

House Price Prediction

Project Report

CS 6350: Big Data management and Analytics

Spring 2019

Name: Rajarshi Chattopadhyay

NetID: RXC170010

Table of Contents

1. Introduction.....	3
2. Problem Definition and Algorithm.....	3
2.1 Task Definition.....	3
2.2 Algorithms and Libraries	3
3. Data Exploration.....	4
3.1 Important features.....	4
3.2 Missing Data.....	6
3.2 Outliers	7
3.3 Skewness	8
4. Building Model.....	9
5. Prediction	10
6. Conclusion.....	10

1. Introduction

The Kaggle house price dataset provides feature variables that describe different aspects of residential houses. A Machine Learning model is built using the data which can be used to predict the final price of a house based on the features.

2. Problem Definition and Algorithm

2.1 Task Definition

The goal is to predict sales prices of the houses using regression models. The task involves data exploration to analyze the structure and form of data followed by necessary feature engineering and pre-processing to create a trainable data set to train the models. This is followed by evaluation to verify the accuracy of the model on the validation data set. The model with the highest accuracy with an appropriate tradeoff between bias and variance to overcome underfitting or overfitting of data is used to predict the sales price of the houses.

2.2 Algorithms and Libraries

The Spark¹ framework for cluster computing is used for performing the Machine Learning using its Scala API. Spark MLlib² library has been used to build the ML models for predictions.

The categorical (non-numeric) features are indexed using Spark MLlib StringIndexer³. It encodes a string column of labels to a column of label indices.

The feature columns are assembled together using Spark MLlib VectorAssembler⁴ so that they can be used to train the model. It combines a given list of columns into a single vector column. It is useful for combining raw features and features generated by different feature transformers into a single feature vector, in order to train ML models.

The regression methods from the Spark MLlib Regression⁵ library is used for model creation. The different regression algorithms used are Linear Regression, Decision Tree Regression, Random Forest Regressor, Gradient Boosting Tree Regressor.

The hyperparameters of the different ML algorithms are tuned to find the best model using Cross-validation⁶ using the training dataset within a Pipeline⁷. Finally, the best model obtained is evaluated over the validation dataset to find out its Root Mean Square Error.

¹ <https://spark.apache.org/docs/latest/index.html>

² <https://spark.apache.org/docs/latest/ml-guide.html>

³ <https://spark.apache.org/docs/latest/ml-features.html#stringindexer>

⁴ <https://spark.apache.org/docs/latest/ml-features.html#vectorassembler>

⁵ <https://spark.apache.org/docs/latest/ml-classification-regression.html#regression>

⁶ <https://spark.apache.org/docs/latest/ml-tuning.html#cross-validation>

⁷ <https://spark.apache.org/docs/latest/ml-pipeline.html>

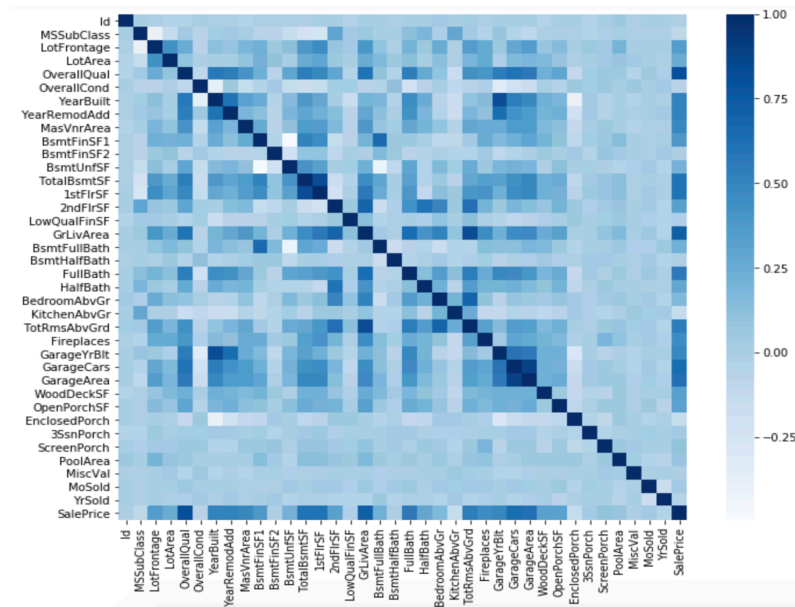
3. Data Exploration

The training data has 1460 rows, 79 feature columns, an Id column, and a SalePrice column, which is the label. The testing data has 1459 rows and does not have the label column.

