**Credit Risk Project**

Goal of this project is to practice what we have discussed so far. We will define a new credit card strategy, and compare it with the existing strategy.

1. Download data from <https://www.kaggle.com/competitions/amex-default-prediction/data>. We will work with “train\_data.csv” and “train\_labels.csv”.
2. Train.csv contains several months of data for each customer. Randomly choose only one month of data for each customer. This will be our development sample. It should have 458,913 observations. Also it should have around 30,000 to 40,000 observations, each year\_month.
3. Merge this data with “train\_labels.csv”, to attach the dependent variable (target). Remove observations with missing target (if any). **Save this data**, so in the future you don’t have to read the original large file again.
4. Explore the data. Data Size, data type of features, a snapshot of data, …
5. Next we want to reduce number of features, and keep only features which have high predictive power. To do so we build an XGB model and will keep features with Feature Importance higher than 0.5%. To run the XGB model, we need to do one-hot encoding. Perform One\_Hot encoding on categorical features.

**Make sure all missing values are stored as NaN, so XGBoost can work with them.**

**As you know, we don’t need to do outlier treatment and normalization for XGB.**

1. Split data to 201705-201801 as Train sample, 201703-201704 as Test1, and 201802-201803 as Test2.
2. Run an XGBoost model on the train sample, with default parameters. Don’t forget to drop unnecessary columns. Calculate feature importance and save the feature importance as a CSV file.

**Note**: **Each time you run the XGB, set random\_state to a number of your choice. That will enable you get the same results, if you need to re-run the model in the future.**

1. Run another XGBoost model, which has 300 trees, 0.5 as learning rate, maximum depth of trees is 4, uses 50% of observation to build each tree, uses 50% of features to build each tree, and assigns a weight of 5 to default observations. Save the feature importance as a CSV file.
2. Keep features that have feature importance of higher 0.5% in any of the two models. We will use only these features after this.

**Note, optimum would be to use all features in the grid search. Also optimum is to test all the possible parameters for grid search! But we sacrifice a little in model’s performance, but gain a lot in computational efficiency. At the end, the sacrifice has minor impact on model’s performance, and even lower impact on the strategy and business results.**

1. Next we run Grid Search for the XGBoost model (using only features we chose in step 9). Use the following combinations in the grid search:
   * Number of trees: 50, 100, and 300
   * Learning Rate: 0.01, 0.1
   * Percentage of observations used in each tree: 50%, 80%
   * Percentage of features used in each tree: 50%, 100%
   * Weight of default observations: 1, 5, 10

Create the following table. **Update the table after each iteration of grid search and save the table**, so in case you got memory error or any other issues, you don’t need to re-run that part of Grid.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| # Trees | LR | Subsample | % Features | Weight of Default | AUC Train | AUC Test 1 | AUC Test 2 |
| 50 | 0.01 | 50% | 50% | 1 | … | … | … |
| … | … | … | … | … | … | … | … |

1. Choose the best model, based on bias and variance. Re-run the model with optimum parameters, and save the final XGB model.
2. Next, grid search for Neural Network. We first need to process the data. We have already done one-hot encoding. We need to do Missing Value Imputation, Outlier Treatment, and Normalization. **We will use only features that we chose in step 9**. Replace outliers with 1 and 99 percentiles. Use StandardScaler for normalization (standardization). Replace missing values with 0.

**As you know, you should get values for 1 and 99 percentiles, as well as Mean and Standard Deviation values for scaling, only based on the Train sample. Then you should apply the same value to Test samples (or any other sample). In other words, for each observation in the test sample (or any other sample), you should first do outlier treatment based on 1 and 99 percentiles of the train sample, and Standardize it based on Mean and Standard Deviation from the train sample.**

1. Next we run Grid Search for the Neural Network model (using only features we chose in step 9). Use the following combinations in the grid search:
   * Number of hidden layers: 2, 4
   * # nodes in each hidden layer: 4, 6
   * Activation function for hidden layers: ReLu, Tanh
   * Dropout regularization for hidden layers: 50%, 100% (no dropout)
   * Batch size: 100, 10000

Use Adam for optimizer, Cross Entropy for Loss function, and 20 for number of Epochs. For everything else, use default parameters.

