Bachelorarbeit

Non-parametric Machine Learning regression under misspecification

Paul Jarschke
Steinfurter Straße 79, 48149 Münster
Matrikelnummer: 461489

21.09.2021

Gutachter: Prof. Dr. Mark Trede

Institut für Ökonometrie und Wirtschaftsstatistik

Westfälische Wilhelms-Universität Münster

Abstract

This thesis deals with determining suitable regression models depending on different metrics to maximise the predictive performance. It therefore investigates how the relative predictive performance of different models varies for changes in the standard deviation of the error term, the number of samples within the provided data and also the degree of misspecification. It focuses on comparing the predictive performance of a parametric regression model with a misspecified functional form to the performance of non-parametric regression machine learning methods. Therefore, two economic production functions are used. First, the different methods of the regression models will be discussed and then used to make predictions on simulated data. Afterwards, it will be investigated how different parameters affect the relative model performance to determine when to choose a parametric model or a non-parametric model. The results of the analysis indicate that certain properties of the data suggest a distinct model choice.

Contents

List of Tables

List of Figures

List of Algorithms

1. Introduction

Machine Learning is a term that describes methods that try to model patterns within different kinds of data and can be divided into three different types of learning, supervised learning being one of them. Supervised learning is exercised, when some given set of input variables is supposed to have an effect on a given output variable. In this thesis, the input variables will be called exogenous variables, whereas the the output variable will be labelled as the endogenous variable. If the endogenous variable is quantitative and not qualitative a regression problem is given. This thesis will only discuss regression methods within machine learning. Linear regression is probably one of the most classical methods for predicting a quantitative variable and has been around for a long time. When linear regression is used to make predictions for a certain relationship between variables, parameters of a previously specified linear function are estimated and optimised to find a function that fits a given data set as well as possible. The method is still one of the most prominent and widely used regression techniques. With the upcoming computing power and new algorithms in the last few decades, a wide variety of different regression methods arose creating an issue of choice. When should which kind of regression model be chosen? When linear regression is performed, a mathematical function has to be specified and a linear relationship between the endogenous and the exogenous variables is assumed. This assumption is quite heavy and does not necessarily need to be correct, resulting in a misspecified regression model that is likely to be biased. As an alternative, there are different algorithmic approaches to regression that do not specify a certain relationship and work without the estimation of parameters. This thesis deals with answering the question, whether to choose a non-parametric model over a parametric regression model that might be misspecified. Chapter 2 and 3 introduce how linear regression works and how non-parametric models like KNN Regression and Random Forest Regression produce predictions and how they can be improved. Afterwards, three economic production functions will be discussed and later used to generate the data for a simulation experiment. In the end, the results of the experiment will be analysed to draw conclusions on when to choose which model under certain circumstances.

2. Parametric Regression

Parametric regression models assume that a relationship between exogenous and endogenous variables is given by some kind of mathematical function. The parameters of the function can then be estimated and optimised to create the regression model. According to Hastie et al. (2009) the goal of parametric regression is to find an appropriate approximation $f(\hat{x})$ to the function $f(x)$, which is the underlying function of the relationship between the input variables and the output variable. The data is assumed to originate from a statistical model with the functional form

$$Y = f(X) + \varepsilon, \tag{2.1}$$

where ε denotes a random error. The random error has the property $E(\varepsilon) = 0$ and is independent of the exogenous variables X. A model that uses an additive error can be interpreted as an appropriate approximation of the reality. Since most systems do not have a deterministic relationship, that can be expressed as $Y = f(X)$, where Y denotes the endogenous variable. Generally there will be some kind of unmeasured variables or measurement errors, that have an effect on Y. This effect can be captured with the additive error ε. Based on Hastie et al. (2009), linear regression and the ordinary least square method will be discussed in the following chapter.

2.1. Linear Regression and Ordinary Least Squares

As mentioned earlier, an input vector $X^T = (X_1, X_2, \cdots, X_p)$ with the exogenous variables is used to make predictions for the endogenous variable Y.
The linear regression model has the form

$$f(X) = \beta_0 + \sum_{j=1}^{p} X_j \beta_j. \tag{2.2}$$

The assumption that the regression function $E(Y|X = x)$ is linear, or that a linear approximation is valid, has to be made when using the linear model for regression. The number of exogenous variables is denoted as p. The parameter β_0 denotes the unknown intercept, β_j the unknown coefficients and X_j the exogenous variables, which can be different in their kind. They can be of quantitative nature, polynomials, for example $X_2 = X_1^2, X_3 = X_1^3$, dummy variables, or even interactions between variables. However,

the model is linear in its parameters. Usually a training data set $(x_1, y_1), \ldots, (x_N, y_N)$ is given and used to estimate the unknown parameters β. Every $x_i = (x_{i1}, x_{i2}, \cdots, x_{iN})^T$ represents a vector of feature measurements for each case. To estimate the unknown parameters β, the *ordinary least squares method* (OLS) will be used. The OLS method chooses the vector of coefficients $\beta = (\beta_0, \beta_1, \cdots, \beta_p)^T$, by minimising the residual sum of squares (RSS) with respect to β. The RSS is defined as

$$RSS(\beta) = \sum_{i=1}^{N} (y_i - f(x_i))^2 \tag{2.3}$$

$$= \sum_{i=1}^{N} (y_i - \beta_0 + \sum_{j=1}^{p} x_{ij}\beta_j)^2, \tag{2.4}$$

where N denotes the number of samples within the given data. It is important to comment, that equation (3.4) does not make any assumption about the validity of the linear model. It only acquires the best possible linear fit to the data it is trained with. To correctly minimise the residual sum of squares, a slightly different notation is required. The variable $\boldsymbol{X}$ is initiated and represents a $N \times (p+1)$ matrix, where each row is an input vector and the first row of the matrix is filled with the number one, to obtain the intercept. In addition to $\boldsymbol{X}$, the variable $\boldsymbol{y}$ is initiated. It resembles the N-vector of the endogenous variable. With these two new variables, the residual sum-of-squares is expressed as a quadratic function with $p+1$ parameters and can be written as follows:

$$RSS(\beta) = (\boldsymbol{y} - \boldsymbol{X}\beta)^T(\boldsymbol{y} - \boldsymbol{X}\beta) \tag{2.5}$$

Now the function can be differentiated with respect to β to obtain the following equations.

$$\frac{\partial RSS}{\partial \beta} = -2\boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\beta) \tag{2.6}$$

$$\frac{\partial^2 RSS}{\partial \beta \partial \beta^T} = -2\boldsymbol{X}^T\boldsymbol{X} \tag{2.7}$$

To continue with the derivation, the assumption that $\boldsymbol{X}$ has full column rank and therefore $\boldsymbol{X}^T\boldsymbol{X}$ is positive definite has to be made. Now the first derivative will be set to 0.

$$0 = \boldsymbol{X}^T(\boldsymbol{y} - \boldsymbol{X}\beta) \tag{2.8}$$

As a solution of the equation (4.10), the vector of estimated coefficients is given as

$$\hat{\beta} = (\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{2.9}$$

and the fitted values, or predicted values of the training data can be written as

$$\hat{\boldsymbol{y}} = \boldsymbol{X}\hat{\beta} = \boldsymbol{X}(\boldsymbol{X}^T\boldsymbol{X})^{-1}\boldsymbol{X}^T\boldsymbol{y} \tag{2.10}$$

With this formula, linear regression is able to make predictions for any quantitative set of data, given the exogenous variables X and the the measured outcome Y. A new prediction for any input vector x_i can be computed with the following formula:

$$\hat{f}(x_i) = (1 : x_i)^T\hat{\beta} \tag{2.11}$$

3. Non-parametric Regression

In comparison to the parametric regression methods, such as linear regression with OLS, non-parametric regression methods do not specify a functional form that can be expressed as a continuous function with estimable parameters. Non-parametric methods make predictions for certain groups of samples or areas of a spaces. In the following, two different estimation methods and their methodology will be presented: K-Nearest-Neighbour (KNN) Regression and Random Forest Regression. The following chapter about KNN Regression is inspired by the work of Kuhn and Johnson (2013)

3.1. KNN Regression

In comparison to the parametric regression, the KNN approach cannot be summarised in form of a mathematical function unlike models that arise from linear OLS estimation. According to Kuhn and Johnson (2013),like linear models, KNN regression models are only constructed with individual values of the exogenous variables and the endogenous variable that are provided by a data set, which is exclusively designated for training purposes. If a regression analysis is done with the KNN approach, an algorithm can determine a value of the endogenous variable, using the K-closest samples, also called neighbours, that are given by the training data set. For K being the total number of neighbours and n being the number of samples in the training data set, the following conditions have to be fulfilled simultaneously: $K \in \mathbb{N}^+$ and $K \leq n$. K has to be an

element of all natural numbers, excluding all negative numbers, decimal numbers and the whole number 0, because computing regression estimations using a KNN model is not possible for negative numbers and decimals and yields no prediction for 0 neighbours. When using a machine learning library to code a KNN regression model, for example scikit-learn for Python, the regressor class specifies K to be an integer. On the other hand, the number of neighbours K must be equal or smaller than the total of samples provided by the training data set. According to Hastie et al. (2009) the computation of a prediction of the value of the endogenous variable $\hat{Y}$ in the training set is defined as follows:

$$\hat{Y}(x) = \frac{1}{k} \sum_{x_i \in N_k(x)} y_i. \tag{3.1}$$

$N_k(x)$ denotes the neighbourhood of x, which is defined by the k-closest points x_i, that are given in the training data. To express the prediction verbally, one can say that first the k observations with x_i closest to x in input space are found and then the value of the endogenous variables are averaged to predict a value. According to Kuhn and Johnson (2013) the mean is the most used metric for KNN regression, but other metrics, for example the median or the with distance weighted mean, can also be used to make predictions.

Whatever metric is used to calculate the prediction, the measurement of distance chosen by the user has a significant effect on the predicted results. The most commonly used measurements for the distance between two samples are the Euclidean distance, also know as straight-line distance, and the Manhattan distance, also known as the city-block distance. They are defined as follows:

$$Euclidean\ distance : \left(\sum_{j=1}^{p} (x_{aj} - x_{bj}) \right)^{1/2} \tag{3.2}$$

$$Manhattan\ distance : \quad \sum_{j=1}^{p} |x_{aj} - x_{bj}| \tag{3.3}$$

The variables x_a and x_b denote the two samples, between which the distance should be quantified. The index j describes the dimension of the sample, with p being the total number of exogenous variables within the given data set. There exists a generalisation for both of these measurements of distance, which is called the Minkowski distance.

For $q > 0$ the Minkowski distance is defined as follows:

$$Minkowski\ distance: \left(\sum_{j=1}^{p} |x_{aj} - x_{bj}|^q \right)^{1/q} \tag{3.4}$$

Inserting $q = 2$ into the Minkowski distance returns the Euclidean distance and for $q = 1$, the Minkowski distance is equal to the Manhattan distance. There are many other metrics for measuring distance, like Tanimoto and Hamming distance, that are used in different scientific cases to make predictions in a specific context. The Manhattan distance for example, is often used in a context where the sampled data includes binary variables.

Since the KNN approach to regression radically depends on the measured distance between samples, the scale of parameters can have a significant effect on the calculated distances between samples, according to Bekkerman et al. (2011). If the scales of the variables within the data differ tremendously, measured distances between samples will be biased towards the variables with the largest scales, since they will contribute the most to the value of distance. One possibility for avoiding this probable bias is feature scaling, whose methods and benefits will be discussed in the following sub chapter, based on Abbott (2014).

3.1.1. Feature Scaling

According to Abbott (2014), there are certain issues with unprocessed data that can lead to biased model predictions. Next to variables with skewed distributions, one of the most prominent issues resulting in biased models is unscaled data. When the variables that are used to train a predictive model have different magnitudes, the model is likely to be biased, especially if it is based on a distance-based algorithm such as KNN. The magnitude can have a significant effect on algorithms like KNN, since variables with a larger scale will result in larger distances and create a bias within the model. Scaling every variable that is used for making predictions removes the bias. When scaling a variable, a function is consistently applied to every value of the unscaled variable and as a result of the application, the variable is returned in a scaled manner.

