Interpretable Machine Learning: Predicting the Popularity of Online News by Integrating Econometrics and Machine Learning

Master Thesis in Econometrics and Management Science
Specialisation: Business Analytics and Quantitative Marketing

Marjolein de With **Student ID number: 483866**

Supervisor: dr. W. Wang **Second assessor: dr. A.A. Naghi**

Abstract

This thesis focuses on the predictive performance and interpretability of several hybrid models that combine econometrics and machine learning, compared to linear or logistic regression and random forests. Comparisons are made using the Online News Popularity data set by the website Mashable, a data set that contains information on different articles published by Mashable during a two-year period, and their number of shares. Additionally, I analyse eight extra data sets to strengthen the conclusions that are made. I discuss both regression and classification models. For regression, a random forest always outperforms the other models, but Multivariate Adaptive Regression Splines and a partially linear model also generally perform well. For classification, attribute selection using univariate trees or a fully grown tree often outperform the other models, sometimes even the random forest. The interpretability of the models mainly depend on their complexity. All models outperform linear regression in terms of predictive performance and random forest in terms of interpretability.

Erasmus University Rotterdam

Erasmus School of Economics

November 23, 2021

Contents

1 Introduction

During the last ten to fifteen years, studying the popularity of online news has become increasingly popular. Thurman and Fletcher (2018) discuss how several newspapers have decided to stop printing and go fully online. News articles, especially online news articles, are extremely time-sensitive. There has always been competition between different news providers to publish articles that are relevant to as many people as possible, and that could go viral. In the past, before the emergence of online news, publishing printed newspapers was costly and was therefore mainly done by large news agencies. However, due to the low cost of creating online news and the ease of sharing these online articles with others, the rules have changed (Bandari et al., 2012).

Due to the competition for attention between news different outlets, it has become very important to accurately predict the popularity of a news article. This is a challenging task for various reasons (Bandari et al., 2012). One reason is that, among others, the local and geographical conditions of the population make prediction complicated. A second reason is that the content of an article plays a crucial role in its popularity. If the content of an article is relevant for a large group of readers, it can be expected that this article attracts much attention.

There are two possibilities for predicting the popularity of online news (Bandari et al., 2012). It is, for example, possible to use the early popularity of an article to predict its success in the future. A more difficult method is predicting article popularity using only features of the article that were known before publication. In doing so, it is possible to try to maximise the expected popularity of an article, by optimising some features of the article that can easily be optimised.

This is not the first paper that tries to predict the popularity of online news. In Section 2.1, I discuss prior literature for this topic. A lot of researchers that deal with this topic, use either an econometric model or a machine learning model for their predictions. A (linear) econometric model often gives an insight into which factors drive the popularity of online news, but does not always give good predictions. A machine learning model, on the other hand, often gives good predictions, but its results are hard to interpret, as the model is not built for the purpose of interpretation (Mullainathan and Spiess, 2017). It is not suitable to look at the features used by a machine learning model, and then decide those features must be important. This has two reasons. The first reason is that machine learning models do not give standard errors, which complicates making inferences. The second, more important, reason, is that the features used by machine learning models can differ greatly, depending on how the data is partitioned into a train and a test set. This is because machine learning

models can fit a wide range of different functions, but this means that the probability of two very different functions having a similar predictive performance increases.

Therefore, I approach the problem in a slightly different way. Namely, I combine econometric models with machine learning models, such that I can obtain both the interpretability of an econometric model and the good predictions of a machine learning model. I evaluate the models I use for this paper on both their interpretability and their accuracy.

This thesis is divided into two parts. In the first part, I use different models, including models combining econometrics and machine learning, to predict the popularity of online news. To do this, I focus on the following research question:

How does a model that combines econometrics and machine learning perform compared to purely econometric or purely machine learning models in terms of predictive performance?

For this question, I compare both the out-of-sample and the in-sample fit using different evaluation criteria. In the second part of my thesis, I compare the models based on their interpretability. To do this, I formulate a second research question:

How does a model that combines econometrics and machine learning perform compared to purely econometric or purely machine learning models in terms of interpretability?

I evaluate my models on the popular Mashable data set, provided by the UCI Machine Learning Repository (Dua and Graff, 2021). This data set was first acquired and pre-processed by Fernandes et al. (2015). It summarises a set of features about articles published by Mashable over the period of two years. The data set contains 39,797 observations and has 61 attributes. The target variable is continuous, but multiple papers (e.g. Fernandes et al., 2015 and Bandari et al., 2012) use binary or multi-class classification, due to the high variance of the target variable. They classify the articles based on how often they were shared. As I do not want to lose valuable information by doing so, I evaluate if new models would yield better results. Afterwards, I also split the articles into two categories and apply classification models to examine their popularity. Then, I evaluate my models on eight different publicly available data sets, four for regression and four for classification. These data sets differ in the number of observations and the number of attributes. In contrast to these eight data sets, I discuss the Mashable data set in far more detail in Section 3.

For my thesis, I study several models, both for regression and classification. For the regression, I first evaluate a regular linear model and a random forest model, as benchmarks for interpretability and predictive performance respectively. I also extend the linear model to include lasso, again as a benchmark for interpretability. I include lasso because its interpretability is similar to that of linear regression, but its predictive performance is usually superior. Then, I evaluate a model tree, a type of decision tree that has linear equations at the leaves instead of values. The advantage of model trees over simple regression trees is

that they are generally much smaller, but give more accurate predictions (Quinlan, 1992). With this model, I hope to be able to rival the performance of the random forest, but with far better interpretability. A slightly different interpretation of combining econometrics and machine learning, is Multivariate Adaptive Regression Splines (MARS), a non-parametric regression technique developed by Friedman (1991). It can be viewed as an extension of linear models, that automatically models non-linearities and interactions between features. I also combine econometrics and machine learning in a third way. Namely, I construct a partially linear model, which is a form of a semiparametric model, as it contains both parametric and non-parametric elements. I evaluate all attributes to see which ones have the highest feature importance for the random forest. These attributes are taken into account in a non-parametric way. My next approach for regression is using decision trees to decide on what attributes to take into account for linear regression. Sarma (2005) and Millard (2004) both approached this in a slightly different way, by using the outcomes of regression trees as inputs for linear regression.

For the classification, I first evaluate a logistic regression and lasso model as benchmarks for interpretability. Second, similar to the regression, I evaluate a random forest model, as a benchmark for predictive performance. Third, I evaluate a *logistic model tree*, which is a tree with logistic regression in the leaves (Landwehr et al., 2003). This tree works similar to a model tree. Then, I extend the MARS technique such that it can handle classification. My fifth approach for classification is using the outcomes of classification trees as inputs for logistic regression.

The remainder of this thesis is structured as follows. In Section 2, I discuss previous literature on the topic of online news popularity, combining econometrics and machine learning, and interpretability. In Section 3, I evaluate my data set and give some descriptive statistics. In Section 4, I give a more detailed explanation of the models I use for this paper. Section 5 discusses the performance and interpretability of these models, and I conclude my research in Section 6.

2 Literature

In Section 1, I mentioned some articles that study the popularity of online news. In Section 2.1, I discuss these articles further. Furthermore, I discuss prior literature on combining econometrics and machine learning in Section 2.2. Finally, in Section 2.3, I discuss interpretability in general.

2.1 Popularity of Online News

Fernandes et al. (2015) first used the Mashable data set to predict the popularity of online news. They proposed a novel Intelligence Decision Support System (IDSS) that analyses articles before they are published. The authors only predicted the popularity using classification models and did not predict the continuous number of shares. They evaluated a random forest (RF), Adaptive Boosting (AdaBoost), a support vector machine (SVM), k-nearest neighbours (KNN) and naive Bayes (NB). Using the random forest, they achieved a ROC-AUC score of 0.73.

Bandari et al. (2012) examined both regression and classification models to predict online news popularity. They collected a set of articles via a news feed aggregator and then found the number of times each of these articles was linked to on Twitter. The authors used linear regression (LR), KNN regression and SVM regression to model the number of tweets per article, where LR performed best. For the classification part, the authors divided the articles into three groups, based on how often they were linked to on Twitter. The authors used Bagging, J48 Decision Trees (an algorithm used to generate the model trees I also discuss), SVM and NB to predict these groups, and achieved an overall accuracy of 84%.

A third paper that discusses the modelling and predicting of online news popularity, is the paper written by Hensinger et al. (2013). They proposed to view popularity as a competitive situation where the popular articles are those that were the most appealing to readers on a particular day. They constructed an 'appeal' function and used only the words found in the article and its title as attributes. Later, they also added tags of the overall topic of an article. First, they used an SVM classifier to predict whether articles were 'Popular' or 'Non-Popular', but this only gave an average accuracy of 55.83% over multiple outlets. Therefore, they modelled the popularity in a relative way instead, as the concept of 'more popular than'. They did this using Ranking SVMs. This method could predict successfully which articles users would prefer in a choice task with two articles, with an accuracy of up to 85.74%.

2.2 Combining Econometrics and Machine Learning

I also discuss some previous papers that attempted to combine econometrics and machine learning. Malhotra (2018) proposed a hybrid approach based on conventional econometrics and advanced machine learning algorithms and used an example of food inflation in India to show its performance. He addressed the issue of finding a measure of the relative importance of independent features in econometric models, as standardised regression coefficients and zero-order correlations had failed in the past. The hybrid approach consists of three steps: in step one, statistically significant features are identified through conventional econometric techniques. In step two, an exploratory model is constructed using machine learning algorithms, with the features identified in step one. In step three, feature importance modules of the used machine learning algorithms are used to give a measure of the relative importance of the features. Malhotra (2018) showed that this hybrid approach outperforms normal econometric models when it comes to finding a measure of relative importance, at least in the case of multicollinearity.

Guerzoni et al. (2021) combined econometrics and machine learning to measure innovation. They created a measure of innovation using an Italian law aimed at boosting new high-tech firms that came into effect in 2012. They analysed the impact of innovativeness on a group of Italian firms that were created in 2008. The authors combined seven learning algorithms to recognise innovative firms on data from 2013 and then applied the model to the 2008 data set to see which firms would have been labelled as innovative according to the 2012 law. They used this indicator in a survival model to explain the ability of firms to stay in the market after 2008, throughout the financial crisis. They concluded that innovative firms are more likely to survive than non-innovative firms.

Dumitrescu et al. (2021) introduced penalised logistic tree regression (PLTR) to obtain the best of both worlds: better classification performance and interpretability, in the context of credit scoring. In the credit risk industry, logistic regression is often the benchmark, as financial regulators often require certain interpretability. Rules are extracted using decision trees with a short depth and are then used as predictors in a penalised logistic regression. Dumitrescu et al. (2021) applied the resulting model to Monte Carlo simulations and four empirical applications and concluded that PLTR significantly outperforms normal logistic regression and competitively compares to random forests.

Finally, an important work is the paper by Farrell et al. (2021), who studied neural networks and discussed how they can be used in semiparametric inference. They provided the first inference results using deep learning methods, focusing on causal effects as a concrete illustration. They explored how to use those methods as tools for economic applications. In their paper, the authors proved that inference can be valid after using deep-learning methods

for first-step estimation. The findings described in the paper can be used to incorporate deep neural networks into standard econometric models. This paper is like my paper in the sense that it combines more complex methods that have good predictive power with standard econometric models. Where I focus on tree-based methods, this paper discusses deep learning methods. Additionally, where this paper comes up with a novel way of combining econometrics and machine learning, I compare and discuss existing ways, and add a new twist to an existing model.

Other examples of researches combining econometrics and machine learning include Van der Voort et al. (1996), who combined ARIMA with neural networks to short-term forecast traffic flow, Lehrer and Xie (2018), who combined support vector regression (SVR) with simple econometric models to capture more heterogeneity in data from the film industry, Hirano and Wright (2017), who proposed a split-sample method where they split the data set into two parts, one for model selection and one for model estimation, and added a bagging step to substantially improve prediction performance, and Li et al. (2019), who proposed a hybrid of a logistic regression model and an extraction algorithm that is built using machine learning, to detect when the blades of wind turbines are iced over.

2.3 Interpretability

Since machine learning has been used for complex problems, researchers have not only been interested in the performance of different models, but also in criteria such as non-discrimination or providing the right to explanation (Doshi-Velez and Kim, 2017), often called *auxiliary criteria*. For example, if a bank predicts that a borrower will not be able to repay a loan, and thus refuses her, the borrower has the right to know why. Additionally, this decision cannot be based on certain discriminatory grounds, such as the ethnicity, gender or age of a person. In cases like this, but also for numerous others, interpretation is an important factor. If the reasoning of a certain model can be explained, it can also be judged whether this reasoning meets the auxiliary criteria (Doshi-Velez and Kim, 2017).

Unfortunately, different authors disagree on what exactly interpretability is, and how to evaluate it (Doshi-Velez and Kim, 2017). There is no mathematical definition for it. To some extent, interpretability is intuitive: if people understand a model, it is interpretable. However, several questions arise. If, say, a model class, such as decision trees, is deemed interpretable, are all models within the class equally interpretable? Do all applications require the same level of interpretability? Miller (2019) provided the following definition: "the degree to which an observer can understand the cause of a decision" (p. 8). Lipton (2018) stated that a model is interpretable if it is "simple enough to be examined all at once by a human" (p. 36). For this, an understanding of the features, weights, structures and other parameters is

required. The observer must have knowledge about which features are important, and how they interact with other features. Interpretability according to this definition is extremely difficult to achieve in practice, as humans cannot conceive a model with more than three dimensions. It is often easier to understand parts of a model (Molnar, 2019). For example, the weights in a linear regression model, or the splits of a decision tree. For individual predictions, the target value often depends on the different features in a less complicated way.

There are several reasons why people would ask for an explanation. Doshi-Velez and Kim (2017) named curiosity and learning. People look for explanations to improve their understanding, such that they can make better, more stable models. People often ask questions about observations they consider unusual or unexpected. Another reason, named by Miller (2019), is the human desire to find meaning in things, both an individual meaning as well as a shared meaning. If an intelligent agent explains something, it creates a shared understanding of the decision she made. Molnar (2019) named some other reasons, for example, many qualitative fields, such as sociology and psychology, are turning more and more to quantitative methods. Their source of knowledge is not the data itself anymore, but the model it has created. Furthermore, being able to interpret models correctly helps in ensuring no errors have been made in either collecting data or building the model.

