A close up of a logo

Description automatically generated

**George Washington University**

**Aman Jaglan  
Instructor: Dr. Reja Jafari**

**25 April 2024**

**Visualization of complex Data**

**DATS 6401**

**Table of content**

|  |
| --- |
| 1. Abstract |
| 2. Introduction |
| 3. Description of the Dataset |
| 4. Pre-processing Dataset |
| 5. Outlier Detection & Removal |
| 6. Principal Component Analysis (PCA) |
| 7. Normality Test |
| 8. Heatmap & Pearson Correlation Coefficient Matrix |
| 9. Statistics |
| 10. Data Analysis and Visualization Insights |
| 11. Conclusion |
| 12. Appendix |
| 13. References |

**Abstract:**

This report presents a comprehensive analysis of American Express customer data with a focus on spend behavior, payment consistency, and delinquency patterns across different account types. By examining key variables such as checking and savings account balances (B\_1 and B\_2), payment behavior (P\_2), spend behavior (S\_3), and delinquency (D\_39), this study aims to identify correlations that influence financial risk and customer behavior dynamics. The analysis employs statistical methods to explore the distribution and correlation of account balances, payments, and spend tendencies, highlighting how these factors contribute to delinquency and risk assessments. Additionally, the report delves into specific behavior patterns, such as the right-skewed distribution of spending and central clustering of balances and payments, to offer insights into effective risk management strategies. Through a detailed examination of the financial behaviors of American Express customers, this report provides essential findings that can assist in enhancing risk assessment models and improving customer relationship management strategies. The goal is to ensure more tailored financial services that meet the needs of diverse customer profiles, ultimately fostering a safer and more efficient banking environment.

**Introduction:**  
In the competitive landscape of financial services, understanding customer behavior is pivotal for optimizing risk management and enhancing customer satisfaction. American Express, a global leader in this industry, continuously seeks deeper insights into its customer base to tailor its services effectively. This report details an analytical project aimed at dissecting the financial behaviors of American Express customers through a multi-dimensional examination of their spend patterns, payment behaviors, and account balances.

The primary data set analyzed comprises several key variables, including Balance Type 1 (Checking Account - B\_1), Balance Type 2 (Savings Account - B\_2), Payment Behavior (P\_2), Spend Behavior (S\_3), and Delinquency Behavior (D\_39), alongside a variable assessing the overall risk factor (R\_1). This study specifically focuses on exploring how these variables interact and what they reveal about the financial health and behavioral trends of the customer base.

The motivation behind this analysis stems from the necessity to mitigate financial risks and predict delinquent behaviors that could adversely affect both the customer and the company. By identifying and understanding the underlying patterns in customer transactions and account management, American Express can implement more robust risk assessment protocols and develop targeted strategies to prevent future delinquencies.

Furthermore, this project aligns with broader corporate goals of achieving greater customer insight and operational efficiency. By leveraging statistical analyses and data visualization techniques, this report provides a detailed narrative and quantitative analysis that aims to contribute to strategic planning and decision-making processes at American Express.

The following sections of the report will delve into the goals of the analysis, the methodologies employed, detailed findings, and the implications of these findings on risk management and customer relationship strategies.

**Description of the Dataset:**

**S\_3: Spend Behavior** - Average spending behavior with a mean of 0.225 and notable variation, ranging from -0.255 to 2.919.

**P\_2: Payment Behavior** - Represents payment consistency with a mean of 0.635, with values stretching **from -0.383 to 1.010.**

**D\_39: Delinquency Behavior** - Indicates delinquency levels with a mean of 0.188, varying significantly across accounts (minimum near zero, maximum of 5.009).

**B\_1: Balance Type 1 (Checking Account)** - Shows checking account balances with an average of 0.143, ranging from -0.295 to 1.324.

**B\_2: Balance Type 2 (Savings Account)** - Details savings account balances with a mean of 0.590, and a range from near zero to 1.010.

**R\_1: Risk Factor** - Measures risk with a mean of 0.093, spanning from nearly zero to 2.508.

**B\_30: Different Types of Balances** - Captures various balance types with most values at zero, extending up to a maximum of 2.0.

**D\_114 and D\_116: Delinquency Indicators** - Binary indicators for specific delinquency behaviors, showing significant categorical differentiation.

A screenshot of a computer screen

Description automatically generated

**A screenshot of a graph

Description automatically generated**

**Pre-processing Dataset:**

**Outlier Detection & Removal:**

**A graph with a line

Description automatically generated**

**A blue rectangle with white text

Description automatically generated**

The two boxplots represent the distribution of a 'Spend Variable' across three categories before and after outlier detection and removal.

**Initial Boxplot (Before Outlier Removal):**

* The initial plot displays a wider range of the 'Spend Variable', including several extreme values which are noticeable as outliers (data points that are significantly higher than the main cluster of the data).
* Categories 0 and 2 show a high presence of outliers that extend the range of the variable considerably, indicating significant variability within these groups.
* Category 1 also contains outliers, but they are less pronounced when compared to the other two categories.

**Adjusted Boxplot (After Outlier Removal):**

* The adjusted plot shows a tighter distribution of the 'Spend Variable', with a significantly reduced range and fewer outliers, indicating that extreme values have been removed.
* The median and interquartile ranges (IQR) are more clearly defined, suggesting a more robust representation of the central tendency and variability within each category.
* The visualization post-outlier removal presents a cleaner, more standardized view of the data, which may be more representative of the "normal" spending behavior across the categories.

**Principal Component Analysis (PCA)**

**A screenshot of a computer screen

Description automatically generated**

* Principal Component Analysis (PCA) is a dimensionality reduction technique that transforms high-dimensional data into a lower-dimensional form, which retains most of the variability present in the original data. This is done by finding new, uncorrelated variables called principal components (PCs), each being a linear combination of the original variables.
* Scree Plot: This plot shows the proportion of variance explained by each principal component. It's commonly used to determine the number of components to retain. In your plot, PC1 accounts for the largest variance in the data (41.76%), and each subsequent component explains progressively less variance, with PC6 explaining the least (5.28%). Typically, one might choose to keep components that contribute significantly to the variance.
* PCA Component Plot: This plot maps data points onto the space defined by the first two principal components. This is a scatter plot where each point's position is determined by its value for the first principal component (horizontal axis) and the second principal component (vertical axis). It's a way to visualize the spread and clustering of the data in this reduced-dimensional space.
* The distribution along the first principal component suggests that it captures a strong gradient in the data, while the distribution along the second principal component may capture another, less pronounced variation.

**Normality Test:**

**A graph showing the value of a graph

Description automatically generated**

* The image shows a Q-Q (Quantile-Quantile) plot which is a graphical technique for determining if two data sets come from populations with a common distribution. A 45-degree reference line is also plotted. Here's what the Q-Q plot and the accompanying test results indicate:
* Deviations from Normality: The data points significantly deviate from the red fit line, especially at the tails of the distribution, suggesting that the data does not follow a normal distribution.
* Statistical Test Confirmations: The Shapiro-Wilk and Kolmogorov-Smirnov tests provide statistical confirmation of the visual evidence. The very low p-values (0.0000) from both tests strongly reject the null hypothesis of normality.
* Implications for Data Analysis: Non-normality in the data might necessitate the use of non-parametric methods for statistical analysis or data transformation techniques if parametric methods are to be applied.

**Heatmap & Pearson Correlation Coefficient Matrix**

**A screenshot of a graph

Description automatically generated**

* Correlation Strengths: The colors vary from blue to red, where deep blue indicates a strong negative correlation, deep red signifies a strong positive correlation, and lighter colors or white represent weaker correlations. The values on the heatmap provide the exact correlation coefficients.
* Highlighted Relationship: The notation at the bottom indicates that the maximum correlation is between B\_1 and B\_2, which, by observing the heatmap, shows a strong positive relationship (0.55).

**Statistics:**

**A graph with a blue line

Description automatically generated**This is the statistics tab from my dashboard where we can test the statistics and density estimates of all the columns. This is just an example of B\_1(balance type one).

**Data Analysis and Visualization Insights**

|  |  |
| --- | --- |
| Plots | Description |
| o Line-plot | Tracking Payment Consistency Across Balance Categories |
| o Bar plot: stack, group | Composition of Payments in Balance Categories  Analysis of Payment Components by Delinquency Categories |
| o Count plot | Comparison of Delinquency Status Across Two Metrics |
| o Pie chart | Proportional Distribution of Balance Categories |
| o Dist plot | Visualizing Risk Across Balance Tiers |
| o Pair plot | Multivariate Relationships in Financial Data |
| o Heatmap with cbar | Interconnected Financial Indicators Revealed by Correlation |
| o Histogram plot with KDE | Skewness in Spending Behavior |
| o QQ-plot | Assessing Normality in Financial Data Using Q-Q Plots |
| o KDE plot | Distribution Insights of Financial Variables via Kernel Density Estimation |
| o Im or reg plot | Exploring the Dynamics Between Account Balance and Payment Behavior |
| o Multivariate Box or Boxen plot | Financial Behavior Insights Through Boxplots.  Spending Patterns Across Different Balance Categories |
| o Area plot | Monthly Expenditure Trends by Balance Categories |
| o Violin plot | Risk Profile Variations by Balance Category |
| o Joint plot with KDE and scatter representation | Joint Distribution of Balance and Payment |
| o Rug plot | Concentration of Log-Transformed Account Balances |
| o 3D plot and contour plot | Multidimensional Financial Dynamics.  Contour Insights into Spending Versus Balance. |
| o Cluster map | Correlation and Clustering in Delinquency Analysis |
| o Hexbin | Concentration of Financial Behaviors with Hexagonal Binning |
| o Strip plot | Spending Behavior Visualized Across Balance Categories |
| o Swarm plot | Visualizing Risk Across Balance Tiers |

**Static Graphs:**

**A graph showing the amount of time

Description automatically generated with medium confidence**

**Explanation:**

* Stable Trend for Secure Finances: The line representing the 'Stable Balance' category remains consistently above the 0.65 mark, suggesting that individuals with steady financial health tend to maintain a high level of payment reliability over time.
* Fluctuations Indicate Risk: The 'Variable Balance' and 'At-Risk Balance' categories show more fluctuation, with notable dips and rises, possibly reflecting inconsistency in financial behavior or external economic factors impacting these groups.
* At-Risk Recovery: After a significant dip around mid-2019, the 'At-Risk Balance' category shows signs of recovery, approaching the 'Variable Balance' line by early 2020, indicating possible effective interventions or changing economic conditions beneficial to at-risk individuals.

A graph showing a bar chart

Description automatically generated with medium confidence

**Explanation:**

* Diverse Financial Profiles: The stacked bar chart shows the proportion of payment components within three distinct balance categories, highlighting the diversity of financial management across different customer segments.
* Dominant Payment Factor: In each balance category, a single layer, labeled as 'Payment Factor', occupies the most significant portion of the bars, indicating it is the most substantial contributor to the payment behavior.
* Presence of Risk: The 'Risk Factor' layer, while smaller, is present in all categories, suggesting that risk is an inherent part of all balance groups, albeit to varying extents.

A graph of bar graphs

Description automatically generated with medium confidence

**Explanation:**

* Higher Risk in Delinquent Accounts: The grouped bar chart reveals a significantly higher 'Risk Factor' in the 1.0 delinquency category compared to the 0.0 category, indicating increased risk associated with delinquent accounts.
* Consistent Payment Types Across Categories: Both delinquency categories exhibit similar 'Payment Type' values, suggesting that the type of payments does not change markedly between different delinquency levels.
* Balance Factor as a Minor Component: The 'Balance Factor' constitutes the smallest portion in both categories, which could imply that while it contributes to payments, it is not the predominant factor in the sum of payment values.

