Complete Capstone: Analysis of Marketing Campaign Effectiveness

Brian Rodney

CSET, Grand Canyon University

DSC-590-O500: Data Science Capstone Project

Chintan Thakkar

December 21st, 2022

Table of Contents

PROJECT PROPOSAL 4

PROJECT INFORMATION 4
PROJECT OVERVIEW AND PROJECT OBJECTIVES 4
State the Problem *4*
Background *4*
Project Objectives *5*
Challenges *5*
Benefits and Opportunities *5*
PROJECT SCOPE 6
Work Breakdown Schedule *7*
PROJECT COMPLETION 7
Project Completion Criteria *7*
Assumptions and Constraints *8*
PROJECT CONTROLS 9
Risk Management *9*
Change Control Log *9*
Roles and Responsibilities *10*
PROJECT SCHEDULE 11
COST ESTIMATE 11
ISSUE LOG 11

REQUIREMENTS ANALYSIS 12

USE CASES 12
SYSTEM DESIGN 13
TECHNICAL REQUIREMENTS 15
DATA SCIENCE MODEL 17
REPORTS 17
SCREEN DEFINITIONS AND LAYOUTS 18
SECURITY 19

MODEL PIPELINE DESIGN 20

DESIGN PLANNING SUMMARY 20
OVERVIEW OF DESIGN CONCEPTS 20
DETAILED MODEL PIPELINE DESIGN 22
Overview *22*
Data Sources *23*
Dataset Types and Formatting *23*
Data Cleaning Procedures *23*
Data Exploration *24*
Data Model *24*
Methodology *25*
Configuration Changes *25*
Security *26*
HARDWARE AND SOFTWARE TECHNOLOGIES 26

IMPLEMENTATION 26

FUNCTIONAL REQUIREMENTS 26
SOURCE CODE 28
IMPLEMENTATION PLAN 35
Overview *35*

Assumptions, Dependencies, Constraints *36*
Operational Readiness *36*
Data Conversion *37*
Phased Rollout *37*
Support *39*
Release Planning *39*
Application Functionality and Execution *39*

PERFORMANCE ANALYSIS **40**

MODULE TEST CASES 40
REQUIREMENTS TESTING 41
SYSTEM TESTING 42

EVALUATION **42**

PROJECT REQUIREMENT RESULTS 42
PLANNED IMPROVEMENTS 43

PROJECT PRESENTATION **43**

USER MANUAL **44**

PREFACE 46
TABLE OF CONTENTS 47
GENERAL INFORMATION 48
SYSTEM SUMMARY 48
USING THE SYSTEM 49
Accessing the Dashboard *49*
Navigating the Dashboard *49*
Exporting Data *49*
TROUBLESHOOTING 49
FAQ (FREQUENTLY ASKED QUESTIONS) 50
HELP AND CONTACT DETAILS 50
GLOSSARY 50

SYSTEM ADMINISTRATION GUIDE **52**

PREFACE 54
TABLE OF CONTENTS 55
SYSTEM OVERVIEW 55
SYSTEM CONFIGURATION 55
SYSTEM MAINTENANCE 55
SECURITY RELATED PROCESSES 56

APPENDIX A – FIGURES **57**

APPENDIX B – REFERENCES **64**

Project Proposal

Project Information

Project name: Analysis of Marketing Campaign Effectiveness

Author: Brian Rodney

Project organization: n/a

Project manager: Brian Rodney

Date submitted: December 21st, 2022

Project Overview and Project Objectives

State the Problem

The purpose of this project is to perform statistical and exploratory analysis on recent marketing campaign data to identify the levels of effectiveness. The goal is to analyze the dataset to understand the effectiveness of global marketing campaigns and propose recommendations that will improve future operations.

Background

Every year, the use of data and its analysis in digital marketing expands. Companies that accept data and the critical analysis of their business models and workflows will initially see growth. Analytics, at its finest, provides companies with practical information into the marketing tasks that are performed daily. Company leaders know exactly how they are faring in terms of performance thanks to the right usage of data (Pepall & Richards, 2021). The project is being undertaken to illustrate how, with the appropriate exploratory and statistical analysis, marketing campaigns can be assessed and potentially replicated to ensure a company's continued marketing success. Without the ability to test and evaluate the success of marketing campaigns, companies

would have no idea what was working and what wasn't, when or if things needed to change, or how.

