Assignment-2_Machine_Learning_Valapadasu_UdayBhaskar

July 1, 2024

[37]:
```
#Assignment 2: Machine Learning
#Name: Uday Bhaskar Valapadasu
#ID: 11696364
```

[38]:
```
#Import Statements

import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
import warnings
# Suppress the FutureWarning
warnings.simplefilter(action='ignore', category=FutureWarning)
```

[39]:
```
# Using the diabetes_df.csv created from assignment - 1 & Created a Pandas␣
↪dataframe from diabetes_df.csv and named it assignment2_df

assignment2_df = pd.read_csv("diabetes_df.csv")
assignment2_df
```

[39]:
```
     Pregnancies  Glucose  BloodPressure  SkinThickness  Insulin   BMI  \
0              6      148             72             35      150  33.6
1              1       85             66             29      150  26.6
2              8      183             64              0      150  23.3
3              1       89             66             23       94  28.1
4              0      137             40             35      168  43.1
..           …        …              …              …       …    …
763           10      101             76             48      180  32.9
764            2      122             70             27      150  36.8
765            5      121             72             23      112  26.2
766            1      126             60              0      150  30.1
767            1       93             70             31      150  30.4

     DiabetesPedigreeFunction  Age  Target
0                       0.627   50       1
1                       0.351   31       0
2                       0.672   32       1
3                       0.167   21       0
```

```
4                        2.288   33       1
..                          …   …       …
763                      0.171   63       0
764                      0.340   27       0
765                      0.245   30       0
766                      0.349   47       1
767                      0.315   23       0

[768 rows x 9 columns]
```

[40]:

```python
# Setup the Machine Learning Model:
#Dividing the data into features (X) array and target (y) array.

#features array
X = assignment2_df.drop(['Target'], axis=1)
#target array
y = assignment2_df['Target']
```

[41]:

```python
# Splitting the dataset into 80-20, 70-30, and 60-40 ratios. (Example: 80-20␣
↪means, 80% training data, 20% testing data, and so on.)

#Split-1 into 80-20
X_train1, X_test1, y_train1, y_test1 = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)

#Split-2 into 70-30
X_train2, X_test2, y_train2, y_test2 = train_test_split(X, y, test_size=0.3,␣
↪random_state=42)

#Split-3 into 60-40
X_train3, X_test3, y_train3, y_test3 = train_test_split(X, y, test_size=0.4,␣
↪random_state=42)
```

[42]:

```python
# For each data split, apply logistic regression machine learning model to␣
↪build confusion matrix and accuracy estimates.
# So, importing the necessary accordinhly.

from sklearn.linear_model import LogisticRegression
from sklearn.metrics import confusion_matrix, accuracy_score, roc_auc_score
```

[43]:

```python
# Performing Logistic Regression & Building Confusion Matrix and Accuracy for␣
↪Split-1 Training:Test(80:20) Ratio
lr_split_1 = LogisticRegression(random_state=42, max_iter=1000)
lr_split_1.fit(X_train1, y_train1)
y_pred1 = lr_split_1.predict(X_test1)
y_pred_proba1 = lr_split_1.predict_proba(X_test1)[:, 1]
cm_split_1 = confusion_matrix(y_test1, y_pred1)
```

```
accuracy_split_1 = accuracy_score(y_test1, y_pred1)
auc_split_1 = roc_auc_score(y_test1, y_pred_proba1)

# Displaying the Confusion Matrix, Accuracy, and AUC
print("Confusion Matrix and Accuracy for Split-1 Training:Test(80:20)")
print("Confusion Matrix:")
print(cm_split_1)
print("Accuracy:", accuracy_split_1)
print("AUC:", auc_split_1)
```

```
Confusion Matrix and Accuracy for Split-1 Training:Test(80:20)
Confusion Matrix:
[[80 19]
 [19 36]]
Accuracy: 0.7532467532467533
AUC: 0.8165289256198347
```

