Modeling Challenge

INDIVIDUAL EXERCISE

Ckalib Nelson

Risk & Fraud Analytics

Table of Contents

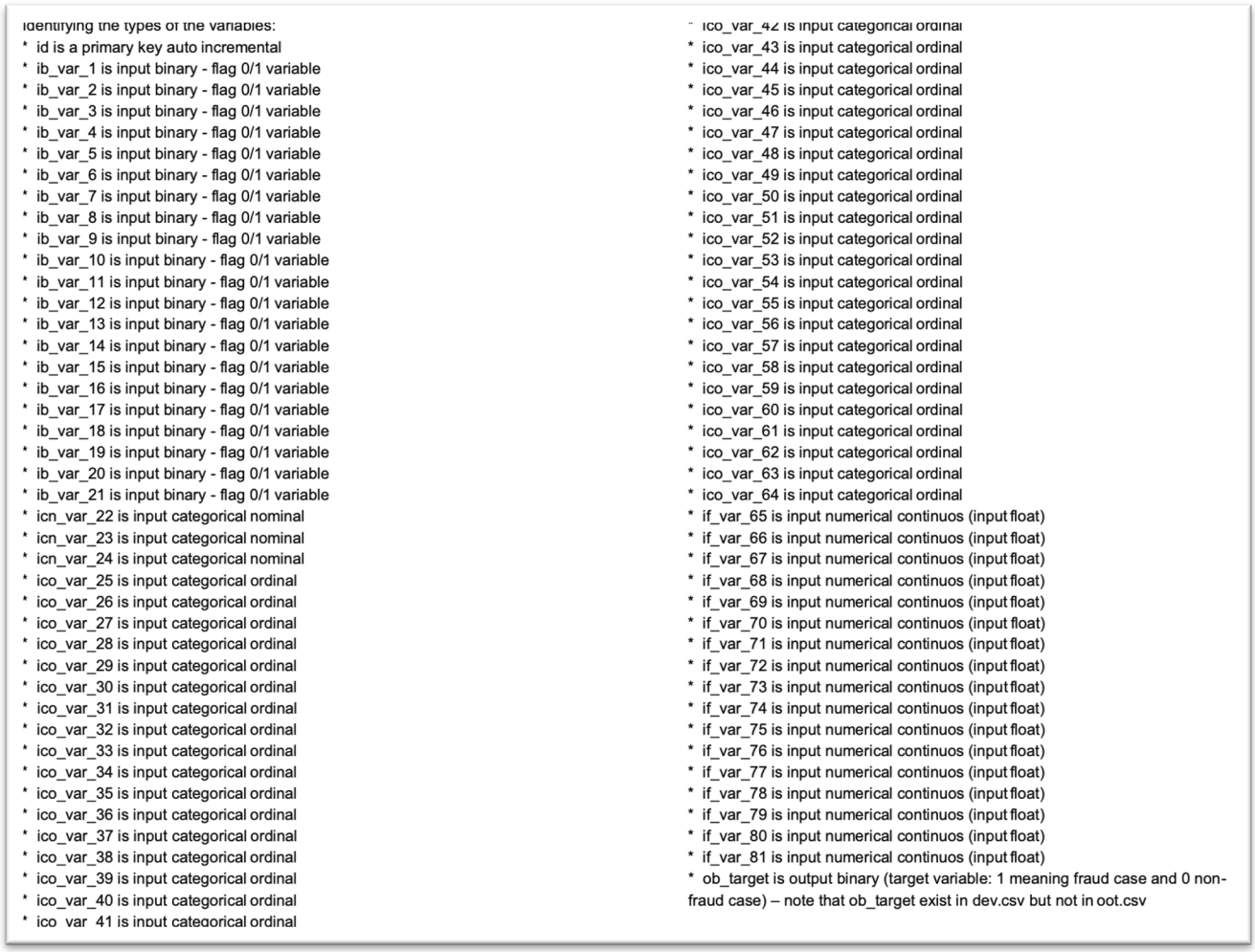
Business Understanding

The purpose of the individual exercise is to build a **fraud detection model**. A fraud detection model is important because organizations need to ensure they can retain the safety and security of the personal information of their customers.