<https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data>

Python programming language has been used to perform the data exploration as it provides useful libraries for performing statistical calculations on the data and help in visualization through plotting of the necessary data.

3.1 Important features



The correlation among the different features are identified by plotting a correlation matrix of the same. The following features are highly correlated to each other and give almost the same info:

- TotalBsmtSF and 1stFlrSF
- Garage... features
- YearBuilt and GarageYrBlt
- GrLivArea and TotRmsAbvGrd

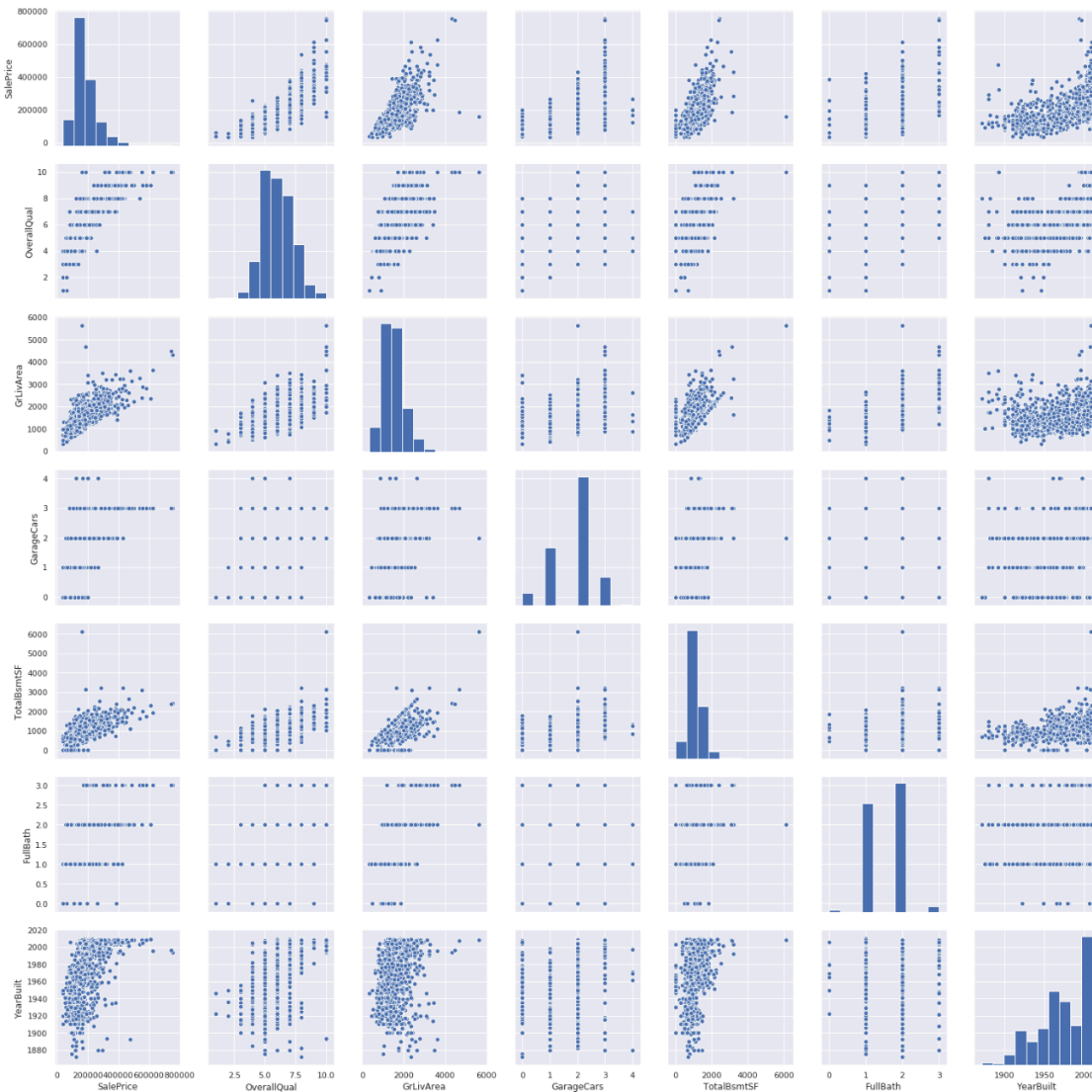
The bottom-last row of the heatmap, shows the relation of SalePrice with the rest of the features and gives an idea on which features the SalePrice is highly correlated with:

