Project Report On

“Human Resource Management: Predicting Employee Promotions Using Machine Learning”

1.INTRODUCTION

1.1 Overview

One of the most delicate topics in any employee's life is promotion. Promotion is the assignment of an employee to a higher-level position in terms of responsibility, authority and pay. When the promotion procedure is applied correctly, the company's success grows as well as the employees' motivation and devotion to the organization. Seniority and qualification are the most important factors in employee advancement. The company's career management success is contingent on establishing a clear and objective promotion policy and applying it fairly. Under what conditions, by whom and how promotions will be made, what qualifications are required for promotion to each position should be determined in advance and presented to all personnel.

1.2 Purpose

The aim is to analyze the various factors that can contribute to the promotion of an employee. Based on the analysis, predict which employees will be promoted.

2.LITERATURE SURVEY

2.1 Existing Problem

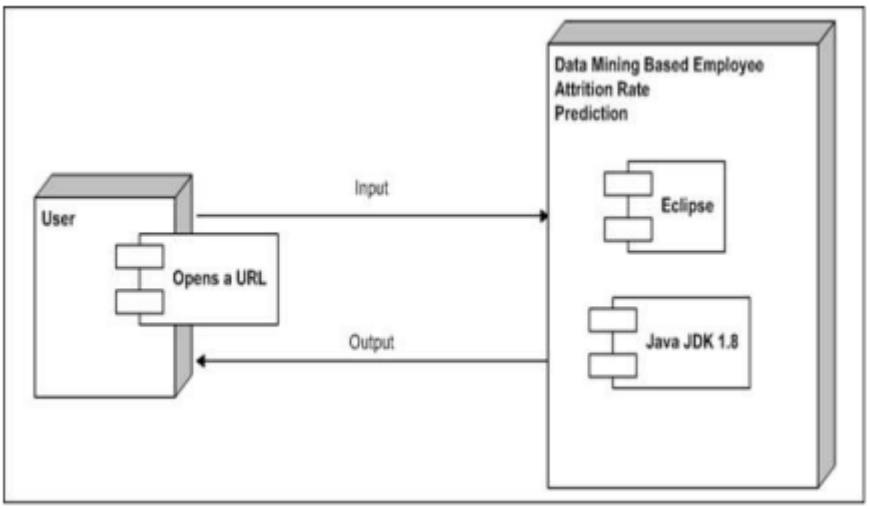
- Deep learning models:
  1. Convolutional Neural Networks (CNN).
  2. Recurrent Neural Networks (RNN).
  3. Boltzmann machine.
  4. Autoencoders etc.
- Classification:
  1. The K-Nearest Neighbours algorithm
  2. Decision Tree
  3. Support Vector Machines
  4. Naive Bayes
- Regression:
  1. Linear Regression
  2. Lasso Regression
  3. Ridge Regression
  4. Support Vector Regression (SVR)
  5. Ensemble Regression
- Clustering:
  1. K means
  2. K means++
  3. K medoids
  4. Agglomerative clustering
  5. DBSCAN

2.2 Proposed Solution

analyse the various factors that can contribute to the promotion of an employee. Based on the analysis, predict which employees will be promoted,using machine learning algorithms.

