Hospital Readmission Prediction Using Machine Learning

presented by:

VEJJU KRISHNA SAI RUTHVIK
ATKURI SAI SURYA
BASSETTI ADINARAYANA
KAKUMANU GEETHA MADHAV

1. INTRODUCTION :

a) Overview:

If a hospital has multiple readmissions, it means that the hospital needs to work on the quality of services it is providing with respect to the health and wellness of its patients. Being able to predict whether a person will be readmitted to the hospital within 30 days or not, will be of great help to the hospital in developing an idea of the incoming number of repeated patients which in turn helps to provide better services for patients with increased risk of disease.

One patient population that is at increased risk of hospitalization and readmission is diabetes. Diabetes is a medical condition that affects approximately 1 in 10 patients in the United States. So in this project, we will be focusing on hospital readmission prediction for patients who are having diabetes.

This study used the Health Facts database (Cerner Corporation, Kansas City, MO), a national data warehouse that collects comprehensive clinical records across hospitals throughout the United States. The Health Facts data we used was an extract representing 10 years (1999–2008) of clinical care at 130 hospitals and integrated delivery networks throughout the United States.

b) Purpose:

The main purpose of this project is to predict whether a person who is suffering from diabetes and consulting a specific hospital will be readmitted or not, based on multiple factors.

We will be using classification algorithms such as Logistic Regression, KNN, Decision tree, Random Forest, AdaBoost, and GradientBoost. We will train and test the data with these algorithms. From this, the best model is selected and saved in pkl format. We will also be

deploying our model locally using Flask.

2. LITERATURE SURVEY:

Some of the existing problem:

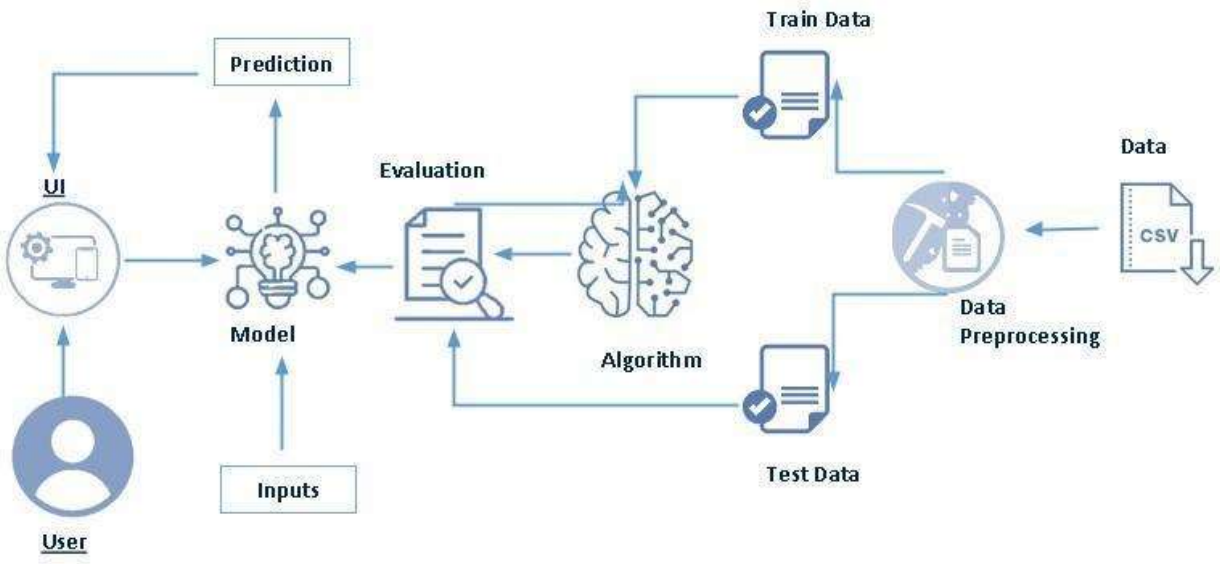
H2O.ai:
The mission at H2O.ai is to democratize AI for all so that more people across industries can use the power of AI to solve business and social challenges. The healthcare industry is a key focus for the company with an initiative to help develop AI healthcare solutions including dedicated, experienced resources for customers, driving healthcare AI events and meetups for healthcare professionals, and membership in Health IT Now, the leading coalition of patient groups, provider organizations, employers, insurers, and other stakeholders. H2O.ai is already working with top healthcare companies including Change Healthcare, Armada Health, Kaiser Permanente, and HCA, and its products include industry leading features for machine learning interpretability required by the healthcare industry for compliance purposes.

Proposed Solution:

The main purpose of this project is to predict whether a person who is suffering from diabetes and consulting a specific hospital will be readmitted or not, based on multiple factors.

3. THEORITICAL ANALYSIS:

a) Block Diagram:



b) Hard/Software Designing:

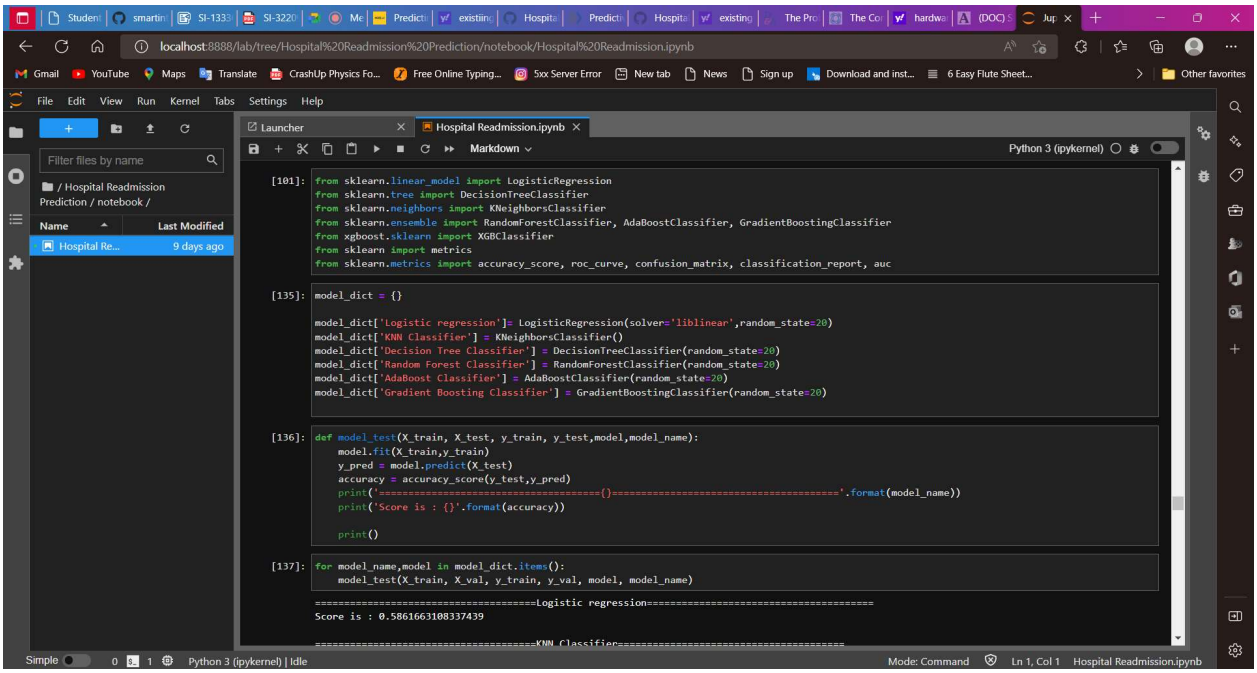
Operating System	Windows, Mac, Linux
CPU (for training)	Multi Core Processors (i3 or above/equivalent)
GPU (for training)	NVIDIA AI Capable / Google's TPU

Software Requirements:

Python	v3.10.0 or Abov
Python Packages	flask, tensorflow, opencv-python, keras, numpy, pandas, virtualenv, pillow
Web Browser	Mozilla Firefox, Google Chrome or any modern web browser

