Predicting Award Prices of First Price Sealed Bid Procurement Auctions

Fabian Blasch

Supervisors:
Univ.Prof.Dr. Jesús Crespo Cuaresma
Dr. Katharina Fenz

October 8, 2022

Abstract

Public auctions are a vital instrument for governments to procure contracts. Accordingly, an award price prediction model may be of great use for public entities like the Colorado Department of Transportation (CDOT) since it would enable them to plan their budget more accurately. This thesis compares different methods to predict the award price of first price sealed bid procurement auctions held by the CDOT. The best performing model is a lasso regression model. Besides performing better than the other applied methods, the ℓ_1-penalized regression model also performs drastically better than the engineer's estimate, which is used as a benchmark. This thesis further examines whether the interaction of certain bidders is associated with a significant change in the award price. The existence of such effects might allude to price collusion. For the dataset containing 430 auctions, no significant interactions are detected.

Contents

1. Introduction

The importance of public procurement quickly becomes apparent when looking at the sheer volume of contracts that is awarded through public auctions. The authorities of the European Union for example spent around 14% of their GDP on public procurement in 2017 (Rodríguez et al., 2020). Similar observations can also be made for many states in the U.S. One example of particular importance for this thesis is Colorado. The Colorado Department of Transportation (CDOT), is responsible for procurement of street and bridge construction contracts. As displayed below the budget for transportation is ranked as number four on the largest capital expenditures in the state's budget, after education, health care and human services. Of the approximate 2 billion dollars spent on transportation in 2021, the CDOT awarded $790 million in contracts, to design, repair and create bridges and highways (CDOT, 2021).
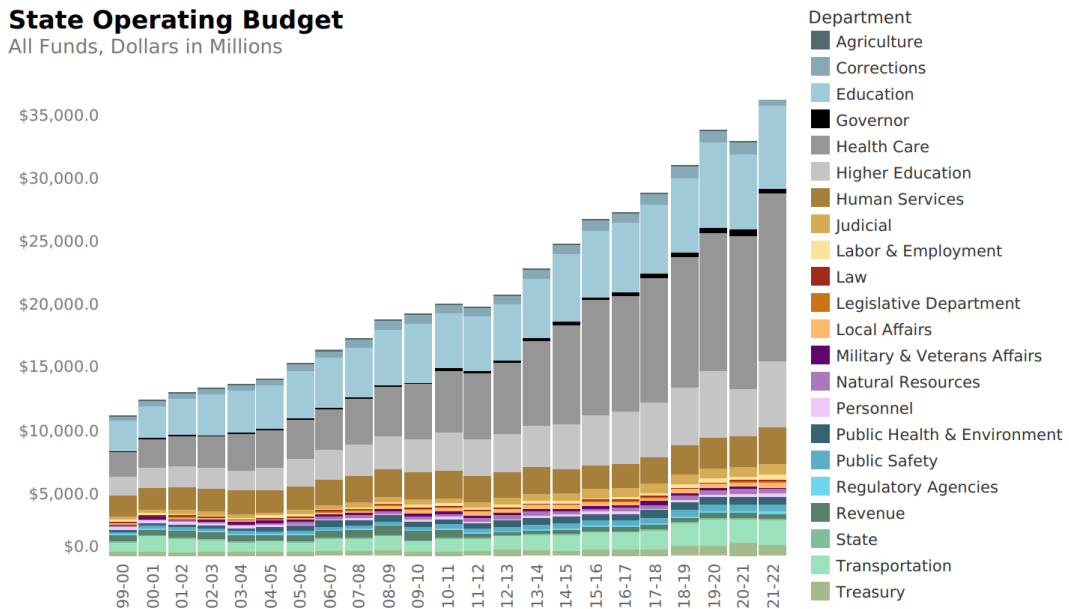
Figure 1: Colorado State Operating Budget (Colorado General Assembly, 2022)

Given that the contracts awarded through procurement auctions are such a substantial part of the budget, every potential improvement to the process may greatly increase the efficiency of the tax payers money that is spent. Accordingly, a closer examination of the process including a prediction model for the auctions' award prices may support public procurement agencies like the CDOT with budget planning. Additionally, further analysis of the underlying data in respect to the interactions of particular bidders and the associated effect on award prices is also highly relevant for public procurement agencies. This thesis thus provides an analysis of different models, that predict the award price of an auction given input information available through the bid tabs that are published on

the official website of the CDOT. In particular, four different model types with varying preprocessing schedules are compared in terms of their predictive power: Standard linear regression, elastic net regression, random forests and an eXtreme gradient boosting model. Those methods were chosen, as they are the most popular machine learning algorithms for tabular data. Additionally, all of the mentioned methods, some of them combined with a preprocessing schedule, are able to deal with a large number of potential explanatory variables. To assess whether the combination of certain bidders leads to higher award prices, recently discovered post-selection inference methods are utilized (Tibshirani et al., 2019).

The presentation of the theoretical concepts in the methods section is kept concise, whenever advanced concepts need introduction a reference is provided. Additionally, great emphasis is put on keeping the application as simple as possible. All applied illustrations of algorithms are written in the simplest form attainable, while attempting to avoid ambiguity. The idea behind this approach is to simultaneously provide an approachable reading experience for people who have not enjoyed formal mathematical training while retaining some formality in the descriptions of the applied methods. The remaining paper is structured as follows, first the data extraction process from the PDFs provided on the CDOT's website is described. Then the general process of procurement is outlined, this also entails a concise description of the auction process utilized by the CDOT. The following section covers the methods that are applied, not only in respect to the different models used for prediction but also for the different preprocessing schedules that are applied and the post-selection inference that is used. The thesis concludes with the results for the best predictive model utilizing linear and quadratic loss functions and the results of the analysis of bidder interactions. All the code that was used to generate the presented results may be found here: *github.com/Base-R-Best-R/Auction*.

2. Data

All the information about the procurement contracts, is obtainable through the bid tab archive on the official website of the Colorado Department of Transportation. The information is provided in PDF documents. In each of those files the following information of the respective auction is provided.

- A table listing all submitted bids, including a unique identifier for each of the participating bidders
- A contract description
- An engineer's estimate
- The contract ID
- The letting date
- Either the amount of time given to complete all the contractual obligations, or a completion due date
- The county in which the contract is to be completed in

For illustrative purposes, Figure 2 displays an example of a bid tab, in particular the second page, which contains the vendor ranking as well as the contract description and the remaining information listed above.

| | | | Colorado Department Of Transportation | | | Printed On: | **11/17/2015** |
| | | | **Vendor Ranking** | | | | Page 1 of 1 |

Letting No:	20151112	Contract ID:	C19868	Project(s):	STU1211-084
Letting Date:	November 12, 2015	Region:	1		
Letting Time:	10:00 AM	Contract Time:	260 WORKING DAYS	Counties:	JEFFERSON, REGION 1

Contract Description:
SH121(WADSWORTH)-HIGHLAND DR-10TH AVE-JEFFERSON CO

THIS PROJECT IS LOCATED ON WADSWORTH BETWEEN HIGHLAND AND 10TH.

CONSTRUCTION WILL INCLUDE A FULL CONSTRUCTION WITH WIDENING OF ONE
LANE IN BOTH DIRECTIONS, AND A MULTI MODAL TRAIL ON BOTH SIDES. THE
MAINLINE PAVING WILL BE CONCRETE. THE WORK ALSO INCLUDES A CONCRETE
BOX CULVERT NEAR HIGHLAND TO CARRY LAKEWOOD GULCH UNDER WADSWORTH.

CDOT WILL ONLY BE ACCEPTING ELECTRONIC BIDS FOR THIS PROJECT. PLEASE
CONTACT BID EXPRESS CUSTOMER SERVICE AT 1-888-352-2439 TO OBTAIN AN
ACCOUNT IF NECESSARY.

Rank	Vendor ID	Vendor Name	Total Bid	Percent Of Low Bid	Percent Of Estimate
0	-EST-	Engineer's Estimate	$9,821,027.20	91.58%	100.00%
1	870A	SEMA CONSTRUCTION, INC.	$10,723,550.00	100.00%	109.19%
2	884A	HAMON INFRASTRUCTURE, INC.	$10,817,000.00	100.87%	110.14%
3	1275A	CASTLE ROCK CONSTRUCTION COMPANY OF COLORADO, LLC	$10,817,845.03	100.88%	110.15%
4	065A	CONCRETE WORKS OF COLORADO INCORPORATED	$11,614,565.78	108.31%	118.26%
5	232A	AMERICAN CIVIL CONSTRUCTORS, INC. dba ACC Mountain West	$12,338,888.00	115.06%	125.64%

Figure 2: Bid Tab Example

2.1. Scraping

In order to obtain all the archived bid tabs, the html code of the website was first examined using a google chrome extension called *SelectorGadget* (Cantino & Maxwell, 2022). This tool allows one to identify html nodes, that website contents are associated with. In the case of the bid tab archive, the html node carrying the links to the individual bid tabs is "<td a>". Once this html node is discovered and the consistency across different years in the archive is ensured, the download is easily achieved by looping over the links and downloading the respective PDFs. The hyperlink extraction was performed utilizing *rvest*, by Wickham (2022). For the remaining steps in the data extraction process, a distinction will be made for text based information and tabular data.

2.1.1. Text Based Information

The structure of the text based information allows us to filter the individual parts via regular expressions. Especially, for the letting data, the contract ID, and the county this required no further data cleaning steps. Unfortunately, this is not the case for the

contract time and the contract description.

The contract time was not as straightforward to obtain, since the way it is reported is inconsistent across documents. Most of the time, it is reported as working days until all contractual obligations have to be fulfilled. Seldom, however, the bid tab contains a completion date instead. Accordingly, to achieve consistency across documents all completion dates were converted to contract time. This was achieved by first adding 60 days to the letting date, as this is the number of days that the CDOT reports as the expected time between the letting date and the start of the work on site. Then, the difference in days between the completion date and the starting date were computed. As said difference is only supposed to contain working days the following holidays as well as all weekends were subtracted from the difference between starting date and completion date.

- New Year's Day
- Dr. Martin Luther King, Jr. Day
- President's Day
- Memorial Day
- Juneteenth
- Independence Day
- Labor Day
- Frances Xavier Cabrini Day
- Veterans Day
- Thanksgiving
- Christmas

The computation was executed utilizing the R package *bizdays* (Freitas, 2022). The package enables the user to generate custom calenders. The difference in starting and completion date was therefore easily calculated by setting up a custom calender with the holidays listed above as well as all Saturdays and Sundays. Then using this calender, the difference between two dates only takes working days into account.