3. THEORETICAL ANALYSIS

3.1 Block Diagram:



3.2 Hardware/Software Designing

1.Software Requirements

1.Downloading of Anaconda Navigator

2.Downloading of python packages like

a. NumPy Package

b. Pandas

c. librosa

d. Tensor Flow

e. Matplotlib

f. scikit-learn

g. Flask

h. pyhton_speech_features

I. mfcc

j. from python_speech_features import mfcc

k. import sklearn.model_selection

l. from sklearn.model_selection import train_test_split

m. import scipy.io.wavfile as wav

n. import os

o. import pickle

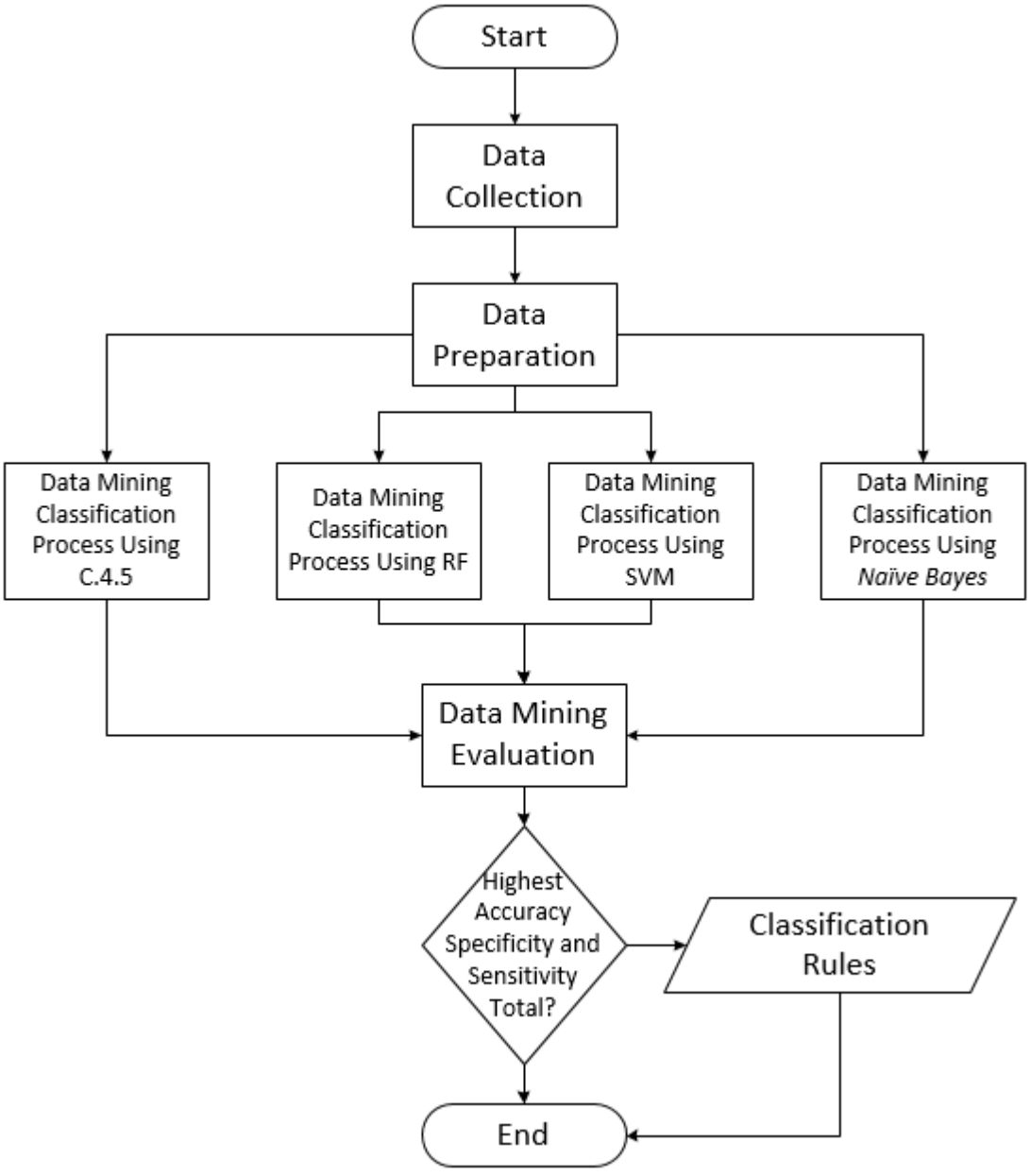
p. import operator

4.EXPERIMENTAL INVESTIGATION

The GTZAN genre collection dataset was collected. It consists of 1000 audio files each having 30 seconds duration. There are 10 classes (10 music genres) each containing 100 audio tracks. Each track is in .wav format. It contains audio files of the following 10 genres:

- Blues
- Classical
- Country
- Disco
- Hip-hop
- Jazz
- Metal
- Pop
- Reggae
- Rock

5.FLOWCHART



6.RESULT

Since Emp_promotion data set consists of different attributes which contains categorical and numerical values, accuracy was used as the main performance metric. Although it can be assumed as a subjective measure in view of an employer, the web application can be used to predict weather an employee will be promoted or not.The best performance in terms of accuracy is observed for the "DeciosionTreeClassifier" model that uses as an input to predict the output with a test accuracy of 89.54. Although performance varies in each classification algorithm.

