Experiment No: 6

Aim: Perform Classification modelling
a. Choose a classifier for classification problems.
b. Evaluate the performance of the classifier.
Perform Classification using the below 4 classifiers on the same dataset:

1. K-Nearest Neighbors (KNN)
2. Naive Bayes
3. Support Vector Machines (SVMs)
4. Decision Tree

Theory:

1. **Decision tree**: A Decision Tree is a supervised learning algorithm for classification and regression that splits data based on feature values to form a tree-like structure. It uses Gini Index or Entropy to determine splits and can be pruned to prevent overfitting. While easy to interpret and handle both numerical and categorical data, it can be unstable and prone to overfitting.

```
X = df[features]
y = df["diabetes"]
X_clean = X.copy()
X_clean.replace([np.inf, -np.inf], np.nan, inplace=True)
X_clean.dropna(axis=1, how='all', inplace=True)
X_clean.dropna(axis=0, how='all', inplace=True)
X_clean = X_clean.apply(lambda col: col.fillna(col.mean()), axis=0)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X_clean)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
dt_model = DecisionTreeClassifier(random_state=42)
dt_model.fit(X_train, y_train)
y_pred_dt = dt_model.predict(X_test)
accuracy_dt = accuracy_score(y_test, y_pred_dt)
print("\nDecision Tree Classifier Results:")
print(f"Accuracy: {accuracy_dt:.2f}")
print("Classification Report for Decision Tree:")
print(classification_report(y_test, y_pred_dt))
```

```
cm = confusion_matrix(y_test, y_pred_dt)

print("Confusion Matrix:")
print(cm)
```

1. The Decision Tree classifier achieved **95% accuracy**, indicating strong overall performance. However, a closer look at the classification report and confusion matrix reveals an imbalance in predictive capability between the two classes:

- **Class 0 (Non-Diabetic)**: The model performed exceptionally well, with **98% precision** and **97% recall**, meaning most non-diabetic cases were correctly classified.
- **Class 1 (Diabetic)**: The model struggled with diabetic cases, achieving **67% precision** and **70% recall**, meaning **30% of actual diabetic cases were misclassified as non-diabetic**.

2. From the **confusion matrix**, we observe:

- **True Negatives (TN):** 17,801 (correctly classified non-diabetic cases)
- **False Positives (FP):** 495 (misclassified non-diabetic cases as diabetic)
- **False Negatives (FN):** 421 (misclassified diabetic cases as non-diabetic)
- **True Positives (TP):** 1,003 (correctly classified diabetic cases)

```
Decision Tree Classifier Results:
Accuracy: 0.95
Classification Report for Decision Tree:
              precision    recall  f1-score   support

           0       0.98      0.97      0.97     18296
           1       0.67      0.70      0.69      1424

    accuracy                           0.95     19720
   macro avg       0.82      0.84      0.83     19720
weighted avg       0.95      0.95      0.95     19720

Confusion Matrix:
[[17801   495]
 [  421  1003]]
```

3. While the model performs well overall, the lower recall for diabetic cases indicates that a significant number of diabetic patients are not being correctly identified. This could be due to **class imbalance** in the dataset, where non-diabetic cases dominate.

2. **Support Vector Machine (SVM):** A Support Vector Machine (SVM) finds the optimal hyperplane to separate classes, using support vectors to maximize the margin. It employs the kernel trick (Linear, Polynomial, RBF) for non-linearly

separable data. SVMs perform well on high-dimensional data and small datasets but can be computationally expensive and require careful kernel selection.

```
X_clean = X.copy()
X_clean.replace([np.inf, -np.inf], np.nan, inplace=True)
X_clean.dropna(axis=1, how='all', inplace=True)
X_clean.dropna(axis=0, how='all', inplace=True)
X_clean = X_clean.apply(lambda col: col.fillna(col.mean()), axis=0)
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X_clean)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
svm_model = SVC(kernel='linear', random_state=42)
svm_model.fit(X_train, y_train)
y_pred_svm = svm_model.predict(X_test)
accuracy_svm = accuracy_score(y_test, y_pred_svm)
print("\nSupport Vector Machine (SVM) Classifier Results:")
print(f"Accuracy: {accuracy_svm:.2f}")
print("Classification Report for SVM:")
print(classification_report(y_test, y_pred_svm))
```

1. The Support Vector Machine (SVM) classifier achieved **96% accuracy**, indicating strong overall performance. However, a closer look at the classification report and confusion matrix reveals an imbalance in predictive capability between the two classes:

- **Class 0 (Non-Diabetic):** The model performed exceptionally well, with **96% precision and 100% recall**, meaning almost all non-diabetic cases were correctly classified.
- **Class 1 (Diabetic):** The model struggled with diabetic cases, achieving **92% precision and 48% recall**, meaning **52% of actual diabetic cases were misclassified as non-diabetic**.

2. From the confusion matrix, we observe:

- **True Negatives (TN):** 17,801 (correctly classified non-diabetic cases)
- **False Positives (FP):** 495 (misclassified non-diabetic cases as diabetic)
- **False Negatives (FN):** 421 (misclassified diabetic cases as non-diabetic)
- **True Positives (TP):** 1,003 (correctly classified diabetic cases)

```
Support Vector Machine (SVM) Classifier Results:
Accuracy: 0.96
Classification Report for SVM:
              precision    recall  f1-score   support

           0       0.96      1.00      0.98     18296
           1       0.92      0.48      0.64      1424

    accuracy                           0.96     19720
   macro avg       0.94      0.74      0.81     19720
weighted avg       0.96      0.96      0.95     19720

Confusion Matrix:
[[17801   495]
 [  421  1003]]
```

3. While the model performs well overall, the **low recall for diabetic cases** indicates that a significant number of diabetic patients are not being correctly identified. This could be due to class imbalance in the dataset, where non-diabetic cases dominate.

Conclusion: In this experiment, we implemented two classification models—Support Vector Machines (SVM), and Decision Tree—on the diabetes dataset to evaluate their performance.

- Decision Tree achieved high accuracy (95%) but showed a bias towards non-diabetic cases due to class imbalance. It had excellent precision and recall for non-diabetic cases but struggled with diabetic cases.
- Support Vector Machine (SVM) performed slightly better overall (96% accuracy) but had low recall (48%) for diabetic cases, meaning it misclassified many diabetic patients.