ML Project. Avito Demand Prediction Challenge

Predict demand for an online classified ad

Executed by: Sodik Umurzakov

Executive summary

In this final project, we are going to use Avito's data from kaggle competition for predicting demand for an online classified ad using different characteristics like regions, description, price etc. We should highlight that there are also photos of the product attached, but we will not use it due to computations reasons.

When selling used goods online, a combination of tiny, nuanced details in a product description can make a big difference in drumming up interest. And, even with an optimized product listing, demand for a product may simply not exist–frustrating sellers who may have over-invested in marketing. Therefore, Avito 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.

Data preparation

In the modelling process we want to use train data and other part of data from below listed files. The data has about 1 503 424 observations and more 17 columns that describe ads' characteristics and related photos of ~140 GB which we will not use. So, there are a lot of features can be calculated using these 17 columns as we will include NLP as well for text descriptions and characteristics of ads and thorough EDA must be done to obtain consistent results. To avoid overfitting issues and fine tune of perparameters, we will follow train / test split by 80% and 20% respectively and use k-fold cross validation which is applicable as we have cross sectional data. However, we should keep in mind that over train and test split must be done properly to account for representativity issues so we assess model on similar sample (shortly, we should compare apples with apples). The description of the data files from the data page:

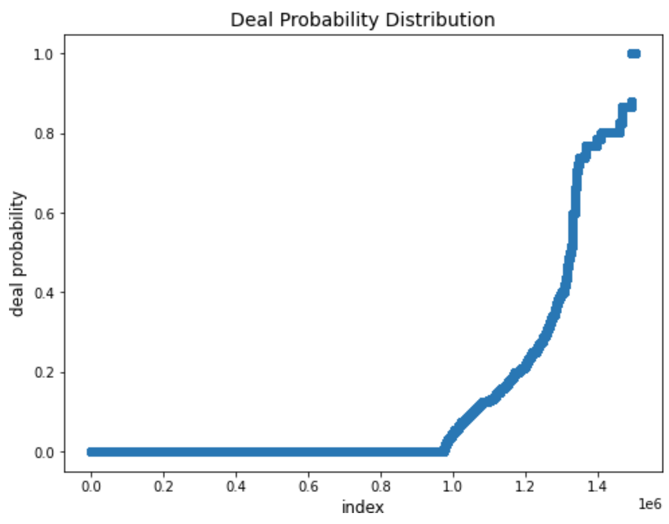
- train.csv - Train data.
- test.csv - Test data. Same schema as the train data, minus deal probability.
- train_active.csv - Supplemental data from ads that were displayed during the same period as train.csv. Same schema as the train data, minus deal probability.
- test_active.csv - Supplemental data from ads that were displayed during the same period as test.csv. Same schema as the train data, minus deal probability.
- periods_train.csv - Supplemental data showing the dates when the ads from train_active.csv were activated and when they were displayed.
- periods_test.csv - Supplemental data showing the dates when the ads from test_active.csv were activated and when they were displayed. Same schema as periods_train.csv, except that the item ids map to an ad in test_active.csv.
- train_jpg.zip - Images from the ads in train.csv.
- test_jpg.zip - Images from the ads in test.csv.

- sample_submission.csv - A sample submission in the correct format.

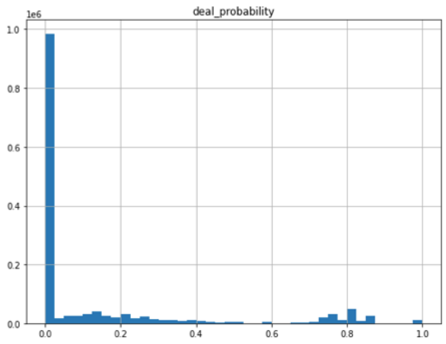
We parsed dates for activation date variable. Furthermore, the train dataset description is as follows:

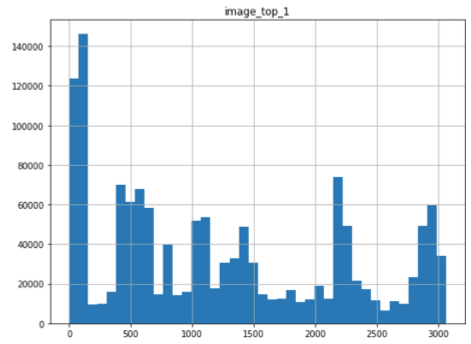
- item_id - Ad id.
- user_id - User id.
- region - Ad region.
- city - Ad city.
- parent_category_name - Top level ad category as classified by Avito's ad model.
- category_name - Fine grain ad category as classified by Avito's ad model.
- param_1 - Optional parameter from Avito's ad model.
- param_2 - Optional parameter from Avito's ad model.
- param_3 - Optional parameter from Avito's ad model.
- title - Ad title.
- description - Ad description.
- price - Ad price.
- item_seq_number - Ad sequential number for user.
- activation_date- Date ad was placed.
- user_type - User type.
- image - Id code of image. Ties to a jpg file in train_jpg. Not every ad has an image.
- image_top_1 - Avito's classification code for the image.
- deal_probability - The target variable. This is the likelihood that an ad actually sold something. It's not possible to verify every transaction with certainty, so this column's value can be any float from zero to one.

