

Drug Classification Using IBM Watson Studio Machine Learning

1 INTRODUCTION

1.1 Overview

Nowadays our lifestyle has been changing. Per family, at least one person has Motorcycles or cars, etc. In the same way, we all have health issues. An earlier generation has proved "Health is Wealth". But, for our generation, this slogan is quite challenging.

We have completely moved with hybrid veggies, junk foods, etc. Due to these foods, we are not getting sufficient nutrition and suffering from health issues. To overcome this, we are consulting doctors and taking some drugs as medicines. In this project, we have some characteristics of the patients as a dataset. The target variable of this dataset is Drugs. The drug names are confidential. So, those names are replaced as DrugX, DrugY, DrugA, DrugB, and DrugC. By consulting a doctor each time, you have to pay a doctor fee and additional charges. For saving money and time, you can use this web application to predict your drug type. The main purpose of the Drug Classification system is to predict the suitable drug type confidently for the patients based on their characteristics. The main problem here is not just the feature sets and target sets but also the approach that is taken in solving these types of problems.

Here we are using classification algorithms such as Decision tree, Random forest, KNN, and xgboost. The data is trained and tested with these algorithms. From this best model is selected and saved in pkl format. And will be doing flask integration and IBM deployment.

1.2 Purpose

The main purpose of the Drug Classification system is to predict the suitable drug type confidently for the patients based on their characteristics.

2 LITERATURE SURVEY

2.1 Existing problem

Nowadays our lifestyle has been changing. Per family, at least one person has Motorcycles or cars, etc. In the same way, we all have health issues. We have completely moved with hybrid

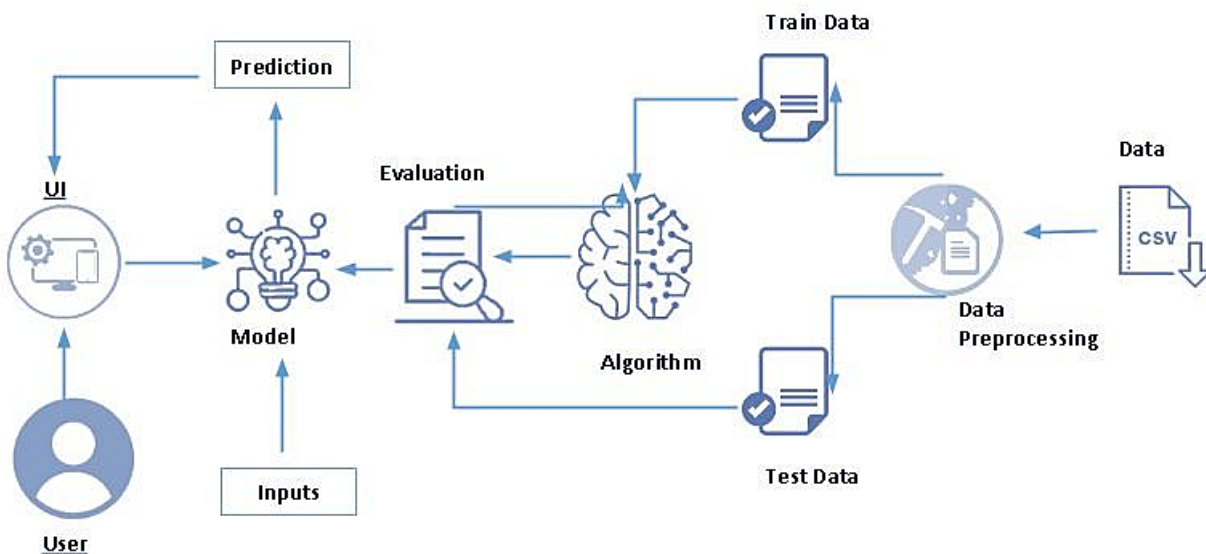
veggies, junk foods, etc. Due to these foods, we are not getting sufficient nutrition and suffering from health issues. To overcome this, we are consulting doctors and taking some drugs as medicines. By consulting a doctor each time, you have to pay a doctor fee and additional charges.

2.2 Proposed solution

By consulting a doctor each time, you have to pay a doctor fee and additional charges. For saving money and time, you can use this web application to predict your drug type. The main purpose of the Drug Classification system is to predict the suitable drug type confidently for the patients based on their characteristics.