Fig 1: Web Application view :

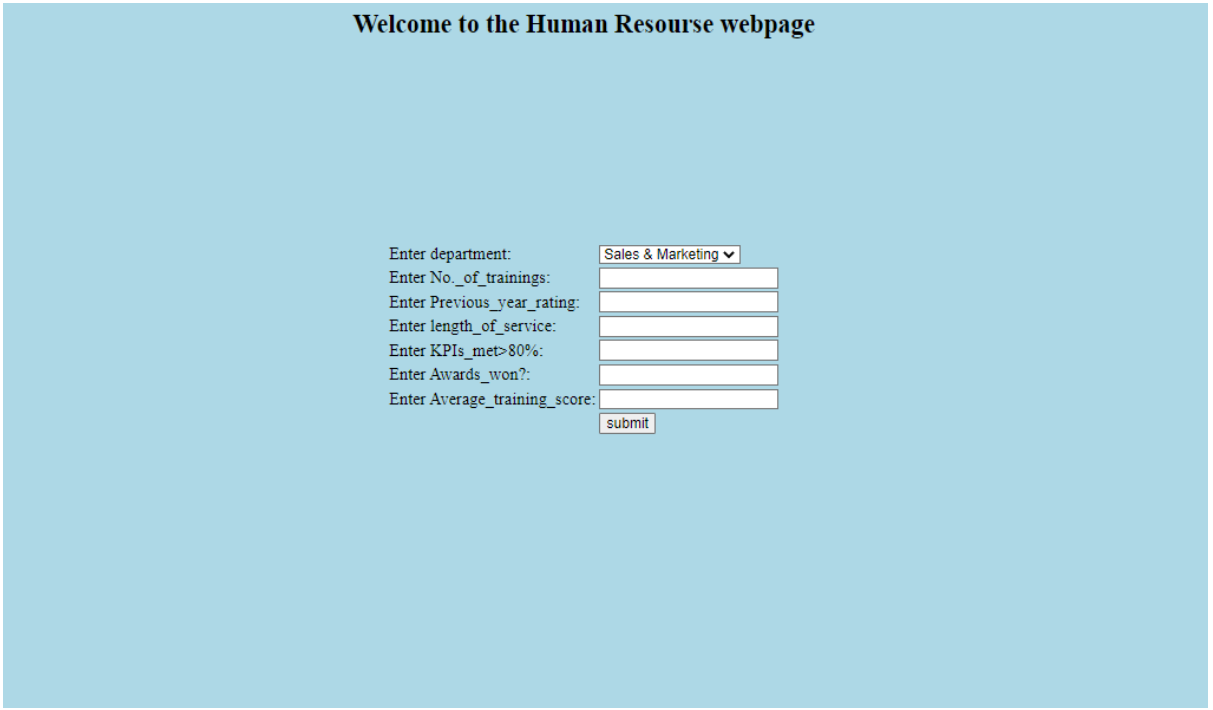
