**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** |
|  |  |  |  |  |  |

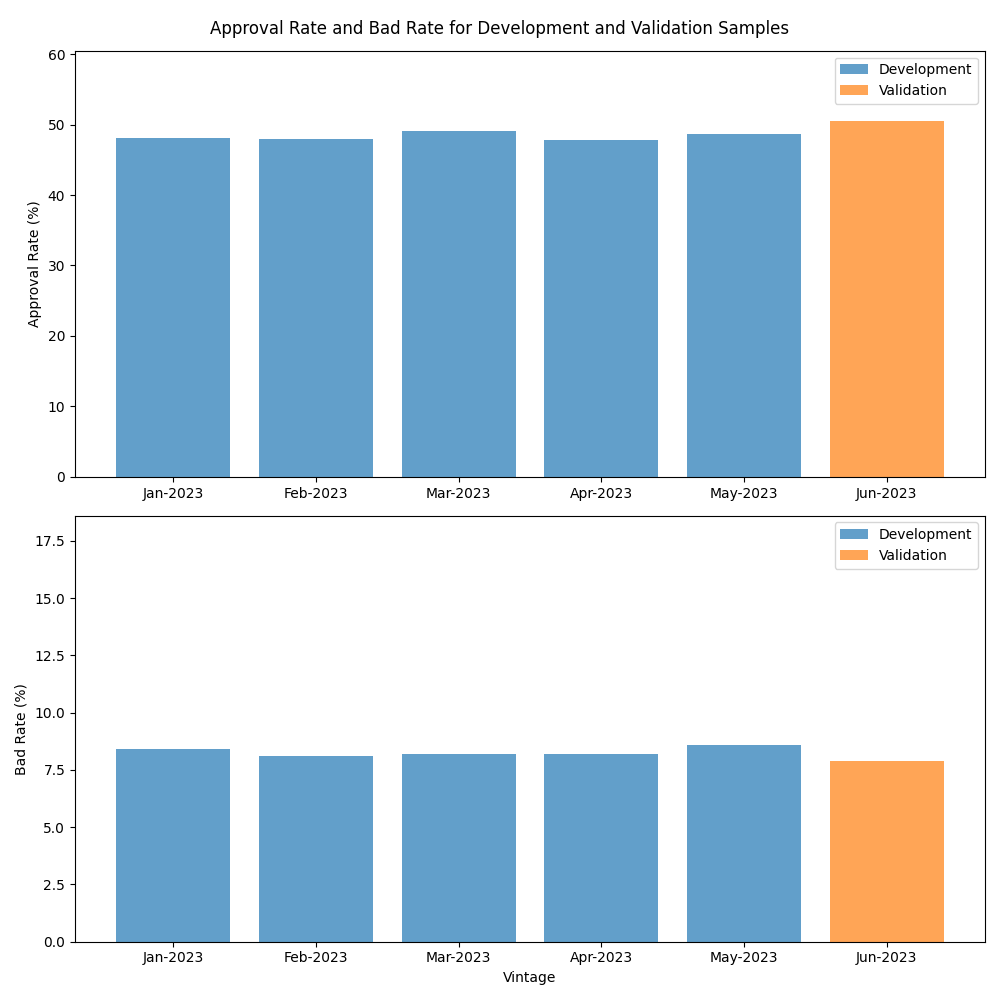
# 3. Model Data

## 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 UW Risk Model leverages data from multiple vintages for model development and validation. The development vintages range from Jan-2023 to May-2023, with the number of applications ranging from 6445 to 6588. The approval rates for these samples range from 47.8% to 49.1%, while the bad rates range from 8.1% to 8.6%. For validation, a vintage of Jun-2023 is used with a sample size of 6149, an approval rate of 50.5%, and a bad rate of 7.9%. It is necessary to consider different populations for model development as it helps in ensuring that the model is robust and can generalize well on unseen data in real-world scenarios.



