Drug Classification Using 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.

1.2 Proposed System

To avoid this,We will be using classification algorithms such as Decision tree, Random forest, KNN, and xgboost. We will train and test the data with these algorithms. From this best model is selected and saved in pkl format. We will be doing flask integration and IBM deployment.

2. LITERATURE SURVEY

2.1 Existing Problem

Some drugs's mechanism of action are unrevealed. Meanwhile, several drugs' mechanism of action have been already discovered. For example, Aspirin's mechanism of action involves irreversible inhibition of the enzyme cyclooxygenase, which reduces inflammation and pain by decreasing the formation of thromboxanes and prostaglandins.

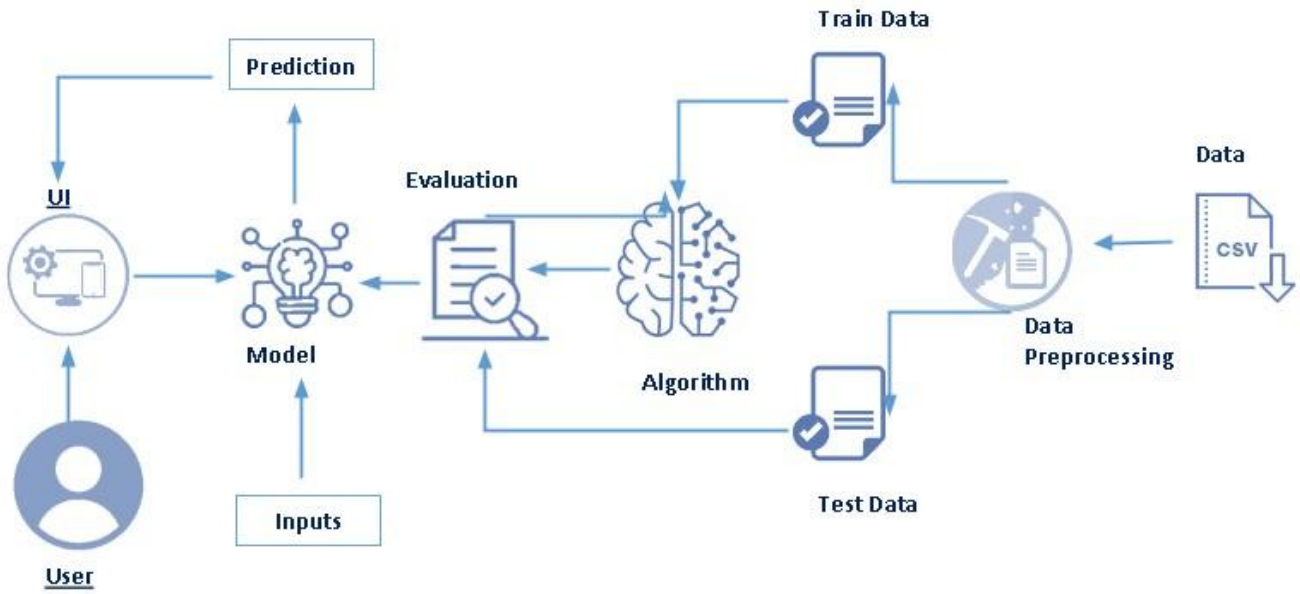
Different drugs can have different mechanism of action. A literature survey of mechanism of action of various drugs.

2.2 Proposed System

To avoid this,We will be using classification algorithms such as Decision tree, Random forest, KNN, and xgboost. We will train and test the data with these algorithms. From this best model is selected and saved in pkl format. We will be doing flask integration and IBM deployment.

3.THE ORETICAL ANALYSIS

3.1 Block Diagram



3.2 Hardware/Software designe

3.2.1 Hardware requirements

- 2 GB ram or above
- Dual core processor or above
- Internet connection

3.2.2 software requirements

- Anaconda Navigator
- Python packages
- Spyder
- VS Code

4. EXPERIMENTAL INVESTIGATION

4.1 Data Pre-Processing

The download data set is not suitable for training the machine learning model as it might have so much of randomness so we need to clean the dataset properly in order to fetch good results. This activity includes the following steps.

