Elements of AIML Lab

NAME: Chaitanaya Sharma

SAP: 500122763

ROLL NO: R2142230335

BATCH: 11

LAB-5

Topic: K-fold cross validation

Experiment Question

How does K-Fold Cross-Validation influence the accuracy of various machine learning classification algorithms (Logistic Regression, Decision Tree, Support Vector Machine, K-Nearest Neighbors, and Linear Discriminant Analysis) when applied to the Pima Indians Diabetes Dataset, Wine Quality Dataset, and Breast Cancer Wisconsin Dataset?

Introduction

In the field of machine learning, evaluating the performance of models is crucial for ensuring their reliability and accuracy. One widely used technique for this purpose is K-Fold Cross-Validation, which helps mitigate the risk of overfitting by dividing the dataset into K subsets, or "folds." This method enables models to be trained on different portions of the data while being tested on unseen data, providing a robust estimate of their performance.

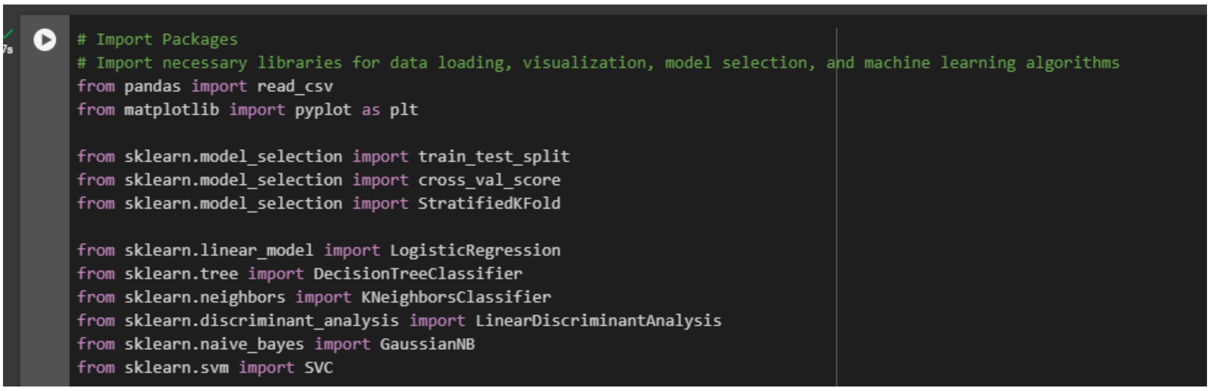
In this report, w will apply K-Fold Cross-Validation to three distinct datasets: the Pima Indians Diabetes Dataset, the Wine Quality Dataset, and the Breast Cancer Wisconsin Dataset. Each dataset presents unique challenges and characteristics that make them suitable for testing various machine learning classification methods. We will evaluate five classification algorithms—Logistic Regression, Decision Tree, Support Vector Machine (SVM), K-Nearest Neighbors (KNN), and Linear Discriminant Analysis (LDA)—to determine which method yields the highest accuracy in predicting outcomes. By analyzing these datasets, we aim to gain insights into the effectiveness of different classification techniques and their applicability in real-world scenarios.

Pima Indians Diabetes Dataset (Pregnant Women) For Predictive Mod Of Diabetes

STEPS OF THE CODE:

1. Import Packages

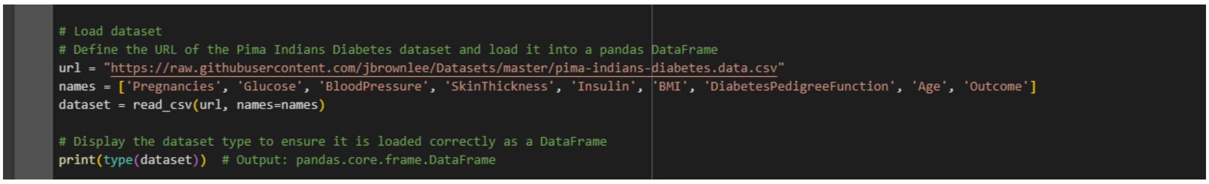
This section imports the necessary libraries required for the experiment. **Pandas** is used for data manipulation, **Matplotlib** for data visualization, and **Sklearn** for applying machine learning algorithms. These libraries provide the essential tools for data preprocessing, model evaluation, and visualization.



```
# Import Packages
# Import necessary libraries for data loading, visualization, model selection, and machine learning algorithms
from pandas import read_csv
from matplotlib import pyplot as plt

from sklearn.model_selection import train_test_split
from sklearn.model_selection import cross_val_score
from sklearn.model_selection import StratifiedKFold

from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.discriminant_analysis import LinearDiscriminantAnalysis
from sklearn.naive_bayes import GaussianNB
from sklearn.svm import SVC
```