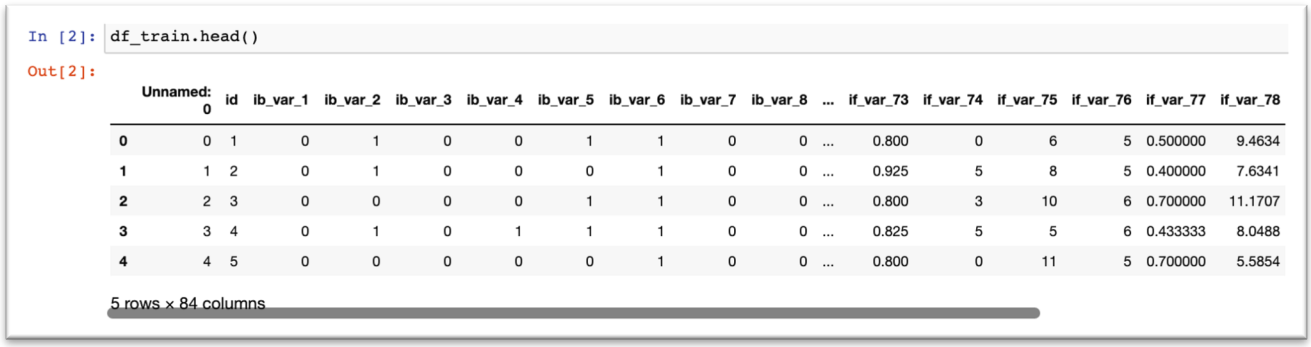
What we really want to do is build a fraud detection model with a strong GINI score. Per MBD01 Group D Gini & KS Statistics in Credit Scoring on YouTube, A GINI score of 100% indicates perfect separation and a GINI score of 0% indicates that fraudulent and non-fraudulent activity is the same. Thus, we want to yield a GINI score as close to 100% as possible.

Data Understanding

Given that the data is fictional, there is not much we can interpret from the data other than the various data types and the number of variables. Here is the synopsis of the data:



