Practical Activity Regression analysis using SVM

1 Practical Activity

1.1 Regression Using SVM

This notebook is an exercise for developing a SVM model for regression. We apply the concepts discussed in Week 8.

Note: this activity is unmarked. It develops your skills for predictive model development using SVM.

2 The Housing dataset

For this practical activity, we will use the housing dataset (same dataset as Activity 4.1)

The Housing dataset is available online at - https://raw.githubusercontent.com/rasbt/python-machine-learning-book-3rdedition/master/ch10/housing.data.txt or - scikit-learn (https://github.com/scikit-learn/scikitlearn/blob/master/sklearn/datasets/data/boston_house_prices.csv)

Goal: our goal is to develop a SVM regerssion model to predict the value of a house given the other attributes i.e., our target is MEDV.

[6]:

2.1 Data Loading

```
#laoding from sklearn
from sklearn.datasets import load_boston
data = load_boston()
```

[8]:

```
import pandas as pd

# Read the DataFrame, first using the feature data
df = pd.DataFrame(data.data, columns = data.feature_names)

# Add a target column, and fill it with the target data
df['target'] = data.target# Show the first five rows
df.head()
```

[8]:

	CRIM	ZN	INDUS	CHAS	NOX	RM	AGE	DIS	RAD	TAX	\
0	0.00632	18.0	2.31	0.0	0.538	6.575	65.2	4.0900	1.0	296.0	
1	0.02731	0.0	7.07	0.0	0.469	6.421	78.9	4.9671	2.0	242.0	
2	0.02729	0.0	7.07	0.0	0.469	7.185	61.1	4.9671	2.0	242.0	
3	0.03237	0.0	2.18	0.0	0.458	6.998	45.8	6.0622	3.0	222.0	
4	0.06905	0.0	2.18	0.0	0.458	7.147	54.2	6.0622	3.0	222.0	

```
   PTRATIO       B  LSTAT  target
0     15.3  396.90   4.98    24.0
1     17.8  396.90   9.14    21.6
2     17.8  392.83   4.03    34.7
3     18.7  394.63   2.94    33.4
4     18.7  396.90   5.33    36.2
```

2.2 Create train and test set

[9]:
```python
from sklearn.model_selection import train_test_split

train, test = train_test_split(df, test_size = 0.3)

X_train = train.drop('target', axis=1)
y_train = train['target']

X_test = test.drop('target', axis = 1)
y_test = test['target']
```

[10]:
```python
from sklearn.preprocessing import MinMaxScaler
scaler = MinMaxScaler(feature_range=(0, 1))

x_train_scaled = scaler.fit_transform(X_train)
#reverting back to df
X_train = pd.DataFrame(x_train_scaled)

x_test_scaled = scaler.fit_transform(X_test)
X_test = pd.DataFrame(x_test_scaled)
```

2.3 Building the SVM Regression Model

[11]:
```python
from sklearn.svm import SVR
from sklearn.metrics import r2_score, mean_squared_error


svm_reg = SVR(kernel = 'rbf')
svm_reg.fit(X_train, y_train)
```

[11]:
```
SVR()
```

2.4 Evaluation

[12]:
```python
import numpy as np

predictions = svm_reg.predict(X_test)
```

```python
mse = mean_squared_error(y_test, predictions)
rmse = np.sqrt(mse)

print(f'RMSE value is: {rmse}')
```

```
RMSE value is: 5.56797557589552
```

[15]:
```python
svm_reg.n_support_
```

[15]:
```
array([346])
```

[17]:
```python
svm_reg.support_vectors_
```

[17]:
```
array([[4.15264132e-03, 0.00000000e+00, 3.71334311e-01, …,
         6.38297872e-01, 9.95814212e-01, 6.11280911e-01],
        [2.33528471e-03, 2.20000000e-01, 1.97947214e-01, …,
         6.91489362e-01, 9.49997478e-01, 4.47346485e-02],
        [4.37788609e-04, 0.00000000e+00, 4.20454545e-01, …,
         8.93617021e-01, 1.00000000e+00, 1.97277021e-01],
        …,
        [2.92795719e-04, 0.00000000e+00, 6.30498534e-02, …,
         6.48936170e-01, 9.94276060e-01, 2.66740761e-02],
        [3.22013248e-01, 0.00000000e+00, 6.46627566e-01, …,
         8.08510638e-01, 5.31166473e-01, 5.02917477e-01],
        [8.29719002e-04, 0.00000000e+00, 2.01612903e-01, …,
         7.02127660e-01, 1.00000000e+00, 1.88663518e-01]])
```