The only remaining text based information is the contract description. So far, none of the text based information required extensive preprocessing to obtain variables that can be represented in a tabular format. In the case of the contract description this is not the case. In order to convert the contract description into a format that may be represented in a table, the descriptions were first tokenized. Tokenization refers to splitting the input text into single unique words, i.e., splitting the sentences on spaces and removing all forms of punctuation. The result is then a vector of tokens. Said tokens were then scanned for spelling mistakes utilizing the R package *hunspell* (Ooms, 2020). Once the misspelled words were corrected, stop words were removed from the

list of tokens. Stop words are words that have no inherent signal associated with their use, examples for such words in the English language would be "a", "is" and "the". In natural language processing there is not necessarily one list of stop words, depending on the context different libraries of stop words may be used to remove as much noise as possible from textual data while leaving the signal associated with a series of words in tact. In the case of this thesis, a combination of five different libraries of stop words was used. All of those libraries, "snowball" , "stopwords-iso", "smart", "marimo" and "nltk" are available through the R package *stopwords*, by Benoit et al. (2021). After filtering out the stop words, the remaining words were then stemmed. Stemming refers to the process in which a word is reduced to its root. This means, that words that carry an identical signal are reduced to the same shortest common substring. Consider the following three words, replacing, replaced, replacement. All those words carry the information that something needs replacement. The language specific circumstances that determine the affixes are not relevant for the information extraction and thus all the aforementioned words are shortened to "replac" (Silge & Robinson, 2017). After stemming, to remove any remaining misspelled words and also for potential removal of unwanted information, all stemmed words were written to an excel file and checked manually. Given that all stop words were already removed and the remaining words were reduced to their stem, this was a very feasible task, resulting in a file with around 2000 words to check. Below we observe the top 40 most frequent words that result from our text mining endeavours.
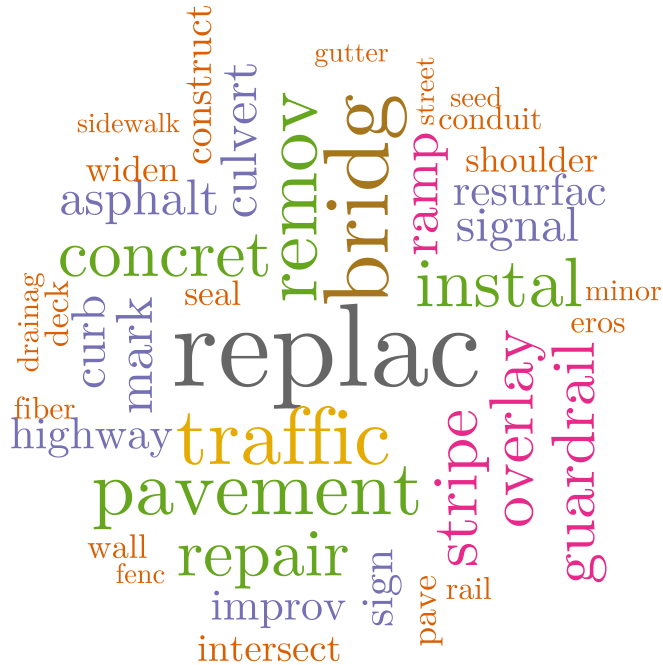

Figure 3: Top 40 Stemmed Description Words

2.1.2. Tabular Data

As displayed in Figure 2, the table in each of the auctions' PDFs contains information on the submitted bids, the bidders' identity and an engineer's estimate. To extract the table containing this information, the package *tabulizer* by Leeper (2018) is utilized. This package provides bindings for the Tabula PDF extractor, written in Java. In particular, two functions of the library were combined to write a wrapper for the table extraction. First the function *extract_tables()* was used to attempt automized table detection and subsequent extraction. Unfortunately, however, there are quite a few cases in which the automatic table detection failed because some auctions only attract one or two bidders. The resulting tables that summarize those auctions have very few rows and thus the automatic detection does not recognize them as tables. Accordingly, if the output of *extract_tables()* is empty, the implemented wrapper calls *extract_areas()*. This function allows the user to specify an area via the R plot-pane to assess where exactly the table is located. Once the location is passed manually, which was necessary for around 20-25% of the cases, the extraction works as intended. To finally obtain the tables, the wrapper is used to loop over the PDFs.

2.2. Descriptive Statistics

The data that results from the scraping process is based on an auction level, meaning that each of the 430 auctions that were scraped between 2015 and 2019 represents one row. The final columns are thus:

- Contract ID
- County
- Letting month
- Letting year
- Contract time
- Number of bidders
- Engineer's estimate
- Award price
- 169 binary variables, representing the bidder identities
- 652 binary variables, representing pair-wise bidder interaction terms
- 258 binary variables, representing the contract description hit words

The interaction terms were generated in order to see whether a combination of certain bidders leads to a lower or higher award price. Those terms may also be used to perform unsupervised collusion detection utilizing post-selection inference for ℓ_1-penalized

models. This idea is further outlined in Section 5.3. To obtain a concise overview of the data, histograms and bar charts of the variables are provided below.
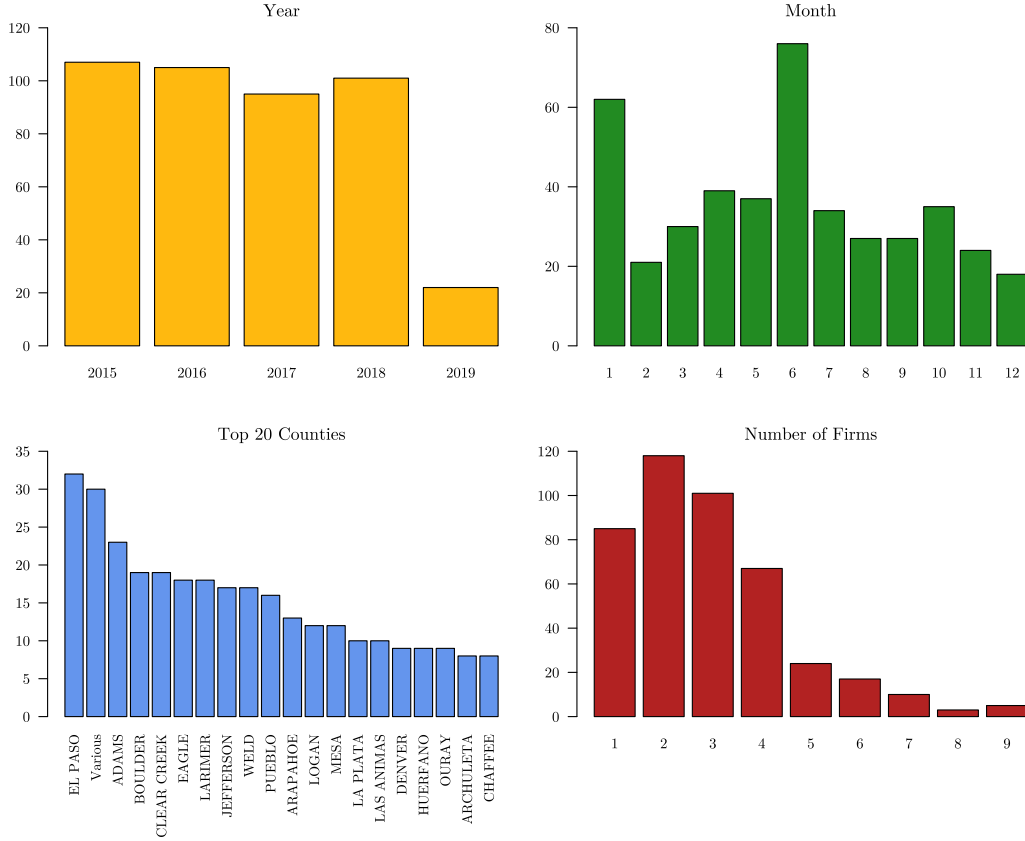
Figure 4: Date, Counties and Number of Firms

We observe, that the sample of auctions in the dataset were held between 2015 and 2019, in which the auction frequency is the highest in January and June. Further we learn, that most contracts are to be completed in El Paso followed by a combination of various counties. In regard to the number of bidders per auction, most of the auctions attract between 1 to 4 bidders, with the maximum being 9 competing firms.

Figure 5: Bidders and their Interactions

The depiction of the top 20 bidders represents the unique identifiers of the firms that submitted the most bids. It is interesting to see that there seem to be a handful of firms that compete in drastically more auctions than others. Additionally, we observe that the two firms that submitted the most bids in the same auctions compete in slightly more than 10% of the auctions in the dataset.
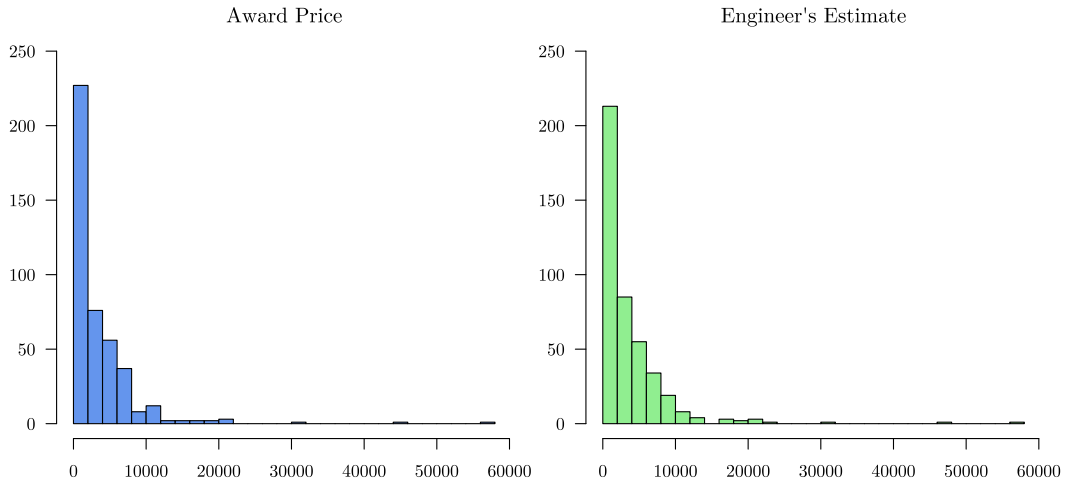
Figure 6: Award Price and Engineer's Estimate

The last two plots, enable us to gain insights about the distribution of the award price and the engineer's estimate. Both variables are right skewed and to the naked eye the

engineer's estimate seems to resemble the distribution of the award price quite well. This is a strong indicator for the engineer's estimate as a predictor. The engineer's estimate is also important as it serves as a benchmark for prediction. Every model that can not beat the engineer's estimate in predicting the award price is not particularly useful, this is further discussed in Section 5.

3. Economic Operationalization

Before presenting the formalization, a quick overview of the auction design of the Colorado Department of Transportation is presented. This renders modelling assumptions more comprehensible. The CDOT provides a document on their website that summarizes the entire auction process including all preregistration processes, that firms have to traverse before they are eligible to submit bids (CDOT, 2018).

The auction design created by the CDOT is a standard sealed bid first price auction, i.e., contracts are advertised for bids online, firms that have been prequalified for bidding for a certain contract may submit their bids online or in person. Once a preset date passes, a representative of the CDOT opens all submitted bids and announces the lowest bidder, who is awarded the contract. The prequalification that was mentioned is of vital importance for this thesis. Given that all firms have to be prequalified for bidding, one may utilize bidder identity and bidder interactions when predicting award prices. If there was no prequalification, then the auctioneer would not have knowledge of all potential bidders until no more bids are to be received. At this point there is also no need for a prediction of the award price. Besides a prequalification process, the CDOT also implements a reserve price. Every project is assigned an engineer, who is responsible to calculate an engineer's estimate of the cost of completion. Should the lowest bid that is submitted exceed the engineer's estimate by more than 10%, then the lowest bidder is only awarded the contract if a secondary assessment of the CDOT ends with the conclusion that the lowest apparent bid is indeed a reasonable price for execution of the contractual obligations. This secondary process to bypass the initial reserve price was necessary for 17% of the auctions in the dataset that is used in this thesis. In regards to the frequency of bid rejection, for our modelling assumption it is important to know that the secondary assessment process very rarely results in a rejection of the lowest apparent bid (CDOT, 2018).

3.1. First Price Sealed Bid Auctions

For the applied nature of this thesis a formalization of the auction process in not a necessary prerequisite for a good prediction model, however, a sound understanding of all the economic agents at play might yield insights into the potential benefit of a prediction model for participating firms and it may also provide us with expectations for the post-selection inference that is explored in Section 4.1.1. Given that this chapter is not strictly necessary for the reader to understand the main findings of this thesis a basic understanding of game theory is assumed. The interested reader is referred to, Tadelis (2012), who provides sound definitions and explanations of all the required concepts.

The description of bidding in the first-price sealed-bid procurement auction (FPSBPA) is denoted in accordance with Milgrom and Weber (1982) and Schmalensee and Willig (1989). For the following formalization let uppercase letters represent random variables

and their lowercase counterpart a single realisation. Consider an auction environment with n potential bidding firms, all of them submit bids in hopes to procure a contract from the auctioneer. Each of those firms, which are indexed by i observe a real valued signal x_i. One can think of X_i as private information that each firm has about their own cost structure and other private information that a bidder can posses, that influences the value of a contract to a firm. Further, let V represent the characteristics that determine the value of the contract to the firms, that are common knowledge among competing bidders. For illustrative purposes, consider an auction for a contract that involves the repair of a street that leads up a mountain. The information that can be extracted from the contract description is common knowledge, i.e., it influences the value of the contract of every firm. Every bidder is aware that a street, which is steep is more cost intensive to repair. When modelling the bidders behaviour we assume that this is common knowledge, such common values are represented by V.

