House Prices: Advanced Regression Techniques

Yingjie Guo

1. Abstract

The purpose of this project is to predict sales prices for houses using regression techniques. Given 79 different types of features including both category and numerical features, I trained models with1460 training data and use them to make predictions on the 1459 test data. And the machine learning models I implemented consists of Ridge regression, Lasso regression, Random Forest, XGBoost and LightGBM. The results are measured by root-mean-squared-error between the logarithm of the predicted value and the logarithm of the observed sales price. The final score achieved is 0.12663.

2. Introduction

2.1 Problem Type, Statement and Goals

The purpose of this project is to predict sales prices for each house using regression techniques. This is a regression problem. What makes this problem interesting is that people can estimate the price of a house based on its various information including its living area, physical location, roof material, type of heating, etc. And these can give both customers and sellers an overview of what matters most for the price of a house.

2.2 Literature review

XGBoost is an optimized distributed gradient boosting library designed to be highly **efficient**, **flexible** and **portable**. It implements machine learning algorithms under the Gradient Boosting framework. XGBoost provides a parallel tree boosting (also known as GBDT, GBM) that solve many data science problems in a fast and accurate way. The same code runs on major distributed environment (Hadoop, SGE, MPI) and can solve problems beyond billions of examples.
LightGBM is a gradient boosting framework that uses tree based learning algorithms. It is designed to be distributed and efficient with the following advantages:
Faster training speed and higher efficiency.
Lower memory usage.

Better accuracy.
Support of parallel and GPU learning.
Capable of handling large-scale data.

