Auction Prescriptive Analytics

Machine Learning under a Modern Optimization Lens

Abhranil Chakrabarti (abhra123@mit.edu)

Bibhabasu Das (bibha175@mit.edu)

Introduction

Internet auctions have gained widespread popularity and are one of the most successful forms of electronic commerce. There are several decisions that the seller and the platform make at the start of an auction (auction format, reserve prices, duration to name a few). These decisions can have a significant impact on the final selling price and thereby, the revenue of the seller and the platform. We place ourselves in the shoes of Swoopo, which is a bidding fee auction platform and seek to maximize our revenue by altering the bid fee.

Motivation

The primary motivation for developing machine learning models for online internet auction sets is to have better interoperability of the results and policy these platforms (like Swoopo, and EBay) need to undertake in improving their auction revenue. This can be further extended to ad auctions and government auctions. Developing accurate and reliable ways of measuring the value of advertising in this environment is essential for advertisers to trade profitably on the exchange and to ensure that acquired ad impressions generate sufficient value.

The major challenge of the problem is that the counterfactuals are not observed and we can only estimate to a reasonable degree of accuracy. The high dimensionality of the dataset adds to the difficulty of this problem, leading to the exploration of machine learning approaches.

Problem Summary

Swoopo is a bidding fee auction platform, which is a type of auction in which all participants must pay a non-refundable fee to place each small incremental bid. Participants pay a fee to purchase bids. Each of the bids increases the price of the item by a small amount and extends the time of the auction by a few seconds. The major source of revenue for Swoopo is the number of bids placed at any auction times the bid fee.

Our hypothesis is that as the bid fee increases, the number of placed bids will decrease. The resulting effect on revenue needs to be studied and a policy for setting the ideal bid fee needs to be established. This policy will however depend on factors such as the item, its brand, original price and bid increment to name a few.

	Pay-per-Bid (Swoopo)
X (Feature Vector)	Type of Auction, Brand, Retail Price, Bid Increment
Z (Treatment)	Bid Fee
Y (Outcome)	Revenue (# bids * bid fee)

Datasets

For our case, we are utilizing the following Swoopo datasets to get the information on the type of auction, the product ID, bid-time, initial and final prices of the product for sale

Swoopo

(121k auctions on Swoopo, 1800 types of Products)

- Treatment: Bid Fee is defined as the cost incurred to make a bid, in cents. Values are either 65 or 75 cents and therefore can be converted into binary form.
- Outcome: Revenue here is defined as product of the number of paid bids placed (placedbids) and the bidfee which is the cost incurred to make a bid, in cents.
- Feature Vectors:
    - Type of Auction: we have four different types of auction ie. NailBiter, beginner auction, fixed-price, 100%-off auction. All of them have binary values.
    - Brand: We obtain this from product_id and item (A text string describing the product) labels in the dataset.
    - Retail Price: The stated retail value of the item, in dollars
    - Bid Increment: The price increment of a bid, in cents.

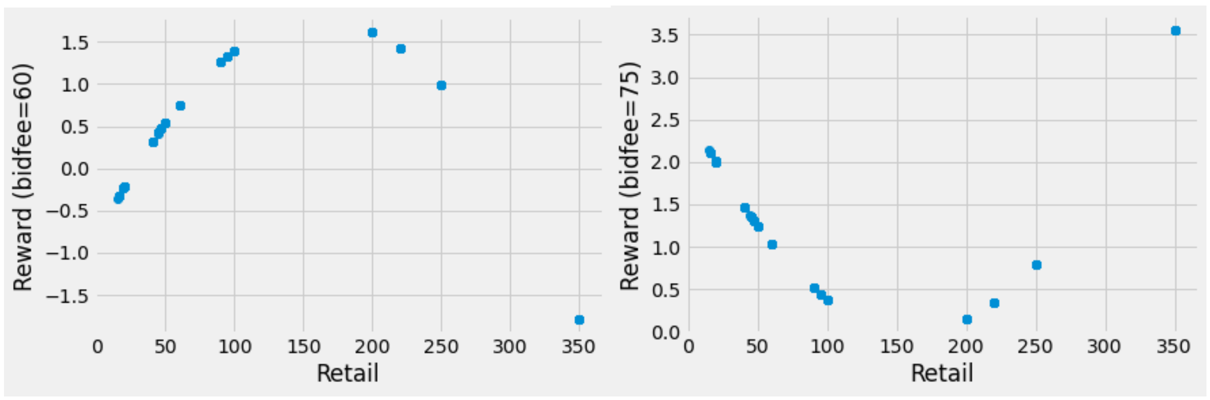
Ebay

(10k auctions on eBay, 4 types of Products)

The Primary Labels comprises auction ID, item, bid, openbid, price and auction type. This dataset is very small compared to Swoopo and was used for initial experimentation on our models.

