```
# Importing the necessary libraries
import pandas as pd
import seaborn as sns
import numpy as np
import matplotlib.pyplot as plt
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.model_selection import train_test_split, GridSearchCV
from sklearn.metrics import accuracy_score, classification_report, confusion_matrix
from sklearn.tree import DecisionTreeClassifier
from sklearn import datasets
from sklearn.tree import DecisionTreeRegressor
from sklearn import tree
from sklearn.model_selection import train_test_split
```

Importing of dataset into the system

```
df1=pd.read_csv('HRDataset_v14.csv')
```

```
df1.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 311 entries, 0 to 310
Data columns (total 36 columns):
 #   Column                      Non-Null Count  Dtype
---  ------                      --------------  -----
 0   Employee_Name               311 non-null    object
 1   EmpID                       311 non-null    int64
 2   MarriedID                   311 non-null    int64
 3   MaritalStatusID             311 non-null    int64
 4   GenderID                    311 non-null    int64
 5   EmpStatusID                 311 non-null    int64
 6   DeptID                      311 non-null    int64
 7   PerfScoreID                 311 non-null    int64
 8   FromDiversityJobFairID      311 non-null    int64
 9   Salary                      311 non-null    int64
 10  Termd                       311 non-null    int64
 11  PositionID                  311 non-null    int64
 12  Position                    311 non-null    object
 13  State                       311 non-null    object
 14  Zip                         311 non-null    int64
 15  DOB                         311 non-null    object
 16  Sex                         311 non-null    object
 17  MaritalDesc                 311 non-null    object
 18  CitizenDesc                 311 non-null    object
 19  HispanicLatino              311 non-null    object
 20  RaceDesc                    311 non-null    object
 21  DateofHire                  311 non-null    object
```

```
 22  DateofTermination           104 non-null    object
 23  TermReason                  311 non-null    object
 24  EmploymentStatus            311 non-null    object
 25  Department                  311 non-null    object
 26  ManagerName                 311 non-null    object
 27  ManagerID                   303 non-null    float64
 28  RecruitmentSource           311 non-null    object
 29  PerformanceScore            311 non-null    object
 30  EngagementSurvey            311 non-null    float64
 31  EmpSatisfaction             311 non-null    int64
 32  SpecialProjectsCount        311 non-null    int64
 33  LastPerformanceReview_Date  311 non-null    object
 34  DaysLateLast30              311 non-null    int64
 35  Absences                    311 non-null    int64
dtypes: float64(2), int64(16), object(18)
memory usage: 87.6+ KB
```

```
df1.head()
```

	Employee_Name	EmpID	MarriedID	MaritalStatusID	GenderID
0	Adinolfi, Wilson K	10026	0	0	1
1	Ait Sidi, Karthikeyan	10084	1	1	1
2	Akinkuolie, Sarah	10196	1	1	0
3	Alagbe,Trina	10088	1	1	0
4	Anderson, Carol	10069	0	2	0

```
df1.isnull().sum()
```

```
Employee_Name                0
EmpID                        0
MarriedID                    0
MaritalStatusID              0
GenderID                     0
EmpStatusID                  0
DeptID                       0
PerfScoreID                  0
FromDiversityJobFairID       0
Salary                       0
```

```
Termd                          0
PositionID                     0
Position                       0
State                          0
Zip                            0
DOB                            0
Sex                            0
MaritalDesc                    0
CitizenDesc                    0
HispanicLatino                 0
RaceDesc                       0
DateofHire                     0
DateofTermination            207
TermReason                     0
EmploymentStatus               0
Department                     0
ManagerName                    0
ManagerID                      8
RecruitmentSource              0
PerformanceScore               0
EngagementSurvey               0
EmpSatisfaction                0
SpecialProjectsCount           0
LastPerformanceReview_Date     0
DaysLateLast30                 0
Absences                       0
dtype: int64
```

