User guide for Customer Risk Prediction Model

Prerequisites

1) Python 3
2) Jupyter Notebook
3) Visual Studio Code
4) GitHub Account

Section 1: Data Cleaning

1) Open Jupyter notebook
2) Open Data Cleaning.ipynb
3) Uncomment and run the first cell to install pandas, numpy, sklearn.

```
# !pip install pandas
# !pip install numpy
# !pip install sklearn
```

4) Run the 2nd cell to import pandas and numpy packages

```
import pandas as pd
import numpy as np
```

5) Import the dataset to be cleaned. Change "train.csv" to the name of the file. This can be both the training file and testing file.
   Note that for training file, labels need to be the at the right most column

```
# labels to be last column
df = pd.read_csv("train.csv")
```

6) Continue running the cells to check how many records there are in the dataset
7) Check number of columns that contain more than 70% NA values and drop these columns.

Remove NA

In [15]:
```
# drop those col with > 70% missing values
pct_null = df.isnull().sum() / len(df)
missing_features = pct_null[pct_null > 0.70].index
df.drop(missing_features, axis=1, inplace=True)
```

In [16]:
```
missing_features   #columns that are dropped
```

Out[16]:
```
Index(['Family_Hist_5', 'Medical_History_10', 'Medical_History_15',
       'Medical_History_24', 'Medical_History_32'],
      dtype='object')
```

In [17]:
```
#input missing values with mean for the other columns
df = df.fillna(df.mean())
```

8) Check for features that are unique to each customer using df.nunique()
    a. If there are features that are unique to each customer for eg. ID, remove them by running the next 2 cells.

Remove columns where features are unique to each customer

In [19]:
```
# get columns where nunique count == no. of records in dataframe
columns = []
for col in df.columns:
    if len(df[col]) == len(df[col].unique()):
        columns.append(col)

columns
```

Out[19]: ['Id']

In [20]:
```
# drop columns where nunique num == no. of records --> meaning columns where it is unique to each customer for eg. Id
df = df.drop(columns = columns)
```

9) Check that columns that have values unique to each customer has been removed by running the next cell, df.head()
10) Next, we will perform label encoding to convert textual data to numerical data as machine learning models cannot work with text.
    a. Add all the columns name into a list called feature

```
# add all column names into a list called 'feature'
feature=[]
for col in df.columns:
    feature.append(col)
```

    b. Check value type of each column

```
#check value type for each column
for i in feature:
    print(df[i].dtypes)
```

    c. If value type of column is ‘object’, it will be encoded into numbers

```
# if valye type is object (which means it is a text), label encoder will encode it into numerical values
from sklearn.preprocessing import LabelEncoder
lb_style = LabelEncoder()

for i in feature:
    if df[i].dtypes=='object':
        df[i]= lb_style.fit_transform(df[i])
```

    d. Check that textual values have been changed into numbers by running df.head()
11) Export the cleaned dataset. You can rename the file to any name you prefer by changing ‘train_cleaned’. Note that the extension ‘.csv’ is required.

```
df.to_csv('train_cleaned.csv', index=False)
```

12) Perform data cleaning for both the train and test file and export them so that they can be uploaded into the flask model (Section 3).

Section 2: Model optimization (Get hyperparameters for models)

1) Open Model Optimisation.ipynb
2) Run the first box to import required modules
3) Import only the training file that you have just cleaned. (Change the filename accordingly if you saved it in another name)

```
In [5]:
df_train = pd.read_csv("train_cleaned.csv")
df_train.head()
```

Out[5]:

	Product_Info_1	Product_Info_2	Product_Info_3	Product_Info_4	Product_Info_5	Product_Info_6	Product_Info_7	Ins_Age	Ht	Wt	...	Medical_K
0	1	16	10	0.076923	2	1	1	0.641791	0.581818	0.148536	...	
1	1	0	26	0.076923	2	3	1	0.059701	0.600000	0.131799	...	
2	1	18	26	0.076923	2	3	1	0.029851	0.745455	0.288703	...	
3	1	17	10	0.487179	2	3	1	0.164179	0.672727	0.205021	...	
4	1	15	26	0.230769	2	3	1	0.417910	0.654545	0.234310	...	

