Deal Probability Prediction for a Classified Advertisement Website

Group 18

Wang Guangyu: A0231903N | Chen Keyi: A0218873U | Liu Huilin: A0231952H | Liu Ziyin: A0231908E | Shi Chenxi: A0232000L

GitHub Link: https://github.com/GY000/BT5153_Group18

1. Objective

Advertising is always living with us even though people may not be aware of it. With new technology in today's world, advertising uses all possible media to get its message to their targeted audience. For example, advertisements are delivered through television, newspaper, radio, audio, internet, phone apps etc. Advertising helps brands to reach more audiences and improve their brand awareness and also helps to drive a low funnel of key performance indicators such as sales. Despite covid situation, the advertising budget has been significantly increased among top advertisers and most of the social media platform's revenue are coming from advertising. This interests our team to study on a Kaggle project which is about the demand prediction of online classified advertisement. Avito is the largest and most popular Russian classified advertisement website. In this project, we are trying to predict demand for online advertisements that run on a website "Avito " that sells various kinds of goods. Leveraging the available data set for various kinds of advertisements, we are going to model and predict the probability of making a deal for those advertisements based on its advertisement title, description, image, context, location etc. With such a prediction, Avito could inform sellers on the optimization of their listing advertisements and share with sellers their potential rate to make a deal successfully.

2. Data Set

2.1 Data source

The data is from Kaggle. From Avito, we get Two databases. One is about ad attribute data which is stored on a single ad basis, including structured data like user id and ad id, text data likead title and ad description, our target – deal probability and so on. Also, we obtain the images from the ads, which is stored in jpg format in the image dataset. Combining the above, we can totally get a sample of 1503424 ads from Avito, with their advertising features, time data and image data. From the perspective of analysis, we will divide the data into two parts – structured data and unstructured data.

2.2 Data Description

2.2.1 Structured Data

Structured data is mainly from ad attribute dataset and period dataset, which are mainly categorical and numerical variables:

Table 1: Structured data description

Variable	Type	Descriptions
item_id	object	Ad id
user_id	object	User id
region	object	Ad region
city	object	Ad city
parent_category	object	Ad category
category_name	object	Ad category
param_1	object	Optional parameter
param_2	object	Optional parameter
param_3	object	Optional parameter
price	float64	Ad price
item_seq	int64	Ad sequential number
activation_date	object	Date ad was placed

user_type	object	User type
image	object	Id code of image
image_top_1	float64	classification code for the image
deal_probability	float64	The target variable

Note that param_1, param_2 and param_3 are features about keyword or classification conclusion of the ads, defined by website, such as the brand of the product, size, etc. Studying a lot of Kaggle contestants' ideas, many of them treated these features as categorical variables. We have reservations on this point. It is possible that these variables can be treated as text data. In the following analysis, we will try and combine both of the two ways of dealing with them. Besides, it's not possible to verify every transaction with certainty, so the Avito supposes that the value of our target variable, deal probability, can be any float from zero to one.

2.2.2 Unstructured Data

Unstructured data contains text data and image data:

Table 2: Unstructured data description

Variable	Type	Descriptions
title	object	Ad title.
description	object	Ad description.
param_1	object	Optional parameter
param_2	object	Optional parameter
param_3	object	Optional parameter
image	jpg	Images from the ads.

Since Avito is a Russian platform, we face the challenge to deal with Russian text features. We plan to use some open-source tools for dealing with Russian, such as nltk 4 russian - Russian texts tagging with Natural Language Toolkit, red hen lab for Russian NLP, word2vec skip-gram model for Russian language, etc. As the tools for analyzing Russian may not perform as well as those for English, we have a backup plan to use text blob or transliterate to translate the text to English for further analysis. For image data, we can extract image features, such as width and height, color and saturation of colors, histogram for contrast, composition of objects, from the jpd documents. Since images are the most intuitive representation of ads to users, we assume that image-related features play an important role in prediction.

3. Exploratory Data Analysis

Team performed exploratory data analysis on the data set. Deal probability is our target variable in this study. It represents the rate of successful deal made between the buyer and selling for a particular item. Deal probability distribution plot can be seen in figure 1. Most of the items have low deal probability in this data set. Around 65 percent of items listed have extremely low deal probability and 5 percent of items have a deal probability of 0.8. It clearly showed that deal on this advertising platform was not performed well.

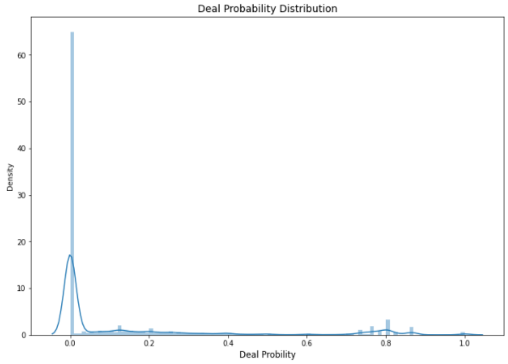

Figure 1. Deal probability distribution for the training data set

All numerical variables are not corrected as shown in figure 2 below. Surprisingly, deal probability and price are not corrected in this study which is kind of inline with our expectation that price may not be the key factor that affects the deal probability.

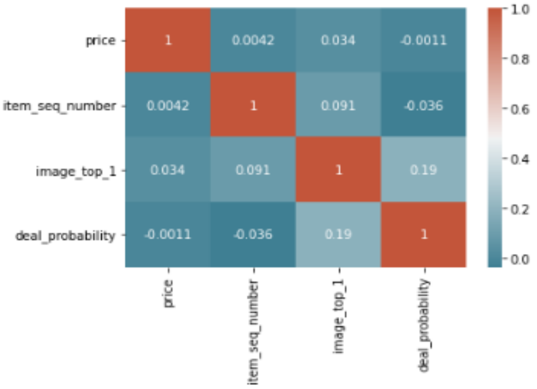

Figure 2. Correlation matrix for the training data set

Advertisement distribution throughout the region seemed to be fairly proportional shown in figure 3 below. The region with the most advertisements was Krasnodar Krai which took 9.41 percent of the total advertisements. On top of that, Sverdlovsk Oblast had 6.28 percent of total advertisements.