- Handling missing values
- Handling categorical data
- Handling outliers
- Scaling Techniques
- Splitting dataset into training and test set

4.2 Model Building

After pre procesing and cleaning the data we ned to train the data with different machine learning algorithms such as Decision Tree, Random Forest, KNN, Xgboost models. Among the four algorithms the best and most accurate algotithm is chosen and saved as .plk file

4.3 Application Building

Now we have to build web pages by using flask framework and HTML. After running the flask code we get a URL in the terminal. Copy the URL and paste it in a browser. We will be directed to the web pages that be built. Now

5. FLOWCHART

5.1 Project Flow:

- The user interacts with the UI to enter the input.
- Entered input is analyzed by the model which is integrated.
- Once the model analyses the input the prediction is showcased on the UI

1 Data collection

  - Collect the dataset or create the dataset

2 Visualizing and analyzing data

  - Univariate analysis
  - Bivariate analysis
  - Multivariate analysis
  - Descriptive analysis

3 Data pre-processing

  - Checking for null values
  - Handling outlier
  - Handling categorical data
  - Splitting data into train and test

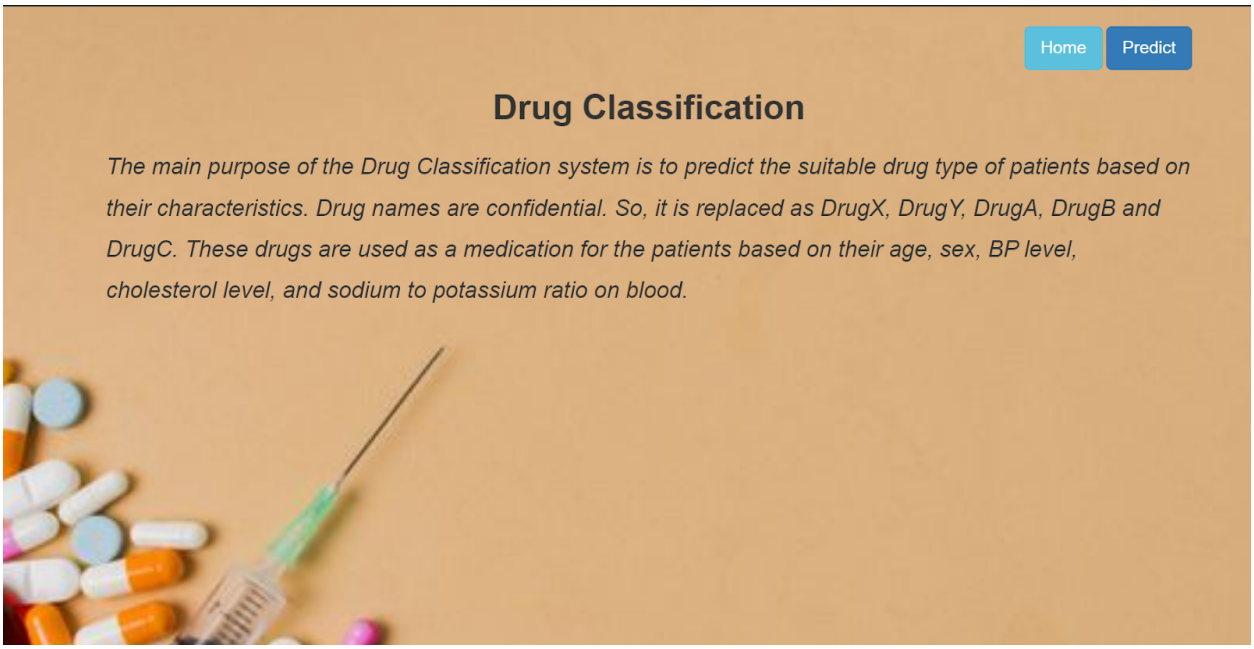
3 Model building

  - Import the model building libraries
  - Initializing the model
  - Training and testing the model
  - Evaluating the performance of the model
  - Save the model

