Automating Fraud Analytics Common Point of Compromise Analysis

Western Governors University

Table of Contents

Project Overview

A. Project Highlights

A1. Research question:

The research question this project aimed to answer is whether a standalone application could be built to automate a frequently used manual fraud analytic process. This common analytic technique, called a CPC or Common Point of Compromise analysis, is used in fraud departments multiple times per day. The purpose of this analysis is to ascertain additional information from card transaction history to predict additional cards that may be at a higher risk of experiencing fraud in the future. This is done by determining where fraudulent cards may have been obtained by the fraudsters.

A2. Project Scope:

This project's scope was to create a standalone application that would automate a routine, manual, and time-consuming fraud analytics process. The application would take an input of card numbers and then output specific information related to a potential merchant compromise, including merchant name, the window of compromise, and additional cards that transacted with the compromised merchant during the compromise window.

A3a. Solution Overview - Tools:

The application was coded in the Python programming language and relied on two key inputs from the end-user. The first input is an Excel file named Capstone.xlsx, which contains card transaction information in a predetermined format. The second key input is a series of card numbers. The application reads the Excel file, then prompts the user to input the card numbers into the application. The application then automatically performs the CPC analysis and outputs the compromised merchant, the date range for the compromise, and any additional card numbers that were used at the compromised merchant during the window of compromise.

A3b. Solution Overview – Methodologies

There were four different types of methodologies used in this project: Project, Data collection, Analytical, and Statistical.

Each methodology played a crucial role in the planning, execution, and verification of this project. Please see further detail and explanation of these methodologies in their corresponding sections below.

Project Plan

B. Project Execution

B1. Project Plan

In this project, I executed the plan without change. All goals, objectives, and deliverables listed below were completed exactly as described in task 2.

The goal of this project is to create a Python application to automate the fraud analytic Common Point of Purchase/Compromise analysis.
The objectives for this goal are:

- Determine which merchant is the source of compromise.
    - The deliverable for this objective is for the application to return the name of the compromised merchant.
- Determine the date range of the compromise.
    - The deliverable for this objective is for the application to return the date range when the merchant was compromised.
- Determine other payment cards that may be at risk of being compromised.
    - The deliverable for this objective is for the application to return a list of cards used at the compromised merchant during the compromise date range.

B2. Project Planning Methodology

The project methodology used was the Waterfall methodology. The five steps involved in this method include Requirements, Design, Implementation, Verification, and Maintenance. This methodology was chosen because of its key provision that each step must be completed before the next is attempted.

Requirements: In this step, I determined the scope of the project, gathered user requirements, and listed the resources needed to complete the project.

Design: In this step, I compiled the tasks that needed to be completed to achieve the project objectives. Some of these tasks included determining the best type of analytical methodology to use to detect compromised merchants, the steps needed to establish the correct date range of compromise, and where to locate the data which would need to be input and output.

Implementation: This step consisted of completing the tasks needed to achieve the project objectives and testing the solution to ensure it produced the desired results. I did this by testing the application on multiple different computers to ensure that an environment without the necessary coding software could reproduce the expected results.

Verification: This stage included creating a project template. When working with various financial institutions, this template could be used to implement the automated CPC application quickly and easily. This template will decrease the time needed to deploy this solution to the firms that need it most.

Maintenance: This stage was not completed during this project, as this solution is not currently deployed at any financial institution. In the future, this portion of the Waterfall methodology will be used to satisfy any contractual SLAs with these 3rd parties.

The execution of this project's methodology did not change from its initial conception to its completion.

B3. Project Timeline and milestones

The actual project timeline and milestones followed a pattern similar to what was initially proposed. The milestones were unchanged and completed as expected. The timeline was the portion of the project that experienced the most drastic changes; these changes are detailed below the table.

