EXPLORATORY DATA ANALYSIS COURSE PROJECT

Ames Housing Price Prediction Dataset

IBM / COURSERA

IDDI ABDUL AZIZ

INTRODUCTION

Data examination is a time-consuming, but necessary, initial step in any analysis that researchers often overlook. With this data **Ames Housing Price Data** set, I will evaluate the impact of missing data, identifies outliers, and tests for the assumptions underlying most multivariate techniques. The objective of these data examination tasks is as much to reveal what is not apparent as it is to portray the actual data, because the “hidden” effects are easily overlooked. For example, the biases introduced by non-random missing data will never be known unless explicitly identified and remedied. My tasked is listed below;

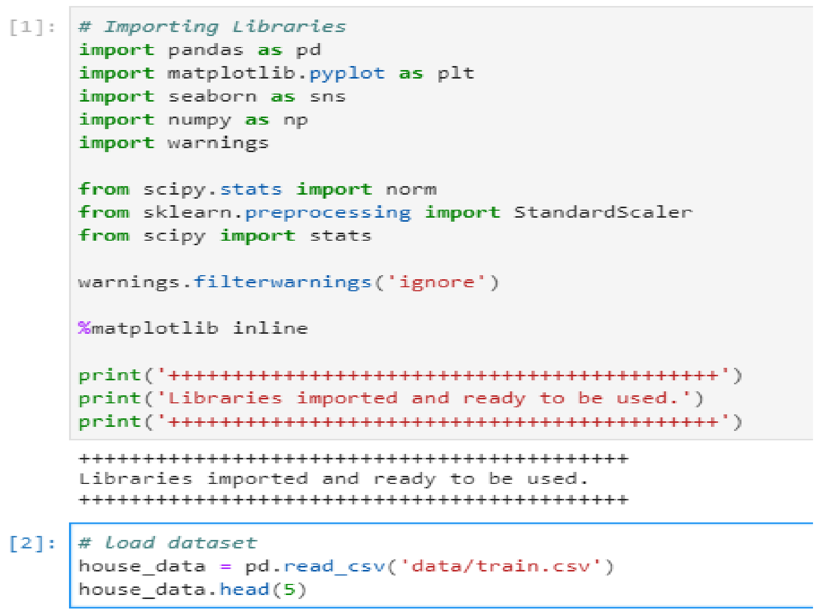
- Brief description of the data set and a summary of its attributes
- Initial plan for data exploration
- Actions taken for data cleaning and feature engineering
- Key Findings and Insights, which synthesizes the results of Exploratory Data Analysis in an insightful and actionable manner
- Formulating at least 3 hypothesis about this data
- Conducting a formal significance test for one of the hypotheses and discuss the results
- Suggestions for next steps in analysing this data
- A paragraph that summarizes the quality of this data set and a request for additional data if needed

In Data Science, 80% of time spent prepare data, 20% of time spent complain about need for prepare data. Let's see how we spend 80% of our time. I did my best to perform a comprehensive data analysis. This report is far from being an exhaustive study on data analysis. It shows how I applied some general principles of data analysis to the **Ames Housing Prices** data set. I will be used Jupyter Lab throughout the project and concentrate more on EDA and Data Pre-processing than modelling.

1. BRIEF DESCRIPTION OF THE DATA SET AND SUMMARY OF ITS ATTRIBUTES

Understanding The Data

Let me first import libraries and dataset



```
[1]: # Importing Libraries
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
import numpy as np
import warnings

from scipy.stats import norm
from sklearn.preprocessing import StandardScaler
from scipy import stats

warnings.filterwarnings('ignore')

%matplotlib inline

print('+++++++++++++++++++++++++++++++++++++++++++++')
print('Libraries imported and ready to be used.')
print('+++++++++++++++++++++++++++++++++++++++++++++')
```