For some applications, such as a Netflix recommendation system, it does not matter why a model predicts something, as long as the predictive performance is good enough (Molnar, 2019). For example, this is the case when a model does not have a significant impact, whether it be social or financial. A second case is when a problem is well-studied already, and experience has shown it works. Third, interpretability could result in people manipulating the system. To get back to the bank example from earlier in this section, a borrower might be able to make short-term adjustments that significantly influence the probability the model provides that she will be able to repay the loan, even if the actual possibility does not change.

Models can be interpretable in two ways (Molnar, 2019). They can be intrinsically interpretable if they are not too complex. Second, a model can be post hoc interpretable, if it is analysed after being trained. Intrinsically interpretable models are, for example, shallow decision trees or linear regression models with not too many features. An example of a method used for post hoc interpretation is feature importance analysis. In my thesis, I focus on intrinsically interpretable models. Such models might need to sacrifice some prediction accuracy, while post hoc interpretable models keep the accuracy intact, but are more limited when it comes to interpretation (Du et al., 2019).

Intrinsic interpretability can also be further differentiated, namely, into global interpretability and local interpretability (Du et al., 2019). Global interpretability is about un-

derstanding how a model makes its decisions. It examines which features play a big role in predicting and which are useless. Local interpretability is about understanding individual predictions of the model. If a potential borrower at a bank needs to be told 'no', her individual situation needs to be taken into account for the explanation.

Post hoc interpretability can be divided into model-specific explanation and model-agnostic explanation (Du et al., 2019). Model-agnostic explanation treats models like black boxes, complex systems with hidden inner workings. It can be applied to any model. An example is permutation feature importance (PFI) (Molnar et al., 2021). PFI states that features that play a big role in the prediction must contain valuable information. Therefore, if the feature values are randomly shuffled, this information is destroyed, and the prediction accuracy should get worse. If this decrease is small, the original information was apparently not very impactful, but if the decrease is large, it was. Model-specific explanation examines the internal structures and parameters of a model. For tree-based models, specifically tree-based ensemble models, an approach is calculating the accuracy gain after adding a feature. Another approach measures the coverage of a feature, namely, the proportion of observations that are related to this feature. A third approach is counting the number of times that a feature is used to split the data.

In my thesis, I approach the topic of interpretability by examining several models that integrate econometrics and machine learning, both for regression and classification. These models are intrinsically interpretable. I discuss their interpretation in detail.

3 Data

In this section, I discuss my data sets. I discuss the Mashable data set in Section 3.1, and the other data sets in Section 3.2.

3.1 Mashable data set

As I have stated before in Section 1, my main data set is the Mashable data set on online news popularity. Fernandes et al. (2015) collected this data set, which covers the time period from 7 January 2013 until 7 January 2015. This data set covers 39,644 articles. Fernandes et al. (2015) deleted some special occasion articles, as they did not follow the general structure. Additionally, they discarded very recent articles, as the number of shares did not always reach convergence yet. As they worked with a rolling window approach, they wanted to keep a constant number of articles per test set. For my research, I follow the same approach. I also delete some articles for which the feature *Number of words in article* equals 0, as those articles do not contain any content. Therefore, I end up with 37,000 articles.

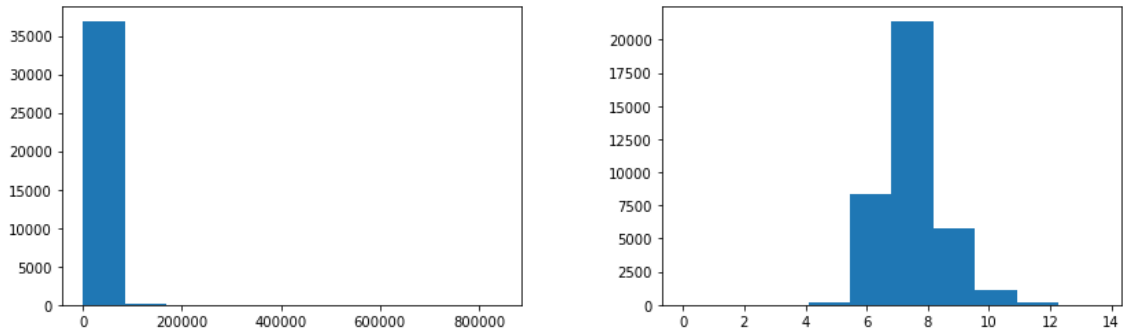
Table 1: Features of Mashable data set

Feature	Type (#)	Feature	Type (#)
Words		**Keywords**	
Number of words in title	number (1)	Number of keywords	number (1)
Number of words in article	number (1)	Worst keyword (min./avg./max. shares)	number (3)
Average word length	number (1)	Median keyword (min./avg./max. shares)	number (3)
Rate of non-stop words	ratio (1)	Best keyword (min./avg./max. shares)	number (3)
Rate of unique words	ratio (1)	Article category	nominal (1)
Rate of unique non-stop words	ratio (1)	**Natural Language Processing**	
Links		Closeness to top 5 LDA topics	ratio (5)
Number of links	number (1)	Title subjectivity	ratio (1)
Number of Mashable article links	number (1)	Article text subjectivity score + absolute difference to 0.5	ratio (2)
Number of shares of Mashable links (min./avg./max.)	number (3)	Title sentiment polarity	ratio (1)
Digital Media		Rate of positive / negative words	ratio (2)
Number of images	number (1)	Positive words rate among non-neutral words	ratio (1)
Number of videos	number (1)	Negative words rate among non-neutral words	ratio (1)
Time		Polarity of pos. words (min./avg./max.)	ratio (3)
Day of the week	nominal (1)	Polarity of neg. words (min./avg./max.)	ratio (3)
Published on a weekend?	bool (1)	Article text polarity score + absolute difference to 0.5	ratio (2)

Fernandes et al. (2015) extracted 47 features from the HTML code to make the data suitable for learning models. For one of these features, namely *Rate of non-stop words*, all values equal 1 after removing the observations for which the feature *Number of words in*

article equals 0. Therefore, I do not take this feature into account. I make dummies for the features *Day of the week* and *Article category*. To avoid multicollinearity, I drop the dummy for the weekend, the dummy for Tuesday and the dummy for the first LDA topic. Following the approach of Fernandes et al. (2015), I scale some unbounded numerical features using a logarithmic transformation, and cap others at the 0.9th quantile. Finally, I standardise all variables. The different features the authors collected describe different aspects of the articles, that they considered relevant to predict the number of shares. For example, they looked at the number of links in the articles, specifically the number of links to other Mashable articles, and also took into account the minimum, average and maximum number of shares for these links per article. Fernandes et al. (2015) made a ranking of all possible keywords in the articles. For each article, they selected the worst keyword and add the minimum, average and maximum shares of articles containing this keyword as features. They did the same for the best keyword and the median keyword. They also added dummies for the categories of the article. These categories are *lifestyle*, *business*, *entertainment*, *social media*, *tech*, *viral* and *world*. Among the 47 features, some are related to *Natural Language Processing*. The *Latent Dirichlet Algorithm* (LDA), proposed by Pritchard et al. (2000) and first applied in machine learning by Blei et al. (2003), has been applied to all Mashable articles in the data set, to identify the five most relevant topics and then measure the closeness of all articles to each of these topics. Finally, Fernandes et al. (2015) used the Pattern web mining module (De Smedt et al., 2014) to compute scores for sentiment polarity and subjectivity. All features, divided into several categories, can be found in Table 1.

For the total number of shares, Fernandes et al. (2015) considered Facebook, Twitter, Google+, LinkedIn, StumbleUpon and Pinterest. The number of times articles were shared ranges from 1 to 843,300, with a median of 1400 and an interquartile range of 1854. This median is also the cut-off value Fernandes et al. (2015) used when they constructed the classes 'popular' and 'not popular'. A histogram of the number of shares can be found in Figure 1. Figure 1a shows the total number of shares for the different articles in the data set, Figure 1b the log of the total number of shares for the different articles in the data set. It is clear from Figure 1a that a log transformation is appropriate in this case, as the data is heavily positively skewed.



(a) Histogram of the number of shares of Mashable articles

(b) Histogram of the log of the number of shares of Mashable articles

Figure 1: Histograms of the number of shares of Mashable articles

Some descriptive statistics of the data set are given in Table 2.

Table 2: Descriptive statistics for the data set

Mean	Median	Standard deviation	Minimum	Maximum
3364.5	1400.0	11693.1	1	843300

3.2 Extra Data Sets

As mentioned in Section 1, I also examine eight other data sets, four of which are for regression and four for classification. All of these are provided by the UCI Machine Learning Repository (Dua and Graff, 2021) or Kaggle (Kaggle, 2021). Doing so allows me to draw more reliable conclusions about the performances of the different models that I examine for this thesis, as they are backed by more than one data set. These data sets are adjusted as necessary. For example, observations with missing values, or columns for which a lot of observations have a missing value, are simply removed, as the main focus of this thesis is not on these data sets. Instead, the data sets are only meant to solidify my conclusions. I do not examine them in detail. However, I do want to give a short description in this section.

Table 3: Short descriptions of extra data sets

Data set				
Regression	**Short Explanation**	**Number of observations**	**Number of attributes**	**Citation**
Fish Market	Data set to predict the weights for seven common fish species	159	11	
Medical Cost Personal Datasets	Data set to predict health insurance costs based on medical information insurance costs	1338	12	Inspired by Lantz (2013)
Bike Sharing Demand	Data set with information on hourly rental data of bikes over a period of two years	17,379	54	Fanaee-T and Gama (2013)
Communities and Crime	Communities within the United States. Data set that combines socio-economic data from the 1990 US Census, law enforcement data from the 1990 US LEMAS survey, and crime data from the 1995 FBI UCR	1993	102	Redmond and Baveja (2002)
Classification	**Short Explanation**	**Number of observations**	**Number of attributes**	**Cite**
Titanic	Data set to predict whether people survived the sinking of the Titanic or not measurements	714	9	
Bank Marketing	Data set related with direct marketing campaigns of a Portuguese bank to predict whether a client will subscribe a term deposit	9042	70	Moro et al., 2014
Adult Income	Data set that is used to predict whether individuals earn more or less than 50K	4522	92	
Job Change of Data Scientists	Data set that is used to predict whether people really want to work at a job or whether they just want free training	19,158	54	

In Appendix 1, Figure A1, histograms for these data sets can be found.

4 Methodology

In this section, I explain the models I use for my thesis. As discussed in Section 1, I approach the prediction of online news popularity as both a regression and a classification task. For the regression task, I directly predict the number of shares for an online article. Then, I divide the articles into two groups in two different ways. First, I predict whether articles are 'popular', or 'not popular'. Second, whether articles are 'not very popular' or 'very popular'. Then, I use the best model for all of the three tasks to find a way to maximise the expected popularity of news articles.

In Section 4.1, I discuss the regression models, and in Section 4.2 the classification models. I assume the reader is already familiar with the benchmark models, so I do not discuss those.

4.1 Regression Models

In this section, I discuss the different hybrid regression techniques that I use for evaluating my different data sets.

Model Tree

Learning algorithms that construct decision trees are often efficient, robust and relatively simple (Quinlan, 1992). Many algorithms that construct trees have been generated, with varying predictive performance and interpretability. Such algorithms include simple regression trees and random forests, but also boosting and bagging trees. In particular boosting algorithms XGBoost (Chen and Guestrin, 2016) and LightGBM (Ke et al., 2017) are known to perform extremely well, but it is practically impossible to interpret them. Therefore, I discuss model trees, which is a method that integrates econometrics and machine learning, a combination that I evaluate to answer my first research question. Quinlan (1992) first came up with the idea for this model. His paper describes M5, a system for learning models that predict values. M5 builds tree-based models, but the trees constructed by the system can have multivariate linear models at their leaves, instead of values. Therefore, model trees are the same as piece-wise linear functions. Model trees are usually smaller than regular regression trees and have proven more accurate for several tasks. Moreover, M5 is an efficient learner and can handle tasks with hundreds of features (Quinlan, 1992). These factors would make M5 models a very attractive alternative to regular trees. A last major difference between regular regression trees and model trees is the fact that model trees are able to give an expected value that lies outside the range that was observed in the train set, such that it is able to extrapolate. This could either be an advantage or a disadvantage. Another algorithm for building model trees is J48. J48 was used by Bandari et al. (2012), who also used it to

predict online news popularity. I discussed their paper in Section 2.1. In the remainder of this section, I describe how M5 model trees are constructed.

First, I have N observations, with a fixed set of features and an associated numeric target value. A set of observations L is always either associated with a leaf node $t \in T$, where T is the set of leaf nodes, or it is split into two subsets corresponding to its target values. This process is applied again to the resulting subsets. Repeating this a large number of times often results in an overcomplicated tree, that overfits the model. Therefore, the tree needs to be pruned back, meaning that subtrees are exchanged for leaves.

The M5 algorithm starts splitting the tree by computing the standard deviation of the target values in the full train set. Then, multiple tests are evaluated. For each test, L_i is the subset of observations that would have outcome i for the test, for $i \in \{0, 1\}$. The standard deviation of these observations, $\text{sd}(L_i)$, is our measure of error. The expected reduction that results from splitting the set of observations according to the test, is

$$\Delta\text{error} = \text{sd}(L) - \sum_i \frac{|L_i|}{|L|} \cdot \text{sd}(L_i). \tag{1}$$

If there are only a few observations left in a set, or if the difference between them is very small, the set is not split into subsets. After every possible test has been examined, the test that maximises (1) is selected. In doing so, M5 deviates from the CART (Classification and Regression Trees) algorithm, as that algorithm tries to maximise the reduction in variance or absolute deviation.

An M5 model tree is constructed using a train set of observations, but it also needs to estimate the model accuracy for unseen observations. To do this, first, the average absolute residual of the model on the train set of observations is derived. Then, as the error on unseen observations is often higher, this value is multiplied by $\frac{n+v}{n-v}$, such that a model with a small number of observations but a large number of features has an increased estimated error.

For each node in the model tree, a multivariate linear model is constructed, using only features that have been referenced before in the subtree of that node. This is done such that a fair comparison can be drawn between the accuracy of the linear model and the accuracy of the subtree.

After the linear models have been constructed for every node, they are simplified by removing features to minimise the estimated error, such that for every leaf, only a subset $V_t \subset V$ of all possible features is taken into account. Doing so generally causes an increase in the average residual, but it also decreases the multiplicative factor $\frac{n+v}{n-v}$, such that the estimated error decreases. Features that do not sufficiently contribute to the model are eliminated using a greedy search.