2.3 Prior and Related Work - None

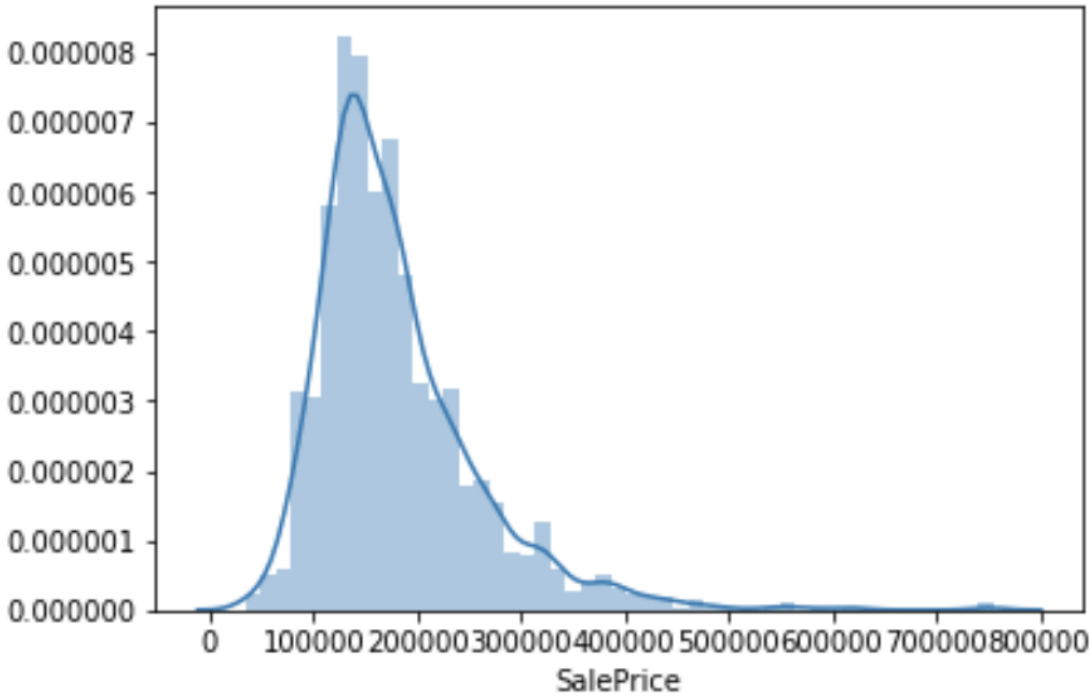
2.4 Overview of Approach

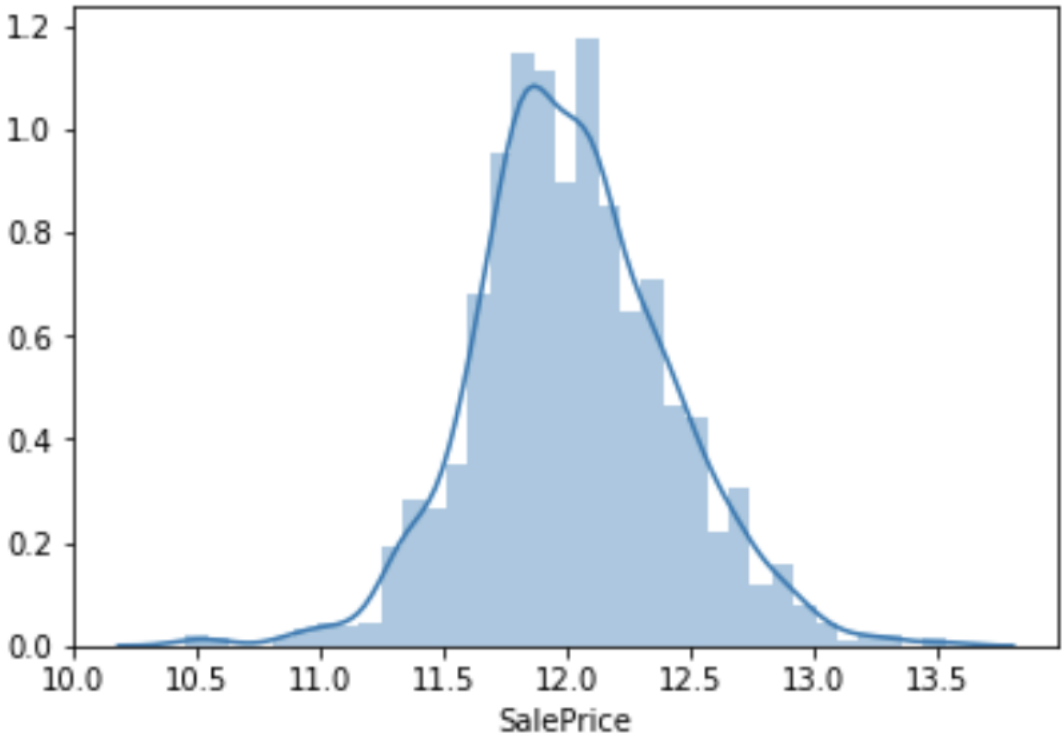
Since the distribution of the target (sale price) is skewed, I applied log on the sale price as the target in the whole process.
Implemented Ridge regression, Lasso linear model, random forest method, XGBoost model and LightGBM model. The LightGBM yield the best performance on the test data and achieved 0.12663 rmse. I used cross validation method in Ridge, Lasso to find the best alpha. And tuned the various parameters in XGBoost and LightGBM using cross validation method to improve the performance.

3. Implementation

3.1 Data Set

Since the distribution of the target (sale price) is skewed, I applied log on the sale price as the target in the whole process.





The latter is the processed one.

Target:

- SalePrice - the property's sale price in dollars. This is the target variable that you're trying to predict.

Features:

Categorical features:

- MSZoning: The general zoning classification
- Street: Type of road access
- Alley: Type of alley access
- LotShape: General shape of property
- LandContour: Flatness of the property

- Utilities: Type of utilities available
- LotConfig: Lot configuration
- LandSlope: Slope of property
- Neighborhood: Physical locations within Ames city limits
- Condition1: Proximity to main road or railroad
- Condition2: Proximity to main road or railroad (if a second is present)
- BldgType: Type of dwelling
- HouseStyle: Style of dwelling
- RoofStyle: Type of roof
- RoofMatl: Roof material
- Exterior1st: Exterior covering on house
- Exterior2nd: Exterior covering on house (if more than one material)
- MasVnrType: Masonry veneer type
- ExterQual: Exterior material quality
- ExterCond: Present condition of the material on the exterior
- Foundation: Type of foundation
- BsmtQual: Height of the basement
- BsmtCond: General condition of the basement
- BsmtExposure: Walkout or garden level basement walls
- BsmtFinType1: Quality of basement finished area
- BsmtFinType2: Quality of second finished area (if present)
- Heating: Type of heating
- HeatingQC: Heating quality and condition
- CentralAir: Central air conditioning
- Electrical: Electrical system

