Exploration of Avito Demand Prediction

Chenhua Shi, Huan Hu, Yunshu Geng, Shengzhe Zhang, Yanzhi Ding, Yilin Piao
University of California, San Diego
{c8shi,huh015, yugeng, shz275, yad027, yipiao}@ucsd.edu

June 16, 2018

Abstract

Avito, Russia's largest classified advertisements website, is challenging us to predict demand for an online advertisement based on its full description (title, description, images, etc.), its context (geographically where it was posted, similar ads already posted) and historical demand for similar ads in similar contexts. With this information, Avito can inform sellers on how to best optimize their listing and provide some indication of how much interest they should realistically expect to receive. In this report, we tried to improve the demand prediction performance by exploring the dataset and training various regression models. We investigated the dataset by examining seven different hypotheses corresponding to the demand and extract new features for demand prediction. With the existed and newly extracted features, we optimized six different regression method including Linear Regression, Random Forest, Support Vector Regression (SVR), Gradient Boosting Regressor, LightGBM and XGboost to predict the demand. The performance of the prediction is evaluated on root mean square error of predicted price for test dataset.

1 Introduction and Dataset

To help finish the prediction of demand for an online advertisement based on its text description, images, and other features by Avito, Russia's largest classified advertisement website, we use Python to do the data exploration, feature engineering and build machine learning models to achieve the prediction in the Kaggle competition, Avito Demand Prediction Challenge. Through the predictions from the built model, Avito can inform sellers on how to best optimize the listing and provide some indication of how much interest they should realistically expect to receive.

Avito Demand Prediction Challenge totally provides `train.csv`, `train_active.csv`, `periods_train.csv`, `test.csv`, `test_active.csv`, `periods_test.csv` for the text features and `train_jpg.zip` for the picture features. We mainly use `train.csv` and `periods_train.csv` files to do the data exploration such as null hypothesis, distributions, and statistics and to build the training model for the regression. Meanwhile, we use `test.csv` and `periods_test.csv` files to test the build models. Now, let us further explore the features provided in the `train.csv`. Totally 17 features are provided in the csv file. They are item_id, user_id, region, city, parent_category_name, category_name, param_1, param_2, param_3, title, description, price, item_seq_number, activation_date, user_type, image, and image_top_1. The training set totally contains 1,503,424 unique item ids, 771,769 unique user ids, 28 unique regions, 1,733 unique cities, 9 distinct parent category names, 47 distinct category names, 28,232 distinct item sequence number, 21 different activation date, and 3 different user types. There exist around 4%, 43.5%, 57.4%, 7.7%, 5.7%, 7.5%, 7.5% missing values in the param_1, param_2, param_3, description, price, image, and image_top_1. Thus, we usually fill zero for these missing values and then investigate the dataset. We have seven different null hypotheses for different features at all for the exploration of dataset. We basically do the following investigations in the data exploration part. They are the influence of region and city on GRP in Russia, the relationship of text description and the deal probability, the influence of price rank difference based on parent category or category on deal probability, the influence of missing price on deal probability, the impact of item sequence number on deal probability, the relationship of the period of the advertisement is displayed and the deal probability, and the influence of images for items on deal probability. After data exploration, we focus on the task of feature engineering since we want to gain more useful features from the data exploration for the train-

ing models. During the feature engineering, we will fill the missing value in a more reliable method such as fill them based on the mean value of different categories. In addition, we do the data transformation so as to make them denser, which will be more useful for training the models. Meanwhile, we apply the one hot encoding to convert category features into a format that works better with regression algorithms. Lastly, natural language process such as TF-IDF, stemming are applied to process the text information. Here, more useful features are built for the aim of improving the performance of the models based on the data exploration. For the provided testing dataset that has the same format and features as the training set, we use them to test the performance of the models based on the metric accuracy root mean square error (RMSE).

For the experiment part, we use six different classical machine learning models that Linear Regression, Random Forest, Support Vector Regression (SVR), Gradient Boosting Regressor, LightGBM and XGboost to do the regression task. In order to finish building the training models for the prediction of demand for an online advertisement, we use sklearn library, LightGBM and XGboost library to achieve the goal. For the multi linear regression model, we will apply adjusted approach and p-value approach to do the selected model. Meanwhile, we can compute the weights for the coefficients of different features. For the Random Forest, we will tune multi hyper-parameters such as the number of trees in the forest, the max depth of the tree, the function to measure the quality of a split, and other parameters. For the SVR, we will also tune multi hyper-parameters such as the penalty parameters, the kernel, the kernel coefficient, and other parameters under grid-search. For the LightGBM and XGboost, the multi hyper-parameters similar to random forest such as the max depth of the tree, minimum children weight, gamma, subsample and other parameters will be tuned. After that, the training set is split into two parts that training and validation, which on the ratio 8:2. Here, the validation data is used to test the performance of the models so as to prevent the overfitting or underfitting and the hyperparameters tuning. Lastly, we will apply the metric accuracy root mean square error (RMSE) to test the predicted result on unseen data. Finally, the predicted deal probability will be uploaded to the Kaggle competition.

Overall, the reference and more detailed background about the Avito Demand Prediction Challenge will be introduced in the related works and background section. The investigations of the null hypotheses behind the dataset are shown in the data exploration. All the theories we use in the data exploration, feature engineering, and experiment such as different methods for computing p-value, natural language processing, multi different machine learning models, metric accuracy and so on are explained in detail in the theory part. For the feature engineering, the detailed explanation of how to process the provided dataset features and more useful features are built are shown in this part. The training process, validating process, feature selections, hyper-parameters tuning, and metric evaluation are explained in the experiment part. The last part is the conclusion that concludes all the work we do for this Avito Demand Prediction Challenge and the comparison of the performance of different machine learning models for this competition.

2 Related Works and Background

Nuance detail and variance could lead to a big change in demand. But analyzing the quantitative effect on demands require considerable amount of work .Avito, Russia's largest classified advertisements website, is deeply familiar with this problem. Sellers on their platform sometimes feel frustrated with both too little demand (indicating something is wrong with the product or the product listing) or too much demand (indicating a hot item with a good description was underpriced).

In their fourth Kaggle competition, Avito is challenging researchers to predict demand for an online advertisement based on its full description (title, description, images, etc.), its context (geographically where it was posted, similar ads already posted) and historical demand for similar ads in similar contexts. With this information, Avito can inform sellers on how to best optimize their listing and provide some indication of how much interest they should realistically expect to receive. The demand is characterized as a continuous number between 0 and 1, namely deal probability. And the goal is to analyze the factors that affect deal probability, and lastly finalize a model that do the best predictions of deal probability using these factors.

Under this initiative, we want to explore the factors that have statistically significant effect on demand using deal probability as numerical representation. We come up with various guess of potential factors. Price for example is a promising factor in demand. As economic

model often introduce price sensitivity[CR09], we want use this idea in our exploration. Region GDP,which could represent general wealth in an area, may also affect the demand within that area.As related work[EHF04] states different consumer behavior across cities, we want to explore if such phenomenon exists in our data.

Besides exploration of factors under intuition and precedent works, we want to empower ourselves with more advanced models dealing with this problem. Linear Regression[Par12] is a classic approach, but it has many constraints in accuracy and feature selection. Machine learning techniques come to our list naturally.We looked into Random Forest, Support Vector Regression, Gradient Boosting Regression, XGBoost, and LightGBM to provide better alternatives. Random forests[Bre01] are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. After a large number of trees is generated, they vote for the most popular class. XGBoost is an extreme gradient boosting. The model of XGBoost[TC16] is the tree ensembles, a set of classification and regression tress. LightGBM[GK17] have the almost best performance contributing to the more complex regularize and split rule but easier to be overfitting.Support Vector Regression is useful for both Linearly Seperable and Non-linearly Seperable data. By exploring these different approaches, we may achieve a better model in representing the relationship between potential factors and demand.

3 Theory

The theory part mainly includes four different pasts: Hypothesis Test Method, Feature Engineering, Machine Learning Models, and Metric Accuracy. Hypothesis Test Method provides us the approaches we use in investigating the null hypothesis. Feature Engineering demonstrates the method we use to pre-process dataset and do data transformation. Machine Learning Models explain how different models work, formula deduction, and hyper-parameters tuning in regression problem. Metric Accuracy demonstrates the way we use to compute loss by RMSE and feature importance by F1 score.

3.1 Hypothesis Test Method

3.1.1 Mann-Whitney U test

Mann-Whitney U test is a non-parametric test that is used to test whether two sample means are equal or not. The following assumptions are assumed: 1. Draw the sample from the population randomly. 2. Independence within the samples and mutual independence is assumed. 3. Ordinal measurement scale is assumed. Thus, we can use the following formula (1)to compute:

$$U = n_1 n_2 + \frac{n_2(n_2+1)}{2} - \sum_{i=n_1+1}^{n_2} R_i \quad (1)$$

Where $n1$ and $n2$ represent the sample size and R represents the rank of the sample size. Here, p-value is assumed an asymptotic normal distribution. If the p-value is smaller than 0.05, we can reject the null hypothesis. Otherwise, we cannot reject the null hypothesis if the p-value is bigger than 0.05.

3.1.2 Wilcoxon signed-rank test

The Wilcoxon signed-rank test is a non-parametric statistical hypothesis test used to compare two related samples, matched samples, or repeated measurements on a single sample to assess whether their population mean ranks differ.

Assumptions

1. Data are paired and come from the same population. 2. Each pair is chosen randomly and independently[citation needed]. 3. The data are measured on at least an interval scale when, as is usual, within-pair differences are calculated to perform the test.

Test procedure

1. Let N be the sample size, i.e., the number of pairs. Thus, there are a total of 2N data points. For pairs i = 1, ..., N, let $x_{1,i}$ and $x_{2,i}$ denote the measurements.

H0: difference between the pairs follows a symmetric distribution around zero.

H1: difference between the pairs does not follow a symmetric distribution around zero.

2. For i=1,...,N i = 1, ..., N, calculate $|x_{2,i} - x_{1,i}|$ and $\text{sgn}(x_{2,i} - x_{1,i})$, where sgn is the sign function.

3. Exclude pairs with $|x_{2,i} - x_{1,i}| = 0$. Let N_r be the reduced sample size.

4. Order the remaining N_r pairs from smallest absolute difference to largest absolute difference,$|x_{2,i} - x_{1,i}|$.

5. Rank the pairs, starting with the smallest as 1. Ties receive a rank equal to the average of the ranks they span. Let R_i denote the rank.

6. Calculate the test statistic W.

If the p-value is smaller than 0.05, we can reject the null hypothesis. Otherwise, we cannot reject the null hypothesis if the p-value is bigger than 0.05.

3.1.3 Kruskal-Wallis test

The Kruskal-Wallis test is a non-parametric (distribution free) test, and is used when the assumptions of one-way ANOVA are not met. Both the Kruskal-Wallis test and one-way ANOVA assess for significant differences on a continuous dependent variable by a categorical independent variable (with two or more groups). In the ANOVA, we assume that the dependent variable is normally distributed and there is approximately equal variance on the scores across groups. However, when using the Kruskal-Wallis Test, we do not have to make any of these assumptions. Therefore, the Kruskal-Wallis test can be used for both continuous and ordinal-level dependent variables.

Method

1. Rank all data from all groups together; i.e., rank the data from 1 to N ignoring group membership. Assign any tied values the average of the ranks they would have received had they not been tied. 2. The test statistic is given by equation (18):

$$H = (N-1)\frac{\sum_{i=1}^{g} n_i(\bar{r}_{i\cdot} - \bar{r})^2}{\sum_{i=1}^{g}\sum_{j=1}^{n_i}(r_{ij} - \bar{r})^2} \qquad (2)$$

, where: n_i is the number of observations in group i a. r_{ij} is the rank (among all observations) of observation j from group i b. N is the total number of observations across all groups $\bar{r}_{i\cdot} = \frac{\sum_{j=1}^{n_i} r_{ij}}{n_i}$ is the average rank of all observations in group i $\bar{r} = \frac{1}{2}(N+1)$ is the average of all the r_{ij}. If the data contain no ties the denominator of the expression for H is exactly $(N-1)N(N+1)/12$and $\bar{r} = \frac{N+1}{2}$.

Thus obtain result by equation (19)

$$H = \frac{12}{N(N+1)}\sum_{i=1}^{g} n_i\left(\bar{r}_{i\cdot} - \frac{N+1}{2}\right)^2 \qquad (3)$$

$$H = \frac{12}{N(N+1)}\sum_{i=1}^{g} n_i\bar{r}_{i\cdot}^2 - \ 3(N+1). \qquad (4)$$

The p-value is approximated by $\Pr(\chi^2_{g-1} \geq H)$. If some n_i values are small (i.e., less than 5) the probability distribution of H can be quite different from this chi-squared distribution. If a table of the chi-squared probability distribution is available, the critical value of chi-squared, $\chi^2_{\alpha:g-1}$, can be found by entering the table at g − 1 degrees of freedom and looking under the desired significance or alpha level.

If the statistic is not significant, then there is no evidence of stochastic dominance between the samples. However, if the test is significant then at least one sample stochastically dominates another sample.

3.1.4 Hypothesis Test

The null hypothesis is that the average random levels are the same H0: MU(h) = MU(r) and the Alternative hypothesis is Hr: MU(h) != MU(r). In the hypothesis testing, we assume that H0 is true and find out how likely our data are under this model. When we make a decision, we can say that the observations can not reject the null hypothesis as p-value > significance level. If the p-value < significance level, then we can conclude that the observations reject the hypothesis and be in favor of the alternative. Here, the cutoff is called significance level of a test.

Generally, the p-value is not the chance that the null hypothesis is true: the hypothesis is either true or not. When we reject the null hypothesis, we do not know if we have been unlucky with our sampling and observe a rare event or if we make the correct decision. Here, Type I error is the significance of the test (alpha). Type II error is the power of a test (1-Beta).

3.1.5 Bootstrap Simulation

Bootstrap simulation is used to construct a confidence interval of unit characteristic. Consider the situation where the data are realizations of random variables $x_1, x_2, ..., x_n$ which satisfy an independent and identical distribution P,i.e.

$$x_1, x_2, ..., x_n i.i.d\ P \qquad (5)$$

As P is unknown, we use the empirical distribution $\widehat{P}$ which places probability mass 1/n on every data point X_i, i = 1, 2, ..., n. The recipe is then to simulate from $\widehat{P}$ and generate simulated data. Such a simulated new data set is called a bootstrap sample. We can now compute our estimator, sample mean in this case, based on the bootstrap sample and then repeat this process many times to get an approximate distribution of our estimator.

3.2 Feature Engineering Method

3.2.1 Natural Language Processing (NLP)

Basically, NLP transforms unstructured text into vectors. And then these vectors can be used for machine learning application. First of all, let us make certain the terminology that we use in NLP.

Corpus: The collection of all documents you are interested in. Here, all the description constitutes of the corpus.

Document: A single entry from the corpus. Here, each item will have its own description that can be treated as the document.

Token: Words in a list. Each document is a list of tokens.
Bag of Words: A vector representation of a document based on word counts.
Stop Words: Words are ignored that too common to be useful.

After that, we apply seven steps to process the text. First of all, we do the tokenization by taking the document and splitting into a list of tokens. After that, we need to convert all the tokens into lower cases. Meanwhile, we have to remove all stop words and punctuation. In addition, we want to reduce words to root form by applying either stemming or lemmatization. Here, we can either draw cloudword to explore top hot words in the corpus or compute the TF-IDF to use them in the model training.