In Table 1, a series of the most common scaling methods is displayed, with the most

commonly used scaling methods being the min-max normalisation method and the z-score standardisation method. Using the magnitude scaling method, the values of the variable are scaled by its highest absolute value. As a result of the magnitude scaling method, all scaled values are in a range between -1 and 1, with one of them being the maximum value of the variable. The entire range of values does not necessarily need to be used, but can be, depending on the maximum and minimum value of the variable. Scaling can also be done with the sigmoid method, which is more popular in the context of deep learning with neural networks and logistic regression since it scales the variable into a range between 0 and 1. The parameter c can be adjusted, to achieve a desired sigmoid shape that is suited for a specific variable. When a variable is scaled with the min-max normalisation method, the scale of the variable can be abbreviated or expanded, since normalisation leads to a variable scale between 0 and 1. Whether the variable is up-scaled or down-scaled depends on the original scale of the variable. Z-score standardisation is the most suitable scaling method if the variable is normally distributed, since this scaling method assumes normally distributed variables. Standardisation is very convenient for interpretation, because each unit refers to one unit of standard deviation from the mean. The different scaling methods are presented in the following table:

Scaling Method	**Formula**	**Range of scaled values**
Magnitude scaling	$x' = \frac{x}{\lvert x \rvert_{max}}$	[-1,1]
Sigmoid scaling	$x' = \frac{1}{\left(1 + e\left(\frac{-x}{c}\right)\right)}$	[0,1]
Min-max normalisation	$x' = \frac{x - x_{min}}{x_{max} - x_{min}}$	[0,1]
Z-score standardisation	$x' = \frac{x - \mu_x}{\sigma_x} = Z$	mostly [–3,3]

Table 1: Scaling Methods (Abbott, 2014)

3.2. Regression Tree Models

Kuhn and Johnson (2013) described Regression Tree models in "Applied Predictive Modeling". The contents of the following chapter are based on their literature. Models that make predictions with decision trees use multiple interlaced if-then-statements for the predictors, that divide the data into different partitions. The model then makes a prediction for each of the created partitions. A simple example for a decision tree with two predictors could look like this: The example portrays a three-dimensional space

Algorithm 1: Decision Tree Model (Kuhn and Johnson, 2013)

```
if Predictor A ≥ 1 then
    if Predictor B ≥ 1 then
        prediction = 10
    else
        prediction = 5
    end
else
    prediction = 0
end
```

with two predictors, that separates the whole prediction space into three sub-spaces. For every sub-space, a number is predicted. When talking about decision tree based model, one would say that there are two splits into three terminal nodes. They are also called leaves of the tree. The splits are cause by the if-statements and the leaves show all possible predictions that the tree can produce. A new prediction for any value can be made by starting at the first if-then-statement and then following the tree, until a terminal node is reached. Instead of a numerical value, a terminal node can also contain a more complex function that will be used to calculate a prediction.

Recognise that for any sample a unique route to a terminal node is defined by the if-then statements of the tree. A rule is defined as a collection of conditions that are concentrated as independent conditions. The rules for the tree would look like this: Tree-based models are a tool of high popularity for different reasons. Since the predic-

Algorithm 2: Rules of a Decision Tree (Kuhn and Johnson, 2013)

if *Predictor A* $\geq$ *1 and Predictor B* $\geq$ *1* **then**
| prediction = 10

end

if *Predictor A* $\geq$ *1 and Predictor B* $<$ *1* **then**
| prediction = 5

end

if *Predictor A* $<$ *1* **then**
| prediction = 10

end

tions are based on a set of conditions, tree-based models are rather easy to interpret and implement. These models can also be used to predict quantitative and qualitative output. Preprocessing the data, e.g. scaling variables, is not necessary for tree-based models, which is an advantage compared to KNN-models. A predefined relationship between exogenous and endogenous variables, that for example is needed when performing linear regression, is not necessary. Tree-based models can even handle missing data and feature selection issues and are very handy for modelling problems that occur when using real-life data. However, if the model is based on a tree or set of rules, it also has certain weaknesses. The most prominent weaknesses of single-tree-models are model instability and sub optimal predictive performance. Small changes within the given data can heavily change the structure of the tree and the rules it is based on, which can result in big changes in terms of interpretation. The sub optimal predictive power is due to the fact, that tree-based models predict the same outcome for rectangular sub spaces of the whole prediction space. If the real relationship between exogenous and endogenous variables cannot be modelled with such rectangular sub spaces, the tree-based model will have a larger prediction error in comparison to a model, that specifies the underlying relationship correctly. Researchers developed ensemble techniques that reduce these issues. Ensemble methods combine many trees

into one predictive model and tend to have better predictive performance than single trees. In the following chapters, regression trees and the ensemble method bagging will be discussed.

3.2.1. Basic Regression Trees

Basic regression trees divide the data into smaller groups. This division is performed in such a manner that the divided groups are homogeneous with respect to the endogenous variables. Therefore, the regression tree first has to determine the predictor to split on and the value of the split. Then the depth and complexity of the tree have to be set, as well as the prediction equation in the terminal node, if desired.

There are different approaches to constructing a decision tree. One of the most used techniques the classification and regression tree (CART) method that was implemented by Breiman et al.(1984) If the output variable is of quantitative nature, which is the case for regression, the model starts with the whole data set S. The model searches for a values of a predictor, that splits S into two subsets, S_1 and S_2, in a way that the the sums of squares error is minimal. The sums of squares error is defined as follows:

$$SSE = \sum_{i \in S_1} (y_i - \overline{y}_1)^2 + \sum_{i \in S_2} (y_i - \overline{y}_2)^2. \tag{3.5}$$

The averages of the values of the endogenous variable in the groups S_1 and S_2 are defined as $\overline{y}_1$ and $\overline{y}_2$ respectively. Within the newly partitioned groups S_1 and S_2 the CART technique seeks the predictor and split value that minimises the SSE. This method is also called recursive partitioning, due to the recursive splitting nature of regression trees. When applying the recursive partitioning to data, the process of splitting within the sets S_1 and S_2 is not stopped until the number of samples within the created splits falls under a predefined value. This could for example be five samples per node. When the threshold is undercut, the tree stops growing and a terminal node is reached. If the used predictors are continuous variables, the data can be ordered and finding the optimal splitting point is unambiguous and binary variables are also rather easy to split. However, finding an optimal split point when a predictor has more than two categories has multiple justifiable processes but is not relevant for this thesis and the simulation study that will be presented in later chapters.

3.2.2. Bagged Trees

Ensemble techniques are methods that combine the predictions of multiple models. They began to appear in the 1990s, when Leo Breiman (Breiman 1996) was one of the first researchers that proposed the bagging, short for bootstrap aggregation, technique. According to Kuhn and Johnson (2013), Bootstrap sampling is a sampling technique that takes random samples of the given data set with replacement. Therefore, a data point can be selected multiple times for a subset and is still available for selection after being drawn and added to the sub sample. Within a bootstrap sample, it is possible that some samples are drawn multiple times and some samples of the training set are not drawn at all for the bootstrap sample. The samples that are not part of the bootstrap subset are called out of bag samples. A model is then built on the selected samples and used to predict the out of bag samples for a given iteration of bootstrap re-sampling. Bagging can therefore be described as an approach that uses bootstrapping in the context of regression to establish an ensemble. The method is rather intuitive and can be expressed as the following algorithm:

Algorithm 3: Bootstrap algorithm (Kuhn and Johnson, 2013)

for $i = 1$ *to* m **do**
  Generate a bootstrap sample of the original data
  Train a decision tree model on this sample
end

Each model in the ensemble is used for predicting a new sample of data. The prediction of the m trees are then averaged to calculate the prediction of the bagged model. Bagged models are likely to perform better than models that are not bagged, since they have several advantages. Bagging does not only reduce the variance of a prediction effectively over the aggregation process, but it also improves models with unstable predictions. Regression trees for example profit from aggregating over multiple versions of the training data, since it decreases the variance of the prediction and therefore results in more stable predictions.

3.3. Fitting and evaluating models

So far, the theory of different models and how their prediction mechanisms work have been presented, whereas this chapter is dedicated to the importance of data allocation and its usage for model evaluation, as explained by Kuhn and Johnson (2013). Within the next two sub chapters, the necessity of splitting data into different subsets for testing and validation and how hyper-parameters can be tuned will be discussed, based on Müller and Guido (2016).

3.3.1. Generalization, Overfitting, and Underfitting

The general idea of supervised learning, as described by Müller and Guido (2016), is to build a model based on training data, that is able to make accurate predictions on a new, unseen set of data with the same characteristics as the training set. If the predictive model performs well, by making accurate predictions on the unseen data, it is able to generalise the training set to the test set. The ultimate goal of supervised learning is to create a model that generalises as good as possible. The first instance of a model is usually built to make accurate predictions on the training set. If training and test set have common characteristics, the model is expected to have a very similar predictive power on both data sets. However, building a model to perfectly fit the training can backfire. This is especially the case for complex models like regression trees and random forests. The only measure of general predictive power for regression models and algorithms is the model evaluation on the test set. If for example a regression tree was constructed to make predictions on a training set, and the tree was constricted in a way, that every single value of the endogenous variable is represented in a terminal node, the models predictions would be 100% accurate for the training data, but would perform very poorly on unseen data. Building a model that is far to complex for the provided information is called overfitting. It occurs when a model is fit too closely to the particularities of the training set and is not able to generalise on new data but fits the training set perfectly. However, the effect can also go into the opposite direction. If a model is not complex enough, it might not be able to capture all the aspects of variability within the data. The model will then perform poorly on both, test and training set. This issue is called underfitting. To summarise, it can be said that prediction accuracy on the training set rises with the complexity of

the model, but too complex models, that focus too much on the individual points of the training set, will not generalise well and perform poorly with new data. The goal is to find the model complexity that maximises prediction accuracy of unseen data.

3.3.2. Measure of Performance

According to Kuhn and Johnson (2013), some sort of metric is needed to evaluate the quality of a regression model. To evaluate the quality of a regression model, some sort of metric is needed. Comparing this metric of different models then enables comparison and model selection. For regression, one of the most common metric is the root mean squared error (RMSE). The RMSE is just the root of the mean squared error (MSE), which is a function based on the difference between predicted values and observed values, also called residuals. The value of the RMSE shows how far the average distance between observed values and predictions is and how fare away the residuals are from 0 on average. It is defined as follows:

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2} \tag{3.6}$$

The mean absolute error (MAE) is also used to evaluate predictive performance and is just slightly different from the MSE, since it calculates the average of the absolute residuals. Another common metric that is especially used for linear regression models, is R^2. The R^2 value ranges between 0 and 1 and can be interpreted as the proportion of the variation within the data, which can be explained by the model. For linear regression models with multiple regressors the proper metric is the *adjusted* R^2. While the interpretation of R^2 is quite intuitive, one has to keep in mind, that R^2 is a measure of correlation and not accuracy. It is also important to mention, that models with the same RMSE can have different R^2 values, depending on the variance of the exogenous variable. Therefore the RMSE will be used as a primary evaluation metric in the later part of this thesis.

3.3.3. Model evaluation and tuning

In the previous chapter the necessity of splitting data into test and training sets and how the predictive performance of a supervised model can be quantified with different metrics was shown, based on Müller and Guido (2016). The procedure of Cross-Validation,

which is a more robust way to judge general performance and how hyperparameters of non-parametric regression models can be tuned and validated with the grid search method will be explained. In the following chapter the methodology of these methods will be shown, based on Müller and Guido (2016).

Cross-validation is another method in statistics that is used for evaluating generalisation performance of models in supervised learning. Cross-validation is a useful tool, since it is more stable and profound than evaluating models solely based on performance on test and training set. Within cross-validation the data is split multiple times and a model is then trained on each split set of data. There are a few different options for doing this, with k-fold cross-validation being one of the most used methods. Applying k-fold cross-validation to a set of data splits the data in to k subsets, which are called folds in this case. The parameter k is specified by the user. In most cases, the data is split into 3, 5, or 10 folds. When 3-fold cross validation is performed, the whole data is split into 3 folds that all have approximately the same size. Then a sequence of three models is trained on the different folds. The first model is for example trained on the second and third fold, while the first fold is used as a test set for the first model. The second model is tested on the second fold and trained with the first and third fold. This process is then repeated one last time for the third model. For each model, the accuracy on the test set is calculated and three differently evaluated models are returned. The mean of the 3 metrics can then be computed to tell, how good the predictions of the models are on average.

