CSE 6363-005: MACHINE LEARNING

REPORT

NAME: JAHNAVI DHANUNJAYA

UTA ID: 1002002747

- **Problem statement**
  - To do classification of iris.data using linear regression and to validate results using accuracy score and cross-validation
  - Coefficients for linear regression equation is obtained using Least Square Estimator

$$\widehat{\beta} = (\mathbf{A}^T\mathbf{A})^{-1}\mathbf{A}^T\mathbf{Y}$$

Here Y is target vector

A is feature vector

$\widehat{\beta}$ is the resultant vector

$\mathbf{A}^T$ is transposed feature vector

$\mathbf{A}^T\mathbf{A}$ Vector must be invertible

- **Data**
  - Iris dataset is a very commonly used data set to evaluate machine learning algorithms
  - Dataset contains five columns that are
    - Petal Length
    - Petal Width
    - Sepal Length
    - Sepal Width
    - Species Type
  - Iris is a flowering plant and it's flower is shown in following figure

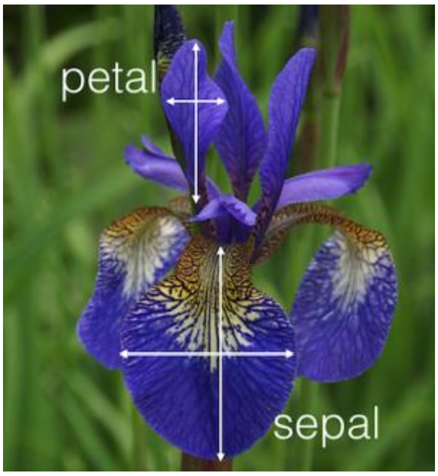

Fig.1-Iris flower

- The information that is present in the Iris.data is the info about it's sepal and petal dimension values and it also has which species does it belongs.
- Sample of data present in the Iris.data is displayed.

	sepallength	sepalwidth	petallength	petalwidth	Species
0	5.1	3.5	1.4	0.2	Iris-setosa
1	4.9	3.0	1.4	0.2	Iris-setosa
2	4.7	3.2	1.3	0.2	Iris-setosa
3	4.6	3.1	1.5	0.2	Iris-setosa
4	5.0	3.6	1.4	0.2	Iris-setosa

Fig.2-Sample data.

- **Training ML model Least Squares Estimator**
  - Data is read through read_csv function
  - Input data is separated for features and targeted output.
  - Targets are labelled by LabelEncoder
  - Entire data is separated for training and testing using train_test_split()

```
#Traning model
x = df.drop('Species', axis=1)
y1 = df['Species']
l = LabelEncoder()
y = l.fit_transform(y1)
x_train,x_test,y_train,y_test = train_test_split(x,y,test_size=1/2,random_state=1)
```

  - Next step is to determine coefficients

```
oness = np.ones(len(x_train))[:, np.newaxis]
A = np.hstack([x_train, oness])
y = y_train[:, np.newaxis]

# Direct least square regression
alpha = np.dot((np.dot(np.linalg.inv(np.dot(A.T,A)),A.T)),y)
print(alpha)
```

- **Results**
  - Alpha and Accuracy score for classification:

```
#Output classification(predction)
y_predect = []
for index, row in x_test.iterrows() :
    z1=alpha[0]*row['sepallength']+alpha[1]*row['sepalwidth']+alpha[2]*row['petallength']+alpha[3]*row['petalwidth']+alpha[4]
    z=[round(num) for num in z1.tolist()]
    y_predect.append(z)

#classification results
print('Accuracy Score for classification: ',accuracy_score(y_test,y_predect))
```

  - Output:

```
=============================================================
[[-0.31898887]
 [ 0.1035784 ]
 [ 0.44950247]
 [ 0.31953335]
 [ 0.46748288]]
Accuracy Score for classification:  0.9866666666666667
```

- Training ML model Linear regression (sklearn)