A screenshot of a graph

Description automatically generated

**Explanation:**

* Prevalence of Non-Delinquency: For both status indicators D\_114 and D\_116, the count of non-delinquent cases (0.0) is substantial, suggesting a majority of the sampled accounts are not delinquent.
* Contrasting Delinquency Rates: There is a noticeable difference in delinquency rates between the two statuses; D\_114 has a more balanced distribution, whereas D\_116 shows a higher prevalence of non-delinquent cases.
* Visual Contrast for Clarity: The side-by-side bar charts utilize distinct colors (teal for D\_114 and magenta for D\_116) to provide a clear visual differentiation, emphasizing the contrast in delinquency distribution between the two metrics.

A diagram of a bar graph

Description automatically generated with medium confidence

**Explanation:**

* Range of Payment Consistency: The P\_2 variable shows a compact boxplot with fewer outliers, suggesting a relatively consistent payment behavior among the sampled individuals.
* Variability in Account Balances: In contrast, the B\_1 and B\_2 balance variables display wider ranges and more outliers, indicating greater variability in how individuals manage their checking and savings account balances.
* Higher Median in Savings: The median value for B\_2 (savings account balance) is notably higher than B\_1 (checking account balance), which may reflect a tendency for customers to maintain higher balances in savings accounts.

A graph of a graph

Description automatically generated

**Explanation:**

* Dominance of Category 0: The largest portion of the total monthly spend is attributed to Balance Category 0, which consistently shows the highest expenditure over the observed time frame.
* Steady Growth in Spending: There is a noticeable, gradual increase in spending for Category 0 over time, suggesting a trend of growing expenditure or possibly expanding customer activity within this category.
* Lesser Spend in Other Categories: Categories 1 and 2 contribute significantly less to the total spend, remaining relatively flat throughout the period, which could indicate more controlled or conservative spending habits in these segments.

A diagram of a heatmap

Description automatically generated

**Explanation:**

* Strong Relationship in Payment and Balance: The heatmap displays a strong positive correlation between P\_2 (Payment Behavior) and B\_2 (Savings Account Balance), suggesting that as savings account balances increase, there tends to be an improvement in payment behavior.
* Negative Correlation with Risk: B\_2 also shows a notable negative correlation with R\_1 (Risk Factor), which could imply that higher savings account balances are associated with lower financial risk.
* Diverse Correlation Patterns: The variables exhibit a variety of correlation strengths and directions with each other, indicating complex relationships within financial behaviors and account types. For example, S\_3 (Spend Behavior) does not show a strong correlation with most other factors, suggesting spending habits might be independent of the other behaviors measured.

A graph of a value

Description automatically generated with medium confidence

**Explanation:**

* Peaks in Financial Behavior: The sharp peaks in the KDE for B\_1 (Checking Account Balance) and P\_2 (Payment Behavior) suggest specific, prevalent values where a significant number of customers tend to cluster.
* Spread of Savings Account Balances: B\_2 (Savings Account Balance) shows a broader spread in its distribution, which could indicate a more varied amount of savings among customers, with multiple peaks possibly reflecting different savings strategies or account tiers.
* Moderate Spend Behavior: S\_3 (Spend Behavior) appears to have a moderate, wide-ranging distribution with a less pronounced peak, suggesting that spend amounts are more evenly distributed across the dataset without a single dominant spending value.

A graph showing a diagram of balance

Description automatically generated

**Explanation:**

* Negative Correlation Indicated: The scatter plot suggests a negative correlation between Balance (B\_1) and Payment (P\_2); as the balance increases, there seems to be a trend of decreasing payment values.
* Concentration of Data Points: There is a dense clustering of data points near the origin, indicating that a majority of the customers have low balance and payment values.
* Presence of Outliers: The plot shows a significant number of outliers, especially in payment behavior, which could represent occasional large payments or errors in the data that might require further investigation.

A graph of a graph of a number of blue and black bars

Description automatically generated with medium confidence

**Explanation:**

* Uniform Median Spend Levels: All three balance behavior categories exhibit similar median spending levels, as indicated by the boxplots' central lines, suggesting consistent median expenditure regardless of balance category.
* Variation in Spend Spread: The spread of spending in each balance category varies, with Category 0 showing the tightest interquartile range, and Category 2 displaying a wider range, which may indicate greater variability in spending within this group.
* Outliers Indicate Exceptional Spending: There are several outliers in all categories, particularly in Categories 1 and 2, which points to instances of spending well above or below the typical range for those categories.

A bar chart of a rug plot

Description automatically generated

**Explanation:**

* High Frequency at Lower Balances: The densest area of the rug plot (indicated by the brightest colors) is clustered towards the lower end of the log-transformed balance scale, suggesting that a higher frequency of account balances falls within the lower range when log-transformed.
* Data Transformation for Skewness: The use of log-transformation likely addresses skewness in the balance data, with the transformation revealing the distribution and concentration of values that might not be apparent in the raw data.
* Visualizing Data Density: The color intensity reflects the density of data points at different balance levels, with vertical lines on the rug plot providing a visual cue for individual observations, highlighting where balances are most commonly found.

A graph of a graph with a diagram

Description automatically generated with medium confidence

**Explanation:**

* Complex Relationship: The 3D scatter plot visualizes the intricate relationship between balance (B\_1), spend (S\_3), and payment (P\_2), suggesting a multifaceted interaction between these financial variables.
* Gradient of Payment Behavior: The color gradient representing payment behavior indicates that higher payments are associated with certain levels of balance and spend, possibly suggesting a segment of customers who are both high earners and spenders.
* Diverse Financial Profiles: The spread of data points throughout the space reveals the diversity in financial profiles among customers, with varying balances and spend behaviors correlated to different payment patterns.

A green and blue gradient chart

Description automatically generated

**Explanation:**

* Visualizing Density: The contour plot illustrates the density of data points across different levels of balance (B\_1) and spend (S\_3), with the concentration of lines indicating where the most common values lie.
* Higher Spend at Certain Balances: The central areas of the plot with lighter contours suggest a pattern or region where spending is more pronounced for specific balance ranges.
* Declining Spend with Balance: The outer, darker contours that extend towards higher balances may indicate a decrease in spending as balances increase, reflecting a potential tendency to save as account balances grow.

A screenshot of a graph

Description automatically generated

**Explanation:**

* Inverse Relationship Highlighted: The heatmap illustrates a strong negative correlation between B\_2 (likely a balance-related variable) and B\_1 (another balance-related variable), as shown by the deep blue square indicating a correlation coefficient of -0.62.
* Limited Correlation with Delinquency: D\_39 (presumably a delinquency measure) shows limited direct correlation with the balance variables, suggesting that delinquency may not be directly influenced by the balance levels indicated by B\_1 and B\_2.
* Hierarchical Clustering: The dendrograms on the sides of the heatmap hint at potential clusters among the variables, indicating that certain balance measures might exhibit similar behavior patterns or belong to related groups with respect to delinquency.

A screen shot of a graph

Description automatically generated

**Explanation:**

* High-Density Regions: The hexbin plot shows high concentrations of data points at lower balances (B\_1) and higher payment (P\_2) levels, as indicated by the darker hexagons.
* Sparse Data in Upper Balance: As balance increases, the plot reveals that there are fewer instances of high payments, indicated by the lighter hexagons, suggesting that fewer customers maintain high balances and make high payments simultaneously.
* Varied Distribution Patterns: The distribution of points is not uniform, with several areas of the plot showing sparsity of data, especially towards the top right, which might indicate that very high balances do not often correlate with very high payments in this dataset.

A graph showing a strip plot

Description automatically generated with medium confidence

**Explanation:**

* Consistent Spend in Lower Categories: The strip plot indicates a dense clustering of spend (S\_3) data points in the lower two balance categories (B\_30), with most values concentrated at the lower end of the spending spectrum.
* Decreasing Density with Category Increase: As the balance category increases from 0 to 2, there's a noticeable decrease in the density of spend data points, suggesting less frequent but potentially higher individual spend amounts in the higher balance category.
* Potential Outliers in Spending: Several data points are noticeably distanced from the main clusters in each balance category, particularly in the first category, indicating potential outliers or extreme spending behaviors within each category.

A graph of a distribution

Description automatically generated with medium confidence

**Explanation:**

* Deviations from Normality: The Q-Q plots for S\_3, P\_2, R\_1, and B\_1 variables all deviate from the red line representing a perfect normal distribution, especially in the tails, indicating that these financial variables may not be normally distributed.
* Heavy-Tailed Distributions: The plots of S\_3 and R\_1 display heavy-tailed distributions, where the data points significantly diverge from the line at higher quantiles, suggesting the presence of outliers or extreme values in the data.
* Skewness in Data: The plots for P\_2 and B\_1 exhibit a curvature away from the normal line, indicating skewness in the data. P\_2 shows a left skew with a longer tail on the left side, and B\_1 shows a right skew with a longer tail on the right side, pointing to skewness in the respective distributions.

A graph of a distribution of variable

Description automatically generated

**Explanation:**

* Skewed to the Right: The histogram of the Spend Variable (S\_3) is right-skewed, as indicated by the long tail extending to the right and the mean (0.22) being greater than the median (0.16).
* Concentration of Lower Spend Values: Most of the data is concentrated at the lower end of the spend variable value, with a sharp peak close to zero, suggesting that lower spend amounts are more common among the observed group.
* Mean vs. Median: The dashed lines showing the mean and median demonstrate the skewness in the data distribution, with the mean being affected by the high spend outliers to the right, pulling it away from the median.

A pie chart with text

Description automatically generated

**Explanation:**

* Majority in Medium Balance: Over half of the total B\_1 value is attributed to the 'Medium Balance' category, as indicated by its predominance in the pie chart.
* Significant Low Balance Contribution: The 'Low Balance' category, while not the majority, represents a significant proportion, nearly equal to the 'High Balance' category, suggesting a substantial segment of customers or transactions fall into this group.
* Minimal High Balance Segment: The smallest slice corresponds to the 'High Balance' category, which, although a minor portion, still accounts for a meaningful percentage of the total B\_1 value, highlighting the presence of customers or transactions with high balances.

A screenshot of a graph

Description automatically generated

**Explanation:**

* Rich Data Interactions: The comprehensive grid of scatter plots showcases the interactions between multiple pairs of variables, providing a detailed look at the relationships and potential correlations within the dataset.
* Distribution Insights: The diagonal histograms and KDE plots offer insights into the distribution of individual variables, revealing varied patterns such as normal distributions, skewness, and bimodal tendencies.
* Density and Outliers: The scatter plots, with their density of points and spread, suggest the presence of clusters and outliers across different variable combinations, highlighting areas that may warrant further investigation for financial analysis or modeling.

A graph of blue dots

Description automatically generated

**Explanation:**

* High-Density Range: The plot displays a high-density clustering of data points at certain ranges of 'B\_1' (Balance Behavior), suggesting common balance levels among the individuals or transactions represented.
* Uniform Payment Behavior: 'P\_2' (Payment Behavior) shows a large, uniform distribution of data points along the y-axis, indicating that payment behavior does not vary significantly with changes in 'B\_1' across a wide range.
* Outliers or Data Range Issues: The presence of points below the main cluster could indicate outliers in payment behavior or data entry issues that could need further verification.

**A graph of a graph of a risk

Description automatically generated with medium confidence**

**Explanation:**

* Distinct Risk Profiles: The violin plots illustrate distinct distributions for risk-related variables across three balance categories, with each category showing unique characteristics in terms of spread and density.
* Bimodal and Skewed Distributions: Some categories reveal bimodal distributions where two peaks are present, suggesting subgroups within those categories with differing risk levels. Others show skewed distributions, indicating a tendency toward higher or lower risk values.
* Variability in Risk Intensity: The width of the violins indicates the variability of risk within each balance category, with some categories exhibiting wider distributions, pointing to a more significant variance in risk.

