Likes Forecaster

Introduction/Business Problem

This project aims to construct a regression model for the task of forecasting the number of likes that has a venue registered in FourSquare. For the purpose of simplicity, we will only consider venues located in New York City. The project is directed to an audience of business owners, or potential business owners, that would like to know which features could enhance their popularity in FourSquare (measured as the number of likes in the platform). We will use the FourSquare API for retrieving the information related with the venues, including their location, category, and number of likes. Also, we will associate each venue with a neighborhood and a borough of New York.

Data

The Datasets that will be used in this project are:

- **newyork_data.json**: Given in the notebook *DS0701EN-3-3-2-Neighborhoods-New-York-py-v1.0.ipynb* from the third week of the Capstone project. It includes the data of the neighborhoods of New York City, their Borough, latitude and longitude.
- **FourSquare Data**: From the FourSquare API we will need the information of the venues that are near to certain neighborhoods of Ney York. The venues information includes their latitude, longitude, id, and category. After that, we will use the id of each venue for getting their number of likes.
- **census_block_loc.csv**: By the moment we have information about New York neighborhoods and bourough. This dataset contains information related with blocks inside New York, which of a lower level territorial division compared with neighborhoods and bourough. In other words, it can be said that a neighborhood is a group of blocks and a bourough is a group of neighborhoods. The blocks are differentiated by an id. The dataset also contains localization information for each block. The following is the source of this dataset: https://www.kaggle.com/muonneutrino/mapping-new-york-city-census-data/data
- **nyc_census_tracts.csv**: This dataset contains demographical and economical information related with the blocks from the dataset **census_block_loc.csv**, distinguished by their id. The following is the source of this dataset: https://www.kaggle.com/muonneutrino/mapping-new-york-city-census-data/data

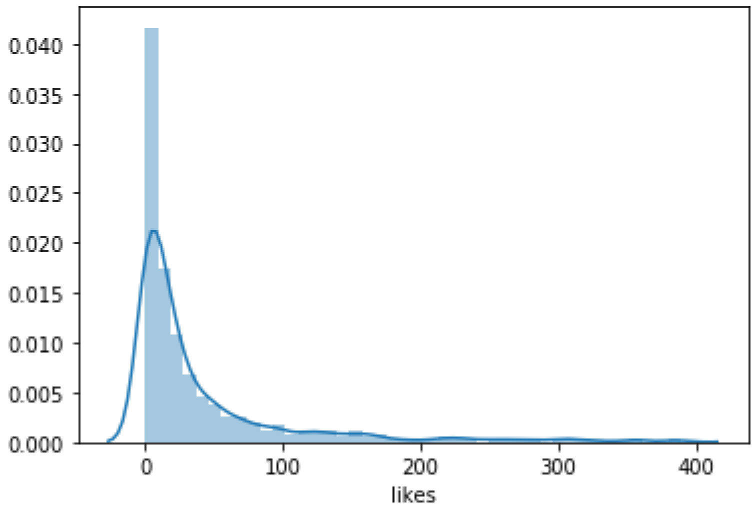
The final dataset include the following variables**:**

- Borough: Categorical variable with 5 levels. Contains the information of the borough where each venue is located.
- Venue: Name of the Venue.