4 Application Building

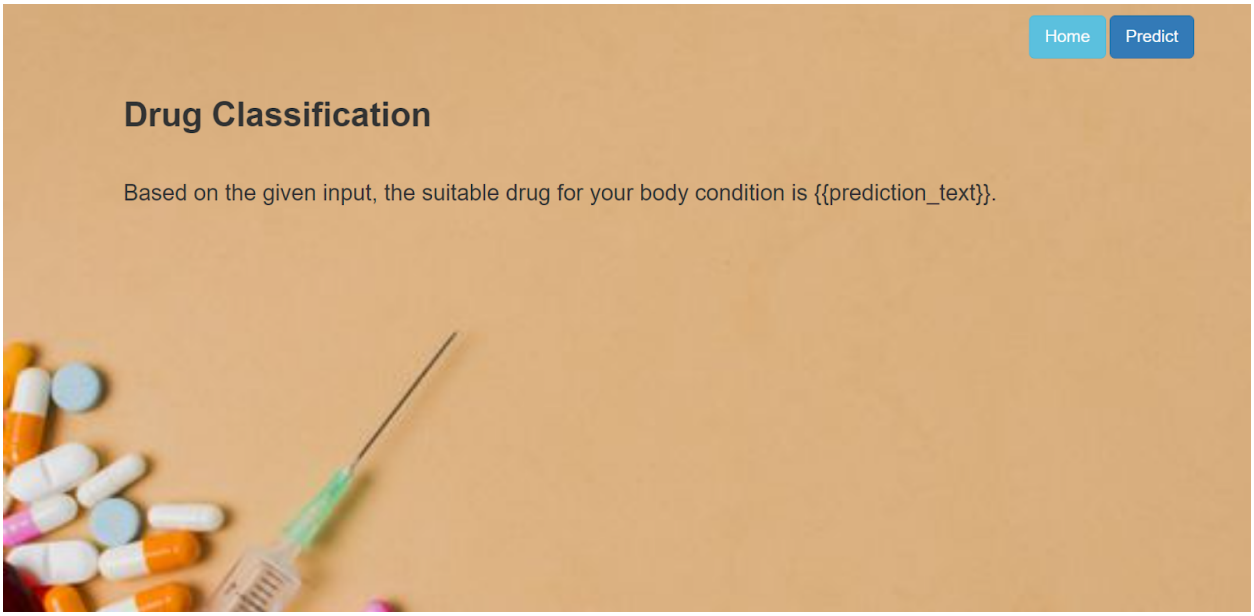
- Create an HTML file
- Build python code

6. Result



Now when you click on predict button from top right corner you will get redirected to predict.html

Now when you click on submit button from left bottom corner you will get redirected to submit.html



7.Advantages and disadvantages

7.1 Advantages:

1) used for manufacturing medicine

2) to cure the diseases

3) recovery of common disease

4) also for the plant growth

5) for feel stressed out

7.2 Disadvantages:

1) addiction of drugs is harmful for health

2) it causes many disease

3) it damage the body parts

4) it effects on family environment

5) person goes in depression in lack of drugs.

8. Applications

- Anaconda Navigator
- Python packages
- Spyder
- VS Code

9 Conclusion

Mechanism of drug can help scientists accelerate the drug discovery process. This paper discussed various machine learning models to predict the mechanism of action of a drug. Also, a Flask-based web application is introduced, through which a user can input a custom testing features data-set containing gene expression and cell viability levels. The output produced is the top classes of drugs, along with their scatter plot. This can help scientists to predict the Mechanism of Action and can help them in the discovery of new drugs.

