Artificial intelegence

ASSIGNMENT # 3

Submitted by : **Muhammad Umar**

Reg no. : **712**

Dep. : **BS.SE. 4th Semester**

Submitted To : Mr. Zubair

Submission Date : 15.June 2024

1. Accuracy Metrics Calculation:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, precision_score, recall_score,
f1_score

# Load dataset (replace 'your_dataset.csv' with your actual file)
data = pd.read_csv('your_dataset.csv')

# Separate features (X) and target variable (y)
X = data.drop('target_column', axis=1)  # Replace 'target_column' with your
target column name
y = data['target_column']

# Train-test split (80% train, 20% test)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)

# Model training
model = LogisticRegression()
model.fit(X_train, y_train)

# Prediction
y_pred = model.predict(X_test)

# Metric calculation
accuracy = accuracy_score(y_test, y_pred)
precision = precision_score(y_test, y_pred, average='weighted')
recall = recall_score(y_test, y_pred, average='weighted')
f1 = f1_score(y_test, y_pred, average='weighted')

# Print and interpret results
print("Accuracy:", accuracy)
print("Precision:", precision)
print("Recall:", recall)
print("F1-Score:", f1)
```

2. Confusion Matrix Interpretation:

```
from sklearn.metrics import confusion_matrix

# Generate confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Print and interpret
print("Confusion Matrix:\n", cm)
print("TP:", cm[0][0])  # True Positives
print("FP:", cm[0][1])  # False Positives
print("TN:", cm[1][1])  # True Negatives
print("FN:", cm[1][0])  # False Negatives

# Analyze the matrix to identify potential issues
# ... (Add your analysis here based on the confusion matrix values)
```

3. ROC/AUC Calculation:

```
from sklearn.metrics import roc_curve, roc_auc_score
import matplotlib.pyplot as plt

# Get ROC curve data
fpr, tpr, _ = roc_curve(y_test, model.predict_proba(X_test)[:, 1])

# Plot ROC curve
plt.plot(fpr, tpr, label='ROC Curve (area = %0.2f)' % roc_auc_score(y_test,
model.predict_proba(X_test)[:, 1]))
plt.plot([0, 1], [0, 1], 'k--')  # Plot perfect classification line
plt.xlabel('False Positive Rate')
plt.ylabel('True Positive Rate')
plt.title('ROC Curve')
plt.legend()
plt.show()

# Calculate AUC
auc = roc_auc_score(y_test, model.predict_proba(X_test)[:, 1])

# Interpret results
print("AUC:", auc)
# ... (Explain the AUC value in the context of your model)
```

4. Cross-Validation Reporting:

```
from sklearn.model_selection import KFold

# K-Fold cross-validation
kfold = KFold(n_splits=5, shuffle=True, random_state=42)
cv_scores = []

# Iterate through folds
for train_index, test_index in kfold.split(X):
    X_train, X_test = X.iloc[train_index], X.iloc[test_index]
    y_train, y_test = y.iloc[train_index], y.iloc[test_index]

    # Train model on current fold
    model.fit(X_train, y_train)

    # Evaluate on hold-out set
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test, y_pred)
    cv_scores.append(accuracy)

# Calculate mean and standard deviation
mean_accuracy = np.mean(cv_scores)
std_dev = np.std(cv_
```

Sources

1. medium.com/mlearning-ai/how-to-manage-an-end-to-end-machine-learning-project-with-mlflow-part-1-ff2e70d81789
2. github.com/AlbertoBarbado/mbti-text-classifier
3. stats.stackexchange.com/questions/490048