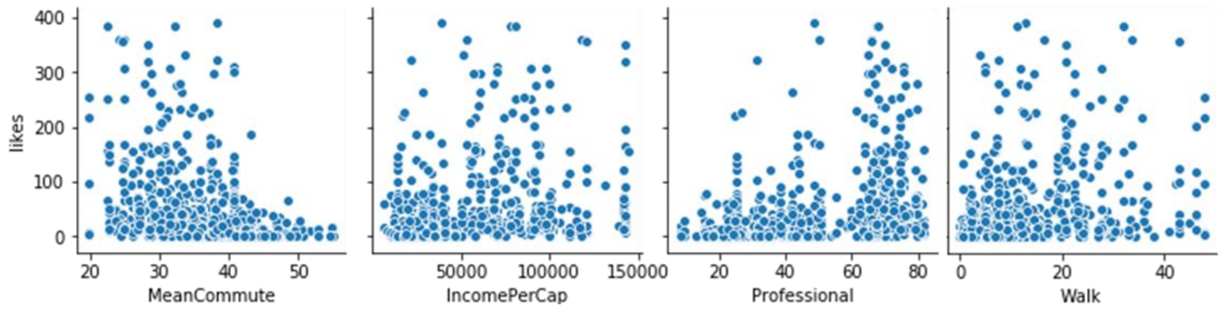
- VenueCategoryGroups: Categorical variable with 6 levels. Contains information related with the category of each venue.
- MeanCommute: Numerical variable. Mean commute to their work of the population of the block.
- IncomePerCap: Numerical variable. Income per capita of the block closest to the venue.
- Professional: Numerical variable. Indicator of the share of the population with a professional work.
- Walk: Numerical variable. Indicator of the share of the population that walks to work.
- Likes: Numerical dependent variable. Likes per venue in the FourSquare platform.
- We associated the venues information with the blocks information by assigning the nearest block to each venue. The final dataset has 753 entries.

Methodology

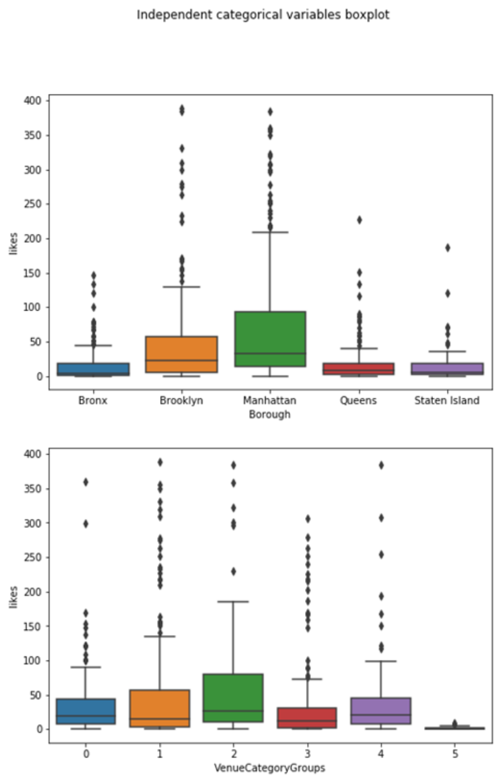
EDA

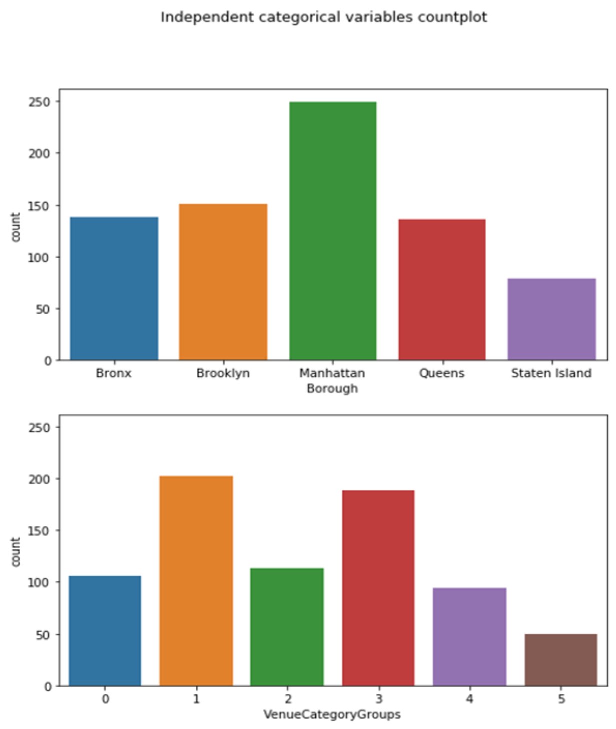


Likes target variable distribution. It is skewed to the right and the lower values are much more than the larger ones.



