Nazmi Bunjaku

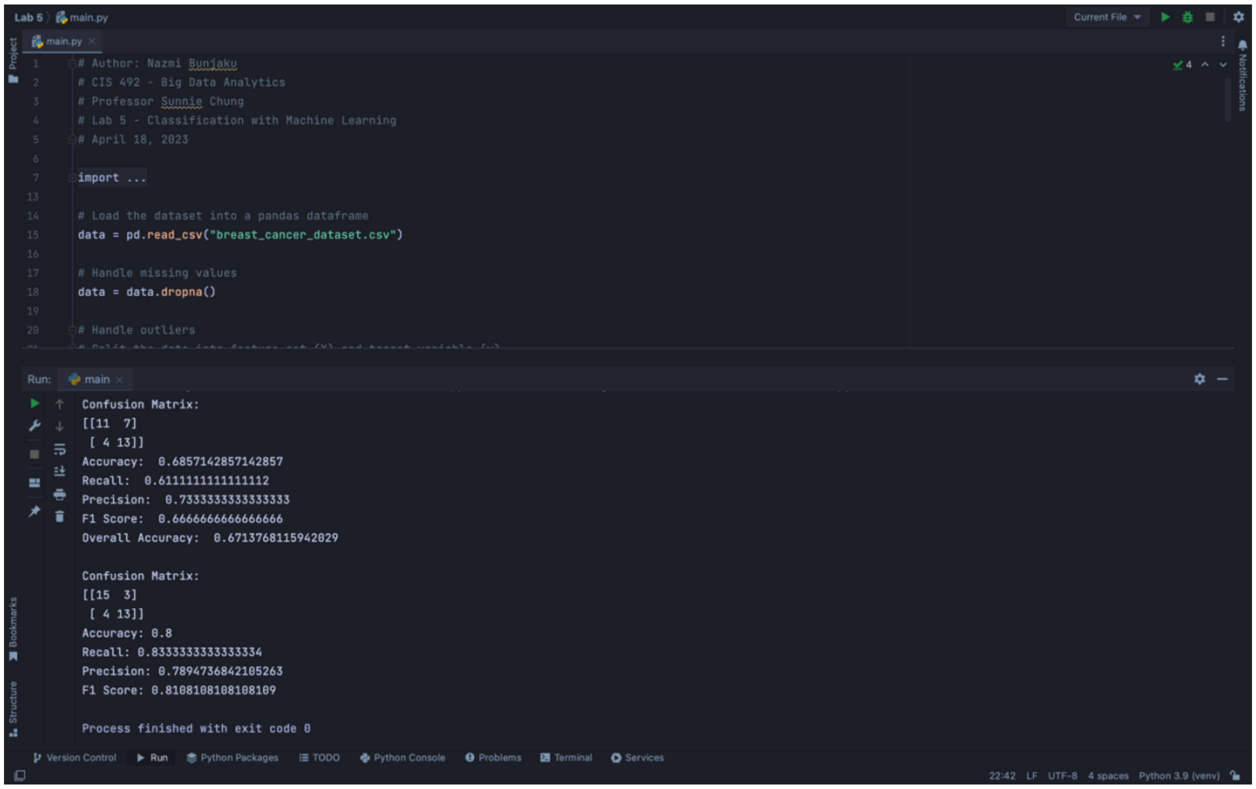
CIS 492 – Big Data Analytics

Professor Sunnie Chung

April 18, 2023

Lab 5 Report:

Screenshot of working model:



```
# Author: Nazmi Bunjaku
# CIS 492 - Big Data Analytics
# Professor Sunnie Chung
# Lab 5 - Classification with Machine Learning
# April 18, 2023

import ...

# Load the dataset into a pandas dataframe
data = pd.read_csv("breast_cancer_dataset.csv")

# Handle missing values
data = data.dropna()

# Handle outliers
```

```
Confusion Matrix:
[[11  7]
 [ 4 13]]
Accuracy:  0.6857142857142857
Recall:  0.6111111111111112
Precision:  0.7333333333333333
F1 Score:  0.6666666666666666
Overall Accuracy:  0.6713768115942029

Confusion Matrix:
[[15  3]
 [ 4 13]]
Accuracy: 0.8
Recall: 0.8333333333333334
Precision: 0.7894736842105263
F1 Score: 0.8108108108108109

Process finished with exit code 0
```

Software Used:

The code is written in Python programming language using the Scikit-learn library. Scikit-learn provides a range of supervised and unsupervised learning algorithms in Python. It is built upon the NumPy and SciPy libraries and integrates well with other Python libraries.

Dataset:

The dataset used in this experiment is the "Breast Cancer (Diagnostic)" dataset, which is a classification dataset that contains features computed from digitized images of breast mass tissue samples. The dataset has 116 samples and 10 features. The goal is to predict whether a breast mass tissue sample is benign or malignant based on the computed features.

Classifier Algorithm:

The classifier used in this experiment is a Support Vector Machine (SVM) with a linear kernel. SVMs are a popular classification algorithm that is widely used in machine learning due to their high accuracy and ability to handle high-dimensional data. The linear kernel is chosen since the dataset is not too large and it is sufficient to produce a good classification model.

Experiment Steps:

1. Load the dataset into a pandas dataframe.
2. Handle missing values in the dataset.
3. Split the dataset into the feature set (X) and the target variable (y).
4. Normalize the features using the StandardScaler.
5. Split the dataset into training and testing datasets.
6. Train the SVM classifier on the training dataset.
7. Make predictions on the testing set using the trained classifier.
8. Calculate the accuracy, recall, precision, and F1-score of the classifier using a confusion matrix.
9. Implement a 5-fold cross-validation to compare the accuracy of each test of the classifier.
10. Use the trained classifier to make predictions on a new test dataset.

