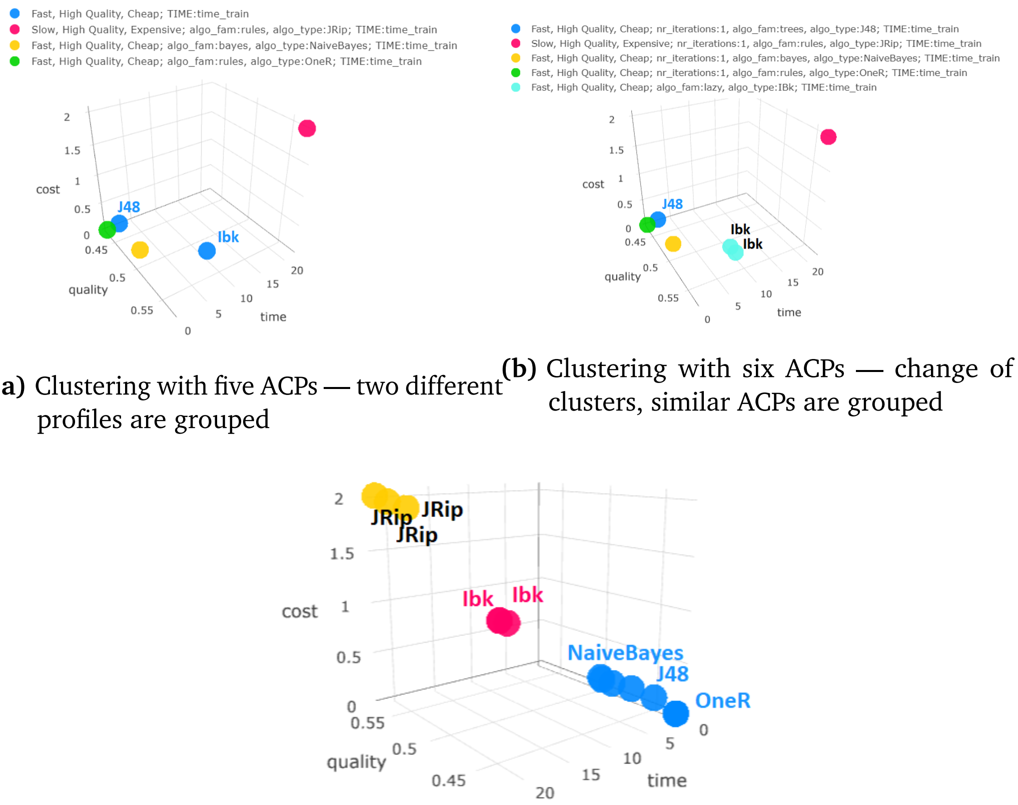
5.4 Implementation Use Case

In this section we would like to introduce use cases of the ACP Dashboard. We assume that given some amount of algorithms that were executed on the same dataset. The user wants to compare the algorithms, using the ACP Dashboard.

Typical use case. Data from the Amazon Food Reviews data source[5] was used to create ACPs, which serve as an input to the ACP Dashboard and are used for comparison. It consists of 568,454 food reviews which contain different information, among it: productID, text and summary of the review, helpfulness score of the review. Text classification algorithms were executed to predict usefulness of the review. Based on performance of the algorithms, the ACPs were constructed. As a result, we got eighteen ACPs for the following algorithms: Naive Bayes (four profiles), JRip (three profiles), IBk (four profiles), OneR (three profiles), J48 (four profiles). Each of the profiles is a .json file (for structure see section 3.2) which weights 14 Kb.

After the visualization part is completed (i.e. user visualized all the ACPs he had), the comparison can begin. The user can compare algorithms by looking at their position on the plot. We could say that the fastest, best quality and cheapest algorithms are those, which appear in the left far down corner of the cube. We start from adding five different ACPs — Ibk, J48, JRip, NaiveBayes, OneR. They cluster in four clusters, and Ibk with J48 are grouped in one cluster (see **figure 5.4a**). At this point we say that clustering is meaningless, as it does not comply with the suggestion that the Execution Time causes points to cluster (see section 4.4) After that we continue adding ACPs, and after adding another Ibk, we see that cluster Ibk-J48 does not exist anymore, as the clustering algorithm grouped both Ibk profiles together (see **figure 5.4b**). Situation changes dramatically, when we add second J48 to the cube — all the data points cluster into three groups (see **figure 5.4c**). From this clustering we can already distinguish — JRip algorithm is the one that costs more than other and is very slow, but at the same time it has the best quality. Algorithms from the blue cluster — Naive bayes, OneR and J48 are the fastest and the cheapest algorithms, but at the same time their quality is the lowest. Ibk algorithm are somewhere between two

[5] https://www.kaggle.com/snap/amazon-fine-food-reviews [ML13]



(a) Clustering with five ACPs — two different profiles are grouped

(b) Clustering with six ACPs — change of clusters, similar ACPs are grouped

(c) Resulting three clusters after adding more data points

Figure 5.4: Number of data points influences clustering

clusters — they are not the cheapest, but have better quality than algorithms from the blue cluster and are faster than JRip. Now user can decide, according to specific requirements, which algorithms is better to use. For example, if the user does not care about time and money, and wants to have accurate results, he should probably chose JRip, or Ibk algorithm. The ACP Dashboard narrows a solution space and eliminates algorithms that perform not in a desired way.

Below we describe two peculiar cases that we discovered when comparing ACPs.

Minimum number of data points. General process of working with our tool can be seen on the interaction diagram (**figure 5.5**). Everything starts with an empty cube, and user gradually adds ACPs on it and discovers new patterns. If number of points is four and more, after each new added point, the clustering algorithm will run again. This might (but does not necessarily have to) affect visualization and naming of the clusters. It could be that after adding one point, number of clusters will change, depending on the new point characteristics. In case if the 3D scatter plot (also cube) was empty before the first point was uploaded, no clustering will be performed.