Identifying the types of the variables:
* id is a primary key auto incremental
* ib_var_1 is input binary - flag 0/1 variable
* ib_var_2 is input binary - flag 0/1 variable
* ib_var_3 is input binary - flag 0/1 variable
* ib_var_4 is input binary - flag 0/1 variable
* ib_var_5 is input binary - flag 0/1 variable
* ib_var_6 is input binary - flag 0/1 variable
* ib_var_7 is input binary - flag 0/1 variable
* ib_var_8 is input binary - flag 0/1 variable
* ib_var_9 is input binary - flag 0/1 variable
* ib_var_10 is input binary - flag 0/1 variable
* ib_var_11 is input binary - flag 0/1 variable
* ib_var_12 is input binary - flag 0/1 variable
* ib_var_13 is input binary - flag 0/1 variable
* ib_var_14 is input binary - flag 0/1 variable
* ib_var_15 is input binary - flag 0/1 variable
* ib_var_16 is input binary - flag 0/1 variable
* ib_var_17 is input binary - flag 0/1 variable
* ib_var_18 is input binary - flag 0/1 variable
* ib_var_19 is input binary - flag 0/1 variable
* ib_var_20 is input binary - flag 0/1 variable
* ib_var_21 is input binary - flag 0/1 variable
* icn_var_22 is input categorical nominal
* icn_var_23 is input categorical nominal
* icn_var_24 is input categorical nominal
* ico_var_25 is input categorical ordinal
* ico_var_26 is input categorical ordinal
* ico_var_27 is input categorical ordinal
* ico_var_28 is input categorical ordinal
* ico_var_29 is input categorical ordinal
* ico_var_30 is input categorical ordinal
* ico_var_31 is input categorical ordinal
* ico_var_32 is input categorical ordinal
* ico_var_33 is input categorical ordinal
* ico_var_34 is input categorical ordinal
* ico_var_35 is input categorical ordinal
* ico_var_36 is input categorical ordinal
* ico_var_37 is input categorical ordinal
* ico_var_38 is input categorical ordinal
* ico_var_39 is input categorical ordinal
* ico_var_40 is input categorical ordinal
* ico_var_41 is input categorical ordinal
* ico_var_42 is input categorical ordinal
* ico_var_43 is input categorical ordinal
* ico_var_44 is input categorical ordinal
* ico_var_45 is input categorical ordinal
* ico_var_46 is input categorical ordinal
* ico_var_47 is input categorical ordinal
* ico_var_48 is input categorical ordinal
* ico_var_49 is input categorical ordinal
* ico_var_50 is input categorical ordinal
* ico_var_51 is input categorical ordinal
* ico_var_52 is input categorical ordinal
* ico_var_53 is input categorical ordinal
* ico_var_54 is input categorical ordinal
* ico_var_55 is input categorical ordinal
* ico_var_56 is input categorical ordinal
* ico_var_57 is input categorical ordinal
* ico_var_58 is input categorical ordinal
* ico_var_59 is input categorical ordinal
* ico_var_60 is input categorical ordinal
* ico_var_61 is input categorical ordinal
* ico_var_62 is input categorical ordinal
* ico_var_63 is input categorical ordinal
* ico_var_64 is input categorical ordinal
* if_var_65 is input numerical continuos (input float)
* if_var_66 is input numerical continuos (input float)
* if_var_67 is input numerical continuos (input float)
* if_var_68 is input numerical continuos (input float)
* if_var_69 is input numerical continuos (input float)
* if_var_70 is input numerical continuos (input float)
* if_var_71 is input numerical continuos (input float)
* if_var_72 is input numerical continuos (input float)
* if_var_73 is input numerical continuos (input float)
* if_var_74 is input numerical continuos (input float)
* if_var_75 is input numerical continuos (input float)
* if_var_76 is input numerical continuos (input float)
* if_var_77 is input numerical continuos (input float)
* if_var_78 is input numerical continuos (input float)
* if_var_79 is input numerical continuos (input float)
* if_var_80 is input numerical continuos (input float)
* if_var_81 is input numerical continuos (input float)
* ob_target is output binary (target variable: 1 meaning fraud case and 0 non-fraud case) – note that ob_target exist in dev.csv but not in oot.csv



```
In [2]: df_train.head()
```

