

HOUSE PRICE PROJECT

Submitted by:

Shahbaz Ali

**ACKNOWLEDGMENT**

The Reference includes the website “**Towards Data Science**”. From there, I got lot of help and knowledge in making this project. The concepts are clear on this website with lots of practical examples so that one can learn easily and quickly.

Not to forget GitHub repositories, I saw and learned about making my precise model from GitHub Repositories.

Also, like to thank my mentor Mr. Shubham Sir for helping me throughout in this project where ever I was stucked. He has good knowledge about Machine Learning and Data Science modules and hence is excel in his field, helped me whenever I raised any concern regarding my project.

**INTRODUCTION**

* Business Problem Framing

Houses are one of the necessary needs of each and every person around the globe and therefore housing and real estate market is one of the markets which is one of the major contributors in the world’s economy. It is a very large market and there are various companies working in the domain.

Data science comes as a very important tool to solve problems in the domain to help the companies increase their overall revenue, profits, improving their marketing strategies and focusing on changing trends in house sales and purchases. Predictive modelling, Market mix modelling, recommendation systems are some of the machine learning techniques used for achieving the business goals for housing companies. Our problem is related to one such housing company.

A US-based housing company named Surprise Housing has decided to enter the Australian market. The company uses data analytics to purchase houses at a price below their actual values and flip them at a higher price. For the same purpose, the company has collected a data set from the sale of houses in Australia. The data is provided in the CSV file below.

* Conceptual Background of the Domain Problem

The objective of the project is to perform data visualization techniques to understand the insight of the data. Machine learning often required to getting the understanding of the data and its insights. This project aims apply various [Python](https://www.python.org/) tools to get a visual understanding of the data and clean it to make it ready to apply machine learning operations on it.

* Review of Literature
* Objective & Data

The competition goal is to predict Sale prices for homes. We’re given a training and testing data set in .csv format as well as a data dictionary.

**Training**: Our training data consists of 1,460 examples of houses with 79 features describing every aspect of the house. We are given sale prices (labels) for each house. The training data is what we will use to “teach” our models.

**Testing**: The test data set consists of 1,459 examples with the same number of features as the training data. Our test data set excludes the sale price because this is what we are trying to predict. Once our models have been built, we will run the best one the test data.

**Task**: Machine learning tasks are usually split into three categories; Supervised, Unsupervised and Reinforcement. For this competition, our task is Supervised learning.

* Supervised learning uses examples and labels to find patterns in data.
* Motivation for the Problem Undertaken

## Project Pipeline

Generally speaking, machine learning projects follow the same process. Data ingestion, data cleaning, exploratory data analysis, feature engineering and finally machine learning.

The pipeline is not linear and we might find to jump back and forth between different stages. It’s important to mention this because tutorials often make us believe the process is much cleaner than in reality.

**Analytical Problem Framing**

* Mathematical/ Analytical Modelling of the Problem

**Libraries**: These are frameworks in python to handle commonly required tasks. I Implore any budding data scientists to familiarise themselves with these libraries:

[**Pandas**](https://pandas.pydata.org/)— For handling Structured data

[**Scikit Learn**](https://scikit-learn.org/stable/) — For Machine Learning

[**NumPy**](https://numpy.org/)— For Linear Algebra and Mathematics

[**Seaborn**](https://seaborn.pydata.org/)— For Data Visualization

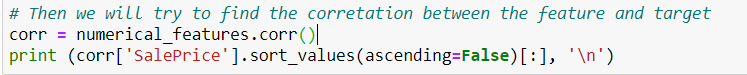
In the project we analyze the numeric features using the NumPy library:



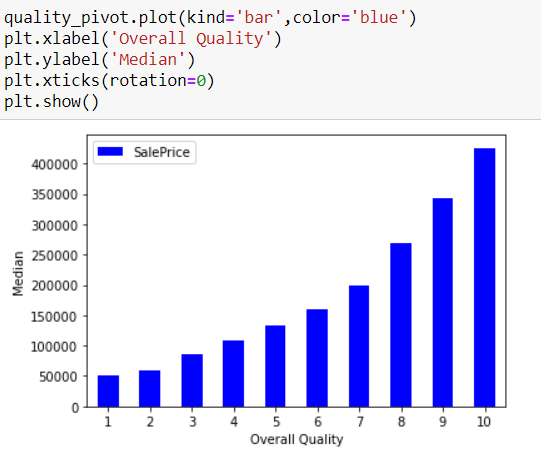
To find out the datatypes of train set of data.

To find the correlation between feature and target.

Correlation can be an important tool for feature engineering in building the Machine Learning Models. The stronger the correlation, the more difficult it is to change one variable without changing another.

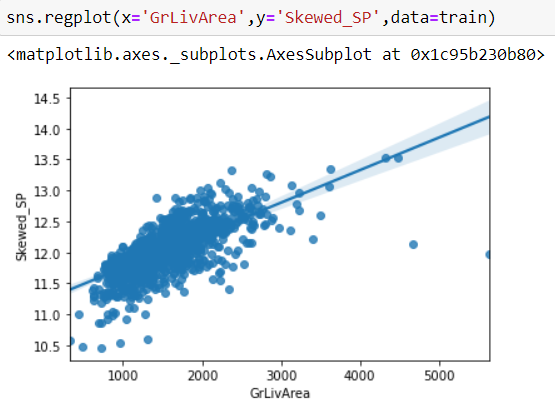


Then we plotted a pivot graph between Overall Quality and median of the data where Sales Prices are indicated.

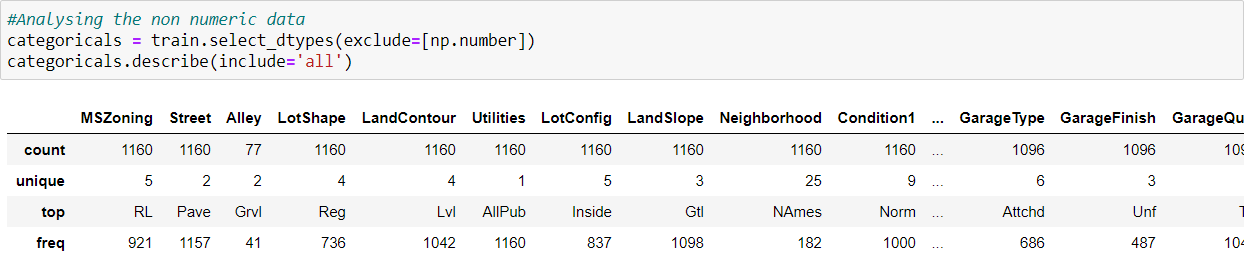


Here we see SalePrice varies directly with the overall quality. As the quality of product improves , the Sales Price goes up.

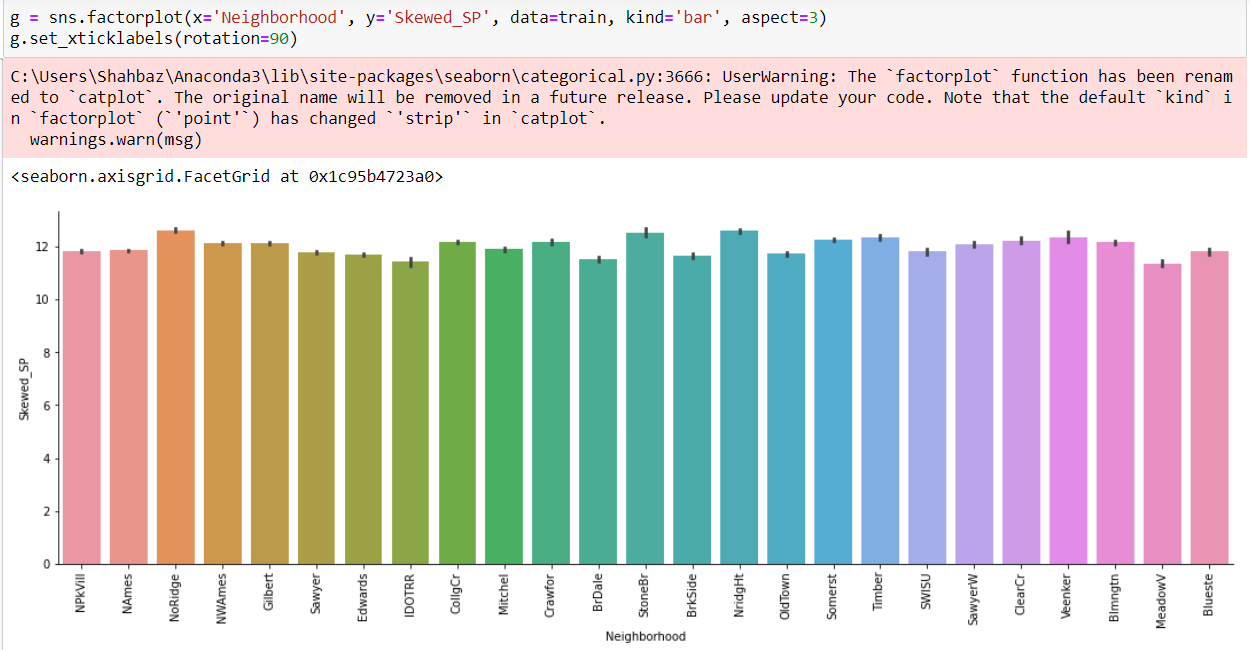
Again, SalePrice increases as the GrLivArea increases. We will also get rid of the outliers which severely affect the prediction of the survival rate.