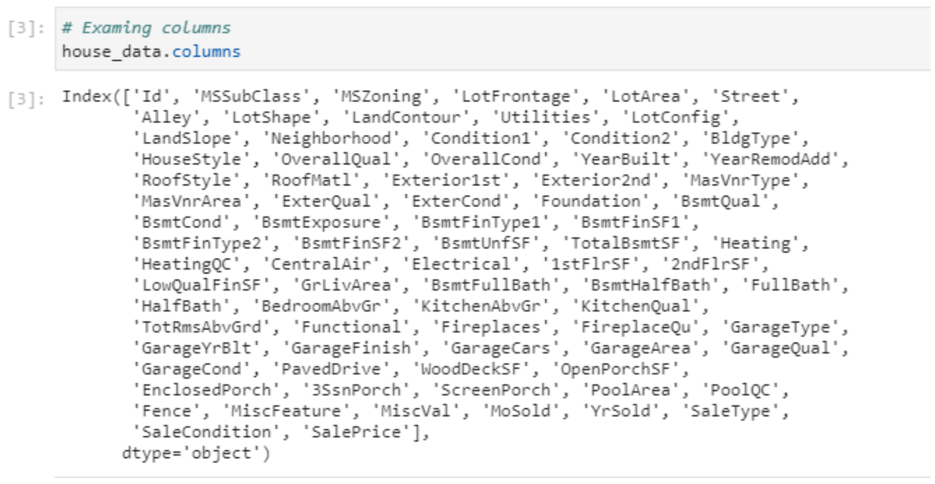
```
+++++++++++++++++++++++++++++++++++++++++++++
Libraries imported and ready to be used.
+++++++++++++++++++++++++++++++++++++++++++++
```

```
[2]: # Load dataset
house_data = pd.read_csv('data/train.csv')
house_data.head(5)
```

Fathoming the data is a crucial aspect of Machine Learning, the better we understand the data, the more we can use to help ourselves in manipulating it to make better predictions. So, let’s look at the available features.

Initial Plan for Data Exploration

In order to understand the data, I will look at each variable and try to understand its meaning and relevance to this problem. I know this is time-consuming, but it will give a flavour of the data set.



```
[3]:  # Examing columns
      house_data.columns

[3]:  Index(['Id', 'MSSubClass', 'MSZoning', 'LotFrontage', 'LotArea', 'Street',
             'Alley', 'LotShape', 'LandContour', 'Utilities', 'LotConfig',
             'LandSlope', 'Neighborhood', 'Condition1', 'Condition2', 'BldgType',
             'HouseStyle', 'OverallQual', 'OverallCond', 'YearBuilt', 'YearRemodAdd',
             'RoofStyle', 'RoofMatl', 'Exterior1st', 'Exterior2nd', 'MasVnrType',
             'MasVnrArea', 'ExterQual', 'ExterCond', 'Foundation', 'BsmtQual',
             'BsmtCond', 'BsmtExposure', 'BsmtFinType1', 'BsmtFinSF1',
             'BsmtFinType2', 'BsmtFinSF2', 'BsmtUnfSF', 'TotalBsmtSF', 'Heating',
             'HeatingQC', 'CentralAir', 'Electrical', '1stFlrSF', '2ndFlrSF',
             'LowQualFinSF', 'GrLivArea', 'BsmtFullBath', 'BsmtHalfBath', 'FullBath',
             'HalfBath', 'BedroomAbvGr', 'KitchenAbvGr', 'KitchenQual',
             'TotRmsAbvGrd', 'Functional', 'Fireplaces', 'FireplaceQu', 'GarageType',
             'GarageYrBlt', 'GarageFinish', 'GarageCars', 'GarageArea', 'GarageQual',
             'GarageCond', 'PavedDrive', 'WoodDeckSF', 'OpenPorchSF',
             'EnclosedPorch', '3SsnPorch', 'ScreenPorch', 'PoolArea', 'PoolQC',
             'Fence', 'MiscFeature', 'MiscVal', 'MoSold', 'YrSold', 'SaleType',
             'SaleCondition', 'SalePrice'],
            dtype='object')
```

To have some discipline in the analysis, I can work with ancient technology and create an Excel spreadsheet with the following columns:

- **Variable** - Variable's name.
- **Type** - Variables can be 'numerical' or 'categorical'. By 'numerical' we mean variables whose values are numbers (e.g., 42). By 'categorical' we mean variables whose values are categories (e.g., 'Excellent').
- **Segment** - We can organize our variables according to three segments: building, area, and location. When we say 'building', we mean a variable that is related to the physical characteristics of the building (e.g., 'OverallQual'). When we say 'space', we mean a variable that reports area properties of the house (e.g., 'TotalBsmtSF'). Finally, when we say 'location', we mean a variable that gives information about the place where the house is located (e.g., 'Neighborhood').
- **Expectation** - Our expectation about the influence of the variable in 'SalePrice'. We can use a categorical scale, such as 'High', 'Medium', and 'Low'.
- **Conclusion** - Our conclusions about the importance of the variable after taking a look at the data. We can use the same categorical scale that we used in 'Expectation'.
- **Comments** - Any general comments that occurred.

