1. Background

It draws our attention that as the development of business services, many people struggle to get loans due to insufficient or non-existent credit histories. And, unfortunately, this population is often taken advantage of by untrustworthy lenders.Under such situations, this ADS is developed to predict loan default probabilities, a common but complex problem in financial services. The system uses machine learning techniques to analyze credit applications, providing predictions that help financial institutions decide on loan approvals and aiming to make sure the underserved population has a positive loan experience.

The complexity and impact of this ADS align with the course's focus on responsible data science. Issues of fairness, bias, and transparency are particularly pertinent given the ADS's influence on financial decisions that significantly affect individuals' lives.

2. Input and Output

Data Description:

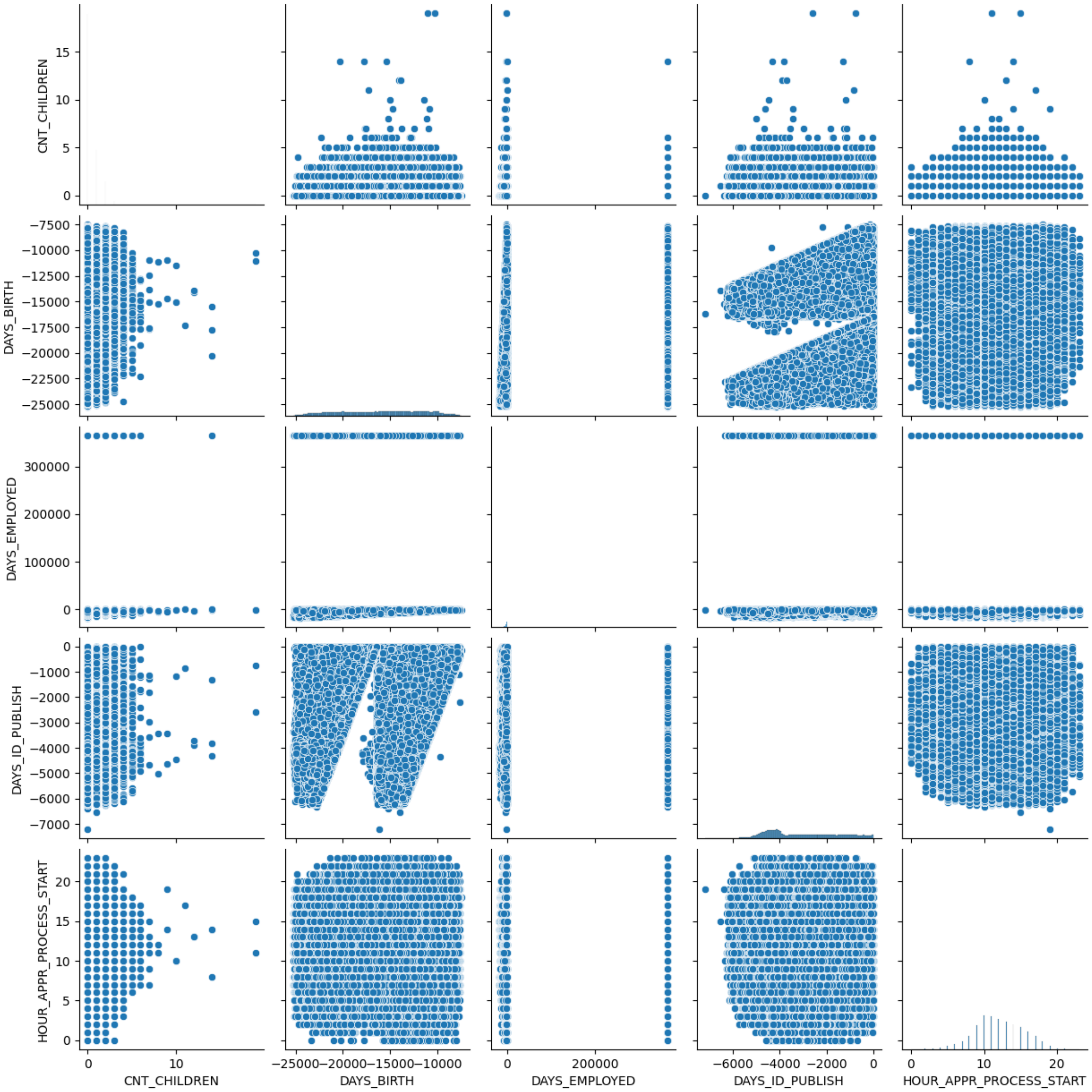
The ADS uses a variety of data sources provided by Home Credit, incorporating features from historical credit performance, current financial behavior, and demographic information.

