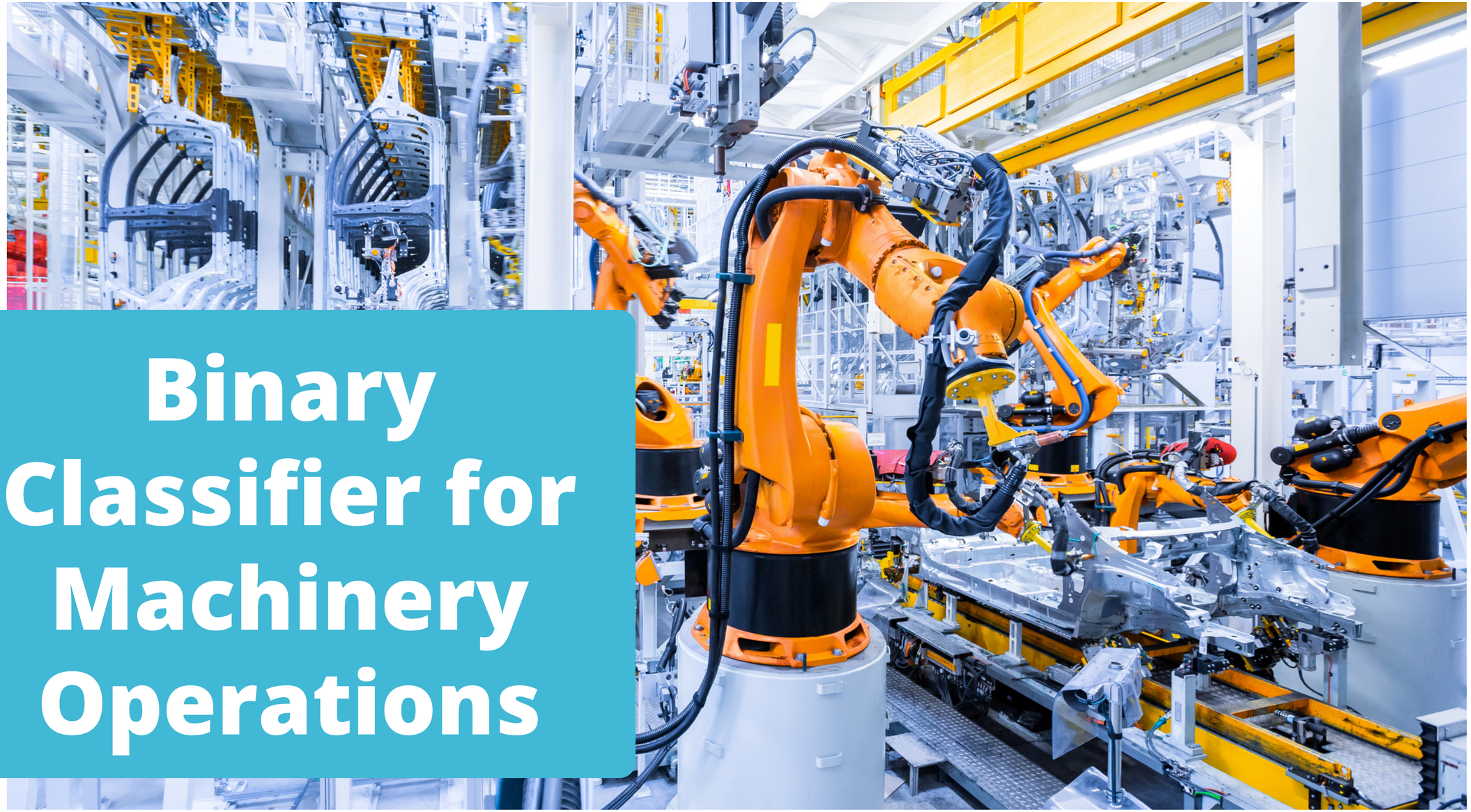
Team KIM

2 June 2022



Binary Classifier for Machinery Operations

Journey Through A Kaggle Competition

The Challenge

Competition Features

1,118	1,142	8,429
Teams	Competitors	Entries

Data Features

Train		Test
900,000	**33**	**700,000**
rows	columns	rows

- We joined a Kaggle Competition to pit our skills against over 1,100 teams
- The competition allowed us to work in a group and provided us a large, feature-rich dataset to work on
- We had a goal of predicting whether a machine is in a 1/0 state based on simulated manufacturing control data - a real world application