5 rows × 122 columns

4) Run all the codes below
5) After running all the codes, go to 'Optimisation' section of each model (as shown below).

Adaboost model

```
In [126]:
classifier = AdaBoostClassifier(
    DecisionTreeClassifier(max_depth=2),
    n_estimators=200
)
classifier.fit(x_sm, y_sm)

print(classification_report(y_validate, classifier.predict(x_validate),digits=4))
```

```
              precision    recall  f1-score   support

           0     0.7134    0.7578    0.7349      3150
           1     0.8760    0.8490    0.8623      6351

    accuracy                         0.8188      9501
   macro avg     0.7947    0.8034    0.7986      9501
weighted avg     0.8221    0.8188    0.8201      9501
```

Optimisation

```
In [34]:
param_grid ={
    "n_estimators": [125,175,225,300],
    "learning_rate": [0.001, 0.01, 0.1,0.3]
}
grid = GridSearchCV(AdaBoostClassifier(DecisionTreeClassifier(max_depth=2),
    n_estimators=200),param_grid=param_grid)
grid_result=grid.fit(x_sm, y_sm)
```

```
In [128]:
print("Best: %f using %s" % (grid_result.best_score_, grid_result.best_params_))
```

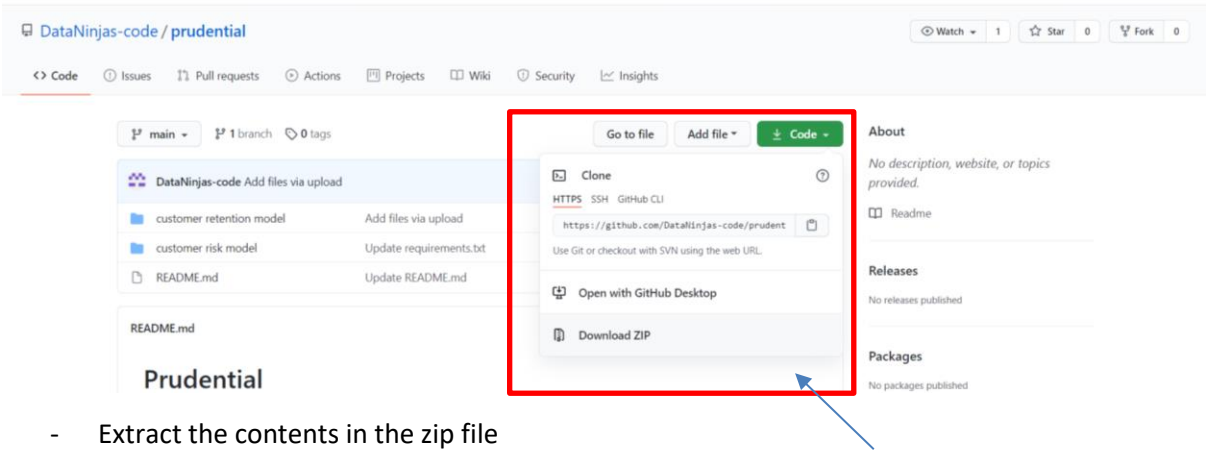
```
Best: 0.855936 using {'learning_rate': 0.3, 'n_estimators': 225}
```

For each of the models, get the best hyperparameters.
For eg. For Adaboost, the best parameters are { 'learning_rate': 0.3, 'n_estimators': 225}