3.2.2 Term Frequency and Inverse Document Frequency (TF-IDF)

Term frequency is the number of times that the token t appears in document d (18) that can be expressed as following:

$$tf(t,d) = freq(t,d) \tag{6}$$

Inverse document frequency is a score for how unique a token is across the corpus. The number of documents with specific token (19) can be expressed as following:

$$idf(t,d) = log(\frac{N}{n}) \tag{7}$$

Here, N is the total documents and n is the number of documents with specific token.

TF-IDF is the combination of the normalized token counts and then multi with IDF. The formula (20) for computing TF-IDF is shown below:

$$tf - idf(t,d) = tf(t,d) * idf(t,d) \tag{8}$$

3.2.3 One Hot Encoding

One hot encoding transforms categorical features to a format that works better with classification and regression algorithms. We will apply sklearn package that LabelEncoder to encode labels with values between 0 and (N-1) classes.

3.2.4 GridSearchCV

GridSearchCV uses k-fold cross-validation to determine best hyper-parameters values. Fitting the model for each possible combinations of parameters on a parameter grid, it will retain the best scored combination. The training set is separated into k-1 folds and the model is trained for each small fold. Each training result is cross validated with the remaining data.

3.2.5 Principal Component Analysis (PCA)

PCA is a linear dimension decomposition method using Singular Value Decomposition(SVD). Due to feature engineering, we have 363 principle components in the dataframe. We use PCA here to decompose the number of dimension of principle components to accelerate SVR process and improve the performance. The first thing to do before applying PCA is to standardize the dataset. Standardization process will scale all the features until they behave as a standard normal distribution. The importance of rescaling is that if the features are not scaled, PCA will misprocess those features which have distinguished variance due to scale. PCA transform the correlated features into some uncorrelated features called principle components, which contains as much as possible variability of the dataset[Wik18]. The first component who carries the most variability (1) is given by:

$$\mathbf{w}_{(1)} = \arg\max_{\|\mathbf{w}\|=1} \left\{ \sum_i (t_1)^2_{(i)} \right\} = \arg\max_{\|\mathbf{w}\|=1} \left\{ \sum_i \left(\mathbf{x}_{(i)} \cdot \mathbf{w}\right)^2 \right\}$$

Figure 1: First Component with the most Variability

Where X is the sample data matrix, w is vectors of weight, and t is principle component scores vector. The following components (2) are calculated by:

$$\hat{\mathbf{X}}_k = \mathbf{X} - \sum_{s=1}^{k-1} \mathbf{X}\mathbf{w}_{(s)}\mathbf{w}_{(s)}^{\mathrm{T}}$$

$$\mathbf{w}_{(k)} = \arg\max_{\|\mathbf{w}\|=1} \left\{ \|\hat{\mathbf{X}}_k \mathbf{w}\|^2 \right\} = \arg\max \left\{ \frac{\mathbf{w}^T \hat{\mathbf{X}}_k^T \hat{\mathbf{X}}_k \mathbf{w}}{\mathbf{w}^T \mathbf{w}} \right\}$$

Figure 2: Components Computation

W(k) is the kth eigenvector of XTX. The maximum values in the bracket are given by corresponding eigenvalues. Then the decomposition of matrix X is shown below. Here, W is the matrix contains W(k) as columns. Then, the Singular Value Decomposition (SVD) of X will do the reduction of dimensions. SVD of X (3) is also shown below where U is the left singular vector of X, W is the right singular vectors of X, and $\sum$ is the matrix of singular values of X.

$$\mathbf{T} = \mathbf{XW}$$

$$\mathbf{X} = \mathbf{U\Sigma W}^T$$

$$\begin{aligned}\mathbf{X}^T\mathbf{X} &= \mathbf{W\Sigma}^T\mathbf{U}^T\mathbf{U\Sigma W}^T \\ &= \mathbf{W\Sigma}^T\mathbf{\Sigma W}^T \\ &= \mathbf{W\hat{\Sigma}}^2\mathbf{W}^T\end{aligned}$$

$$\begin{aligned}\mathbf{T} &= \mathbf{XW} \\ &= \mathbf{U\Sigma W}^T\mathbf{W} \\ &= \mathbf{U\Sigma}\end{aligned}$$

$$\mathbf{T}_L = \mathbf{U}_L\mathbf{\Sigma}_L = \mathbf{XW}_L$$

Figure 3: SVD Computation

3.3 Machine Leaning Models

3.3.1 Linear Regression

In Statistics, linear regression is a tool to modeling a relationship between a dependent variable and one or more independent variables. There are called simple linear regression and multiple regression model. Simple linear regression is bivariate, which contains two variables: y and x. Multiple linear regression contains multiple variables: y and x_1,x_2 and etc. The regression have the form (9):

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \varepsilon_i \quad (9)$$

In linear regression, we want to predict the relationship between and using a function (the linear model) generated from the data we have. Interpreting the regression model, all else holding constant, the intercept, $\beta 0$, of the regression model is the predicted value of outcome when all regressor equals 0, sometimes this value has no real world meaning. The coefficient of each regressor in the model means, all else holding constant, how much the predicted outcome will change when the regressor change by 1 additional unit. The e_i is the called the error term, it contains all the factors that influence the regress and other than x.

Despite of regressor and error term, we also wanted to know how close the real data is to our fitted regression line. The first method is looking at the R^2. And R^2 can be calculated in three ways. First, it can be calculated by squaring the correlation coefficient of x and y. This is the most common way we calculate it. Second, squaring the correlation coefficient of y and $\widehat{y}_i$. Third, the calculation is based on the formula (10):

$$R^2 = \frac{explained\,variability\,in\,y}{total variability\,in\,y} \quad (10)$$

By definition. And, by using ANOVA we can calculate the explained variability and total variability in y.

The residual, $e_i = y_i - \widehat{y}_i$, is the difference between the value of the dependent variable predicted by the model, $\widehat{y}_i$, and the true value of the dependent variable, y_i. One method of estimation is ordinary least squares. This method obtains parameter estimates that minimize the sum of squared residuals, SSR (11):

$$SSR = \sum_{i=1}^{n} e_i^2. \quad (11)$$

Thus, the R^2 equation (12) can be written as,

$$R^2 = \frac{SS\,total - SS\,error}{SS\,total} = \frac{\sum(y-\bar{y})^2 - \sum e_i^2}{\sum(y-\bar{y})^2} = \frac{SS\,model}{SS\,total} \quad (12)$$

Where $\sum e_i^2$ stands for the unexpected variability in the model. In the regression with one regressor, the first method of R^2 calculation is enough.The reason we need three ways to calculate the R^2 is that in the multiple linear regression we can't calculate R^2 as the square of the correlation between x and y because we have multiple x_s.

In our study, we also included another measurement of explained variability, adjusted R^2. Two predictor variables are said to be collinear when they are correlated. We use adjusted R^2 to avoid this collinearity. Because the addition of those collinear variables brings nothing to the model. Thus, we prefer the simplest best model, parsimonious model. For adjusted R^2, if the added variable doesn't really provide any new information, adjusted R^2 will not increase. The formula (13) of adjusted is given by,

$$R^2 adj = 1 - (\frac{SS\,error}{SS\,total}) \times (\frac{n-1}{n-p-1}) \quad (13)$$

where n is the number of cases and p is the number of predictors (explanatory variables) in the model. The adjusted R^2 applies a penalty for the number of predictors included in the model.

The second is the p-value approach. we can generate the hypothesis test of the significance of the predictors. H_0: $B_i = 0$ when other explanatory variables are included in the model. H_1 : $B_i \neq 0$ when other explanatory variables are included in the model.

Having these two methods, the $R^2 adj$ approach and the p-value approach. We have two way, backward-elimination and forward-selection to use these two methods to test our regressors.

When we do the R^2adj elimination, we need to start with the full model, drop one variable at a time and record R^2adj of each smaller model and pick the model with the highest increase in R^2adj. We repeat the process until none of the models yield an increase in R^2adj. When we do the p-value elimination, we drop the variable with the highest p-value and refit a smaller model and repeat the process until all the variables left are significant.

When we do the forward selection, we start with the regressand and each regressor. Pick the model with the highest R^2adj and lowest p-value. Then we add the remaining variables one at a time and pick the vairiables.

Our linear regression model depends on the conditions that the residuals are nearly normal, having constant variability, independent. Also, each variable is linear related to the outcome. These conditions can be checked graphically. The nearly normality can be checked by normal probability plot and/or histogram of residuals. To check the constant variability in residuals we use scatterplot of residuals and/or absolute value of residuals vs. fitted (predicted). The independence can be checked by scatterplot of residuals vs. order of data collection. Last, the linear relationships can be checked using scatterplot of residuals vs. each (numerical) explanatory variable.

3.3.2 Lasso

In our study, we also applied the linear regression via Lasso.Lasso is a regression analysis method that performs both variable selection and regularization in order to enhance the prediction accuracy and interpretability of the statistical model it produces. In Lasso, the outcome y_i, for cases i = 1,2,...,n, features $X\,ij$, j = 1,2,...,p. We want to minimize,

$$\sum(y_i - \sum x_{ij} + \beta_j)^2 + \lambda \sum \mid \beta_j \mid \qquad (14)$$

which is equivalent to minimizing sum of squares with constraint $\sum \mid \beta_j \mid$. And the solution should be,

$$sign(\widehat{\beta})(\mid \widehat{\beta} \mid -\lambda)+ \qquad (15)$$

where $\widehat{\beta}$ is usual least squares estimate.

Lasso penalties have powerful statistical and computational advantages. Three common applications of Lasso are Logistic for classification, Near-isotonic regression and the matrix completion problem. Lasso help us do the variable selection and shrinkage. Lasso penalties provide a natural to encourage/enforce sparsity and simplicity in the solution. Lasso penalties are convex and the assumed sparsity can lead to significant computational advantages.

By using Lasso, we start with large value for (very sparse model) and slowly decrease it. And most coordinates that are zero never become non-zero.

3.3.3 Random Forest

Random forests are a combination of tree predictors such that each tree depends on the values of a random vector sampled independently and with the same distribution for all trees in the forest. The generalization error for forests converges a.s. to a limit as the number of trees in the forest becomes large. The generalization error of a forest of tree regressor depends on the strength of the individual trees in the forest and the correlation between them. Using a random selection of features to split each node yields error rates that compare favorably to Adaboost [FSA99], but are more robust with respect to noise[Bre01].

The common procedure is that for the k-th tree, a random vector Θ_k is generated, independent of the past random vectors $\Theta_1, \Theta_2, ..., \Theta_{k-1}$ but with the same distribution; and a tree is grown using the training set and Θ_k , resulting in a regressor $h(\mathbf{x}, \Theta_k)$ where $\mathbf{x}$ is an input vector. For instance, in bagging the random vector Θ is generated as the counts in N boxes resulting from N darts thrown at random at the boxes, where N is number of examples in the training set. In random split selection Θ consists of a number of independent random integers between 1 and K. The nature and dimensionality of depends on its use in tree construction. After a large number of trees is generated, they vote for the most popular class. We call these procedures random forests.

3.3.4 Support Vector Regression (SVR)

Support Vector Machine (SVM) is a sparse kernel decision machine which can be applicable to regression problems. For linear problem, the function we want to approximate (4) is:

$$y = f(x) = <w, x> + b = \sum_{j=1}^{M} w_j x_j + b,\ y, b \in \mathbb{R}, x, w \in \mathbb{R}^M$$

$$f(x) = \begin{bmatrix} w \\ b \end{bmatrix}^T \begin{bmatrix} x \\ 1 \end{bmatrix} = w^T x + b \quad x, w \in \mathbb{R}^{M+1}$$

Figure 4: Function to approximate for Linear Problem

The second equation is for multivariable re-

gressions. To reach the goal, SVR trains data with a symmetrical loss function, which is called epsilon-intensive loss function[Wel]. Epsilon (5) is threshold which determines whether the point should be penalized (outside the tube) or not (inside the tube) by taking the absolute error compared with epsilon:

$$V_\varepsilon(r) = \begin{cases} 0 & if\ |r| < \varepsilon \\ |r| - \varepsilon & otherwise \end{cases}$$

Figure 5: Absolute Error Compared with Epsilon

By using this, we can generate a tube around the estimate function (6) below:

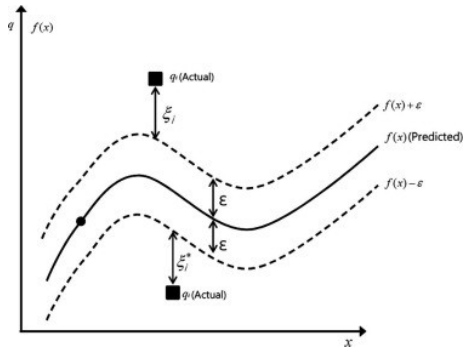

Figure 6: Tube around the Estimate Function

The goal is to find a function with narrowest tube (7), i.e. with least error between predicted value and desired value. Then, the problem becomes:

$$\begin{aligned} \text{minimize} - \mathbf{w}, \xi, \hat{\xi} \quad & \frac{1}{2}||\mathbf{w}||^2 + \frac{C}{2}\sum_i(\xi_i^2 + \hat{\xi}_i^2) \\ \text{subject to} \quad & \mathbf{w}^T\Phi_i + b - y_i \le \varepsilon + \xi_i \ \ \forall i \\ & y_i - \mathbf{w}^T\Phi_i - b \le \varepsilon + \hat{\xi}_i \ \ \forall i \end{aligned}$$

Figure 7: Function with Narrowest Tube

Where ξ is the orthogonal distance of the data point away from the support plane.
C is the penalty coefficient, which can provide a tradeoff between simplicity and the training error[AK70]. It determines how much points within the margin the model should penalize. With higher C, the margin is small, which leads to fewer data points in the margin and a more complex model, and also may result in potential overfitting. As C is small, the training error is large but the model is simpler and robuster[Pai12].
For nonlinear cases (8, 9), we need to use map the data into higher dimension spaces. The problem becomes:

$$\min \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i + \xi_i^*$$

Figure 8: Nonlinear case

subject to

$$\begin{gathered} y_i - w^T\varphi(x_i) \le \varepsilon + \xi_i^* \quad i = 1,\dots,N \\ w^T\varphi(x_i) - y_i \le \varepsilon + \xi_i \quad i = 1,\dots,N \\ \xi_i, \xi_i^* \ge 0 \quad i = 1,\dots,N \\ w = \sum_{i=1}^{N_{SV}}(\alpha_i^* - \alpha_i)\varphi(x_i) \\ \max_{\alpha,\alpha^*} -\varepsilon\sum_{i=1}^{N_{SV}}(\alpha_i + \alpha_i^*) + \sum_{i=1}^{N_{SV}}(\alpha_i^* - \alpha_i)y_i - \frac{1}{2}\sum_{j=1}^{N_{SV}}\sum_{i=1}^{N_{SV}}(\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j)k(x_i, x_j) \\ \alpha_i, \alpha_i^* \in [0, C], i = 1,\dots,N_{SV}, \sum_{i=1}^{N_{SV}}(\alpha_i^* - \alpha_i) = 0 \\ f(x) = \sum_{i=1}^{N_{SV}}(\alpha_i^* - \alpha_i)k(x_i, x) \\ k(x_i, x) = \varphi(x_i).\varphi(x) \end{gathered}$$

Figure 9: Nonlinear case

K is the kernel, which includes two commonly used ones to solve nonlinear problems: Polynomial and Radius Basis Function(RBF) kernels. For polynomial kernel (10),

$$K(x, y) = (x^T y + 1)^d$$

Figure 10: polynomial kernel

And for RBF kernel (11),

$$k(x_n, x_m) = exp(-\gamma||x_n - x_m||_2^2)$$

Figure 11: RBF kernel

In this project, we simply use RBF kernel because its higher accuracy. The figure (12) of comparison of three different kernel is shown below.