2. Loading the Dataset

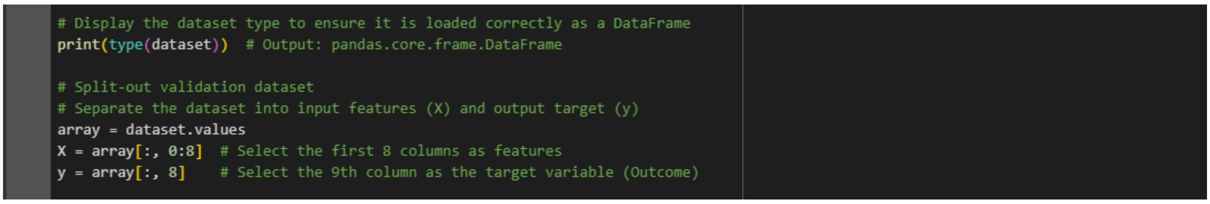
In this step, the **Pima Indians Diabetes dataset** is loaded into a DataFrame using **Pandas**. This dataset includes medical data of women and aims to predict diabetes based on various factors. After loading, the type and structure of the dataset are confirmed to ensure correctness before proceeding with further analysis.



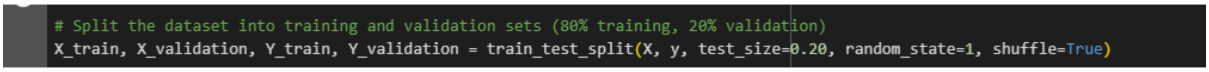
```
# Load dataset
# Define the URL of the Pima Indians Diabetes dataset and load it into a pandas DataFrame
url = "https://raw.githubusercontent.com/jbrownlee/Datasets/master/pima-indians-diabetes.data.csv"
names = ['Pregnancies', 'Glucose', 'BloodPressure', 'SkinThickness', 'Insulin', 'BMI', 'DiabetesPedigreeFunction', 'Age', 'Outcome']
dataset = read_csv(url, names=names)

# Display the dataset type to ensure it is loaded correctly as a DataFrame
print(type(dataset))  # Output: pandas.core.frame.DataFrame
```

3. Data Splitting

The dataset is divided into **features (X)** and **target (y)**, where the features are used for prediction and the target variable indicates the outcome (diabetes presence or absence). Using **train_test_split**, the data is split into training and validation sets to evaluate the models' performance accurately.



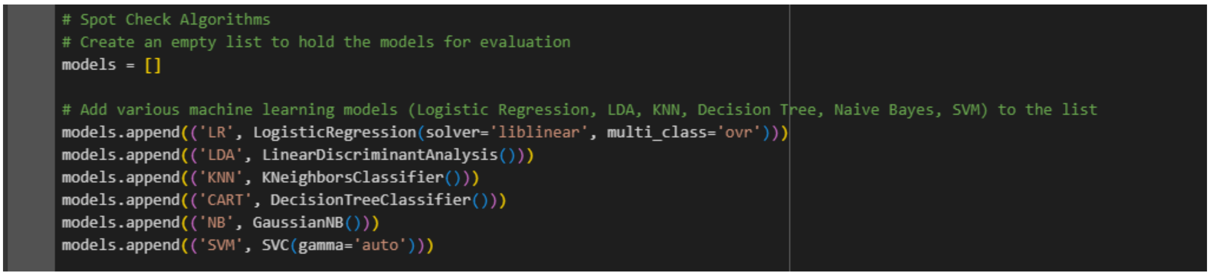
```
# Display the dataset type to ensure it is loaded correctly as a DataFrame
print(type(dataset))  # Output: pandas.core.frame.DataFrame

# Split-out validation dataset
# Separate the dataset into input features (X) and output target (y)
array = dataset.values
X = array[:, 0:8]  # Select the first 8 columns as features
y = array[:, 8]    # Select the 9th column as the target variable (Outcome)
```



```
# Split the dataset into training and validation sets (80% training, 20% validation)
X_train, X_validation, Y_train, Y_validation = train_test_split(X, y, test_size=0.20, random_state=1, shuffle=True)
```