- OverallQual
- GrLivArea (and TotRmsAbvGrd: multi-collinear, so keep only one)
- GarageCars (and GarageArea: multi-collinear, so keep only one)
- TotalBsmtSF (and 1stFlrSF: multi-collinear, so keep only one)
- FullBath
- YearBuilt (and GarageYrBlt: multi-collinear, so keep only one)

So, the following columns can be dropped from the dataset:

1stFlrSF, GarageYrBlt, GarageArea, 1stFloor, TotRmsAbvGrd

The correlation between SalePrice and the identified highly correlated variables is checked:



Interesting findings:

- TotalBsmtSF and GrLivArea: Dots almost forming a linear border with the majority of the dots staying below that border, implying that basement areas are expected to be equal or bigger than the above ground living area.
- The plot between SalePrice and YearBuilt is slightly exponential function implying prices increasing at a higher rate nowadays.

Therefore, the possible important features are identified based on how the SalePrice holds its relationship:

- OverallQual => Linear
- YearBuilt: Original construction date => Exponential
- TotalBsmtSF: Total square feet of basement area => Sometimes Exponential
- GrLivArea: Above ground living area square feet => Linear
- GarageCars: The number of Cars that can be fitted into the Garage => Linear

3.2 Missing Data

Missing data may imply a reduction of the sample size, which can hinder the analysis. So missing data was checked and pattern of the same is identified:

	Total	Percent
PoolQC	1453	99.520548
MiscFeature	1406	96.301370
Alley	1369	93.767123
Fence	1179	80.753425
FireplaceQu	690	47.260274
LotFrontage	259	17.739726
GarageCond	81	5.547945
GarageType	81	5.547945
GarageYrBlt	81	5.547945
GarageFinish	81	5.547945
GarageQual	81	5.547945
BsmtExposure	38	2.602740
BsmtFinType2	38	2.602740
BsmtFinType1	37	2.534247
BsmtCond	37	2.534247
BsmtQual	37	2.534247
MasVnrArea	8	0.547945
MasVnrType	8	0.547945
Electrical	1	0.068493

The following features have more than 15% data missing, so these are excluded from the required features list:

- PoolQC 99.5%
- MiscFeature 96.3%
- Alley 93.8%
- Fence 80.7%
- FireplaceQu 47.3%
- LotFrontage 17.7%

These are not important features and are also outlier candidates.

The following features specific to the Garage and Basement have the same number of missing data (5.5%) and are not critical. So, these are excluded these from the required features list:

- GarageCond 81 5.5%
- GarageType 81 5.5%
- GarageYrBlt 81 5.5%
- GarageFinish 81 5.5%
- GarageQual 81 5.5%
- BsmtExposure 38 2.6%
- BsmtFinType2 38 2.6%
- BsmtFinType1 37 2.5%
- BsmtCond 37 2.53%
- BsmtQual 37 2.53%

The following features are not essential, and are excluded from the required features list:

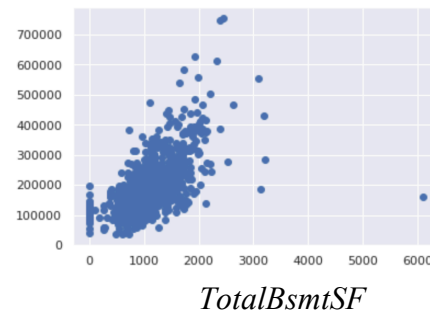
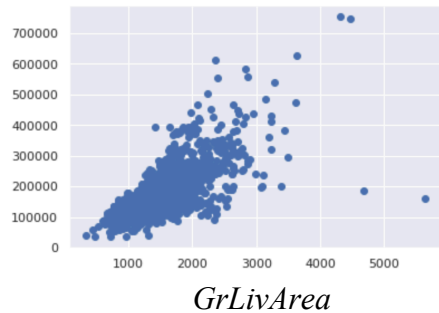
- 'MasVnrArea'
- 'MasVnrType'

The 'Electrical' feature has one missing observation which is delete but the variable is kept.

Thus, essentially, all the other features with missing data are dropped.

3.2 Outliers

Outliers can affect models and can be a valuable source of insightful information. The plots of the SalePrice with the important features are obtained to check for outliers:



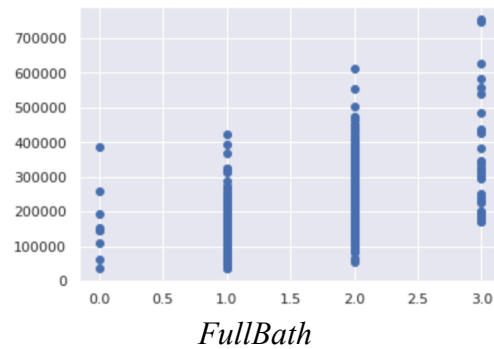
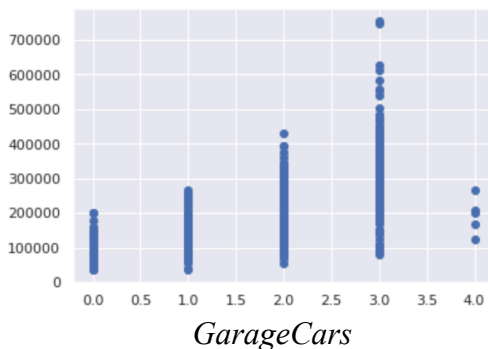
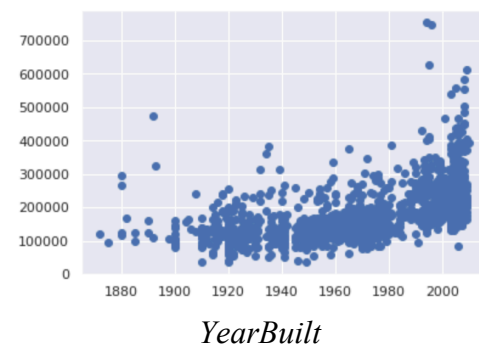
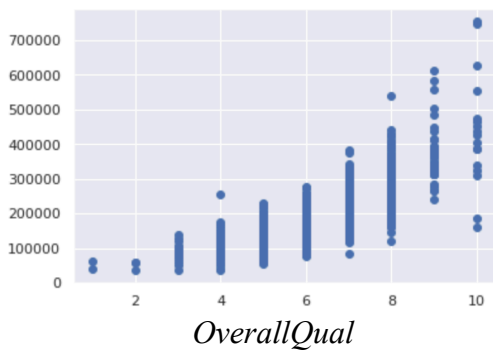
The top two observations with highest GrLivArea but lesser than 200k SalePrice are not typical and do not follow the general trend of price rise. So, they can be considered as outliers.

Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature
1299	60	RL	313.0	63887	Pave	NaN	IR3	Bnk	AllPub	...	480	Gd	NaN	NaN
524	60	RL	130.0	40094	Pave	NaN	IR1	Bnk	AllPub	...	0	NaN	NaN	NaN

The observation with the highest TotalBsmtSF but lesser than 200k 'SalePrice' can be considered as an outlier.

