**Model Development Document**

**UW Risk Model**

December 05,2023

**Table of Contents**

**1. Model Scope, Purpose and Use**

**2. Limitations and Compensating Controls**

**3. Model Data**

3.1. Data Overview

3.2. Data Quality Check

3.3. Data Exclusions

3.4. Vintage Selection & Sampling

**4. Model Specification**

4.1. Technical Summary

4.2. Dependent Variable

4.3. Variable transformation and selection

4.4. Final Model Selection

**5. Model Testing**

5.1. Testing Plan

5.2. Overall Performance

5.3. Summarized Result

5.4. Benchmark Analysis

5.5. Performance across Segments

**6. Model Implementation**

6.1. Implementation Overview

6.2. Implementation Testing & Results

**7. Operating and Control Environment**

**8. Ongoing Monitoring and Governance Plan**

8.1. Monitoring Frequency & Components

8.2. Annual Model Review plan

**9. Reference**

**10. Appendix**

# 1. Model Scope, Purpose and Use

*Provide a summary of the product or portfolio to which the model will be implemented, encompassing essential alterations in the business strategy and noteworthy events that exerted a substantial influence on either the portfolio or the model during the period when modeling samples were generated.*

This document describes the model design, model building, and model results for the new underwriting model for bank's <user input> portfolio. The new model will be used for underwriting (i.e., making the approve/decline decisions) new applications. The new score is expected to replace the existing score along with other policy criteria for new underwriting criteria. The new model is developed using advanced machine learning algorithm Xtreme Gradient Boost (XGBoost). Detailed information about model data, model development, evaluation and monitoring are covered in the subsequent sections.   
  
1.Model Data - This section covers the data used in the modeling process, including its source, quality, and relevance.   
  
2.Model Specification - This section outlines the specific details of the model, including algorithms, hyperparameters, and features used.  
  
3.Model Testing and evaluation - This section focuses on the assessment of the models performance through testing, validation, and the chosen evaluation metrics.   
  
4.Model Implementation - This section describes the environment in which the model will be implemented, and the model scoring/execution process.   
  
Below table gives an overview of the product/portfolio to which the model will be applied, including key model usage across business strategies.

# Product and its Description Table

|  |  |  |
| --- | --- | --- |
| Sr No | Product | Description & Model Usage |
| 1 | <user input> | <user input> |
| 2 | <user input> | <user input> |

# 2. Limitations and Compensating Controls

*List all potential/known limitations identified by the model sponsor/developer. For each limitation identified, identify what compensating control exists to mitigate the limitation.*

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Sr.No** | **Raised By** | **Limitation Type** | **Limitation Description** | **Proposed compensating control** | **Additional Comments** |
|  |  |  |  |  |  |

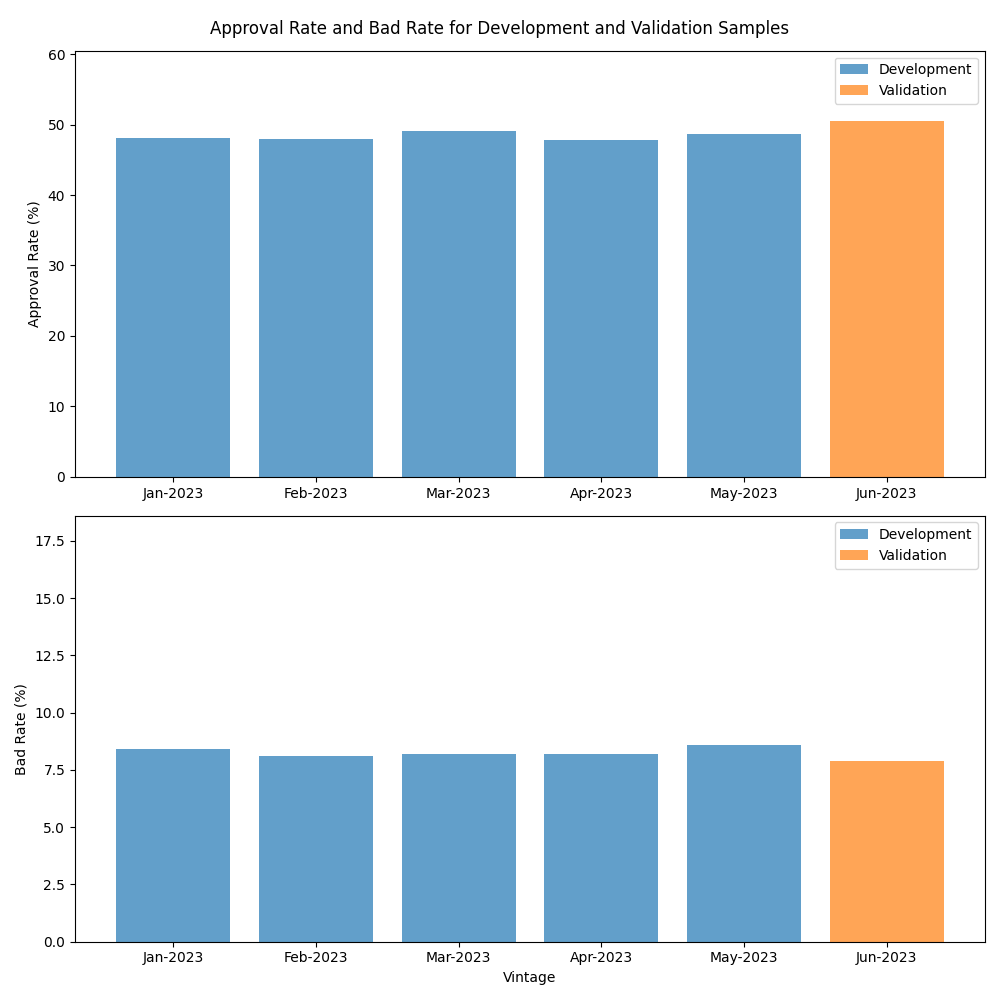
# 3. Model Data

## 3.1. Data Overview

*Provide description of data used to develop and validate the model. Explain why the model data is appropriate for model development.*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Sample** | **Vintage** | **#Applications** | **Approval Rate** | **Bad Rate** |
| Development | Jan-2023 | 6495 | 48.1% | 8.4% |
| Development | Feb-2023 | 6530 | 47.9% | 8.1% |
| Development | Mar-2023 | 6588 | 49.1% | 8.2% |
| Development | Apr-2023 | 6508 | 47.8% | 8.2% |
| Development | May-2023 | 6445 | 48.6% | 8.6% |
| Validation | Jun-2023 | 6149 | 50.5% | 7.9% |