After removing Outliers and Null values, we analyze non numeric data.



We Plotted seaborn factorplot to mention and show Neighbourhood w.r.t Skewed\_SP. The results were something like this:



* Data Sources and their formats

The data sources are provided by my internship company – Flip Robo Technologies as raw data which I further used for Data ingestion, data cleaning, exploratory data analysis, feature engineering and finally machine learning.

The data was provided in .csv format as train and test type of data. Train data is used to “teach” our model and test data is used to evaluate the fit machine learning model.

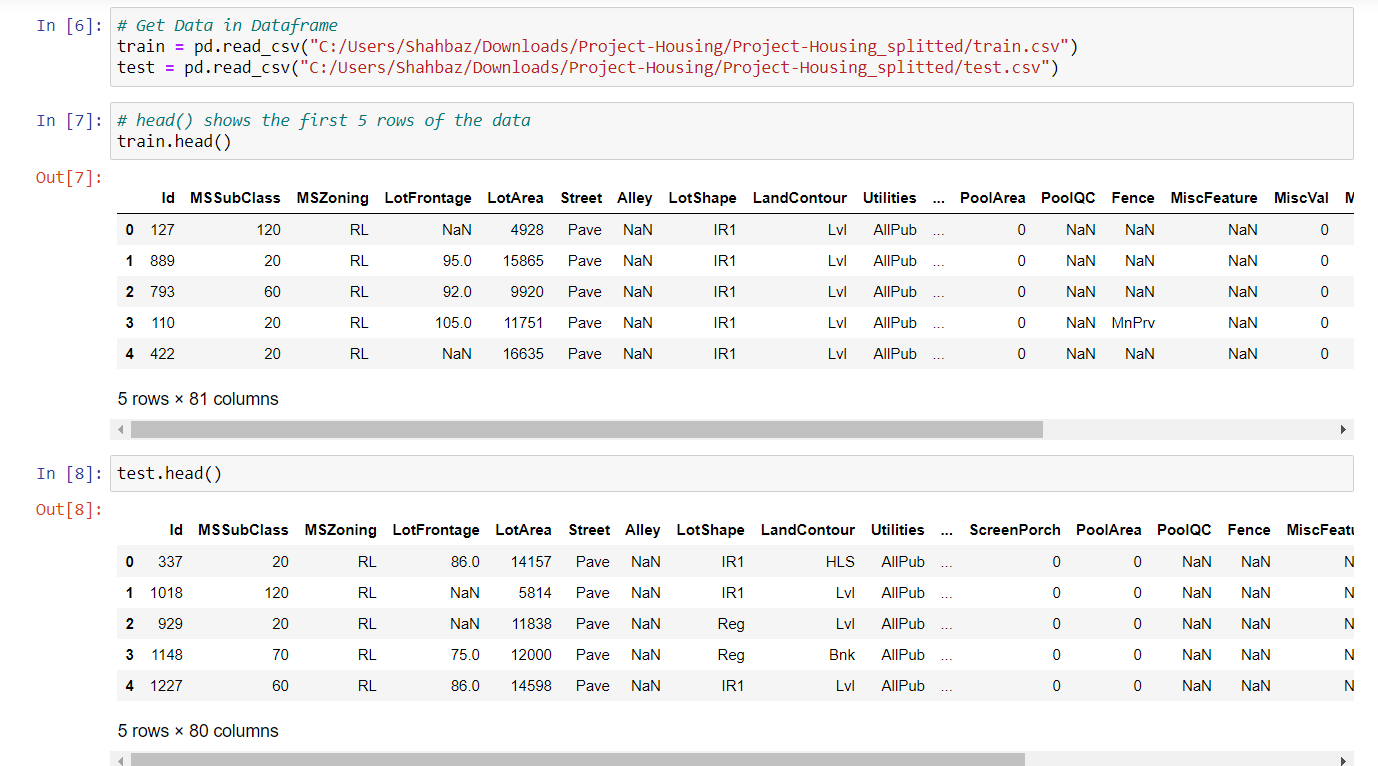
The objective is to estimate the performance of the machine learning model on new data: data not used to train the model.

The train-test split is a technique for evaluating the performance of a machine learning algorithm.

It can be used for classification or regression problems and can be used for any supervised learning algorithm.

**Proper Data Description** 🡪 Our training data consists of 1,460 examples of houses with 79 features describing every aspect of the house. We are given sale prices (labels) for each house.

The test data set consists of 1,459 examples with the same number of features as the training data. Our test data set excludes the sale price because this is what we are trying to predict.



* Data Preprocessing Done

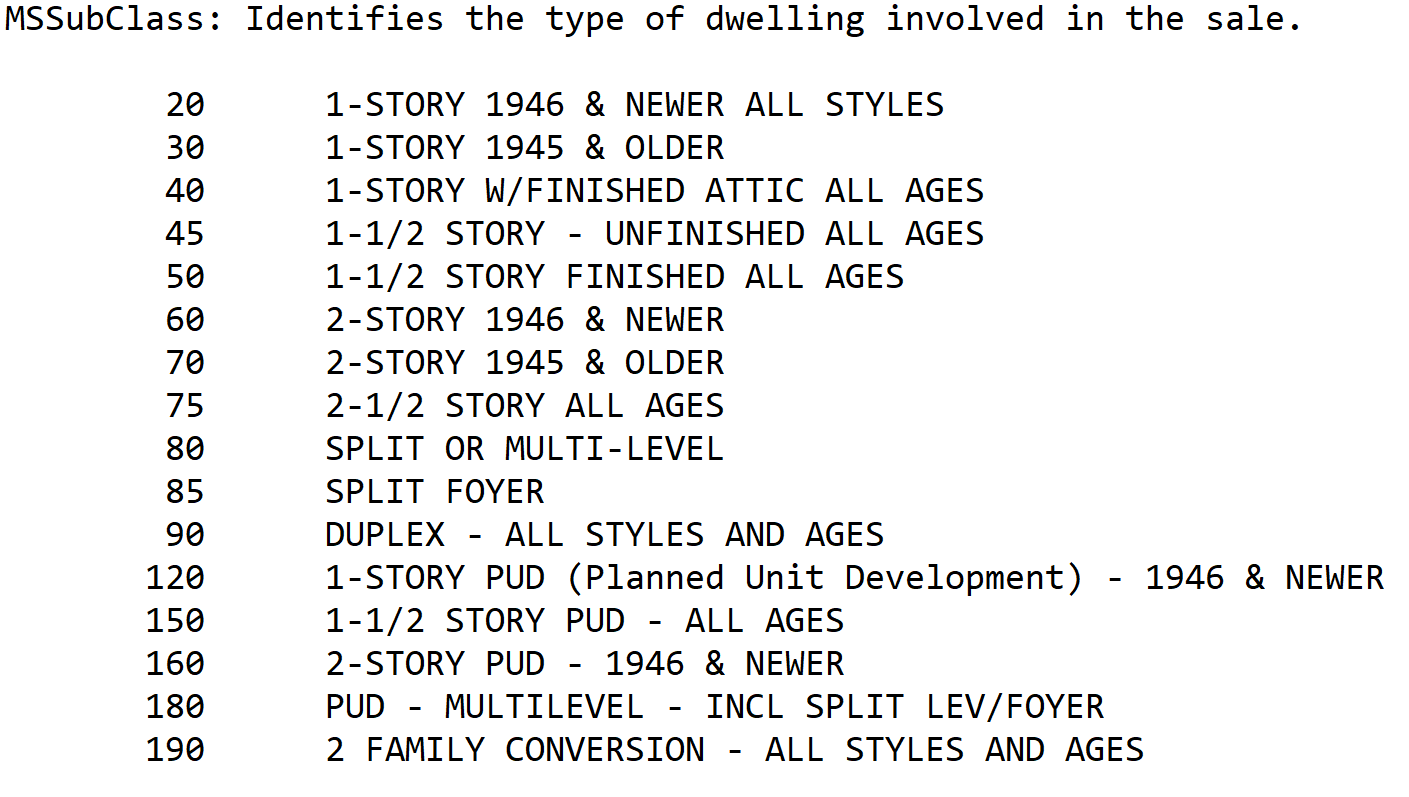
**Steps performed for cleaning of data:**

**1). Duplicates & NANs**: I started by removing duplicates from the data, checked for missing or NAN (not a number) values. It’s important to check for NANs (and not just because it’s socially moral) because these cause errors in the machine learning models.

