Kaggle - Give Me Some Credit

Problem – Credit score prediction is an interesting problem which is used by banks to determine whether a loan should be granted or not. The goal is to predict whether someone will experience financial distress in the next two years using historical data.

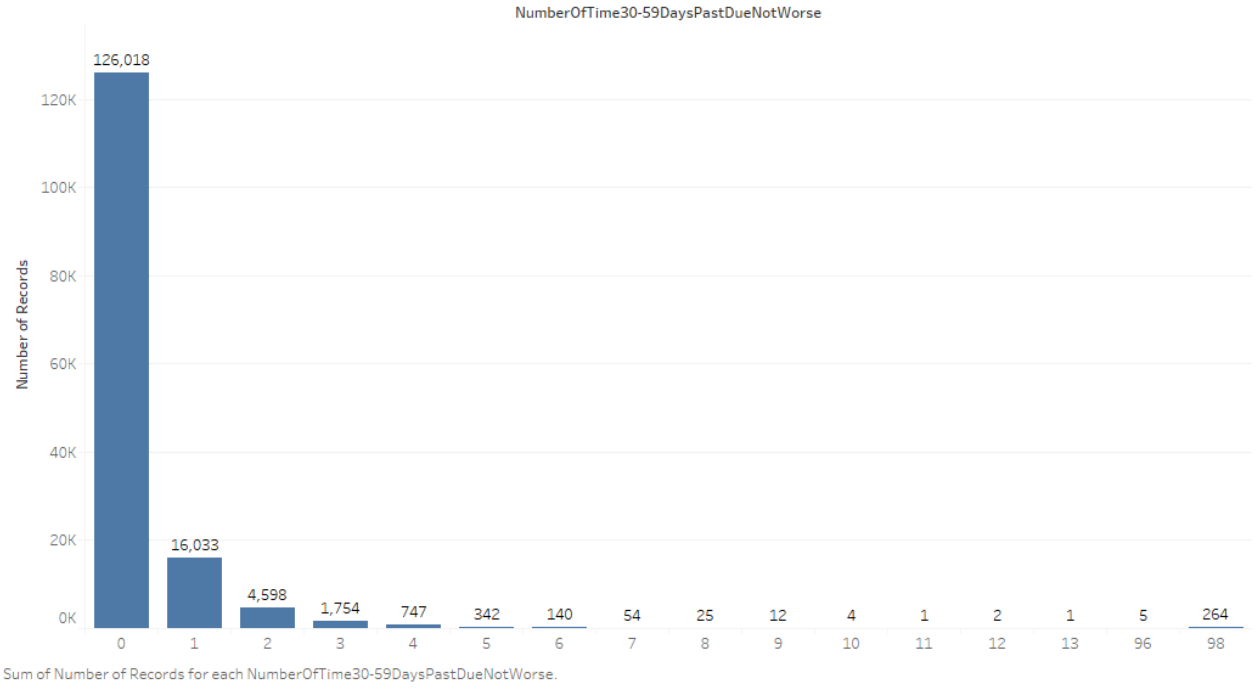
I will describe an approach to predict credit score using random forests

Data Cleaning & Imputation (Graphs were drawn in Tableau & Microsoft PowerPoint)