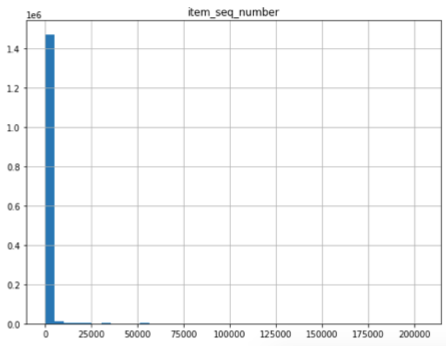
So, deal probability is our target variable and is a float value between 0 and 1 as per the data page. Let us have a look at it.

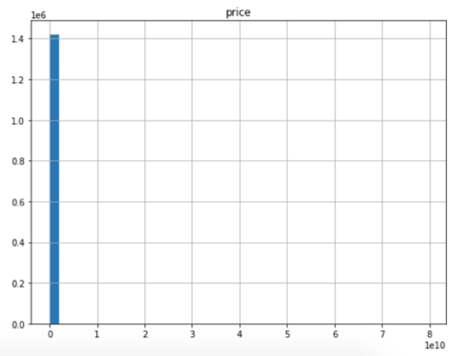


From above plot we can say that almost 100 th. Ads has 0 probaility which means that it did not sell anything, and few ads have a probability of 1. Rest of the deal probabilities have values in between 0 and 1.





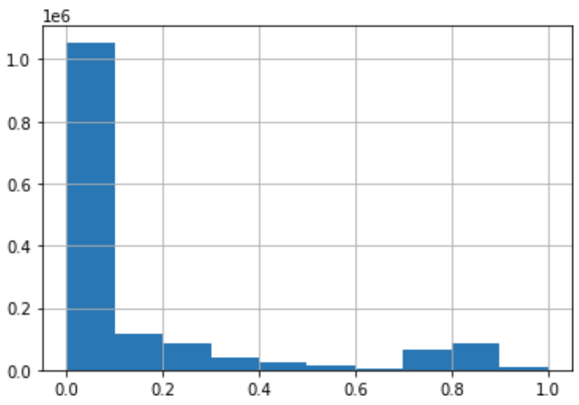


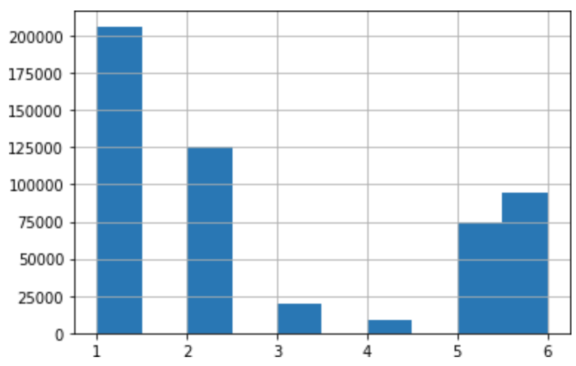


Above plot depicts the countable variables with their distribution. As our target is deal probability and it has most distributed around zero. The target variable is decomposed by regions and category of products that are sold mostly and shows that most of goods are not sold as well as its around zero. Other non-target variables are also mostly around zero. Many suck kinds of histograms are tail heavy, and they extend to the right of the median than to the left. This reason would call some problems for ML algorithms to fix best patterns.

Creating test set

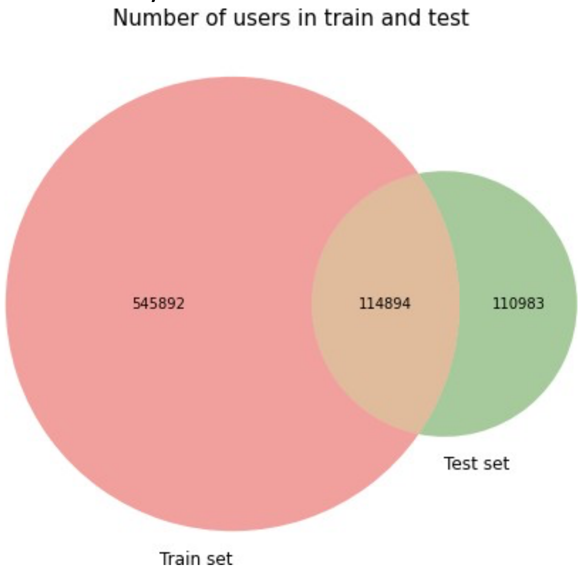
We have divided our data into two parts: train and test by function train test split. Train – 80% and test – 20% of data. Below we can see the characteristics of the target variable one more time and for another new variable from test set – deal hat. The deal hat variable has been created by pandas.cut() function wuth bins and 6 lables.