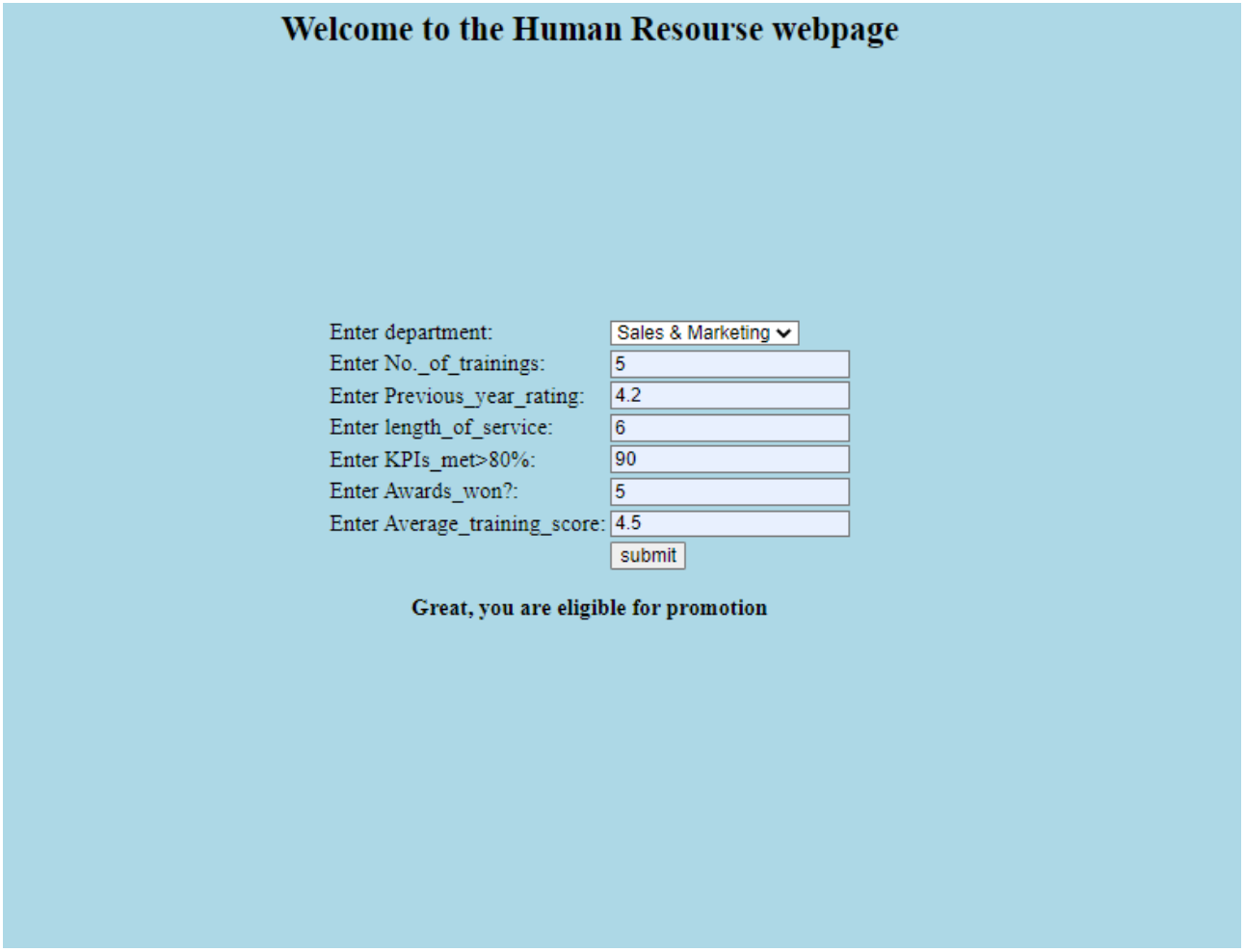