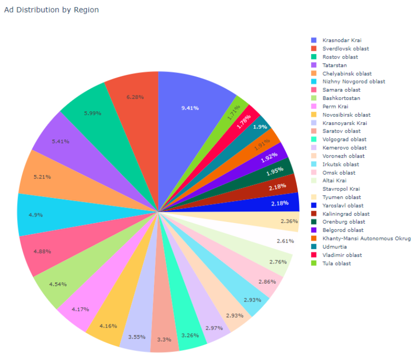

Figure 3. Ads Distribution by Region

Most of the region had a deal probability between 0 to 0.2 shown in figure 4 below. Orenburg Oblast,Bashkortostan and Stavropo Krai seemed to have slightly higher deal probability compared to other regions. High deal probability marked to be outlier clearly reflected the lower volume of high deal rate.

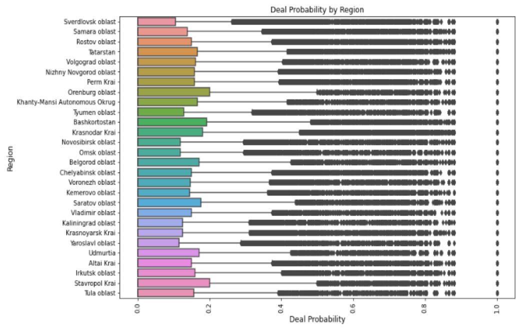

Figure 4. Deal Probability by Region

Figure 5 shows most of the users on the website are private which occupied 71.6 percent of the user group. 23.1 percent are from the company and 5.35 percent actually from the shop. This shows the website mainly for private users for online transactions which is typically the case for classified advertisement websites worldwide.

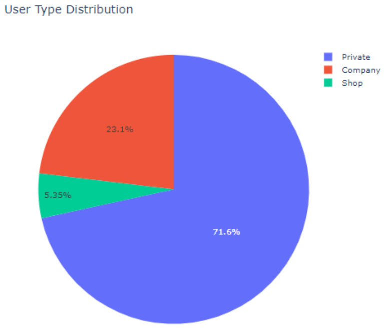

Figure 5. User Type Distribution

Without doubt, private users indeed had the highest deal probability among all user types in this study as reflected in figure 6. Use type distribution pie chart was clearly inline with the deal probability plot in this case.

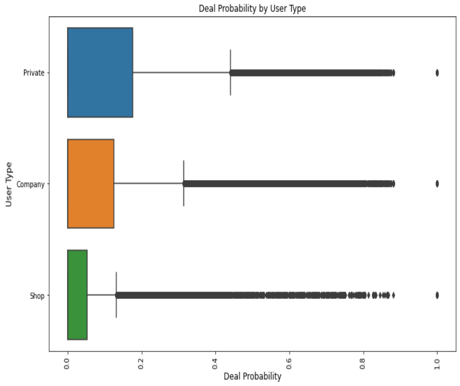

Figure 6. Deal Probability by User Type

Products that belong to personal belongings had 46.4 percent of the total advertisements which is the top category followed by Home/Garden category and Consumer electronic category shown in figure 7 below. These are categories that tend to have more advertisements for a classified website.

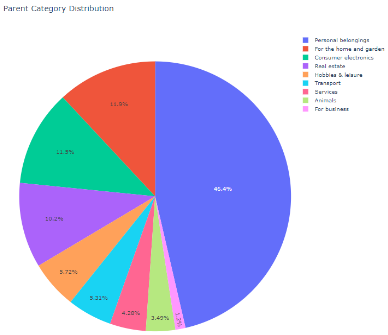

Figure 7. Ads Distribution by Category

Service category though had better and consistent deal probability which reflected on figure 8 below. Most of the categories' deal probability fall into the lower quartile range.

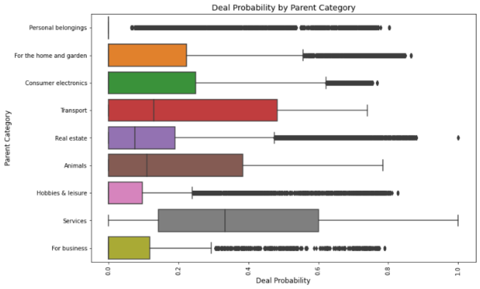

Figure 8. Deal Probability by Category

Another interesting fact is that advertisements without price had higher average deal probability as compared to advertisements with price labeled shown in figure 9. Private buyers and sellers could negotiate better offers offline in order to make the deal successfully which is similar to other platforms such as Carousell in this scenario. Avito website here may not be able to capture the deal information made in this way which could be a potential improvement needed in the future.

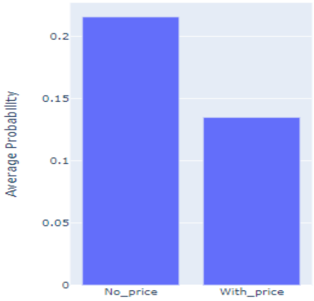

Figure 9. Deal Probability by Price

Advertisements that come with proper description definitely will have higher deal probability as reflected on figure 10 below. Detail and appropriate description do help to make transactions better in this case. Buyers would like to know as detail as possible for every product or service listed on the website.

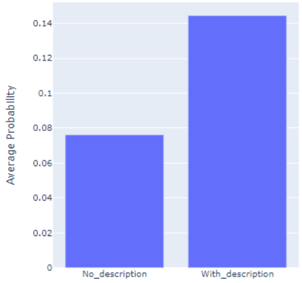

Figure 10. Deal Probability by Description

Lastly, another interesting fact is that advertisements that come without an image have a slightly higher average deal probability.

