A Course Based Project Report on

House Price Prediction using Ml

Submitted to the

Department of CSE-(CyS, DS) and AI&DS

in partial fulfilment of the requirements for the completion of course

Foundations of Data Science LABORATORY(A19PC2CS47)

BACHELOR OF TECHNOLOGY

IN

Department of CSE-(CyS, DS) and AI&DS

Submitted by

G.Akshaya 21071A7223
P.Himaswi 21071A7258
R.Navyasree 21071A7259

Under the guidance of

Dr.P.VIJAYARAGHAVAN

(Course Instructor)

Assistant Professor, Department of CSE-(CYS,DS) AND AI&DS VNRVJIET

Department of CSE-(CyS, DS) and AI&DS

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI INSTITUTE OF ENGINEERING & TECHNOLOGY

An Autonomous Institute, NAAC Accredited with 'A++' Grade, NBA

Vignana Jyothi Nagar, Pragathi Nagar, Nizampet (S.O), Hyderabad – 500 090, TS, India

DECEMBER ,2023

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI
INSTITUTE OF ENGINEERING AND TECHNOLOGY
An Autonomous Institute, NAAC Accredited with ‘A++’ Grade, NBA Accredited for CE, EEE, ME, ECE, CSE, EIE, IT B. Tech Courses, Approved by AICTE, New Delhi, Affiliated to JNTUH, Recognized as “College with Potential for Excellence” by UGC, ISO 9001:2015 Certified, QS I GUAGE Diamond Rated
Vignana Jyothi Nagar, Pragathi Nagar, Nizampet(SO), Hyderabad-500090, TS, India

Department of CSE-(CyS, DS) and AI&DS

TAMASOMA JYOTIRGAMAYA
Estd.1995

CERTIFICATE

This is to certify that the project report entitled “House Price Prediction using ML” is a bonafide work done under our supervision and is being submitted by Miss. Akshaya(21071A7223), Miss. Himaswi (21071A7258), Miss.Navyasree (21071A7259) in partial fulfilment for the award of the degree of Bachelor of Technology in , of the VNRVJIET, Hyderabad during the academic year 2022-2023.

Dr.P.VIJAYARAGHAVAN	Dr.M.RAJASHEKAR
Assistant Professor	Professor & HOD
Dept of CSE-(CyS, DS) and AI&DS	Dept of CSE-(CyS, DS)and AI&DS

VALLURUPALLI NAGESWARA RAO VIGNANA JYOTHI
INSTITUTE OF ENGINEERING AND TECHNOLOGY
An Autonomous Institute, NAAC Accredited with ‘A++’ Grade,
Vignana Jyothi Nagar, Pragathi Nagar, Nizampet(SO), Hyderabad-500090, TS, India

Department of CSE-(CyS, DS) and AI&DS

TAMASOMA JYOTIRGAMAYA
Estd. 1995

DECLARATION

We declare that the course based project work entitled “House Price Prediction using ML” submitted in the Department of CSE-(CyS, DS) and AI&DS, Vallurupalli Nageswara Rao Vignana Jyothi Institute of Engineering and Technology, Hyderabad, in partial fulfilment of the requirement for the award of the degree of Bachelor of Technology in CSE-(CyS, DS) and AI&DS is a bonafide record of our own work carried out under the supervision of Dr.P.VIJAYARAGHAVAN, Assistant Professor, Department of CSE-(CyS, DS) and AI&DS , VNRVJIET. Also, we declare that the matter embodied in this thesis has not been submitted by us in full or in any part there of for the award of any degree/diploma of any other institution or university previously.

G.Akshaya (21071A7223)	P.Himaswi (21071A7258)	R.Navyasree (21071A7259)

ACKNOWLEDGEMENT

We express our deep sense of gratitude to our beloved President, Sri. D. Suresh Babu, VNR Vignana Jyothi Institute of Engineering & Technology for the valuable guidance and for permitting us to carry out this project.

With immense pleasure, we record our deep sense of gratitude to our beloved Principal, Dr. C.D Naidu, for permitting us to carry out this project.

We express our deep sense of gratitude to our beloved Professor Dr.M.RAJASHEKAR, Professor and Head, Department of CSE-(CyS, DS) and AI&DS , VNR Vignana Jyothi Institute of Engineering & Technology, Hyderabad-500090 for the valuable guidance and suggestions, keen interest and through encouragement extended throughout the period of project work.

We take immense pleasure to express our deep sense of gratitude to our beloved Guide, Mr.P.Vijayaraghavan, Assistant Professor in CSE-(CyS, DS) and AI&DS, VNR Vignana Jyothi Institute of Engineering & Technology, Hyderabad, for his/her valuable suggestions and rare insights, for constant source of encouragement and inspiration throughout my project work.

We express our thanks to all those who contributed for the successful completion of our project work.