K-fold cross-validation has several benefits compared to randomly splitting the data into test and training set just once. It might be the case, that the data is randomly split into two very similar sets. Then the model is likely to generalise well on the test set. If the random splits are very different in the characteristics of the data, the trained model will perform poorly on the test set. Therefore, the predictive power of the model could be both over- and underestimated. This can be avoided with the use of cross-validation, since each sample will be used exactly once for testing each model. Every data sample is in one of the folds, and each fold is used as a test set once. Therefore, the model has to generalise well to all samples to achieve high predictive

performance. With cross-validation, the data can also be used more effectively. When data is just divided into two splits, the training set usually contains around 75% of all data and the rest is used as test set. This percentage can deviate, but it is common practice to have a training set that contains between 70% and 80% of all data. When k-fold cross-validation is used to validate a model, this relationship between test and training set size changes. For 5-fold cross validation, 80% of the data can be used for model training, for ten-fold cross validation even 90%. This will result in more accurate models since more data is provided. K-fold cross-validation also has a significant disadvantage of high computational costs. Since k models are trained instead of just one, the cross-validation will be k times slower than doing a single split of the data. This will be observable, when running the KNN and Random Forest model for the simulation.

After being able to evaluate a model that generalises well, the next step is to adjust the hyperparameters of the model to further improve the quality of prediction. Hyperparameters influence the prediction of a model and have to be chosen by the user. For KNN regression, these hyperparameters are for example the number of neighbours and distance, while a tree-based models parameters are for example the depth of a tree or the minimal number of samples per node. This can be a more complex but necessary task and can be done in different ways. One way to find optimal hyperparameters is called grid search. The way grid search finds the optimal parameters is by training a model for each combination of the hyperparameters of interest. If the model fitting process is completed with just the training set, overfitting is very likely to occur. This is where k-fold cross validation comes into play again. With k-fold cross validation k values to measure the performance of the model are computed for each combination of the indicated hyperparameters. Aftwerwards, the mean of the measurements is computed for each combination, which allows comparison of all combinations of hyperparamteres. The model with the highest score is chosen for making predictions. The whole process of splitting data, searching on the grid and evaluating parameters is shown in the following visualisation.

4. Production Functions in Macroeconomics

As described by Olson in "Essentials of Advanced Macroeconomics" (Olson 2013), macroeconomics can be described as the study of the aggregated economy of a country. Macroeconomics tries to understand how decisions of individuals and companies influence the aggregated economic outcome. Topics in macroeconomics include the changes of the gross domestic product (GDP), level of investment, inflation and many more. A common technique to model the output of an economy are production functions. Production functions model the relationship between the means of production and the the output, with the most prominent production function being the Cobb-Douglas production function. Their production function assumes two input factors, labour L and capital K, the technology shift parameter A, and the parameter α, indicating the percentage of all means of production that are classified as labour. In the following sub chapters, some chosen production functions will be discussed. But first, three different metrics that characterise a production function will be presented, to explain how production function can differ from each other and how certain assumptions specify their functional form.

The three main characteristics that describe a production function are the marginal product (MP), the returns to scale (RTS), and the elasticity of substitution (EOS). For explanatory purposes, the general form of a production function, as portrayed by Lipsey (2018), will be reduced to a production function with two input factors, labour and capital, that are defined as x_L and x_K respectively. A production function with the given restrictions would look like follows:

$$y = f(x_L, x_K). \tag{4.1}$$

The idea of the MP is rather simple and the metric can be achieved by differentiating the production function with respect to one input factor (Sickles and Zelenyuk 2019). The marginal product of labour and capital using the function from (6.1) can be expressed as

$$MP_L = \frac{\partial f(x_L, x_K)}{\partial x_L} \quad and \quad MP_K = \frac{\partial f(x_L, x_K)}{\partial x_K}\,. \tag{4.2}$$

The MP of and input factor can then be described as the effect a marginal increase of x_j would have on the output y.

The second presented characteristic of a production function are the RTS. According to Lipsey (2018) a distinction between three different kinds of RTS can be made: constant returns to scale, increasing returns to scale and decreasing returns to scale. RTS can be expressed in the following way:

$$f(\lambda x_L, \lambda x_K) = \alpha_\lambda f(x_L, x_K). \tag{4.3}$$

When $\alpha_\lambda = \lambda$, then constant RTS are indicated. Constant returns to scale are therefore given, when an increase or decrease of only one input factor by one marginal unit results in a proportional change of the output variable. The effect of the change of the input factor is the same for every value of x_j and the relationship between one input factor and the output can be modelled with a linear function. The production function is then called homogeneous. If $\alpha_\lambda > \lambda$, then a case of increasing RTS is portrayed. Increasing returns to scale are therefore given when an increase of only one input factor by one marginal unit results in a more than proportionate increase of the output variable. The effect of the change of the input factor increases with x_j and is especially high for high levels of x_j. The relationship between the input factor x_j and the output y_i can be modelled with a convex function. When $\alpha_\lambda < \lambda$, then decreasing RTS are indicated. Decreasing returns to scale are therefore given, when an increase of only one input factor by one marginal unit results in a less than proportionate increase of the output variable. The effect of the change of the input factor increases with x_j and is especially low for high levels of x_j. The relationship between the input factor x_j and the output y_i can be modelled with a concave function. Every one of the three different types of RTS can be argued for in different situations or for different factors of input, but to construct a production function, an assumption has to be made with care. An assumption about the form of RTS is a very strong assumption, which has fundamental influence on the mathematical expression of the relationship between input factors and output.

Another metric that characterises a production function is the elasticity of substitution. Sickles and Zelenyuk (2019) define the elasticity of substitution σ, as the percentage

change in factor proportions due to a change in the marginal rate of substitution (MRS). The idea behind this concept is, to find outat which rate the input factors, or means of production, can be interchanged to obtain the same values of output. Functions that express this rate are called isoquants. The elasticity of substitution can then be interpreted as the curvature of the isoquant. Sickles and Zelenyuk (2019) establish the metric for a simple production function with two inputs and one output $y = F(L, K)$, where the variable L denotes the factor labour and K denotes the second input variable capital. The first order derivatives, also called marginal product in the context of production functions, are defined as f_L and f_K respectively. With these definitions, the mathematical form of the elasticity of substitution is defined as:

$$\sigma_{LK} = \frac{d\ ln(L/K)}{d\ ln(f_K/f_L)} \tag{4.4}$$

$$= \frac{d\ ln(L/K)}{d\ ln(f_K/f_L)} \times \frac{f_K/f_L}{L/K} \tag{4.5}$$

Since different ecconomists defend different types of the three charactersitsics, there is a broad variety of production functions available, that try to model the relationship of output and input factors. The two most prominent production functions are the Cobb-Douglas function and the constant elasticity of substitution (CES) function. Both of them as well a special case of the CES function will be discussed in the following sub chapters.

4.1. Cobb-Douglas production function

With "A Theory of Production", Charles Cobb and Paul Douglas were one of the first economists that tried to express the relationship between inputs and output for the US manufacturing industry (Cobb and Douglas, 1928). Using the parameters from the previous chapter, the Cobb-Douglas production function is defined as

$$Y = AL^{\alpha}K^{1-\alpha}. \tag{4.6}$$

According to Sickles and Zelenyuk (2019), the function relates output levels to a linear combination of exponentiated levels of the inputs. They defined a general notation for the Cobb-Douglas function and its characteristics. The notation is defined as

$$y = \beta_0 \prod_{j=1}^{N} x_j^{\beta_j}. \tag{4.7}$$

Therefore the marginal products are given by

$$MP_j = \beta_0 \beta_j \; x_J^{-1} \prod_{k=1}^{\beta_k} x_k^{\beta_k}, \; j = 1, \cdots, N \tag{4.8}$$

with the following RTS:

$$RTS = \sum_{k=1}^{N} \beta_k. \tag{4.9}$$

The elasticity of substitution is given by

$$\sigma_{jk} = \frac{\beta_j}{\beta_k} \frac{\beta_k}{\beta_j} = 1. \tag{4.10}$$

As shown by Sickles and Zelenyuk (2019), the Cobb-Douglas production function has an elasticity of substitution that is unitary for any possible set of MPs and combinations of input factors. A useful property of the Cobb-Douglas, is that it can be transformed into log-log form and can be expressed as follows:

$$\ln y = \ln \beta_0 + \sum_{j=1}^{N} \beta_j \ln x_j \tag{4.11}$$

This notation is quite convenient, because it can be used for linear regression and allows the identification of output elasticities and scale economies based on the parameter estimates. Although the Cobb-Douglas function is quite handy for estimation, it comes with the limitation of the constant unitary elasticity of substitution, that is a result of the specification of the functional form. This assumption is quite heavy. Even though the assumption made by Cobb and Douglas seemed consistent for the factor shares of the US manufacturing industry, this assumption might not be valid for other empirical settings. Solow (1956) and Arrow et al. (1961) used this assumption as the motivation for the CES function, which will be discussed in the next chapter.

4.2. CES production function

To address the issue of the heavy assumptions made by Cobb and Douglas, Solow (1956) and Arrow et al. (1961) introduced another function as general alternative, called CES function. According to Sickles and Zelenyuk (2019), the main difference between the Cobb-Douglas function and the CES function, is the non-unitary elasticity of substitution. Therefore the CES function allows factor shares to have constant

changes, and not only levels when the marginal rates of technical substitution change. The general functional form of the CES function, defined by Blackorby and Russell (1981,1989) can be written as:

$$y = \beta_0 \left[\sum_{j=1}^{N} \beta_j x_j^{\rho} \right]^{1/\rho} \tag{4.12}$$

where β_0, $\beta_j > 0$ and $\rho \leq 1$. When the value of the variable ρ is not limited or set to a definite value, the elasticity of substitution is given by

$$\sigma = 1/(1-\rho). \tag{4.13}$$

and for a two-factor case resembles the percentage change in the input ratio when a percentage change in relative factor prices is given. According to Hicks (1932) and Hicks and Allen (1934), the CES function has constant return to scale and is highly flexible, nesting in different functions for different values of ρ. When $\rho \to 0$, then the CES function nests in the Cobb-Douglas function and for $\rho \to -\infty$ the CES function nests in the so called Leontief production function which has and unbounded elasticity of substitution. The functional form can be achieved by applying the L'Hoptial rule to the CES function.

4.3. Translog porduction function

According to Sickles and Zelenyuk (2019), the Translog production function can be described as a generalized form of the Cobb-Douglas function with quadratic and cross-product terms. The function can also be interpreted as the second-order Taylor polynomial approximation of the CES function for $\rho = 0$. The function was introduced by Christensen et al. (1971) and its general expression is as follows:

$$\ln y = \beta_0 + \sum_{j=1}^{N} \beta_j \ln x_j + \sum_{j=1}^{N} \sum_{k=1}^{N} \beta_{jk} (\ln x_j)(\ln x_k) \ , \ \beta_{jk} = \beta_{kj}, \forall j, k. \tag{4.14}$$

The MP is given as

$$MP_j = x \left[\beta_j + 2 \sum_{k=1}^{N} \beta_{jk} \ln(x_k) \right] / x_j \ , \ j = 1, \cdots, N; x_j > 0. \tag{4.15}$$

The Translog function is equal to the Cobb-Douglas function if $\beta_{jk} = 0, \forall j, k$. When the Translog production is derived from the CES function and a two input factor case

with the variables Y, A, L, K is given, according to Berndt and Christensen(1973), the Translog production function can be expressed as:

$$\ln(Y) = \ln(A) + \alpha \ln(L) + (1-\alpha) \ln(K) - \frac{1}{2} \rho\alpha(1-\alpha)[\ln(L) - \ln(K)]^2 \tag{4.16}$$

$$\ln(Y) = \ln(A) + \beta_L \ln(L) + \beta_K \ln(K) + \beta_{L^2} \ln^2(L) + \beta_{K^2} \ln^2(K) + \beta_{LK} \ln(L) \ln(K) \tag{4.17}$$

This function is not only practical for the use of linear regression, it also visualises the connection to the Cobb-Douglas function. Its values for $\ln Y$ are different to the values of the Cobb-Douglas function for $0 < \rho \leq 1$ and when and $\ln(L) - \ln(K) \neq 0$. With increasing ρ, the difference of values between the Translog production function and the Cobb Douglas production function starts to increase. This is the reason why it was chosen for the simulation analysis in the later part of the thesis. The Simulation analysis will be presented in the following chapter.