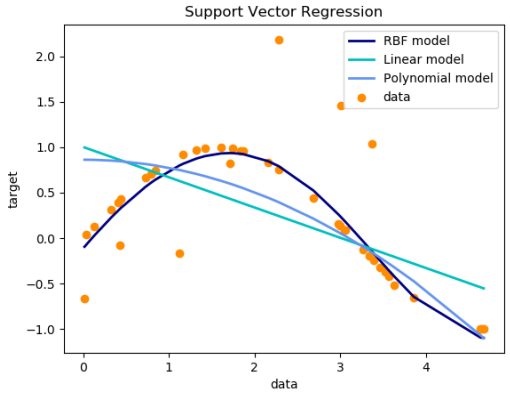

Figure 12: Comparison of Three Different Kernel

In the RBF kernel, the parameter gamma determines how far the influence of a single training example reaches. If gamma is high, them the decision boundary is more dependent on the close points, which may lead to wiggly boundary. Whereas, smaller gamma will take more weight of far points, which results in constrained straight boundary. In our project, our goal is to find the two essential parameters, C and gamma, to make the predicting results optimal. The method we use is GridSearchCV in python "sklearn" package. It searches through a parameter grid using cross-validation to get the optimal estimators, in this case, C and gamma.

3.3.5 Gradient Boosting Regressor

Gradient boosting is another technique used for regression and classification. Here we use the gradient boosting regression model to fit our data. It is an ensemble of weak prediction models based on the optimization of a customizable loss function. These weak prediction models typically consist of decision trees; in other word, the number of estimators is equivalent to the number of trees and this technique is sometimes called the gradient tree boosting.

Generic gradient boosting at the m-th step would fit a decision tree $h_m(x)$to pseudo-residuals. Let J_m be the number of its leaves. The tree partitions the input space into J_m disjoint regions $R_{1m}, \ldots, R_{J_m m}$ and predicts a constant value in each region. Using the indicator notation, the output of $h_m(x)$ for input x (16) can be written as the sum:

$$h_m(x) = \sum_{j=1}^{J_m} b_{jm} \mathbf{1}_{R_{jm}(x)} \tag{16}$$

Then the coefficients b_{jm} are multiplied by some value γ_m, chosen using line search so as to minimize the loss function, and the model (17) is updated as follows:

$$F_m(x) = F_{m-1}(x) + \gamma_m h_m(x),$$
$$\gamma_m = arg\,min \sum_{i=1}^{n} L(y_i, F_{m-1}(x_i) + \gamma h_m(x_i)) \tag{17}$$

3.3.6 XGBoost

XGBoost, an Extreme Gradient Boosting, is used for supervised learning. The model of XGBoost is the tree ensembles, a set of classification and regression tress (CART) [TC16]. Unlike the decision tree, a real score is associated with each of the leaves in the CART. We sum the prediction of multiple trees together to gain the ensemble model. Thus, the model (18) has the following form:

$$\hat{y} = \sum_{k=1}^{K} f_k(x_i), f_k \subset F \tag{18}$$

Here, K is the number of trees, f is a function in the function space F, and F is the set of all possible CARTs. Thus, the objective function (19) can be written as:

$$obj(\theta) = \sum_{i=1}^{n} l(y_i, \hat{y}_i) + \sum_{k=1}^{K} (f_k) \tag{19}$$

Hence, the goal of train boosting is to optimize the objective function by minimizing the training loss and the regularization. Here, the additive strategy is applied to learn those functions f_i, with each containing the structure of the tree and the leaf scores. We need to fix what we learned and then add a new tree at one time. Hence, the changed objective function (20) can be expressed as:

$$obj(\theta)^{(t)} = \sum_{i=1}^{n} [l(y_i, \hat{y}_i^{(t-1)} + g_i f_t(x_i) + \frac{1}{2} h_i f_t(x_i)^2] + (f_t) + constant \tag{20}$$

3.3.7 LightGBM

LightGBM is a gradient boosting framework [GK17] that uses tree-based algorithm. Unlike decision tree or random forest, LightGBM grows

tree vertically based on the leaf with max delta loss to grow. The performance is better than other level-wise algorithm due to more loss reduced. Meanwhile, LightGBM runs faster with low memory and can reach high accuracy. However, LightGBM is easier to cause overfitting so that it is usually used for dealing with large dataset. In order to gain the best fit, the relationship between number of leaves and max depth is the number of $leaves = 2^{(maxdepth)}$. In addition, min_data_in_leaf can not be set too small, which is easier to cause overfitting. For the faster speed to train, the bagging fraction and feature fraction will be turned on.

3.4 Metric Accuracy

3.4.1 Root Mean Square Error

Root mean square error (RMSE) is frequently used to measure the differences between values (sample and population values) predicted by a model or an estimator and the values actually observed. We use the following formula (21) to compute:

$$RMSE = \sqrt{\frac{\sum_{i=1}^{T}(\hat{y_i}^2 - y_i)^2)}{T}} \tag{21}$$

3.4.2 F1 Score

standard F1 metric is also used here since it can fairly judge the quality of the combination of recall and precision. F1 is calculated as following though considering equal weights in both recall and precision. The formulas (22, 23, 24) for computing F1 score is shown below:

$$Precision = \frac{TruePositive}{TruePositive + FalsePositive} \tag{22}$$

$$Recall = \frac{TruePositive}{TruePositive + FalseNegative} \tag{23}$$

$$F1 = \frac{2 * Recall * Precision}{Recall + Precision} \tag{24}$$

4 Data Exploration

In this part, we explore the dataset in 6 different aspects. They are the impact of item sequence number on deal probability, the relationship of text description and the deal probability, the relationship of the period of the advertisement is displayed and the deal probability, the influence of missing price on deal probability, the influence of price rank based on parent category or category on deal probability, and the influence of images for items on deal probability, and the influence of region and city on GRP in Russia. Data Exploration not only provides useful information behind dataset but also helps us accomplish the Feature Engineering.

4.1 Explore the Meaning of Item Sequential Number and its Relation to Deal Probability

4.1.1 Introduction

Among most other properties which either are self-explanatory (e.g. price, region, user_id, etc.) or have a detailed data description (e.g. category_name - Fine grain ad category as classified by Avito's ad model), the true meaning behind the property item_seq_number remains uncleared. It is not self-explanatory, nor can it be well understood by its description: ad sequential number for user. Therefore, interest is placed on exploring the true meaning behind the property: item_seq_number. Only after we understand what it stands for can we begin to study the relation between this property and the target variable, the deal_probability.

4.1.2 Exploration

First, the item sequential number is summarized numerically:

Count	Mean	Std	Min
1503424	743.674	5572.522	1
25%	50%	75%	Max
9	29	88	204429

Table 1: summary of item sequential number

At a glance, we can see the data are extremely sparse and left-skewed. Therefore, the data are log transformed before being plotted in a histogram for graphical data visualization.

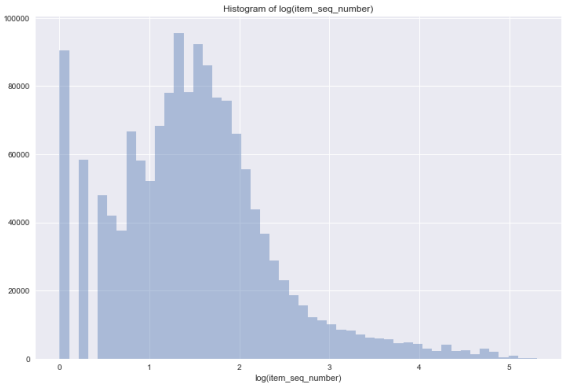

Figure 13: Histogram of item_seq_number

The histogram implies that the item sequential number is still skewed to the left after be-

ing log transformed. The skewness (0.7455) and kurtosis (1.279) of the dataset further validates this observation, that the log transformed item sequential number is left skewed and does not follow normal distribution. Therefore, it is safe to conclude that the item_seq_number is not randomly assigned by a normal distribution, nor can it be fitted using a logarithmic model. The following section tries to explore and discuss the potential definition for this parameter.

Upon investigating the dataset, the data is split into three categories according to the property "user_type". It also appears that each user type (private, company, shop) corresponds to a different range of item sequential number. Graphical and numerical results are summarized below:

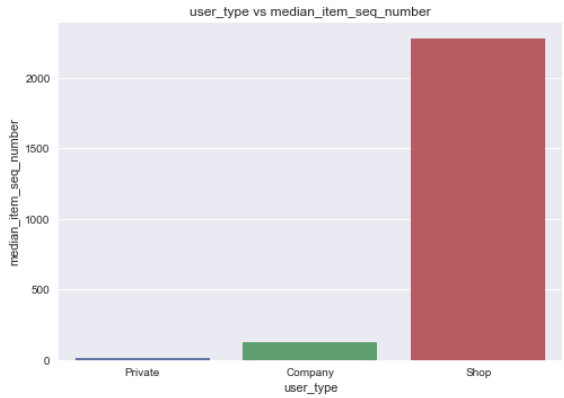

Figure 14: Bar chart of item_seq_number for different user_type

In order to prove that the item sequential number is associated with the user types, student t-test is performed among datasets, with the null hypotheses of:

$$H_{0,a} = \text{no difference in private and company}$$

$$H_{0,b} = \text{no difference in company and shop}$$

$$H_{0,c} = \text{no difference in private and shop}$$

The t-statistics and corresponding p-values are summarized below:

It is necessary to address that the p-values can never be zero - the results here are due to the lack of precision the python package is able to provide. (The three p-values obtained are all extremely small numbers). This suggests that we are fully confident to reject all three null hypotheses, that there exists significant difference among the sequential numbers of three user types. Judging from the t-statistics, we can tell that the shop generally holds the largest item sequential number, followed by company, and private the last. Given this observation, we suspect that the item sequential number represents the number of times the user has posted an advertisement, because it would be logical that a user representing a shop would post the most number of items, followed by the user that represents a company, then the individual users selling their personal items. Therefore, it is safe to postulate that the item_sequential_number stands for the number of times which a certain user posts the advertisement. In order to further explore the relation between item_seq_number and deal_probability, different bar plots are plotted to try to observe the trends. Since price is another crucial factor that may be associated with deal_probability and item_seq_number, they are plotted as well. It is necessary to note that all the price points which are greater than $100,000 are considered as the extreme values/outliers and are thus filtered out of the data. Because multiple deal probabilities and prices are corresponded to one sequential number, the means are calculated as an estimator of the values for that specific sequential number. The mean deal probability of first 20 item sequential number are plotted to see general trends. It can be seen that it follows a generally decreasing trend.

User type	Private	Company	Shop
Mean	35.32	646.21	10633.57
Median	17	125	2279
Std	116.83	3151.65	20798.24

Table 2: Summary of item_seq_number

	H_a	H_b	H_c
t-value	-200.61	-269.91	-528.39
p-value	0.0	0.0	0.0

Table 3: t-statistics and p-values of hypothesis tests

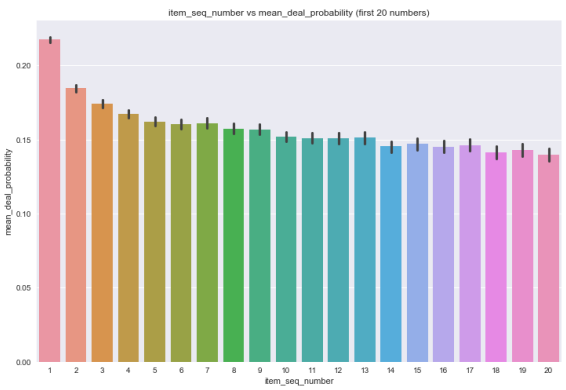

Figure 15: Bar chart for deal probability for first 20 sequential numbers

However, as 400 numbers (90% quantile) are grouped and plotted together, this decreasing trend seems to diminish. In other words, the bigger item sequential numbers tend to have more variabilities.

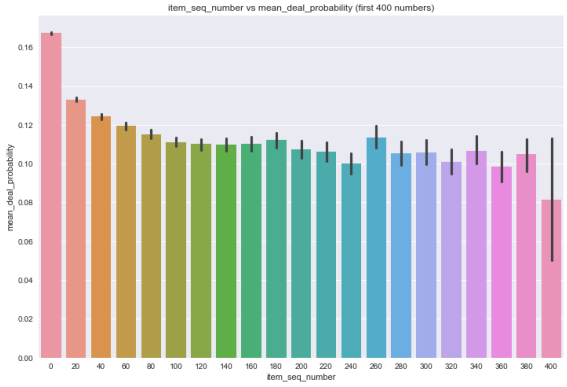

Figure 16: Bar chart for deal probability for first 400 (90%) sequential numbers

Same phenomena are observed for prices, where although there seem to be a decreasing trend for the first 20 item sequential numbers, this trend seems to diminish for bigger item sequential numbers.

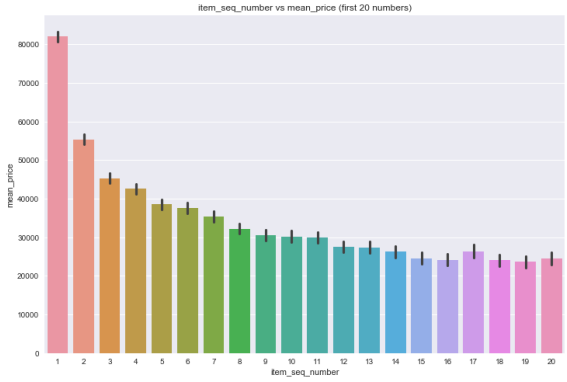

Figure 17: Bar chart for price for first 20 sequential numbers

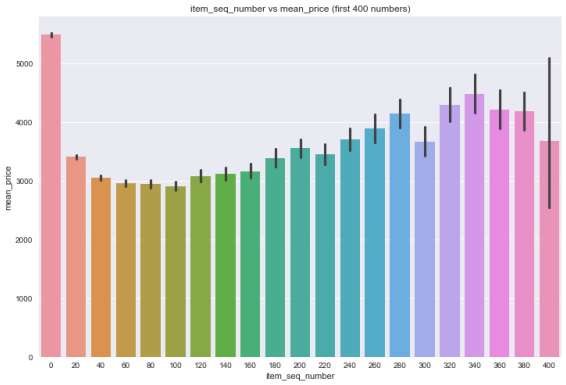

Figure 18: Bar chart for price for first 400 (90%) sequential numbers

Given the proposed definition of item_seq_number, it is speculated that the decreasing trend of deal probability (at least for the first 20 numbers) may be caused by the fact that individual users often times would post their most 'sellable' items first after they went through the trouble of registration, and subsequently post other less attractive items since they already had the account anyways. Intuitively, those individuals will be more 'confident' selling their first few items and thus set them to a higher price. Keeping this speculation in mind, however, it is less likely for the companies and shops to follow the same logic, because they mostly post items randomly and it is harder to predict a larger sample size. Therefore, the data set is further filtered to only consist of user_type 'private;' data are plotted and linear regression is performed to explore the relation between item_seq_number and deal_probability.