G.Akshaya(21071A7223)

P.Himaswi(21071A7258)

R.Navyasree(21071A7259)

TABLE OF CONTENTS

ABSTRACT

People looking to buy a new home tend to be more conservative with their budgets and market strategies. This project aims to analyze various parameters like average income, average area etc. and predict the house price accordingly. This application will help customers to invest in an estate without approaching an agent. To provide a better and fast way of performing operations. To provide proper house price to the customers. To eliminate need of real estate agent to gain information regarding house prices. To provide best price to user without getting cheated. To enable user to search home as per the budget. The aim is to predict the efficient house pricing for real estate customers with respect to their budgets and priorities. By analyzing previous market trends and price ranges, and also upcoming developments future prices will be predicted. House prices increase every year, so there is a need for a system to predict house prices in the future. House price prediction can help the developer determine the selling price of a house and can help the customer to arrange the right time to purchase a house. We use linear regression algorithm in machine learning for predicting the house price trends

CHAPTER-1

INTRODUCTION

Data processing techniques and processes are numerous. We collected data of from various real estate websites. The data would be having attributes such as Sale Price, Overall Quality, Ground Living Area, Garage Cars, Basement area, first floor etc.

People looking to buy a new home tend to be more conservative with their budgets and market strategies. This project aims to analyze various parameters like average income, average area etc. and predict the house price accordingly. This application will help customers to invest in an estate without approaching an agent. To provide a better and fast way of performing operations. To provide proper house price to the customers. To eliminate need of real estate agent to gain information regarding house prices. To provide best price to user without getting cheated. To enable user to search home as per the budget.

The aim is to predict the efficient house pricing for real estate customers with respect to their budgets and priorities. By analyzing previous market trends and price ranges, and also upcoming developments future prices will be predicted. House prices increase every year, so there is a need for a system to predict house prices in the future. House price prediction can help the developer determine the selling price of a house and can help the customer to arrange the right time to purchase a house. We use linear regression algorithm in machine learning for predicting the house price trends

CHAPTER-2

METHOD

Data Collection and Integration:

Data processing techniques and processes are numerous. We collected data of from various real estate websites. The data would be having attributes such as Sale Price, Overall Quality, Ground Living Area, Garage Cars, Basement area, first floor etc.

Data Cleaning and Preprocessing:

Data preprocessing is the process of cleaning our data set. There mightbe missing values or outliers in the dataset. These can be handled by data cleaning. If there are many missing values in a variable we will drop those values or substitute it with the average value.

Linear Regression Model:

Linear Regression is a supervised machine learning model that attempts to model a linear relationship between dependent variables (Y) and independent variables (X). Every evaluated observation with a model, the target (Y)'s actual value is compared to the target (Y)'s predicted value, and the major differences in these values are called residuals. The Linear Regression model aims to minimize the sum of all squared residuals. Here is the mathematical representation of the linear regression:

$$Y = a0 + a1X + \varepsilon$$

CHAPTER-3

TEST CASES/ OUTPUT

```python
import numpy as np
import pandas as pd
import seaborn as sns
from pylab import rcParams
import matplotlib.pyplot as plt
import matplotlib.animation as animation
from matplotlib import rc
import unittest

%matplotlib inline

sns.set(style='whitegrid', palette='muted', font_scale=1.5)

rcParams['figure.figsize'] = 14, 8

RANDOM_SEED = 42

np.random.seed(RANDOM_SEED)

def run_tests():
  unittest.main(argv=[''], verbosity=1, exit=False)
```

Python

```
!wget https://raw.githubusercontent.com/Data-Science-FMI/ml-from-scratch-2019/master/data/house_prices_train.csv
```

Python

```
--2019-04-01 20:29:18--  https://raw.githubusercontent.com/Data-Science-FMI/ml-from-scratch-2019/master/data/house_prices_t
Resolving raw.githubusercontent.com (raw.githubusercontent.com)... 151.101.0.133, 151.101.64.133, 151.101.128.133, ...
Connecting to raw.githubusercontent.com (raw.githubusercontent.com)|151.101.0.133|:443... connected.
HTTP request sent, awaiting response... 200 OK
Length: 460676 (450K) [text/plain]
Saving to: ‘house_prices_train.csv.1’


house_prices_train.   0%[                    ]       0  --.-KB/s
house_prices_train. 100%[===================>] 449.88K  --.-KB/s    in 0.03s

2019-04-01 20:29:19 (16.4 MB/s) - ‘house_prices_train.csv.1’ saved [460676/460676]
```

Data exploration

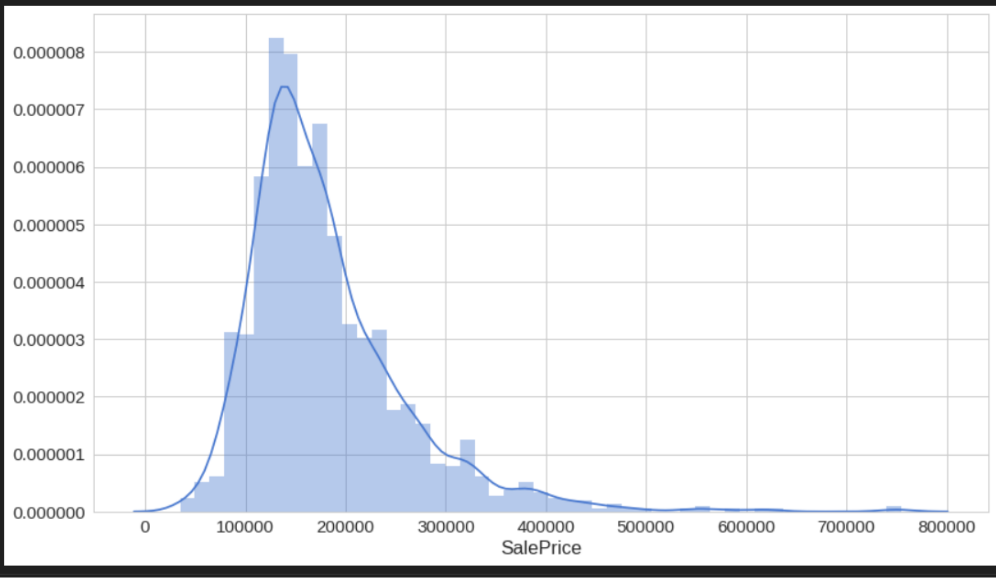
```python
df_train['SalePrice'].describe()
```