10. FUTURE SCOPE

This program allows users to predict if any advance in science and technology finds immediately its application in medicine, in pharmacy, in drug discovery and development. Investments in drug design are worthwhile because as better is designed a given drug candidate during the experimental stage, as less likely is for the drug to fail in the late stages where the tests are more expensive, especially in the clinical trials.

11. BIBLIOGRAPHY

- https://www.researchgate.net/publication/356518190_Classification_of_drugs_based_on_mechanism_of_action_using_machine_learning_techniques
- https://www.sciencedirect.com/science/article/pii/S0016328719303702
- https://www.frontiersin.org/articles/10.3389/fgene.2020.01000/full

12. APPENDIX

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns

df=pd.read_csv('drug200 (2).csv')

df.shape,df.size
df.sample(5)
df.info()
df.isnull().sum()
df.duplicated().sum()
df.describe(include='all')
df.corr()
plt.figure(figsize=(4,4))
sns.heatmap(df.corr(),annot=True,fmt='f',linewidth=True)
plt.show()
df.select_dtypes(exclude='object')
import warnings
warnings.filterwarnings('ignore')
plt.style.use('fivethirtyeight')
plt.figure(figsize=(12,4))
plt.subplot(121)
sns.distplot(df['Age'],hist=False)
plt.subplot(122)
sns.distplot(df['Age'],kde=False)
plt.show()
```

```
plt.figure(figsize=(12,4))
plt.subplot(121)
sns.distplot(df['Na_to_K'],hist=False)
plt.subplot(122)
sns.distplot(df['Na_to_K'],kde=False)
plt.show()
plt.figure(figsize=(12,4))
sns.boxplot(df['Age'])
plt.show()
plt.figure(figsize=(12,4))
sns.boxplot(df['Na_to_K'])
plt.show()
df.select_dtypes(include='object')
print("no. of unique object:",df['Sex'].nunique())
print(df['Sex'].value_counts())
plt.figure(figsize=(12,4))
plt.subplot(121)
sns.countplot(df['Sex'])
plt.subplot(122)
plt.pie(df.Sex.value_counts(),[0,0.1],colors=['orange','blue'],labels=['Ma
le','Female'],autopct="%.2f")
plt.show()
print("no. of unique object:",df['BP'].nunique())
print(df['BP'].value_counts())
plt.figure(figsize=(12,4))
plt.subplot(121)
sns.countplot(df['BP'])
plt.subplot(122)
plt.pie(df.BP.value_counts(),[0,0.1,0.2],colors=['#094c72','#298fca','#8bb
dd9'],labels=['HIGH','NORMAL','LOW'],autopct='%.2f')
plt.show()
plt.figure(figsize=(12,4))
plt.subplot(121)
sns.countplot(df['BP'])
plt.subplot(122)
plt.pie(df.BP.value_counts(),[0,0.1,0.2],colors=['#094c72','#298fca','#8bb
```

```
dd9'],labels=['HIGH','NORMAL','LOW'],autopct='%.2f')
plt.show()
plt.figure(figsize=(12,4))
plt.subplot(121)
sns.countplot(df['Cholesterol'])
plt.subplot(122)
plt.pie(df.Cholesterol.value_counts(),[0,0.1],colors=['#f2a580','#e9692c']
,labels=['NORMAL','HIGH'],autopct='%.2f')
plt.show()
print("no. of unique object:",df['Drug'].nunique())
print(df['Drug'].value_counts())
plt.figure(figsize=(12,4))
plt.subplot(121)
sns.countplot(df['Drug'])
plt.subplot(122)
plt.pie(df.Drug.value_counts(),labels=['drugY','drugX','drugA','drugC','dr
ugB'],autopct='%.2f')
plt.show()
df.select_dtypes(include='object')
df.select_dtypes(exclude='object')
sns.relplot(x='Age',y='Na_to_K',data=df,kind='line')
plt.show()
sns.relplot(y='Age',x='Na_to_K',data=df)
plt.show()
sns.catplot(x='Sex',y='Age',data=df)
plt.show()
sns.catplot(x='Sex',y='Age',kind='swarm',data=df)
plt.show()
sns.catplot(x='BP',y='Age',data=df)
plt.show()
sns.catplot(x='BP',y='Age',kind='swarm',data=df)
plt.show()
sns.catplot(x='Cholesterol',y='Age',data=df)
plt.show()
sns.catplot(x='Cholesterol',y='Age',kind='swarm',data=df)
plt.show()
```

