

**THE SUPERIOR UNIVERSITY**

**Submitted To:**

Sir Rasikh

**Name:**

Waqar Abbas

**Roll No:**

SU92-BSDSM-F23-010

**Section:**

BSDS-4A

**PAI Lab Task-1**

**Support Vector Regression (SVR) for House Price Prediction**

**Introduction**

This script preprocesses house price data and trains a Support Vector Regression (SVR) model to predict house prices based on given features. The process includes handling missing values, removing outliers, encoding categorical variables, normalizing numerical values, and training an SVR model.

**1. Importing Libraries**

The necessary Python libraries are imported:

* numpy and pandas for data handling.
* StandardScaler for feature scaling.
* SVR for regression modeling.
* OrdinalEncoder for encoding categorical data.

**2. Loading Datasets**

Two CSV files, train.csv and test.csv, are loaded into DataFrames df1 and df2 respectively. The training dataset (df1) includes house features and prices, while the test dataset (df2) contains house features but lacks price values.

**3. Checking for Missing Values**

The script prints the number of missing values in each column for both datasets. This helps in identifying which columns need imputation.

**4. Handling Missing Values**

A function FillNA(df) is defined to fill missing values:

* Categorical (object) columns are filled with the most frequent value (mode).
* Numerical columns are filled with the mean.

This function is applied to both df1 and df2, ensuring that there are no missing values.

**5. Removing Outliers**

Outliers are identified and removed based on the 1st and 99th percentiles:

* The lower and upper bounds are calculated for each numerical column.
* Values falling outside this range are considered outliers and removed.

This process is performed separately for both datasets.

**6. Encoding Categorical Variables**

A function ordinalencoder(df) is created to apply OrdinalEncoder on categorical columns. This converts categorical values into numerical representations for machine learning compatibility. The function is applied to both datasets.

**7. Feature Normalization**

A function apply\_z\_score(df) normalizes numerical features using Z-score normalization. This helps in stabilizing the range of values for better model performance.

**8. Feature Selection**

Certain features (MSZoning, LotFrontage, LotArea, Street, LotShape, LandContour) are dropped from both datasets. The target variable SalePrice is separated from df1 for training.

* x contains selected features from df1.
* y contains the target variable (SalePrice).
* df2\_features contains the selected features from df2 (test data).

**9. Training the SVR Model**

An SVR model is initialized and trained using the fit() method on x (features) and y (target variable).

**10. Making Predictions**

The trained SVR model predicts house prices for the test dataset using model.predict(df2\_features). The predictions are stored in predictions.

**11. Saving Predictions to CSV**

The predictions are saved into a new DataFrame output\_df with two columns:

* Id: The house ID from df2.
* SalePrice: The predicted house price.

The DataFrame is then saved as submission.csv using to\_csv() without the index column.

**Conclusion**

This script successfully preprocesses the dataset, removes inconsistencies, and trains an SVR model to predict house prices. The predictions are exported for further evaluation.