Scikit_Assignment 02_Solution

July 3, 2020

1 Assignment 02: Evaluate the Diabetes Dataset

The comments/sections provided are your cues to perform the assignment. You don't need to limit yourself to the number of rows/cells provided. You can add additional rows in each section to add more lines of code.

If at any point in time you need help on solving this assignment, view our demo video to understand the different steps of the code.

Happy coding!

1: Import the dataset

[1]:
```
#Import the required libraries
import pandas as pd
```

[4]:
```
#Import the diabetes dataset
df_diabetes_data = pd.read_csv("pima-indians-diabetes.data", header=None)
```

2: Analyze the dataset

[6]:
```
#View the first five observations of the dataset
df_diabetes_data.head()
```

[6]:

	0	1	2	3	4	5	6	7	8
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

3: Find the features of the dataset

[7]:
```
#Use the .NAMES file to view and set the features of the dataset
features_name =␣
 ↪["Pregnant","glucose","bp","skin","insulin","bmi","pedigree","age","label"]
```

[8]:
```
#Use the feature names set earlier and fix it as the column headers of the␣
↪dataset
df_diabetes_data = pd.read_csv("pima-indians-diabetes.data",␣
↪header=None,names=features_name)
```

[9]:
```
#Verify if the dataset is updated with the new headers
df_diabetes_data.head()
```

[9]:

	Pregnant	glucose	bp	skin	insulin	bmi	pedigree	age	label
0	6	148	72	35	0	33.6	0.627	50	1
1	1	85	66	29	0	26.6	0.351	31	0
2	8	183	64	0	0	23.3	0.672	32	1
3	1	89	66	23	94	28.1	0.167	21	0
4	0	137	40	35	168	43.1	2.288	33	1

[10]:
```
#View the number of observations and features of the dataset
df_diabetes_data.shape
```

[10]:
```
(768, 9)
```

4: Find the response of the dataset

[12]:
```
#Select features from the dataset to create the model
feature_select_cols= ["Pregnant","insulin","bmi","age"]
```

[13]:
```
#Create the feature object
x_feature = df_diabetes_data[feature_select_cols]
```

[14]:
```
#Create the reponse object
y_target = df_diabetes_data["label"]
```

[15]:
```
#View the shape of the feature object
x_feature.shape
```

[15]:
```
(768, 4)
```

[16]:
```
#View the shape of the target object
y_target.shape
```

[16]:
```
(768,)
```

5: Use training and testing datasets to train the model

[18]:
```
#Split the dataset to test and train the model
from sklearn.model_selection import train_test_split
x_train, x_test, y_train, y_test = train_test_split(x_feature, y_target,␣
↪random_state = 1)
```

6: Create a model to predict the diabetes outcome

[21]:
```
# Create a logistic regression model using the training set
from sklearn.linear_model import LogisticRegression
logreg = LogisticRegression()
logreg.fit(x_train,y_train)
```

[21]:
```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
                   intercept_scaling=1, l1_ratio=None, max_iter=100,
                   multi_class='auto', n_jobs=None, penalty='l2',
                   random_state=None, solver='lbfgs', tol=0.0001, verbose=0,
                   warm_start=False)
```

[25]:
```
#Make predictions using the testing set
y_pred = logreg.predict(x_test)
```

7: Check the accuracy of the model

[28]:
```
#Evaluate the accuracy of your model
from sklearn import metrics
print(metrics.accuracy_score(y_test,y_pred))
```

```
0.6927083333333334
```

[35]:
```
#Print the first 30 actual and predicted responses
print("Actual:    ",y_test.values[0:30] )
print("Predicted: ",y_pred[0:30])
```

```
Actual:     [0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 0 0 0 1 1 1 1 0 0 0 1 0 1]
Predicted:  [0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0]
```

[ ]:
```

```