Machine Learning Engineer Nanodegree

Capstone Proposal for AMES house pricing Kaggle competition

Jean-Christophe (JC) PINCE
January 18th, 2017

Proposal

Domain Background

Transactions volume on real estate market in the US only is above 350 billion dollars every year (see http://www.colliers.com/-/media/files/marketresearch/unitedstates/2016-capital-flows-reports/2016-1h-cap-flows-market-report.pdf). This is a huge market with many competitors addressing mostly anyone in the world.
Buying a house is a major step in the life of anyone and requires the buyer to think thoroughly the pros and cons of the house he or she plans to buy whatever what his or her budget is.
There are a lot of professionals in this market allowing the buyers to consider as many houses as possible given their criteria such as the house location, its size, the number of bedrooms but also the look and feel and many other features.

Problem Statement

Given the size of the market and the extremely careful approach of the buyers, estimating properly the price can make a huge difference for any professional. Being able to estimate the right price is important since if a house is overestimated, it will never get sold and if it is underestimated, the seller and selling agency will loose a lot of money.
Predicting the price of a house is a complex task requiring a lot of experience to estimate the different features such as the house location, the number of rooms, size of the house and many other features which have a less evident impact on the price such as the roof style for example

and it is addressable with machine learning.
Zillow and many others already understood that and are using machine learning to estimate the price of the houses on their site; their estimation is an important factor for the house buyer since it gives him or her some confidence that the price is right.

Datasets and Inputs

The Kaggle competition 'House Prices: Advanced Regression Techniques' uses a dataset provided by Dean De Cock: The Ames Housing dataset (http://www.amstat.org/publications/jse/v19n3/decock.pdf). This dataset was compiled by Dean for use in data science education. It's an incredible alternative for data scientists looking for a modernized and expanded version of the often cited Boston Housing dataset.

It is describing the sale of individual residential property in Ames, Iowa from 2006 to 2010 and comprises 79 features with invaluable tiny details such as the type of roof or the proximity of the house to arterial street and many others allowing to fine tune the estimation to a great extent. It also comprises a description of the features.
The 79 features are both numerical and categorical and mainly describe with many details:
- The type of property,
- The size of the different parts of the property (house, basement, garage…),
- The number of principal rooms (bedrooms, bathrooms…),
- The important features such a pool, the number of fireplaces…,
- The neighborhood

Solution Statement

After a deep look into the Kaggle solutions for regression problems, I chose to use a stacking machine learning solution. Largely inspired by Faron's description on the forum (https://www.kaggle.com/getting-started/18153) and the excellent Kaggle ensembling guide (http://mlwave.com/kaggle-ensembling-guide/).

The idea is to split our global model in two or more layers. Each layer should extract different information on the dataset and generate a prediction that will be used as an input to the second layer. The second layer can then make its predictions using the first layer output but also optionally using a subset of the input features to make new predictions. Those predictions can then be used by a next layer until the final layer. Finally, the predictions of the final layer can be averaged or combined to generate the final prediction.

Stacking gives the best results when combining models extracting different information from the data. I plan to use different models (tree based and linear models; namely XGBRegressor, RandomForestRegressor, GradientBoostingRegressor, ExtraTreesRegressor, LinearRegression, Lasso and Ridge) in the first layer with different combinations of the most important features so the predictions should not be identical and the layer 2 will have some variance to deal with. I will also inject one or several subsets of the dataset in the second layer to help it choosing amongst

the different predictions.

As imposed by the competition, the difference between the price predicted and the real price the house has been sold is evaluated using the Root-Mean-Squared-Error (https://en.wikipedia.org/wiki/Root-mean-square_deviation) between the logarithm of the predicted value and the logarithm of the observed sales price. (Taking logs means that errors in predicting expensive houses and cheap houses will affect the result equally).

Benchmark Model

In this section, provide the details for a benchmark model or result that relates to the domain, problem statement, and intended solution. Ideally, the benchmark model or result contextualizes existing methods or known information in the domain and problem given, which could then be objectively compared to the solution. Describe how the benchmark model or result is measurable (can be measured by some metric and clearly observed) with thorough detail.

As a benchmark, I'll use the competition leaderboard which will allow me to compare my solution to the thousands of solutions adopted by the community. Given the number of participants to the competition (roughly 3500) and the density of the scores, I will not take the ranking as a benchmark but the proximity to the best solutions.
As described before, the metric used to measure the result is the RMSE or RMSD on the logs of the price and the estimation; taking the logs allowing to not get a huge penalty on the most expensive houses prediction error.

Evaluation Metrics

The root-mean-square deviation (RMSD) or root-mean-square error (RMSE) is a frequently used measure of the differences between values (sample and population values) predicted by a model or an estimator and the values actually observed. The RMSD represents the sample standard deviation of the differences between predicted values and observed values. These individual differences are called residuals when the calculations are performed over the data sample that was used for estimation, and are called prediction errors when computed out-of-sample. The RMSD serves to aggregate the magnitudes of the errors in predictions for various times into a single measure of predictive power. RMSD is a good measure of accuracy, but only to compare forecasting errors of different models for a particular variable and not between variables, as it is scale-dependent (Source Wikipedia).

I'll use the following formula:
The RMSD of predicted values for times t of a regression's dependent variable y is computed for n different predictions as the square root of the mean of the squares of the deviations:

$$RMSD = \sqrt{\frac{\sum(log(pred) - log(y))^2}{n}}$$

Project Design

This project will imply several steps:
1 - Data exploration
2 - Missing data filling
3 - Categorical data encoding
4 - Basic models implementation and tuning
5 - Stacking

The data exploration is a very important step required to get a good understanding of the data provided and the best way to use them and extract the most pertinent information out of them. It is also very important to understand the possible co-linearities which could affect the performances of the linear models. I would like to determine which data influence most of the price and which data influence the tiny deltas.

The dataset contains a lot of missing data that need to be filled depending on the context extracted from the data available per records, per neighborhood and also globally available. For example, some houses from an expensive neighborhood will have better features than some from a cheapest neighborhood and hence, applying a median value of the neighborhood should be closer to the truth than taking the mean or median of the whole dataset.

Encoding the categorical data can be done in multiple ways; it can be done with one hot encoding and by assigning a real or integer value to each of the possible categorical values. Another technique could be to encode the categories depending on their impact on the sale price (or another reference such as the price per square foot of living area).

Given the comments found on stacking, I plan to use a two layers global model with both layers mixing tree based and linear models so they will extract different information from the dataset and the second layer will be able to choose which model is best. I don’t think I’ll use hundreds or thousands of models since I don’t have the computing power and plan to get the best model possible but a model which could be used in production realistically.