In the auction modelling literature, a distinction is made between an auction environment of private values, i.e., each firm knows their own valuation and is only uncertain about how other firms value the contract. The counterpart to this auction environment is one of pure common values. In this case all bidders have the same common value, however, prior to the execution of the contractual obligations they have no accurate ex ante estimate of the value that may be extracted from completing this contract. This common value paradigm is important when modelling because, given that firms have common metrics that make contracts more or less attractive, it renders the private information that firms may posses relevant for rival bidders and will thus influence the ex post value once all bids are revealed.

To continue the formalization of the auction environment, let firm $i's$ payoff be represented by $U_i = u_i(V, X_i, X_{-i})$, where X_{-i} represents the private information possessed by firm $i's$ rivals. Most of the time it is assumed that firms are risk neutral, i.e., expected utility of profit is simply equal to the expectation of a linear function of capital gained, and maximizing it is equivalent to maximizing expected profit itself. The payoff or the value of a contract to a firm may then be defined as the expectation of U_i given the private information of firm i, i.e., $\mathbb{E}(U_i|X_i = x_i, X_{-i} = x_{-i})$. It is important to point out that in a common values environment the distribution of V conditional on x_i is dependent on the realization of X_{-i}. This means that from the perspective of firm i, other firms may have private information about characteristics of a contract that influence its value to all competing firms and thus also the value in the eyes of firm i. Accordingly, let the bidding strategy be a mapping $\beta : X_i \rightarrow \mathbb{R}^+$, that maps the private information of firm i (cost structure, private knowledge about a contract obtained through on site inspections, ...) into the non-negative real numbers. To ensure that this function is invertible, in the literature it is usually assumed that β is a bijection that is strictly increasing in the signal X_i. Again, assuming that X_i represents the cost structure of firm i and other private information, a ceteris paribus decrease in costs is then associated with a decrease in the price of submitted bids. Now that the basic formalities of the auction environment have been set, we may consider the "winner's curse", which

assumes the event of winning an auction as an informative event in respect to the value of the contract. The idea is that a firm that manages to procure a contract through an auction must be the lowest bidder, and this also means that no other firm was willing to execute the contractual obligations for the same price. More formally, assume that the realization v is unknown and carries a significant common component. Additionally, we may assume that all firms have varying amounts of private information about the contract. They are aware that other firms have private information but they are uncertain about other firms' valuations of the contract. Further, to simplify the model we assume that the signals x_i are drawn from a common distribution F and that the value of the contract is mainly determined by the common component of bidders' valuations, i.e, w.l.o.g for firm 1, the ex ante valuation of the contract is equal to:

$$v_1(x_1) = \mathbb{E}[V|X_1 = x_1] = \int v f_{V|X_1}(v|x_1)dv.$$

Then in equilibrium, assuming that the bidding function is increasing in the cost realization, the winner is the firm with the lowest cost draw from the distribution but also influenced by the firms private information about the contract. In the literature this is formalized by stating that the lowest bid is announced by the firm that incurs the lowest signal (Schmalensee & Willig, 1989). This intuitively makes sense as the signal is defined on the real numbers and the lowest signal will thus be identical to the lowest contract execution costs and consequently the highest profit to be made. By Jensen's inequality, given that the maximum is a convex function[1],

$$\mathbb{E}[\max\{V_i|v\}] \leq \max\{\mathbb{E}[V_i|v]\} = v.$$

This represents the winner's curse as the firm with the lowest signal submits the lowest bid. Further as Milgrom and Weber (1982) state, w.l.o.g, for firm 1 this means:

$$\mathbb{E}[V|X_1 = x, Y_1 > x] < \mathbb{E}[V, X_1 = x].$$

Where Y_1 is the firm with the lowest signal among firm 1's rivals. We thus find that the bidder that incurs the lowest signal associated with the completion of all contractual obligations and therefore the highest potential profit to be extracted from the procurement of the contract should bid less than the firm's ex ante estimate of V_1, simply because the assignment of the contract to the firm as the lowest bidder is an informative event. Especially firms that are not as established and might have difficulties to obtain the expertise to properly assess the difficulties associated with completion of a contractual agreement will suffer more severely from the winner's curse. This is relevant for this thesis as an award price prediction that gives a reasonable estimate for the costs associated with the completion of a contract will change the informational asymmetries and thus also lessen the severity of the winner's curse, should the award price prediction

---

[1]A proof is provided in Appendix A.1.

be shared with the bidders (Rodríguez et al., 2020).

Now that we have discovered another potential beneficiary of an award price prediction model, we may assess the strategic bidding behaviour of firms. To further simplify the strategic behaviour we may again assume that firms signal realisations x_i are drawn from a common distribution, i.e., the values associated with a contract $(V, X_1, ..., X_n)$ are governed by some joint distribution denoted by F. Further, we may assume that the number of firms that submit bids is common knowledge and that the reserve price is not binding. Essentially, this means that the firms believe that bidding above the 110% threshold will not lead to a reissue of the contract but to a contract award in the secondary process previously described. Given that this game is symmetric, we may again focus on firm 1, w.l.o.g. The firm aims to maximize its expected profit, i.e.,

$$\arg\max_b \ (b - x)\{1 - F[\beta^{-1}(b)]\}^{n-1}.$$

Where $(b - x)$ is the profit that the firm expects if it were to win the contract and the remaining term is the probability that all other competing bidders incur a higher signal realization associated with the execution of the contractual obligation, and thus submit higher bids. Accordingly, by the previously mentioned assumption of the strictly increasing function β, the lowest signal leads to the lowest bid. From the maximization problem, we obtain the following FOC:

$$\{1 - F[\beta^{-1}(b)]\}^{n-1} + (b - x)(n - 1)\{1 - F[\beta^{-1}(b)]\}^{n-2}\frac{d\beta^{-1}}{db}\{(-1)f[\beta^{-1}(b)]\} \overset{!}{=} 0$$

$$\Longleftrightarrow \frac{\{1 - F[\beta^{-1}(b)]\}^{n-1}}{\{1 - F[\beta^{-1}(b)]\}^{n-2}} = (b - x)(n - 1)f[\beta^{-1}(b)]\frac{d\beta^{-1}}{db}$$

$$\Longleftrightarrow 1 - F[\beta^{-1}(b)] = (b - x)(n - 1)f[\beta^{-1}(b)]\frac{d\beta^{-1}}{db}.$$

Knowing that in the Bayesian NE, $\beta^{-1}(b) = x$ and $\frac{d\beta^{-1}(b)}{db} = \frac{1}{\beta'(x)}$, we then find:

$$\beta'(x) = \frac{[\beta(x) - x](n - 1)f(x)}{1 - F(x)}.$$

This is a differential equation, from which we may obtain the Bayesian NE bid function (Decarolis, 2018):

$$\beta(x) = x + \frac{\int_x^\infty [1 - F(t)]^{(n-1)} dt}{[1 - F(x)]^{(n-1)}}.$$

We thus learn that the bid resulting from the cost structure of the firm is a surcharge added to the realization of X. Again, we may interpret this as the private information of firm i on the cost of completion of the contractual obligations, e.g., information about the firms cost structure, information about execution costs obtained from on-site

inspections, etc. Further, we observe that the surcharge depends on the number of firms in an auction and the joint distribution F. We have to keep in mind, however, that this simple result is based on quite restrictive assumptions. Especially, the risk neutrality of firms is not particularly realistic. Nonetheless, given those simplifying assumptions, the results displayed in section 5 will show whether the derived economic intuition of firms' behaviour is observable in the data.

4. Methods

This section contains precise descriptions of the methods used to generate the results presented in Section 5.

4.1. Elastic Nets

When the number of features in a dataset is significantly larger than the number of observations, elastic net regularization offers a solution to fit linear models using a convex combination of the lasso and ridge penalty. Further, the regularization may be used to better optimize the bias variance trade-off when compared to a linear regression estimated via ordinary least squares.

The description of the elastic net regularization is denoted in accordance with the seminal paper by Zou and Hastie (2003). Assume that a dataset has n observations and p explanatory variables. As it is the case with the auction dataset, the number of explanatory variables may greatly exceed the number of observations, i.e., $p >> n$. Let $\mathbf{y} = (y_1, ..., y_n)^T$ be the response variable and $\mathbf{X} = [x_1|...|x_p]$ the model matrix, containing the hot encoded explanatory variables. This means that all factor variables have to be converted to individual dummy variables for each level. Subsequently, each column in the model matrix is then centered and standardized. This ensures that the regularization is consistent across variables, since the regularization process is not scale invariant. For any combination of non-negative λ_1 and λ_2, Zou and Hastie (2003), define the elastic net criterion as,

$$l(\lambda_1, \lambda_2, \boldsymbol{\beta}) = |\mathbf{y} - \mathbf{X}\boldsymbol{\beta}|^2 + \lambda_1|\boldsymbol{\beta}|^2 + \lambda_2|\boldsymbol{\beta}|_1,$$

where $|\boldsymbol{\beta}|^2$ represents the ℓ_2 norm of the coefficient vector $\boldsymbol{\beta}$, i.e.,

$$|\boldsymbol{\beta}|^2 = \sum_{i=1}^{p}\boldsymbol{\beta}_i^2$$

and $|\boldsymbol{\beta}|_1$ is equivalent to the ℓ_1 norm, i.e.,

$$|\boldsymbol{\beta}|_1 = \sum_{i=1}^{p}|\boldsymbol{\beta}_i|.$$

The elastic net estimator $\hat{\boldsymbol{\beta}}$ results from the following minimization problem,

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} l(\lambda_1, \lambda_2, \boldsymbol{\beta}).$$

Alternatively, the elastic net penalized regression may also be written as a constrained optimization problem of the least square estimator. Let $\alpha = \frac{\lambda_2}{\lambda_2 + \lambda_1}$, then for some fixed

t,

$$\hat{\boldsymbol{\beta}} = \arg\min_{\boldsymbol{\beta}} |\mathbf{y} - \mathbf{X}\boldsymbol{\beta}|^2,$$

subject to,

$$(1 - \alpha)|\boldsymbol{\beta}|_1 + \alpha|\boldsymbol{\beta}|^2 \leq t. \tag{4.1}$$

The constraint thus encompasses a convex combination of the lasso penalty and the ridge penalty. Accordingly, both ridge and lasso regression are a special cases of an elastic net, where $\alpha = 1$ results in ridge regression and $\alpha = 0$ leaves us with the ℓ_1 penalized lasso regression estimator, initially presented by Tibshirani (1996). In regard to model selection, lasso regression is particularly powerful as it shrinks some candidate variables **exactly** to zero, which is not the case when utilizing the ridge penalty. In the case of ℓ_1 penalized regression, the obtained $\hat{\boldsymbol{\beta}}$ carries only non-zero components for the variables that were deemed important enough in the cross-validation process used to determine the optimal value of the shrinkage parameter t, assuming $\alpha = 0$. This is troublesome for the frequentist statistician, as it implies that the coefficient vector $\boldsymbol{\beta}$ itself is random (Lee et al., 2016). The severity of the implications for frequentist inference is outlined in Benjamini and Yekutieli (2005) and Benjamini et al. (2009).