```
sns.catplot(x='Drug',y='Age',data=df)
plt.show()
sns.catplot(x='Drug',y='Age',kind='swarm',data=df)
plt.show()
sns.catplot(x='BP',y='Na_to_K',data=df)
plt.show()
sns.catplot(x='BP',y='Na_to_K',kind='swarm',data=df)
plt.show()
sns.catplot(x='Cholesterol',y='Na_to_K',data=df)
plt.show()
sns.catplot(x='Cholesterol',y='Na_to_K',kind='swarm',data=df)
plt.show()
sns.catplot(x='Drug',y='Na_to_K',data=df)
plt.show()
sns.catplot(x='Drug',y='Na_to_K',kind='swarm',data=df)
plt.show()
sns.catplot(x='Sex',y='Na_to_K',data=df)
plt.show()
sns.catplot(x='Sex',y='Na_to_K',kind='swarm',data=df)
plt.show()
sns.relplot(x='Age',y='Na_to_K',hue='Sex',data=df)
sns.relplot(x='Age',y='Na_to_K',hue='Sex',style='Drug',data=df)
sns.relplot(x='Age',y='Na_to_K',hue='Sex',style='Drug',size='BP',data=df)
sns.relplot(x='Age',y='Na_to_K',hue='Sex',style='Drug',size='Cholesterol',
data=df)
sns.catplot(x='Drug',y='Age',hue='Sex',split=True,kind='violin',data=df)
plt.show()
sns.catplot(x='Drug',y='Na_to_K',hue='Sex',split=True,kind='violin',data=d
f)
plt.show()
sns.catplot(x='BP',y='Age',split=True,hue='Sex',kind='violin',data=df)
plt.show()
sns.catplot(x='BP',y='Na_to_K',hue='Sex',split=True,kind='violin',data=df)
plt.show()
sns.catplot(x='Cholesterol',y='Age',split=True,hue='Sex',kind='violin',dat
a=df)
```

```
plt.show()
sns.catplot(x='Cholesterol',y='Na_to_K',hue='Sex',split=True,kind='violin'
,data=df)
plt.show()
sns.catplot(x='Drug',y='Age',hue='BP',kind='box',data=df)
plt.show()
sns.catplot(x='Drug',y='Age',hue='Cholesterol',kind='box',data=df)
plt.show()
df.isnull().sum()
sns.boxplot(df['Na_to_K'])
plt.show()
percentile_25=df.Na_to_K.quantile(0.25)
percentile_75=df.Na_to_K.quantile(0.75)
print("25th percentile of Na_to_K:",percentile_25)
print("75th percentile of Na_to_K:",percentile_75)
iqr=percentile_75-percentile_25
print("iqr of Na_to_K:",iqr)
upper_limit=percentile_75+1.5*iqr
lower_limit=percentile_25-1.5*iqr
print("upper limit of Na_to_K:",upper_limit)
print("lower limit of Na_to_K:",lower_limit)
df.Na_to_K.describe()
df['Na_to_K']=np.where(df['Na_to_K']>upper_limit,
                       upper_limit,
                      np.where(df['Na_to_K']<lower_limit,
                               lower_limit,df['Na_to_K']))
df['Na_to_K'].describe()
sns.boxplot(df['Na_to_K'])
plt.show()
X=df.iloc[:,:-1]
y=df.iloc[:,-1]
X
y
from sklearn.model_selection import train_test_split
X_train,X_test,y_train,y_test=train_test_split(X,y,test_size=0.2,random_st
ate=0)
```