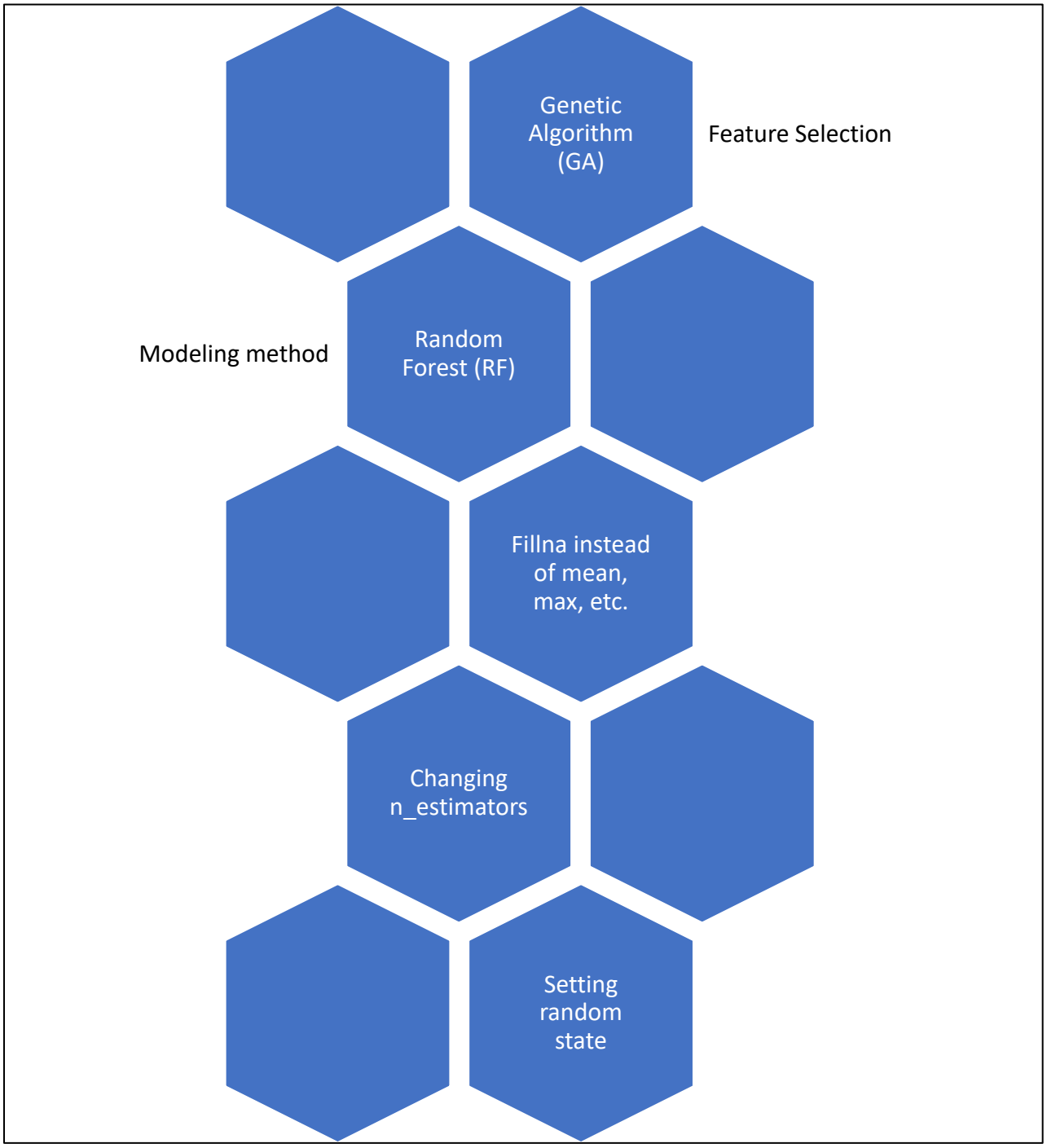
Out[2]:

	Unnamed: 0	id	ib_var_1	ib_var_2	ib_var_3	ib_var_4	ib_var_5	ib_var_6	ib_var_7	ib_var_8	...	if_var_73	if_var_74	if_var_75	if_var_76	if_var_77	if_var_78
0	0	1	0	1	0	0	1	1	0	0	...	0.800	0	6	5	0.500000	9.4634
1	1	2	0	1	0	0	0	1	0	0	...	0.925	5	8	5	0.400000	7.6341
2	2	3	0	0	0	0	1	1	0	0	...	0.800	3	10	6	0.700000	11.1707
3	3	4	0	1	0	1	1	1	0	0	...	0.825	5	5	6	0.433333	8.0488
4	4	5	0	0	0	0	0	1	0	0	...	0.800	0	11	5	0.700000	5.5854

