SURYA GROUP OF INSTITUTION

VIKRAVANDI 605-652

PHASE 2 INNOVATION

PREDICTION HOUSE PRICES USING MACHINE LEARNING

(XG BOOSTER)

NAAN MUDHALVAN

PREPARED BY:

N.PERUMAL

REG NO :422221106012

ECE DEPARTMENT

3RD YEAR 5TH SEM

AI_ PHASE 2:

Consider exploring advanced regression techniques like XG Boost for Improved prediction accuracy.

PROPOSED SYSTEM:

In this proposed system, we focus on predicting the house price values using machine learning

algorithms like XG Boost regression model. We proposed the system “House price prediction using Machine Learning” we have predicted the House price using XG boost regression model. In this proposed system, we were able to train the machine from the various attributes of data points from the past to make a future prediction. We took data from the previous year stocks to train the model .The data set we used was from the official organization. Some of data was used to train the machine and the rest some data is used to test the data. The basic approach of the supervised learning model is to learn the patterns and relationships in the data from the training set and then reproduce them for the test data. We used the python pandas library for data processing which combined different datasets into a data frame. The raw data makes us to prepare the data for feature identification. The attributes were stories, no. of bed rooms, bath rooms, Availability of garage, swimming pool, fire place, year built, area in soft, sale price for a particular house. We used all these features to train the machine on XG boost regression and predicted the house price, which is the price for a given day. We also quantified the accuracy by using the predictions for the test set and the actual values. The proposed system gives the Predicted price.

ALOGORITHM:

We used the python pandas library for data processing which combined different datasets into a data frame. The raw data makes us to prepare the data for feature identification. XG for regression builds an additive model in a forward stage wise fashion. It allows for the optimization of arbitrary differentiable loss functions. In each stage, a regression tree is fit on the negative XG of the given loss function. The idea of boosting came out of the idea of whether a weak learner can be modified to become better. A weak hypothesis is defined as one whose performance is at least slightly better than random chance. The Objective is to minimize the loss of the model by adding weak hypothesis using a XG descent like procedure. This class of algorithms was described as a stage-wise additive model. This is because one new weak learner is added at a time and existing weak learners in the model are frozen and left unchanged.

Step 1: Load the data set df = pd.read csv(“ml_house_data_set.csv”)

Step 2: Replace categorical data with one-hot encoded data

Step 3: Remove the sale price from the feature data

Step 4: Create the features and labels X and Y arrays.

Step 5: Split the data set in a training set (70%) and a test set (30%). Step 6: Fit regression model.

Step 7: Save the trained model to a file trained_house_classifier_model.pkl

Step 8: Predict house worth using predict function

MODEL XG BOOST REGRESSOR :

INPUT:

```
# Importing the libraries
from sklearn.metrics import confusion_matrix
import xgboost as xgb
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, OneHotEncoder
import numpy as np
import matplotlib.pyplot as plt
import pandas as pd

# Importing the dataset
dataset = pd.read_csv('Churn_Modelling.csv')
X = dataset.iloc[:, 3:13].values
y = dataset.iloc[:, 13].values

# Encoding categorical data
labelencoder_X_1 = LabelEncoder()

X[:, 1] = labelencoder_X_1.fit_transform(X[:, 1])
labelencoder_X_2 = LabelEncoder()

X[:, 2] = labelencoder_X_2.fit_transform(X[:, 2])
onehotencoder = OneHotEncoder(categorical_features=[1])

X = onehotencoder.fit_transform(X).toarray()
```

```
X = X[:, 1:]

# Splitting the dataset into the Training set and Test set
X_train, X_test, y_train, y_test = train_test_split(
        X, y, test_size=0.2, random_state=0)

# Fitting XGBoost to the training data
my_model = xgb.XGBClassifier()
my_model.fit(X_train, y_train)

# Predicting the Test set results
y_pred = my_model.predict(X_test)

# Making the Confusion Matrix
cm = confusion_matrix(y_test, y_pred)
```

```
Prediction5 = model_xg.predict(X_test_scal)
```

```
plt.figure(figsize=(12,6))
    plt.plot(np.arange(len(Y_test)), Y_test, label='Actual Trend')
    plt.plot(np.arange(len(Y_test)), Prediction5, label='Predicted Trend')
    plt.xlabel('Data')
    plt.ylabel('Trend')
    plt.legend()
    plt.title('Actual vs Predicted')
```

OUTPUT :

```
Text(0.5, 1.0, 'Actual vs Predicted')
```

INPUT:

```
        sns.histplot((Y_test-Prediction4), bins=50)
```

OUTPUT:

```
<Axes: xlabel='Price', ylabel='Count'>
```