Counterfactual Explanations for Credit Risk: An X-Learner Approach

By Oliver Keyes-Krysakowski
1007157434*

Executive Summary

This study provides interpretable explanations for credit card defaults by examining how changes in repayment behavior, credit limits, education, age, and marital status affect default probabilities. Using data from Taiwan's credit card crisis, it applies a method that estimates what would happen if a borrower's circumstances or behavior were different. These counterfactual explanations reveal actionable insights for both borrowers and lenders, offering clear guidance on how to reduce default risks. The findings reveal significant differences among borrower groups. Improved repayment behavior reduces default probabilities across all age groups, with younger borrowers experiencing the largest absolute reductions. Borrowers with low, medium, and high credit limits see meaningful risk reductions through better repayment behavior, while adjustments to credit limits can further influence default probabilities. Higher education generally reduces default probabilities for both single and married borrowers, although many of these effects are statistically insignificant. These counterfactual explanations make the results practical and interpretable. Financial institutions can use them to design targeted risk management strategies, such as offering repayment improvement programs for younger borrowers or tailoring credit limits based on payment history. Borrowers can understand how specific actions, like improving repayment behavior or pursuing further education, can enhance their creditworthiness. By aligning with regulatory demands for transparency and fairness, these explanations also help foster trust between borrowers and lenders. While the study is based on data from Taiwan, its methods can be applied to other credit markets, making the findings relevant for different risk management strategies.. This research advances the understanding of credit default risks by bridging the gap between financial modeling and real-world decision-making, providing clear, actionable tools for improving credit outcomes.

Keywords: Counterfactual Explanations; Credit Risk; Risk Management; X-Learners

* This paper was written as part of the requirements for the Econometrics course in the Master's in Economics program at the University of Toronto. Email: oliver.keyeskrysakowski@gmail.com. GitHub Repository: `https://github.com/OliverKeyesKrysakowski/CounterfactualExplanationsforCreditRisk`.

I. Introduction

The rapid growth of digital financial systems has driven widespread credit card adoption, providing consumers with convenience and flexibility while introducing significant risks for financial institutions. Poorly managed credit risks can lead to systemic crises, as evidenced by Taiwan's credit card crisis in the early 2000s. During this period, lax lending practices led to a surge in defaults. By 2006, over half a million individuals were unable to meet their credit obligations, resulting in severe societal repercussions, including financial distress, homelessness, and increased suicide rates. These events prompted the Taiwanese government to implement stringent regulatory reforms, emphasizing the critical need for robust credit risk management strategies.

This study employs the dataset from Yeh (2009), which documents Taiwan's credit card crisis, to explore the determinants of credit card defaults. Prior research, including Yeh and Lien (2009) and Islam, Eberle and Ghafoor (2018), has leveraged machine learning (ML) techniques to enhance predictive accuracy. However, these models often lack interpretability, limiting their utility for regulatory compliance and actionable decision-making. While advanced methods such as neural networks and hybrid ML models achieve high performance, their "black box" nature restricts their ability to provide stakeholders with clear insights into the drivers of decisions or actionable guidance to improve outcomes.

To address these challenges, this study builds on the interpretability issues identified by Wachter, Mittelstadt and Russell (2018) and applies the X-Learners framework to generate counterfactual explanations for default probabilities. Counterfactual explanations reveal the minimal changes necessary to achieve desired outcomes, offering actionable insights without requiring transparency into a model's internal mechanics. By focusing on subgroup-specific effects, this methodology investigates key interactions—such as Repayment Status × Age, Credit Limit × Repayment Status, and Education × Marital Status—to uncover how demographic and financial variables influence default probabilities. This approach aligns with Wachter, Mittelstadt and Russell (2018)'s framework for counterfactual explanations, balancing the need for interpretability with privacy protections and compliance with regulatory standards.

Our findings highlight the importance of subgroup-specific effects in understanding credit risk dynamics. We observed that default probabilities vary based on factors such as repayment behavior, credit limits, marital status, age, and educational attainment. By analyzing the key interactions, we derived counterfactual explanations that demonstrate how financial decisions and demographic characteristics shape creditworthiness.

This research contributes to the evolving field of credit risk modeling by integrating counterfactual reasoning with causal inference methodologies. By enhancing the interpretability of ML models, it provides financial institutions with actionable tools for assessing and managing credit risk while meeting regulatory requirements. By bridging the gap between predictive modeling and human-understandable explanations, this study advances the practical application of data-driven methods in shaping policy and decision-making in credit risk management.

A. Literature Review

The growing digitalization of global commerce has led to increased credit card adoption, underscoring the need for effective credit risk management. Early work by Rosenberg and Gleit (1994) laid the foundation for credit risk modeling, using static approaches like discriminant analysis and classification trees, alongside dynamic models like Markov chains. Over time, advances in machine learning (ML) have significantly enhanced predictive accuracy and decision-making capabilities in this domain.

Khandani, Kim and Lo (2010) demonstrated the power of ML algorithms in forecasting consumer credit defaults, achieving high predictive accuracy and reinforcing the transformative potential of these techniques. Similarly, Yeh and Lien (2009) compared various ML methods on Taiwan's credit card default data, finding that neural networks outperformed other approaches, highlighting ML's utility in financial risk management. Islam, Eberle and Ghafoor (2018) advanced the field by integrating heuristic methods with ML, addressing the limitations of traditional statistical models in handling complex, large-scale datasets and further highlighting the growing reliance on ML in credit risk prediction.

Despite their predictive success, ML models often lack interpretability, posing challenges for regulatory compliance and actionable insights. Predictive models like artificial neural networks (ANNs) are often viewed as "black boxes," making decisions difficult to explain. This limitation is critical in contexts such as the General Data Protection Regulation (GDPR), which requires human-understandable explanations for automated decisions. Wachter, Mittelstadt and Russell (2018) addressed this gap by proposing counterfactual explanations, which outline the minimal changes needed to alter outcomes, fostering trust and ensuring compliance. Building on these ideas, Verma et al. (2024) emphasized the value of counterfactual reasoning in enhancing ML transparency, particularly in finance. For example, actionable counterfactuals, such as steps to improve creditworthiness, benefit both applicants and institutions by clarifying decision-making processes while meeting regulatory standards.

Our study expands on this work by applying the X-Learners framework to generate counterfactual explanations for credit risk analysis. This approach moves beyond the predictive focus of traditional ML models by offering actionable insights into default probabilities. Specifically, X-Learners enable the estimation of counterfactual changes for subgroups, providing nuanced explanations of how interactions between variables affect creditworthiness. By focusing on counterfactual reasoning, our method aligns with the principles of interpretability and transparency outlined by Wachter, Mittelstadt and Russell (2018), without requiring direct insights into a model's internal logic.

Kumar and Ravi (2022) demonstrated the utility of counterfactual methods in banking and insurance, showing how changes in inputs influence outcomes like default probabilities. This approach parallels our use of X-Learners to uncover actionable insights for credit risk mitigation. Similarly, Kolodiziev, Chmutova and Lesik (2018) highlighted the broader applicability of counterfactual frameworks in financial decision-making, validating their potential to inform both policy and practice.

II. Data

This study uses the dataset from Yeh (2009), which features credit card payment data from October 2005, collected by a major Taiwanese financial institution. The dataset consists of $30,000$ individual-level observations, with 22.12% of cardholders classified as having defaulted on their payments. The primary outcome variable, Default Payment, is binary, indicating whether a cardholder defaulted. The dataset includes 23 explanatory variables, encompassing financial attributes such as credit limits, repayment history, and payment amounts, as well as demographic characteristics, including age, sex, education, and marital status.

To ensure the robustness of our analysis, we conducted data cleaning and preprocessing. Observations with missing or invalid entries for marital status and education were excluded, and education categories were consolidated into three groups: High School, Undergraduate, and Graduate. Repayment status variables, representing monthly payment behaviors over the previous six months, were recoded to set negative values as -1 to indicate no repayment delays, while other delays were adjusted to reflect incremental delinquency (e.g., one-month delay, two-month delay, and so on).

We further refined the dataset through feature engineering. Categorical variables, such as education and marital status, were one-hot encoded, with High School and Single serving as base cases. Repayment statuses were consolidated and one-hot encoded into categorical variables, such as Mild Delay and Severe Delay, with On-Time repayment serving as the base case. Credit limits were also consolidated into one-hot encoded categorical variables, such as Medium Credit Limit and High Credit Limit, with Low Credit Limit serving as the base case. Interaction terms were introduced to capture relationships, such as Repayment Status $\times$ Age, Repayment Status $\times$ Credit Limit, and Education $\times$ Marital Status.

The final dataset comprises $29,478$ observations and 30 variables, providing a detailed view of cardholder behavior. Using this dataset, we apply the X-Learners framework to generate counterfactual explanations for default probabilities across subgroups defined by our interactions of interest. This section introduces these key interactions and examines their relationship to default and non-default proportions. For comprehensive general summary statistics, detailed descriptions of key variables, and the rationale for their categorizations, please refer to Appendix Section A1: *Dataset Analysis and Variable Construction.*

A. Repayment Status $\times$ Age

The analysis first examines how repayment status and age correlate with default probabilities, offering a baseline for understanding payment behavior across age groups: Young (< 30), Middle-Aged ($30 - 50$), and Old (> 50). Analyzing Table 1, the majority of individuals in the dataset belong to the middle-aged group, with mild delays being the most prevalent repayment status across all age groups. The old group has the fewest individuals, with only 399 categorized as "On-Time," representing just 1.35% of the total sample. Regarding default versus non-default proportions, individuals with severe delays consistently exhibit the highest default rates across all age groups. Interestingly, those with mild delays demonstrate the lowest default rates, even lower than those who paid on time. A possible explanation for this counterintuitive finding is that individuals who pay on time may experience tighter financial constraints, making them less able to absorb additional interest or penalties from delayed payments. This lack of flexibility increases their vulnerability to default when faced with unforeseen expenses.

Table 1—Repayment Status × Age Interaction

Interactions	**Count**	**% of Sample**	**Default (%)**	**Non-Default (%)**
Young				
On-Time	1342	4.55%	14.01%	85.99%
Mild Delay	5566	18.88%	10.55%	89.45%
Severe Delay	3933	13.34%	42.84%	57.16%
Middle-Aged				
On-Time	3355	11.38%	14.19%	85.81%
Mild Delay	7824	26.54%	10.94%	89.06%
Severe Delay	5231	17.75%	42.69%	57.31%
Old				
On-Time	399	1.35%	17.04%	82.96%
Mild Delay	1017	3.45%	12.98%	87.02%
Severe Delay	811	2.75%	45.99%	54.01%

Notes: This table summarizes the distribution of repayment status across age groups. Each combination includes the count of individuals, their percentage of the sample, and the proportions of defaulted and non-defaulted individuals within each category.

