Assessing the Performance of Machine Learning for a Job Matching Algorithm

A Study of Optimal Skill Allocation on an Online Gig Work Platform

Kwan Tsit Richard Chan

Rautistrasse 67
8047 Zurich
+41 78 910 18 68
Student ID: 13739362
kwantsitrichard.chan@uzh.ch
Field of Study: Economics and Data Science

A thesis presented for the degree of
Master of Arts in Economics

University of Zurich
Department of Business Administration
Chair of Marketing
Prof. Dr. Martin Natter
November 8th 2019

Assessing the Performance of Machine Learning for a Job Matching Algorithm

Kwan Tsit Richard Chan

Abstract

Machine learning has a wide range of applications and methods suitable to solve different kinds of problems. The solution is either in form of inference to make better decisions, or in form of predictions of the future. It depends on the particular problem how a machine learning model performs. This thesis evaluates the performance of seven different machine learning models to solve a problem of a matching algorithm for an online gig work platform. The problem consists of the reduction of job requests sent to workers. Different machine learning methods and techniques are created and applied to the platform data, in order to predict a worker's probability to accept a request and a worker's expected rating for a given job. Furthermore, a matching score between a worker's skill set and the required skill is calculated and tested on seven different skill granularities to distinguish the optimal granularity allocation for the matching algorithm. The final model is a tree-based extreme gradient boosting model with a considerable performance. It was able to improve the initial algorithm by reducing the amount of requests by up to 77%, depending on the urgency to hire a worker on a particular job. The results suggests that machine learning has a great potential to improve the matching algorithm, but also emphasize the importance of the correct usage of such a machine learning solution.

Table of Contents

List of Figures

List of Tables

List of Abbreviations

Bal. Acc.	=	Balanced Accuracy
CV	=	Cross-validation
FFNNET	=	Feed Forward Neural Network
FN	=	False Negative
FP	=	False Positive
genImp	=	General Improvements
KNN	=	K-Nearest Neighbor
LM	=	Linear Regression Model
LOG	=	Logistic Regression Model
MAE	=	Mean Absolute Error
ML	=	Machine Learning
NN	=	Neural Networks
PCC	=	Pearsons Correlation Coefficient
pjImp	=	Per Job Improvements
PRAUC	=	Area Under the Precision-Recall curve
PR	=	Precision-Recall
RF	=	Random Forest
RMSE	=	Root Mean Squared Error
ROCAUC	=	Area Under the Receiver Operating Characteristic Curve
ROC	=	Receiver Operating Characteristic
SMOTE	=	Synthetic Minority Oversampling Technique
SVM	=	Support Vector Machine
TH	=	Threshold
TN	=	True Negative
TP	=	True Positive
XGB	=	Extreme Gradient Boosting

1. Introduction

Making use of data has gained tremendous importance over the last years. This is mainly due to the ease of generating data, which leads to an increasing amount of available data, in combination with increasing computational power to analyze data. As a result, more techniques and methods are constantly being developed to cover the needs in this dynamic field. Machine learning (ML) is a method to automatically build analytical models from data, in order to gain insights to crucial information and being able to make decisions, or to make predictions of the future (James, Hastie, Witten, & Tibshirani 2013).

This thesis evaluates the performance of modern ML techniques on a matching algorithm of a digital online gig work platform and measures the improvements to the algorithm brought by ML. Gig work is a short-term based form of work, which requires the worker to be locally present, in order to perform a certain task (Konrad & Pekruhl, 2017). The platform provides the medium with which employers and workers can search and find each other through job requests. The algorithm has been used on the platform to match available workers to jobs with certain requirements, such as skill or geographic and temporal availabilities. The platform only sends out requests to matching workers. Requested workers can then optionally react to the request by accepting or declining the request. Then, the employer can hire workers who accepted and rate them afterwards with one to four stars. However, the problem of the matching algorithm is that it sends out a high number of requests to fill jobs with often low success rate, resulting in workers receiving too many requests. With the use of ML models, patterns from the historical data can be learned. This thesis focuses on predicting matching worker-job pairs for future jobs. In particular, ML is used to reduce the amount of requests sent out by predicting a workers probability to accept a request and the expected rating of the worker for a given job. The predictions of a worker's answer and the worker rating is then compared to their true data point, in order to measure the improvements using ML upon the initial algorithm. Furthermore, skills are grouped using specific skill granularities. A scoring technique is then applied to each granularity to evaluate the matching of a worker's skill set to the required skill of a job. These scores are added to the model for each granularity separately, in order to evaluate the optimal skill granularity allocation for the matching.

The result of this thesis is the quantification of potential improvements for the given algorithm. The improvements come in form of more targeted requests and higher expected ratings of hired workers. This thesis gives an overview of the performance of different ML techniques and models, where the best performing model is as a scalable state-of-the-art ML solution. It can also be implemented to improve the matching algorithm of the platform. Lastly, this thesis answers the question of the optimal skill granularity for a matching algorithm. Chapter 2 of this thesis describes the current state of research in the field and relates this thesis to preceding research. Chapter 3 gives an overview of the data used in this thesis. Chapter 4 explains how the predictive ML models are built and evaluated and describes the approach to find the best performing model. The results and improvements are then quantified in Chapter 5, where the optimal skill granularity allocation is discussed. Finally, the last chapter concludes and proposes trajectories for future research.

2. Related Work

The current state of research in job matching problems focuses on the traditional job matching using a worker's resume and applications to job positions along with the job requirements. Lin, Lei, Addo, and Li (2016), for example, showed that their ML solution has great potential to improve job matching in general. They proposed three ML models to automatically detect similarities in job position and significantly improved search results for job seekers. This normally happens on a rather high level of requirements in education and experience of long-term jobs with a chance of a permanent position.

Another branch in related research is skill matching. There are many approaches that use skills to match workers with jobs. For example, Sayfullina, Kanala, and Malmi (2018) proposed a phrase-matching based approach for different skills in resumes. Their solution has shown a better performance regarding the search of similar positions in resumes, compared to a rule-based solution. This field of research predominantly focuses on jobs of more permanent nature. Contrarily, the thesis of Bottega (2018) with a similar approach analyzed the same gig work platform. He evaluated the drivers of the average rating of a worker. To achieve this, Bottega also data mines information from workers' resumes. Focusing on inference, he found that the proficiency levels are dependent on the industry.

The thesis at hand differs from job or skill matching research projects in this field in the following four points. First, a high amount of applications per job position is generated on the gig work platform at the basis of this thesis, which is not the case in traditional job matching problems. Also, the job positions considered in traditional job matching differ from those on the platform. Second, the ratings given to a worker do not only depend on a worker, but also depend on the requirements of the job. The same worker can receive both high or low ratings, depending on the skills of a worker and the requirements of the job, among others. Contrarily, this thesis analyzes the worker ratings on a job level, rather than for an individual worker on average as in the work of Bottega (2018). The third point is that the platform and the amount of available data has dramatically changed since the creation of the prior thesis. More available features from the workers, the jobs, and more observations are used in this thesis. As a last point, this thesis focuses on creating a prediction model that can be applied to improve the matching algorithm, whereas the goal of Bottega (2018) was inferencial analysis.

Besides its contribution to the job and skill matching literature, this thesis also relates to research on skill granularities. A paper from Pardos, Heffernan, Anderson, and Linquist-Heffernan (2007) compares different skill granularities with ML to determine an optimal granularity. The skills are evaluated on a specific math exam with 29 questions to evaluate math skills that can be measured for different kinds of math problems, such as algebra or geometry problems. The authors built a Bayesian neural network using exam data to evaluate the relation of a student's math skills to the exam outcome on different skill granularities. Their model learns in which way each skill affects the result of a certain math question, such that they can predict the outcome of the exam for each student. The authors showed that in general, the lower and finer skill granularities are, the better the matching results will be.

Unfortunately, the form of the data used in this thesis does not allow to build a Bayesian network that learns from the skills, as the required skills for a job differs between jobs whereas the math exam questions are the same for all students. In that regard, this thesis studies the skill set of a worker matching a given required skill. This thesis differs from the framework of Pardos et al. (2007) by using different ML methods than Bayesian networks and by including additional features aside from skills. Nonetheless, this thesis still seeks to find the optimal skill granularity, also by comparing model performances when using different levels of granularity.

3. The Data

The data used in this thesis was generated on the gig work platform. The data is either formed by the user supplying information to the platform, or by the usage of the platform. In this chapter, a brief description of the used entities and their variables is provided in Section 3.1. Section 3.2 reveals how the data of the different entities enter the ML models. Section 3.3 shows a summary of the used data.

3.1 Description of the Data

The data used for the models can be separated into five main entities, namely the requests, the ratings, the companies, the workers and the skills. In this section, these entities are presented with their features, serving as their predictors. The features have been selected with regard to their exogeneity, which requires the features to be independent from any outcome variables or other features. The company data is generated by the employers of the companies who create and manage their profile and the jobs, hire workers for those jobs, and rate hired workers from 1 to 4 after the job. Likewise, the workers create and manage their personal profile where they provide personal information to the platform and answer requests for particular jobs. Next to mandatory legal documents, the workers also supply information about their skills by choosing from a set of job profiles with corresponding educational levels. This set of job profiles represent the skill set of a worker. Similarly, the employers select from a set of job profiles to define the required skill in a job, before other job information are defined.

3.1.1 The request and the rating data

As mentioned before, the two dependent variables are whether the worker accepted or declined a certain request, hereinafter called *worker answer*, and the rating a worker receives, hereinafter called *worker rating.* If a worker accepts a request and gets hired and rated, this particular observation would then contain both the worker's answer and the worker rating. Hence the ratings are a subset of the requests. Moreover, the predictions are used before requests are actually sent to workers. Therefore, the datasets exclude features using data that is only available after sending, to remove future information. This excludes features such as the amount of time the requested worker needed to answer the request upon receiving. Table 1 provides an overview over all potential features in both datasets of requests and ratings.

Feature	Description
job_id	Unique job identifier
company_id	Unique company identifier
worker_id	Unique worker identifier
answer	Worker answer of the request
rating	Rating a worker received after the job
is_favorite	Whether the worker is a favorite of the company
is_former	Whether the employer hired the worker before
job_duration	Hours the worker is expected to work
salary	The total monetary compensation for the job
wage	The hourly wage a worker receives
distance_to_job	Geographical distance in minutes from worker to job
language_skill_required	Whether certain languages are required
driving_skill_required	Whether driving skills are required
uniform	Whether a work uniform is provided
days_since_ll	Days between last login and considered sending time
job_name_length	Length of the title of the job
job_desc_length	Length of the job description
clothing_requ_length	Length of the clothing requirements
meeting_point_length	Length of the meeting point description

Table 1: Features of the Request and the Rating Data

The requests, where no answer from the worker is observed or no rating of the worker is provided are excluded for the purpose of training the model, as the outcome variables are missing. However, to measure the improvements of the matching algorithm, the observations without an answer or a rating are not excluded for predictions. This is because the improvement is measured in general terms considering all requests that were sent. Nonetheless, the model's performance in terms of accuracy cannot be measured without the actual outcome, hence is measured on the subset without observations with missing outcomes.

3.1.2 The company and the worker data

The company data contains features from the companies. The company characteristics are expected to have a relation to at least one of the two dependent variables and to be stable over time. Some features, like the size of a company, can occasionally change. These features are assumed to be stable over the chosen period. More time-dependent features such as the total number of hired workers violate this assumption and are therefore excluded from the data. As long as such variables cannot be adjusted for past observations, it creates a bias towards the current status of such variables. Table 2 shows the company features that are expected to be related to the outcome variables and to be stable over the chosen time period.

Feature	Description
company__id	Unique company identifier
company__dtfj	Days to first job since company registration
company__size	Size of a company
was__referred	Whether company was invited to the platform
company__favorites	Amount of favorite workers
count__employer	Count of hiring employers of a company
root__count__emp	Count of employers within a company structure
company__industry	Industry of company
avg__worker__rating	Average rating a company gives to workers

Table 2: Features of the Company Data

The worker data holds the set of features from the workers. Besides basic demographic information about gender and age, the data also contains characteristics of the worker measured on the platform. Like the company data, these features of the worker are assumed to be stable over time and are related to at least one of the outcome variables. If a feature is unstable, e.g. the average reaction time to a job offer or the age of a worker, the most recent information on the status is taken because the form of the database does not yet allow for temporal adjustments of such features for each observation. Thus, it is assumed that these changes are relatively small such that the last known status is a good enough approximation. Other, more time-dependent features known to violate this assumption are again excluded from the dataset. Table 3 provides a list of the worker features.