**2). Categorical Features**: There are a lot of categorical variables that are marked as N/A when a feature of the house is nonexistent. For example, when no alley is present. I identified all the cases where this was happening across the training and test data and replaced the N/As with something more descriptive. N/As can cause errors with machine learning later down the line so get rid of them.

**3). Date Features**: For this exercise dates would be better used as categories and not integers. After all, it’s not so much the magnitude that we care about but rather that the dates represent different years. Solving this problem is simple, just convert the numeric dates to strings.

**4). Decoded Variables**: Some categorical variables had been number encoded. See the example below.



The problem here is that the machine learning algorithm could interpret the magnitude of the number to be important rather than just interpreting it as different categories of a feature.

Assumptions done and next action plans taken over that:

I approached this problem with three machine learning models. Decision tree, random forest and gradient boosting machines. I used the decision tree as my baseline model then built on this experience to tune my candidate models. This approach saves a lot of time as decision trees are quick to train and can give you an idea of how to tune the hyperparameters for my candidate models.

[**Decision Tree**](https://towardsdatascience.com/https-medium-com-lorrli-classification-and-regression-analysis-with-decision-trees-c43cdbc58054)— A tree algorithm used in machine learning to find patterns in data by learning decision rules.

[**Random Forest**](https://medium.com/swlh/random-forest-and-its-implementation-71824ced454f)— It uses multiple independent decision trees in parallel to learn from data and aggregates their predictions for an outcome.

[**Gradient Boosting Machines**](https://towardsdatascience.com/understanding-gradient-boosting-machines-9be756fe76ab) — A type of boosting method that uses a combination of decision tree in series. Each tree is used to predict and correct the errors by the preceding tree additively.

Random forests and gradient boosting can turn individually weak decision trees into strong predictive models. They’re great algorithms to use if we have small training data sets like the one, we have.

In the training stage, we’ll tune our model hyperparameters.

**Model Bias** — Models that underfit the training data leading to poor predictive capacity on unseen data. Generally, the simpler the model the higher the bias.

**Model Variance** — Models that overfit the training data leading to poor predictive capacity on unseen data. Generally, the more complexity in the model the higher the variance.

**Hyperparameters**: Hyperparameters help us adjust the complexity of our model. There are some best practices on what hyperparameters one should tune for each of the models.

**Grid search**: Choosing the range of your hyperparameters is an iterative process. With more experience we’ll begin to get a feel for what ranges to set. The good news is once we’ve chosen our possible hyperparameter ranges, grid search allows us to test the model at every combination of those ranges.

**Cross validation**: Models are trained with a 5-fold cross validation. A technique that takes the entirety of your training data, randomly splits it into train and validation data sets over 5 iterations.

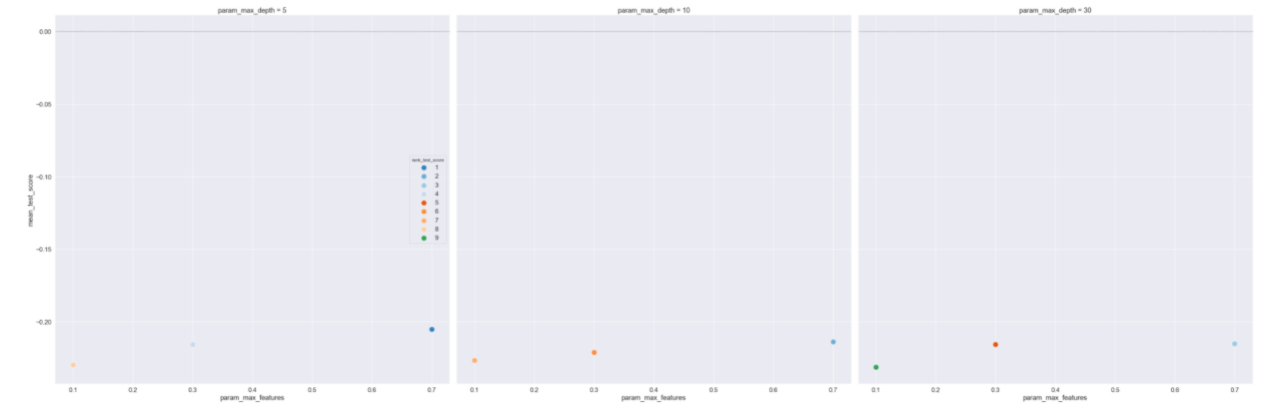
**Implementation**: SciKit Learn helps us bring together hyperparameter tuning and cross validation with ease in using GridSearchCv. It gives us options to view the results of each of our training runs.

* Data Inputs- Logic- Output Relationships

We can use data visualisation to see the results of each of our candidate models. Our performance metric will be the negative root mean squared error (NRMSE).

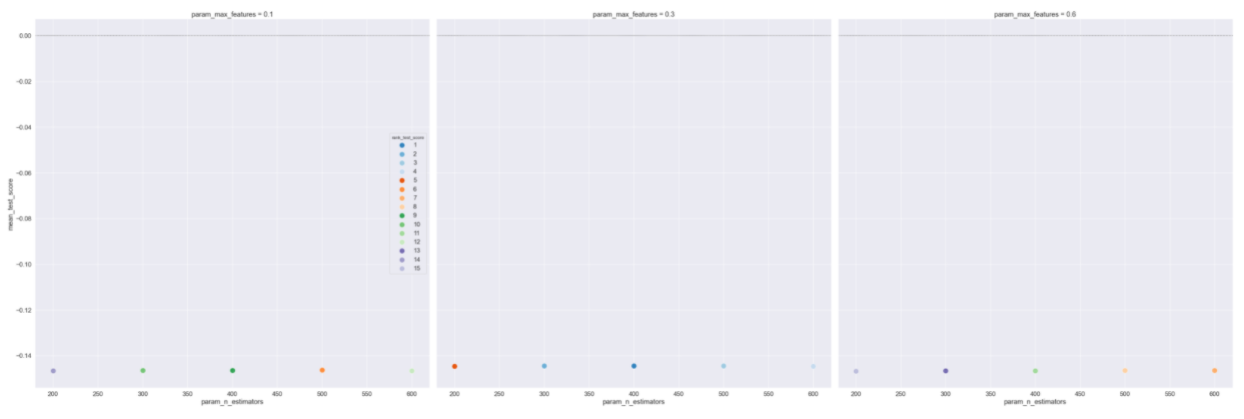
**Decision tree**

Predictably this was our worst performing method. Our best decision tree score -0.205 NRMSE. Tuning the hyperparameters didn’t appear to make much of a difference to the model’s however it trained in under 2 seconds. There is definitely some scope to assess a wider range of hyperparameters.