**Note you would need to run separate For Loops for different number of Hidden Layers.**

Create the following table. **Update the table after each iteration of grid search and save the table**, so in case you got memory error or any other issues, you don’t need to re-run that part of Grid.

|  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- |
| # HL | # Node | Activeation Function | Dropout | Batch Size | AUC Train | AUC Test 1 | AUC Test 2 |
| 2 | 4 | ReLu | 50% | 100 | … | … | … |
| … | … | … | … | … | … | … | … |

1. Choose the best model, based on bias and variance. Re-run the model with optimum parameters, and save the final NN model.
2. Choose the best model among NN and XGB (models of step 11 and step 14)

**Strategy:**

Next, you want to define two strategies: a conservative and an aggressive. For each strategy, you define a threshold to accept/reject applicants based on model’s output. Applicants with probability of default (model’s output) lower than threshold, will be accepted, and those with PD higher than threshold will be rejected. The conservative strategy has a lower threshold compared with aggressive one; hence accepts less applicants.

We will estimate default rate, and Revenue based on each strategy, show it to management, and let them decide which strategy is better.

Calculate Default Rate: You already know how to calculate default rate for a strategy; you just need to calculate default rate in the sample, for those with PD less than threshold.

Calculate Revenue: Revenue on a credit card depends on two factors: how much the customer spends, and how much of monthly balance the customer does not pay (roll over to the next month). Credit Card companies, charge a small amount for each dollar you spend. Also they charge an interest rate on the remaining monthly balance that you do not pay (revolving balance).

For example, assume a CC charges 0.1% on each dollar spent, and charges 24% (annually) on balances. If a customer spend $1000 in a month, company’s revenue from spend of this customer will be 1000×0.001=$1. If customer pays back $200 out of $1000, company will charge 2% monthly interest on the remaining $800, which means $16 interest revenue in that month.

**Note:** As you know interest rates on CC balances are very high, so don’t be manipulated by banks; i.e. don’t spend too much. You are most attractive with a cheap, healthy life, with a lot of exercise. Also Never Default on your Debt; i.e. never miss the minimum monthly payment.

So, to estimate revenue, you need to have a measure of Spend and Balance. In the data, features that start with S\_ are spend variables, and features that start with B\_ are balance variables. Choose one spend and one balance feature (any feature of your choice). Revenue for a strategy would be calculated as:

Calculate sum among customers who are accepted based on your threshold. Assume a revenue of 0 for those who default.

1. Write a function that calculates default rate and revenue based on a threshold. Function gets three inputs:
   * Data (sample)
   * Name of Target Variable (as a text/string)
   * Threshold (a number between 0 and 1)

And will return two outputs: default rate, and revenue.

Use only train sample to try a few thresholds, and choose one conservative and one aggressive strategy. It is up to you how to choose the thresholds. Imagine you want to present it to senior management and want to impress them with your work/results. The only constraint is that company does not want the default rate to be higher than 10%.

**Prepare the presentation:**

**General Guidelines:**

1. Create pretty slides
2. Don’t use any background
3. Format numbers, use 1000 separators. Decimal numbers with 2 decimal places (in case of very small numbers with 3 decimal places)
4. Don’t use small fonts that can not be seen
5. Don’t put too much material in a slide
6. Each slide should be self explanatory. While you don’t want to put too much material, put enough material that explain the stuff in the slide
7. Format tables. Assign appropriate titles to tables and figures
8. Don’t copy paste from your code
9. Have a good story to tell
10. Format everything. Standard fonts …
11. Use colors, but don’t overuse

In general, remember a presentation is like presenting a product. Both packaging and functionality matter. **You need to wrap your good model in a pretty package.**

**Note 1:** In the following steps, feel free to change the format of tables to make the slides easier to follow and understand.

**Note 2:** I have proposed the minimum items to be included in the slides. Feel free to add additional explanations, …

**Note 3:** Due to computational constraints, you may need to simplify the project, read less rows and observations, … Adjust the following tables based on your final sample.

Fill the attached deck with your results.



**Slide #1. Executive Summary**. This is where you sell your model. Show the results of your strategies, and add any explanation that can attract people. In this slide imagine you are a seller. Include the following table. Talk about the project, project’s goal, why this project is important, how it helps the company, and anything that might be interesting (like these days people get excited when they hear AI …)

Propose the strategy that you think help the company better. Explain why you think this is a better strategy.