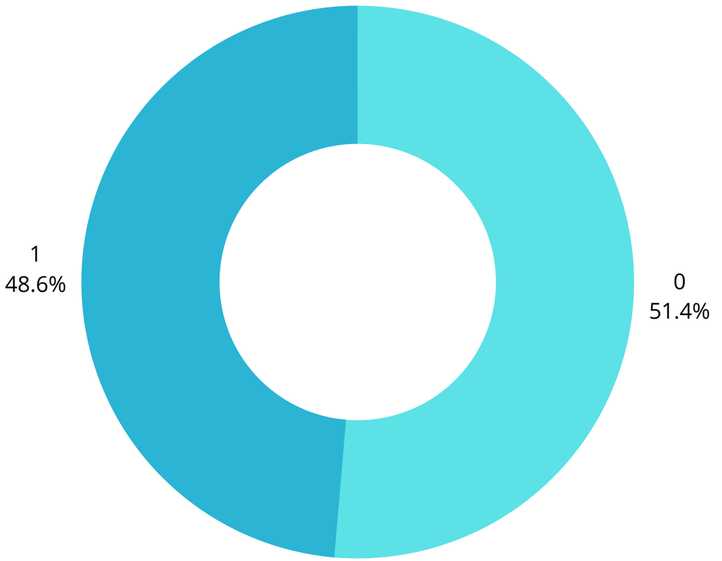
Dataset First Look

Training data sample rows:

	id	f_00	f_01	f_02	f_03	f_04	f_05	f_06	f_07	f_08	...	f_27	f_28	f_29	f_30	target
0	0	-1.373246	0.238887	-0.243376	0.567405	-0.647715	0.839326	0.113133	1	5	...	ABABDADBAB	67.609153	0	0	0
1	1	1.697021	-1.710322	-2.230332	-0.545661	1.113173	-1.552175	0.447825	1	3	...	ACACCADCEB	377.096415	0	0	1
2	2	1.681726	0.616746	-1.027689	0.810492	-0.609086	0.113965	-0.708660	1	0	...	AAAEABCKAD	-195.599702	0	2	1
3	3	-0.118172	-0.587835	-0.804638	2.086822	0.371005	-0.128831	-0.282575	3	2	...	BDBBAACBCB	210.826205	0	0	1
4	4	1.148481	-0.176567	-0.664871	-1.101343	0.467875	0.500117	0.407515	3	3	...	BDBCBBCHFE	-217.211798	0	1	1

- Training data comprises 900,000 lines and 33 columns (**feature rich**) of integer, float and string values
- Column names are anonymized to encourage data exploration and feature engineering
- Dataset is clean: balanced, no nulls, no duplicates
- Our job is to predict 'target' column with 1/0 values, making this a **binary classification** problem
- With our findings, we will have to predict 700,000 lines of unseen data

'target' data is balanced



First Attempt: Baseline Scores

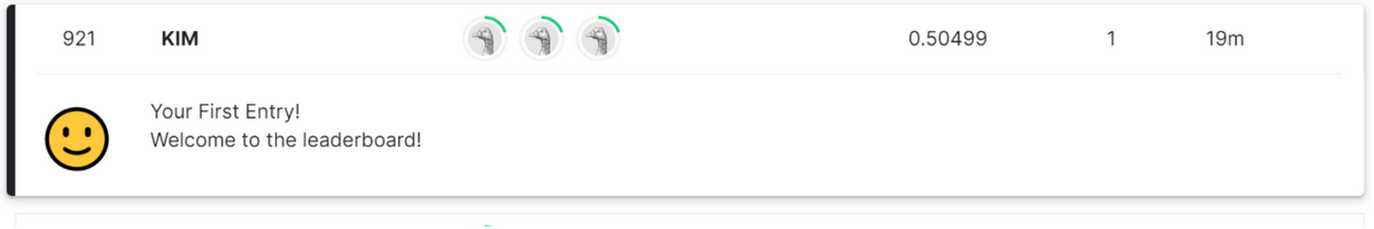
Since the data is clean and balanced, we proceeded to run competing classifiers to establish baseline scores