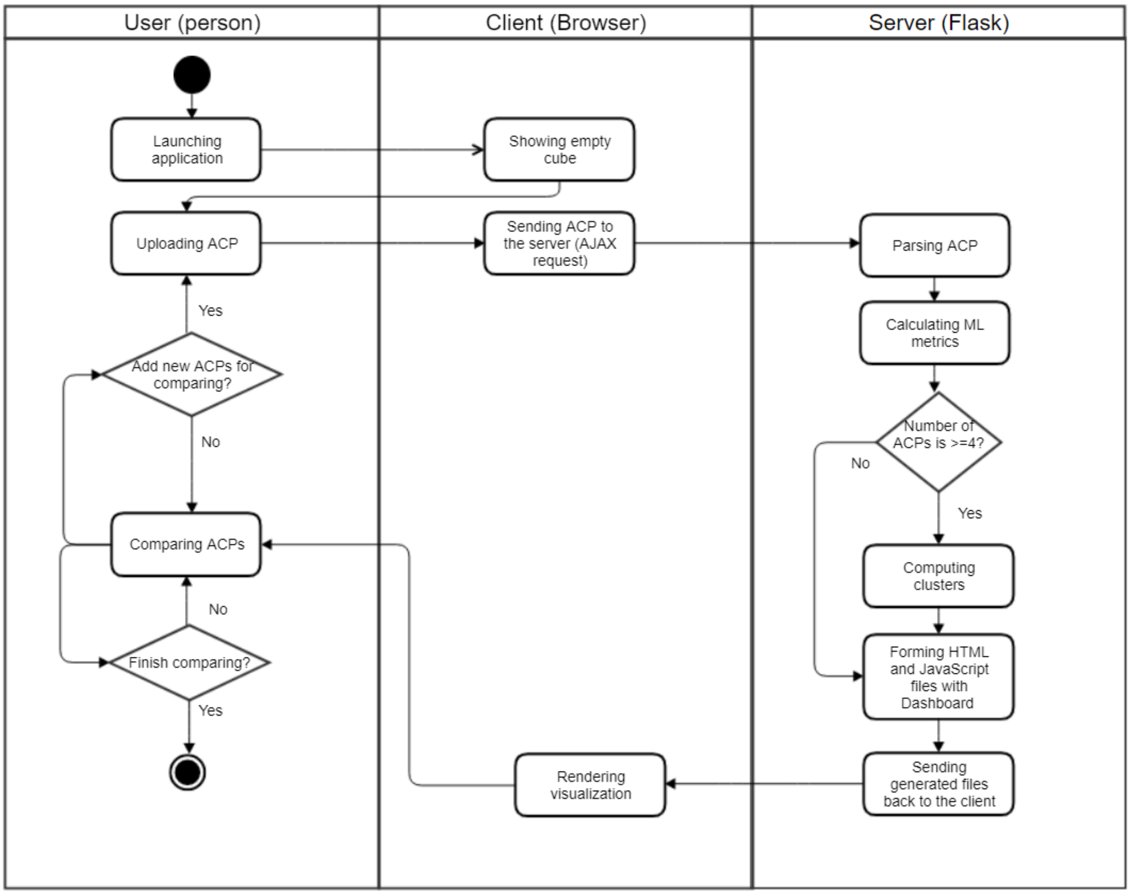


Figure 5.5: Interaction diagram

When testing the use case scenario, we observed interesting features of our solution. Number of data points influence cluster formation and their usefulness. Indeed, experiment shows that clusters formed out of five data points might not have much sense (see **figure 5.4a**), whereas with fifteen ACPs clear picture can be seen (see **figure 5.4c**).

Still, clustering result strongly depends on data which is clustered. Imagine a situation where we feed algorithm with two types of ACPs — for example, Naive Bayes and IbK. Then even four data points will already bring clear clustering (see **figure 5.6**): Naive Bayes ACPs form one cluster (blue) and some of the IbKs — the other (yellow and pink). Therefore we decided to start clustering data points if their number is more or equal to four. First of all, because there is not clear rule of when to start clustering, except of the suggestion discussed above. Secondly — clustering even on early stages (aka when number of data points is less then 2^n) gives a hint which algorithms will lead to the desired result. From the **figure 5.4c** we can definitely say that for cheaper solution user should prefer Naive Bayes over Ibk. At the same time, if the requirement is — to aim for quality, then the Naive Bayes would not be the best option and the user have to consider IbK algorithm.

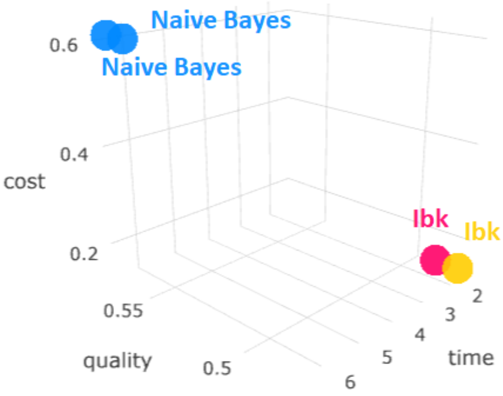


Figure 5.6: Clustering with four data points — Naive Bayes and IbK ACPs form three clusters

Cluster	Values of *Execution Time*	Name from bins: Time
1	[1.75, 1.52, 1.77, 1.51, 1.7, 1.78]	Fast
2	[6.48, 6.47, 6.47]	Fast
3	[0.13, 0.13, 0.12]	Fast
4	[21.5]	Slow

Table 5.1: Cluster names based on *Execution Time* metric

Cluster names — metric that influences the most. Cluster names are designed to help users to identify and group algorithms which have common effects on business metrics. The result of the naming procedure is: every cluster gets a name, which contains in itself bin names for each of the business metric, names of the common configurations for the same cluster and name of the metric that influences clustering the most (see section 4.4). We find that this naming gives detailed information on similarities of algorithms. At the same time there are use cases when naming is not efficient enough. For example, consider four clusters (see **table5.1**). It shows values of *Execution time* metric for each cluster. The element of an arrays — is value of this metric for a single data point. Third column of the table shows which name is assigned to the cluster, according to the bins. Three clusters have name "Fast", and one cluster has name "Slow". However, *cluster two* could have had name **“Medium”** instead. Obviously *clusters one* and *cluster three* are much faster than *clusters two* and *four*. However, *cluster two* is not as slow as *cluster four*, thus we find this bins naming inappropriate.

6 Evaluation and Future Work

In this chapter we are about to discuss quality and value of our solution to the problem introduced in the beginning of the thesis (see section 1.3), and find which enhancements could be done to deal with unsolved issues. We assess answers to each of the question of the problem, check whether all the requirements are fulfilled. We start with evaluation our approach in general (section 6.1). We also evaluate chosen architectural solutions and ponder on whether in complies to the thesis definition (see section 6.2). This chapter is also focused on checking whether the solution shows satisfactory results on from functionality and performance perspective (section 6.3). Moreover, at each point we discuss possible enhancements of the given solutions and future work. Our evaluation of concept uses analytical and descriptive evaluation methods [Hev+04].

6.1 Concept Evaluation

We start concept evaluation from looking at the goal and questions that were defined in the first chapter of this work (see section 1.4). Our goal is to assess whether chosen approach allows to achieve goals that were set and whether it is applicable in the industrial scenario.

6.1.1 Meta-Learning Approach Evaluation

First question that we put in front us, was about defining a method that will allow to compare classification algorithms efficiently and quickly. Among the existing approaches (see chapter 2) we took the way of meta-learning. Data about algorithm performance is structured and kept in ACPs. The profiles. Obvious advantages of this approach are:

- assuming that we already have ACPs, comparison takes very little time, as we do not have to run all the algorithms to get performance data;
- if the scenario allows, there is no need to have all configurations of the algorithms, it is enough to have extremely different configurations, to decide in which direction is better to move;

At the same time, it is not a universal method and has following issues:

- there should be ACPs for every configuration that the user wants to compare;
- meta-data requires to be in an ACP form;

- depends on data, on which the classification algorithm was executed;
- depends on hardware on which the ACP was created.

Some of the issues listed above, can be solved. For example, the missing ACPs can be estimated from the existing profiles. This will save time — because there will be no need to run algorithms in order to get profiles, on the other side the estimated profiles could be not one hundred percent accurate. At this point managers should decide whether they need extremely precise results, or it is enough to have estimated performance, just to limit the solution space.

Moreover, at the beginning it is better to take ACPs of completely different algorithms and configurations. In that case we assume that each ACP will perform differently and appear in completely distinct positions on the cube. This is helpful for the further analysis, it sets boundaries to a solution space.

To deal with a problem of data dependency, we can use a meta data of the data, on which algorithm was executed. Especially, because it is given as a (*Data Quality Profile* by [Vil17]). In that way, it could be possible to compare the algorithms which were executed on the different data set. There just has to be a method to make ACPs even, depending on their DQPs.

6.1.2 Business Metrics Evaluation

Another question was about projecting ML algorithm performance evaluation metrics to the business metric. We presented three metrics: *Quality of Algorithm, Execution Time* and *Execution Cost* (see section 4.2). We find them the most suitable and intuitive, as they are often main objectives, on which performance of a service is accessed. Of course, our solution can be developed further, and more sophisticated business metrics (such as Customer Satisfaction or Service Level, etc.) could be added.