4.1.1. Post-Selection Inference for ℓ_1-Penalized Models

Having discovered the randomness introduced by shrinkage to exactly zero of some elements in the coefficient vector $\hat{\boldsymbol{\beta}}$, it is important to develop a sound understanding of the source of randomness. The randomness introduced by variable selection is one of choice and not inherent to the parameter value itself. Denoted in accordance with Lee et al. (2016), again we assume p explanatory variables. Further, let $M \subset \{1, ..., p\}$ be the subset of covariates which estimate β_j^M has not been set to zero, given the optimal choice of the regularization parameter t. Formally we may write,

$$\{\beta_j^M : M \subset \{1, ..., p\}, j \in M\}.$$

Given $t > 0$, we will only draw inference from the coefficients $\hat{\beta}_j^{\hat{M}}$, i.e., the estimates chosen by the applied ℓ_1 regularization process, $\{\hat{M} : \hat{\beta}_j \neq 0\}$. Thus the randomness is introduced by the choice of $\hat{M}$. Now that the problem inherent with the choice of $\hat{M}$ is established, we may consider a solution to draw valid inference as presented by Lee et al. (2016). Suppose we want to create a confidence interval $C_j^{\hat{M}}$ for a parameter $\hat{\beta}_j^{\hat{M}}$, then fixing α as the desired type I error we want the confidence interval to fulfill:

$$\mathbb{P}(\beta_j^{\hat{M}} \in C_j^{\hat{M}}) \geq 1 - \alpha.$$

Unfortunately, this is obviously undefined for all $j \notin \hat{M}$. Accordingly, to resolve this issue Lee et al. (2016) rely on conditional convergence. Since we construct an interval for $\hat{\beta}_j^{\hat{M}}$ if $\hat{M}$ is selected, we may condition on this event. Thus we obtain,

$$\mathbb{P}(\beta_j^{\hat{M}} \in C_j^{\hat{M}} | \hat{M} = M) \geq 1 - \alpha.$$

In order to obtain the interval mentioned above, Lee et al. (2016) study the conditional distribution,

$$\boldsymbol{\eta}_M^T \boldsymbol{y} | \{\hat{M} = M\},$$

which allows, in theory, to draw inference from parameters of the more general form $\boldsymbol{\eta}_M^T \boldsymbol{\mu}$. To then characterize the event, $\{\hat{M} = M\}$ for lasso regression the resulting event is a union of polyhedra, i.e., $\{\hat{M} = M, \hat{s}_M = s_M\}$, that conditions on the specified explanatory variables as well as the signs of the selected covariates. Formally this may be stated as,

$$\boldsymbol{y} \in \mathbb{R}^n : A(M, s_M)\boldsymbol{y} \leq \boldsymbol{b}(M, s_M)$$

and hence when conditioning on the sign as well as the selected variables, it is sufficient to study,

$$\boldsymbol{\eta}^T \boldsymbol{y} | \{A\boldsymbol{y} \leq \boldsymbol{b}\}.$$

Lee et al. (2016), elegantly derive that the resulting conditional distribution is a univariate truncated Gaussian. Subsequently, they use this result to present a test statistic that may be utilized to draw inference about the significance of the parameter estimates $\hat{\beta}_j^{\hat{M}}$. A description of this derivation is beyond the scope of this thesis, the interested reader is referred to Lee et al. (2016) and Taylor and Tibshirani (2016). The latter provides an extension to the results of Lee et al. (2016), to generalized ℓ_1 penalized linear models and further also provides the implementation in an R package called *selectiveInference*, by Tibshirani et al. (2019).

4.2. Ensemble Methods

Ensemble methods encompass the combination of simple learning algorithms to achieve superior results in regard to predictive power. For this thesis tree-based methods are of particular importance. For the sake of simplicity, we may only distinguish between two methods of combining individual trees, bagging and boosting the former revolves around bootstrap aggregation, which might be a more familiar term to some readers, that have not been exposed to a lot of machine learning literature. Bootstrap aggregation aims to reduce the variance of prediction, by sampling with replacement from all observations, to then apply some simple learning algorithm like a regression tree. In the end, the resulting prediction is a constant derived from the individual learners, most of the times said constant is the arithmetic mean. In contrast, boosting aims to improve the predictive power of simple learners like regression trees by sequentially improving the error of the

previously fit tree. This chapter gives a concise overview of regression trees as the base learners, random forest which utilizes bagging and eXtreme Gradient Boosting which as the name suggests is a special boosting technique discovered by Chen and Guestrin (2016). The formalization of those methods is denoted in accordance with Hastie et al. (2009) for regression trees and random forests. For eXtreme gradient boosting the notation is heavily inspired by the seminal paper by Chen and Guestrin (2016). As previously, it is assumed that the reader is familiar with basic terminology, should some terms be unfamiliar, the reader is referred to James et al. (2013) and Hastie et al. (2009). The former provides a very informal and easy to understand approach, while the latter is more formally motivated.

4.2.1. Regression Trees

The description "simple learner" could not be more fitting than in the case of a regression tree. In essence, the idea may be described in a single sentence. A regression tree finds recursive binary partitions in the space spanned by the explanatory variables that optimizes some metric used to judge the best variable as well as the best available value to form a partition. More formally, again let p represent the number of explanatory variables and n the number of observations, i.e., $i = 1, 2, ..., n$ and $x_i = (x_{i1}, ..., x_{ip})$. Further, let there be M regions R_m, each formed by a split in the data. Additionally, suppose that we model the response variable y as a constant c_m in each region, i.e.,

$$f(x) = \sum_{m=1}^{M} c_m \mathbb{1}_{\{x \in R_m\}}.$$

If we choose to minimize the sum of squares,

$$\sum_{i=1}^{n} (y_i - f(x_i))^2,$$

the optimal $\hat{c}_m$ is the arithmetic mean[2] of the observations in partition R_m (Hastie et al., 2009). A proof for this statement would be identical to deriving the optimal value of the intercept in a linear model fit via ordinary least squares (OLS), that is only made up of one constant. So far, we have determined how to optimally represent the observations grouped in a partition R_i by a single constant. However, we still need to determine how to choose the optimal splitting variable x_j as well as the optimal splitting point s. Suppose that at each node, we consider all possible variables p that may generate the best pair of resulting half planes,

$$R_1(j, s) = \{X | X_j \leq s\} \text{ and } R_2(j, s) = \{X | X_j > s\}.$$

---

[2]A proof is provided in Appendix A.2.

We aim to choose the splitting variable j and the corresponding splitting point s such that,

$$\min_{j,s} \Big[\min_{c_1} \sum_{x_i \in R_1(j,s)} (y_i - c_1)^2 + \min_{c_2} \sum_{x_i \in R_2(j,s)} (y_i - c_2)^2 \Big].$$

Note that it is infeasible to choose those sequential partitions that optimize the sum of squares simultaneously for all possible splits. Accordingly, the optimization process is greedy in a sense that it is memoryless and only considers the current available splits, given all splits that have occurred beforehand. To a degree this is quite unsettling as it does not guarantee that the recursive partitioning will lead to a global minimum in RSS. This is also where ensemble algorithms may make quite a difference, as we will discover at a later stage. Coming back to the greedy minimization, we have previously discovered that the inner minimum in regards to the constant that represent the observations in R_1 and R_2 are optimally represented by the arithmetic mean. The outer minimum in respect to j and s may be discovered by finding the best splitting point s for every candidate variable j to then assess which combination (j, s) leads to the minimization in RSS. This process may then be repeated in the resulting half-planes. If we were to consider trees as a stand alone learner, at this point, we would have to think about how we choose the complexity of the tree. However, as in this thesis the depth of a tree is a hyper parameter chosen via cross validation, we may skip this part. The interested reader is referred to Hastie et al. (2009). Shortly summarized, fixing some threshold for the minimum decrease in RSS to warrant an additional split is not optimal, as a seemingly bad split at the current node might lead to a significant decrease in RSS at a later split. Accordingly, what is usually applied to determine the optimal tree complexity is cost-complexity pruning. In essence this means the tree is grown until some minimum node size is reached and then the tree is "cut down" backwards starting from the terminal nodes.

4.2.2. Random Forest

As previously mentioned, random forests utilize bootstrap aggregation to improve the predictive performance of many noisy but unbiased learners. This is achieved by reducing the variance of the individual simple learning algorithm. The presented regression trees are well suited for this, as they may capture complex non-linear relationships in the data and given that they are grown sufficiently deep, they exhibit low bias. For the sake of simplicity, the notation is slightly changed. Let the dataset be defined as $\mathcal{D}\{\{\boldsymbol{x}_i, y_i\}\}$, with $i = 1, ..., n$, $\boldsymbol{x}_i \in \mathbb{R}^p$ and $y_i \in \mathbb{R}$. The algorithm used to fit a random forest may be concisely described following Hastie et al. (2009):

Algorithm 1: *Random Forest*

Data: $b = 1, 2, ..., B$ bootstrap samples drawn with replacement from the training data, p the number of explanatory variables, m the number of variables sampled for each split, n_{min} the minmum node size and the maximum depth d_{max}.

1. For each bootstrap sample b, grow a regression tree T_b utilizing the bagged data, by recursively repeating the following steps, until the minimum node size n_{min} or the maximum depth d_{max} is reached.

   i. Randomly sample m explanatory variables from the p available candidates.

   ii. Find the best split point s for each of the sampled variables m.

   iii. Choose the best split available.

2. Return the ensemble of trees $\{T_b\}_1^B$

To predict the response of a new observation, we simply average over the trees in our forest, i.e,

$$\hat{f}_{rf}(\boldsymbol{x}_i) = \frac{1}{B}\sum_{b=1}^{B}T_b(\boldsymbol{x}_i).$$

By linearity of the expectation operator, given that each tree generated in bootstrap aggregation is identically distributed, we find that the expectation of the average of an ensemble of trees is identical to any of the individual trees[3]. Put differently, the bias of any single tree is identical to the bias of the bagged ensemble and thus the only way of improving the predictive accuracy is via a reduction in variance. To gain further insights into the workings of a random forest we may benefit from examining the variance of the B random variables T_b that represent the trees in our forest. Note that said trees are identically distributed but not necessarily independent. Let the variance of T_b be represented by σ^2 and further let the positive pairwise correlation be represented by ρ. Accordingly, the prediction that we obtain by averaging over all trees in the forest exhibits the variance,

$$\mathbb{V}\text{ar}(\hat{f}_{rf}) = \mathbb{V}\text{ar}\left[\frac{1}{B}\sum_{b=1}^{B}T_b(\boldsymbol{x}_i)\right] = \frac{1}{B^2}\mathbb{V}\text{ar}\left[\sum_{b=1}^{B}T_b(\boldsymbol{x}_i)\right] \stackrel{4}{=} \frac{1}{B^2}\sum_{k=1}^{B}\sum_{j=1}^{B}\mathbb{C}\text{ov}\left[T_k, T_j\right].$$

Now, given that in the double sum there are B^2 summands and we know that B of those are just $\mathbb{C}\text{ov}\left[T_k, T_k\right] = \mathbb{V}ar\left[T_k\right] = \sigma^2$, we are left with $B^2 - B$ terms for all cases

---

[3]A proof is provided in Appendix A.3.

[4]A proof is provided in Appendix A.4.

$k \neq j$, and we thus find,

$$\frac{1}{B^2}\sum_{k=1}^{B}\sum_{j=1}^{B}\mathbb{C}\text{ov}\left[T_k, T_j\right] = \frac{1}{B^2}\left[B\sigma^2 + (B^2 - B)\rho\sigma^2\right] = \frac{\sigma^2}{B} + \frac{B-1}{B}\rho\sigma^2,$$

knowing that for the identically distributed variables at hand,

$$\rho = \text{Corr}\left[T_k, T_j\right] = \frac{\mathbb{C}\text{ov}\left[T_k, T_j\right]}{\sigma_{T_k}\sigma_{T_j}}$$
$$\iff \mathbb{C}\text{ov}\left[T_k, T_j\right] = \rho\sigma^2.$$

Some trivial rearrangements then lead us to the more easily interpretable expression,

$$\mathbb{V}\text{ar}(\hat{f}_{rf}) = \rho\sigma^2 + \frac{\sigma^2}{B}(1 - \rho).$$

This result simply demonstrates the idea behind the variance reduction achieved via bagging and candidate variable sampling. We observe that the second term approaches zero as the number of trees in the forest approach infinity. Additionally, we also observe that the first term depends on the correlation of the individual trees, thus by sampling different candidate variables at each split for every tree, we aim to reduce the correlation of the individual trees in our forest. This is also of particular importance for the weights that we use to average over our trees. It can be shown that applying an equal weight to the prediction of our simple learners is optimal if the trees are uncorrelated[5].

Given this discovery, imagine a dataset which is made up of very few variables that have a relationship with the dependant variable in the data generating process. Further, assume that there are numerous noise variables that have no predictive power. Then in order to retain some of the variables that possess predictive power in the cross-validation process the optimal number of candidate variables to sample for each split m will tend towards the number of total predictors in the dataset. This happens because low values of m will result in candidate samples that only contain noise variables. A key takeaway is thus, if we observe that the optimal value for m chosen via cross validation is very close to the total number of explanatory variables p, the decorrelation via candidate sampling aiming to reduce $\rho\sigma^2$ might be failing. This is why in Section 4.3.2 a variable selection method is presented that aims to reduce the number of noise variables via recursive feature elimination. Additionally, instead of variable selection methods, reducing the number of variables before fitting the random forest with dimensionality reduction methods is also a viable approach. This is outlined in Section 4.3.1.