Model Features and Baseline Scores

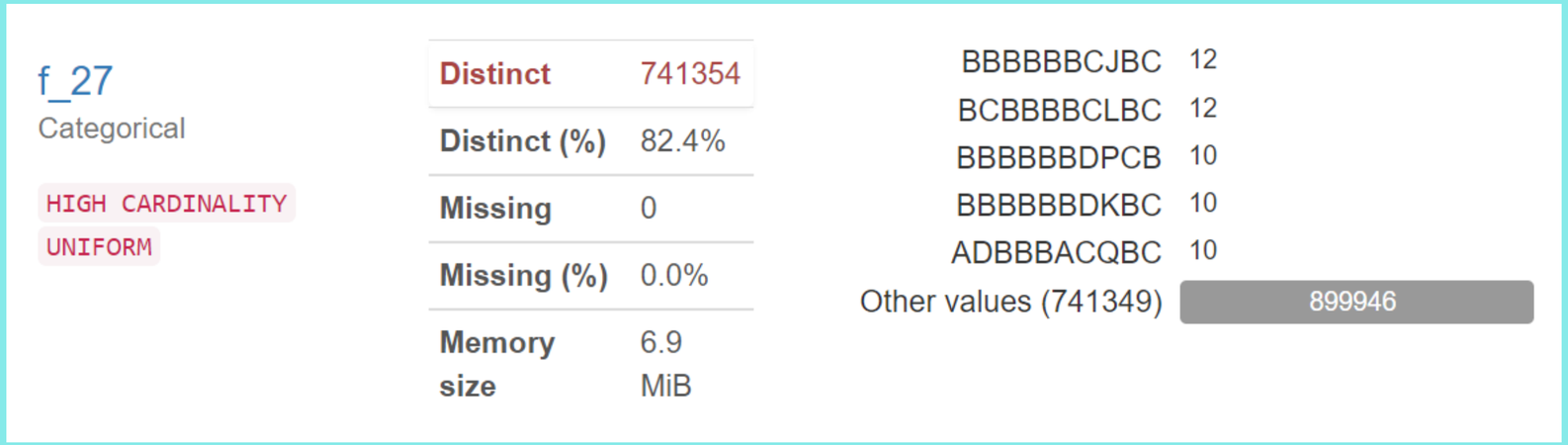
Model Selection	Interpretability	Training Time	Accuracy
Logistic Regression	Easy	Short	62%
Decision Tree	Easy	Short	73%
Random Forest	Hard	Long	83%
XGBoost	Hard	Long	74%

First Attempt: Lessons Learnt

Our opening shot ranked near bottom with results not much better than a coin toss (50-50) despite our Random Forest model showing 83% accuracy in training



We would need to feature engineer the data and re-examine the predictive value of features that we had overlooked in our initial attempt



We would revisit 'f_27' feature that we initially discarded because it appeared to contain mostly unique variables