B. *Repayment Status × Credit Limit*

The analysis next explores how repayment behavior and credit limit categories correlate with default probabilities, providing a baseline for patterns across Low Credit Limit ($\leq 50,000$), Medium Credit Limit ($50,001 - 240,000$), and High Credit Limit ($> 240,000$). Observing the results in Table 2, most individuals fall within the medium credit limit category, with mild delays being the most common repayment status. Among those with low credit limits, "On-Time" individuals are the smallest group, comprising only 425 individuals (1.44% of the sample). Default versus non-default proportions align with trends observed in the Repayment Status × Age interaction. Across all credit limit categories, individuals with severe delays exhibit the highest default rates. In the medium and high credit limit groups, individuals with mild delays consistently show the lowest default rates, even lower than those paying on time. However, in the low credit limit group, default rates for on-time and mild delay individuals are comparable. This pattern may reflect the financial constraints faced by low credit limit individuals, where smaller credit exposures could lessen the perceived stakes of default, influencing their repayment decisions.

Table 2—Repayment Status × Credit Limit Interaction

Interactions	Count	% of Sample	Default (%)	Non-Default (%)
Low Credit Limit				
On-Time	425	1.44%	16.94%	83.06%
Mild Delay	3497	11.86%	16.79%	83.21%
Severe Delay	3661	12.42%	48.37%	51.63%
Medium Credit Limit				
On-Time	2683	9.10%	14.87%	85.13%
Mild Delay	7345	24.92%	9.87%	90.13%
Severe Delay	4916	16.68%	41.90%	58.10%
High Credit Limit				
On-Time	1988	6.74%	13.13%	86.87%
Mild Delay	3565	12.09%	7.38%	92.62%
Severe Delay	1398	4.74%	32.90%	67.10%

Notes: This table summarizes the distribution of repayment status across credit limit categories. Each combination includes the count of individuals, their percentage of the sample, and the proportions of defaulted and non-defaulted individuals within each category.

C. Education × Martial Status

The final interaction examines how education and marital status correlate with default probabilities, providing a baseline for credit risk across combinations of marital status (single, married) and education levels (high school, undergraduate, graduate). In Table 3, single and married individuals are relatively evenly distributed across the dataset. Single individuals with a high school education constitute the smallest group, accounting for 2012 individuals or 6.83% of the sample. Examining default and non-default proportions, a clear trend is evident: higher educational attainment is associated with lower default rates for both single and married individuals. Graduate-level education corresponds to the lowest default rates, likely reflecting the benefits of better-paying jobs and improved financial literacy commonly associated with advanced education. Notably, the reduction in default rates across education levels is more pronounced among married individuals than single individuals. Among single individuals, those with high school and undergraduate education show relatively similar default rates. This difference may stem from the fact that married individuals are often older and in more stable financial situations compared to their younger, single counterparts with high school or undergraduate education.

Table 3—Education × Marital Status Interaction

Interactions	**Count**	**% of Sample**	**Default (%)**	**Non-Default (%)**
Single				
High School	2012	6.83%	23.81%	76.19%
Undergraduate	7182	24.36%	23.16%	76.84%
Graduate	6859	23.27%	18.47%	81.53%
Married				
High School	2861	9.71%	26.35%	73.65%
Undergraduate	6842	23.21%	24.35%	75.65%
Graduate	3722	12.63%	20.66%	79.34%

Notes: This table summarizes the distribution of education levels across marital status categories. Each combination includes the count of individuals, their percentage of the sample, and the proportions of defaulted and non-defaulted individuals within each category.

III. Methodology

This study employs the X-Learners framework to generate counterfactual explanations for credit card default. By combining machine learning with counterfactual reasoning, the methodology addresses the dual challenges of predictive modeling and interpretability in credit risk analysis. Unlike traditional applications, where treatment variables represent binary exposure (e.g., treated vs. control), this study's X-Learner framework redefines treatment as subgroup membership. For example, treatment variables indicate whether an individual belongs to subgroups such as "Younger & On-Time" or "High Credit & Severe Delay," facilitating an analysis of how subgroup-specific attributes influence default probabilities. The framework incorporates machine learning models such as XGBClassifier for propensity score estimation and XGBRegressor for outcome and treatment effect modeling. XGBoost is particularly well-suited to our data, as its scale-invariance and resilience to outliers. Additionally, its reduced sensitivity to collinearity between variables makes it an ideal candidate for credit risk data where payment behaviors are naturally correlated.

First-Stage Models for Conditional Outcomes. — In the first stage, conditional outcome models are trained separately for individuals belonging to a subgroup ($T = 1$) and those who do not ($T = 0$) using XGBRegressor. These models predict the expected outcomes of the outcome variable (default vs. non-default) given subgroup membership and covariates present in our data:

$$\hat{Y}_1(X) = \mathbb{E}[Y \mid T = 1, X], \quad \hat{Y}_0(X) = \mathbb{E}[Y \mid T = 0, X].$$

In our study, these predictions estimate the default probabilities for each subgroup, enabling us to assess how being in a subgroup influences default behavior.

Residual Calculation and Imputation of Treatment Effects. — Next, residuals are computed to impute treatment effects, capturing the counterfactual outcomes for individuals had their treatment status been reversed. For individuals within a subgroup ($T = 1$), the residuals represent their observed outcome adjusted by their estimated counterfactual outcome had they not been part of the subgroup:

$$\hat{R}_1 = Y_1 - \hat{Y}_0(X_1).$$

Conversely, for individuals outside the subgroup ($T = 0$), the residuals represent their observed outcome adjusted by their estimated counterfactual outcome had they belonged to the subgroup:

$$\hat{R}_0 = \hat{Y}_1(X_0) - Y_0.$$

In these equations, Y_1 and Y_0 denote the observed outcomes for treated and untreated individuals, respectively, while $\hat{Y}_0(X_1)$ and $\hat{Y}_1(X_0)$ are the predicted counterfactual outcomes obtained from the first-stage models. These residuals quantify the subgroup-specific treatment effects by estimating the difference between actual and counterfactual outcomes. This step quantifies how subgroup membership affects default probabilities.

Second-Stage Models for Treatment Effect Refinement. — To refine the imputed treatment effects, residuals ($\hat{R}_1$ and $\hat{R}_0$) are used to train second-stage models with XGBRegressor. These models generalize the treatment effects based on covariates:

$$\hat{g}_1(X) = \mathbb{E}[\hat{R}_1 \mid T = 1, X], \quad \hat{g}_0(X) = \mathbb{E}[\hat{R}_0 \mid T = 0, X]$$

For our study, this step refines the estimation of how subgroup-specific characteristics (e.g., being younger or having a severe repayment delay) influence the likelihood of default. By generalizing the residual treatment effects, these models ensure robust predictions across varying covariate values.

Propensity Score Estimation. — The propensity score, $\hat{e}(X)$, is estimated using XGBClassifier. This score represents the likelihood of subgroup membership given the covariates:

$$\hat{e}(X) = P(T = 1 \mid X)$$

Propensity scores balance the probability of subgroup membership across the sample, allowing fair comparisons and avoiding biases in counterfactual estimates. For instance, this step ensures that comparisons between "Younger & On-Time" and "Older & Severe Delay" subgroups are not skewed by differences in covariate distributions. This is particularly important given the imbalance in our data, as propensity scores help reduce the bias introduced by such disparities.

CATE Estimation. — The final CATE combines the predictions of the second-stage models for treated and untreated individuals using the propensity scores. This provides the counterfactual treatment effect for each individual:

$$\hat{\tau}(X) = \hat{e}(X) \cdot \hat{g}_1(X) + (1 - \hat{e}(X)) \cdot \hat{g}_0(X)$$

The weighting mechanism ensures that subgroup-specific predictions are appropriately emphasized: when the propensity score $\hat{e}(X)$ is high, greater weight is assigned to the treated model $\hat{g}_1(X)$, and when $\hat{e}(X)$ is low, greater weight is assigned to the untreated model $\hat{g}_0(X)$. This step delivers the subgroup-specific estimates of how default probabilities change due to subgroup characteristics.

Bootstrapping for Confidence Intervals. — Finally, bootstrapping is employed to quantify uncertainty in the CATE estimates. For each of $B = 1000$ bootstrap samples, propensity scores and second-stage models are recalculated, and CATEs are estimated as:

$$\hat{\tau}_b^*(X) = \hat{e}_b^*(X) \cdot \hat{g}_{1,b}^*(X) + (1 - \hat{e}_b^*(X)) \cdot \hat{g}_{0,b}^*(X)$$

The distribution of the bootstrap samples, $\{\hat{\tau}_b^*\}_{b=1}^B$, is used to compute 95% confidence intervals:

$$\mathrm{CI}_{\text{lower}} = \text{Percentile}_{2.5}(\{\hat{\tau}_b^*\}), \quad \mathrm{CI}_{\text{upper}} = \text{Percentile}_{97.5}(\{\hat{\tau}_b^*\}).$$

IV. Results

This section presents the results of the X-Learners framework for key interactions of interest. Each table reports Conditional Average Treatment Effects (CATEs), reflecting the absolute change in default probabilities attributable to subgroup membership, compared to a counterfactual scenario where individuals do not belong to the subgroup. These findings enable counterfactual explanations, illustrating how changes in attributes affect default probabilities.

A. *Repayment Status × Age*

The results in Table 4 provide a detailed examination of the Conditional Average Treatment Effects (CATEs) of repayment status across different age groups. For younger borrowers, repayment behavior has a pronounced impact on default probabilities. Being in the "Young & Severe Delay" subgroup increases default risk by 5.30% (0.05298^{**}), while transitioning to the "Young & On-Time" subgroup reduces it by 8.24% (-0.08243^{**}). Similarly, moving from "Severe Delay" to "Mild Delay" reduces default risk by 8.24% (-0.08237^{**}). These findings highlight the significant benefits of improving repayment behavior among younger borrowers, though the marginal gain diminishes for those transitioning from "Mild Delay" to "On-Time."

For middle-aged borrowers, repayment improvements also reduce default probabilities but to a lesser extent. Being in the "Middle-Aged & Severe Delay" subgroup increases default probability by 5.63% (0.05631^{**}), while transitioning to "On-Time" reduces it by 4.03% (-0.04026^{**}). Moving from "Mild Delay" to "On-Time" decreases default probability by 6.77% (-0.06774^{**}). These results indicate that while repayment improvements benefit middle-aged borrowers, the effects are less pronounced compared to younger borrowers.

For older borrowers, being in the "Older & Severe Delay" subgroup increases default risk by 8.35% (0.08351^{**}), the largest increase among all groups. Transitioning from "Severe Delay" to "Mild Delay" significantly reduces default probability by 7.88% (-0.07879^{**}). However, the effect of being in the "Older & On-Time" subgroup is not statistically significant (-0.04203), suggesting limited evidence of further benefit from consistent on-time repayments in this demographic.

Therefore, counterfactual insights suggest that improving repayment behavior can significantly reduce default probabilities, with younger borrowers experiencing the largest absolute reductions. Middle-aged borrowers also benefit from improved repayment behavior, though the effects are slightly smaller. For older borrowers, moderate improvements can substantially lower default risk. Comparing Table 1, individuals in the "Severe Delay" category still exhibit the highest default rates. In addition, mild delays continue to have lower default rates than on-time payments, except among young borrowers, where the rates are comparable.