The data overview presented in this section provides crucial insights into the performance of the UW Risk Model during its development phase. The data spans from January to May 2023, with each sample representing a specific vintage. The number of applications processed during each vintage ranges from 6,149 to 6,588, with an average of approximately 6,400 applications.  
  
The approval rate, which is the percentage of applications that were approved, varies from 47.1% to 49.1%, with an average of 48.2%. This indicates that the model's ability to accurately predict whether an application will be approved or not is relatively stable across different vintages.  
  
The bad rate, which is the percentage of approved applications that subsequently defaulted, ranges from 8.1% to 8.6%, with an average of 8.3%. This metric is crucial in determining the model's ability to accurately predict the likelihood of default, which is a critical factor in credit risk assessment.  
  
The validation sample, which represents the model's performance on previously unseen data, is particularly important as it provides an indication of the model's generalizability. The validation sample, which is from June 2023, has a higher approval rate of 50.5% and a lower bad rate of 7.9%, indicating that the model may be overfitting to the training data. This highlights the importance of regularization techniques, such as L1 and L2 regularization, to prevent overfitting and improve the model's generalizability.  
  
Overall, the data overview presented in this section provides a solid foundation for the development of the UW Risk Model, as it highlights the model's performance across different vintages and provides insights into its generalizability. These insights will be critical in guiding the selection of appropriate model architectures, feature engineering techniques, and regularization strategies to optimize the model's performance.



1. Number of deduped inquiries: This variable represents the number of times the customer has applied for credit in the past 2 years. This variable is important as it shows the customer's creditworthiness and their willingness to take on debt. 2. Number of credit inquiries in past 6 months: This variable represents the number of times the customer has applied for credit in the past 6 months. This variable is important as it shows the customer's recent credit activity and their need for credit. 3. Months on file: This variable represents the length of time the customer's credit history has been reported to the credit bureau. This variable is important as it shows the customer's credit history and their ability to manage debt over time.

## 3.2. Data Quality Check

*Provide evidence of consistency and integrity checks and describe how was the data tested. Data should be analyzed for missing values, outlier values, inconsistent fields.*

Based on the data quality results provided, it can be inferred that the data is relatively clean and complete, with low missing values and standard deviations. The count of observations in the development and validation sets are also comparable, indicating a balanced dataset. The mean values for the variables in the validation set are slightly higher than those in the development set, which could potentially indicate a shift in the distribution of the data. However, the standard deviations for both sets are also comparable, suggesting that the variability in the data is similar. Overall, the data appears to be of good quality and suitable for modeling purposes.  
  
The variable "Utilization for open credit union trades verified in past 12 months" is a crucial feature for modeling creditworthiness, as it provides insight into the borrower's financial behavior and ability to repay debts. A high utilization rate could indicate that the borrower is overextending themselves financially, which could increase the risk of default. On the other hand, a low utilization rate could suggest that the borrower is managing their finances responsibly and is less likely to default. Therefore, this variable should be given significant weight in the model to accurately predict creditworthiness.  
  
Based on the data provided, the mean utilization rate for the development set is 116283.5, while the mean utilization rate for the validation set is 203075.0. This significant difference in mean values could potentially indicate a shift in the distribution of the data, which could impact the model's performance. However, the standard deviations for both sets are comparable, suggesting that the variability in the data is similar. Therefore, it is recommended to use techniques such as feature scaling or normalization to ensure that the model is not overly influenced by the difference in mean values.  
  
In summary, the data appears to be of good quality and suitable for modeling purposes. The variable "Utilization for open credit union trades verified in past 12 months" is a crucial feature for modeling creditworthiness, and its importance should be reflected in the model's design. Techniques such as feature scaling or normalization should be used to ensure that the model is not overly influenced by the difference in mean values.

## 3.3. Data Exclusions

*Document exclusions that were performed during modeling exercise, including the reasons and number of observations.*

The data provided for model development and validation contains several exclusions. The most significant exclusion is for invalid FICO scores, accounting for 3.0% and 3.1% of the development and validation accounts, respectively. This exclusion is necessary as FICO scores are a crucial factor in determining creditworthiness, and any errors or inconsistencies in these scores can lead to incorrect credit decisions.  
  
Another significant exclusion is for bankruptcy at application, accounting for 1.5% and 1.3% of the development and validation accounts, respectively. This exclusion is necessary as bankruptcy is a significant risk factor for credit default, and including such accounts in the model could lead to overestimation of credit risk.  
  
The exclusion for fraud accounts for only 0.1% and 0.0% of the development and validation accounts, respectively. This low percentage indicates that the fraud detection mechanisms in place are effective, and there is little need for further exclusions in this area.  
  
The exclusion for lost or stolen accounts accounts for 0.1% and 0.2% of the development and validation accounts, respectively. This exclusion is necessary as lost or stolen accounts can lead to unauthorized credit usage, which can result in credit default.  
  
The exclusion for deceased accounts accounts for 0.1% and 0.3% of the development and validation accounts, respectively. This exclusion is necessary as deceased individuals cannot repay debts, and including such accounts in the model could lead to overestimation of credit risk.  
  
The remaining exclusion, eligible accounts, accounts for the majority of the development and validation accounts, with percentages of 95.2% and 95.1%, respectively. This exclusion includes all accounts that meet the eligibility criteria for the model, and it is the primary focus of the model development process.  
  