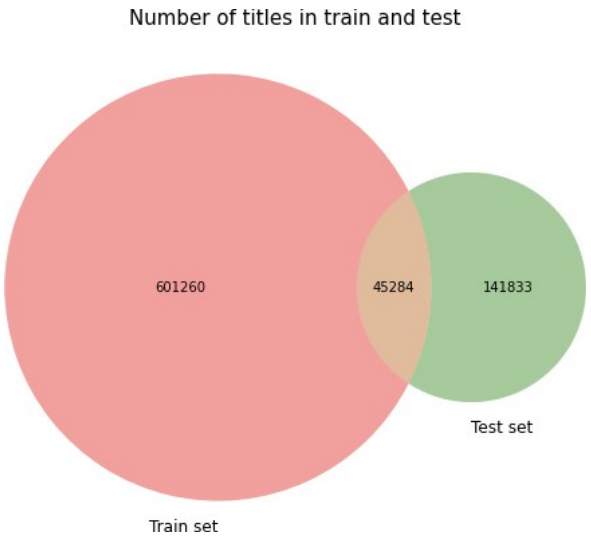




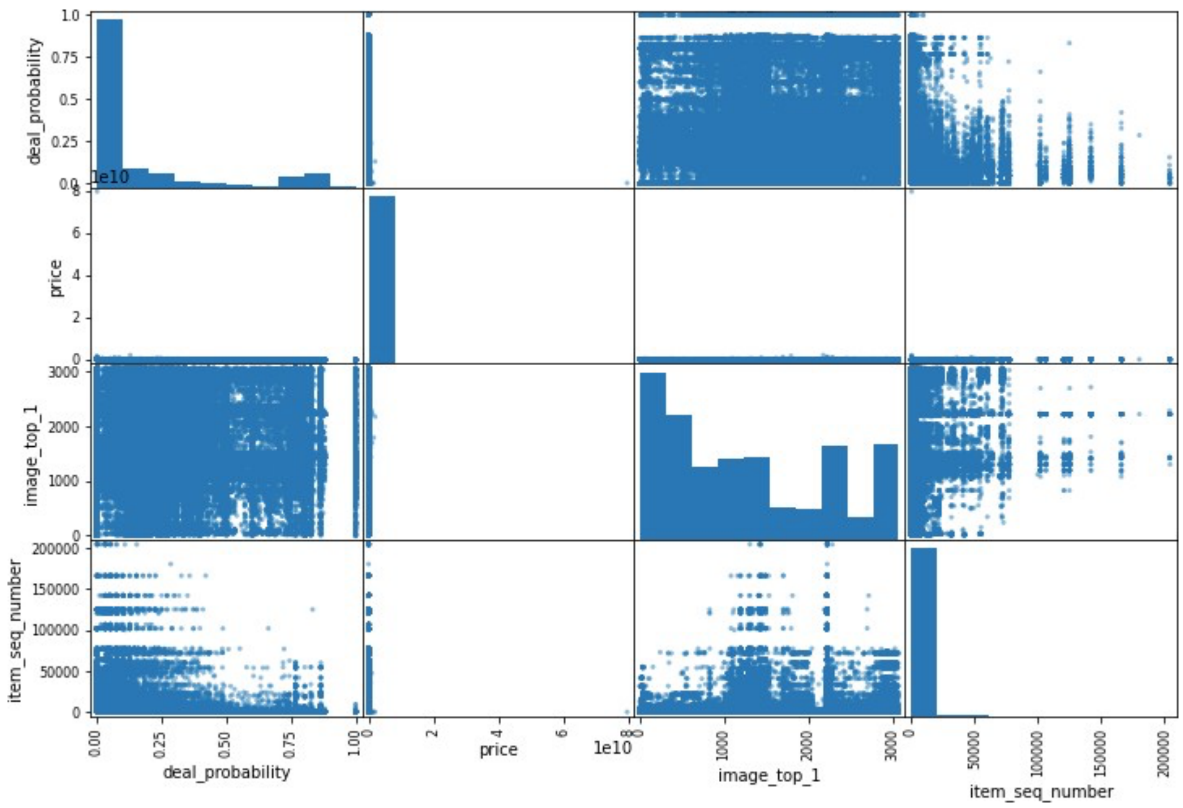
As we can see above that target variable is mostly distributed around zero. The target variable is decomposed by regions and category of products that are sold mostly and shows that most of goods are not sold as well as its around zero. For the new variable – deal hat we see that it also distributed around category 1 and 2.

Now let's try to count the number of users in train and test sets in graphical form.





The number of users in train set – 1 202 739 and in test set – 300 685, but interception of these two sets – 114 894. Uniqueness for train – 545 892 and for test set – 110 983.
The number of titles in train set - 1 202 739 and in test set – 300 685, but interception of these two sets – 45 284. Uniqueness for train – 601 260 and for test set – 141 833.
So, we conclude that with these decomposed sets will deal further in ML algorithmic modelling. Now, let's try to focus on the correlation of the target variable with others.