Subsequently, starting at the bottom, all nodes of the model tree that are not leaves are examined. The M5 algorithm evaluates whether the simplified multivariate linear model or the subtree has a lower estimated error. The subtree at the node is pruned to a leaf in case the linear model is chosen.

Standalone linear regression and ordinary regression trees are special cases of model trees. Linear regression can be seen as a model tree that is completely pruned back to its root, ordinary regression trees as model trees for which $V_t = \emptyset, \forall t \in T$. This makes it possible for model trees to adapt to the data set that is being evaluated: for small data sets that do not exhibit complicated non-linear relationships, standalone linear regression offers the best bias-variance trade-off, while an elaborate tree structure might be more appropriate for more complicated data sets. This is the same logic as for the pruned back subtrees within the tree.

Finally, the model tree is smoothed, as this can improve its prediction accuracy. This means that the values at the leaves of the model tree are adjusted to reflect the predicted values of the nodes that are along the path of the root to the leaf. This works as follows: first, the predicted value at the leaves is computed by the model at that leaf. Then, if an observation follows branch S_i of subtree S, its predicted value backed up to S is

$$PV(S) = \frac{n_i \cdot PV(S_i) + k \cdot M(S)}{n_i + k}, \tag{2}$$

where n_i is the number of observations for branch S_i, $PV(S_i)$ the predicted value for S_i, $M(S)$ the value that the model gives at S and k a smoothing constant that is equal to 15 as default. Smoothing mainly has an effect when either the models along a path give all very different predictions, or if some models were constructed using only a few training cases.

Multivariate Adaptive Regression Splines

Multivariate Adaptive Regression Splines (MARS) is an algorithm designed for multivariate non-linear regression problems. The reason why MARS is included in this research is that it uses a similar process to find the best splits in the data as regression trees. Regression trees take into account the deviation from the mean on both sides of the split, and MARS takes into account the deviation from a spline function on both sides of the split (Friedman, 1991; Hastie et al., 2009). In this section, I explain how MARS works.

Piecewise linear basis functions play an important role in MARS models. Such functions take the form

$$(x - t)_+ = \begin{cases} x - t & x > t, \\ 0 & \text{otherwise}, \end{cases} \tag{3}$$

or

$$(t-x)_+ = \begin{cases} t-x & x < t, \\ 0 & \text{otherwise.} \end{cases} \tag{4}$$

These functions have a *knot* at value t. For each feature X_j, there is a combination of these functions, also called a *reflected pair*, with a knot at each observation x_{ij}. This leads to the following collection:

$$\mathcal{C} = \{(X_j - t)_+, (t - X_j)_+\}, \quad t \in \{x_{1j}, ..., x_{Nj}\}, \quad j = \{1, ..., p\} \tag{5}$$

Then, the model is built like a forward stepwise linear regression. However, instead of the original features, functions from set $\mathcal{C}$ and their products are used. This gives the following model:

$$Y = \beta_0 + \sum_{m=1}^{M} \beta_m h_m(X), \tag{6}$$

where each $h_m(X)$ is either a function in $\mathcal{C}$ or the product of two or more functions in $\mathcal{C}$.

Given h_m, the β_m are estimated by standard linear regression. The functions $h_m(x)$ are constructed as follows: first, the model only contains the constant function $h_0(x) = 1$. I denote the model set that contains the terms of the model by $\mathcal{M}$. Then, at each stage, all products of a function h_m with one of the reflected pairs in $\mathcal{C}$ are considered. Subsequently, the following term is added to $\mathcal{M}$:

$$\hat{\beta}_{M+1} h_\ell(X) \cdot (X_j - t)_+ + \hat{\beta}_{M+2} h_\ell(X) \cdot (t - X_j)_+, \quad h_\ell \in \mathcal{M}, \tag{7}$$

for feature j that gives the largest decrease in the training error. This process is continued until $\mathcal{M}$ contains a predetermined number of terms. As (6) often overfits the data, a backwards deletion procedure is applied. To do this, at each stage, the term for which the removal causes the smallest increase in the residual sum of squares (RSS), is deleted. To determine the optimal model size λ, generalised cross-validation (GCV) is used. This is defined as

$$GCV(\lambda) = \frac{\sum_{i=1}^{n} (y_i - \hat{f}_\lambda(x_i))^2}{(1 - M(\lambda)/N)^2}, \tag{8}$$

where $\hat{f}_\lambda$ is the best model for size λ, and $M(\lambda)$ denotes the effective number of features in the model, which is the number of terms in the model plus the number of parameters that are used in selecting the optimal position of the knots. I choose the model that minimises (8).

As a result of this strategy, non-zero components are only used locally, where they are needed. Therefore, MARS is very suitable for high-dimensional problems.

As the implementation of MARS in Python, which they call *Earth* due to copyright issues, can be slow for some data sets, I set the parameter *use_fast* in those cases equal to *True*. This fast implementation is described by Friedman (1993).

Partially Linear Model

A third method of combining econometrics and machine learning is by using a partially linear model (PLM). A PLM is a form of a semiparametric model, like the model described by Farrell et al. (2021). A semiparametric model is a model with both parametric and non-parametric components. A PLM is given by

$$Y = X\beta + f(U) + \varepsilon, \tag{9}$$

where Y is the dependent variable, X a vector of explanatory variables, β a vector of unknown features and the non-parametric part is given by the unknown function $f(U)$. ε is the error term. This type of model was proposed by Robinson (1988). As it allows only some variables of a model to be modelled in a non-parametric way, the interpretation of the model remains mostly intact, while allowing some flexibility into the model. By choosing the variables that are modelled in a non-parametric way well, I can increase the performance of the model while only giving up a fraction of its interpretability. Several approaches for solving the model have been studied, for example by penalised least squares (Wahba, 1984) or kernel smoothing (Speckman, 1988). Yatchew (1997) and Wang et al. (2011) approached the estimation of this model by first ordering the observations based on their values for the non-parametric part and then estimating the linear component based on the differences of the observations and estimating the non-parametric component using the residuals of the linear fit. This estimation procedure is optimal, as the estimator $\hat{\beta}$ is asymptotically efficient and the estimator for the non-parametric component is minimax rate optimal. Wang et al. (2011) extend the work of Yatchew (1997) by discussing the estimation of not only the linear but also the non-parametric component and by making some smoothness assumptions. As the non-parametric part needs to be continuous, I opt to model one or two features using a model tree. To the best of my knowledge, combining linear regression and a tree-based method like this is new to the literature. A model tree is very suitable in this situation, as the tree only consists of a root node with a linear equation in case it is not appropriate to model some features in a non-parametric way for a certain data set. I decide on the non-parametric features by looking at the feature importances of a random forest and selecting one or two of the three variables with the highest feature importance to be non-parametric. I evaluate those and

select the non-parametric set with the best in-sample performance. In this section, I explain the procedure in more detail.

First, the observations are ordered according to their values for $U_i, i = 1, ..., n$. Then, $d_t, t = 1, ..., m + 1$ is a difference sequence of order m that needs to satisfy $\sum_t d_t = 0$ and $\sum d_t^2 = 1$. D_i is the mth order difference of Y_i, such that $D_i = \sum_{t=1}^{m+1} d_t Y_{i+m+1-t}$ for $i = 1, ..., n - m - 1$. Written differently,

$$D_i = Z_i'\beta + \delta_i + w_i, i = 1, ..., n - m - 1, \tag{10}$$

where $Z_i = \sum_{t=1}^{m+1} d_t X_{i+m+1-t}$, $\delta_i = \sum_{t=1}^{m+1} d_t f(U_{i+m+1-t})$ and $w_i = \sum_{t=1}^{m+1} d_t \varepsilon_{i+m+1-t}$. Written in matrix form, this becomes

$$D = Z\beta + \delta + w. \tag{11}$$

This step eliminates the effect of the non-parametric component. An example of a difference sequence that satisfies the conditions is

$$d_1 = \sqrt{\frac{m}{m+1}}, d_2 = d_3 = ... = d_{m+1} = -\sqrt{\frac{1}{m(m+1)}}. \tag{12}$$

Then, I apply ordinary least squares on the differences to estimate β, and estimate

$$\hat{\beta} = (Z'Z)^{-1}Z'D. \tag{13}$$

Here, the correlation among the w_i should be ignored. The estimates are asymptotically normally distributed with heteroscedastic standard errors. $\hat{\beta}$ is asymptotically efficient if both f and $E(X'|Z)$ have bounded first derivatives. The efficiency relative to the method by Robinson (1988) is $(1+\frac{1}{2m})^{-1}$ (Yatchew, 1997). Hall et al. (1990) have tabulated the optimal differencing weights up to $m = 10$, giving an asymptotic efficiency of 95%. $\hat{\beta}$ can be plugged back into the original model. This yields the residuals

$$r_i = Y_i - X_i'\hat{\beta} = f(U_i) + X_i'(\beta - \hat{\beta}) + \varepsilon_i, i = 1, ..., n. \tag{14}$$

Then, f can be estimated based on r_i.

I order the observations as follows when I model two observations in a non-parametric way: I start with a random observation and select 100 other observations at random. I compute the Euclidean distance from each of those observations to the first observation and select the observation that has the shortest Euclidean distance. This observation is second in the new list. Then, I repeat this process, where the observation selected in the previous round is compared to 100 new observations. I stop when I have a new, ordered, list that

contains all observations. I chose this method as it is easy to implement and relatively quick.

Attribute Selection Using Regression Trees

The last regression approach works differently from the other ones. Namely, I decide on attributes to take into account for linear regression using decision trees, similar to the approach of Dumitrescu et al. (2021), whose paper I discussed in Section 2. I approach attribute selection in two ways, which are briefly discussed below.

Model 1: Univariate Trees

Sarma (2005) constructed several univariate trees, one for each attribute, and used these splits as categorical attributes for logistic regression. I do the same for linear regression, and take a dummy for the optimal split into account for each feature. Furthermore, I also add the feature in its original form.

Model 2: Fully Grown Tree

Millard (2004) constructed a fully grown decision tree, and then treated each leaf of that tree as a dummy variable, taking value 1 if an observation ends up in that leaf. I take these dummy variables into account as inputs for a logistic regression model, in addition to the original features.

Both Sarma (2005) and Millard (2004) examined their methods for classification, but I evaluate them for regression models as well.

4.2 Classification Models

In this section, I discuss the different classification techniques that I use for evaluating my different data sets.

Logistic Model Tree

Landwehr et al. (2003) came up with logistic model trees, that are similar to normal model trees, but have logistic regression models instead of linear regression, such that they can be used for classification. Instead of binary values, logistic model trees produce explicit probability estimates, such that the logistic model tree could be used for the optimisation of the expected popularity of online news articles. Experiments by Landwehr et al. (2003) show that the logistic model tree is competitive with boosted decision trees in terms of predictive performance, but again, is far easier to interpret. As logistic model trees work similarly to normal model trees, I do not explain them in much detail, but only highlight where they differ from normal model trees.

If S is the whole instance space, spanned by all the available features, a tree structure divides S into regions S_i, where

$$S = \cup_{i \in T} S_t, \quad S_t \cap S_{t'} \quad \text{for} \quad t \neq t', \tag{15}$$

where T is the set of leaf nodes. The leaves $t \in T$ do not have a class label, but a logistic regression function f_t. The whole logistic model tree can then be represented by

$$f(x) = \sum_{t \in T} f_t(x) \cdot I(x \in S_t). \tag{16}$$

Similar as for normal model trees, standalone logistic regression and ordinary classification trees are special cases of the logistic model tree: logistic regression can be seen as a pruned back logistic model tree, and ordinary classification trees as logistic model trees with $V_t = \emptyset, \forall t \in T$, where $V_t \in V$ is the subset of features taken into account for leaf t.

The approach to building a logistic model tree differs slightly from the approach to building a normal model tree. Instead of first building a standard decision tree, building a logistic regression model at each node and then pruning the tree back, the logistic regression functions are constructed by "incrementally refining logistic models already fit at higher levels in the tree" (Landwehr et al., 2003, p. 176). Since a logistic regression model is fit at each node, regardless of whether it is an inner node or a leaf node, it is reasonable to use parent nodes as a basis for fitting the logistic regression at a child node, as the features of the model at the parent node already encode a 'global' influence of the features taken into account on the target variable. The model can then be refined by also taking into account the features that are valid for the set of observations in the child node. This can be done naturally using LogitBoost: a boosting algorithm formulated by Friedman et al. (2000). LogitBoost is a convex optimisation problem. Given that I want an additive model of the form

$$\sum_t a_t h_t, \tag{17}$$

LogitBoost minimises the logistic loss:

$$\sum_i \log(1 + e^{-y_i f(x_i)}). \tag{18}$$

This means that a logistic model tree is constructed as follows: for each node, the class membership probabilities can be modelled as

$$P(G = j | X = x) = \frac{e^{F_j^n(x)}}{e^{F_0^n(x)} + e^{F_1^N(x)}}, \tag{19}$$

where $j \in \{0, 1\}$, n is the number of the node and

$$F_j^n(x) = \alpha_0^j + \sum_{v \in V_t} \alpha_v^j \cdot v. \tag{20}$$

First, LogitBoost is run on the full set of observations, including more and more features to the model by adding simple linear regressions f_{mj} to the F_j^n from (20). When adding more features does not increase the model accuracy, the set of observations is split such that the logistic models are refined further in the subdivisions, which might give a better model. This split is by default based on which split would give the highest normalised information gain, which is a synonym for the Kullback-Leibler divergence.

The LogitBoost approach makes it more computationally efficient to build logistic models at the lower levels at the tree, compared to building them from scratch. Child nodes are split until a stopping criterion is met. The growing stops when the node contains less than fifteen observations or when a minimum information gain is not achieved. After the tree has been built, it is pruned using CART-based pruning, a method that uses a combination of the training error and a penalty term for the model complexity to make pruning decisions.

Unfortunately, the implementation of LMT in Python does not follow the exact same steps as Landwehr et al. (2003), but rather the steps described in Section 4.1, but then adjusted for logistic regression. As writing my own implementation is beyond the scope of this thesis, I decided to use the more simple implementation.

