Lab 10 - CCPS 844 Data Mining

Salman AlMaskati

Answer the following questions and submit a PDF file on the D2L.

Select dataset/datasets of your choice.

Q-1 Select a dataset/datasets of your choice.

- Apply/Fit SVM Classification
- Evaluate the results

Q-2 Select a dataset/datasets of your choice.

- Apply/Fit SVM Regression
- Evaluate the results

Q1 - SVM Classification

In [ ]:
```
import pandas as pd

df=pd.read_csv("pima-indians-diabetes.csv",index_col=0)
df

feature_col=['pregnant','glucose','insulin','bmi','age']
target_col=['label']
X=df[feature_col]
y=df[target_col]
X
```

Out[ ]:

	pregnant	glucose	insulin	bmi	age
1	6	148	0	33.6	50
2	1	85	0	26.6	31
3	8	183	0	23.3	32
4	1	89	94	28.1	21
5	0	137	168	43.1	33
...	...	...	...	...	...
764	10	101	180	32.9	63
765	2	122	0	36.8	27
766	5	121	112	26.2	30
767	1	126	0	30.1	47
768	1	93	0	30.4	23

768 rows × 5 columns

In [ ]:
```
from sklearn import svm
clf = svm.SVC()
clf.fit(X, y)

clf.predict([[1,85,0,26.6,31]])
```

```
c:\Users\almas\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversio
nWarning: A column-vector y was passed when a 1d array was expected. Please change the s
hape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
```

Out[ ]:
```
array([0], dtype=int64)
```

In [ ]:
```
#split into training and testing
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size = 0.25, random_stat
```

In [ ]:
```
#scaling
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
X_train = sc.fit_transform(X_train)
X_test = sc.transform(X_test)
```

In [ ]:
```
#fitting into training set
from sklearn.svm import SVC
classifier = SVC(kernel = 'linear', random_state = 0)
classifier.fit(X_train, y_train)
```

```
c:\Users\almas\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversio
nWarning: A column-vector y was passed when a 1d array was expected. Please change the s
hape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
```

Out[ ]:
```
SVC(kernel='linear', random_state=0)
```

In [ ]:
```
y_pred = classifier.predict(X_test)
print(y_pred)
```

```
[1 0 0 1 0 0 1 1 0 0 1 1 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 1
 0 0 1 0 0 0 1 1 0 0 0 0 0 0 0 1 1 0 0 0 1 0 0 1 1 0 0 1 1 1 0 0 0 0 0 0 1
 1 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 1 0 0 1 0 0 0 0 0 0 0 0 0 0 1 0
 0 1 0 1 1 0 1 0 1 0 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1 0 0 0 0 0 0
 0 0 0 1 0 0 1 0 1 0 0 1 1 1 0 0 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 1 0
 0 1 0 0 0 0 0]
```

In [ ]:
```
#support vectors
clf.support_vectors_
```

Out[ ]:
```
array([[  5. , 116. ,   0. ,  25.6,  30. ],
       [ 10. , 115. ,   0. ,  35.3,  29. ],
       [  4. , 110. ,   0. ,  37.6,  30. ],
       ...,
```

```
       [  1. , 128. , 110. ,  36.5,  37. ],
       [  0. , 123. ,   0. ,  36.3,  52. ],
       [  1. , 126. ,   0. ,  30.1,  47. ]])
```

In [ ]:
```
#number of support vectors for each class
clf.n_support_
```

Out[ ]:
```
array([225, 226])
```

In [ ]:
```
from sklearn.metrics import confusion_matrix
confusion_matrix = confusion_matrix(y_test, y_pred)
print(confusion_matrix)
TP = confusion_matrix[1, 1]
TN = confusion_matrix[0, 0]
FP = confusion_matrix[0, 1]
FN = confusion_matrix[1, 0]
print("True Positive: ",TP)
print("True Negative: ",TN)
print("False Positive: ",FP)
print("False Negative: ",FN)
```

```
[[116  14]
 [ 28  34]]
True Positive:  34
True Negative:  116
False Positive:  14
False Negative:  28
```

In [ ]:
```
from sklearn import metrics
import numpy as np
print('Mean Absoulute Error',metrics.mean_absolute_error(y_test, y_pred))
print('Mean Squared Error:',metrics.mean_squared_error(y_test, y_pred))
print('Root Mean Squared Error: ',np.sqrt(metrics.mean_squared_error(y_test, y_pred)))
```

```
Mean Absoulute Error 0.21875
Mean Squared Error: 0.21875
Root Mean Squared Error:  0.46770717334674267
```

Q2 - SVM Regression

In [ ]:
```
import pandas as pd

#read csv
df=pd.read_csv("Advertising.csv",index_col=0)
#set features and target
feature_col=['TV','Radio','Newspaper']
target_col=['Sales']
# X and y data
X=df[feature_col].values
y=df[target_col].values

df
```

Out[ ]:

	TV	Radio	Newspaper	Sales
1	230.1	37.8	69.2	22.1

	TV	Radio	Newspaper	Sales
2	44.5	39.3	45.1	10.4
3	17.2	45.9	69.3	9.3
4	151.5	41.3	58.5	18.5
5	180.8	10.8	58.4	12.9
...	...	...	...	...
196	38.2	3.7	13.8	7.6
197	94.2	4.9	8.1	9.7
198	177.0	9.3	6.4	12.8
199	283.6	42.0	66.2	25.5
200	232.1	8.6	8.7	13.4

200 rows × 4 columns

In [ ]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.svm import SVR
```

In [ ]:
```
#scaling
from sklearn.preprocessing import StandardScaler
scaler_X = StandardScaler()
X_scaled = scaler_X.fit_transform(X)
```

In [ ]:
```
#create and fit SVR using scaled values
regressor = SVR(kernel='rbf', gamma='auto')
regressor.fit(X_scaled, y)
```

```
c:\Users\almas\anaconda3\lib\site-packages\sklearn\utils\validation.py:63: DataConversio
nWarning: A column-vector y was passed when a 1d array was expected. Please change the s
hape of y to (n_samples, ), for example using ravel().
  return f(*args, **kwargs)
```

Out[ ]:
```
SVR(gamma='auto')
```

In [ ]:
```
# data points for prediction
prediction = [[44.5,39.3,45.1]]

#scaled prediction
scaled_predction = scaler_X.transform(prediction)

#predict using sclaed input
predicted_value = regressor.predict(scaled_predction)

# Print the predicted value
print("Predicted Sales:", predicted_value)
```

```
Predicted Sales: [11.21552443]
```