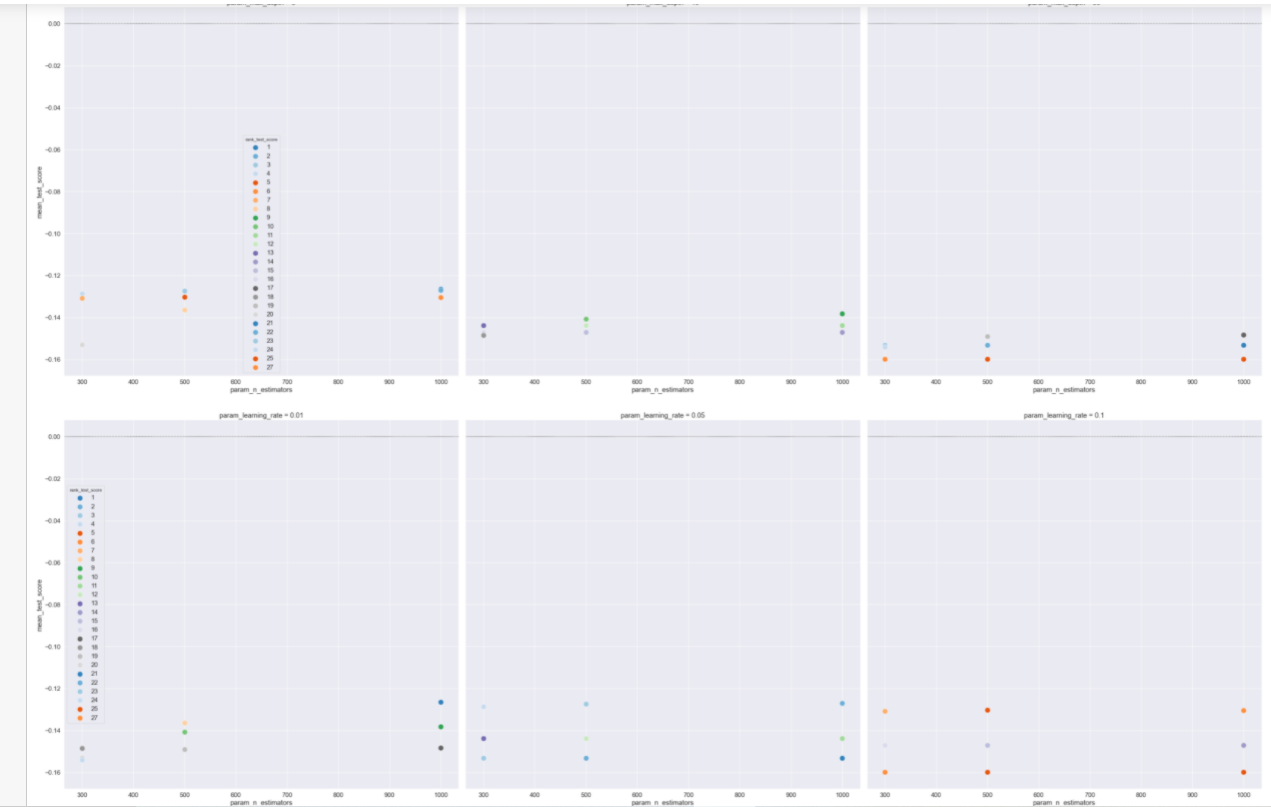
**Random Forest**

Our random forest model was a marked improvement on our decision tree with a NRMSE of -0.144. The model took around 62 seconds to train.



**Gradient Boosting Machine**

This was our best performer by a significant amount with a NRMSE of -0.126. The hyperparameters significantly impact the results illustrating that we must be incredibly careful in how we tune these more complex models. The model took around 241 seconds to train.



* Hardware and Software Requirements and Tools Used

I used Python and Jupyter notebooks for the competition. Jupyter notebooks are popular among data scientist because they are easy to follow and show our working steps.

**Libraries**: These are frameworks in python to handle commonly required tasks. I Implore any budding data scientists to familiarise themselves with these libraries:

[*Pandas*](https://pandas.pydata.org/)*— For handling Structured Data*

[*Scikit Learn*](https://scikit-learn.org/stable/)*— For Machine Learning*

[*NumPy*](https://numpy.org/)*— For Linear Algebra and Mathematics*

[*Seaborn*](https://seaborn.pydata.org/)*— For Data Visualization*

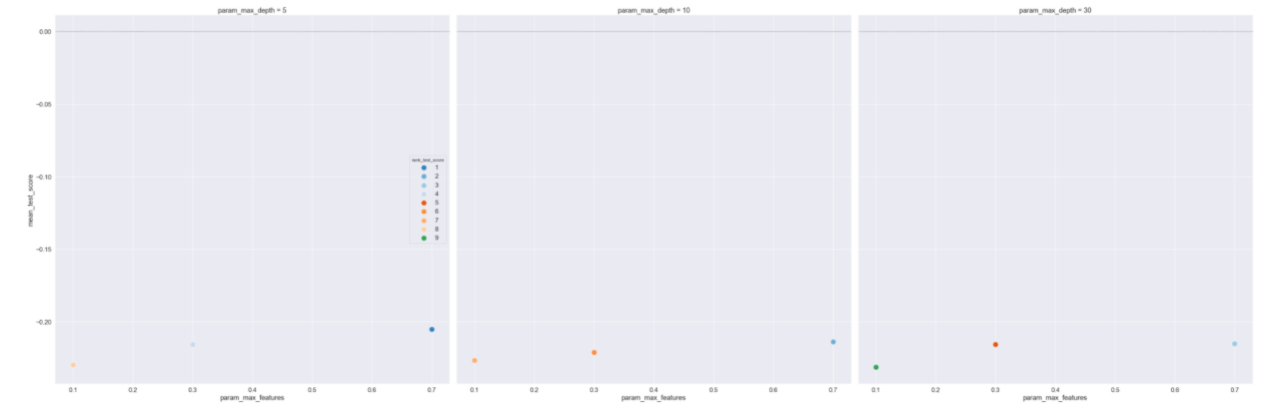
**Model/s Development and Evaluation**

* Identification of possible problem-solving approaches (methods)

We used data visualisation to see the results of each of our candidate models. Our performance metric will be the negative root mean squared error (NRMSE).

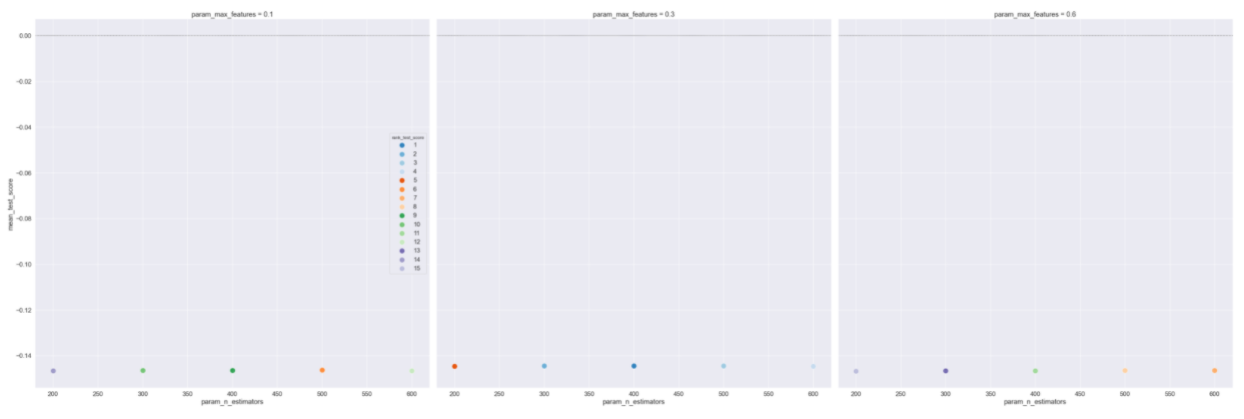
**Decision tree**

Predictably this was our worst performing method. Our best decision tree score -0.205 NRMSE. Tuning the hyperparameters didn’t appear to make much of a difference to the model’s however it trained in under 2 seconds. There is definitely some scope to assess a wider range of hyperparameters.