Fig 2: Predicting weather an employee get promoted or not:



7.ADVANTAGES AND DISADVANTAGES

ADVANTAGES

This paper proposes a decision support system designed for a Human Resource (HR) departments about eligibility of employees' promotion. The study's contribution is the using of imbalanced dataset techniques to cope with imbalanced problem. Another contribution of the paper is to focuse on parameter tuning. Employees who may be promoted as a result of this study will be identified, and HR will be able to use this information to improve key performance indicator

DISADVANTAGES

- This method is not able to be implemented in real time since we need to process the information of whole piece of data.
- Distance based on learning is not clear which type of distance to use and which attribute to use to produce the best results.

8.APPLICATIONS

There are many organizatios uses this type of systems to predict the Empoyee Promotion.That ensures the accuracy and prevent from complaints.

9.CONCLUSION

Promotions have a favorable, significant and beneficial impact on employee work performance in human resources process. In this study a prediction model for employee promotion is proposed by using RF method. A decision support system designed for a Human Resource (HR) departments about eligibility of employees' promotion. SMOTE and ROS imbalanced techniques are used. Then, classification algorithms are applied to predict employee promotion such as SVM, ANN and RF. RF outperformed the other algorithms with 98% accuracy, 96% precision, 1.0 recall and 98% f1-score rate obtained among SVM and ANN. This study indicates that F1-score that is the harmonic mean of precision and recall, should be used. The key reason for utilizing F1 Score instead of accuracy is to avoid selecting an inappropriate model in datasets with imbalanced distribution. Furthermore, the F1-score is critical since it is necessary to have a measurement metric that includes not only False Negative or False Positive, but also all mistake costs. This study can be used by HR in the time efficiency of their performance to improve key performance indicator (KPI) KPIs in promoted positions. Besides, it could assist managers in minimising a person's handicap after receiving a promotion due to a mistake made in the selection of a promotion candidate. For the future work, it is planned to add feature engineering and feature importance to the study by using other data balance techniques.

10. FUTURE SCOPE

turnover. A well-designed network with sufficient hidden layers might improve the accuracy, however the scalability and practical implementation aspect has to be studied as well.For future studies, the authors recommend the capture of data around interventions done by the organization for at-risk at employees and its outcome. This will transform the model into a prescriptive one, addressing not just the question "Who is at risk?" but also "What can we do?". It is also recommended to study the application of deep learning models for predicting

11.BIBLIOGRAPHY

1. https://nevonprojects.com/music-genres-classification-using-knn-system/
2.https://www.kaggle.com/code/rxsraghavagrawal/music-genre-classification-using-knn-be gineers/notebook
3. https://github.com/HetGalia/Music-Genre-Classification-using-KNN

APPENDIX:
Source code:

APPENDIX:

Source code: #notebook_codes

In [106]:
```
import numpy as np
import pandas as pd
from matplotlib import pyplot as plt
import seaborn as sns
```

In [107]:
```
data=pd.read_csv("emp_promotion.csv")
data.head()
```

Out[107]:

	employee_id	department	region	education	gender	recruitment_channel	no_of_trainings	age	previous_year_rating	length_of_service	KPIs_met >80%	awards
0	65438	Sales & Marketing	region_7	Master's & above	f	sourcing	1	35	5.0	8	1	
1	65141	Operations	region_22	Bachelor's	m	other	1	30	5.0	4	0	
2	7513	Sales & Marketing	region_19	Bachelor's	m	sourcing	1	34	3.0	7	0	
3	2542	Sales & Marketing	region_23	Bachelor's	m	other	2	39	1.0	10	0	
4	48945	Technology	region_26	Bachelor's	m	other	1	45	3.0	2	0	

In [108]:
```
data.shape
```

Out[108]: (54808, 14)

2. Data preprocessing

Removing unwanted columns

In [111]:
```
data=data.drop(columns=["employee_id","region","recruitment_channel","gender","age","education"])
```

In [112]:
```
data.head()
```

Out[112]:

	department	no_of_trainings	previous_year_rating	length_of_service	KPIs_met >80%	awards_won?	avg_training_score	is_promoted
0	Sales & Marketing	1	5.0	8	1	0	49	0
1	Operations	1	5.0	4	0	0	60	0
2	Sales & Marketing	1	3.0	7	0	0	50	0
3	Sales & Marketing	2	1.0	10	0	0	50	0
4	Technology	1	3.0	2	0	0	73	0

Checking and handling NULL values in the data

In [113]:
```
data.isnull().sum()    #checking NULL values
```

Out[113]:
```
department                0
no_of_trainings           0
previous_year_rating    4124
length_of_service         0
KPIs_met >80%             0
awards_won?               0
avg_training_score        0
is_promoted               0
dtype: int64
```

In [117]:
```
data["previous_year_rating"].fillna(data["previous_year_rating"].mean(),inplace=True)

#replacing missing value in numerical values
```