The data spans several specific datasets:

- application_train/application_test: These datasets contain static data for all loan applications. Each row represents a single loan application and includes a wide range of input features from the applicant's personal information to details about the loan.
- bureau: This dataset includes information about the client's previous credits from other financial institutions reported to the credit bureau.
- bureau_balance: Monthly balances of previous credits that the client had in the Credit Bureau.
- POS_CASH_balance: This includes monthly balance snapshots of previous POS (Point of Sale) and cash loans that the applicant had with Home Credit.
- credit_card_balance: Monthly balance snapshots of previous credit cards held by the applicant with Home Credit.

- previous_application: Historical data about all previous applications each applicant made before the current application.
- installments_payments: Repayment history for previously disbursed credits.

Input Feature:



The input data includes both categorical data and numerical data – 'CODE_GENDER', 'FLAG_OWN_CAR', 'FLAG_OWN_REALTY' are categorical data(which would be transformed into binary afterwards) and a bunch of numerical data in the bureau.csv file 'DAYS_CREDIT' , 'DAYS_CREDIT_ENDDATE', 'DAYS_CREDIT_UPDATE', 'CREDIT_DAY_OVERDUE', 'AMT_CREDIT_MAX_OVERDUE', 'AMT_CREDIT_SUM', 'AMT_CREDIT_SUM_DEBT', 'AMT_CREDIT_SUM_OVERDUE', 'AMT_CREDIT_SUM_LIMIT', 'AMT_ANNUITY', 'CNT_CREDIT_PROLONG', 'MONTHS_BALANCE_MIN', 'MONTHS_BALANCE_MAX', 'MONTHS_BALANCE_SIZE'(Some of these data are being used to create new features/variables).

- From the scatterplot above, we can see that the clear relationship between age and employment length could be used to predict financial stability or risk, as longer employment might correlate with more stable financial behavior.
- The lack of correlation between the number of children and other factors like age or employment suggests that having children is independent of these factors, which might imply that credit risk related to family size needs to be modeled separately from age or employment.
- The uniform distribution of application times across different demographics could suggest that the time of application isn't a significant factor in application outcomes, which might simplify modeling by omitting time of day as a predictor of loan approval or default risk.

One noticeable point in the dataset is that there are 61% null data overall, which indicates a limited availability of some information. Also we notice that 24% of the input data are normalized. During feature engineering, missing values are handled by the np.where() function and fillna(0) function which directly replace missing values with 0. This is particularly seen after ratios are calculated, and if the denominator was zero even after the substitution by np.where(), the resultant infinity or NaN from operations like division would be converted to 0. This can be useful to maintain numeric consistency across data but might mask potentially valuable signals of missingness. However, It may be better in some cases to use imputation techniques that consider the distribution of the data or the relationships between variables.

Output of ADS:

The output of this ADS is a probability that a client will experience difficulty repaying a loan. More specifically, it quantifies the risk that a client will have a payment that is more than a specified number of days late on at least one of the first few installments of the loan. The probability is between 0 and 1. To use the probability in decision making, we set a threshold. If the probability is above this threshold, the application might be flagged as high risk, potentially leading to a denial of the loan or the provision of the loan under stricter conditions. If the probability is below the threshold, the applicant might be considered low risk, and the loan could be approved under standard conditions.

3. Implementation and Validation

It is quite true that data cleaning and preprocessing is the most time-consuming process in the whole data science project. We can truly feel this from this ADS, which deals with a large amount of data.

It has caught our attention that some preprocessing methods in this ADS like merge, are very useful and common when dealing with large datasets. Since the datasets are various and the data of the same subject may be put in different csv files, it would be nice to merge two related datasets into an integrated one and do the analysis. In addition, we noticed that it would be nice to delete the original dataset after merging – it would save a lot of memory for the computer especially when dealing with large datasets.

For data cleaning in the ADS, especially the way to deal with NULL values, one noticeable point is that the authors simply replace the NULL with 0. No matter the data is about the pos_cash or bureau or others, they just fill all the N/As with 0. We think by this way to deal with the NULL values may not be proper. Maybe it makes sense to some unimportant features, but for example, when dealing with features in dataset, bureau, which contains application data from previous loans that client got from other institutions, it may not be nice to just use 0 to set all NULL values since the features are crucial in the model afterwards. The 0s may cause the data to be biased. Combining with other solutions and our insights, in this case, it may be better to replace the NULLs with computed mean values for the column for a specific feature. The other thing to mention here is that the authors drop the unrelated columns to reduce the complexity of the dataset. Since the dataset is so large and contains too many related features, it would be great to remove some useless features. For example, the authors dropped 'NAME_CONTRACT_STATUS' in pos_cash. This feature would not be used in the further analysis so it is proper to do so and it would make the dataframe more concise. To further organize and simplify the dataset, authors used '_'.join(col).strip() to combine columns. This method joins the elements of the col list (which contains the words of a column name) with underscores.