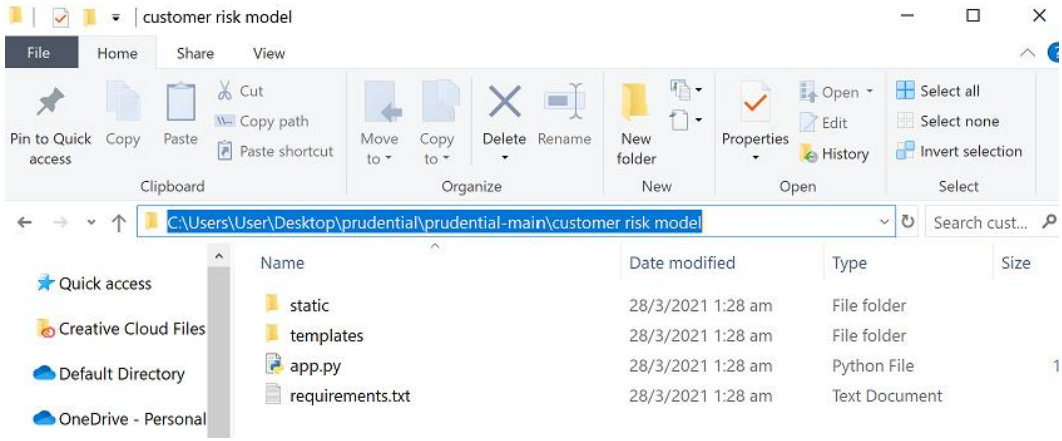
6) Take note of all these best hyperparameters as you need to replace them in the flask application in the next section.

Section 3: Customer Risk Prediction Model

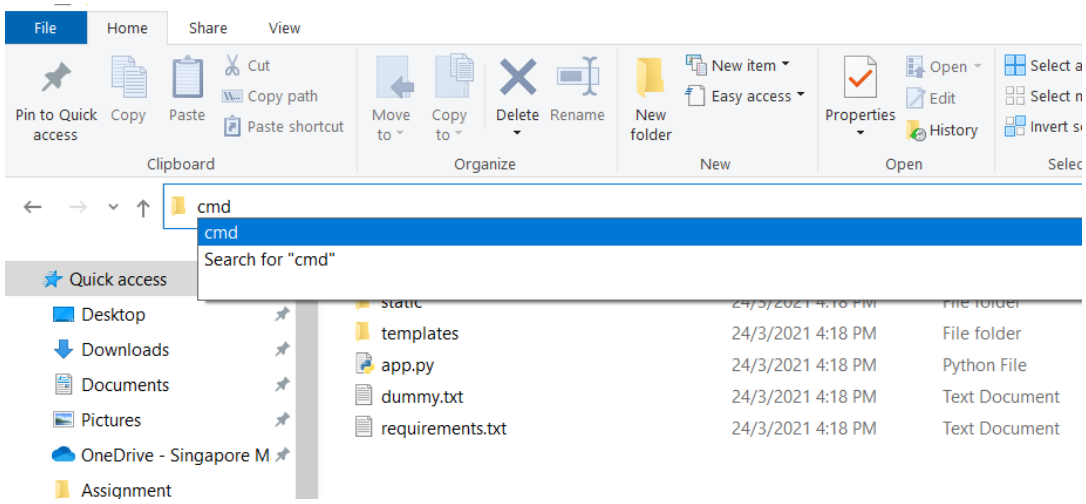
1) Go to https://github.com/DataNinjas-code/prudential
   - Download the file



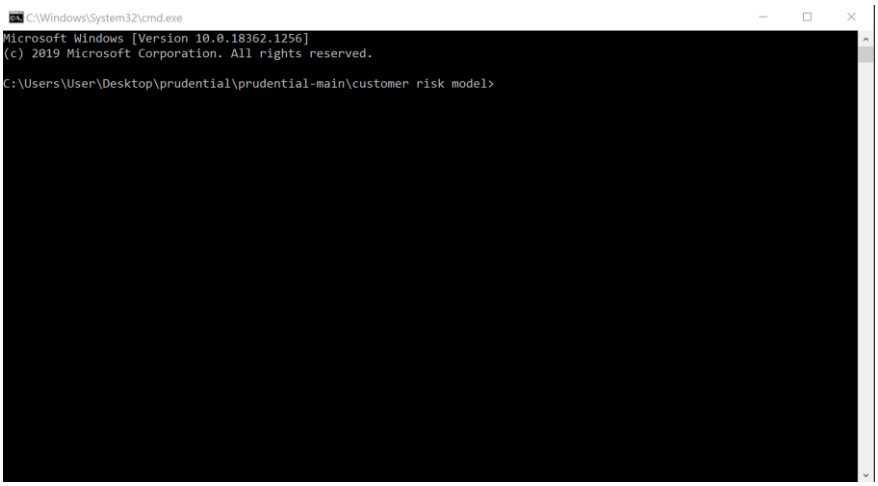
- Extract the contents in the zip file
- Click into the folder and into 'customer retention model'
- Click on the file path (make sure your file path is similar to the screenshot)