Milestone	**Projected Start Date**	**Projected End Date**	**Duration (days/hours)**
Establish requirements for analytics process	12/27/2020	12/27/2020	1 day
Develop system workflow	12/28/2020	12/28/2020	1 day
Create Testing Dataset	12/29/2020	12/30/2020	2 days
Code application	12/31/2020	1/2/2021	3 days
Test application	1/3/2021	1/3/2021	1 day
Create user best practices guidelines	1/4/2021	1/4/2021	1 day

The timeframe of this project changed in the following ways:

- Establishing requirements for the analytic process, Developing system workflow, and Creating Testing Dataset was completed faster than expected. The cumulative time for achieving those milestones was one day.
- Coding the application exceeded the expected duration of three days and instead required a total of five days.
- Testing the application exceeded the anticipated duration of one day and instead required a total of two days.
- Creating user guidelines was completed faster than expected by taking two hours instead of the allocated one day.

Methodology

C. Data Collection Process

- Actual data selection vs. planned collection process:

This project's data collection process faced numerous hurdles due to the extremely sensitive nature of payment card data. These hurdles necessitated that this data was not collected, but pseudo-randomly generated. The section below titled “Obstacles to data collection” further discusses the factors that led to this decision. The data was created as outlined in task 2 with only minor changes to the methodology outlined. One of the key differences between my planned process and the data selection that occurred was adding a single transaction to each

of the four cards that would be used as input for the application. Those cards each had a single transaction added to the end of their transaction history to simulate a pattern of fraudulent activity. This pattern of fraudulent activity is the key catalyst that would drive the analyst to use the application.

- Obstacles to data collection

  There were two primary obstacles to the data collection process for this project. The first is PCI DSS compliance, which prohibits the transmission or publication of unsecured payment card information. This factor led to the creation of a unique set of card numbers that did not overlap with possible existing numbers that could be issued by a financial institution. This project adhered to PCI DSS compliance by creating a new BIN (Bank Identification Number) that was not used by any other institution in the world. This process ensured there would not be any overlap now or in the future with actual card numbers.

  The second major obstacle to the data collection process for this project was ensuring no laws were broken through the inadvertent libel or slander of a business. This project could have been subjected to legal liability had it published the name of a merchant with a potential data breach. To ensure this did not occur, the project utilized fictitious merchant names that would not link an actual merchant to fabricated transaction history.

- Unplanned data governance handling

  Due to the data of this project being created, there were very few issues that needed to be considered when it came to the governance of this data. The most pressing data governance concern for this project comes from when this solution is used by third parties with their own data. This concern necessitates specific language in the user agreement that will forbid the user of this application from disclosing or using the application's output for anything other than internal breach mitigation purposes. Otherwise, a public release of the application's output could lead to liability exposure for both the user's organization and the application developer.

C1. Advantages and Limitations of Data Set

The advantages of this dataset are that it was created with this specific application in mind and satisfied the requirements for this CPC process to function optimally. A secondary advantage is this dataset reduced any prospect of inadvertently violating any laws or PCI DSS compliance concerns through the creation process.

The limitations of this dataset are in the form of the assumptions made through its creation. This dataset has assumed that a single breached merchant exists for a subset of cards with similar fraudulent activity.

Oftentimes, a single breached merchant is not the sole location where fraudulent cards originated. Fraudsters can mix cards of different breaches before they are sold on the black market. This could result in multiple compromises needing to be detected from similar fraud spend patterns.

D. Data Extraction and Preparation Processes

The tool used for this dataset's creation was Microsoft Excel. A list of fictitious card numbers was input, then using Excel's built-in randbetween formula, a number between 8 and 25 was assigned to the card to designate the number of transactions a particular card

would have in its history. Dates were then assigned to the transactions in ascending order per card number. The randbetween formula was again used to create an index number to assign the name of the fictitious merchant for a given transaction from a predefined list. Additional formulas such as vlookup were then used to fill in the associated merchant id and terminal id. Finally, four cards were selected randomly to serve as the cards which would serve as the input for the application. These four cards had a single transaction added to the end of their transaction history to indicate a confirmed fraudulent transaction. A manual CPC analysis was then run on this dataset to ensure a merchant compromise existed and additional cards were found to be used at that merchant during the compromise window to fulfill the requirements for this project to be able to produce the proper output.

E. Data Analysis Process

E1. Data Analysis Methods

The analytical method used in this project was descriptive. This method was most suitable for this project because the goal is to gain additional information from historical pieces of data. The data analysis technique used was data aggregation. Data aggregation was the most appropriate technique because it allows for multiple merchants and time frames to be analyzed to determine the most likely source where a fraudulent card was obtained.