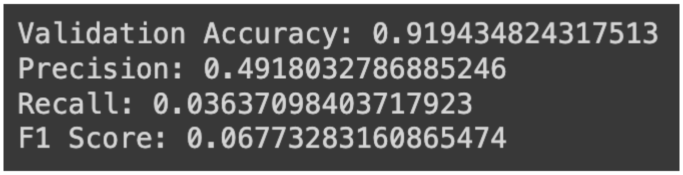
Another point that is worth mentioning when preprocessing data is that the authors aggregated the dataset and used methods like groupby to better organize the data for further analysis. Aggregation helps to summarize data within a dataset by grouping rows and applying a function (e.g., sum, average, count) to each group. It collapses multiple rows of data into a smaller set of summary rows and makes the dataset have fewer rows than the original dataset, where each row represents a group of data with aggregated values. Groupby makes data more organized by specifying a particular feature. The authors use these two methods a lot in this ADS, almost all the datasets – prev_app, pos_cash, inst_pmts etc.

The model used in this ADS is XGBclassifier. XGBoost stands for Extreme Gradient Boosting, which is an efficient and scalable implementation of gradient boosting machines (a machine learning technique used for both regression and classification tasks. It works by combining multiple weak learners (usually decision trees) into a strong learner in an iterative manner). The implementation of XGBclassifier is straightforward in our chosen ADS. But the merging process before applying the model is important. After the preprocessing and data cleaning methods we have mentioned above, the authors merge all the aggregated datasets. Then, they computed and converted data into desired types. We will provide two examples, one for numerical and one for categorical here: 1. “test['LOG_INCOME_CREDIT_RATIO'] = np.log((test['AMT_INCOME_TOTAL'] / np.where(test['AMT_CREDIT'] != 0 ,test['AMT_CREDIT'],1)).fillna(0))” This line calculates the the income_credit_ratio as a natural logarithm based on the available, processed data and get a new feature called 'LOG_INCOME_CREDIT_RATIO'. Notice that, as we mentioned above, the authors still fill 0 in the NULL values. 2. “test['CODE_GENDER'] = test['CODE_GENDER'].isin(['F']).astype(int)” (This looks similar to One-hoe encoder but not the same.) This code checks each value in the 'CODE_GENDER' column to see if it is equal to 'F'. The isin(['F']) method returns a boolean Series with True where the value is 'F' and False otherwise and then converts the boolean values (True/False) to integers (1/0). The astype(int) method converts True to 1 and False to 0. After performing all the similar operations and the transformation process, the ‘test’ is ready to be processed by the model. Then they used the model to predict the probability of loan application for each applicant.

The validation of this ADS is based on the ROC AUC score. ROC AUC (Receiver Operating Characteristic Area Under the Curve) is a popular evaluation metric used to assess the performance of binary classification models. It measures the area under the ROC curve, which is a plot of the true positive rate (TPR) against the false positive rate (FPR) at various threshold settings. It represents the probability that the model will rank a randomly chosen positive instance higher than a randomly chosen negative instance. A higher ROC AUC score indicates better discrimination between positive and negative classes. In the ADS, the authors achieved a score of around 78%. This is not a very high score but considering the size of dataset and the presentation of large NULL values, we think this score is good enough. The ADS did a great job in prediction.

4. Outcome Analysis

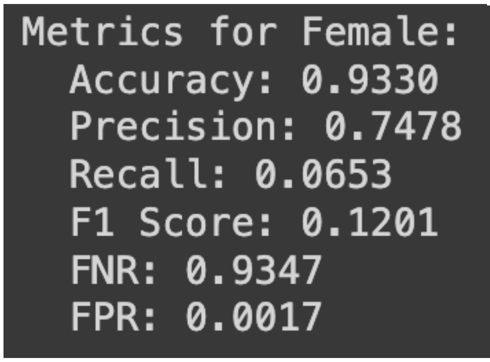
Accuracy metrics of the ADS:



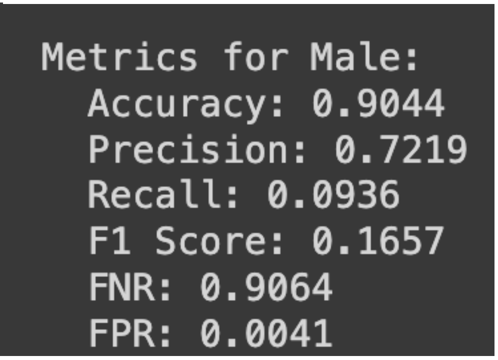
```
Validation Accuracy: 0.919434824317513
Precision: 0.4918032786885246
Recall: 0.03637098403717923
F1 Score: 0.06773283160865474
```

The creator used the Gradient Boosted Tree Model in this ADS. The overall accuracy of the ADS is quite promising, with an accuracy of 0.919. This indicates that the model is generally good at predicting the outcome for the majority class (those without payment difficulties). However, the context of this problem suggests that the high accuracy might be misleading due to the potential class imbalance in the dataset.