- Type 'cmd' and press enter



- You will see your command prompt being opened up.



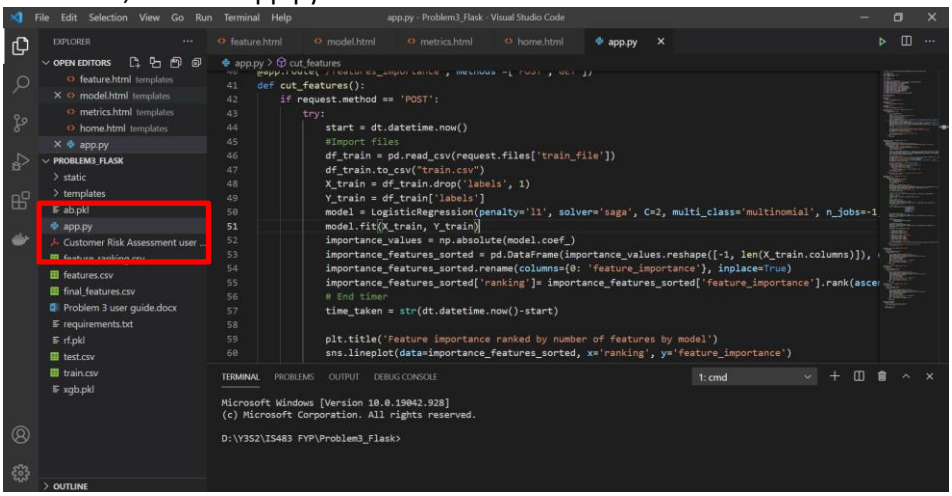
2) Pip install all the required packages using requirements.txt

```
pip install -r path/to/requirements.txt
```

```
C:\Users\User>cd C:\Users\User\Prudential\customer risk model

C:\Users\User\prudential\customer risk model>pip install -r requirements.txt
```

3) Open the flask application in Visual Studio Code or any Software Development Environment. If you are using Visual Studio Code, follow the instructions below:
   a. Open Visual Studio Code
   b. Click on File > Open Folder > Select the 'customer risk model' folder
   c. At the left, click on 'app.py'



   d. Inside app.py, change the hyperparameters of the 3 models (XGBoost, Random Forest and Adaboost) which was derived in section 2.

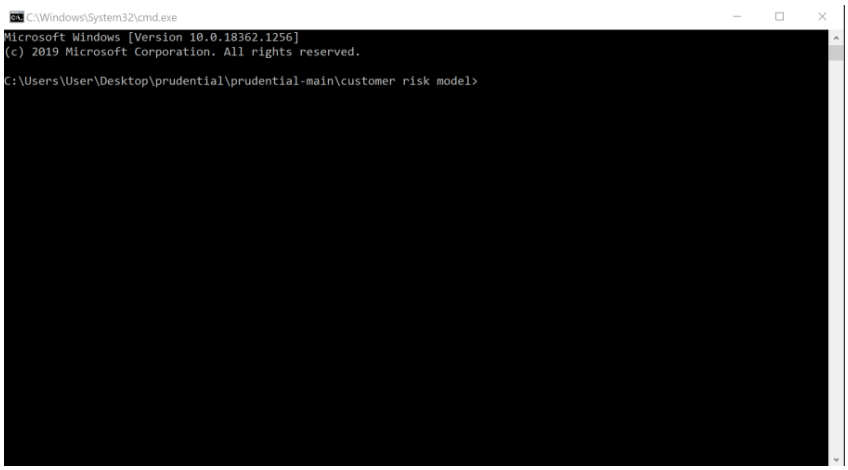
```
#XGBoost
clf_xgb = XGBClassifier(colsample_bytree= 0.3, learning_rate= 0.1, max_depth= 6, reg_alpha = 0.8)
```

```
# Random Forest
clf_rf = RandomForestClassifier(n_estimators= 400, criterion= 'sqrt', min_samples_split= 5, min_samples_leaf = 1,  max_features
```

```
# Adaboost
clf_ab = AdaBoostClassifier(DecisionTreeClassifier(max_depth=2),n_estimators=225, learning_rate = 0.3)
```