Multivariate Adaptive Regression Splines for Classification

Leathwick et al. (2005) describe a way to extend the technique described in Section 4.1 such that it can handle classification. They approach it as follows: first, they fit binary target variable Y on X using MARS, giving fitted values $\hat{Y}$. Then, the basis functions are extracted and used as input for a normal logistic regression model, such that the prediction values are constrained between 0 and 1. For the logistic regression model, I apply lasso regularisation.

Attribute Selection Using Classification Trees

The fifth classification approach works similar to the approach in Section 4.1. Instead of using decision trees to decide on attributes to take into account for linear regression, I use them to decide on attributes to take into account for logistic regression. Again, I follow the approach of Sarma (2005) and Millard (2004).

5 Results

In this section, I present the results I obtained using my models. In Section 5.1, I discuss the results for the regression models and in Section 5.2, I analyse the classification models. In Section 5.3, I discuss the interpretability of all the models, using the Mashable data set as an example. I used Python to obtain the results.

5.1 Regression Models

I evaluate all regression models based on their *mean squared error* (MSE), *relative error* and *correlation*. The relative error is the variance of the absolute values of the residuals of the predicted values divided by the variance of the true values, and the correlation is Pearson's correlation coefficient of the predicted values and the true values. I determine the optimal hyperparameters for the different models according to the MSE. For the MSE, I also discuss the significance of the differences. I do this using the Central Limit Theorem, first proved by De Laplace (1810). This way, the problem is reduced to a simple case of hypothesis testing. I assume that

$$\text{MSE}_1 - \text{MSE}_2 \sim N\left(0, \frac{s_1^2}{N_1} + \frac{s_2^2}{N_2}\right), \tag{21}$$

where MSE_i is the MSE of model i, s_i is the sample standard deviation of the squared errors of model i and N_i the number of observations for model i. My null hypothesis is $H_0 = MSE_1 = MSE_2$. This way, I can determine whether a model with a lower MSE also has a significantly better fit to the data.

Table 4 shows the results of my evaluation. The results for linear regression and lasso are equal, as I selected the regularisation parameter for lasso to be 0 after running grid search. The random forest always outperforms the other models. For the out-of-sample fit, simple models such as linear regression and attribute selection using univariate trees perform surprisingly well, even better than more complicated models such as the model tree or MARS. However, for the in-sample fit, the simple models do not outperform the more complicated models. Instead, the performance of the model tree comes closest to that of the random forest. The performance of PLM does not vary much from the performance of linear regression. For the out-of-sample fit, adding extra dummy variables using univariate trees or a fully grown tree to the regression does not give a much better performance, but for the in-sample fit, those two models do outperform linear regression. None of the models seems to perform exceptionally well for the Mashable data set. The correlation of the out-of-sample random forest estimations with the actual (log) number of shares is only 0.398. There are different

rules of thumb for how Pearson's correlation coefficient should be interpreted. Hinkle et al. (2003), for example, stated that an absolute correlation of 0.00 - 0.30 should be considered a negligible correlation, and an absolute correlation of 0.30 - 0.50 a low correlation. According to this rule of thumb, all the models constructed to predict the number of shares for the Mashable data set have a negligible to a low correlation coefficient. There are no such rules of thumb for the MSE and relative error. However, given that the mean of the logarithm of the number of shares is 7.47, an MSE of 0.72 is rather high. Additionally, a relative error of 0.383 implies that the variance of the true values is only 2.5 times higher than the variance of the absolute values of the residuals. Thus, the relative error is also high for this data set.

Table 5 indicates which models significantly differ in performance compared to other models given their MSE, with significance level $\alpha = 0.05$. The p-values for differences between models that are not significant are displayed in red. There is no significant proof that one of the models is significantly better than the rest. However, this table does show that MARS and attribute selection using a fully grown tree perform significantly worse than the other models for the Mashable data set.

Table 4: Regression results for the Mashable data set

Out-of-sample	**MSE**	**Relative Error**	**Correlation**
Linear regression	0.725	0.392	0.390
Lasso	0.725	0.392	0.390
Random forest	**0.720**	**0.383**	**0.398**
Model tree	0.733	0.395	0.383
MARS	0.807	0.483	0.322
PLM	0.725	0.392	0.391
Univariate trees	0.726	0.390	0.389
Fully grown tree	0.740	0.396	0.374
In-sample	**MSE**	**Relative Error**	**Correlation**
Linear regression	0.715	0.385	0.404
Lasso	0.715	0.385	0.404
Random forest	**0.579**	**0.311**	**0.618**
Model tree	0.706	0.382	0.417
MARS	0.707	0.383	0.416
PLM	0.713	0.385	0.407
Univariate trees	0.707	0.381	0.415
Fully grown tree	0.706	0.380	0.417

Best results are displayed in bold

As these observations are only based on a single data set, I am not able to draw strong conclusions. Therefore, I also discuss the performances of the models on the other four data sets. The exact results for these data sets can be found in Tables A1-A4. I summarise the results in Table 6. This table can be interpreted as follows: for each of the four data sets, I computed how much higher in percentage the evaluation criteria of the different models were

Table 5: Significance of the differences in MSE for the Mashable data set

	LR	Lasso	Random forest	Model tree	MARS	PLM	Univariate trees	Fully grown tree
LR	-	0.500	0.245	0.124	0.000	0.499	0.398	0.019
Lasso	0.500	-	0.245	0.124	0.000	0.499	0.398	0.019
Random forest	0.245	0.245	-	0.033	0.000	0.246	0.172	0.003
Model tree	0.124	0.124	0.033	-	0.000	0.124	0.185	0.177
MARS	0.000	0.000	0.000	0.000	-	0.000	0.000	0.000
PLM	0.499	0.499	0.246	0.124	0.000	-	0.397	0.019
Univariate	0.398	0.398	0.172	0.185	0.000	0.397	-	0.034
Fully grown	0.019	0.019	0.003	0.177	0.000	0.019	0.034	-

than the evaluation criterion for random forest regression, and I took the average of these percentages. For the MSE and the relative error, MARS outperforms the other models, as its MSE is only 16.6% higher than that of random forest regression on average, and its relative error 16.0%. PLM and attribute selection using a fully grown tree are also not far off, but random forest regression greatly outperforms the other models on average. I find similar results for Pearson's correlation coefficient, where attribute selection using a fully grown tree outperforms the other models, as its correlation coefficient is just 1.0% lower than that of random forest regression on average. It is rather surprising that attribute selection using a fully grown tree performs so much better than most of the other models, as it is one of the less complicated models on the list. I discuss the interpretability of attribute selection using a fully grown tree and the other models in Section 5.3. In Tables A5 - A8, I indicate which models significantly differ in performance compared to other models given their MSE, again with significance level $\alpha = 0.05$. Especially for the first two data sets, all differences in performance are significant, meaning that of no pair of two models can be said that their performance is statistically equal. For the third data set, not all differences are significant. The random forest, MARS, the PLM and attribute selection using a fully grown tree all do significantly outperform linear regression and lasso. For the fourth data set, the Communities and Crime data set, none of the models significantly outperforms any of the other models with regards to the MSE. This means that for this data set, no model can be regarded as better than any of the others.

5.2 Classification Models

I evaluate all classification models based on their *receiver operating characteristic area under curve* (ROC-AUC) score, their *accuracy*, *precision*, *recall* and *F1* score. The ROC-AUC score (Green and Swets, 1966) does not directly use the classes predicted by a model, but uses the probability assigned to these classes, and represents the probability that a randomly chosen positive observation would be assigned a higher probability for class 1 than a randomly chosen

Table 6: Average of performances of different models compared to the random forest

Out-of-sample	MSE	Relative Error	Correlation
Linear regression	86.3%	67.4%	-4.3%
Lasso	91.0%	74.9%	-4.4%
Model tree	37.6%	39.6%	-2.9%
MARS	**16.6%**	**16.0%**	-1.5%
PLM	24.0%	21.9%	-2.3%
Univariate trees	89.7%	64.8%	-4.5%
Fully grown tree	28.9%	38.2%	**-1.0%**

Best results are displayed in bold

negative observation. The ROC-AUC lies between 0.5 and 1, where a value of 1 indicates that the model distinguishes perfectly between the two classes. I determine the optimal hyperparameters for the different models according to the ROC-AUC.

Tables 7 and 8 show the results of the evaluation of the two different classification tasks. For the popular/not popular classification, for both the out-of-sample and in-sample fit, random forest classification outperforms the other models for almost all evaluation criteria. The only exception is for the precision in the out-of-sample fit, where random forest regression performs rather poorly compared to the other models, and lasso outperforms the other models. MARS and attribute selection using univariate trees perform remarkably well for the out-of-sample fit. For the in-sample fit, attribute selection using univariate trees also performs well, just like the logistic model tree. Regularisation does not appear to influence the fits very much. The performance of the random forest is almost exactly the same as its performance in the paper by Fernandes et al. (2015) for the different evaluation criteria. In their paper, the random forest was outperformed by naive Bayes for the precision. Clearly, none of the models used in this paper or the paper by Fernandes et al. (2015) has been able to structurally outperform the random forest.

For the very popular/not very popular classification, I find rather similar results. For the out-of-sample fit, attribute selection using a fully grown tree has a higher recall than the other models. Everywhere else, random forest regression is superior. Logistic regression and lasso perform remarkably well for the out-of-sample fit, for most evaluation criteria even better than the other models. For the in-sample fit, mainly attribute selection using univariate trees stands out regarding its performance, but there are no substantial differences.

As I discussed in Section 1, multiple authors applied classification methods rather than regression methods to the Mashable data set, due to the high variance of its target variable. Earlier in this section, I already discussed that none of the regression models performs exceptionally well. Classification models indeed perform better for this data set than regression models. For example, according to the rule of thumb of Hosmer and Lemeshow (2000), a

ROC-AUC score between 0.5 and 0.7 gives rather poor discrimination, a ROC-AUC score between 0.7 and 0.8 gives acceptable discrimination, and a ROC-AUC score between 0.8 and 0.9 gives excellent discrimination. Thus, according to the ROC-AUC score, the classification models do not perform extremely well but are acceptable for both popular/not popular classification and very popular/not very popular classification.

Table 7: Popular/not popular classification results for the Mashable data set

Out-of-sample	**ROC-AUC**	**Accuracy**	**Precision**	**Recall**	**F1**
Linear regression	0.725	0.665	0.680	0.672	0.676
Lasso	0.726	0.667	**0.682**	0.673	0.677
Random forest	**0.731**	**0.671**	0.673	**0.712**	**0.692**
Logistic model tree	0.726	0.667	0.682	0.673	0.677
MARS	0.726	0.668	0.678	0.688	0.683
Univariate trees	0.727	0.669	0.679	0.687	0.683
Fully grown tree	0.724	0.663	0.675	0.678	0.677
In-sample	**ROC-AUC**	**Accuracy**	**Precision**	**Recall**	**F1**
Linear regression	0.733	0.674	0.683	0.695	0.689
Lasso	0.733	0.673	0.683	0.694	0.688
Random forest	**0.796**	**0.716**	**0.715**	**0.753**	**0.734**
Logistic model tree	0.736	0.679	0.688	0.699	0.694
MARS	0.731	0.674	0.678	0.709	0.693
Univariate trees	0.742	0.679	0.687	0.702	0.694
Fully grown tree	0.738	0.677	0.686	0.697	0.691

Best results are displayed in bold

Table 8: Very Popular/not very popular classification results for the Mashable data set

Out-of-sample	**ROC-AUC**	**Accuracy**	**Precision**	**Recall**	**F1**
Linear regression	0.724	0.682	0.186	0.650	0.289
Lasso	0.726	0.683	0.187	0.652	0.291
Random forest	**0.727**	**0.728**	**0.201**	0.577	**0.298**
Logistic model tree	0.724	0.682	0.186	0.650	0.289
MARS	0.718	0.676	0.183	0.651	0.286
Univariate trees	0.712	0.673	0.180	0.641	0.281
Fully grown tree	0.721	0.662	0.179	**0.668**	0.283
In-sample	**ROC-AUC**	**Accuracy**	**Precision**	**Recall**	**F1**
Linear regression	0.734	0.686	0.191	0.662	0.296
Lasso	0.734	0.687	0.191	0.661	0.296
Random forest	**0.810**	**0.713**	**0.221**	**0.744**	**0.341**
Logistic model tree	0.734	0.686	0.191	0.662	0.296
MARS	0.720	0.672	0.181	0.647	0.283
Univariate trees	0.744	0.685	0.193	0.677	0.300
Fully grown tree	0.736	0.668	0.186	0.687	0.292

Best results are displayed in bold

In Table 9, I summarise the results for the four extra classification data sets. The individual results can be found in Tables A9-A12. I approach this the same way as I did for the

regression data sets. For the ROC-AUC score, attribute selection using a fully grown tree is most similar to the random forest in terms of performance, as its ROC-AUC is only 0.5% lower than that of the random forest on average. For accuracy and recall, attribute selection using a fully grown tree even outperforms the random forest on average. For the recall and the F1 score, attribute selection using univariate trees comes closest to the random forest in terms of performance, for the recall it even outperforms it. Linear regression and lasso have on average the lowest performance compared to the random forest. For the recall, most models outperform the random forest model on average, except for MARS and attribute selection using a fully grown tree. It is important to note that the optimal hyperparameters were determined using the ROC-AUC score, to avoid the models predicting that all observations belong to the majority class in case of imbalanced data. If, for example, the accuracy score would be used instead, this could give very different results.

Table 9: Average of performances of different models compared to the random forest

Out-of-sample	ROC-AUC	Accuracy	Precision	Recall	F1
Linear regression	-1.3%	-2.2%	-5.2%	1.2%	-2.8%
Lasso	-1.1%	-2.3%	-5.6%	2.1%	-2.7%
Logistic model tree	-1.4%	-1.5%	-3.7%	1.2%	-2.0%
MARS	-0.9%	-2.3%	-3.6%	-0.5%	-3.1%
Univariate trees	-0.8%	-1.3%	-3.3%	**1.5%**	**-1.7%**
Fully grown tree	**-0.5%**	**0.3%**	**3.5%**	-4.0%	-1.9%

Best results are displayed in bold

5.3 Interpretability

In this section, I discuss the interpretability of the different models discussed in this thesis. I use the Mashable data set as a running example. Using graphs and tables, I explain how the output of the different models lead to a comprehension of the relations within a data set. In contrast to before, I analyse the results without manipulating the data in any way, such as taking logarithms or standardising the explanatory variables. These techniques can increase the performance of models but are not needed when discussing their interpretability.