Dropping the Date columns as they are not relevant to prediction

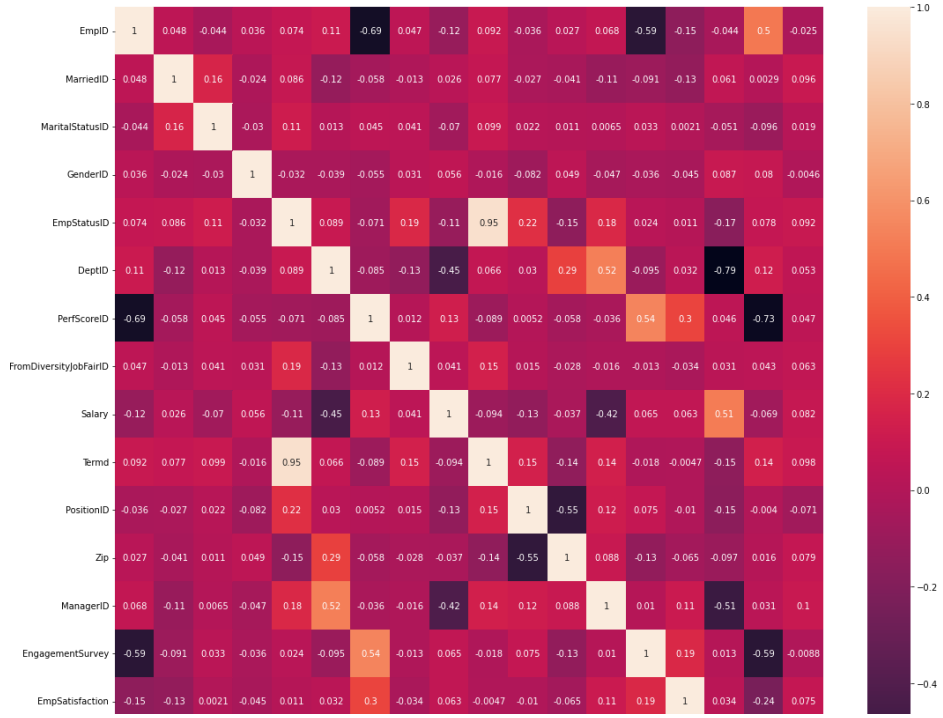
```
df1.drop(['DateofTermination','DateofHire','DOB','LastPerformanceReview_Date'],axis=1,inplace
```

```
df1.head()
```

	Employee_Name	EmpID	MarriedID	MaritalStatusID	GenderID
0	Adinolfi, Wilson K	10026	0	0	1
1	Ait Sidi, Karthikeyan	10084	1	1	1

```
plt.figure(figsize=(18,18))
sns.heatmap(df1.corr(),annot=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd750c21f10>
```

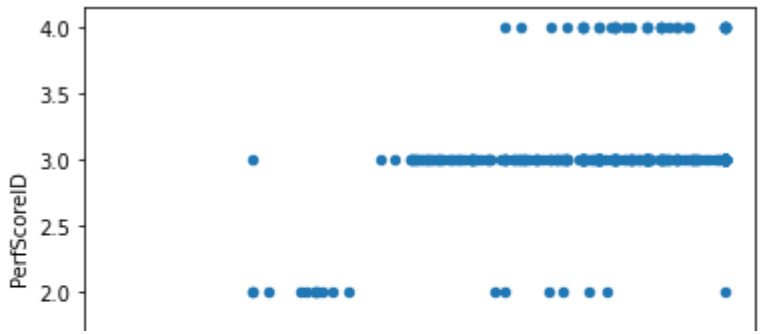



```
plt.figure(figsize=(20,30))
df1.plot(x='Salary',y='PerfScoreID',kind='scatter')
```

```
WARNING:matplotlib.axes._axes:*c* argument looks like a singl
<matplotlib.axes._subplots.AxesSubplot at 0x7fd74ed5ce10>
<Figure size 1440x2160 with 0 Axes>
```


```
df1.plot(x='EngagementSurvey',y='PerfScoreID',kind='scatter')
```

```
WARNING:matplotlib.axes._axes:*c* argument looks like a singl
<matplotlib.axes._subplots.AxesSubplot at 0x7fd750f4fd90>
```




```
df1['ManagerID'] = df1['ManagerID'].replace(np.nan, 39.0)
df1[df1['ManagerName']=='Webster Butler'][['ManagerName','ManagerID']]
```

	ManagerName	ManagerID
4	Webster Butler	39.0

```
df1[['EmpSatisfaction','EmpSatisfaction','SpecialProjectsCount','ManagerID','GenderID','EmpSt
```

```
EmpSatisfaction         0
EmpSatisfaction         0
SpecialProjectsCount    0
ManagerID               0
GenderID                0
EmpStatusID             0
Sex                     0
EmpStatusID             0
SpecialProjectsCount    0
SpecialProjectsCount    0
dtype: int64
```