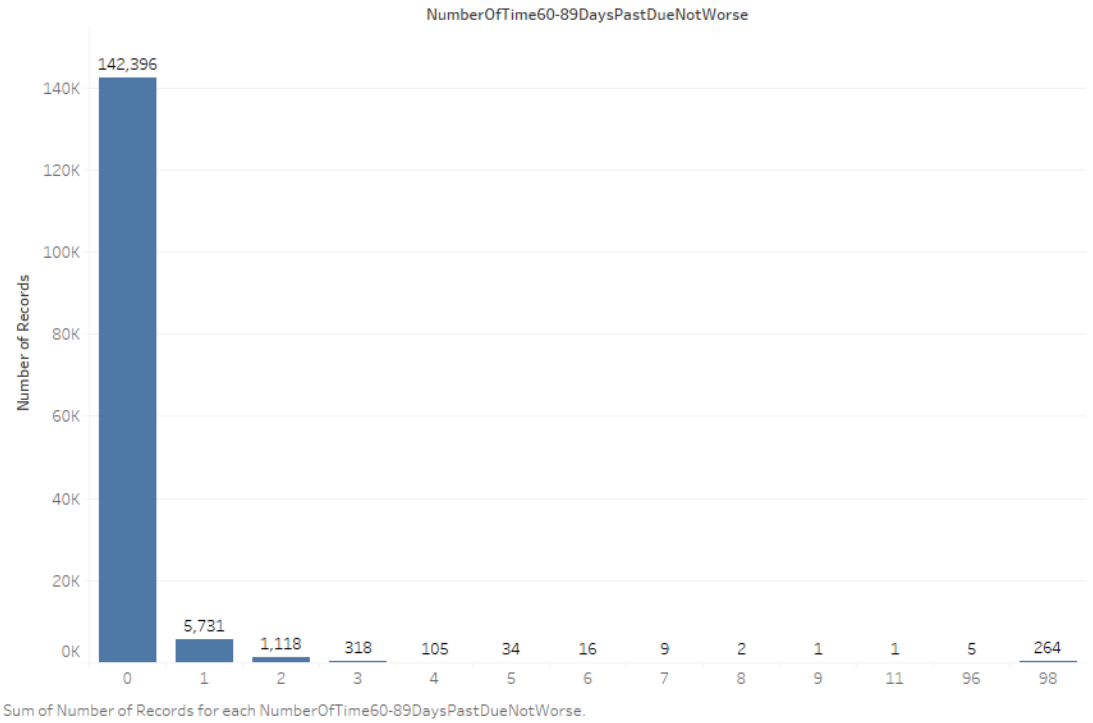
1) Age – Filtered out 1 record which had age=0 as such a case is not possible.

2) NumberOfDependents – Records which has "NA" were replaced with value 0 considering that there are no dependents.

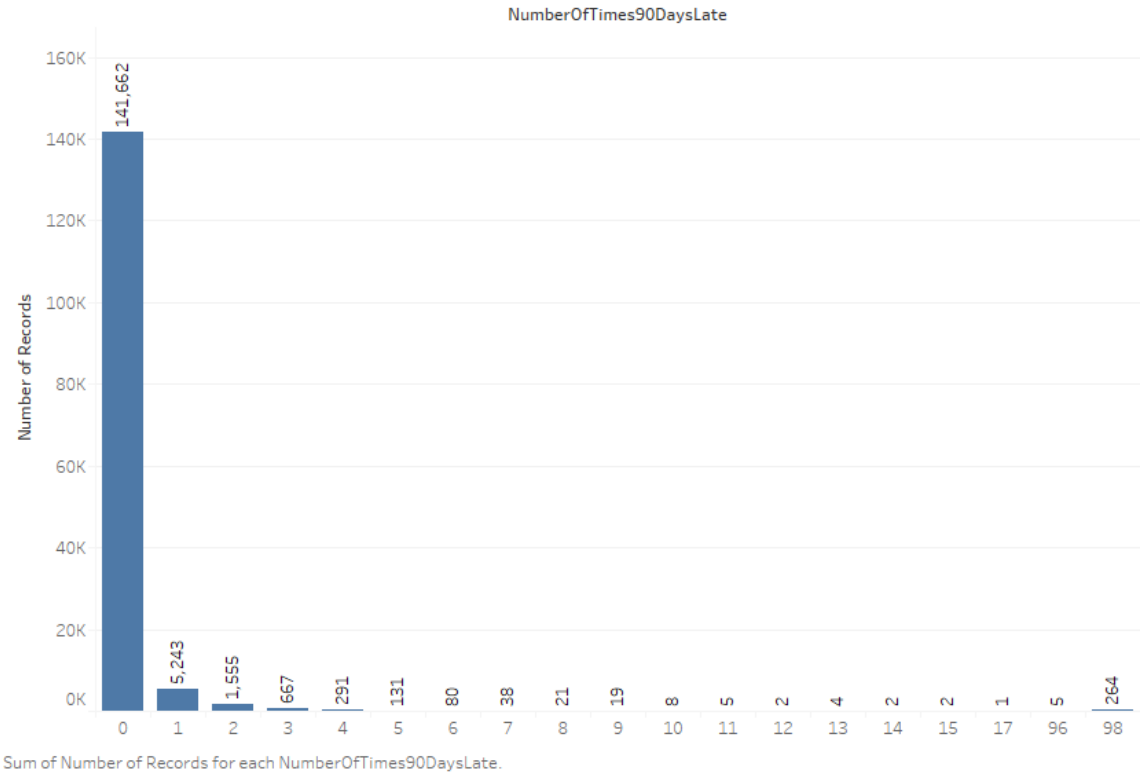
3) NumberOfTime30- 59DaysPastDueNotWorse – There are two values which are very high "96" and "98". It seems that they are either wrongly added or hard-code values for some purpose. To prevent skewing, I filter out these data points.