We constructed *Quality of Algorithm* business metric out of accuracy and f1-score ML metric. Having accuracy and f1-score together is a good decision, because accuracy alone is not enough — it may sometimes give misleading results (see section 3.3). At the same time those metrics show performance of the algorithms from different perspectives that is why using both of values in business metric is justified. However, such metrics as logarithmic loss and area under ROC curve could also bring more value to the *Quality of Algorithm* metric. Or, they could be mapped independently onto other business metrics. The constituents of the *Execution Time* metric, result from the data which is available from the ACP profiles. Except of the times that are in the metric, we think that time for predicting new values should also be included. In this way user will no how much time will be needed to predict values, if the model is already known and built.

Finally, the *Execution Cost* metric serves to show potential cost fro using particular algorithm. It depends strongly on *Execution Time*, and as a result — hardware on which the algorithm is executed. This created a lock, because the better hardware it is — the faster *Execution Time*. It is not yet clear how to manipulate this metric so that it value gives an optimal results. Would it be better to reduce costs for renting the hardware, but prolong time, or the other way around? Enhanced version of the *Execution Cost* might also include time which is need to predict certain amount of new values. The *Cost per Hour* parameter can be split up — separately for the resources which are on the cloud, and for resources that user has on premises (if some).

Also, regarding the third issue from subsection 6.1.1, data could be included into comparison process. Thus, business metric formulas can also take data into account — at least time and cost can be calculated per row, or per some template size of data fro every ACP. Performance, unfortunately still cannot be measured equally for ACPs, because not only data size, but its structure, number of features depend on the classification algorithm results. We could go further and construct a ranking or suitability score for algorithms, allowing not only compare algorithms on business metric, but also show the user which configurations performed good on certain metrics.

6.1.3 Visualization Evaluation

Another goal of this work was to provide the user with an clear and intuitive data representation, regardless of number of points that should be compared. We decided to visualize data on a 3d-scatter plot which has interaction features. Main problem of other visualization techniques were that they either took too much space, could allow to compare profiles on only two dimensions at a time, were not intuitive and required some pre-learning for user to start using them. The benefits of using a 3d-scatter plot are so far:

- intuitive, does not require extra-learning from user;
- simple to implement;
- allows to compare ACPs on three dimensions at a time;
- does not take much space;
- with interaction can avoid cluttering, easy to visualize.

The drawback of this type of visualization is that only three dimensions could be shown at a time. However, with the help of JavaScript and, again, interaction we can change dimensions. Thus, if there were not three business metric (that serve as dimensions for a cube), but four and we would like to change them — user can use set the dimensions which have to be shown manually (see section 5.2). Another possible issue of a 3d-scatter plot — that we do not know to what extend points on it can be added. mostly, it depends on data whether the visualization will look cluttered or not. Thus, we expect that more than one hundred points would be already too much for this kind of visualization. However, more detailed evaluation, with artificial data points should be performed for clarifying this issue.

6.1.4 Clustering Evaluation

Clustering is used to discover patterns in the data and define the ACPs that behave in a same way. We use it as an alternative to statistical analysis (see section 3.3). It groups the profiles that are more similar to each other, according to three business metrics — *Execution Time*, *Execution Cost* and *Quality of Algorithm*. Clustering has following benefits:

- fast;
- does not require input from user;
- can cluster points on multiple dimensions (more than three);
- considers details of data that human might not think of.

Clustering is fast, compared to other approaches. It does not involve human for assigning data to the clusters, machine is doing all necessary calculations, and with today's hardware capability, clustering of one hundred points of not very high-dimensional data takes not more than 5five seconds [Sci]. The clustering algorithm we have chosen — mean-shift (see section 4.4) is a non-parametric clustering technique. This means that user does not have to specify number of clusters to be defined, algorithm will calculate suitable number of clusters independently. Moreover, clustering allows us to compare data points on multiple dimensions at a time — even more than three. What is even more important, because of mathematical background and distance formulas it gives more accurate results than could be achieved, if a human would define clusters. Capabilities of our brain are limited to some extends and sometimes it is impossible to take into account all the details in data. At the same time, machines do not have such limits thus can yield accurate and credible results. Although clustering has significant advantages, there are disadvantages that also should be taken into account:

- no explicit reason of clustering;
- needs certain amount of point to give meaningful clusters.

Clustering on its own does not give clear idea why ACPs cluster. To understand why data was clustered in one way or another, we needed to explore results, make and prove suggestions, and understand how particular clustering algorithm works. Moreover, as already discussed previously, clustering algorithm needs certain amount of data in order to give meaningful results (see section 4.4).

As for the future work, we think that it would be good to provide re-clustering of certain groups. What we mean by that — after the clustering on all data points is done, focus on a specific cluster, and try to re-cluster points within that cluster. This will give the user more information on profiles that form the same cluster. Identify if within one cluster exist profiles that have even more similar performance. Another enhancement of the clustering would be using a dimension reduction technique. It reduces amount of dimensions and leaves only those that are meaningful for clustering and influence it. This is especially useful, in cases when there are many dimensions and many data points. The performance of the clustering algorithm becomes better: fast analysis and more meaningful clusters. This step is a good future work that will make our approach even more effective.

6.2 Architecture Evaluation

In this subsection we discuss architecture choices and check if the implementation satisfies requirements that were defined by thesis definition.

6.2.1 Non-Functional Requirements

One of the prerequisites of this master thesis was to develop a web-application that will be service oriented. This means that first of all, the service should be provided to the user through the network. It should be independent from vendors, technologies that user uses to access the service, and also should be available remotely. Moreover, the task was also to make a prototype compliant to the Representational State Transfer (REST) properties. This includes such important aspects of REST as client-server architecture, statelessness, cacheability, layered system, uniform interface [RR07] [Fie00].

Our solution is a web-service with a two-tier architecture. It has a server side (written in python language) and client side (written in JavaScript and HTML). It is independent from vendors as python application can run on any Operating System (OS) — Windows, Linux, Macintosh. User can reach the client-side (aka HTML web-page) using any of the modern browsers — from Microsoft Edge to the Google Chrome. The service (ACP Dashboard) can be accessed remotely — through the Uniform Resource Locator (URL), as well as it can be executed on a local machine. Our web-service is also a black box because user does not see computations that happen on the server side. Business metric equations as well as clustering procedure are not exposed to the user. Only some part of business logic are known, though. For example, user may know that clustering will start after four ACPs are given for comparison. As soon as the ACPs are loaded, their representation — in form of points on a cube, appear.

We did not reach cacheability as it is meant in its original concept. However, we do store the ACPs that were given previously. Even when the web-application is restarted, all the ACPs that were send to server before the restart, are saved in a special .csv file. This prevents user from processing the whole bunch of points once again, which means that user can continue comparison from the moment where it was stopped.

Stateless property of the RESTful services is implemented via using the HTTP protocols. Each request to the server (aka sending ACP, receiving the HTML page) is complete on its own, and has all the information that is needed for processing the request.

Our implementation is a layered system, which consists of two layers. They are independent from each other and can be substituted by other technologies on demand. For example, back-end side, which is now programmed on Python, using Flask web-serve, can be reprogrammed on Java and Apache Tomcat web-server. This change will not affect client side, because HTTP requests that are sent from a client, do not depend on background, platform or technologies of a back-end side. Which makes our solution flexible and expandable fro future.

Caching and saving results of the user gave us idea that the solution can be developed further. Imagine that multiple users will use the ACP Dashboard, and each of them compares different sets of ACP. The authorization procedure will give access to the personal dashboard of each user. In that case we need to store ACPs for each user separately. We can extend our solution to a three-tier architecture, and add a database layer to it. Database will contain information about users and ACPs that belong to the user. Thus, the web-application will support multiple users.

6.2.2 Functional Requirements

The thesis definition also provides us with non-functional requirements. Main part of it was — to create an interactive visualization that will allow navigating and exploring Analytic Configurations performance data. This includes clustering of configurations and defining common components based on similarities of ACPs. Moreover, it was expected to make possible re-clustering on certain clusters in order to go deeper and find subgroups within the cluster. Also technical details on the ACPs should be provided.