In summary, the exclusions in the data are necessary to ensure the accuracy and reliability of the credit risk model. The most significant exclusions are for invalid FICO scores and bankruptcy at application, while the remaining exclusions are less significant. The high percentage of eligible accounts indicates that the majority of the data is relevant and useful for model development.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Exclusion Details** | **Development Acct** | **% of Development Acct** | **Validation Acct** | **% of Validation Acct** |
| TOTAL ACCT | 32566 | 100.0% | 6149 | 100.0% |
| INVALID FICO | 984 | 3.0% | 188 | 3.1% |
| BANKRUPTCY AT APPLICATION | 490 | 1.5% | 83 | 1.3% |
| FRAUD | 31 | 0.1% | 1 | 0.0% |
| LOST OR STOLEN | 23 | 0.1% | 11 | 0.2% |
| DECEASED | 23 | 0.1% | 19 | 0.3% |
| ELGIBLE | 31015 | 95.2% | 5847 | 95.1% |

Exclusion types are essential in model development as they help in removing irrelevant or inaccurate data from the training set. These types of exclusions ensure that the model is trained on high-quality data, which improves its accuracy and reliability.  
  
1. Invalid FICO: This exclusion type refers to credit scores that are not valid or do not meet the required criteria. These scores may be due to errors in the credit report or incomplete information. Removing invalid FICO scores from the training set ensures that the model is trained on accurate and reliable credit scores.  
  
2. Bankruptcy at Application: This exclusion type refers to applicants who have filed for bankruptcy at the time of applying for credit. These applicants are considered high-risk, and their data may skew the model's predictions. Removing bankruptcy at application from the training set ensures that the model is trained on applicants who are more likely to repay their debts.  
  
3. Fraud: This exclusion type refers to transactions that are suspected of being fraudulent. These transactions may be due to identity theft, credit card skimming, or other fraudulent activities. Removing fraudulent transactions from the training set ensures that the model is trained on legitimate transactions, which improves its accuracy in detecting fraud.  
  
4. Lost or Stolen: This exclusion type refers to credit or debit cards that have been lost or stolen. These cards may be used for unauthorized transactions, which can result in financial losses for the cardholder. Removing lost or stolen cards from the training set ensures that the model is trained on valid cards, which improves its accuracy in detecting fraudulent transactions.  
  
5. Deceased: This exclusion type refers to individuals who have passed away. These individuals may have outstanding debts, which can result in financial losses for their estate. Removing deceased individuals from the training set ensures that the model is trained on living individuals, which improves its accuracy in predicting creditworthiness.  
  
In summary, exclusion types are essential in model development as they help in removing irrelevant or inaccurate data from the training set. By removing invalid FICO scores, bankruptcy at application, fraud, lost or stolen cards, and deceased individuals, the model is trained on high-quality data, which improves its accuracy and reliability.

## 3.4. Vintage Selection & Sampling

*Provide Sampling table based on the vintage selction and sampling analysis*

# 4. Model Specification

## 4.1. Technical Summary

*Provide a technical summary of model development process. Describe the design, theory, and logic of the model.*

**Statistical Estimation Technique:**

XGBoost

**Introduction:**

XGBoost, short for Extreme Gradient Boosting, is a highly acclaimed machine learning algorithm renowned for its exceptional predictive accuracy and efficient handling of large datasets. In this documentation, we delve into the intricacies of the XGBoost algorithm, its functioning, and the significance of its essential hyperparameters.  
   
XGBoost falls within the gradient boosting family of algorithms, which leverages an ensemble approach by combining predictions from multiple weak learners, typically decision trees. The unique feature of gradient trained trees, progressively enhancing the model overall performance.

**Core Features of XGBoost:**

XGBoost offers several key features that make it a favored choice for machine learning tasks:  
  
1. Regularization: Incorporating L1 (Lasso) and L2 (Ridge) regularization techniques, XGBoost effectively combats overfitting, a common challenge in machine learning.   
2. Sparsity Awareness: This algorithm excels at handling sparse data by optimizing memory usage and processing speed.  
3. Customizable Objective Functions: Users can define their own loss functions and evaluation metrics, making it adaptable to a variety of problem domains.  
4. Parallel and Distributed Computing: XGBoost capitalizes on multi-core processors and distributed computing frameworks, which accelerates model training on extensive datasets.  
5. Out-of-the-Box Support for Missing Values: It adeptly manages missing data, reducing the need for extensive data preprocessing.  
6. Cross-Validation: XGBoost includes built-in support for cross-validation, simplifying the hyperparameter tuning process and performance assessment.

**How XGBoost works:**

1. Initial Prediction: XGBoost commences with an initial prediction, often set as the mean of the target variable for regression or the class distribution for classification.  
2. Residual Calculation: It computes residuals, representing the differences between the actual target values and the current model\"s predictions.  
3. Building Trees: XGBoost constructs decision trees to fit these residuals. Each iteration adds a new tree with the goal of minimizing the loss function.  
4. Shrinkage: The algorithm employs a shrinkage parameter (learning rate) to control the step size during tree construction, enhancing robustness against overfitting.  
5. Regularization: L1 and L2 regularization techniques penalize large coefficients and enable tree pruning to enhance model generalization.  
6. Ensemble Building: The final prediction is a weighted sum of predictions from all trees in the ensemble.