Table 4—Repayment Status × Age Interaction: CATE Estimates

Interactions	**CATE Estimate**	**95% Confidence Interval**
Young		
On-Time	-0.08243**	[-0.12699, -0.04639]
Mild Delay	-0.08237**	[-0.11969, -0.09355]
Severe Delay	0.05298**	[0.04800, 0.09291]
Middle-Aged		
On-Time	-0.04026**	[-0.11120, -0.03785]
Mild Delay	-0.06774**	[-0.10706, -0.08184]
Severe Delay	0.05631**	[0.04772, 0.08481]
Old		
On-Time	-0.04203	[-0.09739, 0.02403]
Mild Delay	-0.07879**	[-0.11546, -0.06987]
Severe Delay	0.08351**	[0.06475, 0.15652]

Notes: This table reports the Conditional Average Treatment Effects (CATEs) for default probabilities across repayment status and age subgroups. The CATE represents the estimated change in default probability for individuals in a given subgroup compared to a counterfactual scenario where they do not belong to that subgroup, holding other characteristics constant. Negative values indicate a reduction in default probability due to subgroup membership, while positive values indicate an increase. Confidence intervals represent the range within which the true effect is likely to fall (95%). Significance at the 5% level is denoted by **, indicating intervals that exclude zero.

B. *Repayment Status × Credit Limit*

The results in Table 5 examine the CATEs of repayment status across credit limit categories. For borrowers with low credit limits, "Severe Delay" increases default probability by 6.77% (0.06766**), while transitioning to "On-Time" reduces it by 4.72% (-0.04716^{**}). However, partial improvements, such as moving to "Mild Delay" (-0.00641), show no statistically significant impact. This suggests that for low credit borrowers, only substantial repayment improvements effectively mitigate default risks.

For medium credit borrowers, "Severe Delay" increases default probability by 7.80% (0.07804**), while transitioning to "On-Time" reduces it by 6.74% (-0.06745^{**}). Moving from "Severe Delay" to "Mild Delay" leads to a significant 7.89% (-0.07891^{**}) reduction, showing that even partial repayment improvements meaningfully lower default risk for this group.

For high credit borrowers, "Severe Delay" (0.00362) has no statistically significant impact on default probability, while transitioning to "On-Time" reduces it by 7.48% (-0.07478^{**}), and to "Mild Delay" by 12.41% (-0.12412^{**}). These results suggest that high credit borrowers benefit most from moderate repayment improvements, as severe delays appear less consequential for this group.

To continue, counterfactual insights also reveal how changes in credit limits influence default probabilities for individuals with the same repayment behavior. For borrowers with "Severe Delay", moving from a medium credit limit (7.80%, 0.07804^{**}) to a low credit limit (6.77%, 0.06766^{**}) reduces default probability by 1.03%. For borrowers with "On-Time" repayment behavior, transitioning from a medium credit limit (-6.74%, -0.06745^{**}) to a high credit limit (-7.48%, -0.07478^{**}) results in a modest 0.74% reduction in default probability. Conversely, moving to a low credit limit (-4.72%, -0.04716^{**}) increases default probability by 2.02%. For borrowers in the "Mild Delay" subgroup, transitioning from medium credit (-7.89%, -0.07891^{**}) to high credit (-12.41%, -0.12412^{**}) yields a 4.52% reduction in default probability.

Therefore, borrowers with low, medium, and high credit limits experience significant reductions in default probabilities with improved repayment behavior. Counterfactual insights further demonstrate how adjusting credit limits can change default probabilities. Consistent with Table 2, "Severe Delay" still exhibits the highest default rates across all credit limit groups. Notably, in the low credit category, default rates for "On-Time" and "Mild Delay" individuals differ in this analysis, contrasting with Table 2, where their default rates were comparable. However, as seen in Table 2, mild delays continue to show lower default rates than on-time payments.

Table 5—Repayment Status × Credit Limit Interaction: CATE Estimates

Interactions	**CATE Estimate**	**95% Confidence Interval**
Low Credit Limit		
On-Time	-0.04716**	[-0.09816, -0.02069]
Mild Delay	-0.00641	[-0.05950, 0.00153]
Severe Delay	0.06766**	[0.04462, 0.13230]
Medium Credit Limit		
On-Time	-0.06745**	[-0.11066, -0.01094]
Mild Delay	-0.07891**	[-0.12266, -0.09051]
Severe Delay	0.07804**	[0.06747, 0.10634]
High Credit Limit		
On-Time	-0.07478**	[-0.12678, -0.02430]
Mild Delay	-0.12412**	[-0.15857, -0.12604]
Severe Delay	0.00362	[-0.02245, 0.03652]

Notes: This table reports the Conditional Average Treatment Effects (CATEs) for default probabilities across repayment status and credit limit subgroups. The CATE measures the estimated change in default probability for individuals in a specific subgroup compared to a counterfactual scenario where they are not part of that subgroup, holding other characteristics constant. Negative values indicate reduced default risk due to subgroup membership, while positive values indicate increased risk. The 95% confidence intervals reflect the range within which the true effect is likely to fall. Significance at the 5% level is denoted by **, indicating intervals that exclude zero.

C. *Education × Marital Status*

The results in Table 6 highlight the relationship between education and marital status in shaping default probabilities. For single individuals, the CATE estimates suggest that increased education modestly reduces default probabilities, although only the effect for graduate education (-1.45%,

-0.01453^{**}) is statistically significant. Transitioning from high school (-0.00446) to undergraduate education (-0.00785) does not yield statistically significant changes, as both confidence intervals include zero. However, moving to graduate education significantly lowers default probabilities, emphasizing the protective effect of higher education for single borrowers.

For married individuals, the results demonstrate a different pattern. The CATE estimate for those with an undergraduate degree (1.40%, 0.01399^{**}) is positive and statistically significant, indicating an increased probability of default compared to their high school counterparts (1.34%, 0.01344), although the latter is not statistically significant. Similarly, the effect of graduate education (1.18%, 0.01178) is not statistically significant, suggesting that higher education does not significantly alter default probabilities for married individuals. These findings imply that for married borrowers, default probabilities may be influenced more by other factors, such as household income dynamics, rather than education alone.

The results align with trends in Table 3, where higher education correlates with lower default rates for both single and married individuals. However, while the CATE estimates show a similar pattern, the majority are statistically insignificant, suggesting that the effect of education on default probabilities may be influenced by other factors.

Table 6—Education × Marital Status Interaction: CATE Estimates

Interactions	**CATE Estimate**	**95% Confidence Interval**
Single		
High School	-0.00446	[-0.02156, 0.01117]
Undergraduate	-0.00785	[-0.01839, 0.00220]
Graduate	-0.01453**	[-0.02744, -0.00462]
Married		
High School	0.01344	[-0.00178, 0.03420]
Undergraduate	0.01399**	[0.00559, 0.02734]
Graduate	0.01178	[-0.00588, 0.03015]

Notes: This table reports the Conditional Average Treatment Effects (CATEs) for default probabilities across education and marital status subgroups. The CATE measures the estimated change in default probability for individuals in a specific subgroup compared to a counterfactual scenario where they are not part of that subgroup, holding other characteristics constant. Negative values indicate reduced default risk due to subgroup membership, while positive values indicate increased risk. The 95% confidence intervals reflect the range within which the true effect is likely to fall. Significance at the 5% level is denoted by **, indicating intervals that exclude zero.

V. Discussion

A. *Significance*

The results derived from our X-Learners framework highlight the value of counterfactual explanations in identifying subgroup-specific determinants of default probabilities. By estimating Conditional Average Treatment Effects (CATEs) across interactions such as Repayment Status × Age, Repayment Status × Credit Limit, and Education × Marital Status, this study provides actionable insights that bridge predictive modeling and practical decision-making in credit risk management.

The significance of these findings is twofold. First, they provide granular, quantifiable insights into how individual characteristics influence default risks, offering valuable guidance for financial institutions aiming to enhance their risk assessment models. For example, the Repayment Status $\times$ Age interaction highlights that improving repayment behavior can significantly lower default probabilities across all age groups, with younger borrowers experiencing the largest absolute reductions. Similarly, the Repayment Status $\times$ Credit Limit interaction demonstrates that borrowers across low, medium, and high credit limits can achieve meaningful reductions in default probabilities through improved repayment behavior, while adjustments to credit limits can also alter default probabilities. The Education $\times$ Marital Status interaction further illustrates the role of educational attainment, showing a general reduction in default probabilities at higher education levels for single and married borrowers; however, many of these effects are statistically insignificant. Second, these counterfactual explanations have practical implications for credit risk management. Financial institutions can leverage the insights to construct subgroup-specific risk profiles, enabling a more nuanced understanding of market dynamics. Such profiles enhance the precision of credit assessments and strengthen decision-making processes related to lending and credit issuance. Moreover, by integrating counterfactual explanations with predictive models, institutions can better capture and quantify risks, particularly in imbalanced datasets where certain subgroups, such as those with severe delays or older borrowers, may be underrepresented.

For borrowers, these counterfactual explanations offer transparency and actionable recommendations, aligning with regulatory standards such as GDPR that emphasize fairness and interpretability in credit decisions. Borrowers can understand how adjustments in behaviors or characteristics—such as improving repayment patterns, altering credit limits, or pursuing further education—can positively influence their creditworthiness. This transparency not only enhances consumer trust but also provides a roadmap for individuals aiming to improve their financial standing.

The X-Learners framework's robustness in handling imbalanced datasets further underscores its utility. By estimating missing counterfactual outcomes, such as default probabilities for individuals in subgroups they do not currently belong to, the framework isolates the effects of subgroup membership. This capability is particularly important in credit markets, where natural imbalances in default and non-default populations can distort traditional models. Consequently, the framework ensures reliable and interpretable results, offering both academic and practical contributions to the field of credit risk analysis.

B. Limitations

Although the dataset includes a substantial sample size of 30,000 individuals, certain subgroups within the key interactions are relatively small. For example, the "Old & On-Time" subgroup comprises only of 399 individuals, representing 1.35% of the total sample—the smallest of all subgroups. While XGBoost, the model used in our framework, is less sensitive to outliers compared to many traditional models, small sample sizes inherently amplify the influence of individual observations. This heightened sensitivity can affect the robustness of CATE estimates for these subgroups and may limit the generalizability of the findings. In addition, the natural imbalance between default and non-default cases within the dataset presents another challenge. Credit datasets typically exhibit a much larger non-default population compared to default cases, which can influence estimates. Although the X-Learners framework helps mitigate such imbalances, they remain a potential source of bias in the results.

To continue, the X-Learners framework relies on multiple models to estimate counterfactual outcomes, and each model introduces potential bias. While this study incorporates hyperparameter tuning to optimize performance, the complexity of using multiple models still presents challenges. Small deviations in predictions from one stage can propagate through the framework, potentially influencing the final CATE estimates. Additionally, while XGBoost is well-suited for our data due to its robustness to outliers and its ability to handle imbalanced datasets, it may not necessarily be the most appropriate model for all scenarios. Alternative algorithms might better capture specific patterns or relationships within the data, potentially yielding improved estimates.

