VASANTDADA PATIL PRATISHTHAN'S COLLEGE OF ENGINEERING AND VISUAL ARTS
ISO 9001:2015 Certified Institute
Department of Information Technology
NBA Accredited Course (Dated 01/07/2024 to 30/06/2027)

EXPERIMENT - 6

Aim: Classification modeling
a. Choose classifier for classification problem.
b. Evaluate the performance of classifier.

Theory:

1. Introduction to Classification

Classification is a type of supervised learning where the goal is to categorize data into predefined labels or classes. It involves training a model on labeled data so that it can make predictions on new, unseen data. In this experiment, we use the **Random Forest Classifier** to predict hair fall based on different features.

2. Dataset and Preprocessing

Feature Selection:

- The dataset consists of various attributes related to hair fall.
- The target variable (hair_fall) is a categorical feature indicating whether hair fall occurs.
- The independent variables (X) consist of different factors that may contribute to hair fall.

Data Splitting:

- The dataset is split into **training (80%)** and **testing (20%)** sets using train_test_split().
- The training set is used to train the model, while the testing set evaluates its performance.

Feature Scaling:

- Since features may have different units and magnitudes, we apply **Standardization** using StandardScaler().
- Standardization transforms the features to have a **mean of 0** and **standard deviation of 1**, improving the model's performance.

3. Random Forest Classifier

Random Forest is an **ensemble learning method** that constructs multiple decision trees and combines their outputs for more accurate predictions.

Key Advantages of Random Forest:

- Handles missing values and noise well.
- Reduces overfitting compared to a single decision tree.
- Can handle both classification and regression tasks.

Working of Random Forest:

1. Multiple decision trees are trained on different random subsets of the training data.
2. Each tree makes a prediction, and the majority vote is taken for classification.
3. The final prediction is based on the combined outputs of all trees.

4. Model Evaluation

Accuracy Score:

- The **accuracy score** is calculated using accuracy_score(y_test, y_pred).
- It measures how many predictions were correct compared to the total number of samples.

Classification Report:

- Provides metrics like **Precision, Recall, and F1-score** for each class.
- classification_report(y_test, y_pred) generates the report.

Program:

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.metrics import accuracy_score, confusion_matrix
from sklearn.metrics import classification_report
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt
import seaborn as sns

# Load the dataset
df = pd.read_csv("hair_loss.csv")

# Split the data into training and testing sets
X = df.drop('hair_fall', axis=1)  # Features (all columns except hair_fall)
y = df['hair_fall']  # Target variable

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
scaler = StandardScaler()
X_train_scaled = scaler.fit_transform(X_train)
X_test_scaled = scaler.transform(X_test)

# Classification model, random forest
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
```

```
RandomForestClassifier
RandomForestClassifier(random_state=42)
```

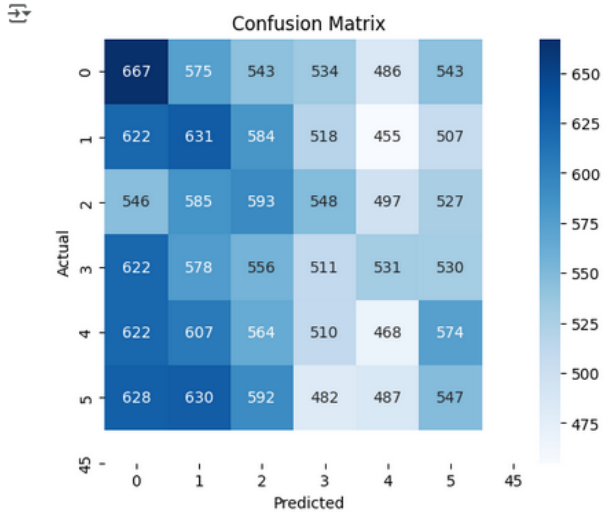
```
[12] # Evaluate the model
     y_pred = model.predict(X_test)
     accuracy = accuracy_score(y_test, y_pred)
     print("Accuracy:", accuracy)
     print(classification_report(y_test, y_pred))
```

```
Accuracy: 0.17085
              precision    recall  f1-score   support

           0       0.18      0.20      0.19      3348
           1       0.17      0.19      0.18      3317
           2       0.17      0.18      0.18      3296
           3       0.16      0.15      0.16      3328
           4       0.16      0.14      0.15      3345
           5       0.17      0.16      0.17      3366

    accuracy                           0.17     20000
   macro avg       0.17      0.17      0.17     20000
weighted avg       0.17      0.17      0.17     20000
```

```
[14] cm = confusion_matrix(y_test, y_pred)
     plt.figure(figsize=(6, 5))
     sns.heatmap(cm, annot=True, fmt='d', cmap="Blues", xticklabels=model.classes_, yticklabels=model.classes_)
     plt.xlabel("Predicted")
     plt.ylabel("Actual")
     plt.title("Confusion Matrix")
     plt.show()
```



Conclusion: Thus, we have successfully implemented classification modeling using random forest