While 'Type' and 'Segment' is just for future reference, the column 'Expectation' is important because it will help us to develop a “sixth sense.” To fill this column, we should read the description of all variables and, for each one, ask ourselves:

- If we would buy a house, would this variable influence our decision? (e.g., When we think about the house of our dreams, do we care about its 'Masonry veneer type'?).
- If so, how important this variable would be? (e.g. What is the impact of having 'Excellent' material on the exterior instead of 'Poor'? And of having 'Excellent' instead of 'Good'?).
- Does this variable repeat information given by another variable? (e.g., If 'LandContour' gives the flatness of the property, do we really need to know the 'LandSlope'?).

After this daunting exercise, I filtered the spreadsheet and look carefully at the variables with 'High' 'Expectation'. Then, I rushed into some scatter plots between those variables and 'SalePrice', filling in the 'Conclusion' column (which just matters to correct expectations).

I went through this process and concluded that the following variables seem to be important:

- OverallQual (which is a variable that I don't like because I don't know how it was computed; a funny exercise

would be to predict 'OverallQual' using all the other variables available).
- YearBuilt.
- TotalBsmtSF.
- GrLivArea.

I ended up with two “building” variables ('OverallQual' and 'YearBuilt') and two “area” variables ('TotalBsmtSF' and 'GrLivArea'). This might be a little bit unexpected as it goes against the real estate mantra that all that matters is “location, location, and location.”

It is possible that this quick data examination process was a bit harsh for categorical variables. For example, I expected the 'Neigborhood' variable to be more relevant, but after examining the data I ended up excluding it. Maybe this is related to the use of scatter plots instead of box plots, which are more suitable for categorical variables visualization. The way we visualize data often influences our conclusions.

However, the main point of this exercise was to reflect a little on our data, so I think I achieved the goal.

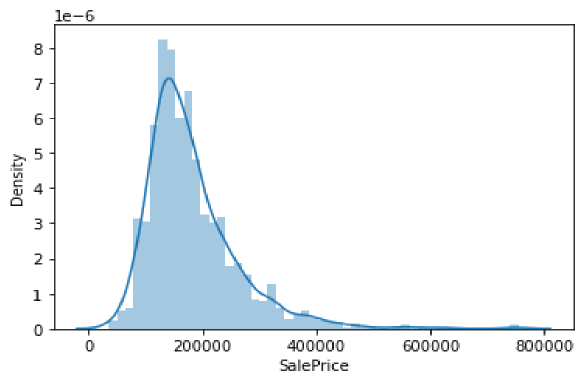
Actions taken for Data cleaning and Feature Engineering

'SalePrice' is the reason of our quest, thus our target. It's like when we're going to a party. We always have a reason to be there. So, let's start our journey with a story. The story of “How we met 'SalePrice'.”

```
[6]: # Descriptive statistics summary
     house_data['SalePrice'].describe()
```

```
[6]: count      1460.000000
     mean     180921.195890
     std       79442.502883
     min       34900.000000
     25%      129975.000000
     50%      163000.000000
     75%      214000.000000
     max      755000.000000
     Name: SalePrice, dtype: float64
```

```
[7]: # Plot histogram
     sns.distplot(house_data['SalePrice']);
```



Great! With the seaborn visualisation outlook, we are able to see that it:

- *Deviate from the normal distribution.*
- *Have appreciable positive skewness.*
- *Show peakedness.*