Python

```
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
```

```
sns.distplot(df_train['SalePrice']);
```

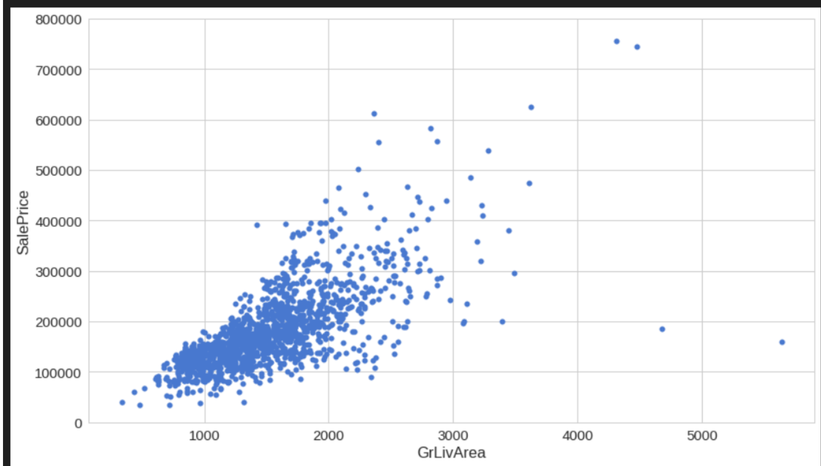
```
/usr/local/lib/python3.6/dist-packages/matplotlib/axes/_axes.py:6521: MatplotlibDeprecationWarning:
The 'normed' kwarg was deprecated in Matplotlib 2.1 and will be removed in 3.1. Use 'density' instead.
  alternative="'density'", removal="3.1")
```




```
var = 'GrLivArea'
data = pd.concat([df_train['SalePrice'], df_train[var]], axis=1)
data.plot.scatter(x=var, y='SalePrice', ylim=(0,800000), s=32);
```

Python

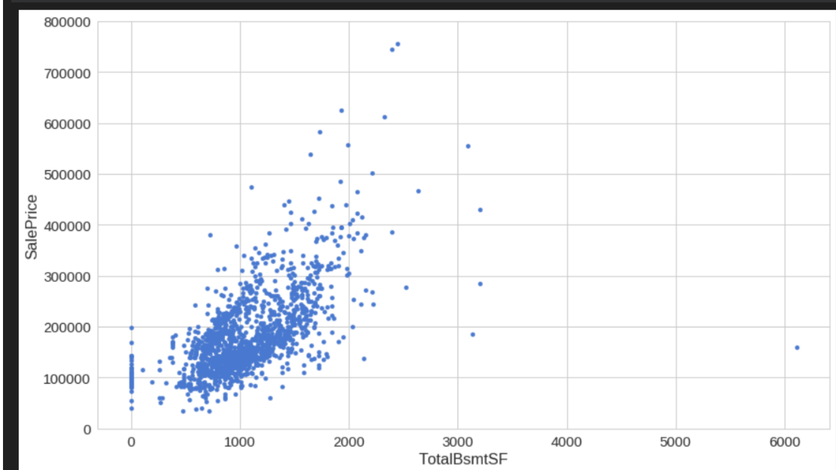
```
'c' argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-mapping will have precedence in case its length matches with 'x' & 'y'.  Please use a 2-[
```




```
var = 'TotalBsmtSF'
data = pd.concat([df_train['SalePrice'], df_train[var]], axis=1)
data.plot.scatter(x=var, y='SalePrice', ylim=(0,800000));
```

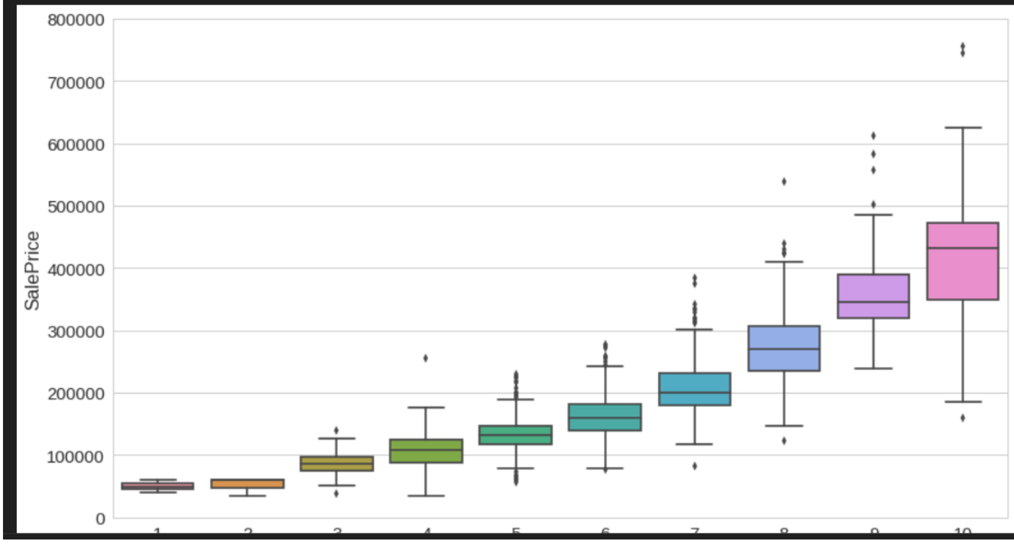
Python

```
'c' argument looks like a single numeric RGB or RGBA sequence, which should be avoided as value-mapping will have precedence in case its length matches with 'x' & 'y'.  Please use a 2-[
```