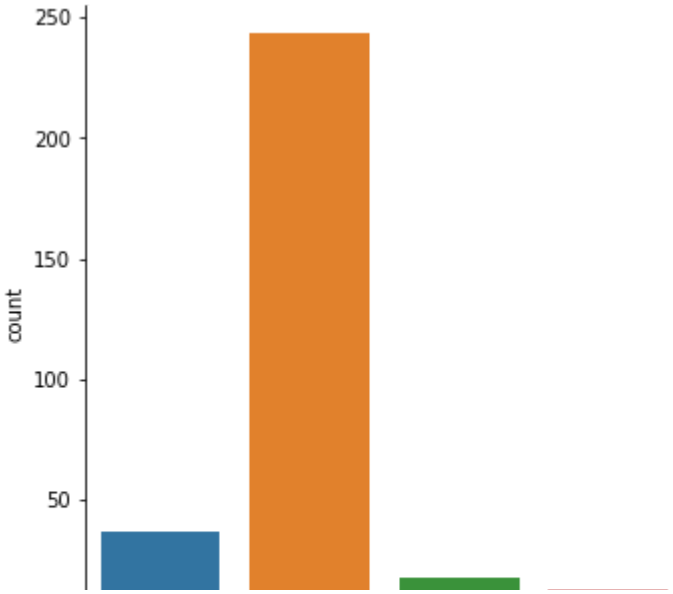
```
df1.corr()
```

	EmpID	MarriedID	MaritalStatusID
MarriedID	0.048058	1.000000	0.164044

Double-click (or enter) to edit

```
color_palette = sns.color_palette("tab10")
sns.set_palette(color_palette)
sns.catplot(x = "PerformanceScore",kind='count', data = df1)
plt.xticks(rotation=45)
```

```
(array([0, 1, 2, 3]), <a list of 4 Text major ticklabel
objects>)
```




```
X = df1[['MaritalStatusID','FromDiversityJobFairID','Salary','SpecialProjectsCount','EmpSatis
y = df1['PerfScoreID']

X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=0)
```

Logistic Regression model

```
from sklearn.base import ClassifierMixin
from sklearn.metrics.cluster import entropy
RandomForestClassifier
regressor = DecisionTreeClassifier(criterion='entropy', random_state=0)
```

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

```
#Running a logistic regression model
# Training the model
from sklearn.linear_model import LogisticRegression
model_logr = LogisticRegression()
model_logr.fit(X_train,y_train)
```

```
LogisticRegression()
```

```
y_predict_log = model_logr.predict(X_test)
```

```
# Finding accuracy, precision, recall and confusion matrix
print(accuracy_score(y_test,y_predict_log))
print(classification_report(y_test,y_predict_log))
```

```
0.8191489361702128
              precision    recall  f1-score   support

           1       0.00      0.00      0.00         3
           2       0.00      0.00      0.00         3
           3       0.82      1.00      0.90        77
           4       0.00      0.00      0.00        11

    accuracy                           0.82        94
   macro avg       0.20      0.25      0.23        94
weighted avg       0.67      0.82      0.74        94

/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: Undefin
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: Undefin
  _warn_prf(average, modifier, msg_start, len(result))
/usr/local/lib/python3.7/dist-packages/sklearn/metrics/_classification.py:1318: Undefin
  _warn_prf(average, modifier, msg_start, len(result))
```

```
confusion_matrix(y_test,y_predict_log)
```

```
array([[ 0,  0,  3,  0],
       [ 0,  0,  3,  0],
       [ 0,  0, 77,  0],
       [ 0,  0, 11,  0]])
```

Finding the confusion matrix for different classes

```
# Finding accuracy, precision, recall and confusion matrix when all classes are considered
print(accuracy_score(y_test,y_pred))
print(classification_report(y_test,y_pred))
```

```
0.6914893617021277
```

```
                   precision    recall  f1-score   support

                1       0.50      0.67      0.57         3
                2       0.12      0.33      0.18         3
                3       0.85      0.79      0.82        77
                4       0.10      0.09      0.10        11

         accuracy                           0.69        94
        macro avg       0.39      0.47      0.42        94
     weighted avg       0.73      0.69      0.71        94
```

```
confusion_matrix(y_test,y_pred)
```

```
array([[ 2,  0,  0,  1],
       [ 0,  1,  1,  1],
       [ 2,  7, 61,  7],
       [ 0,  0, 10,  1]])
```