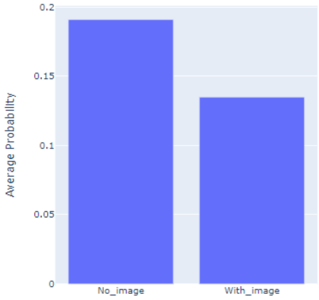

Figure 11. Deal Probability by Image

4. Data Preprocessing

Due to computing capacity, we only selected 10000 image samples and its corresponding structured and text data for the modeling. Also, since some of the ads did not have images, we included all none-image ads in the dataset. We get a total 212588 samples.

4.1 Read and merge Image Data

We use opencv to read images. Due to the large amount of images, 100,000 images are randomly selected. Team noted that images in different advertisements have different width and height. After converting the image into an array, the shape of the array will be different, which will cause challenges in the modeling part. Thus, we decide to unify the size of images and resize all images' shapes. The weight of images is 150 while the height of images is 200. After resizing, the shape of one image becomes (200,150,3). Finally, we get a dataframe of image data with the same size and merge it with structured data by image name, which is 'image' in the ad attribute dataset. We get a dataset with all structure data, text data and image data now.

4.2 Handle missing values

For structured data, only price has missing values. For most ads platforms, price is the compulsory information required to provide when posting ads. Here, we keep such assumptions of compulsory price and focus on those with price information. As price is related to so many factors like product types and new or old types, we do not handle it but drop all samples without price by list-wise method.

For text data, since information of advertisements is filled by users and some information is not mandatory for posting advertisements, we find there are null values in 'param_1', 'param_2', 'param_3' and 'description'. We fill these features with empty strings and these rows remain in the dataset.

For image-related data, some advertisements don't have pictures so their image-related data are null. 'image_top_1' is the classification code for the image, defined by Avito. We fill missing values by 0 to represent the ads without images. For image data, we fill nan with a zero matrix that has the same size with other image data.

4.3 Encode categorical features

In this part, we would like to transform categorical variables into numerical ones. Of all the variables, there are 10 categorical variables, which are listed in Table 1 in Appendix. We will first visualize the relationship between such variables and then try several encoding methods combined with domain knowledge. Before encoding, we first split train and test data to avoid data leakage.

Since city names are duplicated across regions, we combined city and region to identify specific cities. There are 28 regions and 1597 cities in total. We could not just do the one-hot encoding, which will lead to data sparsity. From figure 12, arranging the mean deal probability for each region in ascending order, we could find the deal probability is significantly different between certain regions, while some regions are with similar values. The red line is the mean deal probability of all training data. We consider using the mean of deal probability in each region to represent them, which refers to the transaction level per area. Also, we plot the mean price distribution by region, and we find that consumption level in Orenburg Oblast is much higher than others. Compared with the mean price level of all training samples, only two regions are higher than the mean level. It is possible that the wealth gap is large and the most wealth is concentrated in a few wealthy areas. Consumption level in areas is correlated to salary, economic development, citizen types and so on, which will also influence their deal probability in Avito. Thus, we also encode the region by mean price level of each town to represent the consumption level of each town. These two encoding methods are also used in the city variable. For the encoding in test data, we assume that the data distribution of the training set can be representative of the general cases. Therefore, we transfer the mean of train data to test data corresponding to each type of the variables.

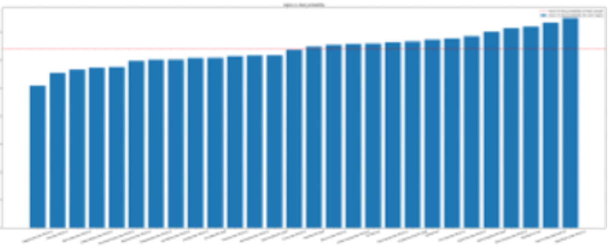

Figure 12. Deal Probability V.S. Region

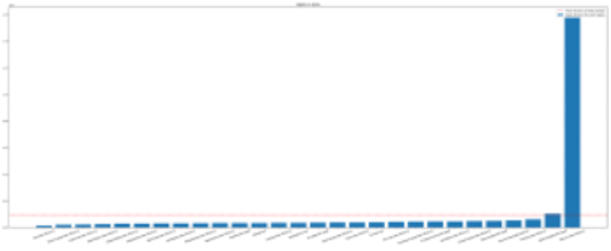

Figure 13. Price V.S. Region

Variable - parent_category_name and category_name is the product type in the ads. As parent_category_name only has 9 categories, I use one-hot encoding to transfer it. Also, the demand of different product categories on classified advertised platforms is different. Besides, price is highly correlated with product categories. For example, for some daily-use types of goods, although their price is low, it is seldom to make a deal on the classified advertising platform. That is, an iPhone must be more expensive than an apple. However, few people buy apples on such platforms. We could also get such insights in figure 14 and 15. Therefore, mean deal probability and mean price of each product type. In addition to one-hot encoding, I also use the above two methods – mean deal probability and mean price – to encode parent_category_name. As category_name has 47 different types, we only use the latter two methods to encode it, to control data sparsity.

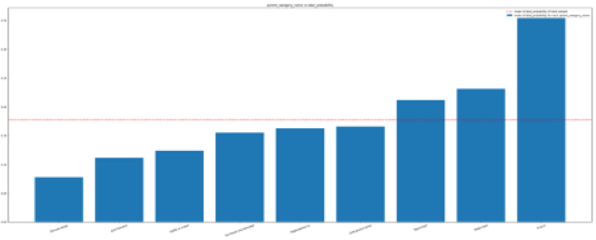

Figure 14. Deal Probability V.S. Parent_category_name

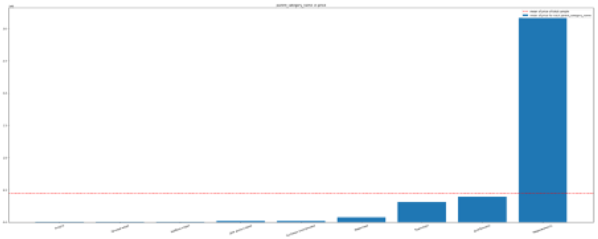

Figure 15. Price V.S. Parent_category_name

From figure 6 above, different user types have different deal possibility distributions. That is, private is the most likely to make a transaction, followed by the company, and the shop last. Therefore, we would conduct one-hot encoding for these three types of users. The same as before, we use the mean of deal probability and price for each class to encode them, due to the significantly different distribution between user types. For the mean price, we could gain information about pricing levels for different users.