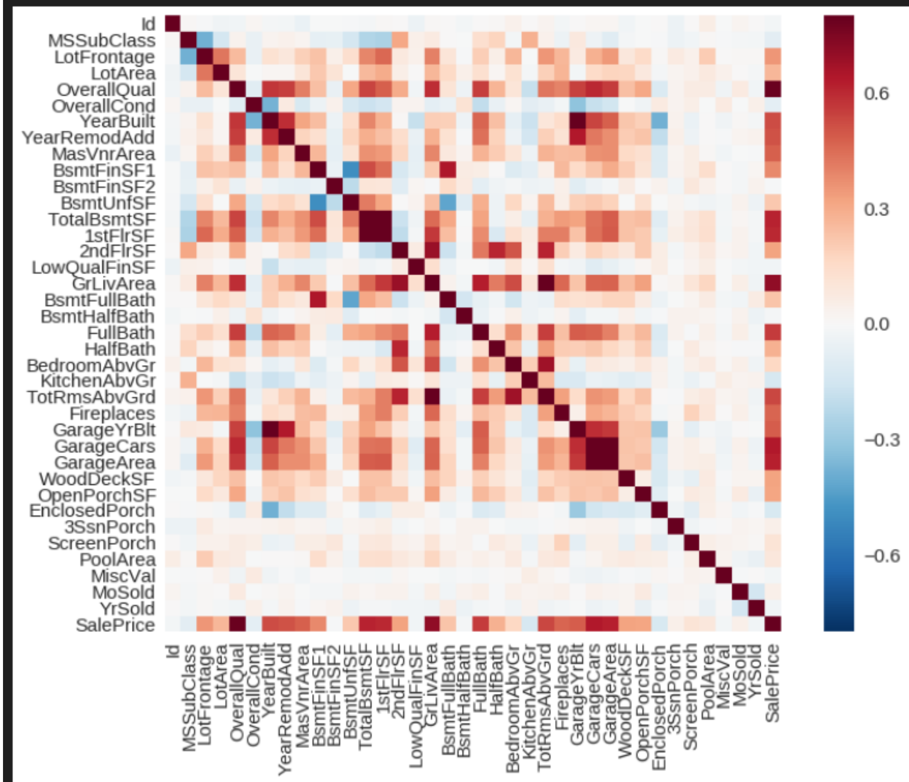


```python
var = 'OverallQual'
data = pd.concat([df_train['SalePrice'], df_train[var]], axis=1)
f, ax = plt.subplots(figsize=(14, 8))
fig = sns.boxplot(x=var, y="SalePrice", data=data)
fig.axis(ymin=0, ymax=800000);
```

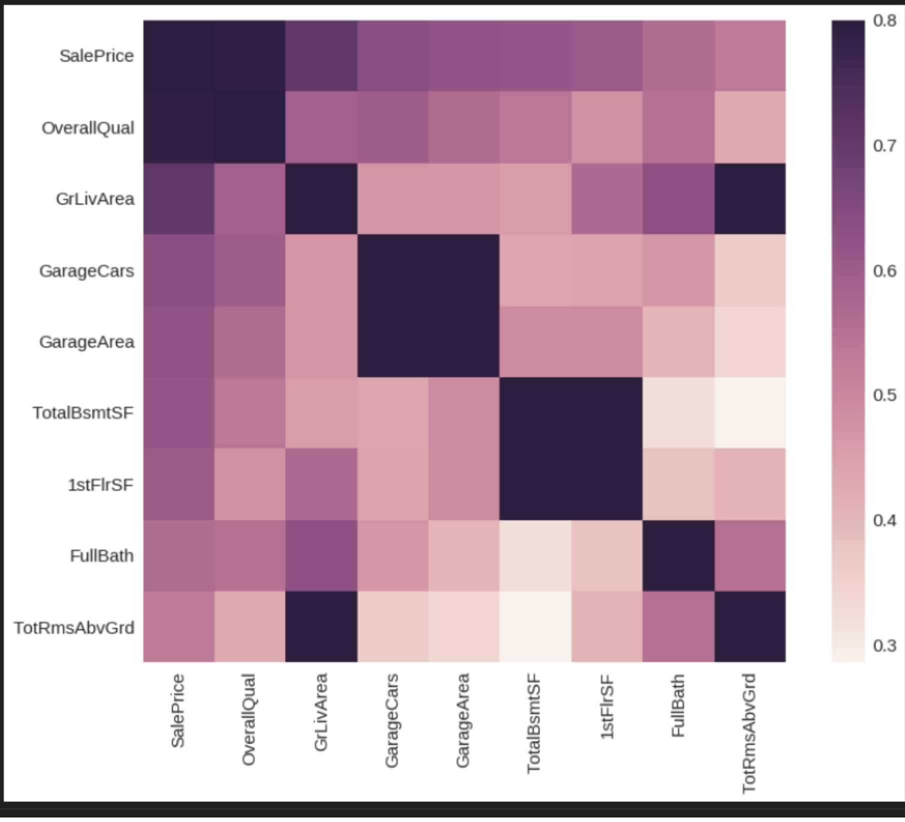
```
/usr/local/lib/python3.6/dist-packages/seaborn/categorical.py:454: FutureWarning: remove_na is deprecated and is a private function. Do not use.
  box_data = remove_na(group_data)
```



