**CHURN RISK SCORE (Online Shopping App) CLASSIFICATION PROJECT**

**Problem Statement**

The problem statement of this project is to construct a multiclass classification model which is going to figure out the chances of the customer of an online shopping application of leaving the platform.

This is multiclass classification problem where the model is going to give a churn risk score in between 1 to 5 .

If for a customer the churn risk score is of 5 it means that the customer is highly likely to leave the platform,

If the churn risk score is of 1 it implies that the customer is not very likely to leave the shopping platform. Overall it is a 5 class classification problem !

**About the data set**

For this project I picked up a publicly available data set , which is having about 40k samples and 25 features including the dependent column.

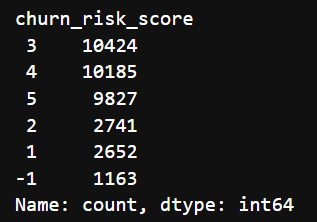
The independent features are as follows

* Customer ID
* Name
* Age
* Gender
* Region Category
* Membership Category
* Joining Date
* Average Time Spent
* Average Transaction Value
* Points In Shopping wallet
* Past Complaint
* Complaint Status
* Feedback

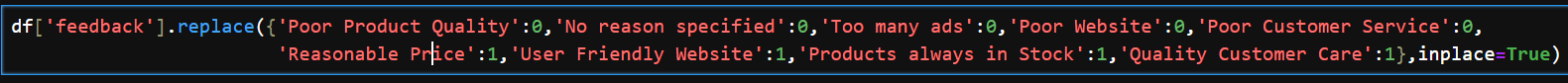
And more features !!

**Preprocessing**

* **Missing Values –** noticed the fact that there were few features with missing values
* **Imbalanced Categories -** from this one can clearly see the degree of imbalance that the categories have , also do notice the fact that there is a ghost category with the label -1



* **Dropping Insignificant Features -** there were a few insignificant features which I dropped
* Customer\_id
* Name
* Security\_no
* Referral\_id
* Medium of operation
* Internet option
* Last visit time
* **Analysed the effect of each feature on churn risk score –**
* Gender was not having any effect on churn risk score (to figure this out I looked at the number of males and females in churn score category )
* The feature **Region** i.e City , Town, Village to which the customer belonged was not having much effect on the churn risk score
* **Membership category -**  did similar analysis on this feature as well and noticed that the customers with platinum memberships were least likely to leave
* **Preferred Offer Types -**  noticed that people without offers were more likely to leave
* **Past complaint -**  surprisingly this feature was also not having any effect on the churn risk score , each churn risk category was having almost equal number of people with and without past complaint
* **The story of the feedback feature-** the feedback feature is a categorical feature with 9 categories I noticed that customers who gave -ve feedbacks were more likely to leave so I wanted to ordinally encode this categorical feature , so there were +ve and -ve feedbacks , it wasn’t possible to figure out which -ve feedback is the most -ve one because all of them were having almost the same churn score composition , thus I decided to group all -ve feed backs as one particular category and I did a similar thing with positive feedbacks as well



* **Handling Columns With GHOST Categories –**
* **Gender –** The gender feature in my dataset was having a third category “Unknown” , I decided to divided this unknown category into M and F, I decided so because the categories M and F were pretty well balanced so as to maintain the balance I divided the GHOST category into M and F randomly
* **Churn Risk Score –** The feature churn risk score was having a ghost category -1 , I am calling this category ghost because this category wasn’t mentioned anywhere in the description of the data set , I assigned these samples category 5 , because I noticed that almost all the samples belonging to this category were having -ve feedbacks.