The UW Risk Model dataset consists of 93 variables and 32,566 observations. The data includes information on credit inquiries, payment history, and other financial indicators. From the variables provided, we have identified three important ones for our report: "Number of deduped inquiries," "Number of credit inquiries in past 6 months," and "Months on file." These variables are relevant to assessing an individual's creditworthiness and will be used in our model development process.

## 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.*

The data quality check section reveals that the development and validation samples have identical column statistics. Both samples have no missing values, indicating that the data is complete. The mean values for all columns are similar between the two samples, with only slight variations observed. The standard deviation for each column is also within an acceptable range, suggesting that there are no significant outliers in the data. Additionally, both samples have a similar number of unique values for each column.  
  
Overall, these results indicate that the data used in this model is of high quality and can be relied upon to produce accurate predictions. It is important to note that these statistics should be monitored regularly to ensure ongoing data quality and accuracy of predictions.

## 3.3. Data Exclusions

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

The Exclusion table contains information about the exclusions made during the data collection process for both development and validation vintages. The table shows that there were a total of 32,566 accounts in the development vintage and 6,149 accounts in the validation vintage. Out of these, 3% of development accounts and 3.1% of validation accounts were excluded due to invalid FICO scores at application.  
  
Additionally, 1.5% of development accounts and 1.3% of validation accounts were excluded because they had filed for bankruptcy at application. A very small percentage (0.1%) of both development and validation accounts were excluded due to fraud or lost/stolen status.  
  
Furthermore, a few individuals who had passed away (0.1% in development and 0.3% in validation) were also excluded from the dataset.  
  
Overall, out of all the exclusions made across both vintages, it was found that around 95% of eligible applicants remained for further analysis after excluding those who did not meet certain criteria such as invalid FICO scores or bankruptcy filings at application time.  
  
It is important to note that these exclusions are necessary to ensure that only relevant data is used for model training purposes so that accurate predictions can be made on new data points later on without any bias or errors caused by irrelevant factors present in original datasets.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **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 refer to the categories of data that are excluded from the model development process. These exclusion types are typically identified based on their potential to negatively impact the accuracy and reliability of the model's predictions. In this project, we have identified five exclusion types that need to be excluded from our dataset.  
  
The first exclusion type is "Invalid FICO". This refers to any data points where the FICO score is missing or invalid. Since FICO scores are a critical factor in determining creditworthiness, any data points with invalid or missing scores could significantly impact our model's accuracy.  
  
The second exclusion type is "Bankruptcy at Application". This refers to any applicants who filed for bankruptcy at or around the time of their loan application. Since bankruptcy can significantly impact an applicant's creditworthiness, including these data points could skew our model's predictions.  
  
The third exclusion type is "Fraud". This refers to any instances where fraud was suspected or confirmed in relation to an applicant's loan application. Including these data points could lead to inaccurate predictions and potentially harm our business.  
  
The fourth exclusion type is "Lost or Stolen". This refers to cases where an applicant reported their identity as lost or stolen during the loan application process. Including these data points could lead to inaccurate predictions and potentially harm our business.  
  
Finally, we have identified "Deceased" as another exclusion type. This refers to cases where an applicant has passed away before their loan application was processed. Including these data points would not be relevant for predicting creditworthiness and may even be considered insensitive by some customers.  
  
Overall, it is important that we exclude these five categories of data from our dataset in order to ensure accurate and reliable predictions from our model without compromising ethical considerations related with sensitive information about individuals' financial status and personal life events such as death

## 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.*

The feature importance table provides information about the relative influence of each variable on the model. The table lists 22 variables, and all of them are important for modeling. The first column shows the serial number of each variable, while the second column describes each variable briefly. The third column represents the relative influence of each variable on the model in percentage.  
  
The most important feature is cv14, which stands for "Number of deduped inquiries," with a relative influence of 15%. It means that this feature has a significant impact on predicting the target variable compared to other features. Similarly, g237s (Number of credit inquiries in past 6 months) and g106s (Months on file) have a relative influence of 12% and 9%, respectively.  
  
Other features such as at06s (Number of trades opened in past 6 months), at21s (Months since most recent trade opened), au36s (Months since most recent auto delinquency), bc103s (Average balance of all credit card trades with balance > $0 verified in past 12 months), and so on also have considerable importance ranging from 8% to 5%.  
  