While this study focuses on counterfactual explanations rather than causal inference, the issue of endogeneity may still influence the results. Endogenous variables are a common challenge in credit modeling, even in industry-standard predictive models, and their presence introduces bias. However, similar to predictive models, the X-Learners framework is designed to explain subgroup-specific effects using the available data. It serves as a decision-support tool for institutions, complementing but not replacing the judgment of decision-makers.

Finally, while the methodology generalizes individuals to their respective subgroups, the data itself is drawn from a Taiwanese credit market in October 2005. As such, the results may not fully generalize to other credit markets or time periods. Nonetheless, the X-Learners framework remains adaptable and can be applied to new datasets from specific markets or timeframes, ensuring relevance for financial institutions working with their own borrower data.

C. Future Works

Future research could compare CATE estimates with alternative frameworks, such as the DR-Learner model, to validate consistency and assess robustness. Additionally, experimenting with different models within the X-Learners framework may yield more precise or statistically significant estimates. Expanding the analysis to include new interaction types or additional variables, such as credit utilization, prior delinquency, or macroeconomic factors, could reveal new counterfactual insights. Examining interactions beyond the current focus, such as geographic or industry-specific effects, may further broaden the framework's applicability. Analyzing temporal dynamics with longitudinal data offers another avenue for exploration. Such an approach could uncover how subgroup-specific effects evolve over time in response to macroeconomic trends or changes in credit policies, providing insights into the stability of counterfactual explanations. Lastly, applying the framework to datasets from different regions or credit markets would test its generalizability and adaptability. This would ensure its relevance across diverse borrower populations and economic contexts, enabling financial institutions to tailor strategies to their specific needs.

VI. Conclusion

This study applies the X-Learners framework to generate counterfactual explanations for credit risk analysis, focusing on subgroup-specific determinants of default probabilities. By estimating Conditional Average Treatment Effects (CATEs) for interactions such as Repayment Status $\times$ Age, Repayment Status $\times$ Credit Limit, and Education $\times$ Marital Status, we provide actionable insights into how demographic and financial factors influence default risk.

Our findings demonstrate the value of counterfactual explanations for credit risk management. Improved repayment behavior significantly reduces default probabilities across all age groups, with younger borrowers experiencing the largest benefits. Borrowers across all credit limit levels also see meaningful reductions in risk with better repayment behavior, while changes to credit limits can further influence default probabilities. Additionally, higher education generally lowers default risk for both single and married borrowers, although many of these effects are statistically insignificant. These insights support the creation of subgroup-specific risk profiles, enhancing credit assessments and decision-making. For borrowers, counterfactual explanations provide transparency and actionable guidance, aligning with GDPR regulations and fostering greater trust.

The X-Learners framework's ability to handle imbalanced datasets and estimate missing counterfactual outcomes highlights its robustness and adaptability. While based on 2005 Taiwanese credit market data, the methodology is flexible and applicable to diverse markets and timeframes, ensuring broad relevance.

This study bridges the gap between predictive modeling and interpretability, advancing the integration of causal inference techniques into credit risk modeling. Future research can refine this approach by experimenting with alternative models, expanding interactions, and exploring longitudinal effects, further enriching credit risk analysis and decision-making in the financial industry.

REFERENCES

Islam, Sheikh Rabiul, William Eberle, and Sheikh K. Ghafoor. 2018. "Credit Default Mining Using Combined Machine Learning and Heuristic Approach." *arXiv e-prints.*

Khandani, Amir E., Adlar J. Kim, and Andrew W. Lo. 2010. "Consumer Credit Risk Models Via Machine-Learning Algorithms." *AFA 2011 Denver Meetings Paper.*

Kolodiziev, Oleh, Iryna Chmutova, and Vitaliy Lesik. 2018. "Use of causal analysis to improve the monitoring of the banking system stability." *Banks and Bank Systems*, 13: 62–76.

Kumar, Satyam, and Vadlamani Ravi. 2022. "Application of Causal Inference to Analytical Customer Relationship Management in Banking and Insurance." *arXiv*, abs/2208.10916.

Rosenberg, Eric S., and Alan Gleit. 1994. "Quantitative Methods in Credit Management: A Survey." *Operations Research*, 42(4): 589–613.

Verma, Sahil, Varich Boonsanong, Minh Hoang, Keegan E. Hines, John P. Dickerson, and Chirag Shah. 2024. "Counterfactual Explanations and Algorithmic Recourses for Machine Learning: A Review." *ACM Computing Surveys*, 56(12).

Wachter, Sandra, Brent Mittelstadt, and Chris Russell. 2018. "Counterfactual Explanations without Opening the Black Box: Automated Decisions and the GDPR." *Harvard Journal of Law & Technology*, 31(2).

Yeh, I-Cheng. 2009. "Default of Credit Card Clients." *UCI Machine Learning Repository*, DOI: https://doi.org/10.24432/C55S3H.

Yeh, I-Cheng, and Che-Hui Lien. 2009. "The comparisons of data mining techniques for the predictive accuracy of probability of default of credit card clients." *Expert Systems with Applications*, 36(2, Part 1): 2473–2480.

APPENDIX

A1. Dataset Analysis and Variable Construction

General Summary Statistics. — After data cleaning and preprocessing, we identified the variables summarized in Table A1, which serve as the foundation for examining key interactions, including Repayment Status $\times$ Age, Repayment Status $\times$ Credit Limit, and Education $\times$ Marital Status. The data reveal a clear imbalance in payment status, with non-defaults significantly outweighing defaults. Education levels exhibit notable diversity, with undergraduate education comprising the largest category, followed by smaller shares for high school and graduate education. Age distributions show that middle-aged individuals constitute the largest group, followed by younger individuals, while older individuals represent the smallest share of the sample. Marital status is relatively evenly distributed between married and single individuals. Repayment behavior varies, with mild delays being the most common, followed by severe delays and a smaller proportion of on-time payments. Credit limits are distributed relatively evenly across low, medium, and high categories. Next, we focus on the Age, Repayment Status, and Credit Limit variables, outlining the rationale behind their categorizations.

Table A1—Summary Statistics for Selected Variables of Interest

Category	**Count**	**% of Sample**
Payment Status		
Default	6,598	22.38%
Non-Default	22,880	77.62%
Education		
High School	4,873	16.53%
Undergraduate	14,024	47.57%
Graduate	10,581	35.89%
Marital Status		
Married	13,425	45.54%
Single	16,053	54.46%
Age		
Young	10,841	36.78%
Middle-Aged	16,410	55.67%
Old	2,227	7.55%
Repayment Status		
On Time	5,096	17.29%
Mild Delay	14,407	48.87%
Severe Delay	9,975	33.84%
Credit Limit		
Low Credit Limit	7,583	25.72%
Medium Credit Limit	14,944	50.70%
High Credit Limit	6,951	23.58%

Notes: This table summarizes the demographic and financial characteristics of the dataset used in the study. Counts represent the number of observations in each category, and percentages represent their share of the total sample.

Age. — In the dataset, Age was initially recorded as a continuous variable. To enable compatibility with our interactions of interest and the X-Learner framework, we categorized individuals into age buckets. Table A2 presents summary statistics for Age, revealing that 25% of individuals are aged 28 or younger, the median age is 34, and 75% are above 41. Figure A1 further illustrates this distribution, showing a positive skew with a concentration of individuals in the 30-40 age range. To achieve balanced sample sizes and align with logical groupings, we used these percentiles and the distribution plot to define three age categories: Young (< 30), Middle-Aged ($30 - 50$), and Old (> 50).

Table A2—Summary Statistics for Age

Count	**Mean**	**Std**	**Min**	**25%**	**Median**	**75%**	**Max**
29,478	35.47	9.22	21.00	28.00	34.00	41.00	79.00

Notes: This table provides summary statistics for the Age variable.

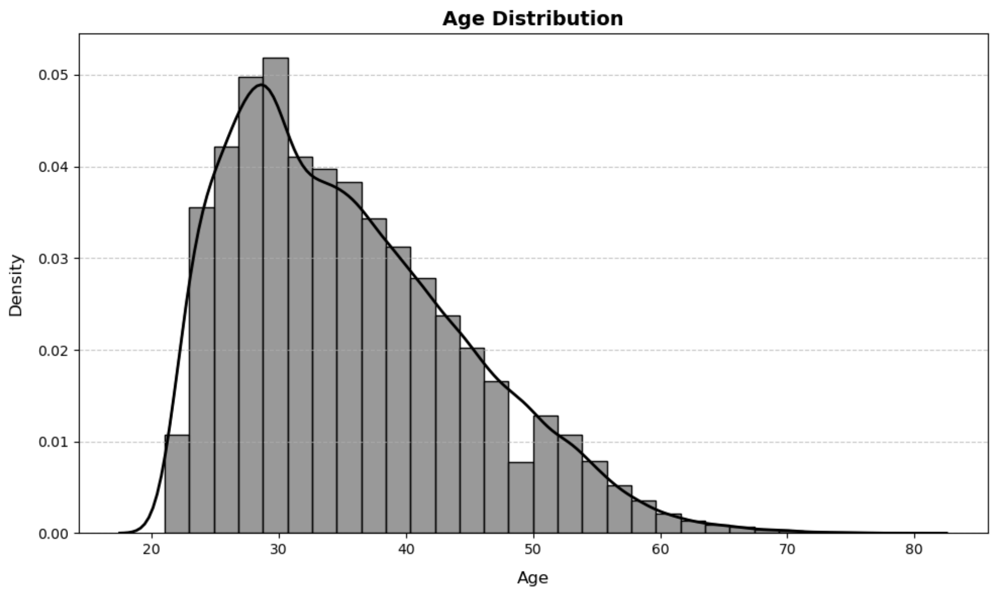

Figure A1. Age Distribution

Notes: This figure shows the distribution of ages in the dataset, highlighting the density of observations. The distribution is positively skewed, with a concentration of individuals in the 30-40 age range, tapering off toward older ages.

Repayment Status. — In the original dataset, repayment status was recorded for the six months preceding October 2005, the period during which the data was retrieved from the Taiwanese financial institution. Each variable (period) represents repayment behavior for a specific month: the most recent month, the second most recent month, and so on, up to the sixth most recent month. The repayment status is coded on a scale from -1 to 9, where -1 indicates payments made on time, and 1 through 9 represent delayed payments, with the number corresponding to the months of delay. As shown in Figure A2, the distribution of repayment status reveals that most individuals either made payments on time across all six periods or experienced a delay of one month in one of the six periods. Notably, only the most recent period shows a significant proportion of two-month delays, while delays of three months are pronounced across the six periods. After this, delays decrease sharply. Given the large number of repayment variables, using all six in their raw form for interactions would be inefficient and analytically redundant. To address this, repayment behavior was consolidated into three categories. The first category, On-Time, includes individuals who made on-time payments across all six periods. The second category, Mild Delay, comprises individuals with exactly one delayed payment in any of the six periods. Finally, the third category, Severe Delay, includes individuals with two or more delayed payments in any of the six periods. This categorization simplifies the analysis while preserving the essential variation in repayment behavior.