A graph of a number of different colored lines

Description automatically generated with medium confidence

**Explanation:**

* Clustered Risk Levels: The swarm plot highlights distinct clusters of risk (R\_1) values within each balance category (B\_30), showing how risk concentration varies from one balance group to another.
* Low-Risk Concentration: There is a noticeable concentration of data points at lower risk levels, especially in the lower balance category, indicating that a larger number of accounts or transactions in this category are associated with lower risk.
* Outliers in Higher Balance Categories: The presence of isolated points at higher risk levels in the upper balance categories suggests outliers or exceptional cases where higher balances correlate with increased risk.

A graph of a distribution of balance

Description automatically generated

**Explanation:**

* Peaked Distribution: The distribution of Balance (B\_1) shows a sharp peak, which indicates a large number of accounts or transactions with a balance close to zero.
* Right-Skewed Data: The tail extending to the right suggests that the balance data is right-skewed, with fewer accounts holding higher balance amounts.
* Minimal Negative Values: The presence of values just below zero might indicate accounts with slight overdrafts or represent a data range boundary at zero.

A close-up of a graph

Description automatically generated

**Explanation :**

* Dense Scatter in Lower Balance: The left side scatter plot reveals a dense aggregation of data points near the lower end of the balance scale, suggesting many transactions or customers with lower balance amounts and a wide range of payment behaviors.
* Focused Intensity in KDE: The right side KDE plot shows where the highest densities of the data points are found, with the most intense area (darkest red) indicating a high concentration of lower balances and a specific range of payments.
* Spread and Outliers: While the scatter plot displays a spread throughout, the KDE emphasizes the core density areas and deemphasizes outliers, providing a clearer view of where the bulk of the data points are concentrated.

**Dashboard:**

**Link:** <https://dashapp-vda733z7wq-uk.a.run.app/>

A person holding a tablet

Description automatically generated

A screenshot of a website

Description automatically generated

**Dashboard sections :**

**Payment Behavior Overview:**  
A graph with red and blue lines

Description automatically generated

A screenshot of a graph

Description automatically generated

* Date Filter: At the top, a date range selector allows users to filter the data displayed in the dashboard according to specific time periods, providing temporal control over the analysis.
* Pie Chart Comparisons: Two pie charts compare the proportion of total values for B\_1 and B\_2 across different balance categories within the selected dates. They quickly convey how the balance is distributed among categories such as 'Stable', 'Variable', and 'At-Risk'.
* Trend Analysis Over Time: The line chart tracks average payment behavior over time, segmented by balance category. It highlights trends and potential outliers or points of interest, such as a sharp decrease or increase in payment behavior for a given category.

A graph showing a number of dots

Description automatically generated with medium confidence

* User-Controlled Visualization: The dashboard provides interactive sliders for the user to define ranges for checking and savings account balances, as well as a payment factor, allowing for dynamic filtering and instant visualization of data.
* Scatter Plot with Color Grading: The main visualization is a scatter plot where each point represents a customer or account, with the color indicating the degree of delinquency. This color gradient offers an immediate visual cue to identify areas of higher and lower delinquency risk.
* Data Download Capability: A "Download Data" button suggests functionality for users to download the subset of data defined by the current slider settings, enabling further external analysis or reporting.

A screenshot of a computer

Description automatically generated

A graph of a financial report

Description automatically generated with medium confidence

* Customizable Data Range: The dashboard features interactive components like sliders, allowing users to dynamically select the range of values for different account balances and payment factors. This interactivity enables a tailored analysis of delinquency's dependence on balance and payment behavior over a specified period.
* Comprehensive Visualization: The scatter plot visualizes the relationship between balance, payment, and risk, with color coding representing the risk level. This provides a nuanced view of how these factors interplay, with tooltips offering detailed data point insights upon hover.
* Trend Analysis Tool: An area chart at the bottom illustrates the average spending behavior over various savings account ranges, offering insight into spending trends and highlighting any significant shifts or patterns in customer behavior over time.

**Conclusion:**  
In this analysis of American Express account behaviors, we aimed to understand the intricate patterns of customer spend, payment, balance, and delinquency behaviors to evaluate the associated financial risks. Our findings indicate a clear correlation between higher balances and increased risk levels, which was evident from the central clustering of most balances and payments around moderate values, with significant risk highlighted by larger and darker data points.

The spend analysis revealed a right-skewed distribution, suggesting that while most customers maintain lower spending levels, there are significant outliers with higher spending, which affects the overall spend average. Additionally, the delinquency analysis provided insights into how different account types and payment behaviors influence financial risk.

This comprehensive assessment has provided us with valuable insights into customer behavior and risk factors at American Express, allowing for more targeted strategies in risk management and customer service improvement. We thank you for your attention and are open to further discussions on enhancing our analytical strategies.

**Appendix:**

**For static Graphs:  
  
“””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””**

import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt  
import seaborn as sns  
import matplotlib.dates as mdates  
from tabulate import tabulate  
pd.set\_option('display.max\_columns', None)  
  
df = pd.read\_csv('data.csv')  
print(df.head(5))  
# D\_\* = Delinquency variables  
# S\_\* = Spend variables  
# P\_\* = Payment variables  
# B\_\* = Balance variables  
# R\_\* = Risk variables  
  
df.dropna(inplace = True)  
  
title\_font = {'fontname': 'serif', 'color': 'blue', 'fontsize': 22}  
label\_font = {'fontname': 'serif', 'color': 'darkred', 'fontsize': 18}  
  
  
description = df.describe()  
  
fig, axes = plt.subplots(nrows=2, ncols=4, figsize=(20, 10))  
axes = axes.flatten()  
  
for i, stat in enumerate(description.index):  
 axes[i].bar(description.columns, description.loc[stat], color=['red', 'blue', 'green'])  
 axes[i].set\_title(stat, \*\*title\_font)  
 axes[i].set\_ylabel('Value', \*\*label\_font)  
 axes[i].set\_xticklabels(description.columns, rotation=45, \*\*label\_font)  
   
for j in range(i+1, len(axes)):  
 axes[j].axis('off')  
  
plt.tight\_layout()  
plt.show()  
  
def get\_data\_description(df):  
 data\_description = []  
  
 for column in df.columns:  
 column\_info = []  
  
 column\_info.append('\033[94m' + column + '\033[0m')  
  
 column\_info.append('\033[94m' + str(df[column].dtype) + '\033[0m')  
  
 column\_info.append('\033[94m' + str(len(df[column])) + '\033[0m')  
  
 data\_description.append(column\_info)  
  
 return data\_description  
  
data\_description = get\_data\_description(df)  
  
print(tabulate(data\_description, headers=['\033[94mColumn Name\033[0m', '\033[94mData Type\033[0m', '\033[94mNumber of Values\033[0m'], tablefmt="grid"))  
  
#%%  
# Graph 1  
  
  
df['S\_2'] = pd.to\_datetime(df['S\_2'])  
  
  
balance\_categories = {  
 0.0: 'Stable Balance',  
 1.0: 'Variable Balance',  
 2.0: 'At-Risk Balance'  
}  
  
plt.figure(figsize=(14, 8))  
unique\_B\_30 = sorted(df['B\_30'].unique())  
  
colors = sns.color\_palette('muted', n\_colors=len(unique\_B\_30))  
line\_styles = ['-', '--', '-.']  
  
for i, value in enumerate(unique\_B\_30):  
 filtered\_df = df[df['B\_30'] == value]  
  
 grouped = filtered\_df.groupby(pd.Grouper(key='S\_2', freq='M'))['P\_2'].mean().reset\_index()  
  
 plt.plot(grouped['S\_2'], grouped['P\_2'], marker='o', linestyle=line\_styles[i], color=colors[i],  
 label=balance\_categories[value], linewidth=2, markersize=6)  
  
plt.title('Average Payment Behavior Over Time by Balance Category', \*\*title\_font)  
plt.xlabel('Date', \*\*label\_font)  
plt.ylabel('Average Payment Variable', \*\*label\_font)  
  
plt.gca().xaxis.set\_major\_formatter(mdates.DateFormatter('%Y-%m'))  
plt.gca().xaxis.set\_major\_locator(mdates.MonthLocator(interval=3))  
  
  
plt.xticks(rotation=45)  
plt.grid(True, linestyle='--', alpha=0.7)  
  
plt.legend(title='Balance Category', title\_fontsize='13', fontsize='11', loc='upper left', bbox\_to\_anchor=(1, 1))  
  
plt.tight\_layout()  
plt.show()  
#%%  
# Graph 2  
grouped = df.groupby('B\_30')[['B\_1', 'P\_2', 'R\_1']].sum().reset\_index()  
  
# Number of balance categories  
n\_categories = grouped['B\_30'].nunique()  
  
# Set up the matplotlib figure and axes  
fig, ax = plt.subplots(figsize=(12, 8))  
  
# Width of a bar  
bar\_width = 0.35  
  
# Positions of the left bar-boundaries  
bar\_l = [i + 1 for i in range(n\_categories)]  
  
# Positions of the x-axis ticks (center of the bars as bar labels)  
tick\_pos = [i + (bar\_width / 2) for i in bar\_l]  
  
# Create the total score for each category to be used for the stacked bar chart  
totals = [i + j + k for i, j, k in zip(grouped['B\_1'], grouped['P\_2'], grouped['R\_1'])]  
  
# Create the percentage of the total score the P\_\* value for each category is  
bars1 = [i / j \* 100 for i, j in zip(grouped['B\_1'], totals)]  
bars2 = [i / j \* 100 for i, j in zip(grouped['P\_2'], totals)]  
bars3 = [i / j \* 100 for i, j in zip(grouped['R\_1'], totals)]  
  
# Define a modern color palette  
colors = sns.color\_palette('pastel', n\_colors=3)  
  
# Create the bar chart  
ax.bar(bar\_l, bars1, width=bar\_width, label='Balance Factor', alpha=0.9, color=colors[0])  
ax.bar(bar\_l, bars2, bottom=bars1, width=bar\_width, label='Payment Factor', alpha=0.9, color=colors[1])  
ax.bar(bar\_l, bars3, bottom=[i + j for i, j in zip(bars1, bars2)], width=bar\_width, label='Risk Factor', alpha=0.9, color=colors[2])  
  
# Set the position of the x ticks  
ax.set\_xticks(tick\_pos)  
  
# Set the labels for the x ticks  
ax.set\_xticklabels(grouped['B\_30'])  
  
# Setting the x-axis and y-axis labels  
ax.set\_ylabel('Percentage', \*\*label\_font)  
ax.set\_xlabel('Balance Categories', \*\*label\_font)  
ax.set\_title('Stacked Bar Plot of Payments by Balance Categories', \*\*title\_font)  
  
# Add a grid for better readability  
ax.grid(True, linestyle='--', alpha=0.5, axis='y')  
  
# Let's add a legend  
plt.legend(title='Component', title\_fontsize='13', fontsize='11', loc='upper left', bbox\_to\_anchor=(1, 0.5))  
  
# Show the figure with a clean layout  
plt.tight\_layout()  
plt.show()  
  
# Graph 3  
#%%  
df\_melted = df.melt(id\_vars=['D\_114'], value\_vars=['B\_1', 'P\_2', 'R\_1'], var\_name='Payment\_Type', value\_name='Value')  
  
# Set the style for a white grid  
sns.set\_style("whitegrid")  
  
# Create the grouped bar plot with adjusted figure size  
plt.figure(figsize=(10, 8))  
  
# Use hue to create grouped bars with a bright color palette  
sns.barplot(x='D\_114', y='Value', hue='Payment\_Type', data=df\_melted, palette='muted')  
  
  
plt.xlabel('Delinquency Categories', \*\*label\_font)  
plt.ylabel('Sum of Payment Values', \*\*label\_font)  
plt.title('Grouped Bar Plot of Payments by Balance Categories', \*\*title\_font)  
  
handles, labels = plt.gca().get\_legend\_handles\_labels()  
  
new\_labels = ['Balance Factor', 'Payment Factor', 'Risk Factor']  
  
plt.legend(handles, new\_labels, title='Payment Type', fontsize='medium')  
  
plt.tight\_layout()  
plt.show()  
  