---

[5]A proof is provided in Appendix A.5.

4.2.3. eXtreme Gradient Boosting

Instead of averaging over a multitude of trees, the ensemble generated by eXtreme Gradient Boosting (XGBoost) is created by starting with some constant as our initial prediction. Using this constant, residuals for every observation are generated, then trees are sequentially fit to improve on the error of the previous prediction. The formalization is denoted in accordance with the paper that first introduced XGBoost, Chen and Guestrin (2016).

Let the dataset be defined as $\mathcal{D}\{\{\boldsymbol{x}_i, y_i\}\}$, with $i = 1, ..., n$, $x_i \in \mathbb{R}^p$ and $y_i \in \mathbb{R}$. Further, let $\eta \in (0, 1]$, which is sometimes referred to as the learning rate. The ensemble of trees is then defined additively as:

$$\hat{y}_i = \sum_{k=1}^{K} \eta f_k(\boldsymbol{x}_i).$$

f_k represents a tree structure that, somewhat similarly as previously described in Section 4.2.1, recursively partitions the space spanned by the explanatory variables to then represent the observations grouped in a leaf node as some constant w_j. The learning process is governed by the following regularized objective,

$$\mathcal{L}^t = \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)} + f_t(\boldsymbol{x}_i)) + \gamma T + \frac{1}{2}\lambda f_t(\boldsymbol{x}_i)^2.$$

As already mentioned, the initial prediction for $t = 0$ is some constant, which is identical for all observations. Starting from the first iteration $t = 1$, trees are built via optimization of the above objective. Where l represents the loss function, usually of quadratic nature, e.g., $l(y_i, \hat{y}_i) = \frac{1}{2}(y_i - \hat{y}_i)^2$. Further, γT, where $\gamma \in \mathbb{R}^+$, is used to penalize the complexity of the model where T represents the number of terminal nodes (leafs). The last term , $\frac{1}{2}\lambda|w|^2$ offers additional regularization to avoid over-fitting. It utilizes the ℓ_2 norm, similar to the ridge penalty described in Section 4.1. Note that we assume the result of the previous iteration to be fixed and thus this is also a greedy optimization procedure. For the sake of computational complexity the presented objective is not minimized directly, XGboost utilizes a second order Taylor polynomial, as an approximation for the loss function, i.e.,

$$\mathcal{L}^t \approx \sum_{i=1}^{n} l(y_i, \hat{y}_i^{(t-1)}) + g_i f_t(\boldsymbol{x}_i) + \frac{1}{2}h_i f_t(\boldsymbol{x}_i)^2 + \gamma T + \frac{1}{2}\lambda f_t(\boldsymbol{x}_i)^2.$$

Where

$$g_i = \frac{dl(y_i, \hat{y}_i^{(t-1)})}{d\hat{y}_i^{(t-1)}} \text{ and } h_i = \frac{d^2 l(y_i, \hat{y}_i^{(t-1)})}{d\left[\hat{y}_i^{(t-1)}\right]^2},$$

represent first and second order derivatives respectively. To reduce the notational burden let I_j be the instance set of node j, we may then remove the terms that do not have an

influence on the output weight w_j to obtain,

$$\mathcal{L}^t \approx \sum_{i \in I_j} g_i f_t(\boldsymbol{x}_i) + \frac{1}{2} h_i f_t(\boldsymbol{x}_i)^2 + \frac{1}{2} \lambda f_t(\boldsymbol{x}_i)^2. \tag{4.2}$$

Now we find the optimum output weight w_j for the leaf j, by finding the derivative with respect to the output weight,

$$\frac{d\mathcal{L}^t}{dw_j} = \sum_{i \in I_j} g_i + h_i w_j + \lambda w_j \overset{!}{=} 0$$

$$\iff w_j^* = -\frac{\sum_{i \in I_j} g_i}{\sum_{i \in I_j} h_i + \lambda}$$

Considering the previously mentioned loss function, $l(y_i, \hat{y}_i) = \frac{1}{2}(y_i - \hat{y}_i)^2$, we find g_i and h_i to be:

$$g_i = -(y_i - \hat{y}_i) \text{ and } h_i = 1.$$

Thus for the quadratic loss function at hand, we find the optimal leaf weight to be:

$$w_j^* = \frac{\sum_{i \in I_j}(y_i - \hat{y}_i)}{|I_j| + \lambda} = \frac{\text{RS}_{i \in I_j}}{|I_j| + \lambda},$$

where $|I_j|$ represents the cardinality and $\text{RS}_{i \in I_j}$ the sum of the residuals of the instance set I_j. If we plug the optimal output weight back into Equation 4.1, some algebraic manipulation then yields the value that we may use to evaluate the similarity score of leaf j.

$$\mathcal{L}_{opt} = \frac{1}{2} \frac{[\sum_{i \in I_j}(y_i - \hat{y}_i)]^2}{|I_j| + \lambda} \approx \frac{[RS_{i \in I_j}]^2}{|I_j| + \lambda},$$

given that the coefficient is constant for all evaluated splits, it may be omitted. Additionally, in the original manuscript the minimization of the regularized objective is changed into a maximization by multiplying by (-1), the reason for that is to achieve a more easily interpretable concept of the similarity score. Now that we have a tool for calculating the optimal output weight w_j as well as a procedure to evaluate the quality of a split, we can formulate XGBoost as an algorithm. The following formulation omits some of the tunable parameters that are available in the implementation by Chen et al. (2022). The additional hyperparameters are:

- *min_child_weight*: The minimum sum of instances in a child node resulting from a split. This is an alternative stopping variable that can be used together or instead of the maximum depth outlined in Algorithm 2.

- *subsample*: A ratio passed to the algorithm that determines the relative amount of data points sampled for each round n_{iter}, it is used to further prevent over-fitting.

- *colsample_bytree*: The subsample ratio of candidate variables p when constructing each tree.

- α: An additional ℓ_1 regularization term, similar to the one used in lasso regression outlined in Section 4.1.

Algorithm 2: *XGboost*

Data: A training dataset $\mathcal{D}\{\{\boldsymbol{x}_i, y_i\}\}$, with $i = 1, ..., n$, $x_i \in \mathbb{R}^p$ and $y_i \in \mathbb{R}$. The structure complexity paremeter γ, the regularization parameter λ, the learning rate η, the maximum depth d_{max} and the number of trees to be fit in succession n_{iter}.

1. Start with some initial prediction that is identical for all observations.

2. For n_{iter} times, repeat:

   I. While the number of nodes in the current tree is smaller or equal to d_{max}:

      i. Compute the residuals using the predicted values of the previous iteration, and then calculate the current similarity score.

      ii. Find the best splitting point w_j^* for all candidate variables p, execute the best split, by choosing the the variable with highest gain at the optimal splitting point. Gain is defined as the sum of the similarity scores of the child nodes minus the similarity score of the root node.

   II. Prune the tree of the current iteration backwards by subtracting γ from the gain of the respective split in the tree. If the result of the difference is negative, the split is removed.

   III. Multiply the predicted values of the created tree structure by η and add it to the existing ensemble.

3. Return the ensemble, $\sum_{n=1}^{n} \eta f_n(\boldsymbol{x}_i)$.

4.3. Nested Cross Validation

The idea of simple k-fold cross validation for hyperparameter optimization is to split the training data into K equally-sized parts. Suppose the indexing function $\kappa : \{1, ..., n\} \rightarrow \{1, ..., K\}$, it indicates which observation i out of n is allocated to the $K-1$ training folds and the K'th validation fold, respectively. Further, let $\hat{f}^{-k}(x)$, be the fitted predictive

function computed with the k'th part of the data removed and let $\boldsymbol{\alpha}$ represent a vector of hyper-parameters used to fit $\hat{f}^{-k}(x, \boldsymbol{\alpha})$ (Hastie et al., 2009). We can then represent the cross validation error as

$$CV(\hat{f}) = \frac{1}{n}\sum_{i=1}^{n} l(y_i, \hat{f}^{-\kappa(i)}(x, \boldsymbol{\alpha})),$$

where l represents some loss function, e.g., of quadratic or linear nature. Now given that some preprocessing schedules like Logistic PCA or feature selection have an effect on the choice of the optimal hyper-parameter vector $\boldsymbol{\alpha}$, within the K folds a secondary CV process is established. Consider PCA as an example, the optimal number of principle components is a parameter to be chosen via cross validation, further we cannot apply our predictive function $\hat{f}^{-k}(x)$ on the validation fold without previously using the training data to determine the loading vectors used to apply the data set transformation inherent to PCA. Accordingly, the idea of nested cross validation is to assess the optimal value of principle components in an inner loop, to then use the acquired loading vectors to apply the required data transformations. Subsequently, one may then then run the outer loop over the hyper-parameter set $\boldsymbol{\alpha}$. Note, that it considered a severe data leak to just apply PCA to the validation set directly. Accordingly, this section briefly describes two preprocessing schedules that require nested cross validation, logistics PCA applied to binary variables by group and recursive feature elimination. Both methods aim to reduce noise in the data set via dimensionality reduction.

4.3.1. Logistic PCA

The idea of principle component analysis (PCA) was first proposed by Pearson (1901). The first introduced data processing technique aims to generate a projection of rank k, that is as close to the original data in terms of the mean squared error as possible. Let the data be made up of $\boldsymbol{x}_i \in \mathbb{R}^d$, for $i = 1, ..., n$. To then project the initially d-dimensional data into $k < d$ dimensions, Pearson suggested representing each $\boldsymbol{x}$ by $\boldsymbol{\mu} + \boldsymbol{U}\boldsymbol{U}^T(\boldsymbol{x} - \boldsymbol{\mu})$, where $\mu \in \mathbb{R}^d$ and $\boldsymbol{U}$ is a $d \times k$ matrix with orthonormal columns, i.e., $\boldsymbol{U}^T\boldsymbol{U} = \boldsymbol{I_k}$. The objective is thus,

$$\min_{\boldsymbol{\mu}\in\mathbb{R}^d,\ \boldsymbol{U}^T\boldsymbol{U}=\boldsymbol{I_k}} \sum_{i=1}^{n}|(\boldsymbol{x_i} - \boldsymbol{\mu}) - \boldsymbol{U}\boldsymbol{U}^T(\boldsymbol{x_i} - \boldsymbol{\mu})|^2$$

Pearson (1901), showed that $\boldsymbol{\mu}$ equal to the sample mean and $\boldsymbol{U}$ as a matrix with the first k eigenvectors of the sample covariance matrix, yields the minimum of the stated objective. Recently, Landgraf and Lee (2020) proposed an extension to the idea of principle component analysis that aims to perform dimensionality reduction specifically for binary data. They also implemented their work in an R package called *logisticPCA*, Landgraf and Lee (2015). The idea is based on an alternative interpretation of PCA

as a technique for dimensional projection of the natural parameters of the saturated model. Assume $\boldsymbol{x}_i \sim \mathcal{N}(\boldsymbol{\theta}_i, I_d)$, then the original idea of PCA may be interpreted as a projection to minimize the Gaussian deviance which is identical to the minimization of the mean square error (Collins et al., 2001). Where $\mathbb{E}(\boldsymbol{x}_i)$ can be thought of as the natural parameter, while $\boldsymbol{x}_i$ is the natural parameter of the saturated model. Natural parameter refers to a definition that is derived from the representation as part of the exponential family, i.e.,

$$f(x|\theta) = h(x)\exp(T(x)\theta - A(\theta)),$$

where θ is the so called natural parameter, $T(x)$ is also referred to as the sufficient statistic, $A(\theta)$ is the log normalizer and $h(x)$ is often referred to as the base measurement. An introduction to exponential families can be found in Casella and Berger (2001). Given that the Bernoulli(p_{ij}) distribution is also part of the exponential family, this alternative interpretation of minimization of the deviance may be applied in a similar fashion. The natural parameter of the Bernoulli distribution is $\theta_{ij} = \text{logit}(p_{ij})$, then the saturated model occurs when $p_{ij} = x_{ij}$, i.e., the natural parameter in the saturated model is:

$$\tilde{\theta}_{ij} = \begin{cases} -\infty & \text{if } x_{ij} = 0 \\ \infty & \text{if } x_{ij} = 1. \end{cases} \tag{4.3}$$

Here Landgraf and Lee (2020) mention that instead of ∞ they use a large number m to achieve computational feasibility. The natural parameters of the Bernoulli model are then estimated via

$$\hat{\boldsymbol{\theta}}_i = \boldsymbol{\mu} - \boldsymbol{U}\boldsymbol{U}^T(\tilde{\boldsymbol{\theta}}_i - \boldsymbol{\mu}),$$

where $\tilde{\boldsymbol{\theta}}_i$ represents the d-dimensional vector of natural parameters from the saturated model. Finally, $\boldsymbol{\mu}$ and $\boldsymbol{U}$ are found that minimize the Bernoulli deviance,

$$\sum_{i=1}^{n}\sum_{j=1}^{d} - 2x_{ij}\hat{\theta}_{ij} + 2\log(1 + \exp(\hat{\theta}_{ij})).$$

Now that we have established the formalities of this approach, an algorithm of the nested CV process is provided below. The logPCA preprocessing is only implemented for the random forest for reasons outlined in Section 4.2.2.