Project Objectives

- Build a basic customer profile from the data available.
- Identify which marketing campaign is the most successful.
- Determine if a marketing campaigns success is influenced by a geographic region.
- Identify which products are performing at higher levels.

Challenges

- Do any variables warrant transformation? Can the variables be organized into a more useful form?
- Are there any outliers or null values present? How will they be addressed?
- Does the data contain any correlated factors?

Benefits and Opportunities

The benefits resulting from a successful marketing analysis are overwhelming to any identified company. For instance, by building a basic customer profile and identifying the appropriate marketing channels, a company can begin to use a targeted marketing strategy to further increase its sales revenue potential (Pepall & Richards, 2021). An additional benefit of a successful analysis is it allows companies to be proactive rather than reactive in the scope of its market. This leads to less wasted resources and a higher chance of staying ahead of the trend curve (Pepall & Richards, 2021). Opportunities do exist and can be as simple as improperly interpreting data results or reviewing growth potential without digging into the supporting data sets (Hair Jr. & Sarstedt, 2021).

Project Scope

The project will commence once the data set has been exported from the host system. The team will begin by performing an exploratory data analysis where they will be responsible for wrangling any identified outliers, consolidating like variables, and explaining any potential correlation found within.

Following the EDA, a series of statistical tests in the form of regression modeling will begin to address the identified project objectives. The results will be plotted and visualized, accompanied by a layman explanation for ease of interpretation among the audience. Additional pertinent information that is not addressed in the project objectives may be included, depending upon the level of significance. The EDA and following analysis will not be used during this project for any other goal other than the identified objectives.

In Scope Features:

- EDA
- Regression modeling

Out of Scope Features:

- Unrelated predictions that do not address marketing.
- The addition of variables into the existing data set.