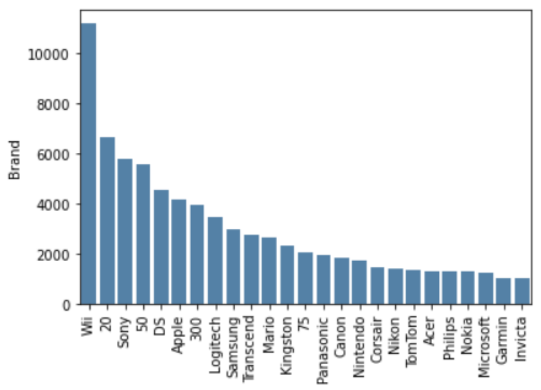
Synthetic DataSet

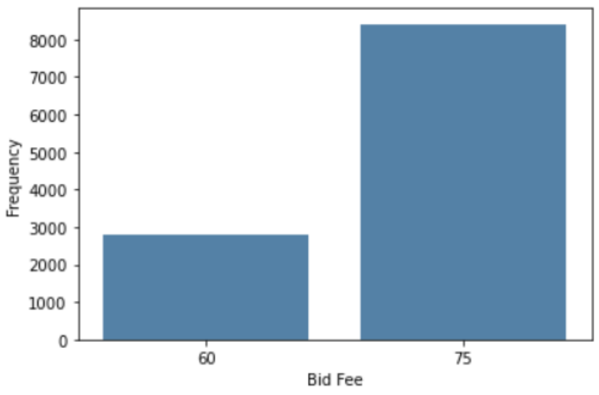
Since the counterfactuals cannot be observed on the actual dataset, we also simulate a dataset to compare the actual rewards instead of estimated rewards. The reward varies as a function of the treatment and retail price of the product as shown in the graphs below. An ideal strategy in this case would be to have a bid fee of 75 (treatment=1) for high and low retail prices and a bid fee of 60 (treatment=0) for products with moderate retail prices.



Exploratory Data Analysis

We extract some new features such as the Brand name of the product from the available product description. We also create a revenue metric as the product of the bid fee and the number of placed bids as this is the metric we need to maximize. We observe that most of the products are concentrated among a small fraction of the brands. We restrict our analysis to the top few brands as there is very little data available for the remaining brands.





The treatment values have some imbalance in their distribution, but there are a sufficient number of observations of each treatment for a decent analysis. The estimated uplift in revenue was highest when the brands were modeled separately and we decided to perform our analysis on all the auctions for "Wii", having the largest number of observations.

Methodology

We try three different approaches and perform a comparative study of the same as described below

I. Optimal Policy Trees

Optimal Policy Trees yield interpretable prescription policies that are scalable, and handle both discrete and continuous treatments. Optimal Policy Trees combine methods for counterfactual estimation in training globally-optimal decision trees from observational data.

We use Optimal Policy Trees to develop and estimate the rewards using a doubly Robust Estimator to determine the heterogeneous treatment effects. Then we develop policy trees to maximize the rewards by choosing the best policy.

Although the treatment is numeric, the only available values are 75 and 60. We can therefore view this as a categorical treatment with only two categories. We first build a doubly robust estimator on the training set and use it to learn the policy trees. To avoid any leakage of information, we build a separate doubly robust reward estimator on the test set and use it for evaluation. GridSearch is used for hyperparameter tuning and the max_depth is kept low (varying from 3 to 6) for interpretability.

II. Optimal Prescriptive Trees

Optimal prescriptive trees have an objective function that balances between optimality and accuracy. OPTs are interpretable and highly scalable, accommodate multiple treatments, and provide high-quality prescriptions.

Our goal is to maximize the outcome (revenue) and the Prescriptive Tree Maximizer is used as the model. The prescriptive trees are fit for multiple prescription factors and the results are compared.