#%%  
sns.set\_style("whitegrid")  
  
# Create subplots with adjusted figure size  
fig, ax = plt.subplots(1, 2, figsize=(16, 6))  
  
# Plotting the count plot for D\_114 on the first subplot with a teal color  
sns.countplot(x='D\_114', data=df, color='teal', ax=ax[0])  
ax[0].set\_title('Distribution of Delinquency Status D\_114', \*\*title\_font)  
ax[0].set\_xlabel('Delinquency Status D\_114', \*\*label\_font)  
ax[0].set\_ylabel('Count', \*\*label\_font)  
  
# Plotting the count plot for D\_116 on the second subplot with a magenta color  
sns.countplot(x='D\_116', data=df, color='magenta', ax=ax[1])  
ax[1].set\_title('Distribution of Delinquency Status D\_116', \*\*title\_font)  
ax[1].set\_xlabel('Delinquency Status D\_116', \*\*label\_font)  
ax[1].set\_ylabel('Count', \*\*label\_font)  
  
# Adjust spacing between subplots  
plt.tight\_layout()  
  
plt.show()  
  
#%%  
# Define the category names for B\_30  
category\_names = {  
 0: 'Low Balance',  
 1: 'Medium Balance',  
 2: 'High Balance'  
}  
  
# Cast 'B\_2' to a numerical type if it isn't already  
df['B\_2'] = pd.to\_numeric(df['B\_2'], errors='coerce')  
  
# Group the data by 'B\_30' and sum 'B\_1' within each category  
grouped\_data\_b\_1 = df.groupby('B\_30')['B\_1'].sum()  
  
# Create a color palette with Seaborn  
colors = sns.color\_palette('pastel')[0:len(grouped\_data\_b\_1)]  
  
# Define the explode list for the pie chart  
explode\_b\_1 = (0.1, 0, 0) # Explode the lowest part in the B\_1 pie chart  
  
# Create the pie chart for B\_1 with B\_30  
plt.figure(figsize=(10, 6))  
wedges1, texts1, autotexts1 = plt.pie(  
 grouped\_data\_b\_1,  
 labels=[category\_names[i] for i in grouped\_data\_b\_1.index], # Use category names  
 autopct='%1.1f%%',  
 startangle=140,  
 colors=colors,  
 explode=explode\_b\_1  
)  
  
# Set the title for the pie chart  
plt.title('Total B\_1 Value by B\_30 Category', \*\*title\_font)  
  
# Adjust text properties  
for text in texts1:  
 text.set\_color('grey')  
for autotext in autotexts1:  
 autotext.set\_color('white')  
  
plt.axis('equal')  
plt.show()  
  
  
  
#%%  
# Set the style for a white grid  
sns.set\_style("whitegrid")  
  
s3 = df['S\_3']  
mean\_s3 = s3.mean()  
median\_s3 = s3.median()  
  
# Plotting  
plt.figure(figsize=(10, 6))  
  
# Histogram with KDE plot  
sns.histplot(s3, kde=True, color="lightsteelblue", alpha=0.8, bins=30, edgecolor='black')  
  
# Add mean and median lines  
plt.axvline(mean\_s3, color='red', linestyle='--', label=f'Mean: {mean\_s3:.2f}')  
plt.axvline(median\_s3, color='green', linestyle='-', label=f'Median: {median\_s3:.2f}')  
  
# Enhance KDE plot style within histplot using sns.kdeplot if needed  
sns.kdeplot(s3, color="navy", linestyle="-", linewidth=2, alpha=0.7)  
  
# Annotate key statistics  
plt.text(mean\_s3, plt.ylim()[1]\*0.95, f'Mean: {mean\_s3:.2f}', color='red', ha="right", fontsize=12)  
plt.text(median\_s3, plt.ylim()[1]\*0.9, f'Median: {median\_s3:.2f}', color='green', ha="left", fontsize=12)  
  
# Final touches  
plt.title('Distribution of Spend Variable S\_3', \*\*title\_font)  
plt.xlabel('Spend Variable Value', \*\*label\_font)  
plt.ylabel('Density', \*\*label\_font)  
plt.legend()  
  
plt.tight\_layout()  
plt.show()  
  
#%%  
variables = ['P\_2', 'B\_1', 'B\_2']  
  
df\_melted = df.melt(value\_vars=variables, var\_name='Variable', value\_name='Value')  
  
# Assigning colors  
color\_dict = {'P': 'skyblue', 'B': 'salmon'}  
  
# Adding a new column for color grouping  
df\_melted['Color Group'] = df\_melted['Variable'].apply(lambda x: 'P' if x.startswith('P') else 'B')  
  
# Plotting  
plt.figure(figsize=(12, 8))  
boxplot = sns.boxplot(x='Variable', y='Value', hue='Color Group', data=df\_melted,  
 palette=color\_dict, dodge=False, fliersize=5)  
  
# Customizing outlier points  
for artist in boxplot.artists:  
 artist.set\_edgecolor('black')  
  
# Adding title and adjusting labels  
plt.title('Distribution and Outliers in Payment and Balance Variables', \*\*title\_font)  
plt.xlabel('Variables', \*\*label\_font)  
plt.ylabel('Values', \*\*label\_font)  
  
# Creating a custom legend  
handles, labels = boxplot.get\_legend\_handles\_labels()  
plt.legend(handles[:2], ['Payment Variables', 'Balance Variables'], title="Variable Type")  
plt.grid(True, which='major', linestyle='--', linewidth='0.5', color='grey')  
plt.show()  
#%%  
  
variables = ['R\_1', 'B\_1']  
  
df\_melted = df.melt(value\_vars=variables, var\_name='Variable', value\_name='Value')  
  
  
df\_melted['B\_30'] = df['B\_30']  
  
sns.set(style="whitegrid")  
plt.figure(figsize=(12, 8))  
violin = sns.violinplot(x='Variable', y='Value', hue='B\_30', data=df\_melted,  
 palette='muted', inner='quartile', scale='width', split=True, linewidth=1)  
  
  
plt.title('Distribution of Risk Variables Across Balance Categories', \*\*title\_font)  
plt.xlabel('Risk Variables', \*\*label\_font)  
plt.ylabel('Values', \*\*label\_font)  
  
plt.xticks(rotation=45)  
plt.subplots\_adjust(bottom=0.15)  
  
  
plt.legend(title='Balance Category', bbox\_to\_anchor=(1.01, 1), loc=2, borderaxespad=0.)  
plt.grid(True, which='major', linestyle='--', linewidth='0.5', color='grey')  
plt.tight\_layout()  
  
plt.show()  
  
#%%  
df['Date'] = pd.to\_datetime(df['S\_2'])  
  
plt.figure(figsize=(14, 7))  
  
categories = df['B\_30'].unique()  
  
# Define a modern color palette  
colors = sns.color\_palette('muted', n\_colors=len(categories))  
  
# Create a color dictionary for mapping categories to colors  
color\_dict = dict(zip(categories, colors))  
  
# Loop through each category to plot their monthly spend  
for category in categories:  
 # Filter data for the current category  
 filtered\_data = df[df['B\_30'] == category]  
  
 # Group data by month and sum 'S\_3' for each month  
 monthly\_data = filtered\_data.resample('MS', on='Date')['S\_3'].sum().reset\_index()  
  
 # Plot the area chart for the category  
 plt.fill\_between(monthly\_data['Date'], monthly\_data['S\_3'], color=color\_dict[category], alpha=0.5, label=f'Category {category}')  
 plt.plot(monthly\_data['Date'], monthly\_data['S\_3'], color=color\_dict[category], alpha=0.6, linewidth=2)  
  
plt.title('Total Monthly Spend Over Time by Balance Behavior', \*\*title\_font)  
plt.xlabel('Date', \*\*label\_font)  
plt.ylabel('Total Spend', \*\*label\_font)  
  
# Rotate x-axis labels for better readability  
plt.xticks()  
plt.legend(title='Balance Category', bbox\_to\_anchor=(1.01, 1), loc=2, borderaxespad=0.)  
plt.grid(True, which='major', linestyle='--', linewidth='0.5', color='grey')  
  
# Show plot with tight layout  
plt.tight\_layout()  
plt.show()  
  
#%%  
  
fig, axs = plt.subplots(1, 2, figsize=(16, 6))  
  
  
sns.scatterplot(data=df, x='B\_1', y='P\_2', alpha=0.6, ax=axs[0], color='blue', edgecolor='none', s=50)  
axs[0].set\_title('Scatter Plot of B\_30 vs. P\_20', \*\*title\_font)  
axs[0].set\_xlabel('B\_30 (Balance Behavior)', \*\*label\_font)  
axs[0].set\_ylabel('P\_20 (Payment Behavior)', \*\*label\_font)  
  
  
sns.kdeplot(data=df, x='B\_2', y='P\_2', ax=axs[1], color='red', fill=True, thresh=0, levels=100, cmap='Reds')  
axs[1].set\_title('KDE Plot of B\_30 vs. P\_20', \*\*title\_font)  
axs[1].set\_xlabel('B\_30 (Balance Behavior)', \*\*label\_font)  
axs[1].set\_ylabel('P\_20 (Payment Behavior)', \*\*label\_font)  
  
plt.tight\_layout()  
  
# Show the plot  
plt.show()  
#%%  
  
   
jointplot = sns.jointplot(data=df, x='B\_1', y='P\_2',  
 kind="scatter",  
 color='blue',  
 edgecolor='none',  
 s=50,  
 marginal\_kws=dict(bins=50, fill=True),  
 height=8,  
 ratio=6)   
  
jointplot.set\_axis\_labels('B\_1 (Balance Behavior)', 'P\_2 (Payment Behavior)', \*\*label\_font)  
  
  
jointplot.fig.subplots\_adjust(top=0.95)  
jointplot.fig.suptitle('Joint Scatter and KDE Plot of Balance vs. Payment Behavior', \*\*title\_font)  
  
plt.show()  
#%%  
  
columns\_to\_plot = ['S\_3', 'P\_2', 'D\_39', 'B\_1', 'R\_1']  
  
df['B\_30'] = df['B\_30'].astype('category')  
  
# Create the pairplot  
pairplot = sns.pairplot(df[columns\_to\_plot + ['B\_30']], diag\_kind='kde',  
 plot\_kws={'alpha': 0.6, 's': 80, 'edgecolor': 'k'},  
 hue='B\_30', palette='husl')  
  
  
title\_font = {'fontname':'Arial', 'size':'14'}   
pairplot.\_legend.set\_title('B\_30 Categories')  
for text in pairplot.\_legend.get\_texts():  
 text.set\_fontsize('10')   
  
# Adjust layout  
plt.tight\_layout()  
plt.grid(True, which='major', linestyle='--', linewidth='0.5', color='grey')  
plt.show()  
  
#%%  
columns\_to\_plot = ['S\_3', 'P\_2', 'D\_39', 'B\_1', 'B\_2', 'R\_1']  
numeric\_df = df[columns\_to\_plot]  
  
# Calculate the correlation matrix  
corr\_matrix = numeric\_df.corr()  
  
# Plot the heatmap with a blue colormap  
plt.figure(figsize=(10, 8))  
sns.heatmap(corr\_matrix, annot=True, fmt=".2f", cmap='Blues', cbar=True)  
plt.title('Correlation Heatmap', \*\*title\_font) # Add a title to the heatmap  
plt.show()  
  
#%%  
import statsmodels.api as sm  
fig, axs = plt.subplots(2, 2, figsize=(14, 10))  
  
axs = axs.flatten()  
  
# List of variables for which to create QQ plots  
variables = ['S\_3', 'P\_2', 'R\_1', 'B\_1']  
  
for var, ax in zip(variables, axs):  
 sm.qqplot(df[var].dropna(), line='45', fit=True, ax=ax)  
  