3 THEORITICAL ANALYSIS

3.1 Block diagram



3.2 Hardware / Software designing

Software requirements:

- Anaconda navigator
- Python packages
- IBM watson studio

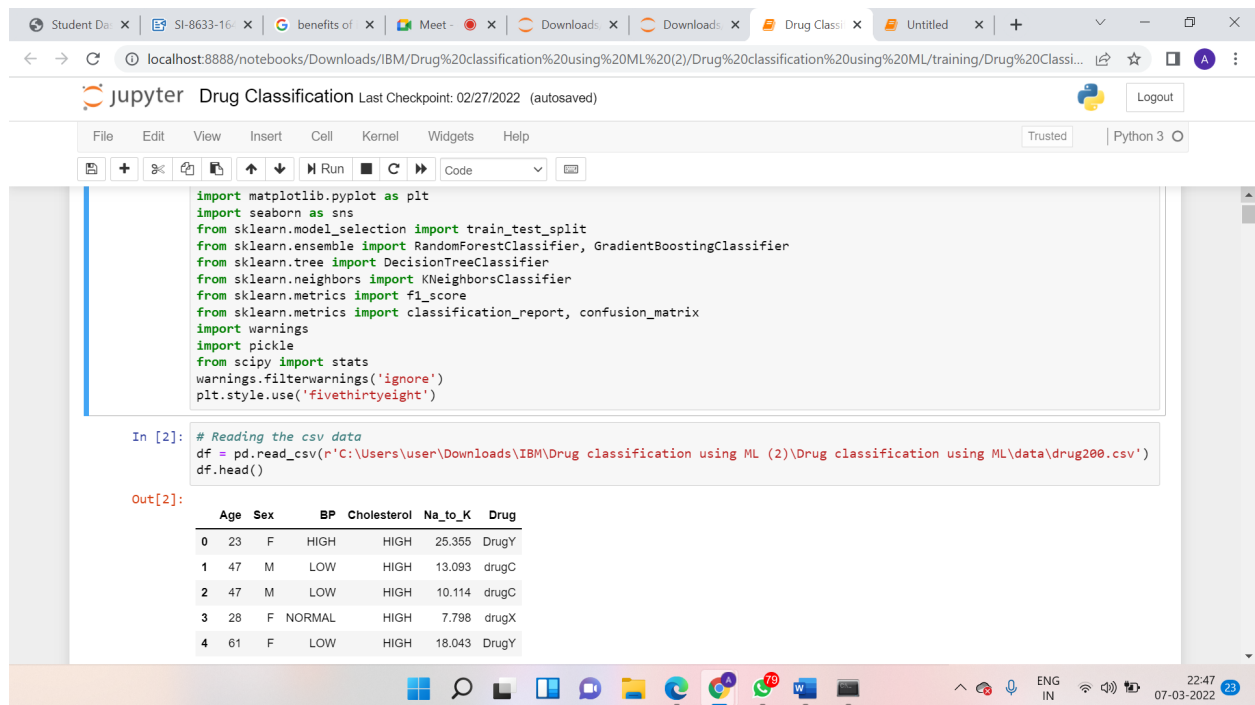
- IBM Watson Machine Learning
- IBM Cloud Object Storage

4 EXPERIMENTAL INVESTIGATIONS

- Shape of inputs plays a major role in the correctness of the model.
- IBM Cloud helps to deploy machine learning models and test the correctness of our model.
- Integrating Flask with the machine learning model involves a lot of data preprocessing to make the predictions correctly.

5 RESULT

source code



The screenshot shows a Jupyter Notebook titled "Drug Classification" with the following code and output:

```
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import f1_score
from sklearn.metrics import classification_report, confusion_matrix
import warnings
import pickle
from scipy import stats
warnings.filterwarnings('ignore')
plt.style.use('fivethirtyeight')
```

