Employee Promotion Prediction Using IBM Watson Studio Machine Learning

1. Introduction:

1.1 Overview:

Promotion or career advancement is a process through which an employee of a company is given a higher share of duties, a higher pay scale, or both. Promotion is not just beneficial for employees but is also highly crucial for the employer or business owners. It boosts the morale of promoted employees, increases their productivity, and hence improves upon the overall profits earned by the organization.

1.2 Purpose:

As global competition intensifies, the competition among various
enterprises become more and more fierce. Employees are the key
part of enterprises and significantly affect enterprises.

But what if you could have a reliable predictor for an employee that could predict if they would get promoted.

Then, you could have a secure and much fulfilling life.

2. Literature Survey

2.1 Existing Problem:

According to others using this dataset, some of the values for the employees are incorrect, meaning that some of our predictions will be off by a large amount, but we shouldn’t always trust the listed value.

It is a very time-consuming task to predict the if a person will get promoted or not by actual testing.

Using a machine learning model, we can do the same in seconds.

2.2 Proposed Solution:

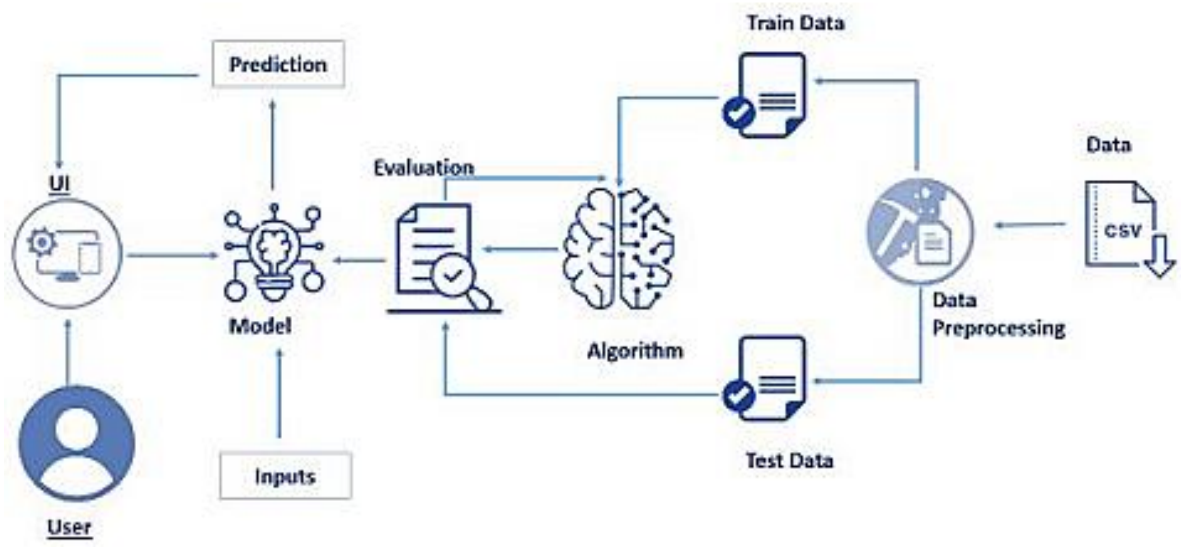
Using advanced machine learning model which is trained using the verified dataset of employees who got promoted and who are not promoted and its various attributer the model can be trained to predict the promotion of an employee provided necessary values given.

The model can predict the performance with an accuracy of 94% given the fact that the result can be seen in seconds the model is reliable.

Anyone with prior knowledge of using a web browser can operate the application easily. Generally, a model is only as good as the data passed into it, and the data preprocessing we do ensure that the model has as accurate a dataset as possible.

3. Theoretical Analysis

3.1 Block Diagram:



3.2 Hardware and Software Designing:

- IBM Watson Studio

Watson Studio accelerates the machine and deep learning workflows required to infuse AI into your business to drive innovation. It provides a suite of tools for data scientists, application developers and subject matter experts, allowing them to collaboratively connect to data, wrangle that data and use it to build, train and deploy models at scale. Successful AI projects require a combination of algorithms + data + team, and a very powerful compute infrastructure.

