**Initial Models**

Initially we ran the models without any cross-validation. As label of the sets are binary, we chose tree-based models and logistic regression.

As our data is imbalanced, we were expecting better results from Boosting algorithms. On each iteration algorithms would increase the weight of wrong answer which would possibly be the minority labels. As a result, we would have automatically increased weights in our minority labels.

On terms of F1-Score, Logistic Regression and Random Forest performed worst whereas other three models performed similarly. Looking at the AUC score, XGBoost performed best with slightly better than other good models. As our data is imbalanced we did not prefer to note the accuracy scores as they would lead us to wrong results.

**Models with Cross-Validation**

After running the models on train data only first time, we decided to run models with cross-validation before moving further. In this phase, we used 5-fold cross-validation with the same models in previous phase.

Our results were mostly consistent with the previous phase. Boosting models were best in term of AUC score where results slightly differ. In terms of F1-score Random Forest performed much better relative to the previous phase.

There is a considerable decrease in F1 score of XGBoost when model is run with cross-validation. Gradient Boosting seems to be the most resilient model, performing very good in phase 1 and phase 2. On the other hand, Logistic Regression is the worst performing model in both phases.

After the cross-validation phase, we decided to select 2 models and finetune the final models’ parameters. As Gradient Boosting performed very well in both phases and also increased performance on cross-validation, it was our first candidate.

XGBoost was the most preferred model in Kaggle for the last two years. Although the model didn’t perform any better after cross-validation, we decided to consider it as second candidate.

**Finetuning the Parameters of Gradient Boosting**

As Gradient Boosting was one of our final models, we decided to finetune the parameters to increase performance. During this process we decided to keep using cross-validation. Skicit-learn library allowed us for using parallel computing, which shortened the calculation time. The definition of model is as following:

**learning\_rate:** Learning rate shrinks the contribution of each tree by learning\_rate. There is a trade-off between learning\_rate and n\_estimators. Initial values is chosen as 0.1

**n\_estimators:** The number of boosting stages to perform. Gradient boosting is fairly robust to over-fitting so a large number usually results in better performance. This value was our first choice to perform grid search.  
**min\_samples\_split:** The minimum number of samples required to split an internal node. Initial value is chosen as 500.

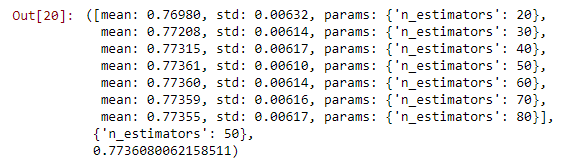
**min\_samples\_leaf:** The minimum number of samples required to be at a leaf node. Initial values is set to 50.  
**max\_depth:** Maximum depth of the individual regression estimators. The maximum depth limits the number of nodes in the tree. Initial value is set to 8.

**max\_features:** The number of features to consider when looking for the best split. Initial value is chosen as ‘sqrt’. As there are 138 features in our model this value corresponds to 12.

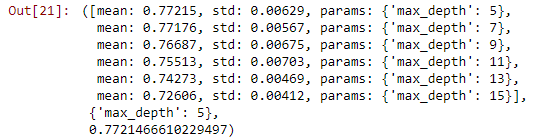
**subsample:** The fraction of samples to be used for fitting the individual base learners. Initial value is set to 0.8

**random\_state:** Random state is the seed used by the random number generator. We set this value to 10 which is chosen randomly.

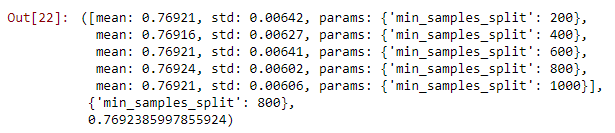
**Applying Grid Search for Finding Better Parameters**

We started with trying different values for n\_estimators. Test data was not used until all parameters were finetuned. Value of 50 for n\_estimators parameter is better than the other, so this value is set.  


After number of estimators, maximum depth is chosen for finetuning. Value of 5 performed best in this phase.



On third stage, minimum samples for split is chosen. Value of 800 performed the best in the given interval.



On the fourth stage, minimum samples at leaf is chosen. Value of 60 performed the best in the given interval.