- KitchenQual: Kitchen quality
- FireplaceQu: Fireplace quality
- GarageType: Garage location
- GarageFinish: Interior finish of the garage
- GarageQual: Garage quality
- GarageCond: Garage condition
- PavedDrive: Paved driveway
- PoolQC: Pool quality
- Fence: Fence quality
- MiscFeature: Miscellaneous feature not covered in other categories
- SaleType: Type of sale
- SaleCondition: Condition of sale

Numerical features:

- MSSubClass: The building class
- LotFrontage: Linear feet of street connected to property
- LotArea: Lot size in square feet
- OverallQual: Overall material and finish quality
- OverallCond: Overall condition rating
- YearBuilt: Original construction date
- YearRemodAdd: Remodel date
- MasVnrArea: Masonry veneer area in square feet
- BsmtFinSF1: Type 1 finished square feet
- BsmtFinSF2: Type 2 finished square feet
- BsmtUnfSF: Unfinished square feet of basement area

- TotalBsmtSF: Total square feet of basement area
- 1stFlrSF: First Floor square feet
- 2ndFlrSF: Second floor square feet
- LowQualFinSF: Low quality finished square feet (all floors)
- GrLivArea: Above grade (ground) living area square feet
- BsmtFullBath: Basement full bathrooms
- BsmtHalfBath: Basement half bathrooms
- FullBath: Full bathrooms above grade
- HalfBath: Half baths above grade
- Bedroom: Number of bedrooms above basement level
- Kitchen: Number of kitchens
- TotRmsAbvGrd: Total rooms above grade (does not include bathrooms)
- Fireplaces: Number of fireplaces
- GarageYrBlt: Year garage was built
- Functional: Home functionality rating
- GarageCars: Size of garage in car capacity
- GarageArea: Size of garage in square feet
- WoodDeckSF: Wood deck area in square feet
- OpenPorchSF: Open porch area in square feet
- EnclosedPorch: Enclosed porch area in square feet
- 3SsnPorch: Three season porch area in square feet
- ScreenPorch: Screen porch area in square feet
- PoolArea: Pool area in square feet
- MiscVal: $Value of miscellaneous feature
- MoSold: Month Sold