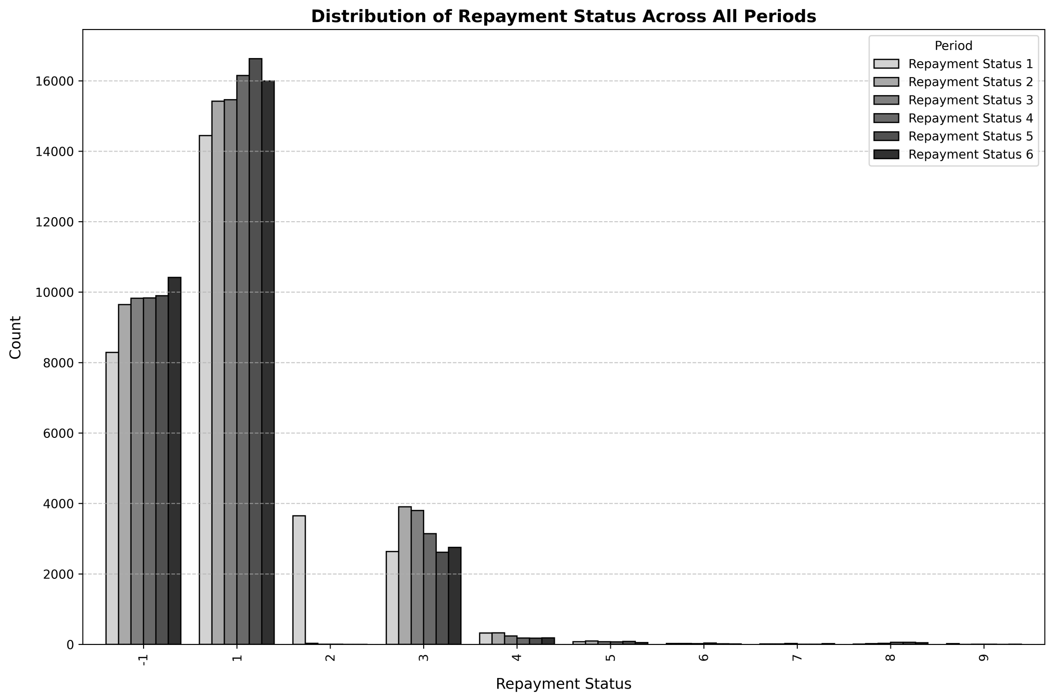

Figure A2. Distribution of Repayment Status Across All Periods

Notes: The dataset captures repayment status for the six months preceding October 2005, sourced from a Taiwanese financial institution. Each period represents repayment behavior for a specific month: the most recent month, the second most recent month, and so on, up to the sixth most recent month. Repayment status is coded from -1 to 9, where -1 indicates payments made on time, and 1 through 9 represent delayed payments, with the number corresponding to months of delay.

Credit Limit. — In the dataset, Credit Limit was initially recorded as a continuous variable. To enable compatibility with our interactions of interest and the X-Learner framework, we categorized individuals into credit limit buckets. Table A3 provides summary statistics for Credit Limit, revealing that 25% of individuals have credit limits at or below \$50,000, the median credit limit is \$140,000, and 75% of individuals have credit limits at or below \$240,000. Figure A3 illustrates the distribution of credit limits, showing a positive skew, with most individuals concentrated at lower credit limits and a tapering off as limits increase. To achieve balanced sample sizes and align with logical groupings, we used these percentiles and the distribution plot to define three credit limit categories: Low Credit Limit ($\leq 50,000$), Medium Credit Limit ($50,001 - 240,000$), and High Credit Limit ($> 240,000$).

Table A3—Summary Statistics for Credit Limit

Count	**Mean**	**Std**	**Min**	**25%**	**Median**	**75%**	**Max**
29,478	167,328	129,971	10,000	50,000	140,000	240,000	1,000,000

Notes: This table provides summary statistics for the Credit Limit variable.

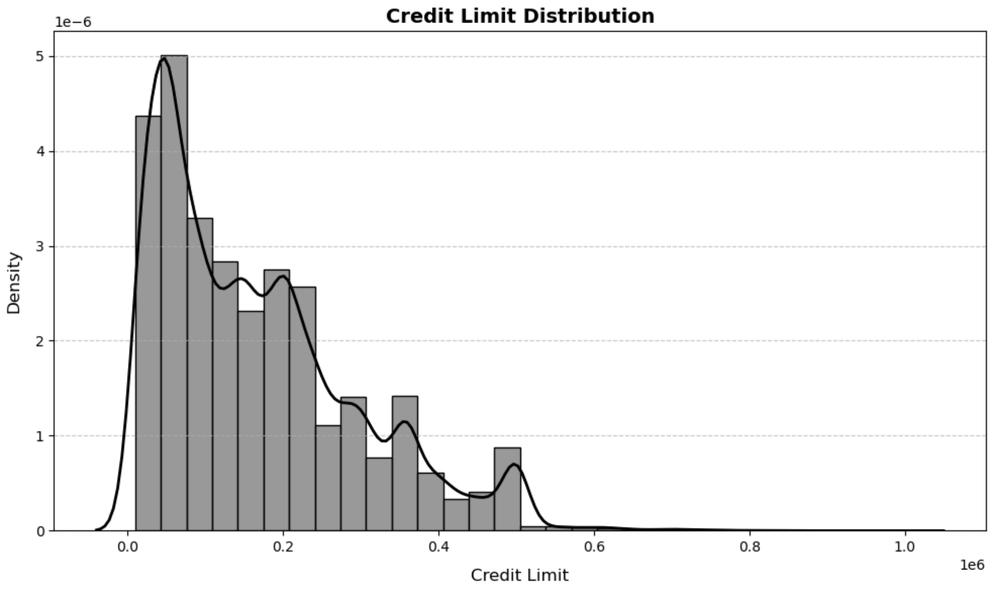

Figure A3. Credit Limit Distribution

Notes: This figure shows the distribution of credit limits in the dataset, highlighting the density of observations. The distribution is positively skewed, with most individuals having a relatively low credit limit, with a few individuals having much higher credit limits.

Code

Data Cleaning

```
[ ]: import pandas as pd
     import matplotlib.pyplot as plt
     import seaborn as sns
     import numpy as np
     from xgboost import XGBClassifier, XGBRegressor, DMatrix, cv
     from sklearn.utils import resample

     # Load dataset
     file_path = 'defaultdata.csv'
     data = pd.read_csv(file_path, header=1)

     # Rename columns
     column_mapping = {
         'LIMIT_BAL': 'Credit_Limit',
         'SEX': 'Gender',
         'EDUCATION': 'Education_Level',
         'MARRIAGE': 'Marital_Status',
         'AGE': 'Age',
         'PAY_0': 'Repayment_Status_1',
         'PAY_2': 'Repayment_Status_2',
         'PAY_3': 'Repayment_Status_3',
         'PAY_4': 'Repayment_Status_4',
         'PAY_5': 'Repayment_Status_5',
         'PAY_6': 'Repayment_Status_6',
         'BILL_AMT1': 'Bill_Amount_1',
         'BILL_AMT2': 'Bill_Amount_2',
         'BILL_AMT3': 'Bill_Amount_3',
         'BILL_AMT4': 'Bill_Amount_4',
         'BILL_AMT5': 'Bill_Amount_5',
         'BILL_AMT6': 'Bill_Amount_6',
         'PAY_AMT1': 'Payment_Amount_1',
         'PAY_AMT2': 'Payment_Amount_2',
         'PAY_AMT3': 'Payment_Amount_3',
         'PAY_AMT4': 'Payment_Amount_4',
         'PAY_AMT5': 'Payment_Amount_5',
         'PAY_AMT6': 'Payment_Amount_6',
         'default payment next month': 'Default_Payment'
     }
     data.rename(columns=column_mapping, inplace=True)

     # Clean data
     data = data[(data['Marital_Status'] != 0) & data['Education_Level'].isin([1, 2,␣
      ↪3])]
```

```
pay_features = ['Repayment_Status_1', 'Repayment_Status_2',␣
 ↪'Repayment_Status_3',
                'Repayment_Status_4', 'Repayment_Status_5',␣
 ↪'Repayment_Status_6']
for feature in pay_features:
    data.loc[data[feature] < 0, feature] = -1
    data.loc[data[feature] >= 0, feature] += 1
    data[feature] = data[feature].astype('int64')

# Encode features
data['Grad_School'] = (data['Education_Level'] == 1).astype('int')
data['University'] = (data['Education_Level'] == 2).astype('int')
data.drop('Education_Level', axis=1, inplace=True)
data['Female'] = (data['Gender'] == 2).astype('int')
data['Married'] = (data['Marital_Status'] == 1).astype('int')
data.drop(['Gender', 'Marital_Status'], axis=1, inplace=True)
data.drop('ID', axis=1, inplace=True)
```

Summary Statistics

Age

[ ]:
```
# Summary statistics for Age
age_summary = data['Age'].describe(percentiles=[0.25, 0.5, 0.75])
print("Summary Statistics for Age:")
print(age_summary)

# Histogram and density plot for Age
plt.style.use('default')
plt.figure(figsize=(10, 6))
sns.histplot(data['Age'], bins=30, kde=True, stat='density', color='gray',␣
 ↪edgecolor='black', alpha=0.8)
sns.kdeplot(data['Age'], color='black', linewidth=2)
plt.title('Age Distribution', fontsize=14, weight='bold')
plt.xlabel('Age', fontsize=12, labelpad=10)
plt.ylabel('Density', fontsize=12, labelpad=10)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()

# Create Age categories
data['Middle_Aged'] = ((data['Age'] > 30) & (data['Age'] <= 50)).astype(int)
data['Older'] = (data['Age'] > 50).astype(int)
```

Repayment Status

```
[ ]: # Bar plot of Repayment Statuses
repayment_status_vars = [
    'Repayment_Status_1', 'Repayment_Status_2', 'Repayment_Status_3',
    'Repayment_Status_4', 'Repayment_Status_5', 'Repayment_Status_6'
]

renamed_status_vars = {
    'Repayment_Status_1': 'Repayment Status 1',
    'Repayment_Status_2': 'Repayment Status 2',
    'Repayment_Status_3': 'Repayment Status 3',
    'Repayment_Status_4': 'Repayment Status 4',
    'Repayment_Status_5': 'Repayment Status 5',
    'Repayment_Status_6': 'Repayment Status 6'
}

repayment_status_counts = {renamed_status_vars[var]: data[var].
  ↪value_counts(sort=False) for var in repayment_status_vars}
repayment_status_df = pd.DataFrame(repayment_status_counts).fillna(0)
colors = ['#D3D3D3', '#A9A9A9', '#808080', '#696969', '#505050', '#303030']
ax = repayment_status_df.plot(kind='bar', figsize=(12, 8), width=0.8,␣
  ↪color=colors, edgecolor='black')
plt.title('Distribution of Repayment Status Across All Periods', fontsize=14,␣
  ↪weight='bold')
plt.xlabel('Repayment Status', fontsize=12, labelpad=10)
plt.ylabel('Count', fontsize=12, labelpad=10)
plt.legend(title='Period', loc='upper right', fontsize=10)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.savefig('Repaybar.png', dpi=600)
plt.show()

# Create Repayment Status categories
data['Mild_Delay'] = ((data[repayment_status_vars] == 1).any(axis=1) &
                      (data[repayment_status_vars] < 2).all(axis=1)).astype(int)
data['Severe_Delay'] = (data[repayment_status_vars] >= 2).any(axis=1).
  ↪astype(int)
data = data[(data['Mild_Delay'] + data['Severe_Delay']) <= 1]
```