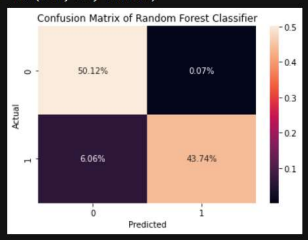
4. EXPERIMENTAL INVESTIGATIONS:

Training the train dataset:



```
[137]: for model_name,model in model_dict.items():
           model_test(X_train, X_val, y_train, y_val, model, model_name)

       ======================================Logistic regression=======================================
       Score is : 0.5861663108337439

       ========================================KNN Classifier========================================
       Score is : 0.7962719658400395

       ====================================Decision Tree Classifier====================================
       Score is : 0.8781956533639897

       ======================================Random Forest Classifier======================================
       Score is : 0.917309903103958

       ======================================AdaBoost Classifier=======================================
       Score is : 0.8183336070509662

       ====================================Gradient Boosting Classifier====================================
       Score is : 0.8846280177369026

[138]: for model_name, model in model_dict.items():
           p = model.predict(X_test)
           print('Testing accuracy of ',model_name,'=',accuracy_score(y_test,p))

       Testing accuracy of  Logistic regression = 0.6683777536034811
       Testing accuracy of  KNN Classifier = 0.6339067174326897
       Testing accuracy of  Decision Tree Classifier = 0.7998368234974164
       Testing accuracy of  Random Forest Classifier = 0.8569146042969812
       Testing accuracy of  AdaBoost Classifier = 0.858784335055752
       Testing accuracy of  Gradient Boosting Classifier = 0.8802352461245581

[144]: cf_matrix = confusion_matrix(y_val, pred_rfc)
       sb.heatmap(cf_matrix/np.sum(cf_matrix), annot=True, fmt='.2%')
```

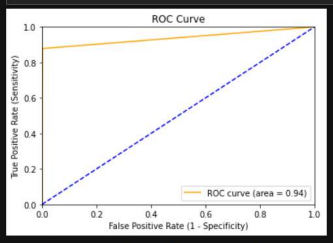
```
[144]: cf_matrix = confusion_matrix(y_val, pred_rfc)
       sb.heatmap(cf_matrix/np.sum(cf_matrix), annot=True, fmt='.2%')
       plt.title('Confusion Matrix of Random Forest Classifier')
       plt.xlabel('Predicted')
       plt.ylabel('Actual')

[144]: Text(33.0, 0.5, 'Actual')
```



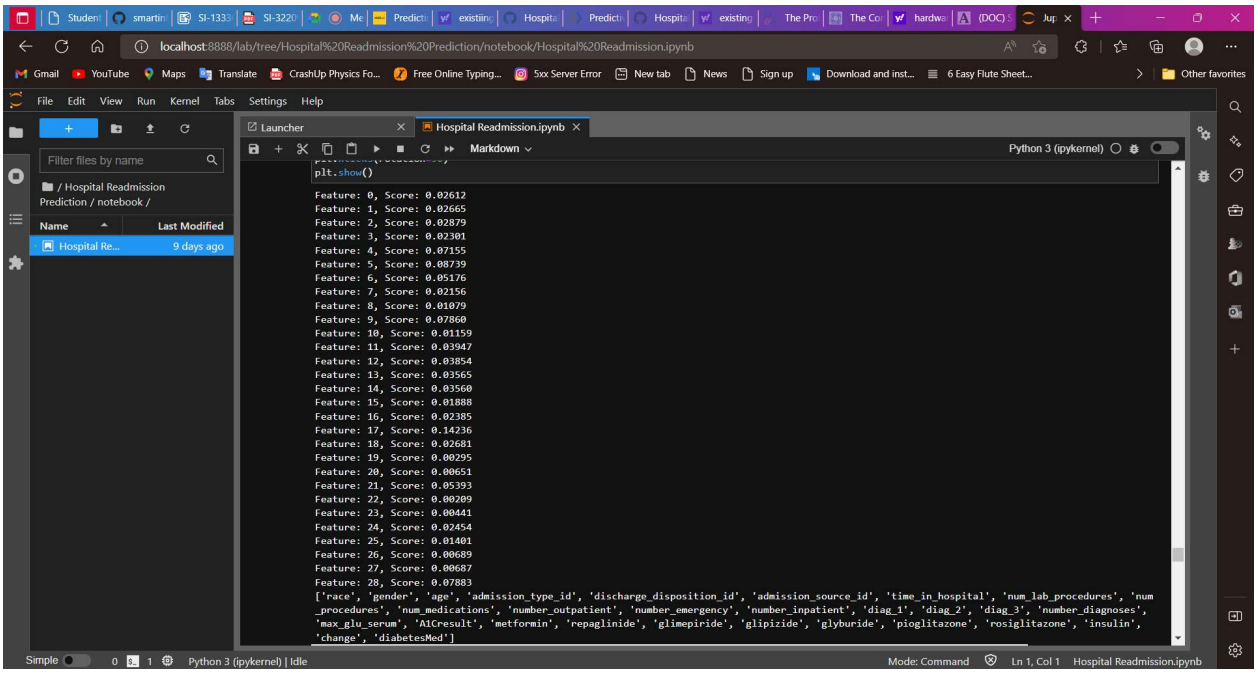

```
[146]: fpr_rfc, tpr_rfc, thresholds_rfc = roc_curve(y_val, pred_rfc)
       roc_auc_rfc = metrics.auc(fpr_rfc, tpr_rfc)
       plt.plot(fpr_rfc, tpr_rfc, color='orange', label='ROC curve (area = %0.2f)' % roc_auc_rfc)
       plt.plot([0, 1], [0, 1], color='blue', linestyle='--')
       plt.xlim([0.0, 1.0])
       plt.ylim([0.0, 1.0])
       plt.title('ROC Curve')
       plt.xlabel('False Positive Rate (1 - Specificity)')
       plt.ylabel('True Positive Rate (Sensitivity)')
       plt.legend(loc="lower right")
```