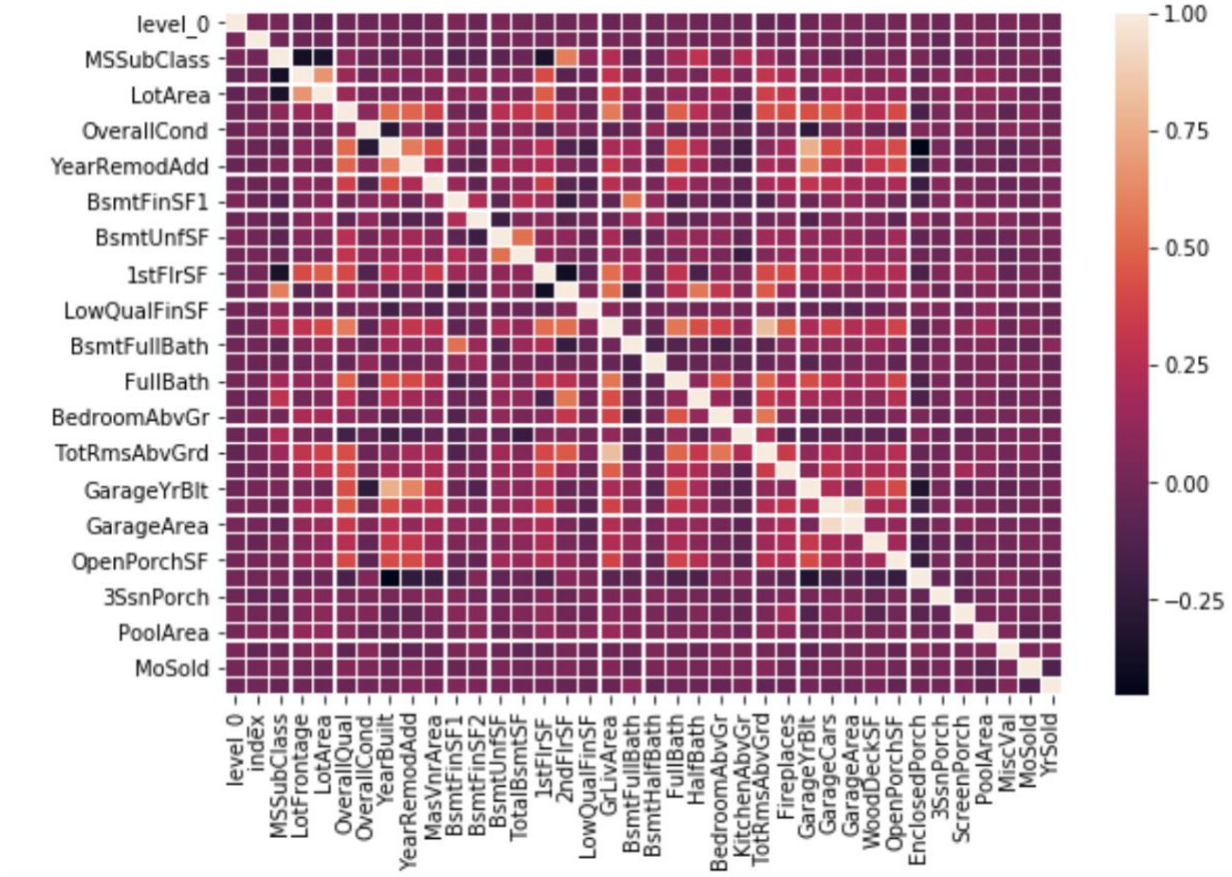
- YrSold: Year Sold

3.2 Preprocessing, Feature Extraction, Dimensionality, Adjustment

Firstly, I separate the training data into two parts: train(0.8) and mytest(0.2).

Secondly, dealing with the missing data. I separate the data into two parts: categorical features and numerical featuresAccording to the description of the data offered in the project: much NaN data can be represented by “None”, such as ‘PoolQC’, ‘MiscFeature’, ‘None’, etc. And filling the missing by the most common category such as ‘Electrical’, ‘Functional’, etc. Besides, I filled the missing numerical features in mytest and test data by the median of the train data when it is an integer such as ‘GarageCars’, and filling with mean value of the train data when it is not integer such as ‘BsmtFinSF1’.

Calculate the correlation of features and draw the map. There are no too obvious reliability between features. So I did not delete features.



Next, I used OneHotEncoding method to process the category features. And Standardized the numerical features.
Since some of the numerical features are skewed. I used boxcox method to process these features for better performance.

3.3 Dataset Methodology

1168 data are used as my training data which are also used for my 5-fold cross validation in the modeling part. And 292 Data are used as my test data. And above both from training data provided from the Kaggle project.

3.4 Training Process

Implemented Ridge regression, Lasso linear model, random forest method, XGBoost model and LightGBM model.

Used cross validation method (RidgeCV) to choose the best alpha, which is 0.28005, rmse of mytest is 0.13351

Used cross validation method (LassoCV) to choose the best alpha, which is 0.00016, rmse of mytest is 0.12944

Implemented XGBoost method and tuned parameters with greedy method using 5-fold cross validation:
'eta', 'max_depth', 'subsample', 'min_child_weight', 'lambda', 'alpha', 'objective'.
The best manually tuned parameters: {'eta': 0.05, 'max_depth': 2, 'subsample': 0.85, 'objective': 'reg:linear', 'eval_metric': 'rmse', 'lambda': 0.03, 'alpha': 0.3, 'base_score': 12.030658310971578, 'silent': 1, 'min_child_weight': 0.01}
rmse of mytest is 0.13569

Implemented LightGBM method and tuned parameters with greedy method using 5-fold cross validation:
'learning_rate', 'n_estimators', max_bin', 'max_depth', 'num_leaves', 'bagging_fraction', 'bagging_freq', 'min_gain_to_split', 'feature_fraction', 'min_data_in_leaf', 'min_sum_hessian_in_leaf', 'lambda_l2', 'lambda_l1', 'objective'
The best manually tuned parameters: {'learning_rate': 0.03, 'metric': 'rmse', 'bagging_freq': 1, 'objective': 'regression', 'num_threads': 2,