**Did this type of preprocessing with few more features !!**

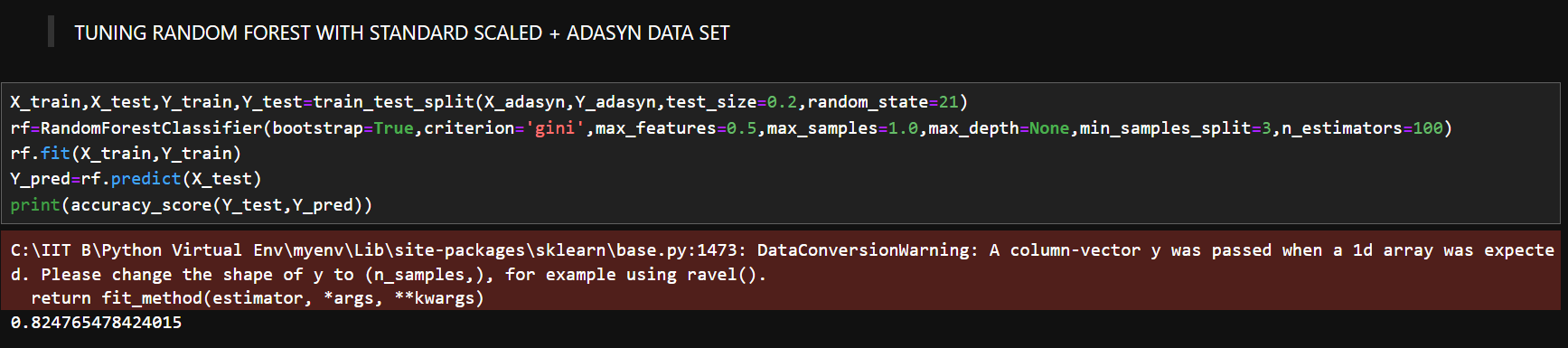
**NOW POST IMPUTING , ENCODING AND SCALING it was time to solve the problem of imbalance in the data set !! (for imputing at places I went for mean imputing and relevant methods , for Nominal categorical features performed OHE and for ordinal features underwent Ordinal Encoding, also for scaling I went for Standard Scaling after trying Min-Max Scaling )**

* **I went for scaling prior to applying balancing techniques like SMOTE and ADASYN because these oversampling techniques use KNN in their internal workings and as we are aware of the fact that KNN uses distance metric and if not scaled the feature with entries of higher scale will have more say in figuring out the neighbours , thus to get precise synthetic samples I went for Standard Scaling prior to applying oversampling techniques !!**

**SO POST APPLYING SMOTE AND ADASYN I have X\_smote , Y\_smote and X\_adasyn , Y\_adasyn .**

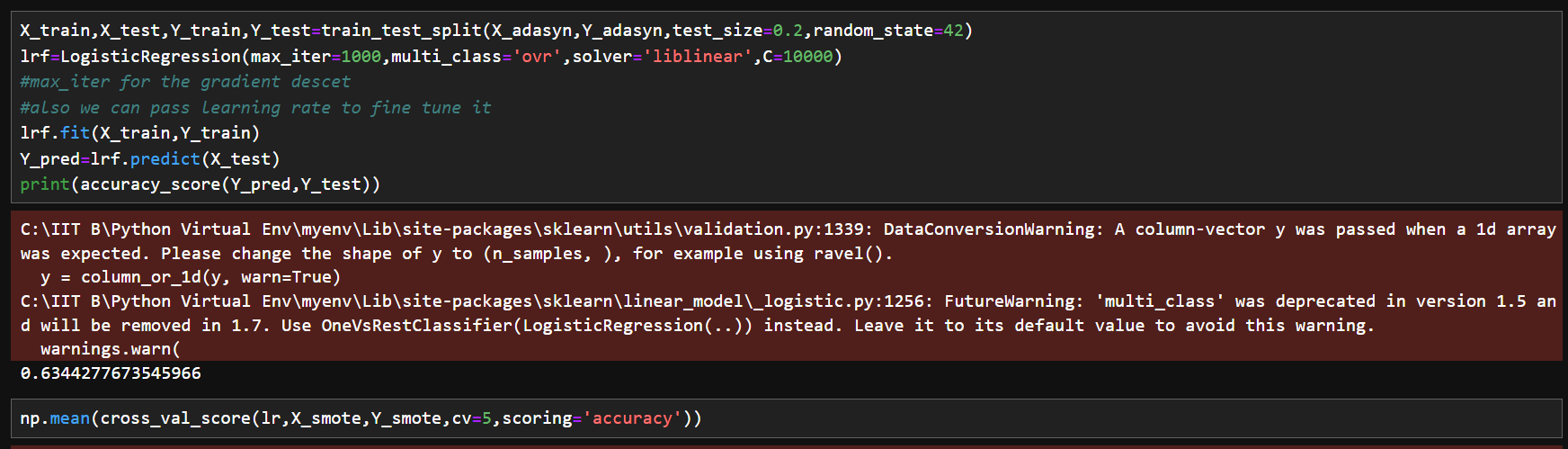
**Random Forest Classifier**

I trained and fine tuned random forest classifier with X\_smote ,Y\_smote and with X\_adasyn , Y\_adasyn, tuned and cross validated and received an accuracy score of about 83 percent.