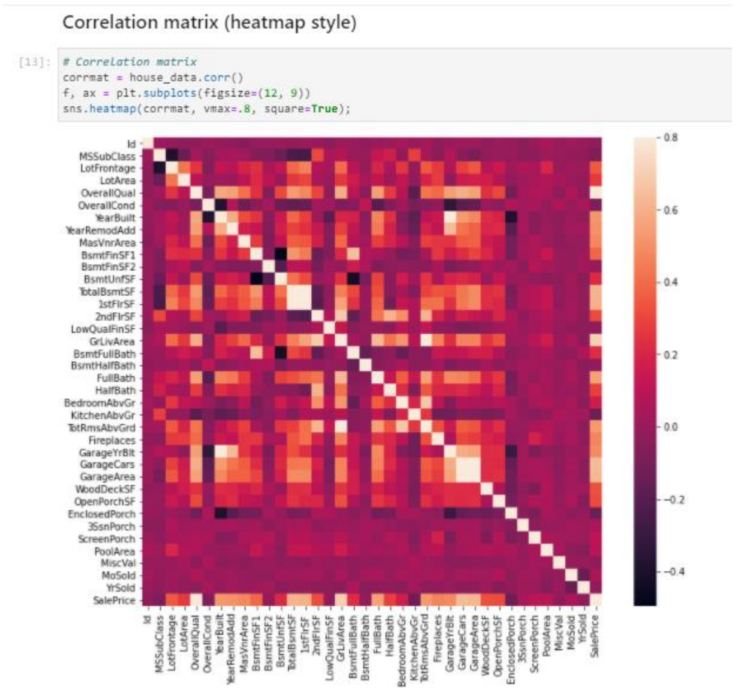
This shows that ‘SalePrice’ does not follow normal distribution. It has positive skewness. It deviates from normal distribution. The SalePrice is not linear, which means we cannot find a straight line that would fit through the SalePrice data.

In the very beginning there was nothing except for our data. Which we'll try our best to analyse and explore in time.

To explore the data, we will start with some practical recipes to make sense of our “data”:

- Correlation matrix (heatmap style).
- 'SalePrice' correlation matrix (zoomed heatmap style).

- Scatter plots between the most correlated variables.

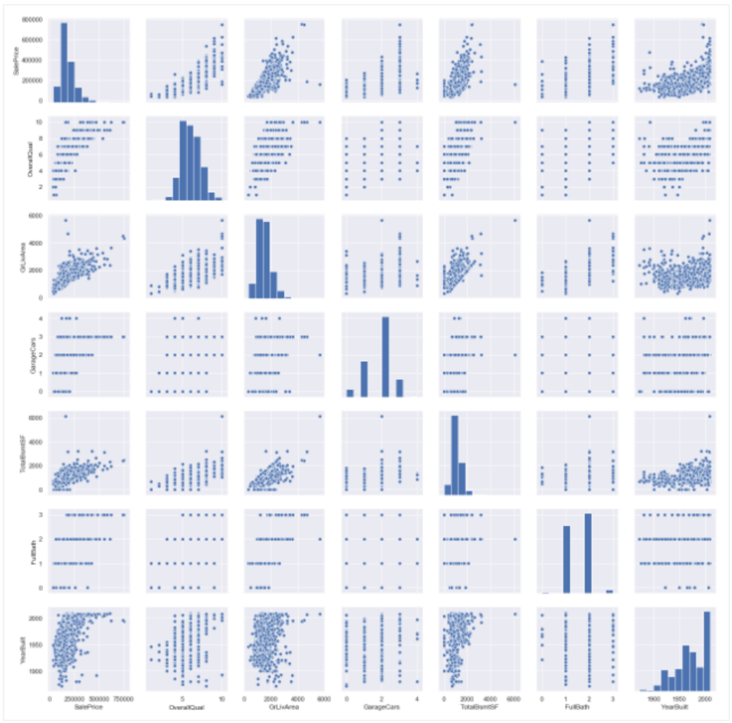

Correlation matrix (heatmap style)

```
[13]: # Correlation matrix
corrmat = house_data.corr()
f, ax = plt.subplots(figsize=(12, 9))
sns.heatmap(corrmat, vmax=.8, square=True);
```

Accordingly, the visualisation says it all, these are the variables whose correlation with 'SalePrice' is strongest:

- **'OverallQual', 'GrLivArea', and 'TotalBsmtSF'** are strongly correlated with 'SalePrice'.
- **'GarageCars' and 'GarageArea'** are also some of the most strongly correlated variables. I kept 'GarageCars' since its correlation with 'SalePrice' is stronger.
- **'TotalBsmtSF' and '1stFloor'** also seem to be twin brothers
- **'FullBath'** is relevant.
- **'TotRmsAbvGrd' and 'GrLivArea',** twin brothers again.
- **It seems that 'YearBuilt'** is slightly correlated with 'SalePrice'.