- IBM Watson Machine Learning
- IBM Cloud Object Storage

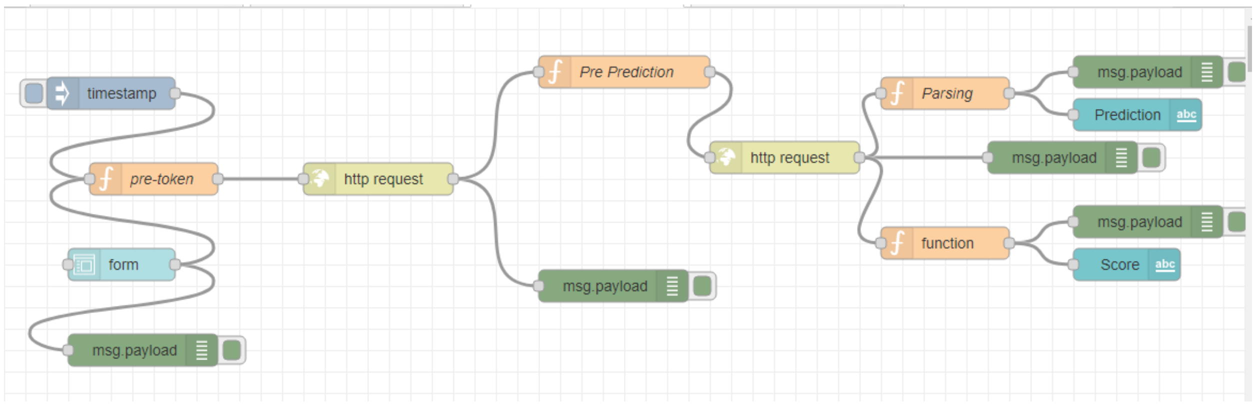
4. Experimental Investigation

In this section, we will be creating and training our model for predicting if a employee could get promoted or not. Since there are multiple algorithms, we can use to build our model, we will compare the accuracy scores after testing and pick the most accurate algorithm.

From this list, we are using DecisionTree, RandomForest, and KNeighborsRegressor to perform our predictions. We then see which algorithm produces the highest accuracy and select it as our algorithm of choice for future use.

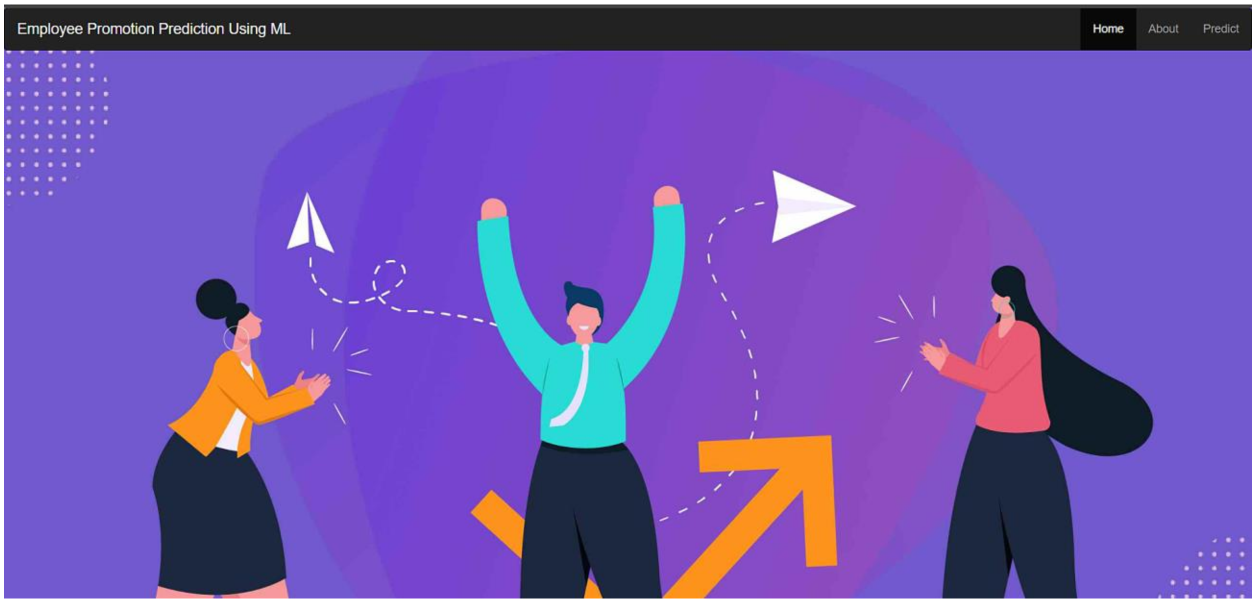
On the results of the following algorithms, we have done the conclusion the Random Forest model is the most accurate out of all the models which we have tested.