Work Breakdown Schedule

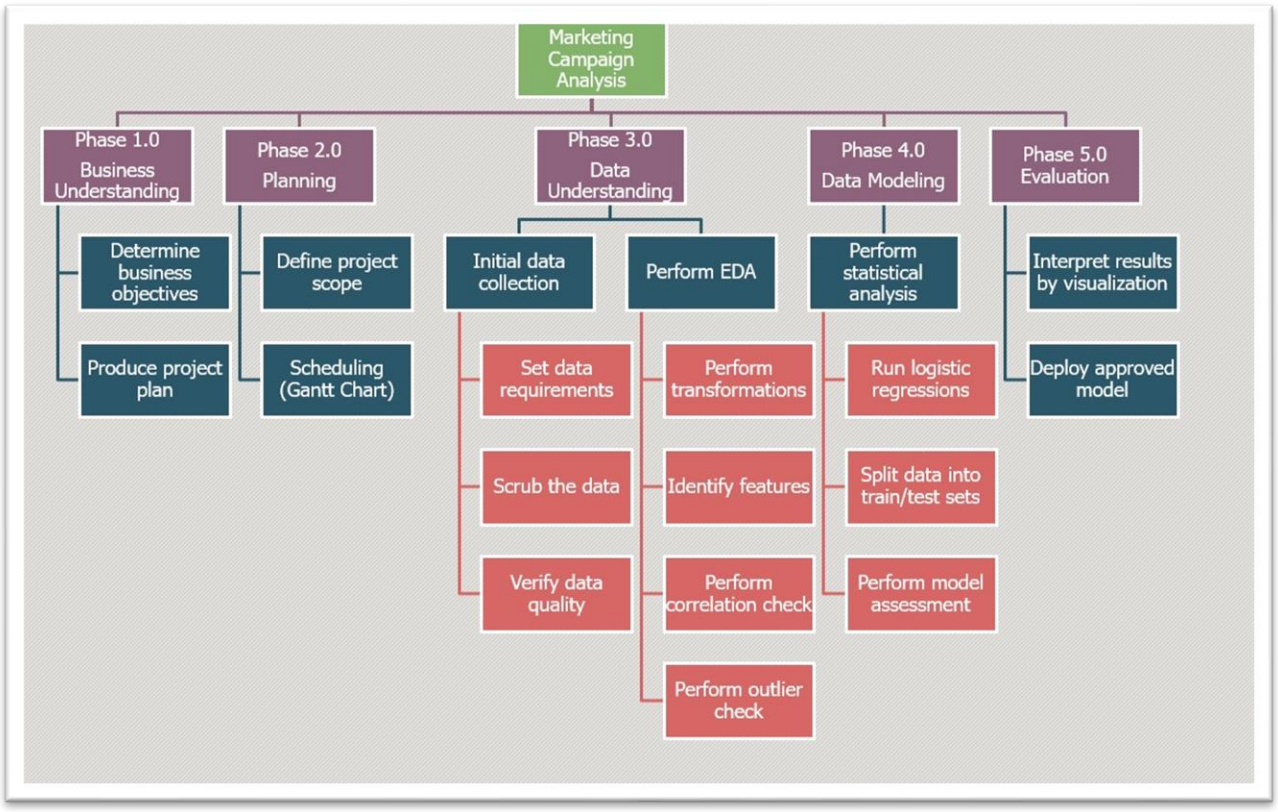


Figure 1. Work Breakdown Structure

The WBS in Figure 1 consists of 5 phases (Business Understanding, Planning, Data Understanding, Data Modeling, & Evaluation), each with applicable subphases that must be completed prior to moving onto the next per the waterfall methodology.

Project Completion

The project will be considered complete once all criteria has been met and the analysis has been completed. The criteria listed below is not in any specific order of prerequisite or importance.

Project Completion Criteria
1. The analysis provided results that inform which marketing campaign is the most successful.

2. Accepts or rejects the hypothesis.
3. The results of the project can be replicated.
4. Accomplishes all set forth project objectives.
5. The analysis completes within the approved timeline.

In addition to a set criterion for completion, the project is beholden to the assumptions and constraints that influence it.

Assumptions and Constraints					
ID	**Description**	**Comments**	**Type**	**Status**	**Date Entered**
1	Usable Data	The data is usable, even if it requires scrubbing	Assumption		01/03/2022
2	Programming Language	A compatible programming language is understood for EDA & statistical analysis.	Assumption		01/03/2022
3	Limited timeframe	The timeframe to complete the project to a relevant and satisfactory status is limited.	Constraint		01/12/2022

Project Controls

Risk Management				
Event Risk	**Risk Probability (high, medium,**	**Risk Impact**	**Risk Mitigation**	**Contingency Plan**
Data privacy violations	Medium	If the data contains PII, it could stop the project in the appropriate permissions are not obtained.	If the data contains PII, determine its relevancy and if there is minimal impact, remove it from the dataset.	If relevancy is high, seek the appropriate permissions or source another dataset.
Poor data quality	High	If the data quality is poor, it may extend the timeline for project completion or stall the project entirely.	The data needs to be collected from a quality and pertinent source, as it pertains to the project.	Ensure there is a backup dataset from the same organization in the event the original quality is subpar.
Incorrect analytical approach	High	If the data is processed using the wrong algorithm or analytical approach, the results would fall out of scope of the project.	The analytical approach o be used will be defined prior to project start and will be followed throughout its lifecycle.	If the incorrect methodology was used, determine if the results can be utilized to explain any of the objectives.
Results are rejected	Low	If the results are rejected, it would require either reanalysis of the dataset or redefining the project scope.	Ensure that all the criteria set by the stakeholders is clear and has been met before offering results.	Meet with the stakeholders to redefine objectives.

Change Control Log									
ID	**Change Description**	**Priority (C/H/M/L)**	**Originator**	**Date Entered**	**Date Assigned**	**Evaluator**	**Status**	**Date of Decision**	**Rev #**

1	Removed CC information from dataset	C	B. Rodney	01/19/2022	01/19/2022	Self	Approved	01/20/2022	1
2	New workflow option	M	B. Rodney	01/21/2022	01/22/2022	Self	Denied	01/23/2022	2
3	Updated documentation (objectives, constraints, reports)	C	B. Rodney	11/05/2022	11/05/2022	Self	Approved	11/06/2022	3

Roles and Responsibilities			
Name	Team	Project Role	Responsibility
Brian Rodney	GCU Capstone	Project manager, analyst	All work prescribed including data collection, processing, modeling, and reporting the results.