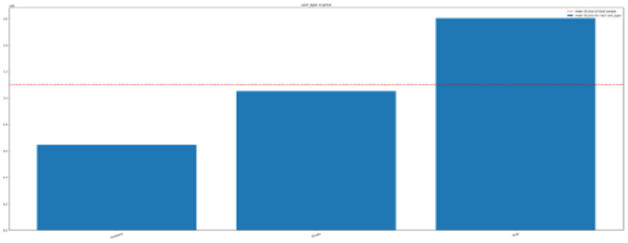

Figure 16. Price V.S. user_type

As mentioned before, param_1 with 340 types, param_2 with 243 types and param_3 with 723 types, which are the optional description information of products, could also be treated as categorical variables. These descriptions can be used as detailed classification of products. Also, variable-image_top_1 is about 2959 types of classification of ads image. We also use the above two methods – mean deal probability and mean price – to encode all of them.

Variable - activation_date is the date the ad was placed. In the dataset, the activation period is from March 2017 to April 2017. Therefore, we would like to extract the weekday, the day of week like Monday or Sunday, from the activation data. Since the number of weekday is not ordinary for our prediction problem, we still need to encode it. From figure 17, Wednesday and Saturday have high deal probability while the deal price is much higher than others on Wednesday. To capture such different patterns of deal probability and price, we also use the above two methods – mean deal probability and mean price – to encode weekday.

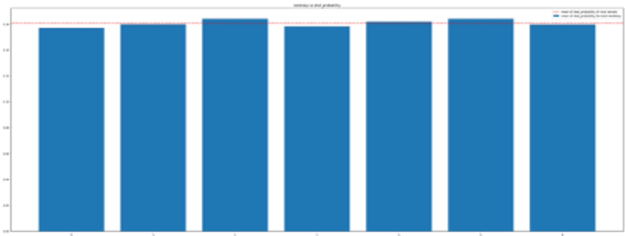

Figure 17. Deal probability V.S. weekday

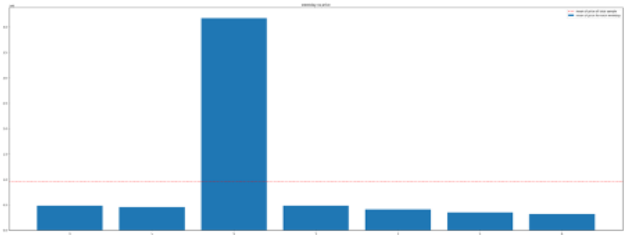

Figure 18. Price V.S. weekday

The summary of the encoding methods is shown in Table I in Appendix. After encoding, we get 34 numerical variables now.

5. Feature Engineering

One of the objectives is to understand how the different quality of image and text are affecting the final deal probability. Feature engineering for them is necessary because it is the way that can help interpret what kind of images and text are more attractive or have higher quality**.**

5.1 Text Features

Textual data is composed of three parts: param, title, and description. Text 'description' that we mainly focus on provides the most comprehensive information. The features we chose to depict 'description' contain three parts: 1. Basic features, like the number of characters and words, etc; 2. Part-of-Speech tagging features, like numerals, adjectives, etc. 3. Sentiment Analysis features, like polarity scores, neutral scores, etc. The reason for choosing these three dimensions is that we assume the length of a sentence (can reflect how much information has been provided), the rendering (how the users describe their products), and the emotions conveyed by 'description' may make a difference when people read it and will then reflect on the final deal probability. Below is the table of complete text features.

Table 3: Text features description

Type	Feature Name	Feature Description
Basic Features	char_count	Number of characters
	word_count	Number of words
	word_density	char_count /(word_count+1)
	punctuation_count	Number of punctuations
	title_word_count	Number of words that begin with a capital letter in title
POS Features	NUM	Number of numerals
	A	Number of adjectives
	ADV	Number of adverbs
	S	Number of nouns
	V	Number of verbs
Sentiment Analysis Features	polarity_score	The combination of the next 3 scores
	neutral_score	Measures how neutral the sentence is
	negative_score	The negativity of a sentence
	positive_score	How positive the sentence is

5.2 Image Features

As the image vectors are both memory consuming and hard to explain, we derived some features to access image quality in hope to provide more information to the model.

Table 4: Image features description

Type	Feature Name	Feature Description
Size	height	Height of the image
	width	Width of the image
Color	color_mean, color_std	Mean and standard deviation of all pixels
	r_mean, r_std, b_mean, b_std, g_mean, g_std	Mean and std of Red/Blue/Green pixels (compare color variations between images for RBG)
	h_mean, h_std, s_mean, s_std, v_mean, v_std	Mean and std of Hue/Saturation/Value (compare different color intensities on top of RBG)
Bright-ness	exposure	measure the exposure level: the perceived brightness
	contrast	image contrast in grayscale
Clarity	edge_score	the number of edges, used to measure pixel variation
	blurriness	Number of adjectives

These features are included into the training and testing data as numerical data.

6. Deep Learning Model

When running deep learning models, we encounter computing capacity issues. Our computers are not able to deal with 212588 samples. As a result, we randomly sample 60% data from our train set and test set respectively to complete model building. What's more, we detect that there are null values in image related features after feature engineering because some advertisements don't have images. We fill these null values with -1 before training our model.

Recurrent Neural Network (RNN) is an ideal model to do Natural Language Processing (NLP) since it can recognize the sequential characteristics of text and therefore it can understand text better. Meanwhile, RNN can handle not only short text but also long text, which is an advantage compared with Convolutional Neural Network (CNN) in NLP. Researchers study different deep learning models in NLP and find that RNN performs well and robustly in a broad range of tasks except when the task is essentially a keyphrase recognition task as in some sentiment detection and question-answer matching settings[1]. Our target is to predict deal probability through understanding the descriptions of products in advertisements. RNN will perform well in this scenario.