Linear Regression and Lasso

First, I discuss the Mashable regression results for linear regression in a more detailed manner, specifically focusing on its interpretability. Inference for lasso is not straightforward. Belloni et al. (2013) discussed the inference of lasso-type methods, that find sparse models. Hastie et al. (2009) stated that lasso causes the estimates of the non-zero coefficients to be biased towards zero. Additionally, they are usually not consistent. They state several ways for

reducing this bias, for example running lasso to identify which coefficients should be set to zero, and then running a regular regression on the remaining coefficients. Another method would be running lasso to identify the zero-coefficients, and then applying lasso again, just to the selection from the previous step. In the second step, there would be less competition from 'noise variables', such that the penalty parameter is usually smaller. A third method would be modifying the penalty function such that large coefficients are shrunk in a different way than small coefficients. As the results for linear regression and lasso on the Mashable regression data set are the same, I do not discuss lasso separately. Also when the results are not exactly the same, the interpretation is similar. However, in general, it is crucial to account for the biasedness of the lasso estimates, such that both the estimates and the standard errors account for the uncertainty in the selection lasso makes. The linear regression coefficients can be found in Table 10.

Table 10: Linear regression estimates

Feature	Coefficient (Standard Error)	Feature	Coefficient (Standard Error)
Rate of unique words	53.1 (1539.5)	Day of the week: Wednesday	197.9 (197.4)
Rate of unique non-stop words	-50.5 (1660.1)	Day of the week: Thursday	-4.4 (198.2)
Number of links	33.5 (7.0)***	Day of the week: Friday	8.8 (212.5)
Number of Mashable article links	-52.3 (18.4)***	Day of the week: Saturday	606.2 (283.3)**
Number of images	16.5 (9.4)*	Day of the week: Sunday	274.9 (271.1)*
Number of videos	15.1 (16.3)	Closeness to LDA topic 1	-707.4 (540.4)***
Average word length	-521.0 (185.3)***	Closeness to LDA topic 2	-1216.8 (515.5)**
Number of keywords	49.0 (38.6)**	Closeness to LDA topic 3	-402.2 (510.0)
Article category: lifestyle	-1050.9 (411.7)**	Closeness to LDA topic 4	-392.2 (467.6)***
Article category: entertainment	-1132.8 (272.1)***	Article text subjectivity score (ATSS)	2375.7 (872.1)***
Article category: business	-740.4 (392.2)*	Article text polarity score (ATPS)	-586.1 (1694.5)
Article category: social media	-605.1 (382.0)	Rate of positive words	-13,979.0 (7307.1)*
Article category: tech	-587.0 (386.0)**	Rate of negative words	590.5 (13314.2)
Article category: world	-527.1 (396.3)**	Positive words rate among non-neutral tokens	295.6 (1053.9)
Worst keyword (minimum number of shares)	2.3 (1.6)	Average polarity of positive words	-1888.1 (1396.1)
Worst keyword (maximum number of shares)	0.1 (0.1)*	Minimum polarity of positive words	-1984.2 (1172.3)*
Worst keyword (average shares)	-0.4 (0.3)	Maximum polarity of positive words	548.7 (449.8)
Best keyword (minimum number of shares)	-0.002 (0.001)	Average polarity of negative words	-1134.8 (1286.7)
Best keyword (maximum number of shares)	-0.005 (0.001)	Minimum polarity of negative words	-338.5 (472.8)
Best keyword (average shares)	-0.001 (0.001)	Maximum polarity of negative words	-231.2 (1076.2)
Average keyword (minimum number of shares)	-0.4 (0.1)***	Title subjectivity	-89.1 (287.7)
Average keyword (maximum number of shares)	-0.2 (0.0)***	Title sentiment polarity	117.0 (263.5)
Average keyword (average shares)	1.7 (0.1)***	Absolute difference to 0.5 of ATSS	515.6 (379.5)
Minimum number of shares of referenced Mashable articles	0.03 (0.01)***	Absolute difference to 0.5 of ATPS	687.6 (415.8)*
Maximum number of shares of referenced Mashable articles	0.01 (0.004)	Number of words in title	800.4 (277.3)***
Average shares of referenced Mashable articles	-0.01 (0.01)***	Number of words in article	-239.1 (136.3)*
Day of the week: Monday	525.7 (203.8)***		

Significance: $*** < 0.01, ** < 0.05, * < 0.1$

Table 10 shows all features used for the models and their linear regression coefficient. Approximately half the features are statistically significant when $\alpha = 0.1$. The interpretation of

linear regression models is very straightforward. Multiplying all features by their respective coefficients and adding them to the intercept (1440.9) gives an estimate for the total number of shares for that article. For example, adding a new link increases the expected number of shares by approximately 35. If the category of the article is *entertainment*, it decreases the expected number of shares by approximately 1133 shares. The day of the week an article is published on only matters significantly in case that day is Monday, Saturday or Sunday. On all these days, the expected amount of shares is significantly higher than it would be on a Tuesday, for which the variable is left out due to multicollinearity. The more words an article has, the lower the estimated number of shares. However, the number of words in the title has a positive effect on the estimated number of shares. The closeness of each article to the most important topics, that were identified using the Latent Dirichlet Algorithm (LDA), is usually significant. The other features related to Natural Language Processing (NLP) are mostly not significant in the linear regression model. However, the subjectivity score of the article is significant, even if $\alpha = 0.01$. As I discussed in Section 1, being able to interpret models that estimate the popularity of an article is important, such that an article can be written in such a way, that the expected popularity is maximised. For this data set, a good understanding of the factors that cause an article to have a higher number of shares can help Mashable authors write new articles. Even though, as shown in Figure 4, the random forest outperforms the linear regression, it cannot be used as easily to investigate how to improve future articles.

Logistic Regression and Lasso

The coefficients and standard errors for logistic regression can be found in Table 11. Logistic regression is not as directly interpretable as linear regression. However, the odds ratio (Cox and Snell, 1970) makes logistic regression more interpretable, as the conditional odds ratio equals the exponent of the estimated coefficients, e^{β}. The odds ratio indicates how much the odds change following an increase in the corresponding explanatory variable while keeping the other features constant. For example, if the day of the week is Sunday, the odds of an article being popular are 1.000014 times the odds of an article that is not published on Sunday being popular. Likewise, if the minimum amount of shares of referenced Mashable articles goes up by one unit, the odds of an article being popular are 1.0000065 times the odds of when the minimum amount is not increased. In contrast to the linear regression, neither the closeness of each article to the most important topics plays a significant role in logistic regression, nor the other features related to NLP.

Random Forest

Table 11: Logistic regression estimates

Feature	Coefficient (Standard Error)	Feature	Coefficient (Standard Error)
Rate of unique words	4.35E-06 (1.62E-04)	Day of the week: Wednesday	-1.31E-05 (1.68E-04)
Rate of unique non-stop words	2.31E-06 (1.50E-04)	Day of the week: Thursday	-1.10E-05 (1.13E-04)
Number of links	4.35E-04 (2.57E-03)	Day of the week: Friday	-6.80E-07 (3.31E-05)
Number of Mashable article links	6.15E-05 (5.75E-04)	Day of the week: Saturday	2.03E-05 (2.34E-04)
Number of images	1.55E-04 (8.17E-04)	Day of the week: Sunday	1.34E-05 (1.67E-04)
Number of videos	-6.19E-05 (7.28E-04)	Closeness to LDA topic 1	-1.73E-05 (2.06E-04)
Average word length	-5.82E-05 (5.39E-04)	Closeness to LDA topic 2	-2.06E-05 (2.50E-04)
Number of keywords	-1.13E-06 (2.21E-04)	Closeness to LDA topic 3	-1.32E-05 (1.45E-04)
Article category: lifestyle	6.03E-07 (1.61E-05)	Closeness to LDA topic 4	2.43E-05 (2.93E-04)
Article category: entertainment	-4.16E-05 (4.53E-04)	Article text subjectivity score (ATSS)	-4.53E-07 (8.78E-06)
Article category: business	6.44E-06 (9.82E-05)	Article text polarity score (ATPS)	3.35E-06 (3.84E-05)
Article category: social media	1.87E-05 (2.19E-04)	Rate of positive words	1.87E-07 (3.60E-06)
Article category: tech	3.85E-05 (4.45E-04)	Rate of negative words	-5.11E-07 (5.35E-06)
Article category: world	-2.90E-05 (3.47E-04)	Positive words rate among non-neutral tokens	1.92E-07 (2.53E-05)
Worst keyword (minimum number of shares)	-1.20E-03 (3.63E-04)***	Average polarity of positive words	-3.62E-06 (3.42E-05)
Worst keyword (maximum number of shares)	7.41E-05 (2.08E-05)***	Minimum polarity of positive words	-3.78E-06 (3.81E-05)
Worst keyword (average shares)	-5.35E-04 (1.49E-04)***	Maximum polarity of positive words	-1.63E-06 (1.64E-05)
Best keyword (minimum number of shares)	-2.05E-07 (2.20E-07)	Average polarity of negative words	4.44E-06 (5.00E-05)
Best keyword (maximum number of shares)	-1.26E-06 (7.22E-08)	Minimum polarity of negative words	5.91E-06 (7.50E-05)
Best keyword (average shares)	-1.63E-06 (1.48E-07)	Maximum polarity of negative words	2.28E-06 (2.30E-05)
Average keyword (minimum number of shares)	-5.30E-05 (1.42E-05)***	Title subjectivity	1.34E-06 (8.00E-06)
Average keyword (maximum number of shares)	-9.01E-05 (6.12E-06)***	Title sentiment polarity	8.28E-06 (8.60E-05)
Average keyword (average shares)	6.88E-04 (3.09E-05)***	Absolute difference to 0.5 of ATSS	-3.90E-06 (1.87E-05)
Minimum number of shares of referenced Mashable articles	6.45E-06 (3.40E-06)*	Absolute difference to 0.5 of ATPS	2.34E-06 (1.91E-05)
Maximum number of shares of referenced Mashable articles	2.71E-06 (9.41E-07)***	Number of words in title	-2.93E-05 (2.75E-04)
Average shares of referenced Mashable articles	-1.18E-06 (2.70E-06)	Number of words in article	-3.20E-05 (2.73E-04)
Day of the week: Monday	-8.39E-06 (8.43E-05)		

Significance: $***< 0.01, **< 0.05, *< 0.1$

I have also evaluated a random forest, a machine learning algorithm that Fernandes et al. (2015) concluded worked best for classification in their analysis. The advantage of random forests over regular decision trees is that they counteract the problem of overfitting the data (Hastie et al., 2009). Random forests are often used as black-box models, as they often need little configuration but make rather accurate predictions.

In situations where the relationship between the dependent variable and the explanatory variables is nonlinear, or where explanatory variables interact with each other, decision trees can be a good alternative to linear regression. Interpreting a simple decision tree that is not very deep is not hard: it can be seen as a collection of rules, that apply in different situations, for different feature combinations. Random forests are harder to interpret, as they consist of multiple relatively deep trees. Feature importances can be calculated, but, when using random forests, interpretability is traded in for higher accuracy. The feature importances for the ten most important features of the random forest regression are given in Table 12. The importances are calculated based on the reduction in the Gini criterion, which was used

to decide how variables should be split. A majority of the most important features are also significant for the linear regression. The estimated number of shares for an article heavily depends on how often other articles with the same keywords were shared, and how often other articles, that are referenced in the article were shared. When it comes to the NLP topics, the third LDA topic and the average polarity of negative words seem to play an important role in estimating the number of shares.

Table 12: Feature importances of the random forest regression

	Feature	Importance
1	Average keyword (average shares)	0.093
2	Average keyword (maximum number of shares)	0.087
3	Average share of referenced Mashable articles	0.072
4	Maximum number of shares of referenced Mashable articles	0.065
5	Minimum number of shares of referenced Mashable articles	0.062
6	Closeness to LDA topic 3	0.045
7	Number of links	0.034
8	Rate of unique words	0.034
9	Average polarity of negative words	0.028
10	Rate of unique non-stop words	0.028

The hyperparameters for the random forest, for both regression and classification, are given in Table 13. It is not feasible to properly interpret 90 to 140 trees with a depth of up to

Table 13: Random forest hyperparameters

	Regression	Popular/not popular classification	Very popular/not very popular classification
Number of trees	90	110	140
Maximum number of features per tree	8	8	8
Maximum depth	12	15	9
Minimum fraction of samples required to split an internal node	0.001	0.01	0.005
Minimum fraction of samples required to be at a leaf node	0.001	0.001	0.005
Use of bootstrapped data	False	False	False

15. Therefore, I discussed other models, which combine econometrics and machine learning, to improve interpretability while outperforming linear regression when it comes to prediction accuracy.

Model Tree

I also discuss the interpretation of the (logistic) model tree. A visualisation plot for the Mashable regression can be found in Figure 2. Like I mentioned in Section 4.1, model trees are usually smaller but more accurate than regular decision trees. A model tree can be interpreted as a clustering method that creates four groups in the data and then runs a linear regression for each of those four groups. Unfortunately, as seen in Table 4, model trees are not more accurate than linear regression for the out-of-sample fit. However, for other

data sets, it is more accurate, as shown in Table 6. This inaccuracy might have to do with the fact that model trees only take into account the features that were previously split on for the regression in the leaf nodes, such that the regressions in this case only have a few features. This was done such that a fair comparison could be drawn between the accuracy of the linear model and that of the subtree.