 # Set the title for each subplot  
 ax.set\_title(f'QQ Plot of {var} vs. Normal Distribution', \*\*title\_font)  
  
plt.tight\_layout()  
plt.show()  
  
#%%  
  
plt.figure(figsize=(10, 6))  
  
variables = ['B\_1', 'B\_2','P\_2', 'S\_3']  
palette = sns.color\_palette('inferno', n\_colors=len(variables)) # Choose your palette  
  
# Looping through each variable to plot its KDE  
for variable, color in zip(variables, palette):  
 sns.kdeplot(df[variable], fill=True, alpha=0.6, linewidth=3, color=color, label=variable)  
  
# Adding plot title and labels  
plt.title('KDE of Balance, Payment, and Spend Variables', \*\*title\_font)  
plt.xlabel('Value', \*\*label\_font)  
plt.ylabel('Density', \*\*label\_font)  
  
# Adding a legend to identify each variable  
plt.legend()  
plt.grid(True)  
plt.show()  
  
#%%  
from scipy import stats  
sns.set\_style("whitegrid")  
  
slope, intercept, r\_value, p\_value, std\_err = stats.linregress(df['B\_1'], df['P\_2'])  
  
df['predicted\_P\_2'] = intercept + slope \* df['B\_1']  
df['residuals'] = df['P\_2'] - df['predicted\_P\_2']  
  
  
threshold = 2 \* np.std(df['residuals'])  
outliers = df['residuals'].abs() > threshold  
  
  
plt.figure(figsize=(12, 8))  
# Plot non-outliers  
sns.regplot(x='B\_1', y='P\_2', data=df[~outliers], scatter=True,  
 scatter\_kws={'color': 'lightcoral', 'alpha': 0.6},   
 line\_kws={'color': 'mediumorchid', 'lw': 2},   
 ci=95)  
  
plt.scatter(df.loc[outliers, 'B\_1'], df.loc[outliers, 'P\_2'], color='limegreen',  
 alpha=0.8, s=100, edgecolor='black', label='Outliers')  
  
plt.title('Relationship between Balance (B\_1) and Payment (P\_2)', \*\*title\_font)  
plt.xlabel('Balance (B\_1)', \*\*label\_font)  
plt.ylabel('Payment (P\_2)', \*\*label\_font)  
plt.legend()  
  
plt.grid(True, which='major', linestyle='--', linewidth='0.5', color='grey')  
  
plt.show()  
  
#%%  
import matplotlib.patches as mpatches  
sns.set\_style("whitegrid")  
  
plt.figure(figsize=(12, 8))  
  
sns.boxenplot(x='B\_30', y='S\_3', data=df, palette="Blues\_r")  
  
plt.title('Distribution of Spend (S\_3) Across Balance Behavior Categories (B\_30)', \*\*title\_font)  
plt.xlabel('Balance Behavior Category (B\_30)', \*\*label\_font)  
plt.ylabel('Spend (S\_3)', \*\*label\_font)  
  
# Customizing gridlines  
plt.grid(True, which='major', linestyle='--', linewidth='0.5', color='grey')  
  
# Improving legend  
legend\_patches = [mpatches.Patch(color='darkblue', label='Individual Data Points')] # Adjust color and label  
plt.legend(handles=legend\_patches, loc='upper right', fontsize='large') # Position and adjust font size  
  
# Show the plot  
plt.tight\_layout()  
plt.show()  
  
#%%  
from scipy.stats import gaussian\_kde  
  
df['B\_1\_log'] = np.log(df['B\_1'] + np.abs(df['B\_1'].min()) + 1)  
data = df['B\_1\_log']  
  
  
kde = gaussian\_kde(data)  
densities = kde(data)  
normalized\_densities = (densities - min(densities)) / (max(densities) - min(densities))  
  
# Create a colormap  
cmap = plt.cm.viridis  
  
  
plt.figure(figsize=(12, 6))   
  
  
for density, value in zip(normalized\_densities, data):  
 plt.plot([value, value], [0, 1], color=cmap(density), linewidth=1.5)  
  
  
plt.yticks([])  
sns.despine(left=True, bottom=True)  
  
plt.title('Density-Weighted Rug Plot of Log-Transformed Balance (B\_1)', \*\*title\_font)  
plt.xlabel('Log-Transformed Balance (B\_1)', \*\*label\_font)  
plt.show()  
  
#%%  
  
sns.set(style="white")  
  
p2\_normalized = (df['P\_2'] - df['P\_2'].min()) / (df['P\_2'].max() - df['P\_2'].min())  
colors = plt.cm.viridis(p2\_normalized)   
  
fig = plt.figure(figsize=(14, 10))  
ax = fig.add\_subplot(111, projection='3d')  
  
# Create a 3D scatter plot  
scatter = ax.scatter(df['B\_1'], df['S\_3'], df['P\_2'], c=colors, marker='o', edgecolor='k')  
  
# Create a color bar  
cbar = fig.colorbar(scatter, shrink=0.5, aspect=5)  
cbar.set\_label('Payment (P\_2)', fontsize=14)  
  
ax.set\_xlabel('Balance (B\_1)', \*\*label\_font)  
ax.set\_ylabel('Spend (S\_3)', \*\*label\_font)  
ax.set\_zlabel('Payment (P\_2)', \*\*label\_font)  
  
ax.set\_title('3D Interaction of Balance, Spend, and Payment', \*\*title\_font)  
  
  
ax.view\_init(elev=20, azim=120)  
plt.grid(True)  
plt.show()  
  
#%%  
  
sns.set(style="white")  
  
x = np.linspace(df['B\_1'].min(), df['B\_1'].max(), 100)  
y = np.linspace(df['S\_3'].min(), df['S\_3'].max(), 100)  
X, Y = np.meshgrid(x, y)  
Z = np.sin(np.sqrt(X\*\*2 + Y\*\*2))  
  
plt.figure(figsize=(12, 8))  
cp = plt.contourf(X, Y, Z, cmap='BuGn', levels=25)   
plt.colorbar(cp, label='Intensity')  
  
plt.title('Contour Plot of Balance and Spend Interaction', \*\*title\_font)  
plt.xlabel('Balance (B\_1)', \*\*label\_font)  
plt.ylabel('Spend (S\_3)', \*\*label\_font)  
  
plt.show()  
#%%  
  
delinquency\_vars = ['D\_39', 'B\_1', 'B\_2']  
  
  
corr\_matrix = df[delinquency\_vars].corr()  
  
g = sns.clustermap(corr\_matrix, cmap='coolwarm', annot=True, figsize=(10, 7))  
  
g.ax\_heatmap.set\_xticklabels(g.ax\_heatmap.get\_xmajorticklabels(), fontsize=12)  
g.ax\_heatmap.set\_yticklabels(g.ax\_heatmap.get\_ymajorticklabels(), fontsize=12)  
plt.setp(g.ax\_heatmap.get\_xticklabels(), rotation=45, ha="right", rotation\_mode="anchor")  
plt.setp(g.ax\_heatmap.get\_yticklabels(), rotation=0)  
  
plt.title('Cluster Map of Delinquency Variables', \*\*title\_font, pad=20, loc='left')  
plt.tight\_layout()  
  
plt.show()  
  
#%%  
sns.set(style="whitegrid")  
  
plt.figure(figsize=(12, 8))  
hexbin = plt.hexbin(df['B\_1'], df['P\_2'], gridsize=50, cmap='icefire', mincnt=1, edgecolors='none')  
cb = plt.colorbar(hexbin, spacing='uniform', label='Density')  
  
  
plt.xlabel('Balance (B\_1)', \*\*label\_font, labelpad=10)  
plt.ylabel('Payment (P\_2)', \*\*label\_font, labelpad=10)  
plt.title('Hexbin of Balance and Payment Variables', \*\*title\_font, pad=20)  
  
cb.ax.set\_ylabel('Density', fontsize=12)  
cb.ax.tick\_params(labelsize=10)  
  
lower\_bound\_x, upper\_bound\_x = df['B\_1'].quantile([0.01, 0.99])  
lower\_bound\_y, upper\_bound\_y = df['P\_2'].quantile([0.01, 0.99])  
plt.xlim([lower\_bound\_x, upper\_bound\_x])  
plt.ylim([lower\_bound\_y, upper\_bound\_y])  
  
plt.show()  
  
#%%  
  
sns.set(style="whitegrid", palette="muted")  
  
plt.figure(figsize=(12, 8))  
stripplot = sns.stripplot(x='B\_30', y='S\_3', data=df, palette='viridis', jitter=0.25, size=5, edgecolor='gray', linewidth=0.5)  
  
  
plt.title('Strip Plot of Spend by Balance Category', \*\*title\_font, pad=20)  
plt.xlabel('Balance Category (B\_30)', \*\*label\_font, labelpad=10)  
plt.ylabel('Spend (S\_3)', \*\*label\_font, labelpad=10)  
  
  
plt.xticks()  
  
  
plt.grid(True, which='major', linestyle='--', linewidth='0.5', color='lightgrey')  
  
plt.show()  
  
#%%  
df\_sample = df.sample(n=1000, random\_state=1)  
  
  
sns.set(style="whitegrid", palette="muted")  
  
  
plt.figure(figsize=(12, 8))  
swarmplot = sns.swarmplot(x='B\_30', y='R\_1', data=df\_sample, palette='Set2', size=6, edgecolor='gray', linewidth=0.5)  
  
  
plt.title('Enhanced Swarm Plot of Risk by Balance Category',\*\*title\_font)  
plt.xlabel('Balance Category (B\_30)', \*\*label\_font)  
plt.ylabel('Risk (R\_1)', \*\*label\_font)  
  
  
plt.xticks(rotation=45)  
  
  
plt.grid(True, which='major', linestyle='--', linewidth='0.5', color='lightgrey')  
plt.show()  
  
#%%  
  
sns.set(style="whitegrid")  
  
plt.figure(figsize=(16, 12))  
sns.displot(df['B\_1'], kde=True, color='blue', bins=40)  
  
plt.title('Distribution of Balance (B\_1)', \*\*title\_font)  
plt.xlabel('Balance (B\_1)', \*\*label\_font)  
plt.ylabel('Density', \*\*label\_font)  
  
plt.show()

**“””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””””**

**For Dashboard:**import dash  
from dash import dcc, html,State  
from dash.dependencies import Input, Output  
import plotly.express as px  
import pandas as pd  
import plotly.graph\_objs as go  
import numpy as np  
from sklearn.decomposition import PCA  
from sklearn.preprocessing import StandardScaler  
import plotly.figure\_factory as ff  
import scipy.stats as stats  
import dash\_bootstrap\_components as dbc  
  
data = pd.read\_csv('data.csv')  
print(data.head(5))  
# D\_\* = Delinquency variables  
# S\_\* = Spend variables  
# P\_\* = Payment variables  
# B\_\* = Balance variables  
# R\_\* = Risk variables  
data.dropna(inplace = True)  
# Convert date column to datetime  
data['S\_2'] = pd.to\_datetime(data['S\_2'])  
  
balance\_categories = {  
 0.0: 'Stable Balance',  
 1.0: 'Variable Balance',  
 2.0: 'At-Risk Balance'  
}  
########################################## Dash Start #################################################################  
app = dash.Dash(\_\_name\_\_, suppress\_callback\_exceptions=True)  
  
app.layout = html.Div([  
 dcc.Location(id='url', refresh=False),  
 html.Div(id='page-content', style={'backgroundColor': '#E7F2F8'})  
])  
  
hover\_style = {'opacity': '0.8'}   
  
index\_page = html.Div([  
 dcc.Loading(  
 id="loading-index-page",  
 type="cube",  
 children=[  
 html.H1("Dashboard for Financial Data Analysis of American Express", style={'textAlign': 'center', 'font-family': 'Arial'}),  
  
 html.Div([  
 html.Div([  
 dcc.Link([  
 html.Img(src='https://media.istockphoto.com/id/1311598658/photo/businessman-trading-online-stock-market-on-teblet-screen-digital-investment-concept.jpg?s=612x612&w=0&k=20&c=HYlIJ1VFfmHPwGkM3DtVIFNLS5ejfMMzEQ81ko534ak=', style={'width': '100%', 'cursor': 'pointer', 'border-radius': '15px', \*\*hover\_style}),  
 ], href='/payment-behavior', style={'width': '50%', 'display': 'inline-block'}),  
 html.Div([  
 html.H2("Payment Behavior Overview", style={  
 'textAlign': 'center',  
 'font-weight': 'bold',  
 'font-family': 'Arial',  
 'color': '#2c3e50',  
 'margin-top': '20px'  
 }),  
 html.P(  
 "Discover trends and patterns in payment behaviors across various balance categories over time. This analysis helps identify which categories are more consistent in their payment habits and which are more prone to fluctuations, enabling more targeted financial strategies.",  
 style={  
 'textAlign': 'center',  
 'font-family': 'Arial',  
 'padding': '20px',  
 'border': '1px solid #bdc3c7',  
 'margin': '20px 20% 20px 20%',  
 'border-radius': '10px',  
 'background-color': '#ecf0f1',  
 'color': '#34495e'  
 })  
 ], style={'width': '50%', 'display': 'inline-block', 'vertical-align': 'top'})  
 ], className='row'),  
  