Feature	Description
worker_id	Unique worker identifier
age	Age of the worker in years
gender	Gender of a worker
description_length	Length of characters of the self-description
is_domestic	Whether worker is a domestic worker
avg_rating	Average worker rating, on a scale from 1 to 4
photos	Count of profile pictures of a worker
worker_language	Profile language setting
count_cv	Count of uploaded resumes
count_certificate	Count of uploaded certificates
count_diploma	Count of uploaded diplomas
count_testimonial	Count of uploaded testimonials
count_drivinglicenses	Count of driving licenses
was_referred	Whether worker was invited to the platform
receive_newsletter	Whether the worker subscribed to newsletter
receive_phone_calls	Whether worker allows phone calls
driving_skills	Count of driving skills
speaking_level_a	Count of languages spoken on level a
speaking_level_b	Count of languages spoken on level b
speaking_level_c	Count of languages spoken on level c
speaking_level_l	Count of languages spoken on mother-tongue level
writing_level_a	Count of languages written on level a
writing_level_b	Count of languages written on level b
writing_level_c	Count of languages written on level c
writing_level_l	Count of languages written on mother-tongue level
avg_reactiontime	Average time between receiving and reacting to the request
avg_jobduration	Average length of the jobs accepted
avg_wage	Average wage of the jobs accepted
min_wage	Lowest wage of the jobs accepted
avg_salary	Average salary of the jobs accepted
avg_distance	Average distance of the jobs accepted

Table 3: Features of the Worker Data

3.1.3 The skill data

The skill set of a worker is defined as the amount of chosen job profiles over wich they want to work. Each job profile has at least one corresponding proficiency or education level. The latter is referred to as the education level in the following. A skill is defined as the combination of a job profile and its education level. Employers can search for workers that possess the required skills for a job. For a job in the hospitality industry with a required job profile *chef* and education level *skilled* for example, all workers fulfilling these requirements can be found on the platform. Some skills do not require an education whereas other skills require a certain background. For instance, a worker can choose the skills *Chef: Skilled* and *Waiter: Unskilled* because the worker is an educated chef but has no experience, nor is educated as a waiter. For the first skill, the platform asks the worker to provide a proof for the proficiency, but not for the latter skill. On the other hand, the employer chooses the job profile and education level for the job which fits the task. All skills can be grouped into different granularities using different grouping dimensions. The proficiecy or education and the industry of a skill are the used grouping dimensions. When grouping by education, a skilled chef is more similar to a skilled waiter than an unskilled chef, as the education level is the same. In the dimension of industries, however, the skilled chef is considered closer to an unskilled chef than to any waiter. In the used knowledge base, five different granularities exist for skill education and two more granularities can be formed using available information of the skill industry. Table 4 gives an overview of the seven different skill granularities for education and industries used in this setting.

Granularity ID	Description
E191	Job profile with education levels
E91	Job profile without education levels
E85	Education level without job profiles
E31	Grouped education levels, small groups
E6	Grouped education levels, six big groups
I5	Industry of the job profile and education level
I10	Subindustry or sector of the job profile and education level

Table 4: Skill Granularity Overview

The number in the skill granularity ID states the amount of distinct levels within each granularity. The education levels range from unskilled to having a university degree and combine proficiency and education of a worker. The more levels a granularity includes, the less skills are grouped together to build one granularity level. The lowest and finest skill granularity has 191 different levels and mirrors the job profiles and education levels used on the platform. Thus, this granularity contains as many levels as there are skills to choose from. The examples in Table 5 show how skills can take on different values based on different granularities.

Granularity ID	Granularity levels		
E191	Chef: Skilled	Concierge: Unskilled	Office work: Semi-skilled
E91	Chef	Concierge	Office work
E85	Federal Education	No education	No education but experienced
E31	Skilled	Unskilled	Unskilled but experienced
E6	Skilled	Unskilled	Unskilled
I5	Hospitality	Hospitality	Office
I10	Gastronomy	Hotel sector	Office support

Table 5: Skill Granularity for Three Different Skills

The required skill of a job and the supplied skill set of a worker could be added to the set of features. However, both required and provided skills have strongly skewed distributions on the platform as it focuses on certain industries. Including the levels of skill granularities individually would bias the results towards the more used skills and does not pick up any matching effects. For this reason, a matching score is calculated for each required skill of a job. It compares the required skill to the provided skill set of a worker regarding the amount of other relevant skills than the required skill. The relevance is defined by the granularity used to calculate the score. It is measured by the percentage of relevant skills a worker provides from all relevant skills of a job.

$$\begin{matrix}\text{Matching}\\\text{score}\end{matrix} = \frac{\text{amount of provided skills within required granularity level}}{\text{total of relevant skills of required granularity level}} \times 100\% \quad (1)$$

For example, if the required skill of a job is a skilled chef and the E31 granularity is chosen to calculate the matching score, then a score of 0.5 is assigned to a worker which has half of all the skills with the E31 granularity level *Skilled*. The score is higher, the more relevant skills a worker can provide. IT is then added as a feature for each granularity separately to increse a models explanatory power and is used to determine the optimal matching granularity.

3.1.4 Interaction terms

To capture more complex behaviors that might also have additional explanatory power, interaction terms are introduced into the data. The matching of the worker information with the requirements of the job are picked up by variables interacting between the entities. For example, the difference between the wage of the job and the average wage a worker accepting a job will provide explanatory power for the outcomes. In this regard, the matching score of the skill data is also an interaction term. Table 6 shows the interaction terms used in this setting.

Interaction term	Description
wage_diff	Diff. between job wage and average wage of jobs a worker accepted
min_wage_diff	Diff. between job wage and lowest wage of jobs a worker accepted
duration_diff	Diff. between job duration and average duration of jobs a worker accepted
salary_diff	Diff. between job salary and average salary of jobs a worker accepted
distance_diff	Diff. between distance to job and the average distance to jobs a worker accepted
maching_score	Measures how much a skill set matches the required skill for a certain granularity

Table 6: Interaction Terms

3.2 Data Models

This section describes the data models used to train the machine learning methods. As mentioned, the matching algorithm is improved by modelling the probability of a worker accepting a request and by modelling the rating a worker would receive for a given job. The base data model contains all relevant features from the request or rating data, the company data, the worker data, and the interaction terms, but excludes the matching score. It can be represented using the equation (2). Y_{rwc} is the outcome variable measuring how worker w answered the request r from company c. X_w is the set of worker features, X_c is the feature set of the company c, X_r is the feature set of the request or the ratings and X_i is the set of interaction terms between worker w and request r.

$$\begin{aligned} Y_{rwc} &= \beta_r X_r + \beta_w X_w + \beta_c X_c + \beta_i X_i + \epsilon_{rwc}, \\ Y_{rwc} &= \begin{cases} \text{accepted} \\ \\ \text{declined} \end{cases} \quad \text{or } Y_{rwc} = \{1,2,3,4\} \end{aligned} \tag{2}$$

In a further step, the matching score of each granularity is added separately to the base model, in order to find the optimal granularity allocation. The goal of the data model is to minimize the error term ϵ_{rwc} by optimizing the coefficient vectors β_r, β_w, β_c, and β_i, which are the partial effects of the features on the outcome variable. Since the rating data is a subset of the request data, the same feature set can be used for both data models. Thus, the rating data model is similar to the request data model, except that the outcome variable in the rating data model takes on an integer from 1 to 4 while it takes on *accepted* or *declined* in the request data model.

3.3 Summary of the Outcomes

The data at hand is restricted to requests and ratings that were generated within a time period of 15 months between 2018 and 2019. During this period, 9.75 million requests were sent to $66'145$ different workers on the platform, each receiving one of three potential outcomes. On average, a worker received 230 requests during this period, answered 29.4% and accepted 10.5% of them. Figure 1 shows the distribution of the worker answers of all requests.

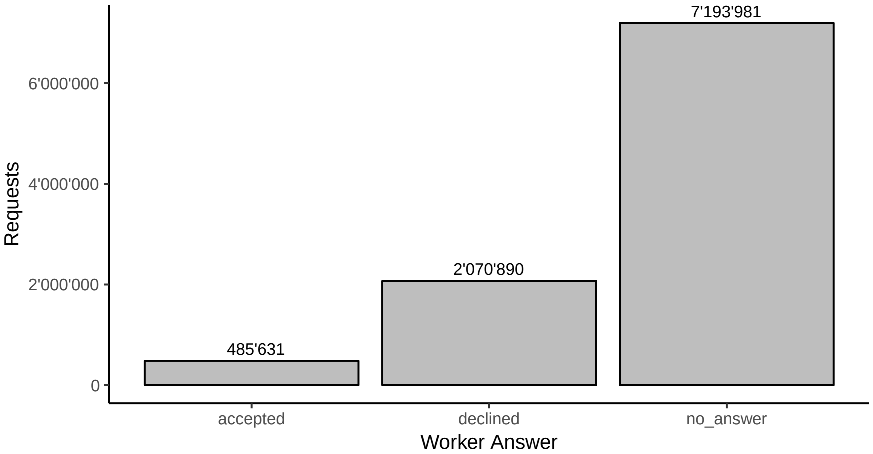

Figure 1: Worker Answer Distribution

About 73% of the requests are unanswered, and only about 19% of answered requests are accepted by the worker. This highlights that the algorithm sending out requests has a large potential for improvements in targeting workers. Since the hypothetical answer of those that did not react to the request are unknown, these observations are excluded from the request dataset for the purpose of training. The remaining data contains 2.55 million requests where the worker either accepted or declined. As stated previously, unanswered requests are included to measure final improvement results.

From these answered requests, $485'631$ requests have been accepted and $76'422$ ratings of workers have been created. The ratings range from 1 to 4, where 4 is the highest. Figure 2 shows the distribution of those ratings. The majority of workers were rated with a value of 4, with a sharp drop to values of 3 and an even lower amount of rating of 2. Compared to ratings of 2, a rating of 1 was given relatively often. This could be explained by the design of the rating system. As there is no center, employers tend to choose the higher one of the intermediate ratings 2 and 3, if the quality of the work was neither exceptionally good nor bad. However, if the job was executed poorly, employers tend towards the worst rating of 1. A full summary statistics of the features used is presented in Table 18 in the appendix.

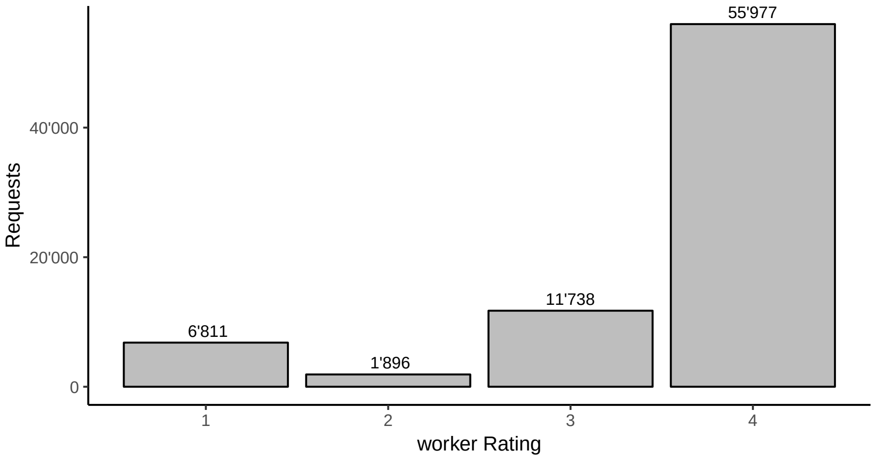

Figure 2: Worker Rating Distribution

4. Machine Learning Models

ML models are tools used to better understand data. Generally, ML models can be classified as supervised and unsupervised models. Supervised statistical learning models require known outcome variables, while unsupervised learning models do not. For the latter case, certain relationships in the data can still be learned and it is used especially in clustering problems. Supervised statistical learning methods are tools used to generalize relationships between features and an outcome variable. Many learning methods can only be used in specific kinds of problems regarding the form of the outcome variables, others can be used in a more general setting. A problem involving continuous outcome variables is called regression problem whereas it's called a classification problem if the outcome is categorical. Methods that increase the performance of a learning model are called techniques. Some techniques are mutually exclusive, such that they cannot be applied together. Lastly, there are techniques to combine different learning methods together (James et al., 2013). Ultimately, the goal is to find learning methods and techniques, such that an outcome can accurately be predicted.