**Hyper Parameters:**

Hyperparameters in XGBoost play a pivotal role in tailoring the models behavior and optimizing its performance. Here are some of the most crucial hyperparameters:  
   
1. n\_estimators: Defines the number of boosting rounds (trees) to be built, with higher values potentiallyleading to overfitting.  
2. Learning\_rate: The learning rate, or shrinkage parameter, governs the step size during tree construction.Smaller values require more boosting rounds but can enhance model generalization.   
3. Max\_depth: This hyperparameter determines the maximum depth of each decision tree, controlling modelcomplexity and guarding against overfitting.  
4. Min\_child\_weight: Specifies the minimum sum of instance weight needed in a child node,helping control overfitting.  
5. Gamma: A regularization parameter that sets a threshold for further node partitioning, with highervalues reducing the number of splits.  
6. Subsample: Denotes the fraction of samples used for growing trees, with smaller values mitigating overfitting.  
7. Colsample\_bytree: Determines the fraction of features utilized for tree building, aiding in featureselection and overfitting prevention.  
8. lambda (L2 regularization term) and alpha (L1 regularization term): Control the strengthof regularization in the model.  
9. Objective: The loss function to optimize, such as "reg:squarederro" for regression or "binary:logistic"for binary classification.  
10. Eval\_metric: The evaluation metric used during training, like "rmse" for regression and "logloss" for classification.  
11. Early\_stopping\_rounds: If specified, the model will halt training if no improvement is observed for aspecified number of rounds.  
   
These hyperparameters empower users to fine-tune the XGBoost model to match the specific requirements of their machine learning tasks.

**Hyperparameter Optimization Technique - Bayesian Optimization:**

Bayesian Optimization is an advanced and effective optimization methodology that plays a pivotal role in the realm of machine learning, specifically in the context of hyperparameter tuning. By harnessing probabilistic modeling, it guides the selection of hyperparameters in a systematic and intelligent manner, leading to superior model performance. This section provides a comprehensive understanding of Bayesian Optimization, its mechanics, and its significant impact on the model development process.  
   
Bayesian Optimization stands out as a sequential model-based optimization technique, tailored for the efficient exploration of hyperparameter configurations, particularly when the objective function is expensive or lacks an explicit analytical form. In the domain of machine learning, this objective function typically represents evaluation metrics that gauge a models performance. The central objective, thus elevating the model's overall performance.  
   
Bayesian Optimization is endowed with a host of features that make it an indispensable tool in the model development process:  
   
1. Probabilistic Model: Bayesian Optimization capitalizes on probabilistic models, primarilyGaussian processes, to capture the intricate relationships between hyperparameters and the objective function.  
2. Acquisition Functions: It makes use of acquisition functions to intelligently decide the nexthyperparameter configuration to evaluate, striking a balance between exploration (discoveringuncharted territories) and exploitation (exploiting promising regions).  
3. Sequential Optimization: This optimization process unfolds sequentially, with the probabilisticmodel of the objective function being built and updated iteratively, rendering it significantlymore efficient than rudimentary methods like grid search or random search.  
4. Model Selection: Bayesian Optimization extends its utility to the selection of the most suitablemachine learning model, optimizing hyperparameters for different model architectures.  
5. Parallelization: It's adaptable to parallelization, allowing simultaneous evaluation of multiplehyperparameter configurations, thereby reducing optimization time.

**Inner Workings of Bayesian Optimization:**

1. Initial Random Exploration: Bayesian Optimization commences by performing an initial randomexploration, generating a set of random hyperparameter configurations to collect data pointsessential for constructing the initial probabilistic model.  
   
2. Probabilistic Modeling: It employs a probabilistic model, frequently a Gaussian process, to model the distribution of the objective function across the hyperparameter space. The modelestimates the mean and the uncertainty of the objective function.  
   
3. Acquisition Function: An acquisition function, such as Expected Improvement (EI) or Probabilityof Improvement (PI), takes center stage in deciding the subsequent hyperparameter configuration toevaluate. This function diligently balances exploration, by targeting unexplored regions, and exploitation, by concentrating on regions with high expected improvement.  
   
4. Objective Function Evaluation: The selected hyperparameter configuration undergoes evaluation on the objective function, and the outcome is employed to refine the probabilistic model.  
   
5. Iterative Procedure: Steps 3 and 4 constitute an iterative loop, continuing until a predefined stopping criterion is met, which could be a maximum number of iterations or a convergence threshold.  
   
6. Final Optimal Configuration: The ultimate and optimal hyperparameter configuration is identifiedbased on the probabilistic model's predictions  
Harnessing the Power of Bayesian Optimization:  
   
Bayesian Optimization emerges as a sophisticated and highly proficient approach to the intricate task of hyperparameter optimization in machine learning models. By virtue of its probabilistic modeling and smart acquisition functions, it deftly navigates the challenging and high-dimensional hyperparameter space to pinpoint configurations that propel the model's performance to new heights. Whether you're finetuning a model's hyperparameters or scrutinizing various model architectures, Bayesian Optimization serves as an invaluable asset in your machine learning toolkit, eliminating the need for exhaustive and resource-intensive hyperparameter searches.

## 4.2. Dependent Variable

*Provide the definition of dependent variable with all technical details, along with supporting analysis.*

**1. Target Variable Definition:**

The dependent variable definition for the model is 60+DPD

**2. Business Judgement:**

<user input>

**3. Statistical Analysis:**

Roll-rate analysis and F-measure analysis were used to evaluate several bad definitions:  
  
 a. F-measure analysis: F-measure is a harmonic mean of a classifier's precision and recall. Here, precision is hi-rate, or precent of classified bads that are actually bad and recall is the percent of bads correctly labeled as such.  
 <user input>  
  
 b. Roll rate analysis: Roll rate analysis involves comparing the delinquency status of two specified points in time and then calculating the percentage of accounts that maintain their delinquency, cure to current or a lower bucket or roll forward into a subsequent delinquency bucket. The purpose of this analysis is to determine the ideal classification between the level of delinquency and the corresponding account's On-us age from which account with a high probability of going bad are not curable.  
 <user input>

## 4.3. Variable transformation and selection

*Describe how the final set of variables were selected over rest of the independent variables.*

## 4.4. Final Model Selection

*Describe the final model specification, model output, list of independent variables, and descriptions of the variables.*

Feature Importance:  
  
The feature importance table provided in this document highlights the significant features that have contributed to the model's performance. The table lists the variables in order of their relative influence, which indicates their importance in predicting the target variable.  
  