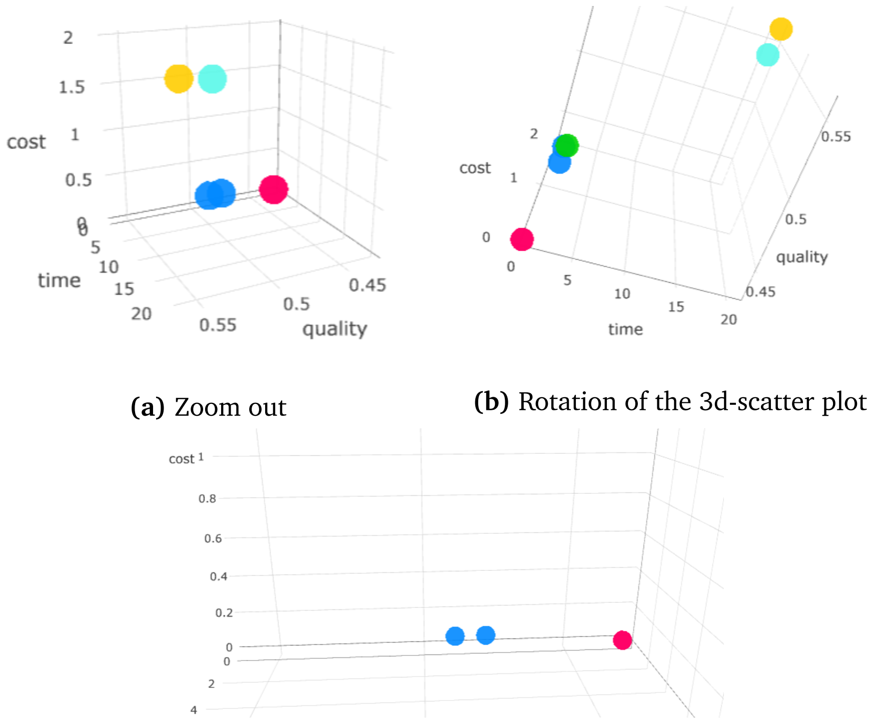
An ACP Dashboard is a tool for comparing ACPs with an interactive visualization (in form of 3d-scatter plot) and a control panel. The control panel shows technical details of the the ACPs and has controls for:

- adding new ACP to a Dashboard,
- changing dimensions on a visualization.

This fulfills two of functional requirements stated above. The visualization part is a plot created with a plotly.js library. Points on a plot represent ACP profiles, which can be compared to each other on three dimensions at a time: *Execution Cost*, *Quality of Algorithm* and *Execution Time*. User can navigate through the data and explore it: rotate, zoom in, zoom out (see **figure 6.1**), hide and show points that are grouped in a cluster. Other libraries that work with python, such as Bokeh[1] and matplotlib[2] do not provide that interactivity. Bokeh does not have 3d-scatter plots, instead it offers surface plots which only bring complexity to the visualization. Matplotlib, on the other hand provides only static plots that cannot be rotated, zoomed in or out on demand. This solution could be developed do bring user more opportunity of manipulating points. For example, we think it is necessary to add functions for deleting points. It could be done either by selecting a point on a cube and deleting it, or deleting the corresponding row from the table with

[1] https://bokeh.pydata.org/en/latest/

[2] https://matplotlib.org

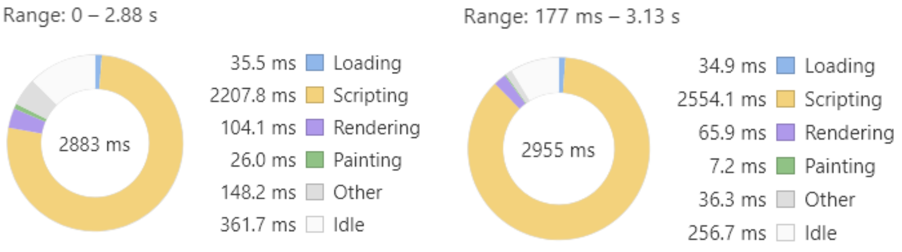


(a) Zoom out

(b) Rotation of the 3d-scatter plot

(c) Zoom in

Figure 6.1: Demonstration of interaction features



(a) Performance with empty cube

(b) Performance with eighteen profiles

Figure 6.2: Performance results, measured on Opera browser in the "Inspect element" mode

details. Unfortunately, as of now, plotly.js does not support selection of points for 3d-scatter plot. Which means we would have to either change the visualization library, or develop this feature on our own.

6.3 Performance Evaluation

We tested performance of the ACP Dashboard in the Opera v.45.0 browser, using "Inspect element" mode (Ctrl+Shift+C) its "Performance" and "Network" tabs. It provides JavaScript CPU Profiler and a Timeline of load. As an input data we took same Amazon food Reviews data set (see section 5.4)

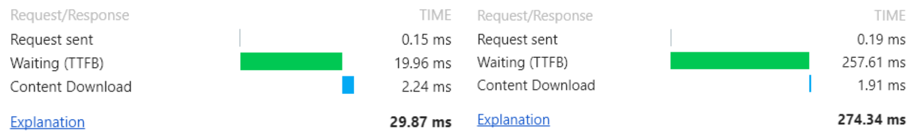
First we started with and empty Dashboard, which is a web-page with a 3d-scatter plot that has no points on it. As can be seen from the performance analysis, the most time took scripting events — 2207.8 ms. The load of the page, rendering and painting events were relatively fast — 35.5 ms, 104.1 ms and 26.0 ms respectively. This makes us think that rendering and painting of the cube itself are fast procedures. In this case rendering means computing Cascading Style Sheets (CSS) which is associated to the Document Object Model (DOM) elements, positioning DOM elements on the page; painting means — literally painting pixels and resizing or decoding images. Scripting, for its part, includes evaluating scripts, time for calling events, function calls and also sending and receiving requests and seems to be the most time consuming procedure.

We went deeper and decided to figure out what takes it almost three seconds for a page to appear. The performance logs from the "Network" tab of the "Inspect element" (see **figure 6.4**) mode and show that all the libraries (plotly.js and bootstrap) are loading fast — no more than 2 ms, but the response from server itself (line 127.0.0.1) takes a lot of time. This can be seen explicitly well at "Waiting (Time to First Byte (TTFB)" row. With zero profiles time is only about 20 ms, with eighteen profiles this value reaches mark 257 ms (see **figure 6.3**). The TTFB depends on a server response time and time to transfer bytes. Because we are working with a web-application which is running on a local machine, time to transfer bytes should not be considered as a reason of such high TTFB. In that case, a server response time is most probably a reason why it takes so long to process the page. According to Google PageSpeedInsigts[3], an appropriate server response should not be more than 200ms. And causes of low response could be following [Lim15]:

- slow database queries,
- slow logic,
- resource starvation,
- too many slow frameworks/libraries/dependencies,
- slow hardware.