This chapter presents an overview of the techniques and learning methods applied in this thesis, as well as the approach to measure the performance of the models and the improvements to the initial algorithm. Section 4.1 briefly explains the applied statistical learning methods. Section 4.2 illustrates how the performance of the models and the improvements of the matching algorithm are measured. Section 4.3 introduces the enhancement techniques to increase the model performance. Finally, Section 4.4 explains the approach of selecting the best performing model.

4.1 Theoretical Background of Applied Statistical Models

Generally, the performance of a model is related to the bias-variance trade-off (Faber, 1999). A bias arises if a model is not flexible enough in generalization leading to the inability to estimate the real effect of features to the outcome variable. In linear regressions, for example, only linear relationships between the features and the dependent variable are allowed. Models with a low flexibility are well interpretable and can generalize easier to other datasets, but are often too simple to find complex patterns. A more flexible model, on the other hand, will lead to less bias. Variance is defined as the change in the estimates when a different training dataset is used. A model with very high variance has the ability to learn many relations within a training set which also includes noise. This is called overfitting and the model can no longer generalize to other datasets (Dietterich, 1995). Overfitting is one of the central problems to take care of in ML. Since the model's flexibility is inversely related to its interpretability, there exists a trade-off. (Faber, 1999). This trade-off is not only used to select an appropriate learning method which is presented in this section, it is also used in Section 4.3 to legitimize the applied enhancement techniques.

A linear regression model (LM) with a low flexibility is straightforward to interpret, compared to a complex neural network (NN) with high flexibility. However, a high interpretability does not necessarily mean that the model will not be competitive in making prediction. Likewise, high flexibility does not imply a good performance in predicting as the model might overfit faster (Faber, 1999). Since there is no method that overperforms in every aspect, different methods need to be compared (James et al., 2013). For comparative purposes, a wide range of models in regard to flexibility and interpretability are considered, as presented in Table 7.

Learning method	Models
Regression	Linear Regression Model (LM), Logistic Regression Model (LOG)
Classification methods	K-Nearest Neighbor (KNN), Support Cector Machines (SVM)
Tree-based methods	Random Forest (RF), Extreme Gradient Boosting (XGB)
Neural networks	Feed Forward Neural Network (FFNNET)

Table 7: Learning Methods Overview

4.1.1 Regression

Due to the simplicity and good interpretability of regressions, they are widely used in machine learning and have served as a starting point of many newer approaches. It requires a mapping of a function through a set of data points, such that the average difference between the data points and the function is minimized. However, it is not flexible as it assumes a certain form of the relationship between the features and the outcome. For example, the relationship is assumed to be linear between all the features and the outcome when using a LM (Goldberger, 1962). On the other hand, the simplicity allows for straightforward interpretation of the model and thus is commonly used for inference (James et al., 2013). It is often considered as a benchmark for the performance of ML models. Moreover, it is possible to solve classification problems using a logistic regression model (LOG). This ensures the predictions to not exceed the boundaries of a binary outcome probability (Press & Wilson, 1978).

4.1.2 Classification methods

Classification methods were initially built to solve classification problems with categorical outcome variables where two or more qualitative levels are given. For this, an observation is assigned to a specific outcome class. The classification methods are highly flexible and do not assume any specific form of relationships and have shown to perform well for a range of problems (Chang, 1979). The downfall of classification methods is its reduced interpretability, as no coefficient table like in regression methods can be created (James et al., 2013). However, some modifications enable these methods to be used for regression problems (Weiss & Indurkhya, 1995). A widely-used classifier is the K-nearest neighbor classifier (KNN), where a neighbor is defined as the closest observation for a given data point with respect to its feature space. KNN assigns a class to the outcome according to the majority of chosen neighbors of that observation (Chang, 1974). The choice of the amount of neighbors depends again on the bias-variance trade-off where a lower number increases flexibility (James et al., 2013). Another widely used classifier is the support vector machine (SVM), which separates spaces of features with flexible functions called kernels. These kernels allow the model to learn non-linear relationships between the features and the outcome variable. The SVM can also be applied on both classification and regression problems (Amari & Wu, 1999).

4.1.3 Tree-based methods

Tree-based methods can also be used for both regression and classification problems, where at least one decision tree is built. It involves segmenting the feature space into regions. Splitting rules are applied for this segmentation, which can be displayed as a tree and gives this type of methods its name. The simplicity of a tree is useful for interpretation but, in terms of prediction, the highly flexible and non-parametric approach makes a single tree not competitive with other approaches (Weiss & Indurkhya, 1995). Therefore, tree-based methods are typically used with ensemble techniques described in Section 4.3.4, which involves more than one tree (James et al., 2013). By bootstrapping the training set and randomly subsetting the features, different trees are created and combined into one ensemble model, the random forest (RF) (Breiman, 2001). On the other hand, the extreme gradient boosting algorithm (XGB) boosts decision trees, which enables XGB to learn over time. A benefit of the XGB is the fast computing time, which makes it more convenient to train and test models (Chen & Guestrin, 2016).

4.1.4 Neural networks

NN are another branch of machine learning. Essentially, NN enables the modelling of complex real-world environments by mapping it in a nested hierarchy of layers of neurons. Each neuron, which is a model itself, represents a function that learns from input data and provides the output as an input for another neuron in the next layer. The last layer of a network forms the output into the desired form, which is the prediction of the outcome variable. The initial form of NN is inspired by neuroscience and gives NN its name. Due to the flexibility using neurons and layers, the requirement of input data is relatively high. The simplest but fundamental model is the multilayer perceptron, also called the feed forward neural network (FFNNET). FFNNET receives its name from the information flow which has only one direction. It has no feedback connections which feeds output of a NN back into the NN itself (Bengio, Courville, & Goodfellow, 2016).

4.2 Performance Measurement

The performance of these models is assessed by letting each trained model predict on the test set that was previously separated from the training set. This way of assessing is called out-of-sample measurement. The prediction is compared to the actual outcome, while the difference between outcome and prediction is called error (Tashman, 2000). However, there are many different performance measures. The choice of the measure depends on the outcome variable and the problem that needs to be solved (Hajian-Tilaki, 2013). In the following, different measures of performances for classification and regression problems are introduced.

4.2.1 Performance measures for regression problems

A measure that evaluates the performance of regression models relates the predicted outcome $\hat{y}_i$ to the observation y_i and measures the distance between them. The following measures are widely used to measure the performance of said models (Becker & Kennedy, 1992; Benesty, Chen, & Huang, 2008; Wilmott & Matsuura, 2005).

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2} \tag{3}$$

$$\mathrm{MAE} = \frac{1}{n}\sum_{i=1}^{n}|y_i - \hat{y}_i| \tag{4}$$

$$\mathrm{R}^2 = 1 - \frac{\sum_{i=1}^{n}(y_i - \hat{y}_i)^2}{\sum_{i=1}^{n}(y_i - \bar{y})^2} \tag{5}$$

$$\text{Adjusted } \mathrm{R}^2 = 1 - (1 - \mathrm{R}^2)\frac{N-1}{N-M-1} \tag{6}$$

$$\begin{aligned}\mathrm{PCC}^2 &= \frac{n\sum^n y_i\hat{y}_i - \sum^n y_i \sum \hat{y}_i}{\sqrt{n\sum y_i^2 - (\sum y_i)^2}\sqrt{n\sum \hat{y}_i^2 - (\sum \hat{y}_i)^2}} \\ &= corr(y_i, \hat{y}_i)^2\end{aligned} \tag{7}$$

The root mean squared error (RMSE) calculates the square root of the average squared difference between the prediction and the observation, while the mean absolute error (MAE) is the average of the absolute difference between the prediction and the observation. The difference between the two is that the MAE weights each observation equally and is more interpretable. The RMSE squares the errors, which results in giving higher weights to larger errors (Wilmott & Matsuura, 2005). In other words, the RMSE penalizes larger errors more than MAE. However, even though the goal is to minimize both RMSE and MAE, they can range between zero and infinity, meaning that both measures alone cannot tell whether the fit of the model is good or not. On the other hand, the R^2 and the adjusted R^2 are absolute measures for the fit of a model, where a perfect fit results in $R^2 = 1$ and a negative value implies that the model predicts worse than taking the mean of the outcome $\bar{y}$ as the prediction. Hence the R^2 and the adjusted R^2 are scale-free measures and ranges between minus infinity and one (Becker & Kennedy, 1992). The R^2 however, assumes that every feature helps in explaining the outcome, which is not always true. Thus, the adjusted R^2 is also calculated, which adjusts for features with low explanatory power by considering the amount of observations N and the amount of features M (James et al., 2013). Finally, the squared pearson correlation coefficient (PCC^2) is another measure of fit and is defined as the squared correlation between the predictions and the observed outcomes. It is used similarly to the R^2 and ranges between 0 and 1 (Benesty et al., 2008).

4.2.2 Performance measures for classification problems

The predictions of a classification problem is an assignment of probabilities to each of the possible outcomes (Sokolova & Lapalm, 2009). Since the prediction is a probability of a class, the performance can be measured as the rate of correctly predicted classes for a certain threshold of the probability. The measures used for comparing models with a threshold are best explained by the confusion matrix in Table 8 (James et al., 2013).

	Prediction		
Reference	Declined	Accepted	Total
Declined	TN	FP	N
Accepted	FN	TP	P
Total	N*	P*	

Table 8: Confusion Matrix

The confusion matrix takes a threshold of probability p and assigns the prediction to one class, e.g. all $p > 0.5$ are assigned to positive (P^*), and negative (N^*) otherwise. True positives (TP) and true negatives (TN) are the amount of correctly predicted observations in each class while false positives (FP) and false negatives (FN) are incorrectly predicted observations for each class. P is the total amount of positive outcomes and N is the total amount of negative outcomes among the actual observations (Sokolova & Lapalm, 2009). Furthermore, there are cases when FP should be valued more than FN and vice versa. The appendix contains two examples from Hajian-Tilaki (2013) and Keijser, de Lange, and van Wilsem (2014), in which the error types are of opposite importance.

The threshold must be chosen carefully depending on the weight given to the error types (Keijser et al., 2014). A higher threshold results in more FN and lowering the threshold increases the FP. The threshold for the results in Chapter 5 are chosen by minimizing both error types equally. For the improvement results in Section 5.5, two additional thresholds are added. Each additional threshold is minimizing one of the two error types, while keeping the remaining error type constant. This results in a boundary of the improvement, depending on the importance of each error type. Starting from the confusion matrix, different performance measures can be calculated. The FP rate, the TP rate and the positive predictive value, also known as precision, are commonly used performance measures. The accuracy and the balanced accuracy are performance measures of the model for both outcome classes simultaneously. The former measures the percentage of correctly classified observations while the latter one adjusts for class imbalance of the outcome variable (Sokolova & Lapalm, 2009). Table 9 gives an overview of the aforementioned measures (James et al., 2013; Keijser et al., 2014; Sokolova & Lapalm, 2009).

Name	Definition	Synonyms
FP rate	FP/N	Type I error, 1-specificity
TP rate	TP/P	1-Type II error, recall, sensitivity, 1-FN rate
Pos. Pred. value	TP/P*	Precision, positive predictive power
Accuracy	(TP+TN)/(N+P)	
Balanced Acc.	(TP/P + TN/N)/2	

Table 9: Error Measures for Categorical Outcomes

Next to these static measures, there are measures for categorical outcomes that are independent of the threshold (Sun, Wong, & Kamel, 2009). One of such measures is called the Receiver Operating Characteristic (ROC) and trades off the TP rate with the FP rate. Plotting the rates for all possible thresholds results in a curve. The area under the ROC curve (ROCAUC) indicates how accurate a model is for any threshold (Bradley, 1997). However, Saito and Rehmsmeier (2015) showed that the ROC curve does not change with increasing class imbalance, resulting in the ROCAUC to be independent of class imbalance. Therefore, the Precision-Recall (PR) curve should be used instead (Saito & Rehmsmeier, 2015). The PR curve trades off the positive predictive value with the TP rate. The PR curve also creates a measurable area under the curve (PRAUC). An example of these curves is presented in Figure 3, including a scale indicating the thresholds on each point on the curve.