'feature_fraction_seed': (9,), 'bagging_seed': 9, 'n_estimators': 500, 'max_bin': 20, 'max_depth': 3, 'num_leaves': 8, 'bagging_fraction': 0.78, 'min_gain_to_split': 0.01, 'feature_fraction': 0.2, 'min_data_in_leaf': 15, 'min_sum_hessian_in_leaf': 0.0001, 'lambda_l2': 0.001, 'lambda_l1': 0.01}
rmse of mytest is 0.12431

Implemented random forest method with max_depth set to 3 and n_estimatos set to 100.
rmse of mytest is 0.20487

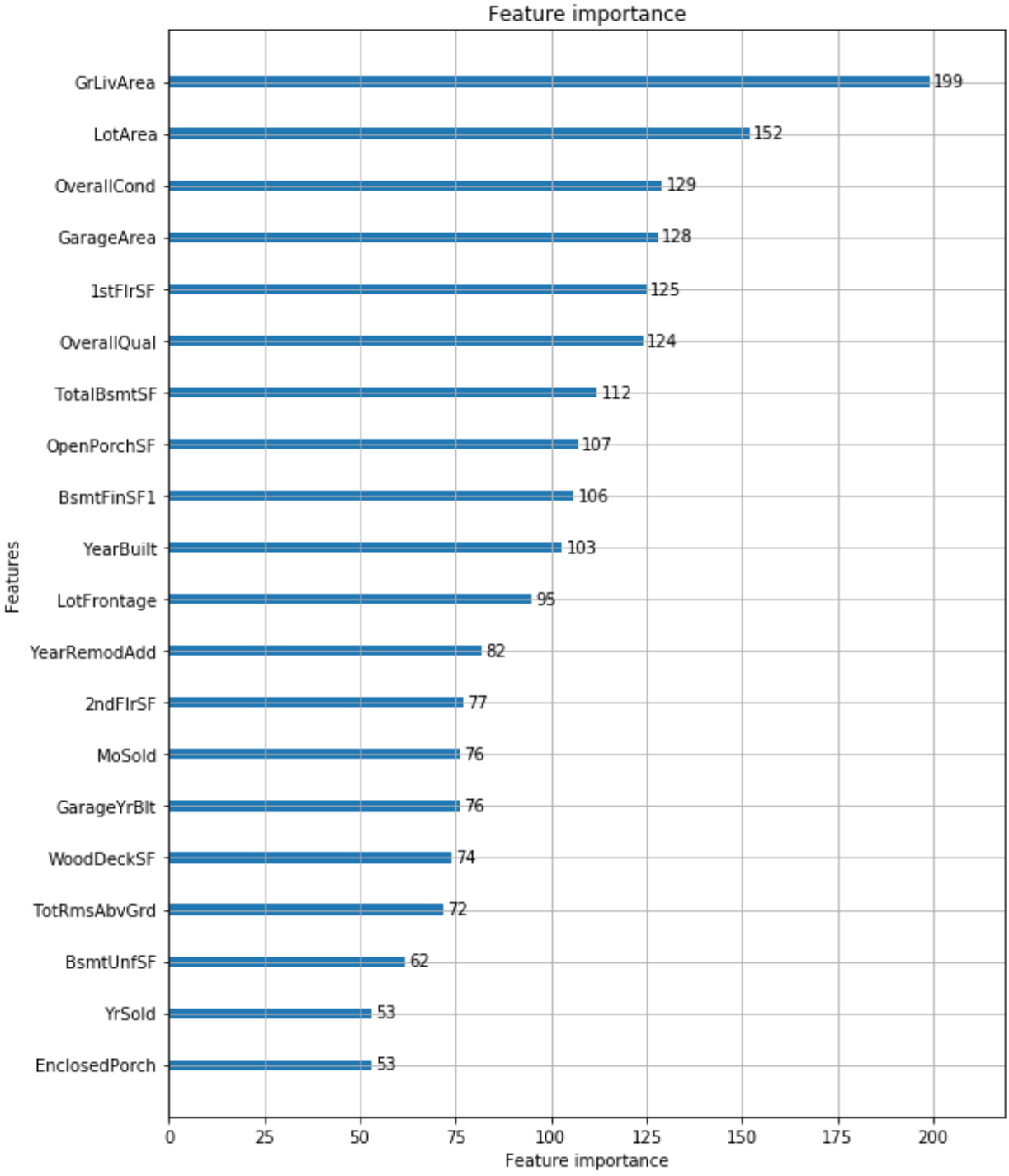
model	Train rmse	Mytest rmse
Ridge regression	0.10194	0.13351
Lasso regression	0.10600	0.12953
Random forest	0.18284	0.20487
XGBoost	0.01398	0.13569
LightGBM	0.09237	0.12431

