Department Of CREATIVE TECHNOLOGIES

AIR UNIVERSITY
FCAI, ISLAMABAD

Machine Learning Lab (AI332L)

Class:	BS Artificial Intelligence	**Semester:**	5th (Fall 2023)
Instructor:	Dr. M. Ehatisham-ul-Haq	**Email:**	ehtisham@mail.au.edu.pk

| Lab 07 |

Applying Classifiers in Machine Learning Problems Using Python

Lab Objective:

In this lab tutorial, we will apply different supervised learning classifiers on machine learning datasets using Python. This tutorial will guide you through the entire process, from loading the dataset to evaluating the classifiers.

We will also apply K-means clustering, a commonly used machine-learning technique for unsupervised classification tasks.

Setting up Python with Google Colab

1. Go to Google Colab.
2. Create a new notebook by clicking on **File** > **New Notebook**.
3. You're now ready to start writing and executing Python code in the notebook!

Building Supervised Machine Learning Classification Models

1. Importing Necessary Libraries

```
import numpy as np
from sklearn import datasets
from sklearn.model_selection import train_test_split
from sklearn.tree import DecisionTreeClassifier
from sklearn.svm import SVC
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import accuracy_score, classification_report
```

2. Load and Explore Sample Dataset: Iris Dataset

```
# Load the Iris dataset
iris = datasets.load_iris()
X = iris.data
y = iris.target

# Display the first few rows of the dataset
print(iris.feature_names)
```

```
print(X[:5])
print(iris.target_names)
print(y[:5])
```

3. Load and Explore Sample Dataset: Iris Dataset

```
# Split the data into training and testing sets (80% training, 20% testing)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=42)
```

4. Applying Classifiers: Decision Tree and SVM

```
# Create and train a Decision Tree Classifier
dt_classifier = DecisionTreeClassifier()
dt_classifier.fit(X_train, y_train)

# Make predictions on the test set
dt_predictions = dt_classifier.predict(X_test)

# Create and train a Support Vector Machine (SVM) Classifier
svm_classifier = SVC()
svm_classifier.fit(X_train, y_train)

# Make predictions on the test set
svm_predictions = svm_classifier.predict(X_test)
```

5. Evaluating Classifiers: Decision Tree and SVM

```
# Calculate accuracy and generate a classification report
dt_accuracy = accuracy_score(y_test, dt_predictions)
dt_report = classification_report(y_test, dt_predictions)

print("Decision Tree Classifier:")
print(f"Accuracy: {dt_accuracy}")
print("Classification Report:")
print(dt_report)

# Calculate accuracy and generate a classification report
svm_accuracy = accuracy_score(y_test, svm_predictions)
svm_report = classification_report(y_test, svm_predictions)

print("\nSupport Vector Machine (SVM) Classifier:")
print(f"Accuracy: {svm_accuracy}")
print("Classification Report:")
print(svm_report)
```

Building Unsupervised Machine Learning Classification Model

Applying K-Means Clustering for Classification:

```
from sklearn.cluster import KMeans

# Create a K-means clustering model with 3 clusters (as there are 3 species in
the Iris dataset)
```

```
kmeans = KMeans(n_clusters=3, random_state=42)
kmeans.fit(X)
# Get the cluster centers and labels
cluster_centers = kmeans.cluster_centers_
labels = kmeans.labels_


# Visualize the clusters
plt.figure(figsize=(8, 6))
plt.scatter(X[:, 0], X[:, 1], c=labels, s=50, cmap='viridis')
plt.scatter(cluster_centers[:, 0], cluster_centers[:, 1], s=200, c='red',
marker='X')
plt.title("K-means Clustering")
plt.xlabel(iris.feature_names[0])
plt.ylabel(iris.feature_names[1])
plt.show()


# Evaluating results
inertia = kmeans.inertia_
print(f"Inertia: {inertia}")
```

Lab Task(s):

1. Explore K-NN, Random Forest, and Naive Bayes classifiers in Python and use them for the classification of the Iris dataset used in the tutorial.
2. Take any multiclass dataset and apply different machine learning classifiers to train and test the model. Compare the performance of all the classifiers on the given dataset and present your analysis in a 1-2 page report.
3. Apply K-Means clustering on any classification dataset and evaluate the clustering performance.