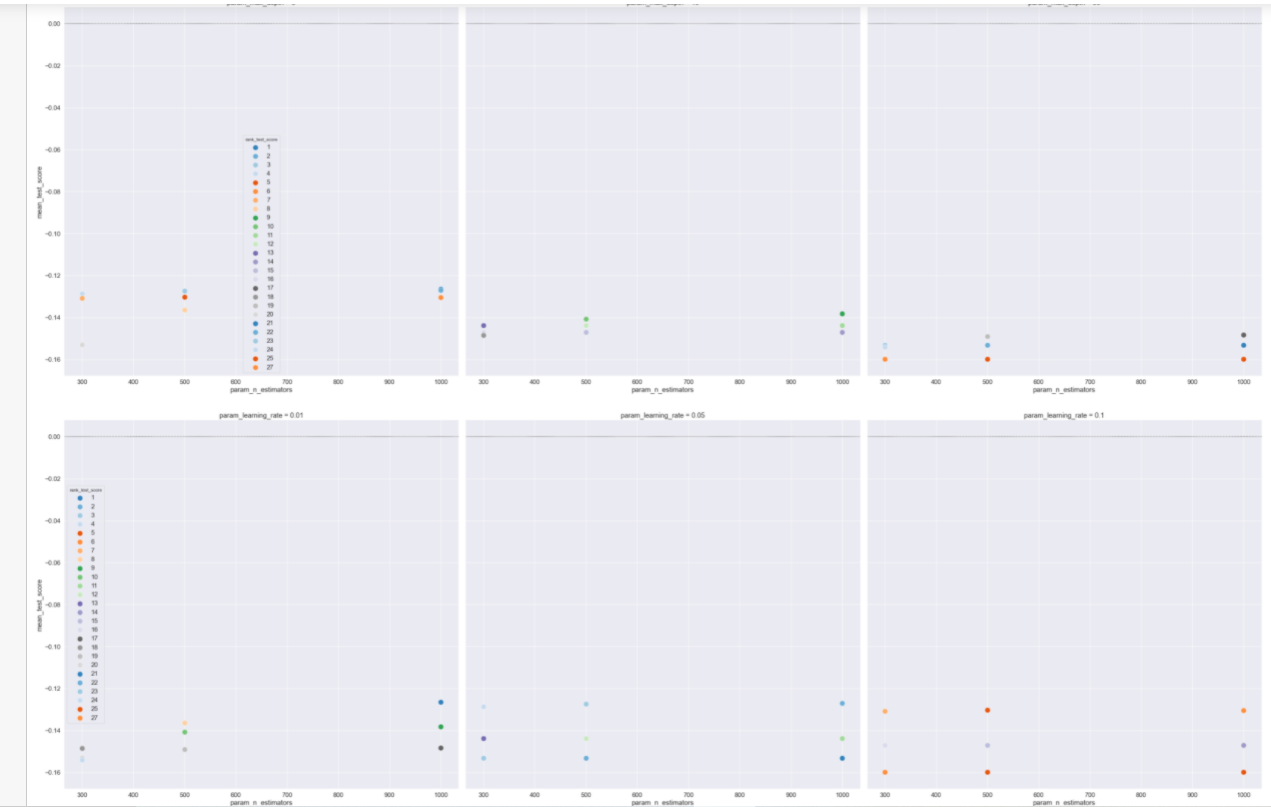
**Random Forest**

Our random forest model was a marked improvement on our decision tree with a NRMSE of -0.144. The model took around 62 seconds to train.



**Gradient Boosting Machine**

This was our best performer by a significant amount with a NRMSE of -0.126. The hyperparameters significantly impact the results illustrating that we must be incredibly careful in how we tune these more complex models. The model took around 241 seconds to train.



* Testing of Identified Approaches (Algorithms)

Machine learning tasks are usually split into three categories; Supervised, Unsupervised and Reinforcement. For this competition, our task is Supervised learning.

**Supervised learning** uses examples and labels to find patterns in data.

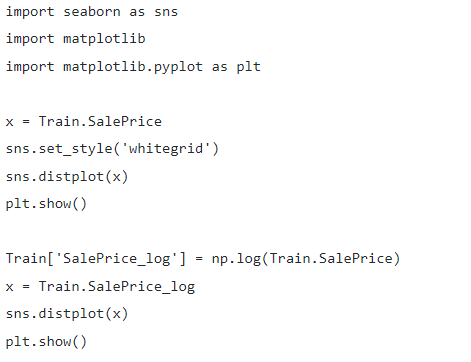
It’s easy to recognise the type of machine learning task in front of you from the data you have and your objective. We’ve been given housing data consisting of features and labels, and we’re tasked with predicting the labels for houses outside of our training data.

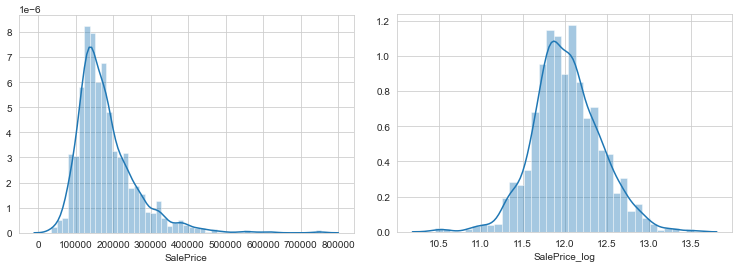
* Run and evaluate selected models

Describe all the algorithms used along with the snapshot of their code and what were the results observed over different evaluation metrics.

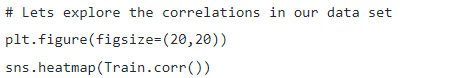
**Labels**: I plotted sales price on a histogram. The distribution of sale prices is right skewed, something that is expected. In your neighbourhood it might not be unusual to see a few houses that are relatively expensive.

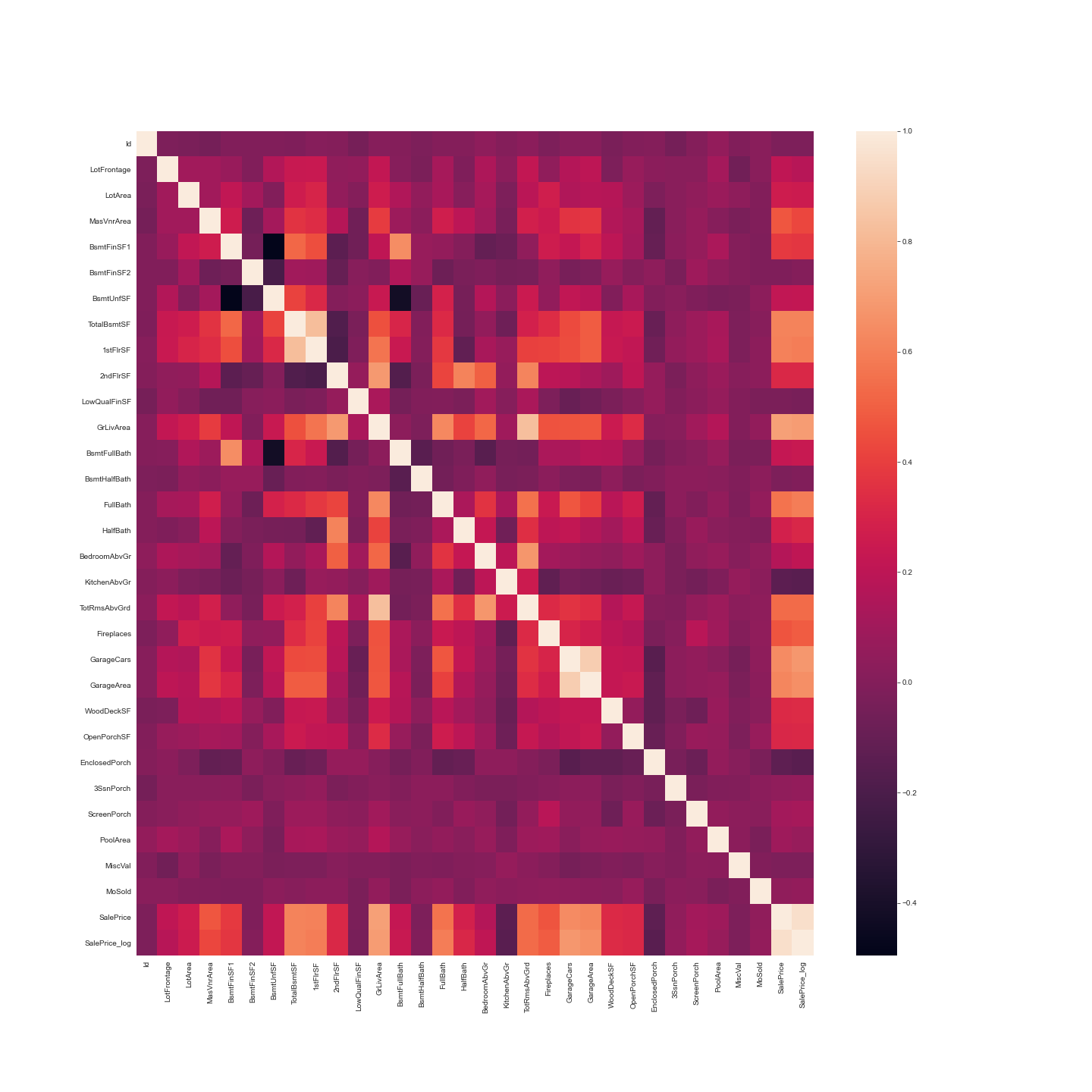
**Outliers** can have devastating effects on models that use loss functions minimising squared error. Instead of removing outliers try applying a transformation.