CNN is the mainstream method to deal with images. Images have high dimensionality. For traditional neural networks, images are very difficult to deal with due to the computing capacity issues. However, CNN not only can automatically detect the important features without any human supervision but also is computationally efficient. CNN is very effective in reducing the number of parameters without losing on the quality of models. As such, CNN is very suitable for image processing.

Generally, our techniques are that we use RNN to handle text data, MLP to handle numerical data and CNN to handle image data. Then, we will combine outputs from three models and build MLP on combined outputs to do final prediction. Our work flow can be illustrated by figure 19.

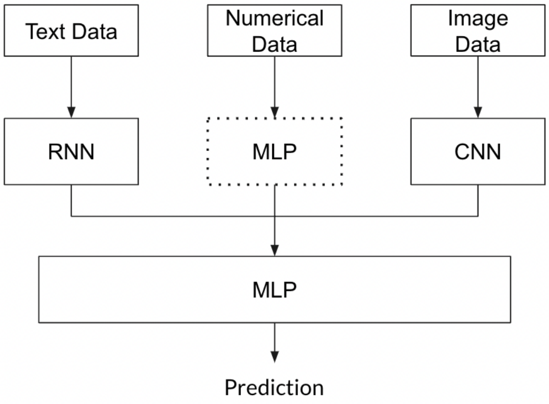

Figure 19. Work Flow of Deep Learning Model

6.1 Word Embedding

Word embedding is a fundamental part of NLP. Through word embedding, words that have similar meaning will be grouped together and have close vectors in vector space. We combine text in param_1, param_2, param_3, title and description and rename it as full text. Then we implement text cleaning by removing http links, html tags, special characters, stopwords and punctuations. Also, we implement lemmatization. Typically, pre-trained word embedding matrices will have better performance than self-trained word embedding matrix in word representation. Here, since the text is Russian, we use pre-trained word embedding from fasttext, which is trained for Russian on Common Crawl and Wikipedia. This model was trained using CBOW with position-weights, in dimension 300, with character n-grams of length 5, a window of size 5 and 10 negatives [2]. First, we keep all words in our dataset to do word embedding. Then, we convert the text to index. After that, we unify the length of the text through padding. The unified length is 400. Words not found in the embedding matrix will be all-zeros.

6.2 Model 1

In this model, we have an embedding layer and LSTM architecture in RNN. Then, we input numerical data directly without any processing. For image data, we build 2D CNN with 3 convolutional layers and use max pooling right after each convolutional layer. The activation function in three convolutional layers is relu. We concate outputs from these models and input them into MLP. We first normalize these inputs and then build a hidden layer with activation function to be relu and with the he uniform method to set the initial random weights of layers. Also, we set dropout layers to control risk of overfitting. The dropout rate of this hidden layer is 0.4. The second hidden layer has the same activation function and initializer with 64 nodes, only half of the first hidden layer. We use sigmoid as the activation function in the output layer since we need a scaled regression target from 0 to 1 to represent probability. The optimizer is rmsprop, loss is mse and metrics is rmse. Figure20 (Appendix A) shows the architecture of model 1.

The train RMSE of model 1 is 0.30 and the test RMSE is 0.3. The number of trainable parameters in this model is 13,145,849. That’s a quite big number. Too many parameters result in longer training time. What’s more, we do not process numerical data before inputting them to MLP for prediction. We want to improve these two points in a later model.

6.3 Model 2

In model 2, the architecture is quite similar to model 1. We add MLP after the input layer of numerical data. Also, we change the filter size of CNN to (16, 32, 64). With smaller filter size, the dimension of CNN after flattening is greatly reduced and thus, trainable parameters of the whole model are greatly reduced. Beside these, we change the optimizer from rmsprop to adam. Figure 21(Appendix A) shows the architecture of model 2.

The train RMSE of model 2 is 0.23 and the test RMSE is 0.25. The number of trainable parameters in this model is 556,297. We greatly reduce the model size and improve the performance by 0.05. We find that the dimension of output of RNN is different from other output’s dimensions. We want to unify the output dimension of different type data in a later model.

6.4 Model 3

In model 3, we add additional hidden layers to make the number of nodes of the output layer 4, which is the same as MLPs and CNNs. After concatenating, the dimension becomes 12. Given the small dimension, we reduce the number of layers in MLP for prediction. Figure 22 (Appendix A) shows the architecture of model 3.

The train RMSE of model 3 is 0.23 and the test RMSE is 0.24. The number of trainable parameters in this model is 540,365. We slightly improved the performance by 0.01.

7. Why Deep Learning Model Did Not Perform Well

The RMSE of deep learning models is around 0.25. Since our target variable is from zero to one, the performance is not very good. Next, we will analyze why deep learning models do not perform well in this project.

First, the pre-trained word embedding matrix is not suitable for this project. Since the text data is Russian, we can only choose fasttext as embedding. However, fasttext is trained from Common Crawl and Wikipedia. Text in these two websites may have different scenarios from descriptions in Avito advertisements. Some words not found in the embedding matrix also remain zeros. That will bring negative effects on later modeling training, especially in RNN training.

Second, due to computing capacity, we compress images a lot. That may result in misleading information in images and our models may fail to capture appropriate information in image data. Also, unifying the pictures into one size will cause some pictures to stretch and deform, resulting in confused information of image data. As a result, images data may not be very helpful in our models, even may mislead the model training.

Third, our target variable is derived from Avito’s prediction model and cannot be verified with certainty. It is possible that when getting target variables, companies only use linear models. Therefore, deep learning models are too complex and twist the relationships between features and target variables.

Fourth, due to our encoding method, we use target variables to encode categorical variables and transform them into numerical variables. As a result, this encoding method will enhance the linear relationship between features and target variables, making nonlinear systems of neural networks less effective.