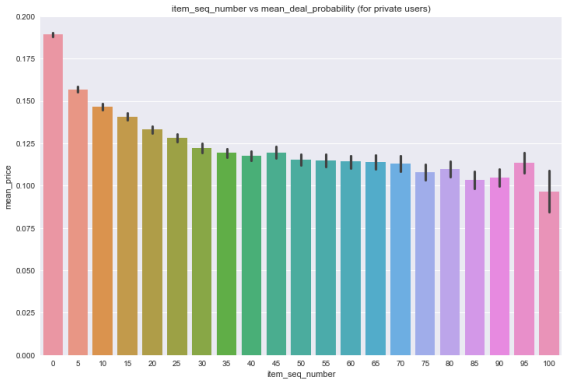

Figure 19: Bar chart for deal probability for first 100 (90%) sequential numbers for private users

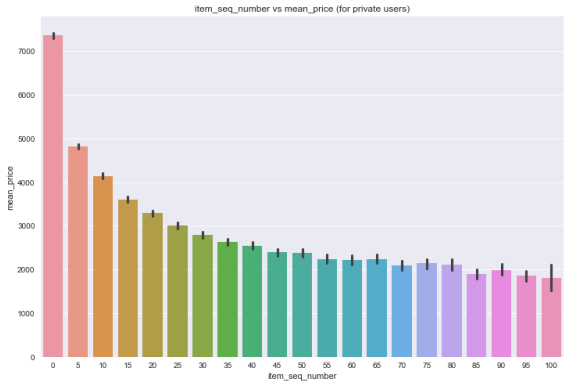

Figure 20: Bar chart for price for first 100 (90%) sequential numbers for private users

The mean deal probability and mean price of first 100 item sequential numbers for private users (90% quantile) are plotted to see general trends. As expected, the trend remains consistently decreasing for most of the data. Therefore, we can begin to perform linear regression to investigate if there exists any relation between item_seq_number and the target variable, deal_probability. The hypothesis are formed as follows:

$$H_0 = \text{no relation between item\_seq\_number and deal\_probability}$$

$$H_1 = \text{relation between item\_seq\_number and deal\_probability}$$

The results for linear regression are summarized below:

OLS Regression Results

Dep. Variable:	deal_probability	R-squared:	0.007
Model:	OLS	Adj. R-squared:	0.007
Method:	Least Squares	F-statistic:	5827.
Date:	Sun, 10 Jun 2018	Prob (F-statistic):	0.00
Time:	17:54:05	Log-Likelihood:	-96502.
No. Observations:	850563	AIC:	1.930e+05
Df Residuals:	850561	BIC:	1.930e+05
Df Model:	1		
Covariance Type:	nonrobust		

	coef	std err	t	P>|t|	[0.025	0.975]
Intercept	0.1675	0.000	393.473	0.000	0.167	0.168
item_seq_number	-0.0009	1.24e-05	-76.333	0.000	-0.001	-0.001

Figure 21: Summary of ordinary linear regression

Here we care mostly about two values, the R-squared value (since it is a univariable fit, there is no need to look at the adjusted R-squared value) and the p-value for fitted coefficient for item_seq_number (the estimator regarding the coefficient itself does not matter that much because we are not trying to fit a model purely based on item_seq_number; instead, we want to study if whatever we fitted has statistical significance). The R-squared value is 0.007 and the p-value is 0.000, which, again, indicates an extremely small number. The small p-value lets us successfully reject the null hypothesis, which states that item_seq_number has no statistical relation with deal_probability. Though counterintuitive, a small R-squared value does not contradict our conclusion. The low R-squared value indicates that the dataset is highly spread out, that there exists a high variability among the data. However, it does not disqualify this dataset from having a significant trend. Therefore, after performing linear regression, we reject the null hypothesis and conclude that the item_seq_number has significant correlation with the target variable, the deal_probability. In our future step of constructing model, the item_seq_number actually acts as a more crucial feature to the overall model.

4.2 Exploration the Relationship of Description and Deal Probability

4.2.1 Exploration of Deal Probability

We draw the histogram of deal probability to explore the distribution. Through exploring the statistical values of deal probability, we find that the mean value is 0.139 and the standard deviation is 0.26. The quantities of 25%, 50%, and 75% are 0, 0, and 0.151 respectively. From the histogram (22), we can observe that the majority of deal probability is in the range of 0 to 0.1. I compute that 64.827% of deal probability is 0, which means there is no likelihood that an advertisement can actually sold something for the majority. In order to better explore the relationship of deal probability and description, we do the binary classification for the deal probability. If the value of deal probability is larger than 0.399, then we regard the advertisement is useful and label it as 1. Otherwise, the label of the advertisement labeled as 0 if the value of deal probability is smaller than 0.399. Thus, the useful advertisement for the deal probability is classified as 1 that accounts for 13.6% and the rest classified as 0 accounts for 86.4%.

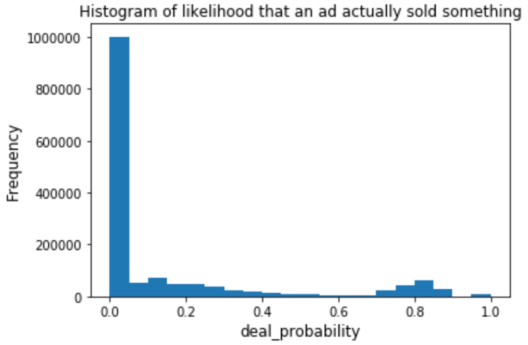

Figure 22: The Histogram of Likelihood that an Advertisement Actually Sold Something through the Comparison of Frequency vs Deal Probability

Figure 23: Wordcloud for Advertisement Description in English

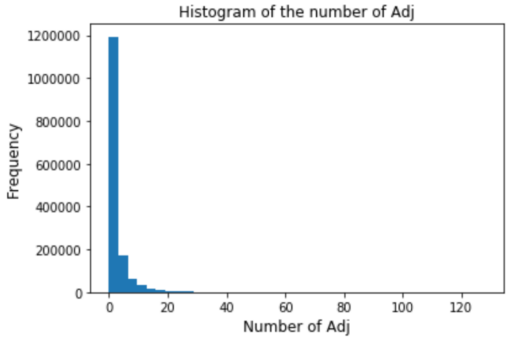

Figure 24: The Histogram of the Number of Adjective Words through the Comparison of Frequency vs The Number of Adjective Words

4.2.2 Exploration of Description

The description probably be a useful feature in influencing deal probability. We process the description by applying NLTK package to find the number of adjective words in each description. Meanwhile, we count each description length and compute the ratio of adjective words in each description. Since all descriptions are in Russian, which are hard for us to understand, we translate them into English and draw the word cloud to show the top 200 hot key words in the description. From the word cloud (23), We can see that these top key words are adjective words that are worthy of further exploration. In order to improve the efficiency of the counting adjective words in the description, we need to do the natural language process. First of all, we do the word tokenization and convert all words into lower cases. After that, we remove all stop words. Lastly, we count the number of adjective words including the comparative level and the most advanced in each description. We draw the histogram (24) of the number of adjective words in each description. Through the statistical computation, we find that the max value is 128, mean value is 2.533, standard deviation is 4.037, and the quantities of 25%, 50%, and 75% are 0,1, and 3 respectively. Similarly, we also draw the histogram (25) of the length of the description and do the statistical computation. we find that the max value is 3212, mean value is 178, standard deviation is 288, and the quantities of 25%, 50%, and 75% are 41,87, and 186 respectively.

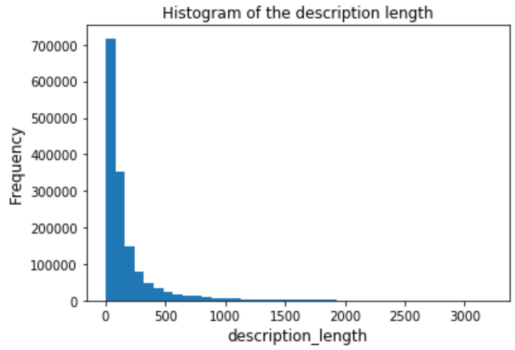

Figure 25: The Histogram of the Description Length through the Comparison of Frequency vs The Description Length

4.2.3 Null Hypothesis for the Relationship of the Description and Deal Probability

We first want to explore whether the number of adjective words in each description influences the deal probability. Through drawing the Q-Q plot of the number of adjective words count in class 1 vs the number of adjective words count in class 0 (26), we see that they are obviously not normal distribution. Thus, we apply mann-whitney test to compute the p-value. Thus, we can have the following null hypothesis:

H0: The two populations are equal

H1: The two populations are not equal

We apply two-sided test to gain the p-value, which is 0.207. Since the p-value is larger than 0.05, we can not reject the null hypothesis. As we can see, the count of adjective words in the description does not have a big impact on the deal probability due to the two populations are equal. Similarly, we explore whether the length of the description influences the deal probability.Through drawing Q-Q plot (27), we also find that they are not normal distribution. Thus,we still use mann-whitney test to compute the p-value and have the same null hypothesis.After computation, we gain the p-value of 8.37e-07 that is much smaller than the 0.05. Thus, we can reject the null hypothesis that two populations are not equal. Hence, we can see that the length of the description influences the deal probability. Lastly, we explore whether the percentage of the adjective words in the description has an effect on the deal probability. The Q-Q plot (28) shows that they are not normal distribution. Thus, we can still use mann-whitney test to compute the p-value and have the same null hypothesis as before. We gain the p-value of 0.126 that is much bigger than the 0.05. Thus, we can not reject the null hypothesis that two populations are equal. Hence, we can see that the percentage of the adjective words in the description not influences the deal probability.

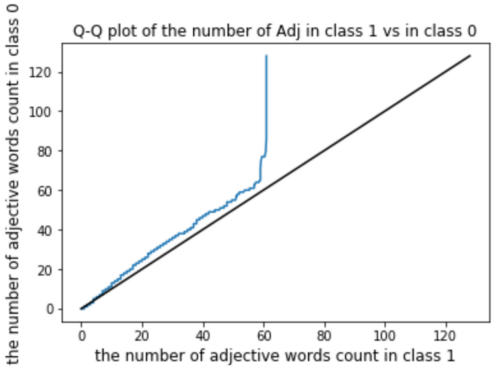

Figure 26: Q-Q Plot of the Number of Adjective Words in Class 1 vs in Class 0

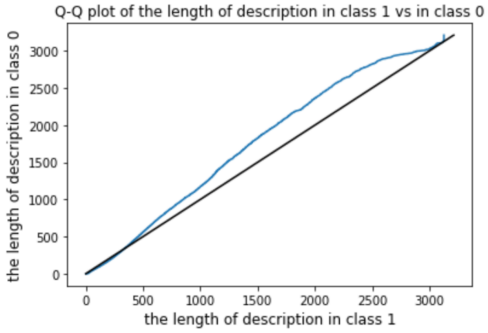

Figure 27: Q-Q Plot of the Length of Description Words in Class 1 vs in Class 0

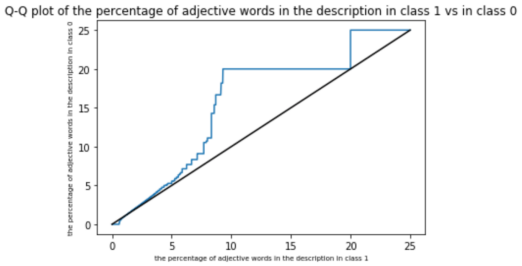

Figure 28: Q-Q Plot of the Percentage of Adjective Words in the Description Words in Class 1 vs in Class 0

4.2.4 Conclusion for the Relationship of Description and Deal Probability

As three different null hypotheses shown above, we can see that the length of the description influences the deal probability. While, the count of adjective words in the description and the percentage of the adjective words in the description

do not have a big impact on the deal probability due to the p-value larger than 0.05. Thus, we will apply the length of the description in feature engineering.

4.3 Explore the Relationship Between Periods of Advertisements Displayed and Deal Probability

4.3.1 Exploration of Deal Probability and Displayed Days

Theoretically, the longer the time interval an advertisement is displayed on website, the more people will see it, and then larger the probability the good will be sold. But does shorter period mean that the good is sold out quickly (large deal probability)? Thus, we are interested in seeking for the relationship between the length of the period during which an advertisement is displayed and the deal probability. We explore the potential relation from two perspectives: grouping the period and testing the difference of the two period groups; grouping deal probability and determining how significant the difference of the two groups is. The reason why we simply make the periods or deal probability data into two groups respectively is that the histogram (29, 30) of the two features are all highly skewed. Then basically what are doing is to discover the difference of extreme large groups versus the others. The null hypothesis is that there is no significant difference between two groups of periods or deal probability. We will use independent 2 sample t-test to test them.

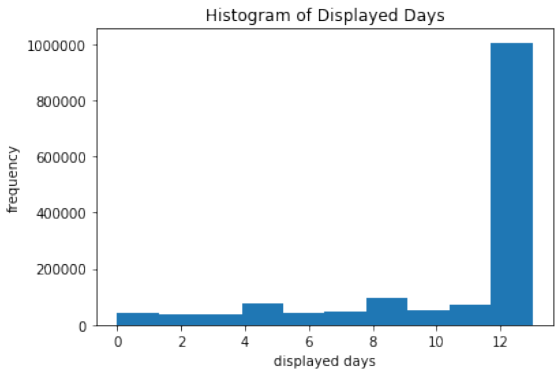

Figure 29: Histogram of Deal Probability (periods)

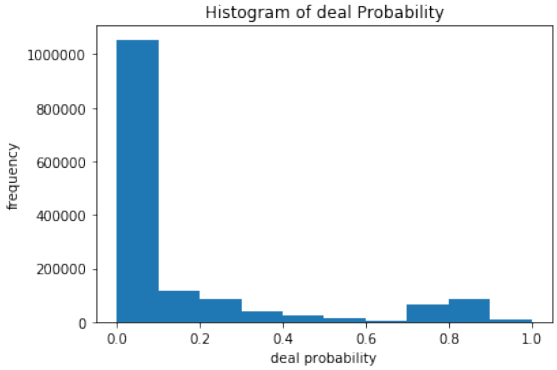

Figure 30: Histogram of Displayed Days

4.3.2 Data Processing

Since the periods of data in $train_active.csv$ are the same as those in *train.csv*, then we can combine *train.csv* and $periods_train.csv$ to get intuitive connection between deal probability and time periods. Given the starting and ending date, the first thing we do is calculating time difference using python package "datetime" for each item id in $periods_train.csv$. Since one id may appear several times in the table, then we combine all the periods for identical ids. To combine *train.csv* and manipulated periods, we use "merge" function to combine them using "$item_id$" as the index column. Since there are some extra supplemental data in period dataset, we ignore the extra item ids and keep all the item ids in *train.csv*. The combined table (31) looks like:

image	image_top_1	deal_probability	diff
:1caf64c...	1008.0	0.12789	13
7142700...	692.0	0.00000	13
db4b48a...	3032.0	0.43177	5
a1c8413...	796.0	0.80323	8
)16fd0a2...	2264.0	0.20797	13

Figure 31: Screenshot of part of the Combined Table, where "diff" is the Time Difference (periods).

From below scatter plot (32), we cannot find any special relation between the two features. So, we decide to explore the relation in the following two parts.