For each prescription factor, hyperparameter tuning is done via a GridSearch. The max depth is varied from 3 to 6. The maximum depth is not chosen to be very high as the interpretability of the policies is an important factor and larger trees can become difficult to interpret. The same test set reward estimator used for the Policy Trees was used for evaluation of the prescriptive trees.

III. Causal Trees

Causal Tree Learning leverages a decision tree learning to identify an optimal strategy for splitting observed individuals into groups to estimate heterogeneous treatment effects. The splitting criterion of a causal tree learning algorithm must optimize for the balance between observed cases in each leaf which are exposed and unexposed to treatment, and must incorporate the expected accuracy of a CATE estimation made within a particular leaf.

We use the causal trees to identify groups in which the treatment effect is maximized. Subsequently, for each subgroup, a treatment is recommended or not depending on whether the treatment effect leads to positive or negative outcomes in that group (leaf). A bid fee of 75 is taken to be a positive treatment (treatment=1) while a bid fee of 60 is assumed to be no treatment (treatment=0). The Causal-ML package is used for building the causal trees. We assess the quality of estimation of the treatment effect heterogeneity using the Qini plot and use the same test set reward estimation as the OPTs to assess the performance for prescription.

We use 60% of the dataset as training and the remaining 40% for testing across all the approaches, while fixing the seed for consistency.

For reward estimation, we use a doubly robust estimator. This is used both for training the policy trees and for evaluation of all the different approaches. The advantage of the doubly robust estimator is that it will be strong if either the direct method or the inverse propensity weighted method is strong.

Hyperparameter Tuning

The tuning of hyperparameters for the models is done via cross-validation. In this case, we use k-fold cross validation. In this approach, we split the dataset into k random equal sized partitions. After this, k iterations are performed in which one partition is treated as the validation set while the remaining 'k-1' partitions are used for training. For each set of hyperparameters, the average of the decided criterion is calculated and the best-performing hyperparameter set is chosen.

Comparison Metric

We compare the revenue based on the estimation by the doubly robust reward estimator. The estimated revenue per auction is compared for the current treatment in the test set (baseline) against the estimated revenue based on the policy prescribed by each of the approaches.

$$\%\ \text{Uplift} = \left(\frac{model\ score - baseline}{baseline}\right) * 100$$

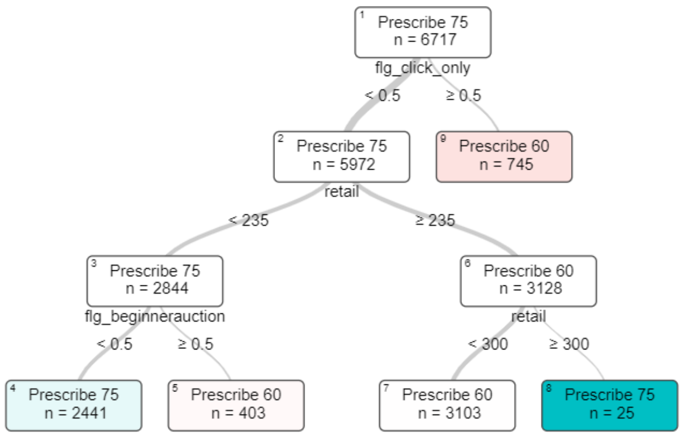
$$\text{Closeness to Benchmark} = \left(\frac{model\ score - baseline}{benchmark - baseline}\right) * 100$$

Results

The two key factors considered while assessing the effectiveness of an approach was the uplift in expected revenue and the interpretability of the recommended policy. The results are compared to the baseline (existing treatment in the test set) which had a revenue of $ 39.22 per auction. The benchmark (maximum attainable revenue based on the reward estimated) was computed to be $ 50.47 per auction (29% higher than the current score).

