ASAD ASHRAF KARL

PYTHON LIBARAIES

To avoid warnings:

In [ ]:
```
from warnings import filterwarnings
filterwarnings('ignore')
```

Python Fundamental libraries:

In [1]:
```
import numpy as np

import pandas as pd

import math

import random
```

Visualizations:

In [10]:
```
import matplotlib.pyplot as plt

import seaborn as sns
```

STATISTICS:

In [3]:
```
import scipy.stats as st

from itertools import combinations
# list(combinations(data,x)) , x=1,2,3,.... number of combinations

import statsmodels.api as sm
from statsmodels.formula.api import ols
# model = ols('numerical_variale'~'Categorical_variable', data).fit()
# sm.stats.anova_lm(model)

# If we reject null hypothesis, and wish to all possibilites in difference o
from statsmodels.stats.multicomp import pairwise_tukeyhsd
# pairwise_tukeyhsd('numerical_variable', 'categorecal_variable', alpha).sum

# Varias functions from scipy:
from scipy import stats
from scipy.stats import shapiro
```

Split

In [ ]:
```
from sklearn.model_selection import train_test_split

x=df.drop('target_column', axis = 1)#drop the target variable
y=df['target_column']
xtrain, xtest, ytrain, ytest = train_test_split(x, y, test_size = 0.2, rando
```

FEATURE ENGINEERING:

STANDARDIZATION:

z_score

In [12]:
```
from scipy.stats import zscore
# df['New_column'] = zscore(df['On_which_column'])
```

NORMALIZATION:

MinMax scaling

In [5]:
```
from sklearn.preprocessing import MinMaxScaler
# mm = MinMaxScaler(feature_range=(0, 1)) To fix theh range according to use
# df['New_column'] = mm.fit_transform(df[['On_which_column']])
```

OR

In [ ]:
```
df['New_Column'], lmda = st.boxcox(df['Column_name'])
```

LABEL ENCODING:

In [7]:
```
from sklearn.preprocessing import LabelEncoder
# LE = LabelEncoder()
# df['New_column'] = LE.fit_transform(df['On_which_column'])
```

ONE HOT ENCODING:

In [9]:
```
# pd.get_dummies(df['Column_name'])
```

Various functions from statsmodel to perform linear regression

In [ ]:
```
import statsmodels
import statsmodels.api as sm
import statsmodels.stats.api as sms
from statsmodels.compat import lzip
from statsmodels.stats.outliers_influence import variance_inflation_factor
from statsmodels.graphics.gofplots import qqplot
from statsmodels.stats.anova import anova_lm
from statsmodels.formula.api import ols
from statsmodels.tools.eval_measures import rmse
from statsmodels.formula.api import ols
```

Sklearn libraries:

In [ ]:
```
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.linear_model import Linear_regression, Lasso, Ridge, LassoCV, R
from sklearn.feature_selection import RFE
from sklearn.ensemble import RandomForestRegressor,  BaggingRegressor
# RandomForestRegressor Learns the maximum of the data. (Suitable model) , B
```

'metrics' from sklearn is used for evaluating the model performance

In [ ]:
```
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_scor
```

END