Assignment 4: Kanishka Verma(21BCE1412)

Importing libraries.

```
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler, OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.metrics import accuracy_score, confusion_matrix, classification_report
```

Loading the dataset.

```
data = pd.read_csv('/content/drive/MyDrive/Data/WA_Fn-UseC_-HR-Employee-Attrition.csv')
```

Data Preprocessing

```
label_encoder = LabelEncoder()
data['Attrition'] = label_encoder.fit_transform(data['Attrition'])
```

```
X = data.drop('Attrition', axis=1)
y = data['Attrition']
```

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

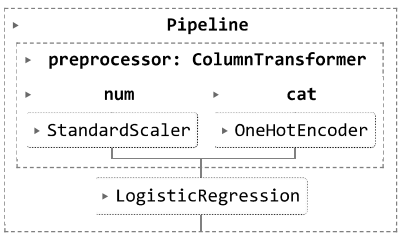
```
numeric_features = X_train.select_dtypes(include=['number']).columns
categorical_features = X_train.select_dtypes(exclude=['number']).columns
```

```
numeric_transformer = Pipeline(steps=[
    ('scaler', StandardScaler())
])

categorical_transformer = Pipeline(steps=[
    ('onehot', OneHotEncoder(handle_unknown='ignore'))
])

preprocessor = ColumnTransformer(
    transformers=[
        ('num', numeric_transformer, numeric_features),
        ('cat', categorical_transformer, categorical_features)
    ])
```

Model Building using Logistic Regression.

```
logistic_model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('classifier', LogisticRegression(random_state=42))
])
```

```
logistic_model.fit(X_train, y_train)
```

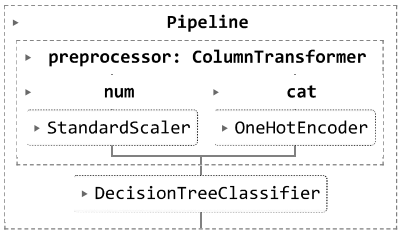


Model Building using Decision Tree

```
tree_model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('classifier', DecisionTreeClassifier(random_state=42))
```

```
])
```

```
tree_model.fit(X_train, y_train)
```



Calculating Performance metrics

For logistic regression.

```
logistic_predictions = logistic_model.predict(X_test)
logistic_accuracy = accuracy_score(y_test, logistic_predictions)
logistic_confusion_matrix = confusion_matrix(y_test, logistic_predictions)
logistic_classification_report = classification_report(y_test, logistic_predictions)
```

For Decision Tree.

```
tree_predictions = tree_model.predict(X_test)
tree_accuracy = accuracy_score(y_test, tree_predictions)
tree_confusion_matrix = confusion_matrix(y_test, tree_predictions)
tree_classification_report = classification_report(y_test, tree_predictions)
```

```
print("Logistic Regression Accuracy:", logistic_accuracy)
print("Logistic Regression Confusion Matrix:\n", logistic_confusion_matrix)
print("Logistic Regression Classification Report:\n", logistic_classification_report)
```

```
Logistic Regression Accuracy: 0.8945578231292517
Logistic Regression Confusion Matrix:
 [[245  10]
 [ 21  18]]
Logistic Regression Classification Report:
               precision    recall  f1-score   support

           0       0.92      0.96      0.94       255
           1       0.64      0.46      0.54        39

    accuracy                           0.89       294
   macro avg       0.78      0.71      0.74       294
weighted avg       0.88      0.89      0.89       294
```

```
print("\nDecision Tree Accuracy:", tree_accuracy)
print("Decision Tree Confusion Matrix:\n", tree_confusion_matrix)
print("Decision Tree Classification Report:\n", tree_classification_report)
```

```
Decision Tree Accuracy: 0.7789115646258503
Decision Tree Confusion Matrix:
 [[220  35]
 [ 30   9]]
Decision Tree Classification Report:
               precision    recall  f1-score   support

           0       0.88      0.86      0.87       255
           1       0.20      0.23      0.22        39

    accuracy                           0.78       294
   macro avg       0.54      0.55      0.54       294
weighted avg       0.79      0.78      0.78       294
```