EDA: Feature Engineering

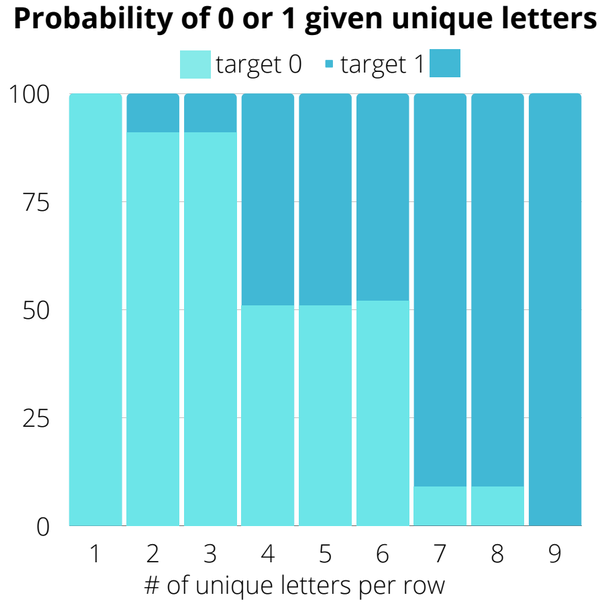
At first glance 'f_27' looked unremarkable but was one of the most predictive features

Raw data

f_27
ABABDADBAB
ACACCADCEB
AAAEABCKAD
BDBBAACBCB
BDBCBBCHFE
BDAEDBDEDA
ACBCAAAHHE
BBBDBBEPDB
ACAEBADDAA
BABCBBBABD
ADBBBBBSDC
BEBBBADTEA
BCBDCBDSAA
ADBCDAGKBE
BAACBBDDAC
BBBBFAAECB
ACAHABHHAB
ABBIBBCJFD
AGBCBACCBA
ADAEDAFQBA

Feature Engineering

1. Examined count of unique letters in each row, which shows relationship to 'target'



2. Examined occurance of letters in each row, then assigned numeric value to each letter

```
At Position 0: {'B', 'A'}
At Position 1: {'H', 'C', 'I', 'D', 'A', 'F', 'J', 'K', 'L', 'B', 'M', 'E', 'G', 'N'}
At Position 2: {'B', 'A'}
At Position 3: {'H', 'C', 'I', 'D', 'A', 'O', 'F', 'J', 'K', 'L', 'B', 'M', 'E', 'G', 'N'}
At Position 4: {'H', 'C', 'I', 'D', 'A', 'O', 'F', 'J', 'K', 'L', 'B', 'M', 'E', 'G'}
At Position 5: {'B', 'A'}
At Position 6: {'H', 'C', 'I', 'D', 'A', 'O', 'F', 'J', 'K', 'L', 'B', 'M', 'E', 'G', 'N'}
At Position 7: {'H', 'A', 'P', 'Q', 'N', 'S', 'C', 'R', 'K', 'T', 'I', 'D', 'O', 'L', 'J',
At Position 8: {'H', 'C', 'I', 'D', 'A', 'O', 'F', 'J', 'L', 'K', 'B', 'M', 'E', 'G', 'N'}
At Position 9: {'H', 'C', 'I', 'D', 'A', 'O', 'F', 'J', 'K', 'L', 'B', 'M', 'E', 'G', 'N'}
```