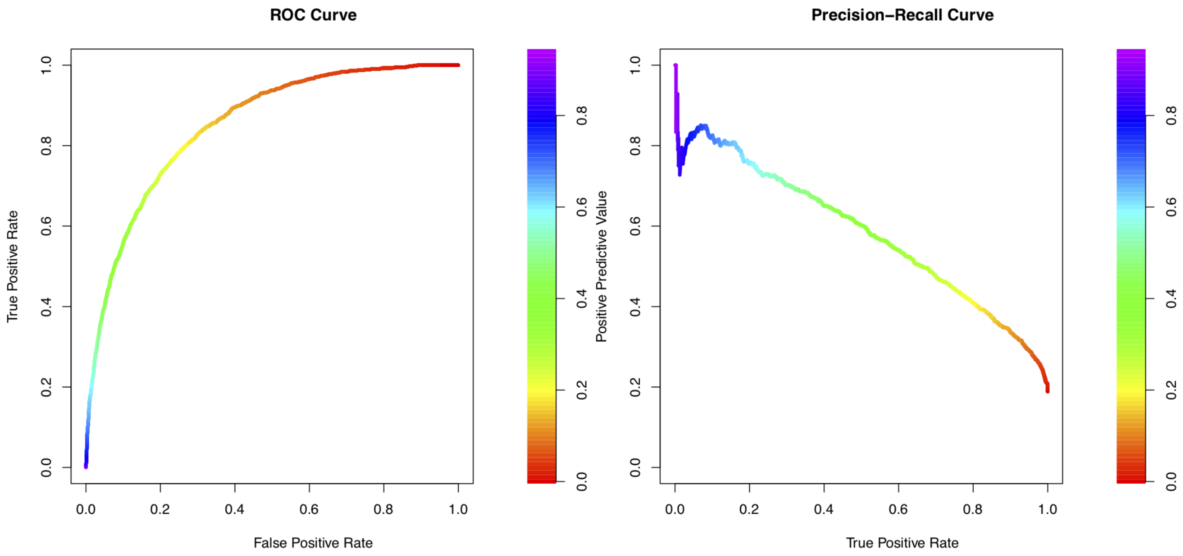

Figure 3: ROC Curve and PR Curve Example

The plot to the left shows the ROC curve and the plot to the right shows the PR curve. The more accurate a model, the closer the ROC curve is to the top left corner for the ROC curve, and the closer to the top right corner for the PR curve, each results in an area closer to 1.

4.2.3 Measuring algorithm improvements

The main problem of the initial matching algorithm on the gig work platform is that too many workers are being requested for a job. This thesis addresses this problem using two different approaches. First, workers which are likely to accept a request are identified. The second approach is to find the workers with the highest probability to receive a good rating. Both predictions are used to reduce the number of unsuited requests separately. In addition, the latter approach can be used to increase the user experience for employers and workers on the platform and could lead to an increase in user retention. This section describes the process of measuring the improvements using the two approaches.

The initial matching algorithm sends an amount of requests to workers for a job. A threshold of the prediction then defines the amount of requests that should not have been sent, in order to reduce the requests to an optimum number. The improvements to the algorithm are then measured by the amount of all predicted requests that could have been reduced. Besides the FP rate where workers were wrongly requested, also the FN rate is measured, where workers were wrongly excluded from the requested workers. The latter two measures provide an intuition of how accurately the ML model performs when applied to new data. The same is applied to predicted worker ratings. Figure 4 illustrates how the improvements are measured.

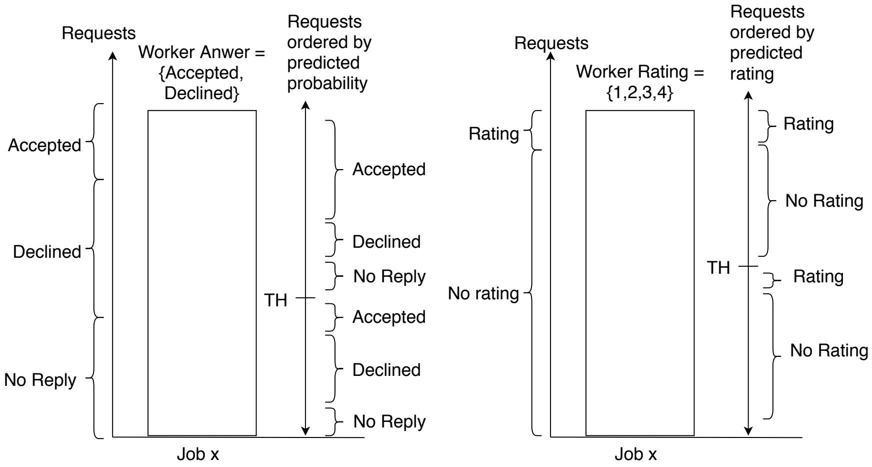

Figure 4: Measuring Improvements of the Algorithm

The figure shows a given job x with a certain amount of requests grouped into the outcomes for both outcome variables. The threshold (TH) defines the amount of reduced requests regarding the predicted worker answers and predicted worker ratings. However, unlike before, the threshold also affects workers that have not answered the request or were not rated by the employer. These observations cannot be used to measure the accuracy of the model using the threshold. Thus, next to the reduced requests for all requests, the performance is measured as the accuracy of those, for which the outcome is observed. The plot on the left of Figure 4 shows the amount of requests and the amount of each answer for a given job. It also indicates that a potential threshold defining the requests which should be sent, will predict workers to be requested which declined or have not answered the request. The plot on the right of Figure 4 illustrates the same job with certain workers being rated by the employer. The workers with a predicted rating below a threshold are excluded from being requested. Finally, it shows that the choice of the threshold is central for the degree of improvement.

Moreover, the choice of the threshold depends on the acuteness to cover the job with workers, subsequently referred to as the *urgency* of a job. On one side, there are urgent jobs in which it is difficult to find interested workers and on the other side, there are less urgent jobs in which plenty of workers are interested. Besides the time left until the beginning of the job, the urgency also depends on the location of the job, the required skill, the wage and many other variables. Therefore, the urgency of a job cannot be measured in a straightforward approach. However, in the case of an urgent job the threshold must be lowered to increase the amount of requests sent, which increases the TP rate but also increases the FP rate. In the case of a less urgent job, the threshold can be increased to send less requests, which results in an increase of the TN rate but also increases the FN rate. By giving weight to either the FN rate or the FP rate, different optima of the threshold can be calculated. For this reason, the results of improvements are presented in Chapter 5 using bounded limits of the threshold.

4.3 Model Enhancement Techniques

Model enhancement focuses on improving existing models by either preprocessing the data appropriately, further improving the data handling or by increasing the complexity of the model. In the following, these techniques are referred to as *model enhancement techniques* and are intended to optimize the bias-variance trade-off. This section describes the four general enhancement techniques applied to the different models in this thesis, namely preprocessing, feature selection, resampling, and ensemble learning.

4.3.1 Preprocessing

Before the data is used to train the ML models, it is transformed by a method called preprocessing. It involves different kinds of data cleaning and data transformation techniques. It has been shown that preprocessing can increase the performance of machine learning (Rasekhi, Mollaei, Bandarabadi, Teixeira, & Dourado, 2013). As a first step, missing values in the dataset are cleaned. An approach called mean imputation is used here, which replaces the missing values with the mean of the variable. It has the advantage of leaving the mean of the variable unchanged, as opposed to other approaches such as median imputation (Donders, van der Heijden, Stijnen, and Moons, 2006). In the data at hand, missing values are often a result of new functionalities being introduced on the platform, creating missing values for all observations created before the new functionality was introduced. This is the reason why the data used here is limited to this specific time period of 15 months.

ML models do not allow for qualitative variables, thus all features with a qualitative value are one-hot encoded (Potdar, Pardawala, & Pai, 2017). To achieve this, a dummy variable is created for each level of the variable. In order to avoid perfect multicollinearity, one dummy variable per feature is excluded. This is feasible as the variable has no qualitative order such as *small* and *big*. Additionally, all quantitative variables are standardized, that is, to linearly transform the values of the features into a standard range by subtracting the mean and dividing by the standard deviation. These procedures increase the performance of models by increasing the internal consistency of the data (Kotsiantis, Kanellopoulos, & Pintelas, 2006). The last transformation is applied to the rating outcome itself by dividing it by the average rating a company assigned to workers. This reduces the differences in individual employer behavior when they rate the workers.

4.3.2 Feature selection

Feature selection is a model-independent technique to select a subset of features of a model with high explanatory power. If a feature is not related to the outcome, this feature will only add noise to the data. The removal of these kinds of features helps reducing overfitting and thus increases the accuracy of out-of-sample predictions. Furthermore, less features leads to a reduced training time, which plays a role if high computational power is required (Guyon & Elisseeff, 2003). There are many different techniques for feature selection. The features under study here have gone through a manual selection process during feature engineering, such that only features expected to be related with one of the outcomes are included. Among others, business experience and correlations have been applied to identify important features. Moreover, the applied feature selection wrapper algorithm confirms the selected features to be important. In essence, this algorithm compares different combinations of features to find the set of the most important features to be used and leads to the set of features shown in Chapter 3 (Kursa & Rudnicki, 2010).

4.3.3 Resampling

Resampling techniques are used to obtain additional information by fitting a given model on repeatedly drawn samples of a dataset. These techniques are independent of the model and can be applied to a wide range of methods. However, resampling is computationally expensive, as it requires to fit the same model multiple times. The most commonly used resampling technique is the k-fold Cross-validation (CV) which allows to calculate an out-of-sample error for a given dataset (Picard & Cook, 1984). A dataset is split into k parts, where $k \in \mathbb{N}$. A given model then takes one of those k parts as a test set and fits the model on the remaining $k - 1$ parts. After that, the held out test set is used for computing an out-of-sample error, replacing an in-sample fitting error. This is repeated for each of the k parts to create k models which are then averaged into a single model. The choice of k is again a problem of the bias-variance trade-off. A high k results in more correlated models trained on more similar training sets than a low k. The mean of correlated models tends to have a higher variance than the mean of less correlated models (Picard & Cook, 1984). Lastly, the needed computational power increases with k. Values between $k = 5$ and $k = 10$ have been empirically shown to have good balance between bias and variance (James et al., 2013). This technique is applied on the ML models in this thesis to further increase the performance.

Another common resampling technique is bootstrapping, where random samples are repeatedly drawn from the dataset. Since the sampling happens randomly, the samples and thus the created models from each sample are less correlated with each other than with non-random sampling techniques (Efron & Tibshirani, 1993). This can further reduce the variance of a given model (James et al., 2013).

Finally, resampling techniques can be used to counter class imbalance in a dataset. As shown in Chapter 3, the outcome variables for the categorical outcomes are highly imbalanced. This does not only cause issues for interpreting results, but also reduces the accuracy during model training. To counter this, samples are not randomly created as with bootstrapping, but are created in a way that balances the outcomes. This is subsequently referred to as *rebalancing*. In particular, such a sample can be created by undersampling the majority or by oversampling the minority of the outcomes with replacement (Chawla, Bowyer, Hall, & Kegelmeyer, 2002). Chawla et al. (2002) proposed the synthetic minority oversampling technique (SMOTE), where the minority and the majority is synthetically oversampled, and undersampled, respectively. This procedure is called synthetic because new observations are introduced by copying existing observations with certain transformation operations. To counter the class imbalance in the data at hand, these rebalancing techniques are also applied to each method separately.

4.3.4 Ensemble learning

Ensemble learning is a method that combines different models. Widely speaking, applying resampling to a model can also be seen as a kind of ensemble learning as it combines models that were trained on different subsets of the training data. However, ensemble learning addresses another problem than resampling techniques. In some cases, a model suffers from high variance, meaning that the model is not able to generalize. In an ensemble model, the model resulting from model combination is less volatile than the individual models (Dietterich, 2000). The combination of multiple models is the average of the individual predictions of the underlying models and thus reduces the variance. Hence ensemble learning can increase the prediction accuracy. Models gain the most from ensemble learning if the correlation between models is low (James et al., 2013). Less flexible models such as regressions are more robust and suffer less from high variance, which is why such models profit less from ensemble learning than more unstable models (Dietterich, 2000).

Boosting is another technique that can be applied to a range of models. Similar to ensemble techniques, boosting combines multiple models to form one prediction (Chen & Guestrin, 2016). The idea of boosting, however, is that the models are built sequentially. Each model is built with the target of minimizing the error of the previous model. Thus, with boosting, a model learns over time instead of fitting once to the available data. Unlike in ensemble learning, choosing too many sequential models can lead to overfitting. This is avoided by combining boosting with other resampling techniques such as CV, to find the optimal amount of sequential models (James et al., 2013).

4.4 Model Selection Approach

In order to find the best performing model to improve the matching algorithm, the performance of models are evaluated in three steps. A fair comparison of model performances is required, such that models only differ in one aspect while everything else remains constant. To do so, the base data described in Section 3.2 is enhanced with the enhancement techniques described in Section 4.3 to train and evaluate a given learning method. In particular, model specifications are identified that maximizes the performance for a given model. This leads to the first model performance results for each model with the best fitting augmenting technique and the best model specifications. In a second step, the best skill granularity is chosen by evaluating the two best performing models from step one including the matching scores on different granularities. Finally, the optimal combinations of specifications and techniques of the best performing model is then used to draw a comparison to the original matching algorithm. This procedure is described comprehensively in the following.