5. Data Simulation and Model Comparison

Now that the theoretical part of the thesis is completed, the analytical part of the thesis can commence. As explained in the Motivation of the thesis, the main question is, how non-parametric machine learning regression methods perform in comparison to misspecified parametric regression models. To find a sophisticated answer for this question, a simulation study was held. The results of this simulation study will be discussed in the following chapters. For the experiment, four different regression models were programmed with the use of the programming language Python and the scikit-learn library, which is provided by Pedregosa et al. since 2011 and is one of the most used packages for machine learning in Python. The digital appendix includes four different jupyter-notebook files, one for each model. The first model is a linear regression model that minimises the sum of squared residuals (OLS method) and assumes the functional form, of a Cobb-Douglas production function with two input factors. This model will be called the Cobb-Douglas regression model. For the Cobb-Douglas regression model the following parametric form is assumed and used to make predictions for the endogenous variable $\ln(Y)$:

$$\ln(Y) = \ln(A) + \beta_L \ln(L) + \beta_K \ln(K).$$

The regression parameter β_L estimates the parameter α of the two-factor Cobb-Douglas function and therefore the parameter β_K can also be interpreted as the estimation for $(1-\alpha)$ or, generally speaking, the marginal products of the production function. The second model is also a linear regression model that minimises the sum of squared residuals using the OLS method. But instead of the functional form of a Cobb-Douglas production function, the functional form of the Translog production function is assumed to make predictions for $\ln(Y)$. This regression model will be called Translog regression model an can be written as follows:

$$\ln(Y) = \ln(A) + \beta_L \ln(L) + \beta_K \ln(K) + \beta_{L^2} \ln^2(L) + \beta_{K^2} \ln^2(K) + \beta_{LK} \ln(L) \ln(K)$$

The remaining two regression models are based on non-parametric methods, that do not assume any kind of functional form and therefore cannot be expressed with a parametric function. The third regression model uses the random forest regression algorithm, while the last regression model is based on the KNN approach. They will be called Random Forest regression model and KNN regression model respectively.

The data that is used for training, validating, and testing of the models was artificially generated. The generated data includes three variables. The two exogenous variables are denoted as $\ln(L)$ and $\ln(K)$, whereas the exogenous the exogenous variable is denoted as $\ln(Y)$. The values of the two exogenous variables were generated by randomly drawing values between 0 and 10 from a continuous uniform distribution. A seed was set before generating random values, to make the generation of the data and the outcomes reproducible, when running the scripts. The values of $\ln(Y)$ were then computed, by applying the Translog production function (5.16) to the randomly generated exogenous variables. The given function is:

$$\ln(Y) = \ln(A) + \alpha \ln(L) + (1-\alpha) \ln(K) - \frac{1}{2} \rho\alpha(1-\alpha)[\ln(L) - \ln(K)]^2.$$

After computing the values of $\ln(Y)$, the relationship can be described perfectly and has no variance at all. To add some variation to the data, an error term is added to the computed values of the endogenous variable, to create a statistical model with an additive error. The values of the error term are randomly drawn from a normal distribution with the standard deviation σ and the expectation of the distribution is given

by $\mu = 0$. For the experiment the following values of α $\ln(A)$ and were kept constant for the simulation. A production function, where the inputs have the same marginal products and and the intercept is close to 0 was assumed, resulting in the following values: $\alpha = 0.5$ and $\ln(A) = 0.1$. When taking a look at the formula that simulates $\ln(Y)$, it can be observed that for $\rho = 0$ the functional form of a production function is given, and that the Cobb-Douglas regression model is correctly specified for predicting the values of the endogenous variables. For $\rho > 0$, the Cobb-Douglas regression model is not capable of modelling the underlying relationship between the variables. The regression model then is misspecified. With increasing ρ the difference between the values of the Translog production function and the Cobb-Douglas production will increase for $\ln(L) \neq \ln(K)$ and therefore the value of the parameter ρ can be interpreted as the degree of misspecification. Then one can say that the degree of misspecification increases with ρ. To find out how the parametric models perform in comparison to the misspecified Cobb-Douglas regression model, the RMSE of all 4 Models was computed for different combinations of ρ and n, which denotes the total number of samples. For each combination of ρ and n, the RMSE of the models can be compared, to find out which model performs best for a certain combination. A model that has the lowest RMSE dominates all other models. This is done for different values of the standard deviation of the error term, to see if variance has an effect on the relative predictive power of the models. The analysis was exercised for $n \in \{125, 250, 500, 1000\}$, $\rho \in \{0, 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, 1\}$ and $\sigma \in \{0.5, 1, 1.5\}$ to limit the amount of data that can be interpreted. The notebook files allow for a relatively free change of those variables, and the user is encouraged to try running the models for different values of the parameters. The following chapter it will discussed how the RMSE of each model changes with n, ρ and σ. Afterwards, the visual representation of the models will be show for two edge cases. Finally the models will be compared to each other, aiming to gather insights on when which model is a better choice compared to the other models.

5.1. RMSE Analysis

To recall, the values of the endogenous variable in the simulated data set were computed by applying the Taylor approximated formula of the Translog function(5.16) to

the randomly drawn values of the exogenous variables. Therefore, the Translog regression model correctly specifies the underlying relationship of the data for every value of ρ, by accounting for the influence of the squared variables and and interaction term. Therefor, a change of ρ should not affect the predictive power of the model. This hypothesis can be confirmed, looking at the RMSE values of the Translog regression model. For $\sigma = 1$, the RMSE values for different numbers of samples are presented in Table 6. For a constant value of n, the RMSE of the Translog regression model is constant for all possible values of ρ. One can also observe that the RMSE of the model continuously decreases with the number of samples and is independent of ρ. The value of the RMSE is maximal for $n = 125$ with RMSE = 1.046781 an minimal for $n = 100$ with RMSE = 0.928443. When the standard deviation of the normal distribution is given as $\sigma = 1$, the Translog regression model seems to fit to the data well, since all RMSE values are very close to 1. Similar insights can be gained for different values of σ. The RMSE values for $\sigma = 0.5$ and $\sigma = 1.5$ are shown in Table 5 and Table 7. The RMSE is constant for every value of ρ and decreases for an increasing number of samples resulting in a minimum for $n = 1000$. For $\sigma = 0.5$ the RMSE values of the Translog regression model are close to 0.5 and for $\sigma = 1.5$ close to 1.5. The lowest RMSE = 0.464222 is achieved for $\sigma = 0.5$ and $n = 1000$, whereas the maximum RMSE =1.570172 is achieved for $\sigma = 1.5$ and $n = 125$. Therefore, it can be said that the model has its best predictive performance for a data set with low unexplained variance and a large number of samples, while the predictions are worst for a small number of samples with high variance. An increase of unexplained variance increases the RMSE of the model.

In the previous chapter, one could observe that the Cobb-Douglas production function and the Translog production function are the same for $\rho = 0$. Therefore, the Cobb-Douglas regression model correctly specifies the relationship that underlies the simulated data for $\rho = 0$, but is misspecified for every other value of ρ. The RMSE values of the Coob-Douglas regression model are shown in Table 2, 3 and 4. Looking at the different achieved values, one can observe the effect of ρ on the RMSE of the Cobb-Douglas regression model. Independent of σ and n, the RMSE increases significantly when ρ increases. The Cobb-Douglas regression model has its best predictive perfor-

mance when $\rho = 0$ and model misspecification is not the case. For $\rho = 0$, the RMSE is close to σ. Unlike the Translog regression model, the Cobb-Douglas regression model does not consistently show an increase of the RMSE for an increasing number of samples. The relationship between these variables seems ambiguous. The lowest value of the RMSE = 0.464332 is achieved for $\sigma = 0.5$, $n = 1000$ and $rho = 0$, whereas the the maximum value of the RMSE = 3.267709 is achieved for $\sigma = 1.5$, $n = 500$ and $rho = 0$. Therefore it can be said that the models has its best predictive performance, when it correctly specifies the underlying relationship between the given variables and when the amount of unexplained variance is low. The worst predictive performance of the Cobb-Douglas regression model is achieved, when the model is heavily misspecified and the variance is high. An increase of unexplained variance increases the RMSE of the model.

Now the values of the first non-parametric model, the KNN regression model will be looked at to find possible relationships. Since the KNN regression model is non-parametric, misspecification is not possible and therefore, an increase of the RMSE for increasing values of ρ is not necessarily expected. Looking at the experimental data, one can say that a relationship between the RMSE and ρ for the KNN regression model is not really observable. For different numbers of samples and values of the standard deviation, the RMSE reacts differently to an increase of ρ. Both an increasing and decreasing RMSE values are observable for, and therefore an observable relationship does not appear to exist. However, one can see that RMSE is minimal for $n = 1000$ for every value of ρ and σ. This might be the case, because a data set with a sufficient number of samples results in a much smaller predictor space, since the distance between the datapoints and the coordinates of the predicted value are much smaller. A much smaller predictor space could then lead to a more precise prediction of the KNN regression model. It is also possible, that with an increasing number of samples, the number of neighbours can be increased to make a more robust prediction. For $n = 1000$ the RMSE seems to slightly decrease when ρ increases. A probable explanation for this is that the distances between points increase when the effect of the squared difference of the exogenous variables on the endogenous variable increases. The lowest value of the RMSE = 0.00438 is achieved for $\rho = 0$, $\sigma = 0,5$ and $n = 1000$. whereas the the maximum value of the RMSE = 1.718382 is achieved for $\sigma = 1.5$, $n = 500$ and $rho = 1$.

To summarise, it can be said that ρ does not appear to have a significant effect on the RMSE, but the model appears to predict best, when the number of provided samples is large. An increase of the unexplained variance increases the RMSE and therefore reduces predictive power of the KNN model.

The last model with interpretable data is the Random Forest regression model and the results are pretty similar to the KNN regression model. Since the Random Forest regression model is also non-parametric, misspecification is not possible and therefore, an increase of the RMSE for increasing ρ is not necessarily expected. Looking at the data, a relationship between the RMSE and ρ for is not observable for the Random Forest regression model. The RMSE seems to deviate for different values of ρ, but a strict relationship between ρ and the RMSE cannot be observed. One can see that the RMSE is again minimal, for the largest number of samples. Just like the other regression model, the Random Forest regression model also seems to profit from a large number of samples. This could be due to that fact, that more data enables the decision trees to divide the data into more splits. Then averaging would yield a better prediction for the rectangular subspace of one tree. As a result, the Random Forest regression model can produce more precise predictions, because of the improved trees. The lowest value of the RMSE = 0.500072 is achieved for $\rho = 0.2$, $\sigma = 0,5$ and $n = 1000$, whereas the the maximum value of the RMSE = 1.760972 is achieved for $\sigma = 1.5$, $n = 125$ and $\rho = 1$. To summarise, it can be said that ρ does not appear to have a significant effect on the RMSE, but the model appears to predict most precisely, if the number of provided samples is large. Just like any other model, an increase of the unexplained variance increases the RMSE and therefore reduces predictive power of the Random Forest regression model.

5.2. Model Comparison

After examining the relationships between the variables σ, ρ, n an the RMSE of the different models, this chapter is dedicated to the relative performance of the different parametric and non-parametric models. At first, it will be discussed which model has the highest predictive performance, which is equal to the lowest RMSE, for all combinations of σ, ρ and n. Then it will be identified, when the non-parametric machine

learning models will perform better than the misspecified Cobb-Douglas regression model, and how the RMSE values of the Translog regression model compare to the values of the Cobb-Douglas regression model. After answering the question, when to use which model, the predictions of the models will be visualised for two edge cases to see why a certain model performs better than the others.