Algorithm 3: *Nested CV: Logistic PCA*

Data: A training dataset $\mathcal{D}\{\{\boldsymbol{x}_i, y_i\}\}$, with $i = 1, ..., n$, $x_i \in \mathbb{R}^p$ and $y_i \in \mathbb{R}$. A $n \times k$ tuning grid for the random forest hyperparameters, a $l \times p$ tuning grid for the logistic PCA hyperparameters, K cross validation folds and a loss function $l(y_i, \hat{y}_i)$ to determin the CV error.

1. For all K cross validation folds, repeat:

   a) For all l rows in the logPCA tuning grid, repeat:

      I. Perform CV for the logPCA parameter m, using the number of principle components of the current row in the logPCA tuning grid.

      II. Perform the logPCA preprosessing procedure on the training folds.

      III. Apply the transformation that minimizes the Bernoulli deviance of the training data to the validation data.

      IV. For all n rows in the RF tuning grid, repeat:

         i. Fit the random forest with the hyperparameters specified in the current row of the RF tuning grid, utilizing the logPCA transformed data obtained from the training folds.

         ii. Predict the response in the validation set.

         iii. Calculate the error using the loss function $l(y_i, \hat{y}_i)$.

         iv. Save the error as well as the input hyperparameters.

2. Compute the CV error by averaging the error of each model specification across all folds.

Note that it is vital that the logPCA process is not directly applied to the validation data in the CV process. This would be considered a severe data leak. Instead the logPCA process is executed on the training data and the resulting transformation is then applied to the test data. This is also where the implementation by Landgraf and Lee (2015) shines, as the application of an obtained logPCA transformation to a new dataset is implemented in a computationally efficient way. Further, note that in this thesis the actual implementation of this CV process is slightly more complicated and would require another loop. This is because the logPCA process was not applied to all columns in the data set at once. The columns were grouped into contract description, firm identification and firm interaction columns, which represent all sources of binary dummy variables besides the contract date and the county. Logistic PCA was then

applied to these groups of columns separately. This was done to be able to assess variable importance with the source of the input information sustained. Meaning that we may not be able to say which variable in the original form of the data is more influential for the prediction of our model. However, we may tell if a potentially influential principle component was generated by contract description data, firm identification information or firm interaction data.

4.3.2. Recursive Feature Elimination

As outlined in Section 4.2.2, when a dataset contains many noise variables, the predictive accuracy of a random forest model might benefit from the removal of some of the variables with little or no predictive power. The idea behind recursive feature elimination is to fit a random forest, identify a share of the $p \in (0, 1)$ most important variables and remove the remaining covariates. This may be repeated for a number of times k, where the optimal choice of k and p is to be determined via cross validation. Gregorutti et al. (2013) present simulation studies that convey that the recursive feature elimination algorithm performs well in reducing the number of unimportant noise variables. They also compare different approaches to asses the importance of a predictor and conclude that permutation importance delivers the best results. Permutation importance may be efficiently computed by using the out of bag (OOB) sample. When the b-th tree is constructed, OOB samples are predicted using the current tree structure, then the values of one of the covariates is randomly permuted and the OOB samples are predicted again. The decrease in accuracy measured by some loss function $l(y_i, \hat{y}_i)$ is then averaged over all trees in the forest. The resulting measure may then be thought of as the importance of the variable. This procedure is repeated for all variables. The R package *ranger* by Wright and Ziegler (2017) enables the user to compute the variable importance as outlined above. The algorithm of recursive feature elimination may be summarized as follows:

Algorithm 4: *Nested CV: Recursive Feature Elimination*

Data: A training dataset $\mathcal{D}\{\{\boldsymbol{x}_i, y_i\}\}$, with $i = 1, ..., n$, $x_i \in \mathbb{R}^d$ and $y_i \in \mathbb{R}$. A $n \times f$ tuning grid for the random forest hyperparameters, in this case this tuning grid also contains the parameters for recursive feature elimination p and k, K cross validation folds and a loss function $l(y_i, \hat{y}_i)$ to determine the CV error.

1. For all K cross validation folds, repeat:

   a) For all n rows in the RF tuning grid, repeat:

      I. Fit a random forest using the data in the training folds.

      II. Calculate the permutation importance of each covariate.

      III. While the number of iterations is smaller or equal to k, specified in the current row of the tuning grid:

         i. Retain the $\lceil d * p \rceil$ most important variables, in the training and in the validation set.

         ii. Fit the random forest using the remaining variables using the training data.

         iii. Calculate the feature importance.

      IV. Predict the response in the validation set.

      V. Calculate the error using the loss function $l(y_i, \hat{y}_i)$.

      VI. Save the error as well as the input hyperparameters.

2. Compute the CV error by averaging the error of each model specification across all folds.

Note that the model that includes all covariates may be included as the model that is fit if $k = 0$ or $p = 1$. Albeit $p = 1$ is possible, in terms of computational efficiency it is the inferior solution to retaining all variables. Accordingly, if the tuning grid is set up correctly the standard random forest will be fit in the recursive feature elimination CV process.

5. Results

This section covers the comparison of the predictive accuracy based on the out of sample root mean square error and mean absolute error. For each model type the optimal hyper parameters are displayed. Additionally, the results of the post selection inference are presented.

5.1. Performance Evaluation Metrics

To asses the error of our predictions, two metrics are used. Let m be the number of observations in the test dataset. Then we may calculate the mean error as:

$$\text{Err} = \frac{1}{m}\sum_{i=1}^{m}l(y_i, \hat{y}_i),$$

where $l(y_i, \hat{y}_i) = (y_i - \hat{y}_i)^2$, or $l(y_i, \hat{y}_i) = |y_i - \hat{y}_i|$. The plot below displays the loss associated with the corresponding prediction error. When using a quadratic loss function, the plot enables us to realize that we interpret large deviations from the actual values as drastically worse when compared to the linear loss function. Depending on the nature of our problem, putting great emphasis on large errors might be desirable. Given that for this thesis the main objective is to ease budget planing of the auctioning entity, the interpretation of using a quadratic loss function is that we want to avoid to allocate drastically more or less capital prior to the bidding process.
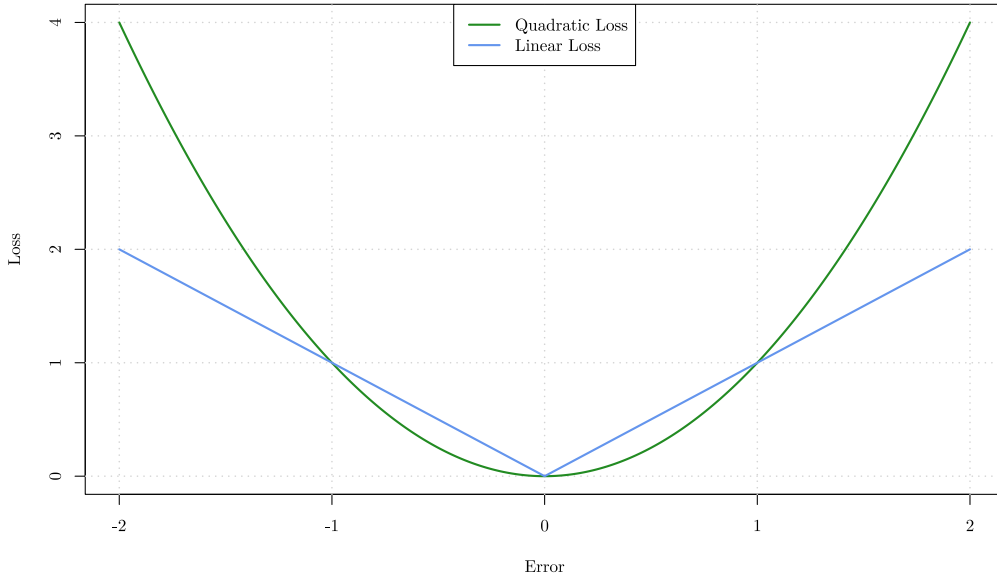

Figure 7: Loss Functions

5.2. Prediction

For all models the CV for hyperparameter optimization was written in R (R Core Team, 2022). At first, a hyperparameter grid is built that spans a wide range for all input values. Then the CV process is repeated multiple times, in each iteration the ranges for the different parameters are altered such that for each case, the optimal parameter is never the minimum or maximum of the input set. The rationale behind this is, that a hyperparameter value that represents the minimum or maximum might not be optimal. Suppose we have a tuning grid for an elastic net, i.e., we are looking for the values of α and λ that minimize our CV error. Further suppose, that we choose the sets $\alpha \in \{0, 0.25, 0.5, 0.75, 1\}$ and $\lambda \in [0, 100] \cap \mathbb{N}_0$. We then set up the tuning grid, that contains all combinations of our parameter values. Formally, this may be expressed using set theory. The tuning grid is just the Cartesian product of all hyperparameter sets, i.e., assume the hyperparameter sets for α and λ are called A and B respectively. Then the set that represents the tuning grid is just $A \times B$. Now assume that the lowest CV error is achieved for $\alpha = 0.25$ and $\lambda = 100$, then we observe that the λ value is the maximum in B. Accordingly, it could be the case that a higher value for λ leads to a lower CV error. For the next CV run the tuning grid may be altered shrinking the range of α values while extending the λ set.

All cross validation runs were conducted in Google Colaboratory, which allows users to remotely execute Python code. The PRO+ version of their service offers 24h of uninterrupted code execution. Thus the size of the tuning grids is chosen such that execution finishes within 24 hours. To execute the R code in Python the package *rpy2* was utilized (Gautier, 2022).

5.2.1. Elastic Net

As previously described the elastic net offers two tunable parameters. To find the optimal specification, the package *glmnetUtlis*, by Ooi (2021) was used. This is an R package that extends the original package *glmnet* and enables the user to tune both t and α from Equation 4.1 (Friedman et al., 2010). Unfortunately, the definition of the regularized objective is defined differently in the package. However, this difference is just of notational nature. The optimal five fold CV root mean square error (RMSE) is achieved for $\alpha = 1$ and $t = 48.7$, where t from Equation 4.1 is called λ in *glmnetUtils*. Given the different setup, $\alpha = 1$ implies the optimal elastic net is actually a lasso regression model. This also stems from a difference in the definition in the regularized objective of the package.

5.2.2. Random Forest

The random forest is fit incorporating two different preprocessing schedules as previously described. The cross validation process is implemented parallelly using the R package *parallel* and *ranger* was used to fit the forests (R Core Team, 2022; Wright & Ziegler, 2017). The following table contains the optimal hyperparameters that were found via five fold cross validation.