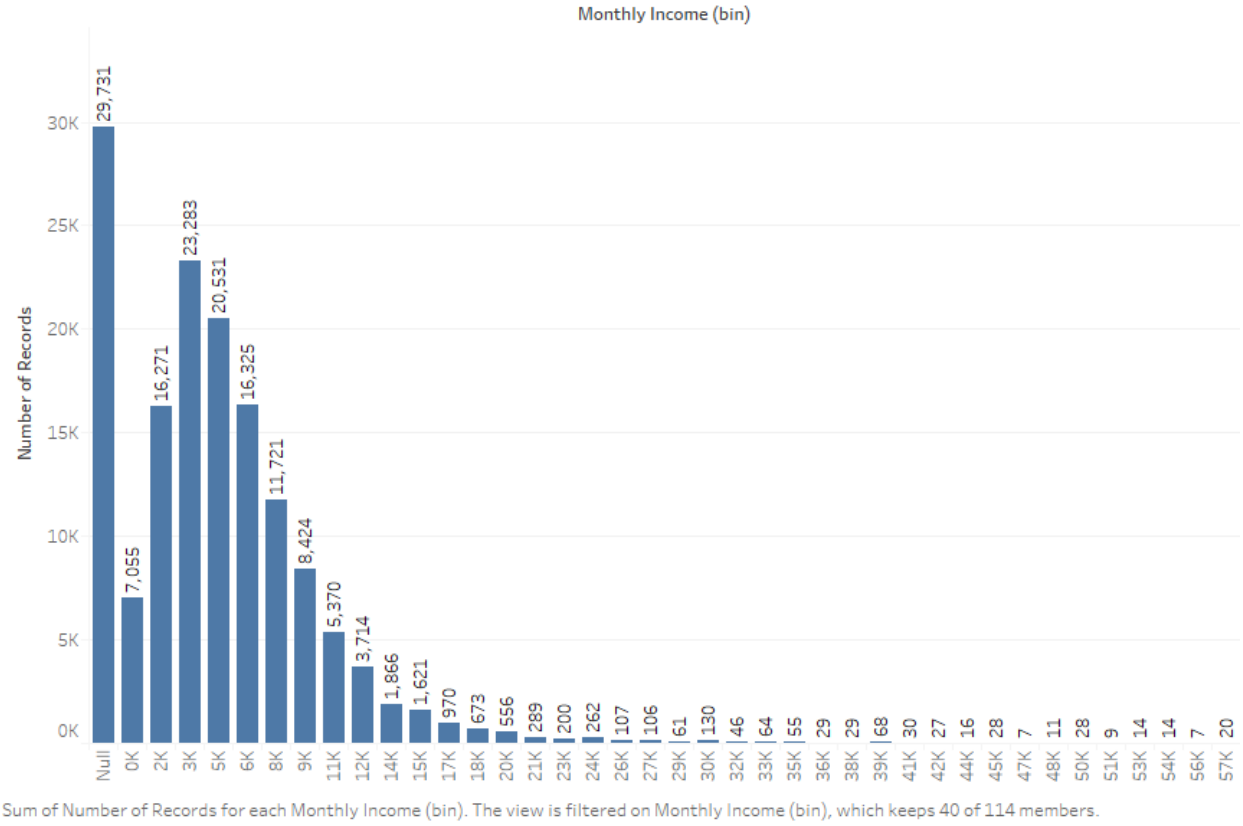
4) NumberOfTime60- 89DaysPastDueNotWorse – Similar to above, I filter out records with values "96" and "98".



5) NumberOfTimes90DaysLate - Similar to above, I filter out records with values “96” and “98”.