First, the RMSE values of the Translog regression model will be compared to the values of the Cobb-Douglas regression model. Looking at the RMSE values of each model respectively, it can be observed that the RMSE values are very similar for $\rho \in \{0, 0.1\}$. For the majority of parameter combinations, the Translog regression model dominates the predictive power of the Cobb-Douglas regression model significantly. The Cobb-Douglas regression model only dominates the Translog regression model for the following parameter combinations:

$$(\sigma, \rho, n) \in \{(0.5, 0, 500), (1, 0, 500), (1, 0.1, 125), (1.5, 0, 500), (1.5, 0.1, 125)\}$$

This could be caused by the degree of complexity, that is enabled by the Translog regression model, since it allows the estimation of a quadratic function, even though the underlying relationship is linear. The coefficients β_{L^2}, β_{K^2} and β_{LK} are 0, but the estimated coefficients are only close to 0, therefore resulting in the estimation of a quadratic function. For $n = 500$ and $\rho = 0$, the Translog regression model could be slightly overfitting, resulting in a better performance of the Cobb-Douglas regression model. For the combinations that have 125 samples, the small amount of random data is and the rising standard deviation could be the reason for the dominance of the Cobb-Douglas regression model. To summarise, it can be said, that both models perform very similar when they correctly specify the given relationship, or the relationship or if the model is just marginally misspecified. For higher levels of ρ, the misspecified Cobb-Douglas regression model has predictive power that is significantly lower if compared to the correctly specified Translog regression model.

Now that both parametric regression models have been compared to each other, the difference between Cobb-Douglas regression model and the non-parametric regression models can be investigated, starting with the KNN regression model. For $\sigma = 0.5$ the RMSE of the Cobb-Douglas regression model is lower for all values of $\rho \in \{0, 0.1\}$,

except $n = 1000$ and $\rho = 0.1$. For all other combinations of ρ and n, the RMSE of the KNN regression model is lower than the RMSE of the Cobb-Douglas regression model. When the standard deviation of the normally distributed error term is increased from 0.5 to 1, the number parameter combination, where the Cobb-Douglas regression model dominates the KNN regression model, increases by two. The new set of parameters, where the the RMSE of the Cobb-Douglas regression model is lower are the following:

$$(\sigma, \rho, n) \in \{(1, 0.2, 125), (1, 0.1, 1000)\}$$

Increasing σ even further, from 1 to 1.5, three additional sets of are dominated by the Cobb Douglas regression model:

$$(\sigma, \rho, n) \in \{(1.5, 0.3, 125), (1.5, 0.2, 250), (1.5, 0.2, 1000)\}$$

The results suggest that an increase of the standard deviation of the error term allows the misspecifed Cobb-Douglas regression model to dominate the KNN regression model for higher levels of ρ, especially if the number of samples is small.

When comparing the RMSE of the Random Forest regression model to the Cobb-Douglas regression model, an increase of σ appears to affect the dominance relation of the models in a similar way. For $\sigma = 0.5$ the the Cobb-Douglas regression model has a lower RMSE for all values of n. It also dominates the predictive power of the Random Forest regression model for $\rho = 0.1$ and $n \in \{125, 250\}$. Increasing the standard deviation of the normally distributed error term from 0.5 to 1 results in an increasing number of sets that are dominated by the Cobb-Douglas regression model. The additional sets that are dominated by The Cobb-Douglas regression are:

$$(\sigma, \rho, n) \in \{(1, 0.1, 1000), (1, 0.2, 125)\}$$

Increasing the standard deviation again from 1.0 to 1.5 has the same effect and adds 2 additional sets to the dominated stes of the modes. For $\sigma = 1.5$ the Cobb Douglas regression model dominates the Random Forest regression model for all values of n and for $\rho \in \{(0, 0.1)$. Additionally, the RMSE is also lower for the following sets:

$$(\sigma, \rho, n) \in \{(1.5, 0.2, 125), (1.5, 0.2, 250)\}$$

As a conclusion, it can be said that an increase of σ affects the relative performance of the regression models, and allows the misspecified Cobb-Douglas regression model to perform better than the Random Forest regression model, especially for lower values of n.

5.3. Visual analysis

As described in the previous chapter, the non-parametric regression models appeared to perform worse than the Cobb-Douglas regression model, when the Cobb-Douglas regression model correctly specifies the underlying relationship of the data and when the standard deviation of the error term is large. On the Other hand, the non-parametric models seem to perform significantly better for a high degree of misspecification and a large number of samples.

In this chapter, the heat maps of the different regression model will be looked at for two different sets of parameters. The sets of parameters are defined as follows:

$$Set_1 : (\sigma, \rho, n) = (1.5, 0, 125) \tag{5.1}$$

$$Set_2 : (\sigma, \rho, n) = (1.5, 1, 1000) \tag{5.2}$$

Keeping the insights of the last chapter in mind, the Cobb-Douglas regression model is expected to perform better than the non-parametric regression models with data that is simulated with the first set of parameters, whereas the second set of parameters should suit the non-parametric regression models better than the Cobb-Douglas regression model. The visualisation of the Translog regression model will be used as a reference for comparison, since it models the true relationship of the variables within the data set. In the written part of the thesis, heat maps of the regression model will be compared visually, since they are a good option for visualising a relationship of three variables in two dimensions. The digital appendix also offers three-dimensional plots that can be interacted that can be use to achieve a better understanding of the regression models.

For Set_1 the Cobb Douglas regression model and the Translog regression models both correctly specify the relationship between the endogenous and the exogenous variable,

because $\rho = 0$ When taking a look at the heat maps of both models(Figure 1 and Figure 2), both heat maps visualise the linear relationship between the natural logarithm of the input variables and the natural logarithm of and significant difference between the models is observable. The highest fitted values of both models are predicted for the values of $\ln(L) = 10$ and $\ln(K) = 10$ and the smallest predict values are observed for $\ln(L) = 0$ and $\ln(K) = 0$. This is not the case for the KNN regression model, which is shown in Figure 3. A continuous shift between the colours of the scale that represent the predicted values is no more visible. Instead, there are certain areas where the colour of the heat map and therefore the predicted values are the same for certain combinations of input variables. This is because of the nature of the KNN Algorithm. The predicted values are the same for combinations for each predictor space, where the k-closest neighbours are the same. The KNN regression seems to be slightly off the real relationship, because the lowest predicted values are observable for $\ln(L) \approx 2$ and $\ln(K) \approx 0$ and the highest fitted values are given for $\ln(L) \approx 7$ and $\ln(K) \approx 10$. It still appears that the KNN model is able to roughly describe the underlying relationship of the data. The predictions of the Random Forest regression model are visualised in Figure 4. Like the KNN Regresison model, the smallest and highest predicted values are not given for $\ln(L) = 10$, $\ln(K) = 10$ and $\ln(L) = 0$, $\ln(K) = 0$. The rectangular sub-spaces of the tree, where predictions are the same can be observed. Remembering that a decision tree splits the data into partitions and then predicts the same values for one of the leaves of the tree. After averaging the predictions of the multiple regression trees, the rectangular structure is still visible. All in all, the Random Forest regression model is also able to visualise the underlying relationship, but not as smoothly as the Cobb-Douglas and the Translog regression model.

For Set_2 the Translog regression model correctly specifies the underlying relationship, while the Cobb-Douglas regression model is heavily misspecified, because $\rho = 0$. Looking at the predicted values of the Translog regression model (Table 5), one can observe, that there are two areas, where the predicted values of the Translog regression model are minimal. This is the case when the difference of $\ln(L)$ and $\ln(K)$ is maximised. The predicted values when $\ln(L)$ and $\ln(K)$ are big and their difference is 0. When the Cobb-Douglas regression model is used to make predictions for Set_2, it can be

observed, that the model is not able to capture the quadratic function at all, and looks Figure 4 looks almost exactly as 2. The KNN regreession model seems closer to the Translog, regression model and identifies the relationship correctly. The size of the subspace has decreased tremendously. A high number of samples, which is given in Set_2, reduces the size of the predictor spaces and therefore enabling a higher number of predicted values. There are some pixels in Figure 7 that look significantly darker or lighter than the area around it. This occurs, because the model that was chosen after hyper-parameter tuning and cross-validation uses the distance as a weight before averaging values and calculating the prediction value. Overall the KNN model shows a relationship that is very close to the Translog regression model. The last Graph (Figure 8) shows the predicted values of the Random Forest regression model for Set_2. Again the rectangular sub-spaces are visible and the model correctly identifies the areas, where the predicted values are supposed to be at minimum and maximum. Because of the rectangular subspaces, the transition between predicted values does not appear to be continuous, unlike the Translog regression model and the KNN regression model, but seems to model the underlying relationship much better in comparison to the misspecified Cobb-Douglas regression model.

6. Conclusion

This thesis discussed how the standard deviation, the degree of misspecification and the size of the data set affect the predictive performance of regression models. It also investigated the effect of misspecification on the model choice to find out when a non-parametric model might be a better choice than a misspecified parametric model.

The results that were discussed in this thesis show that, independent of the model choice, the standard deviation of the normally distributed error term has a significant effect on the RMSE of every model. If the standard deviation of the error term and therefore the unexplained variance increases, the RMSE of a regression model will ceteris paribus also increase and vice versa. Independent of the form of the model and every other possible parameter, the prediction error of all models was minimal for the largest number of samples. As one of the main insights, the analysis showed that the RMSE of a linear model, that is used to predict a relationship of quadratic nature, drastically increases, when the the impact of the quadratic term on the values of the endogenous variable increases. Finally, the models were compared to visualise when which model choice is best, for a certain set of parameters. The analysis showed that the parametric models produced better predictions, when the underlying relationship of the data was correctly specified or just marginally misspecified. The Parametric models performed significantly better when the unexplained variance of the model was relatively high and the amount of data was relatively low. The non-parametric regression models produced more precise predictions than the linear model, if the linear model is heavily misspecified and the number of samples within the data set is relatively high. Increasing the unexplained variance within the data allowed the linear model to still dominate the regression model for marginally higher degrees of misspecification. These insights can be used to decide when which kind of model is suitable for making good predictions. If the relationship within of the exogenous and endogenous variables is given and the number of samples is low, then a parametric model should be chosen to produce good predictions. This is especially the case when the standard deviation of the error term is high. However, if the functional form of the correct model is not known beforehand and a relatively large number of samples is provided, non-parametric regression models appear to be the better choice to minimise prediction error.

References

[Abbott 2014] Abbott, D. (2014). Applied predictive analytics: Principles and techniques for the professional data analyst. John Wiley & Sons.

[Arrow et al. 1961] Arrow, K. J., Chenery, H. B., Minhas, B. S., & Solow, R. M. (1961). Capital-labor substitution and economic efficiency. The review of Economics and Statistics, 43(3), 225-250.

[Bekkerman et al. 2011] Bekkerman, R., Bilenko, M., & Langford, J. (Eds.). (2011). Scaling up machine learning: Parallel and distributed approaches. Cambridge University Press.

[Blackorby and Russell 1981] Blackorby, C., & Russell, R. R. (1981). The Morishima elasticity of substitution; symmetry, constancy, separability, and its relationship to the Hicks and Allen elasticities. The Review of Economic Studies, 48(1), 147-158.

[Blackorby and Russell 1989] Blackorby, C., & Russell, R. R. (1989). Will the real elasticity of substitution please stand up?(A comparison of the Allen/Uzawa and Morishima elasticities). The American economic review, 79(4), 882-888.

[Christensen et al. 1973] Christensen, L. R., Jorgenson, D. W., & Lau, L. J. (1973). Transcendental logarithmic production frontiers. The review of economics and statistics, 28-45.

[Cobb and Douglas 1928] Cobb, C. W., & Douglas, P. H. (1928). A theory of production. The American Economic Review, 18(1), 139-165.

