Assignment_4

September 29, 2023

1 Model building on Employee Attrition Dataset

Anand Misra 21BAI1105

1.1 Importing the Libraries

[82]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder
```

1.2 Importing the Dataset

[83]:
```
data = pd.read_csv('Employee-Attrition.csv')
```

1.3 Checking for Null Values

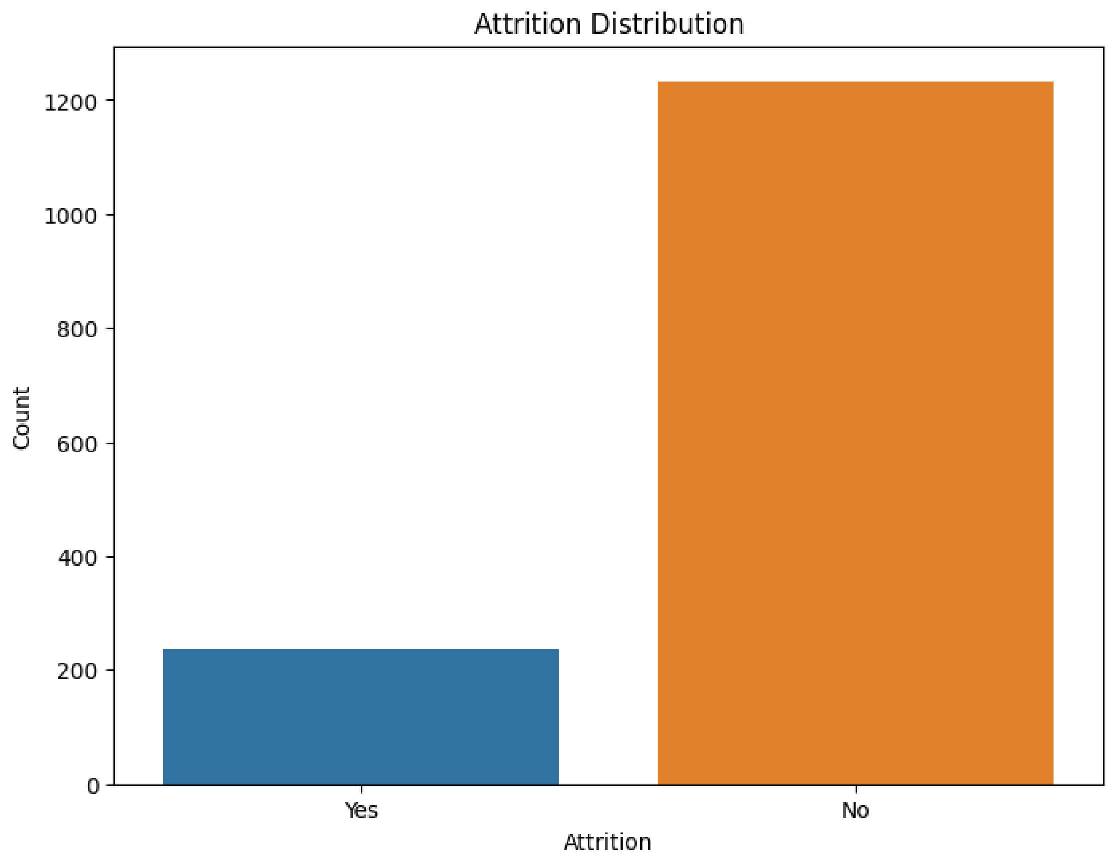
[84]:
```
null_counts = data.isnull().sum()
null_counts
```

[84]:
```
Age                         0
Attrition                   0
BusinessTravel              0
DailyRate                   0
Department                  0
DistanceFromHome            0
Education                   0
EducationField              0
EmployeeCount               0
EmployeeNumber              0
EnvironmentSatisfaction     0
Gender                      0
HourlyRate                  0
JobInvolvement              0
JobLevel                    0
JobRole                     0
```

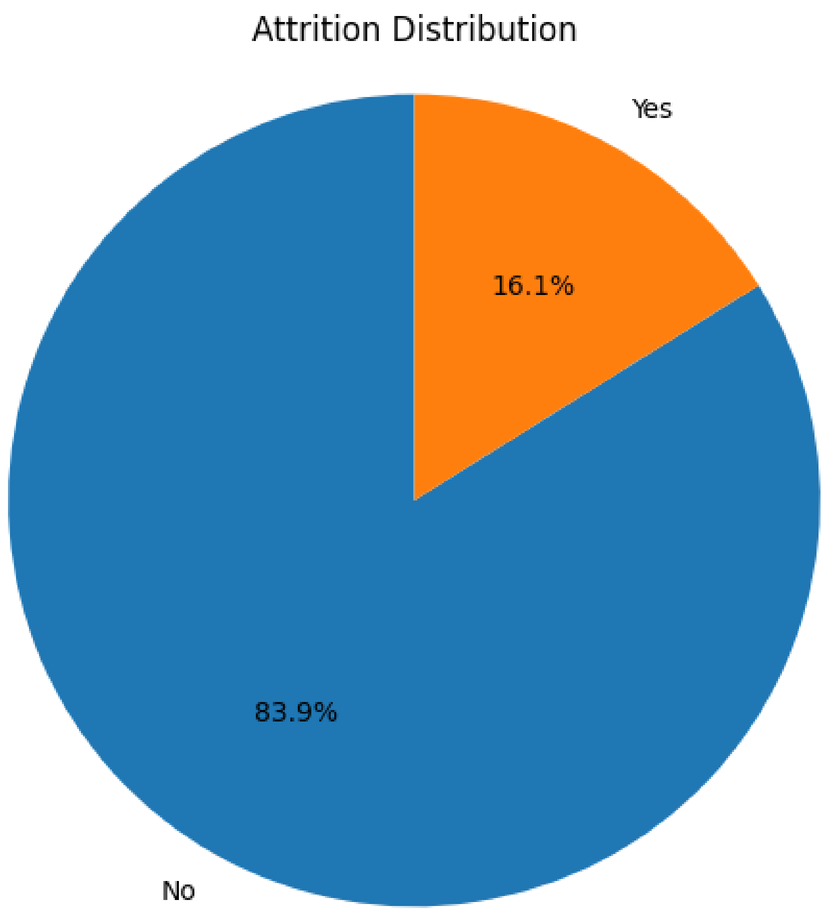
```
JobSatisfaction             0
MaritalStatus               0
MonthlyIncome               0