5 rows × 84 columns

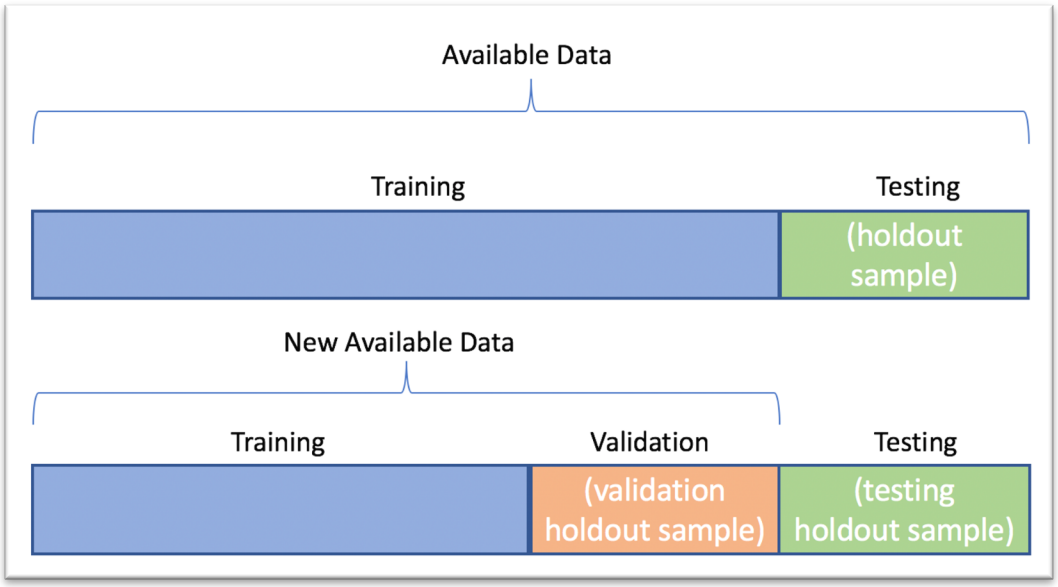
Data Preparation

What did work?

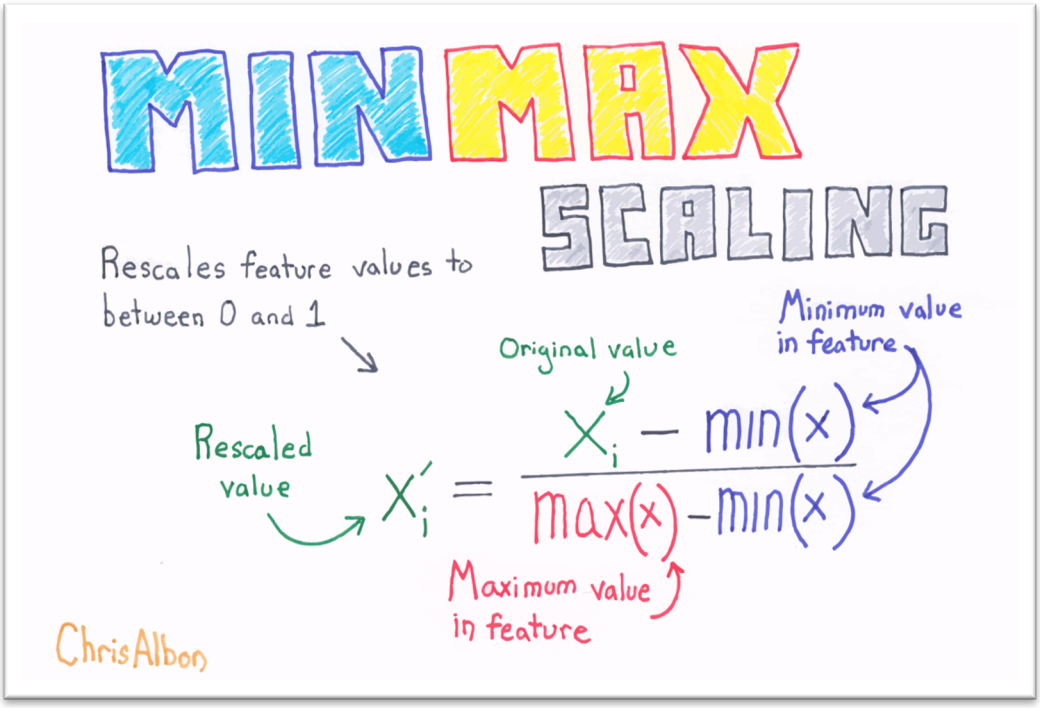


What did not work?