[Hastie et al. 2009] Hastie, T., Tibshirani, R., & Friedman, J.(2009). The elements of statistical learning. New York: Springer series in statistics.

[Hicks 1932] Hicks, J. R. (1932). The Theory of Wages. London: Macmillan.

[Hicks and Allen 1934] Hicks, J. R., & Allen, R. G. (1934). A reconsideration of the theory of value. Part II. A mathematical theory of individual demand functions. Economica, 1(2), 196-219.

[Kuhn and Johnson 2013] Kuhn, M., & Johnson, K. (2013). Applied predictive modelling (Vol. 26, p. 13). New York: Springer.

[Lipsey 2018] Lipsey, R. G. (2018). A Reconsideration of the Theory of Non-Linear Scale Effects: The Sources of Varying Returns to, and Economies of, Scale. Cambridge University Press.

[Müller and Guido 2016] Müller, A. C., & Guido, S. (2016). Introduction to machine learning with Python: a guide for data scientists. " O'Reilly Media, Inc.".

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

[Olsson 2013] Olsson, O. (2013). Essentials of advanced macroeconomic theory. Routledge.

[Sickel and Zelenyuk 2019] Sickles, R. C., & Zelenyuk, V. (2019). Measurement of productivity and efficiency. Cambridge University Press.

[Solow 1956] Solow, R. M. (1956). A contribution to the theory of economic growth. The quarterly journal of economics, 70(1), 65-94.

A. Tables

$\sigma = 0.5$	RMSE of the Cobb-Douglas regression model			
$\boldsymbol{\rho}$ / $\boldsymbol{n}$	**125**	**250**	**500**	**1000**
0	0.525465	0.484259	0.475501	0.464332
0.1	0.524375	0.517248	0.575242	0.515483
0.2	0.650852	0.656918	0.769947	0.657359
0.3	0.849723	0.852347	1.005906	0.845457
0.4	1.081774	1.073503	1.260153	1.055346
0.5	1.329746	1.307394	1.523561	1.276319
0.6	1.586189	1.548259	1.792095	1.503498
0.7	1.847579	1.793291	2.063754	1.734446
0.8	2.112081	2.040989	2.337450	1.967836
0.9	2.378657	2.290488	2.612542	2.202891
1.0	2.646680	2.541258	2.888631	2.439132

Table 2: RMSE of the Cobb-Douglas regression model for $\sigma = 0.5$, $\ln(A) = 0.1$ and $\alpha = 0.5$

$\sigma = 1$	RMSE of the Cobb-Douglas regression model			
$\boldsymbol{\rho}$ / $\boldsymbol{n}$	**125**	**250**	**500**	**1000**
0	1.05093	0.968518	0.951002	0.928664
0.1	1.013545	0.968827	1.017573	0.951059
0.2	1.048750	1.034495	1.150485	1.030966
0.3	1.149896	1.154424	1.329996	1.156524
0.4	1.301703	1.313837	1.539895	1.314718
0.5	1.488752	1.500199	1.769399	1.495225
0.6	1.699447	1.704695	2.011812	1.690915
0.7	1.926042	1.921543	2.262987	1.897094
0.8	2.163548	2.147005	2.520307	2.110692
0.9	2.408739	2.378633	2.782067	2.329668
1.0	2.659491	2.614787	3.047123	2.552639

Table 3: RMSE of the Cobb-Douglas regression model for $\sigma = 1$, $\ln(A) = 0.1$ and $\alpha = 0.5$

$\sigma = 1.5$	RMSE of the Cobb-Douglas regression model			
$\boldsymbol{\rho}$ / $\boldsymbol{n}$	**125**	**250**	**500**	**1000**
0	1.576395	1.452777	1.426504	1.392996
0.1	1.527025	1.441776	1.480610	1.405174
0.2	1.525900	1.475748	1.583229	1.457693
0.3	1.573125	1.551743	1.725727	1.546449
0.4	1.664588	1.664013	1.899149	1.665659
0.5	1.793536	1.805804	2.095832	1.809313
0.6	1.952555	1.970755	2.309842	1.972077
0.7	2.134936	2.153551	2.536797	2.149614
0.8	2.335213	2.350030	2.773522	2.338562
0.9	2.549170	2.557042	3.017717	2.536372
1.0	2.773644	2.772227	3.267709	2.741127

Table 4: RMSE of the Cobb-Douglas regression model for $\sigma = 1.5$, $\ln(A) = 0.1$ and $\alpha = 0.5$

$\sigma = 0.5$	RMSE of the Translog regression model			
$\boldsymbol{\rho}$ / $\boldsymbol{n}$	**125**	**250**	**500**	**1000**
0	0.523391	0.482503	0.476789	0.464222
0.1	0.523391	0.482503	0.476789	0.464222
0.2	0.523391	0.482503	0.476789	0.464222
0.3	0.523391	0.482503	0.476789	0.464222
0.4	0.523391	0.482503	0.476789	0.464222
0.5	0.523391	0.482503	0.476789	0.464222
0.6	0.523391	0.482503	0.476789	0.464222
0.7	0.523391	0.482503	0.476789	0.464222
0.8	0.523391	0.482503	0.476789	0.464222
0.9	0.523391	0.482503	0.476789	0.464222
1.0	0.523391	0.482503	0.476789	0.464222

Table 5: RMSE of the Translog regression model for $\sigma = 0.5$, $\ln(A) = 0.1$ and $\alpha = 0.5$

$\sigma = 1$	RMSE of Translog the regression model			
$\boldsymbol{\rho}$ / $\boldsymbol{n}$	**125**	**250**	**500**	**1000**
0	1.046781	0.965006	0.953578	0.928443
0.1	1.046781	0.965006	0.953578	0.928443
0.2	1.046781	0.965006	0.953578	0.928443
0.3	1.046781	0.965006	0.953578	0.928443
0.4	1.046781	0.965006	0.953578	0.928443
0.5	1.046781	0.965006	0.953578	0.928443
0.6	1.046781	0.965006	0.953578	0.928443
0.7	1.046781	0.965006	0.953578	0.928443
0.8	1.046781	0.965006	0.953578	0.928443
0.9	1.046781	0.965006	0.953578	0.928443
1.0	1.046781	0.965006	0.953578	0.928443

Table 6: RMSE of the Translog regression model for $\sigma = 1$, $\ln(A) = 0.1$ and $\alpha = 0.5$

$\sigma = 1.5$	RMSE of the Translog regression model			
$\boldsymbol{\rho}$ / $\boldsymbol{n}$	**125**	**250**	**500**	**1000**
0	1.570172	1.447510	1.430367	1.392665
0.1	1.570172	1.447510	1.430367	1.392665
0.2	1.570172	1.447510	1.430367	1.392665
0.3	1.570172	1.447510	1.430367	1.392665
0.4	1.570172	1.447510	1.430367	1.392665
0.5	1.570172	1.447510	1.430367	1.392665
0.6	1.570172	1.447510	1.430367	1.392665
0.7	1.570172	1.447510	1.430367	1.392665
0.8	1.570172	1.447510	1.430367	1.392665
0.9	1.570172	1.447510	1.430367	1.392665
1.0	1.570172	1.447510	1.430367	1.392665

Table 7: RMSE of the Translog regression model for $\sigma = 1.5$, $\ln(A) = 0.1$ and $\alpha = 0.5$

$\sigma = 0.5$	RMSE of the KNN regression model			
$\boldsymbol{\rho}$ / $\boldsymbol{n}$	**125**	**250**	**500**	**1000**
0	0.646461	0.549033	0.575158	0.500438
0.1	0.619273	0.553231	0.589371	0.501689
0.2	0.602792	0.565323	0.580463	0.502091
0.3	0.597904	0.581063	0.546893	0.504547
0.4	0.604890	0.593265	0.548472	0.509183
0.5	0.623349	0.610309	0.551310	0.509528
0.6	0.675853	0.629488	0.555387	0.514412
0.7	0.694507	0.646783	0.560677	0.515325
0.8	0.648443	0.667204	0.567146	0.520554
0.9	0.728206	0.722673	0.574753	0.525857
1.0	0.758973	0.751771	0.583455	0.531955

Table 8: RMSE of the KNN regression model for $\sigma = 0.5$, $\ln(A) = 0.1$ and $\alpha = 0.5$

$\sigma = 1$	RMSE of the KNN regression model			
$\boldsymbol{\rho}$ / $\boldsymbol{n}$	**125**	**250**	**500**	**1000**
0	1.194599	1.005181	1.035648	0.978423
0.1	1.168141	1.002765	1.047280	0.974556
0.2	1.128508	1.011603	1.064230	0.977285
0.3	1.097928	1.055935	1.086554	0.981268
0.4	1.085831	1.065210	1.106463	0.984237
0.5	1.078307	1.076505	1.134941	0.988377
0.6	1.082638	1.089759	1.167168	0.989405
0.7	1.096451	1.104901	1.170384	0.993828
0.8	1.119395	1.121853	1.113329	0.998838
0.9	1.130730	1.140536	1.127428	1.004427
1.0	1.136218	1.162892	1.048501	1.006985

Table 9: RMSE of the KNN regression model for $\sigma = 1$, $\ln(A) = 0.1$ and $\alpha = 0.5$

$\sigma = 1.5$	RMSE of the KNN regression model			
$\boldsymbol{\rho}$ / $\boldsymbol{n}$	**125**	**250**	**500**	**1000**
0	1.735231	1.493799	1.493698	1.455246
0.1	1.685410	1.481608	1.503084	1.456609
0.2	1.639682	1.483986	1.534676	1.459062
0.3	1.604916	1.490191	1.552155	1.459666
0.4	1.577366	1.500175	1.573329	1.463584
0.5	1.557413	1.513864	1.598049	1.468403
0.6	1.545352	1.575848	1.626156	1.468171
0.7	1.541368	1.585445	1.657476	1.470647
0.8	1.545524	1.596401	1.672868	1.475010
0.9	1.557755	1.608687	1.705753	1.475889
1.0	1.577872	1.622274	1.718382	1.480093

Table 10: RMSE of the KNN regression model for $\sigma = 1.5$, $\ln(A) = 0.1$ and $\alpha = 0.5$

$\sigma = 0.5$	RMSE of the Random Forest regression model			
$\boldsymbol{\rho}$ / $\boldsymbol{n}$	**125**	**250**	**500**	**1000**
0	0.626672	0.583686	0.559109	0.503703
0.1	0.646116	0.580471	0.530148	0.508686
0.2	0.630124	0.580116	0.532513	0.500072
0.3	0.619070	0.555161	0.535436	0.497854
0.4	0.601188	0.585329	0.525521	0.493940
0.5	0.610765	0.581103	0.540203	0.506591
0.6	0.686684	0.597369	0.564364	0.517213
0.7	0.802438	0.602198	0.576919	0.518490
0.8	0.938416	0.641984	0.605512	0.521414
0.9	1.060800	0.681985	0.641018	0.533511
1.0	1.139218	0.733231	0.663947	0.539985

Table 11: RMSE of the Random Forest regression model for $\sigma = 0.5$, $\ln(A) = 0.1$ and $\alpha = 0.5$

$\sigma = 1$	RMSE of the Random Forest regression model			
$\boldsymbol{\rho}$ / $\boldsymbol{n}$	**125**	**250**	**500**	**1000**
0	1.118431	1.062959	1.029879	0.973639
0.1	1.092438	1.036677	1.002686	0.980068
0.2	1.077211	1.031290	1.020834	0.967644
0.3	1.027273	1.037089	1.039214	0.966765
0.4	0.985752	1.049556	1.046565	0.992044
0.5	0.974721	1.060933	1.037390	0.980793
0.6	1.024058	1.061553	1.053429	0.971908
0.7	1.114700	1.078191	1.057702	0.989651
0.8	1.260447	1.094248	1.048250	0.988294
0.9	1.393189	1.132500	1.074904	0.994085
1.0	1.362333	1.138107	1.107434	1.009140

Table 12: RMSE of the Random Forest regression model for $\sigma = 1$, $\ln(A) = 0.1$ and $\alpha = 0.5$