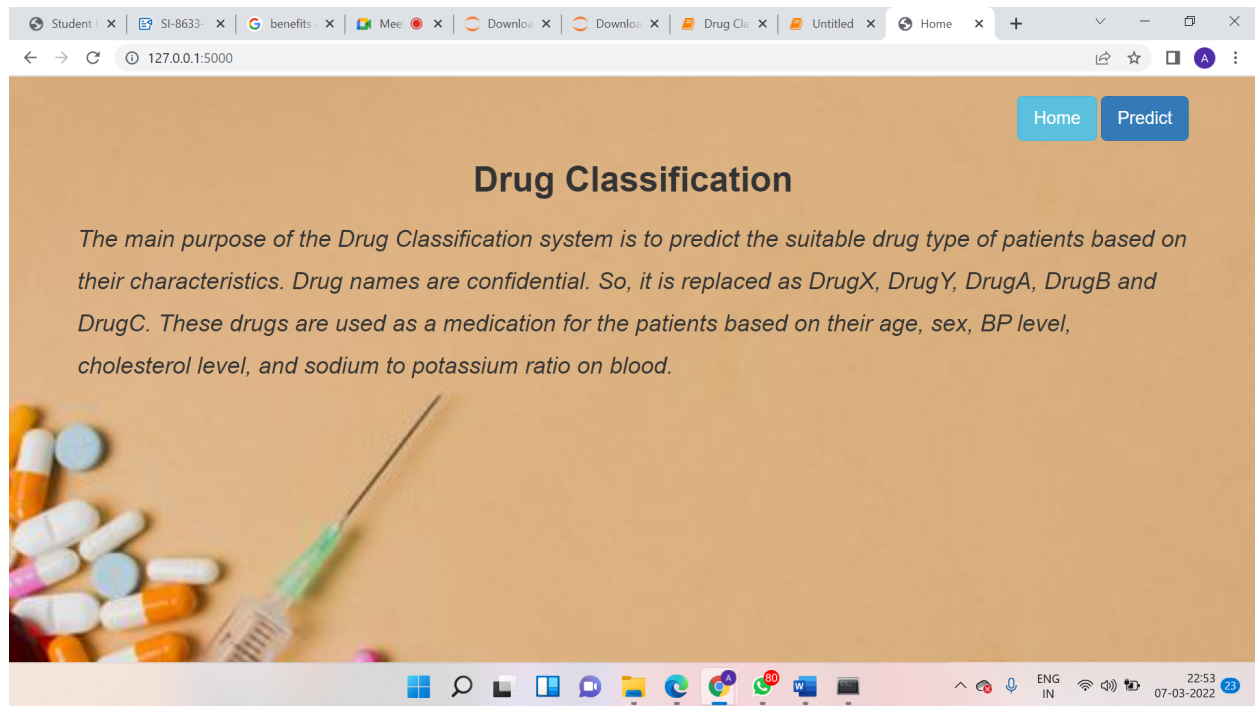
In [2]: # Reading the csv data

```
df = pd.read_csv(r"C:\Users\user\Downloads\IBM\Drug classification using ML (2)\Drug classification using ML\data\drug200.csv')
df.head()
```

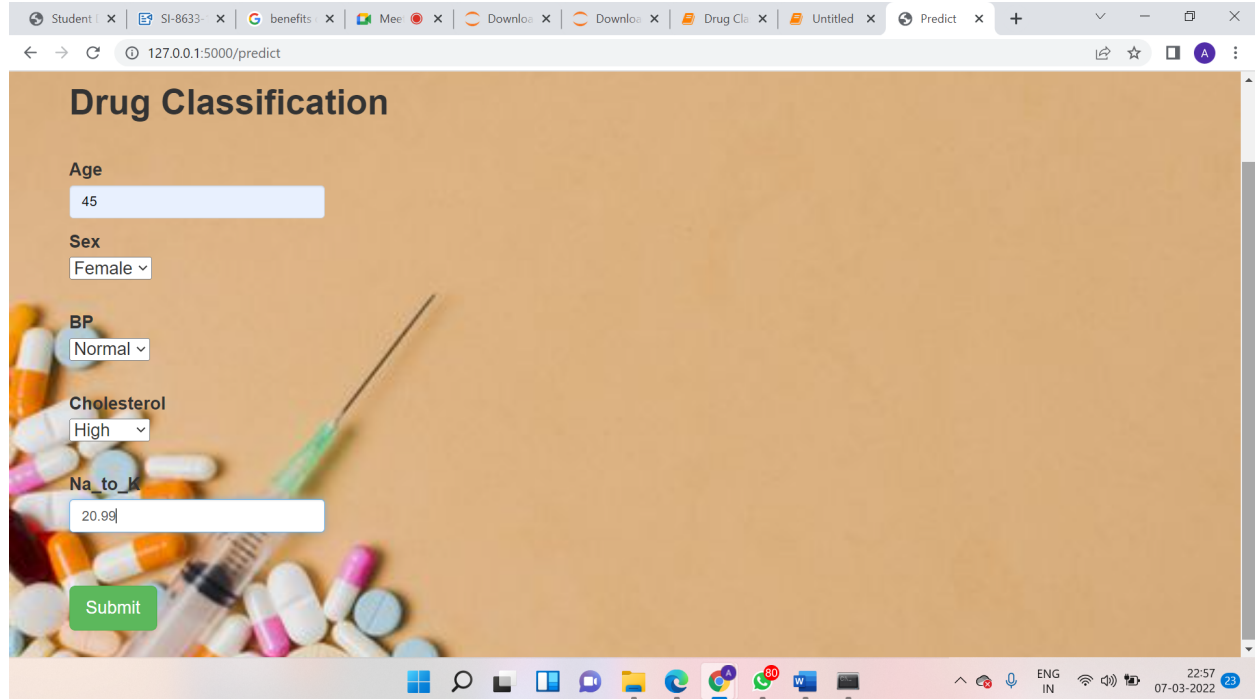
Out[2]:

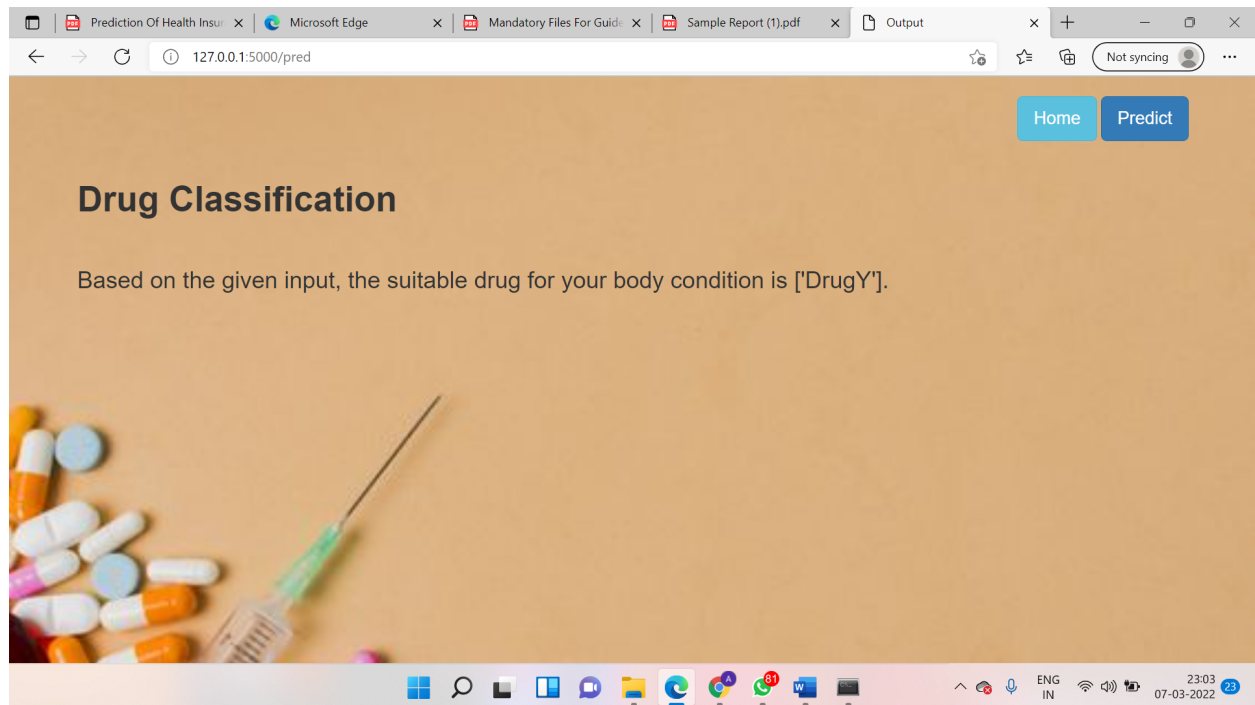
	Age	Sex	BP	Cholesterol	Na_to_K	Drug
0	23	F	HIGH	HIGH	25.355	DrugY
1	47	M	LOW	HIGH	13.093	drugC
2	47	M	LOW	HIGH	10.114	drugC
3	28	F	NORMAL	HIGH	7.798	drugX
4	61	F	LOW	HIGH	18.043	DrugY

UI OUTPUT:



AFTER ENTERING VALUES:





6 ADVANTAGES AND DISADVANTAGES

ADVANTAGES

- It saves money
- It saves time
- Helps to predict drug type

DISADVANTAGES

- predicting the drug is little bit risky
- no prescription of doctor, sometimes drug may cause allergies
- less comfort and satisfaction

7 APPLICATION

- Medical store
- Shopping malls

8 CONCLUSION

The Drug Classification system help us to predict the suitable drug type confidently for the patients based on their characteristics. It help us to save time and money. Prescription drugs are

a key component in the healthcare. Discusses the challenges to providing accessible prescription drugs. Explores technological advances and new business models in providing pharmacy services. Prescription drugs are a key component in the healthcare.