The scatter plot of the likes variable against the other numerical variables show us that they don't have a strong linear relationship.





The categorical variables have different values in each level. Also, the distribution of likes per level is different, as shown in the boxplots. That's why they can give valuable information to the models.

Regression models

We used the following models for the likes forecasting:

- Linear Regression. Classic linear model.
- Support Vector Regression. Classic nonlinear model.
- XGBoost Regressor. Novel nonlinear model, based on ensembles of trees models.

Their performance is measured with the following metrics.

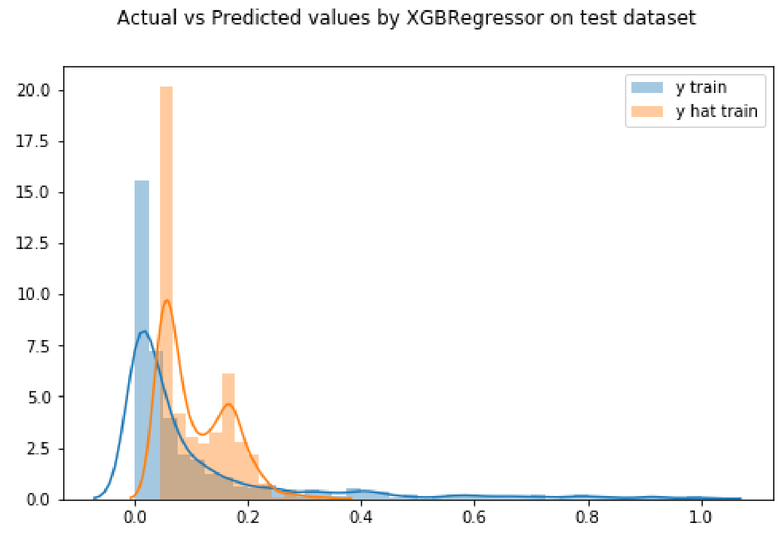
- R squared.
- Root Mean Squared Error.
- Mean Absolute Error.

Results

At first, we show a summary with the resultant metrics per model.

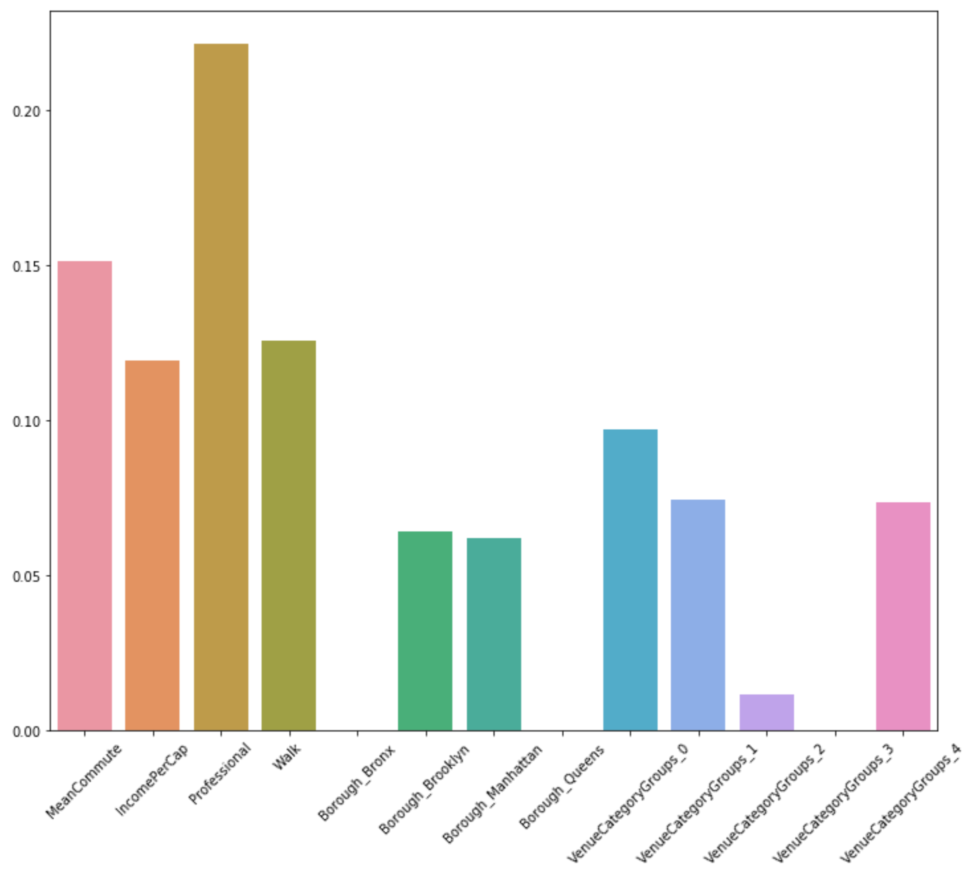
Model	**Train/Test**	**R^2**	**RSME**	**MAE**
Linear Regression	Train	0.196	0.158	0.1
Linear Regression	Test	0.111	0.143	0.094
SVR	Train	0.177	0.16	0.102
SVR	Test	0.117	0.143	0.096
XGBRegressor	Train	**0.262**	**0.151**	**0.095**
XGBRegressor	Test	**0.172**	**0.138**	**0.090**