Credit Limit

```
[ ]: # Summary statistics for Credit Limit
credit_limit_summary = data['Credit_Limit'].describe(percentiles=[0.25, 0.5, 0.
  ↪75])
print("Summary Statistics for Credit Limit:")
print(credit_limit_summary)
```

```
# Histogram and density plot for Credit Limit
plt.style.use('default')
plt.figure(figsize=(10, 6))
sns.histplot(data['Credit_Limit'], bins=30, kde=True, stat='density',␣
 ↪color='gray', edgecolor='black', alpha=0.8)
sns.kdeplot(data['Credit_Limit'], color='black', linewidth=2)
plt.title('Credit Limit Distribution', fontsize=14, weight='bold')
plt.xlabel('Credit Limit', fontsize=12, labelpad=10)
plt.ylabel('Density', fontsize=12, labelpad=10)
plt.grid(axis='y', linestyle='--', alpha=0.7)
plt.tight_layout()
plt.show()

# Create Credit Limit categories
data['Medium_Credit_Limit'] = ((data['Credit_Limit'] > 50000) &␣
 ↪(data['Credit_Limit'] <= 240000)).astype(int)
data['High_Credit_Limit'] = (data['Credit_Limit'] > 240000).astype(int)
```

Key Interactions

Create Interactions

```
[ ]: # Interaction terms for Repayment Status × Age
data['Mild_Delay × Middle_Aged'] = data['Mild_Delay'] * data['Middle_Aged']
data['Mild_Delay × Older'] = data['Mild_Delay'] * data['Older']
data['Severe_Delay × Middle_Aged'] = data['Severe_Delay'] * data['Middle_Aged']
data['Severe_Delay × Older'] = data['Severe_Delay'] * data['Older']

# Interaction terms for Repayment Status × Credit Limit
data['Medium_Credit_Limit × Mild_Delay'] = data['Medium_Credit_Limit'] *␣
 ↪data['Mild_Delay']
data['Medium_Credit_Limit × Severe_Delay'] = data['Medium_Credit_Limit'] *␣
 ↪data['Severe_Delay']
data['High_Credit_Limit × Mild_Delay'] = data['High_Credit_Limit'] *␣
 ↪data['Mild_Delay']
data['High_Credit_Limit × Severe_Delay'] = data['High_Credit_Limit'] *␣
 ↪data['Severe_Delay']

# Interaction terms for Education × Marital Status
data['Grad_School × Married'] = data['Grad_School'] * data['Married']
data['University × Married'] = data['University'] * data['Married']
```

Interaction Statistics

[ ]:

```
# Define combinations
repayment_age_combinations = {
    'Younger & On-Time': (data['Middle_Aged'] == 0) & (data['Older'] == 0) &
                         (data['Mild_Delay'] == 0) & (data['Severe_Delay'] ==␣
 ↪0),
    'Middle_Aged & On-Time': (data['Middle_Aged'] == 1) & (data['Older'] == 0) &
                             (data['Mild_Delay'] == 0) & (data['Severe_Delay']␣
 ↪== 0),
    'Older & On-Time': (data['Older'] == 1) &
                       (data['Mild_Delay'] == 0) & (data['Severe_Delay'] == 0),
    'Younger & Mild_Delay': (data['Middle_Aged'] == 0) & (data['Older'] == 0) &
                            (data['Mild_Delay'] == 1) & (data['Severe_Delay']␣
 ↪== 0),
    'Middle_Aged & Mild_Delay': (data['Middle_Aged'] == 1) & (data['Older'] ==␣
 ↪0) &
                                (data['Mild_Delay'] == 1) &␣
 ↪(data['Severe_Delay'] == 0),
    'Older & Mild_Delay': (data['Older'] == 1) &
                          (data['Mild_Delay'] == 1) & (data['Severe_Delay'] ==␣
 ↪0),
    'Younger & Severe_Delay': (data['Middle_Aged'] == 0) & (data['Older'] == 0)␣
 ↪&
                              (data['Mild_Delay'] == 0) & (data['Severe_Delay']␣
 ↪== 1),
    'Middle_Aged & Severe_Delay': (data['Middle_Aged'] == 1) & (data['Older']␣
 ↪== 0) &
                                  (data['Mild_Delay'] == 0) &␣
 ↪(data['Severe_Delay'] == 1),
    'Older & Severe_Delay': (data['Older'] == 1) &
                            (data['Mild_Delay'] == 0) & (data['Severe_Delay']␣
 ↪== 1)
}

credit_limit_repayment_combinations = {
    'Low Credit & On-Time': (data['Medium_Credit_Limit'] == 0) &␣
 ↪(data['High_Credit_Limit'] == 0) &
                            (data['Mild_Delay'] == 0) & (data['Severe_Delay']␣
 ↪== 0),
    'Medium Credit & On-Time': (data['Medium_Credit_Limit'] == 1) &␣
 ↪(data['Mild_Delay'] == 0) &
                               (data['Severe_Delay'] == 0),
    'High Credit & On-Time': (data['High_Credit_Limit'] == 1) &␣
 ↪(data['Mild_Delay'] == 0) &
                             (data['Severe_Delay'] == 0),
    'Low Credit & Mild_Delay': (data['Medium_Credit_Limit'] == 0) &␣
 ↪(data['High_Credit_Limit'] == 0) &
                                (data['Mild_Delay'] == 1),
    'Medium Credit & Mild_Delay': (data['Medium_Credit_Limit'] == 1) &␣
 ↪(data['Mild_Delay'] == 1),
    'High Credit & Mild_Delay': (data['High_Credit_Limit'] == 1) &␣
 ↪(data['Mild_Delay'] == 1),
    'Low Credit & Severe_Delay': (data['Medium_Credit_Limit'] == 0) &␣
 ↪(data['High_Credit_Limit'] == 0) &
                                  (data['Severe_Delay'] == 1),
    'Medium Credit & Severe_Delay': (data['Medium_Credit_Limit'] == 1) &␣
 ↪(data['Severe_Delay'] == 1),
    'High Credit & Severe_Delay': (data['High_Credit_Limit'] == 1) &␣
 ↪(data['Severe_Delay'] == 1)
}

education_marital_combinations = {
    'High School & Single': (data['Grad_School'] == 0) & (data['University'] ==␣
 ↪0) & (data['Married'] == 0),
    'Grad School & Single': (data['Grad_School'] == 1) & (data['Married'] == 0),
    'University & Single': (data['University'] == 1) & (data['Married'] == 0),
    'High School & Married': (data['Grad_School'] == 0) & (data['University']␣
 ↪== 0) & (data['Married'] == 1),
    'Grad School & Married': (data['Grad_School'] == 1) & (data['Married'] ==␣
 ↪1),
    'University & Married': (data['University'] == 1) & (data['Married'] == 1)
}

# Summary table generation
def calculate_combinations(combinations, total_count):
    stats = {k: v.sum() for k, v in combinations.items()}
    percentages = {k: (v / total_count) * 100 for k, v in stats.items()}
    return pd.DataFrame({'Count': stats, 'Percentage (%)': percentages})

total_count = len(data)
repayment_age_table = calculate_combinations(repayment_age_combinations,␣
 ↪total_count)
credit_limit_repayment_table =␣
 ↪calculate_combinations(credit_limit_repayment_combinations, total_count)
education_marital_table =␣
 ↪calculate_combinations(education_marital_combinations, total_count)

print("Repayment Status × Age Combinations:")
print(repayment_age_table)
print("\nCredit Limit × Repayment Status Combinations:")
print(credit_limit_repayment_table)
print("\nEducation × Marital Status Combinations:")
print(education_marital_table)

# Default vs. Non-Default proportions
def calculate_default_proportions(combinations):
    proportions = {}
    for label, condition in combinations.items():
        subset = data[condition]
        default_counts = subset['Default_Payment'].value_counts(normalize=True)␣
 ↪* 100
        proportions[label] = default_counts.reindex([0, 1], fill_value=0)
    return pd.DataFrame(proportions).T.rename(columns={0: 'Non-Default (%)', 1:␣
 ↪'Default (%)'})

repayment_age_default_proportions =␣
 ↪calculate_default_proportions(repayment_age_combinations)
credit_limit_repayment_default_proportions =␣
 ↪calculate_default_proportions(credit_limit_repayment_combinations)
education_marital_default_proportions =␣
 ↪calculate_default_proportions(education_marital_combinations)

print("\nProportion of Default vs Non-Default for Repayment Status × Age␣
 ↪Combinations:")
print(repayment_age_default_proportions)
print("\nProportion of Default vs Non-Default for Credit Limit × Repayment␣
 ↪Status Combinations:")
print(credit_limit_repayment_default_proportions)
print("\nProportion of Default vs Non-Default for Education × Marital Status␣
 ↪Combinations:")
print(education_marital_default_proportions)
```

General Summary Statistics

[ ]:
```
# Calculate base cases
base_cases = {
    "Young (<30)": len(data.query("Middle_Aged == 0 & Older == 0")),
    "On Time Payment": len(data.query("Mild_Delay == 0 & Severe_Delay == 0")),
    "Low Credit Limit": len(data.query("Medium_Credit_Limit == 0 &␣
 ↪High_Credit_Limit == 0")),
    "High School": len(data[(data["Grad_School"] == 0) & (data["University"] ==␣
 ↪0)]),
    "Not Married": data["Married"].value_counts().get(0, 0),
}

# Variables of interest
categorical_vars = [
```

```python
    "Default_Payment", "Grad_School", "University", "Female", "Married",
    "Middle_Aged", "Older", "Mild_Delay", "Severe_Delay",
    "Medium_Credit_Limit", "High_Credit_Limit"
]

# Initialize summary dictionary
summary = {"Variable": [], "Category": [], "Count": [], "% of Sample": []}

# Summarize categorical variables
for var in categorical_vars:
    for category in data[var].unique():
        count = len(data[data[var] == category])
        pct = (count / len(data)) * 100
        summary["Variable"].append(var)
        summary["Category"].append(f"{var} = {category}")
        summary["Count"].append(count)
        summary["% of Sample"].append(round(pct, 2))

# Add base cases
for base_case, count in base_cases.items():
    summary["Variable"].append(base_case)
    summary["Category"].append("Base Case")
    summary["Count"].append(count)
    summary["% of Sample"].append(round((count / len(data)) * 100, 2))

# Add stats for Default_Payment
default_1_count = len(data[data["Default_Payment"] == 1])
default_0_count = len(data[data["Default_Payment"] == 0])
summary["Variable"].extend(["Default_Payment", "Default_Payment"])
summary["Category"].extend(["Default = 1", "Not Default = 0"])
summary["Count"].extend([default_1_count, default_0_count])
summary["% of Sample"].extend([
    round((default_1_count / len(data)) * 100, 2),
    round((default_0_count / len(data)) * 100, 2)
])

# Convert to DataFrame and display
summary_df = pd.DataFrame(summary)
print(summary_df)
```