	Recursive Feature Elimination	logistic PCA
Node Feature Share	85%	-
Mtry	-	55
Min. Node Size	3	1
Max. Depth	20	30
Num. Trees	1500	1000
Nrounds	16	-
Covariate Share	75%	-
Num. PC Description	-	20
Num. PC Vendors	-	10
Num. PC Vendor Interaction	-	20

Table 1: Random Forest Best CV Hyperparameters

Node Feature Share is a wrapper variable for the number of covariates to sample at each node to find another split in the data. Given that the number of explanatory variables is not constant in the recursive feature elimination algorithm, the number of sampled covariates at each new split is passed via a percentage of total variables left in the current iteration. "Mtry" which is an argument of the random forest function in the *ranger* package represents the absolute number of covariates to sample for a new split. "Nrounds", represents the number of feature elimination rounds, while "Covariate Share" represents the relative number of covariates to retain in each iteration. Accordingly, knowing that we start the first iteration with 1086 explanatory variables and we iteratively remove 25% in each of the 16 iterations, we find that the forest with the lowest CV error only utilizes roughly $\lceil 0.75^{16} \times 1086 \rceil = 11$ covariates. The actual number might be slightly higher since the algorithm retains the ceiling of the share times the current number of remaining covariates in each iteration. The optimal number of principle components for the contract description, the vendor ID and the vendor interaction was discovered using the nested CV process outlined in Section 4.3. The best model reduces the 169 bidder identities to 10 principle components (PCs), the bidder interactions are reduced to 20 PCs from 652 initial pairwise interaction terms. Lastly, the 258 description words are represented by 20 PCs. All remaining variables in the tuning grid are thoroughly explained in Section 4.2.2.

5.2.3. eXtreme Gradient Boosting

The XGBoost CV process is implemented using the R package *xgboost*, that was released with the original manuscript (Chen et al., 2022). All of the listed hyperparameters are explained in Section 4.2.3.

	XGBoost
N_{iter}	338
Min. Child Weight	1
Max. Depth	7
Subsample	90%
Colsample by Tree	90%
η	0.04
γ	0.01
λ	8
α	3

Table 2: XGBoost Best CV Hyperparameters

5.2.4. Performance Comparison

The following plots aim to assess the out of sample performance of the respective model. Out of the total 430 observations 80% were used as training data, while 20% were held back for performance evaluation. In the following plots we observe the actual versus predicted award price in thousands of U.S. dollars. Additionally, Table 3 displays the RMSE and MAE values for all models. In Figure 8 we can observe that the predicted values provided by the lasso regression are quite close to the actual values for all of the observations in the test set. Further, we observe that for many observations the lasso regression delivers a better prediction when compared to the benchmark of the engineer's estimate. Unfortunately, all other models did not lead to an improvement when compared to the engineer's estimate. Now, one might be tempted to assume that the relationship of the most important predictors in the dataset with the award price might be linear. The reasoning for this assumption would be, that the recursive partitioning associated with tree based models may only approximate a linear relationship. However, even though this hypothesis is not testable, it might be the case that the tree based methods suffer more severely from the small number of observations. Further, especially XGBoost is known to perform poorly in very sparse datasets with few observations in the training data set. Accordingly, the framing of our performance analysis has to include the fact that with a growing number of observations for model training, it could very well be the case that the more complex models would outperform the lasso regression.
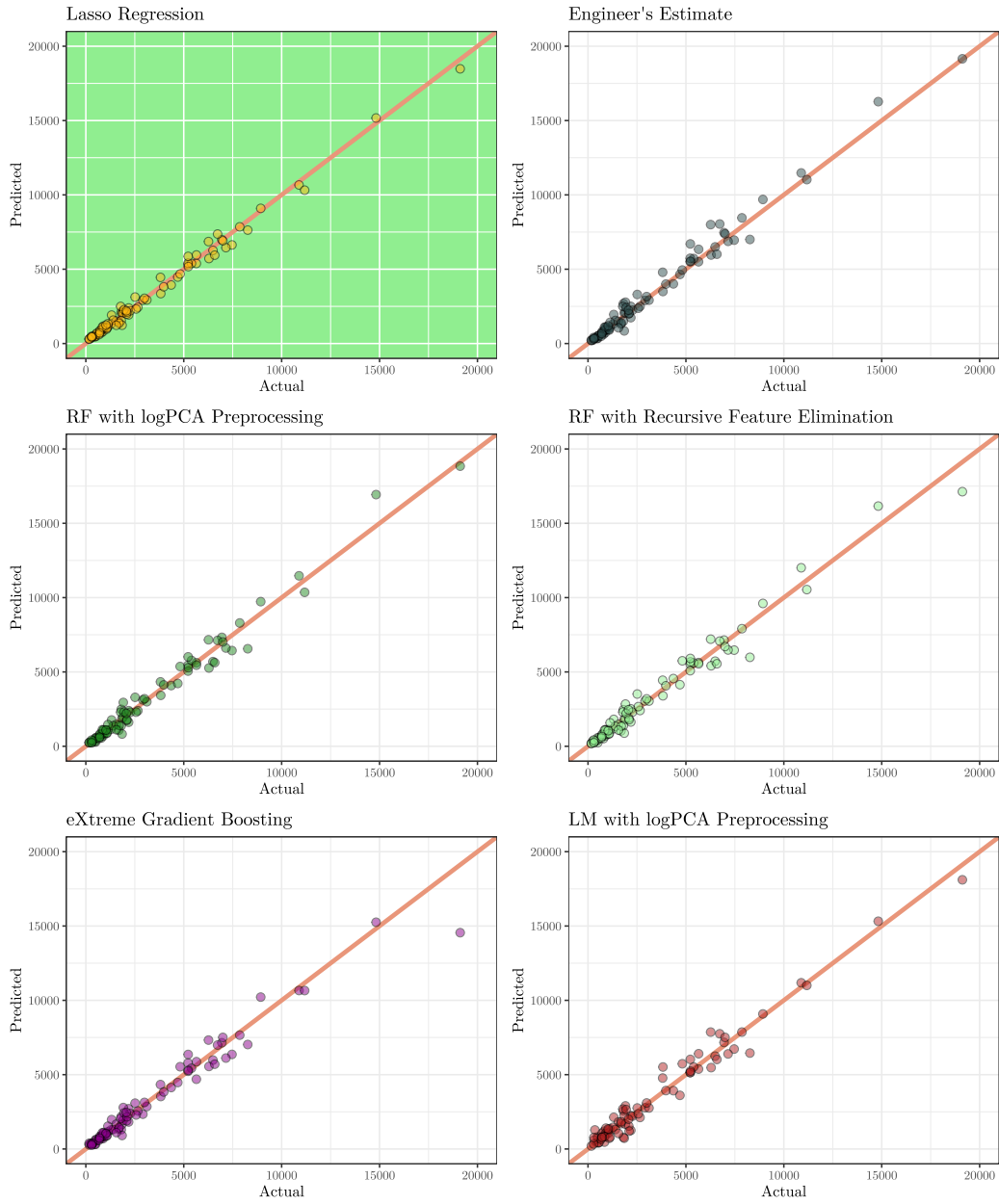
Figure 8: Actual vs. Predicted Values

	Lasso Reg.	Eng. Est.	logPCA_RF	rfe_RF	logPCA_LM	XGB
RMSE	326.1261	497.0567	509.7934	560.7600	609.4673	671.6634
MAE	241.7894	327.9388	348.2869	373.1132	448.2662	376.3191

Table 3: Out of Sample Error

5.3. Unsupervised Collusion Detection

Besides a good ex ante estimate of the required budget for a contract auctioned by the CDOT, it is of high relevance whether the interaction of certain bidders systematically leads to higher award prices. If this would be the case, the state of Colorado might want to initiate further investigations to discover potential bid rigging schemes. Accordingly, this section provides the post selection inference results for the lasso regression, which has proven itself as the best predictor. The required inference for ℓ_1-penalized regression models is outlined in Section 4.1.1. The confidence intervals were obtained using the previously mentioned R package *selectiveInference* (Tibshirani et al., 2019). To keep the type II error of our inferential endeavours as low as possible, the cross-validation process for t as described in Section 4.3 is repeated for the entire dataset. The resulting optimal value is $t = 69.8$. Utilizing this hyperparameter value, the lasso regression shrinks 1061 out of 1078 variables exactly to zero. The model is made up of the following variables:

- Engineer's estimate
- Contract time
- Two stemmed description words: toll and street
- One county: Prowers
- One bidder identifier
- 11 bidder interaction terms

Fixing the type I error to 5%, we find that the only confidence interval that does not contain zero is the one for the engineer's estimate. Recalling our initial assumptions from the theoretical discussion in Section 3.1, we expected the number of firms to play a tangible role in determining the award price of an auction, which does not seem to be the case. The fact that only the engineer's estimate's confidence interval does not contain zero may hint at the fact that we might be suffering to a high type II error rate. Accordingly, from the perspective of the CDOT, it is advisable to repeat the process using more auctions as input for the model. The increase in observations might allow the type II error to decrease to a point were significant bidder interactions might be detected. Given that 11 out of 17 variables that were not shrinked to 0 were bidder interactions, this does not seem particularly unlikely.

6. Conclusion

This thesis offers a concise description of the process that is applied to obtain the data from the website of the CDOT. Further, the auction process of the CDOT is explained and an economic operationalization of bidders' behaviour is provided. After establishing some economic intuition, the methods that are used for the award price prediction are

presented. The best performing model, both in terms of RMSE and MAE, is the elastic net. In particular, the realized regression is a lasso regression. The obtained model is then used to provide confidence intervals for the best performing model. For the sample of 430 auctions at hand, no significant effect of bidder interactions is detected.

References

Benjamini, Y., Heller, R., & Yekutieli, D. (2009). Selective inference in complex research. *Philosophical transactions. Series A, Mathematical, physical, and engineering sciences, 367*, 4255–71.

Benjamini, Y., & Yekutieli, D. (2005). False discovery rate–adjusted multiple confidence intervals for selected parameters. *Journal of the American Statistical Association, 100*(469), 71–81.

Benoit, K., Muhr, D., & Watanabe, K. (2021). *Stopwords: Multilingual stopword lists* [R package version 2.3].

Cantino & Maxwell. (2022). *Selectorgadget: An open source bookmarklet that makes css selector generation and discovery on complicated sites a breeze.*

Casella, G., & Berger, R. (2001). *Statistical inference.* Duxbury Resource Center.

CDOT. (2018). *Rules governing construction bidding for cdot public projects.*

CDOT. (2021). *Goods and services procurement presentation for vendors.*

Chen, T., & Guestrin, C. (2016). Xgboost: A scalable tree boosting system. *CoRR, abs/1603.02754.*

Chen, T., He, T., Benesty, M., Khotilovich, V., Tang, Y., Cho, H., Chen, K., Mitchell, R., Cano, I., Zhou, T., Li, M., Xie, J., Lin, M., Geng, Y., Li, Y., & Yuan, J. (2022). *Xgboost: Extreme gradient boosting* [R package version 1.6.0.1].

Collins, M., Dasgupta, S., & Schapire, R. E. (2001). A generalization of principal components analysis to the exponential family. In T. Dietterich, S. Becker, & Z. Ghahramani (Eds.), *Advances in neural information processing systems.* MIT Press.

Colorado General Assembly. (2022). Colorado State Operating Budget [Accessed: 2022-07-18].

Decarolis, F. (2018). Comparing Public Procurement Auctions. *International Economic Review, 59*(2), 391–419.

Freitas, W. (2022). *Bizdays: Business days calculations and utilities* [R package version 1.0.11].

Friedman, J., Hastie, T., & Tibshirani, R. (2010). Regularization paths for generalized linear models via coordinate descent. *Journal of Statistical Software, 33*(1), 1–22.

Gautier, L. (2022). *Rpy2: An interface to run r embedded in a python process.*

Gregorutti, B., Michel, B., & Saint-Pierre, P. (2013). Correlation and variable importance in random forests [cite arxiv:1310.5726].

Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). *The elements of statistical learning: Data mining, inference, and prediction* (Vol. 2). Springer.

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). *An introduction to statistical learning: With applications in r.* Springer.

Landgraf, A. J., & Lee, Y. (2015). *Dimensionality reduction for binary data through the projection of natural parameters* (tech. rep. No. 890). Department of Statistics, The Ohio State University.