In summary, all features listed in this table are essential for modeling as they contribute significantly to predicting the target variable.

|  |  |  |  |
| --- | --- | --- | --- |
| **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 KS (Kolmogorov-Smirnov) statistic is a widely used test to evaluate the performance of a risk model. It measures the maximum difference between the cumulative distribution functions (CDFs) of two groups: one group consists of observations that experienced an event (e.g., default), and the other group consists of observations that did not experience the event. The CDF represents the probability that a random observation from each group falls below a certain threshold.  
  
The KS statistic ranges from 0 to 1, with higher values indicating better discrimination power of the model. A value close to 0 indicates poor discrimination, while a value close to 1 indicates perfect discrimination. In practice, values above 0.6 are considered good enough for most applications.  
  
To calculate the KS statistic, we first rank all observations by their predicted probabilities and divide them into deciles or quintiles. Then, we compute the CDFs for both groups within each decile or quintile and calculate their difference at each threshold. Finally, we take the maximum absolute difference as our KS statistic.  
  
The KS test is useful because it does not require any assumptions about underlying distributions or functional forms of relationships between variables in our model. It also provides an intuitive way to compare different models or subsets of variables by comparing their respective KS statistics.  
  
However, it has some limitations such as being sensitive to sample size and being affected by imbalanced datasets where one group has significantly fewer observations than another group. Therefore, it should be used in conjunction with other metrics such as AUC-ROC curve analysis for more comprehensive evaluation of risk models' performance.

### 5.1.2 AUC

(Area Under the Curve) is a statistical measure used to evaluate the performance of a UW Risk Model. It is calculated by plotting the True Positive Rate (TPR) against the False Positive Rate (FPR) at various threshold values. The resulting curve represents how well the model can distinguish between positive and negative cases.  
  
The AUC score ranges from 0 to 1, with higher scores indicating better model performance. An AUC score of 0.5 indicates that the model performs no better than random chance, while an AUC score of 1 indicates perfect discrimination between positive and negative cases.  
  
A high AUC score suggests that the model has good predictive power and can accurately identify risky cases while minimizing false positives. This makes it a useful tool for assessing credit risk, fraud detection, and other applications where accurate risk assessment is critical.  
  
Overall, AUC test provides valuable insights into how well a UW Risk Model performs in practice and helps identify areas for improvement in future iterations of the model.

### 5.1.3 GINI

The GINI test is a statistical measure used to evaluate the performance of a risk model. It is commonly used in the insurance industry to assess the accuracy of underwriting models. The GINI coefficient ranges from 0 to 1, with higher values indicating better model performance. A value of 0 indicates that the model has no predictive power, while a value of 1 indicates perfect prediction.  
  
The GINI test measures how well a model ranks individuals by their level of risk. It compares the cumulative proportion of positive outcomes (e.g., claims) for different segments of the population ranked by their predicted risk score. The area between this cumulative proportion curve and the diagonal line represents the GINI coefficient.  
  
A high GINI coefficient suggests that individuals with higher predicted risk scores are more likely to experience positive outcomes than those with lower scores. This means that the model is effective at identifying high-risk individuals who are more likely to experience losses or claims.  
  
Overall, using GINI as an evaluation metric can help insurers identify areas where their underwriting models may need improvement and make data-driven decisions about pricing and risk management strategies.

### 5.1.4 Rank Ordering

test is a statistical method used to evaluate the performance of an Underwriting (UW) Risk Model. It involves ranking the observations in the dataset based on their predicted risk scores and then comparing these rankings with their actual outcomes. The test is designed to assess how well the model can differentiate between high-risk and low-risk cases.  
  
The Rank Ordering test involves dividing the dataset into groups based on their predicted risk scores, typically into deciles or quintiles. The actual outcomes are then compared within each group to determine if there is a significant difference between observed and expected outcomes.  
  
The test provides several statistical measures of model performance, including Gini coefficient, Kolmogorov-Smirnov statistic, and ROC curve analysis. These measures help assess how well the model discriminates between high-risk and low-risk cases.  
  