Let's move on to scatter plots.



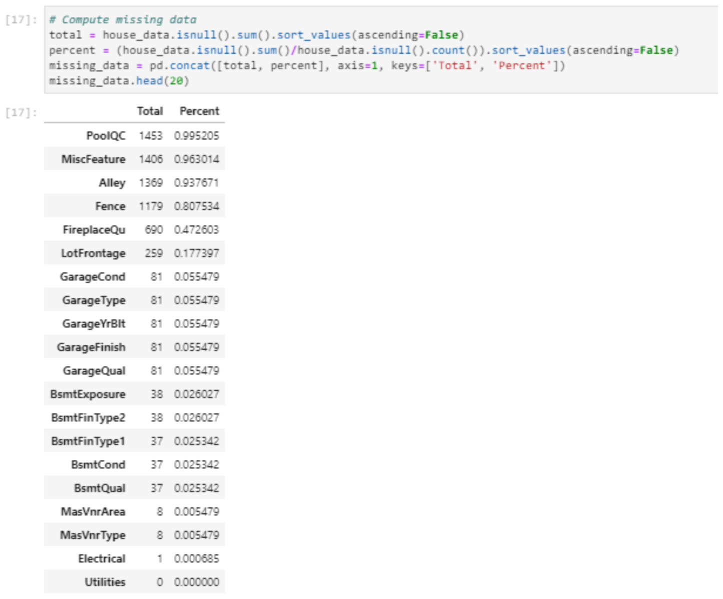
This mega scatter plot gives a reasonable idea about the relationships between variables.

One of the figures that I find interesting is the one between 'TotalBsmtSF' and 'GrLiveArea'. In this figure, we can see the points drawing a linear line, almost acting like a border. It makes sense that most of the points stay below this border. The basement area can be at least equal to the above grade area, but the basement area is not expected to be larger than the above grade area.

Another interesting plot is the one that relates 'YearBuilt' to 'SalePrice'. In the bottom of the "points cloud", we can see what appears to be a shy exponential function. In the same fashion, we can see an exponential function in the upper limit of the "points cloud". Finally, notice how the points related to the last few years tend to be above the lower limit and to break the upper limit (prices are increasing faster now as you may have noticed the last time you tried to buy a house).

Missing Data

Important questions when thinking about missing data:



```
[17]: # Compute missing data
      total = house_data.isnull().sum().sort_values(ascending=False)
      percent = (house_data.isnull().sum()/house_data.isnull().count()).sort_values(ascending=False)
      missing_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])
      missing_data.head(20)
```

[17]:

	Total	Percent
PoolQC	1453	0.995205
MiscFeature	1406	0.963014
Alley	1369	0.937671
Fence	1179	0.807534
FireplaceQu	690	0.472603
LotFrontage	259	0.177397
GarageCond	81	0.055479
GarageType	81	0.055479
GarageYrBlt	81	0.055479
GarageFinish	81	0.055479
GarageQual	81	0.055479
BsmtExposure	38	0.026027
BsmtFinType2	38	0.026027
BsmtFinType1	37	0.025342
BsmtCond	37	0.025342
BsmtQual	37	0.025342
MasVnrArea	8	0.005479
MasVnrType	8	0.005479
Electrical	1	0.000685
Utilities	0	0.000000

- How significant is the missing data?
- Is the data missing at random or not?

The answer to these questions defines how we should deal with missing data. For example, if the missing data is not that significant, we can just delete all the observations that have missing data. However, this may not be a great solution if we are already lacking data. Also, if the data is not missing at random, to delete observations with missing data may bias our analysis.

I'll start by considering that when data of one variable is missing in more than 15% of the data set, that variable should be deleted. Thus, we should delete a set of variables (e.g., 'PoolQC', 'MiscFeature', 'Alley'). Will we miss this data? I don't think so. None of these variables seem to be important, since most of them are not aspects in which we think about when we are buying a house (maybe that's the reason why data is missing). Moreover, thinking a little bit more, we can realize that variables like 'PoolQC', 'MiscFeature', and 'FireplaceQu' are strong candidates to be outliers, which should reinforce our desire to eliminate them.