Decision tree visual representation

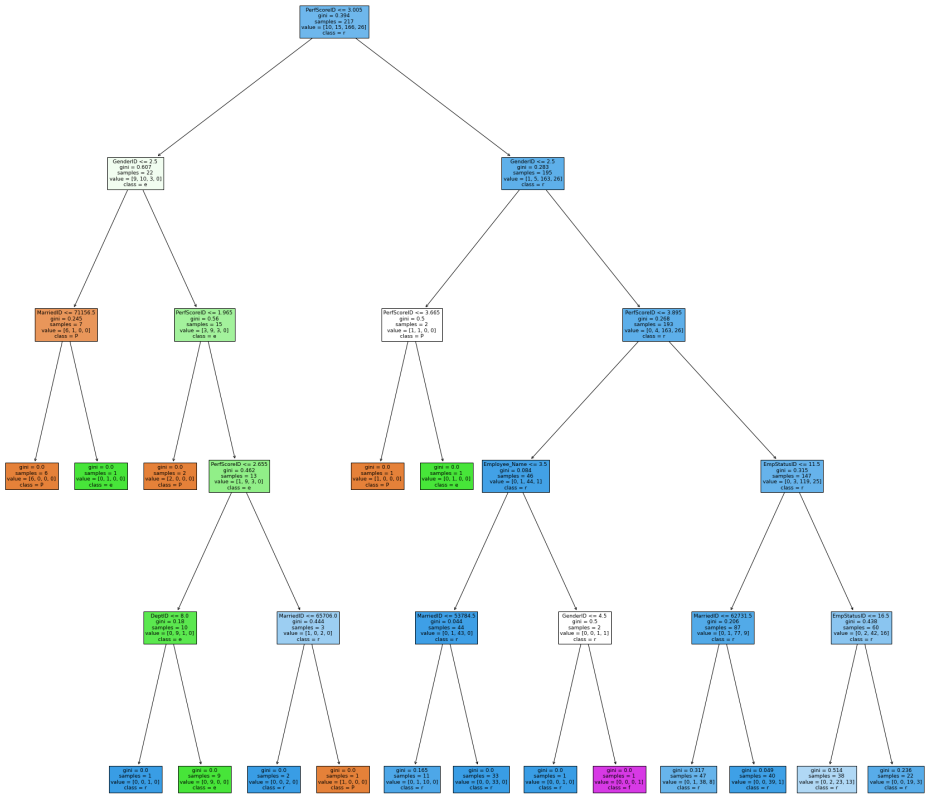
```
clf = DecisionTreeClassifier(max_depth=5, random_state=0)
model = clf.fit(X_train,y_train)
```

```
text_representation = tree.export_text(clf)
print(text_representation)
```

```
|--- feature_7 <= 3.00
|   |--- feature_4 <= 2.50
|   |   |--- feature_2 <= 71156.50
|   |   |   |--- class: 1
|   |   |--- feature_2 >  71156.50
|   |   |   |--- class: 2
|   |--- feature_4 >  2.50
|   |   |--- feature_7 <= 1.96
|   |   |   |--- class: 1
|   |   |--- feature_7 >  1.96
|   |   |   |--- feature_7 <= 2.65
|   |   |   |   |--- feature_6 <= 8.00
|   |   |   |   |   |--- class: 3
|   |   |   |   |--- feature_6 >  8.00
|   |   |   |   |   |--- class: 2
|   |   |   |--- feature_7 >  2.65
|   |   |   |   |--- feature_2 <= 65706.00
|   |   |   |   |   |--- class: 3
|   |   |   |   |--- feature_2 >  65706.00
|   |   |   |   |   |--- class: 1
|--- feature_7 >  3.00
|   |--- feature_4 <= 2.50
|   |   |--- feature_7 <= 3.67
```

```
|   |   |   |--- class: 1
|   |   |--- feature_7 >  3.67
|   |   |   |--- class: 2
|   |--- feature_4 >  2.50
|   |   |--- feature_7 <= 3.90
|   |   |   |--- feature_0 <= 3.50
|   |   |   |   |--- feature_2 <= 53784.50
|   |   |   |   |   |--- class: 3
|   |   |   |   |--- feature_2 >  53784.50
|   |   |   |   |   |--- class: 3
|   |   |   |--- feature_0 >  3.50
|   |   |   |   |--- feature_4 <= 4.50
|   |   |   |   |   |--- class: 3
|   |   |   |   |--- feature_4 >  4.50
|   |   |   |   |   |--- class: 4
|   |   |--- feature_7 >  3.90
|   |   |   |--- feature_5 <= 11.50
|   |   |   |   |--- feature_2 <= 62731.50
|   |   |   |   |   |--- class: 3
|   |   |   |   |--- feature_2 >  62731.50
|   |   |   |   |   |--- class: 3
|   |   |   |--- feature_5 >  11.50
|   |   |   |   |--- feature_5 <= 16.50
|   |   |   |   |   |--- class: 3
|   |   |   |   |--- feature_5 >  16.50
|   |   |   |   |   |--- class: 3
```

```
Dig = plt.figure(figsize=(30,30))
_ = tree.plot_tree(clf,
                   feature_names=df.columns,
                   class_names='PerfScoreID',
                   filled=True)
```