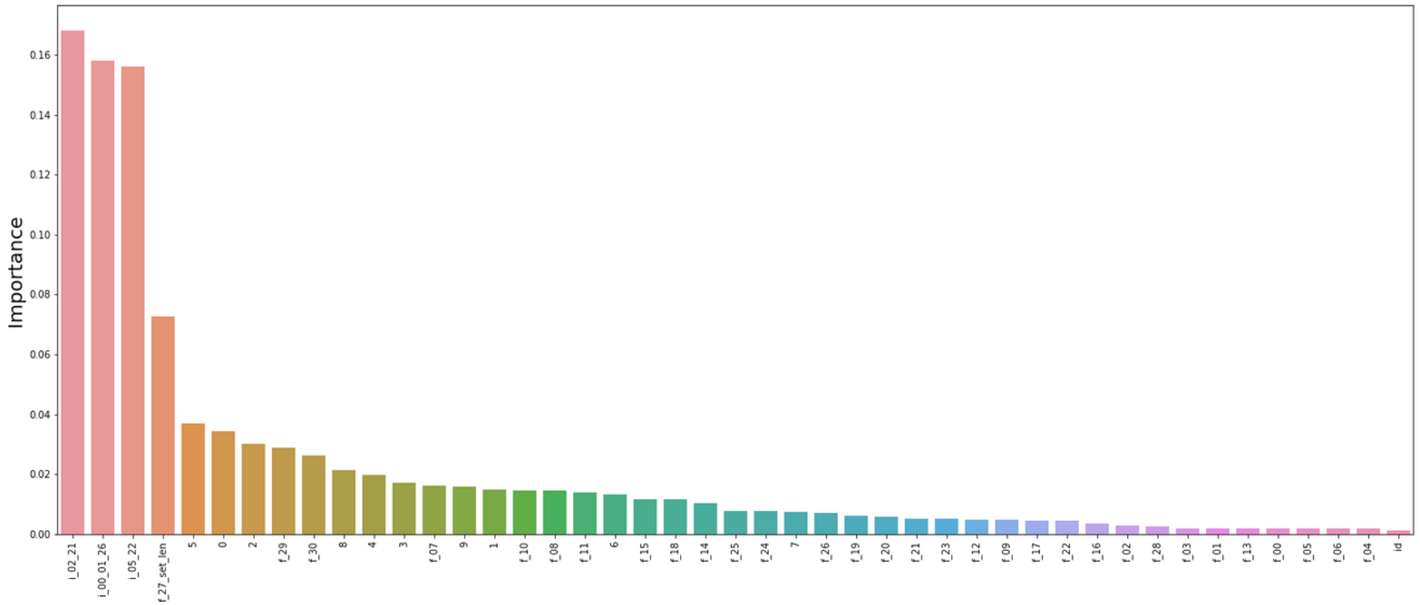
Output as new features

	f_27_set_len
0	3
1	5
2	6
3	4
4	6

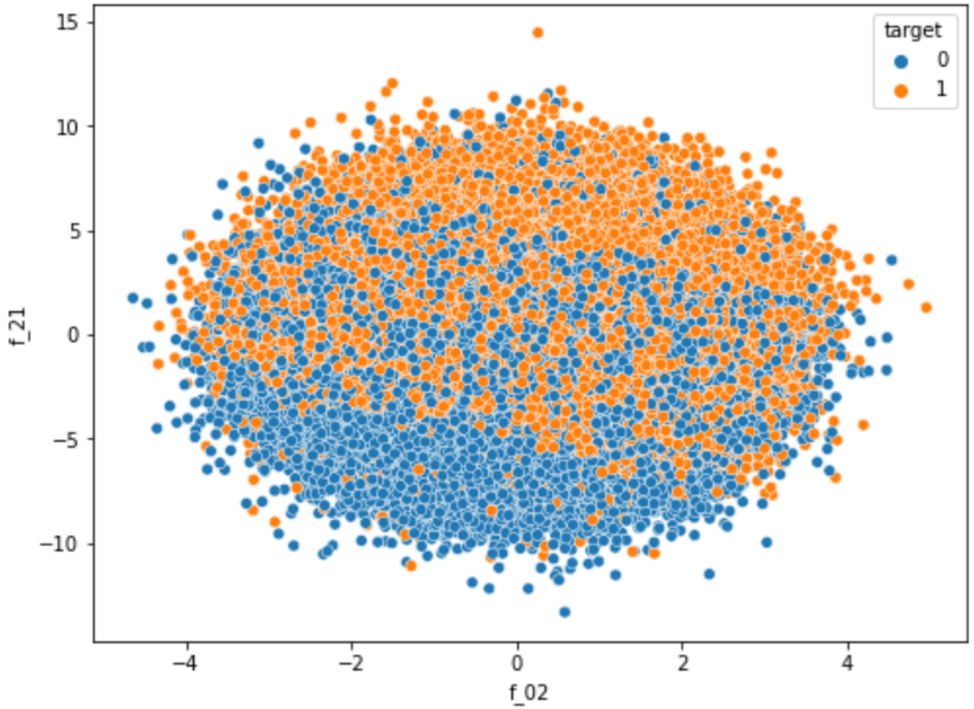
	0	1	2	3	4	5	6	7	8	9
0	0	1	0	1	3	0	3	1	0	1
1	0	2	0	2	2	0	3	2	4	1
2	0	0	0	4	0	1	2	10	0	3
3	1	3	1	1	0	0	2	1	2	1
4	1	3	1	2	1	1	2	7	5	4

