Machine Learning Engineer Nanodegree

Capstone Proposal

Ratna Bearavolu

May 21, 2018

Proposal

For my capstone project, I have chosen to work on a competition posted on kaggle.com - Avito Demand Prediction challenge (https://www.kaggle.com/c/avito-demand-prediction)

Domain Background

Avito Demand Prediction competition wants to predict the likelihood/probability that an advertisement that is posted on their social advertisement website sells. Based on various details of an advertisement posted, the company would like to give feedback to the seller on the likelihood that the ad will sell. This will help set expectations of the seller, so they are not frustrated with the website if the item does not sell or give the seller an opportunity to review the the posting, to help improve the likelihood of the item selling (e.g. may be include a detailed description, better picture, etc)

I am interested in solving this problem because of the nature of the inputs that need to be analyzed - tabular data, textual data and image data to predict the likelihood of selling. Typically, in the homeworks/assignments that I have worked on so far, the data was of single type (either tabular data or textual data or image data). When I saw that this problem had a mix of inputs, it piqued my interest in how I would address this issue and how I would set up this problem. I feel that this problem will give me valuable experience in applying various ML algorithms. Also at a personal level, in the past, there were many of my ads (that I posted on various social advertisement websites) that never sold. I would have definitely welcomed any feedback on the likelihood of the ad selling from any one of these websites.

Problem Statement

Given an advertisement, i.e, information on item to be sold, such as description, price, image, etc, a '***deal probability***' should be predicted by the system. Deal probability is a continuous variable ranging between 0.0 to 1.0. Higher probability will indicate greater likelihood that the advertisement will sell and lower probability will indicate less likelihood that the advertisement will sell the item.

Datasets and Inputs

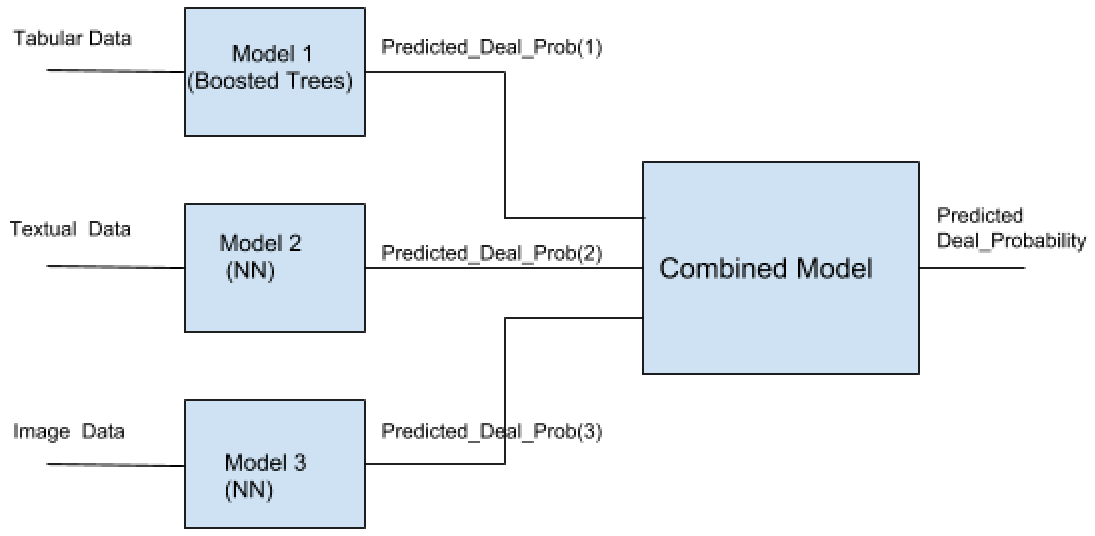
The competition data has been provided by Avito (https://www.kaggle.com/c/avito-demand-prediction/data). Of interest for my capstone proposal are:

- train.csv - Training data.
    - 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.
- train_jpg.zip - Images from the ads in train.csv.

As the published test.csv does not have the actual target values (as the competition is active), I will not be able to use the test.csv to rate the predictions of my model(s). I will split the given training data to be used for both training and testing. Also, depending on the resource availability (memory, computing power), I may use a subset of the data for training purposes. Train.csv has 1.5 Million rows of data. Each row indicates an ad item that was published by the user on Avito website. Not all rows have description and not all rows have an associated image (i.e. the image of the item the user is trying to sell). The size of the file is 908MB. The size of train_jpg.zip is ~40GB.