Train-test split

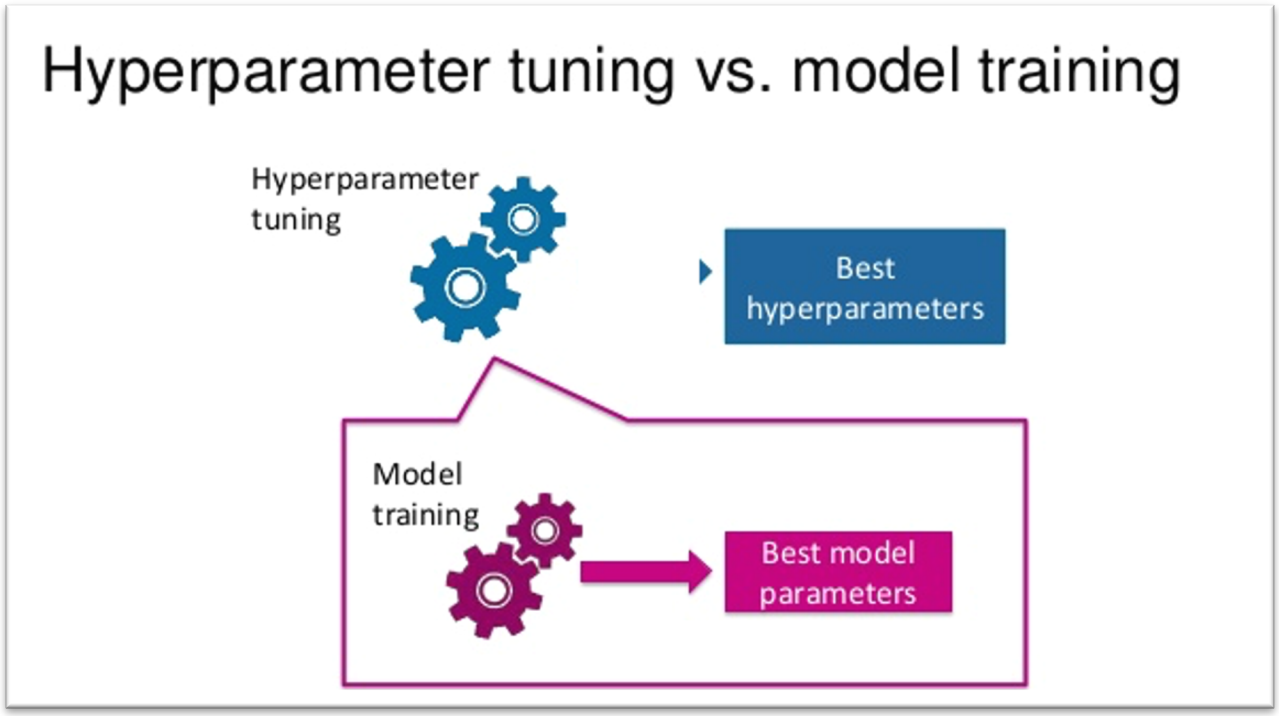


Scaling

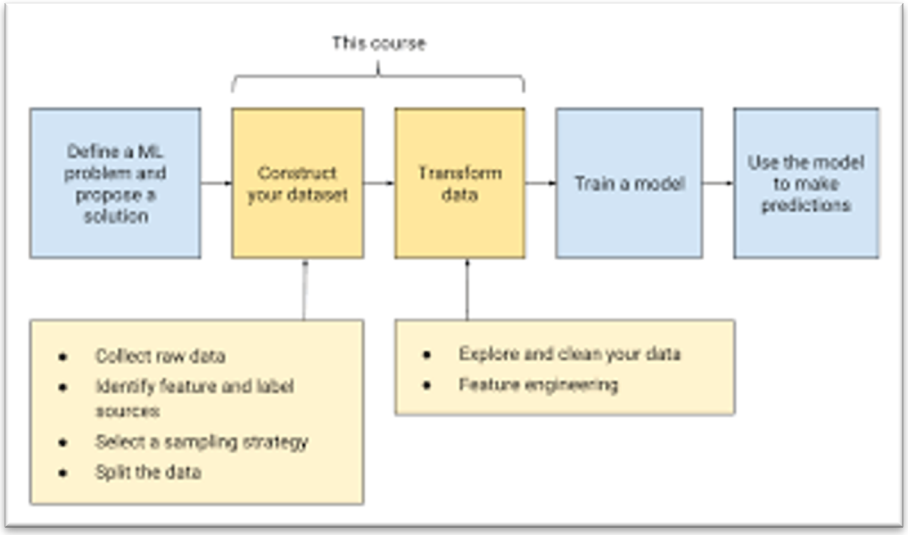


What I wish I could have done?