9 FUTURE SCOPE

The Drug Classification helps us to save time and money and without consulting a doctor we will be able to predict the drug type. We can also add doctor prescriptions as well as test results such as blood test, ECG, Bp ect. At the same time users can view their old data.

10 BIBLIOGRAPHY

<https://medlineplus.gov/cholesterolmedicines.html>

<https://www.nhs.uk/conditions/high-cholesterol/medicines-for-high-cholesterol/>

<https://medlineplus.gov/ency/article/007278.htm>

<https://www.mayoclinic.org/diseases-conditions/high-blood-pressure/diagnosis-treatment/drc-20373417>

11 APPENDIX

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier, GradientBoostingClassifier
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.metrics import f1_score
from sklearn.metrics import classification_report, confusion_matrix
import warnings
import pickle
from scipy import stats
warnings.filterwarnings('ignore')
plt.style.use('fivethirtyeight')
# Reading the csv data
df = pd.read_csv(r'C:\Users\user\Downloads\IBM\Drug classification using ML (2)\Drug classification using ML\data\drug200.csv')
df.head()
# Checking the distribution (normal or skewed)
plt.figure(figsize=(12,5))
plt.subplot(121)
sns.distplot(df['Age'],color='r')
```

```

plt.subplot(122)
sns.distplot(df['Na_to_K'])
plt.show()
# Creating a data frame with categorical features for following visualization
df_cat = df.select_dtypes(include='object')
df_cat.head()
# Visualizing the count of categorical variable.
plt.figure(figsize=(18,4))
for i,j in enumerate(df_cat):
    plt.subplot(1,4,i+1)
    sns.countplot(df[j])
# Visualizing the relation between drug, BP, sex & cholesterol
plt.figure(figsize=(20,5))
plt.subplot(131)
sns.countplot(df['Drug'],hue=df['BP'])
plt.legend(loc='upper right')
plt.subplot(132)
sns.countplot(df['Drug'],hue=df['Sex'])
plt.subplot(133)
sns.countplot(df['Drug'],hue=df['Cholesterol'])
# Creating a new column Age_. This column shows the categorized age
df['Age_'] = ['15-30' if x<=30 else '30-50' if x>30 and x<=50 else '50-75' for x in df['Age']]
df.head()
# Finding the relation between categorized age and drug
pd.crosstab(df['Age_'],[df['Drug']])
# Removing the Age_ column
df.drop('Age_',axis=1,inplace=True)
df.head()
sns.swarmplot(df['Drug'],df['Na_to_K'],hue=df['BP'])
# DrugC is used for low BP patient, DrugY is used on patients having Na_to_K > 15.
df.describe(include='all')
# Shape of csv data
df.shape
# Finding null values
df.isnull().sum()
# Checking the information of features
df.info()
# Finding outliers
plt.figure(figsize=(12, 5))
sns.boxplot(df['Na_to_K'])
plt.show()
# From the above plot age column is normally distributed. Na_to_k is
right skewed (mean>mode). To overcome skewness transformation
techniques can be used.

```

```

print(stats.mode(df['Na_to_K']))
print(np.mean(df['Na_to_K']))
# Na_to_K has 8 outliers. In this project we are not going to handle
outliers. Most of the classification algorithms are not sensitive to
outliers.
q1 = np.quantile(df['Na_to_K'],0.25)
q3 = np.quantile(df['Na_to_K'],0.75)
IQR = q3-q1
upper_bound = q3+(1.5*IQR)
lower_bound = q1-(1.5*IQR)
print('q1 :',q1)
print('q3 :',q3)
print('IQR :',IQR)
print('Upper Bound :',upper_bound)
print('Lower Bound :',lower_bound)
print('Skewed data :',len(df[df['Na_to_K']>upper_bound]))
print('Skewed data :',len(df[df['Na_to_K']<lower_bound]))
# To handle outliers transformation techniques are used.
def transformationPlot(feature):
    plt.figure(figsize=(12,5))
    plt.subplot(1,2,1)
    sns.distplot(feature)
    plt.subplot(1,2,2)
    stats.probplot(feature,plot=plt)
transformationPlot(np.log(df['Na_to_K']))
df['Na_to_K']=np.log(df['Na_to_K'])
# Replacing low, normal & high with 0, 1 & 2...
df[['BP']] = [0 if x=='LOW' else 1 if x=='NORMAL' else 2 for x in
df['BP']]
# Replacing normal and high cholesterol with 0 & 1
df[['Cholesterol']] = [0 if x=='NORMAL' else 1 for x in
df['Cholesterol']]
# Replacing female and male with 0 & 1
df['Sex'] = [0 if x=='F' else 1 for x in df['Sex']]
df.head()
# With the help of pairplot we find the correct classification
algorithms
sns.pairplot(df,hue='Drug')
x = df.drop('Drug',axis=1)
x.head()
y = df['Drug']
y.head()

```