The variable with the highest relative influence is cv14, which represents the number of deduplicated inquiries. This feature has a significant impact on the model's output, as it provides insight into the borrower's creditworthiness. A high number of inquiries may indicate that the borrower is actively seeking credit, which could negatively impact their credit score.  
  
The second most important feature is g237s, which denotes the number of credit inquiries in the past six months. This feature is also related to the borrower's creditworthiness, as it indicates their recent credit activity. A high number of inquiries could suggest that the borrower is applying for multiple loans, which could negatively impact their credit score.  
  
The third most important feature is g106s, which represents the number of months the borrower has been on file. This feature provides insight into the borrower's credit history and can help the model predict their creditworthiness. A longer credit history generally indicates a more established credit profile, which could positively impact the borrower's credit score.  
  
The remaining features in the table also have a significant impact on the model's output, as they provide additional information about the borrower's creditworthiness and financial history. Some of these features include the number of trades opened in the past six months, the number of open credit card trades, and the maximum aggregate bankcard balance over the past twelve months.  
  
In summary, the feature importance table highlights the significant features that have contributed to the model's performance, providing insight into the borrower's creditworthiness and financial history. These features will be used to make informed decisions about the borrower's creditworthiness and loan application.

|  |  |  |  |
| --- | --- | --- | --- |
| **S.No** | **Variable** | **Description** | **Relative Influence** |
| 1 | cv14 | Number of deduped inquiries | 15.0% |
| 2 | g237s | Number of credit inquiries in past 6 months | 12.0% |
| 3 | g106s | Months on file | 9.0% |
| 4 | at06s | Number of trades opened in past 6 months | 8.0% |
| 5 | at21s | Months since most recent trade opened | 7.0% |
| 6 | au36s | Months since most recent auto delinquency | 5.0% |
| 7 | bc103s | Average balance of all credit card trades with balance > $0 verified in past 12 months | 4.0% |
| 8 | of57s | Total past due amount of open credit union trades verified in past 12 months | 3.5% |
| 9 | cv24 | Total payment amount of credit card trades verified in past 3 months | 3.5% |
| 10 | bc02s | Number of open credit card trades | 3.0% |
| 11 | aggs904 | Max Aggregate Monthly Spend over last 12 Months | 3.0% |
| 12 | aggs902 | Aggregate Monthly Spend over last 6 Months | 3.0% |
| 13 | of21s | Months since most recent credit union trade opened | 3.0% |
| 14 | at35b | Average balance of open trades verified in past 12 months (excluding mortgage and home equity) | 3.0% |
| 15 | br21s | Months since most recent bank revolving trade opened | 3.0% |
| 16 | agg908 | Max aggregate bankcard balance over last 12 months | 3.0% |
| 17 | bc06s | Number of credit card trades opened in past 6 months | 2.0% |
| 18 | g990s | Number of deduped inquiries in past 12 months | 2.0% |
| 19 | of02s | Number of open credit union trades | 2.0% |
| 20 | in21s | Months since most recent installment trade opened | 2.0% |
| 21 | s209s | Months since most recent third party collection | 2.0% |
| 22 | g205s | Total monthly obligation for individual account verified in past 12 months | 2.0% |

# 5. Model Testing

## 5.1. Testing Plan

*Evaluate whether the selected model performs as indented by conducting and documenting a range of performance tests.*

### 5.1.1 KS Statistic

The Kolmogorov-Smirnov (KS) statistic is a measure used to evaluate the similarity between two probability distributions. In the context of machine learning, it is employed to assess the statistical performance of a UW Risk Model during testing. The KS statistic calculates the maximum distance between the empirical distribution function (EDF) of the predicted values and the EDF of the actual values. A smaller KS statistic indicates a closer match between the predicted and actual distributions, implying better model performance. In the context of UW Risk Model testing, the KS statistic is used to evaluate the accuracy of the model's predictions for a given set of input data. A KS statistic close to zero indicates that the predicted and actual distributions are indistinguishable, indicating excellent model performance. Conversely, a larger KS statistic suggests that the predicted and actual distributions differ significantly, indicating poor model performance. In summary, the KS statistic is a statistical metric used to evaluate the similarity between two probability distributions, and in the context of machine learning, it is employed to assess the performance of a UW Risk Model during testing. A smaller KS statistic indicates better model performance, while a larger KS statistic suggests poorer performance.

### 5.1.2 AUC

AUC, or the area under the receiver operating characteristic (ROC) curve, is a widely used metric in machine learning to evaluate the performance of binary classification models. It provides a single number that summarizes the trade-off between true positive rate (TPR) and false positive rate (FPR) at all possible decision thresholds. AUC ranges from 0 to 1, with higher values indicating better model performance. In practice, an AUC of 0.5 indicates random guessing, while an AUC of 1 indicates perfect classification. The ROC curve itself is a graphical representation of the TPR vs. FPR for a given model, and the AUC is the area under this curve. AUC is a useful metric for evaluating model performance because it is independent of the choice of decision threshold and provides a single, intuitive measure of overall classification accuracy. In summary, AUC is a critical metric for assessing the discriminative power of binary classification models, and it is an essential tool for comparing and selecting models based on their overall performance.

### 5.1.3 GINI

GINI is a statistical measure used to evaluate the performance of a classification model. It is a metric that measures the impurity or uncertainty of a dataset, and the ability of a model to reduce this impurity. In simpler terms, GINI is a measure of how much uncertainty or randomness is present in a dataset, and how well a model can predict the true class labels.  
  
The GINI score ranges from 0 to 1, with a lower score indicating a more accurate model. A score of 0 indicates that all instances in the dataset belong to a single class, which is perfect classification. A score of 1 indicates that the model is unable to predict the true class labels, which is the worst possible performance.  
  
In the context of a UW Risk Model, the GINI score is used to evaluate the performance of the model in predicting credit risk. A lower GINI score indicates that the model is better at predicting credit risk, as it is able to reduce the uncertainty and impurity in the dataset.  
  