$\sigma = 1.5$	RMSE of the Random Forest regression model			
$\boldsymbol{\rho}$ / $\boldsymbol{n}$	**125**	**250**	**500**	**1000**
0	1.581803	1.542096	1.510934	1.455896
0.1	1.583603	1.527369	1.500857	1.445525
0.2	1.566761	1.532623	1.504633	1.436045
0.3	1.520034	1.538438	1.521321	1.442069
0.4	1.499252	1.529365	1.525310	1.423859
0.5	1.484012	1.532470	1.553927	1.448748
0.6	1.524904	1.541773	1.560504	1.471996
0.7	1.529059	1.586140	1.529009	1.469444
0.8	1.589788	1.595488	1.524218	1.444211
0.9	1.684694	1.627173	1.560598	1.457166
1.0	1.760972	1.618951	1.670376	1.460495

Table 13: RMSE of the Random Forest regression model for $\sigma = 1.5$, $\ln(A) = 0.1$ and $\alpha = 0.5$

B. Figures

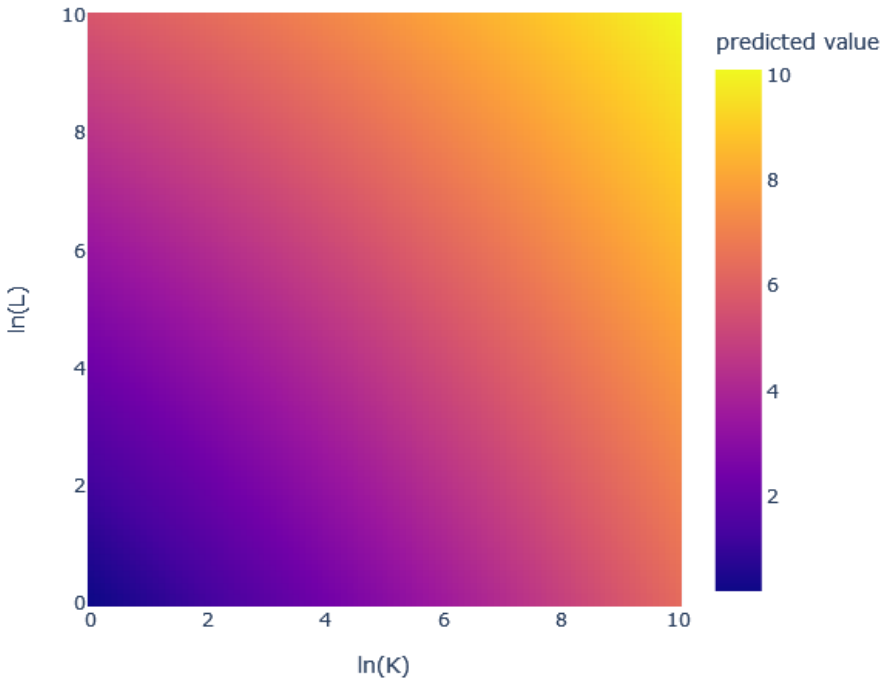

Figure 1: Heat map of the Translog regression model for Set_1

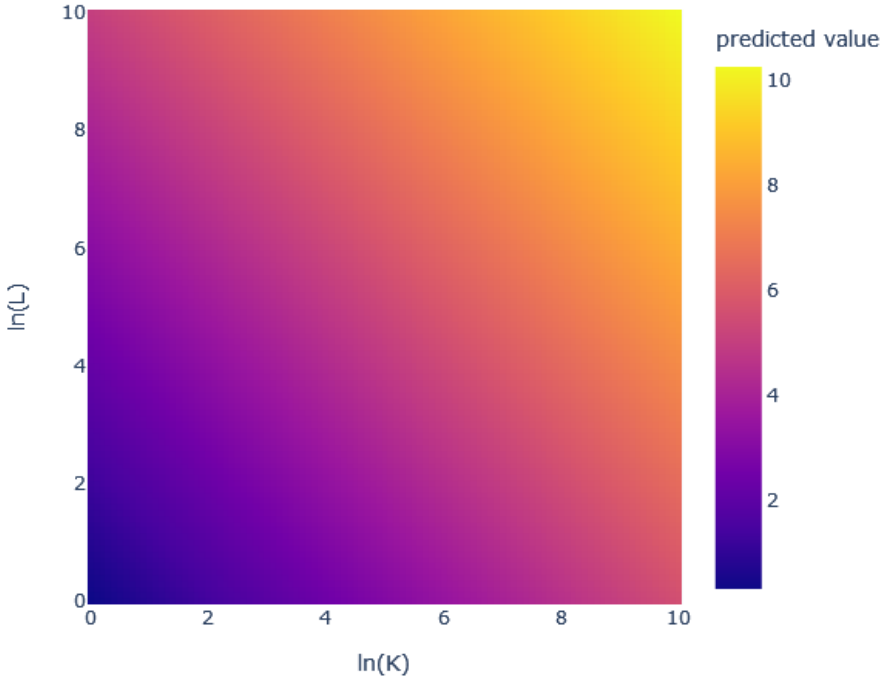

Figure 2: Heat map of the Cobb-Douglas regression model for Set_1

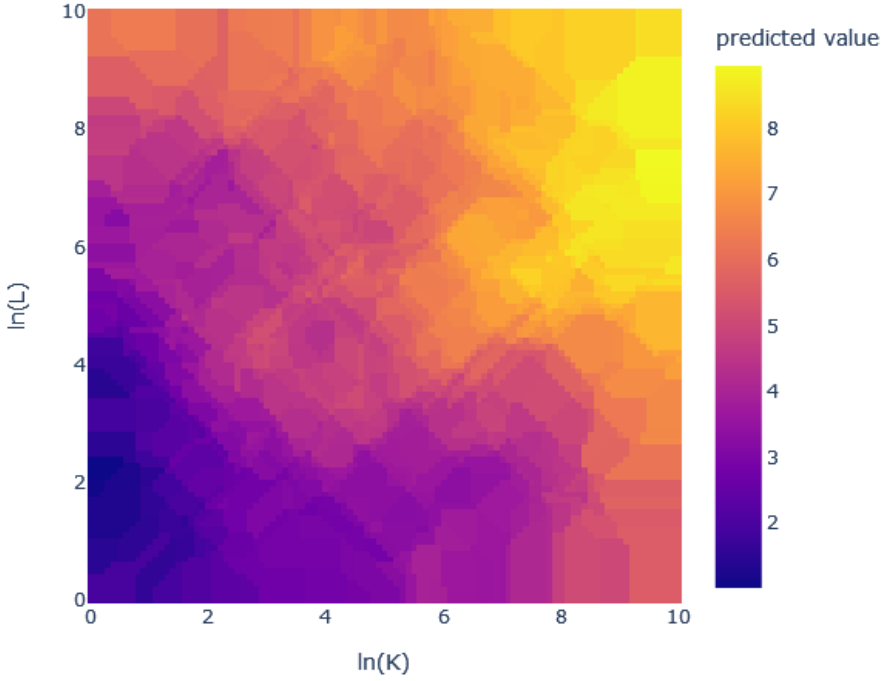

Figure 3: Heat map of the KNN regression model for Set_1

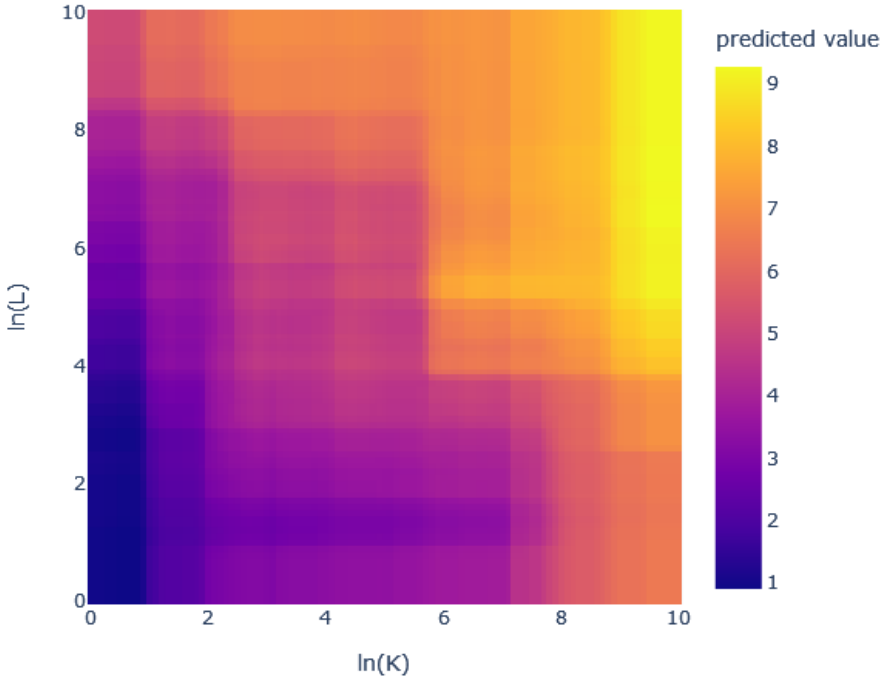

Figure 4: Heat map of the Random Forest regression model for Set_1

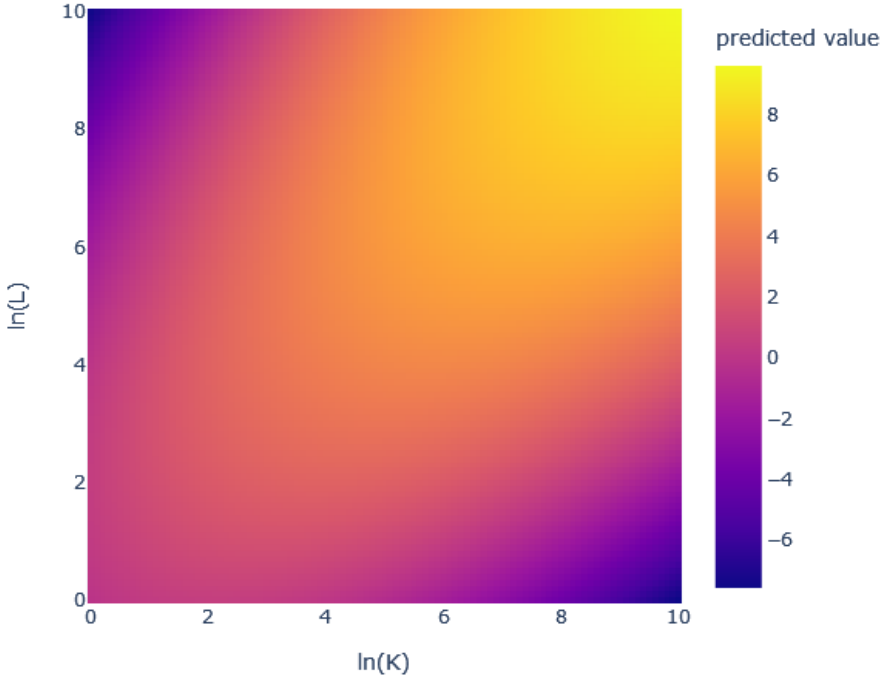

Figure 5: Heat map of the Translog regression model for Set_2

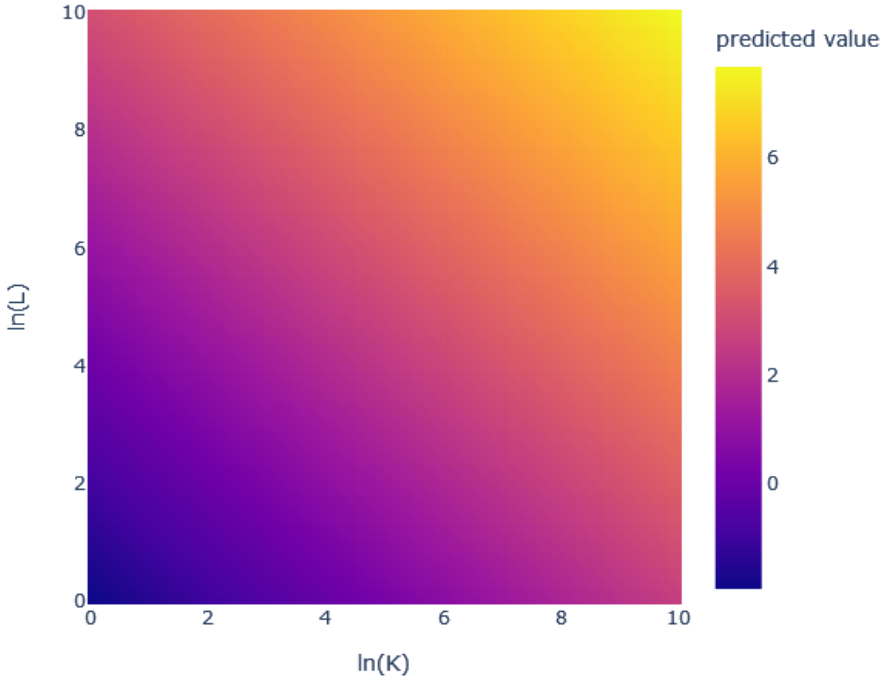

Figure 6: Heat map of the Cobb-Douglas regression model for Set_2

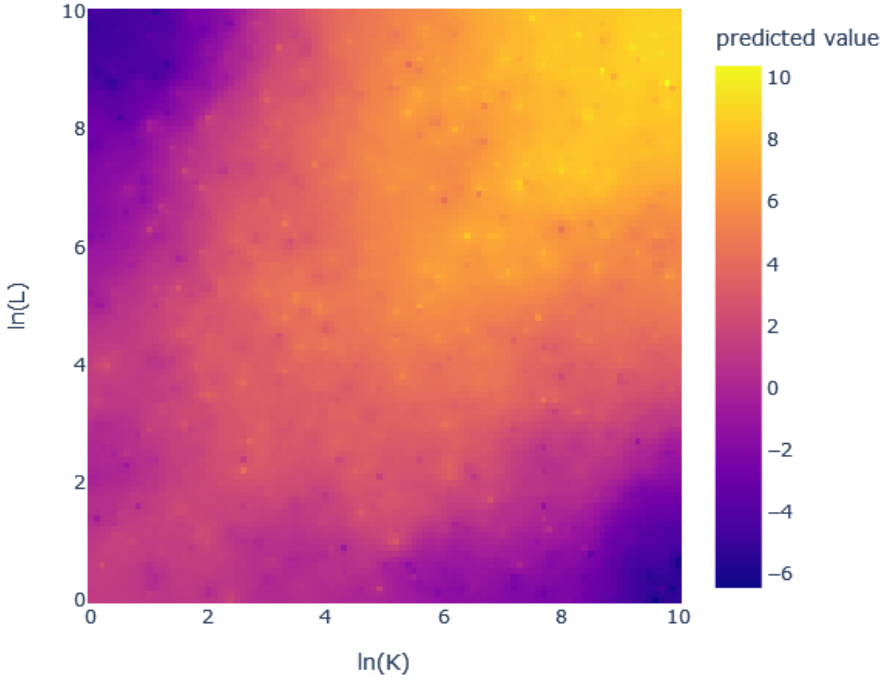

Figure 7: Heat map of the KNN regression model for Set_2

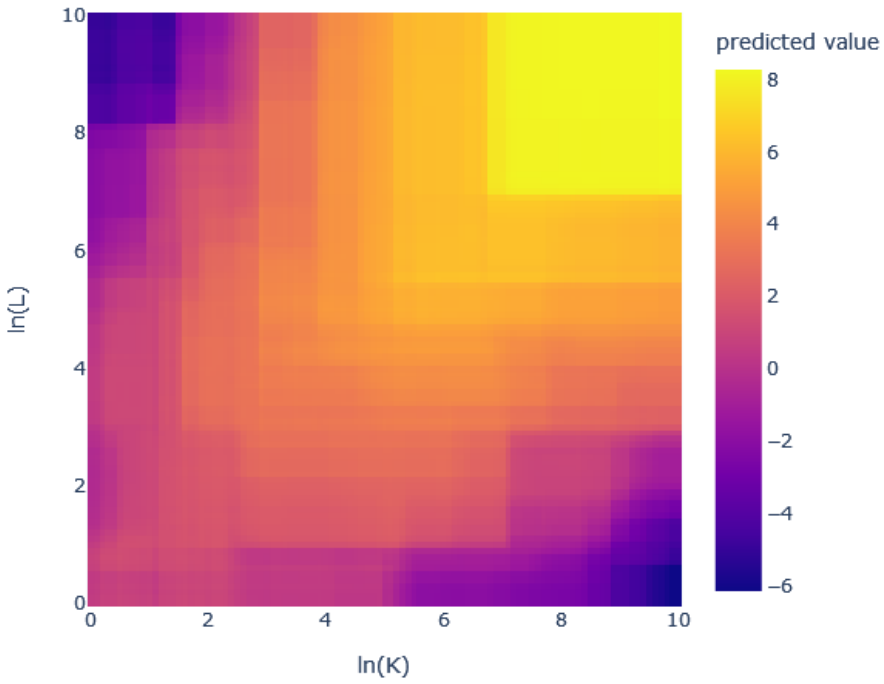

Figure 8: Heat map of the Random Forest regression model for Set_2

C. Digital Appendix

The digital Appendix shows a folder called *Bachelorarbeit*, that includes the thesis as a pdf file and four different jupyter-notebook files with the code that was used to program the different regression models and to generate the data. Each file represents on regression model. To run those files, jupyter-notebook and all necessary packages have to be installed. Downloading the Anconda-distribution for installation purposes is highly recommended. The individual version can be downloaded under the following link:

`https://www.anaconda.com/products/individual-d`

For the thesis, different software libraries were used to construct the notebook, and to build and visualise the regression models. The following libraries were used and need to be installed by executing the respective commands in the terminal:

scikit-learn `conda install -c anaconda scikit-learn`

numpy `conda install -c anaconda numpy`

pandas `conda install -c anaconda pandas`

ipywidgets `conda install -c anaconda ipywidgets`

ipython `conda install -c anaconda ipython`

plotly `conda install -c plotly plotly`

With Anaconda, the majority of libraries should already be installed. The only package that has to be installed via the terminal is plotly. If the installation processes are completed, the folder should be copied to the desktop. Afterwards, jupyter notebook can be launched in the home menu of the anaconda navigator or by executing the following command in the terminal:

```
jupyter notebook
```

If the start was successful, a tab in your browser should open, showing different folders. If the folder was copied to the desktop, just find the desktop folder, then the folder *Bachelorarbeit* and open it. The notebook files can now be accessed. As a last step,

you have to run the notebook. This can be done by pressing the following button, which starts the kernel and executes the code. From there on the code will be commented and explained inside each notebook file.

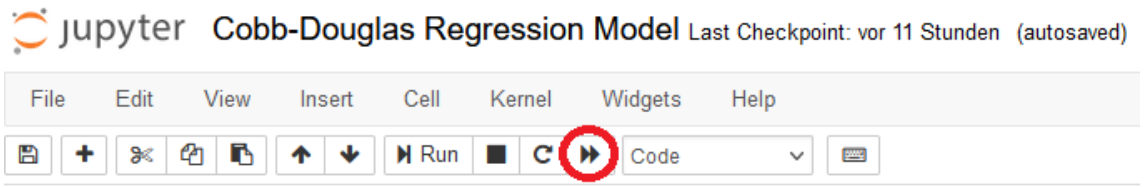

Figure 9: Running the Jupyter Notebook

Declaration of authorship

I hereby declare in lieu of an oath that this thesis is my own work and that I have not used any sources other than those listed in the bibliography. Content from published or unpublished works that has been quoted directly or indirectly or paraphrased is indicated as such. The thesis has not been submitted in the same or similar form for any other academic award. The electronic version I have submitted is completely identical to the hard copy version submitted.