8. Traditional Machine Learning Model

8.1 Regression Results

Traditional regression models have faster training speed and are easier to interpret, therefore, we tested different regression models on top of the deep learning models. To make the model more explainable, original text and image data are excluded, only the numerical data are inputted into the model, which includes the encoded data and the features newly generated from texts and images.

Table 5: Model Results

Model	**Train RMSE**	**Test RMSE**
Linear Regression*	0.23	0.24
Elastic Net*	0.25	0.26
XGBoost (gblinear)*	0.23	0.24
Random Forest**	0.08	0.24
XGBoost (gbtree)**	0.20	0.24
LGBM**	0.21	**0.23**

*linear models, ** non linear models

The result shows that traditional models have comparable performance to the deep learning modes, Light GBM, being the best performing model, has better performance than the later. The simple linear models also have relatively good performance, which add weight to our guess that the relationship between the main features and deal probability is a linear one.

We also performed hyperparameter tuning on the best performing model, namely Light GBM, but it only increased the test RMSE by a small margin. This further implies that a simple linear model may be a good choice for its fast training speed and interpretability.

8.1. Interpretation of Model Outputs

To understand the effects of different features on the deal probability, we plotted the feature importance (Figure 23,24,25 in Appendix A) using coefficients for linear models and SHAP for LGBM. As the categorical features from the original data are not subjected to change for the sellers in Avito, the image and text features are more useful in providing actionable insights. However, we realized that none of the text features are important across all the models, whereas some image features rank quite high in terms of importance.

Table 6: Important image features

Types	Features
color	h_mean, h_std, s_std
	r_mean, b_mean, g_mean, colour_mean
size	width,height
clarity	blurriness

9.Evaluation and Conclusion

9.1What make up good product images?

The highly ranked image features provided us with the information on the correlation between some image attributes and the deal probability.

For the colors, mean for red/blue/green all have negative coefficients while overall pixel mean has positive coefficients, suggesting that images that are mono color or strong in one color tone may not be preferred. The positive coefficients for h_std and s_std suggests that images with larger color variation and contrasts are better.

For size, height and width suggests that larger images are preferred, which aligns with the common intuition.

For clarity, less blurriness is preferred, but it is less important than use of color. This may be due to the fact that users are not able to detect small differences in blurriness given the small display area on the website.

On top of the features we generated, Avito's classification of images, image_top_1, is also an important factor in the xgboost regression. Therefore, we compared the images from the image_top_1 class of highest deal probability and the class with the lowest deal probability. On top of the effect of colors, size and blurriness as mentioned above, the two groups also differ largely in edge score, which suggests that images that have higher pixel variation may be preferred. This is also consistent with our intuition as such images allow us to better differentiate the different objects and also different the subject from the background.

9.2 Why are text features not important?

As the plot of feature importance shows, text features do not play a crucial role in the prediction of deal probability. There are two possible reasons.

First of all, the design of the Avito website diminishes the role of text. Textual data is placed in an unobtrusive position, which is not easy to be seen by users. The homepage of the website only presents the picture, name, price, user address, and release date. There is no description of the product or any textual data. When we click on a product and go to its details page, a giant image appears on the first screen and the user has to scroll down to the bottom of the screen to see the description.

Second, based on the feature importance results, we found that 'mean_dp_param _1' which represents the average deal probability for each param_1 is important. As mentioned above, param_1 is extracted automatically by Avito from the description. It provides labeled characteristics of a product, for example: 'for girls',' for boys', etc. Then we divided the description into two groups based on 'mean_dp_param_1':

'High_dp_param_1' means this group of param_1 has a higher deal probability and vice versa.

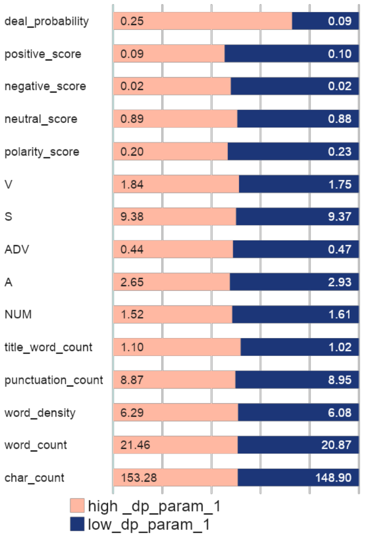

Figure 26. Text Feature Statistics Based on 'mean_dp_param_1'

Figure 26 shows that there is no significant difference in the text features between the two groups, indicating that it doesn't matter how well the description itself is written, as long as it covers the needed characteristics.

9.3 Business Insights

With our findings in the previous sections, we would provide the following suggestions to the sellers.

Firstly, having a good product image is more important than having a good product description. A good product image is defined in section 9.1.

Secondly, external features such as city, product category and user types can also greatly affect deal probability. Section 3 shows relationships of the different external features with the deal probabilities. Therefore, sellers need to form more realistic expectations given these external conditions.

Lastly, Avito's internal classification of product image and description, the features including (image_top_1, para_1, para_2, para_3), are also important in predicting deal probability. That suggests that Avito has rich internal data and is able to accurately classify high selling items from the low ones. Given the classification codes, the sellers can mimic the characteristics of images and descriptions of the products in the high deal probability class. On the other hand, the sellers can also work with the website to gain more information on improving sales.

9.4 Future improvements

The areas of improvements for our project are listed as follow:

1. Our current model shows a strong linear relationship between the features and the deal probability. This is likely due to the fact that most of the important features, such as image_top_1, para_1 and user_type, are encoded using mean of deal probability. With this, the model makes the assumption that products of the same image class or same user type will have similar deal probability. Therefore, the model will not be effective for new products with new image class, para_1, etc. The assumption will also be challenged if there is a frequent change in user buying pattern and preferences. Therefore, we can work on generating more features relating to the image and product itself for more robust prediction.
2. We were not able to utilize all the data provided by Avito due to memory constraint, using more power devices and parallel processing will allow us to include more information into the model.
3. We can explore more encoding methods to compare the changes brought to the model and select the best of all.