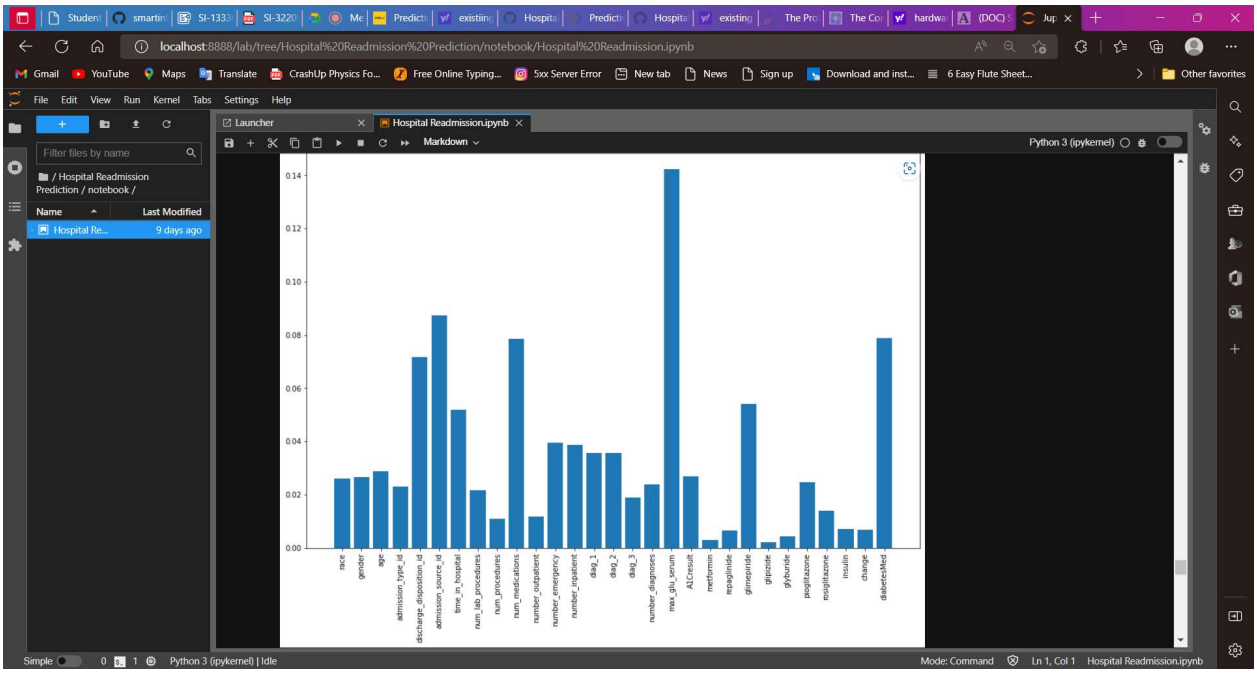
```
[106]: rfc = RandomForestClassifier(random_state=20)
       rfc.fit(X_train,y_train)
       pred_rfc = rfc.predict(X_val)
       print('Training Accuracy of Random Forest=',accuracy_score(y_val,pred_rfc))

       Training Accuracy of Random Forest= 0.9386325067060082

[108]: cols = list(data.columns)
       cols.remove('readmitted')

[109]: importance = rfc.feature_importances_
       for i,v in enumerate(importance):
           print('Feature: %0d, Score: %.5f' % (i,v))
       print(cols)
       plt.figure(figsize=(15,10))
       plt.bar([cols[x] for x in range(len(importance))], importance)
       plt.xticks(rotation=90)
       plt.show()

       Feature: 0, Score: 0.02612
       Feature: 1, Score: 0.02665
       Feature: 2, Score: 0.02879
       Feature: 3, Score: 0.02301
       Feature: 4, Score: 0.07155
       Feature: 5, Score: 0.08739
       Feature: 6, Score: 0.05176
       Feature: 7, Score: 0.02156
       Feature: 8, Score: 0.01079
       Feature: 9, Score: 0.07860
       Feature: 10, Score: 0.01159
       Feature: 11, Score: 0.03947
       Feature: 12, Score: 0.03854
       Feature: 13, Score: 0.03565
       Feature: 14, Score: 0.03560
       Feature: 15, Score: 0.01888
```

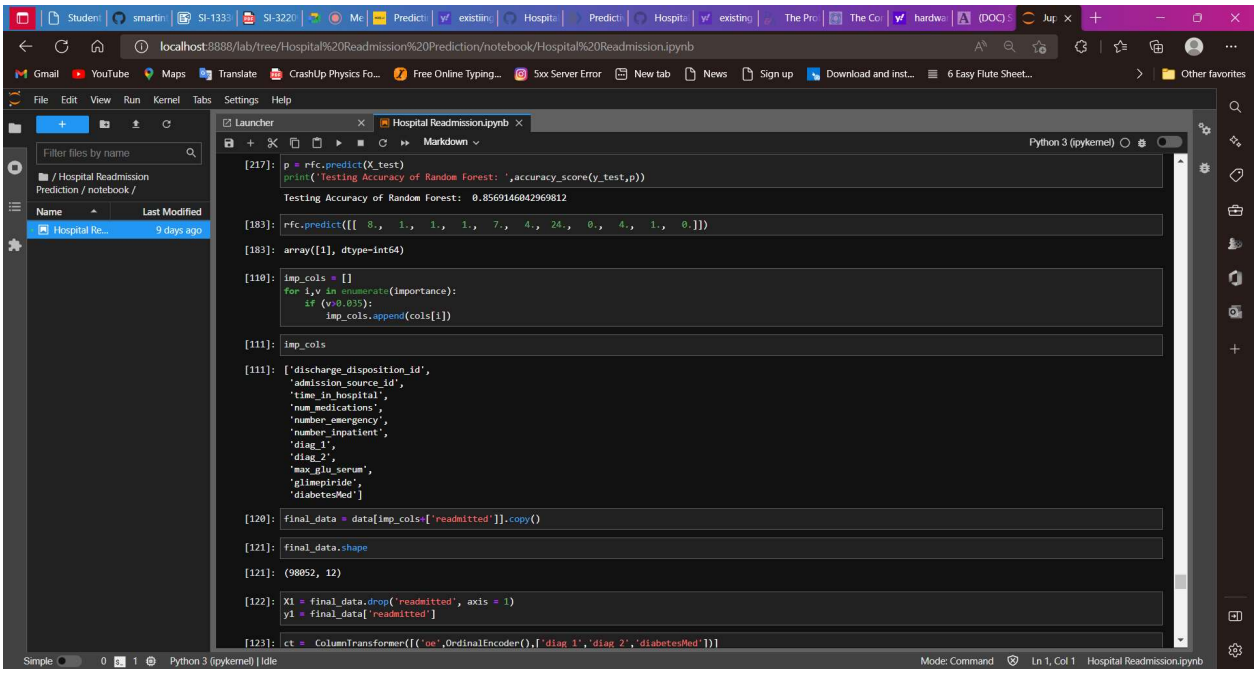
```
roc_curve(y_val, pred_rfc)
```



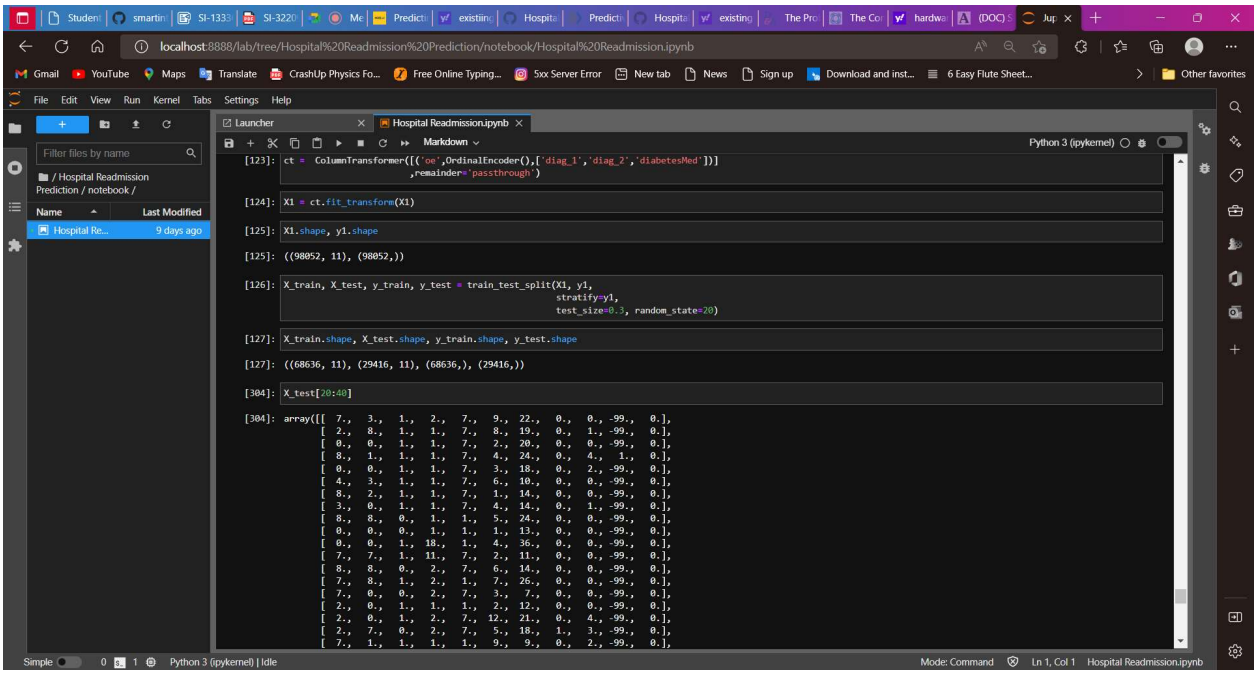

```
[146]: (array([0.        , 0.00147235, 1.        ]),
        array([0.        , 0.87826995, 1.        ]),
        array([2, 1, 0], dtype=int64))

[147]: print(classification_report(y_val,pred_rfc))

                   precision    recall  f1-score   support

                0       0.89      1.00      0.94     18338
                1       1.00      0.88      0.93     18196

         accuracy                           0.94     36534
        macro avg       0.95      0.94      0.94     36534
     weighted avg       0.94      0.94      0.94     36534

[106]: rfc = RandomForestClassifier(random_state=20)
```