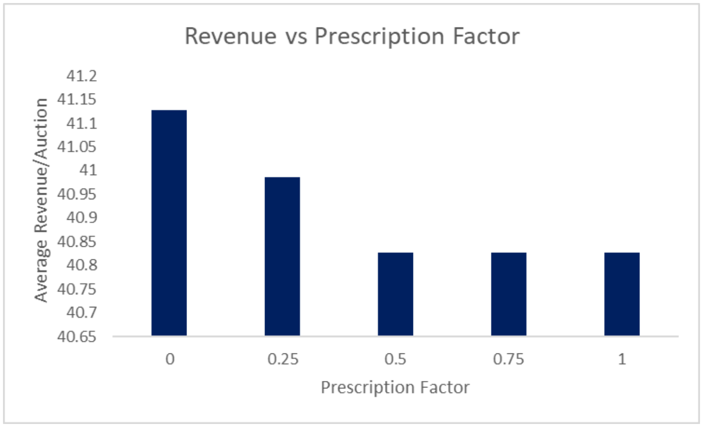
Policy Trees

The tree for the best hyperparameters is shown in the diagram below. The retail price and the type of auction are important variables used in the recommendation of a strategy. The depth of the tree is low and the policy is very interpretable. The revenue per auction for the best policy tree was $ 41.27, an increase of 5% over the baseline and 18% of the benchmark uplift.

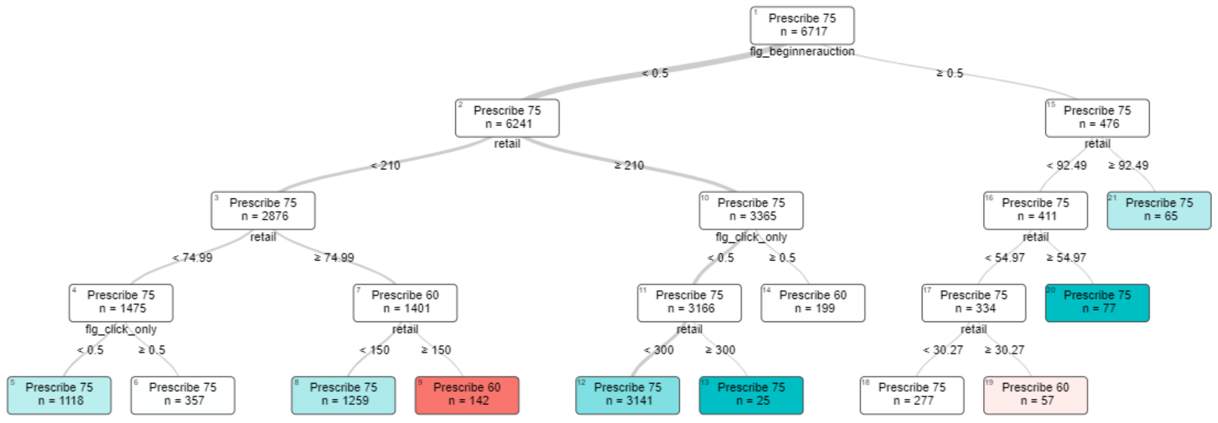


Prescriptive Trees

The expected revenue per auction is compared for different prescription factors and we observe that lower prescription factors seem to be performing better on the test set.

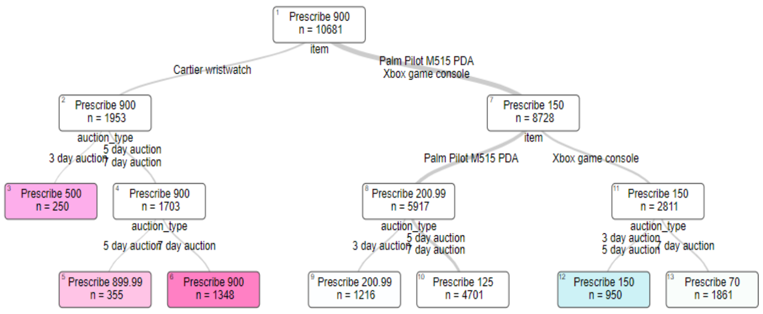


The tree for the best hyperparameters is shown in the diagram below. The retail price and the type of auction are important variables used in the recommendation of a strategy, while bid increment