With regard to the remaining cases, we can see that 'Garage*X*' variables have the same amount of missing data. I bet missing data refers to the same set of observations. Since the most important information regarding garages is expressed by 'GarageCars', and considering that we are just talking about 5% of missing data, I'll delete the mentioned 'Garage*X*' variables. The same logic applies to the 'Bsmt*X' variables.*
Regarding 'MasVnrArea' and 'MasVnrType', we can consider that these variables are not essential. Furthermore, they have a strong correlation with 'YearBuilt' and 'OverallQual', which are already considered. Thus, we will not lose information if we delete 'MasVnrArea' and 'MasVnrType'.

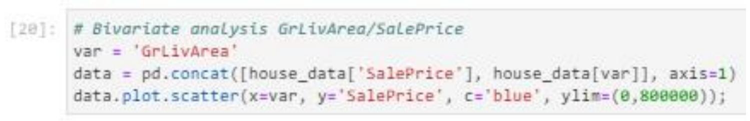
Finally, we have one missing observation in 'Electrical'. It is just one observation, so we'll delete this observation and keep the variable.

In summary, to handle missing data, we'll delete all the variables with missing data, except the variable 'Electrical'. In 'Electrical', we'll just delete the observation with missing data.

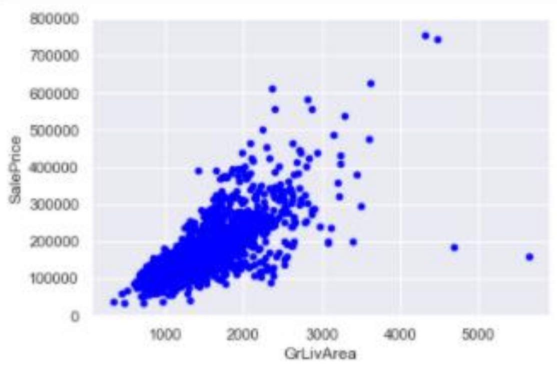
Outliers

Outliers are observations that are clearly different from other observations in the data set. It is important to identify outliers because they can affect our models and they can give us important insights about specific behaviours happening in the reality hiding behind our data set.

Here, we'll just do a quick analysis using the standard deviation of 'SalePrice' and a set of scatter plots. Using Bivariate analysis, there’s always something to discover.



```
[20]: # Bivariate analysis GrLivArea/SalePrice
      var = 'GrLivArea'
      data = pd.concat([house_data['SalePrice'], house_data[var]], axis=1)
      data.plot.scatter(x=var, y='SalePrice', c='blue', ylim=(0,800000));
```



Consequences of the new perspective:

- The two values with bigger 'GrLivArea' seem strange and they are not following the crowd. We can speculate why this is happening. Maybe they refer to agricultural area and that could explain the low price. I'm not sure about this but I'm quite confident that these two points are not representative of the typical case. Therefore, we'll define them as outliers and delete them.
- The two observations in the top of the plot look like two special cases; however, they seem to be following the trend. For that reason, we will keep them.

Hypothesis Testing

Testing the assumptions underlying the statistical bases for multivariate analysis is a step towards answering this question. We already discovered a lot about our data. Now, it's time to dig deeper and understand if our data complies with the statistical assumptions that enables us to apply multivariate techniques.

The four assumptions to test:

- **Normality** - Normality means that our data follows a normal distribution. This is important because several statistic tests rely on this (e.g., t-statistics). In this exercise, we'll just check univariate normality for 'SalePrice' (which is a limited approach).

- **Homoscedasticity** - Homoscedasticity refers to the “assumption that dependent variable(s) exhibit equal levels of variance across the range of predictor variable(s). Homoscedasticity is desirable because we want the error term to be the same across all values of the independent variables.

- **Linearity**- The most common way to assess linearity is to examine scatter plots and search for linear patterns. If patterns are not linear, it would be worthwhile to explore data transformations. However, I'll not get into this because most of the scatter plots appear to have linear relationships.