  - First objective is to train a model to do the classification of data.
  - If the target output that has to be predicted is Species which is becomes a multinominal class logistic regression is used for better result analysis.
  - Python 3.9 is used to perform the task
  - Loading data : following statements are used

```
import numpy as np
from numpy import mean,std
import seaborn as sns
import matplotlib.pyplot as plt

#loaind data
df = pd.read_csv(r'iris.csv')
#print(df.head())
```

- Data is loaded into ‘df’.
- Next step is to train model:

```
#Traning model

x = df.drop('Species', axis=1)
y = df['Species']
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.33, random_state=1)
logmodel = LogisticRegression(multi_class='multinomial', solver='lbfgs' , max_iter=100)
logmodel.fit(x_train, y_train)
```

- Data is separated into two variable ‘x’, ‘y’ for features and targeted output
- Then for x, y are separated to perform training and testing the model using train_test_split() function imported from sklearn.
- Model is trained using training samples of x, y.

• Output evaluation

- Classification report of model:

```
predictions = logmodel.predict(x_test)
print('Classification Report of model: ',classification_report(y_test, predictions))
```

- Output of above command

```
Classification Report of model:
                 precision    recall  f1-score   support

    Iris-setosa       1.00      1.00      1.00        17
Iris-versicolor       1.00      0.95      0.97        19
 Iris-virginica       0.93      1.00      0.97        14

       accuracy                           0.98        50
      macro avg       0.98      0.98      0.98        50
   weighted avg       0.98      0.98      0.98        50
```

- Confusion matrix of model

```
print('Confusion matrix of model:\n',confusion_matrix(y_test, predictions))
sns.heatmap(pd.DataFrame(confusion_matrix(y_test,predictions)))
plt.show()
```

- Output

```
Confusion matrix of model:
 [[17  0  0]
 [ 0 18  1]
 [ 0  0 14]]
```

- Heatmap representation

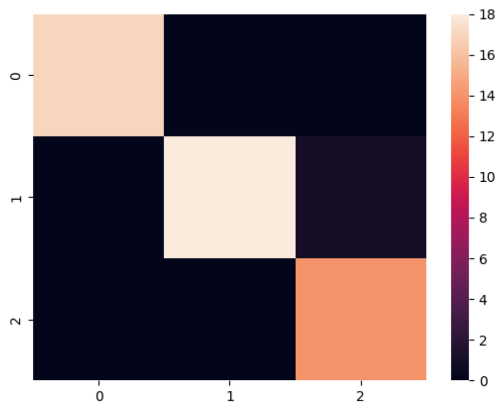

Fig.3- Heatmap representation of Confusion matrix of model.

- Accuracy and cross-validation:

```
print('Accuracy score of model: ',accuracy_score(y_test, predictions))

cv = RepeatedStratifiedKFold(n_splits=10, n_repeats=3, random_state=1)
# evaluate the model and collect the scores
n_scores = cross_val_score(logmodel, x_train, y_train, scoring='accuracy', cv=cv, n_jobs=-1)
# report the model performance
print('Mean Accuracy: %.3f (%.3f)' % (mean(n_scores), std(n_scores)))
```

- Output

```
Accuracy score of model:  0.98
Mean Accuracy: 0.970 (0.046)
```

- Another ML model
  - In the previous model Species are the targeted output and there can be other like sepallength for which linear regression can be used and program follows.

```
import pandas as pd
import numpy as np
from numpy import mean,std
import seaborn as sns
import matplotlib.pyplot as plt

#loaind data
df = pd.read_csv(r'iris.csv')
#print(df.head())


from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.metrics import classification_report
from sklearn.metrics import confusion_matrix,accuracy_score
from sklearn.model_selection import cross_val_score,RepeatedStratifiedKFold

#Traning model
mapping = {
    'Iris-setosa' : 1,
    'Iris-versicolor' : 2,
    'Iris-virginica' : 3
}

x = df.drop('sepallength', axis=1).replace(mapping)
y = df['sepallength']

x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.33, random_state=1)
linmodel = LinearRegression()
linmodel.fit(x_train, y_train)

print('Coefficients: ', linmodel.coef_)
```

  - Coefficient of linear regression are as followes

```
Coefficients:  [ 0.5424607   0.82435319 -0.45945426 -0.37504998]
```