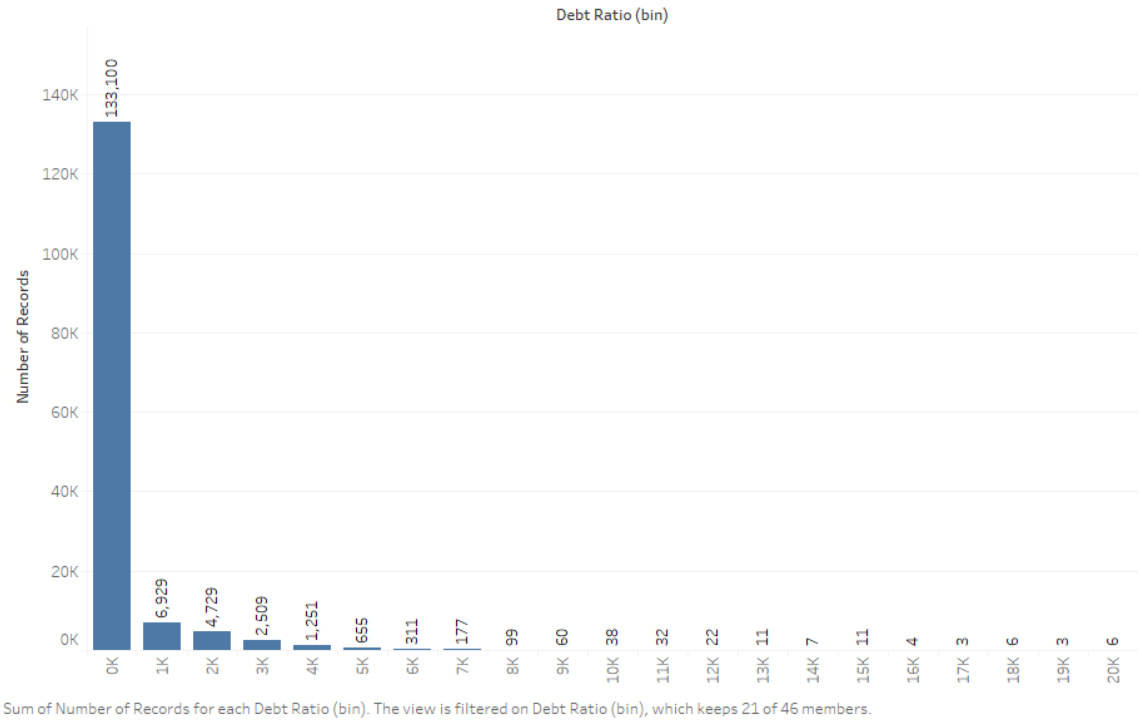


6) MonthlyIncome – A large number of records (29,731) are null, i.e., no values are available for them. There are several things that can be done to solve this issue: replace with “0”, replace with “-1”, replace with mean or median of the distribution. I tried all these combinations and find that the model performs the best when “NA” values are replaced with “0”.



Sum of Number of Records for each Monthly Income (bin). The view is filtered on Monthly Income (bin), which keeps 40 of 114 members.

7) DebtRatio – Debt ratio represents monthly debt payments divided by monthly gross income. The values vary quite a lot, thus, I only select values of DebtRaio < 15000.