As a first step, the data is split into a training, a test and a validation set. The validation set contains the data of the last three months out of the 15 months period. The remaining requests and ratings are the ones created in the first 12 months of the specified period. The remaining dataset is then randomly split into the training and the test set with a 6 : 4 ratio, while retaining the same outcome proportions. Using the data from the last three months out of the 15-month period as the validation set has the advantage of pretending to predict new data where outcomes are already observed.

A learning model is trained using the training set and different model hyper-parameters and enhancement techniques. Hyper-parameters, such as the amount of trees for a RF algorithm are external to the models. They must be specified before training the model. These specifications were selected and optimized through manual random search, which is more effective than grid search when new models are created (Bergstra & Bengio, 2012). It has been shown that different hyper-parameters are important in different datasets and only a minority of hyper-parameters really matter. Manual search involves observing the performance of a model and compare it to the same model with an adjusted hyper-parameter. In grid search, the hyper-parameter adjustments are predefined as a range of hyper-parameters to find the best specification. The trained

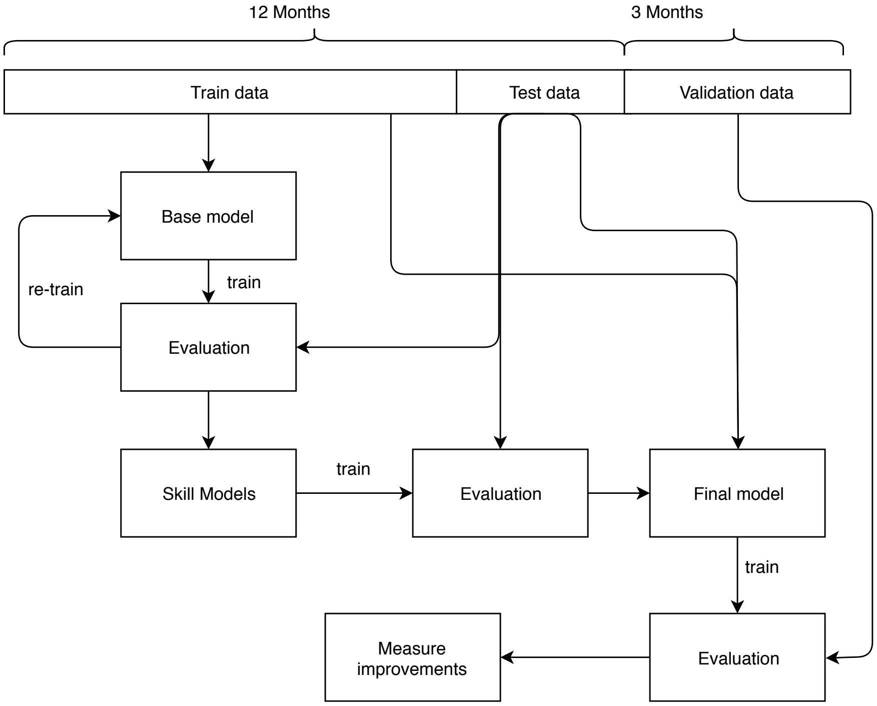

Figure 5: Data Usage Overview for Model Selection Approach

model then predicts on the test set to evaluate the out-of-sample error. The techniques and specifications with which the given statistical learning model is performing best are considered for the final model. This is repeated for all the different learning methods, such that it is possible to select only the best models for further evaluation. The best models are then enhanced with the matching score of different skill granularities, to find the optimal skill granularity allocation and the final best performing model. Lastly, the final model with the optimal skill granularity is re-trained using the combination of training and test set, in order to predict the validation set. The final predictions are then used to measure the improvements to the initial matching algorithm. The overview of the data usage to find the best model is shown in Figure 5.

5. Empirical Analysis and Results

This chapter presents the results of the empirical analysis. The models and their specification used are described in Section 5.1. The results of the base models are then shown for both predictions of worker answers in Section 5.2 and of worker ratings in Section 5.3. In Section 5.4 the optimal skill granularity is evaluated by adding the scores of the skill granularities to the best two base models individually. The best model with the optimal skill granularity is then used to predict the validation set. The potential improvement of the matching algorithm is then quantified by evaluating the predictions of the validation set in Section 5.5 for both worker answers and worker ratings.

5.1 Model Overview

This section is dedicated to the models under evaluation and their specification of hyper-parameters and enhancement techniques used in this thesis. Table 10 presents an overview over the best specifications chosen for each model and the outcome types for which the models are used.

Model	Outcome	Specification
LM	Continuous	5-fold CV, 5 repeats
LOG	Categorical	5-fold CV, 5 repeats
KNN	Both	10-fold CV, 10 repeats, k=11
SVM	Both	CV, radial kernel, C=0.1, sigma=0.0156
RF	Both	10-fold CV, 5 repeats, 10 trees, 7 features
XGB	Both	5-fold CV, rounds=600, depth=4, eta=0.1
FFNNET	Both	5 layers, 50 units or less per layer, regularization, dropout rate=0.1, 75 epochs

Table 10: Specifications Used in Learning Models

The LM is applied on worker ratings and is used with a 5-times repeated 5-fold CV, which increases the accuracy of predicted ratings. The other regression model is the LOG and is applied on worker answers. It works similarly to the LM, except that it fits the sigmoid function between 0 and 1 to the data instead of a straight line. This method is also applied with a 5-times repeated 5-fold CV specification. As mentioned previously, these regression models are used as a benchmark.

The classifier SVM applied in this thesis uses a radial kernel. Furthermore, the smoothing parameter sigma is set to 0.0156 and the cost parameter C is set to 0.1, which controls the complexity of the model. The remaining classifier KNN is applied using 11 neighbors. Both the SVM and the KNN models are used with a 10-times repeated 10-fold CV to prevent overfitting of these highly flexible non-parametric models. Since both models entail modifications that enable them to be used in a regression context, they are applied to both worker answers and worker ratings.

The RF model is created with 10 trees, 7 randomly selected features per tree and is used with a 5-times repeated 10-fold CV. On the other hand, the number of trees on top of each other for the XGB is chosen to be 600, while allowing for only four nodes per tree to prevent overfitting. Furthermore, a 5-fold CV is used in between each XGB iteration. The last hyper-parameter to be determined for the XGB is the shrinkage parameter, which controls the rate of learning to reduce overfitting and is chosen to be at 0.008. RF and XGB are applied on both the worker answers and the worker ratings.

The applied FFNNET uses five layers of neurons. The first layer consists of 50 neurons, the second layer 30 neurons, the third and fourth layer contains 20 and 10 neurons, respectively. The layers use rectified linear unit activation functions and regularization, which weights the features individually and reduces overfitting. The last layer has one neuron only, to format the output. For the binary outcome of worker answers, it uses the sigmoid function instead of the rectified linear unit. Between each layer, outputs are normalized and some features are randomly dropped to prevent overfitting. A dropout rate of 0.1 is chosen, which randomly excludes 10% of all features between each layer. Lastly, it is specified with 75 epochs, the number of times the network encounters the entire training set, and a batch size of 32 observations that run through the model in each learning iteration. The FFNNET is used on both the worker answers and the worker ratings.

5.2 Finding High Potential Workers

Since in the training set the worker either accepts or declines requests, the categorical measures shown in Section 4.2.2 are used. As there is a class imbalance in the worker answers, the rebalancing methods are compared first by applying them to the above-mentioned models. Table 11 shows the results of the different rebalancing methods for the XGB model. The rebalancing results of the other models can be found in the appendix from Table 19 to Table 23.

Model	TP rate	TN rate	Precision	Accuracy	Bal. Acc.	ROCAUC	PRAUC
XGB no rebal.	0.7778	0.7055	0.3829	0.7193	0.7417	0.8206	0.5234
XGB over	0.7647	0.7165	0.3879	0.7257	0.7406	0.8184	0.5210
XGB under	0.7848	0.6871	0.3708	0.7057	0.7359	0.8135	0.5046
XGB smote	0.6855	0.4839	0.2379	0.5223	0.5847	0.6164	0.2611

Table 11: XGB Rebalancing Results

The results show that no rebalancing method can consistently improve the performance of XGB when predicting worker answers, as the ROCAUC and PRAUC is the highest without rebalancing methods. However, this is not true for all the other models. This allows to conclude that there is no best rebalancing technique for this dataset, but it depends on the model used. Since using rebalancing techniques removes class imbalance, the ROCAUC measure is used to determine the best rebalancing technique for each model. The best rebalancing technique for each model is then used to compute the performance measures for the base data, shown in Table 12.

Model	TP rate	TN rate	Precision	Accuracy	Bal. Acc.	ROCAUC	PRAUC
LOG	0.6748	0.7117	0.3549	0.7047	0.6933	0.7603	0.4262
KNN	0.6599	0.6826	0.3282	0.6783	0.6713	0.7335	0.3936
SVM	0.7696	0.6536	0.3345	0.6750	0.7116	0.7843	0.4663
RF	0.6889	0.8146	0.4493	0.7920	0.7518	0.8356	0.5597
FFNNET	0.6725	0.6346	0.3019	0.6418	0.6536	0.7725	0.4081
XGB	0.7936	0.7523	0.4295	0.7602	0.7730	0.8571	0.6000

Table 12: Performance of Worker Answer Models

By using the two threshold-independent measures PRAUC and ROCAUC, the table shows that the XGB model and the RF model are performing better compared to the others. The TP rate shows that approximately 79% of all workers accepting the request were correctly predicted by the XGB and 75% of the declines were identified as well. On the other hand, the RF model was able to predict the workers answer correctly in more than 81% from all workers declining, but it only found 69% from all workers accepting. Compared to that, the initial algorithm sends all requests, which results in finding 100% of all accepted requests and 0% of declined requests. The RF also has the highest precision and accuracy. The precision means that from the workers predicted to accepting the request, more than 44% actually did so. This result means that the RF is better at predicting only one of the two outcomes, whereas the XGB is more balanced, resulting in a highest ROCAUC, PRAUC and balanced accuracy (Bal. Acc.).

It might be the case that the best performing model according to ROCAUC and PRAUC is not overall performing best. For a certain threshold implying different weighting of the error types, another model could perform better, even with a lower area under the curves. Figure 6 shows the ROC curves and PR curves of the best performing models. It suggests that the XGB is indeed performing better independent of the weighting of error types. It can be seen in the curves of XGB being closer to the optimal point (0,1) for the ROC curve and to the optimal point (1,1) for the PR curve, compared to any other models for all thresholds. Nonetheless, both the well performing RF and the XGB are chosen for the further analysis.

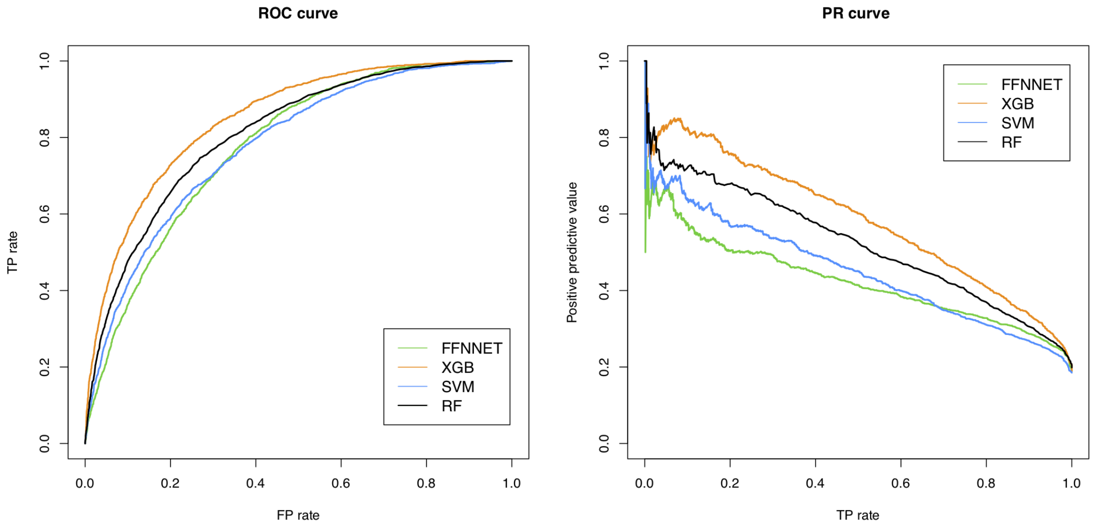

Figure 6: ROC Curve and PR Curves for Worker Answer Predictions of XGB, SVM, FFNNET, and RF

5.3 Finding the Best Workers

The worker rating is a continuous variable, as such rebalancing is not needed. Instead, the rating was scaled to be used in the models. The results for the base models of worker ratings are shown in Table 13.

