Machine Learning Engineer Nanodegree

Capstone Proposal

Yang Yang
June 19th, 2017

Domain Background

This is a kaggle competition sponsored by Sberbank, Russia's oldest and largest bank. It challenge kagglers by making predictions of house price using house transaction data and macroeconomic data. A good algorithm will mean a lot for both the bank and its customers, with which Sberbank can provide better service and analysis for their customers and customers can be more confident whether to lease or purchase a house. And tons of reseach has been done to understand what factors influence house prices, but, in a more finance oriented way. In this project, I will try to predict the house price using data.

Problem Statement

In this competition, we will predict the house price with two datasets, one is about the properties of house itself. The other is about the macroeconomics of Russia in a certain period of time. The problem is straightforward, more specifically, we will use more than 400 both numeric and categorical features to predict a numeric target, the house price.

Datasets and Inputs

Data has already been provided by Kaggle. It mainly consists of 2 parts. The first part is train and test files. The train set is 30000*200 big and the test set is 7000*200 and the macro data set is 2500*100. more than 90% of features are numeric features and the left are categorical features. The whole data set has many null values and outliers and this should be a real problem to achieve good performance. They are information about individual transactions. The rows are indexed by the "id" field, which refers to individual transactions (particular properties might appear more than once, in separate transactions). These files also include supplementary information about the local area of each property.

The other part is macro file, which is data on Russia's macroeconomy and financial sector. For more details please refer to the README.md file.

Solution Statement

Firstly I will process the data, including filling null values, removing outliers, selecting features. And then I will build some single models and tune each of them with grid search cross validation. To increase the diversity among model I will try random forest, adaboost, neuron nets, and of course, xgboost. The performance of models can be evaluated by local hold out set and the Kaggle public leader board. In the final stage, I will ensemble a bunch of models with decent performance by stacking and bagging. the method will usually yield a slightly better performance.

Benchmark Model

XGBoost is fast, invariant to scale, handles missing values and has good performance. So it is a perfect choice to start as our benchmark. Since it’s a competition, we’d better start off from a higher point. So, just encode all the data into numeric and feed it into XGBoost, we will have our benchmark model.

Evaluation Metrics

As for the evaluation, we will use RMSLE (Root Mean Squared Logarithmic Error). This is quite similar to RMSE but it gives less punishment on large predicted values. This error will be calculated on my local cross validation folder, local hold out set, and Kaggle leaderboard (a place where you compete your solution with other data scientists).

Project Design

Firstly, explore the data. In this process, I will get familiar with the underneath meaning of each feature by reading the definition document and plots some graph to see the fluctuation trend, correlation, distribution.

Secondly, build some single models. This is not an easy job and sometimes we might not put too much hope on the ensemble part since single model can also bring us to the top of leaderboard with some awesome feature engineering and model tuning. So, in this stage, I will process the data by filling null values, detecting outliers and get a feel of the messy data. I will improve the XGBoost performance by tuning parameters using grid search cross validation. Other models like neuron nets, random forest tree, extra tree, adaboost are all under consideration.

Lastly, ensemble all the appropriate models. Detailly, I will code a class for automating the whole process: generate of-out-sample data from train set and generate predicted data on test set. These my first level features that will be feed to a second level model(should be XGBoost still).

To sum up, this is what I think I will do for now, but all others methods posted on the kaggle forum will all be concerned to help me climb up a few more rankings on the leaderboard.