Appendix A

Table I: Summary of encoding methods

Variable name	Encoding method	New variable
region	mean of deal probability in each region	mean_dp_region
	mean of price in each region	mean_p_region
city	mean of deal probability in each city	mean_dp_city
	mean of price in each city	mean_p_city
parent_category_name	One-hot encoding	all parent_category_name
	mean of deal probability in each parent_category_name	mean_dp_par_category_name
	mean of price in each parent_category_name	mean_p_par_category_name
category_name	mean of deal probability in each category_name	mean_dp_category_name
	mean of price in each category_name	mean_p_category_name
param_1, param_2, param_3	mean of deal probability in each type	mean_dp_param_1, mean_dp_param_2, mean_dp_param_3
	mean of price in each type	mean_p_param_1,mean_p_param_2,mean_p_param_3
user_type	One-hot encoding	all user_type
	mean of deal probability in each user_type	mean_dp_user_type
	mean of price in each user_type	mean_p_user_type
image_top_1	mean of deal probability in each image_top_1	mean_dp_image_top_1
	mean of price in each image_top_1	mean_p_image_top_1
activation_date	mean of deal probability in each weekday	mean_dp_weekday
	mean of price in each weekday	mean_p_weekday

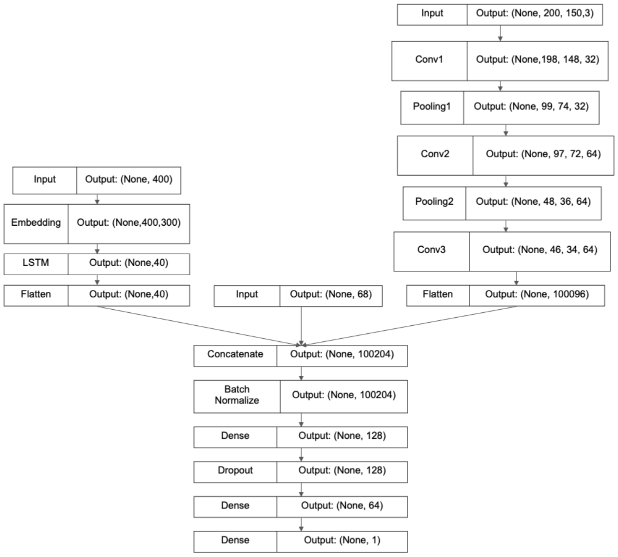

Figure 20. The First Version of Deep Learning Model

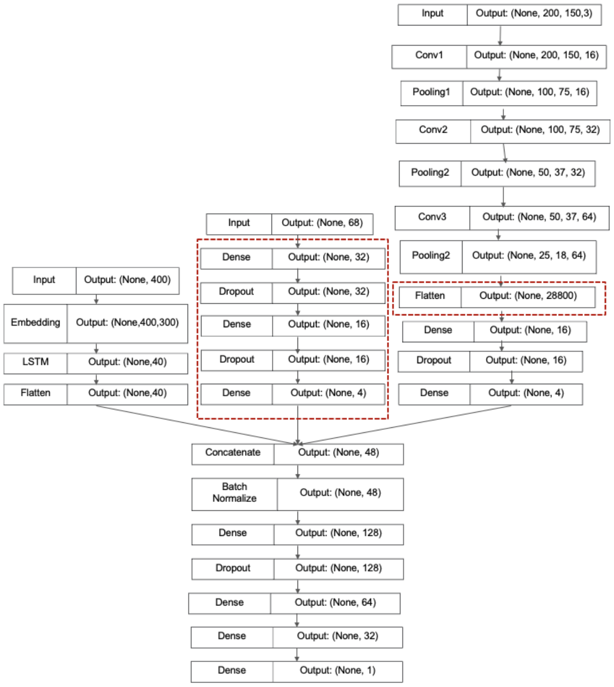
Input | Output: (None, 200, 150,3)
Conv1 | Output: (None, 200, 150, 16)
Pooling1 | Output: (None, 100, 75, 16)
Conv2 | Output: (None, 100, 75, 32)
Pooling2 | Output: (None, 50, 37, 32)
Conv3 | Output: (None, 50, 37, 64)
Pooling2 | Output: (None, 25, 18, 64)
Flatten | Output: (None, 28800)
Dense | Output: (None, 16)
Dropout | Output: (None, 16)
Dense | Output: (None, 4)
Input | Output: (None, 68)
Dense | Output: (None, 32)
Dropout | Output: (None, 32)
Dense | Output: (None, 16)
Dropout | Output: (None, 16)
Dense | Output: (None, 4)
Input | Output: (None, 400)
Embedding | Output: (None,400,300)
LSTM | Output: (None,40)
Flatten | Output: (None,40)
Concatenate | Output: (None, 48)
Batch Normalize | Output: (None, 48)
Dense | Output: (None, 128)
Dropout | Output: (None, 128)
Dense | Output: (None, 64)
Dense | Output: (None, 32)
Dense | Output: (None, 1)

Figure 21. The Second Version of Deep Learning Model

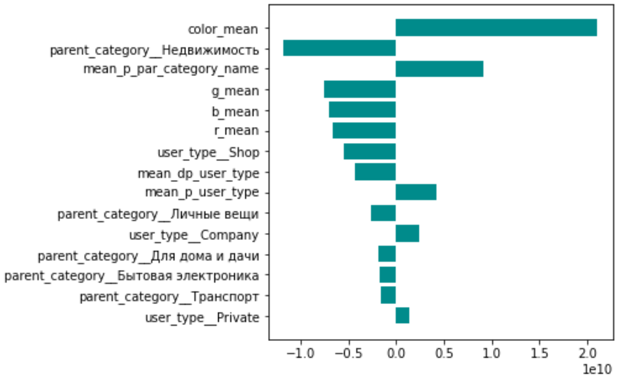

Figure 23. Coefficient Plot for Linear Regression

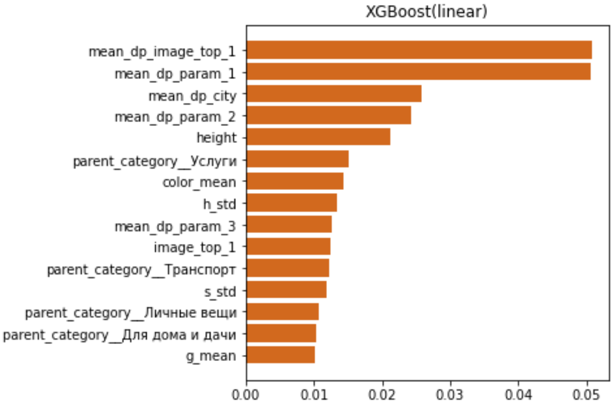

Figure 24. Coefficient Plot for XGBoost (gblinear)