Accuracy of ADS across different genders:



```
Metrics for Female:
  Accuracy: 0.9330
  Precision: 0.7478
  Recall: 0.0653
  F1 Score: 0.1201
  FNR: 0.9347
  FPR: 0.0017
```



```
Metrics for Male:
  Accuracy: 0.9044
  Precision: 0.7219
  Recall: 0.0936
  F1 Score: 0.1657
  FNR: 0.9064
  FPR: 0.0041
```

We created a function called evaluate_metrics to compute the performance metrics. It takes the true labels of the data set which is the TARGET feature, the predicted labels, the entire dataset which contains the features and labels, and the column name in the dataset that identifies the subgroups for which the metrics are to be computed.

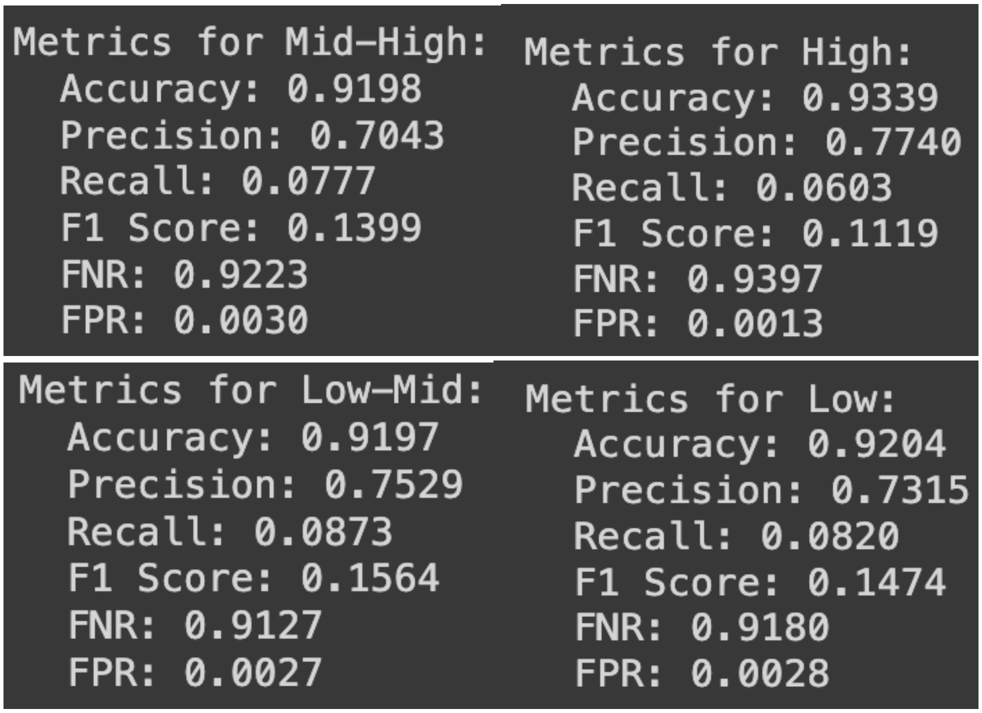
Male:

- Recall is very low (0.0936), indicating that the model struggles to correctly identify males who have payment difficulties. This results in a high FNR (0.9064), meaning most true positive cases are missed for males.
- Precision (0.7219) and Accuracy (0.9044) are relatively better, but the low recall significantly impacts the F1 score (0.1657), showing an imbalance in the performance measures.

Female:

- Similar to males, recall is also low (0.0653) for females, leading to a high FNR (0.9347), suggesting that the model also misses identifying most females with payment difficulties.
- Precision (0.7478) and Accuracy (0.9330) are higher than those for males, which could indicate a slightly better identification of true negatives, but the low recall still severely impacts the F1 score (0.1201).