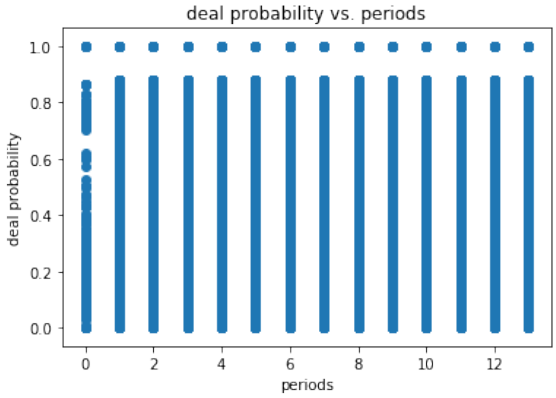

Figure 32: Scatter Plot of Deal Probability vs. Periods

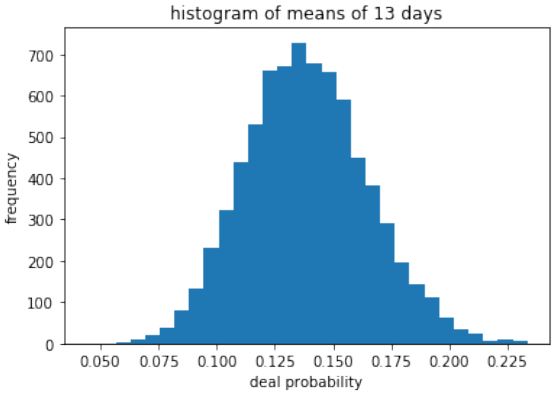

Figure 33: Histograms of Means of 13 Days

4.3.3 Explore the Probability Difference Between Long and Short Displayed Days Groups

From the histogram (30), it is obvious that a large amount of data gathering at 13 days. Hence, we make the data with displayed days less than 13 days into a group, and another group of data with displayed days exactly 13 days. There are 752358 data rows in long period group, and 750300 data rows (exclude 0 days) in shorter period group. To make things easier, we randomly select 750300 data rows from long period group to make it equal to the number of shorter period group. We apply hypothesis test to see if there is significant probability difference between the two groups. Our null hypothesis is that there is no significant difference between the group with long period and the one with shorter periods. We determine to use t-test to show the comparison. To perform an independent 2 sample t-test, we need the distribution of the two groups to be normal, and to have the same variance. To normalize the probability data for each group, according to Central Limit Theorem, we randomly extract 7503 subsets with 100 data per group, and distribution of the means of all the subsets will be normal distribution. To verify that they are following normal distribution, we draw the histograms (33, 34) and Q-Q plots (35, 36) below:

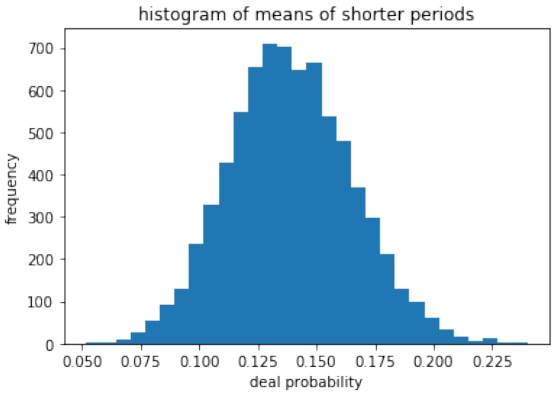

Figure 34: Histograms of Means of Shorter Periods

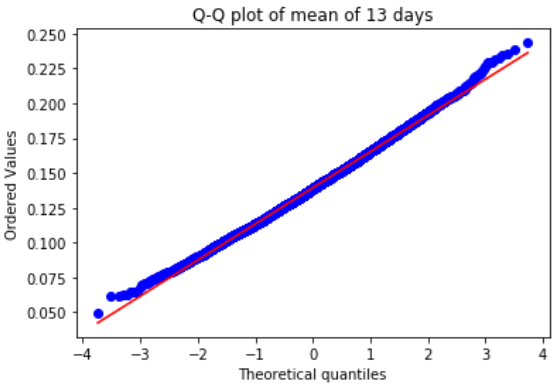

Figure 35: Q-Q Plot of Long Period Dataset

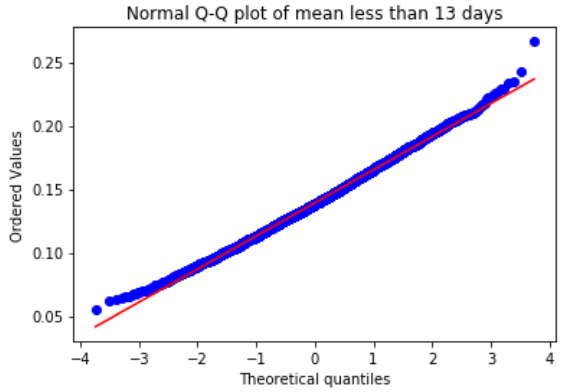

Figure 36: Q-Q Plot of Short Period Dataset

From above Q-Q plots (35, 36), we can clearly see that the two distributions of probability means are all normal. Moreover, the variance of the means of the long period dataset is 0.000673, and the variance of the means of the shorter period dataset is 0.000680 which is close to 0.000673. So, two sample t-test (37) is eligible to be applied here:

```
Ttest_indResult(statistic=0.3880266174203556, pvalue=0.6980018408578844)
```

Figure 37: t-test Result

The result shows that p-value is 0.698, which is way larger than the significance level 0.05. So, we fail to reject the null hypothesis. There is no significant probability difference between long and short displayed period groups.

4.3.4 Explore the Displayed Period Difference Between High and Low Probability Groups

After examining the probability discrepancy, we would like to do it inversely—cut the probability into two groups and explore the period difference of the two groups. Observing from the histogram 4.2.1, most of the data gathering between 0 to 0.2, hence we make the data with probability greater than 0.2 into a group and those with probability less than 0.2 into a group (exclude 0.0). We randomly extract 206200 data rows out from each dataset to make the size equal. Our null hypothesis is that there is no significant difference between the groups with high and low probabilities. Still, we use t-test to perform the comparison. Similarly, we need to normalize the period distribution to conduct t-test. We randomly extract 2062 subsets from each group with 100 data per subset, the distribution of the 2062 means should be normal. The histograms (38, 40) and Q-Q plots (40, 41) show below:

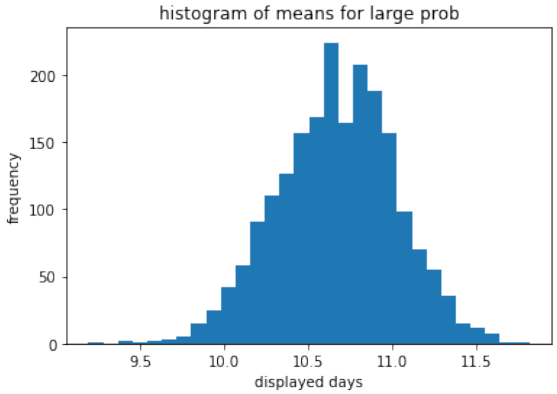

Figure 38: Histograms of Means for Larger Probability

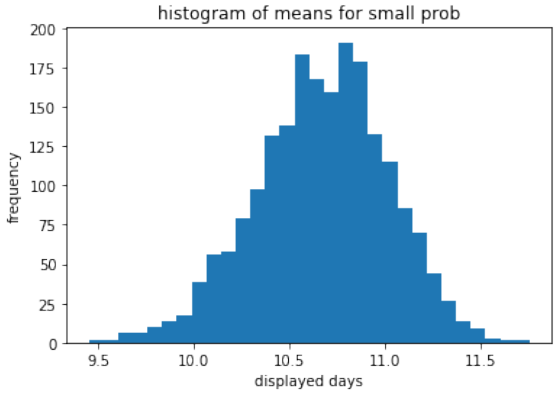

Figure 39: Histograms of Means for Small Probability

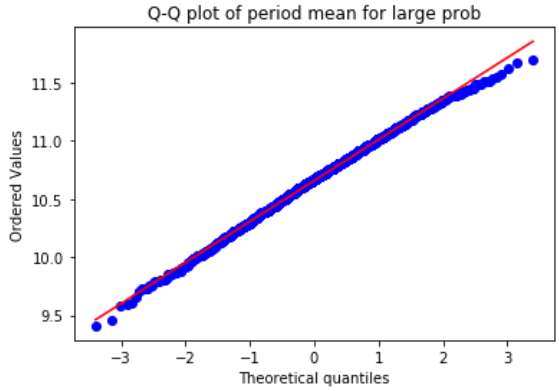

Figure 40: Q-Q Plot of Period Means for Large Probability Group

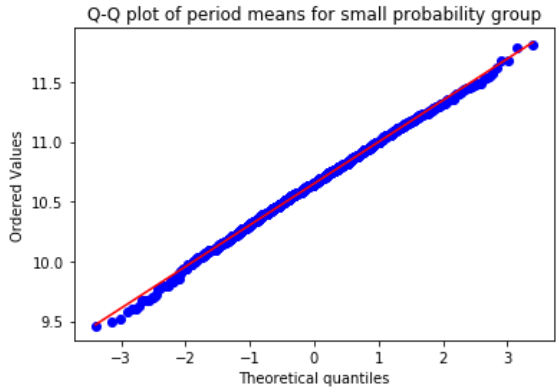

Figure 41: Q-Q Plot of Period Means for Small Probability Group

From the Q-Q plots (40, 41) above, we can clearly see that the means of the two groups are following normal distribution. The variance of small probability group is 0.1235 and that of large probability group is 0.1205, so they are close enough. Then we are safe to use independent two sample t-test:

```
Ttest_indResult(statistic=-0.47067678681331243, pvalue=0.6378967123425947)
```

Figure 42: t-test Result

The t-test result (42) shows that p-value is 0.638, which is larger than significance level 0.05, which means we also fail to reject this null hypothesis. There is no significant period difference between the two datasets with different probabilities. In conclusion, we perform t-test to examine if there is significant probability difference between long and short period groups, and whether high or low probability show difference in item displayed days. The null hypothesis are all fail to be rejected, thus we can conclude that the advertisement displayed periods will not lead to effects on items' deal probability.

4.4 Explore the Missing Price and Non-missing Price Groups' Difference on Deal Probability

4.4.1 Introduction

Price is potentially one of the most important feature which affects deal probability. However, price is NA, which indicate data missing, for around 6 percents of the total data. We want to take a look at if price missing or not has impact on deal probability.

Furthermore, the result of this section could give us insight on whether to drop this part of data or simulate price to be used in later sections' model train.

4.4.2 Graphical overview and data processing

Before investigating the impact of price missing, we suggested an assumption that the deal probability distribution and price distribution for different categories are different.

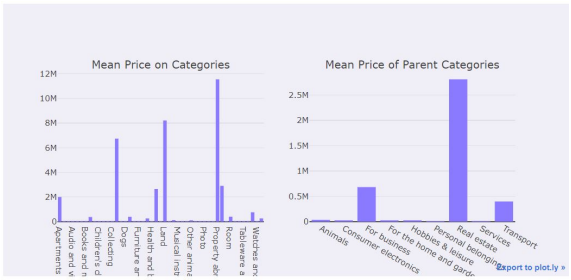

Figure 43: Histogram of mean price

From the figure of mean price of different categories (43), we can see there's explicit difference in the means of price across different categories.

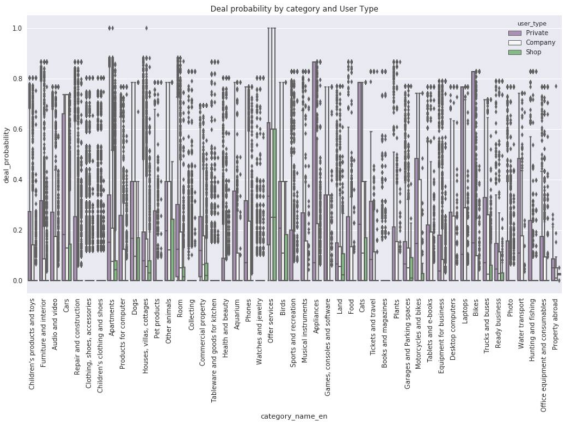

Figure 44: Box plot of deal probability

Then the detailed box-plot (44) represent that the deal probability among different categories.

The deal probability varies a lot across categories.

Thus we gain support of our assumption.

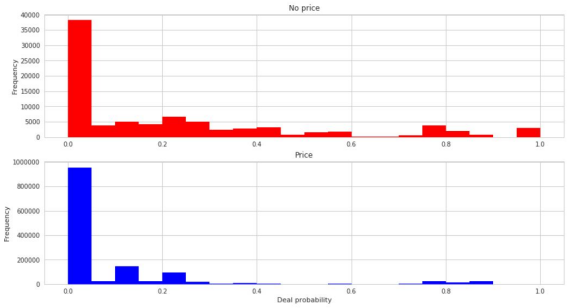

Figure 45: Histogram of deal probability

Then we perform graphical analysis of deal probability between price missing and non price missing groups. As the two histograms (45) suggest, there's a difference between the two groups. We would further test it in later section.

Also, based on previous assumption, we suggest a potential answer to the difference in deal probability between the two groups as the difference in category distribution could be the main factor.

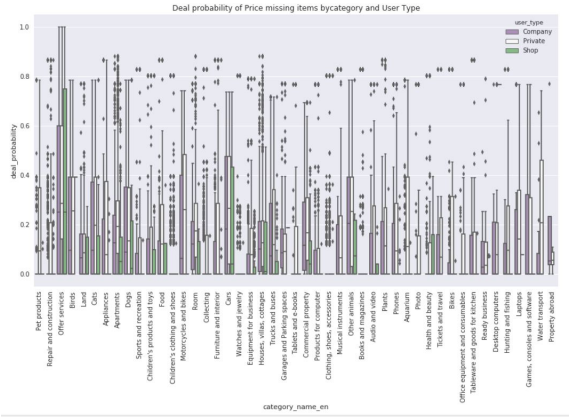

Figure 46: Box plot of deal probability

From the figure (46) represent that the deal probability among different categories within the population without price. We can see it looks not much different from previous figure. And we would test our intuition in later section.

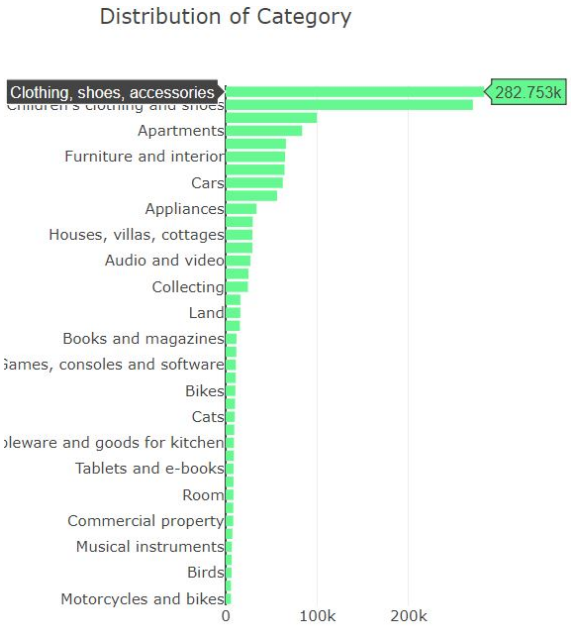

Figure 47: Histogram of categories of all items

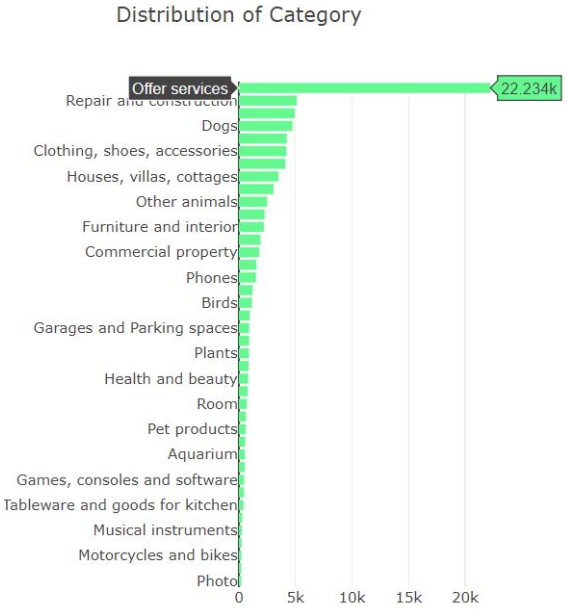

Figure 48: Histogram of categories of price missing items

From the figures (47, 48), we can see the category distribution of the two group are significantly different. As clothing and children clothing made up most of the original population, offer services made up most of the price missing group.