Id	MSSubClass	MSZoning	LotFrontage	LotArea	Street	Alley	LotShape	LandContour	Utilities	...	PoolArea	PoolQC	Fence	MiscFeature
1299	60	RL	313.0	63887	Pave	NaN	IR3	Bnk	AllPub	...	480	Gd	NaN	NaN

These two observations with Id 1299 and 524 are deleted.

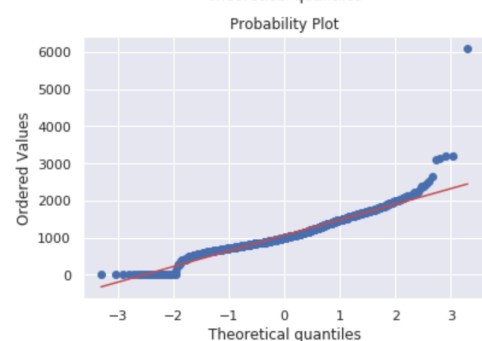
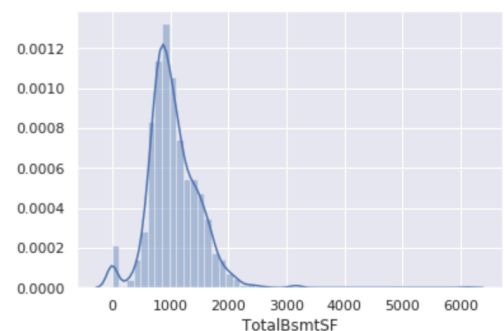
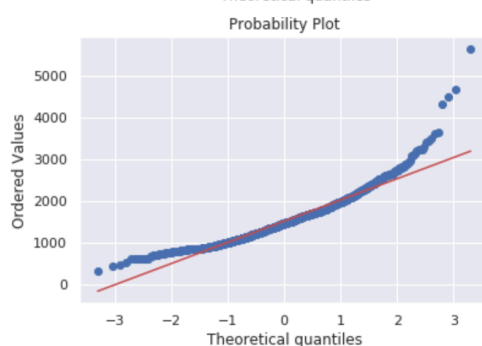
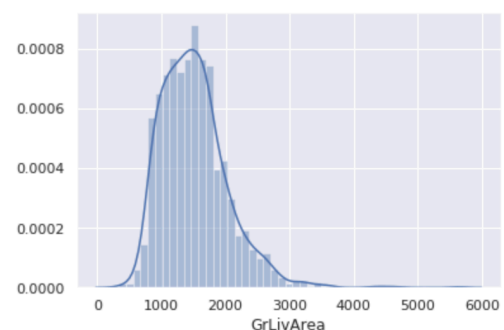
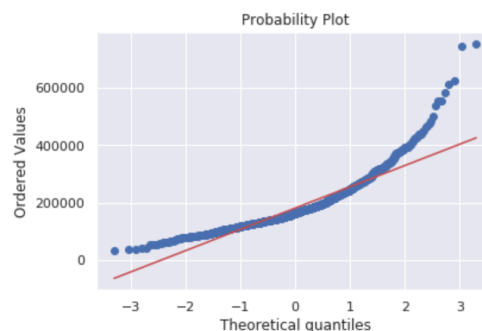
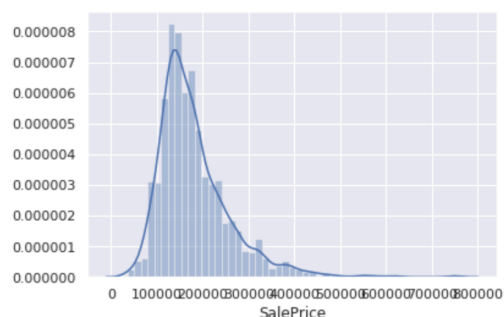


The other important features – OverallQual, YearBuilt, GarageCars, and FullBath do not reveal any outlier.