MonthlyRate                 0
NumCompaniesWorked          0
Over18                      0
OverTime                    0
PercentSalaryHike           0
PerformanceRating           0
RelationshipSatisfaction    0
StandardHours               0
StockOptionLevel            0
TotalWorkingYears           0
TrainingTimesLastYear       0
WorkLifeBalance             0
YearsAtCompany              0
YearsInCurrentRole          0
YearsSinceLastPromotion     0
YearsWithCurrManager        0
dtype: int64
```

1.4 Data Visualization

[85]:
```python
plt.figure(figsize=(8, 6))
sns.countplot(data=data, x='Attrition')
plt.title('Attrition Distribution')
plt.xlabel('Attrition')
plt.ylabel('Count')
plt.show()
```



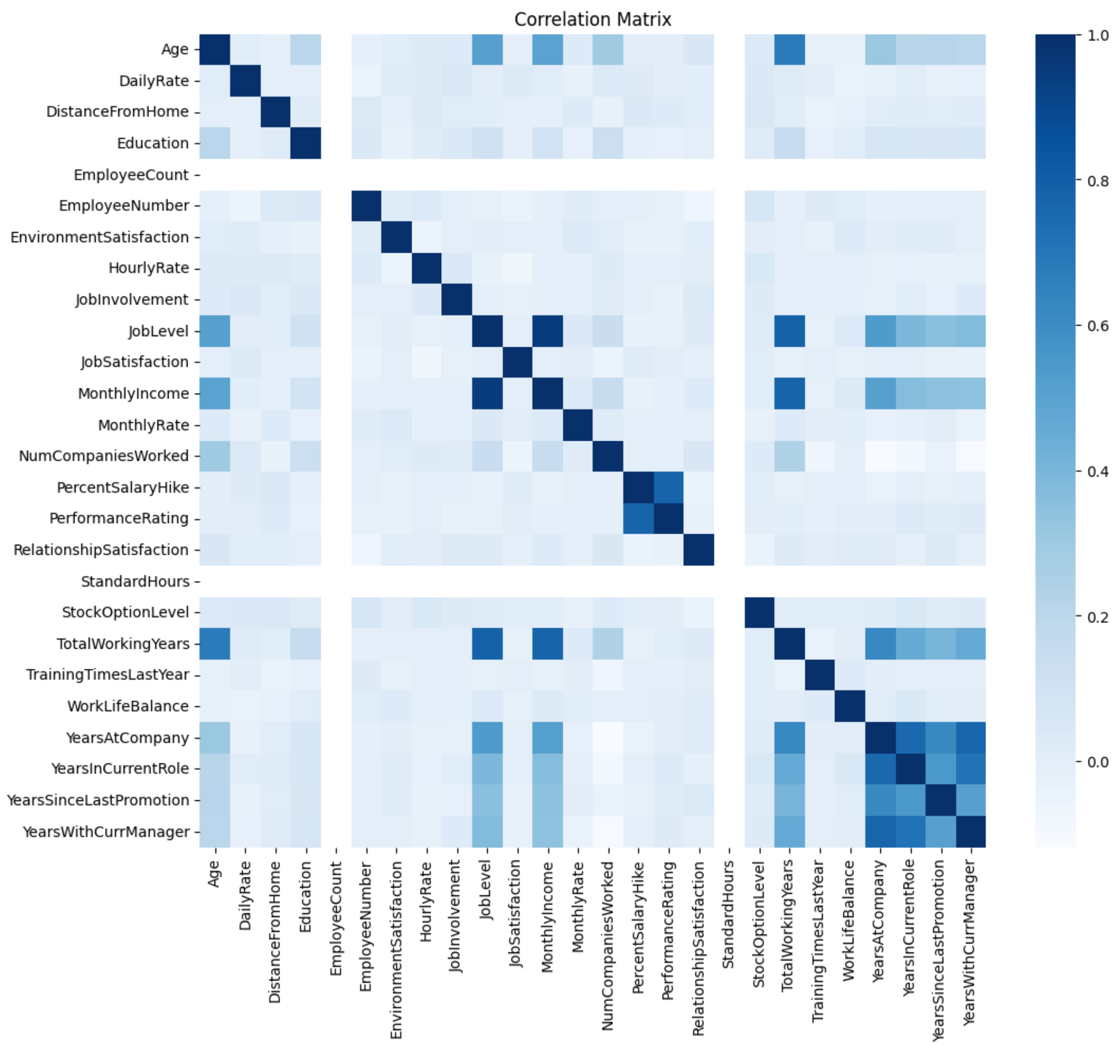
[86]:
```
attrition_counts = data['Attrition'].value_counts()
labels = attrition_counts.index
sizes = attrition_counts.values
plt.figure(figsize=(6, 6))
plt.pie(sizes, labels=labels, autopct='%1.1f%%', startangle=90)
plt.title('Attrition Distribution')
plt.axis('equal')
plt.show()
```



[87]:

```
correlation_matrix = data.corr()
plt.figure(figsize=(12, 10))
sns.heatmap(correlation_matrix, cmap='Blues')
plt.title('Correlation Matrix')
plt.show()
```

```
<ipython-input-87-9b4c1a72ae78>:1: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  correlation_matrix = data.corr()
```



1.5 Removing columns that are not useful or not related

[88]:
```
data = data.drop(['BusinessTravel', 'EmployeeCount', 'EmployeeNumber',␣
↪'Over18', 'StandardHours', 'PerformanceRating'], axis=1)
```

1.6 Encoding the required attributes

[89]:
```
data.head()
```

[89]:
```
   Age Attrition  DailyRate              Department  DistanceFromHome  \
0   41       Yes       1102                   Sales                 1
1   49        No        279  Research & Development                 8
2   37       Yes       1373  Research & Development                 2
3   33        No       1392  Research & Development                 3
```

```
4   27        No        591  Research & Development                 2

   Education EducationField  EnvironmentSatisfaction  Gender  HourlyRate  …  \
0          2  Life Sciences                        2  Female          94  …
1          1  Life Sciences                        3    Male          61  …
2          2          Other                        4    Male          92  …
3          4  Life Sciences                        4  Female          56  …
4          1        Medical                        1    Male          40  …

   PercentSalaryHike  RelationshipSatisfaction StockOptionLevel  \
0                 11                         1                0
1                 23                         4                1
2                 15                         2                0
3                 11                         3                0
4                 12                         4                1

   TotalWorkingYears TrainingTimesLastYear  WorkLifeBalance  YearsAtCompany  \
0                  8                     0                1               6
1                 10                     3                3              10
2                  7                     3                3               0
3                  8                     3                3               8
4                  6                     3                3               2

   YearsInCurrentRole YearsSinceLastPromotion  YearsWithCurrManager
0                   4                       0                     5
1                   7                       1                     7
2                   0                       0                     0
3                   7                       3                     0
4                   2                       2                     2