4. Model Definition

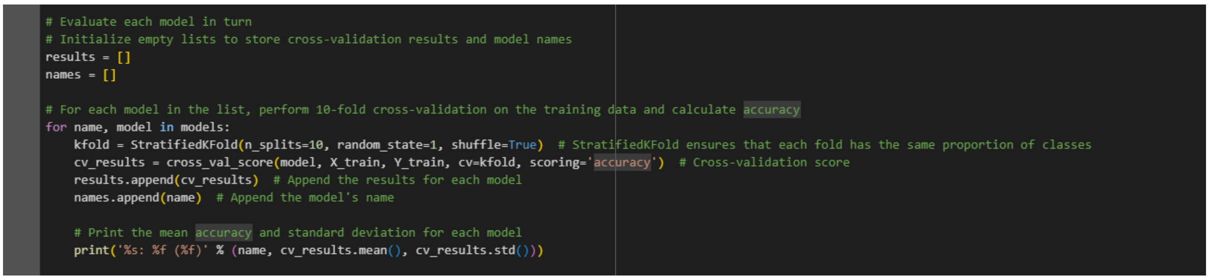
Multiple machine learning models are selected for comparison: **Logistic Regression, Linear Discriminant Analysis, K-Nearest Neighbors, Decision Tree, Naive Bayes,** and **Support Vector Machine (SVM)**. These models represent various algorithms used in classification tasks and will be compared based on their accuracy.’



```
# Spot Check Algorithms
# Create an empty list to hold the models for evaluation
models = []

# Add various machine learning models (Logistic Regression, LDA, KNN, Decision Tree, Naive Bayes, SVM) to the list
models.append(('LR', LogisticRegression(solver='liblinear', multi_class='ovr')))
models.append(('LDA', LinearDiscriminantAnalysis()))
models.append(('KNN', KNeighborsClassifier()))
models.append(('CART', DecisionTreeClassifier()))
models.append(('NB', GaussianNB()))
models.append(('SVM', SVC(gamma='auto')))
```

5. Cross-Validation

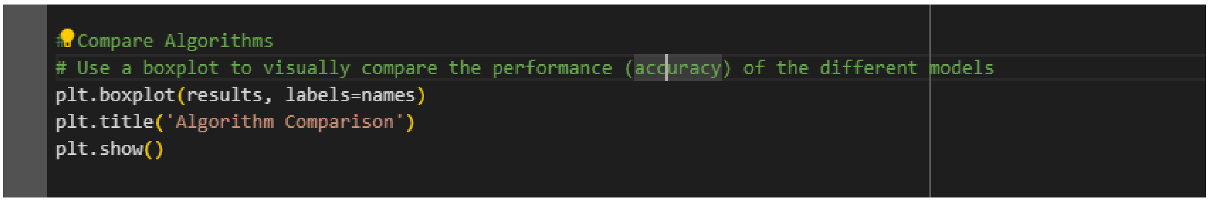
Each machine learning model is evaluated using **Stratified K-Fold Cross-Validation** to ensure robust performance testing. This technique divides the training data into 10 folds, allowing each model to be trained and tested on different parts of the dataset. The accuracy scores for each model are calculated and recorded.



```
# Evaluate each model in turn
# Initialize empty lists to store cross-validation results and model names
results = []
names = []

# For each model in the list, perform 10-fold cross-validation on the training data and calculate accuracy
for name, model in models:
    kfold = StratifiedKFold(n_splits=10, random_state=1, shuffle=True)  # StratifiedKFold ensures that each fold has the same proportion of classes
    cv_results = cross_val_score(model, X_train, Y_train, cv=kfold, scoring='accuracy')  # Cross-validation score
    results.append(cv_results)  # Append the results for each model
    names.append(name)  # Append the model's name

    # Print the mean accuracy and standard deviation for each model
    print('%s: %f (%f)' % (name, cv_results.mean(), cv_results.std()))
```

6. Visualization

To visualize the performance of each model, the accuracy scores from cross-validation are plotted using a **boxplot**. This allows for a quick comparison of the models' accuracy distribution, highlighting the best-performing models and the variance in their results.



```
# Compare Algorithms
# Use a boxplot to visually compare the performance (accuracy) of the different models
plt.boxplot(results, labels=names)
plt.title('Algorithm Comparison')
plt.show()
```

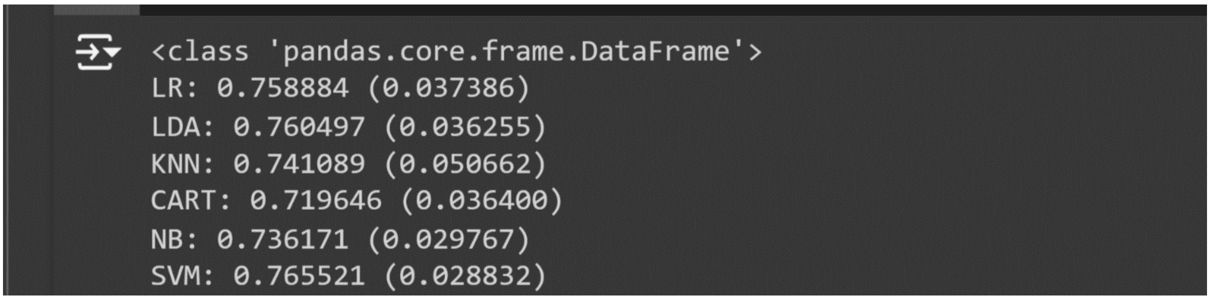
---

7. Conclusion:
Based on the results of the cross-validation, the **Support Vector Classifier (SVC)** emerged as the best-performing model, demonstrating the highest average accuracy across the different folds. This indicates that the SVC model is likely the most effective choice for predicting diabetes in the Pima Indians Diabetes Database.