Model	RMSE	MAE	PCC2	R2	AdjR2
FFNNET	0.8252	0.6125	0.2025	0.1750	0.1752
RF	0.6100	0.3697	0.5509	0.5493	0.5493
LM	0.7009	0.4189	0.4229	0.4048	0.4029
KNN	0.6985	0.4187	0.4174	0.4089	0.4070
SVM	0.6469	0.3464	0.5133	0.4930	0.4931
XGB	0.5858	0.3546	0.5844	0.5842	0.5843

Table 13: Performance of Worker Rating Models

Surprisingly, the FFNNET was not able to generalize resulting in the lowest performance measures in this comparison. It shows that complex models such as a NN are not always competitive and perform worse than a simple LM in this case. Measuring by RMSE, the XGB also dominates the others. This result is also supported by PCC^2, R^2 and adjusted R^2 measures. The SVM and the RF model perform worse than XGB but are still competitive. The SVM model also displays the lowest MAE value, which shows that different models have different strengths. Nonetheless, the RMSE is preferred over MAE measure due to its higher weighting of larger errors and therefore, the XGB model and the RF model are chosen for further analysis.

5.4 Optimal Skill Granularity

After evaluating the performance of the different methods, it is time to find out how the different matching scores of the skill granularities affect the outcomes. To do so, the two best models for both worker answers and worker ratings, namely RF and XGB, are used to measure the impact of adding the skill matching score for each granularity. The skill granularity results for worker answers are shown in Table 14 and the results for worker ratings are shown in Table 15.

Model	TP rate	TN rate	Precision	Accuracy	Bal. Acc.	ROCAUC	PRAUC
RF E191	0.7686	0.7853	0.4681	0.7820	0.7769	0.8566	0.6057
RF E91	0.8125	0.7355	0.4302	0.7507	0.7740	0.8553	0.6088
RF E85	0.8328	0.7184	0.4210	0.7410	0.7756	0.8559	0.6010
RF E31	0.7787	0.7687	0.4528	0.7707	0.7737	0.8534	0.5998
RF E6	0.7889	0.7674	0.4547	0.7717	0.7781	0.8555	0.6037
RF I5	0.8057	0.7442	0.4364	0.7563	0.7750	0.8562	0.6018
RF I10	0.8007	0.7512	0.4418	0.7610	0.7760	0.8541	0.6002
XGB E191	0.8234	0.7422	0.4294	0.7577	0.7828	0.8615	0.5928
XGB E91	0.8112	0.7595	0.4427	0.7693	0.7853	0.8633	0.5976
XGB E85	0.8077	0.7640	0.4464	0.7723	0.7858	0.8640	0.5981
XGB E31	0.8392	0.7290	0.4218	0.7500	0.7841	0.8629	0.5951
XGB E6	0.7395	0.8245	0.4982	0.8083	0.7820	0.8621	0.5951
XGB I5	0.8322	0.7397	0.4296	0.7573	0.7859	0.8635	0.5984
XGB I10	0.8427	0.7232	0.4177	0.7460	0.7829	0.8616	0.5949

Table 14: Results of Worker Answers of XGB and RF for each Skill Granularity

Comparing the ROCAUC measure to the base model results in Table 12 shows that the matching scores increases the explanatory power for both models. It also shows that the matching scores affect the RF and the XGB differently. While for the RF the finest granularity E191 improves the performance the most, it is the E85 granularity which improves the performance of the XGB the most. However, it shows that the I5 granularity ranks second for both models and highlights the importance of the used dimensions to create the granularities, but also that it depends on the used model to determine the optimal skill allocation. The PRAUC measure emphasize that the RF performs better than the XGB for any granularity. Since rebalancing techniques could not improve the performance of both RF and XGB, the ROCAUC is used to determine the optimal skill granularity. Therefore, the E85 skill granularity is ascertained as the optimal skill allocation for matching, using the XGB model on worker answers. These results are not in line with the findings of Pardos et al. (2007), as they found that in general, a lower skill granularity increases the model performance.

Model	RMSE	MAE	PCC2	R2	AdjR2
XGB E191	0.5782	0.3602	0.5811	0.5797	0.5800
XGB E91	0.5782	0.3603	0.5811	0.5797	0.5800
XGB E85	0.5785	0.3614	0.5807	0.5792	0.5795
XGB E31	0.5781	0.3602	0.5812	0.5798	0.5801
XGB E6	0.5781	0.3603	0.5812	0.5797	0.5801
XGB I5	0.5780	0.3601	0.5814	0.5799	0.5803
XGB I10	0.5783	0.3603	0.5810	0.5795	0.5799
RF E191	0.5879	0.3518	0.5682	0.5654	0.5658
RF E91	0.5865	0.3506	0.5701	0.5675	0.5679
RF E85	0.5881	0.3526	0.5680	0.5651	0.5655
RF E31	0.5853	0.3515	0.5716	0.5692	0.5696
RF E6	0.5870	0.3526	0.5692	0.5667	0.5670
RF I5	0.5838	0.3527	0.5740	0.5714	0.5718
RF I10	0.5828	0.3501	0.5757	0.5730	0.5733

Table 15: Results of Worker Ratings of XGB and RF for each Skill Granularity

The results of worker ratings using different granularities in Table 15 show that the results differ more between granularities for the RF than using the XGB model. The MAE reveals that the RF performs better, while all other measures support the usage of the XGB model. The measures show that the industry granularities improve the performance more than any education granularities for both models. This allows to conclude that the industry dimension of skills has more explanatory power to explain worker ratings than the education dimensions. Comparing across models and granularities shows that all measures support the usage of the XGB model with the I5 granularity, except for the MAE which favors the RF model using the I10 granularity. Thus, the XGB is considered as the final model using the I5 granularity to calculate the matching scores. Lastly, the results on the worker rating are again not confirming the findings of Pardos et al. (2007).

5.5 Measure Improvements of the Matching Algorithm

Applying the findings above to the validation data allows to compare the ML model to the initial gig work algorithm. As mentioned in Section 4.2.3 both an upper and a lower threshold boundary is reported depending on the urgency of the request. A third threshold, which optimizes both the FN rate and the FP rate equally indicates the intermediate results.

Recalling the goal of reducing requests, the improvement is measured first in general (genImp) by calculating the percentage of requests that are not sent by ML for the three mentioned thresholds. Then, it is also calculated as an average per job (pjImp). Furthermore, the FN rate and the FP rate for aforementioned thresholds is calculated for both in general and per job as well, if the workers answered requests. Table 16 shows the performance measures of the XGB model and the E85 granularity using the upper, the optimal and the lower bound threshold, respectively.

Boundary	ROCAUC	PRAUC	genImp	genFN	genFP	pjImp	pjFN	pjFP
Optimum	0.84	0.60	77.82	50.10	21.63	75.75	5.84	24.19
optimum	0.84	0.60	58.08	24.44	41.25	55.54	2.77	44.00
Lower	0.84	0.60	38.75	9.45	60.67	35.50	1.06	63.97

Table 16: Improvement Results of the XGB Model for Worker Answers

The lower ROCAUC compared to the results before shows that the prediction on the validation set is worse than on the test set. Due to the change in time of the datasets, a systematic change in the underlying data might have occurred. This also shows the difficulty when using historical data for prediction. Nonetheless, the results suggest improvements between 38.8% and 77.8% in terms of general reduced request, depending on the urgency of a job. The less urgent a job is, the less requests should be sent which results in more improvements compared to the initial algorithm. Using the upper boundary further implies that in general, 50.1% of workers accepting the job offers are predicted to not being requested. Lowering the boundary increases the amount of requests which can decrease this FN rate to 9.4%, while 60.6% of workers declining the requests are also requested. The reason for a lower FN rate per job is that there is a large amount of jobs predicted well by the model. The predictions are less precise for accepting workers in some jobs with a tremendous amount of requests sent. This results in slightly lower improvements per job compared to the general setting, as the potential for improvements per job increases with the amount of sent requests.

On the other hand, the predicted ratings can be used to improve the matching algorithm in two different ways. A threshold of the predicted rating can be chosen to restrict the amount of requests sent. Again, the threshold depends on the amount of requests that needs to be reduced and thus depends on the urgency of the job. The distribution of predicted ratings is shown in Figure 7 in the left plot while the right plot shows the same distribution but excludes unobserved ratings. The latter plot describes a similar distribution to the actual ratings distribution shown in Chapter 3 and implies that the predicted ratings are not too far from the actual ratings. Figure 7 also shows that the predictions of ratings can exceed the boundaries, i.e. it contains ratings below 1 and above 4.

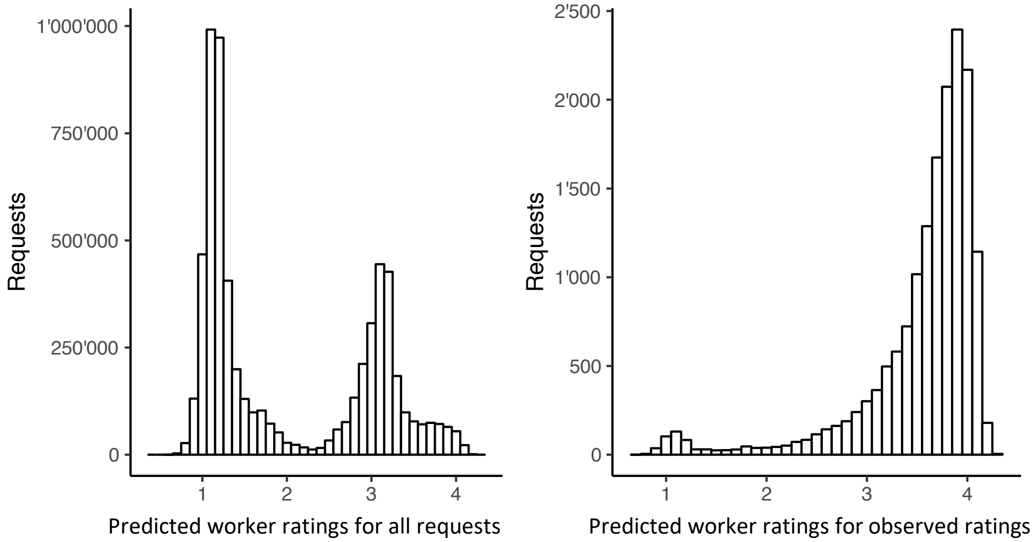

Figure 7: Predicted Ratings Distribution of XGB for all Requests and for Observed Ratings

The cumulative distribution of all predicted ratings in Figure 8 shows the percentage of reduced requests for all possible thresholds of the predicted rating. For example, excluding workers being requested with a threshold of the predicted rating below 3 reduces the amount of sent requests by more than 70%.

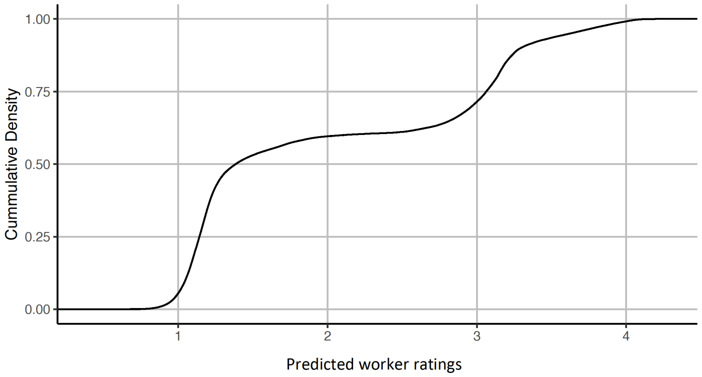

Figure 8: Cumulative Distribution of Predicted Ratings

Since there are only a few ratings per job compared to the requests sent, a measure of differences between predicted and actual rating per job is not representative enough and therefore, a similar trade-off as for worker answers cannot be made. Thus, the worker rating predictions are an improvement of quality without a measure of errors per job. The results of predicted ratings in general for requests with observable ratings are presented in Table 17. The MAE shows that on average, the difference between predicted and actual ratings is about 0.4 rating points. The second way to use predicted ratings is to sort all interested workers according to their predicted ratings in order to improve the experience for the employer and the worker, which is again, not objectively measurable.