Filter applied on different columns

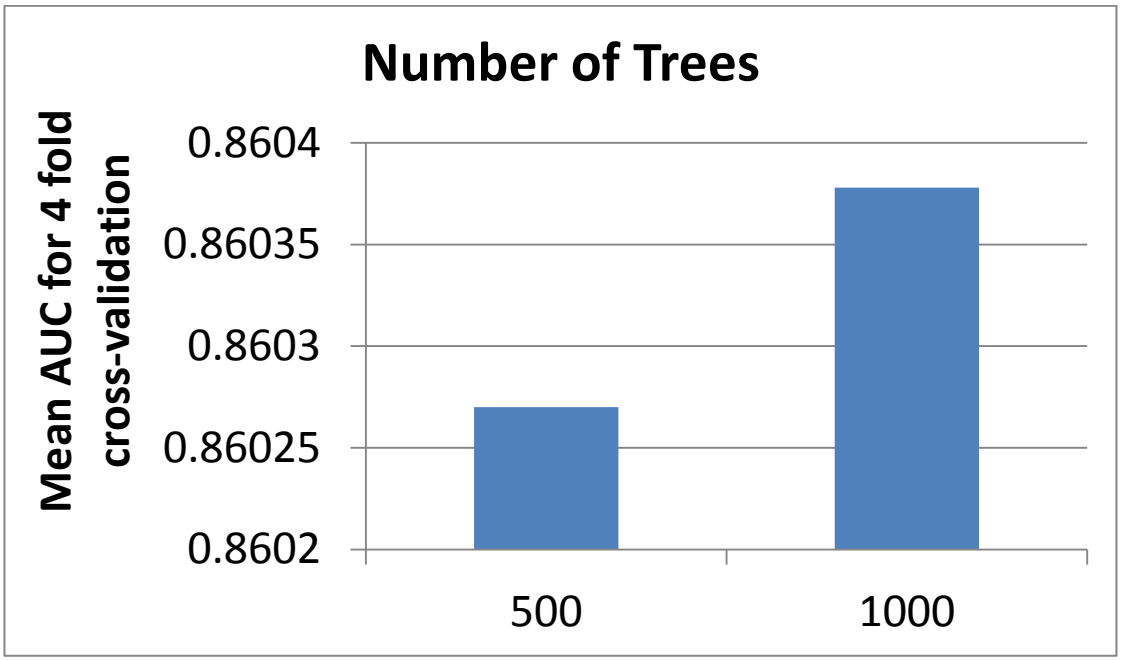
Feature	Filter
Age	**!=0**
NumberOfDependents	**“NA” replaced with “0”**
NumberOfTime30-59DaysPastDueNotWorse	**!=96 and !=98**
NumberOfTime60-89DaysPastDueNotWorse	**!=96 and !=98**
NumberOfTimes90DaysLate	**!=96 and !=98**
MonthlyIncome	**“NA” replaced with “0”**
DebtRatio	**<15,000**

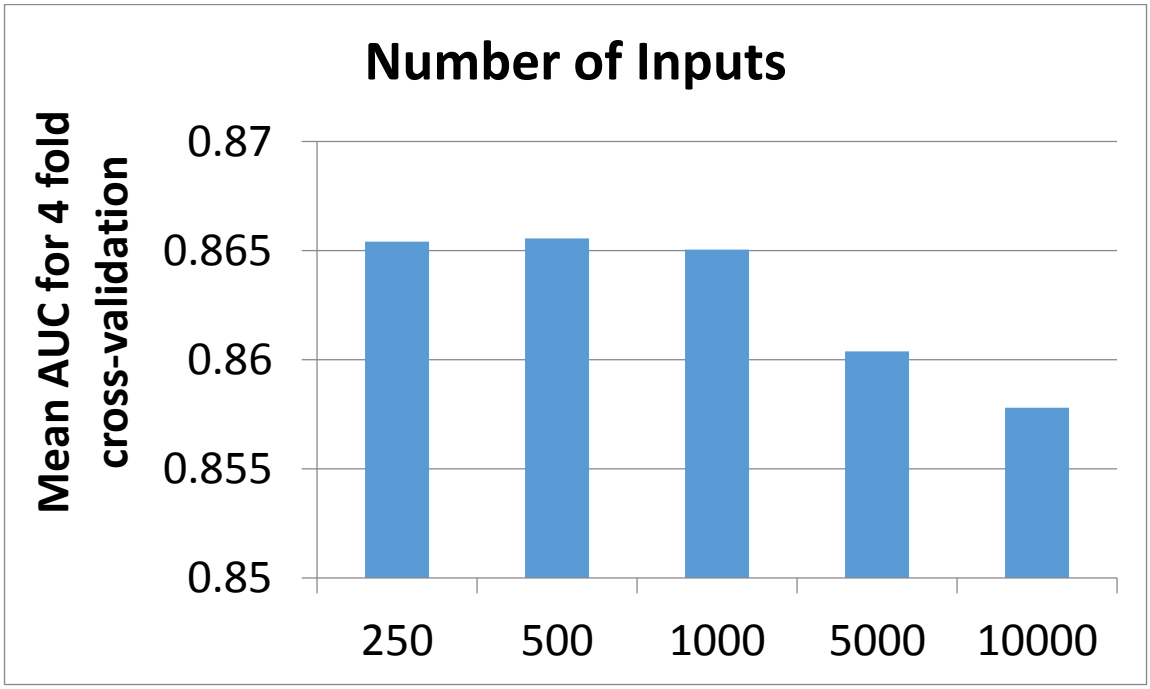
Parameter Tuning

1) Number of Trees – This value represents the number of trees to grow. Usually, higher number of trees give better results as it increases the probability that input record gets predicted few times. I

select 500 and 1000 for this parameter and find out that the model performs slightly better with the number of trees = 1000.