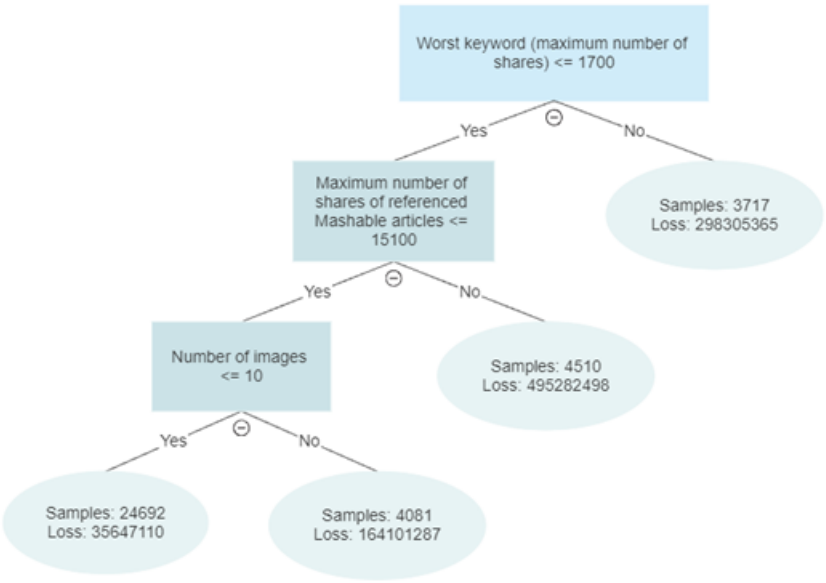


Figure 2: Visualisation of Mashable regression model tree

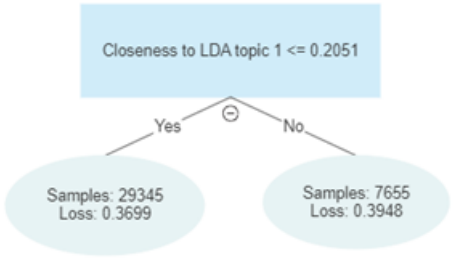


Figure 3: Visualisation of Mashable classification model tree

Figure 3 shows the logistic model tree for the popular/not-popular classification. This is a very simple tree, that only generates two groups of articles, depending on their closeness to the first LDA topic. For the Mashable classification, the logistic model tree does slightly outperform linear regression for the out-of-sample fit, but not lasso. For the other data sets I evaluated, the logistic model tree did outperform the other models for all evaluation criteria

except the ROC-AUC score.

Multivariate Adaptive Regression Splines

MARS works differently for regression and classification. For regression, the regression splines are directly taken into account. For classification, first MARS for regression is run, where the binary response variable is the dependent variable. Then, the resulting regression splines are inputs for logistic regression.

Table 14 displays the regression splines for the MARS regression. Table 4 shows that MARS does not outperform linear regression. In fact, it performs worse than any of the other models. For the other data sets, however, MARS is one of the best models, if not the best. For the Mashable data set, the most used feature in the regression splines is *Average share of referenced Mashable articles*. Furthermore, the rate of (unique) non-stop words also appears multiple times. MARS uses far fewer regression splines than linear regression uses features, such that they can more easily be comprehended at once. For example, the first regression spline in Table 14 equals h(4466.83 − *Average keyword (average shares)*). This regression spline translates to max(4466.83 − *Average keyword (average shares)* , 0). The median of this feature is 2852, such that for more than half the observations, the new feature is 4466.83 minus the old feature. For the other part of the observations, the new feature equals 0. The coefficient of this regression spline, −0.96, is negative. Therefore, the higher the average shares of the average keyword are, the lower the negative effect on the estimated shares of the Mashable article itself is. If the average number of shares of the average keyword is higher than 4467, the regression spline has no more effect on the estimated shares. It becomes more complicated, however, when terms are multiplied by each other, for example for the second regression spline. The interpretation for this regression spline is as follows: if the article category is *business*, and the average number of shares of the average keyword is higher than 4467, each share over 4467 adds 2298 to the estimated number of shares. If the article category is not *business*, or the average number of shares of the average keyword is lower than 4467, there is no effect on the estimated number of shares. When a regression spline uses two features that are both not dummy features, it becomes very hard to give an intuitive interpretation. A way to handle this is only allowing one term per regression spline, thus avoiding multiplications. This will, however, affect the performance accuracy.

For the Mashable classification, after running MARS, I ended up with 36 regression splines. For the popular/not popular classification, MARS does outperform linear regression. However, since logistic regression already requires extra steps to become interpretable, exchanging the regular features for regression splines makes interpretability even harder. The model can be interpreted as a regular logistic regression model, but with different features.

Table 14: MARS regression splines

Regression spline	Coefficient
h(4466.83 - Average keyword (average shares))	-0.96
Article category: business * h(Average keyword (average shares) - 4466.83)	2298.48
h(0.34 - Rate of unique words) * h(Average shares of referenced Mashable articles - 407)	26.97
h(Rate of unique non-stop words - 0.37) * h(Average shares of referenced Mashable articles - 407)	-1192.81
h(0.368276 - Rate of unique non-stop words) * h(Average share of referenced Mashable articles - 407)	7308.7
h(Closeness to LDA topic 3 - 0.02) * h(Average share of referenced Mashable articles - 407)	0.27
h(Rate of unique non-stop words - 0.39) * h(Average share of referenced Mashable articles - 407)	-238.67
h(Rate of unique non-stop words - 0.37) * h(Average share of referenced Mashable articles - 407)	1431.48
h(0.37 - Rate of unique non-stop words) * h(Average share of referenced Mashable articles - 407)	-12,900.6
h(0.38 - Rate of unique non-stop words) * h(Average share of referenced Mashable articles - 407)	5567.46
Day of the week: Monday * h(Average keyword (average shares) - 4466.83)	977.74
Average polarity of negative words * h(Average share of referenced Mashable articles - 407)	-0.58
h(285 - Average keyword (minimum amount of shares))	2.73

As I have explained regression splines before, I will not go into more detail.

Partially Linear Model

The partially linear model is a combination of two of the other models included in this analysis, namely linear regression and a model tree. The partially linear model uses the fact that not all features affect the dependent variable in a non-linear way, and allows one or two of them to be modelled using a model tree. As can be seen in Table 4, the partially linear model does not outperform linear regression for the Mashable data set, but it does outperform the model tree. This can be explained by the fact that the model tree might overfit the data, and the (more simple) partially linear model does not. From Table 6, it follows that the partially linear model outperforms on average both linear regression and model trees for the other data sets I evaluated. Figure 4 shows the non-parametric part of the partially linear model, namely the model tree. With a depth of 6, the tree is rather deep. I used grid search to determine the maximum depth of the tree, resulting in this depth. As this tree only uses two features, namely *Average shares of referenced Mashable articles* and *Average keyword (maximum number of shares)*, interpretation is still not very complicated. Using these two features, seven different groups of observations are created, of which the smallest group contains 523 observations, and the biggest 13,716. For the other features in the data set, I ran a linear regression, the result of which can be found in Table 15. These results have the same interpretation as for the regular linear regression I discussed. Compared to Table 10, the coefficients and their signs are rather similar. After running the linear regression, all the coefficients were multiplied by the values of their corresponding features. Afterwards, for each observation, I added its value according to the model tree in Figure 4. This resulted in an estimate for the number of shares of each article. This model is rather interpretable, especially for more shallow regression trees. The model mainly consists of linear regression,

and only a few non-linear features are computed differently. By using a model tree for these few features, which divides the observations into different groups and runs separate regressions for each of those groups, even the non-linear features remain very interpretable, and they give much insight into the underlying structure of the data set.

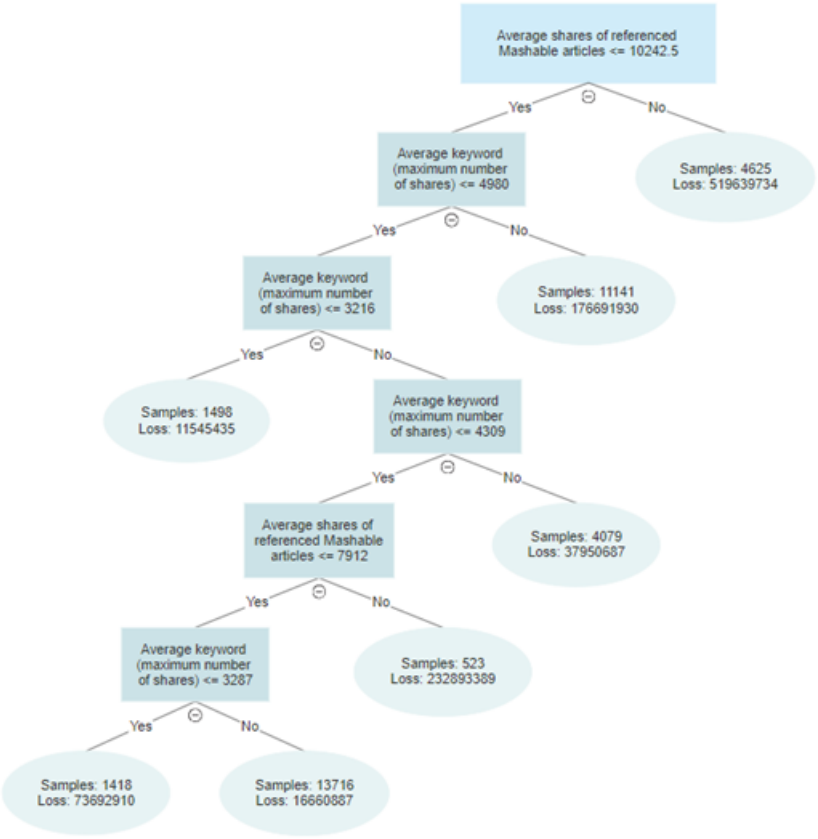


Figure 4: Visualisation of the non-parametric part of the partially linear model

Attribute Selection Using Decision Trees

The last two models do attribute selection using decision trees. These decision trees are either different univariate trees, where each split is an extra feature in the linear or logistic model, or one fully grown tree, where observations get dummies for the leaf node they end up in, which are then extra features in the linear or logistic model. Table 6 shows that the fully grown tree performs very well on average for the out-of-sample fit for the extra regression data sets, even better than the more complicated model tree. Table 9 shows that for the out-of-sample fit for the extra classification data set, attribute selection using a fully grown tree even outperforms the other models for some criteria and univariate trees for the other.

Figure 5 gives an example of one of the univariate regression trees that were used in the

Table 15: Estimates for the linear part of the partially linear model

Feature	Coefficient	Feature	Coefficient
Rate of unique words	29.1	Day of the week: Wednesday	172.3
Rate of unique non-stop words	-24.4	Day of the week: Thursday	-41.5
Number of links	36.5	Day of the week: Friday	-70.0
Number of Mashable article links	-52.3	Day of the week: Saturday	654.0
Number of images	13.8	Day of the week: Sunday	227.1
Number of videos	16.4	Closeness to LDA topic 1	-969.9
Average word length	-581.2	Closeness to LDA topic 2	-1068.5
Number of keywords	-13.0	Closeness to LDA topic 3	-595.4
Article category: lifestyle	-1205.3	Closeness to LDA topic 4	-324.1
Article category: entertainment	-994.8	Article text subjectivity score (ATSS)	1973.1
Article category: business	-665.1	Article text polarity score (ATPS)	625.8
Article category: social media	-564.8	Rate of positive words	-10,916.5
Article category: tech	-481.1	Rate of negative words	-252.9
Article category: world	-450.3	Positive words rate among non-neutral tokens	44.3
Worst keyword (minimum number of shares)	0.9	Average polarity of positive words	-1815.1
Worst keyword (maximum number of shares)	0.1	Minimum polarity of positive words	-1741.3
Worst keyword (average shares)	-0.2	Maximum polarity of positive words	370.7
Best keyword (minimum number of shares)	-0.001	Average polarity of negative words	-1330.0
Best keyword (maximum number of shares)	-0.001	Minimum polarity of negative words	-90.9
Best keyword (average shares)	-0.001	Maximum polarity of negative words	-633.2
Average keyword (minimum number of shares)	-0.3	Title subjectivity	-46.7
Average keyword (maximum number of shares)	×	Title sentiment polarity	198.3
Average keyword (average shares)	1.4	Absolute difference to 0.5 of ATSS	660.1
Minimum number of shares of referenced Mashable articles	0.1	Absolute difference to 0.5 of ATPS	796.8
Maximum number of shares of referenced Mashable articles	0.01	Number of words in title	732.9
Average shares of referenced Mashable articles	×	Number of words in article	-112.1
Day of the week: Monday	489.4		

regression model. For the feature *Rate of unique words* itself, the corresponding coefficient is -1710. For the dummy variable corresponding to whether *Rate of unique words* is smaller than 0.24 or not, it is -8403.43. For example, for the first observation, the rate of unique words is 0.605. Therefore, the contribution of this feature to the estimated number of shares for this article is $0.605 \cdot -1710 + 0 \cdot -8403.43$, which equals -1034.55. In other words, for each article, the higher the rate of unique words, the lower the estimated number of shares. If the rate of unique words is under 0.24, however, the estimated number of shares goes down with approximately 8403 extra shares. Therefore, holding all the other features equal, the estimated optimal value for the rate of unique words would be just over 0.24. Adding dummies for the splits of univariate decision trees to the linear regression allows for some non-linear modelling. Still, this does not significantly decrease the interpretability of the model. For logistic regression, the interpretation is similar. I will not go into more detail, as I already interpreted logistic regression above.

For the fully grown tree, I did not work with multiple univariate trees, but with one deep, fully grown tree. This tree had a depth of 9 for regression and 10 for classification. This indicates how deep a regular tree can be compared to a model tree for the same data set.

For linear regression, the tree had 39 leaves, such that I added 39 dummy variables to the linear regression model. The dummy variable equals 1 for observations that end up in the leaf the dummy corresponds to. Compared to attribute selection using univariate trees, the

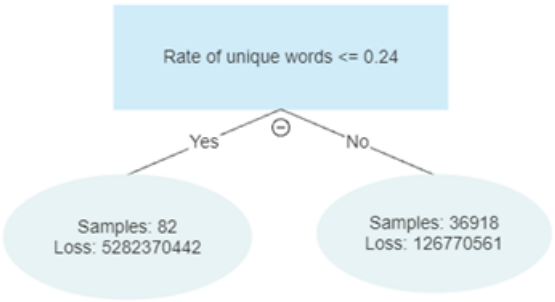


Figure 5: Visualisation of the univariate tree for *Rate of unique words*

interpretation of this model is slightly more complicated. Similar to regular linear regression, each feature has a corresponding coefficient, and these were multiplied with each other to estimate the number of shares of each article. However, in this case, for each observation an additional coefficient was added, depending on the leaf of the decision tree it ended up in. The latter part is hard to interpret. For example, the first observation ended up in the 21st node of the article. Therefore, approximately 172 shares were added to the expected number of shares. Again, for logistic regression, the model works similarly, so I will not go into more detail.

Attribute selection using a fully grown tree often outperforms attribute selection using univariate trees, as can be seen in Tables 6 and 9. For the regression data sets, this difference is rather big. For the classification data sets, the differences between the two models are smaller. In that case, if interpretability would be taken into account, attribute selection using univariate trees would be the more attractive alternative.