Solution Statement

As the data set is labeled and target variable is continuous, I will be using supervised learning algorithms to design the model (e.g. Boosted Tree Regression and/or Neural Net Regression techniques). Also, as the input is primarily of three types of data (tabular data, textual data and images), I plan to build three different models for each type of input and then use another regression model to combine the outputs of the three models to get the final prediction.



For Model 1, the input (X_train) will be all the columns found in train.csv, except the ‘description’ column and the label (y_train) will be the corresponding ‘deal_probability’ value. I will be using either XGBoost or DecisionTreeRegressor package from python to build and train the model. Similarly for Model 2, the X_train will be the vector representation of the ‘description’ and y_train will be the corresponding ‘deal_probability’ from train.csv. For Model 3, X_train will be the matrix representation of the image and the corresponding ‘deal_probability’ from train.csv. For all three models, the output will be the deal_probability value given their respective X_train input.

As the next step in the process, the combined model will take the outputs of these three models (individual deal_probability values) and combine them to predict the final value for the target variable. As I am still in the exploring stage, I am unclear as to what may comprise of the combined model. I am planning to either use Linear Regression or Neural Net or DecisionTreeRegressor to see which will give me the ‘best’ prediction.

Benchmark Model

As this an active Kaggle competition, there hasn’t been a published model yet. So, as an benchmark model, I intend to use a simple linear model based on tabular data. I will compare the results of only using tabular data with the results of the combined model that will use tabular, textual as well as image data.

Evaluation Metrics

As the competition is being evaluated on the Root Mean Squared Error (RMSE) score, I intend to use this metric for evaluating my models.

RMSE is defined as

$$RMSE = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(target_{i\ actual} - target_{i\ predicted})2}$$

Where n = the number of testing data points, target_actual = actual target variable value (between 0.0 and 1.0), target_predicted = predicted target variable value (between 0.0 and 1.0). The goal would be to build a model that has a value of RMSE close to 0.0

Given the nature of the target variable (continuous), I think RMSE is a good evaluation metric.

Project Design

Data Exploration:

I will begin by exploring the data using both code and visualizations. My primary goal would be understand how various attributes (in train.csv) relate to one another and their correlation to the target variable. The exploration analysis will help me in understanding feature relevance and provide me good insight into trimming dimensions off, especially when I convert the categorical data into one-hot encoded columns.

Data Preprocessing:

- Prepare the data in train.csv.

- For Model 1:
    - Classify the number of columns as categorical, ordinal, continuous, binary
    - Check for missing values - calculate the percent missing
        - Drop columns that have “high” percent missing (may be > 75% of data is missing. However, I may revisit this and fill in a “special” value and use this data for training).
        - Columns with some percent missing:
            - For ordinal value column - use the mode to fill the data gaps
            - For continuous value column - use the mean to fill the data gaps
            - For categorical columns - create a placeholder category and mark them accordingly
        - Convert categorical columns
            - If the number of categories is high, don’t want to use one-hot encoding to avoid the curse of dimensionality. From the Data Exploration step above, I may start with the top 5-10% of the categories, build the model and see how it performs. Via some experimentation, I will determine the percent of categories I will use to train on.
        - Handle outliers in continuous columnar data.
- For Model 2:
        - Model 2 will use the ‘description’ column of train.csv data. I am thinking of using either word2vec approach or tf-idf to get the vector representation of the textual data. Again, I plan to try both the models and choose the model that gives best performance.
- For Model 3:
    - Model 3 will use the images data. I will use the Keras package to read them and convert them into 3d array representations.

After the above steps, as outlined in Solution Statement section, for model 1, I intend to use Gradient Boosting Regression technique to predict the target variable value. For model 2, I plan to use Keras package to develop sequential Neural Net (NN) model. As I am dealing with a continuous target variable, I will have my last layer of the net as a single neuron with a sigmoid activation function. Similarly for model 3, I plan to use Keras package to build a CNN model, with the last layer being a single neuron with sigmoid activation function.

Once I have output from model1, model2, model3, I plan to use NN or Linear Regression or another GradientBoostingRegressor to figure out how to best combine the outputs to give me the final prediction. For all my models, I plan to use RMSE as the evaluation metric.