In [118]:
```
data["previous_year_rating"].head(10)
```

Out[118]:
```
0    5.0
1    5.0
2    3.0
3    1.0
4    3.0
5    3.0
6    3.0
7    3.0
8    4.0
9    5.0
Name: previous_year_rating, dtype: float64
```

In [119]:
```
data.isnull().sum()
```

Out[119]:
```
department              0
no_of_trainings         0
previous_year_rating    0
length_of_service       0
KPIs_met >80%           0
awards_won?             0
avg_training_score      0
is_promoted             0
dtype: int64
```

Finding and removing negative data

In [120]:
```
n=data[(data["KPIs_met >80%"]==0) & (data["awards_won?"]==0) & (data["previous_year_rating"]==1.0) & (data["is_promoted"]==1) &
      (data["avg_training_score"]<60)]

n
```

Out[120]:

	department	no_of_trainings	previous_year_rating	length_of_service	KPIs_met >80%	awards_won?	avg_training_score	is_promoted
31860	Sales & Marketing	1	1.0	2	0	0	58	1
51374	Sales & Marketing	1	1.0	5	0	0	58	1

In [121]:
```
data.drop(index=[31860,51374],inplace=True)
```

Handling outliers

In [122]:
```
q1=np.quantile(data["length_of_service"],0.25)
q3=np.quantile(data["length_of_service"],0.75)
```

In [123]:
```
result=q3-q1

upperBound=(1.5*result)+q3
lowerBound=(1.5*result)-q1
```

In [124]:
```
print("q1:",q1)
print("q3:",q3)
print("result:",result)

print("UpperBound:",upperBound)
print("LowerBound:",lowerBound)
print("Skewed data:",len(data[data['length_of_service']>upperBound]))
```

```
q1: 3.0
q3: 7.0
result: 4.0
UpperBound: 13.0
LowerBound: 3.0
Skewed data: 3489
```

In [125]:
```
pd.crosstab([data['length_of_service']>upperBound],data['is_promoted'])
```

Out[125]:

is_promoted	0	1
length_of_service		
False	46885	4432
True	3255	234

In [101]:
```
data["length_of_service"]=[upperBound if x>upperBound else x for x in data['length_of_service']]
data["length_of_service"]
```

Out[101]:
```
0         8.0
1         4.0
2         7.0
3        10.0
4         2.0
        ...
54803    13.0
54804     6.0
54805     3.0
```

Handling Categorical values

In [126]:
```
from sklearn.preprocessing import LabelEncoder
le=LabelEncoder()

data["department"]=le.fit_transform(data["department"])
```

In [127]:
```
data["department"].unique()
```

Out[127]:
```
array([7, 4, 8, 0, 6, 5, 1, 2, 3])
```

Handling Imbalanced data

In [128]:
```
x=data.drop('is_promoted',axis=1)
y=data['is_promoted']

print(x.shape)
print(y.shape)
```

```
(54806, 7)
(54806,)
```

In [129]:
```
from imblearn.over_sampling import SMOTE
sm=SMOTE()

x_update,y_update=sm.fit_resample(x,y)

print(x_update.shape)
print(y_update.shape)
```

```
(100280, 7)
(100280,)
```

Splitting data into train and test

In [130]:
```
from sklearn.model_selection import train_test_split
x_train,x_test,y_train,y_test=train_test_split(x_update,y_update,test_size=0.2,random_state=0)
```

Model building ¶

Using Logistic regression

In [131]:
```
from sklearn.preprocessing import StandardScaler
ss=StandardScaler()
x_train=ss.fit_transform(x_train)
x_test=ss.transform(x_test)
```

In [132]:
```
from sklearn.tree import DecisionTreeClassifier
dt= DecisionTreeClassifier(criterion="entropy",random_state=0)
dt.fit(x_train,y_train)
```

Out[132]:
```
DecisionTreeClassifier(criterion='entropy', random_state=0)
```

In [133]:
```
dtpred=dt.predict(x_test)
```

In [134]:
```
dtpred
```

Out[134]:
```
array([1, 1, 1, ..., 1, 0, 0], dtype=int64)
```

In [135]:
```
y_test
```

Out[135]:
```
92988    1
77138    1
65836    1
88329    1
85776    1
        ..
91647    1
27994    0
58334    1
23270    0
20055    0
Name: is_promoted, Length: 20056, dtype: int64
```