2) Sample Size – This value represents the size of sample drawn at each iteration of the split. The value selected for this parameter is 500.





Feature Engineering

I added new features to improve the performance of the model. These features are derived from the original features. The data type of each feature is specified in the brackets.

- NumberOfTime30-89DaysPastDueNotWorse (Count) = NumberOfTime30-59DaysPastDueNotWorse + NumberOfTime60-89DaysPastDueNotWorse

- NumberOfTimePastDue (Count) = NumberOfTime30-59DaysPastDueNotWorse + NumberOfTime60-89DaysPastDueNotWorse + NumberOfTimes90DaysLate

- NeverDue (Boolean) = (NumberOfTime30-59DaysPastDueNotWorse < 1 * NumberOfTime60-89DaysPastDueNotWorse < 1 * NumberOfTimes90DaysLate < 1)

- HasRealestateLoan (Boolean) = NumberRealEstateLoansOrLines > 0)

- MultipleLoans (Boolean) = NumberRealEstateLoansOrLines > 1

- LogMonthlyIncome (Float) = log(MonthlyIncome)

- HasDependants (Boolean) = Number of Dependents >0

- LogHasDependants (Float) = log(Number of Dependents)

- LogMonthlyIncome (Float) = log(MonthlyIncome)

- IncomePerPerson (Float) = MonthlyIncome/NumberofDependents

- LogIncomePerPerson (Float) = log(IncomePerPerson)

- LogDebtRatio (Float) = log(DebtRatio)

Evaluation Metrics & Results

I used Area Under the Curve (AUC) which is a common evaluation metric used in binary classification problems. This metric represents the area under the graph when true positive rate (TPR) (also called as sensitivity) and false positive rate are plotted against each other. TPR corresponds to the number of positive data points correctly classified as positive with respect to all the positive data points, whereas FPR corresponds to the proportion of negative points considered as positive.

This metric is also referred as Area Under the Receiver Operating Characteristic Curve (AUROC). If the classifier is good, it can increase the number of true positive and thus increasing AUC, whereas if the

classifier randomly classifies the AUC will be close to 0.5. As we are performing binary classification, AUC seems appropriate to evaluate our model's performance.

In this challenge, using the above features I obtain AUC = 0.867301 on the test (unseen) data. The ranking on the leaderboard comes out to be around 96. We can observe that model can predict credit score with a high value of AUC. This shows that the model performs well after performing some data cleaning, imputation and feature engineering. We can also observe that Random Forests can generate good prediction results with tweaking of its parameters.

Other Evaluation Metrics:

In this analysis, I have used AUC but other metrics are also commonly used by organizations.

Accuracy – Accuracy represents the proportion of samples classified as positive over all the samples examined.

$$Accuracy = TP + TN/TP + TN + FP + FN$$

where

TP – True Positive, i.e., Number of positive samples classified as positive

TN – True Negative, i.e., Number of negative samples classified as negative

FP- False Positive, i.e., Number of negative samples classified as positive

FN – False Negative, i.e., Number of positive samples classified as negative

However, accuracy alone is not a good measure. For example, if we the number of True Negatives (TN) increases the accuracy will increase, however, the true positive can be zero. In this case, the model has zero predictive power when it comes to correctly classifying the positive labels whereas we want the model to have higher TP to correctly classify positive labels. Thus, accuracy must be used in combination with other measures to avoid making wrong conclusions.

F-measure (F1) – F1 is the harmonic mean of precision and recall.

$$F1 = 2 * Precision * Recall/Precision + Recall$$

$$Precision = TP/TP + FP$$

$$Recall = TP/TP + FN$$

Precision represents the proportion of samples that are correctly classified as positive out of the all the samples, i.e., the ability of the classifier not to label positive samples as negative. Recall represents the proportion of samples classified as positive out of all the positive samples, i.e., the ability of the classifier to find all the positive samples. F1 measure is more stable than accuracy as it takes into account precision and recall. Depending on different situations, we can tweak the model to achieve higher precision or recall.