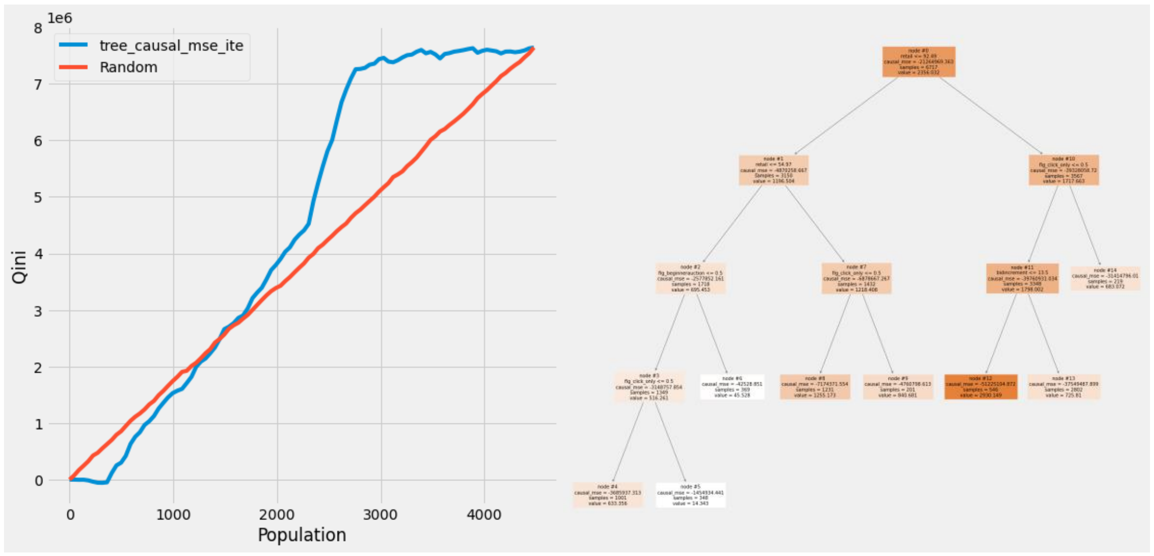
does not seem to play an important role in the recommendations. The complexity of the tree is slightly higher but the policy is still interpretable. The revenue per auction for the best prescriptive tree was $ 41.12, an increase of 5% over the baseline and 17% of the benchmark uplift.

Although the experiments on the EBay dataset did not yield any noticeable improvements in the expected revenue, it is worth noting that the policy learnt is very interpretable and depends on the item auctioned and the duration of the auction. The policy is intuitive and recommends lower opening bid prices for longer auctions with a performance very close to the baseline.



Causal Trees

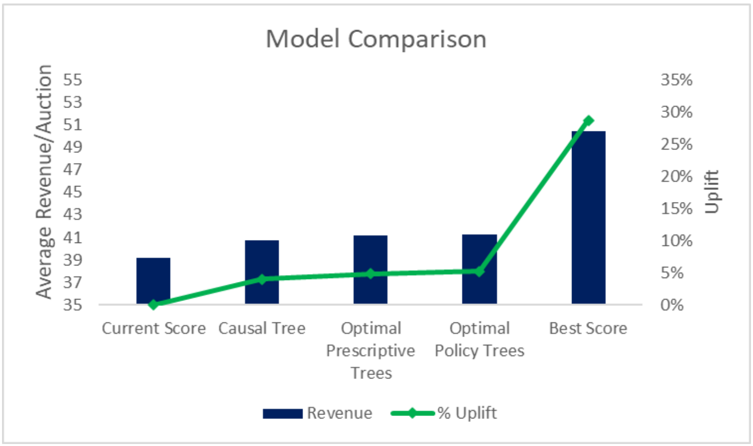
The Qini plot suggests that the split generated is decent for maximizing heterogeneity in the treatment effects for individuals in the group. The tree with the generated splits is also shown below. Using the same reward estimator, the revenue per auction for this tree was observed to be $ 40.8, an increase of 4% over the baseline and 14% of the benchmark uplift.



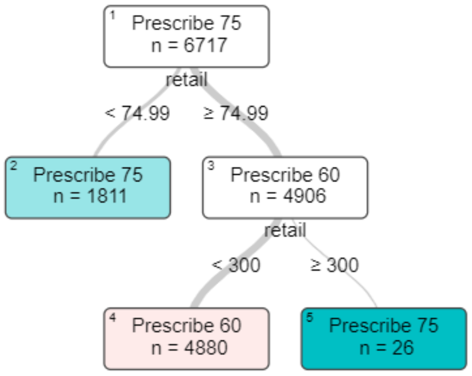
Model Comparison

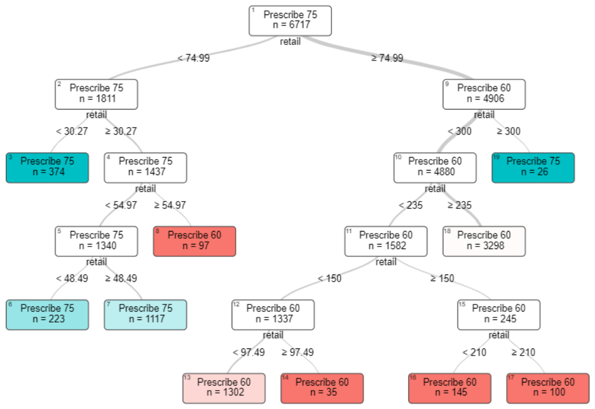
The results for each of the models on the Swoopo dataset are compiled below. We compare the average estimated revenue per auction, the percentage uplift from the baseline and the closeness to the benchmark for each of the three models. For the best model, we observe an uplift of approximately $2 per auction (from $39 to $41) which can be significant as the number of auctions run increases.