localhost:8888/lab/tree/Hospital%20Readmission%20Prediction/notebook/Hospital%20Readmission.ipynb
Launcher
Hospital Readmission.ipynb
Markdown
Python 3 (ipykernel)
plt.show()
Feature: 0, Score: 0.02612
Feature: 1, Score: 0.02665
Feature: 2, Score: 0.02879
Feature: 3, Score: 0.02301
Feature: 4, Score: 0.07155
Feature: 5, Score: 0.08739
Feature: 6, Score: 0.05176
Feature: 7, Score: 0.02156
Feature: 8, Score: 0.01079
Feature: 9, Score: 0.07860
Feature: 10, Score: 0.01159
Feature: 11, Score: 0.03947
Feature: 12, Score: 0.03854
Feature: 13, Score: 0.03565
Feature: 14, Score: 0.03560
Feature: 15, Score: 0.01888
Feature: 16, Score: 0.02385
Feature: 17, Score: 0.14236
Feature: 18, Score: 0.02681
Feature: 19, Score: 0.00295
Feature: 20, Score: 0.00651
Feature: 21, Score: 0.05393
Feature: 22, Score: 0.00209
Feature: 23, Score: 0.00441
Feature: 24, Score: 0.02454
Feature: 25, Score: 0.01401
Feature: 26, Score: 0.00689
Feature: 27, Score: 0.00687
Feature: 28, Score: 0.07883
['race', 'gender', 'age', 'admission_type_id', 'discharge_disposition_id', 'admission_source_id', 'time_in_hospital', 'num_lab_procedures', 'num_procedures', 'num_medications', 'number_outpatient', 'number_emergency', 'number_inpatient', 'diag_1', 'diag_2', 'diag_3', 'number_diagnoses', 'max_glu_serum', 'A1Cresult', 'metformin', 'repaglinide', 'glimepiride', 'glipizide', 'glyburide', 'pioglitazone', 'rosiglitazone', 'insulin', 'change', 'diabetesMed']
Filter files by name
/ Hospital Readmission Prediction / notebook /
Name
Last Modified
Hospital Re...
9 days ago
Simple
Python 3 (ipykernel) | Idle
Mode: Command
Ln 1, Col 1
Hospital Readmission.ipynb

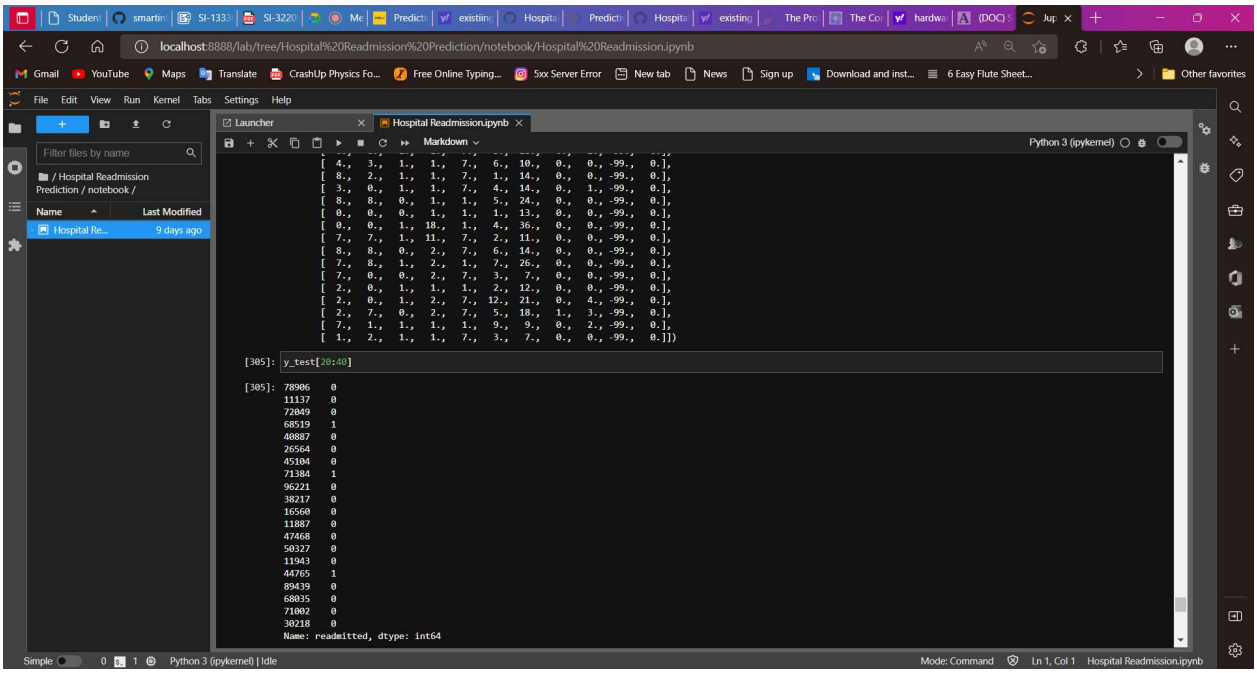

localhost:8888/lab/tree/Hospital%20Readmission%20Prediction/notebook/Hospital%20Readmission.ipynb
Launcher
Hospital Readmission.ipynb
Markdown
Python 3 (ipykernel)
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
race
gender
age
admission_type_id
discharge_disposition_id
admission_source_id
time_in_hospital
num_lab_procedures
num_procedures
num_medications
number_outpatient
number_emergency
number_inpatient
diag_1
diag_2
diag_3
number_diagnoses
max_glu_serum
A1Cresult
metformin
repaglinide
glimepiride
glipizide
glyburide
pioglitazone
rosiglitazone
insulin
change
diabetesMed
Filter files by name
/ Hospital Readmission Prediction / notebook /
Name
Last Modified
Hospital Re...
9 days ago
Simple
Python 3 (ipykernel) | Idle
Mode: Command
Ln 1, Col 1
Hospital Readmission.ipynb