To further validate the model's predictive capabilities, we will test it on a new patient dataset. The input for this patient includes relevant medical features such as glucose levels, body mass index, and age. The prediction will reveal whether this new patient is likely to develop diabetes (1) or not (0). This step is crucial as it demonstrates the practical application of the machine learning model, providing valuable insights for healthcare professionals in assessing diabetes risk in new patients.

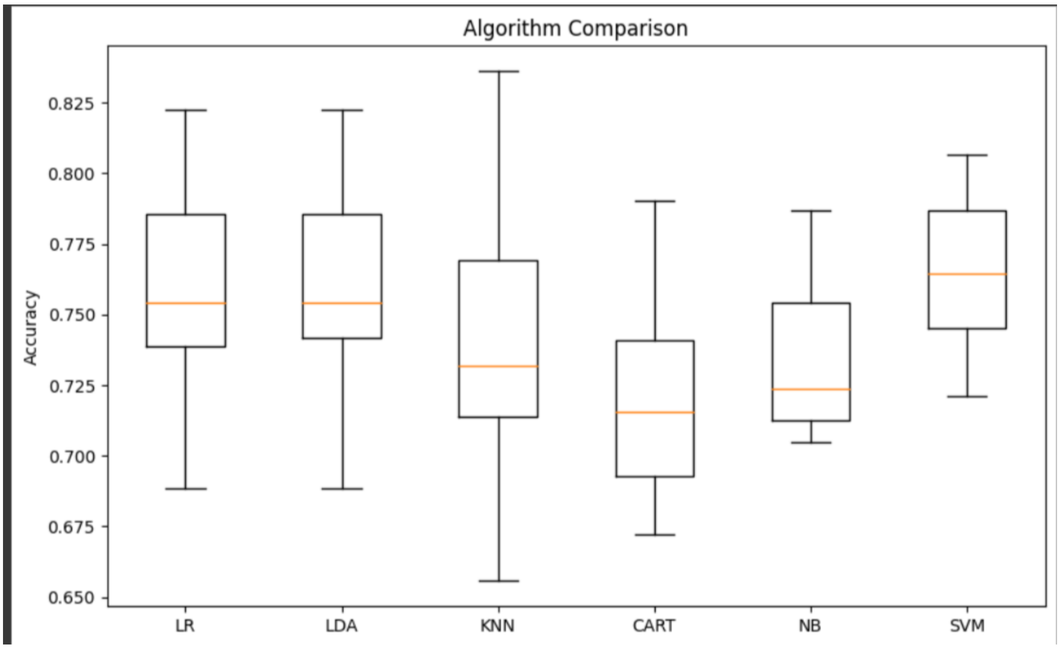
OUTPUT:

1. 5 ML ALGORITHMS:



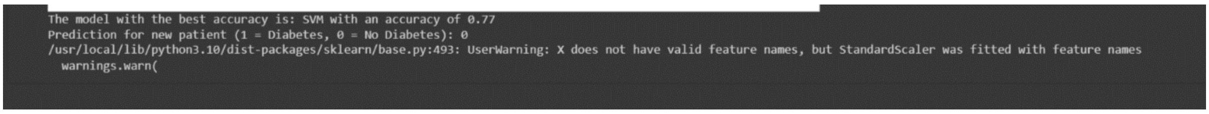
```
<class 'pandas.core.frame.DataFrame'>
LR: 0.758884 (0.037386)
LDA: 0.760497 (0.036255)
KNN: 0.741089 (0.050662)
CART: 0.719646 (0.036400)
NB: 0.736171 (0.029767)
SVM: 0.765521 (0.028832)
```

2. COMPARISON:



3. BEST ALGORITHM ACCURACY:

4. NEW DATASET PATIENT PREDICTION:



```
The model with the best accuracy is: SVM with an accuracy of 0.77
Prediction for new patient (1 = Diabetes, 0 = No Diabetes): 0
/usr/local/lib/python3.10/dist-packages/sklearn/base.py:493: UserWarning: X does not have valid feature names, but StandardScaler was fitted with feature names
  warnings.warn(
```