# PREDICTING HOUSE PRICE USING MACHINE LEARNING

## Phase – 4

We further building our project by loading the Data cleansing, Splitting Data set into Testing and Training, Model and accuracy like Random forest and Linear regression or Support Vector Machine in Google colab Notebook.

## Data Cleansing:

Downloaded the **USA_Housing.csv** dataset from the Kaggle

We perform data cleansing operation

```
dataset.drop(['Avg. Area Income'],
      axis=1,
      inplace=True)
```

```
[8]  new_dataset = dataset.dropna()
```

```
[9]  new_dataset.isnull().sum()
```

```
Avg. Area House Age             0
Avg. Area Number of Rooms       0
Avg. Area Number of Bedrooms    0
Area Population                 0
Address                         0
dtype: int64
```

## Splitting Data Sets :

Now, we categorize the features depending on Categorial features and Calculate the No. of Categorial features of them.

```
from sklearn.preprocessing import OneHotEncoder
s = (new_dataset.dtypes == 'object')
object_cols = list(s[s].index)
print("Categorical variables:")
print(object_cols)
print('No. of. categorical features: ',
len(object_cols))
```

```
Categorical variables:
['Address']
No. of. categorical features:  1
```

## Splitting Dataset into Training and Testing :

X and Y splitting (i.e. Y is the SalePrice column and the rest of the other columns are X)

```
import pandas as pd
from sklearn.metrics import mean_absolute_error
from sklearn.model_selection import train_test_split
dataset = pd.read_csv("USA_Housing.csv")
X = dataset[['Avg. Area House Age','Avg. Area Number of Rooms','Avg. Area Number of Bedrooms']]
Y = dataset['Price']
X_train, X_valid, Y_train, Y_valid = train_test_split(
  X, Y, train_size=0.8, test_size=0.2, random_state=0)
```

## Model And Accuracy :

As we have to train the model to determine the continuous values, so we will be using these models like

- Support Vector Machine ( SVM )
- Linear Regression
- Random Forest Regressor

## Support Vector Machine (SVM):

SVM can be used for both regression and classification model. It finds the hyperplane in the n-dimensional plane

```
from sklearn import svm
from sklearn.svm import SVC
from sklearn.metrics import mean_absolute_percentage_error

model_SVR = svm.SVR()
model_SVR.fit(X_train,Y_train)
Y_pred = model_SVR.predict(X_valid)

print(mean_absolute_percentage_error(Y_valid, Y_pred))
```

```
0.28617436171038496
```

## Linear Regression :

Linear Regression predicts the final output-dependent value based on the given independent features. Like, here we have to predict SalePrice depending on features like MSSubClass.

```
from sklearn.linear_model import LinearRegression

model_LR = LinearRegression()
model_LR.fit(X_train, Y_train)
Y_pred = model_LR.predict(X_valid)

print(mean_absolute_percentage_error(Y_valid, Y_pred))
```

```
0.22488357290972008
```

## Random Forest Regressor

Random Forest is an ensemble technique that uses multiple of decision trees and can be used for both regression and classification tasks.

```
from sklearn.ensemble import RandomForestRegressor

model_RFR = RandomForestRegressor(n_estimators=10)
model_RFR.fit(X_train, Y_train)
Y_pred = model_RFR.predict(X_valid)

mean_absolute_percentage_error(Y_valid, Y_pred)
```

```
0.23724727747567836
```

## Evaluation :

The machine learning model is given the test data but without the price of the properties in order to predict the price for them given the various features for the properties. The predicted price is then compared to the actual price in the test data.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_squared_error, r2_score
data = pd.read_csv('USA_Housing.csv')
X = data[['Avg. Area House Age','Avg. Area Number of Rooms','Avg. Area Number of Bedrooms']]
y = data['Price']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
model = LinearRegression()
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)
print(f"Mean Squared Error: {mse:.2f}")
print(f"R-squared: {r2:.2f}")
new_data = pd.DataFrame({'Avg. Area House Age':[8],'Avg. Area Number of Rooms':[5],'Avg. Area Number of Bedrooms':[4]})
predicted_price = model.predict(new_data)
print(f"Predicted Price: {predicted_price[0]:.2f}")
```

```
Mean Squared Error: 81281121833.21
R-squared: 0.34
Predicted Price: 1330233.42
```


**Team Members:**

K.Santhosh

J.Saravanakumar

M.Sangara Sequvar

C.Mohan

**JP COLLEGE OF ENGINEERING**