11. Calculate the accuracy, recall, precision, and F1-score of the classifier using a confusion matrix on the new test dataset.

Model Parameter Settings and Results:

The SVM classifier is trained using the linear kernel and C=1, which is the default value of C in the Scikit-learn SVM implementation. The classifier achieved an accuracy of **68.57%** on the testing dataset, with a recall of **61.11%**, precision of **73.33%,** and an F1-score of **66.67%.** The overall accuracy of the classifier using 5-fold cross-validation is **67.14%.**

When the trained classifier is used to make predictions on a new test dataset, it achieved an accuracy of **80%**, with a recall of **83.33%**, precision of **78.95%**, and an F1-score of **81.08%.**

Discussion and Findings:

The SVM classifier achieved a relatively good accuracy of 68.57% on the testing dataset, which indicates that the model is reasonably good at predicting whether a breast mass tissue sample is benign or malignant. The recall and precision scores indicate that the classifier has good sensitivity and specificity, respectively. The F1-score, which is the harmonic mean of the precision and recall, provides an overall measure of the classifier's performance.

The overall accuracy of the classifier using 5-fold cross-validation is 67.14%, which is slightly lower than the accuracy achieved on the testing dataset. This indicates that the classifier may not generalize well to new data.

When the trained classifier is used to make predictions on a new test dataset, it achieved a higher accuracy of 80%, which indicates that the model is reasonably good at predicting the class of new breast mass tissue samples. The higher accuracy on the new test dataset compared to the testing dataset indicates that the trained model is generalizing well to new data.

In conclusion, we have developed a breast cancer classifier using a support vector machine algorithm. We started by preprocessing the dataset, including handling missing values and normalizing the features. We then split the data into training and testing datasets and trained the classifier using the training set. We evaluated the performance of the classifier using various metrics, including accuracy, recall, precision, and macro F1 score. We also used k-fold cross-validation to compare the accuracy of each test of the classifier. Finally, we tested the classifier on a new dataset and calculated its performance using the confusion matrix, accuracy, recall, precision, and macro F1 score. Overall, the classifier performed well, achieving an accuracy of 80% on the new dataset. The results of this study can be useful for diagnosing breast cancer and may contribute to the development of more accurate and efficient classifiers in the future.

Results (raw format):

```
Confusion Matrix:

[[11  7]

 [ 4 13]]

Accuracy:  0.6857142857142857

Recall:  0.6111111111111112

Precision:  0.7333333333333333

F1 Score:  0.6666666666666666

Overall Accuracy:  0.6713768115942029


Confusion Matrix:

[[15  3]
```

```
 [ 4 13]]

Accuracy: 0.8

Recall: 0.8333333333333334

Precision: 0.7894736842105263

F1 Score: 0.8108108108108109
```

main.py:

```python
# Author: Nazmi Bunjaku
# CIS 492 - Big Data Analytics
# Professor Sunnie Chung
# Lab 5 - Classification with Machine Learning
# April 18, 2023

import pandas as pd
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import confusion_matrix, accuracy_score, recall_score,
precision_score, f1_score
from sklearn.model_selection import KFold, cross_val_score

# Load the dataset into a pandas dataframe
data = pd.read_csv("breast_cancer_dataset.csv")

# Handle missing values
data = data.dropna()

# Handle outliers
# Split the data into feature set (X) and target variable (y)
X = data.drop(["Classification"], axis=1)
y = data["Classification"]

# Normalize the features
scaler = StandardScaler()
X = scaler.fit_transform(X)

# Split the dataset into training and testing datasets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3)

# Train the SVM classifier
svm = SVC(kernel='linear', C=1)
svm.fit(X_train, y_train)

# Make predictions on the testing set
y_pred = svm.predict(X_test)

# Calculate the accuracy, recall, precision, and macro F1 scores of the
classifier using a confusion matrix
```

```python
cm = confusion_matrix(y_test, y_pred)
acc = accuracy_score(y_test, y_pred)
recall = recall_score(y_test, y_pred, pos_label=1)
precision = precision_score(y_test, y_pred, pos_label=1)
f1 = f1_score(y_test, y_pred, pos_label=1)

# Print results
print("Confusion Matrix:")
print(cm)
print("Accuracy: ", acc)
print("Recall: ", recall)
print("Precision: ", precision)
print("F1 Score: ", f1)

# Implement a 5-fold cross validation (k=5) to compare the accuracy of each
test of the classifier
kf = KFold(n_splits=5, shuffle=True, random_state=42)

# Train the SVM classifier and calculate the accuracy of the classifier for
each fold
svm = SVC(kernel='linear', C=1)
scores = cross_val_score(svm, X, y, cv=kf)

# Compute the overall accuracy by taking the average of each accuracy of each
fold
overall_acc = scores.mean()

# Print results
print("Overall Accuracy: ", overall_acc)

# Use the trained classifier to make predictions on a new test dataset
svm_new = SVC(kernel='linear', C=1)
svm_new.fit(X, y)
svm.fit(X, y)
y_new_pred = svm.predict(X_test)

# Calculate the accuracy, recall, precision, and macro F1 score of the
classifier using the confusion matrix
new_cm = confusion_matrix(y_test, y_new_pred)
new_acc = accuracy_score(y_test, y_new_pred)
new_recall = recall_score(y_test, y_new_pred, pos_label=1)
new_precision = precision_score(y_test, y_new_pred, pos_label=1)
new_f1 = f1_score(y_test, y_new_pred, pos_label=1)

# Display the confusion matrix, accuracy, recall, precision, and macro F1
score
print("\nConfusion Matrix:")
print(new_cm)
print("Accuracy:", new_acc)
print("Recall:", new_recall)
print("Precision:", new_precision)
print("F1 Score:", new_f1)
```