Project Schedule

Name	Duration	Start	Finish
Determine business objectives	1 day	1/10/22 8:00 AM	1/10/22 5:00 PM
Produce project plan	2 days	1/11/22 8:00 AM	1/12/22 5:00 PM
Define the project scope	4 days	1/11/22 8:00 AM	1/14/22 5:00 PM
Fill in a Gantt chart with the project schedule	1 day	1/17/22 8:00 AM	1/17/22 5:00 PM
Intial data collection	2 days	1/18/22 8:00 AM	1/19/22 5:00 PM
Data requirements	2 days	1/18/22 8:00 AM	1/19/22 5:00 PM
Scrubbing the data	5 days?	1/20/22 8:00 AM	1/26/22 5:00 PM
Verify the data quality	2 days	1/27/22 8:00 AM	1/28/22 5:00 PM
Perform data transformations	2 days	1/31/22 8:00 AM	2/1/22 5:00 PM
Perform exploratory data analysis	2 days?	2/2/22 8:00 AM	2/3/22 5:00 PM
Identify important features	1 day	2/4/22 8:00 AM	2/4/22 5:00 PM
Perform correlation check	1 day	2/7/22 8:00 AM	2/7/22 5:00 PM
Determine prevelance of outliers	1 day	2/7/22 8:00 AM	2/7/22 5:00 PM
Data modeling	2 days	2/8/22 8:00 AM	2/9/22 5:00 PM
Perform statistical analysis tests	3 days?	2/10/22 8:00 AM	2/14/22 5:00 PM
Run logistic regression models	2 days	2/15/22 8:00 AM	2/16/22 5:00 PM
Split data into training and test sets	1 day	2/17/22 8:00 AM	2/17/22 5:00 PM
Assess the model to determine relevance	1 day	2/18/22 8:00 AM	2/18/22 5:00 PM
Interpret results by visualization	2 days	2/21/22 8:00 AM	2/22/22 5:00 PM
Manage project deliverables for records and hand-off	35 days?	1/10/22 8:00 AM	2/25/22 5:00 PM
Assess and manage risks throughout the project life cycle	35 days?	1/10/22 8:00 AM	2/25/22 5:00 PM
Develop the final project report	3 days	2/23/22 8:00 AM	2/25/22 5:00 PM
Perform project reflections	1 day	2/23/22 8:00 AM	2/23/22 5:00 PM

Figure 2. Timeline of the Marketing Effectiveness Project

Cost Estimate

There is no cost associated with this project.

Issue Log

Issues Log								
ID	Description	Impact	Action Plan/Resolution	Owner	Importance	Date Entered	Date to Review	Date Resolved

1	Software failed to open properly; root file corrupted.	Severe	Located source of corruption (bad Windows update). Uninstalled and reinstalled Anaconda. Resolved.	Brian R	Critical	01/23/2022	01/23/2022	01/24/2022
2	Processing time of the regression models are higher than expected.	Medium	Review build to determine what steps can be removed or adjusted to cut down on processing time.	Brian R	Medium	01/25/2022	01/26/2022	IP

Requirements Analysis

Use Cases

Every organization that offers a product or service and contains a marketing department wants to know if they are doing the right thing and reaching out to the correct customers. A Chief Marketing Officer [CMO] may receive a report that a recent marketing campaign has not performed as anticipated and clarification would be required before launching another global marketing initiative. Without knowing the appropriate customer segment, product purchase history, marketing channel, or success rate of previous marketing campaigns, they would be potentially spending millions of dollars on a failed cause. The goal of this report is to produce insights that can be used to propose data-backed recommendations for improvements on the next

marketing initiative. The following steps would break down the process as it would be expected in this use case.

- The organization should provide a data file that contains details on the previous marketing campaigns, including:
    - Customer demographics
    - Products marketed
    - Campaign name
    - Marketing channel used
- The data will then be cleaned, modeled, and visualized using Python
- The resulting predictions and classifications will be used to identify which marketing campaign was successful and what components made up its framework.

This information produced by the use case can then be used by the organization to model future marketing campaigns and increase their overall effectiveness.