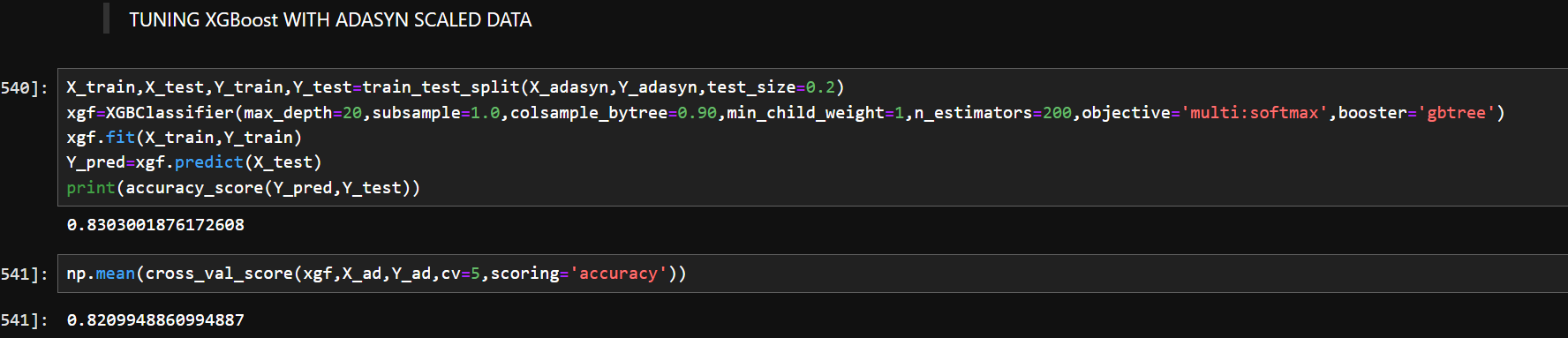
**LOGISTIC REGRESSION**

I trained Logistic Regression with X\_smote ,Y\_smote and with X\_adasyn , Y\_adasyn, tuned and cross validated and received an accuracy score of about 63-64 percent.



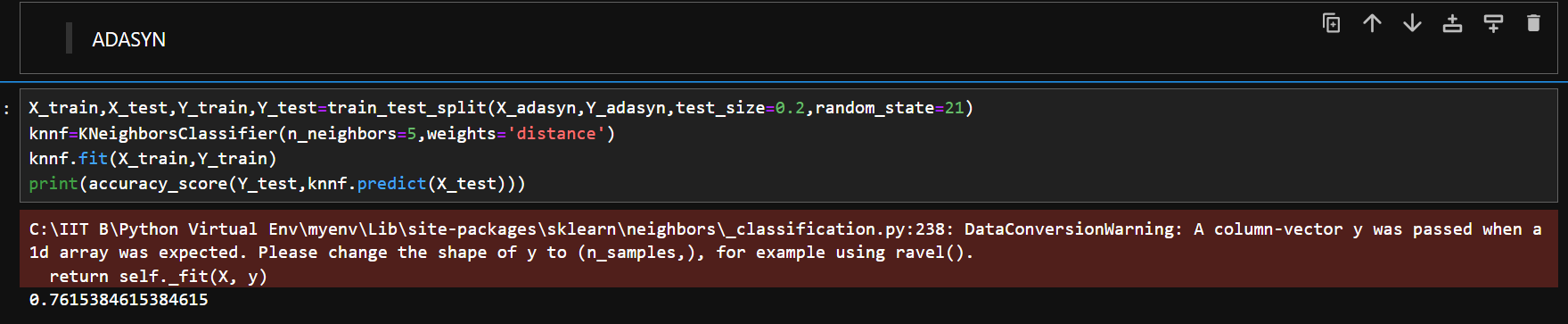
**XGBClassifier**

I trained and fine tuned XGBClassifier with X\_smote ,Y\_smote and with X\_adasyn , Y\_adasyn, tuned and cross validated and received an accuracy score of about 82-83 percent.



**KNNClassifier**

I trained and finetuned KNNclassifier with X\_smote ,Y\_smote and with X\_adasyn , Y\_adasyn, tuned and cross validated and received an accuracy score of about 75-76 percent.