[44]:

```
# Performing Logistic Regression & Building Confusion Matrix, Accuracy, and AUC␣
 ↪for Split-2 Training:Test(70:30) Ratio
lr_split_2 = LogisticRegression(solver='lbfgs', random_state=42, max_iter=1000)
lr_split_2.fit(X_train2, y_train2)
y_pred2 = lr_split_2.predict(X_test2)
y_pred_proba2 = lr_split_2.predict_proba(X_test2)[:, 1]
cm_split_2 = confusion_matrix(y_test2, y_pred2)
accuracy_split_2 = accuracy_score(y_test2, y_pred2)
auc_split_2 = roc_auc_score(y_test2, y_pred_proba2)

# Displaying the Confusion Matrix, Accuracy, and AUC
print("Confusion Matrix, Accuracy, and AUC for Split-2 Training:Test(70:30)")
print("Confusion Matrix:")
print(cm_split_2)
print("Accuracy:", accuracy_split_2)
print("AUC:", auc_split_2)
```

```
Confusion Matrix, Accuracy, and AUC for Split-2 Training:Test(70:30)
Confusion Matrix:
[[121  30]
 [ 30  50]]
Accuracy: 0.7402597402597403
AUC: 0.7964403973509934
```

[45]:

```
# Performing Logistic Regression & Building Confusion Matrix, Accuracy, and AUC␣
 ↪for Split-3 Training:Test(60:40) Ratio
lr_split_3 = LogisticRegression(random_state=42, max_iter=1000)
lr_split_3.fit(X_train3, y_train3)
y_pred3 = lr_split_3.predict(X_test3)
y_pred_proba3 = lr_split_3.predict_proba(X_test3)[:, 1]
cm_split_3 = confusion_matrix(y_test3, y_pred3)
```

```
accuracy_split_3 = accuracy_score(y_test3, y_pred3)
auc_split_3 = roc_auc_score(y_test3, y_pred_proba3)

# Displaying the Confusion Matrix, Accuracy, and AUC
print("Confusion Matrix, Accuracy, and AUC for Split-3 Training:Test(60:40)␣
↪Ratio")
print("Confusion Matrix:")
print(cm_split_3)
print("Accuracy:", accuracy_split_3)
print("AUC:", auc_split_3)
```

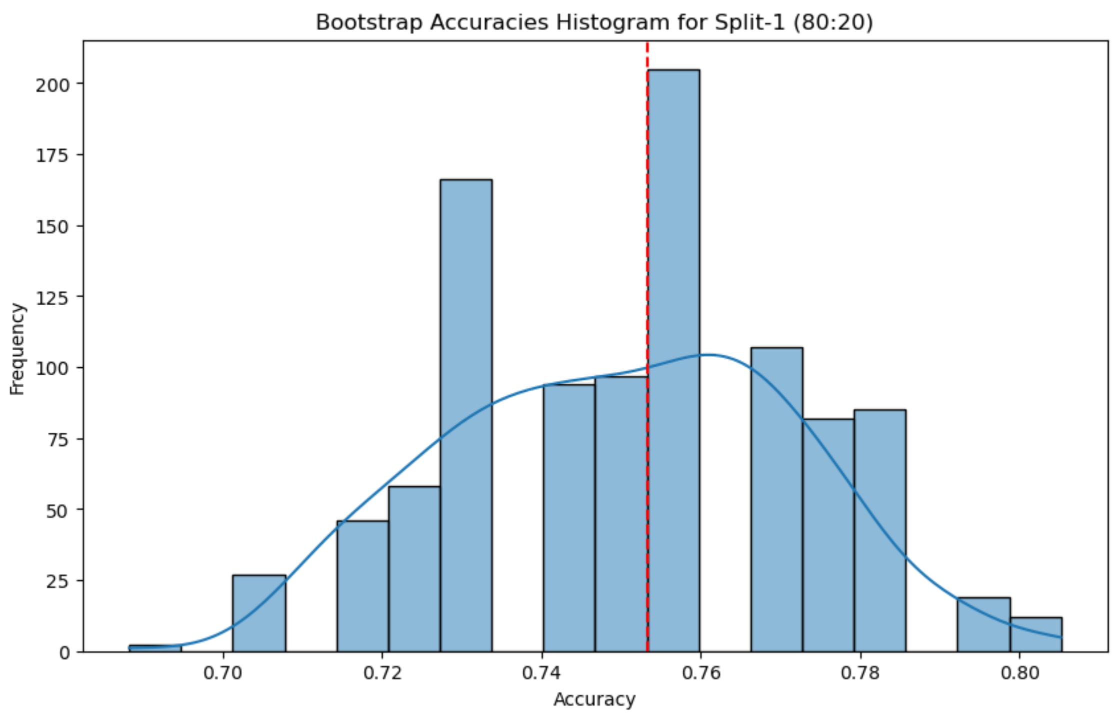
```
Confusion Matrix, Accuracy, and AUC for Split-3 Training:Test(60:40) Ratio
Confusion Matrix:
[[166  40]
 [ 36  66]]
Accuracy: 0.7532467532467533
AUC: 0.8213401865600609
```