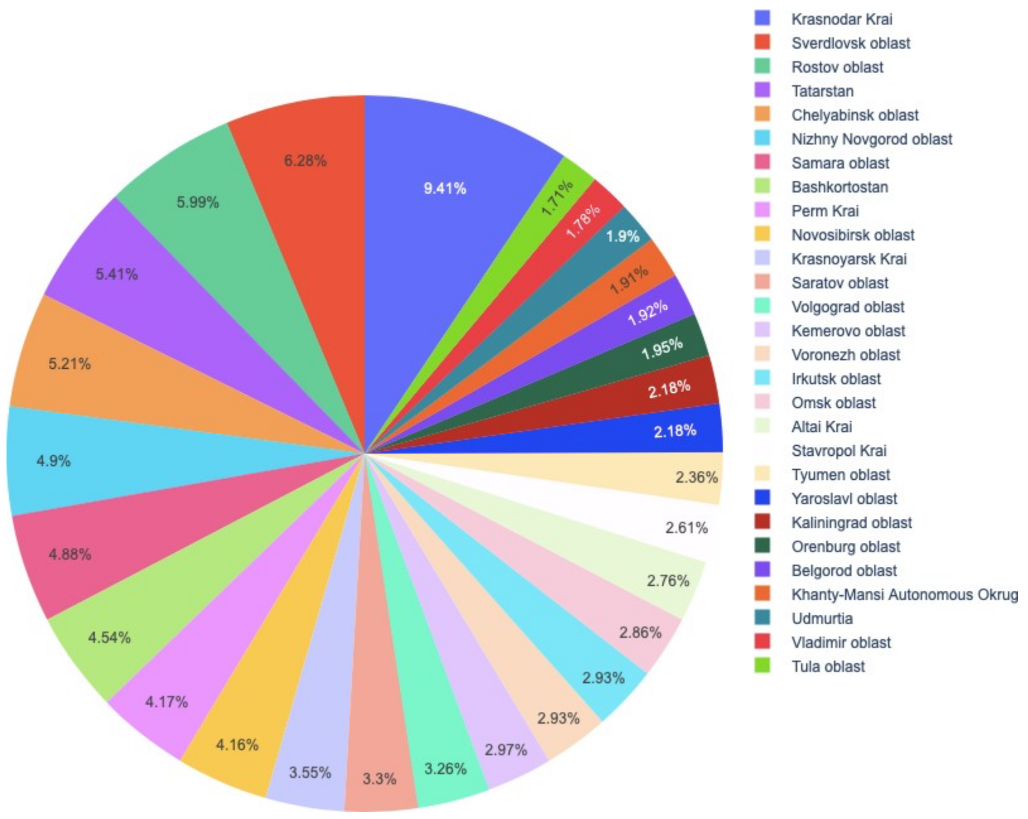


There are some correlations among these variables. Usually, the correlation coefficient which is ranged from –1 to 1. In case when it is close to 1, it means that there is a good positive correlation. In case when the coefficient is around –1, it means that there is a negative correlation. As we see negative correlation between our target variable with price and Ad sequential number for user. Because these two variables can't be useful for deal probability. Finally, coefficients close to zero means that there is no linear correlation.

For now, we want to skip with test set for EDA analysis as we know that test set helps us in modelling part to predict the demand for classified ads. Therefore, we would like to focus on train set for EDA analysis for this part.

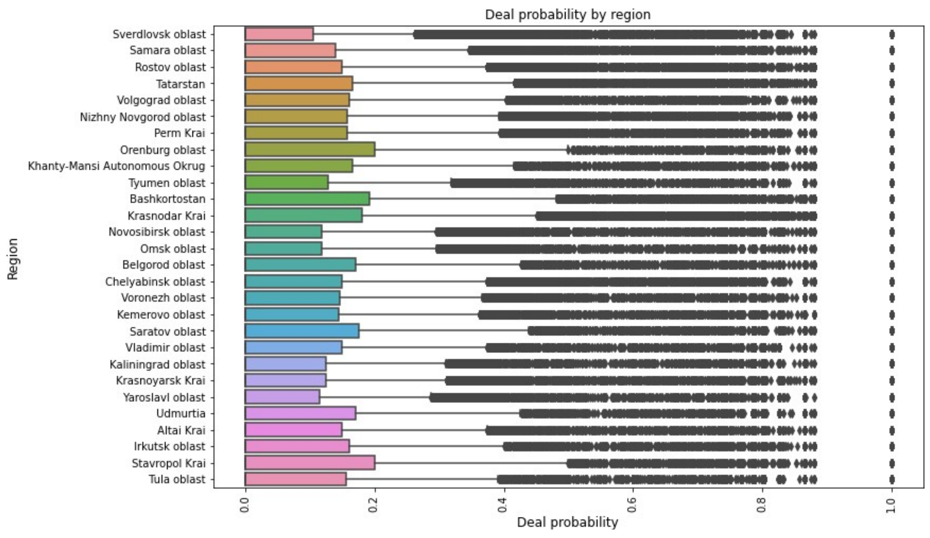
Data Visualization

Region wise distribution of Ads:



The main purpose of region distribution is about how to make a decision in what region actively use Ads for sales. Further, we have to work on with those regions to find a pattern how to predict demand for Ads and sales among avito users. The regions have percentage of ads between 1.71% to 9.41%. The top regions are:

1. Krasnodar region - 9.41%
2. Sverdlovsk region - 6.28%
3. Rostov region - 5.99%
4. Tatarstan - 5.41%



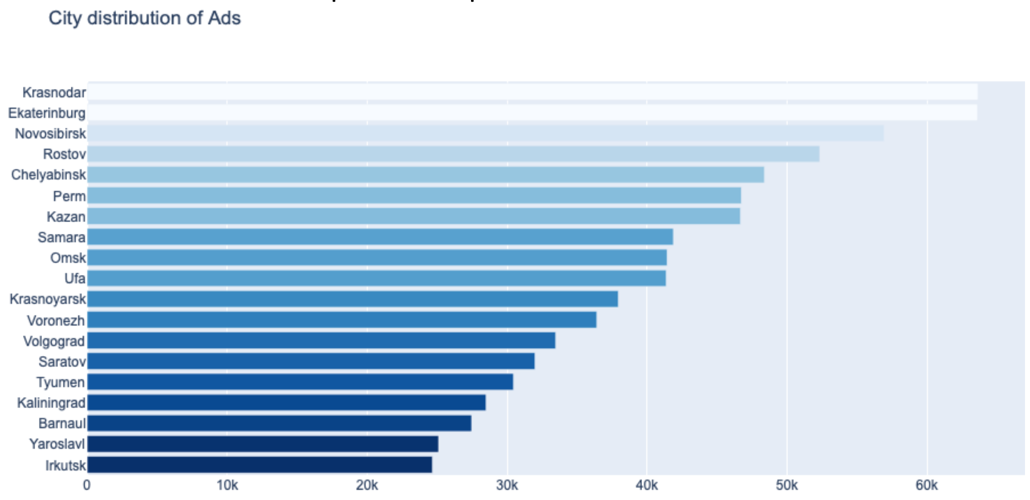
The plot shows the probability of deal conducting among various regions. It means in what regions ads effectively generate used goods for sales. Below the deal probability by region and leading regions are:

1. Sverdlovs oblast
2. Samara oblast
3. Rostov oblast
4. Tatarstan
5. Volgograd oblast

Sverdlovs and Samara oblasts are the highest one, we have to make sure the reason of this. In my point of view, it depends on wage level of the regions and they have not enough high wages and population density in Ural comparing to other regions. Most part of population prefers to buy used goods because of wages. Further, we try to analyze the situation in case of top cities distributed by ads.

City wise distribution of Ads:

Now let us have a look at the top 20 cities present in the dataset.



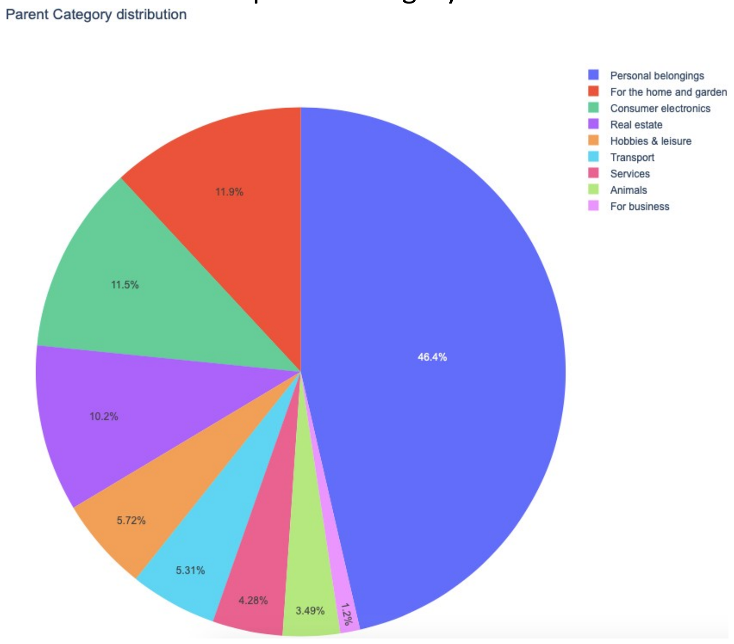
In this plot we see the top cities where the ads are depicted.

- Krasnodar
- Ekaterinburg
- Novosibirsk

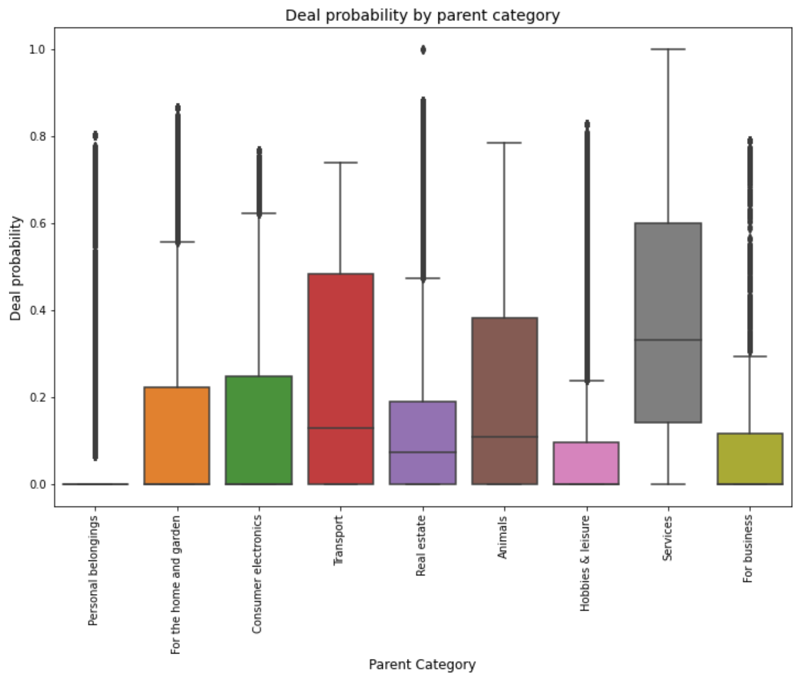
- Rostov

If we compare cities with regions, then we can make an option that two factors effect on Ads motivation: 1. Population density; 2. Wage level. As we see from plot approximately 65k ads come from that category of cities. Regarding the current situation of COVID-19 when Further, we can create a pattern for other non-top cities to create a demand.

