Project Report on best model selection in multiple linear regression



**Submitted To: Submitted By:**

**Dr. Kaushal Kumar Name: Sukhpal**

**RollNo: 1893534**

**Subject : Statistics II**

**M.Sc (OR) Sem III**

**Abstract**

Data collected from kaggle.com (that allows students to download data) to develop a multivariate regression model to determine house’s price based on a variety of characteristics such as no. of bedrooms, no. of bathrooms, square ft living and no. of floors. We learn to understand data, explore techniques for variable selection and develop specially constructed variables.

**1.Introduction:**

**Regression analysis** helps one understand how the typical value of the dependent variable (or 'criterion variable') changes when any one of the independent variables is varied, while the other independent variables are held fixed. Thus, it provides a good basis for estimating the cost and duration. If y is a dependent variable and x1, x2, …, xk are independent variables then the multiple regression model provides a prediction of y from the xi of the form:

Y = β0 + β1x1 + β2x2 + … + βkxk + ε

Where β0 + β1x1 + β2x2 + … + βkxk is the deterministic portion of the model and ε is the random error. We further assume that for any given values of the xi the random error ε is normally and independently distributed.

The multiple regression model is based on the following assumptions:

**1. Linearity:** The dependent variable y can be expressed as a linear combination of the independent variables x1, …, xk.

**2. Independence:** Observations are selected independently and randomly from the population.

**3. Normality:** Observations are normally distributed.

**4. Homogeneity of variances:** Observations have the same variance.

**Dependent variable:** A variable (often denoted by *y*) whose value depends on that of another

**Independent variable:** An independent variable (often denoted by *X*) is the variable that is changed or controlled in a scientific experiment to test the effects on the dependent variable.

**2.Data Description (that I am using in project):**

The dataset consisted of 21 variables and 21613 observations. I have to predict the price of a house that is in King Country ( is a region of the western [North Island](https://en.wikipedia.org/wiki/North_Island) of [New Zealand](https://en.wikipedia.org/wiki/New_Zealand) ).

Source: <https://www.kaggle.com/shivachandel/kc-house-data>

**Language used:-**

Python language: Python is an interpreted , high-level , general-purpose programming language. Created by Guido van Rossum and first released in 1991, Python’s design philosophy emphasizes code readability with its notable use of significant whitespace. Its language constructs and object-oriented approach aim to help programmers write clear , logical code for small and large-scale projects.

**Softwares used:**

1. Spyder (Python 3.7)
2. Jupyter Notebook
3. MS Excel (To understand data)

**Library used:**

**pandas :** pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming.

**numpy:**NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

**matplotlib.pyplot:** matplotlib.pyplot is a collection of command style functions that make matplotlib work like MATLAB. Each pyplot function makes some change to a figure: e.g., creates a figure, creates a plotting area in a figure, plots some lines in a plotting area, decorates the plot with labels, etc.

**seaborn:** Seaborn is a Python visualization library based on matplotlib. It provides a high-level interface for drawing attractive statistical graphics.

**statsmodels:** Statsmodels is a Python package that allows users to explore data, estimate statistical models, and perform statistical tests. An extensive list of descriptive statistics, statistical tests, plotting functions, and result statistics are available for different types of data and each estimator.

**scipy.stats:** Statistical functions ( scipy. stats ) This module contains a large number of probability distributions as well as a growing library of statistical functions. Each univariate distribution is an instance of a subclass of rv\_continuous ( rv\_discrete for discrete distributions).

**Sklearn.feature\_selection:** It is a supervised learning estimator with a fit method that provides information about feature importance either through a coef\_ attribute or through a feature\_importances\_ attribute. n\_features\_to\_select : int or None (default=None) The number of features to select.

**3.Description of dataset columns:**

price : Price of a house i.e dependent variable for dataset.

bedrooms : No. of bedrooms that are in a house.

bathrooms: No. of bathrooms.

sqft\_living: Size of living space.

sqft\_lot: Plot size.

floors : No. of floors.

waterfront: A part of a town that borders the sea or a lake or river.

view: view around the house.

condition: condition of the house.

grade: grade of house.

sqft\_above: square foot above.

sqft\_basement: square foot basement size.

yr\_built: when house built.

yr\_renovated: when house renovated.

lat : Latitude of house.

long: Longitude of house.

Method used to select variables:

**4.STEPWISE SELECTION:**

It is a method under sequential selection for selection of best model in case of number of independent variables and hence no. of models.