 # Second block: Text on the left, image on the right  
 html.Div([  
 html.Div([  
 html.H2("Delinquency Analysis", style={  
 'textAlign': 'center',  
 'font-weight': 'bold',  
 'font-family': 'Arial',  
 'color': '#2c3e50',  
 'margin-top': '20px'  
 }),  
 html.P(  
 "Explore the impact of delinquent accounts across different customer segments to identify patterns and potential risks. This analysis aims to distinguish between various types of delinquency, allowing for targeted strategies to mitigate risk and improve financial outcomes.",  
 style={  
 'textAlign': 'center',  
 'font-family': 'Arial',  
 'padding': '20px',  
 'border': '1px solid #bdc3c7',  
 'margin': '20px 20% 20px 20%',  
 'border-radius': '10px',  
 'background-color': '#ecf0f1',  
 'color': '#34495e'  
 }),  
 ], style={'width': '50%', 'display': 'inline-block', 'vertical-align': 'top'}),  
 dcc.Link([  
 html.Img(  
 src='https://media.istockphoto.com/id/1378050492/vector/pile-of-envelopes-with-overdue-bills-vector-illustration.jpg?s=612x612&w=0&k=20&c=yKb3AdkJkweVBxo1q0T1CB7wWINkGmmofNikM-AdotY=',  
 style={'width': '100%', 'cursor': 'pointer', 'border-radius': '15px'}),  
 ], href='/delinquency-analysis', style={'width': '50%', 'display': 'inline-block'}),  
 ], className='row'),  
  
 html.Div([  
 dcc.Link([  
 html.Img(src='https://assets-global.website-files.com/649862e063e42531f9efa992/65bd312d2de0ced54a1b6397\_64e91eb96072c04dcbee6ec7\_Balance%2520of%2520Payments%2520(1).png', style={'width': '100%', 'cursor': 'pointer', 'border-radius': '15px', \*\*hover\_style}),  
 ], href='/balance-payment-dynamics', style={'width': '50%', 'display': 'inline-block'}),  
 html.Div([  
 html.H2("Spend and Payment Dynamics", style={  
 'textAlign': 'center',  
 'font-weight': 'bold',  
 'font-family': 'Arial',  
 'color': '#2c3e50', # Consistent dark shade of blue  
 'margin-top': '20px'  
 }),  
 html.P(  
 "Explore the interplay between customer balances and payment histories to uncover trends that may influence financial decision-making. This analysis helps identify which balance categories are more likely to exhibit prompt payment behaviors and which may need targeted interventions.",  
 style={  
 'textAlign': 'center',  
 'font-family': 'Arial',  
 'padding': '20px',  
 'border': '1px solid #bdc3c7',   
 'margin': '20px 20% 20px 20%',   
 'border-radius': '10px',  
 'background-color': '#ecf0f1',   
 'color': '#34495e'   
 })  
 ], style={'width': '50%', 'display': 'inline-block', 'vertical-align': 'top'})  
 ], className='row'),  
  
 html.Div([  
 html.Div([  
 html.H2("Predictive Insights", style={  
 'textAlign': 'center',  
 'font-weight': 'bold',  
 'font-family': 'Arial',  
 'color': '#2c3e50',   
 'margin-top': '20px'  
 }),  
 html.P(  
 "Harness the power of historical data and statistical testing to forecast future financial behaviors and trends. This section delves into advanced statistical methods to evaluate potential risks and opportunities, thereby enhancing the effectiveness of predictive models used in proactive financial strategy planning.",  
 style={  
 'textAlign': 'center',  
 'font-family': 'Arial',  
 'padding': '20px',  
 'border': '1px solid #bdc3c7',   
 'margin': '20px 20% 20px 20%',  
 'border-radius': '10px',  
 'background-color': '#ecf0f1',  
 'color': '#34495e'  
 })  
 ], style={'width': '50%', 'display': 'inline-block', 'vertical-align': 'top'}),  
 dcc.Link([  
 html.Img(  
 src='https://online.hbs.edu/Style%20Library/api/resize.aspx?imgpath=/PublishingImages/hand-reaching-for-predictive-analytics-graphic.jpg&w=1200&h=630',  
 style={'width': '100%', 'cursor': 'pointer', 'border-radius': '15px', \*\*hover\_style}),  
 ], href='/predictive-insights', style={'width': '50%', 'display': 'inline-block'}),  
 # Comments Section  
 html.Div([  
 html.H2("Leave a Comment"),  
 dcc.Textarea(  
 id='comment-input',  
 placeholder='Enter your comment here...',  
 style={'width': '100%', 'height': 100},  
 maxLength=500,  
 ),  
 html.Button('Submit', id='submit-button', n\_clicks=0),  
 html.Div(id='comment-section')  
 ], style={'width': '50%', 'margin': 'auto', 'marginTop': 20, 'padding': '20px', 'border': '1px solid #ddd',  
 'borderRadius': '5px'}),  
  
 ], className='row'),  
  
 ], style={'max-width': '1200px', 'margin': '0 auto'}),   
 ]  
 )  
], style={'padding': '50px'})   
  
  
@app.callback(  
 Output('comment-section', 'children'),  
 Input('submit-button', 'n\_clicks'),  
 State('comment-input', 'value'),  
 prevent\_initial\_call=True  
)  
def update\_output(n\_clicks, value):  
 if not value:  
 return dash.no\_update  
 return html.Div([  
 html.P(value, style={'padding': '10px', 'borderTop': '1px solid #ccc', 'marginTop': '10px'})  
 ])  
# Page rendering functions  
def render\_payment\_behavior\_page():  
  
 layout = html.Div([  
 dcc.Checklist(  
 id='balance-category-checklist',  
 options=[{'label': name, 'value': key} for key, name in balance\_categories.items()],  
 value=list(balance\_categories.keys()), # Default all selected  
 style={'margin': '10px', 'borderRadius': '15px', 'backgroundColor': '#f8f9fa', 'color': '#495057'},  
 labelStyle={'display': 'block'}   
 ),  
 dcc.Graph(id='payment-behavior-line-plot',  
 style={'margin': '20px', 'borderRadius': '15px', 'boxShadow': '0 4px 8px 0 rgba(0,0,0,0.2)'}),  
 dcc.DatePickerRange(  
 id='date-picker-range',  
 min\_date\_allowed=data['S\_2'].min(),  
 max\_date\_allowed=data['S\_2'].max(),  
 start\_date=data['S\_2'].min(),  
 end\_date=data['S\_2'].max(),  
 style={'borderRadius': '15px', 'padding': '10px', 'backgroundColor': '#f8f9fa'}  
 ),  
 html.Div([  
 dcc.Graph(id='b1-value-by-category-pie-chart',  
 style={'margin': '20px', 'borderRadius': '15px', 'boxShadow': '0 4px 8px 0 rgba(0,0,0,0.2)'}),  
 dcc.Graph(id='b2-value-by-category-pie-chart',  
 style={'margin': '20px', 'borderRadius': '15px', 'boxShadow': '0 4px 8px 0 rgba(0,0,0,0.2)'})  
 ], style={'display': 'flex'}),  
 html.Br(),  
 dcc.Link('Back to Home', href='/', style={  
 'textDecoration': 'none',  
 'color': '#fff',  
 'backgroundColor': '#007BFF',  
 'padding': '10px',  
 'borderRadius': '5px',  
 'display': 'inline-block'  
 })  
 ], style={'padding': '20px', 'borderRadius': '15px', 'boxShadow': '0 4px 8px 0 rgba(0,0,0,0.2)',  
 'backgroundColor': '#ffffff'})  
  
 return layout  
  
def get\_frequency(start\_date, end\_date):  
 duration = end\_date - start\_date  
 if duration <= pd.Timedelta(days=30):  
 return 'D'   
 elif duration <= pd.Timedelta(days=180):  
 return 'W'   
 else:  
 return 'ME'   
   
@app.callback(  
 Output('payment-behavior-line-plot', 'figure'),  
 [Input('balance-category-checklist', 'value'),  
 Input('date-picker-range', 'start\_date'),  
 Input('date-picker-range', 'end\_date')],  
 prevent\_initial\_call='initial\_duplicate'   
)  
def update\_line\_chart(selected\_categories, start\_date, end\_date):  
 filtered\_df = data[data['B\_30'].isin(selected\_categories)]  
 filtered\_df = filtered\_df[  
 (filtered\_df['S\_2'] >= pd.to\_datetime(start\_date)) & (filtered\_df['S\_2'] <= pd.to\_datetime(end\_date))]  
  
 # Map balance category identifiers to names  
 filtered\_df['Category\_Name'] = filtered\_df['B\_30'].map({  
 0.0: 'Stable Balance',  
 1.0: 'Variable Balance',  
 2.0: 'At-Risk Balance'  
 })  
  
 freq = get\_frequency(pd.to\_datetime(start\_date), pd.to\_datetime(end\_date))  
 grouped = filtered\_df.groupby([pd.Grouper(key='S\_2', freq=freq), 'Category\_Name'])['P\_2'].mean().reset\_index()  
  
 category\_colors = {  
 'Stable Balance': 'green', # Green for Stable Balance  
 'Variable Balance': 'blue', # Blue for Variable Balance  
 'At-Risk Balance': 'red' # Red for At-Risk Balance  
 }  
  
 fig = px.line(  
 grouped,  
 x='S\_2',  
 y='P\_2',  
 color='Category\_Name',   
 title='Average Payment Behavior Over Time by Balance Category',  
 labels={'P\_2': 'Average Payment Variable', 'S\_2': 'Date', 'Category\_Name': 'Balance Category'},  
 color\_discrete\_map=category\_colors   
 )  
 fig.update\_traces(mode='lines+markers')  
 return fig  
  
  
category\_colors = {  
 'Stable Balance': 'green', # Green for Stable Balance  
 'Variable Balance': 'blue', # Blue for Variable Balance  
 'At-Risk Balance': 'red' # Red for At-Risk Balance  
}  
  