[46]:
```
#Which data split is providing you the best accuracy?

#Ans: As, we observe split_1(80-20), split_3(60-40) ratio are on a tie giving␣
↪the best accuracy of 75% when compared to data split_2(70:30)
# Here, I am selecting split_1(80-20) for the further analysis.
```

[47]:
```
# Importing all the necessary libraries
from sklearn.utils import resample
import numpy as np
import matplotlib.pyplot as plt
from sklearn.metrics import roc_curve, roc_auc_score
```

[48]:
```
# Compute ROC curve and ROC area for Split-1
fpr_split1, tpr_split1, thresholds_split1 = roc_curve(y_test1, y_pred_proba1)
roc_auc_split1 = auc(fpr_split1, tpr_split1)

# Bootstrap analysis for Split-1 (80:20)
n_iterations = 1000
bootstrap_accuracies = []

# Perform bootstrap sampling
for i in range(n_iterations):
    X_resampled, y_resampled = resample(X_train1, y_train1, replace=True,␣
↪random_state=i)
    lr_split_1.fit(X_resampled, y_resampled)  # Fit logistic regression on␣
↪resampled data
    y_pred_resampled = lr_split_1.predict(X_test1)  # Predict on test set
    accuracy_resampled = accuracy_score(y_test1, y_pred_resampled)  # Calculate␣
↪accuracy
```

```python
    bootstrap_accuracies.append(accuracy_resampled)  # Store accuracy from each␣
 ↪iteration

# Calculate p-value and confidence intervals
p_value_split1 = np.mean(np.array(bootstrap_accuracies) >= accuracy_split_1)
confidence_interval_split1 = np.percentile(bootstrap_accuracies, [2.5, 97.5])

# Plot histogram of bootstrap accuracies
plt.figure(figsize=(10, 6))
sns.histplot(bootstrap_accuracies, kde=True)
plt.axvline(accuracy_split_1, color='r', linestyle='--')
plt.xlabel('Accuracy')
plt.ylabel('Frequency')
plt.title('Bootstrap Accuracies Histogram for Split-1 (80:20)')
plt.show()

# Write the report with metrics for Split-1 (80:20)
report_split1 = f"""
Selected Data Split Ratio (80:20) Metrics:
  Confusion Matrix:
{cm_split_1}
  Accuracy: {accuracy_split_1}
  AUC: {auc_split_1}
  ROC AUC: {roc_auc_split1}
  ROC Thresholds: {thresholds_split1}

Bootstrap Analysis for Split-1:
  P-Value: {p_value_split1}
  Confidence Interval: {confidence_interval_split1}
"""

print(report_split1)
```




```
Selected Data Split Ratio (80:20) Metrics:
  Confusion Matrix:
[[80 19]
 [19 36]]
  Accuracy: 0.7532467532467533
  AUC: 0.8165289256198347
  ROC AUC: 0.8165289256198347
  ROC Thresholds: [1.97062516 0.97062516 0.96040546 0.87284274 0.87075569
0.78031972
 0.77299541 0.77209963 0.76866991 0.72395837 0.70889906 0.70883013
 0.69285014 0.66792411 0.66001388 0.65390286 0.65205796 0.63552015
 0.62281485 0.61803155 0.57749256 0.56826283 0.55545133 0.54548516
 0.5319286  0.5140412  0.49408098 0.49150511 0.4436292  0.42606807
 0.40481265 0.37964985 0.36247904 0.327527   0.28621408 0.27489661
 0.26885186 0.26091845 0.26086909 0.25787497 0.23546844 0.23035508
 0.21334583 0.20356076 0.15930276 0.15109309 0.1416267  0.14056454
 0.13153163 0.12834498 0.12069508 0.11672998 0.04015114 0.0384741
 0.00415931]

Bootstrap Analysis for Split-1:
  P-Value: 0.51
  Confidence Interval: [0.70779221 0.79220779]
```