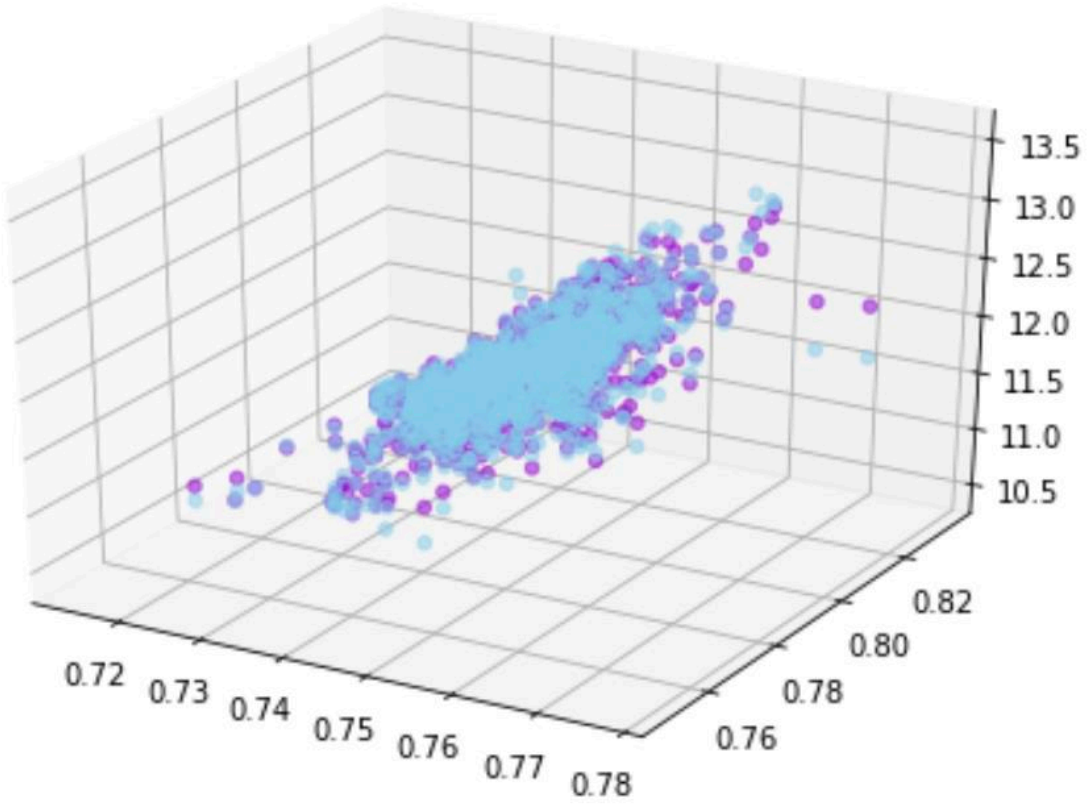
The LightGBM yield the best performance on the test data and achieved 0.12663 rmse.