Model	RMSE	MAE	PCC2	R2	AdjR2
Final XGB	0.6732	0.4020	0.4753	0.4721	0.4723

Table 17: Performance of XGB for Observed Worker Ratings

The results show that for the task of predicting worker answers, ML has the ability to improve this particular matching algorithm by up to 77.8% in terms of a reduced amount of requests with an FP rate of 21.6%. This means that on average, 21.6% of workers declining the request are predicted to be requested. The results of the rating predictions unveil that improvements strongly depend on the threshold of the predicted rating. Different to predicting the replies of requests, the improvements cannot be optimized by construction. It would involve that the threshold is defined in the beginning to create low and high rating groups, which turns the regression task into a classification task of whether a worker receives a high or a low rating. However, this would not allow to use the quantitative predictions qualitatively, e.g. to rank interested workers according to the predicted rating presented to the employer. Nonetheless, the MAE measure shows an acceptable absolute prediction error lower than half a rating point on average. Finally, both prediction tasks led to significant potential improvements on the initial matching algorithm. The results showed that quantifying improvements depend strongly on a jobs urgency to hire workers while a qualitative use of worker ratings is feasible. Even though the performance of the models in both tasks are adequate and usable, the results still suggests room for improvements.

6. Conclusion

The thesis evaluated the performance of ML on the issue of a matching algorithm by creating models to predict a workers answer for a given request and to predict a workers expected rating for a given job. It has examined different statistical models and methods and has studied the allocation of the optimal skill granularity for skill matching. Resulting ROCAUC measures not exceeding 0.84 for worker answers or R2 not exceeding 0.47 emphasize the difficulty to generalize relationships between the used features and the outcomes on new data. The results on worker ratings showed that generalization is particularly difficult when qualitative measures such as ratings are predicted. One explanation is that each person perceives quality in a different way, which leads to difficulties when predicting it. Additionally, error measures on improvement results for worker ratings per job are not representative and cannot be used, as only a fraction of requested workers are rated. The XGB model had a low depth and a low learning rate but by using 600 rounds it outperformed all other prediction models. The SVM and the RF model performed less accurate and also take much longer to train. Clustering skills into finer granularities did not improve the models, which is in contrast with the contemporary literature of skill matching and skill granularities. Still, the E85 and the I5 skill granularity have been found to be the optimal skill granularity allocation for worker answers and worker ratings, respectively. The thesis shows that ML is able to improve the matching algorithm by predicting worker answers and worker ratings for a given job, such that the amount of requests reduces to an optimum regarding a jobs urgency to hire a worker.

Overall, a stronger performance of the ML models was expected. The following list shows possible reasons for a low performance in descending importance.

- The time frame of the particular 15 months might have unknown implications. Another time period might have led to improved performance. It might be worth to also adjust for seasonality and specific events such as marketing activities or new platform functionalities.

- The class imbalance of the datasets was addressed with scaling and different resampling methods. Other strategies might have led to better results.
- Like the matching score, other interaction term features can be added, if the data is available. Instead of full time averages, time dependent averages can be added to improve the performance.
- A suboptimal amount of features could have decreased the performance, even though a specific selection algorithm was applied. Another selection strategy might have led to an improved performance.
- The assumption that features are stable during the time frame could be unreasonable. Adding more time dependent but computationally more expensive features might increase the performance.
- Only one matching score per granularity was added. The score itself can be enhanced by combining different granularity dimensions to increase the performance. Other dimensions, such as a temporal dependent dimension can also be added if the underlying data is available.

The results of this thesis are of hypothetical nature and represent results that would have occurred if the final model was actually used on the platform. However, uncertainty is still involved because it subsequently depends on human behavior. Therefore, the results of the final model after implementing it on the platform might differ from these hypothetical results. Hence future research should consider solutions to implement the final model on the platform. Only then the effects of adjustments of the thresholds can be directly measured and optimized. Since the improvements depends strongly on the threshold, future research must also focus on optimizing the threshold, which requires to measure the urgency of a job. A potential solution must be able to adjust the threshold according to a jobs dynamic urgency. Moreover, future research should aim at improving the performance of the models. Other strategies to reduce the class imbalance should be tested, for example by using a stratified resampling or reworking the problem itself. New sets of features could be included from the text mining approach of Bottega (2018), which enhances the feature set with information from worker resumes, combined with the estimated average ratings of workers, for example. The gig work platform itself can be enhanced with new functionalities, such as a possibility to state declining reasons for the worker answers. Working closely with the development of the platform could rise a new set of features with high explanatory power critical for the success of such a model. The research must also focus on a scalable database structure of the underlying data. Not only a pipeline of data must be built to constantly feed the models with new observations, also a pipeline to process outputs must be built in order to measure real time improvements. Furthermore, a more temporal related database system can increase the amount of features. Finally, modeling an intermediate process such as the skill matching score through a Bayesian network might increase the performance as well. Future research might find a better solution to make use of the skill data from the platform.

Being able to take advantage of ML will become even more important in the future for any data related company, especially with an increasing amount of data. New models and methods are constantly created in this dynamic field of ML such that insights and predictions of data will become more accurate. Not only will it help decision makers with their decisions, also automation of tasks and other predictions can be made. Besides automated solutions, the ML model presented in this thesis can be used to create a tool for internal Operation Managers to adjust the amount of requests dynamically, as they are constantly observing and learning about each job's urgency. Lastly, the model can be used to create new products on the platform for more prestigious clients by using the predicted worker ratings qualitatively, e.g. by identify premium workers. This shows that ML has not only the potential for improvements but also for new valuable conceptions in all business applications.

Bibliography

Amari, S., and S. Wu. 1999. “Improving Support Vector Machine Classifiers by Modifying Kernel Functions.” *Neural Networks* 12 (6): 783–89.

Becker, W., and P. Kennedy. 1992. “A Lesson in Least Squares and R Squared.” *The American Statistician* 64 (4): 282–83.

Benesty, J., J. Chen, and Y. Huang. 2008. “On the Importance of the Pearson Correlation Coefficient in Noise Reduction.” *IEEE Trans. Audio, Speech & Language Processing* 16 (4): 757–65.

Bergstra, J., and Y. Bengio. 2012. “Random Search for Hyper-Parameter Optimization.” *The Journal of Machine Learning Research* 13: 281–305.

Bottega, C. 2018. “The Influence of Cv Characteristics on Employers’ Satisfaction in Short-Time Employment. A Text Mining Approach.” Master thesis, University of Zurich.

Bradley, A. P. 1997. “The Use of the Area Under the ROC Curve in the Evaluation of Machine Learning Algorithms.” *Pattern Recognition* 30 (7): 1145–59.

Breiman, L. 2001. “Random Forests.” *Machine Learning* 45 (1): 5–32.

Chang, C. 1974. “Finding Prototypes for Nearest Neighbor Classifiers.” *IEEE Transactions on Computers* 23 (11): 1179–84.

Chawla, N. V., K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. 2002. “SMOTE: Synthetic Minority over-Sampling Technique.” *Journal of Artificial Intelligence Research* 16: 321–57.

Chen, T., and C. Guestrin. 2016. “XGBoost: A Scalable Tree Boosting System.” In *Proceedings of the 22Nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining*, 785–94. New York, NY: ACM.

Dietterich, T. G. 1995. “Overfitting and Undercomputing in Machine Learning.” *ACM Computing Surveys* 27 (3): 326–27.

Dietterich, T. G. 2000. “Ensemble Methods in Machine Learning.” *Multiple Classifier Systems* Lecture Notes in Computer Science (1857): 1–15.

Donders, A. R. T., G. J. M. G. van der Heijden, T. Stijnen, and K. G. M. Moons. 2006. “Review: A Gentle Introduction to Imputation of Missing Values.” *Journal of Clinical Epidemiology* 59 (10): 1087–91.

Efron, B., and R. J. Tibshirani. 1993. *An Introduction to the Bootstrap.* New York, NY: Chapman & Hall.

Faber, N. K. M. 1999. "A Closer Look at the Bias–variance Trade-off in Multivariate Calibration." *Journal of Chemometrics* 13 (2): 185–92.

Goldberger, A. S. 1962. "Best Linear Unbiased Prediction in the Generalized Linear Regression Model." *Journal of the American Statistical Association* 57 (298): 369–75.

Goodfellow, I. J., Y. Bengio, and A. C. Courville. 2016. *Deep Learning*. Adaptive Computation and Machine Learning. Cambridge, Massachusetts, MIT Press.

Guyon, I., and A. Elisseeff. 2003. "An Introduction to Variable and Feature Selection." *Journal of Machine Learning Research* 3: 1157–82.

Hajian-Tilaki, K. 2013. "Receiver Operating Characteristic (Roc) Curve Analysis for Medical Diagnostic Test Evaluation." *Caspian Journal of Internation Medicine* 4 (2): 627–35.

James, G., D. Witten, T. Hastie, and R. Tibshirani. 2013. *An Introduction to Statistical Learning: With Applications in R*. Vol. 103. Springer Texts in Statistics. Berlin, Heidelberg.

Keijser, J. W. de, E. G. M. de Lange, and J. A. van Wilsem. 2014. "Wrongful Convictions and the Blackstone Ratio: An Empirical Analysis of Public Attitudes." *Punishment & Society* 1 (16): 32–49.

Konrad, J., and U. Pekruhl. 2017. *Trends zur Flexibilisierung in der Platform Economy*. In M. Zölch, M. Oertig, & V. Calabro (Hrsg.), Flexible Workforce - Fit für die Herausforderungen der modernen Arbeitswelt? (S. 45-77). Bern: Haupt.

Kotsiantis, S. B., D. Kanellopoulos, and P. E. Pintelas. 2006. "Data Preprocessing for Supervised Leaning." *International Journal of Computer Science* 1 (1): 111–17.

Kursa, M. B., and W. R. Rudnicki. 2010. "Feature Selection with the Boruta Package." *Journal of Statistical Software* 36 (11): 1–13.

Lin, Y., H. Lei, P. C. Addo, and X. Li. 2016. "Machine Learned Resume-Job Matching Solution." *arXiv Preprint* arXiv:1607.07657.

Pardos, Z. A., N. T. Heffernan, B. Anderson, and C. Linquist-Heffernan. 2007. "The Effect of Model Granularity on Student Performance Prediction Using Bayesian Networks." In *User Modeling*, 4511:435–39. Lecture Notes in Computer Science. Berlin: Springer.

Picard, R. R., and R. D. Cook. 1984. "Cross-Validation of Regression Models."

Journal of the American Statistical Association 79 (387): 575–83.

Potdar, K., T. S. Pardawala, and C. D. Pai. 2017. "A Comparative Study of Categorical Variable Encoding Techniques for Neural Network Classifiers." *International Journal of Computer Applications* 175 (4): 7–9.

Press, S. J., and S. Wilson. 1978. "Choosing Between Logistic Regression and Discriminant Analysis." *Journal of the American Statistical Association* 73 (364): 699–705.

Rasekhi, J., M. R. K. Mollaei, M. Bandarabadi, C. A. Teixeira, and A. Dourado. 2013. "Preprocessing Effects of 22 Linear Univariate Features on the Performance of Seizure Prediction Methods." *Journal of Neuroscience Methods* 217 (1-2): 9–16.

Saito, T., and M. Rehmsmeier. 2015. "The Precision-Recall Plot Is More Informative Than the Roc Plot When Evaluating Binary Classifiers on Imbalanced Datasets." *PLoS One* 10 (3): e0118432.

Sayfullina, L., E. Malmi, and J. Kannala. 2018. "Learning Representations for Soft Skill Matching." *CoRR* abs/1807.07741.

Sokolova, M., and G. Lapalme. 2009. "A Systematic Analysis of Performance Measures for Classification Tasks." *Information Processing and Management* 45 (4): 427–37.

Sun, Y., A. K. C. Wong, and M. S. Kamel. 2009. "Classification of Imbalanced Data: A Review." *IJPRAI* 23 (4): 687–719.

Tashman, L. J. 2000. "Out-of-Sample Tests of Forecasting Accuracy: An Analysis and Review." *International Journal of Forecasting* 16 (4): 437–50.

Weiss, S. M., and N. Indurkhya. 1995. "Rule-Based Machine Learning Methods for Functional Prediction." *Journal of Artifcial Intelligence Research* 3: 383–403.

Willmott, C. J., and K. Matsuura. 2005. "Advantages of the Mean Absolute Error (Mae) over the Root Mean Square Error (Rmse) in Assessing Average Model Performance." *Climate Research* 30 (1): 79.