6 Conclusion

In this thesis, I focused on the interpretability of several models that integrated machine learning and econometrics in different ways. I did this for both regression and classification models and used linear/logistic regression, lasso and a random forest as benchmarks. First, I discussed the predictive performance of these models. I did this using the following research question:

How does a model that combines econometrics and machine learning perform compared to purely econometric or purely machine learning models in terms of predictive performance?

Afterwards, I discussed the interpretability of the models, using the second research question:

How does a model that combines econometrics and machine learning perform compared to purely econometric or purely machine learning models in terms of interpretability?

My main data set was the Mashable Online News Popularity data set, a data set concerned with estimating the number of shares of different articles that were published on the website Mashable during a period of two years. I analysed both the predictive performance and the interpretability of the models for this data set. Afterwards, I evaluated eight other data sets, four for regression and four for classification. I did this such that I could draw stronger conclusions about the predictive performance of the models, as one data set is not sufficient to do so.

Aside from my benchmark models, I analysed (logistic) model trees, Multivariate Adaptive Regression Splines (MARS), partially linear models (PLM), and attribute selection using decision trees. For the last model, I used univariate trees and fully grown trees.

My main findings for the regression models were as follows: not surprisingly, the random forest usually performs best on average. MARS outperforms all the other models when it comes to the MSE and the relative error of its predictions, but also attribute selection using fully grown trees performs very well; it even outperforms the other models when it comes to the correlation on average. Also, the partially linear model performs well on average compared to the other models. When it comes to interpretability, all models have advantages and disadvantages. Generally, keeping the models simple improves the interpretability, for example by not allowing multiplication terms for MARS, and not letting a model tree grow too deep. However, doing so can also affect the predictive performance of those models.

For the classification models, I can conclude the following: the random forest does not always perform best on average for the four extra data sets I analysed. Attribute selection using fully grown trees gives higher accuracy and precision, and attribute selection using univariate trees gives a higher recall. In general, these two models perform very well on

average for classification. The logistic model tree and MARS generally outperform linear regression and lasso, but not by much. This raises the question of whether the slight gain in predictive performance is worth the decline in interpretability. For the interpretability of the classification models, I draw the same conclusions as for the regression models. MARS regression splines and attribute selection using either univariate trees or a fully grown tree add features to logistic regression, which worsens the interpretability. The extend of this decrease in interpretability depends on various factors such as the amount of added features. However, in contrast to the random forest, in the end, an interpretation can be given. Again, keeping the models simple makes giving this interpretation much easier.

Many of the hybrid models I discussed in my thesis can easily be adapted to real-life situations. A shallow (logistic) model tree can give interesting insights into different groups of observations within the data and differences of estimation between those groups. Attribute selection using univariate trees or a fully grown tree performs very well while keeping computation time low and interpretation relatively easy. A major advantage of MARS is that for certain values, a feature does not have any impact on the estimation of the dependent variable. There are situations and data sets where this would be especially relevant. A partially linear model is a good fit when some of the features within a data set should be modelled in a non-linear way. This improves the interpretability of the model, as the other features are still modelled linearly. All the hybrid models I discussed perform on average better than linear regression and lasso for the four extra data sets I analysed. So, in any situations where the predictive performance of linear regression is not good enough but where a random forest is too abstract, one of the hybrid models could be picked instead.

For future research, the models discussed in my thesis should be tested for a wider range of applications and more complicated data sets, to solidify the claims I have made about their performance. Other combinations of econometrics and machine learning could be explored, such as combining a random forest with (logistic) model trees to decrease the number of trees in the forest and their depth. Another option would be treating the output of each tree in a random forest as a dummy variable, and adding this to a logistic regression model, similar to the attribute selection using a fully grown tree approach. There are also numerous other options for combining econometrics and machine learning, that do not involve decision trees. Those models could be improved upon or made more efficient, or new models could be created.

References

Bandari, R., Asur, S., & Huberman, B. A. (2012). The Pulse of News in Social Media: Forecasting Popularity.

Belloni, A., Chernozhukov, V., & Hansen, C. (2013). Inference on Treatment Effects after Selection among High-Dimensional Controls†. *The Review of Economic Studies*, *81*(2), 608–650.

Blei, D. M., Ng, A. Y., & Jordan, M. I. (2003). Latent Dirichlet Allocation. *Journal of Machine Learning Research*, *3*, 993–1022.

Chen, T., & Guestrin, C. (2016). XGBoost: A Scalable Tree Boosting System. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Cox, D., & Snell, E. (1970). *Analysis of Binary Data* (2nd ed.). Routledge.

De Laplace, J. (1810). Mémoire sur les Approximations des Formules Qui Sont Fonctions de Très Grands Nombres et sur Leur Application aux Probabilités. *Mémoires de l'Académie Royale des Sciences de Paris*, *10*.

De Smedt, T., Nijs, L., & Daelemans, W. (2014). Creative Web Services with Pattern. *Proceedings of the 5th International Conference on Computational Creativity (ICCC 2014)*.

Doshi-Velez, F., & Kim, B. (2017). Towards A Rigorous Science of Interpretable Machine Learning.

Du, M., Liu, N., & Hu, X. (2019). Techniques for Interpretable Machine Learning. *Communications of the ACM*, *63*(1), 68–77.

Dua, D., & Graff, C. (2021). UCI Machine Learning Repository. http://archive.ics.uci.edu/ml

Dumitrescu, E., Hué, S., Hurlin, C., & Tokpavi, S. (2021). *Machine Learning or Econometrics for Credit Scoring: Let's Get the Best of Both Worlds* (Working Papers). HAL.

Fanaee-T, H., & Gama, J. (2013). Event Labeling Combining Ensemble Detectors and Background Knowledge. *Progress in Artificial Intelligence*, 1–15.

Farrell, M. H., Liang, T., & Misra, S. (2021). Deep Neural Networks for Estimation and Inference. *Econometrica*, *89*(1), 181–213.

Fernandes, K., Vinagre, P., & Cortez, P. (2015). A Proactive Intelligent Decision Support System for Predicting the Popularity of Online News. *Progress in Artificial Intelligence*, 535–546.

Friedman, J. H., Hastie, T., & Tibshirani, R. (2000). Additive Logistic Regression: A Statistical View of Boosting. *The Annals of Statistics*, *38*(2).

Friedman, J. H. (1991). Multivariate Adaptive Regression Splines. *Annals of Statistics*.

Friedman, J. H. (1993). *Fast MARS* (tech. rep.). Stanford University, Department of Statistics.

Green, D. M., & Swets, J. A. (1966). *Signal Detection Theory and Psychophysics*. Wiley.

Guerzoni, M., Nava, C. R., & Nuccio, M. (2021). Start-UPS Survival Through a Crisis. Combining Machine Learning With Econometrics to Measure Innovation. *Economics of Innovation and New Technology*, *30*(5), 468–493.

Hall, P., Kay, J. W., & Titterington, D. M. (1990). Asymptotically Optimal Difference-Based Estimation of variance in Nonparametric Regression. *Biometrika*, *77*(3), 521–528.

Hastie, T., Tibshirani, R., & Friedman, J. (2009). *The Elements of Statistical Learning: Data Mining, Inference and Prediction* (2nd ed.). Springer International Publishing.

Hensinger, E., Flaounas, I., & Cristianini, N. (2013). Modelling and Predicting News Popularity. *16*(4), 623–635.

Hinkle, D., Wiersma, W., & Jurs, S. (2003). *Applied Statistics for the Behavioral Sciences* (Vol. 663). Houghton Mifflin.

Hirano, K., & Wright, J. H. (2017). Forecasting With Model Uncertainty: Representations and Risk Reduction. *Econometrica*, *85*(2), 617–643.

Hosmer, D. W., & Lemeshow, S. (2000). *Applied Logistic Regression*. John Wiley; Sons.

Kaggle. (2021). https://www.kaggle.com/

Ke, G., Meng, Q., Finley, T., Wang, T., Chen, W., Ma, W., Ye, Q., & Liu, T.-Y. (2017). LightGBM: A Highly Efficient Gradient Boosting Decision Tree, 3149–3157.

Landwehr, N., Hall, M., & Frank, E. (2003). Logistic Model Trees. *Machine Learning: ECML 2003*, 241–252.

Lantz, B. (2013). *Machine Learning with R*. Packt Publishing.

Leathwick, J. R., Rowe, D., Richardson, J., Elith, J., & Hastie, T. (2005). Using Multivariate Adaptive Regression Splines to Predict the Distributions of New Zealand's Freshwater Diadromous Fish. *Freshwater Biology*, *50*(12), 2034–2052.

Lehrer, S. F., & Xie, T. (2018). *The Bigger Picture: Combining Econometrics with Analytics Improve Forecasts of Movie Success* (tech. rep. No. 24755). National Bureau of Economic Research.

Li, T., Tan, W., & Liu, Z. (2019). A Hybrid Model Based on Logistic Regression Algorithm and Extraction Algorithm Using Reward Extremum to Real-Time Detect Blade Icing Alarm. *International Journal of Pattern Recognition and Artificial Intelligence*, *33*(14).

Lipton, Z. C. (2018). The Mythos of Model Interpretability. *Queue*, *16*(3), 31–57.

Malhotra, A. (2018). *A Hybrid Econometric-Machine Learning Approach for Relative Importance Analysis: Prioritizing Food Policy* (tech. rep.).

Millard, T. (2004). Using Classification Tree Outcomes to Enhance Logistic Regression Models.

Miller, T. (2019). Explanation in Artificial Intelligence: Insights From the Social Sciences. *Artificial Intelligence*, *267*, 1–38.

Molnar, C. (2019). *Interpretable Machine Learning: A Guide for Making Black Box Models Explainable*. Lulu Press.

Molnar, C., König, G., Bischl, B., & Casalicchio, G. (2021). Model-Agnostic Feature Importance and Effects with Dependent Features – A Conditional Subgroup Approach.

Moro, S., Cortez, P., & Rita, P. (2014). A Data-Driven Approach to Predict the Success of Bank Telemarketing. *Decisision Support Systems, Elsevier*, *62*, 22–31.

Mullainathan, S., & Spiess, J. (2017). Machine Learning: An Applied Econometric Approach. *Journal of Economic Perspectives*, *31*, 87–106.

Pritchard, J. K., Stephens, M., & Donnelly, P. (2000). Inference of Population Structure Using Multilocus Genotype Data. *Genetics*, *155*(2), 945–959.

Quinlan, J. R. (1992). Learning With Continuous Classes, 343–348.

Redmond, M., & Baveja, A. (2002). A Data-Driven Software Tool for Enabling Cooperative Information Sharing Among Police Departments. *European Journal of Operational Research*, *141*(3), 660–678.

Robinson, P. (1988). Root- N-Consistent Semiparametric Regression. *Econometrica*, *56*(4), 931–54.

Sarma, K. S. (2005). Combining Decision Trees with Regression in Predictive Modeling with SAS ® Enterprise Miner TM.

Speckman, P. (1988). Kernel Smoothing in Partial Linear Models. *Journal of the Royal Statistical Society: Series B (Methodological)*, *50*(3), 413–436.

Thurman, N., & Fletcher, R. (2018). Are Newspapers Heading Toward Post-Print Obscurity? *Digital Journalism*, *6*(8), 1003–1017.

Van der Voort, M., Dougherty, M., & Watson, S. (1996). Combining Kohonen Maps With Arima Time Series Models to Forecast Traffic Flow". *Transportation Research. Part C: Emerging Technologies*, *4*(5), 307–318.

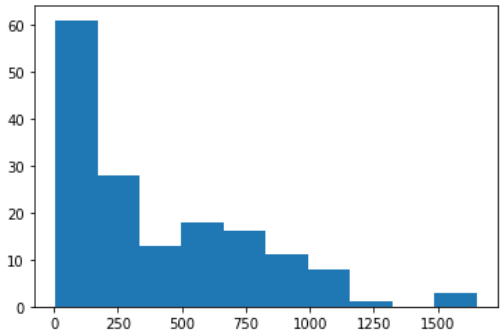
Wahba, G. (1984). Cross Validated Spline Methods for the Estimation of Multivariate Functions From Data on Functionals. In H. A. David (Ed.), *Statistics: An Appraisal. Proceedings 50th Anniversary Conference Iowa State Statistical Laboratory*. Iowa State University Press.

Wang, L., Brown, L. D., & Cai, T. T. (2011). A Difference Based Approach to the Semiparametric Partial Linear Model. *Electronic Journal of Statistics*, *5*, 619–641.

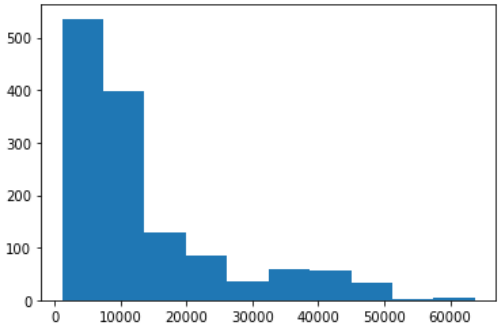
Yatchew, A. (1997). An Elementary Estimator of the Partial Linear Model. *Economics Letters*, *57*(2), 135–143.