   e. Save app.py using Ctl+S

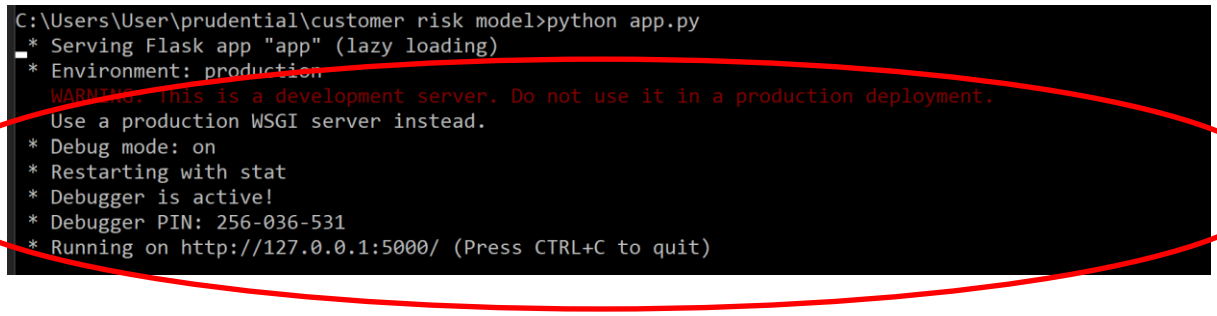
4) Go back to cmd



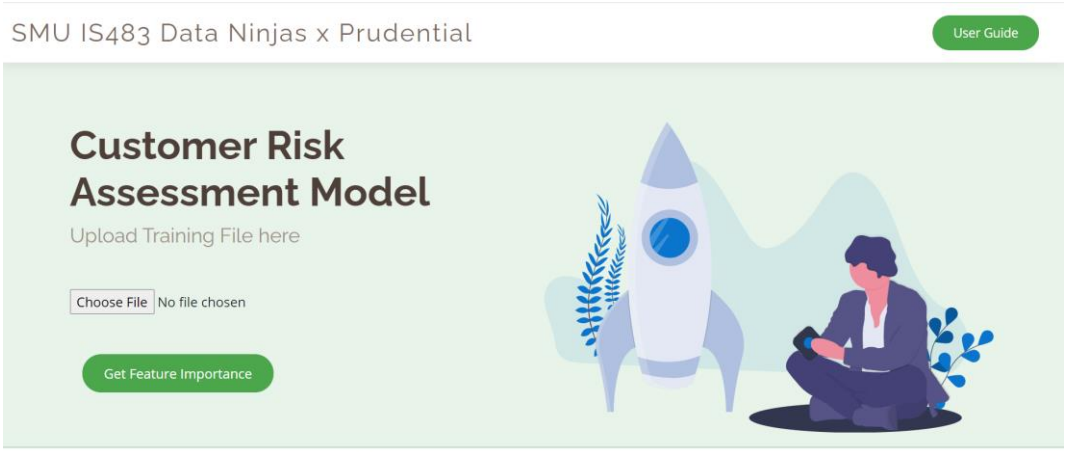
5) Run the Flask App

```
python app.py
```

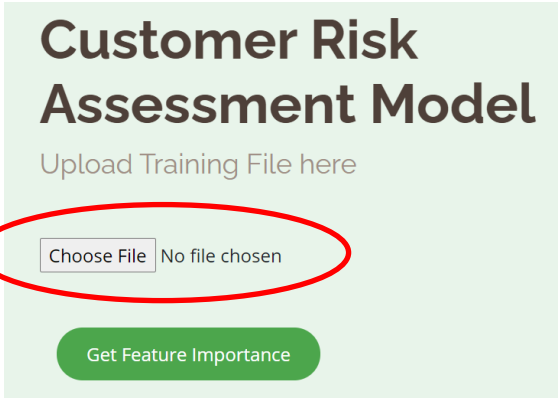
You should see this which means the flask app is live and running.



