ay42-implementation-svm-regression

November 9, 2023

```
Day42 SVM Regression        By:Loga Aswin
```

[1]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Data Pre-processing :

[2]:
```
df = pd.read_csv('/content/Salary_dataset.csv')
```

[3]:
```
df.head()
```

[3]:
```
   Unnamed: 0  YearsExperience   Salary
0           0              1.2  39344.0
1           1              1.4  46206.0
2           2              1.6  37732.0
3           3              2.1  43526.0
4           4              2.3  39892.0
```

[4]:
```
df.drop('Unnamed: 0', axis=1, inplace=True)
```

[5]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 30 entries, 0 to 29
Data columns (total 2 columns):
 #   Column           Non-Null Count  Dtype
---  ------           --------------  -----
 0   YearsExperience  30 non-null     float64
 1   Salary           30 non-null     float64
dtypes: float64(2)
memory usage: 608.0 bytes
```

[6]:
```
df.describe()
```

[6]:
```
       YearsExperience        Salary
count        30.000000     30.000000
mean          5.413333  76004.000000
std           2.837888  27414.429785
```

```
min           1.200000   37732.000000
25%           3.300000   56721.750000
50%           4.800000   65238.000000
75%           7.800000  100545.750000
max          10.600000  122392.000000
```

[7]:

```
df.isnull().sum()
```

[7]:

```
YearsExperience    0
Salary             0
dtype: int64
```

[8]:

```
X = df.drop('Salary', axis=1)
Y = df['Salary']
```

[9]:

```
from sklearn.model_selection import train_test_split

x_train,x_test,y_train,y_test = train_test_split(X, Y, test_size=0.20,␣
 ↪random_state=42)
```

Create and Train SVM

[10]:

```
from sklearn.svm import SVR

svr = SVR(kernel='linear')
svr.fit(x_train, y_train)
```

[10]:

```
SVR(kernel='linear')
```

Predict Test Results :

[11]:

```
y_pred = svr.predict(x_test)
pd.DataFrame({'Actual':y_test, 'Predicted':y_pred})
```

[11]:

```
      Actual  Predicted
27  112636.0   62472.76
15   67939.0   62205.33
23  113813.0   62393.10
17   83089.0   62228.09
8    64446.0   62108.60
9    57190.0   62137.05
```

[13]:

```
# Evaluate Model Performance :

from sklearn.metrics import mean_absolute_error, mean_squared_error

mae = mean_absolute_error(y_test, y_pred)
mae
```

[13]:

```
22577.028333225255
```

[14]:

```
mse = mean_squared_error(y_test, y_pred)
mse
```

[14]:

```
943057673.9043975
```

[15]:

```
rmse = np.sqrt(mse)
rmse
```

[15]:

```
30709.244111576525
```