[5 rows x 29 columns]
```

[90]:
```python
label_encoders = {}
categorical_columns = data.select_dtypes(include=['object']).columns
for col in categorical_columns:
    le = LabelEncoder()
    data[col] = le.fit_transform(data[col])
    label_encoders[col] = le
```

[91]:
```python
data.head()
```

[91]:
```
   Age  Attrition  DailyRate  Department  DistanceFromHome  Education  \
0   41          1       1102           2                 1          2
1   49          0        279           1                 8          1
2   37          1       1373           1                 2          2
3   33          0       1392           1                 3          4
4   27          0        591           1                 2          1
```

```
   EducationField  EnvironmentSatisfaction  Gender  HourlyRate  …  \
0               1                        2       0          94  …
1               1                        3       1          61  …
2               4                        4       1          92  …
3               1                        4       0          56  …
4               3                        1       1          40  …

   PercentSalaryHike  RelationshipSatisfaction  StockOptionLevel  \
0                 11                         1                 0
1                 23                         4                 1
2                 15                         2                 0
3                 11                         3                 0
4                 12                         4                 1

   TotalWorkingYears  TrainingTimesLastYear  WorkLifeBalance  YearsAtCompany  \
0                  8                      0                1               6
1                 10                      3                3              10
2                  7                      3                3               0
3                  8                      3                3               8
4                  6                      3                3               2

   YearsInCurrentRole  YearsSinceLastPromotion  YearsWithCurrManager
0                   4                        0                     5
1                   7                        1                     7
2                   0                        0                     0
3                   7                        3                     0
4                   2                        2                     2

[5 rows x 29 columns]
```

1.7 Splitting Dependent and Independent variables

[92]:
```python
X = data.drop('Attrition', axis=1)
y = data['Attrition']
```

[93]:
```python
X.head()
```

[93]:
```
   Age  DailyRate  Department  DistanceFromHome  Education  EducationField  \
0   41       1102           2                 1          2               1
1   49        279           1                 8          1               1
2   37       1373           1                 2          2               4
3   33       1392           1                 3          4               1
4   27        591           1                 2          1               3

   EnvironmentSatisfaction  Gender  HourlyRate  JobInvolvement  …  \
0                        2       0          94               3  …
```

```
1                        3       1          61                2  …
2                        4       1          92                2  …
3                        4       0          56                3  …
4                        1       1          40                3  …

   PercentSalaryHike  RelationshipSatisfaction  StockOptionLevel  \
0                 11                         1                 0
1                 23                         4                 1
2                 15                         2                 0
3                 11                         3                 0
4                 12                         4                 1

   TotalWorkingYears  TrainingTimesLastYear  WorkLifeBalance  YearsAtCompany  \
0                  8                      0                1               6
1                 10                      3                3              10
2                  7                      3                3               0
3                  8                      3                3               8
4                  6                      3                3               2

   YearsInCurrentRole  YearsSinceLastPromotion  YearsWithCurrManager
0                   4                        0                     5
1                   7                        1                     7
2                   0                        0                     0
3                   7                        3                     0
4                   2                        2                     2

[5 rows x 28 columns]
```

[94]:
```
y.head()
```

[94]:
```
0    1
1    0
2    1
3    0
4    0
Name: Attrition, dtype: int64
```

1.8 Splitting Data into Train and Test

[95]:
```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
↪random_state=42)
```

1.9 Logistic Regression

[96]:
```
from sklearn.linear_model import LogisticRegression

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

```
/usr/local/lib/python3.10/dist-packages/sklearn/linear_model/_logistic.py:458:
ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
```

[96]:
```
LogisticRegression()
```

1.10 Decision Tree

[97]:
```
from sklearn.tree import DecisionTreeClassifier

decision_tree_model = DecisionTreeClassifier(random_state=42)
decision_tree_model.fit(X_train, y_train)
```

[97]:
```
DecisionTreeClassifier(random_state=42)
```

1.11 Performance Metrics

[98]:
```
from sklearn.metrics import accuracy_score, classification_report,␣
↪confusion_matrix
```

1.11.1 Performance of logistic regression

[99]:
```
logistic_predictions = logistic_model.predict(X_test)
logistic_accuracy = accuracy_score(y_test, logistic_predictions)
logistic_classification_report = classification_report(y_test,␣
↪logistic_predictions)
logistic_confusion_matrix = confusion_matrix(y_test, logistic_predictions)
```