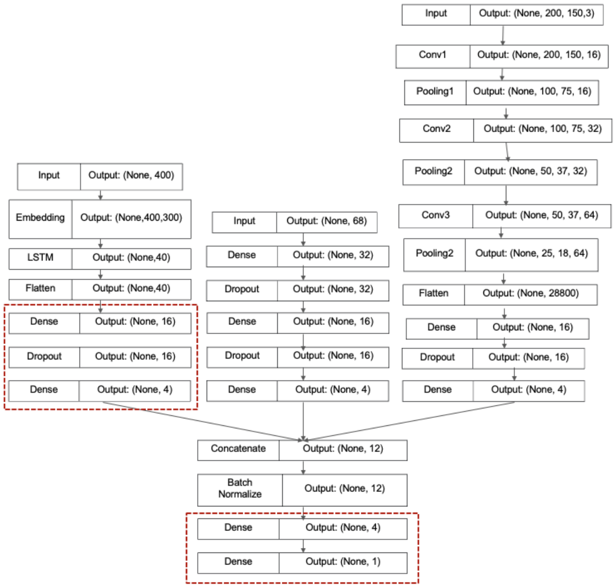
Input | Output: (None, 200, 150,3)
Conv1 | Output: (None, 200, 150, 16)
Pooling1 | Output: (None, 100, 75, 16)
Conv2 | Output: (None, 100, 75, 32)
Pooling2 | Output: (None, 50, 37, 32)
Conv3 | Output: (None, 50, 37, 64)
Pooling2 | Output: (None, 25, 18, 64)
Flatten | Output: (None, 28800)
Dense | Output: (None, 16)
Dropout | Output: (None, 16)
Dense | Output: (None, 4)
Input | Output: (None, 68)
Dense | Output: (None, 32)
Dropout | Output: (None, 32)
Dense | Output: (None, 16)
Dropout | Output: (None, 16)
Dense | Output: (None, 4)
Input | Output: (None, 400)
Embedding | Output: (None,400,300)
LSTM | Output: (None,40)
Flatten | Output: (None,40)
Dense | Output: (None, 16)
Dropout | Output: (None, 16)
Dense | Output: (None, 4)
Concatenate | Output: (None, 12)
Batch Normalize | Output: (None, 12)
Dense | Output: (None, 4)
Dense | Output: (None, 1)

Figure 22. The Third Version of Deep Learning Model

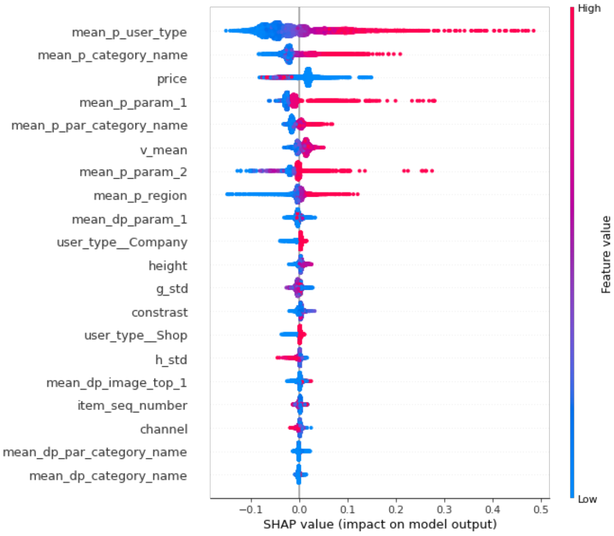

Figure 24. Feature Importance for XGBoost (gblinear)

References

[1] Wenpeng Yin, Katharina Kann, Mo Yu, Hinrich Schütze, Comparative Study of CNN and RNN for Natural Language Processing, 2017.

[2] Anon. Word vectors for 157 languages · fasttext. Retrieved April 24, 2022 from https://fasttext.cc/docs/en/crawl-vectors.html

[3]Rich et al. 2021. Keras: Multiple inputs and mixed data. (July 2021). Retrieved April 24, 2022 from https://pyimagesearch.com/2019/02/04/keras-multiple-inputs-and-mixed-data/#pyis-cta-modal

[4]MridulMazumdar.MridulMazumdar/avito_demand_prediction: Prediction of deal probability using advertisement data. Retrieved April 24, 2022 from https://github.com/MridulMazumdar/avito_demand_prediction

[5]Shivamb. 2018. Ideas for image features and image quality. (May 2018). Retrieved April 24, 2022 from https://www.kaggle.com/code/shivamb/ideas-for-image-features-and-image-quality

[6]Marian Stefanescu. 2022. Measuring & Enhancing Image Quality attributes. (January 2022). Retrieved April 24, 2022 from https://towardsdatascience.com/measuring-enhancing-image-quality-attributes-234b0f250e10

[7]somang1418. 2019. Tuning hyperparameters under 10 minutes (LGBM). (March 2019). Retrieved April 24, 2022 from https://www.kaggle.com/code/somang1418/tuning-hyperparameters-under-10-minutes-lgbm/notebook

[8]Anon.Retrieved April 24, 2022 from https://www.nltk.org/api/nltk.html