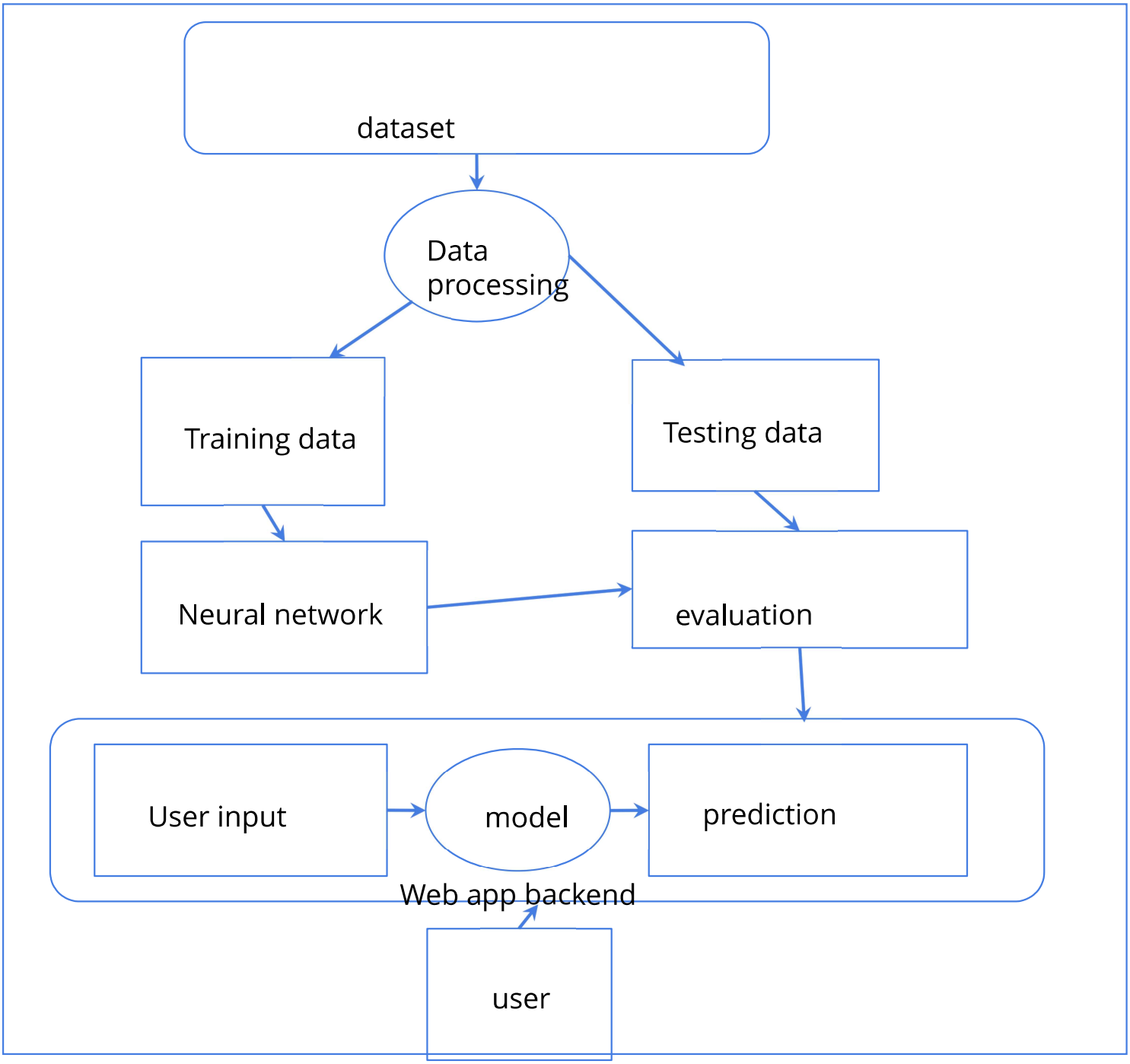
```
[217]: p = rfc.predict(X_test)
       print('Testing Accuracy of Random Forest: ',accuracy_score(y_test,p))

       Testing Accuracy of Random Forest:  0.8569146042969812

[183]: rfc.predict([[ 8.,  1.,  1.,  1.,  7.,  4., 24.,  0.,  4.,  1.,  0.]])

[183]: array([1], dtype=int64)

[110]: imp_cols = []
       for i,v in enumerate(importance):
           if (v>0.035):
               imp_cols.append(cols[i])

[111]: imp_cols

[111]: ['discharge_disposition_id',
        'admission_source_id',
        'time_in_hospital',
        'num_medications',
        'number_emergency',
        'number_inpatient',
        'diag_1',
        'diag_2',
        'max_glu_serum',
        'glimepiride',
        'diabetesMed']

[120]: final_data = data[imp_cols+['readmitted']].copy()

[121]: final_data.shape

[121]: (98052, 12)

[122]: X1 = final_data.drop('readmitted', axis = 1)
       y1 = final_data['readmitted']

[123]: ct =  ColumnTransformer([('oe',OrdinalEncoder(),['diag_1','diag_2','diabetesMed'])]
```



```
[123]: ct =  ColumnTransformer([('oe',OrdinalEncoder(),['diag_1','diag_2','diabetesMed'])]
                               ,remainder='passthrough')

[124]: X1 = ct.fit_transform(X1)

[125]: X1.shape, y1.shape

[125]: ((98052, 11), (98052,))

[126]: X_train, X_test, y_train, y_test = train_test_split(X1, y1,
                                                           stratify=y1,
                                                           test_size=0.3, random_state=20)

[127]: X_train.shape, X_test.shape, y_train.shape, y_test.shape

[127]: ((68636, 11), (29416, 11), (68636,), (29416,))

[304]: X_test[20:40]

[304]: array([[ 7.,  3.,  1.,  2.,  7.,  9., 22.,  0.,  0., -99.,  0.],
               [ 2.,  8.,  1.,  1.,  7.,  8., 19.,  0.,  1., -99.,  0.],
               [ 0.,  0.,  1.,  1.,  7.,  2., 20.,  0.,  0., -99.,  0.],
               [ 8.,  1.,  1.,  1.,  7.,  4., 24.,  0.,  4.,  1.,  0.],
               [ 0.,  0.,  1.,  1.,  7.,  3., 18.,  0.,  2., -99.,  0.],
               [ 4.,  3.,  1.,  1.,  7.,  6., 10.,  0.,  0., -99.,  0.],
               [ 8.,  2.,  1.,  1.,  7.,  1., 14.,  0.,  0., -99.,  0.],
               [ 3.,  0.,  1.,  1.,  7.,  4., 14.,  0.,  1., -99.,  0.],
               [ 8.,  8.,  0.,  1.,  1.,  5., 24.,  0.,  0., -99.,  0.],
               [ 0.,  0.,  0.,  1.,  1.,  1., 13.,  0.,  0., -99.,  0.],
               [ 0.,  0.,  1., 18.,  1.,  4., 36.,  0.,  0., -99.,  0.],
               [ 7.,  7.,  1., 11.,  7.,  2., 11.,  0.,  0., -99.,  0.],
               [ 8.,  8.,  0.,  2.,  7.,  6., 14.,  0.,  0., -99.,  0.],
               [ 7.,  8.,  1.,  2.,  1.,  7., 26.,  0.,  0., -99.,  0.],
               [ 7.,  0.,  0.,  2.,  7.,  3.,  7.,  0.,  0., -99.,  0.],
               [ 2.,  0.,  1.,  1.,  1.,  2., 12.,  0.,  0., -99.,  0.],
               [ 2.,  0.,  1.,  2.,  7., 12., 21.,  0.,  4., -99.,  0.],
               [ 2.,  7.,  0.,  2.,  7.,  5., 18.,  1.,  3., -99.,  0.],
               [ 7.,  1.,  1.,  1.,  1.,  9.,  9.,  0.,  2., -99.,  0.],
```


localhost:8888/lab/tree/Hospital%20Readmission%20Prediction/notebook/Hospital%20Readmission.ipynb
File Edit View Run Kernel Tabs Settings Help
Launcher
Hospital Readmission.ipynb
Markdown
Python 3 (ipykernel)
Filter files by name
/ Hospital Readmission Prediction / notebook /
Name
Last Modified
Hospital Re...
9 days ago
[4., 3., 1., 1., 7., 6., 10., 0., 0., -99., 0.],
[8., 2., 1., 1., 7., 1., 14., 0., 0., -99., 0.],
[3., 0., 1., 1., 7., 4., 14., 0., 1., -99., 0.],
[8., 8., 0., 1., 1., 5., 24., 0., 0., -99., 0.],
[0., 0., 0., 1., 1., 1., 13., 0., 0., -99., 0.],
[0., 0., 1., 18., 1., 4., 36., 0., 0., -99., 0.],
[7., 7., 1., 11., 7., 2., 11., 0., 0., -99., 0.],
[8., 8., 0., 2., 7., 6., 14., 0., 0., -99., 0.],
[7., 8., 1., 2., 1., 7., 26., 0., 0., -99., 0.],
[7., 0., 0., 2., 7., 3., 7., 0., 0., -99., 0.],
[2., 0., 1., 1., 1., 2., 12., 0., 0., -99., 0.],
[2., 0., 1., 2., 7., 12., 21., 0., 4., -99., 0.],
[2., 7., 0., 2., 7., 5., 18., 1., 3., -99., 0.],
[7., 1., 1., 1., 1., 9., 9., 0., 2., -99., 0.],
[1., 2., 1., 1., 7., 3., 7., 0., 0., -99., 0.]])
[305]: y_test[20:40]
[305]: 78906 0
11137 0
72049 0
68519 1
40887 0
26564 0
45104 0
71384 1
96221 0
38217 0
16560 0
11887 0
47468 0
50327 0
11943 0
44765 1
89439 0
68035 0
71002 0
30218 0
Name: readmitted, dtype: int64
Simple
Python 3 (ipykernel) | Idle
Mode: Command
Ln 1, Col 1
Hospital Readmission.ipynb