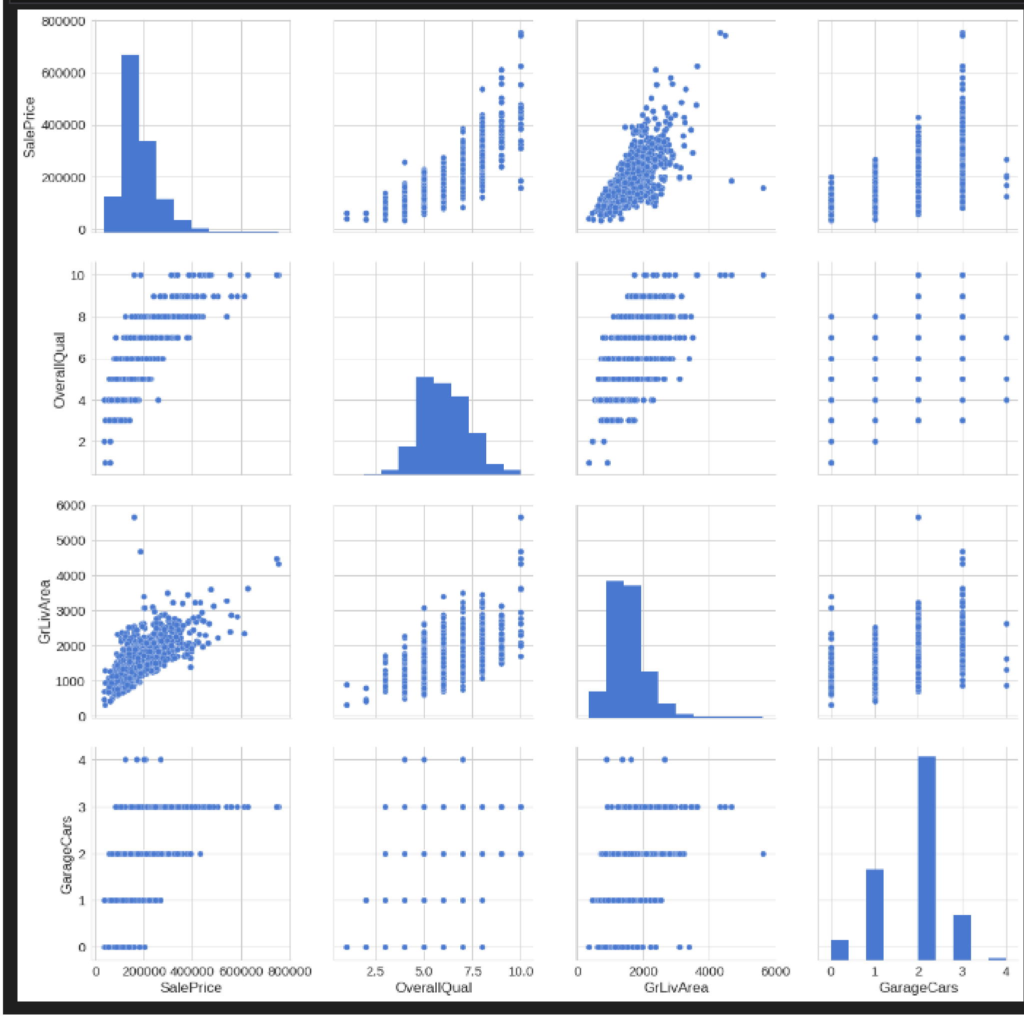

```python
corrmat = df_train.corr()
f, ax = plt.subplots(figsize=(12, 9))
sns.heatmap(corrmat, vmax=.8, square=True);
```



```
k = 9 #number of variables for heatmap
cols = corrmat.nlargest(k, 'SalePrice')['SalePrice'].index
f, ax = plt.subplots(figsize=(14, 10))
sns.heatmap(df_train[cols].corr(), vmax=.8, square=True);
```




```
cols = ['SalePrice', 'OverallQual', 'GrLivArea', 'GarageCars']
sns.pairplot(df_train[cols], size = 4);
```



Do we have missing data?

```
total = df_train.isnull().sum().sort_values(ascending=False)
percent = (df_train.isnull().sum()/df_train.isnull().count()).sort_values(ascending=False)
missing_data = pd.concat([total, percent], axis=1, keys=['Total', 'Percent'])
missing_data.head(20)
```

	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

Preparing the data

Feature scaling

We will do a little preprocessing to our data using the following formula (standardization):

$$x' = \frac{x - \mu}{\sigma}$$

where μ is the population mean and σ is the standard deviation.

