**Problem Framing**

|  |  |  |  |
| --- | --- | --- | --- |
|  | qualitative | quantitative | question |
| Current State | too many fraudulent transactions => less customers => less revenue => loss to the bank | 10% fraudulent transactions => 5% less customers => 5% loss in revenue | what is the avg. number of fraudulent transactions in the current situation and what we can do to decrease the number? |
| Objectives | * build a model that can detect fraud transaction before it gets completed * decrease fraudulent transaction => improve customer experience => more revenue | reduce fraudulent transaction by at least 5% => 5% more revenue | How we can detect the fraudulent transactions? |
| Benefit/ Cost Tradeoff and Prioratization | * cost of errors:   FN => Fraudulent transaction marked as non-fraudulent => risk increases => bad user experience => loss of revenue  FP => non-fraudulent transaction marked as fraudulent => less revenue   * benefits of correct predictions:   TP => correctly identified fraudulent transaction => customers are protected => better user experience => more revenue  TN => correctly marked non-fraudulent transactions => maintained user experience as expected => no significant impact on revenue | cost-benefit matrix   |  |  | | --- | --- | | c(TP) | c(FP) | | c(FN) | c(TN) |   1% TP =>  + 0.5% revenue  1% FN => 1% very bad experiences => 0.1% risk of customers’ assets => 10% less revenue  1% FP => -0.5% revenue  1% TN => no significant impact on revenue | what is the cost of errors/benefits of correct predictions and why? |
| Constraints | can only afford a small FN percent => very small percent of very bad user experience => limited risk of customers’ assets => limited loss of revenue | at most 10% FN => 1% very bad experiences => 0.1% churn => 0.1% risk of revenue loss => acceptable risk for 5% potential upside in revenue | what are the acceptable risks/budgets and why? |
| Desired State | * benefit: significantly lesser fraudulent transactions => significantly better user experience => significantly more customers => significantly better revenue * cost: very few false negatives => limited risk of very bad user experience => limited risk of customers’ assets => limited risk to revenue | * at least 50% decrease in fraudulent transactions (from 20% to 10%) => 5% more revenue * at most 10% false negatives => 1% very bad experience => 0.1% risk to revenue | what is the desired outcome (benefits/costs) that we want to see and why? |

**Why ML**

|  |  |  |  |
| --- | --- | --- | --- |
|  | qualitative | quantitative | question |
| best non-ML alternative  hypothesis | classify based on amount of money transaction => too many FP and FN => very bad user experience => lesser customer => loss of revenue | 50% FN 70% FP => not cleaning enough fraudulent transactions and causing more complaints for misplacing fraudulent transactions as genuine => 5% revenue loss risk | what are the non-ML alternatives and why are they problematic? (pains/missed gains)? |
| ML value proposition hypothesis | much fewer FP and FN => much better user experience => much better revenue | 10% FN 50% FP => 50% decrease in fraudulent transactions (from 20% to 10%) at the expense of 1% bad engagements => 5% increase in revenue at the expense of 0.1% risk | what are the advantages (pain relievers/gain creators) of ML solution and why? |
| ML feasibility  hypothesis | * data: labelled samples of each person’s transaction data * model: state of the art review suggests promising candidates are available | * data: around five thousand samples * model: state of the art claim solutions with 10% FN 20% FP | what data and model are good candidates and why? |

## ML Solution Design

|  |  |  |  |
| --- | --- | --- | --- |
|  | choices | metrics | experiment |
| data | (labelled) money transaction data | * label imbalance | * randomized 70/15/15 train/validation/test split |
| model | pr(fraud) | * AUCPR   (Precision recall curve) | * rule based heuristic * tf-idf + logistic regression * tf-idf + random forest * BERT + logistic regression   train these benchmark models (from simpler to more complex) using train data. validate and tune using validation data. select the model with best AUCPR on test data |
| action | if pr(fraud) > threshold: auto take down | * precision * recall * confusion matrix | * choose a threshold to maximize the recall (estimated reward) subject to recall > 90% |
| reward | * decrease in fraudulent transaction * cost of misclassification | * % decrease in fraud * % increase in daily active users | * shadow test * A/B test |