Programs and Code

Chawla, N. V., K. W. Bowyer, L. O. Hall, and W. P. Kegelmeyer. 2002. “SMOTE: Synthetic Minority over-Sampling Technique [Computer Software].” *Journal of Artificial Intelligence Research* 16: 321–57.

Chen, T., and C. Guestrin. 2016. “XGBoost: A Scalable Tree Boosting System [Computer Software].” In *Proceedings of the 22Nd Acm Sigkdd International Conference on Knowledge Discovery and Data Mining*, 785–94. New York, NY: ACM.

Chollet, F. 2013. “Keras Temporal Convolutional Network [Computer Software].” Retrieved October 22, 2019, from https://github.com/philipperemy/keras–tcn.

Dimitriadou, E., K. Hornik, F. Leisch, D. Meyer, and A. Maintainer. 2006. “The E1071 Package [Computer Software].” Retrieved October 3, 2019, from https://CRAN.R–project.org/package=e1071.

Kuhn, M. 2008. “Building Predictive Models in R Using the Caret Package [Computer Software].” *Journal of Statistical Software, Articles* 28 (5): 1–26.

Liaw, A., and M. Wiener. 2002. “Classification and Regression by randomForest [Computer Software].” *R News* 2 (3): 18–22.

Miron, B. K., and W. R. Rudnicki. 2012. “Feature Selection with the Boruta Package [Computer Software].” *Journal of Statistical Software* 36 (11): 1–13.

R Core Team. 2013. “R: A Language and Environment for Statistical Computing [Computing Language].” *R Foundation for Statistical Computing,* Vienna, Austria. URL http://www.R-project.org/.

Appendix

Table 18: Data Summary of the Used Features

	Mean	SD	Min	Q1	Median	Q3	Max
worker_rating	3.53	0.91	1.00	3.00	4.00	4.00	4.0
avg_rating	3.79	0.22	1.00	3.70	3.87	3.95	4.0
applicants	30.12	38.79	0.00	8.00	17.00	37.00	448.0
is_favorite	0.06	0.23	0.00	0.00	0.00	0.00	1.0
is_former	0.10	0.30	0.00	0.00	0.00	0.00	1.0
job_duration	23.66	71.74	1.00	5.00	8.00	20.00	3888.0
salary	629.54	1998.06	0.00	130.00	216.00	504.00	121577.8
distance_to_job	28.96	59.72	0.00	8.62	21.89	37.53	16387.1
language_skill_requ	0.07	0.25	0.00	0.00	0.00	0.00	1.0
driving_skill_requ	0.00	0.06	0.00	0.00	0.00	0.00	1.0
labels_defined	0.00	0.05	0.00	0.00	0.00	0.00	1.0
uniform	21.23	130.15	0.00	0.00	0.00	0.00	1636.0
job_name_len	36.48	19.58	2.00	22.00	34.00	48.00	177.0
job_requ_len	306.87	314.92	0.00	77.00	187.00	451.00	2019.0
clothing_requrements_len	134.56	142.26	0.00	42.00	78.00	164.00	609.0
meeting_point_len	33.39	30.05	0.00	12.00	25.00	49.00	287.0
add_skills_requ	0.79	1.01	0.00	0.00	1.00	1.00	9.0
days_since_ll	17.75	35.67	1.00	1.00	3.00	14.00	798.0
validation_set	0.64	0.48	0.00	0.00	1.00	1.00	1.0
use_public_wrks	0.98	0.15	0.00	1.00	1.00	1.00	1.0
use_former_wrks	0.40	0.49	0.00	0.00	0.00	1.00	1.0
use_favorite_wrks	0.87	0.33	0.00	1.00	1.00	1.00	1.0
use_own_recruited_wrks	0.11	0.31	0.00	0.00	0.00	0.00	1.0
use_dedicated_wrks	0.07	0.25	0.00	0.00	0.00	0.00	1.0
use_explicit_wrks	0.10	0.29	0.00	0.00	0.00	0.00	1.0
worker_age	30.66	10.29	18.00	23.00	27.00	36.00	102.0
months_on_platform	28.39	21.89	2.00	10.00	22.00	41.00	103.0
description_len	125.79	281.34	0.00	0.00	0.00	124.00	4768.0
is_domestic	0.79	0.41	0.00	1.00	1.00	1.00	1.0
average_rating	2.37	1.80	0.00	0.00	3.50	3.89	4.0
has_picture	0.83	0.37	0.00	1.00	1.00	1.00	1.0
photos	1.29	1.00	0.00	1.00	1.00	2.00	7.0

count_cv	0.93	0.40	0.00	1.00	1.00	1.00	12.0
count_certificate	1.32	2.00	0.00	0.00	1.00	2.00	24.0
count_diploma	0.05	0.29	0.00	0.00	0.00	0.00	7.0
count_testimonial	0.90	1.67	0.00	0.00	0.00	1.00	31.0
count_drivinglicens	0.07	0.34	0.00	0.00	0.00	0.00	7.0
count_rtw	1.63	1.14	0.00	1.00	1.00	2.00	20.0
was_referred	0.22	0.41	0.00	0.00	0.00	0.00	1.0
receive_newsletter	0.64	0.48	0.00	0.00	1.00	1.00	1.0
driving_skills	0.88	1.39	0.00	0.00	1.00	1.00	18.0
speaking_level_a	0.41	0.68	0.00	0.00	0.00	1.00	6.0
speaking_level_b	0.66	0.80	0.00	0.00	0.00	1.00	6.0
speaking_level_c	0.43	0.67	0.00	0.00	0.00	1.00	5.0
speaking_level_l	0.97	0.76	0.00	0.00	1.00	1.00	11.0
writing_level_a	0.48	0.74	0.00	0.00	0.00	1.00	6.0
writing_level_b	0.66	0.81	0.00	0.00	0.00	1.00	6.0
writing_level_c	0.38	0.64	0.00	0.00	0.00	1.00	5.0
writing_level_l	0.91	0.72	0.00	0.00	1.00	1.00	11.0
mean_reactiontime	25.16	53.17	0.02	8.84	12.00	25.34	4244.0
mean_salary	773.63	2206.31	0.00	166.28	321.21	662.73	139747.5
mean_jobduration	17.92	46.51	0.13	6.84	8.35	15.59	3105.5
mean_wage	31.09	64.07	19.93	24.97	25.76	26.91	10500.0
min_wage	24.29	20.75	16.00	23.09	23.92	25.00	10500.0
gender_Female	0.57	0.50	0.00	0.00	1.00	1.00	1.0
worker_language_DE	0.86	0.35	0.00	1.00	1.00	1.00	1.0
worker_language_FR	0.07	0.26	0.00	0.00	0.00	0.00	1.0
platform_id	1.00	0.00	1.00	1.00	1.00	1.00	1.0
com_com_dtfj	55.94	237.43	-1259.00	4.00	21.00	56.00	2785.0
com_size	2.21	0.85	1.00	1.00	2.00	3.00	3.0
com_was_referred	0.61	0.49	0.00	0.00	1.00	1.00	1.0
com_is_mothercompany	0.42	0.49	0.00	0.00	0.00	1.00	1.0
com_favorite_wrks	126.09	183.47	0.00	18.00	57.00	151.00	884.0
com_former_wrks	474.05	937.83	0.00	60.00	154.00	488.00	5883.0
com_recruited_wrks	48.43	137.93	0.00	0.00	0.00	6.00	868.0
com_employers	31.92	120.34	1.00	3.00	6.00	16.00	800.0
com_root_employers	144.72	276.66	1.00	6.00	30.00	90.00	993.0
com_ind_Hospitality	0.51	0.50	0.00	0.00	1.00	1.00	1.0
com_ind_NA	0.00	0.03	0.00	0.00	0.00	0.00	1.0

com_ind_EventPromotion	0.26	0.44	0.00	0.00	0.00	1.00	1.0
com_ind_RetailLogistics	0.20	0.40	0.00	0.00	0.00	0.00	1.0
com_ind_Airports	0.02	0.13	0.00	0.00	0.00	0.00	1.0
com_ind_All	0.00	0.04	0.00	0.00	0.00	0.00	1.0
com_ind_Healthcare	0.00	0.01	0.00	0.00	0.00	0.00	1.0
E6_score	0.34	0.21	0.04	0.15	0.33	0.48	1.0
E31_score	0.40	0.18	0.00	0.29	0.38	0.50	1.0
E85_score	0.34	0.20	0.04	0.15	0.35	0.46	1.0
E91_score	1.00	0.02	0.00	1.00	1.00	1.00	1.0
E191_score	0.51	0.28	0.00	0.33	0.33	0.67	1.0
I5_score	0.35	0.15	0.00	0.23	0.33	0.46	1.0
I10_score	0.36	0.22	0.00	0.22	0.33	0.44	1.0

Unequal Importance of FN and FP

This section presents two cases, in which FN and FP are of inequal importance. In medical cancer prediction, FN is weighted more severely versus to FP as the former indicates no cancer when there actually is, while the latter indicates cancer when there is no cancer (Hajian-Tilaki, 2013). Another example arises from the blackstone ratio for judicial errors in criminal law. The idea is that it’s more severe to consider one innocent person as guilty (FP) than to consider criminals as not guilty (FN) (Keijser, de Lange, & van Wilsem, 2014). This results in the opposite importance of FP and FN compared to the medical treatment example.

Model	TP rate	TN rate	Precision	Accuracy	Bal. Acc.	ROCAUC	PRAUC
FFNNET no rebal.	0.9026	0.4634	0.2895	0.5490	0.6830	0.7251	0.3345
FFNNET over	0.2410	0.5776	0.1214	0.5120	0.4093	0.6040	0.1561
FFNNET under	0.6256	0.6037	0.2766	0.6080	0.6147	0.6270	0.2569
FFNNET smote	0.8359	0.3540	0.2387	0.4480	0.5950	0.6032	0.2381

Table 19: FFNNET Rebalancing Results

Model	TP rate	TN rate	Precision	Accuracy	Bal. Acc.	ROCAUC	PRAUC
KNN no rebal.	0.7287	0.6447	0.3220	0.6605	0.6447	0.7390	0.3711
KNN over	0.7580	0.5911	0.3003	0.6225	0.5911	0.7462	0.3904
KNN under	0.6676	0.7303	0.3643	0.7185	0.7303	0.7666	0.4283
KNN smote	0.7263	0.7979	0.7687	0.7635	0.7979	0.8312	0.8051

Table 20: KNN Rebalancing Results

Model	TP rate	TN rate	Precision	Accuracy	Bal. Acc.	ROCAUC	PRAUC
LOG none	0.7135	0.7252	0.3805	0.7230	0.7194	0.7822	0.4655
LOG over	0.7104	0.7229	0.3775	0.7205	0.7166	0.7817	0.4555
LOG under	0.7303	0.6990	0.3647	0.7050	0.7146	0.7776	0.4509
LOG smote	0.7543	0.6619	0.3455	0.6796	0.7081	0.7716	0.4303

Table 21: LOG Rebalancing Results

Model	TP rate	TN rate	Precision	Accuracy	Bal. Acc.	ROCAUC	PRAUC
RF no rebal.	0.8177	0.5861	0.3039	0.6280	0.7019	0.7536	0.3671
RF over	0.7624	0.5775	0.2851	0.6110	0.6700	0.7215	0.3515
RF under	0.7459	0.6190	0.3020	0.6420	0.6825	0.7274	0.3512
RF smote	0.7735	0.5617	0.2806	0.6000	0.6676	0.7163	0.3533

Table 22: RF Rebalancing Results

Model	TP rate	TN rate	Precision	Accuracy	Bal. Acc.	ROCAUC	PRAUC
SVM no rebal.	0.8177	0.5861	0.3039	0.6280	0.7019	0.7536	0.3671
SVM over	0.7624	0.5775	0.2851	0.6110	0.6700	0.7215	0.3515
SVM under	0.7459	0.6190	0.3020	0.6420	0.6825	0.7274	0.3512
SVM smote	0.7735	0.5617	0.2806	0.6000	0.6676	0.7163	0.3533

Table 23: SVM Rebalancing Results

Statutory Declaration

I hereby declare that the thesis with title

Assessing the Performance of Machine Learning for a Job Matching Algorithm

A Study of Optimal Skill Allocation on an Online Gig Work Platform

has been composed by myself autonomously and that no means other than those declared were used. In every single case, I have marked parts that were taken out of published or unpublished work, either verbatim or in a paraphrased manner, as such through a quotation. This thesis has not been handed in or published before in the same or similar form.

Zurich, November 8th 2019

Kwan Tsit Richard Chan