EDA: Iterative Discovery

Investigating feature importance of each column



Examining relationship of features vs target



- We pored over all features looking for opportunity to further feature engineer
- The process was largely trial and error, with many ideas proving unsatisfactory
- We experimented iteratively with various features and this was the most resource and time-consuming phase

Ensemble Learning Models

With EDA and feature engineering done, we moved onto updating our model by testing ensemble learning techniques and favoring XGB

Updated Model Selection and Scores

Model Selection	Interpretability	Training Time	Accuracy
Random Forest	Hard	Long	83%
Light GBM	Hard	2min 20sec	96.26%
XGBoost	Hard	2min 22sec	97.07%

Final Attempt: XGBoost & Tuning

Manual HPT

- Set up a loop to tune with different parameter inputs
- Found this method superior to RandomizedSearch

```
n_estimators: 250, max_depth: 4
Accuracy:  0.9201388888888888
n_estimators: 250, max_depth: 6
Accuracy:  0.9463944444444444
n_estimators: 250, max_depth: 8
Accuracy:  0.9585555555555556
n_estimators: 250, max_depth: 10
Accuracy:  0.9637055555555556
n_estimators: 500, max_depth: 4
Accuracy:  0.9346777777777778
n_estimators: 500, max_depth: 6
Accuracy:  0.95695
...
n_estimators: 1750, max_depth: 14
Accuracy:  0.9691888888888889
n_estimators: 1750, max_depth: 16
Accuracy:  0.9688888888888889
n_estimators: 2000, max_depth: 10
Accuracy:  0.9702222222222222
n_estimators: 2000, max_depth: 12
Accuracy:  0.9695666666666667
n_estimators: 2000, max_depth: 14
Accuracy:  0.9694611111111111
n_estimators: 2000, max_depth: 16
Accuracy:  0.9690055555555556
```

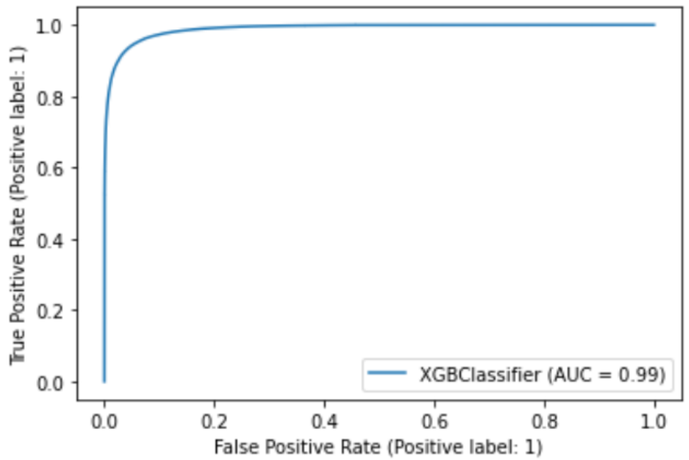
Pre-HPT

Pre-tuning baseline score was 94.6%

```
Accuracy is:  94.62
[[131552   6630]
 [  7890 123928]]
              precision    recall  f1-score   support

           0       0.94      0.95      0.95    138182
           1       0.95      0.94      0.94    131818

    accuracy                           0.95    270000
   macro avg       0.95      0.95      0.95    270000
weighted avg       0.95      0.95      0.95    270000
```