E2. Advantages and Limitations of Tools/Techniques

The primary tool for the analytic technique used in this project was Python. The decision to use this tool was based on the robust development environment Python provides. This environment facilitated the analysis and manipulation of the data through its many packages such as pandas, openpyxl, os, and itertools. One of the primary limitations to using Python is that it does not perform operations as quickly as other programming languages; this can lead to problems with developing solutions that perform analysis across large datasets.

The primary technique of data analysis was data aggregation. This technique made it possible to determine the compromised merchant, date range, and additional cards at risk. On the other hand, this technique has limitations; it relies on historical data and thus the organization must conduct this type of analysis only after the fraud has both occurred and been reported. This limitation can lead to increased losses when compared to other mitigation strategies such as real-time anomaly detection.

E3. Application of Analytical Methods

This project applied the analytical methods and techniques above through the following steps:

- Creating a dataset of only transactions on cards that had seen fraud.
- Aggregating transactions by merchant and card where transactions were not indicated as fraudulent.
- Selecting only the merchant where all cards had transacted.
- Aggregating dates of card transactions at the compromised merchant.
- Iterating through every possible combination of date ranges.
- Filtering date range by the smallest window of compromise.
- Filtering full dataset to determine additional cards at risk.

There were a few key assumptions that factored into this analysis. The first assumption was that only one compromised merchant existed, and it would be detectable by all cards with a fraud transaction having transacted at the compromised merchant. The next assumption was that the window of compromise would be the smallest timeframe in which all four cards transacted at the compromised merchant. The final assumption was that the cards most likely to be at risk from the detected breach would transact with the merchant during the window of compromise. All of the assumptions listed above were verified before the start of application development by conducting an initial manual CPC analysis once the dataset was created.

Results

F. Project Success

F1. Statistical Significance

To determine if this project met its goals for a statistically significant reduction in the amount of time it takes to run a CPC analysis, I sent my mock dataset to five former colleagues. I asked that they each time themselves while conducting their own CPC analysis on my data. I used this sample to create a distribution of the amount of time it takes a human analyst to perform this analytics process. The mean time was 15.4 minutes, with a standard deviation of 2.154 minutes. I then ran my application five different times to record an average run time, which yielded 2.19 seconds, or .0365 minutes. To prove significance, I formulated two hypotheses. The first hypothesis stated this result is significant, and the second hypothesis was a null hypothesis that noted the result was not significant. I then set my significance level(α) to .01, which would indicate a 99% chance of correctly rejecting the null hypothesis when false. Using this criterion, the P-value must be < .01 to reject the null hypothesis. I then calculated the Z score using the following formula: $Z = (.0365 – 15.4) / 2.154$. This calculation resulted in a Z score of -7.1325. I then calculated the P-value through the socscistatistics.com calculator, which yielded a P-value < .00001. Since this P value was less than .01, the null hypothesis was rejected. This confirmed the original hypothesis; there is a statistically significant reduction in the amount of time needed to conduct this analysis when done through my application.

F2. Practical Significance

The practical significance of this application's reduction in time spent conducting CPC analysis means the user will reclaim an average of 15 minutes of work time per use of the application. In fraud departments around the world, saving 15 minutes will save organizations money in terms of the amount of fraud that can be stopped in that time. It will also allow analysts to devote their time to precious other fraud mitigation tasks. Suppose we extrapolate these results out over a larger timeframe, using a conservative estimate of one use of this application per day. The amount of time saved in a year is 5,621 minutes or 93.68 hours of work. From my experience working in a fraud department, it was quite common for the team to conduct three or more of these analyses per day. If we extrapolate this application running three times per day, this could save 281 working hours or just over seven weeks of full-time work per year.

F3. Overall Success

I view this application as an unbridled success. All four of the criteria laid out in task two were met. The application correctly returned the merchant name, date range, and all additional cards associated with the breach. The application also returned the correct information in a time that

was statistically significant when compared to this process done manually. An organization using this application could save an immense amount of time and money. This application's ability to detect compromises such as a card skimmer at an ATM or gas pump could allow a fraud department to streamline operations in a manner not possible with manual analysis. While this solution will not be an all-encompassing solution to compromise detection, it has demonstrated its ability to automate a time-consuming manual process. I look forward to seeing how this tool could help combat fraud in the coming years.

