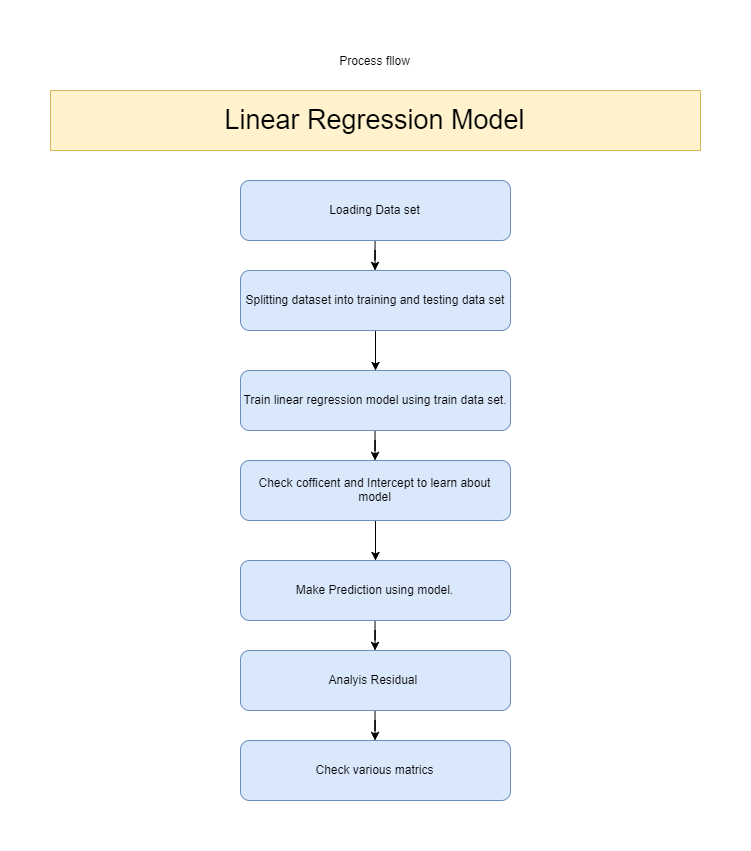
Linear Regression Model

Regression is a method of modeling a target value based on independent predictors. This method is mostly used for forecasting and finding out cause and effect relationship between variables. Regression techniques mostly differ based on the number of independent variables and the type of relationship between the independent and dependent variables.

**Process Flow**



**Codes:-**

* **To import following libraries.**

import pandas as pd

import numpy as np

import matplotlib.pyplot as plt

import seaborn as sns

%matplotlib inline

* To read CSV file using pandas and load dataset into df

df = pd.read\_csv('USA\_Housing.csv')

* To view, info, describe and to see column of data set

df.head()

df.info()

df.describe()

df.columns

* To import machine learning library with various models.

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

* To Create variable X and y for training and testing sets. [X = Column names] [y = Column against which we have to plot]

X = df[['Avg. Area Income','Avg. Area House Age','Avg. Area Number of Rooms','Avg. Area Number of Bedrooms', 'Area Population']]

y = df['Price']

* To Create Test and Training data sets.

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.4, random\_state=101)

* Import Linear regression library from sklearn

from sklearn.linear\_model import LinearRegression

* Create linear regression object.

lm = LinearRegression()

* To Train model.

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

* To view Intercept and Coefficient

print(lm.intercept\_)

lm.coef\_

* To put coefficient in data frame.

cdf = pd.DataFrame(lm.coef\_,X.columns,columns=['Coeff'])

**Predictions**

* To create prediction

predictions = lm.predict(X\_test)

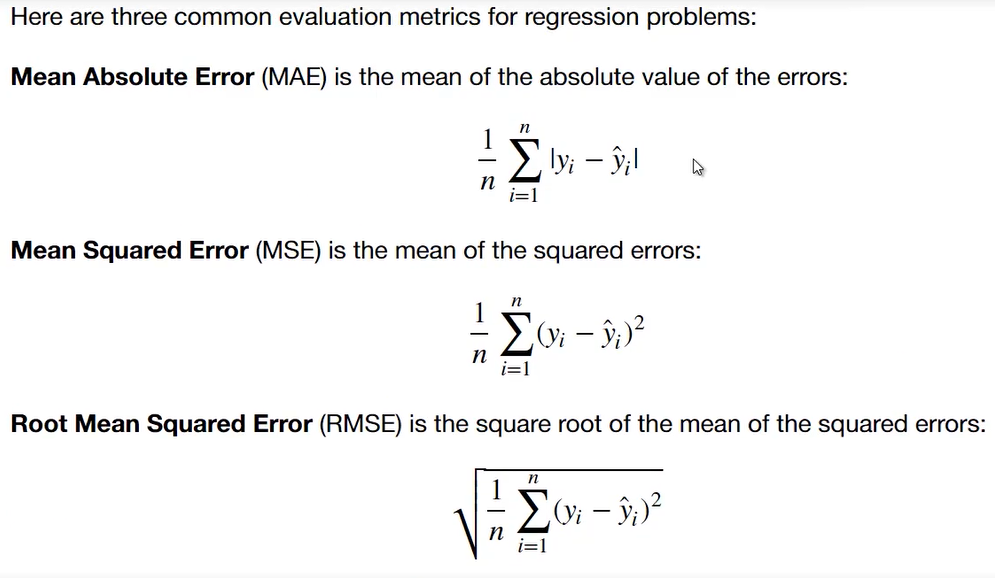
* To Create Scatter plot

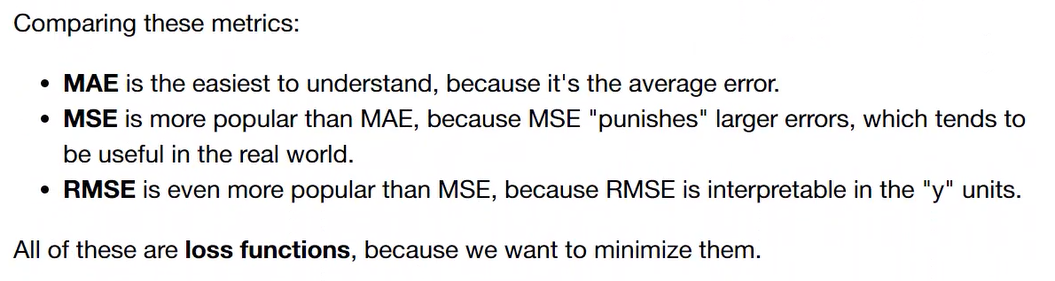
plt.scatter(y\_test,predictions)

* To plot histogram

sns.distplot((y\_test-predictions))

**To Test Predictions**





**Code to use metrics**

from sklearn import metrics

metrics.mean\_absolute\_error(y\_test,predictions)

metrics.mean\_squared\_error(y\_test,predictions)

np.sqrt(metrics.mean\_squared\_error(y\_test,predictions))

Final percentage:

metrics.explained\_variance\_score(y\_test,predictions)