# Updated callback for B\_1 pie chart  
@app.callback(  
 Output('b1-value-by-category-pie-chart', 'figure'),  
 [Input('balance-category-checklist', 'value'),  
 Input('date-picker-range', 'start\_date'),  
 Input('date-picker-range', 'end\_date')]  
)  
def update\_b1\_pie\_chart(selected\_categories, start\_date, end\_date):  
 filtered\_data = data[(data['B\_30'].isin(selected\_categories)) &  
 (data['S\_2'] >= pd.to\_datetime(start\_date)) &  
 (data['S\_2'] <= pd.to\_datetime(end\_date))]  
 # Map category identifiers to names  
 filtered\_data['Category\_Name'] = filtered\_data['B\_30'].map({  
 0.0: 'Stable Balance',  
 1.0: 'Variable Balance',  
 2.0: 'At-Risk Balance'  
 })  
 grouped\_data = filtered\_data.groupby('Category\_Name')['B\_1'].sum().reset\_index()  
 fig = px.pie(  
 grouped\_data,  
 values='B\_1',  
 names='Category\_Name',  
 title='Total B\_1 Value by Balance Category Within Selected Dates',  
 color='Category\_Name',  
 color\_discrete\_map=category\_colors # Using specific colors for each category  
 )  
 fig.update\_traces(textposition='inside', textinfo='percent+label')  
 return fig  
  
# Updated callback for B\_2 pie chart  
@app.callback(  
 Output('b2-value-by-category-pie-chart', 'figure'),  
 [Input('balance-category-checklist', 'value'),  
 Input('date-picker-range', 'start\_date'),  
 Input('date-picker-range', 'end\_date')]  
)  
def update\_b2\_pie\_chart(selected\_categories, start\_date, end\_date):  
 filtered\_data = data[(data['B\_30'].isin(selected\_categories)) &  
 (data['S\_2'] >= pd.to\_datetime(start\_date)) &  
 (data['S\_2'] <= pd.to\_datetime(end\_date))]  
 # Map category identifiers to names  
 filtered\_data['Category\_Name'] = filtered\_data['B\_30'].map({  
 0.0: 'Stable Balance',  
 1.0: 'Variable Balance',  
 2.0: 'At-Risk Balance'  
 })  
 grouped\_data = filtered\_data.groupby('Category\_Name')['B\_2'].sum().reset\_index()  
 fig = px.pie(  
 grouped\_data,  
 values='B\_2',  
 names='Category\_Name',  
 title='Total B\_2 Value by Balance Category Within Selected Dates',  
 color='Category\_Name',  
 color\_discrete\_map=category\_colors # Using specific colors for each category  
 )  
 fig.update\_traces(textposition='inside', textinfo='percent+label')  
 return fig  
  
def render\_delinquency\_analysis\_page():  
 return html.Div([  
 html.Div([  
 html.H2("Dependency of Delinquency on Balance and Payment Behavior", style={'textAlign': 'center'}),  
 html.Div([  
 html.Label('Checking account balance Range:'),  
 dcc.RangeSlider(  
 id='b1-range-slider',  
 min=data['B\_1'].min(),  
 max=data['B\_1'].max(),  
 value=[data['B\_1'].min(), data['B\_1'].max()],  
 marks={i: f'{i:.2f}' for i in np.linspace(data['B\_1'].min(), data['B\_1'].max(), 5)},  
 step=(data['B\_1'].max() - data['B\_1'].min()) / 100  
 ),  
 html.Label('Saving account balance Range:'),  
 dcc.RangeSlider(  
 id='b2-range-slider',  
 min=data['B\_2'].min(),  
 max=data['B\_2'].max(),  
 value=[data['B\_2'].min(), data['B\_2'].max()],  
 marks={i: f'{i:.2f}' for i in np.linspace(data['B\_2'].min(), data['B\_2'].max(), 5)},  
 step=(data['B\_2'].max() - data['B\_2'].min()) / 100  
 ),  
 html.Label('Payment factor Range:'),  
 dcc.RangeSlider(  
 id='p2-range-slider',  
 min=data['P\_2'].min(),  
 max=data['P\_2'].max(),  
 value=[data['P\_2'].min(), data['P\_2'].max()],  
 marks={i: f'{i:.2f}' for i in np.linspace(data['P\_2'].min(), data['P\_2'].max(), 5)},  
 step=(data['P\_2'].max() - data['P\_2'].min()) / 100  
 ),  
 ], style={'padding': '20px'}),  
 dcc.Graph(id='delinquency-scatter-plot'),  
 dbc.Button("Download Data", id="btn-download", color="primary", className="me-1", style={'margin': '10px'}),  
 dcc.Download(id="download-dataframe-csv")  
 ], style={'padding': '20px', 'border': '1px solid #ccc', 'border-radius': '5px'}),  
 ], style={'padding': '20px'})  
  
# Callback to update the scatter plot  
@app.callback(  
 Output('delinquency-scatter-plot', 'figure'),  
 [Input('b1-range-slider', 'value'),  
 Input('b2-range-slider', 'value'),  
 Input('p2-range-slider', 'value')]  
)  
def update\_graph(b1\_range, b2\_range, p2\_range):  
 global filtered\_data   
 filtered\_data = data[  
 (data['B\_1'].between(\*b1\_range)) &  
 (data['B\_2'].between(\*b2\_range)) &  
 (data['P\_2'].between(\*p2\_range))  
 ]  
  
 fig = px.scatter(  
 filtered\_data,  
 x='P\_2',  
 y='D\_39',   
 color='D\_39',  
 title="Dependency of Delinquency on Balance and Payment Behavior",  
 labels={"P\_2": "Payment Behavior (P\_2)", "D\_39": "Delinquency Measure"}  
 )  
 return fig  
  
# Callback to handle the download action  
@app.callback(  
 Output("download-dataframe-csv", "data"),  
 Input("btn-download", "n\_clicks"),  
 prevent\_initial\_call=True  
)  
def download\_data(n\_clicks):  
 if n\_clicks is None:  
 raise dash.exceptions.PreventUpdate  
 return dcc.send\_data\_frame(filtered\_data.to\_csv, "filtered\_data.csv")  
  
  
  
def render\_balance\_payment\_dynamics\_page():  
 return html.Div([  
 html.H1('Balance and Payment Dynamics', style={'textAlign': 'center'}),  
  
 dcc.Tabs([  
 dcc.Tab(label='Balance and Spend Dynamics', children=[  
 html.Div([  
 dcc.Graph(id='balance-spend-bubble-chart'),  
 dcc.DatePickerRange(  
 id='bubble-chart-date-picker-range',  
 min\_date\_allowed=data['S\_2'].min(),  
 max\_date\_allowed=data['S\_2'].max(),  
 start\_date=data['S\_2'].min(),  
 end\_date=data['S\_2'].max(),  
 style={'padding': '10px', 'margin': '10px'}  
 ),  
 html.Div([  
 html.Label('Select B\_1 Range:'),  
 dcc.RangeSlider(  
 id='b1-slider',  
 min=data['B\_1'].min(),  
 max=data['B\_1'].max(),  
 value=[data['B\_1'].min(), data['B\_1'].max()],  
 marks={i: f'{i:.2f}' for i in np.linspace(data['B\_1'].min(), data['B\_1'].max(), 5)},  
 step=(data['B\_1'].max() - data['B\_1'].min()) / 100,  
 tooltip={"placement": "bottom", "always\_visible": True}  
 ),  
 html.Label('Select P\_2 Range:'),  
 dcc.RangeSlider(  
 id='p2-slider',  
 min=data['P\_2'].min(),  
 max=data['P\_2'].max(),  
 value=[data['P\_2'].min(), data['P\_2'].max()],  
 marks={i: f'{i:.2f}' for i in np.linspace(data['P\_2'].min(), data['P\_2'].max(), 5)},  
 step=(data['P\_2'].max() - data['P\_2'].min()) / 100,  
 tooltip={"placement": "bottom", "always\_visible": True}  
 ),  
 html.Label('Select S\_3 Range:'),  
 dcc.RangeSlider(  
 id='s3-slider',  
 min=data['S\_3'].min(),  
 max=data['S\_3'].max(),  
 value=[data['S\_3'].min(), data['S\_3'].max()],  
 marks={i: f'{i:.2f}' for i in np.linspace(data['S\_3'].min(), data['S\_3'].max(), 5)},  
 step=(data['S\_3'].max() - data['S\_3'].min()) / 100,  
 tooltip={"placement": "bottom", "always\_visible": True}  
 ),  
 ], style={'padding': '20px'})  
 ])  
 ]),  
  
 dcc.Tab(label='Area Chart Dynamics', children=[  
 html.Div([  
 dcc.Dropdown(  
 id='variable-area-dropdown',  
 options=[  
 {'label': 'Spend (S\_3)', 'value': 'S\_3'},  
 {'label': 'Payment (P\_2)', 'value': 'P\_2'},  
 {'label': 'Delinquency (D\_39)', 'value': 'D\_39'},  
 {'label': 'Risk Factor (R\_1)', 'value': 'R\_1'}  
 ],  
 value='S\_3', # Default selected variable  
 style={'width': '50%', 'padding': '10px'}  
 ),  
 dcc.Graph(id='area-chart')  
 ])  
 ])  
 ]),  
  
 dcc.Link('Back to Home', href='/', style={'marginTop': '20px', 'display': 'block'})  
 ])  
  
@app.callback(  
 Output('balance-spend-bubble-chart', 'figure'),  
 [Input('bubble-chart-date-picker-range', 'start\_date'),  
 Input('bubble-chart-date-picker-range', 'end\_date'),  
 Input('b1-slider', 'value'),  
 Input('p2-slider', 'value'),  
 Input('s3-slider', 'value')]  
)  
def update\_balance\_spend\_bubble\_chart(start\_date, end\_date, b1\_range, p2\_range, s3\_range):  
 # Filter data based on date range and slider values  
 filtered\_data = data[  
 (data['S\_2'] >= pd.to\_datetime(start\_date)) &  
 (data['S\_2'] <= pd.to\_datetime(end\_date)) &  
 (data['B\_1'].between(\*b1\_range)) &  
 (data['P\_2'].between(\*p2\_range)) &  
 (data['S\_3'].between(\*s3\_range))  
 ]  
  
 filtered\_data['S\_3\_abs'] = filtered\_data['S\_3'].abs()  
  
 # Create the bubble chart  
 fig = px.scatter(  
 filtered\_data,  
 x='B\_1',   
 y='P\_2',   
 size='S\_3\_abs',   
 color='R\_1',   
 hover\_data=['B\_1', 'P\_2', 'S\_3', 'R\_1'],  
 title='Relationship between Balance, Payment, Spend, and Risk'  
 )  
 return fig  
  
@app.callback(  
 Output('area-chart', 'figure'),  
 [Input('variable-area-dropdown', 'value')]  
)  
def update\_area\_chart(selected\_variable):  
 # Define bin edges for B\_2  
 bins = pd.interval\_range(start=data['B\_2'].min(), end=data['B\_2'].max(), freq=(data['B\_2'].max() - data['B\_2'].min()) / 10)  
  
 # Create a new 'Binned\_B2' column  
 data['Binned\_B2'] = pd.cut(data['B\_2'], bins=bins)  
  
 # Aggregate data by the new binned column  
 aggregated\_data = data.groupby('Binned\_B2')[selected\_variable].mean().reset\_index()  
  
 # Convert the IntervalIndex to string for better plotting and format to 2 decimal places  
 aggregated\_data['Binned\_B2'] = aggregated\_data['Binned\_B2'].apply(lambda x: f"{x.left:.2f} to {x.right:.2f}")  
  
 fig = px.area(  
 data\_frame=aggregated\_data,  
 x='Binned\_B2',  
 y=selected\_variable,  
 title=f'Area Chart of {selected\_variable} over Binned Saving Account',  
 labels={  
 'Binned\_B2': 'Saving Account (Binned)',  
 selected\_variable: f'Mean {selected\_variable}'  
 }  
 )  
 variable\_names = {  
 'S\_3': 'Spend Variable',  
 'P\_2': 'Payment Variable',  
 'D\_39': 'Delinquency Measure',  
 'R\_1': 'Risk Factor'  
 }  
  
 fig.update\_layout(  
 template='plotly\_white',  
 xaxis\_title='Saving Account Ranges',  
 yaxis\_title=f'Mean {variable\_names.get(selected\_variable, selected\_variable)}',  
 font=dict(  
 family="Arial, sans-serif",  
 size=12,  
 color="RebeccaPurple"  
 )  
 )  
 fig.update\_xaxes()  
  
 return fig  
  
columns\_to\_plot = ['S\_3', 'P\_2', 'D\_39', 'B\_1', 'B\_2', 'R\_1']  
numeric\_df = data[columns\_to\_plot]  
corr\_matrix = numeric\_df.corr()  
  
np.fill\_diagonal(corr\_matrix.values, np.nan)  
max\_corr\_value = corr\_matrix.unstack().dropna().abs().idxmax()  
max\_corr\_text = f"Max correlation is between {max\_corr\_value[0]} and {max\_corr\_value[1]}."  
  