5. FLOWCHART:

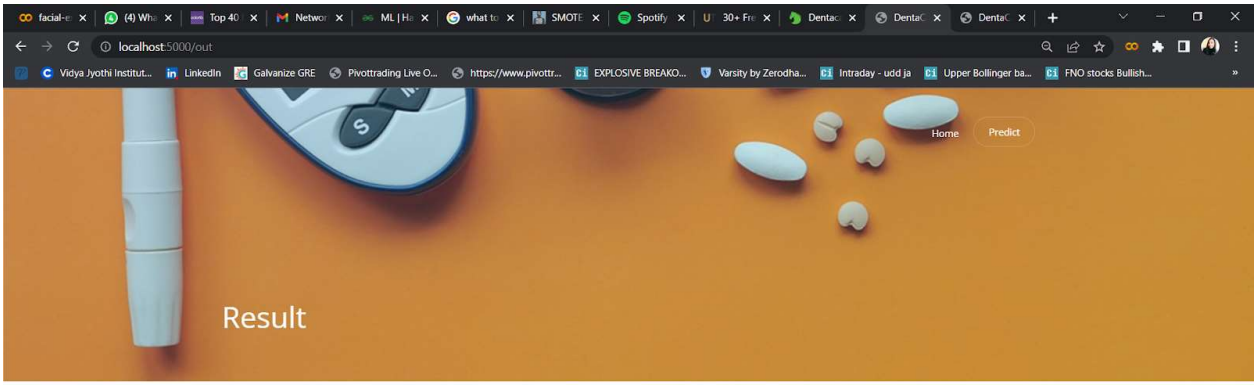

dataset
Data processing
Training data
Testing data
Neural network
evaluation
User input
model
prediction
Web app backend
user

6. RESULT:

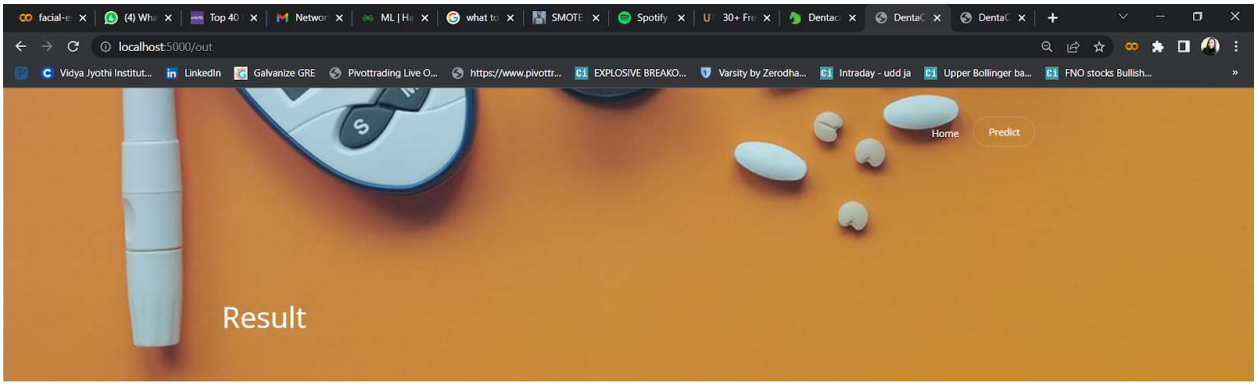
The problem associated is a binary classifcation. Since the target variable in the dataset is quite imbalanced, SMOTE algorithm was used to oversample the minority class.

A number of models were used such as logistic regression, KNN Classifier, Decision Tree Classifier, Random Tree Classifier, Adaboost Classifier etc. We will train and test the data with these algorithms. From this, the best model is selected and saved in pkl format. We will also be deploying our model locally using Flask. Out of all the models, random forest classifier is found to perform the best.

The accuracy for Random Forest Classifier is 91.73 percent.



This patient will be readmitted



This patient will not be readmitted

7. ADVANTAGES AND DISADVANTAGES:

Advantages:

1. The hospital will be able to be ready for any number of patients that will be readmitted in the future.

Disadvantage:

1. The accuracy is only 91.73 percentage.
2. It will not be able to predict 100% accurately.

Applications:

1. It can be used by clinics realted to hospitals
2. It can be used by diabetes hospitals
3. It can be used by diabetic patients

7 CONCLUSION

In this work we adopted machine learning methods to identify high risk patients and evaluated different machine learning algorithms. Compared to the previous analysis, our study achieved high accuracy due to the sophisticated pre processing procedure. The XGBoost method is reported to be the best method for prediction of the readmission rate for diabetes patients.

We identified the most important factors as the time_in_hospital and number of inpatient, number of diagnosis, which appears to associate with the severity of the disease. Further studies could conduct more exploration when analyzing these factors individually.

Future Scope:

Having a technology that can predict the number of people that will be readmitted in the future will help the hospitals in being ready for for any number of patients that will be readmitted.

Bibilography:

- ML Concepts

Supervised learning: https://www.youtube.com/watch?v=QeKshry8pWQ

Unsupervised learning:
https://www.youtube.com/watch?v=D6gtZrsYi6c&feature=emb_logo

Metrics: https://www.youtube.com/watch?v=aWAnNHXIKww&t=1s

Flask : https://www.youtube.com/watch?v=lj4I_CvBnt0&feature=emb_logo

APPENDIX:

```
from sklearn.model_selection import train_test_split
X_train, X_test, y_train, y_test = train_test_split(X, y,
                                                    stratify=y,
                                                    test_size=0.3,random_state=20)
```

```
X_train.shape, X_test.shape, y_train.shape, y_test.shape
```

```
((68636, 29), (29416, 29), (68636,), (29416,))
```

```
from imblearn.over_sampling import SMOTE
```

```
sm = SMOTE(random_state = 20)
X1_res, y1_res = sm.fit_sample(X_train,y_train)
```

```
X1_res = np.array(X1_res)
```

```
y1_res = np.array(y1_res)
```

```

X_train, X_val, y_train, y_val = train_test_split(X1_res,y1_res,test_size=0.3, random_state=20)
```

```
X_train.shape, X_val.shape, y_train.shape, y_val.shape
```

```
((85246, 29), (36534, 29), (85246,), (36534,))
```

```
from imblearn.over_sampling import SMOTE
```

```
sm = SMOTE(random_state = 20)
X1_res, y1_res = sm.fit_sample(X_train,y_train)
```

```
X1_res = np.array(X1_res)
```

```
y1_res = np.array(y1_res)
```

```

X_train, X_val, y_train, y_val = train_test_split(X1_res,y1_res,test_size=0.3, random_state=20)
```

```
X_train.shape, X_val.shape, y_train.shape, y_val.shape
```

```
((85246, 29), (36534, 29), (85246,), (36534,))
```

```
from sklearn.linear_model import LogisticRegression
from sklearn.tree import DecisionTreeClassifier
from sklearn.neighbors import KNeighborsClassifier
from sklearn.ensemble import RandomForestClassifier, AdaBoostClassifier, GradientBoostingClassifier
from xgboost.sklearn import XGBClassifier
from sklearn import metrics
from sklearn.metrics import accuracy_score, roc_curve, confusion_matrix, classification_report, auc
```

```
#Creating a dictionary of all models
model_dict = {}

model_dict['Logistic regression']= LogisticRegression(solver='liblinear',random_state=20)
model_dict['KNN Classifier'] = KNeighborsClassifier()
model_dict['Decision Tree Classifier'] = DecisionTreeClassifier(random_state=20)
model_dict['Random Forest Classifier'] = RandomForestClassifier(random_state=20)
model_dict['AdaBoost Classifier'] = AdaBoostClassifier(random_state=20)
model_dict['Gradient Boosting Classifier'] = GradientBoostingClassifier(random_state=20)
model_dict['XGB Classifier'] = XGBClassifier(random_state=20)
```

```python
#function to print accuracy of all models
def model_test(X_train, X_test, y_train, y_test,model,model_name):
    model.fit(X_train,y_train)
    y_pred = model.predict(X_test)
    accuracy = accuracy_score(y_test,y_pred)
    print('======================================{}======================================'.format(model_name))
    print('Score is : {}'.format(accuracy))

    print()
```

```python
for model_name,model in model_dict.items():
    model_test(X_train, X_val, y_train, y_val, model, model_name)
```

```
======================================Logistic regression======================================
Score is : 0.6002080254009964

======================================KNN Classifier======================================
Score is : 0.7935621612744292

======================================Decision Tree Classifier======================================
Score is : 0.8801664203207971

======================================Random Forest Classifier======================================
Score is : 0.938632506706082

======================================AdaBoost Classifier======================================
Score is : 0.9139158044561231

======================================Gradient Boosting Classifier======================================
Score is : 0.9317621941205453

======================================XGB Classifier======================================
Score is : 0.9375376361745223
```