Parent Category Name:
Now let us look at the distribution of parent category names.

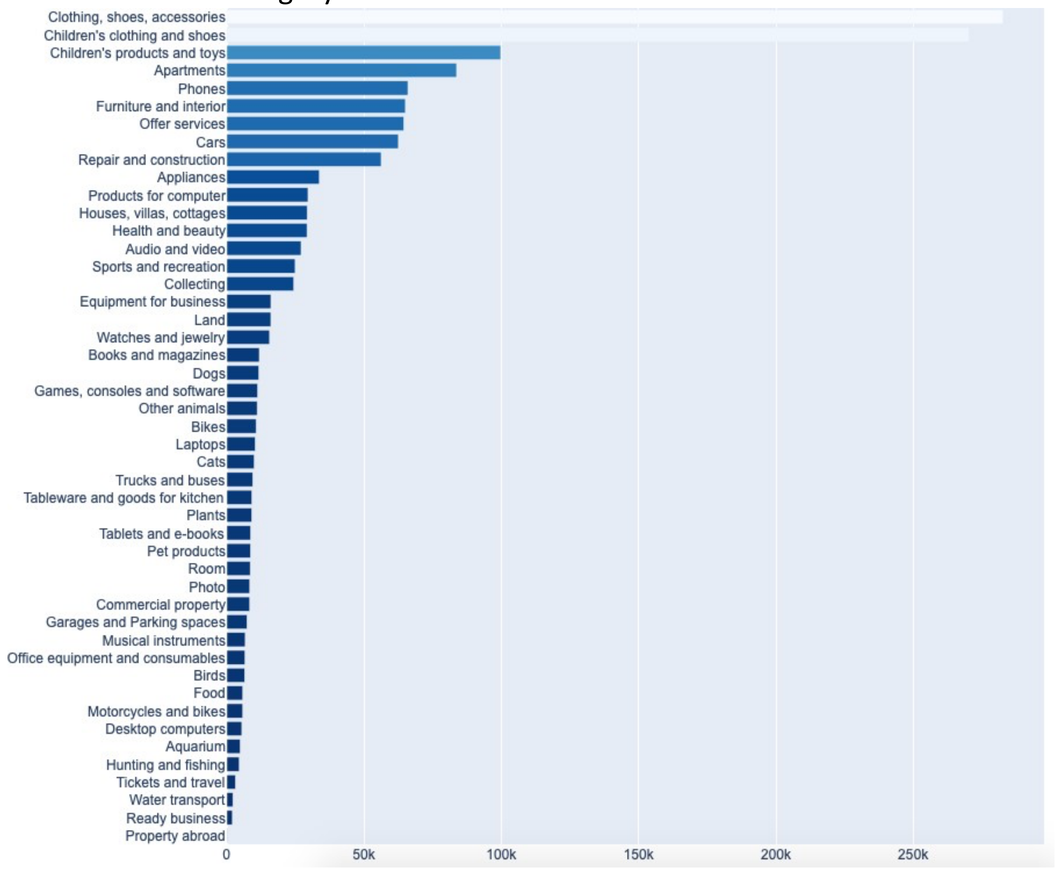


We see the situation of category distribution, 46.4% of the ads are for Personal belongings, 11.9% are for home and garden, 11.5% for consumer electronics, 10.2% for real estate. As usual in Avito platform where used goods are sold has high amount of share in personal belongings. And, we have to work on this category and to predict demand for Ads, therefore, it follows to make an accent in ML pipeline for demand prediction.



In this box we see that Services, Transport and Animals categories seem to have slightly higher deal probability compared to others. The key reason is non-durable products as services promotion are less expensive, therefore, the deal probability is higher to compare with other categories. As we see from above plot avito ads platform is not effectively suits for business development and hobbies and leisure, first, these categories are not so popular, second, for suck kinds of categories need to promote with high qualified persons from official webs or from Universities. So, our main goal should be filtering interested categories and focus on them for prediction.

Category of Ads:
Now let us look at the category of ads.