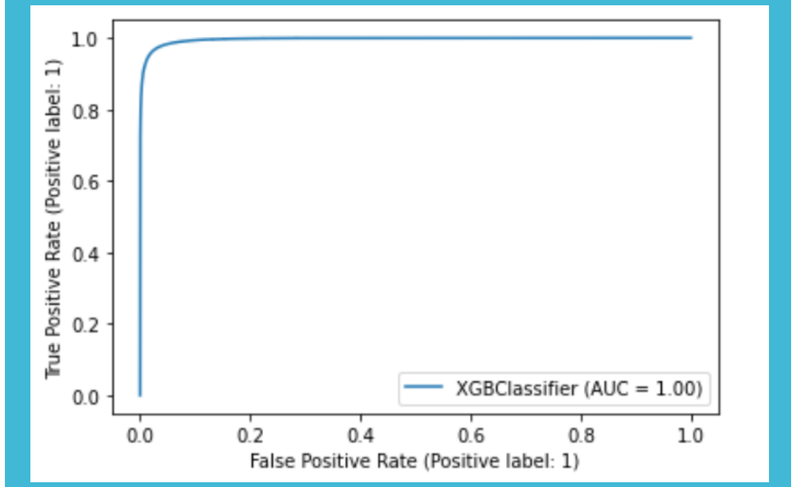
With HPT

Loading our HPT params into model yielded nearly 97.1% accuracy

```
Accuracy is:  97.07
[[134386   3796]
 [  4122 127696]]
              precision    recall  f1-score   support

           0       0.97      0.97      0.97    138182
           1       0.97      0.97      0.97    131818

    accuracy                           0.97    270000
   macro avg       0.97      0.97      0.97    270000
weighted avg       0.97      0.97      0.97    270000
```



Results: Moving up the Field

Final Attempt

- XGBoost
- Post-feature engineering
- Post-tuning

Baseline Attempt

- Random Forest
- Pre-feature engineering
- Pre-tuning

#	Team	Members	Score	Entries	Last	Code
1	The Team		0.99836	80	17h	
2	Joe Cooper		0.99830	73	3d	
3	Aman Singh		0.99830	39	3h	
589	▲ 1	**KIM**	0.97062	11	1d	
639	Kalyani Bandgar		0.93378	3	4d	
640	Devashish_Mahajan		0.93378	3	4d	
641	Abhishek Mehendale		0.93378	4	4d	
642	onoyu1012		0.93374	8	25d	
643	Nikhil's Kaggle Account		0.93372	4	6d	
921	**KIM**		0.50499	1	19m	
	Your First Entry! Welcome to the leaderboard!					
1072	Zain Ali Shakeel		0.48934	1	7d	
1073	Arqum Ashraf		0.48901	1	10d	
1074	RAKSHITHA BS		0.48808	2	8d	

Challenges

Dataset

The Dataset was sprawling and the feature names were anonymized so we could not glean any clues from 'eyeballing' the data

EDA

- Data was abstract so we could not intuit trends by eyeing features
- Features engineering was challenging; Even when we could identify useful features it was not immediately obvious how to engineer
- Skills gap to elite Kagglers who are able to 'see' more trends than within the same data

Models & Tuning

- Testing and tuning took large amounts of time, until we discovered GPU calculation
- Structured tuning techniques, ie. GridSearch, were less convincing than manual tuning

Conclusion

1. The project was a good test of our newfound skills against other data scientists, while simulating (via anonymized features) a real world scenario where we had a dataset in an area that we lacked domain knowledge
2. EDA is indispensible and consumed most of our time and resources; Without feature engineering we could not have crossed the 90% accuracy scoreline; But we did not uncover the 'last mile' to match the Kaggle elites
3. Parameter tuning delivered less accuracy gains than expected but we became more efficient at it after incorporating GPU calculations; As our inputs became more complex, XGBoost outstripped Random Forest in delivering results

Appendix: Workflow

Conceptualization

1. Jointly brainstormed ideas for ML project
2. Once decided, came up with common strategy

Version Control

1. Master versions of train/test files kept so members ran individual ML on same data
2. Latest findings from EDA or tuning captured iteratively

Submission

All work consolidated to make one submission

EDA & Features

Feature engineering was conducted separately but joint decisions made to implement most successful features in Master

Daily calls to discuss findings

Iterative

Model Selection & Tuning

Each member experimented with models and parameter tuning to find best results