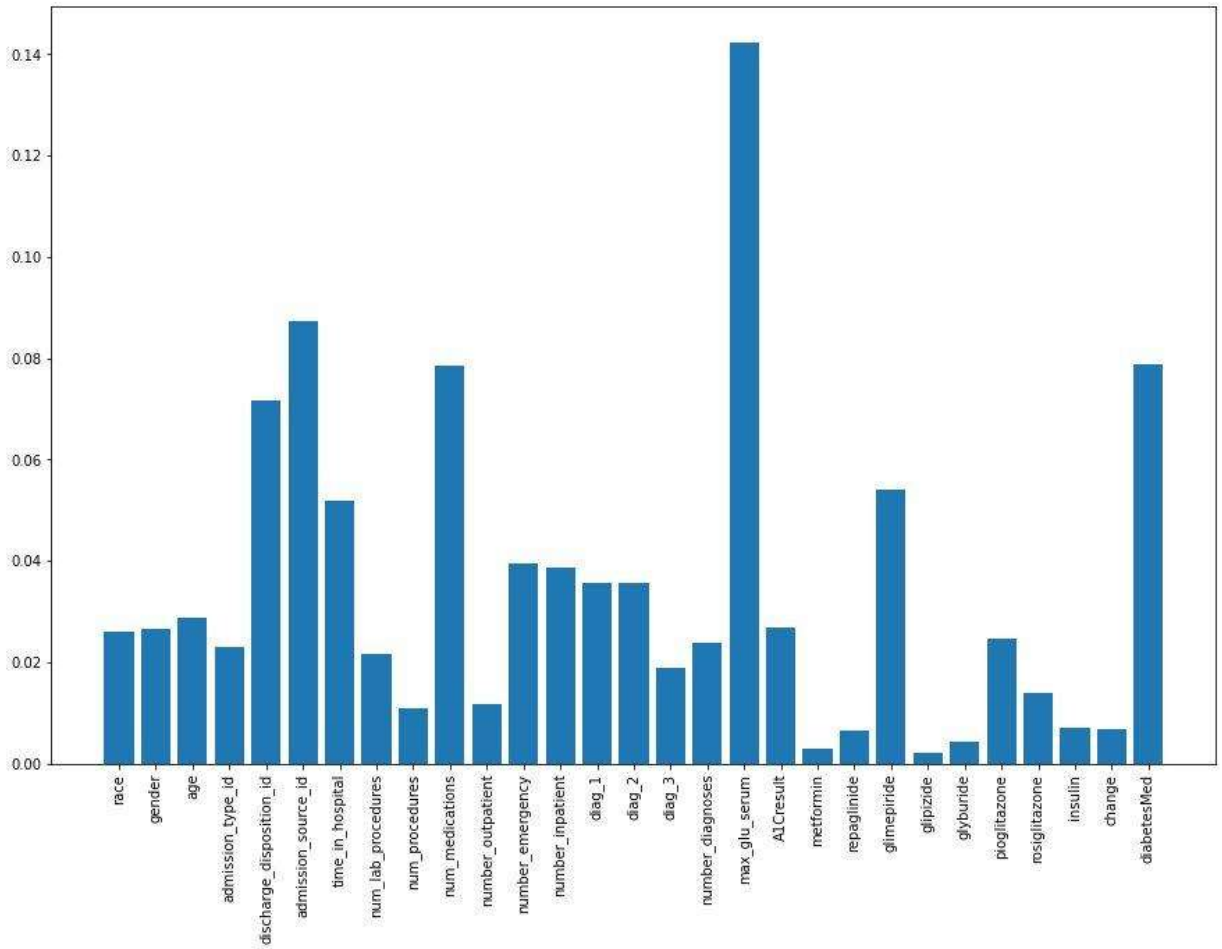
```python
for model_name, model in model_dict.items():
    p = model.predict(X_test)
    print('Testing accuracy of ',model_name,'=',accuracy_score(y_test,p))
```

```
Testing accuracy of  Logistic regression = 0.6484226271416916
Testing accuracy of  KNN Classifier = 0.5848517813434866
Testing accuracy of  Decision Tree Classifier = 0.7861028011966277
Testing accuracy of  Random Forest Classifier = 0.8871702474843622
Testing accuracy of  AdaBoost Classifier = 0.8744220832200164
Testing accuracy of  Gradient Boosting Classifier = 0.8873742181125918
Testing accuracy of  XGB Classifier = 0.8872382376937721
```

```python
#Fitting data to Random Forest classifier
rfc = RandomForestClassifier(random_state=20)
rfc.fit(X_train,y_train)
pred_rfc = rfc.predict(X_val)
print('Training Accuracy of Random Forest=',accuracy_score(y_val,pred_rfc))
```

```
Training Accuracy of Random Forest= 0.938632506706082
```

```
importance = rfc.feature_importances_
# summarize feature importance
for i,v in enumerate(importance):
    print('Feature: %0d, Score: %.5f' % (i,v))
# plot feature importance
print(cols)
plt.figure(figsize=(15,10))
plt.bar([cols[x] for x in range(len(importance))], importance)
plt.xticks(rotation=90)
plt.show()
```



```
imp_cols = []
for i,v in enumerate(importance):
    if (v>0.035):
        imp_cols.append(cols[i])
```

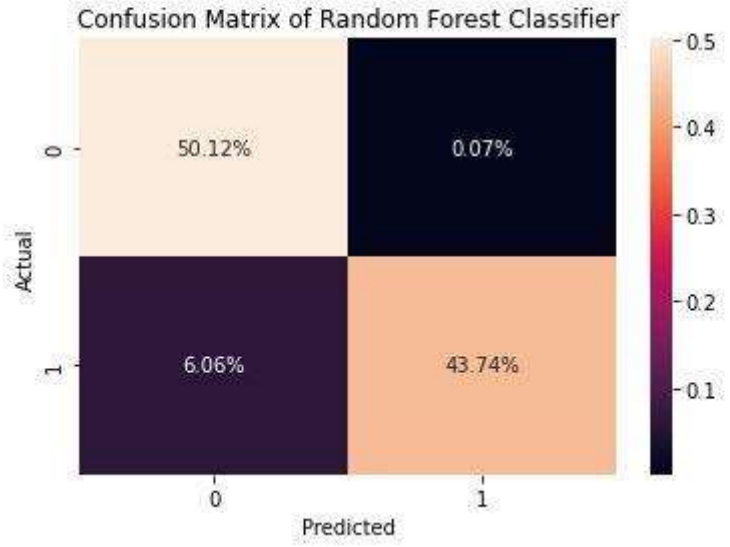
```
imp_cols
```

```
['discharge_disposition_id',
 'admission_source_id',
 'time_in_hospital',
 'num_medications',
 'number_emergency',
 'number_inpatient',
 'diag_1',
 'diag_2',
 'max_glu_serum',
 'glimepiride',
 'diabetesMed']
```