We think that slow logic is exactly the reason that causes the delay. As of now the whole HTML page, together with a JavaScript code is generated on a server side and sent to a client. This is also true, because with empty cube we do not have much data to process and send to a server, but with eighteen data points situation is different. That is why waiting takes longer, when amount of data points increases. So our future work would be to move

[3] https://developers.google.com/speed/docs/insights/Server



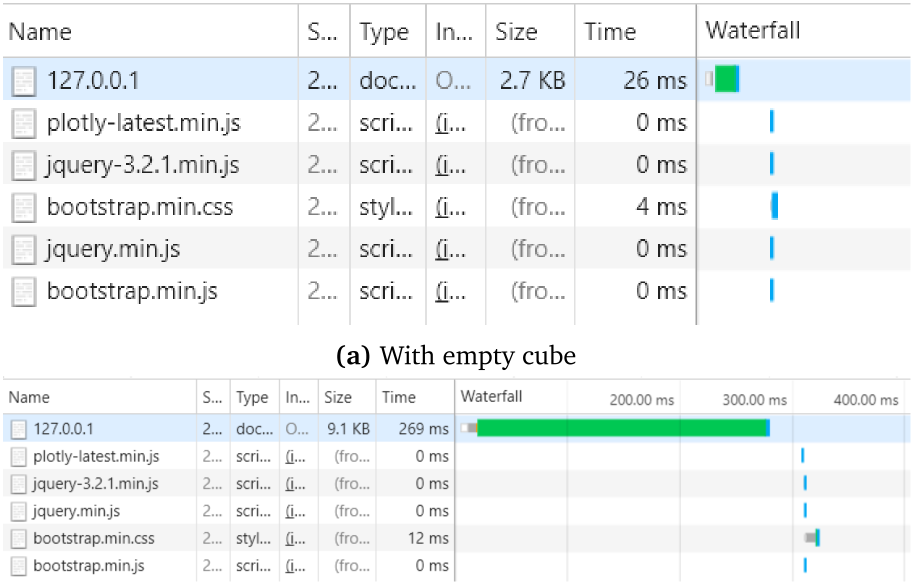
Request/Response	TIME
Request sent	0.15 ms
Waiting (TTFB)	19.96 ms
Content Download	2.24 ms
Explanation	**29.87 ms**

(a) With empty cube

Request/Response	TIME
Request sent	0.19 ms
Waiting (TTFB)	257.61 ms
Content Download	1.91 ms
Explanation	**274.34 ms**

(b) With eighteen profiles

Figure 6.3: Performance results — Time to First Byte



Name	S...	Type	In...	Size	Time	Waterfall
127.0.0.1	2...	doc...	O...	2.7 KB	26 ms	
plotly-latest.min.js	2...	scri...	(i...	(fro...	0 ms	
jquery-3.2.1.min.js	2...	scri...	(i...	(fro...	0 ms	
bootstrap.min.css	2...	styl...	(i...	(fro...	4 ms	
jquery.min.js	2...	scri...	(i...	(fro...	0 ms	
bootstrap.min.js	2...	scri...	(i...	(fro...	0 ms	

(a) With empty cube

Name	S...	Type	In...	Size	Time	Waterfall 200.00 ms 300.00 ms 400.00 ms
127.0.0.1	2...	doc...	O...	9.1 KB	269 ms	
plotly-latest.min.js	2...	scri...	(i...	(fro...	0 ms	
jquery-3.2.1.min.js	2...	scri...	(i...	(fro...	0 ms	
jquery.min.js	2...	scri...	(i...	(fro...	0 ms	
bootstrap.min.css	2...	styl...	(i...	(fro...	12 ms	
bootstrap.min.js	2...	scri...	(i...	(fro...	0 ms	

(b) With eighteen profiles

Figure 6.4: Performance results on a "Network" tab — resource requests

part of the server logic to a client side. For example, HTML page could be generated on client, and only changing parts — data points location on cube, and clustering information, should be sent from server in a .json file. this will reduce server load and amount of data that should be sent to a client and thus resulting in a lower server response time.

Moreover, the evaluation of performance could be more demanding — it is recommended to test the web-application with more ACP. In this work we tested performance of the solution using eighteen profiles, but if in real scenario user probably has to compare forty, or even hundred profiles at a time. In this case one could artificially generate ACPs, load them to Dashboard and measure the performance with the same Opera developer tool. We did not do it due to the lack of time, but we believe that such investigation will contribute much to the solution, helping to find flaws and weak sides.

Our solution can be applicable to different industrial settings, if they follow same scenario — comparison of classification algorithms. Although the solution was primarily designed for the industrial data, we managed to show results with Amazon food service [4] data

[4] https://www.kaggle.com/snap/amazon-fine-food-reviews

also. Which proves, that ACP Dashboard does not depend on the scenario on which it is applied.

7 Conclusion

In conclusion, the contribution of this work is fourfold. In the conceptual solution chapter, we presented the meta-learning approach, where we use ACPs to compare algorithms. The feature of this approach is that the metadata is already given, and we do not require to run algorithms in order to get the performance results. The eighteen ACPs were created on the Amazon food review data set and used as an input data for comparison. There we also introduce business metrics that allow to evaluate and compare text classification algorithms from business perspective (*Quality of Algorithm, Execution Time, Execution Cost*). We proved that decision of business metrics is compliant to the business objectives and thus makes sense to the people from industry.

Since one of the problems of the thesis was data representation, we proposed using 3D scatter plot as a visualization technique. Next, clustering technique — mean shift was used to define patterns in the data, form groups of algorithms that have similar performance. We also developed a method that gives meaningful names to cluster, it shows the parameters that are common for the algorithms which form a cluster. Next we designed a web-application that allows comparison of the algorithms based on ACPs. Our solution has a two-tier (client-server) architecture, and is implemented with python, JavaScript and HTML, together with visualization and clustering libraries. We showed a use-case of the application, tested its usability and fulfillment of the functional requirements, set by a thesis definition. Finally, we evaluated our approach from different points — method for comparing algorithms, chosen business metrics, used visualization and clustering. We also tested performance of the web-application and discussed possible future work that can enhance the proposed solution.

Overall, the approach we presented in the thesis satisfies most of the defined goals. The ACP Dashboard allows to compare machine learning algorithms from the industrial perspective, thus business user may compare algorithms without consulting with ML experts. The chosen data representation (3D scatter plot) is intuitive and does not require extra learning, but more important — it helps to see patterns of the data immediately. The clustering technique shows common configurations of the algorithm, thus giving a hint which algorithm settings influence business metric values. Still, this solution needs enhancements, as discussed in chapter 6. More sophisticated business metrics can be added, dependency of the algorithm performance on data should be also taken into account.

Bibliography

[AG98] Y Asiedu, P Gu. “Product life cycle cost analysis: state of the art review.” In: *int. j. prod. res* 36.4 (1998), pp. 883–908. URL: http://www.lcis.com.tw/paper_store./paper_store/Asiedu+_ProductLifeCycleCostAnalysis_-201412102295156.pdf (cit. on p. 9).

[Aha92] D. Aha. “Generalizing from case studies: A case study.” In: *Proceedings of the Ninth International Conference on Machine Learning* Section 2 (1992), pp. 1–10. DOI: 10.1.1.44.4156. URL: https://pdfs.semanticscholar.org/0b1f/7bc084d00e0277e8ec8c0ebf851a12017498.pdf (cit. on p. 19).

[Atk99] R. Atkinson. “Project management: cost, time and quality, two best guesses and a phenomenon, its time to accept other success criteria.” In: *International Journal of Project Management* 17.6 (1999), pp. 337–342. URL: https://notendur.hi.is/vio1/Project_management_Cost_time_and_quality.pdf (cit. on p. 30).

[BI16] A. H. Beg, M. Z. Islam. “Advantages and limitations of genetic algorithms for clustering records.” In: *Proceedings of the 2016 IEEE 11th Conference on Industrial Electronics and Applications, ICIEA 2016* June (2016), pp. 2478–2483. DOI: 10.1109/ICIEA.2016.7604009. URL: http://ieeexplore.ieee.org/abstract/document/7604009/ (cit. on p. 17).

[BID14] E. Bradner, F. Iorio, M. Davis. “SimAUD 2014 Symposium on Simulation for Architecture and Urban Design Parameters Tell the Design Story: Ideation and Abstraction in Design Optimization.” In: (2014). URL: https://d2f99xq7vri1nk.cloudfront.net/legacy_app_files/pdf/64_final.pdf (cit. on p. 18).