```
X_train.shape,X_test.shape
X_train.sample(5)
from sklearn.preprocessing import OrdinalEncoder
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline,make_pipeline
tr1=ColumnTransformer(transformers=[
    ('ohe',OneHotEncoder(sparse=False,dtype='int32'),[1])
],remainder='passthrough')
tr2=ColumnTransformer(transformers=[
    ('oe3',OrdinalEncoder(categories=[['LOW','NORMAL','HIGH']]),[3])
],remainder='passthrough')
tr3=ColumnTransformer(transformers=[
    ('oe4',OrdinalEncoder(categories=[['NORMAL','HIGH']]),[4])
],remainder='passthrough')
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report, confusion_matrix
tr4=DecisionTreeClassifier()
pipe_dt=Pipeline([
    ('tr1',tr1),
    ('tr2',tr2),
    ('tr3',tr3),
    ('tr4',tr4)
])
pipe_dt.fit(X_train,y_train)
y_pred_dt=pipe_dt.predict(X_test)
print('decision tree:')
print('confusion matrix:')
print(confusion_matrix(y_test,y_pred_dt))
print('classification report:')
print(classification_report(y_test,y_pred_dt))
tr5=KNeighborsClassifier()
pipe_knn=Pipeline([
```

```
    ('tr1',tr1),
    ('tr2',tr2),
    ('tr3',tr3),
    ('tr5',tr5)
])
pipe_knn.fit(X_train,y_train)
y_pred_knn=pipe_knn.predict(X_test)
print('KNeighborsClassifier:')
print('confusion matrix:')
print(confusion_matrix(y_test,y_pred_knn))
print('classification report:')
print(classification_report(y_test,y_pred_knn))
tr6=GradientBoostingClassifier()
pipe_gb=Pipeline([
    ('tr1',tr1),
    ('tr2',tr2),
    ('tr3',tr3),
    ('tr6',tr6)
])
pipe_gb.fit(X_train,y_train)
y_pred_gb=pipe_gb.predict(X_test)
print('GradientBoostingClassifier:')
print('confusion matrix:')
print(confusion_matrix(y_test,y_pred_gb))
print('classification report:')
print(classification_report(y_test,y_pred_gb))
tr7=RandomForestClassifier()
pipe_rf=Pipeline([
    ('tr1',tr1),
    ('tr2',tr2),
    ('tr3',tr3),
    ('tr7',tr7)
])
pipe_rf.fit(X_train,y_train)
y_pred_rf=pipe_rf.predict(X_test)
print('RandomForestClassifier:')
```

```
print('confusion matrix:')
print(confusion_matrix(y_test,y_pred_rf))
print('classification report:')
print(classification_report(y_test,y_pred_rf))
from sklearn.model_selection import cross_val_score
from sklearn.metrics import f1_score
print('rf:',y_pred_rf,'dt:',y_pred_dt,'knn:',y_pred_knn,'gb:',y_pred_gb)
score_rf=f1_score(y_pred_rf,y_test,average='weighted')
score_dt=f1_score(y_pred_dt,y_test,average='weighted')
score_knn=f1_score(y_pred_knn,y_test,average='weighted')
score_gb=f1_score(y_pred_gb,y_test,average='weighted')
print('rf:',score_rf,'dt:',score_dt,'knn:',score_knn,'gb:',score_gb)
pd.crosstab(y_test,y_pred_gb)
plt.figure(figsize=(8,8))
sns.heatmap(pd.crosstab(y_test,y_pred_gb),annot=True,linewidth=True,fmt='f
')
plt.show()
import joblib
joblib.dump(pipe_gb,'model.pkl')
```