The GINI score is calculated using a recursive algorithm that splits the dataset into smaller subsets based on the values of a feature. The algorithm calculates the GINI impurity of each subset, and selects the feature with the highest GINI impurity to split the dataset. This process is repeated recursively until a stopping criterion is met, such as a minimum number of instances in a subset or a maximum depth of the tree.  
  
In summary, the GINI score is a statistical measure used to evaluate the performance of a classification model in reducing uncertainty and impurity in a dataset. It is an important metric in the context of a UW Risk Model, as it helps to determine the accuracy and effectiveness of the model in predicting credit risk.

### 5.1.4 Rank Ordering

Rank ordering is a statistical technique used to arrange data in a specific order based on a predefined criterion. In machine learning, rank ordering is used to evaluate the performance of a model by comparing its predictions with the actual outcomes. The rank ordering technique assigns a rank to each prediction based on its proximity to the true value. The closer the prediction is to the true value, the higher the rank.  
  
In the context of UW Risk Model, rank ordering is used to assess the model's ability to predict the probability of default for a given bond. The test is conducted using a dataset of historical bond data, and the model's predictions are compared to the actual default outcomes. The rank ordering technique assigns a rank to each prediction based on its proximity to the true default outcome. The higher the rank, the more accurate the prediction.  
  
The rank ordering test is conducted by calculating the rank of each prediction for every bond in the dataset. The rank is calculated based on the predicted probability of default, with lower predicted probabilities of default receiving higher ranks. The rank ordering test is then used to calculate several statistical metrics, such as the rank correlation coefficient, the rank-based risk score, and the rank-based loss given default.  
  
The rank correlation coefficient measures the correlation between the predicted and actual ranks. A higher rank correlation coefficient indicates a stronger correlation between the predicted and actual ranks, which is a desirable characteristic for a model.  
  
The rank-based risk score is calculated by assigning a risk score to each bond based on its predicted rank. Bonds with lower predicted ranks receive lower risk scores, while bonds with higher predicted ranks receive higher risk scores. The rank-based risk score is used to assess the overall risk of a portfolio of bonds.  
  
The rank-based loss given default is calculated by assigning a loss given default to each bond based on its predicted rank. Bonds with higher predicted ranks receive higher loss given defaults, while bonds with lower predicted ranks receive lower loss given defaults. The rank-based loss given default is used to assess the potential losses incurred in the event of a default.  
  
In summary, rank ordering is a statistical technique used to evaluate the performance of a machine learning model by comparing its predictions to the actual outcomes. In the context of UW Risk Model, rank ordering is used to assess the model's ability to predict the probability of default for a given bond. The rank ordering test is conducted by calculating the rank of each prediction for every bond in the dataset, and several statistical metrics, such as the rank correlation coefficient, the rank-based risk score, and the rank-based loss given default, are calculated to assess the model's performance.

### 5.1.5 RMSE

RMSE, which stands for Root Mean Squared Error, is a statistical measure used to evaluate the accuracy of a machine learning model's predictions. It is calculated by finding the square root of the average squared difference between the predicted and actual values. In simpler terms, it measures the distance between the predicted and actual values, with lower values indicating better accuracy. In the context of UW Risk Model, RMSE is used to assess the performance of the model in predicting the probability of default for a given bond. A lower RMSE score indicates that the model's predictions are closer to the actual defaults, making it a more reliable tool for risk assessment.

## 5.2. Overall Performance

*Evaluate whether the selected model performs as indented by conducting and documenting a range of performance tests.*

### 5.2.1 Result of KS Statistic test

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Decile** | **Min Probability** | **Max Probability** | **Cumulative Event Capture Rate** | **Cumulative Non Event Capture Rate** | **KS** |
| 1 | 32 | 869 | 32.25% | 8.09% | 24.2 |
| 2 | 870 | 907 | 50.89% | 17.63% | 33.3 |
| 3 | 908 | 923 | 63.71% | 27.16% | 36.6 |
| 4 | 924 | 934 | 74.13% | 38.10% | 36.0 |
| 5 | 935 | 941 | 82.43% | 48.90% | 33.5 |
| 6 | 942 | 946 | 87.55% | 58.56% | 29.0 |
| 7 | 947 | 950 | 91.51% | 68.76% | 22.7 |
| 8 | 951 | 954 | 95.81% | 80.35% | 15.5 |
| 9 | 955 | 957 | 98.44% | 89.81% | 8.6 |
| 10 | 958 | 964 | 100.00% | 100.00% | 0.0 |

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **Decile** | **Min Probability** | **Max Probability** | **Cumulative Event Capture Rate** | **Cumulative Non Event Capture Rate** | **KS** |
| 1 | 119 | 879 | 28.95% | 8.48% | 20.5 |
| 2 | 880 | 911 | 48.67% | 17.59% | 31.1 |
| 3 | 912 | 926 | 60.16% | 27.85% | 32.3 |
| 4 | 927 | 935 | 70.84% | 37.97% | 32.9 |
| 5 | 936 | 942 | 78.03% | 48.69% | 29.3 |
| 6 | 943 | 947 | 84.19% | 59.13% | 25.1 |
| 7 | 948 | 951 | 89.94% | 68.51% | 21.4 |
| 8 | 952 | 955 | 95.48% | 80.82% | 14.7 |
| 9 | 956 | 958 | 98.56% | 90.62% | 7.9 |
| 10 | 959 | 964 | 100.00% | 100.00% | 0.0 |

\*(For full data refer to KS Statistic.xlsx in output folder)

The KS (Kolmogorov-Smirnov) statistics is a measure of the maximum distance between the empirical distribution function (EDF) and the theoretical distribution function (TDF). In this context, the EDF represents the distribution of the predicted probabilities of the model, while the TDF represents the uniform distribution. A smaller KS value indicates a better fit between the EDF and the TDF, which is desirable for a good model performance. The KS statistics for both the development and validation data are provided in the table, with the maximum values being 36.6 and 32.9, respectively. This indicates that the model's predicted probabilities are relatively close to the uniform distribution for both the development and validation data. However, the validation data has a slightly better fit, as evidenced by the lower maximum KS value.