PerfScoreID <= 3.005
gini = 0.394
samples = 217
value = [10, 15, 166, 26]
class = r
GenderID <= 2.5
gini = 0.407
samples = 29
value = [9, 10, 3, 0]
class = e
GenderID <= 2.5
gini = 0.282
samples = 188
value = [1, 5, 163, 26]
class = r
MarriedID <= 721564.5
gini = 0.245
samples = 7
value = [6, 1, 0, 0]
class = P
PerfScoreID <= 1.965
gini = 0.56
samples = 15
value = [3, 9, 3, 0]
class = e
PerfScoreID <= 1.665
gini = 0.5
samples = 2
value = [1, 1, 0, 0]
class = P
PerfScoreID <= 3.935
gini = 0.266
samples = 163
value = [0, 4, 163, 26]
class = r
gini = 0.0
samples = 6
value = [6, 0, 0, 0]
class = P
gini = 0.0
samples = 1
value = [0, 1, 0, 0]
class = e
gini = 0.0
samples = 2
value = [2, 0, 0, 0]
class = P
PerfScoreID <= 2.655
gini = 0.462
samples = 13
value = [1, 8, 3, 0]
class = e
gini = 0.0
samples = 1
value = [1, 0, 0, 0]
class = P
gini = 0.0
samples = 1
value = [0, 1, 0, 0]
class = e
Employee_Name <= 3.5
gini = 0.084
samples = 46
value = [0, 1, 44, 1]
class = r
EmpStatusID <= 11.5
gini = 0.315
samples = 117
value = [0, 3, 119, 25]
class = r
DeptID <= 6.0
gini = 0.153
samples = 12
value = [0, 8, 1, 0]
class = e
MarriedID <= 657706.0
gini = 0.444
samples = 1
value = [1, 0, 2, 0]
class = r
MarriedID <= 521984.5
gini = 0.042
samples = 44
value = [0, 1, 43, 0]
class = r
GenderID <= 4.5
gini = 0.5
samples = 2
value = [0, 0, 1, 1]
class = r
MarriedID <= 627311.5
gini = 0.262
samples = 85
value = [0, 1, 77, 9]
class = r
EmpStatusID <= 16.5
gini = 0.436
samples = 32
value = [0, 2, 42, 16]
class = r
gini = 0.0
samples = 7
value = [0, 0, 1, 0]
class = e
gini = 0.0
samples = 5
value = [0, 5, 0, 0]
class = e
gini = 0.0
samples = 2
value = [0, 0, 2, 0]
class = r
gini = 0.0
samples = 1
value = [1, 0, 0, 0]
class = P
gini = 0.145
samples = 13
value = [0, 1, 12, 0]
class = r
gini = 0.0
samples = 31
value = [0, 0, 31, 0]
class = r
gini = 0.0
samples = 1
value = [0, 0, 1, 0]
class = r
gini = 0.0
samples = 1
value = [0, 0, 0, 1]
class = P
gini = 0.317
samples = 47
value = [0, 1, 38, 8]
class = r
gini = 0.469
samples = 40
value = [0, 0, 39, 1]
class = r
gini = 0.514
samples = 28
value = [0, 2, 23, 13]
class = r
gini = 0.236
samples = 22
value = [0, 0, 19, 3]
class = r

SVM model Trainign

```
print(accuracy_score(y_test,y_pred))
```

```
0.6914893617021277
```