6) Go to the link: http://127.0.0.1:5000/ and you should see this page below:

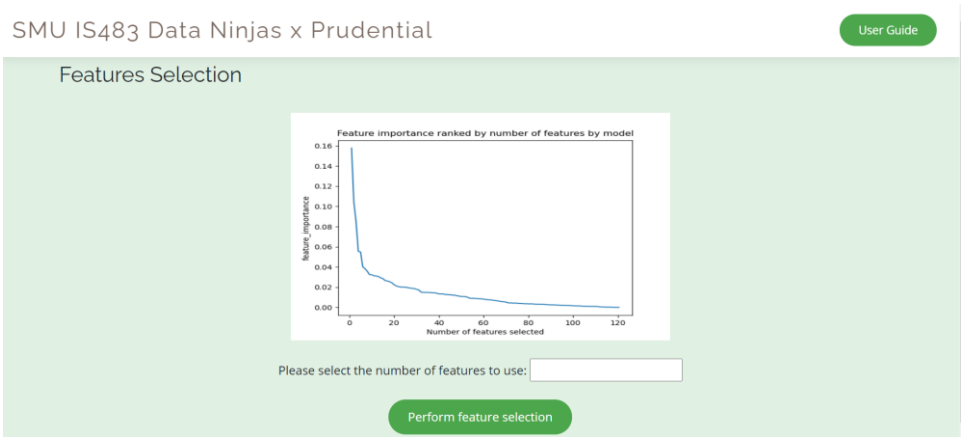


7) Upload cleaned training file by clicking on the 'Choose File' button.



**** Note: Training file needs to contain at least 10,000 rows of data**

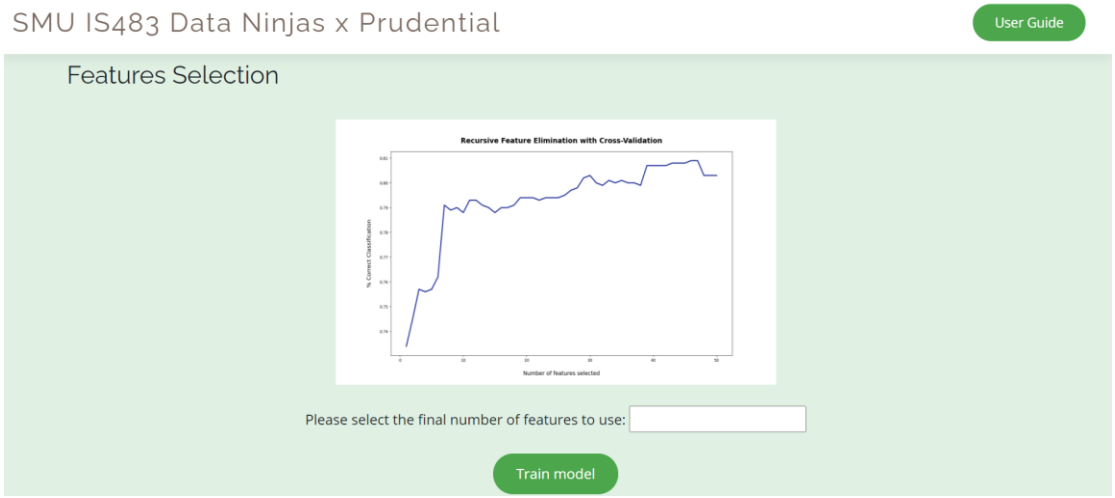
8) Click on "Get Feature Importance" button to run feature importance ranking across all the features. A feature importance graph will be shown, and you will need to select the number of features to use to run feature selection. This is because feature selection is an expensive process and it would be better to cut down unimportant features first at the start.