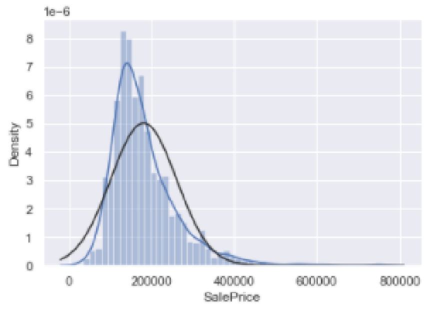
- **Absence of correlated errors** - Correlated errors, like the definition suggests, happen when one error is correlated to another one. This is common in time series because some patterns are time related. I'll also not get into this

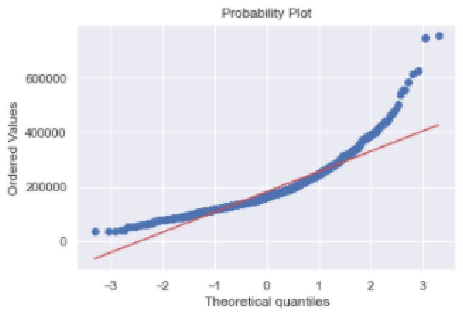
Conducting a formal significance test of the hypotheses

Normality – In the search of normality, the goal is to test 'SalePrice' in a very lean way. I'll do this, paying attention to:

- **Histogram** - Kurtosis and skewness.
- **Normal probability plot** - Data distribution should closely follow the diagonal that represents the normal distribution.

```
[23]: # Histogram and normal probability plot
      sns.distplot(house_data['SalePrice'], fit=norm);
      fig = plt.figure()
      res = stats.probplot(house_data['SalePrice'], plot=plt)
```



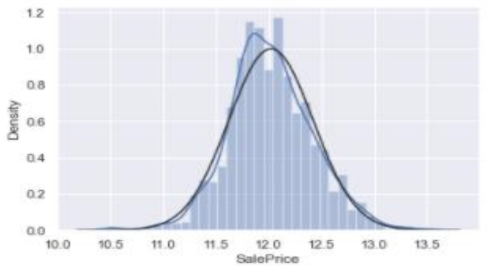


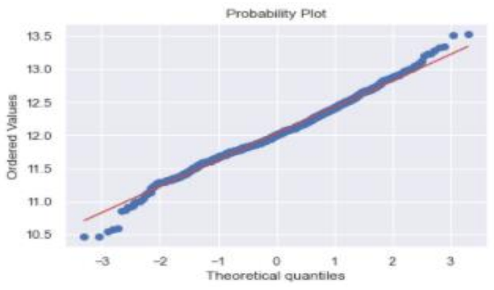
'SalePrice' is not normal. It shows 'peakedness', positive skewness, and does not follow the diagonal line.

I used simple data transformation to solve the problem. This is one of the awesome things that one can learn in statistics books: in case of positive skewness, log transformations usually work well.

```
[25]: # Apply log transformation
      house_data['SalePrice'] = np.log(house_data['SalePrice'])
```

```
[26]: # Transformed histogram and normal probability plot
      sns.distplot(house_data['SalePrice'], fit=norm);
      fig = plt.figure()
      res = stats.probplot(house_data['SalePrice'], plot=plt)
```



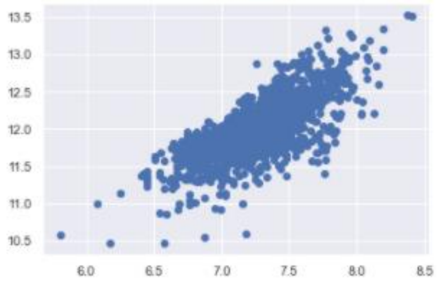


Homoscedasticity -

We can test homoscedasticity in a visual way, using scatter plots. Departures from an equal dispersion are shown by such shapes as cones (small dispersion at one side of the graph, large dispersion at the opposite side), or diamonds (a large number of points at the centre of the distribution).

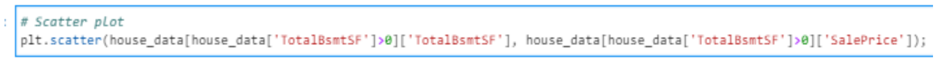
Starting with 'GrLivArea' and 'SalePrice'

```
[36]: # Scatter plot
      plt.scatter(house_data['GrLivArea'], house_data['SalePrice']);
```

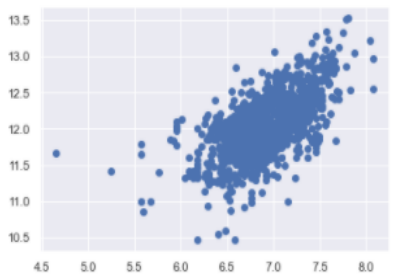


The log transformation of this will have a conic shape, now, the current scatter plot doesn't have a conic shape. That's the power of normality. Just by ensuring normality in some variables, we solved the homoscedasticity problem.