Accuracy of ADS across different income levels:



```
Metrics for Mid-High:
  Accuracy: 0.9198
  Precision: 0.7043
  Recall: 0.0777
  F1 Score: 0.1399
  FNR: 0.9223
  FPR: 0.0030
```

```
Metrics for High:
  Accuracy: 0.9339
  Precision: 0.7740
  Recall: 0.0603
  F1 Score: 0.1119
  FNR: 0.9397
  FPR: 0.0013
```

```
Metrics for Low-Mid:
  Accuracy: 0.9197
  Precision: 0.7529
  Recall: 0.0873
  F1 Score: 0.1564
  FNR: 0.9127
  FPR: 0.0027
```

```
Metrics for Low:
  Accuracy: 0.9204
  Precision: 0.7315
  Recall: 0.0820
  F1 Score: 0.1474
  FNR: 0.9180
  FPR: 0.0028
```

We used the cut method in pandas to segment AMT_INCOME_TOTAL feature into 4 categories. And I examined each category with different accuracy metrics.

- Across all income brackets from Low to High, the recall is extremely low (ranging from 0.0603 to 0.0873), resulting in very high FNRs (above 0.9127 in all cases). This demonstrates the model's general inability to correctly identify individuals who face payment difficulties across all income levels.
- Precision varies more significantly between brackets, being highest for the High income bracket (0.7740) and lowest for the Mid-High bracket (0.7043).
- Accuracy remains high across brackets, suggesting that the model is effective at identifying true negatives but at the cost of failing to catch true positives.
- F1 scores are low across all brackets due to low recall rates, highlighting that the model is not effectively balancing the precision and recall.

Advantages of Choosing These Metrics:

1. Precision:

   In the context of loan repayment predictions, precision is crucial because it measures the reliability of the model in predicting clients who may face payment difficulties. High precision means that when the model predicts a loan will not be repaid, it is likely correct. This helps in minimizing the financial risk by focusing attention on truly high-risk clients.

   Resource Allocation: High precision ensures that resources are efficiently allocated, such as offering special intervention programs only to those who are likely to need them.
2. Recall:

   Recall is important in this context because a high recall rate means the model is effective at identifying most of the actual defaulters. This is critical to minimize losses from bad loans.

   Customer Support: Identifying more individuals who are likely to face payment issues allows the institution to proactively offer support or restructuring options, potentially aiding in recovery efforts and customer retention.
3. F1 Score:

   The F1 Score is particularly useful because it provides a single metric that balances both precision and recall. This is important when both identifying as many positive cases as possible (recall) and ensuring the cases identified are indeed positive (precision) are equally important.

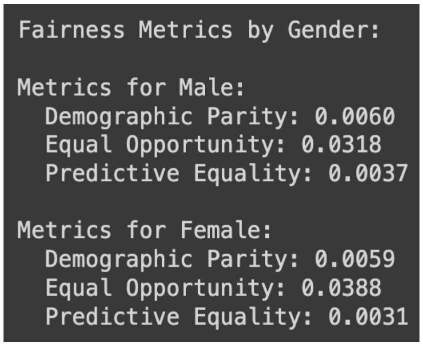
Model Tuning: Helps in tuning the model towards an equilibrium where neither precision nor recall is sacrificed excessively, promoting a more robust predictive model.

4. False Negative Rate (FNR) and False Positive Rate (FPR):

   Minimizing False Negatives: In loan default prediction, minimizing false negatives (FNR) is critical because failing to identify a potential defaulter can lead to significant financial losses.

   Controlling False Alerts: Minimizing false positives (FPR) is also crucial to prevent the financial institution from unnecessarily penalizing or inconveniencing clients who are likely to repay their loans. This helps maintain trust and satisfaction among clients.

Evaluation using fairness metrics:



```
Fairness Metrics by Gender:

Metrics for Male:
  Demographic Parity: 0.0060
  Equal Opportunity: 0.0318
  Predictive Equality: 0.0037

Metrics for Female:
  Demographic Parity: 0.0059
  Equal Opportunity: 0.0388
  Predictive Equality: 0.0031
```