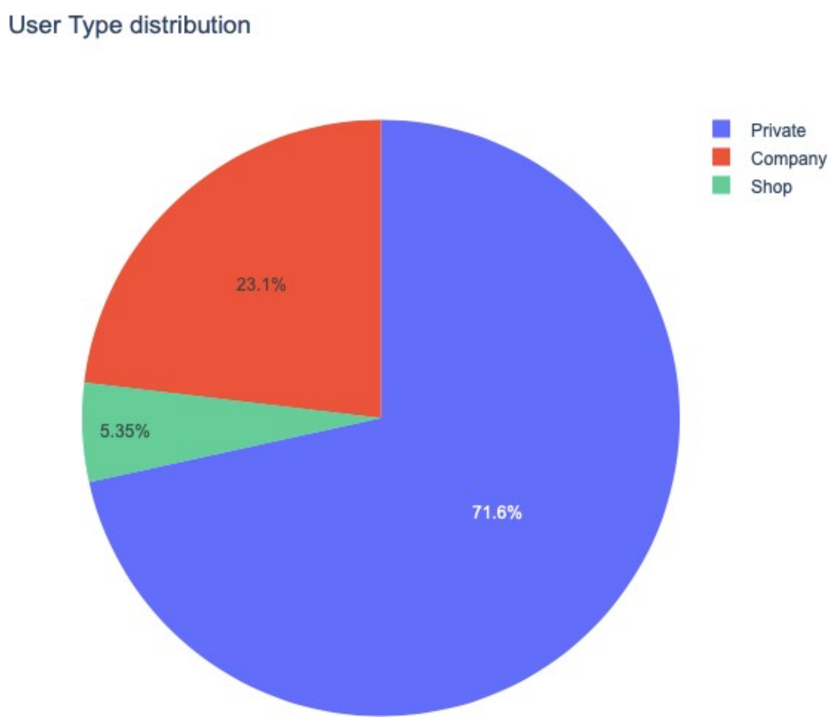


There are the top 3 categories demonstrated:

1. Clothes, shoes, accessories
2. Children's clothing and footwear
3. Goods for children and toys

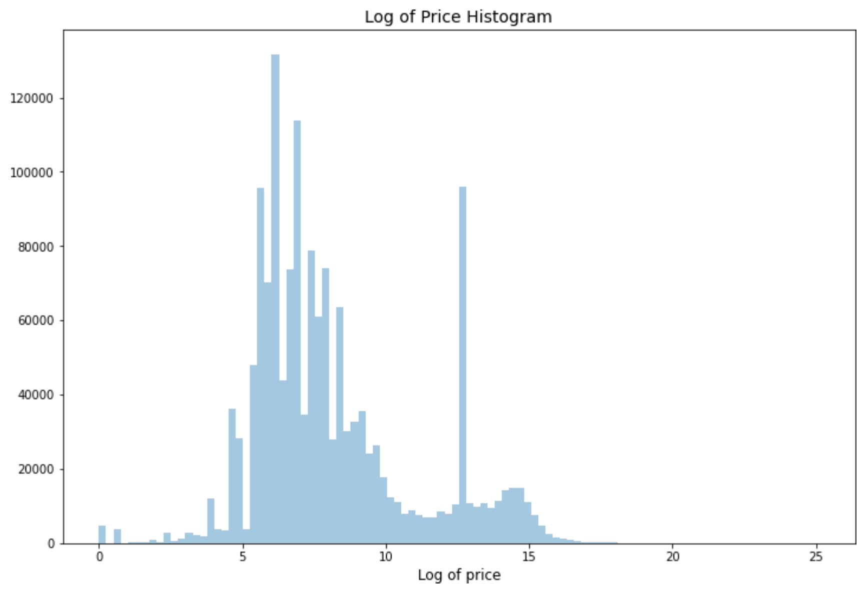
From this picture we can say, approximately 60% of ads come from top 3 categories for adult and children, and it means that avito platform targeted on clothes, shoes, accessories.

User Type
Now let us look at the user type.

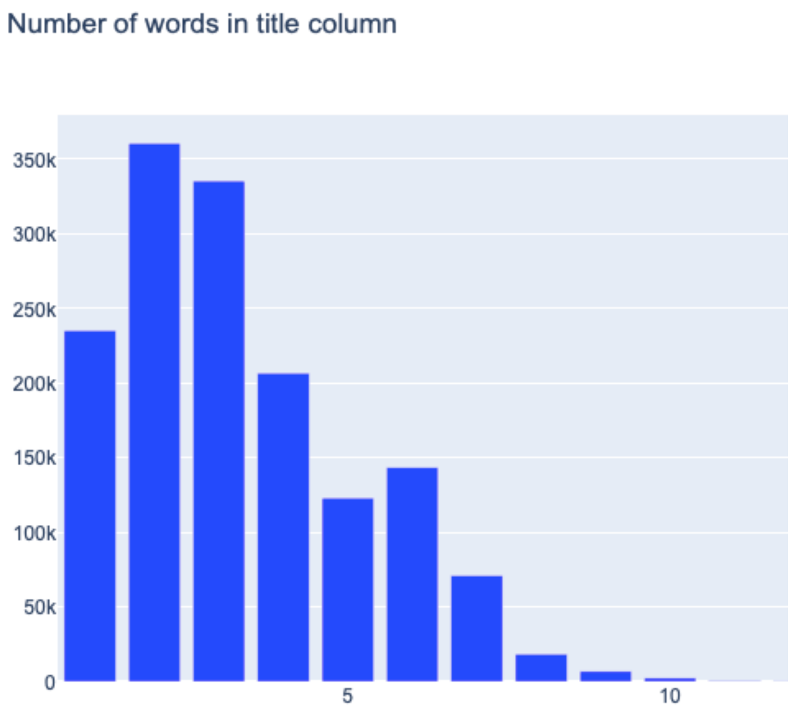


Private users constitute 72% of the data followed by company and shop. But company share's also huge and it is necessary to care about the quality of ads to increase this category. Because most users think about the security of the bought products. When the private is 71.6% out of 100% then it is more challengeable to predict the behaviour of the users and is not easy to analyse the ads content.