**STEPWISE SELECTION ALGORITHM:**

**Step-1** Start with the model having no independent variable.

**Step-2** Compute partial F-values for each of the independent variable.

**Step-3** Identify the variable which corresponds to highest partial F-value and add it to the model.

**Step-4** Compute partial F-values for remaining variables in presence of previously selected variables and identify the variable which corresponds to highest partial F-value and add it to the model if corresponding partial F-value > some threshold value.

**Step-5** All the variables which have been included in the model are tested for their significance in presence of other variables in model. All the variables which are no longer significant in presence of other variables in model are removed.

**Step-6** This algorithm terminates if at any stage we cannot add any additional variable into the model and all the variables which are there in the model are significant.

**Partial F-value = (( ESS(full model)- ESS(full model except xj ))/(RMS(full model))**

**f\_regression:** f\_regression in python is used for calculating F-value for each model to check significance of model, such as price ~ sqft\_living, price ~ lot, price ~ floors and etc. for all explanatory variables.

**anova\_lm:** anova\_lm in pythonis used for calculating partial f-value to check significance of explanatory variables in the model.

**5.Python Code and results:**

**# Importing the libraries**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import statsmodels.api as sm

import scipy.stats as stats

from sklearn.feature\_selection import f\_regression

from statsmodels.stats.anova import anova\_lm

import seaborn as sns

sns.set()

pd.set\_option('display.notebook\_repr\_html', True)

pd.set\_option('display.precision', 2)

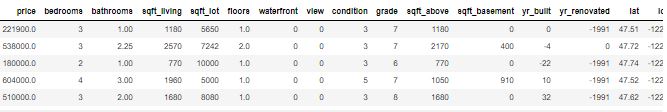
%matplotlib notebook

plt.rcParams['figure.figsize'] = 10, 10

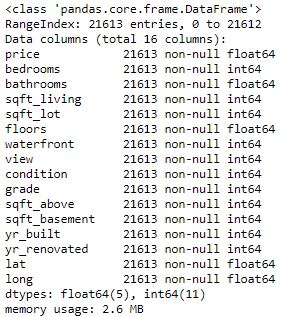
**#Load dataset into dataframe**

df = pd.read\_csv("kc\_house\_data1.csv")

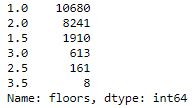
display(df.head())



df.info()



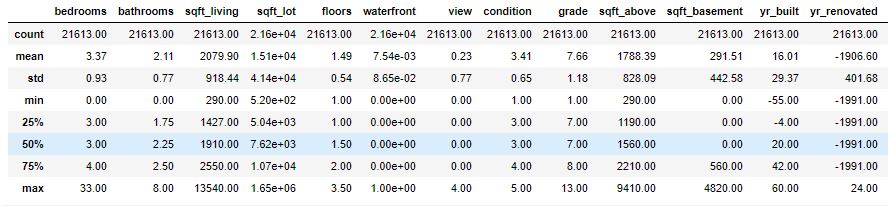
df['floors'].value\_counts()



y = df.iloc[:, 0]

X = df.iloc[:, 1:]