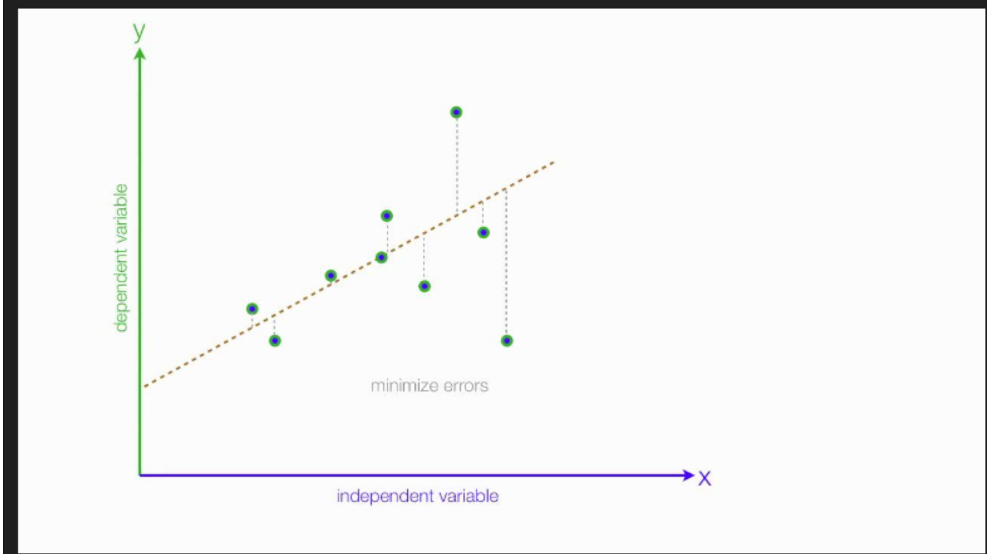
Source: Andrew Ng

```
x = df_train['GrLivArea']
y = df_train['SalePrice']

x = (x - x.mean()) / x.std()
x = np.c_[np.ones(x.shape[0]), x]
```

```
x.shape
```

```
(1460, 2)
```

Linear Regression

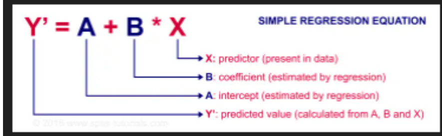


Source: MyBookSucks

Simple Linear Regression

Simple linear regression uses a traditional slope-intercept form, where a and b are the coefficients that we try to "learn" and produce the most accurate predictions. X represents our input data and Y is our prediction.

$$Y = bX + a$$



Source: SPSS tutorials

Multivariable Regression

A more complex, multi-variable linear equation might look like this, where w represents the coefficients, or weights, our model will try to learn.

$$Y(x_1, x_2, x_3) = w_1x_1 + w_2x_2 + w_3x_3 + w_0$$

The variables x_1, x_2, x_3 represent the attributes, or distinct pieces of information, we have about each observation.

Loss function

Given our Simple Linear Regression equation:

$$Y = bX + a$$

We can use the following cost function to find the coefficients:

Mean Squared Error (MSE) Cost Function

The MSE is defined as:

$$MSE = J(W) = \frac{1}{m}\sum_{i=1}^{m}(y^{(i)} - h_w(x^{(i)}))^2$$

where

$$h_w(x) = g(w^Tx)$$

The MSE measures how much the average model predictions vary from the correct values. The number is higher when the model is performing "bad" on the training set.

The first derivative of MSE is given by:

$$MSE' = J'(W) = \frac{2}{m}\sum_{i=1}^{m}(h_w(x^{(i)}) - y^{(i)})$$

One Half Mean Squared Error (OHMSE)

We will apply a small modification to the MSE - multiply by $\frac{1}{2}$ so when we take the derivative, the 2s cancel out:

$$OHMSE = J(W) = \frac{1}{2m}\sum_{i=1}^{m}(y^{(i)} - h_w(x^{(i)}))^2$$

The first derivative of OHMSE is given by:

$$OHMSE' = J'(W) = \frac{1}{m}\sum_{i=1}^{m}(h_w(x^{(i)}) - y^{(i)})$$

```python
def loss(h, y):
  sq_error = (h - y)**2
  n = len(y)
  return 1.0 / (2*n) * sq_error.sum()
```

```python
class TestLoss(unittest.TestCase):

  def test_zero_h_zero_y(self):
    self.assertAlmostEqual(loss(h=np.array([0]), y=np.array([0])), 0)

  def test_one_h_zero_y(self):
    self.assertAlmostEqual(loss(h=np.array([1]), y=np.array([0])), 0.5)

  def test_two_h_zero_y(self):
    self.assertAlmostEqual(loss(h=np.array([2]), y=np.array([0])), 2)

  def test_zero_h_one_y(self):
    self.assertAlmostEqual(loss(h=np.array([0]), y=np.array([1])), 0.5)

  def test_zero_h_two_y(self):
    self.assertAlmostEqual(loss(h=np.array([0]), y=np.array([2])), 2)
```

```python
run_tests()
```

```
.....
----------------------------------------------------------------------
Ran 5 tests in 0.007s

OK
```

```python
class LinearRegression:

  def predict(self, X):
    return np.dot(X, self._W)

  def _gradient_descent_step(self, X, targets, lr):

    predictions = self.predict(X)

    error = predictions - targets
    gradient = np.dot(X.T,  error) / len(X)

    self._W -= lr * gradient

  def fit(self, X, y, n_iter=100000, lr=0.01):

    self._W = np.zeros(X.shape[1])

    self._cost_history = []
    self._w_history = [self._W]
    for i in range(n_iter):

        prediction = self.predict(X)
        cost = loss(prediction, y)

        self._cost_history.append(cost)

        self._gradient_descent_step(x, y, lr)

        self._w_history.append(self._W.copy())
    return self
```

```python
class TestLinearRegression(unittest.TestCase):

    def test_find_coefficients(self):
      clf = LinearRegression()
      clf.fit(x, y, n_iter=2000, lr=0.01)
      np.testing.assert_array_almost_equal(clf._W, np.array([180921.19555322,  56294.90199925]))
```

```python
run_tests()
```

```
......
----------------------------------------------------------------------
Ran 6 tests in 1.094s

OK
```