Price:
This is the price shown in the Ad.



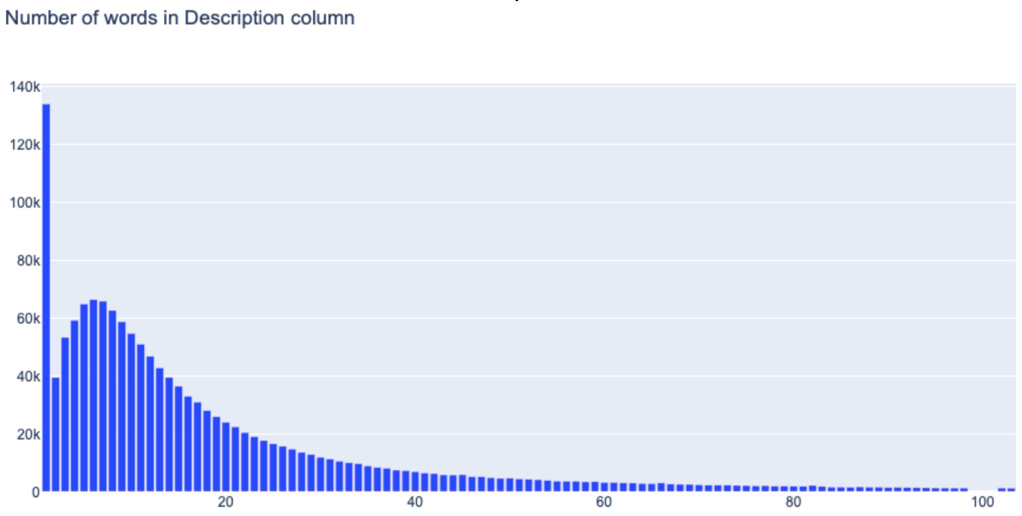
Here the log of price distribution is depicted it shows us the approximately 140k goods have log of price around 6-7. But we see that there are ***outliers*** in the dataset, and we have to drop them from our sample and try to get normally distributed log of price.
We have around 110K common titles between train and test set. Now let us look at the number of words present in the title column.



As we see above the number of words in the combined title column where decomposed to 10 categories. And, in the second category of goods the number of words is ~ 370k. 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. Further, we have to care about how to optimize the volume of words for sellers and recommend optimal amount to concentrate on key information.

Description:

Let us check the number of words in the description column.



Description column has a huge right tail till about 700 words. We have cut the same till top 100 for better visualization. But most of them left tail till approximately 135k goods that are generated for clothes, shoes, accessories etc. Further, we have to manipulate with the number of words for sellers to create online Ads.

Modelling

In this section we are going to deal with the modelling process and try to follow the framework how to build a model. We have decided to use advanced ML algorithm. This task is a regression problem as we have deal probability which is continuous variable between 0 and 1 where we are going to use state-of-the-art classical ML models such as XGBoost and Light GBM. Eventually, we will try to stack the models to see if there is an uplift in prediction power. Thus, optimization metric is MSE (mean squared error) that is default ones for such tasks and based on how we should punish deviations from true values use one of them. MSE metric punishes deviations more due to squared power. Now, let's proceed the XGBoost model.

XGBoost

For this model we have used hyperparameters and GridSearchCV, but we have more than 1,5 mln observations and 20 features for fitting and we spent 29 hours to generated optimum, therefore, because of time shortage we used optimal hyperparameters based on GridSearchCV results.
Best parameters:
```
{'max_depth': 5, 'learning_rate': 0.15}
```
Best score
0.0513

Light GBM

In this model we used the following:
- Number of estimators = 1 000;
- Max_debth = 5;
- Learning_rate = 0.08;
- Seed = 42.

Following by notebook we can see the fitting part of the model with train data and target variable.

Conclusion

In this task, we tried to predict the deal probability of the ad given qualitative and numeric features of the ad. Yet, we skipped photos and NLP as this course did not cover it and would take more time to make something worthy.

So, we used two tree-based models: XGBoost and LightGBM but fine-tuned only XGBoost as it takes to much computation power to train dataset with such many observations. Yet, we decided to use VotingRegressor from sklearn.ensemble, but it also took to much time and we stopped it. We believe, stacking these two models might give better results on test set.

To summarize our steps, we did comprehensive EDA; feature engineering: preprocessed categorical features, calculated etc.; fine-tuned parameters using KFold cross validation technique with 10 folds as we have sufficient amount to do so and tried to stack using VotingRegressor from sklearn.ensemble

On the table below, you can see that even LightGBM has not been fine-tuned it gave better results compared to XGBoost. This can be due to a novel technique of Gradient-based One-Side Sampling (GOSS) to filter out the data instances for finding a split value XGBoost uses pre-sorted algorithm & Histogram-based algorithm for computing the best split. Here instances are observations/samples.

	XGBoost	LightGBM
RMSE	0.232	0.229

These results can be improved further if decide to use at least NLP for textual data as from Kaggle notebooks we see that some of the features give significant uplift in score. We hope to improve our solution in such way using NLP and might be phots as well (but not sure, it requires GPU)