  
heatmap\_figure = go.Figure(data=go.Heatmap(  
 x=columns\_to\_plot,  
 y=columns\_to\_plot,  
 z=corr\_matrix.values,  
 text=np.around(corr\_matrix.values, decimals=2),  
 texttemplate="%{text}",  
 hoverinfo='none',  
 colorscale='RdBu',  
 reversescale=True,  
 showscale=True  
),  
layout=go.Layout(  
 title='Correlation Heatmap of Variables'  
))  
  
variables = ['B\_1', 'B\_2', 'R\_1', 'P\_2', 'D\_39', 'S\_3']  
def render\_predictive\_insights\_page():  
 return html.Div([  
 html.H1("Predictive Insights", style={'text-align': 'center'}),  
 dcc.Tabs(id="stats-tabs", children=[  
 dcc.Tab(label='PCA Analysis', children=[  
 html.Button("Run PCA", id='run-pca-button'),  
 dcc.Graph(id='scree-plot'),  
 dcc.Graph(id='component-plot')  
 ]),  
  
 dcc.Tab(label='Normality Test', children=[  
 html.Div([  
 html.H1("Normality Tests"),  
 html.Button("Run Normality Test", id='run-normality-test-button', n\_clicks=0),  
 html.Div(id='normality-test-result'),  
 dcc.Graph(id='qq-plot'),  
 html.Div(id='normality-test-result')  
 ])  
 ]),  
 dcc.Tab(label='Heatmap & Correlation', children=[  
 html.Div([  
 html.H1("Correlation Heatmap with Annotations"),  
 dcc.Graph(  
 id='correlation-heatmap',  
 figure=heatmap\_figure  
 ),  
 html.Div([  
 html.P(max\_corr\_text, style={'fontSize': 16, 'marginTop': '10px', 'color': 'green'})  
 ], style={'textAlign': 'center'}) # Text displayed below the heatmap  
 ])  
 ]),  
 dcc.Tab(label='Outlier Detection', children=[  
 html.Div([  
 dcc.Graph(id='box-plot'),  
 html.Button("Remove Outliers", id='remove-outliers-btn', n\_clicks=0)  
 ])  
 ]),  
 dcc.Tab(label='Statistics & Density Estimates', children=[  
 html.Div([  
 html.H1("Statistics and Density Estimates"),  
 dcc.RadioItems(  
 id='variable-selector',  
 options=[{'label': var, 'value': var} for var in variables if var in data.columns],  
 value='B\_1', # Default value; ensure 'B\_1' is in 'data'  
 labelStyle={'display':'block'}  
 ),  
 html.Div(id='output-stats'),  
 dcc.Graph(id='variable-graph')  
 ])  
 ])  
 ]),  
 dcc.Link('Back to Home', href='/', style={'marginTop': '20px', 'display': 'block'})  
 ])  
  
@app.callback(  
 Output('box-plot', 'figure'),  
 [Input('remove-outliers-btn', 'n\_clicks')],  
 prevent\_initial\_call=False  
)  
def update\_box\_plot(n\_clicks=0):  
 filtered\_data = data.copy()   
 if n\_clicks > 0:  
 Q1 = filtered\_data['S\_3'].quantile(0.25)  
 Q3 = filtered\_data['S\_3'].quantile(0.75)  
 IQR = Q3 - Q1  
 multiplier = 2.5   
 filtered\_data = filtered\_data[(filtered\_data['S\_3'] >= (Q1 - multiplier \* IQR)) & (filtered\_data['S\_3'] <= (Q3 + multiplier \* IQR))]  
  
 z\_scores = np.abs((filtered\_data['S\_3'] - filtered\_data['S\_3'].mean()) / filtered\_data['S\_3'].std())  
 filtered\_data = filtered\_data[z\_scores < 3]   
  
 fig = px.box(  
 filtered\_data,  
 x='B\_30',  
 y='S\_3',  
 title="Box Plot of Spend Variable vs Categories",  
 labels={'B\_30': 'Category', 'S\_3': 'Spend Variable'}  
 )  
 return fig  
  
  
@app.callback(  
 [Output('scree-plot', 'figure'),  
 Output('component-plot', 'figure')],  
 [Input('run-pca-button', 'n\_clicks')]  
)  
def run\_pca\_analysis(n\_clicks):  
 if n\_clicks is None or n\_clicks <= 0:  
 raise dash.exceptions.PreventUpdate  
  
   
 selected\_columns = ['S\_3', 'P\_2', 'D\_39', 'B\_1', 'B\_2', 'R\_1']  
 pca\_data = data[selected\_columns]  
  
 scaled\_data = StandardScaler().fit\_transform(pca\_data)  
  
 pca = PCA()  
 pca.fit(scaled\_data)  
 pca\_scores = pca.transform(scaled\_data)  
  
 scree\_fig = px.bar(  
 x=[f'PC{i + 1}' for i in range(len(pca.explained\_variance\_ratio\_))],  
 y=pca.explained\_variance\_ratio\_,  
 labels={'x': 'Principal Component', 'y': 'Variance Explained'},  
 title='Scree Plot',  
 text=pca.explained\_variance\_ratio\_  
 )  
 scree\_fig.update\_traces(marker\_color='rgba(100, 149, 237, 0.7)', marker\_line\_color='rgb(25, 25, 112)',  
 marker\_line\_width=1.5, opacity=0.6, texttemplate='%{text:.2%}', textposition='outside')  
 scree\_fig.update\_layout(plot\_bgcolor='white', xaxis=dict(title='Principal Components'),  
 yaxis=dict(title='Proportion of Variance Explained', tickformat='.2%'),  
 title\_x=0.5, title\_font=dict(size=20, color='rgb(25, 25, 112)'),  
 font=dict(family='Arial, sans-serif'))  
  
 components = pd.DataFrame(pca\_scores, columns=[f'PC{i + 1}' for i in range(len(pca.explained\_variance\_ratio\_))])  
 component\_fig = px.scatter(  
 components, x='PC1', y='PC2',  
 title='PCA Component Plot',  
 hover\_data={i: False for i in components.columns} # Disable hover for all columns; enable as needed  
 )  
 component\_fig.update\_traces(marker=dict(size=10, opacity=0.8, line=dict(width=0.5, color='DarkSlateGrey')))  
 component\_fig.update\_layout(plot\_bgcolor='white', xaxis=dict(title='First Principal Component'),  
 yaxis=dict(title='Second Principal Component'),  
 title\_x=0.5, title\_font=dict(size=20, color='rgb(25, 25, 112)'),  
 font=dict(family='Arial, sans-serif'))  
  
 return scree\_fig, component\_fig  
  
  
@app.callback(  
 [Output('qq-plot', 'figure'),  
 Output('normality-test-result', 'children')],  
 [Input('run-normality-test-button', 'n\_clicks')],  
 prevent\_initial\_call=True  
)  
def update\_normality\_tests(n\_clicks):  
 if n\_clicks > 0:  
  
 data\_sample = data['S\_3'].dropna()   
  
 shapiro\_stats, shapiro\_p = stats.shapiro(data\_sample)  
  
 ks\_stats, ks\_p = stats.kstest(data\_sample, 'norm', args=(np.mean(data\_sample), np.std(data\_sample)))  
  
 qq\_plot\_fig = go.Figure()  
 qq = stats.probplot(data\_sample, dist="norm")  
 qq\_plot\_fig.add\_trace(go.Scatter(x=qq[0][0], y=qq[0][1], mode='markers', name='Data Points'))  
 qq\_plot\_fig.add\_trace(go.Scatter(x=qq[0][0], y=qq[1][1] + qq[1][0] \* qq[0][0], mode='lines', name='Fit Line'))  
 qq\_plot\_fig.update\_layout(title="Q-Q Plot", xaxis\_title="Theoretical Quantiles", yaxis\_title="Sample Quantiles")  
  
 results\_text = [  
 html.P(f"Shapiro-Wilk Test Statistic: {shapiro\_stats:.4f}, p-value: {shapiro\_p:.4f}"),  
 html.P(f"Kolmogorov-Smirnov Test Statistic: {ks\_stats:.4f}, p-value: {ks\_p:.4f}"),  
 ]  
  
 if shapiro\_p < 0.05 or ks\_p < 0.05:  
 results\_text.append(  
 html.P("The data does not appear to follow a normal distribution.", style={'color': 'red'}))  
 else:  
 results\_text.append(html.P("The data appears to follow a normal distribution.", style={'color': 'green'}))  
  
 return qq\_plot\_fig, results\_text  
 else:  
 # Return empty if not clicked yet  
 return go.Figure(), "Click the button to run the normality test."  
  
@app.callback(  
 Output('density-plot', 'figure'),  
 [Input('variable-dropdown', 'value')]  
 )  
def update\_density\_plot(variable):  
 fig = ff.create\_distplot([data[variable].dropna()], [variable], bin\_size=.2)  
 return fig  
  
def update\_statistics(variable):  
 data\_series = data[variable].dropna()  
 desc\_stats = data\_series.describe()  
 stats\_text = f"Mean: {desc\_stats['mean']:.2f}, Std: {desc\_stats['std']:.2f}"  
 return stats\_text  
  
@app.callback(  
 [Output('output-stats', 'children'),  
 Output('variable-graph', 'figure')],  
 [Input('variable-selector', 'value')]  
)  
def update\_content(selected\_variable):  
 stats\_text = update\_statistics(selected\_variable)  
 stats\_div = html.Div(stats\_text)  
  
 # Generate graph  
 data\_series = data[selected\_variable].dropna()  
 fig = px.histogram(data\_series, title=f'Distribution of {selected\_variable}')  
  
 return stats\_div, fig  
  
  
def render\_segmentation\_analysis\_page():  
 return html.Div([  
 html.H1('Payment Behavior Over Time by Balance Category'),  
 dcc.Graph(id='payment-behavior-graph'),  
 html.Div([dcc.Link('Back to Home', href='/')])  
 ])  
  
# Add similar functions for other pages  
  
@app.callback(Output('page-content', 'children'),  
 [Input('url', 'pathname')])  
  
def display\_page(pathname):  
 print("Received pathname:", pathname)  
 if not pathname or pathname == '/':  
 return index\_page   
 elif pathname == '/payment-behavior':  
 return render\_payment\_behavior\_page()  
 elif pathname == '/delinquency-analysis':  
 return render\_delinquency\_analysis\_page()  
 elif pathname == '/balance-payment-dynamics':  
 return render\_balance\_payment\_dynamics\_page()  
 elif pathname == '/predictive-insights':  
 return render\_predictive\_insights\_page()  
 elif pathname == '/segmentation-analysis':  
 return render\_segmentation\_analysis\_page()  
 else:  
 return '404'   
  
if \_\_name\_\_ == '\_\_main\_\_':  
 app.run\_server(debug=True)

**References:  
"Predictive Analytics with Microsoft Azure Machine Learning" by Valentine Fontama, Roger Barga, and Wee Hyong Tok.**

**"Python for Data Analysis" by Wes McKinney.**

**"Interactive Applications Using Matplotlib and Dash" by R. T. Pringadi.**

**"Dash documentation by Plotly" - Official Documentation.**