[BS07] R. Bhagwat, M. K. Sharma. “Performance measurement of supply chain management : A balanced scorecard approach.” In: 53 (2007), pp. 43–62. DOI: 10.1016/j.cie.2007.04.001 (cit. on pp. 22, 32).

[Bow+12] P. Bowen, K. Cattel, K. Hall, P. Edwards, R. Pearl. “Perceptions of Time, Cost and Quality Management on Building Projects.” In: *Construction Economics and Building* 2.2 (2012), pp. 48–56. ISSN: 2204-9029. DOI: 10.5130/CEB.v2i2.2900. URL: http://epress.lib.uts.edu.au/journals/index.php/AJCEB/article/view/2900 (cit. on p. 30).

[Bra03] P. B. Brazdil. "Ranking Learning Algorithms: Using IBL and Meta-Learning on Accuracy and Time Results." In: *Machine Learning* 50 (2003), pp. 251–277. URL: https://www.researchgate.net/profile/Joaquim_Costa2/publication/220344215_Ranking_Learning_Algorithms_Using_IBL_and_Meta-Learning_on_Accuracy_and_Time_Results/links/0912f5072b5c7e6672000000/Ranking-Learning-Algorithms-Using-IBL-and-Meta-Learning-on-Accuracy-and-Time-Results.pdf (cit. on p. 18).

[CM02] D. Comaniciu, P. Meer. "Mean shift: a robust approach toward feature space analysis." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 24.5 (2002), pp. 603–619. ISSN: 0162-8828. DOI: 10.1109/34.1000236. URL: http://cyrille.nathalie.free.fr/computer%20vision/color%20segmentation/comanici/Papers/MsRobustApproach.pdf (cit. on p. 26).

[CT08] N. B. Ciza Thomas. "Improvement in minority attack detection with skewness in network traffic." In: *Proc.SPIE* 6973 (2008), pp. 6973 –6973 –12. DOI: 10.1117/12.785623. URL: https://www.researchgate.net/profile/Ciza_Thomas/publication/253383113_Improvement_in_minority_attack_detection_with_skewness_in_network_traffic/links/540826060cf2bba34c249dd1/Improvement-in-minority-attack-detection-with-skewness-in-network-traffic.pdf (cit. on p. 26).

[Che95] Y. Cheng. "Mean shift, mode seeking, and clustering." In: *IEEE Transactions on Pattern Analysis and Machine Intelligence* 17.8 (1995), pp. 790–799. ISSN: 0162-8828. DOI: 10.1109/34.400568. URL: http://home.ku.edu.tr/mehyilmaz/public_html/mean-shift/00400568.pdf (cit. on p. 26).

[Cho+17] D. S. Cho, F. Khalvati, D. A. Clausi, A. Wong. "A Machine Learning-Driven Approach to Computational Physiological Modeling of Skin Cancer." In: *14th International Conference, ICIAR* (2017), pp. 79–86. DOI: 10.1007/978-3-319-59876-5. URL: https://www.springerprofessional.de/en/a-machine-learning-driven-approach-to-computational-physiologica/12451520 (cit. on p. 30).

[DT13] N. Dogan, Z. Tanrikulu. "A comparative analysis of classification algorithms in data mining for accuracy, speed and robustness." In: *Inf Technol Manag* 14 (2013), pp. 105–124. DOI: 10.1007/s10799-012-0135-8. URL: https://link.springer.com/article/10.1007/s10799-012-0135-8 (cit. on p. 15).

[Dem06] J. Demšar. "Statistical Comparisons of Classifiers over Multiple Data Sets." In: *Journal of Machine Learning Research* 7 (2006), pp. 1–30. URL: http://www.jmlr.org/papers/volume7/demsar06a/demsar06a.pdf (cit. on p. 17).

[Die98] T. G. Dietterich. "Approximate Statistical Tests for Comparing Supervised Classification Learning Algorithms." In: *Neural Comput.* 10.7 (1998), pp. 1895–1923. ISSN: 0899-7667. DOI: 10.1162/089976698300017197. URL: http://sci2s.ugr.es/keel/pdf/algorithm/articulo/dietterich1998.pdf (cit. on pp. 16, 36).

[Dol02] S. Dolnicar. “A Review of Unquestioned Standards in Using Cluster Analysis for Data-Driven Market Segmentation A Review of Unquestioned Standards in Using Cluster Analysis for Data- Driven Market Segmentation.” In: 2002, pp. 2–4. URL: http://ro.uow.edu.au/cgi/viewcontent.cgi?article=1286&context=commpapers (cit. on p. 38).

[Dom15] R. T. Domingo. *THE QCD APPROACH TO OPERATIONS MANAGEMENT*. 2015. URL: http://www.rtdonline.com/BMA/MM/qcd.htm (visited on 12/04/2017) (cit. on p. 32).

[Ete+16] R. Etemadpour, L. Linsen, J. G. Paiva, C. Crick, A. G. Forbes. “Choosing Visualization Techniques for Multidimensional Data Projection Tasks: A Guideline with Examples.” In: *Computer Vision, Imaging and Computer Graphics Theory and Applications: 10th International Joint Conference, VISIGRAPP 2015, Berlin, Germany, March 11–14, 2015, Revised Selected Papers*. Ed. by J. Braz, J. Pettré, P. Richard, A. Kerren, L. Linsen, S. Battiato, F. Imai. Cham: Springer International Publishing, 2016, pp. 166–186. ISBN: 978-3-319-29971-6. DOI: 10.1007/978-3-319-29971-6_9. URL: https://doi.org/10.1007/978-3-319-29971-6_9 (cit. on p. 27).

[Fie00] R. T. Fielding. “Representational State Transfer (REST).” PhD thesis. Irvine: University of California, 2000. Chap. 5, pp. 76–105. URL: https://www.ics.uci.edu/{~}fielding/pubs/dissertation/fielding{_}dissertation{_}2up.pdf (cit. on p. 59).

[For03] G. Forman. “An Extensive Empirical Study of Feature Selection Metrics for Text Classification.” In: *Journal of Machine Learning Research* 3 (2003), pp. 1289–1305. URL: http://www.jmlr.org/papers/volume3/forman03a/forman03a.pdf (cit. on pp. 11, 15, 25).

[For84] A. K. Formann. *Die Latent-Class-Analyse : Einführung in Theorie und Anwendung*. Beltz-Monographien. Weinheim: Beltz, 1984. ISBN: 3-407-54657-2 (cit. on p. 38).

[GHE08] S. García, F. Herrera, H. U. Es. “An Extension on " Statistical Comparisons of Classifiers over Multiple Data Sets " for all Pairwise Comparisons.” In: *Journal of Machine Learning Research* 9 (2008), pp. 2677–2694. URL: http://www.jmlr.org/papers/volume9/garcia08a/garcia08a.pdf (cit. on p. 17).

[GS00] P. D. Gardiner, K. Stewart. “Revisiting the golden triangle of cost, time and quality: the role of NPV in project control, success and failure.” In: *International Journal of Project Management* 18 (2000), pp. 251–256. URL: https://notendur.hi.is/vio1/Revisiting_the_%20golden_triangle_of_cost_time_and%20%20%20%20%20%20quality_the%20role%20of%20NPV%20in%20project%20control_success%20and%20failure.pdf (cit. on p. 30).

[GS14] S. Gupta, M. Starr. *Production and Operations Management Systems*. Boca Raton, FL: CRC Press, 2014, pp. 279–280. ISBN: 978-1-4665-0734-0. URL: http://lib.vcomsats.edu.pk/library/LSM733/Course%20Contents/HANDOUTS/PDF % 20Books / Gupta , %20Sushil _ %20Starr , %20Martin % 20Kenneth -

Production%20and%20Operations%20Management%20Systems-CRC%20Press%20(2014).pdf (cit. on p. 21).