The model with best performance in every metric is the XGBRegressor. We need higher values in the R squared metric and lower values in both RMSE and MAE metrics. Now, we plot the distribution of the target variable and the predicted variables in both train and test set for the XGBRegressor.



Finally, we plot the feature importance provided by the best XGBRegressor model.

Feature importance XGBRegressor



Discussion

It seems that the regression models had a bad time trying to extract the non-linear relationship between the independent variables and the dependent variables. Since the EDA and the correlation plot we could be aware of this limitation, because regression models require a high linear relationship in order to have good results. Now, the model that had a better performance extracting the mentioned non-linear relationship was the XGBRegressor, as it is an ensemble machine learning model based on trees (for further information see https://machinelearningmastery.com/xgboost-for-regression/). The affirmation stated before is based on the R squared values, as it is a statistic that indicates the percentage of the variance of the dependent variable that is explained by the independent variables.

However, we have to say that the performance of the models is not good enough, based on the value the metrics for each model. Another important observation is that the models seems to overfit the train dataset and their R squared metric is worse in the test dataset. In the other metrics (**RSME, MAE**), the models have a good performance in the test dataset, in comparation with the train dataset. In general, the model with the best performance is the XGBRegressor. But if we plot the distribution of the predicted and actual values of the target variable, we see that the model is very limited. It doesn't seem that the model can output values in the whole range of the actual values of the target variable.

Another important result is the weight of the variables in the XGBRegressor model, shown in the Results section. The variables with a higher weight in the model are Professional and MeanCommute.

Some dummies categorical variables don't have a weight in the model at all. But others have a great weight. For example, the model gives a huge weight to the VenueCategoryGroups_0 dummy variable.

Conclusions and further recommendations

- The geolocation data can be combined with demographical and economical variables for constructing projects that can be solved through machine learning problems.
- Both problems of overfitness and bad performance in the metrics can be addressed by increasing the size of the dataset. In a further implementation of this Likes Forecaster, we could have a higher tier FourSquare API developer license with more calls and extract a big dataset, i.e. with more than 10,000 entries or even more. We could extract information from other cities and include them as a variable.
- The most suitable model for the task of forecasting likes of the venues of FourSquare is XGBRegressor as it beats the others in every metric, in both training and test dataset. This is due to the nature of the model, as it can extract the nonlinear relationship between the variables.