**Correlations**: It’s often good to plot a correlation matrix to give you an idea of relationships that exist in your data. It can also guide our model building. For example, if we see a lot of features are correlated with each other we might want to avoid linear regression.





The correlation measure used here is Pearson’s correlation. In our case the lighter the square the stronger the correlation between two variables.

Features related to space such as lot frontage, garage area, ground living area were all positively correlated with sale price as one might expect. The logic being that larger properties should be more expensive. No correlations look suspicious here.

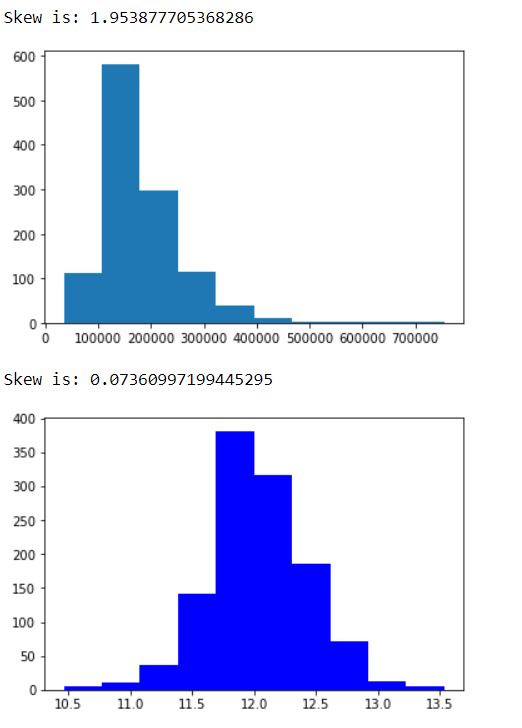
**Categorical Relations:** Sales price appears to be approximately normally distributed within each level of each category. No observations appear, untoward. Some categories contain little to no data, whilst other show little to no distinguishing ability between sales class.

* Visualizations

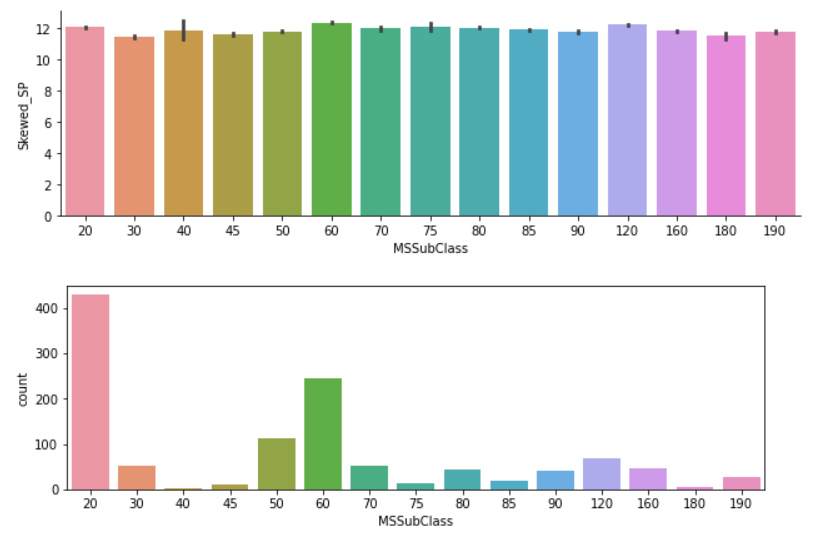
We performed various visualisations in our model.

Firstly, we check SalePrice description and found Sales price is right skewed. So, we perform log transformation so that the skewness is nearly zero.

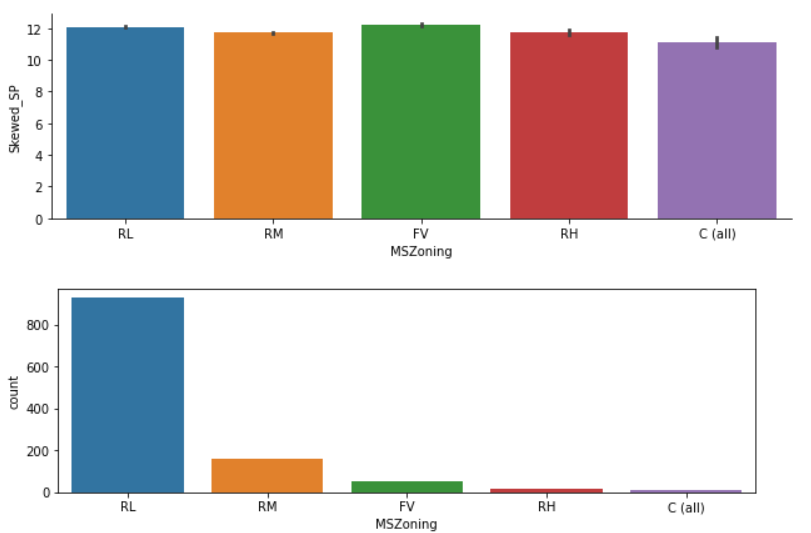
In first visualisation we observed that the SalePrice is more skewed that expected. After performing the log transformation of data, then it looks more centre aligned, which is a good observation.



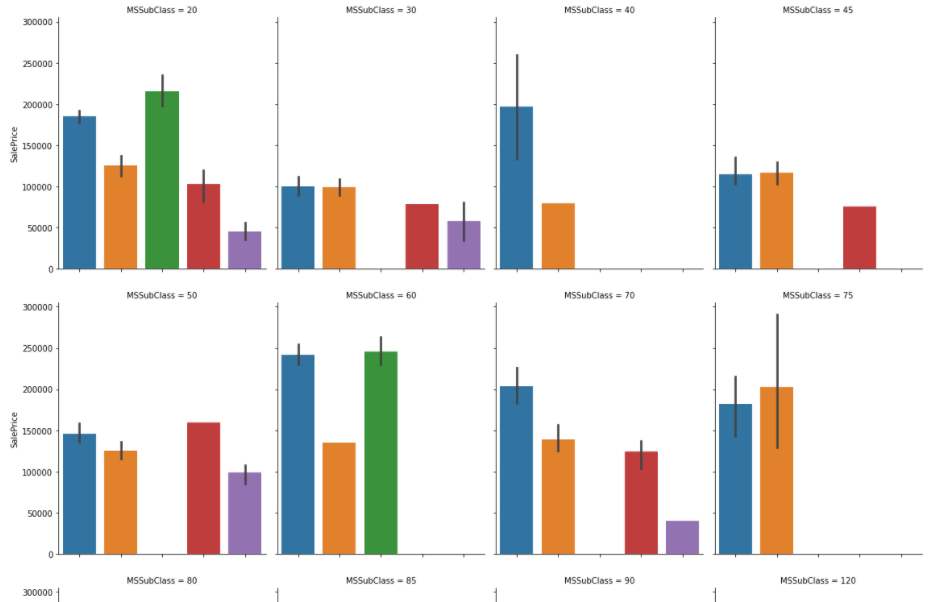
Second visualisation, we plot a Bar plot of MSSubClass w.r.t Skewed\_SP. Here we observe MSSubClass = 60 has the highest SalesPrice while the sales of houses with MSSubClass=20 is the highest.