[GSW12] G. Garrison, K. Sanghyun, R. L. Wakefield. "Success Factors for Deploying Cloud Computing." In: *Communications of the ACm* 55.9 (2012), pp. 62–68. DOI: 10.1145/2330667.2330685. URL: http://www.yildiz.edu.tr/~aktas/courses/CE-0112822/13-05-4-1.pdf (cit. on p. 32).

[Gar87] D. A. Garvin. "Competing on the Eight Dimensions of Quality." In: *Harvard Business Review* (1987). URL: https://hbr.org/1987/11/competing-on-the-eight-dimensions-of-quality (cit. on p. 22).

[Hev+04] A. R. Hevner, S. T. March, J. Park, S. Ram. "DESIGN SCIENCE IN INFORMATION SYSTEMS RESEARCH." In: *Design Science in IS Research MIS Quarterly* 28.1 (2004), pp. 75–105. URL: https://pdfs.semanticscholar.org/fa72/91f2073cb6fdbdd7c2213bf6d776d0ab411c.pdf (cit. on p. 55).

[JS92] B. Johnson, B. Shneiderman. "Tree-Maps: A Space-filling Approach to the Visualization of Hierarchical Information Structures." In: *Proceedings of the 2Nd Conference on Visualization '91*. VIS '91. IEEE Computer Society Press, 1992, pp. 284–291. ISBN: 0-8186-2245-8. URL: http://delivery.acm.org/10.1145/150000/142833/p369-johnson.pdf?ip=141.72.229.55&id=142833&acc=ACTIVE%20SERVICE&key=2BA2C432AB83DA15%2EA83A5A66E0DD4B84%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=1015417557&CFTOKEN=45585583&__acm__=1512982937_ee40b58777dce6ec0e68a19f4f6acc01 (cit. on p. 28).

[Jai+99] A. K. Jain, M. N. Murty, P. J. Flynn, A. Rosenfeld, K Bowyer, N Ahuja, A Jain. "Data Clustering: A Review." In: *ACM Computing Surveys* 31.3 (1999). URL: http://delivery.acm.org/10.1145/340000/331504/p264-jain.pdf?ip=141.72.229.55&id=331504&acc=PUBLIC&key=2BA2C432AB83DA15%2EA83A5A66E0DD4B84%2E4D4702B0C3E38B35%2E4D4702B0C3E38B35&CFID=1015417557&CFTOKEN=45585583&__acm__=1512982826_76adea99e47189c6ac5e631cab370bdf (cit. on p. 26).

[Joh92] B. Johnson. "TreeViz: Treemap Visualization of Hierarchically Structured Information." In: *CHI* (1992), pp. 369 –370. URL: http://delivery.acm.org/10.1145/150000/142833/p369-johnson.pdf?ip=141.72.229.55{\&}id=142833{\&}acc=ACTIVESERVICE{\&}key=2BA2C432AB83DA15.A83A5A66E0DD4B84.4D4702B0C3E38B35.4D4702B0C3E38B35{\&}CFID=1010209112{\&}CFTOKEN=20193513{\&}{_}{_}acm{_}{_}=1511708737{_}70ff3551bf21140255da1ab86441da1a (cit. on p. 28).

[KGH04] A. Kalousis, J. Gama, M. Hilario. "On data and algorithms: Understanding inductive performance." In: *Machine Learning* 54.3 (2004), pp. 275–312. ISSN: 08856125. DOI: 10.1023/B:MACH.0000015882.38031.85. URL: https://link.springer.com/article/10.1023%2FB%3AMACH.0000015882.38031.85?LI=true (cit. on p. 18).

[KN93] R. S. Kaplan, D. P. Norton. “Putting the Balanced Scorecard to Work.” In: *Harvard Business Review: On Point* (1993), pp. 2–18. URL: https://hbr.org/1993/09/putting-the-balanced-scorecard-to-work (cit. on p. 23).

[Kei+08] D. Keim, G. Andrienko, J.-D. Fekete, C. Görg, J. Kohlhammer, G. Melançon. “Visual Analytics: Definition, Process, and Challenges.” In: *Information Visualization: Human-Centered Issues and Perspectives*. Ed. by A. Kerren, J. T. Stasko, J.-D. Fekete, C. North. Berlin, Heidelberg: Springer Berlin Heidelberg, 2008, pp. 154–175. ISBN: 978-3-540-70956-5. DOI: 10.1007/978-3-540-70956-5_7. URL: https://doi.org/10.1007/978-3-540-70956-5_7 (cit. on p. 34).

[Kot07] S. B. Kotsiantis. “Supervised Machine Learning: A Review of Classification Techniques.” In: *Informatica* 31 (2007), pp. 249–268. ISSN: 09226389. DOI: 10.1115/1.1559160. arXiv: /www.informatica.si/index.php/informatica/article/view/148 [http:]. URL: http://citeseerx.ist.psu.edu/viewdoc/download?doi=10.1.1.95.9683&rep=rep1&type=pdf (cit. on p. 16).

[LDA17] D. J. Lohan, E. M. Dede, J. T. Allison. “Topology optimization for heat conduction using generative design algorithms.” In: *Structural and Multidisciplinary Optimization* 55.3 (2017), pp. 1063–1077. DOI: 10.1007/s00158-016-1563-6. URL: https://www.semanticscholar.org/paper/Topology-Optimization-for-Heat-Conduction-Using-Ge-Lohan-Dede/60eb8705b6bc24a594e5bacfafd89440e77afc6a (cit. on p. 17).

[LGL05] H. Liu, L. Gao, X. Liu. “Generative Design in an Agent Based Collaborative Design System.” In: *LNCS* 3168 (2005), pp. 105–116. URL: https://link.springer.com/chapter/10.1007%2F11568421_11 (cit. on p. 18).

[LL13] M.-H. Lin, L.-C. Lee. “An Experimental Study for Applying Generative Design to Electronic Consumer Products.” In: *Part IV LNCS* 8015 (2013), pp. 392–401. URL: https://link.springer.com/chapter/10.1007/978-3-642-39253-5_43 (cit. on p. 17).

[Leh+12] D. J. Lehmann, G. Albuquerque, M. Eisemann, M. Magnor, H. Theisel. “Selecting Coherent and Relevant Plots in Large Scatterplot Matrices.” In: xx (2012), pp. 1–12. URL: https://pdfs.semanticscholar.org/82b5/44b32cb92c8767f7d3c126f9c3a3f584d54b.pdf (cit. on p. 28).

[Lim15] C. Limpalair. *What is Waiting (TTFB) in DevTools, and what to do about it*. 2015. URL: https://scaleyourcode.com/blog/article/27 (cit. on p. 62).

[MDT99] M. Martinsons, R. Davison, D. Tse. “The balanced scorecard: a foundation for the strategic management of information systems.” In: *Decision Support Systems* 25.1 (1999), pp. 71–88. ISSN: 01679236. DOI: 10.1016/S0167-9236(98)00086-4. URL: http://ai2-s2-pdfs.s3.amazonaws.com/eec4/4862b2d58434ca7706224bc0e9437a2bc791.pdf (cit. on p. 22).

[ML13] J. McAuley, J. Leskovec. “From amateurs to connoisseurs: modeling the evolution of user expertise through online reviews.” In: *WWW* (2013) (cit. on p. 50).

[MNSS09] H. R. Motahari-Nezhad, B. Stephenson, S. Singhal. “Outsourcing Business to Cloud Computing Services: Opportunities and Challenges.” In: *IEEE Internet Computing* (2009). URL: http://www.hpl.hp.com/techreports/2009/HPL-2009-23.pdf (cit. on p. 32).