Thus in later section we want to test if such difference in category distribution leads to the difference in deal probability, with price missing having no significant impact.

4.4.3 General Hypothesis on No Impact of NA Price on Deal Probability

As previous sections explore, we suggest a general hypothesis that NA price has no impact on deal probability, holding else constant. Difference on deal probability between price missing group and non price missing group is due to different categories.

To verify this hypothesis, we conduct three hypothesis tests.

4.4.4 Hypothesis Test of Missing Price's Impact on Deal Probability with Items of Various Categories

It is the first step of our investigation, as we haven't exclude the influence of categories and other potential correlated parameters, we want to test if there's a difference in general between group of price missing items and the opposite group. Also, since the population is not normally distributed, instead of using t-test, we used Wilcoxon signed-rank test which is non parametric and could be applied to our data.

H0: difference between the price missing and non price missing items follows a symmetric distribution around zero.

H1: difference between the price missing and non price missing items does not follow a symmetric distribution around zero.

4.4.5 Hypothesis Test of Category "offer services" 's Impact on Deal Probability

In data overview, we found most items with price missing are in category of "offer services". Then we want to test if items in this category and other categories has significant difference in deal probability.

H0: difference between items with category of offer services and other categories follows a symmetric distribution around zero.

H1: difference between items with category of offer services and other categories does not follow a symmetric distribution around zero.

We applied the two-sided test to gain w-statistic and p-value, which are 5264742.5 and 0.0 (in float 64) respectively, we reject the null hypothesis since the p-value for two side test is significantly small, suggesting the difference in mean is not zero.

Thus items of category "offer services" indeed differ from the rest in deal probability

4.4.6 Hypothesis Test of Missing Price's Impact on Deal Probability with Items of Same Category "offer services"

To continue our exploration, we control the impact of category in this step. As previous test validated that items with category of "offer services" differ from others in deal probability, we want to check if price missing has impact on deal probability with category holding constant. Then we want to compare if within that category of "offer services", price missing's impact on deal probability is statistically significant.

H0: difference between the price missing and non price missing items with category of offer services follows a symmetric distribution around zero.

H1: difference between the price missing and non price missing items with category of offer services does not follow a symmetric distribution around zero.

We applied the two-sided test to gain w-statistic and p-value, which are 75814.5 and 0.8031310504564 respectively, we fail to reject the null hypothesis since the p-value for two side test is greater than the threshold of 0.05, suggesting the difference in mean is zero.

Thus it validated our guess in graphical analysis, as after controlling categories, there exist no difference of deal probability between price missing and non price missing items with category of "offer services".

4.4.7 Conclusion on Missing Price's Impact on Deal Probability

The three hypothesis tests' results verified our general hypothesis. Holding else constant, price missing has no significant impact on deal probability.

The potential difference in deal probability between price missing group and non price missing group is most likely due to their different distribution of categories as deal probability between different categories differ.

This result also provides us the ease of ignore the effect of NA in prices on deal probability. Thus in further investigations and model building, we could fill price NA items with mean of price of that item's category.

4.5 Explore the Relationship Between Price Rank within Category and Deal Probability

4.5.1 Introduction

Price could be one of the most important parameters in our data. Intuitively, we want to explore its relationship with deal probability. But as previous section revealed that deal probability and prices differ across different categories, making price itself a less explanatory parameter for deal probability.

Thus we introduced a new parameter, price rank, which is converted from an item's price ranking in ascending order within its category. Price rank is a float number, representing the proportion of items with same category more expensive than it. An item with price rank of 0.01, for example, means only 1 percents of items with same category has a higher price than it.

With that new parameter, we explore its relationship with deal probability in a more explanatory approach.

Plus, we introduce a categorical parameter, price class, which is the 5 percent interval an item's price rank falls into on [0,1]. It has 20 values, naming, [0,0.05), [0.05,0.10) .... [0.95, 1]. Given that we are able to split items into different groups based on its price rank.

Then we could check if there's a difference on deal probability among any of these groups.

4.5.2 Data Processing and Graphical Overview

We firstly eliminated data with missing values in our test since we need to rank our data based on price.

As previous section's figures (44, 46) suggested, deal probability and price varies across different categories. Thus we want to control category constant as we perform rank.

Then we performed the calculation which gives each item a price rank, representing the proportion of items with same category more expensive than it, and a price class based on which of the 20 intervals on [0,1] its price rank falls into.

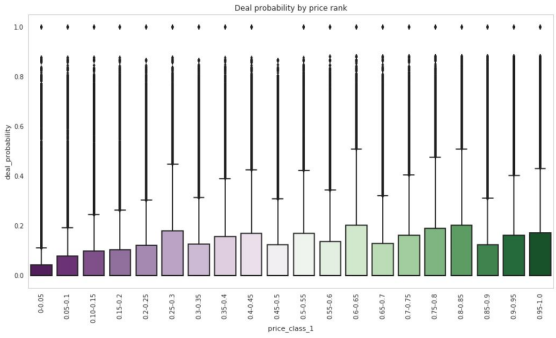

Figure 49: Box plot of deal probability of different price rank class

After that we generated a box plot (49) on deal probability of different price rank class. From it we can see some trend like, "0-0.05" group tend to have a lower deal probability than the rest, with "0.30-0.35" group have high deal probability.

But this figure (49) cannot give numerical proof to our intuition, and we would perform hypothesis test in later section.

4.5.3 Hypothesis Test of Deal Probability of Items with Different Price Rank Class

Since the non-normal distribution of data we could not use ANOVA to test if the samples originated from same distribution, indicating the groups we assumed have no impact on it. Instead, we use Kruskal–Wallis test, which is immune to non-normal distribution, to test if the samples originated from same distribution.

H0: Deal probability of samples with different price class originated from same distribution.

H1: Deal probability of samples of at least one price class originated from a different distribution than the rest.

We applied the Kruskal-Wallis test and get test statistic of 132.83231437075716 and p-value of 3.0994401241401968e-24. Since p-value is smaller than the threshold 0.05, we reject the null hypothesis that deal probability of samples with different price class originated from same distribution.

4.5.4 Conclusion on the Relationship

The graphical analysis and hypothesis test result are consistent with each other. Some group like 0-0.05 and 0.25-0.30 show noticeable difference in the graph and since we reject the null hypothesis, there's at least one group show different distribution of deal probability.

Thus price rank should be considered as a statistically significant factor for deal probability.

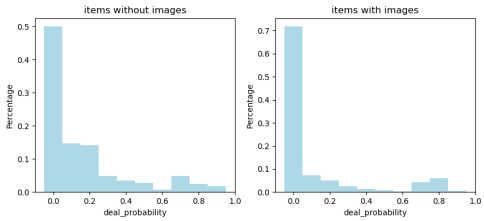

Figure 50: Histogram of deal probability for items with and without images

This give us a new parameter to be used in later sections' models in exploring the factors of deal probability.

4.6 Explore the Relationship Between Image and Deal Probability

In this section we explore whether an item with an image has impact on deal probability. In second-hand products shopping, the detailed description of the product including an image help customers make a better decision whether the item fits his or her expectation. So the null hypothesis to be tested is that there is no significant differences in deal probability between items with images and items without images.The histogram of deal probability of items with and without images are plot in Fig. 50 Whether an item has an image can be retrieved from the feature named 'image'. 'Null' in 'image' category indicates the item has no images. We calculate the mean and variance of the items with and without images, which is listed as follows Table.(4):

	with image	without image
mean	0.13493	0.19098
variance	0.061953	0.06787
count	1390836	112588

Table 4: Comparison between items with images and without images

A student t-test can be used to test our hypothesis. originally, t-test is applied to random variables with normal distribution. However, when sample size is large, which is true in this case, the t-test is valid. Assume we divide the samples into many equal sized blocks and take the mean. Because of the central limit theorem, the distribution of these means, in repeated sampling, converges to a normal distribution, irrespective of the distribution of in the population. We perform a t-test and the calculated the p-value is almost 0, indicating we should reject the null hypothesis. The mean deal probability for items without images is larger than those with images. This is an interesting observation because it contradicts our intuition that items with images should make themselves more reliable and easier to sell.

To further investigate this problem, we compare the mean deal probability within different categories. We group our data by 'parent_category_name' and divide the data into 9 different parent category. Within each parent category, we calculate the mean and variance of deal probability for items with and without images respectively. Two sample t-test is then performed to find out if there is a significant difference between deal probability for each of the parent category. Results are show in Table .(5).

We noticed that in the parent category of 'Personal things', 'For home and cottages', 'Consumer electronics', 'The property', 'Transport' and 'The services', the p-value is close to 0 and we can reject the null hypothesis. In the 'The property' category, the deal probability is higher for items with images. For other categories, the deal probability is always lower for items with images. For 'Hobbies and Recreation' and 'For business' it's safe to accept the null hypothesis and conclude that images have no effects on those items. For 'Animal' category, the p value is just above 0.05 and we don't have a good idea on the effect of images on deal probability.

Our conclusion in this section is that whether items with images affect the deal probability is category dependent. The influence can be positive, negative or neutral for different categories.

4.7 Explore the Relationship Between GRP of Regions in Russia and Deal Probability

4.7.1 Introduction

Normally [U+FF0C] the higher the Gross Regional Product (GRP) of the region, the purchasing power of the people will be greater. That means, the deal probability of the goods online will tend to be higher. We are wondering if the GRP of the region is a factor which influences deal probability. Thus, we generate a study about this factor. We grouped the column region and find the mean of the deal probability each of those 28 regions. We collected the 2008 GRP data we find on Wikipedia. Then, we divided those 28 regions into two groups, the high and low GRP groups.

	count	mean dp with image	mean dp without image	p value
Personal things	697623	0.0757	0.0899	0
For home and cottages	178823	0.1819	0.1493	0
Consumer electronics	173008	0.1774	0.1349	0
The property	153190	0.1297	0.1681	0
Hobbies and Recreation	86011	0.1239	0.1179	0.2141
Transport	79839	0.2693	0.1769	0
The services	64385	0.4314	0.3366	0
Animals	52470	0.2369	0.2309	0.0818
For business	18075	0.1111	0.1104	0.9219

Table 5: mean deal probability for items with and without images within different categories

4.7.2 Exploration

First, we visualized the size of the item numbers in each cities on the map of the Russia. We can see that some regions (51) are more popular and denser than other regions.

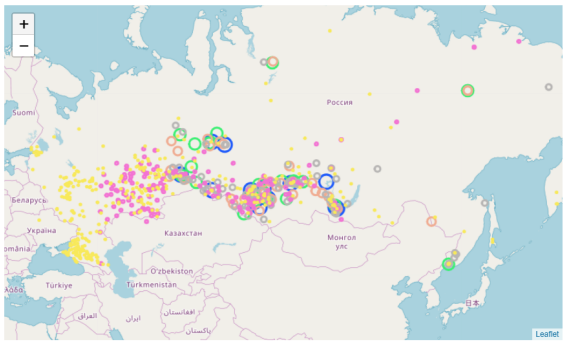

Figure 51: Visualization of the size of the total items in each city of Russia

Then, we collected the GRP values of the 28 regions in a table (52).

```
{'Алтайский край': 275084,
 'Башкортостан': 749481,
 'Белгородская область': 273213,
 'Владимирская область': 0,
 'Волгоградская область': 431748,
 'Воронежская область': 268842,
 'Иркутская область': 456706,
 'Калининградская область': 176257,
 'Кемеровская область': 575942,
 'Краснодарский край': 808704,
 'Красноярский край': 740233,
 'Нижегородская область': 597988,
 'Новосибирская область': 460087,
 'Омская область': 352938,
 'Оренбургская область': 425045,
 'Пермский край': 609230,
 'Ростовская область': 576386,
 'Самарская область': 706514,
 'Саратовская область': 330564,
 'Свердловская область': 944408,
 'Ставропольский край': 251726,
 'Татарстан': 923206,
 'Тульская область': 230606,
 'Тюменская область': 3143607,
 'Удмуртия': 215937,
 'Ханты-Мансийский АО': 0,
 'Челябинская область': 664974,
 'Ярославская область': 319071}
```

Figure 52: The GRP of 28 regions

Next, we analyze the two distribution (53,54) of the deal probability in the low and high GRP groups.

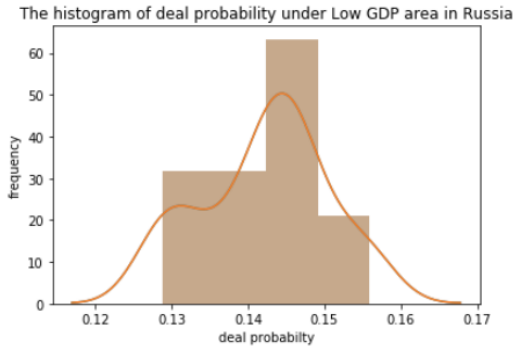

Figure 53: The distribution of the deal probability in low GRP group

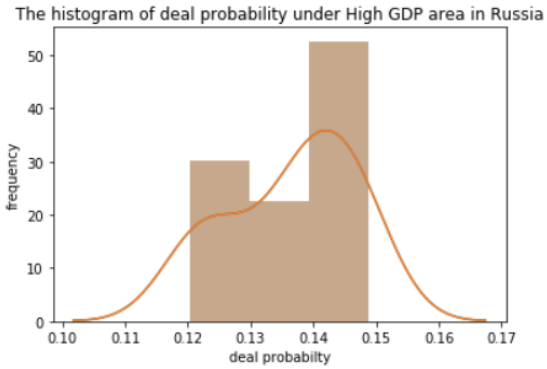

Figure 54: The distribution of the deal probability in high GRP group

The null hypothesis we set here is that:
H0: the deal probability of the low and high GRP groups are equal
H1: the deal probability of the low and high GRP groups are not equal

We can see from the two graphs that the mean of the this two groups are different. However, we want to study whether the difference is statistically significant. Then we generate the t-test. And we found out the p-value equals 0.07389919, which is greater than the 5 percent significance level. Thus, we can't reject the null hypothesis. Thus, we accepted that the Region GRP doen't influence the deal probability.

5 Feature Engineering

In this part, we explain how we do the data pre-processing and data transformation from the given dataset and our exploration. We first connect training data and test data because we need to process both of them. Next, the new dataframe becomes the size of 2,011,862 x 18. For the image_top_1 feature, we fill the missing value with -999 based on the exploration of this feature range. Similarly, we fill the missing value for features param_1, param_2, param_3, description, tilte with NaN due to string type. For the price feature, we take log of them and plus 0.001 so as to make them more dense and avoid 0. Meanwhile, we fill the missing value based on the mean. For the activation date, we also use datetime to grab the detailed day of the month, day of the week, and week of the year. More importantly, we also build additional features based on the relation of region, city, parent_category_name, category_name, user_type, image_top_1, item_seq_number, day of the month, day of the week, and week of the year and deal probability and price to gain the mean and standard deviation through groupby method. For the title and description, we apply natural language processing such as TF-IDF to gain 200 max features from description, and 100 max features from title under the consideration of stopwords. In addition, we count title length, description length, and percentage of unique words in the description after cleaning words by removing special characters. After that, all the category features are applied dummy variable to be transferred into machine code (one-hot-encoding). We also build the extra features from the data exploration. We compute the price rank score, the range from zero to one based on the parent category. In addition, we compute the difference displayed date from the provided periods tables. Lastly, we assign the item with images one and others be zero. After finish building the additional features and processing the dataset, we totally have around 350 features in training machine models.

6 Experiment

6.1 Multiple Regression

6.1.1 General Hypothesis and idea

Given the data with tens of columns as parameter, we want to explore if we can fit our data of deal probability into a multiple regression model with some, if any, of these parameters as coefficients.