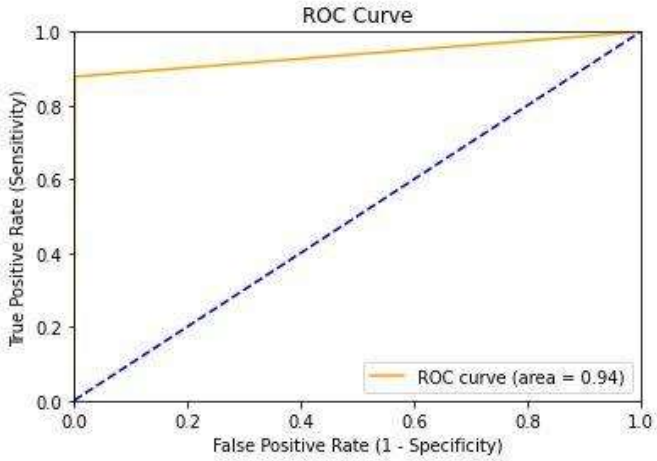
Confusion matrix:

```
#Plotting confusion matrix
cf_matrix = confusion_matrix(y_val, pred_rfc)
sb.heatmap(cf_matrix/np.sum(cf_matrix), annot=True, fmt='.2%')
plt.title('Confusion Matrix of Random Forest Classifier')
plt.xlabel('Predicted')
plt.ylabel('Actual')
```

```
Text(33.0, 0.5, 'Actual')
```



```python
#Plotting ROC curve
fpr_rfc, tpr_rfc, thresholds_rfc = roc_curve(y_val, pred_rfc)
roc_auc_rfc = metrics.auc(fpr_rfc, tpr_rfc)
plt.plot(fpr_rfc, tpr_rfc, color='orange', label='ROC curve (area = %0.2f)' % roc_auc_rfc)
plt.plot([0, 1], [0, 1], color='blue', linestyle='--')
plt.xlim([0.0, 1.0])
plt.ylim([0.0, 1.0])
plt.title('ROC Curve')
plt.xlabel('False Positive Rate (1 - Specificity)')
plt.ylabel('True Positive Rate (Sensitivity)')
plt.legend(loc="lower right")
plt.show()
roc_curve(y_val, pred_rfc)
```




```python
print(classification_report(y_val,pred_rfc))
```

```
              precision    recall  f1-score   support

           0       0.89      1.00      0.94     18338
           1       1.00      0.88      0.93     18196

    accuracy                           0.94     36534
   macro avg       0.95      0.94      0.94     36534
weighted avg       0.94      0.94      0.94     36534
```

```python
import pickle
pickle.dump(rfc,open('model.pkl','wb'))
```

Web application in Flask:

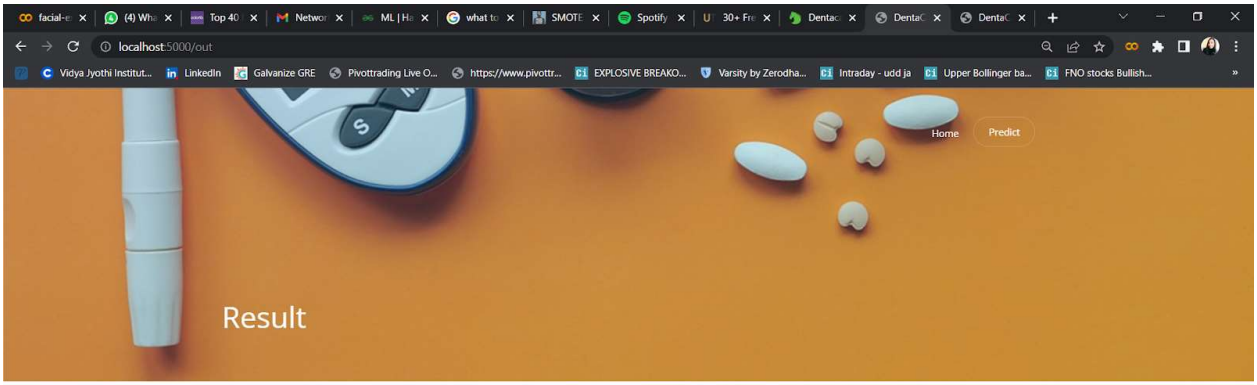
```
from flask import Flask, render_template, request
import pickle, joblib
import pandas as pd

app = Flask(__name__)

model = pickle.load(open("model.pkl","rb"))
ct = joblib.load('feature_values')
```

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

@app.route('/pred')
def predict():
    return render_template("index.html")
```

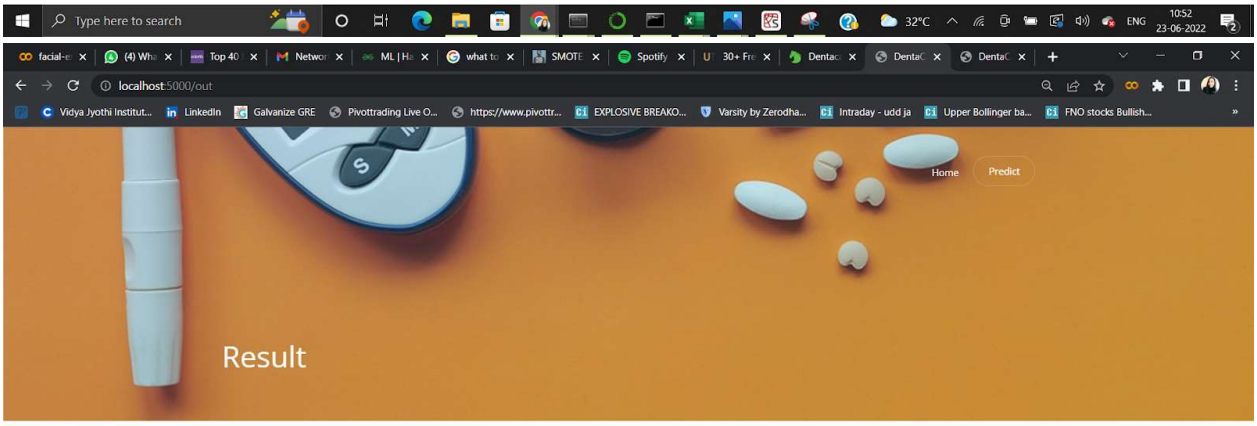
```
@app.route('/out', methods =["POST"])
def output():
    discharge_disposition_id = request.form["discharge_disposition_id"]
    admission_source_id = request.form["admission_source_id"]
    time_in_hospital = request.form["time_in_hospital"]
    num_medications = request.form["num_medications"]
    number_emergency = request.form["number_emergency"]
    number_inpatient = request.form["number_inpatient"]
    diag_1 = request.form["diag_1"]
    diag_2 = request.form["diag_2"]
    max_glu_serum = request.form["max_glu_serum"]
    glimepiride = request.form["glimepiride"]
    diabetesMed = request.form["diabetesMed"]

    if max_glu_serum == '>200' or max_glu_serum=='>300':
        max_glu_serum=1
    elif max_glu_serum=='Norm':
        max_glu_serum=0
    else:
        max_glu_serum=-99

    if glimepiride == 'No':
        glimepiride = 0
    else:
        glimepiride=1



    data = [[discharge_disposition_id,admission_source_id,time_in_hospital,
             num_medications, number_emergency, number_inpatient,diag_1, diag_2,
             max_glu_serum, glimepiride, diabetesMed]]


    feature_cols = ['discharge_disposition_id', 'admission_source_id', 'time_in_hospital',
                    'num_medications','number_emergency', 'number_inpatient',
                    'diag_1', 'diag_2','max_glu_serum', 'glimepiride', 'diabetesMed']

    pred = model.predict(ct.transform(pd.DataFrame(data,columns=feature_cols)))
    pred = pred[0]
    if pred:
        return render_template("output.html",y="This patient will be readmitted ")
    else:
        return render_template("output.html",y="This patient will not be readmitted")

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

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

Output:



This patient will be readmitted



This patient will not be readmitted