System Design

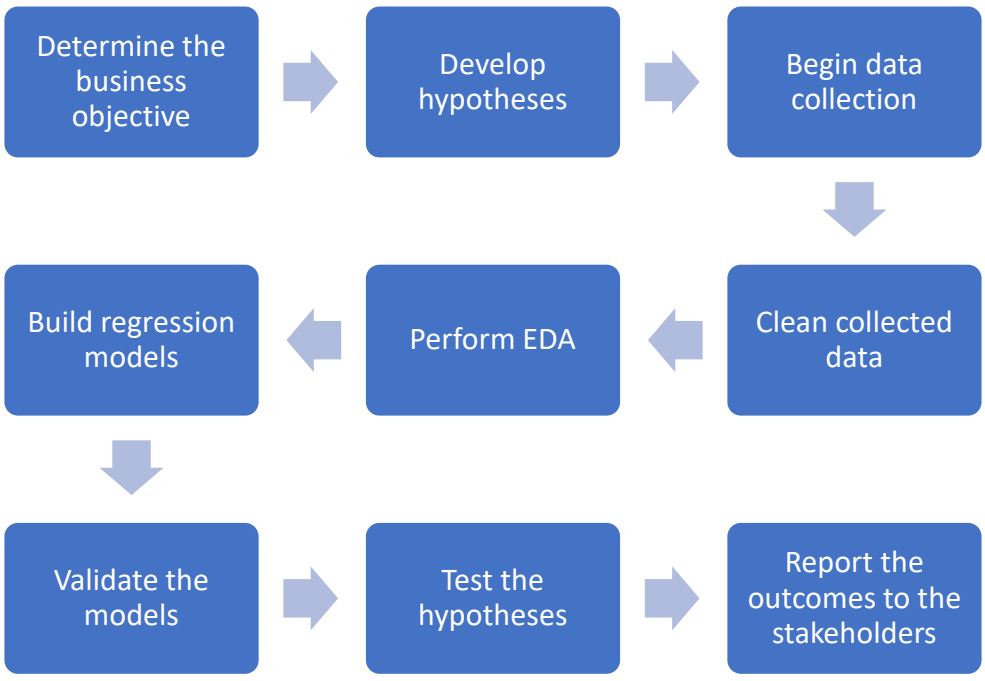


Figure 3. System design flowchart

1. Determine the business objective
    a. Without determining what the businesses end goal for the data analysis is, there is no direction and, by default, no project.
2. Develop hypotheses
    a. Using the identified business objectives, the researcher can develop a hypothesis or hypotheses in accordance with quantitative research design.
3. Begin data collection
    a. The data collection process will query and store primary data, as the project is focused on establishing the effectiveness of an internal workflow. Important variables will be earmarked as to move into the next process.
4. Clean collected data
    a. Redundant variables, outliers, and null values will be removed. Variables of interest are identified and confirmed while supporting variables will be transformed for ease of use.
5. Perform EDA
    a. Exploratory data analysis will commence to highlight any relationships between the variables. EDA will identify which characteristics might benefit the model, which features may be removed, and how missing values and correlations will be addressed.
6. Build regression models
    a. Data will be split into separate training and testing data sets. The initial model will be built following a regression methodology, where it will be trained with the respective data. This provides a building block for the validation step.
7. Validate the models

    a. Following the building of the models, they will be tested for predictive qualities. Additionally, the models will have the performance measured using statistical methods include MSE, MAE, RMSE, and confusion matrices, to name a few.

8. Test the hypotheses
    a. The resulting model information will provide the researcher enough information to accept or reject the null hypotheses defined previously.
9. Report the outcome
    a. Once all previous steps have been completed and the resulting data has been compiled, it will be assembled into a final report that will be presented to the project stakeholders.

Technical Requirements

1. Performance
    a. The code must compile in < 60 seconds.
    b. If the code compilation takes too long, it can negatively impact the recompilation time if changes are required to be made. Excessive wait times will add up and could overextend the project lifecycle.
2. Availability
    a. The dataset used will be hosted in Github & PythonAnywhere for public use.
    b. By hosting the dataset publicly, this will allow others to replicate the results.
3. Hardware requirements
    a. Operating System
        i. Windows 8.1 or later
        ii. Mac OS X 10.x or later
        iii. Linux-based OS

1. The OS options are open to whichever the user feels more comfortable in using. The identified software has builds for various OS applications, but there are minimum requirements associated with that software.

b. RAM and Processor

i. 16GB RAM or more

ii. 2GHz processor or higher

1. Due to the system power requirements to compile large projects, it is important to meet the identified minimum requirements as failure to do so may result in longer than usual processing times.

4. Software requirements

a. Python IDE or RStudio

i. As the project code is originally compiled in Python, it would be important that an applicable Python IDE (Jupyter, Spyder, PyCharm) is used.

ii. If Python is not used, the project may be written and compiled in R where RStudio would prove beneficial for use.