X-Learners

Data Preparation

```
[ ]: # Remove specified interaction terms
interaction_terms_to_remove = [
    'Mild_Delay × Middle_Aged', 'Mild_Delay × Older', 'Severe_Delay ×␣
 ↪Middle_Aged',
    'Severe_Delay × Older', 'Medium_Credit_Limit × Mild_Delay',
    'Medium_Credit_Limit × Severe_Delay', 'High_Credit_Limit × Mild_Delay',
    'High_Credit_Limit × Severe_Delay', 'Grad_School × Married',
    'University × Married'
]
data = data.drop(columns=interaction_terms_to_remove, errors='ignore')

# Define features and target variable
X = data.drop(columns=['Default_Payment'])
y = data['Default_Payment']
```

Repayment Status × Age Interaction

```
[ ]: # Define subgroups for Repayment Status × Age
interaction_groups = {
    'Younger & On-Time': (X['Middle_Aged'] == 0) & (X['Older'] == 0) &
                         (X['Mild_Delay'] == 0) & (X['Severe_Delay'] == 0),
    'Middle_Aged & On-Time': (X['Middle_Aged'] == 1) & (X['Older'] == 0) &
                             (X['Mild_Delay'] == 0) & (X['Severe_Delay'] == 0),
    'Older & On-Time': (X['Older'] == 1) &
                       (X['Mild_Delay'] == 0) & (X['Severe_Delay'] == 0),
    'Younger & Mild_Delay': (X['Middle_Aged'] == 0) & (X['Older'] == 0) &
                            (X['Mild_Delay'] == 1) & (X['Severe_Delay'] == 0),
    'Middle_Aged & Mild_Delay': (X['Middle_Aged'] == 1) & (X['Older'] == 0) &
                                (X['Mild_Delay'] == 1) & (X['Severe_Delay'] ==␣
 ↪0),
    'Older & Mild_Delay': (X['Older'] == 1) &
                          (X['Mild_Delay'] == 1) & (X['Severe_Delay'] == 0),
    'Younger & Severe_Delay': (X['Middle_Aged'] == 0) & (X['Older'] == 0) &
                              (X['Mild_Delay'] == 0) & (X['Severe_Delay'] == 1),
    'Middle_Aged & Severe_Delay': (X['Middle_Aged'] == 1) & (X['Older'] == 0) &
                                  (X['Mild_Delay'] == 0) & (X['Severe_Delay']␣
 ↪== 1),
    'Older & Severe_Delay': (X['Older'] == 1) &
                            (X['Mild_Delay'] == 0) & (X['Severe_Delay'] == 1)
}

# Dictionary to store final CATE results and confidence intervals
cate_results = {}
```

```python
# CV function
def xgb_cv_optimization(X, y, params, model_type, nfold=5, verbose_eval=False):
    dmatrix = DMatrix(data=X, label=y)

    # Determine metrics and stratification
    if model_type == 'classifier':
        metrics = 'auc'
        stratified = True
    else:
        metrics = 'rmse'
        stratified = False

    # Perform CV
    cv_results = cv(
        params=params,
        dtrain=dmatrix,
        num_boost_round=200,
        nfold=nfold,
        metrics=metrics,
        stratified=stratified,
        early_stopping_rounds=10,
        seed=42,
        verbose_eval=verbose_eval
    )
    return len(cv_results)

# Bootstrapping function
def bootstrap_cate(X, y, treat, num_bootstrap):
    bootstrap_cates = []
    for i in range(num_bootstrap):
        X_resampled, y_resampled, treat_resampled = resample(X, y, treat,␣
 ↪random_state=i)

        # Fit models for bootstrap sample
        propensity_model.fit(X_resampled, treat_resampled)
        propensity_scores = propensity_model.predict_proba(X_resampled)[:, 1]

        treated_indices = treat_resampled == 1
        control_indices = treat_resampled == 0

        treated_model.fit(X_resampled[treated_indices],␣
 ↪y_resampled[treated_indices])
        control_model.fit(X_resampled[control_indices],␣
 ↪y_resampled[control_indices])

        treated_preds = treated_model.predict(X)
        control_preds = control_model.predict(X)
        treatment_effects = treated_preds - control_preds

        # Combine effects
        refined_treated_model.fit(X_resampled[treated_indices],␣
 ↪treatment_effects[treated_indices])
        refined_control_model.fit(X_resampled[control_indices],␣
 ↪treatment_effects[control_indices])

        refined_treated_preds = refined_treated_model.predict(X)
        refined_control_preds = refined_control_model.predict(X)

        cate_combined = (propensity_scores * refined_treated_preds +
                         (1 - propensity_scores) * refined_control_preds)
        bootstrap_cates.append(cate_combined.mean())
    return np.array(bootstrap_cates)

# Loop through each group
for group_name, condition in interaction_groups.items():
    print(f"Processing group: {group_name}")

    # Define treatment variable
    treat = condition.astype(int)

    # Step 1: Propensity Score Estimation with XGBClassifier
    propensity_params = {
        'objective': 'binary:logistic',
        'learning_rate': 0.1,
        'max_depth': 5,
        'subsample': 0.8,
        'colsample_bytree': 0.8,
        'eval_metric': 'auc',
        'seed': 42
    }
    best_num_boost_rounds = xgb_cv_optimization(X, treat, propensity_params,␣
 ↪model_type='classifier', verbose_eval=False)
    propensity_model = XGBClassifier(
        n_estimators=best_num_boost_rounds,
        random_state=42,
        **propensity_params
    )
    propensity_model.fit(X, treat)
    propensity_scores = propensity_model.predict_proba(X)[:, 1]

    # Step 2: Fit Conditional Outcome Models
    treated_indices = X[treat == 1].index
    control_indices = X[treat == 0].index
    treated_outcome = y.loc[treated_indices]
    control_outcome = y.loc[control_indices]

    treated_params = {
        'objective': 'reg:squarederror',
        'learning_rate': 0.1,
        'max_depth': 5,
        'subsample': 0.8,
        'colsample_bytree': 0.8,
        'eval_metric': 'rmse',
        'seed': 42
    }
    best_num_boost_rounds_treated = xgb_cv_optimization(
        X.loc[treated_indices], treated_outcome, treated_params,␣
↪model_type='regressor', verbose_eval=False
    )
    treated_model = XGBRegressor(
        n_estimators=best_num_boost_rounds_treated,
        random_state=42,
        **treated_params
    )
    treated_model.fit(X.loc[treated_indices], treated_outcome)

    best_num_boost_rounds_control = xgb_cv_optimization(
        X.loc[control_indices], control_outcome, treated_params,␣
↪model_type='regressor', verbose_eval=False
    )
    control_model = XGBRegressor(
        n_estimators=best_num_boost_rounds_control,
        random_state=42,
        **treated_params
    )
    control_model.fit(X.loc[control_indices], control_outcome)

    treated_preds = treated_model.predict(X)
    control_preds = control_model.predict(X)
    treatment_effects = treated_preds - control_preds

    refined_treated_model = XGBRegressor(**treated_params)
    refined_control_model = XGBRegressor(**treated_params)

    refined_treated_model.fit(X[treat == 1], treatment_effects[treat == 1])
    refined_control_model.fit(X[treat == 0], treatment_effects[treat == 0])

    refined_treated_preds = refined_treated_model.predict(X)
    refined_control_preds = refined_control_model.predict(X)
    cate_combined = (propensity_scores * refined_treated_preds +
                     (1 - propensity_scores) * refined_control_preds)

    # Bootstrap CATE
    bootstrap_cates = bootstrap_cate(X, y, treat, num_bootstrap=1000)
    cate_mean = cate_combined.mean()
    cate_ci_lower = np.percentile(bootstrap_cates, 2.5)
    cate_ci_upper = np.percentile(bootstrap_cates, 97.5)

    # Store results
    cate_results[group_name] = {
        'CATE Mean': cate_mean,
        'CI Lower': cate_ci_lower,
        'CI Upper': cate_ci_upper
    }

# Convert results to a DataFrame
cate_df = pd.DataFrame.from_dict(cate_results, orient='index')
print(cate_df)
```