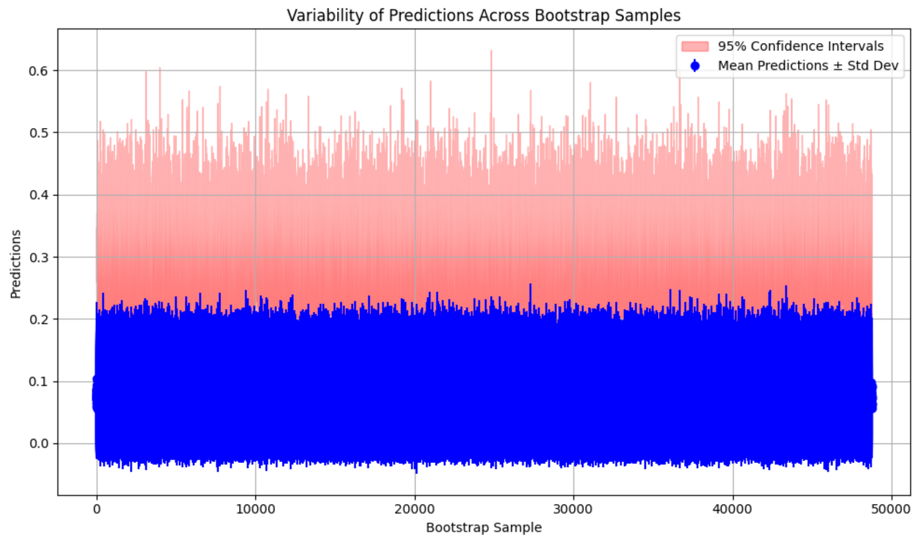
To evaluate the fairness of the ADS across different genders, we used several fairness metrics including Demographic Parity, equal opportunity and predictive equality.

- We choose to evaluate the Demographic Parity because we want to make sure that the prediction of whether clients will pay loans is only dependent on related features like income levels etc. As we can see from the above figure, the DP is quite close between males and females, suggesting that the model gives positive outcomes to both genders at almost equal rates. This closeness in values indicates a relatively fair approach by the model regarding decision frequency across genders.
- Evaluating the Equal Opportunity metrics ensures that the model is equally good at identifying true positive outcomes (loan repayment) for all genders. It helps in minimizing bias that could prevent a specific gender from accessing loans due to a model’s inability to recognize their potential to repay. In credit lending, it's vital that all individuals who are

likely to repay are given an opportunity to get a loan. Equal Opportunity shows a slightly higher rate for females compared to males. This suggests that the model is slightly more accurate at identifying true positive outcomes for females than for males, which could be seen as beneficial fairness if historical data has underrepresented female positives.

- Predictive equality is crucial for trust and ethical standing. Incorrectly labeling a potential customer as a defaulter can have severe personal consequences, affecting their ability to access financial services and potentially leading to financial exclusion. By focusing on the false positive rate, this metric ensures that the model does not unfairly predict loan default for one gender more than another. It helps in avoiding scenarios where individuals are incorrectly judged as risky and thus unfairly denied credit. Predictive Equality has a lower value for females compared to males, indicating that the model is less likely to falsely predict a positive outcome for females. This could mean the model is more cautious or accurate with females regarding false alarms.

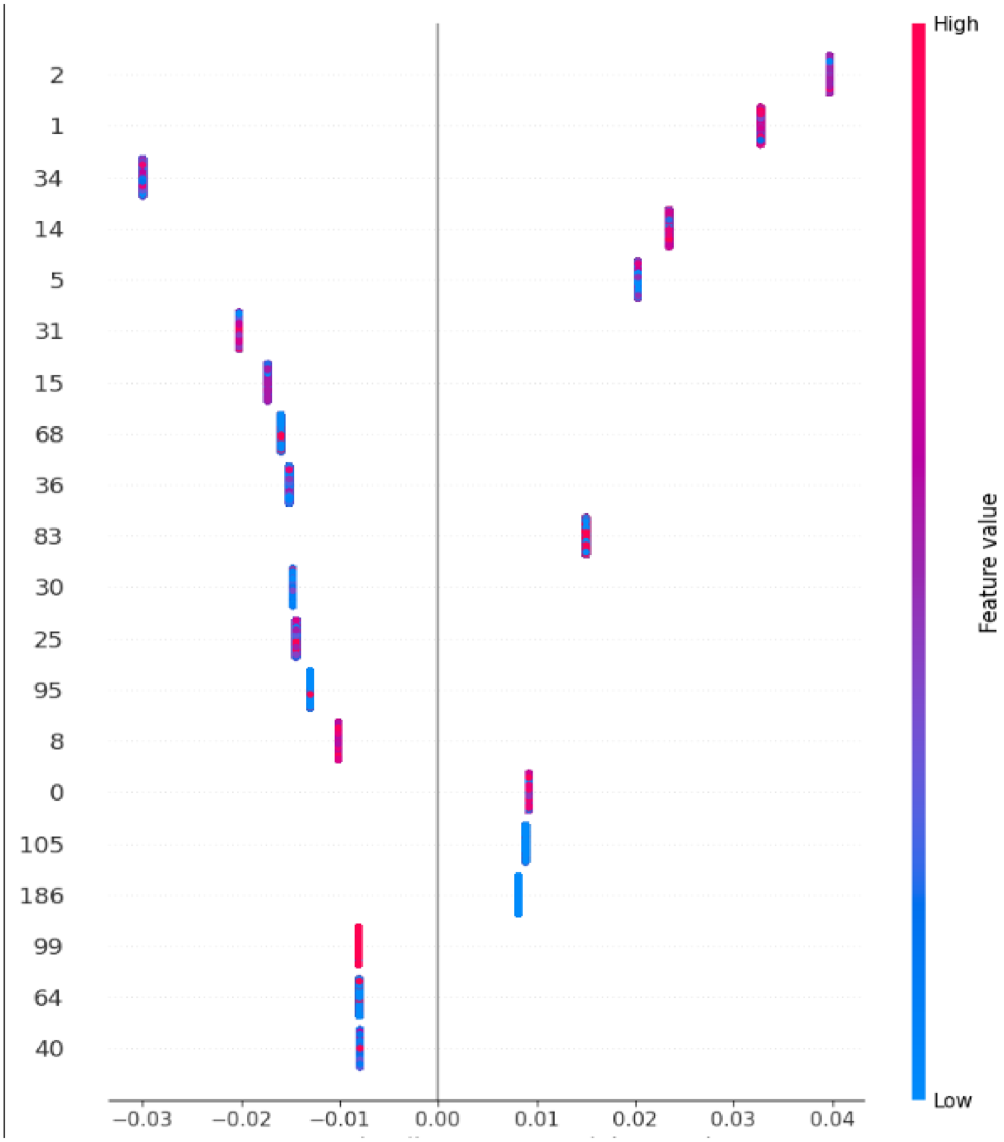
Analyze the model stability:



Graph 1: Model Variability with bootstrap samples

We used Bootstrap to test the variability of the model. By bootstrapping a number of samples and using the model to make predictions based on each sample, we successfully plotted the graph as above. Specifically, the blue-shaded area represents the mean prediction plus/minus the standard deviation across all bootstrap samples and the red-shaded area shows the confidence intervals for each sample. By drawing sample with replacement repeatedly we can have an insight of how the model works for different sample – namely different datasets constructed by randomly drawing samples. Looking at the graph of variability of predictions, we would have a sense of how variable is the XGB model when dealing with different datasets.

Analyze feature importance based on the model:

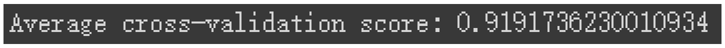


Graph 2: Feature importance by SHAP

Similar to what has been done in the lab, we developed SHAP to analyze the feature importance when applying the XGB classifier to the dataset. By doing so, we can have a sense of what features are important in the prediction with this model and what are not. The plot shown above only contains some of the features among all relative features – this is because though the other features may have contribution to the prediction, they may affect the result to a little extent. In other words, their absolute Shapley value is very close to 0, so the plot exclude those features.

Analyze the model robustness:

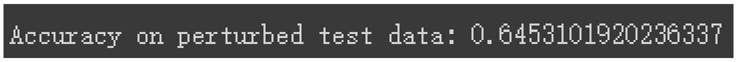
Cross-Validation:



```
Average cross-validation score: 0.9191736230010934
```

We used Cross Validation to test the robustness of the model and found average cross-validation score is around 0.919. The cross-validation process trains the model on different number folds(cv-1) as our choice and tests it on the 1 remaining fold. This process is repeated several times(value of cv), with each of the folds used exactly once as the test set. The reason why we use cv=5 is that: 1. This value can reduced bias: By using most of the data for training (80%) and a portion for testing (20%), the model has a good chance of learning robust patterns and not just memorizing the data; 2. Reduced Variance: Repeating the process across 5 different subsets helps in reducing the variance of the model estimate since the model is validated multiple times against different data sets.

Data Perturbation:



```
Accuracy on perturbed test data: 0.6453101920236337
```

We also used Data Perturbation to further test the robustness of the model. This method tests the performance of the model with perturbed data. If the model can still achieve a high accuracy after data perturbation, it means the model is quite robust and has a low sensibility. As the choice of noise (= np.random.normal(np.mean(mean_predictions), 1.5, X_test.shape)), we found the accuracy of Accuracy on perturbed test data: 0.645. The reason why we use the mean of mean predictions(as mentioned in the bootstrap) of one parameter is due to CLT – it ensures that the mean of bootstrap the sample means equals the true population mean. Thus it is valid to do so. And from the bootstrap, we can see the standard deviation is not that large so we choose a relatively large value of standard deviation as the second parameter to see whether the model can still be robust to such perturbed data.

5. Summary

Data Appropriateness:

Talking about fitness of the data to this ADS, we'd like to first talk about the model used in this ADS: XGBoost.