In [136]:
```
from sklearn.metrics import accuracy_score
accuracy=accuracy_score(dtpred,y_test)

accuracy
```

Out[136]:
```
0.8915037893897089
```

In [137]:
```
new=dt.predict([[0,1,2.0,1,10,1,10]])
new
```

Out[137]:
```
array([1], dtype=int64)
```

Saving the model for Application development

In [138]:
```
import pickle
pickle.dump(dt,open('promotion.pkl','wb'))
```

project.py × main_index.html × main.css ×

```python
from flask import Flask,render_template,request

app=Flask(__name__)

import pickle

model=pickle.load(open("promotion.pkl","rb"))


@app.route('/')
def index():
    return render_template("main_index.html")

@app.route('/datas',methods=["POST"])
def do():
    d=request.form["dept"]

    if d=="Sales & Marketing":
        d=7
    elif d=="Operations":
        d=4
    elif d=="Technology":
        d=8
    elif d=="Analytics":
        d=0
    elif d=="R&D":
        d=6
    elif d=="Procurement":
        d=5
    elif d=="Finance":
        d=1
    elif d=="HR":
        d=2
    elif d=="Legal":
        d=3
```

```python
    num_of_training=request.form["not"]
    pre_yr_rating=request.form["pyr"]
    len_of_service=request.form["los"]
    kpi=request.form["kpi"]
    award=request.form["aw"]
    avg_training_score=request.form["ats"]

    data=[[d,num_of_training,pre_yr_rating,len_of_service,kpi,award,avg_training_score]]

    p=model.predict(data)

    if p == 0:
        text = 'Sorry, you are not eligible for promotion'
    else:
        text = 'Great, you are eligible for promotion'


    return render_template("main_index.html",data=text)

app.run(debug=True)
```

project.py × main_index.html × main.css ×

```html
<html>
<head>
<link rel="stylesheet" href="{{url_for('static',filename='css/main.css')}}">
</head>
<body>
<p> <h2> <center> Welcome to the Human Resourse webpage </center> </h2> </p>
<form method="post" action="/datas">
<center>
 <br> <br> <br> <br> <br> <br> <br> <br>
<table>

<tr>  <td> Enter department: </td>
      <td> <select name="dept">
           <option value="Sales & Marketing"> Sales & Marketing </option>
           <option value="Operations"> Operations </option>
           <option value="Technology"> Technology </option>
           <option value="Analytics"> Analytics </option>
           <option value="R&D"> R&D </option>
           <option value="Procurement"> Procurement </option>
           <option value="Finance"> Finance </option>
           <option value="HR"> HR </option>
           <option value="Legal"> Legal </option>  </td>
                                                   </select> <br> </tr>

<tr>  <td> Enter No._of_trainings: </td>
      <td> <input type="text" name="not" </td>
                                              </tr>

<tr>  <td> Enter Previous_year_rating: </td>
      <td> <input type="text" name="pyr" </td>
                                               </tr>

<tr>  <td> Enter length_of_service: </td>
      <td> <input type="text" name="los" </td>
                                               </tr>

<tr>  <td> Enter KPIs_met>80%: </td>
      <td> <input type="text" name="kpi" </td>
```

```
40
41  <tr>  <td> Enter Awards_won?: </td>
42        <td> <input type="text" name="aw" </td>
43                                                  </tr>
44
45  <tr>  <td> Enter Average_training_score: </td>
46        <td> <input type="text" name="ats" </td>
47                                                  </tr>
48
49  <tr>  <td>  </td>
50        <td> <input type="submit" value="submit" </td>
51                                                  </tr>
52
53  </table>
54  </center>
55  </form>
56
57  <center>
58  <p style="color:black"> <b> {{data}} </b> </p>
59  </center>
60
61  </body>
62  </html>
```

project.py × | main_index.html × | main.css ×

```
1  body {
2        background-color: lightblue;
3        text-align=center;
4  }
5
```