[100]:
```
print("Logistic Regression Accuracy:", logistic_accuracy)
print("\nLogistic Regression Classification Report:\n",␣
↪logistic_classification_report)
print("\nLogistic Regression Confusion Matrix:\n", logistic_confusion_matrix)
```

```
Logistic Regression Accuracy: 0.8673469387755102
```

```
Logistic Regression Classification Report:
              precision    recall  f1-score   support

           0       0.87      1.00      0.93       255
           1       0.50      0.03      0.05        39

    accuracy                           0.87       294
   macro avg       0.68      0.51      0.49       294
weighted avg       0.82      0.87      0.81       294


Logistic Regression Confusion Matrix:
 [[254   1]
 [ 38   1]]
```

1.11.2 Performance of decision tree

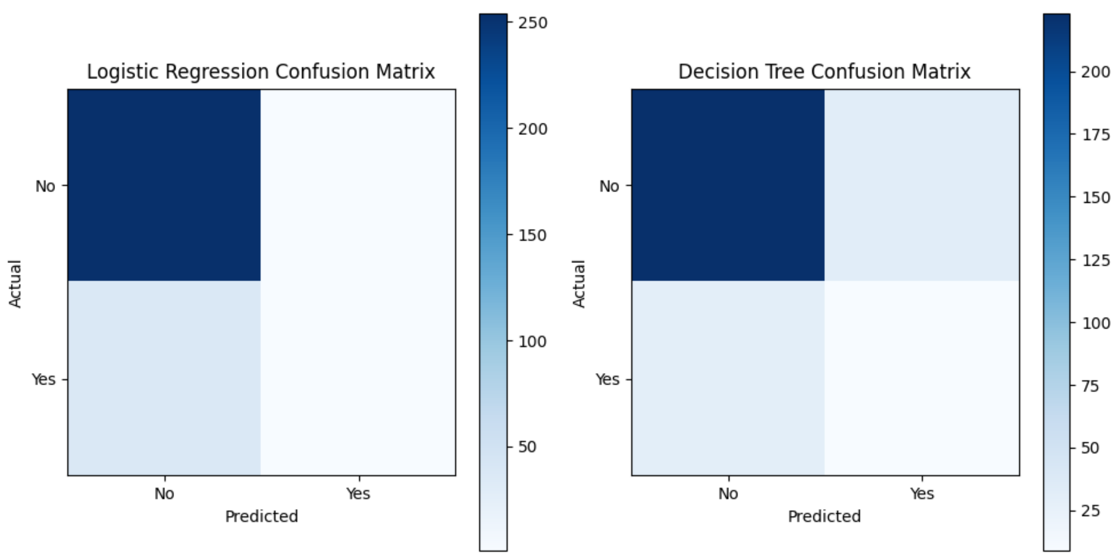
[101]:
```
decision_tree_predictions = decision_tree_model.predict(X_test)
decision_tree_accuracy = accuracy_score(y_test, decision_tree_predictions)
decision_tree_classification_report = classification_report(y_test,␣
 ↪decision_tree_predictions)
decision_tree_confusion_matrix = confusion_matrix(y_test,␣
 ↪decision_tree_predictions)
```

[102]:
```
print("\nDecision Tree Accuracy:", decision_tree_accuracy)
print("\nDecision Tree Classification Report:\n",␣
 ↪decision_tree_classification_report)
print("\nDecision Tree Confusion Matrix:\n", decision_tree_confusion_matrix)
```

```
Decision Tree Accuracy: 0.7891156462585034

Decision Tree Classification Report:
              precision    recall  f1-score   support

           0       0.88      0.87      0.88       255
           1       0.22      0.23      0.23        39

    accuracy                           0.79       294
   macro avg       0.55      0.55      0.55       294
weighted avg       0.79      0.79      0.79       294


Decision Tree Confusion Matrix:
 [[223  32]
 [ 30   9]]
```

1.11.3 Visualization of performance metrics

[103]:
```
plt.figure(figsize=(10, 5))
plt.subplot(1, 2, 1)
plt.title("Logistic Regression Confusion Matrix")
plt.imshow(logistic_confusion_matrix, cmap='Blues', interpolation='nearest')
plt.colorbar()
plt.xticks([0, 1], ['No', 'Yes'])
plt.yticks([0, 1], ['No', 'Yes'])
plt.xlabel('Predicted')
plt.ylabel('Actual')

plt.subplot(1, 2, 2)
plt.title("Decision Tree Confusion Matrix")
plt.imshow(decision_tree_confusion_matrix, cmap='Blues',␣
 ↪interpolation='nearest')
plt.colorbar()
plt.xticks([0, 1], ['No', 'Yes'])
plt.yticks([0, 1], ['No', 'Yes'])
plt.xlabel('Predicted')
plt.ylabel('Actual')

plt.tight_layout()
plt.show()
```



[ ]:
```
```