I also tried using SVM classifier but it wasn’t performing well even after tuning it , it wasn’t crossing the accuracy score of 45 percent , thus I decided to not to stick with it.

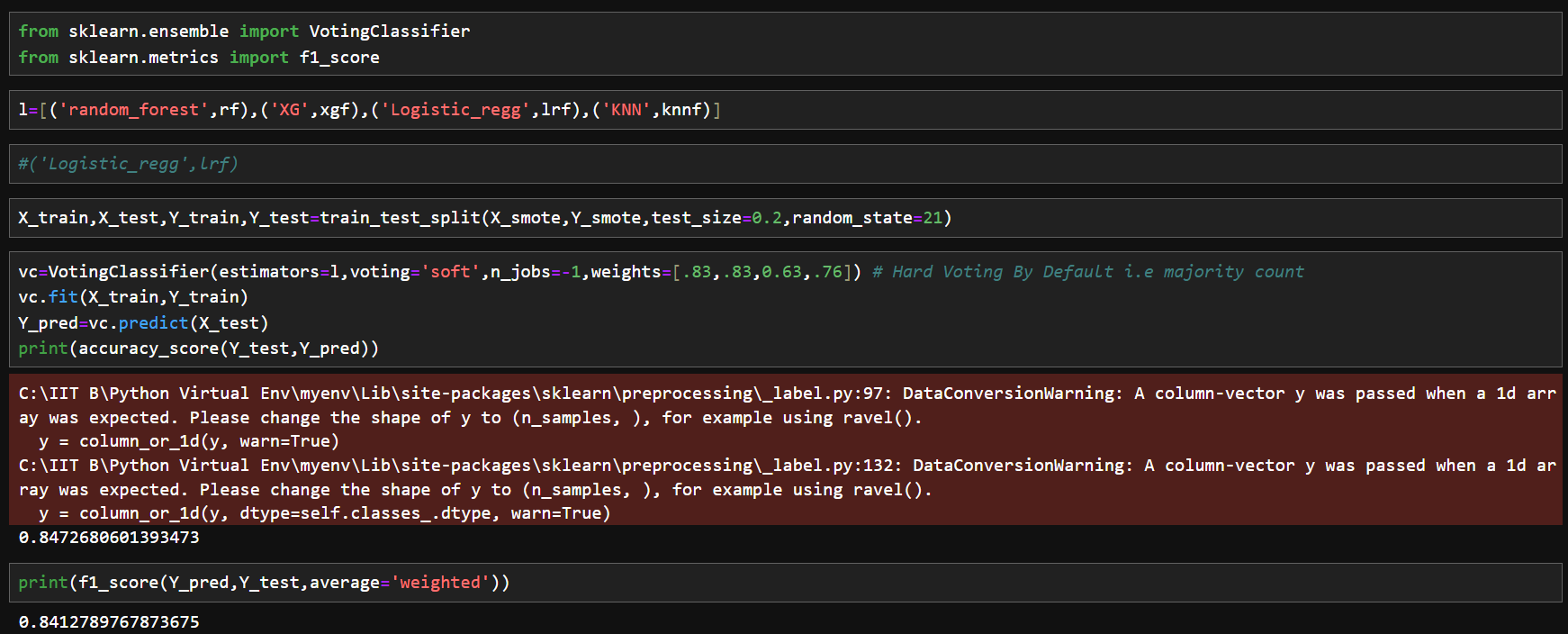
**VOTING CLASSIFIER**

I decided to construct an ensemble structure using all the well performing models I trained previously , Voting ensemble is known to improve the performance of you model significantly by working on bias variance thingi also the distinct base models of a voting structure tend to capture different aspects of the training data and these factors significantly enhance the performance of your overall model.

So I constructed a voting structure with the base models

* RandomForestClassifier
* LogisticRegression
* KNNClassifier
* XGBClassifier
* one can clearly notice the fact that I have already used the boosting and bagging ensemble in my overall classifier

This overall voting ensemble gave me the best accuracy score of 86 percent and somewhat similar F1 score , which is an improvement of 4 percent over the best performing base model.



**I fine tuned my voting classifier by assigning weights to each base model on the basis of their individual performance. Also I checked my voting structures performance in both the cases i.e ‘soft’ and ‘hard’ voting type .**