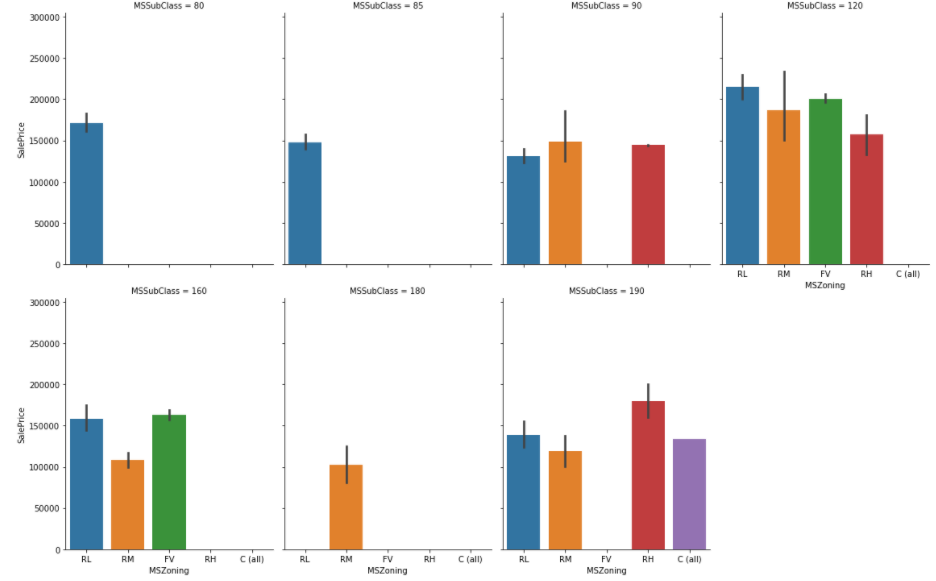


Below visualisation represents Bar plot showing the MSZoning (General zoning classification of sale) w.r.t Skewed\_SP & count of numbers against them. Here, RL (Residential Low Density) reached the maximum state until around 900. RM (residential medium density) was measured around 190. FV (Floating village residential) measured 65 and RH (Residential High density) and C (Commercial) measured 16 and 10 respectively.



Here showing the seaborn representation of different MSSubClass values w.r.t SalePrice. We see changing values on MSSubClass (Dwelling involved) variates the Sale Price which is quite obvious. Hence the representation.





20 1-STORY 1946 & NEWER ALL STYLES

30 1-STORY 1945 & OLDER

40 1-STORY W/FINISHED ATTIC ALL AGES

45 1-1/2 STORY - UNFINISHED ALL AGES

50 1-1/2 STORY FINISHED ALL AGES

60 2-STORY 1946 & NEWER

70 2-STORY 1945 & OLDER

75 2-1/2 STORY ALL AGES

80 SPLIT OR MULTI-LEVEL

85 SPLIT FOYER

90 DUPLEX - ALL STYLES AND AGES

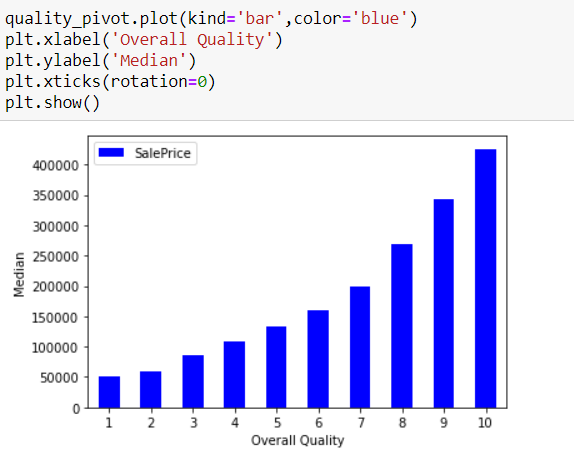
120 1-STORY PUD (Planned Unit Development) - 1946 & NSEWER

150 1-1/2 STORY PUD - ALL AGES

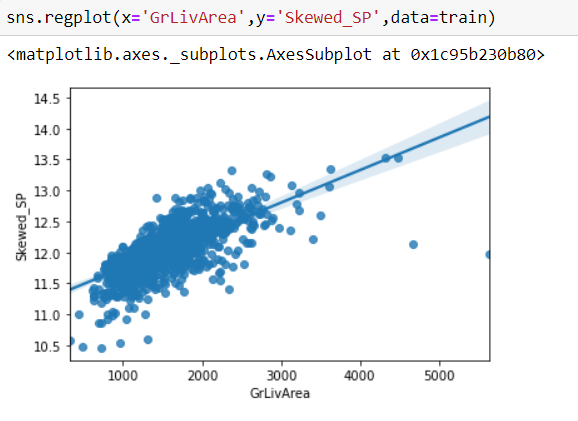
160 2-STORY PUD - 1946 & NEWER

180 PUD - MULTILEVEL - INCL SPLIT LEV/FOYER

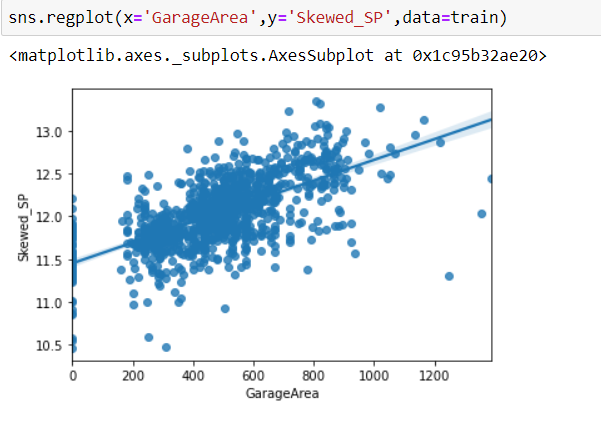
190 2 FAMILY CONVERSION - ALL STYLES AND AGES.



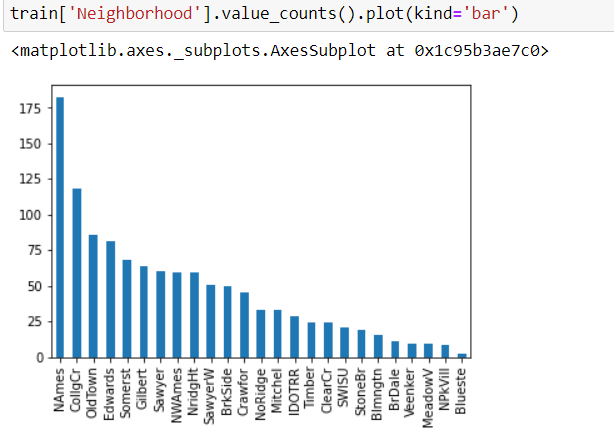
Here SalePrice varies directly with the Overall quality.



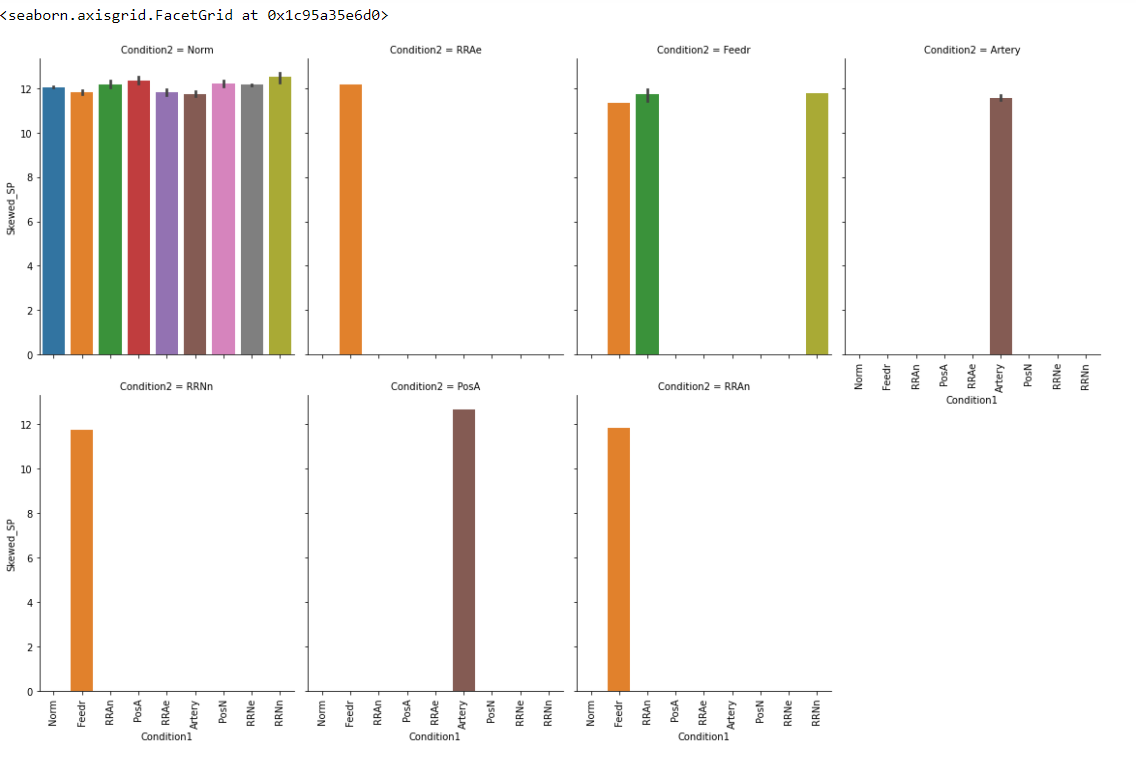
SalePrice increases as the GrLivArea increases. We will also get rid of the outliers which severely affect the prediction of the survival rate.