[Mar+11] S. Marston, Z. Li, S. Bandyopadhyay, J. Zhang, A. Ghalsasi. “Cloud computing — The business perspective.” In: *Elsevier* 51 (2011), pp. 176–189. DOI: 10.1016/j.dss.2010.12.006. URL: http://www.cs.joensuu.fi/{~}parkkine/LuK2015/CloudComputing-DecisionSupportSystems2011.pdf (cit. on p. 32).

[Mar07] D. Marghescu. “Multidimensional Data Visualization Techniques For Exploring Financial Performance Data.” In: *AMCIS* 509 (2007). URL: https://www.researchgate.net/profile/Dorina_Rajanen_marghescu/publication/220891806_Multi-dimensional_Data_Visualization_Techniques_for_Exploring_Financial_Performance_Data/links/53ee4ba70cf23733e80bcf18/Multi-dimensional-Data-Visualization-Techniques-for-Exploring-Financial-Performance-Data.pdf (cit. on p. 27).

[McC89] D. K. McClish. “Analyzing a Portion of the ROC Curve.” In: *Medical Decision Making* 9.3 (1989). PMID: 2668680, pp. 190–195. DOI: 10.1177/0272989X8900900307. eprint: https://doi.org/10.1177/0272989X8900900307. URL: https://doi.org/10.1177/0272989X8900900307 (cit. on p. 25).

[Ols71] R. P. Olsen. “Can project management be defined?” In: *Project management quarterly* 2.1 (1971), pp. 12–14. DOI: 0147-5363. URL: https://www.pmi.org/learning/library/project-management-defined-concept-1950 (cit. on p. 30).

[PWR04] W. Peng, M. O. Ward, E. A. Rundensteiner. “Clutter Reduction in Multi-Dimensional Data Visualization Using Dimension Reordering.” In: (2004). URL: http://digitalcommons.wpi.edu/computerscience-pubs/71/ (cit. on p. 28).

[Pow11] D. M. W. Powers. “Evaluation: From Precision, Recall And F-measure, To Roc, Informedness, Markedness Correlation.” In: *Journal of Machine Learning Technologies ISSN* 2.1 (2011), pp. 2229–3981. URL: https://bioinfopublication.org/files/articles/2_1_1_JMLT.pdf (cit. on p. 25).

[RR07] L. Richardson, S. Ruby. *Restful Web Services*. First. O’Reilly, 2007. ISBN: 9780596529260. URL: http://shop.oreilly.com/product/9780596529260.do (cit. on p. 59).

[Ros+03] L Rosasco, E De Vito, A Caponnetto, M Piana, A Verri. “Are Loss Functions All the Same?” In: (2003). URL: http://web.mit.edu/lrosasco/www/publications/loss.pdf (cit. on p. 25).

[SSS12] J. K. Shim, J. G. Siegel, A. I. Shim. “Manufacturing Costs: Sales Forecasts and Realistic Budgets.” In: *Budgeting Basics and Beyond*. 4th ed. John Wiley & Sons, 2012. Chap. 9, pp. 191–202. DOI: 10.1002/9781118387023.ch9. URL: http://onlinelibrary.wiley.com/doi/10.1002/9781118387023.ch9/summary (cit. on p. 22).

[SW10] C. Sammut, G. I. Webb. *Encyclopedia of Machine Learning*. Boston: Springer, Boston, MA, 2010. ISBN: 978-0-387-30164-8. URL: https://link.springer.com/referencework/10.1007%2F978-0-387-30164-8 (cit. on p. 24).

[Saa08] T. L. Saaty. “Decision making with the analytic hierarchy process.” In: *International Journal of Services Sciences* (2008). ISSN: 1753-1446. DOI: 10.1504/IJSSCI.2008.017590. URL: https://www.colorado.edu/geography/leyk/geog_5113/readings/saaty_2008.pdf (cit. on p. 16).

[Sci] *Comparing different clustering algorithms on toy datasets — scikit-learn 0.19.1 documentation*. URL: http://scikit-learn.org/stable/auto_examples/cluster/plot_cluster_comparison.html (visited on 12/09/2017) (cit. on p. 58).

[Seb01] F. Sebastiani. “Machine Learning in Automated Text Categorization.” In: (2001). arXiv: 0110053v1 [arXiv:cs]. URL: https://arxiv.org/pdf/cs/0110053.pdf (cit. on p. 9).

[She05] Y. Shen. “Loss Functions For Binary Classification And Class Probability Estimation.” PhD thesis. University of Pensilvania, 2005, pp. 1–119. URL: http://stat.wharton.upenn.edu/~buja/PAPERS/yi-shen-dissertation.pdf (cit. on p. 25).

[Sta88] G. Stalk. “Time -The Next Source of Competitive Advantage.” In: *Harward Business Review* (1988). URL: https://hbr.org/1988/07/time-the-next-source-of-competitive-advantage (cit. on p. 30).

[Sze+16] V. Sze, Y.-H. Chen, J. Emer, A. Suleiman, Z. Zhang. “Hardware for Machine Learning: Challenges and Opportunities.” In: (2016). URL: https://arxiv.org/pdf/1612.07625.pdf (cit. on p. 16).

[TDL11] D. Tomaskovic-Devey, K.-H. Lin. “Income Dynamics, Economic Rents, and the Financialization of the U.S. Economy.” In: *American Sociological Review* 76.4 (2011), pp. 538–559. DOI: 10.1177/0003122411414827. eprint: https://doi.org/10.1177/0003122411414827. URL: https://doi.org/10.1177/0003122411414827 (cit. on p. 21).

[Tat+11] A. Tatu, G. Albuquerque, M. Eisemann, P. Bak, H. Theisel, M. Magnor, D. Keim. “Automated analytical methods to support visual exploration of high-dimensional data.” In: *IEEE Transactions on Visualization and Computer Graphics* (2011). DOI: 10.1109/TVCG.2010.242. URL: http://wwwisg.cs.uni-magdeburg.de/visual/files/publications/2010/Tatu_2010_TVCG.pdf (cit. on pp. 28, 33).

[Vil17] A. Villanueva. “A framework to guide the selection and configuration of advanced data analytics solutions in manufacturing.” In: *Internal Manuscript* (2017) (cit. on pp. 10, 12, 23, 29, 34, 56).

[Wil05] S. Williams. *Unnatural Selection - MIT Technology Review*. 2005. URL: https://www.technologyreview.com/s/403655/unnatural-selection/ (visited on 07/18/2017) (cit. on p. 18).

[Woe10] J. Woe. *QCD – Quality, Cost and Delivery*. 2010. URL: https://johanneswoe.wordpress.com/2010/05/27/qcd-quality-cost-and-delivery/ (cit. on p. 21).

[ZKM17] A. G. Zacarias Villanueva, L. Kassner, B. Mitschang. "Exploring Text Classification Configurations A bottom-up approach to customize text classifiers based on the visualization of performance." In: (2017) (cit. on p. 19).

[Jas+12] D. Jason Gerber, S. - Hsin Lin, B. Pan, A. Senel Solmaz. "Design Optioneering: Multi - disciplinary Design Optimization through Parameterization, Domain Integration and Automation of a Genetic Algorithm." In: (2012). URL: http://journals.sagepub.com/doi/abs/10.1177/0037549713482027 (cit. on p. 17).

[mat] matplotlib. *The 3D scatter plot made with matplotlib library*. URL: https://matplotlib.org/examples/mplot3d/scatter3d_demo.html (visited on 06/27/2017) (cit. on p. 27).

All links were last followed on December 11, 2017.

Declaration

I hereby declare that the work presented in this thesis is entirely my own and that I did not use any other sources and references than the listed ones. I have marked all direct or indirect statements from other sources contained therein as quotations. Neither this work nor significant parts of it were part of another examination procedure. I have not published this work in whole or in part before. The electronic copy is consistent with all submitted copies.

place, date, signature