Appendix

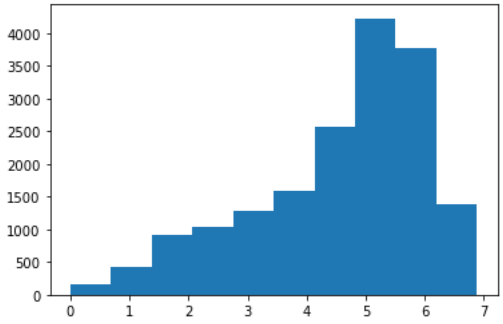
Appendix 1: Figures



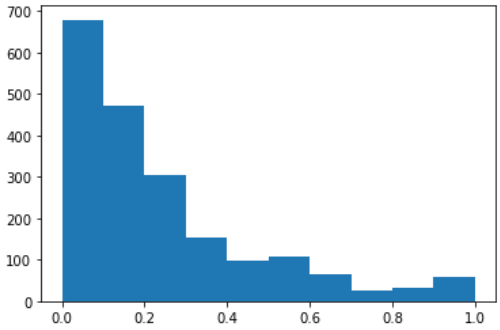
(a) Fish Market



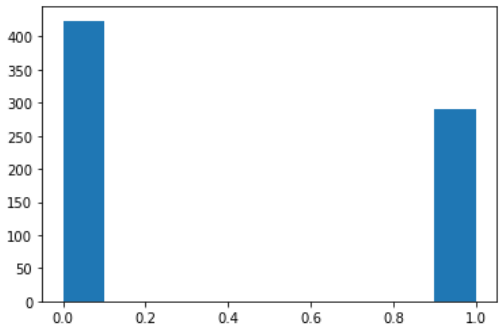
(b) Medical Cost Personal Datasets



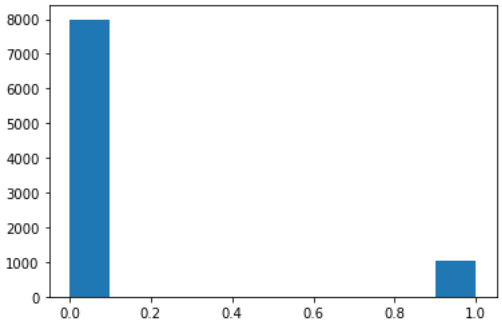
(c) Bike Sharing Demand (log)



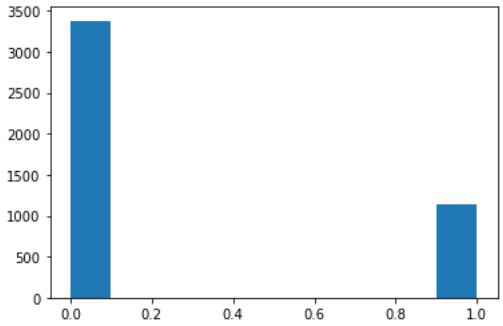
(d) Communities and Crime



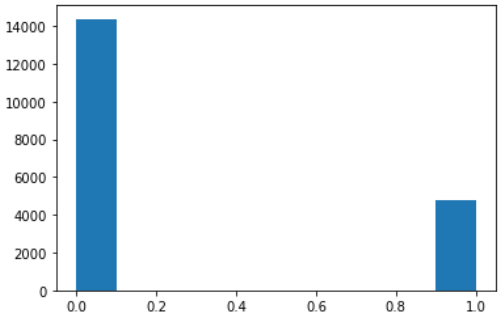
(e) Titanic



(f) Bank Marketing



(g) Adult Income



(h) Job Change of Data Scientists

Figure A1: Histograms of the target variables of the extra data sets

Appendix 2: Tables

Table A1: Regression results for the Fish Market data set

Out-of-sample	**MSE**	**Relative error**	**Correlation**
Linear regression	8695	0.029	0.977
Lasso	9479	0.033	0.975
Random forest	3643	0.013	0.989
Model tree	3786	0.016	0.989
MARS	2923	0.012	0.992
PLM	**2195**	**0.008**	**0.994**
Univariate trees	9126	0.029	0.977
Fully grown tree	7263	0.031	0.977
In-sample	**MSE**	**Relative error**	**Correlation**
Linear regression	9253	0.037	0.960
Lasso	9477	0.038	0.959
Random forest	860	0.005	0.997
Model tree	2348	0.011	0.990
MARS	2811	0.014	0.988
PLM	4526	0.017	0.981
Univariate trees	7233	0.028	0.969
Fully grown tree	**48**	**0.000**	**1.000**

Best results are displayed in bold

Table A2: Regression results for the Medical Cost Personal Datasets data set

Out-of-sample	**MSE**	**Relative error**	**Correlation**
Linear regression	36,656,681	0.155	0.841
Lasso	36,511,108	0.154	0.841
Random forest	21,763,150	0.118	0.908
Model tree	21,550,132	0.123	0.909
MARS	22,326,555	**0.115**	0.906
PLM	24,474,943	0.127	0.896
Univariate trees	36,875,799	0.144	0.840
Fully grown tree	**20,753,116**	0.118	**0.912**
In-sample	**MSE**	**Relative error**	**Correlation**
Linear regression	36,445,793	0.119	0.875
Lasso	36,787,225	0.121	0.874
Random forest	**17,566,262**	**0.076**	**0.942**
Model tree	18,684,343	0.084	0.938
MARS	22,992,488	0.094	0.923
PLM	21,836,837	0.088	0.927
Univariate trees	35,219,935	0.110	0.879
Fully grown tree	18,061,250	0.081	0.940

Best results are displayed in bold

Table A3: Regression results for the Bike Sharing Demand data set

Out-of-sample	**MSE**	**Relative error**	**Correlation**
Linear regression	0.453	0.083	0.894
Lasso	0.453	0.083	0.894
Random forest	**0.200**	**0.039**	**0.955**
Model tree	0.433	0.082	0.899
MARS	0.344	0.067	0.921
PLM	0.422	0.083	0.900
Univariate trees	0.450	0.083	0.895
Fully grown tree	0.213	0.042	0.952
In-sample	**MSE**	**Relative error**	**Correlation**
Linear regression	0.440	0.086	0.894
Lasso	0.440	0.086	0.894
Random forest	**0.160**	**0.033**	**0.963**
Model tree	0.414	0.084	0.900
MARS	0.344	0.070	0.918
PLM	0.428	0.085	0.898
Univariate trees	0.437	0.086	0.895
Fully grown tree	0.177	0.035	0.959

Best results are displayed in bold

Table A4: Regression results for the Communities and Crime data set

Out-of-sample	**MSE**	**Relative error**	**Correlation**
Linear regression	0.019	0.175	0.812
Lasso	0.019	0.175	0.812
Random forest	**0.017**	**0.168**	**0.831**
Model tree	0.023	0.211	0.782
MARS	0.019	0.179	0.810
PLM	0.020	0.180	0.811
Univariate trees	0.020	0.169	0.807
Fully grown tree	0.020	0.179	0.807
In-sample	**MSE**	**Relative error**	**Correlation**
Linear regression	0.016	0.147	0.840
Lasso	0.018	0.182	0.813
Random forest	**0.006**	**0.065**	**0.947**
Model tree	0.013	0.118	0.872
MARS	0.018	0.170	0.820
PLM	0.016	0.149	0.841
Univariate trees	0.014	0.124	0.863
Fully grown tree	0.015	0.140	0.850

Best results are displayed in bold

Table A5: Significance of the differences in MSE for the Fish Market data set

	LR	Lasso	Random forest	Model tree	MARS	PLM	Univariate trees	Fully grown tree
LR	-	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Lasso	0.000	-	0.000	0.000	0.000	0.000	0.000	0.000
Random forest	0.000	0.000	-	0.000	0.000	0.000	0.000	0.000
Model tree	0.000	0.000	0.000	-	0.000	0.000	0.000	0.000
MARS	0.000	0.000	0.000	0.000	-	0.000	0.000	0.000
PLM	0.000	0.000	0.000	0.000	0.000	-	0.000	0.000
Univariate	0.000	0.000	0.000	0.000	0.000	0.000	-	0.000
Fully grown	0.000	0.000	0.000	0.000	0.000	0.000	0.000	-

Table A6: Significance of the differences in MSE for the Medical Cost Personal Datasets data set

	LR	Lasso	Random forest	Model tree	MARS	PLM	Univariate trees	Fully grown tree
LR	-	0.000	0.000	0.000	0.000	0.000	0.000	0.000
Lasso	0.000	-	0.000	0.000	0.000	0.000	0.000	0.000
Random forest	0.000	0.000	-	0.000	0.000	0.000	0.000	0.000
Model tree	0.000	0.000	0.000	-	0.000	0.000	0.000	0.000
MARS	0.000	0.000	0.000	0.000	-	0.000	0.000	0.000
PLM	0.000	0.000	0.000	0.000	0.000	-	0.000	0.000
Univariate	0.000	0.000	0.000	0.000	0.000	0.000	-	0.000
Fully grown	0.000	0.000	0.000	0.000	0.000	0.000	0.000	-

Table A7: Significance of the differences in MSE for the Bike Sharing Demand data set

	LR	Lasso	Random forest	Model tree	MARS	PLM	Univariate trees	Fully grown tree
LR	-	0.480	0.000	0.061	0.000	0.010	0.386	0.000
Lasso	0.480	-	0.000	0.068	0.000	0.011	0.405	0.000
Random forest	0.000	0.000	-	0.000	0.000	0.000	0.000	0.069
Model tree	0.061	0.068	0.000	-	0.000	0.215	0.105	0.000
MARS	0.000	0.000	0.000	0.000	-	0.000	0.000	0.000
PLM	0.010	0.011	0.000	0.215	0.000	-	0.020	0.000
Univariate	0.386	0.405	0.000	0.105	0.000	0.020	-	0.000
Fully grown	0.000	0.000	0.069	0.000	0.000	0.000	0.000	-

Table A8: Significance of the differences in MSE for the Communities and Crime data set

	LR	Lasso	Random forest	Model tree	MARS	PLM	Univariate trees	Fully grown tree
LR	-	0.494	0.459	0.435	0.500	0.498	0.492	0.490
Lasso	0.494	-	0.465	0.429	0.494	0.492	0.486	0.484
Random forest	0.459	0.465	-	0.394	0.459	0.457	0.450	0.449
Model tree	0.435	0.429	0.394	-	0.434	0.436	0.443	0.444
MARS	0.500	0.494	0.459	0.434	-	0.498	0.491	0.490
PLM	0.498	0.492	0.457	0.436	0.498	-	0.493	0.492
Univariate	0.492	0.486	0.450	0.443	0.491	0.493	-	0.499
Fully grown	0.490	0.484	0.449	0.444	0.490	0.492	0.499	-

Table A9: Classification results for the Titanic data set

Out-of-sample	**ROC-AUC**	**Accuracy**	**Precision**	**Recall**	**F1**
Linear regression	0.868	0.805	0.761	0.778	0.769
Lasso	0.868	0.805	0.755	**0.789**	0.772
Random forest	0.867	0.809	0.788	0.744	0.766
Logistic model tree	0.861	0.819	0.793	0.767	**0.780**
MARS	0.873	0.791	0.785	0.689	0.734
Univariate trees	0.861	0.800	0.758	0.767	0.762
Fully grown tree	**0.874**	**0.819**	**0.807**	0.744	0.775
In-sample	**ROC-AUC**	**Accuracy**	**Precision**	**Recall**	**F1**
Linear regression	0.857	0.792	0.731	0.760	0.745
Lasso	0.858	0.792	0.731	0.760	0.745
Random forest	**0.923**	**0.854**	**0.832**	0.795	**0.813**
Logistic model tree	0.883	0.834	0.806	0.770	0.788
MARS	0.890	0.836	0.811	0.770	0.790
Univariate trees	0.872	0.808	0.739	**0.805**	0.770
Fully grown tree	0.891	0.834	0.794	0.790	0.792

Best results are displayed in bold

Table A10: Classification results for the Bank Marketing data set

Out-of-sample	**ROC-AUC**	**Accuracy**	**Precision**	**Recall**	**F1**
Linear regression	0.898	0.832	0.393	0.819	0.531
Lasso	0.903	0.832	0.393	0.819	0.531
Random forest	**0.913**	**0.853**	**0.424**	0.746	**0.541**
Logistic model tree	0.898	0.832	0.393	0.819	0.531
MARS	0.895	0.830	0.387	0.803	0.523
Univariate trees	0.904	0.829	0.390	0.829	0.530
Fully grown tree	0.903	0.816	0.372	**0.848**	0.517
In-sample	**ROC-AUC**	**Accuracy**	**Precision**	**Recall**	**F1**
Linear regression	0.922	0.857	0.440	0.852	0.580
Lasso	0.921	0.858	0.441	0.845	0.580
Random forest	**0.971**	**0.903**	**0.549**	**0.933**	**0.692**
Logistic model tree	0.922	0.857	0.440	0.852	0.580
MARS	0.917	0.856	0.438	0.837	0.575
Univariate trees	0.927	0.857	0.440	0.863	0.583
Fully grown tree	0.936	0.852	0.435	0.906	0.588

Best results are displayed in bold

Table A11: Classification results for the Adult Income data set

Out-of-sample	**ROC-AUC**	**Accuracy**	**Precision**	**Recall**	**F1**
Linear regression	0.898	0.801	0.582	0.839	0.688
Lasso	0.903	0.799	0.577	0.859	0.690
Random forest	**0.912**	0.807	0.585	**0.893**	0.707
Logistic model tree	0.898	0.801	0.582	0.839	0.688
MARS	0.909	0.822	0.615	0.856	**0.715**
Univariate trees	0.904	0.809	0.596	0.833	0.695
Fully grown tree	0.901	**0.842**	**0.723**	0.636	0.677
In-sample	**ROC-AUC**	**Accuracy**	**Precision**	**Recall**	**F1**
Linear regression	0.909	0.811	0.583	0.848	0.691
Lasso	0.903	0.800	0.564	0.860	0.681
Random forest	**0.937**	0.816	0.583	**0.921**	**0.714**
Logistic model tree	0.909	0.811	0.583	0.848	0.691
MARS	0.907	0.814	0.587	0.841	0.692
Univariate trees	0.917	0.817	0.590	0.859	0.700
Fully grown tree	0.925	**0.866**	**0.766**	0.665	0.712

Best results are displayed in bold

Table A12: Classification results for the Job Change of Data Scientists data set

Out-of-sample	**ROC-AUC**	**Accuracy**	**Precision**	**Recall**	**F1**
Linear regression	0.764	0.714	0.448	0.731	0.555
Lasso	0.764	0.714	0.447	0.731	0.555
Random forest	**0.782**	0.752	0.494	0.758	**0.598**
Logistic model tree	0.770	0.722	0.457	0.743	0.566
MARS	0.768	0.707	0.442	**0.773**	0.563
Univariate trees	0.776	0.739	0.478	0.746	0.583
Fully grown tree	0.780	**0.753**	**0.496**	0.753	0.598
In-sample	**ROC-AUC**	**Accuracy**	**Precision**	**Recall**	**F1**
Linear regression	0.787	0.735	0.483	0.748	0.587
Lasso	0.787	0.735	0.483	0.748	0.587
Random forest	**0.833**	**0.770**	**0.530**	0.785	**0.632**
Logistic model tree	0.794	0.739	0.488	0.758	0.594
MARS	0.790	0.720	0.467	**0.785**	0.586
Univariate trees	0.800	0.759	0.514	0.750	0.610
Fully grown tree	0.814	0.769	0.529	0.765	0.625

Best results are displayed in bold