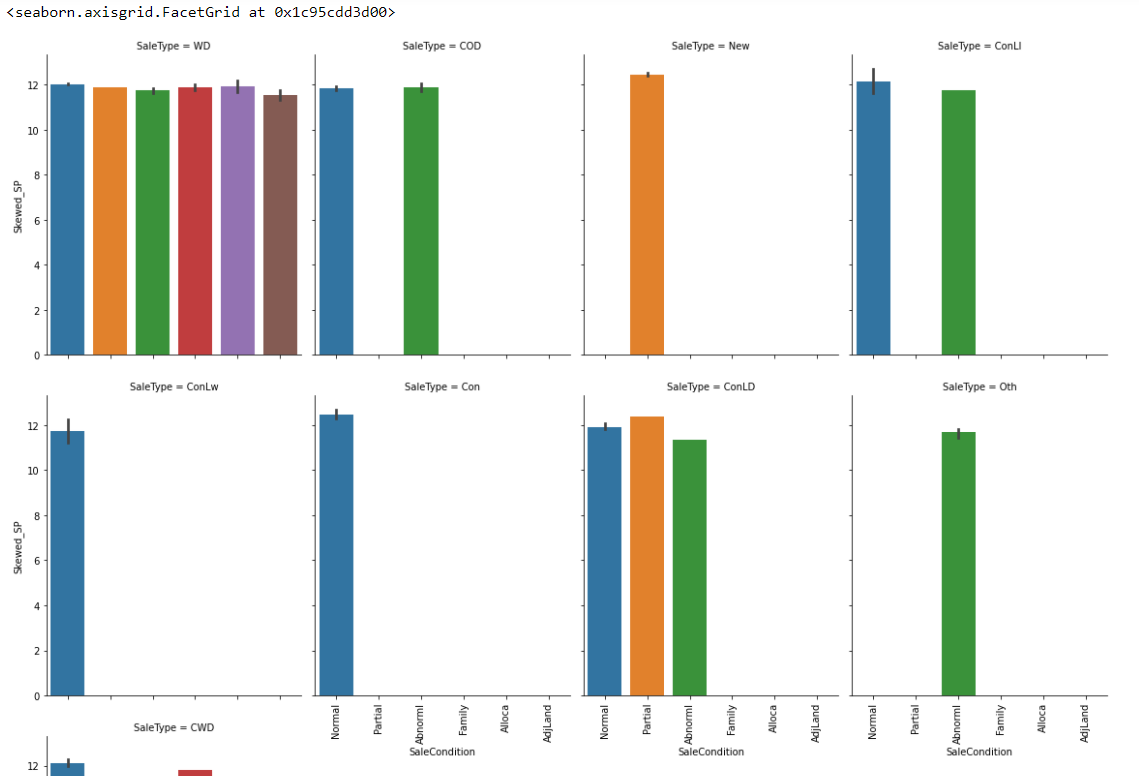
We noticed here that GarageArea and SalePrice are directly proportional.



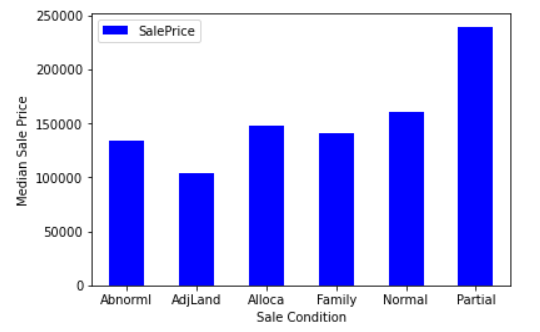
Here, we noticed the Physical locations within Ames city limits and various portions based on bar plot. This represents the value counts of different physical locations.



In this bar plot representation, we are trying to show the Proximity to various conditions (Condition1 & Condition 2). There it shows classification among different imputes as per the conditions (1 &2). Norm, RRAe, Feedr, Artery, RRNn, PosA, RRAn all these are imputes of Conditions. The data is showed w.r.t to their context. The data showed is trained data.



Next is Seaborn bar plot indicating the Sale Type and Sale Condition of the training data. The highest of which is found in Warranty Deed – Conventional followed by Home just constructed and sold & Contract Low Down.



At last, we did feature engineering (histogram) on Sale Condition and review insights of the SalePrice. Here we see Condition of sale is fluctuating based on the different categories. The highest Sale Price was with Partial (Home was not completed when last assessed) around 240000 and lowest being with AdjLand (Adjoining Land Purchase) at 110000.

* Interpretation of the Results

I believe the model used could be optimized and tuned more to add accuracy either by adding new features or engineering new features. The model can be used to predict the house prices in any geographic location by just slightly fine tuning the features and parameters.

Through Visualisations, it was made easy and possible for us to locate different models’ parameters and behaviour/ characteristics of different modules of datasets.

**Steps performed for Preprocessing & Modelling of data:**

**1). Duplicates & NANs**: I started by removing duplicates from the data, checked for missing or NAN (not a number) values. It’s important to check for NANs because these cause errors in the machine learning models.

**2). Categorical Features**: There are a lot of categorical variables that are marked as N/A when a feature of the house is nonexistent. For example, when no alley is present. I identified all the cases where this was happening across the training and test data and replaced the N/As with something more descriptive. N/As can cause errors with machine learning later down the line so get rid of them.

**3). Date Features**: For this exercise dates would be better used as categories and not integers. After all, it’s not so much the magnitude that we care about but rather that the dates represent different years. Solving this problem is simple, just convert the numeric dates to strings.

**4). Decoded Variables**: Some categorical variables had been number encoded.

**CONCLUSION**

* Key Findings and Conclusions of the Study

**Key Findings & Inferences**:

We used data visualisation to see the results of each of our candidate models. Suppose we’re not happy with our results, we might have to revisit our process at any of the stages from data cleaning to machine learning. I believe our model could be optimized and tuned more to add accuracy either by adding new features or engineering new features. The model can be used to predict the house prices in any geographic location by just slightly fine tuning the features and parameters.

Observations from problem through which we could improve our model:

**Categorical variables**: Some of our categorical features in the data have a high cardinality. Tree models can be biased to these features because of this. We might be able to improve model performance by recategorizing these high dimensional features into lower dimensions.

**Hyperparameter tuning**: We can look to widen our solutions space for our hyperparameters in the hope of finding a more optimal position. Be warned, this will require heavy computation power if you’re just working on your laptop.

* Learning Outcomes of the Study in respect of Data Science

Data visualization is the representation of data or information in a graph, chart, or other visual format. It communicates relationships of the data with images. ... Machine learning makes it easier to conduct analyses such as predictive analysis, which can then serve as helpful visualizations to present.

**Power of Visualization:** -

Visualisation of data helps us when we're performing the process over and over again to ensure our model is optimized and can generalize well. It both shortens the machine learning process and provides more accuracy for its outcome. When evaluating the models, visualizing the results of hyperparameter tuning can help data scientists narrow down the groupings of hyperparameters that are most important.

**Data Cleaning:** -

The main aim of Data Cleaning is to identify and remove errors & duplicate data, in order to create a reliable dataset. This improves the quality of the training data for analytics and enables accurate decision-making. Data cleaning is considered a foundational element of the basic data science.

Different Algorithms used: -

We use three algorithms in this model of House Price prediction.

1). Decision Trees

2). Random Forest

3). Gradient Boosting Machine

From **Decision trees**, it was our worst performing method and we got a merely score of -0.205 NRMSE. Tuning the hyperparameters didn’t appear to make much of a difference to the model.

**Random Forest** model as improved than decision trees with NRMSE of -0.144.

**Gradient Boosting** Machine was by far the best performing model with NRMSE of -0.126. The hyperparameters significantly impact the results illustrating that we must be incredibly careful in how we tune these more complex models.

Challenges faced while working on models and overcome from that:

**Categorical variables**: Some of our categorical features in the data have a high cardinality. Tree models can be biased to these features because of this. We might be able to improve model performance by recategorizing these high dimensional features into lower dimensions.

**Hyperparameter tuning**: We can look to widen our solutions space for our hyperparameters in the hope of finding a more optimal position.

<< End Of Report>>

ThankYou 😊