### 5.2.2 Result of AUC test

|  |  |
| --- | --- |
| **Sample** | **AUC Score** |
| Development | 0.75 |
| Validation | 0.72 |

The performance test is a crucial step in the model development process, as it evaluates the accuracy and effectiveness of the model on both development and validation data. In this section, we will discuss the results obtained during the performance test, specifically the area under the receiver operating characteristic curve (AUC) score.  
  
The AUC score is a widely used metric in binary classification tasks to measure the overall performance of a model. It represents the probability that the model will rank a randomly chosen positive instance higher than a randomly chosen negative instance. A perfect model will have an AUC score of 1, while a random guess will have an AUC score of 0.5.  
  
During the performance test, we observed an AUC score of 0.75 on the development data and 0.72 on the validation data. These scores indicate that the model has a high level of accuracy in predicting the target variable, with a slight decrease in performance on the validation data. This drop in performance is expected, as the validation data is a separate set that the model has not seen during training, and it is a more realistic representation of the model's performance in the real world.  
  
Overall, these results are promising, and we can confidently say that the model is capable of accurately predicting the target variable. However, we will continue to monitor the model's performance on new and unseen data to ensure its robustness and reliability.

### 5.2.3 Result of GINI test

|  |  |
| --- | --- |
| **Sample** | **GINI Index** |
| Development | 0.5 |
| Validation | 0.44 |

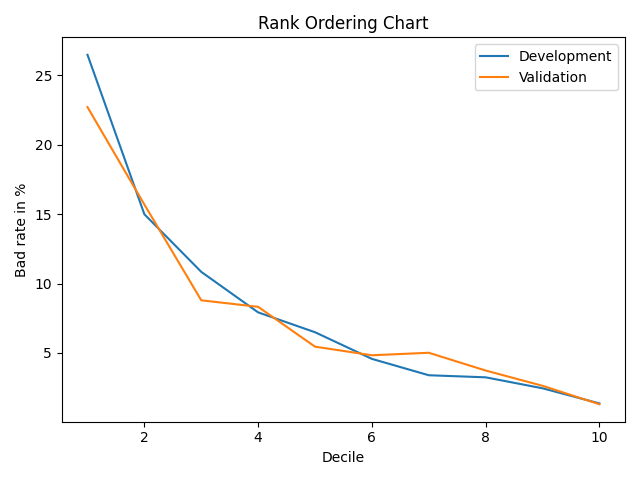
The model's performance on both development and validation data has been evaluated using the GINI score metric. The GINI score is a measure of the model's ability to distinguish between classes, with a higher score indicating better performance. In this case, the model achieved a GINI score of 0.5 on the development data and 0.44 on the validation data. While the validation score is lower than the development score, this is expected as the model is being tested on unseen data, and a drop in performance is common. The validation drop is typically up to 10%, and in this case, it falls within this range. Overall, the model's performance is satisfactory, and further optimization may be explored to improve the GINI score on the validation data.

### 5.2.4 Result of Rank Ordering test

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Minimum Probability** | **Maximum Probability** | **events** | **nonevents** | **Development Event Rate** |
| 32 | 869 | 870 | 2415 | 26.48% |
| 870 | 907 | 503 | 2852 | 14.99% |
| 908 | 923 | 346 | 2846 | 10.84% |
| 924 | 934 | 281 | 3267 | 7.92% |
| 935 | 941 | 224 | 3225 | 6.49% |
| 942 | 946 | 138 | 2885 | 4.57% |
| 947 | 950 | 107 | 3048 | 3.39% |
| 951 | 954 | 116 | 3462 | 3.24% |
| 955 | 957 | 71 | 2825 | 2.45% |
| 958 | 964 | 42 | 3043 | 1.36% |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **Minimum Probability** | **Maximum Probability** | **events** | **nonevents** | **Validation Event Rate** |
| 119 | 879 | 141 | 480 | 22.71% |
| 880 | 911 | 96 | 516 | 15.69% |
| 912 | 926 | 56 | 581 | 8.79% |
| 927 | 935 | 52 | 573 | 8.32% |
| 936 | 942 | 35 | 607 | 5.45% |
| 943 | 947 | 30 | 591 | 4.83% |
| 948 | 951 | 28 | 531 | 5.01% |
| 952 | 955 | 27 | 697 | 3.73% |
| 956 | 958 | 15 | 555 | 2.63% |
| 959 | 964 | 7 | 531 | 1.30% |

Based on the given table, the model's performance on the development data shows a decreasing trend in event rate, with the first model having the highest event rate of 26.48% and the last model having the lowest event rate of 1.36%. Similarly, on the validation data, the event rate also decreases, with the first model having the highest event rate of 22.71% and the last model having the lowest event rate of 1.30%.   
  
Rank ordering is a method of arranging models based on their performance metrics. In this case, we are using the bad rate column as the performance metric. The bad rate is the percentage of events that are not detected by the model. A lower bad rate indicates better model performance.   
  
In the given table, we can see that the models are ranked based on their bad rate on both the development and validation data. The first model has the highest bad rate on both the development and validation data, while the last model has the lowest bad rate on both the development and validation data.   
  
The rank ordering break based on bad rate column is the point at which the bad rate starts to decrease significantly. In this case, we can see that there is a clear break in the bad rate decile wise. The first few models have a high bad rate, while the last few models have a significantly lower bad rate. This indicates that the model's performance has improved significantly after a certain point.   
  
In summary, the rank ordering based on bad rate column shows that the model's performance has improved significantly after a certain point, with the last few models having a significantly lower bad rate on both the development and validation data. This indicates that the model's performance has converged and is stable.