**Slide #2. Data.** Explain your data (data of step 3). Include the following table, explain why you decided to use this data, explain your target variable (you can generate a story for what default means), …



**Slide #3. Features.** Talk about categories of independent variables used in the development process (data of step 3).



Also show summary statistics for the top 5 features with highest SHAP values in the best XGBoost model (Step 11). Use raw (un-processed) data to create summary statistics (following table). Note that at this point you don’t need to talk about the XGBoost model and SHAP. You can just mention that based on your analyses these are among the most important attributes.



**Slide #4. Sampling.** Define Train, Test 1, and Test 2 samples. Explain why you chose test and train samples in this way. Include the following table.



**Slide #5. Data Processing / One-Hot Encoding.** Show the categorical variables, and show how you treated them. Show the results after One-Hot Encoding. Include your code to do one-hot encoding.

**Slide #6. Feature Selection.** Add a graph that explains your feature selection technique (steps 7 to 9). Create a pretty graph. Attach an excel file with results of feature importance for two models (steps 7 and 8). Add a column to table of slide 3, that shows # features selected from each category to be used in grid search.



**Slide #7. XGBoost - Grid Search.** Include your grid search code. Explain why you chose these parameters (don’t say because you said …). Talk about your experience with grid search, how many models you trained, any lessons learned, …

**Slide #8. XGBoost - Grid Search.** In this slide, we create scatter plots for models of grid search, and will choose the best model based on the scatter plot. For each of the models of grid search, calculate average and standard deviation of AUC across three samples (train and tests). Then include 2 scatter plots in the slide:

* In the first one, X\_Axis shows Average AUC, and Y-Axis shows Standard Deviation of AUC.
* In the second one, X-Axis is AUC of train sample and Y-Axis is AUC of Test 2 sample.

Explain which model you would choose based on each scatter plot.

**Slide #9. XGBoost – Final Model.** Show the parameters of the final model, also AUC of model on each sample. Also show how model Rank Orders on each of the three samples. Check the last part of XGBoost sample code, for rank ordering. Note you need to define score bins based on the train sample, and apply the same thresholds to test samples. Show rank orderings in a Bar-Chart, where each sample is one series in Bar Chart, X-Axis shows score bins (intervals), and Y-Axis shows default rate in each bin.

**Slide #10. XGBoost – SHAP Analysis.** Show Beeswarm Graph for the final model, based on Test 2 sample. Add some explanation of your choice. You can talk about ranking of attributes, correlation between attribute and the output, …

**Slide #11. XGBoost – SHAP Analysis.** Show Waterfall Graph for the final model, based on one observation in Test 2 sample. Add some explanation of your choice. You can talk about which attributes are driving the score, how to improve the score, …

**Slide #12. Neural Network – Data Processing.** Explain your data processing for Neural Network. Feel free to add code, tables, … Format this slide, so it is easy to follow and understand.

**Slide #13. Neural Network - Grid Search.** Include your grid search code. Explain why you chose these parameters (don’t say because you said …). Talk about your experience with grid search, how many models you trained, any lessons learned, …

**Slide #14. Neural Network - Grid Search.** In this slide, we create scatter plots for models of grid search, and will choose the best model based on the scatter plot. For each of the models of grid search, calculate average and standard deviation of AUC across three samples (train and tests). Then include 2 scatter plots in the slide:

* In the first one, X\_Axis shows Average AUC, and Y-Axis shows Standard Deviation of AUC.
* In the second one, X-Axis is AUC of train sample and Y-Axis is AUC of Test 2 sample.

Explain which model you would choose based on each scatter plot.

**Slide #15. Neural Network - Grid Search.** Show the parameters of the final model, also AUC of model on each sample. Also show how model Rank Orders on each of the three samples. Check the last part of XGBoost sample code, for rank ordering. Note you need to define score bins based on the train sample, and apply the same thresholds to test samples. Show rank orderings in a Bar-Chart, where each sample is one series in Bar Chart, X-Axis shows score bins (intervals), and Y-Axis shows default rate in each bin.

**Slide #16. Final Model.** Talk about the final model (XGBoost or Neural Net), and why you chose this one. Add tables or graphs from previous steps to support your reasoning …

**Slide #17. Strategy.** Include the function you have written in part 16. Also include the following table. Explain what thresholds you chose for conservative and aggressive strategy, and explain your rationale.