The general hypothesis is, when fitting deal probability and these parameters in to a multiple regression model, at least some of these parameters' coefficients are not zero, indicating some of these parameters has linear impact on deal probability.

The further goal is to find the best multiple regression model with some of these parameters, using RMSE as the evaluation standard. To achieve that, we perform feature selection pro-

cedure with p-value approach.

Besides, due to the large data set and number of parameters, we may using only a proportion of data set during the feature selection procedure subject to the computing power constraint. However, as we select fewer parameters during feature selection, we could increase the proportion we use to achieve more precise result.

6.1.2 Multiple regression at first look

Before selecting any feature, We fit our data in to a multiple regression using most of parameters we have with a smaller proportion due to computing power limit55.

For categorical parameters, we create dummy variables to represent each category.

parameter name	coef estimator
category_name_deal_probability_mean	9.759600e-01
image_y_or_n	3.212433e-02
price_rank1	5.754584e-03
description_num_unique_words	7.478816e-04
day_of_week	6.894398e-04
title_unique_percentage	3.714938e-04
title_length	3.238590e-04
category_name	2.901878e-04
param_1	5.532344e-05
diff	3.366421e-05
city	1.270164e-07
parent_category_name_price_mean	8.514010e-08
region_price_mean	8.216618e-09
category_name_price_mean	2.213604e-09
image_top_1_price_std	6.539330e-10
city_price_std	4.972655e-10
user_type_price_std	1.697606e-10
item_seq_number_price_mean	2.005020e-12
item_seq_number_price_std	1.627241e-12
param_2	0.000000e+00
user_type	-0.000000e+00
week_of_year	-0.000000e+00
description_unique_percentage	0.000000e+00
category_name_deal_probability_std	0.000000e+00
region_price_std	-3.704804e-11
category_name_price_std	-4.483112e-11
parent_category_name_price_std	-8.406788e-10
image_top_1_price_mean	-8.627357e-09
city_price_mean	-1.879546e-08
user_type_price_mean	-8.197137e-08
item_seq_number	-1.407759e-07
image_top_1	-5.257359e-06
param_3	-9.976949e-06
region	-2.164677e-04
day_of_month	-3.168933e-04
description_length	-6.643097e-04
parent_category_name	-1.772135e-03
price	-5.223073e-03

Figure 55: multiple regression with 38 parameters

The resulting RMSE is around 0.267.

Though we could avoid omitted variable bias in our coefficients, we can not rule out colinearity. And it would affect our result of coefficients estimators.

6.1.3 Feature Selection

In order to achieve lowest RMSE as well as avoid over fitting. We want to select some of all the parameters we have in our regression model.

For each feature, there's an implicit null hypothesis that its coefficient in multiple regression is equal to zero, with alternative hypothesis that it is not zero.

We want to test the hypothesizes, and if we fail to reject a null hypothesis, we exclude the corresponding parameter from the multiple regression model in next round.

The procedure is as follow: For each fit of multiple regression model, check the t-statistic and p-value for each parameter. If the p-value is significantly large, greater than the threshold of 0.01, we fail to reject the null hypothesis and remove the corresponding parameter in next round of fit of multiple regression. If the p-value is significantly small, smaller than the threshold of 0.01, we reject the null hypothesis and keep the corresponding parameter in next round of fit of multiple regression. During the

To avoid multicolinearity, we also used Lasso regression as an aid in feature selection. In parallel to each fit of multiple regression, we perform a fit to lasso regression using the same parameters as input. Lasso regression output would automatically omit the parameters that are perfectly correlated to some others. If we found such parameter, we would drop it from both models and perform the fit again.

And we check the coefficients of lasso regression to give us a cross validation on multiple regression. If lasso regression show a parameter's coefficient is significantly close to 0, that it is less than 1e-10, we would create two multiple regression model with and without that parameter, and choose the model with lower RMSE.

6.1.4 Final Multiple Regression Model and Conclusion on General Hypothesis

After feature selection procedure, we achieved our final model of multiple regression in table **??**.

parameter name	coef estimator
category_name_deal_probability_mean	9.906844e-01
image_y_or_n	2.713823e-02
price_rank1	1.202478e-02
title_length	1.997626e-04
category_name	1.943120e-04
param_1	4.308838e-05
parent_category_name_price_mean	1.742194e-08
region_price_mean	-5.988090e-09
user_type_price_mean	-9.744843e-08
item_seq_number	-1.846966e-07
image_top_1	-6.393068e-06
param_3	-1.360783e-05
param_2	-1.780523e-05
description_num_unique_words	-9.218076e-05
parent_category_name	-1.848458e-03
price	-4.094080e-03

Figure 56: multiple regression with 16 parameter

Using this model to predict deal probability of the validation set, and compare with the true deal probability on Kaggle, we achieved a RMSE of 0.2465. (57) And it is our best score using multiple regression.

Figure 57: rmse score of multiple regression

And in figure56 we can see the descending order of coefficient value. With most parameters as categorical, we can use this order as a reference of feature importance. It coincides with many of our previous founding in previous sections, as category name, whether include a image and price rank have greater coefficient than others.

We can also justify our final hypothesis that feature of our choice have impact on deal probability as the p-values shown in the table 58 suggest no parameter has a p-value greater than our threshold 0.01.

6.2 Random Forest

In this section, we trained the random forest model and made deal probability prediction. By tuning various parameters, we got an deal probability estimation for test dataset. The root mean square error of deal probability was calculated as an evaluation of prediction performance. The best rmse performance is **0.2356** with random forest model of 100 estimators(the number of trees in the forest.) and 15 maximum features (the number of features to consider when looking for the best split). More estimators and higher number of features made less estimation mean square error, but higher RMSE for test dataset, indicating overfitting. The largest 15 feature importance is plotted in Fig. 59. The higher the feature importance, the more important the feature. We noticed that the most important features are mean deal probability of image_top1, standard deviation of deal probability of image_top1, price_rank1 and price. From the investigation we conclude that the Avito's classification code for the image (image_top1) and price plays the most important roles in the deal probability.

parameter name	p value
description_num_unique_words	1.099895e-04
region_price_mean	6.304433e-06
parent_category_name_price_mean	5.374635e-25
title_length	9.367386e-32
parent_category_name	1.973413e-92
category_name	8.453563e-109
price	3.988890e-132
item_seq_number	2.860326e-141
price_rank1	7.186882e-286
category_name_deal_probability_mean	0.000000e+00
user_type_price_mean	0.000000e+00
param_3	0.000000e+00
param_2	0.000000e+00
param_1	0.000000e+00
image_top_1	0.000000e+00
image_y_or_n	0.000000e+00

Figure 58: multiple regression with 16 parameter

6.3 Support Vector Regression (SVR)

To begin with, we need to determine the number of essential principle components left after PCA decomposition process. The remaining principal components should contain as much as variability of the original dataset and simplify the datasets into as many as least features to improve the efficiency of the model. Before all manipulations, we should standardize the dataframe using Standarscaler() function from sklearn to scale all the features to unit variance. Then firstly we assign the number of remaining principal components k from 1 to 350 with step size 50. Using default RBF kernel settings for SVM model, we get predicted for training and validation datasets, and calculate the RMSE for training dataset and validation dataset respectively for each k value. The result (60) is shown below:

On the graph, the blue dots are for the validation set and the red ones are for training

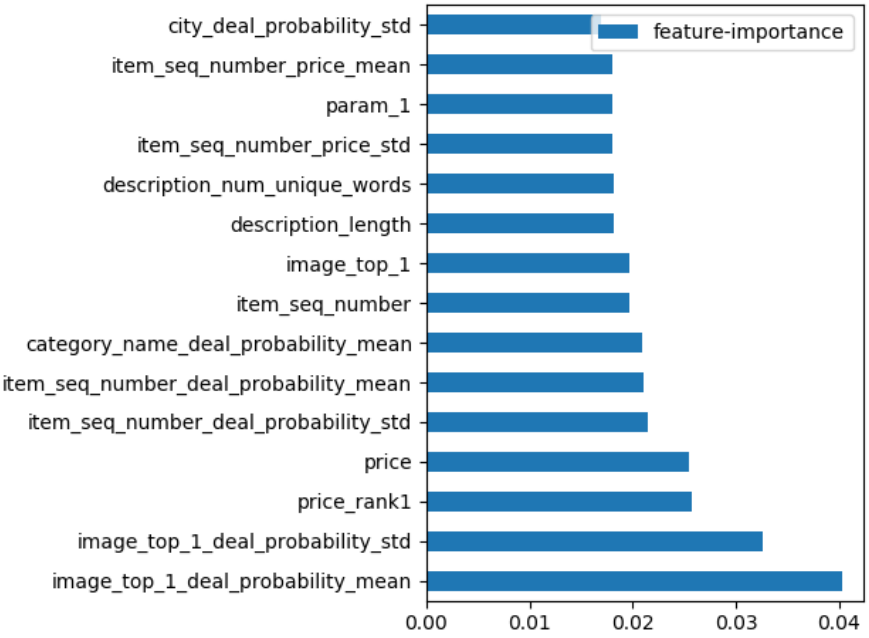

Figure 59: feature importance from random forest

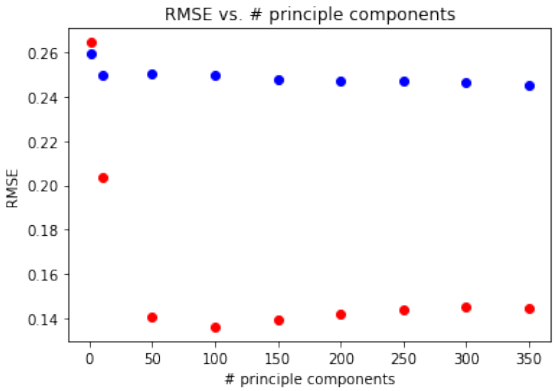

Figure 60: RMSE of Training and Validation Dataset for k from 1 to 350

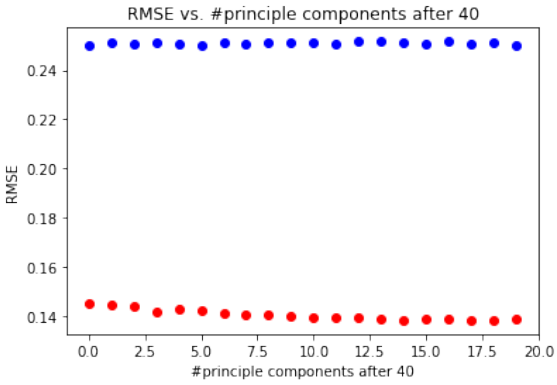

Figure 61: RMSE of Training and Validation Sets for k Ranging from 40 to 60

sets. Obviously, for both sets, the rmse value decreases around k=50, and maintains flat after that. Thus, we infer that the optimal number for principal components we should keep is around 50 after which the accuracy does not change dramatically. Let's look into the interval for k from 40 to 60 with step size 1 (61) below:

Still, the blue dots are the validation set and the red ones are the training sets. From the graph, we can not see much fluctuations already, but after 10 (k=40+10=50), both trends are almost perfectly flat without any tilts. Hence, we conclude that the optimal principle components remaining should be 50 when doing PCA.

To determine optimal values of essential parameters C and gamma for SVR, we use GridSearchCV() function in python by inputting training dataset after PCA transformation. Empirically, we assign C and Gamma (62) with values ranging from 10e-2 to 10e2:

```
C_list = [10**-2, 10**-1, 1, 10, 100]
gamma_list = [10**-2, 10**-1, 10**-0, 10**1, 10**2]

parameters = {'C':C_list, 'gamma':gamma_list}
model_cv = GridSearchCV(model, parameters, return_train_score='true')
model_cv.fit(pca_xtrain, y_train1)
```

Figure 62: Grid Search to Find Best Parameters

After fitting the model, calling the embedded function best_estimator_(63) gives us:

```
In [55]: model_cv.best_estimator_

Out[55]: SVR(C=0.1, cache_size=200, coef0=0.0, degree=3, epsilon=0.1, gamma=0.01,
           kernel='rbf', max_iter=-1, shrinking=True, tol=0.001, verbose=False)
```

Figure 63: Result of Best Parameters

We also try the different C and gamma under different kernels. The RMSE are compared in the table (64) shown below:

RMSE of SVR under Different Kernel, C, and Gamma

Kernel	C	Gamma	RMSE (training)	RMSE (validation)	RMSE (test)
Linear	1	None	0.2289	0.2328	0.2952
Polynomial	1	None	0.2271	0.2311	0.2846
RBF	1	Auto	0.2226	0.2296	0.2745
RBF	0.1	0.01	0.2197	0.2251	0.2663

Figure 64: Comparison of RMSE for Different Kernel, C, and Gamma under SVR

We can see that RBF kernel outperforms the linear and polynomial kernel. For the RBF kernel, when the kernel coefficient gamma is too small, the model does not perform well since it cannot capture the shape of the data points and lower its performance in higher-dimensional feature space. However, the larger kernel coefficient gamma will result in over-fitting due to the majority influence on support vectors under larger radius of the area. Thus, the intermediate values of gamma with larger C have better performance since it can not only reasonably regularize the number of support vectors but also avoid over-fitting. Overall, the best RMSE result on Kaggle is 0.2663. The screenshot (65) of RMSE is shown below:

Figure 65: The Best Result of RMSE under SVR Model

6.4 Gradient Boosting Regression

The training data set is again split into 80% train data and 20% validation data. The training data is used to train the gradient boosting model and the validation data is used to generate a score of this model based on root-mean-squared error (RMSE). The test data set is then generated using the one provided by Kaggle, and the score as provided by Kaggle is provided as well to study the goodness-of-fit of models.

Upon reading the documentation, there are two parameters mainly in charge of fitting different models: the n_estimators and max_depth. The number of estimators stands for the number of iterations; in other words, the number of trees fitted. The maximum depth stands for the number of leaves in one tree. In theory, the bigger the n (number of trees) and m (number of leaves) are, the better the model will fit. The n is set at 100 (default), 500, 1000 while the m is kept at 3 (default). The train loss (defined by a linear regression function) is plotted against certain iterations too see the effect of adjusting m.

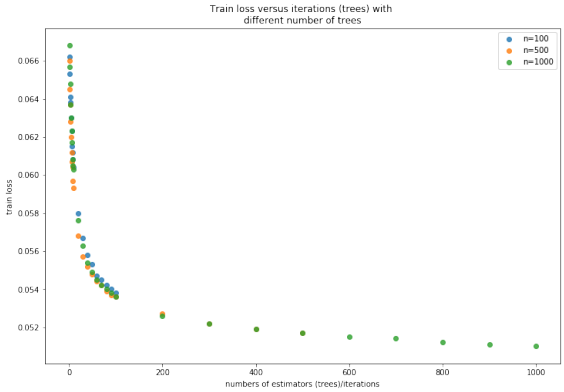

Figure 66: Scatter1

It is apparent that as m is increased, the number of iteration extends. Even though the scatter points stay close to each other for the first 100 iterations, they would keep decrease as iteration goes up, eventually leading to an overall better train loss result.

This, m is kept at 100 (default), and m is set at 3 (default), 5, 7, respectively, to see the effects of adjusting n.

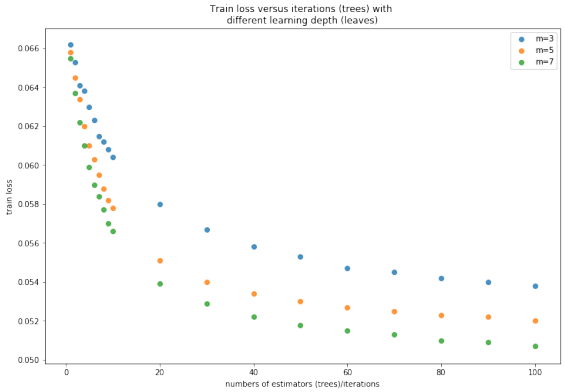

Figure 67: Scatter2

Again, as n is increased, the model seems to be more efficient decreasing the train loss. Therefore, it is expected that a higher m and n would lead to a better fitted model.