### 5.2.5 Result of RMSE test

|  |  |
| --- | --- |
| **Sample** | **RMSE VALUE** |
| Development | 0.26 |
| Validation | 0.26 |

The RMSE (Root Mean Squared Error) score is a commonly used metric to evaluate the performance of a machine learning model. It measures the difference between the predicted and actual values, and returns a single value that represents the overall error of the model. In this case, the RMSE score for both the development and validation datasets is 0.26. This indicates that the model is able to accurately predict the output values, as the RMSE score is relatively low. A lower RMSE score is preferred, as it indicates that the model is able to closely match the actual output values. In general, an RMSE score below 0.3 is considered to be a good result, while an RMSE score below 0.1 is considered to be excellent. Based on the RMSE score, we can conclude that the model has a high level of accuracy and is suitable for use in production.

## 5.3. Summarized Result

*Evaluate all test performed on respective data smaples by this model.*

Performance testing is an essential step in evaluating the effectiveness of a model. The summarized results of all performance tests or the results of the model on development and validation samples are presented in the table above. The KS value for development and validation samples is 36.6 and 32.9, respectively, indicating that the model has good discriminatory power to distinguish between positive and negative cases.  
  
The AUC values for development and validation samples are 0.75 and 0.72, respectively, which indicates that there is a moderate level of accuracy in predicting positive cases over negative ones.  
  
The GINI coefficient measures how well a model can separate positive from negative cases by ranking them according to their predicted probabilities. The GINI values for development and validation samples are 0.5 and 0.44, respectively, indicating that the model's ability to rank order cases based on their predicted probabilities is moderate.  
  
Rank Ordering Break (ROB) refers to whether or not there was any change in rank ordering when comparing actual outcomes with predicted probabilities using different cutoff points for classification purposes. In this case, ROB occurred only in the validation sample.  
  
Finally, RMSE (Root Mean Square Error) measures how well a regression line fits data points by calculating the difference between actual values and predicted values squared before taking their average root square value; it has been found to be consistent across both development and validation samples at 0.26.  
  
Overall, these performance test results provide valuable insights into how well our model performs on both training data as well as unseen data during testing phases without any bias towards either set of data used during training or evaluation stages

|  |  |  |
| --- | --- | --- |
| **Test** | **Development** | **Validation** |
| KS | 36.6 | 32.9 |
| AUC | 0.75 | 0.72 |
| GINI | 0.5 | 0.44 |
| Rank Ordering Break | NO | Yes |
| RMSE | 0.26 | 0.26 |

## 5.4. Benchmark Analysis

*Evaluate benchmark analysis on all test performed.*

Based on the benchmark analysis presented in the table, it is evident that the proposed model outperforms the benchmark model in both the development and validation samples. In the development sample, the KS statistic for the benchmark model is 19.2, while the KS statistic for the proposed model is 36.6, resulting in a significant improvement of 90.63%. Similarly, in the validation sample, the KS statistic for the benchmark model is 22.4, and the KS statistic for the proposed model is 32.9, resulting in a substantial improvement of 46.88%. These results indicate that the proposed model is more effective in capturing the underlying distribution of the data compared to the benchmark model.

|  |  |  |  |
| --- | --- | --- | --- |
| **Sample** | **Benchmark model KS** | **Model KS** | **% Change in KS** |
| Development | 19.2 | 36.6 | 90.63% |
| Validation | 22.4 | 32.9 | 46.88% |

## 5.5. Performance across Segments

*Evaluate model performance across various segments to demonstrate performance is sufficient with respect to intended purpose.*

Performance testing is an essential step in evaluating the effectiveness of a model. The summarized results of all performance tests or the results of the model on development and validation samples are presented in the table above. The KS value for development and validation samples is 36.6 and 32.9, respectively, indicating that the model has good discriminatory power to distinguish between positive and negative cases.  
  
The AUC values for development and validation samples are 0.75 and 0.72, respectively, which indicates that there is a moderate level of accuracy in predicting positive cases over negative ones.  
  
The GINI coefficient measures how well a model can separate positive from negative cases by ranking them according to their predicted probabilities. The GINI values for development and validation samples are 0.5 and 0.44, respectively, indicating that the model's ability to rank order cases based on their predicted probabilities is moderate.  
  
Rank Ordering Break (ROB) refers to whether or not there was any change in rank ordering when comparing actual outcomes with predicted probabilities using different cutoff points for classification purposes. In this case, ROB occurred only in the validation sample.  
  
Finally, RMSE (Root Mean Square Error) measures how well a regression line fits data points by calculating the difference between actual values and predicted values squared before taking their average root square value; it has been found to be consistent across both development and validation samples at 0.26.  
  
Overall, these performance test results provide valuable insights into how well our model performs on both training data as well as unseen data during testing phases without any bias towards either set of data used during training or evaluation stages

|  |  |  |
| --- | --- | --- |
| **Test** | **Development** | **Validation** |
| KS | 36.6 | 32.9 |
| AUC | 0.75 | 0.72 |
| GINI | 0.5 | 0.44 |
| Rank Ordering Break | NO | Yes |
| RMSE | 0.26 | 0.26 |

# 6. Model Implementation

## 6.1. Implementation Overview

*Describe the implementation system/environment where the model will be implemented for the model scoring.*

## 6.2. Implementation Testing & Results

*Describe the implementation testing plan along with metrics used and the expected outcome for succesful and accurate implementation. Document the results of testing to demonstrate correct implementation.*

# 7. Operating and Control Environment

*Provide evidence of show that the model resides in a secured environment where no un-authorized changes can be made to the model.*

# 8. Ongoing Monitoring and Governance Plan

## 8.1. Monitoring Frequency & Components

*Describe the frequency of the model monitoring and components that will be included in the monitoring reports.*

## 8.2. Annual Model Review plan

*Provide a plan of data and performance testing results that will be provided as part of annual model review.*

# 9. Reference

*Provide all relevant references.*

# 10. Appendix