Münster, 21.09.2021

Eidesstattliche Erklärung

Declaration in lieu of an oath

Name, Vorname:
Surname, Name:

Matrikelnummer:
Student ID Number:

Abschluss:
Degree:
☐ Bachelor ☐ Master

Studienfach:
Degree programme:

Titel der Abschlussarbeit:
Title of the thesis:

Hiermit versichere ich an Eides statt, dass ich die vorliegende Arbeit selbstständig und ohne die Benutzung anderer als der angegebenen Hilfsmittel angefertigt habe. Alle Stellen, die wörtlich oder sinngemäß aus veröffentlichten und nicht veröffentlichten Schriften entnommen wurden, sind als solche kenntlich gemacht. Die Arbeit ist in gleicher oder ähnlicher Form im Rahmen einer anderen Prüfung noch nicht vorgelegt worden. Ich versichere, dass die eingereichte elektronische Fassung der eingereichten Druckfassung vollständig entspricht.

I hereby declare in lieu of an oath that this thesis is my own work and that I have not used any sources other than those listed in the bibliography. Content from published or unpublished works that has been quoted directly or indirectly or paraphrased is indicated as such. The thesis has not been submitted in the same or similar form for any other academic award. The electronic version I have submitted is completely identical to the hard copy version submitted.

Ort *Place* | Datum *Date* | Unterschrift *Signature*

Prüfungsamt
der Wirtschaftswissenschaftlichen Fakultät
Corrensstraße 1
48149 Münster

E-Mail: pam@wiwi.uni-muenster.de
pam.wiporecht@wiwi.uni-muenster.de
Telefon: 0251 – 83 37915

Version: 29.12.2020

WESTFÄLISCHE
WILHELMS-UNIVERSITÄT
MÜNSTER

Wirtschafts-
wissenschaftliche
Fakultät

Einverständniserklärung
zur Prüfung meiner Arbeit mit einer Software zur Erkennung von Plagiaten

Name: _____ **Vorname:** _____

Matrikelnummer: _____ **Studiengang:** _____

Adresse: _____

Titel der Arbeit: _____

Was ist ein Plagiat?
Als ein Plagiat wird eine Übernahme fremden Gedankengutes in die eigene Arbeit angesehen, bei der die Quelle, aus der die Übernahme erfolgt, nicht kenntlich gemacht wird. Es ist dabei unerheblich, ob z.B. fremde Texte wörtlich übernommen werden, nur Strukturen (z.B. argumentative Figuren oder Gliederungen) aus fremden Quellen entlehnt oder Texte aus einer Fremdsprache übersetzt werden.

Softwarebasierte Überprüfung
Alle Bachelor- und Masterarbeiten werden vom Prüfungsamt mit Hilfe einer entsprechenden Software auf Plagiate geprüft. Die Arbeit wird zum Zweck der Plagiatsüberprüfung an einen Software-Dienstleister übermittelt und dort auf Übereinstimmung mit anderen Quellen geprüft. Zum Zweck eines zukünftigen Abgleichs mit anderen Arbeiten wird die Arbeit dauerhaft in einer Datenbank gespeichert. Ein Abruf der Arbeit ist ausschließlich durch die Wirtschaftswissenschaftliche Fakultät der Westfälischen Wilhelms-Universität Münster möglich. Der Studierende erklärt sich damit einverstanden, dass allein zum beschriebenen Zweck der Plagiatsprüfung die Arbeit dauerhaft gespeichert und vervielfältigt werden darf. Das Ergebnis der elektronischen Plagiatsprüfung wird dem Erstgutachter mitgeteilt.

Sanktionen
Liegt ein Plagiat vor, ist dies ein Täuschungsversuch i.S. der Prüfungsordnung, durch den die Prüfungsleistung als „nicht bestanden“ gewertet wird. Es erfolgt eine Mitteilung an das Prüfungsamt und die dortige Dokumentation. In schwerwiegenden Täuschungsfällen kann der Prüfling von der Prüfung insgesamt ausgeschlossen werden. Dies kann unter Umständen die Exmatrikulation bedeuten. Plagiate können auch nach Abschluss des Prüfungsverfahrens und Verleihung des Hochschulgrades zum Entzug des erworbenen Grades führen.

Hiermit erkläre ich, dass ich die obigen Ausführungen gelesen habe und mit dem Verfahren zur Aufdeckung und Sanktionierung von Plagiaten einverstanden bin.

_____
Datum und Unterschrift des Studierenden