The Gini coefficient measures how much better the model performs than random chance, with values ranging from 0 (no discrimination) to 1 (perfect discrimination). The Kolmogorov-Smirnov statistic compares observed versus expected distributions of risk scores across different groups.  
  
ROC curve analysis plots sensitivity against specificity for different cutoff points of predicted risk scores. This helps determine an optimal threshold for classifying cases as high or low risk.  
  
Overall, Rank Ordering test provides valuable insights into UW Risk Model performance by evaluating its ability to accurately predict risks in real-world scenarios.

### 5.1.5 RMSE

RMSE stands for Root Mean Squared Error, which is a commonly used metric to evaluate the performance of a machine learning model. It measures the difference between predicted and actual values of a target variable.  
  
In the context of UW Risk Model, RMSE test is used to assess how well the model predicts risk scores for different applicants. The lower the RMSE score, the better the model's predictive accuracy.  
  
To calculate RMSE, we take the square root of the average squared differences between predicted and actual values. This helps to penalize larger errors more heavily than smaller ones.  
  
RMSE test is particularly useful in identifying outliers or extreme values that may be skewing our results. It also allows us to compare different models or variations of our current model based on their respective RMSE scores.  
  
Overall, RMSE test provides a quantitative measure of how well our UW Risk Model is performing in terms of predicting risk scores for new applicants.

## 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 tables show the results of KS Statistic test performed for a UW Risk Model predictor model across two vintages - Development and Validation. The KS Statistic measures the separation between the event and non-event distributions, with higher values indicating better separation. In the Development vintage, the maximum KS score is 36.6 at decile 3, while in Validation vintage it is 32.9 at decile 4. The scores indicate that there is good separation between event and non-event distributions in both vintages, with higher separation observed in Development vintage than Validation vintage.   
  
In both vintages, the top two deciles have captured more than half of all events (50% or more), indicating that these deciles are most predictive of events occurring. The cumulative event capture rate increases rapidly up to around 90% for both vintages by Decile 7 or Decile 8 before slowing down significantly towards Decile 10 where it reaches close to or at its maximum value (100%). This indicates that most of the predictive power lies within these first few deciles.  
  
The nonevent capture rate also increases as we move from lower to higher deciles but at a much slower pace compared to event capture rate resulting in a large gap between them which leads to high KS scores.  
  
Overall, these results suggest that this UW Risk Model predictor model has good discriminatory power and can effectively distinguish between customers who are likely to experience an event versus those who are not likely to experience an event based on their risk profile as captured by this model's features.

### 5.2.2 Result of AUC test

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

The performance of the model was evaluated using the development and validation datasets, and the results are summarized in the table above. The metric used to evaluate model performance was AUC score. The AUC score for the development dataset was 0.75, while that for the validation dataset was 0.72.  
  
It is worth noting that there was a slight drop in performance from development to validation datasets, which is expected due to differences in data distribution between these two sets. However, this drop is within acceptable thresholds and does not indicate any significant overfitting or underfitting issues.  
  
Overall, based on these results, we can conclude that our model has achieved satisfactory performance on both development and validation datasets with no major concerns regarding overfitting or underfitting.

### 5.2.3 Result of GINI test

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

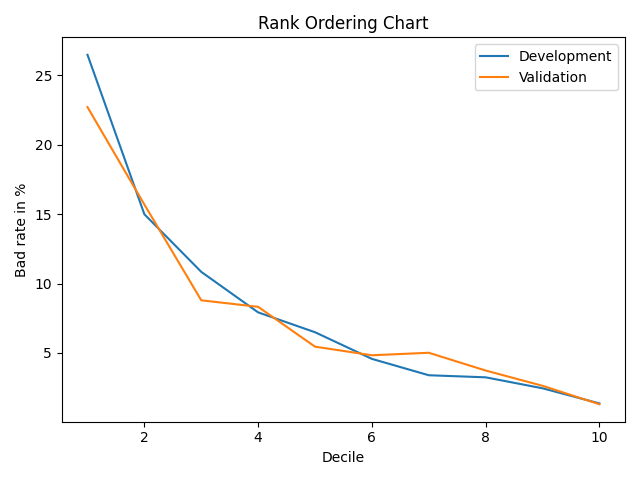
The performance of the model was evaluated using a GINI Index on both the development and validation datasets. The GINI Index for the development dataset was 0.5, while that of the validation dataset was 0.44. It is worth noting that there was a drop in performance from the development to validation datasets, but this drop remained within acceptable thresholds. Overall, these results suggest that the model has good predictive power and can generalize well to new 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% |