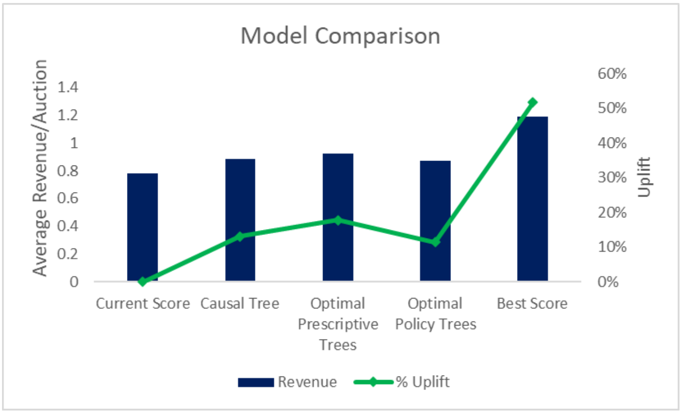
Model	Revenue	% Uplift	Closeness to Benchmark
Current Score	$ 39.22	0%	0%
Causal Tree	$ 40.78	4%	14%
Optimal Prescriptive Trees	$ 41.13	5%	17%
Optimal Policy Trees	$ 41.27	5%	`18%
Best Score	$ 50.47	29%	100%



Synthetic Data







The trees are very interpretable and perform as expected. For very low and very high retail prices, the recommended strategy for both the trees is to have a bid fee of 75 while for moderate retail prices, both trees recommend a bid price of 60 which is in accordance with the simulated data. We observe similar uplift (highest for prescriptive trees at 18%) even for the simulated dataset with actual (known) values for both the treatment effects.

Conclusion

The policy trees are observed to have lower depth than the prescriptive trees for the same experiments and this is observed to be true across experiments. A possible reason could be that the prescriptive trees have a dual objective, leading to more complexity. The observations are summarized below and hold true for this dataset (and are not a generalization for all datasets)

	Performance	**Interpretability**
Policy Trees	Good performance on the test set	Low Depth, High Interpretability
Prescriptive Trees	Performance almost as good as Policy Trees	Higher Complexity, Lower Interpretability
Causal Trees	Lower performance as objective is different	Moderate Complexity and Interpretability

As discussed, we observe an uplift of approximately \$2 per auction (from \$39 to \$41) which becomes significant as the number of auctions run rises. Our approaches are able to provide effective recommendations and result in an uplift of 5% in the revenue for Swoopo. The dataset is still quite small and as the amount of data increases, it has the potential to provide more powerful recommendations, which can play a significant role in improving revenues, be it Swoopo or other auctions with different mechanisms and decisions.

Future Work

The assumption of unconfoundedness may not always hold. For instance, shill bidding is a practice wherein the seller submits fake bids to artificially inflate the final price. This affects both the decision Z and the final outcome Y and is therefore, a confounding variable. One approach would be to treat such auctions as anomalies, identifying them through techniques such as clustering. Variables such as the time of bids (shill bids are likely to occur early), number of bids could be useful indicators of rigged auctions.

Appendix

Abhranil came up with developing and implementing different causal inference models and decision tree learning methods over the auction field. We started by testing simple models over the eBay dataset we found on Kaggle. But since the eBay dataset was comparatively smaller, Bibha found and extracted a new auction dataset for Swoopo, a pay-per-bid platform. We then started preprocessing the dataset and undertook basic EDA to better understand the nature of the data. Both of them studied and tried to understand the field of auctions and how each of these different types of auction work in varied setups. Bibha worked on developing causal trees and how we could scale it up for Swoopo. Abhranil worked on prescriptive and Policy trees and further tried different evaluation methods. We then split up the work with Bibha on the presentation and Abhranil on the report