Now, let's check 'TotalBsmtSF' with 'SalePrice'.



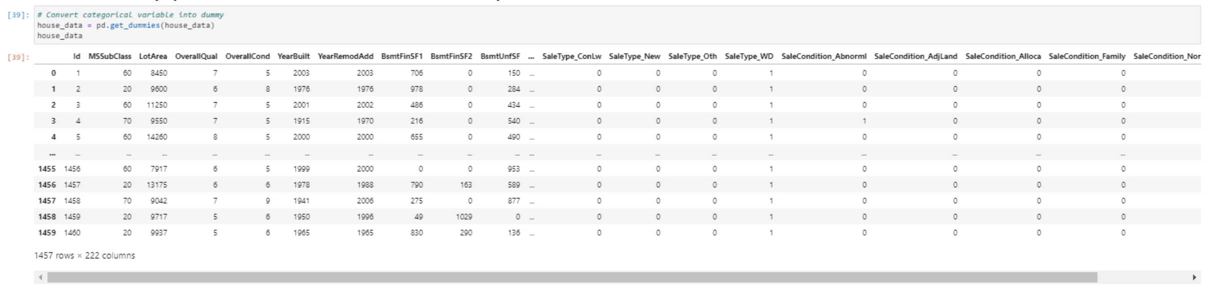
```
# Scatter plot
plt.scatter(house_data[house_data['TotalBsmtSF']>0]['TotalBsmtSF'], house_data[house_data['TotalBsmtSF']>0]['SalePrice']);
```



In general, 'SalePrice' exhibits equal levels of variance across the range of 'TotalBsmtSF'. Cool.

Dummy variables

Using variable coding as a mean of transforming nonmetric data into metric data. It involves the creation of so-called dummy variables, in which 1s and 0s are assigned to subjects, depending on whether they possess a characteristic in question.



```
[39]: # Convert categorical variable into dummy
house_data = pd.get_dummies(house_data)
house_data
```

[39]:

	Id	MSSubClass	LotArea	OverallQual	OverallCond	YearBuilt	YearRemodAdd	BsmtFinSF1	BsmtFinSF2	BsmtUnfSF	...	SaleType_ConLw	SaleType_New	SaleType_Oth	SaleType_WD	SaleCondition_Abnorml	SaleCondition_AdjLand	SaleCondition_Alloca	SaleCondition_Family	SaleCondition_Nor
0	1	60	8450	7	5	2003	2003	706	0	150	...	0	0	0	1	0	0	0	0	
1	2	20	9600	6	8	1976	1976	978	0	284	...	0	0	0	1	0	0	0	0	
2	3	60	11250	7	5	2001	2002	486	0	434	...	0	0	0	1	0	0	0	0	
3	4	70	9550	7	5	1915	1970	216	0	540	...	0	0	0	1	1	0	0	0	
4	5	60	14260	8	5	2000	2000	655	0	490	...	0	0	0	1	0	0	0	0	
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	...	
1455	1456	60	7917	6	5	1999	2000	0	0	953	...	0	0	0	1	0	0	0	0	
1456	1457	20	13175	6	6	1978	1988	790	163	589	...	0	0	0	1	0	0	0	0	
1457	1458	70	9042	7	9	1941	2006	275	0	877	...	0	0	0	1	0	0	0	0	
1458	1459	20	9717	5	6	1950	1996	49	1029	0	...	0	0	0	1	0	0	0	0	
1459	1460	20	9937	5	6	1965	1965	830	290	136	...	0	0	0	1	0	0	0	0	

1457 rows × 222 columns

Conclusion

It is never good idea to skip EDA for machine learning projects, it will only lead to melancholic outcomes, like getting poor accuracy from the machine learning model. With that in mind, I philosophized about the variables, I analysed 'SalePrice' alone and together with some of the variables with the highest correlation, I dealt with missing data and outliers, I tested some of the fundamental statistical assumptions, and transformed categorial variables into dummy variables. That's a lot of work that Python made easier.

But the quest is not over. In future, I will use more data sets and other functions to get the clear idea related to the outcome. I will try to predict the behaviour. Do you think it will enjoy regularized linear regression approaches? Or do you think it prefers ensemble methods? Or maybe something else?