Input the number of features to use and click on the "Perform feature selection" button.

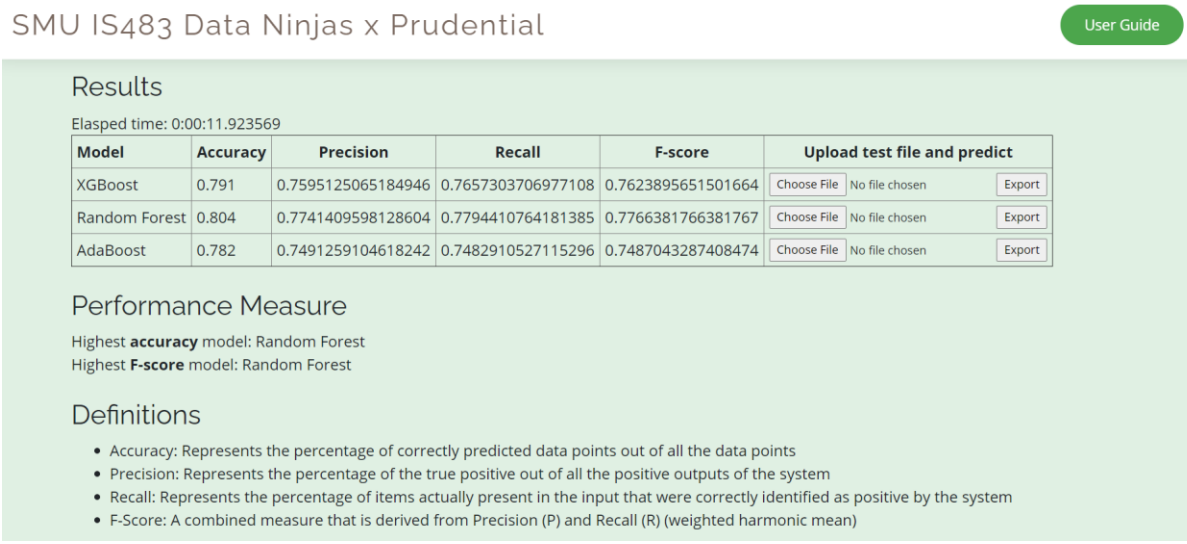
Note: I*t may take awhile for the feature selection to run, depending on the number of features you have. Took around 40 minutes to cut from 120 features to 60 features.*

9) A feature selection graph would be shown once feature selection is done.

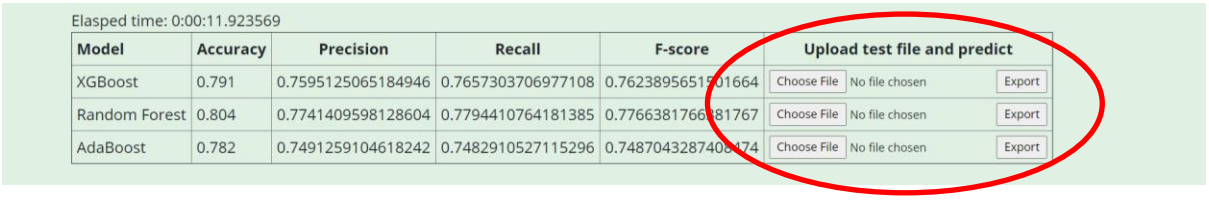


From the graph, you can pick the final number of features you want to use to train your model with based on the number of features that gives the highest % of correct classification. Input the final number of features to use and click on “Train model” to start model training.

10) After model has finished training, you will be bought to the results page. At this page, you can see the accuracy, precision, recall and f-score of the 3 models – AdaBoost, XGBoost and Random Forest.



11) To choose the best model to predict the labels on your test file:
For the model chosen:
i) Upload test file
ii) Click “export”



12) Once you click on ‘export’, a file will be downloaded. This file contains the labels predicted by the respective model on the test file you uploaded.