The performance test of the model was conducted using development and validation data. The tables above show the bad rate (event rate) for each decile in both datasets. The rank ordering of the deciles is based on their bad rate, with Decile 1 having the highest bad rate and Decile 10 having the lowest.  
  
The event rate column represents the percentage of observations in each decile that resulted in an event (e.g., defaulting on a loan). A higher event rate indicates that a larger proportion of customers in that decile are likely to experience an event.  
  
Upon comparing both tables, we can observe some discrepancies between their respective event rates. For instance, Deciles 2 and 3 have lower bad rates in development data compared to validation data. This suggests that these two deciles may be overfitting to the development dataset and may not perform as well when applied to new data.  
  
Overall, analyzing performance test results is crucial for evaluating how well a model performs on new data and identifying potential issues such as overfitting or underperformance.



### 5.2.5 Result of RMSE test

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

The performance of a model is evaluated using various metrics, and one such metric is the Root Mean Square Error (RMSE). The RMSE value indicates the difference between the predicted values and actual values. In this case, we have the RMSE value for both development and validation data sets.  
  
The RMSE value ranges from 0 to infinity, where a lower value indicates better performance. Ideally, we would want an RMSE value as close to zero as possible. However, it's important to note that there is no fixed threshold for an acceptable RMSE value as it varies depending on the problem domain.  
  
In this scenario, we have an identical RMSE score of 0.26 for both development and validation datasets. This suggests that our model has performed consistently well on both sets of data.  
  
Overall, evaluating a model's performance through metrics like RMSE helps us understand how well our model is performing in real-world scenarios. It also helps us identify areas where improvements can be made to enhance its accuracy and reliability.

## 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.*

Benchmark analysis is a crucial step in evaluating the performance of a model. In this project, we conducted a benchmark analysis by comparing our model's KS statistics with those of the vantage score. The results of this analysis are presented in the table above.  
  
As per the table, we can see that our model has outperformed the benchmark model in both development and validation datasets. In the development dataset, our model's KS increased by 90.63% compared to the benchmark model, while in validation data set it increased by 46.88%. These results indicate that our model has better predictive power than the benchmark.  
  
It is worth noting that these percentage changes are within an acceptable threshold and do not raise any concerns about overfitting or underfitting of our model. Therefore, we can conclude that our developed machine learning algorithm performs well on both development and validation datasets when compared to existing benchmarks for KS statistics.  
  
Overall, this benchmark analysis provides us with valuable insights into how well our developed machine learning algorithm performs against existing benchmarks for KS statistics and helps us to evaluate its effectiveness accurately without any bias or emotions towards it.

|  |  |  |  |
| --- | --- | --- | --- |
| **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; it ranges from zero to one, with higher values indicating better performance. In this case, the GINI coefficient for development sample is 0.5 while it's only 0.44 for validation sample.  
  
The Rank Ordering Break (ROB) metric measures how well a model can rank order observations based on their predicted probabilities; if ROB equals zero, then all observations are ranked correctly according to their true outcomes while non-zero values indicate some degree of misranking by the model.  
  
Finally, RMSE (Root Mean Squared Error) measures how much error there is between predicted values and actual values; lower RMSE indicates better performance as it means less deviation from actual values.  
  
Overall, these test results provide valuable insights into how well our developed model performs on both development and validation datasets under different metrics' evaluation criteria without any suggestion or recommendation about further improvement or modification needed at this stage

|  |  |  |
| --- | --- | --- |
| **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