XGBoost classifier is designed to be highly efficient and scalable, which makes it faster than many other implementations of gradient boosting. It utilizes both multi-threading and distributed computing to train models quickly, making it suitable for large datasets. In addition, XGBoost is equipped to handle sparse data (data with many zeros) originating from various sources like missing values or encoding of categorical variables. Its handling of sparse data allows for efficient use of memory and improves computation speed.

Based on those advantages, let's look back to our data: The Home Credit dataset contains tons of data, even after data cleaning and preprocessing. There are so many features and applicants in the dataset, with a lot of missing values. This ADS, equipped with XGBooster, can efficiently deal with this large amount of data and be able to take care of the missing values. So we think the data was appropriate for this ADS.

Stability and Robustness:

According to the Bootstrap we've done, we believe that the implementation is stable. As we can see from the plotted graph, though small variation in the interval(mean plus/minus standard deviation) can be seen, generally, the shape of the intervals are smooth, which indicates a good stability – there are no such outlying intervals. For robustness, no matter testing the model by cross-validation or the data perturbation, we can conclude that the implementation is relatively robust. The model gets a cross-validation score of 0.919, meaning that it performs well under the application of those folds. What's better, the model remains an accuracy score of 0.645 even with a large value of standard deviation chosen for the noise. We believe that this score is high enough for the model performance with an input of large noise.

Accuracy and fairness:

The model shows high overall accuracy, which typically indicates that it performs well in identifying clients who will repay loans. However, the consistently low recall across all groups reveals that the model fails to identify a significant portion of clients who will face payment difficulties. This could result in those who might default on a loan being approved, potentially leading to financial losses. This could lead to adverse financial and social consequences for individuals and institutions alike. To improve, the model needs enhancements in its ability to detect true positives without sacrificing fairness across sensitive attributes. Enhancements could include revising the training data, feature selection, or model algorithm to better capture the complexities and variabilities in loan repayment behaviors across different demographic groups.

Comfortness of deploying this ADS in public sectors with Stakeholder Considerations:

We would like to deploy this ADS in the public sector. For public sectors, the most desired thing of an ADS is its accuracy. This ADS is well performed in accuracy and is expected to be stable and robust with the input of some noise and the potential interruption of other factors. In addition, the ADS ensures fairness among gender groups, which provides public sectors with confidence to deploy this ADS to make fair decisions. Let’s be more specific by talking about the stakeholders.

The primary user of this model will be the Financial institutions. They are interested in both high accuracy to minimize risk and fairness to ensure compliance with regulations and maintain their reputation. They also require a balance between not granting loans to likely defaulters (precision) and not missing out on clients who would repay (recall). Therefore, the model likely needs to improve its recall rate in order to attract Financial institutions as users.

Loan applicants are directly affected by the model's decisions. Fair treatment across genders and equitable opportunities for loan approval are crucial for them. High predictive equality ensures that no subgroup is unfairly judged based on biased or inaccurate assessments.

For regulators, they would be concerned with both the fairness metrics and the model's ability to serve all sections of society equitably. They would endorse measures like EO and DP to ensure that discriminatory biases do not exist in automated decision-making processes.

In conclusion, the model shows great accuracy while it fails to flag individuals who will face repayment difficulties due to the extreme imbalance of the dataset. This could lead to scenarios where risky loans are approved, and cause a higher default rate. We suggest that model needs to be tuned to handle such imbalance by adjusting class weight or resampling the data.

Data Collection, Processing & Analysis:

Though we have mentioned that the XGBooster can handle the NULL values properly, it would be better if in the process of data collection, more data can be collected and less NULL values remain in the dataset. By doing so, we believe that the model's performance would be better and the accuracy may have the potential to be higher.

For data processing, like we have mentioned above, it would be better to find an alternative way to compute the NULL values – in order to get more accurate predictions. In addition, if it is able to do so, we should further assess the precision and correctness of the data used by the system and do consistency checks – ensure that the data does not contain contradictory entries and follows the same formats and standards throughout.

Based on how the authors chose and trained the model, we believe that the author did a great job on the analysis.They examined the performance of many different models and by comparing all those models, they finally picked XGBooster. Also, they used the ROC AUC curve to validate the implementation, which is a nice way to measure the model's ability to distinguish between classes and check the model performance.

Partner Contribution:

We browsed on the Kaggle and picked our ADS together. We did the first 3 parts together as well – we firstly added our own perspectives to each section and after some discussion, we revised some parts and combined them all together. For part4, the analysis of accuracy and fairness is mainly done by Patric and the testing of stability, feature importance and robustness is primarily done by me. After we finished our code and posted the result on the report, we discussed the plots and results, and wrote the analysis for our own part. Then we talked about the remaining questions in part5 and finished that part together. Glad to work with Patric and so proud to have this beautiful project :)