11

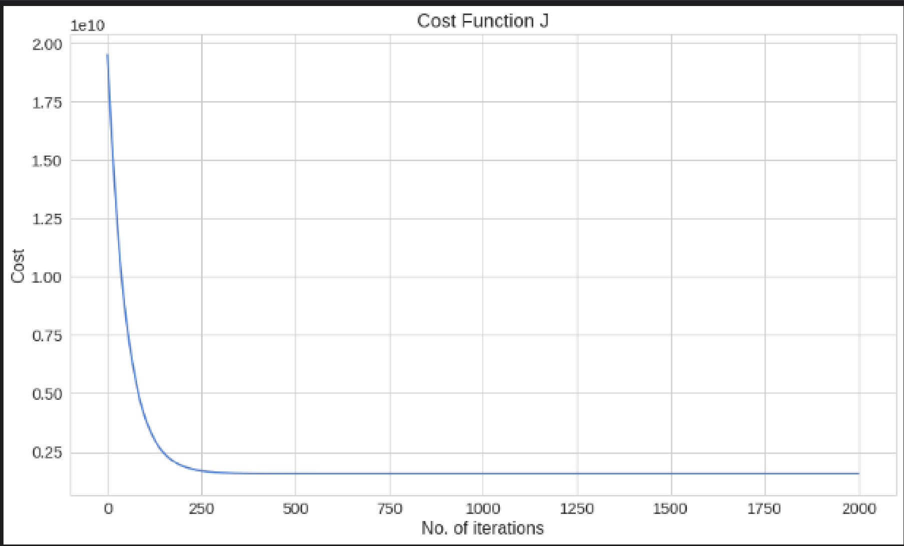
```
clf = LinearRegression()
clf.fit(x, y, n_iter=2000, lr=0.01)
```

```
<__main__.LinearRegression at 0x7fe8e6b74240>
```

```
clf._W
```

```
array([180921.19555322,  56294.90199925])
```

```
plt.title('Cost Function J')
plt.xlabel('No. of iterations')
plt.ylabel('Cost')
plt.plot(clf._cost_history)
plt.show()
```



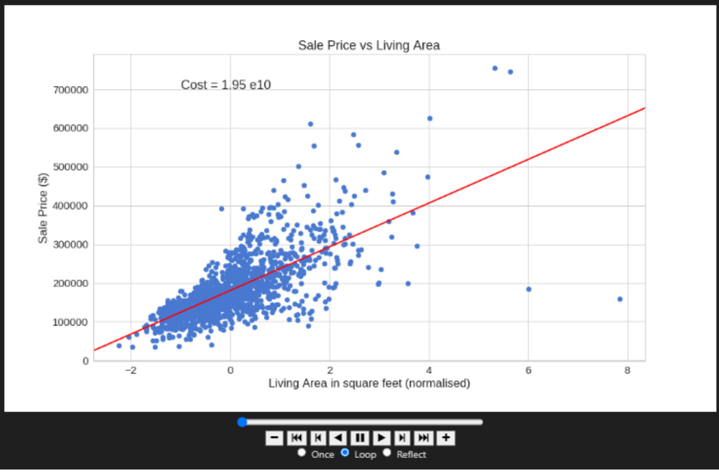
+ Code + Markdown

```
clf._cost_history[-1]
```

```
1569921604.8332634
```

```
#Animation

#Set the plot up,
fig = plt.figure()
ax = plt.axes()
plt.title('Sale Price vs Living Area')
plt.xlabel('Living Area in square feet (normalised)')
plt.ylabel('Sale Price ($)')
plt.scatter(x[:,1], y)
line, = ax.plot([], [], lw=2, color='red')
annotation = ax.text(-1, 700000, '')
annotation.set_animated(True)
plt.close()

#Generate the animation data,
def init():
    line.set_data([], [])
    annotation.set_text('')
    return line, annotation

# animation function.  This is called sequentially
def animate(i):
    x = np.linspace(-5, 20, 1000)
    y = clf._w_history[i][1]*x + clf._w_history[i][0]
    line.set_data(x, y)
    annotation.set_text('Cost = %.2f e10' % (clf._cost_history[i]/10000000000))
    return line, annotation

anim = animation.FuncAnimation(fig, animate, init_func=init,
                               frames=300, interval=10, blit=True)

rc('animation', html='jshtml')

anim
```

```
Animation size has reached 21048368 bytes, exceeding the limit of 20971520.0. If you're sure you want a larger animation embedded, set the animation.embed_limit rc parameter to a larger value (in MB). This and further frames will be dropped.
```



Multivariable Linear Regression

Let's use more of the available data to build a Multivariable Linear Regression model and see whether or not that will improve our OHMSE error:

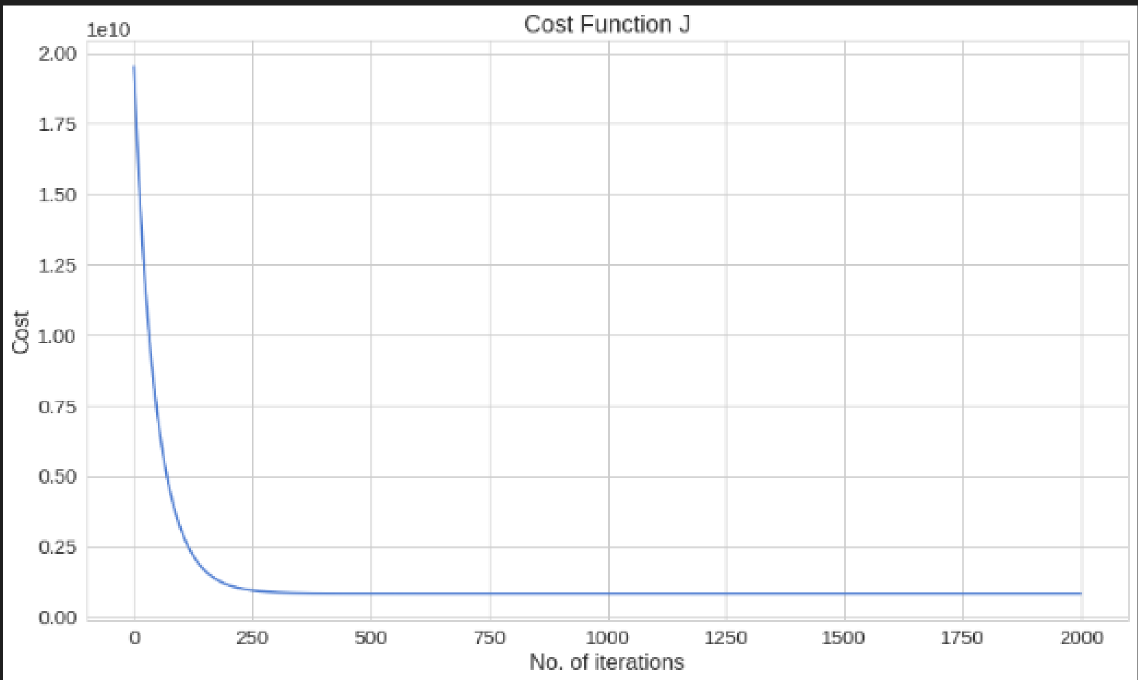
```
x = df_train[['OverallQual', 'GrLivArea', 'GarageCars']]

x = (x - x.mean()) / x.std()
x = np.c_[np.ones(x.shape[0]), x]

clf = LinearRegression()
clf.fit(x, y, n_iter=2000, lr=0.01)
```

```
<__main__.LinearRegression at 0x7fe8e8ada860>
```

```
clf._W
```

```
array([180921.19555322,  37478.604254  ,  26631.93830568,  15921.22581327])
```

```python
plt.title('Cost Function J')
plt.xlabel('No. of iterations')
plt.ylabel('Cost')
plt.plot(clf._cost_history)
plt.show()
```




```python
clf._cost_history[-1]
```

```
822817042.8437098
```

14

1

1

1

17

CHAPTER-4

RESULTS

The r square error or mean square error for good accuracy of the model in predicting the data is indicatednumerically also. A model is good if these error values are less then 5. Then during testing process we predict the future house prices using present and past data parameters ofhouses in an location. Then we plot this graphically as a house price over time graph. For training the model , the error needs to be minimum for greater accuracy of model. The error between the actual and predicted price is plotted graphically using scatter plot. Here we can see that error is minimum sincethe data points of actual and predicted value are close to each other

CHAPTER 5

SUMMARY, CONCLUSION, RECOMMENDATION

SUMMARY:

The project conducted a comprehensive analysis of house price data. Through meticulous data processing and visualization, two primary observations emerged. Firstly, the user exhibited a frequent purchase pattern between sales price and basement area, suggesting a significant connection between these locations, potentially indicating residential or occupational ties. Secondly, the data indicated a purchase for first floor as we analyzed the data for high standard of living.

CONCLUSION:

The machine learning model to predict the house price based ongiven dataset is executed successfully using Linear Regression. This model further helps people understand whether this place is more suited for them based on heatmap correlation. It also helps people looking to sell a house at best time for greater profit. Any house price in any location can be predicted with minimum error by giving appropriate dataset.

RECOMMENDATIONS:

Advanced Feature Engineering:

Explore advanced feature engineering techniques such as PCA (Principal Component Analysis) or interaction features.

Data Augmentation:

Generate synthetic samples using techniques like SMOTE (Synthetic Minority Over-sampling Technique) to balance the dataset.

Time Series Considerations:

If your data includes a time dimension, consider time-series-specific models or features.

Automated Machine Learning (AutoML):

Explore AutoML tools to automate the process of model selection and hyperparameter tuning.

Monitoring and Updating:

Implement monitoring for model performance in production and update the model as needed.

REFERENCES

1. https://raw.githubusercontent.com/Data-Science-FMI/ml-from-scratch-2019/master/data/house_prices_train.csv
2. https://www.geeksforgeeks.org/house-price-prediction-using-machine-learning-in-python/