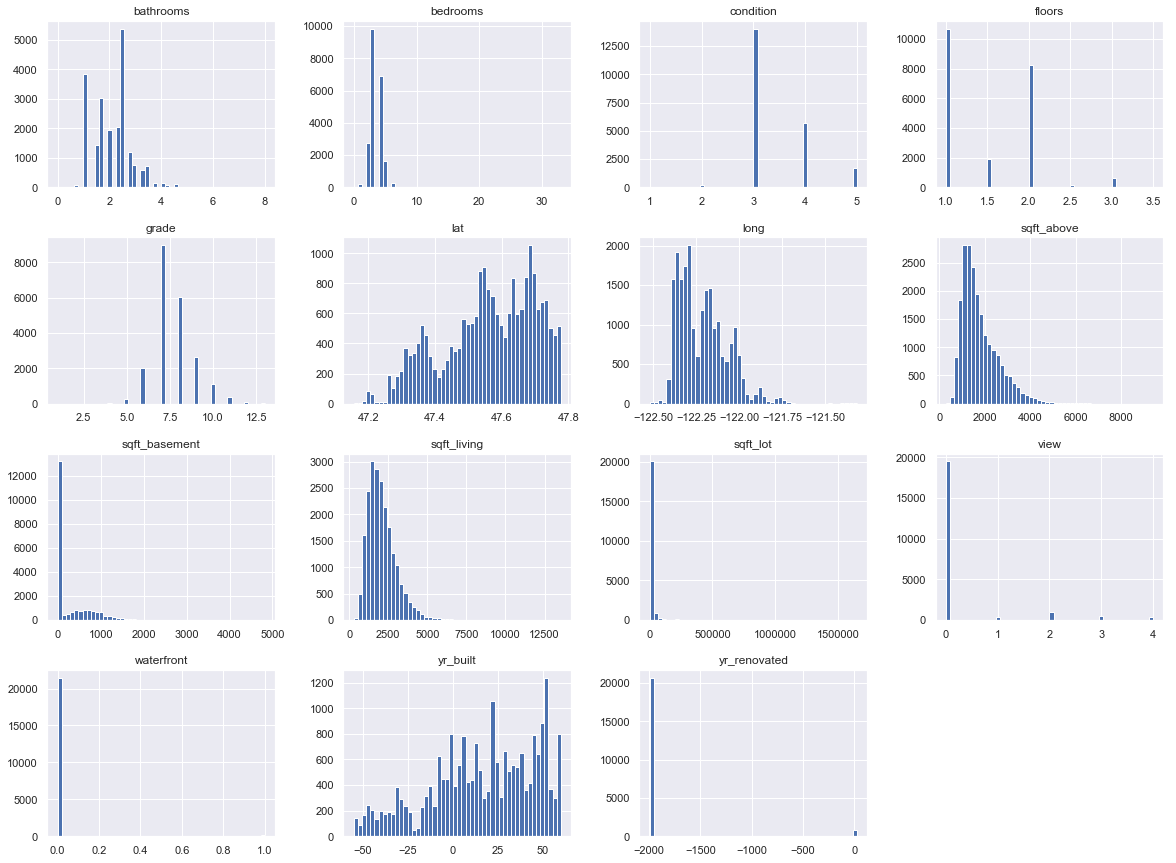
X.describe()



**#Plotting histogram**

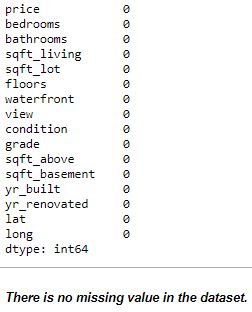
X.hist(bins=50, figsize=(20, 15))

plt.show()



**#For missing values:**

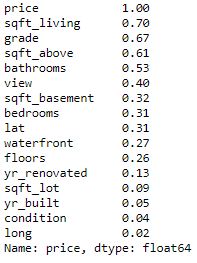
print(df.isnull().sum())



**#Looking for correlations**

corr\_matrix = df.corr()

corr\_matrix['price'].sort\_values(ascending=False)



**#Stepwise selection**

**# Initial Iteration**

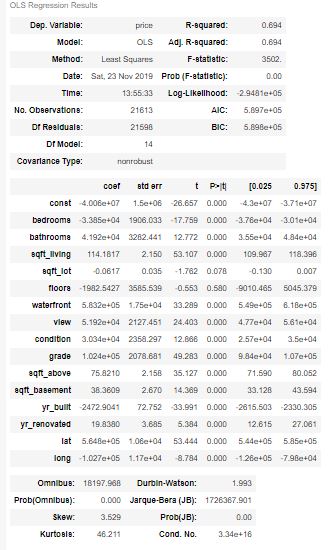
y = df.iloc[:, 0]

X = df.iloc[:, 1:]

X = sm.add\_constant(X)

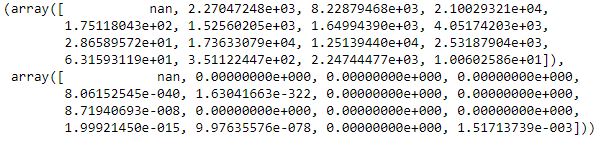
mreg = sm.OLS(y, X).fit()

display(mreg.summary())



(F, pval) = f\_regression(X, y)

F,pval



index = list()

for i in range(1,len(df.columns)+1):

if max(F[1:len(df.columns)+1]) == F[i] and max(F[1:len(df.columns)+1]) > stats.f.ppf(q = 1-0.05, dfn = 2, dfd = len(df) - (2+1)):

index.append(i)

break

modeled\_X = df.iloc[:,index]

modeled\_X.head()



for j in range(len(df.columns)-2):