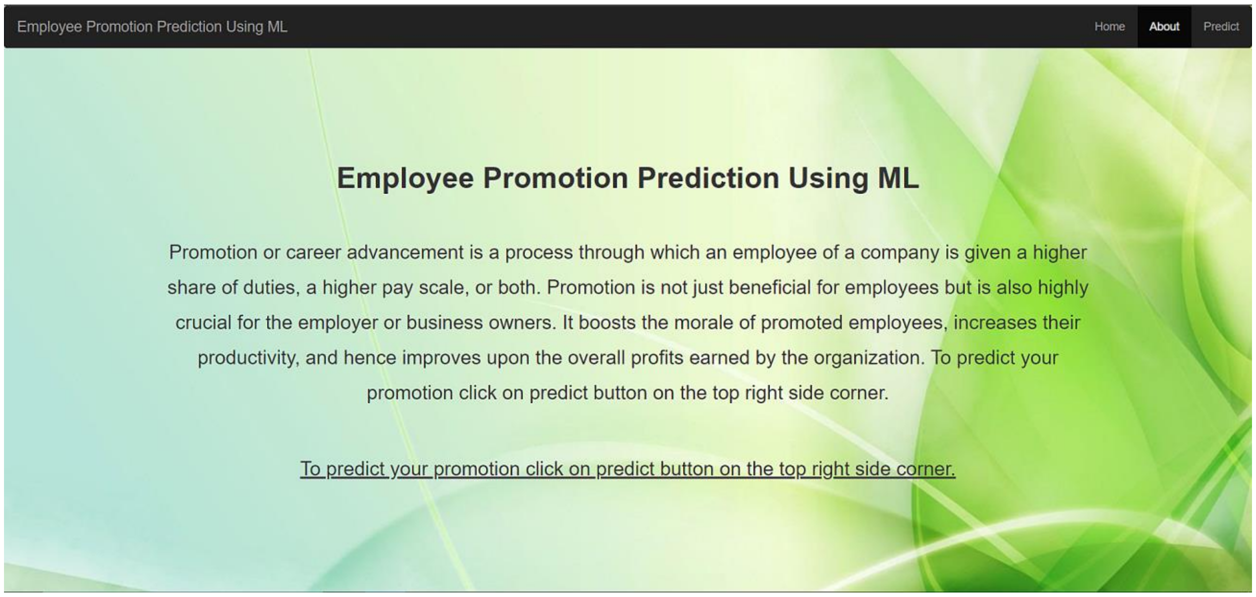
5. Flowchart



6. Result

The final result of the project Employee Promotion prediction . The output would be in this format:

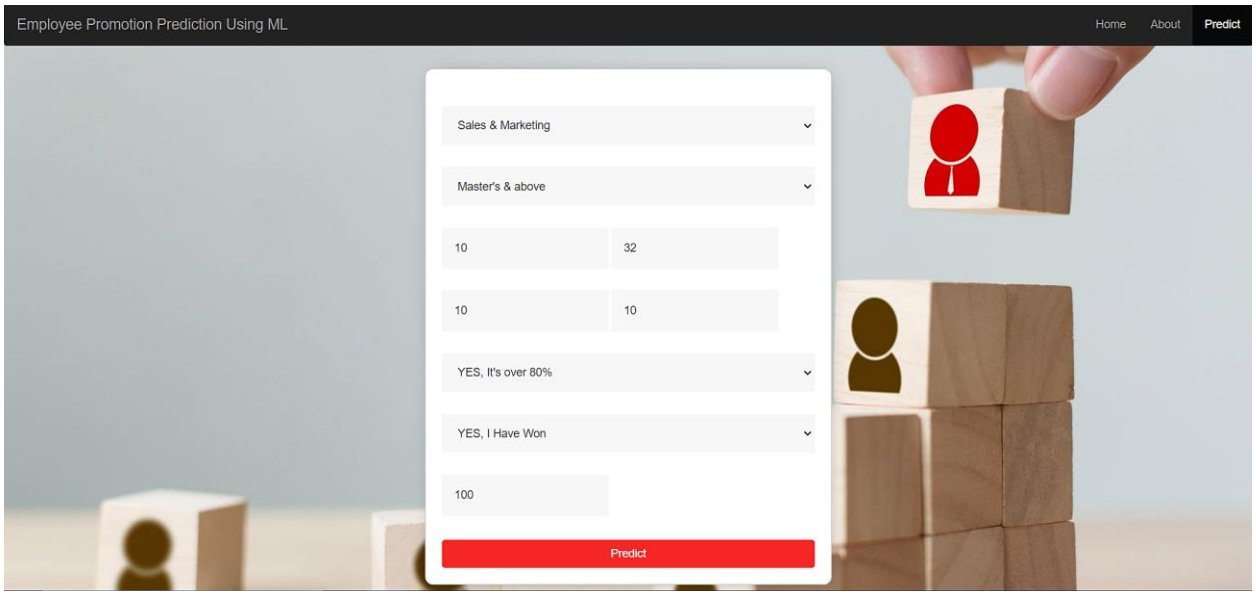




Employee Promotion Prediction Using ML

Promotion or career advancement is a process through which an employee of a company is given a higher share of duties, a higher pay scale, or both. Promotion is not just beneficial for employees but is also highly crucial for the employer or business owners. It boosts the morale of promoted employees, increases their productivity, and hence improves upon the overall profits earned by the organization. To predict your promotion click on predict button on the top right side corner.

To predict your promotion click on predict button on the top right side corner.



Employee Promotion Prediction Using ML

Sales & Marketing

Master's & above

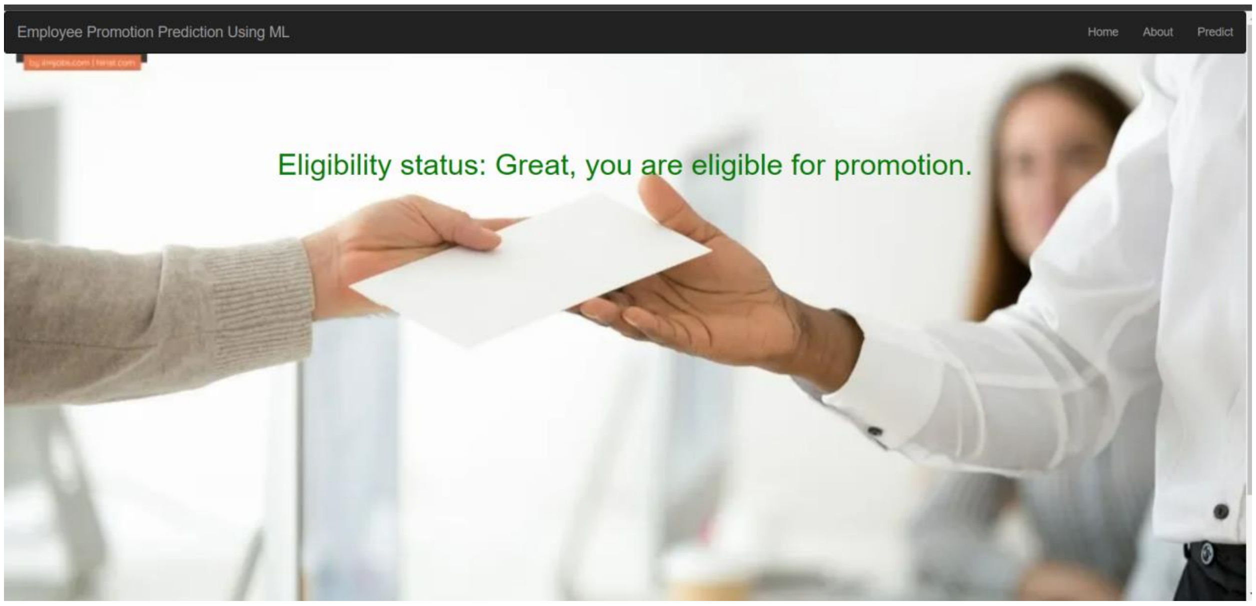
10 32

10 10

YES, It's over 80%

YES, I Have Won

100



Employee Promotion Prediction Using ML

Eligibility status: Great, you are eligible for promotion.