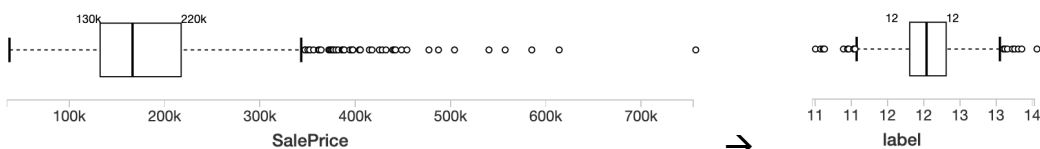
3.3 Skewness

Check the distribution of the SalePrice.

The plotted histogram of the SalePrice (label) column shows that it is skewed to the left.



SalePrice and GrLivArea deviate from the normal distribution and have positive skewness while peaking. These are handled by performing log operation on them. Log is useful to remove skewness in cases where only few points are much larger than the bulk of the data.



Thus, the label data gets more balanced and suitable for training and prediction purposes.

For TotalBsmtSF, not only does it deviate from the normal distribution, and have positive skewness while peaking, but also has a number of observations with value 0 implying houses without basement. Log operation cannot be performed on zero values.

To handle this, the log operation is applied to only those TotalBsmtSF that are greater than 0.

4. Building Model

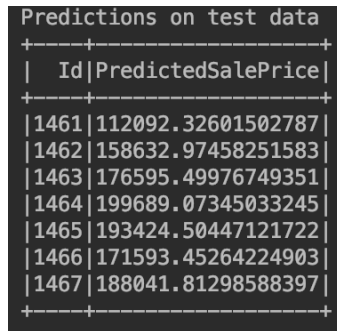
The following steps are performed to build the required Machine Learning model that is to be used to predict the SalePrice of houses:

- Training Data is read into a Spark DataFrame
 - The unimportant columns are dropped
 - The null values in the DataFrame are handled as required
 - Data skewness is removed
 - Feature indexing is performed to convert categorical features to numerical form
 - All the Features are assembled together to form a trainable feature vector
 - Regressors are selected
 - A Pipeline is created, and Parameter grid is built specifying the hyperparameter values
 - A Cross validator is set for finding best model parameters based on RMSE evaluator
 - Training and Validation sets are created by splitting the training data
 - Cross-validation is performed to train the different models and the best model is chosen
- The best model is the one that has the least Root Mean Square Error (RMSE)

```
Connected to Spark
Data read into DataFrame
Dropped unimportant columns
Handled null values
Removed data skewness
Feature indexing completed
Features assembled
Regressors selected
Pipeline and Parameter grid built for models
Cross validator set for finding best model parameters
Training and Validation sets formed
Training models...
Running cross-validation on training set to choose the best model...
Best Model:
{
  linReg_610dcd27429d-elasticNetParam: 0.7,
  linReg_610dcd27429d-maxIter: 10,
  linReg_610dcd27429d-regParam: 0.5,
  pipeline_9901a92912ea-stages: [Lorg.apache.spark.ml.PipelineStage;@16c58781
}
Predictions made on validation set
RMSE of Best Model on validation set is: 0.129102130712155
```

5. Prediction

The best model obtained is used to predict the House Price of the different houses whose features are specified in the test data set.

A terminal window with a dark background and light green text. It displays a table titled 'Predictions on test data'. The table has two columns: 'Id' and 'PredictedSalePrice'. It contains seven rows of data, each representing a house with its ID and predicted sale price.

Id	PredictedSalePrice
1461	112092.32601502787
1462	158632.97458251583
1463	176595.49976749351
1464	199689.07345033245
1465	193424.50447121722
1466	171593.45264224903
1467	188041.81298588397

6. Conclusion

Overall, it is found that best model is the Linear Regression model with:

elasticNetParam: 0.7 (Elastic Net Parameter)

regParam: 0.5 (Regularization Parameter)

maxIter: 10 (Number of iterations used in training)

The Root Mean Square error of the best model is ~ 0.12 i.e. about 12% is moderate.

Further improvements could be made by gathering more training data for the different features.

In the future, the plan is to incorporate Principal Component Analysis for Dimensionality Reduction. The PCA has high potential to significantly improvement the learning and make the feature engineering task easier.