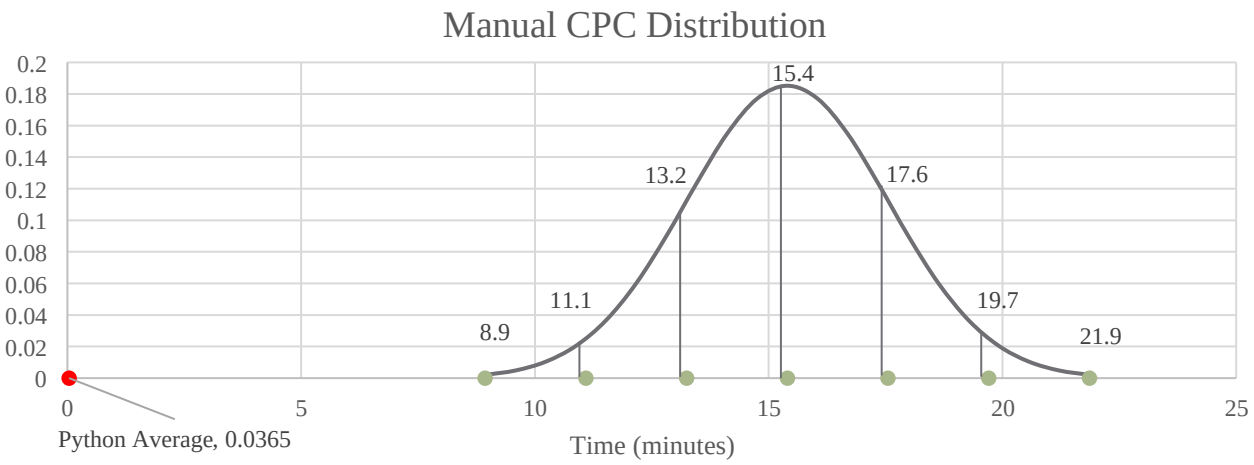
G. Key Takeaways

G1. Summary of Conclusions

This project set out to create an application that would return a merchant name, compromise window and set of additional compromised cards. The application also needed to complete this task in significantly less time than the task would take to complete manually. The following chart summarizes whether the application accomplished the first three objectives.

Metric	Answer	
	Yes	No
Did the application return the correct compromise name?		
Did the application return the correct compromise window?		
Did the application return the correct set of additional compromised cards?		

The graph below displays a bell curve for the distribution of the amount of time this analysis takes when done manually versus the average time the application took to run.



In this graph, we can see the fourth objective was achieved, as this reduction in the amount of time it takes to run this analysis is statistically significant.

Due to all four of these objectives having been achieved, we can conclude this project was a success.

G2. Effective Storytelling

The tools and visual representations chosen were the best methods for compelling storytelling because the primary three objectives were binary yes or no achievements. The best way to convey these objectives' satisfaction is through a table that displays the objective and whether it was achieved.
The bell curve graph is the best visual representation of statistical significance as it allows the viewer to see how far outside the curve the run time of this application happens to be.

G3. Findings-based Recommendations

The research question asked if an application could be created which would automate a manual fraud analysis process known as a CPC. This has been demonstrated to be possible. The two logical courses of action based on these findings are, first, to extend the testing on this application by allowing financial institutions to use their data to see if they could benefit from its use in their fraud operations. The second course of action is to expand the application's functionality to account for additional fringe CPC scenarios that may also benefit from automated detection. One such example of this type of fringe scenario would be a network-level compromise where multiple merchants use the same payment processor that has been compromised.

H. Panopto Presentation

My Panopto Presentation can be found at the following link:
<deleted>

Appendices

I. Evidence of Completion

The following files will serve as evidence of completion for this project.
1. Python code
2. Excel data file
3. Screenshot of executable output and expected result.

Sources

Statistics Calculators. (n.d.). Retrieved January 8, 2021, from https://www.socscistatistics.com/tests/