**#f test for adding column or model significance**

fvalue1 = []

for i in range(len(df.columns)-1):

if i+1 not in index:

index1 = index + [i+1]

X = df.iloc[:,index1]

X = sm.add\_constant(X)

mreg = sm.OLS(y, X).fit()

#display(mreg.summary())

fvalue1.append(mreg.fvalue)

else:

fvalue1.append(0)

for i in range(len(df.columns)-1):

if max(fvalue1) == fvalue1[i] and max(fvalue1) > stats.f.ppf(q = 1-0.05, dfn = len(index)+1, dfd = len(df) - (len(index)+2)):

index.append(i+1)

break

modeled\_X = df.iloc[:,index]

**#partial f test for removing insignificanct columns**

cols = list(modeled\_X.columns)

X = modeled\_X

X = sm.add\_constant(X)

mreg = sm.OLS(y, X).fit()

for i in range(len(index)):

X = modeled\_X.drop([cols[i]],axis = 1)

X = sm.add\_constant(X)

mreg1 = sm.OLS(y, X).fit()

res = anova\_lm(mreg1,mreg)

if res.F[1] == np.nan or res.F[1]>=stats.f.ppf(q = 0.95, dfn = len(df) - res.df\_resid[1], dfd = res.df\_resid[1]):

continue

elif res.F[1]<stats.f.ppf(q = 0.95, dfn = len(df) - res.df\_resid[1], dfd = res.df\_resid[1]):

modeled\_X = modeled\_X.drop([cols[i]],axis = 1)

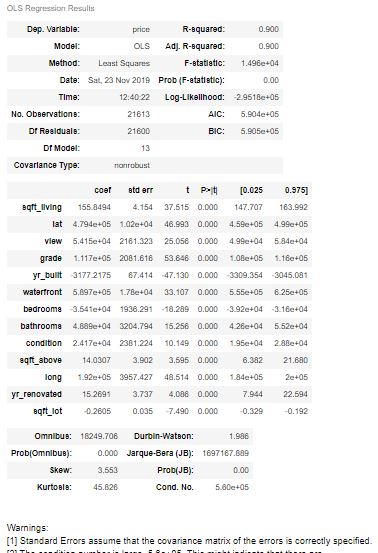
**# Final model**

modeled\_X.head()



mreg = sm.OLS(y, modeled\_X).fit()

mreg.summary()



**# Splitting the dataset into Training set and Test set**

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(modeled\_X, y, test\_size = 0.2, random\_state = 42)

**#Fitting multiple linear regression to the Training set**

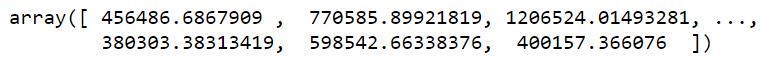
from sklearn.linear\_model import LinearRegression

regressor = LinearRegression()

regressor.fit(X\_train, y\_train)

**# Predicting y\_test results using model y\_test ~ X\_test**

regressor.predict(X\_test)



**#Plotting price ~ sqft\_living**

**#Because sqft\_living is highly correlated with price**

y = df.iloc[:,0].values.reshape(-1,1)

X = df.iloc[:,3].values.reshape(-1,1)

reg = LinearRegression()

reg.fit(X,y)

**#Visualising the results**

plt.scatter(X, y, color = 'red')

plt.plot(X, reg.predict(X), color = 'blue')

plt.title('sqft\_living vs price (Training set)')

plt.xlabel('sqft\_living')

plt.ylabel('price')

plt.show()



**Conclusion:**

Finally, we got 13 independent variables (sqft\_living, lat, view, grade, yr\_built, waterfront, bedrooms, bathrooms, condition, sqft\_above, long, yr\_renovated, sqft\_lot) and 21613 observations in the dataset.

In final results we got R2 = 0.839 and Adjusted R2 = 0.839 that are very good performance by the model.

**Acknowledgements:**

Partial support for this work was provided by the course (M.Sc Operational research). I would like to thank Dr. Kaushal Kumar for giving to do this project and other friends for their helpful suggestions.

**References:**

Basic Econometrics 5th Edition by Damodar N. Gujarati and Dawn C. Porter:

<https://himayatullah.weebly.com/uploads/5/3/4/0/53400977/gujarati_book.pdf>

Machine Learning free course:

<https://www.superdatascience.com/machine-learning>

All python softwares:

<https://www.anaconda.com/distribution/>