7. Advantages and Disadvantages

Advantages:

The main advantage the proposed model is it that it can predict whether certain attributes like number of trainings, length of service, awards won, etc. have any effect on the employee who might get promoted.

Therefore, employees can have extra opinion on how to make their chances high.

Disadvantages:

- Need more datasets, to increase the accuracy of the algorithms.
- The accuracy of the application depends on the dataset used to train the model
- The proposed application is Web-based, hence cannot be used in Mobile devices.
- The result of the application depends upon the accuracy of the algorithms

8. APPLICATION

By HR Department

The HR department can use this application to check weather the person they are going to promote meets the standards of the all promoted employees in the company in this way they can find employees to promote easily.

By Employees

The employees who wish to get promoted can check weather they are apt for the promotion and if how to improve themselves if they want promotion.

9. CONCLUSION

During this notebook, we built a model that could reliably predict if an employee could get promoted if nessary values are given

This model could be trained with newer employee data and be used to predict other companies’ employee promotion and from that increase the standard of the company

While our model may be inaccurate in some cases, we talked about how our dataset can contain inaccurate values for the education and previous year rating, and oftentimes, our predictions are more accurate than the values in the dataset.

For newer employees, the collected data is significantly more reliable, so our model will be able to perform better with a different, more accurate dataset.

10. FUTURE SCOPE

The purpose of this model is to help those in the Multi national companies. We all know that every industry never stays the same so they innovate to help humans move forward as a species. With every innovation new feature will be added in the form of hardwire or software with these changes it can be a bit hard for the model to predict the performance so in the future we hope to include a better prediction model using a larger dataset of values with a lot of features from the popular car manufacturers which will help in prediction the performance more accurately

We can also redesign the web application to follows the latest trends and also so support different languages in the future.