We then try to validate this by plotting the heat map of results from different m and n values. The result here is the RMSE of our validation data, the first scoring methodology. Since the score stands for the deviation from the ac-

tual deal probability of validation data, a smaller value indicates a better fitted model.

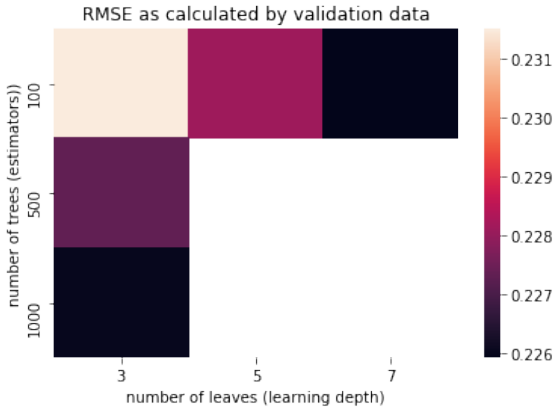

Figure 68: Heat1

The results are intuitive, in a sense that as m and n are increasing, the RMSE scores tend to decrease (indicated by a darker color). The model where it utilizes 100 trees and 3 leaves per tree yields the worst result.

Then, we use the second scoring methodology and try to run the test data downloaded at Kaggle in the models to see what score it would generate. The results are presented in another heat map,

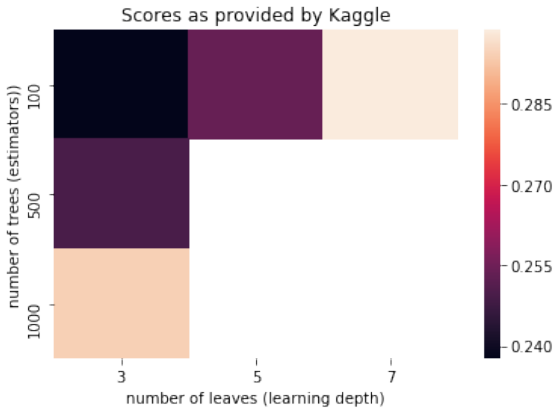

Figure 69: Heat2

It is surprising to find out that the heat map this time yields contradictory results! This conflict can possibly be explained by over-fitting of the model. Since the model is fitted in such a way that it aims to decrease the errors in validation data, the machine may be solely gearing towards that purpose, while the model it fits is not actually that accurate when predicting the test data. Therefore, we conclude that the Gradient Boosting Regression model is not actually a great choice for our data, as the model utilizing the default values (the lowest numbers of trees and leaves) is the one that best predicts the test data.

6.5 XGBoost

We use XGBoost as the machine learning model to train the dataset. Since 80% of the dataset has been used as the training set, 20% of the dataset has been used as the validation set, the training procedure will stop until the RMSE has not been improved in 50 rounds in validation part. For the parameters in the XGBoost, we use the tree based models because it usually has better performance than the linear models. The learning rate is chosen as 0.3, the min_child_weight, the minimum sum of weights of all observations required in a child, is chosen as 0 that is used for controlling the overfitting and underfitting in the imbalanced calss, and almost the rest parameters are used in default settings. Alpha is used here that means the L1 regularize can run faster in the higher dimension and avoid overfitting. In addition, the logic regression is used as the loss function for the objective and RMSE is chosen as the evaluation metric. In addition, we also try another hyper-parameters tuning method. We increase the max_depth to be 18. At the same time, min_child_weight needs to be increased as well to 3. We increase the training iterations due to the decrease of learning rate. The same early stop rule is applied here. One of the training procedure (70) is shown below:

```
[0]     train-rmse:0.352962     valid-rmse:0.353753
Multiple eval metrics have been passed: 'valid-rmse' will be used for early stopping.

Will train until valid-rmse hasn't improved in 50 rounds.
[5]     train-rmse:0.227352     valid-rmse:0.231942
[10]    train-rmse:0.217243     valid-rmse:0.224727
[15]    train-rmse:0.212824     valid-rmse:0.223472
[20]    train-rmse:0.209165     valid-rmse:0.223256
[25]    train-rmse:0.205846     valid-rmse:0.2233
[30]    train-rmse:0.202744     valid-rmse:0.22345
[35]    train-rmse:0.199752     valid-rmse:0.223556
[40]    train-rmse:0.196963     valid-rmse:0.223764
[45]    train-rmse:0.194361     valid-rmse:0.223956
[50]    train-rmse:0.191993     valid-rmse:0.224176
[55]    train-rmse:0.189508     valid-rmse:0.224369
[60]    train-rmse:0.186987     valid-rmse:0.224613
[65]    train-rmse:0.184688     valid-rmse:0.224804
[70]    train-rmse:0.182336     valid-rmse:0.22497
Stopping. Best iteration:
[22]    train-rmse:0.207766     valid-rmse:0.22322
```

Figure 70: The Training Procedure of XGBoost

In order to get the better performance under XGBoost model, we also vary the range of hyper-parameters such as max_depth, min_child_weight, subsample, and learning rate. The tuning procedure is hard because even if the RMSE is low in validation set, it might be extremely high during the test due to the overfittig. After multiple experiments, we find that the best RMSE result is 0.2303 in test. The screenshot (71) of RMSE is shown below:

Figure 71: The Best Result of RMSE under XGBoost Model

We also plot the feature importance (72) under the best XGBoost model. Undoubtedly, price, item_seq_number, image_top_1, description length, param_1, and city are top crucial features in training the model. These points satisfy our expectation because we indeed find the relationship between these features and deal probability during the data exploration.

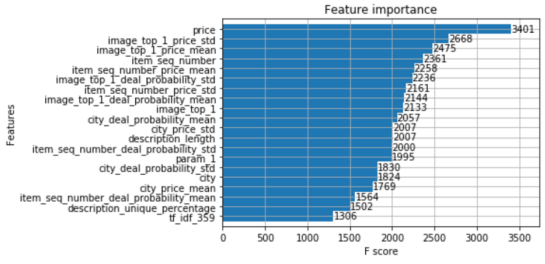

Figure 72: The Feature Importance for XGBoost Model

6.6 LightGBM

We use LightGBM as the machine learning model to train the dataset. Similarly, 80% of the dataset will be used as the training set, 20% of the dataset will be used as the validation set. The training procedure will stop until the RMSE has not been improved in 150 rounds in validation part due to the larger number of iterations so as to avoid overfitting. The hyper-parameters are hard to tune since LightGBM is easier to cause the overfitting. Even if the RMSE is low in the validation set, it will be ridiculous high in the test. Bascially, we choose regression as the objective function, RMSE as the metric accuracy. In order to gain the better accuracy, we use large num_leaves and smaller learning rate. The bagging_fraction and feature_fraction are also used for pursing the faster training. During the multiple experiments, we also vary the max_depth and lamda to control the overfitting or underfitting. Due to the larger iteration times, part of the screenshot for the training procedure is shown below:

```
[3180]  valid_0's rmse: 0.22526
[3200]  valid_0's rmse: 0.225259
[3220]  valid_0's rmse: 0.225268
[3240]  valid_0's rmse: 0.225271
[3260]  valid_0's rmse: 0.225267
[3280]  valid_0's rmse: 0.225264
[3300]  valid_0's rmse: 0.225257
[3320]  valid_0's rmse: 0.225256
[3340]  valid_0's rmse: 0.225258
[3360]  valid_0's rmse: 0.225258
[3380]  valid_0's rmse: 0.225253
[3400]  valid_0's rmse: 0.225254
[3420]  valid_0's rmse: 0.225255
[3440]  valid_0's rmse: 0.225256
[3460]  valid_0's rmse: 0.22526
[3480]  valid_0's rmse: 0.225258
[3500]  valid_0's rmse: 0.22526
Early stopping, best iteration is:
[3406]  valid_0's rmse: 0.225251
```

Figure 73: The Training Procedure of LightGBM

The best RMSE result on Kaggle is 0.2296. The screenshot (74) of RMSE is shown below:

Figure 74: The Best Result of RMSE under LightGBM Model

We still plot the feature importance (75) under the best LightGBM model. Similarly, most top features are the same as before that satisfy our expectation due to the investigation in data exploration. These null hypothesis tests show us which features probably potentially be useful in machine learning models training.

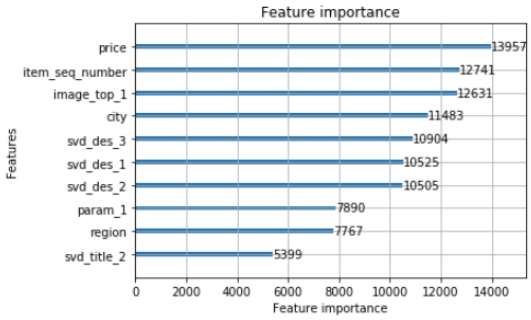

Figure 75: The Feature Importance for LightGBM Model

7 Conclusion and Discussion

7.1 Hypothesis Test

To begin with, we start by exploring the true meaning behind the feature item_seq_number and its relation with the target variable, deal probability. We conclude that the item_seq_number stands for the number of

times a certain user posts the advertisement on the website. While studying the relationship, we found out that the trend is more apparent when only considering the first 90% of sequential numbers for private users. This is because the behavior of private users (filtering out extremely large sequential numbers) are more predictable. It is found that there is a negative correlation between item sequential numbers and deal probability for these users. Therefore, item_seq_number is selected as a potential feature in our models.

The description is then explored to study how it affects the deal probability. Intuitively, a longer, more detailed description would lead to a higher deal probability. Here, we further divide description into three features: the length of description, the count of adjectives and the percentage of adjectives in the description. Results from hypothesis tests show that the length of description does cause a difference in deal probability (null hypothesis rejected), while the count of adjectives and the percentage of adjectives does not make a significant difference in deal probability (null hypothesis failed to reject). Therefore, only length of description is selected as a feature in feature engineering.

The next feature we study is the period of time the advertisements are displayed. Since a longer display time will lead to more people seeing the advertisement, and subsequently an increase in deal probability, but a shorter display time might imply that the item is sold out quickly, the seemingly contradictory observations are studied using hypothesis tests. Results present that there is no significant difference (null hypothesis failed to reject) between deal probabilities of long and short displayed period groups. Therefore, we conclude that the displayed time of advertisement does not affect the deal probability, and is not considered as a feature in later on models.

Whether the item with missed price affects the deal probability is studied next. Since the price column of some advertisements are NA, these items are grouped to see if it has a significantly different deal probability between these two groups. After detailed exploration, we conclude that overall, 'missing price' does not have significant impact on deal probability, and thus feature is thus eliminated for future model constructing.

Next, price is further studied within categories to explore its impact on deal probability. Hypothesis tests are performed within all categories and some p-values goes above the significance level, 0.05. Therefore, it is concluded that there is at least one group which is showing a significant difference of deal probabilities. Therefore, the price rank within categories can serve as a new feature for future model constructing.

Whether the advertisement has image(s) changes its deal probability is then studied. Intuitively, an advertisement with image should be more informative and tend to be sold more easily. However, the result indicates that the effect of images come in three ways: it can serve as either a positive, neutral, or negative effect on deal probability, depending on the category. This makes sense because while some images may serve as an informative tool that gives a more comprehensive review of the item, other images may have negative influence due to their lack of quality or showcase of the actual and not-so-much appealing condition of a (second-hand) item, thus preventing the item from being purchased.

Our last data exploration studies if the GRP of regions has a relation to deal probability, where the GRP corresponds to the city of the advertisement posted.

7.2 Models

After data exploration and feature engineering, we build six different machine learning models: **Linear Regression**, **Random Forest**, **Support Vector Regression**, **Gradient Boosting Regression**, **XGBoost**, and **LightGBM**. The results of metric accuracy RMSE under six different machine learning are shown below (76). Undoubtedly, Boosting approaches outerperform linear regrssion and SVR due to the tree structure. Linear Regression is limited to that easier model that can not fit complex dataset enough. While, SVR is limited to the extremely longer training time under RBF kernel. Meanwhile, we only use part of the train dataset to train the model, which causes the worst RMSE result in test. Random Forest and Gradient Boosting Regression have close performance. Their performances can not be improved due to the easy decision rule. But they can still reduce bias and variance effectively during the training. XGBoost and LightGBM have the almost best performance contributing to the more complex regularize and split rule. But, it is hard to tune so many hyper-parameters such as the parameters used for controlling tree structure, learning rate, and regularize, which is easier to cause the overfitting. Overall, Boosting algorithm is the better machine learning model for this dataset in our exploration and experiment.

Best RMSE Results in test under Different Machine Learning Models

	Linear Regression	Random Forest	SVR	GBR	XGBoost	LightGBM
Best RMSE (test)	0.2465	0.2356	0.2663	0.2377	0.2303	**0.2296**

Figure 76: The Best RMSE Result in Test for Different Machine Learning Models

7.3 Discussion

It's difficult to achieve good performance of the deal demand prediction with simple models directly taking the dataset features. Feature extraction from an initial set of measured data and building derived features intended to be informative and non-redundant, facilitate the subsequent learning and regression performance. In this report, we tried to explore how features affect the deal probability by testing 7 hypothesis, and then incorporated the exploration results in feature engineering. Finally 6 different models were applied to predict the deal demand. However, due to limited computation resources, limited features were extracted. More unfortunately, we found images have significant impact on deal probability but we could not make full use of the huge amount of data (over 50 GB) yet. As a future work, feature extraction from images, with deep learning techniques for example, is of vital importance. With sufficient computer memories, study on subset of images could also be used to explore the relationship between the images and deal probability. We believe by taking full advantages of the images, we could make much better prediction performance in the future.

References

[AK70] Mariette Awad and Rahul Khanna. Support vector regression, January 1970. link.springer.com/chapter/10.1007/978-1-4302-5990-9_4.

[Bre01] Leo Breiman. Random forests. *Machine learning*, 45(1):5–32, 2001.

[CR09] Yen-Soon Kim Carola Raab, Karl Mayer. Price-sensitivity measurement: A tool for restaurant menu pricing. *Journal of Hospitality & Tourism Research, Vol. 33, No. 1,*, 2009.

[EHF04] David McHardy Reid Eugene H. Fram, Lu Le. Consumer behavior in china: An exploratory study of two cities. *Journal of Asia-Pacific Business Volume 5,2004 - Issue 4*, 2004.

[FSA99] Yoav Freund, Robert Schapire, and Naoki Abe. A short introduction to boosting. *Journal-Japanese Society For Artificial Intelligence*, 14(771-780):1612, 1999.

[GK17] Thomas Finley Guolin Ke, Qi Meng. Lightgbm: A highly efficient gradient boosting decision tree. *Advances in Neural Information Processing Systems 30*, 2017.

[Pai12] Paul Paisitkriangkrai. Linear regression and support vector regression, October 2012. cs.adelaide.edu.au/~chhshen/teaching/ML_SVR.pdf.

[Par12] Iain Pardoe. Applied regression modeling. 2012.

[TC16] Carlos Guestrin Tianqi Chen. Xgboost: A scalable tree boosting system. *22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 785–794, 2016.

[Wel] Max Welling. Support vector regression. www.ics.uci.edu/~welling/teaching/KernelsICS273B/SVregression.pdf.

[Wik18] Wikipedia. Principal component analysis, June 2018. en.wikipedia.org/wiki/Principal_component_analysis.