Landgraf, A. J., & Lee, Y. (2020). Dimensionality reduction for binary data through the projection of natural parameters. *Journal of Multivariate Analysis*, *180*, 104668.

Lee, J. D., Sun, D. L., Sun, Y., & Taylor, J. E. (2016). Exact post-selection inference, with application to the lasso. *The Annals of Statistics*, *44*(3).

Leeper, T. J. (2018). *Tabulizer: Bindings for tabula pdf table extractor library* [R package version 0.2.1].

Milgrom, P. R., & Weber, R. J. (1982). A theory of auctions and competitive bidding. *Econometrica*, *50*(5), 1089–1122.

Ooi, H. (2021). *Glmnetutils: Utilities for 'glmnet'* [R package version 1.1.8].

Ooms, J. (2020). *Hunspell: High-performance stemmer, tokenizer, and spell checker* [R package version 3.0.1].

Pearson, K. (1901). Liii. on lines and planes of closest fit to systems of points in space. *The London, Edinburgh, and Dublin Philosophical Magazine and Journal of Science*, *2*(11), 559–572.

R Core Team. (2022). *R: A language and environment for statistical computing.* R Foundation for Statistical Computing. Vienna, Austria.

Rodríguez, M. J. G., Montequín, V. R., Fernández, F. O., & Balsera, J. M. V. (2020). Bidders recommender for public procurement auctions using machine learning: Data analysis, algorithm, and case study with tenders from spain. *Complexity*, *2020*, 1–20.

Rudin, W. (1976). *Principles of mathematical analysis* (Third) [International Series in Pure and Applied Mathematics]. McGraw-Hill Book Co.

Schmalensee, R., & Willig, R. (Eds.). (1989). *Handbook of Industrial Organization* (Vol. 2). Elsevier.

Shahar, D. J. (2017). Minimizing the variance of a weighted average. *Open Journal of Statistics*, 216–224.

Silge, J., & Robinson, D. (2017). *Text mining with r: A tidy approach* (1st). O'Reilly Media, Inc.

Tadelis, S. (2012). *Game theory: An introduction* (1st ed.). Princeton University Press.

Taylor, J., & Tibshirani, R. (2016). Post-selection inference for l1-penalized likelihood models.

Tibshirani, R. (1996). Regression shrinkage and selection via the lasso. *Journal of the Royal Statistical Society. Series B (Methodological)*, *58*(1), 267–288.

Tibshirani, R., Tibshirani, R., Taylor, J., Loftus, J., Reid, S., & Markovic, J. (2019). *Selectiveinference: Tools for post-selection inference* [R package version 1.2.5].

Wickham, H. (2022). *Rvest: Easily harvest (scrape) web pages.*

Wright, M. N., & Ziegler, A. (2017). ranger: A fast implementation of random forests for high dimensional data in C++ and R. *Journal of Statistical Software*, *77*(1), 1–17.

Zou, H., & Hastie, T. (2003). Regularization and variable selection via the elastic net. *Journal of the Royal Statistical Society: Series B (Statistical Methodology)*, *67*(2), 301–320.

A. Appendix: Proofs

A.1. The Maximum is Convex on $\mathbb{R}^n$

By definition provided by Rudin (1976), a mapping $f : \mathbb{R}^n \to \mathbb{R}$ is convex if

$$f(tx + (1-t)y) \leq (1-t)f(x) + tf(y) \ \forall \ x, y \in \mathbb{R}^n \text{ and } t \in [0,1].$$

Proof: The fact that the maximum fulfils this definition may be shown via straightforward computation. Let $f(\boldsymbol{x}) = \max_i(x_1, ..., x_n)$, then

$$f(tx + (1-t)y) = \max_i(tx_i + (1-t)y_i) \overset{*}{\leq} (1-t)\max_i(y_i) + t\max_i(x_i) = (1-t)f(x) + tf(y).$$

The inequality marked with * holds because,

$$\max_i(x_i) \geq x_j \text{ and } \max_i(y_i) \geq y_j,$$

for any $j \in \{1, ..., n\}$. Thus, given that $t \geq 0$ and $(1-t) \geq 0$, we find that the initially stated inequality in our proof must hold. $\square$

A.2. The Optimal Constant $\hat{c}_m$ in R_m

Proof: We are set out to minimize the squared error within R_m, i.e., we want to find:

$$\arg\min_{\hat{c}_m} \sum_{i=1}^{n} (y_i - \hat{c}_m)^2,$$

assuming that there are n observations in R_m. The squared error function at hand is convex and differentiable, thus it has a unique minimum which may be found by setting the derivative with respect to $\hat{c}_m$ to zero. From the FOC, we obtain:

$$-2\sum_{i=1}^{n}(y_i - \hat{c}_m) \overset{!}{=} 0 \iff \sum_{i=1}^{n}y_i - \sum_{i=1}^{n}\hat{c}_m \overset{!}{=} 0 \iff \sum_{i=1}^{n}y_i - n\hat{c}_m \overset{!}{=} 0 \iff \hat{c}_m = \frac{1}{n}\sum_{i=1}^{n}y_i.$$

$\square$

A.3. Bias of Individual Trees and the Resulting Forest

We want to show that a random forest made up of B identically distributed trees $\{T_b\}_1^B$, exhibits the same bias as an individual tree T_b.

Proof:

$$\mathbb{E}\left[\hat{f}_{rf}(\boldsymbol{x}_i)\right] = \mathbb{E}\left[\frac{1}{B}\sum_{b=1}^{B}T_b(\boldsymbol{x}_i)\right] = \frac{1}{B}\sum_{b=1}^{B}\mathbb{E}\left[T_b(\boldsymbol{x}_i)\right] \overset{i.d}{=} B\frac{1}{B}\mathbb{E}\left[T_1(\boldsymbol{x}_i)\right] = \mathbb{E}\left[T_1(\boldsymbol{x}_i)\right].$$

$\square$

A.4. An Identity for the Variance of a Sum

Suppose we have B identically distributed random variables $\{T_b\}_1^B$ with finite variances. We aim to show, that

$$\mathbb{V}\mathrm{ar}\left[\sum_{b=1}^{B}T_b\right] = \sum_{k=1}^{B}\sum_{j=1}^{B}\mathbb{C}\mathrm{ov}\left[T_k, T_j\right].$$

Proof: By Steiner's displacement theorem,

$$\mathbb{V}\mathrm{ar}\left[\sum_{b=1}^{B}T_b\right] = \mathbb{E}\left[\left(\sum_{b=1}^{B}T_b\right)^2\right] - \left[\mathbb{E}\left(\sum_{b=1}^{B}T_b\right)\right]^2.$$

Squaring a sum is identical to multiplying each element of the two multiplicands pairwise, i.e.,

$$\mathbb{E}\left[\left(\sum_{b=1}^{B}T_b\right)^2\right] = \mathbb{E}\left[\sum_{k=1}^{B}\sum_{j=1}^{B}T_kT_j\right] = \sum_{k=1}^{B}\sum_{j=1}^{B}\mathbb{E}\left[T_kT_j\right],$$

and

$$\left[\mathbb{E}\left(\sum_{b=1}^{B}T_b\right)\right]^2 = \left[\sum_{b=1}^{B}\mathbb{E}\left(T_b\right)\right]^2 = \sum_{k=1}^{B}\sum_{j=1}^{B}\mathbb{E}\left[T_k\right]\mathbb{E}\left[T_j\right].$$

We thus find,

$$\mathbb{V}\mathrm{ar}\left[\sum_{b=1}^{B}T_b\right] = \sum_{k=1}^{B}\sum_{j=1}^{B}\mathbb{E}\left[T_kT_j\right] - \sum_{k=1}^{B}\sum_{j=1}^{B}\mathbb{E}\left[T_k\right]\mathbb{E}\left[T_j\right],$$

and hence,

$$\mathbb{V}\mathrm{ar}\left[\sum_{b=1}^{B}T_b\right] = \sum_{k=1}^{B}\sum_{j=1}^{B}\mathbb{E}\left[T_kT_j\right] - \mathbb{E}\left[T_k\right]\mathbb{E}\left[T_j\right] = \sum_{k=1}^{B}\sum_{j=1}^{B}\mathbb{C}\mathrm{ov}\left[T_k, T_j\right].$$

$\square$

A.5. A Lower Bound for the Variance of a Weighted Sum

A lot of times, the equal weights implied by a standard random forest are taken as given without providing further arguments as to why such a specific weighting is optimal. This proof aims to show, that for B uncorrelated and identically distributed random variables $\{T_b\}_1^B$ with finite variance, there exists a lower bound for the variance of the weighted sum of said random variables. Note that in a random forest, the trees are seldom uncorrelated, and accordingly the correlation structure should be modelled when calculating the weights. This is, however, a very complex endeavour. Alternatively, we may look into the lower bound for uncorrelated random variables to gain further understanding of why a low correlation of trees in a random forest is of such importance. As a prerequisite, the following proof utilizes the the Cauchy-Schwarz inequality (CS). In the euclidean space $\mathbb{R}^n$ with the standard inner product, it states that,

$$\left[\sum_{b=1}^{B}X_bY_b\right]^2 \leq \left[\sum_{b=1}^{B}X_b^2\right]\left[\sum_{b=1}^{B}Y_b^2\right].$$

Proof: The idea for this proof is heavily inspired by Shahar (2017). Independent of how we weigh the simple learners in our forest we require that our weights fulfil,

$$\sum_{b=1}^{B}w_b = 1.$$

Thus we find,

$$1 = \left[\sum_{b=1}^{B}w_b\right]^2 = \left[\sum_{b=1}^{B}w_b\frac{\sqrt{\mathbb{V}\mathrm{ar}\left(T_b\right)}}{\sqrt{\mathbb{V}\mathrm{ar}\left(T_b\right)}}\right]^2 \overset{CS}{\leq} \left\{\sum_{b=1}^{B}\left[w_b\sqrt{\mathbb{V}\mathrm{ar}\left(T_b\right)}\right]^2\right\}\left\{\sum_{b=1}^{B}\left[\frac{1}{\sqrt{\mathbb{V}\mathrm{ar}\left(T_b\right)}}\right]^2\right\}.$$

Now, some further simplification and a division by,

$$\frac{1}{\sum_{b=1}^{B}\mathbb{V}\mathrm{ar}\left(T_b\right)},$$

yields the following expression,

$$\frac{1}{\sum_{b=b}^{B} \frac{1}{\mathbb{V}\mathrm{ar}(T_b)}} \leq \sum_{b=1}^{B} w_b^2 \mathbb{V}\mathrm{ar}\left(T_b\right) \overset{*}{=} \mathbb{V}\mathrm{ar}\left[\sum_{b=1}^{B} w_b T_b\right].$$

Where * signifies the point at which we need the assumption that our random variables $\{T_b\}_1^B$ are uncorrelated. Further, we know that the CS inequality is binding iff there exists a $\lambda \in \mathbb{R}$ such that,

$$w_b\sqrt{\mathbb{V}\mathrm{ar}\left(T_b\right)} = \lambda\frac{1}{\sqrt{\mathbb{V}\mathrm{ar}\left(T_b\right)}} \ \forall \ b \in \{1, ..., B\}.$$

Since our weights sum up to one we discover that,

$$w_b = \frac{\lambda}{\mathbb{V}\mathrm{ar}\left(T_b\right)} \implies \sum_{b=1}^{B} \frac{\lambda}{\mathbb{V}\mathrm{ar}\left(T_b\right)} = 1 \iff \lambda = \frac{1}{\sum_{b=b}^{B} \frac{1}{\mathbb{V}\mathrm{ar}(T_b)}}.$$

This result directly leads us to the expression that we were set out to show, the weights w_b for which the CS inequality is binding and thus the weights that minimize the variance of our sum are,

$$w_b = \frac{\frac{1}{\mathbb{V}\mathrm{ar}(T_b)}}{\sum_{b=b}^{B} \frac{1}{\mathbb{V}\mathrm{ar}(T_b)}}.$$

For our identically distributed variables $\{T_b\}_1^B$, this simplifies to,

$$w_b = \frac{1}{B} \ \forall \ b \in \{1, ..., B\}.$$

$\square$