11. BIBLIOGRAPHY

Data Science for Beginners

www.wikipedia.org

www.google.com

www.github.org

11 .1 Appendix

Employee Promotion Prediction.ipynb

```
#!/usr/bin/env python
# coding: utf-8

# In[8]:


import numpy as np
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
import warnings
warnings.filterwarnings('ignore')
from sklearn.preprocessing import LabelEncoder
from sklearn.tree import DecisionTreeClassifier
from sklearn.ensemble import RandomForestClassifier,
GradientBoostingClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.model_selection import cross_val_score
import pickle
from sklearn.metrics import classification_report,confusion_matrix

plt.style.use('fivethirtyeight')



import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3

def __iter__(self): return 0

# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It
includes your credentials.
# You might want to remove those credentials before you share the
notebook.
client_ac4ae00215fd46288be593d83ddbe528 =
ibm_boto3.client(service_name='s3',
    ibm_api_key_id='gx1UZMzT-nJ08E16hDKvC3cbFfgGV10Vo51QiXQQTELF',
    ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
    config=Config(signature_version='oauth'),
    endpoint_url='https://s3.private.us.cloud-object-
storage.appdomain.cloud')

body =
client_ac4ae00215fd46288be593d83ddbe528.get_object(Bucket='employeepromoti
onprediction-donotdelete-pr-
cdsyvyuo6z1r19',Key='emp_promotion.csv')['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType(
__iter__, body )

data = pd.read_csv(body)
data.head()
 data

data.shape


data.describe(include='all')


# Data is imbalanced

plt.figure(figsize=(10,4))
plt.subplot(121)
sns.countplot(data['is_promoted'])
```

```
plt.subplot(122)
data['is_promoted'].value_counts().plot(kind='pie',autopct =
'%.2f%%',shadow=True)
plt.show()


plt.figure(figsize=(16,10))
plt.subplot(231)
plt.axis('off')
plt.title('KPIs_met >80%')
data['KPIs_met >80%'].value_counts().plot(kind='pie',shadow=True,autopct =
'%.2f%%')
plt.subplot(232)
plt.axis('off')
plt.title('awards_won?')
data['awards_won?'].value_counts().plot(kind='pie',shadow=True,autopct =
'%.2f%%')
plt.subplot(233)
plt.axis('off')
plt.title('previous_year_rating')
data['previous_year_rating'].value_counts().plot(kind='pie',shadow=True,au
topct = '%.2f%%')
plt.show()


plt.figure(figsize=(14,6))
plt.subplot(121)
sns.boxplot(data['length_of_service'],color='g')
plt.subplot(122)
sns.boxplot(data['avg_training_score'],color='b')
# In[16]:


plt.figure(figsize=(24,20))
plt.subplot(231)
```

```
#sns.countplot(data['gender'],hue=data['is_promoted'])
plt.title('Promotion based on gender')
plt.subplot(232)
plt.xticks(rotation=45)
sns.countplot(data['department'],hue=data['is_promoted'])
plt.title('Promotion based on department')
plt.subplot(233)
sns.countplot(data['no_of_trainings'],hue=data['is_promoted'])
plt.title('Promotion based on no_of_trainings')
plt.subplot(234)
sns.countplot(data['previous_year_rating'],hue=data['is_promoted'])
plt.title('Promotion based on previous_year_rating')
plt.subplot(235)
sns.countplot(data['awards_won?'],hue=data['is_promoted'])
plt.title('Promotion based on awards_won?')
plt.subplot(236)
sns.countplot(data['KPIs_met >80%'],hue=data['is_promoted'])
plt.title('Promotion based on department')
plt.show()


plt.figure(figsize=(20,6))
sns.barplot(data['avg_training_score'],data['previous_year_rating'],data['
is_promoted'])


# #### Drop Unwanted Stuff


data.drop(['employee_id','gender','region','recruitment_channel'],axis=1,i
nplace=True)



data
```

```
# #### Checking for null


data.isnull().sum()


print(data["education"].value_counts())


mv=data[data["education"].isnull()].index.tolist()
data.drop(mv, axis=0, inplace=True)


data.isnull().sum():


print(data["previous_year_rating"].value_counts())


data["previous_year_rating"]=data["previous_year_rating"].fillna(data["pre
vious_year_rating"].mode()[0])


negative=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)]


negative


data.drop(index=[31860,51374],inplace=True)


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


IQR= q3-q1
```

```
ub=(1.5*IQR)+q3
lb=(1.5*IQR)-q1


print("q1" ,q1)
print('q3 :',q3)
print('IQR :',IQR)
print('Upper bound :',ub)
print('Lower bound :',lb)
print ('skewed data : ',len(data[data[ 'length_of_service' ]>ub]))
]:


print ('skewed data : ',len(data[data[ 'length_of_service' ]>ub]))


pd.crosstab([data['length_of_service']>ub],data['is_promoted'])


data['length_of_service']=[ub if x>ub else x for x in
data['length_of_service']]


data[ 'education' ].isnull()


data['education']=data['education'].replace(("Below Secondary",
"Bachelor's","Master's & above"),(1,2,3))




lb = LabelEncoder()
data['department']=lb.fit_transform(data['department'])


# ##### spliting and resampling
```

```
x=data.drop('is_promoted',axis=1)
y = data['is_promoted']
print(x.shape)
print(y.shape)


# In[39]:


x


# In[42]:


from sklearn.model_selection import train_test_split


# In[44]:


x_train,x_test,y_train,y_test =
train_test_split(x,y,test_size=0.3,random_state=10)


# In[45]:


print('Shape of x_train {}'.format(x_train.shape))
print('Shape of y_train {}'.format(y_train.shape))
print('Shape of x_test {}'.format(x_test.shape))
print('Shape of y_test {}'.format(y_test.shape))


# In[46]:
```

```python
rf= RandomForestClassifier()
rf.fit(x_train,y_train)
yPred = rf.predict(x_test)


# In[47]:


pickle.dump(rf,open('model.pkl','wb'))


# In[48]:


get_ipython().system('tar -zcvf employee-promotion-model_new.tgz
model.pkl')


# In[49]:


ls -1


# In[50]:


get_ipython().system('pip install watson-machine-learning-client --
upgrade')


# In[51]:


from ibm_watson_machine_learning import APIClient
wml_credentials = {
    "url": "https://us-south.ml.cloud.ibm.com",
    "apikey": "tO2_3DRl4UbNH9hZzRwmbp3pqdJYb7bsfBnzif10rJ6R"
}
client = APIClient(wml_credentials)


# In[52]:


def guid_from_space_name(client, space_name):
    space = client.spaces.get_details()
    #print(space)
    return(next(item for item in space['resources'] if
item['entity']["name"] == space_name)['metadata']['id'])


# In[56]:


space_uid = guid_from_space_name(client, space_name='EmployeeDeployment')
print("Space UID = " + space_uid)


# In[59]:


client.set.default_space(space_uid)


# In[60]:


client.software_specifications.list()


# In[61]:


software_spec_uid=
client.software_specifications.get_uid_by_name("default_py3.8")
```

```
software_spec_uid


# In[62]:


model_details = client.repository.store_model(model='employee-promotion-
model_new.tgz',

meta_props={client.repository.ModelMetaNames.NAME:"EmployeeDeployment",

client.repository.ModelMetaNames.TYPE:"scikit-learn_0.23",

client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_spec_uid
                                                          },
                                              training_data=x_train,
                                              training_target=y_train)



# In[63]:


model_id = client.repository.get_model_id(model_details)
model_id


# In[64]:


deployment = client.deployments.create(
    artifact_uid=model_id,

meta_props={client.deployments.ConfigurationMetaNames.NAME:"Employee_deplo
y",
    client.deployments.ConfigurationMetaNames.ONLINE: {}
})


# In[ ]:


```

App_IBM.py

```
import pickle
from flask import Flask, render_template, request
import requests

# NOTE: you must manually set API_KEY below using information retrieved
from your IBM Cloud account.
API_KEY = "tO2_3DRl4UbNH9hZzRwmbp3pqdJYb7bsfBnzif10rJ6R"
token_response = requests.post('https://iam.cloud.ibm.com/identity/token',
data={"apikey": API_KEY, "grant_type": 'urn:ibm:params:oauth:grant-
type:apikey'})
mltoken = token_response.json()["access_token"]

header = {'Content-Type': 'application/json', 'Authorization': 'Bearer ' +
mltoken}


app = Flask(__name__)


@app.route('/')
def home():
    return render_template('home.html')


@app.route('/home')
```

```python
def home1():
    return render_template('home.html')


@app.route('/about')
def about():
    return render_template('about.html')


@app.route('/predict')
def predict():
    return render_template('predict.html')


@app.route('/pred', methods=['POST'])
def pred():
    department = request.form['department']
    education = request.form['education']
    if education == '1':
        education = 1
    elif education == '2':
        education = 2
    else:
        education = 3
    no_of_trainings = request.form['no_of_trainings']
    age = request.form['age']
    previous_year_rating = request.form['previous_year_rating']
    length_of_service = request.form['length_of_service']
    KPIs = request.form['KPIs']
    if KPIs == '0':
        KPIs = 0
    else:
        KPIs = 1
    awards_won = request.form['awards_won']
    if awards_won == '0':
        awards_won = 0
    else:
        awards_won = 1
    avg_training_score = request.form['avg_training_score']
    total = [[department, education, no_of_trainings, age,
float(previous_year_rating), float(length_of_service),
              KPIs, awards_won, avg_training_score]]


    payload_scoring = {"input_data": [{"field": [['department',
'education', 'no_of_trainings', 'age', 'previous_year_rating',
'length_of_service', 'KPIs','awards_won','avg_training_score']],
                                       "values": total}]}

    response_scoring = requests.post('https://us-
south.ml.cloud.ibm.com/ml/v4/deployments/a70ae845-cc08-46dd-b178-
6e8f5e0a525a/predictions?version=2022-03-07', json=payload_scoring,
headers={'Authorization': 'Bearer ' + mltoken})
    print("Scoring response")
    print(response_scoring.json())
    #print(predictions)

    prediction = response_scoring.json()

    prediction = pred['predictions'][0]['values'][0][0]


    if prediction == 0:
        text = 'Sorry, you are not eligible for promotion'
    else:
        text = 'Great, you are eligible for promotion'
    return render_template('submit.html', predictionText=text)


if __name__ == '__main__':
    app.run(debug=True)
```