Parameter optimization



Feature Engineering

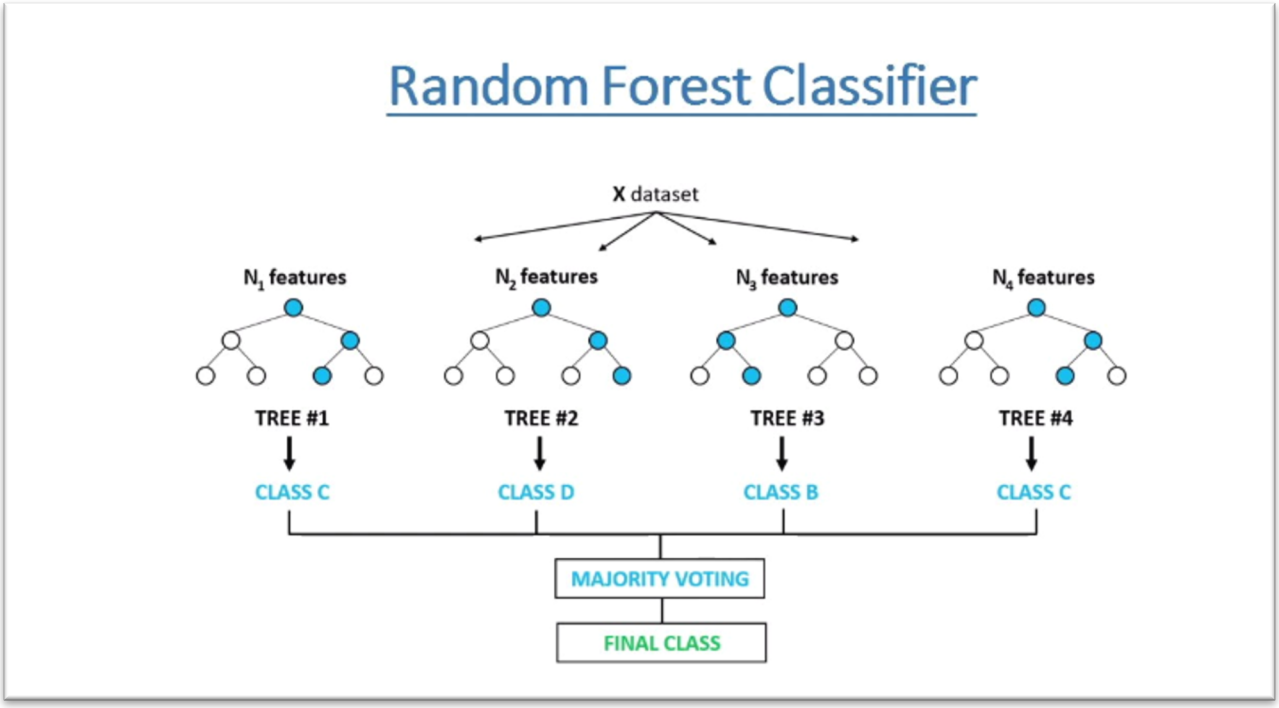


Modeling

Random Forest Classifier

After much trial and error, I discovered that the **Random Forest Classifier** provided the best results relative to Logistic Regression and Decision Tree Classifier.

```
######
# CHANGE_HERE - START: YOU ARE VERY LIKELY USING A DIFFERENT TECHNIQUE BY NOW. SO CHANGE TO YOURS.
#####
from sklearn.datasets import make_classification
X, y = make_classification(n_samples=1500, n_features=25, n_informative=15, n_redundant=7)
RF = RandomForestClassifier(n_estimators=10000, oob_score=True, random_state=123 ,max_features='log2', bootstrap =
model = RF.fit(X_train, Y_train)
Y_predict = model.predict_proba(X_test)[:,1]
Y_predict_train = model.predict_proba(X_train)[:,1]
gini_value = submission(Y_predict)
```



Submission

As of May 28th, my best GINI score was roughly 52% w/ a SD of 0.037.