```

x_train, x_test, y_train, y_test = train_test_split(x, y,
test_size=0.3, random_state=10)
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))
def decisionTree(x_train, x_test, y_train, y_test):
    dt=DecisionTreeClassifier()
    dt.fit(x_train,y_train)
    yPred = dt.predict(x_test)
    print('***DecisionTreeClassifier***')
    print('Confusion matrix')
    print(confusion_matrix(y_test,yPred))
    print('Classification report')
    print(classification_report(y_test,yPred))
def randomForest(x_train, x_test, y_train, y_test):
    rf = RandomForestClassifier()
    rf.fit(x_train,y_train)
    yPred = rf.predict(x_test)
    print('***RandomForestClassifier***')
    print('Confusion matrix')
    print(confusion_matrix(y_test,yPred))
    print('Classification report')
    print(classification_report(y_test,yPred))
def KNN(x_train, x_test, y_train, y_test):
    knn = KNeighborsClassifier()
    knn.fit(x_train,y_train)
    yPred = knn.predict(x_test)
    print('***KNeighborsClassifier***')
    print('Confusion matrix')
    print(confusion_matrix(y_test,yPred))
    print('Classification report')
    print(classification_report(y_test,yPred))
def xgboost(x_train, x_test, y_train, y_test):
    xg = GradientBoostingClassifier()
    xg.fit(x_train,y_train)
    yPred = xg.predict(x_test)
    print('***GradientBoostingClassifier***')
    print('Confusion matrix')
    print(confusion_matrix(y_test,yPred))

```

```

    print('Classification report')
    print(classification_report(y_test, yPred))
def compareModel(x_train, x_test, y_train, y_test):
    decisionTree(x_train, x_test, y_train, y_test)
    print('-'*100)
    randomForest(x_train, x_test, y_train, y_test)
    print('-'*100)
    KNN(x_train, x_test, y_train, y_test)
    print('-'*100)
    xgboost(x_train, x_test, y_train, y_test)
compareModel(x_train, x_test, y_train, y_test)
# Decision tree and Random forest performs well
from sklearn.model_selection import cross_val_score
# Random forest model is selected
rf = RandomForestClassifier()
rf.fit(x_train, y_train)
yPred = rf.predict(x_test)
f1_score(yPred, y_test, average='weighted')
cv = cross_val_score(rf, x, y, cv=5)
np.mean(cv)
pickle.dump(rf, open('model.pkl', 'wb'))
!tar -zcvf drug-classification-model_new.tgz model.pkl
ls -l
!pip install watson-machine-learning-client --upgrade
from ibm_watson_machine_learning import APIClient
wml_credentials = {
    "url": "https://us-south.ml.cloud.ibm.com",
    "apikey": "NW6_YjwnxreHnFE-dgSFLYScfWDFhMW2WOY0qggTfdns"
}
client = APIClient(wml_credentials)
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'])
space_uid = guid_from_space_name(client,
space_name='DrugClassification')
print("Space UID = " + space_uid)
client.set.default_space(space_uid)
client.software_specifications.list()

```

```

import sklearn
sklearn.__version__
software_spec_uid=
client.software_specifications.get_uid_by_name("default_py3.8")
software_spec_uid
model_details = client.repository.store_model(model='drug-
classification-model_new.tgz',

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

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)
model_id = client.repository.get_model_id(model_details)
model_id
# Deploy
deployment = client.deployments.create(
    artifact_uid=model_id,

meta_props={client.deployments.ConfigurationMetaNames.NAME:"DrugClassifi
cation_deploy",
    client.deployments.ConfigurationMetaNames.ONLINE: {}
})
#'00a67b76-022e-47d1-a1d0-031cc258204f'

```