Repayment Status × Credit Limit Interaction

```
[ ]: # Define subgroups for Credit Limit × Repayment Status
interaction_groups = {
    'Low Credit & On-Time': (X['Medium_Credit_Limit'] == 0) &␣
 ↪(X['High_Credit_Limit'] == 0) &
                            (X['Mild_Delay'] == 0) & (X['Severe_Delay'] == 0),
    'Medium Credit & On-Time': (X['Medium_Credit_Limit'] == 1) &␣
 ↪(X['Mild_Delay'] == 0) &
                               (X['Severe_Delay'] == 0),
    'High Credit & On-Time': (X['High_Credit_Limit'] == 1) & (X['Mild_Delay']␣
 ↪== 0) &
                             (X['Severe_Delay'] == 0),
    'Low Credit & Mild_Delay': (X['Medium_Credit_Limit'] == 0) &␣
 ↪(X['High_Credit_Limit'] == 0) &
                               (X['Mild_Delay'] == 1),
    'Medium Credit & Mild_Delay': (X['Medium_Credit_Limit'] == 1) &␣
 ↪(X['Mild_Delay'] == 1),
    'High Credit & Mild_Delay': (X['High_Credit_Limit'] == 1) &␣
 ↪(X['Mild_Delay'] == 1),
    'Low Credit & Severe_Delay': (X['Medium_Credit_Limit'] == 0) &␣
 ↪(X['High_Credit_Limit'] == 0) &
                                 (X['Severe_Delay'] == 1),
    'Medium Credit & Severe_Delay': (X['Medium_Credit_Limit'] == 1) &␣
 ↪(X['Severe_Delay'] == 1),
    'High Credit & Severe_Delay': (X['High_Credit_Limit'] == 1) &␣
 ↪(X['Severe_Delay'] == 1)
}

# Dictionary to store final CATE results and confidence intervals
cate_results = {}

# CV function
def xgb_cv_optimization(X, y, params, model_type, nfold=5, verbose_eval=False):
    dmatrix = DMatrix(data=X, label=y)

    # Determine metrics and stratification
    if model_type == 'classifier':
        metrics = 'auc'
        stratified = True
    else:
        metrics = 'rmse'
        stratified = False

    # Perform CV
    cv_results = cv(
        params=params,
        dtrain=dmatrix,
        num_boost_round=200,
        nfold=nfold,
        metrics=metrics,
        stratified=stratified,
        early_stopping_rounds=10,
        seed=42,
        verbose_eval=verbose_eval
    )
    return len(cv_results)

# Bootstrapping function
def bootstrap_cate(X, y, treat, num_bootstrap):
    bootstrap_cates = []
    for i in range(num_bootstrap):
        X_resampled, y_resampled, treat_resampled = resample(X, y, treat,␣
 ↪random_state=i)

        # Fit models for bootstrap sample
        propensity_model.fit(X_resampled, treat_resampled)
        propensity_scores = propensity_model.predict_proba(X_resampled)[:, 1]

        treated_indices = treat_resampled == 1
        control_indices = treat_resampled == 0
        treated_model.fit(X_resampled[treated_indices],␣
 ↪y_resampled[treated_indices])
        control_model.fit(X_resampled[control_indices],␣
 ↪y_resampled[control_indices])

        treated_preds = treated_model.predict(X)
        control_preds = control_model.predict(X)
        treatment_effects = treated_preds - control_preds

        # Combine effects
        refined_treated_model.fit(X_resampled[treated_indices],␣
 ↪treatment_effects[treated_indices])
        refined_control_model.fit(X_resampled[control_indices],␣
 ↪treatment_effects[control_indices])

        refined_treated_preds = refined_treated_model.predict(X)
        refined_control_preds = refined_control_model.predict(X)

        cate_combined = (propensity_scores * refined_treated_preds +
                         (1 - propensity_scores) * refined_control_preds)
        bootstrap_cates.append(cate_combined.mean())
    return np.array(bootstrap_cates)

# Loop through each group
for group_name, condition in interaction_groups.items():
    print(f"Processing group: {group_name}")

    # Define treatment variable
    treat = condition.astype(int)

    # Step 1: Propensity Score Estimation with XGBClassifier
    propensity_params = {
        'objective': 'binary:logistic',
        'learning_rate': 0.1,
        'max_depth': 5,
        'subsample': 0.8,
        'colsample_bytree': 0.8,
        'eval_metric': 'auc',
        'seed': 42
    }
    best_num_boost_rounds = xgb_cv_optimization(X, treat, propensity_params,␣
 ↪model_type='classifier', verbose_eval=False)
    propensity_model = XGBClassifier(
        n_estimators=best_num_boost_rounds,
        random_state=42,
        **propensity_params
    )
    propensity_model.fit(X, treat)
    propensity_scores = propensity_model.predict_proba(X)[:, 1]

    # Step 2: Fit Conditional Outcome Models
    treated_indices = X[treat == 1].index
    control_indices = X[treat == 0].index

    treated_outcome = y.loc[treated_indices]
    control_outcome = y.loc[control_indices]

    treated_params = {
        'objective': 'reg:squarederror',
        'learning_rate': 0.1,
        'max_depth': 5,
        'subsample': 0.8,
        'colsample_bytree': 0.8,
        'eval_metric': 'rmse',
        'seed': 42
    }
    best_num_boost_rounds_treated = xgb_cv_optimization(
        X.loc[treated_indices], treated_outcome, treated_params,␣
↪model_type='regressor', verbose_eval=False
    )
    treated_model = XGBRegressor(
        n_estimators=best_num_boost_rounds_treated,
        random_state=42,
        **treated_params
    )
    treated_model.fit(X.loc[treated_indices], treated_outcome)

    best_num_boost_rounds_control = xgb_cv_optimization(
        X.loc[control_indices], control_outcome, treated_params,␣
↪model_type='regressor', verbose_eval=False
    )
    control_model = XGBRegressor(
        n_estimators=best_num_boost_rounds_control,
        random_state=42,
        **treated_params
    )
    control_model.fit(X.loc[control_indices], control_outcome)

    treated_preds = treated_model.predict(X)
    control_preds = control_model.predict(X)
    treatment_effects = treated_preds - control_preds

    refined_treated_model = XGBRegressor(**treated_params)
    refined_control_model = XGBRegressor(**treated_params)
    refined_treated_model.fit(X[treat == 1], treatment_effects[treat == 1])
    refined_control_model.fit(X[treat == 0], treatment_effects[treat == 0])

    refined_treated_preds = refined_treated_model.predict(X)
    refined_control_preds = refined_control_model.predict(X)

    cate_combined = (propensity_scores * refined_treated_preds +
                     (1 - propensity_scores) * refined_control_preds)

    # Bootstrap CATE
    bootstrap_cates = bootstrap_cate(X, y, treat, num_bootstrap=1000)
    cate_mean = cate_combined.mean()
    cate_ci_lower = np.percentile(bootstrap_cates, 2.5)
    cate_ci_upper = np.percentile(bootstrap_cates, 97.5)

    # Store results
    cate_results[group_name] = {
        'CATE Mean': cate_mean,
        'CI Lower': cate_ci_lower,
        'CI Upper': cate_ci_upper
    }

# Convert results to a DataFrame
cate_df = pd.DataFrame.from_dict(cate_results, orient='index')
print(cate_df)
```

Education × Martial Status Interaction

```
[ ]: # Define subgroups for Education × Marital Status
interaction_groups = {
    'High School & Single': (X['Grad_School'] == 0) & (X['University'] == 0) &␣
 ↪(X['Married'] == 0),
    'Grad School & Single': (X['Grad_School'] == 1) & (X['Married'] == 0),
    'University & Single': (X['University'] == 1) & (X['Married'] == 0),
    'High School & Married': (X['Grad_School'] == 0) & (X['University'] == 0) &␣
 ↪(X['Married'] == 1),
    'Grad School & Married': (X['Grad_School'] == 1) & (X['Married'] == 1),
    'University & Married': (X['University'] == 1) & (X['Married'] == 1)
}

# Dictionary to store final CATE results and confidence intervals
cate_results = {}

# CV function
def xgb_cv_optimization(X, y, params, model_type, nfold=5, verbose_eval=False):
    dmatrix = DMatrix(data=X, label=y)
    # Determine metrics and stratification
    if model_type == 'classifier':
        metrics = 'auc'
        stratified = True
    else:
        metrics = 'rmse'
        stratified = False

    # Perform CV
    cv_results = cv(
        params=params,
        dtrain=dmatrix,
        num_boost_round=200,
        nfold=nfold,
        metrics=metrics,
        stratified=stratified,
        early_stopping_rounds=10,
        seed=42,
        verbose_eval=verbose_eval
    )
    return len(cv_results)

# Bootstrapping function
def bootstrap_cate(X, y, treat, num_bootstrap):
    bootstrap_cates = []
    for i in range(num_bootstrap):
        X_resampled, y_resampled, treat_resampled = resample(X, y, treat,␣
 ↪random_state=i)

        # Fit models for bootstrap sample
        propensity_model.fit(X_resampled, treat_resampled)
        propensity_scores = propensity_model.predict_proba(X_resampled)[:, 1]

        treated_indices = treat_resampled == 1
        control_indices = treat_resampled == 0

        treated_model.fit(X_resampled[treated_indices],␣
 ↪y_resampled[treated_indices])
        control_model.fit(X_resampled[control_indices],␣
 ↪y_resampled[control_indices])

        treated_preds = treated_model.predict(X)
        control_preds = control_model.predict(X)
        treatment_effects = treated_preds - control_preds

        # Combine effects
        refined_treated_model.fit(X_resampled[treated_indices],␣
↪treatment_effects[treated_indices])
        refined_control_model.fit(X_resampled[control_indices],␣
↪treatment_effects[control_indices])

        refined_treated_preds = refined_treated_model.predict(X)
        refined_control_preds = refined_control_model.predict(X)

        cate_combined = (propensity_scores * refined_treated_preds +
                         (1 - propensity_scores) * refined_control_preds)
        bootstrap_cates.append(cate_combined.mean())
    return np.array(bootstrap_cates)

# Loop through each group
for group_name, condition in interaction_groups.items():
    print(f"Processing group: {group_name}")

    # Define treatment variable
    treat = condition.astype(int)

    # Step 1: Propensity Score Estimation with XGBClassifier
    propensity_params = {
        'objective': 'binary:logistic',
        'learning_rate': 0.1,
        'max_depth': 5,
        'subsample': 0.8,
        'colsample_bytree': 0.8,
        'eval_metric': 'auc',
        'seed': 42
    }
    best_num_boost_rounds = xgb_cv_optimization(X, treat, propensity_params,␣
↪model_type='classifier', verbose_eval=False)
    propensity_model = XGBClassifier(
        n_estimators=best_num_boost_rounds,
        random_state=42,
        **propensity_params
    )
    propensity_model.fit(X, treat)
    propensity_scores = propensity_model.predict_proba(X)[:, 1]

    # Step 2: Fit Conditional Outcome Models
    treated_indices = X[treat == 1].index
    control_indices = X[treat == 0].index

    treated_outcome = y.loc[treated_indices]
    control_outcome = y.loc[control_indices]
    treated_params = {
        'objective': 'reg:squarederror',
        'learning_rate': 0.1,
        'max_depth': 5,
        'subsample': 0.8,
        'colsample_bytree': 0.8,
        'eval_metric': 'rmse',
        'seed': 42
    }
    best_num_boost_rounds_treated = xgb_cv_optimization(
        X.loc[treated_indices], treated_outcome, treated_params,␣
↪model_type='regressor', verbose_eval=False
    )
    treated_model = XGBRegressor(
        n_estimators=best_num_boost_rounds_treated,
        random_state=42,
        **treated_params
    )
    treated_model.fit(X.loc[treated_indices], treated_outcome)

    best_num_boost_rounds_control = xgb_cv_optimization(
        X.loc[control_indices], control_outcome, treated_params,␣
↪model_type='regressor', verbose_eval=False
    )
    control_model = XGBRegressor(
        n_estimators=best_num_boost_rounds_control,
        random_state=42,
        **treated_params
    )
    control_model.fit(X.loc[control_indices], control_outcome)

    treated_preds = treated_model.predict(X)
    control_preds = control_model.predict(X)
    treatment_effects = treated_preds - control_preds

    refined_treated_model = XGBRegressor(**treated_params)
    refined_control_model = XGBRegressor(**treated_params)

    refined_treated_model.fit(X[treat == 1], treatment_effects[treat == 1])
    refined_control_model.fit(X[treat == 0], treatment_effects[treat == 0])

    refined_treated_preds = refined_treated_model.predict(X)
    refined_control_preds = refined_control_model.predict(X)

    cate_combined = (propensity_scores * refined_treated_preds +
                     (1 - propensity_scores) * refined_control_preds)
    # Bootstrap CATE
    bootstrap_cates = bootstrap_cate(X, y, treat, num_bootstrap=1000)
    cate_mean = cate_combined.mean()
    cate_ci_lower = np.percentile(bootstrap_cates, 2.5)
    cate_ci_upper = np.percentile(bootstrap_cates, 97.5)

    # Store results
    cate_results[group_name] = {
        'CATE Mean': cate_mean,
        'CI Lower': cate_ci_lower,
        'CI Upper': cate_ci_upper
    }

# Convert results to a DataFrame
cate_df = pd.DataFrame.from_dict(cate_results, orient='index')
print(cate_df)
```