3.5 Model Selection and Comparison of Results

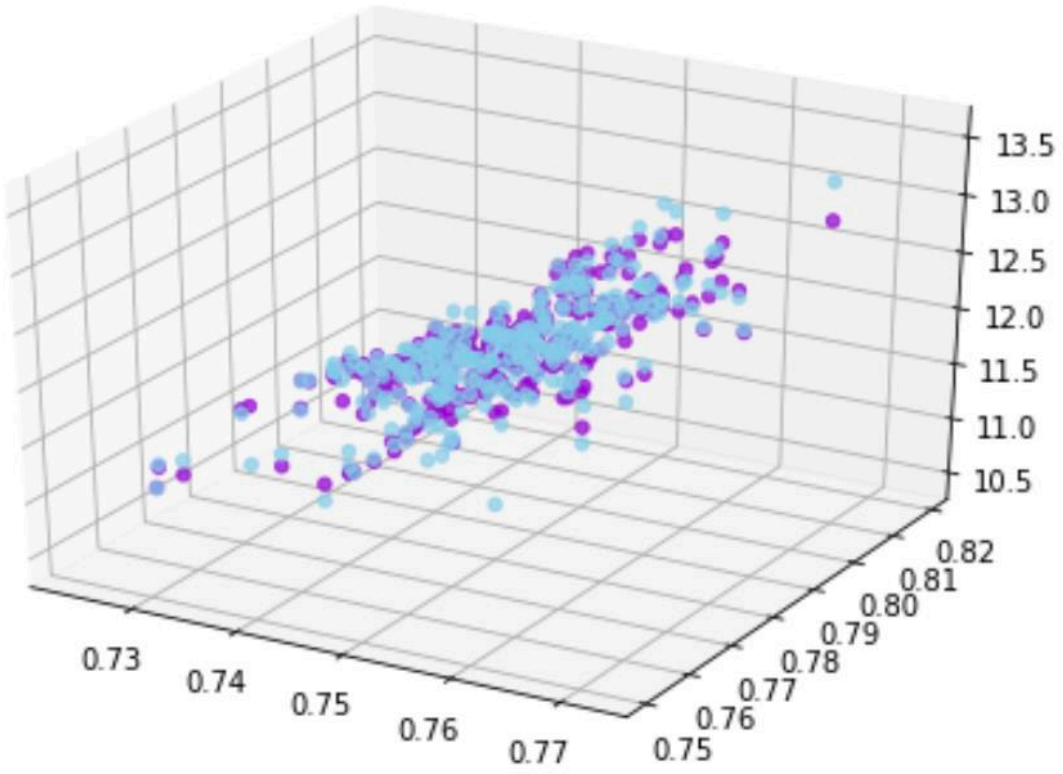
Based on the performance on mytest data. I choose the LightGBM as my final model and achieved 0.12663 rmse.

4. Final Result and Interpretation


Feature importance
GrLivArea
LotArea
OverallCond
GarageArea
1stFlrSF
OverallQual
TotalBsmtSF
OpenPorchSF
BsmtFinSF1
YearBuilt
LotFrontage
YearRemodAdd
2ndFlrSF
MoSold
GarageYrBlt
WoodDeckSF
TotRmsAbvGrd
BsmtUnfSF
YrSold
EnclosedPorch
199
152
129
128
125
124
112
107
106
103
95
82
77
76
76
74
72
62
53
53
Features
0
25
50
75
100
125
150
175
200
Feature importance



Above is the plot showing the price(z- axis) based on the two most important features (GrLivArea and LotArea) of the train data.



Above is the plot showing the price(z- axis) based on the two most important features (GrLivArea and LotArea) of the mytest data.

Eout <= Etest + sqrt(1/2N* ln(2M/δ))
Let δ be 0.05, M=5, N=292, Etest=0.12431
Eout<= 0.12431+0.09525 =0.21956

Based on the performance on mytest data. I choose the LightGBM as my final model and achieved 0.12663 rmse.

5. Summary and conclusions

The reason why LightGBM perform best is that it is complicated enough for the model. Ridge and Lasso are too simple. XGBoost is overfitting since the train rmse is much smaller than mytest rmse. As for the random forest, further parameter tuning will help to improve the performance.

I found that during the parameter tuning process, LightGBM calculated much faster than XGBoost and also yield better results. I think this is because lightGBM is based on XGBoost and have improved not only the speed but also the performance of prior model.
According to LightGBM method, GrLivArea and LotArea are the two most important features in predicting the sale price of the house. This make sense because, GrLivArea represents above grade (ground) living area square feet and LotArea represents Lot size in square feet. These are reasonable since people actually care about these when they are choosing a house. And the top 10 important features are all numerical features which is interesting. Maybe numerical features can describe the details of a house more precisely. I think it can be challenging to collect more data and more features of different houses in different cities of different countries. And build different models based on cities or countries. And compare the results. This is really cool and may take much time. People can get more insight into the housing and models learnt via this research.

6. References

https://lightgbm.readthedocs.io/en/latest/Parameters.html
https://xgboost.readthedocs.io/en/latest/parameter.html
https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data
https://www.kaggle.com/aharless/xgboost-lightgbm-and-ols
https://www.kaggle.com/serigne/stacked-regressions-top-4-on-leaderboard