Data Science Model

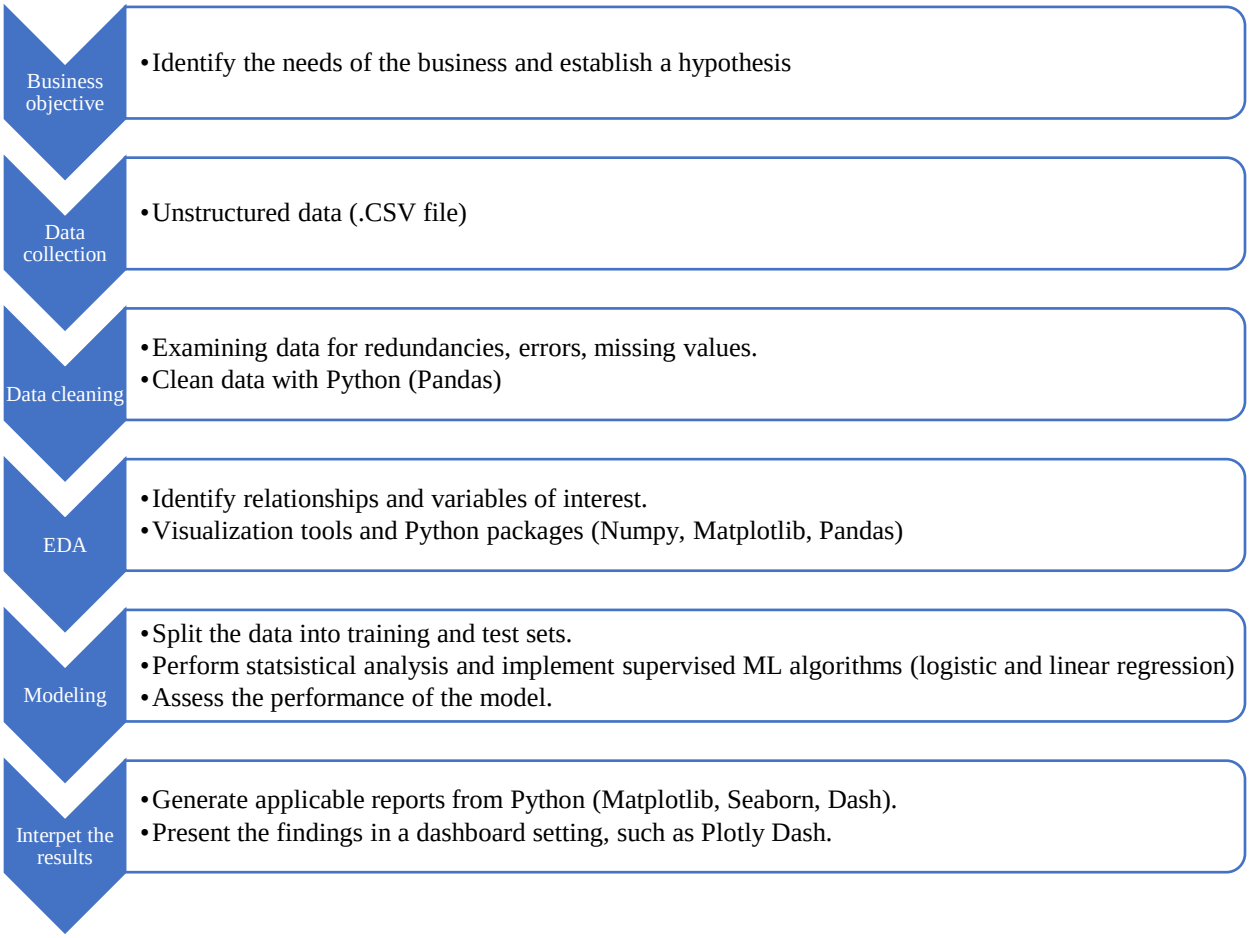


Figure 4. Data science pipeline model

Reports

To properly interpret and present the results of the analysis, several visualization templates will be used. Python offers visualization tools such as Plotly and Seaborn, both with their own specific functions. Plotly is a visualization library within Python, but the graphics produced are interactive. A choropleth map, one of Plotly's offerings, would prove useful in highlighting which countries responded best to which marketing campaigns (Plotly, 2022). An example of this type of this graphic is found in figure A8 of Appendix A. Seaborn is powered off matplotlib and integrates very well with the Pandas package of Python and produces statistical graphics including heatmaps and combination scatterplot matrices, as found in figures A6 & A7

(Waskom, 2012). A dashboard, built with Plotly Dash, will be used to display the noted visualizations with the capability of exporting data. In addition to a PDF report containing the analysis results, the dashboard product includes the ability to export the raw data file used for the visualization and subsequent statistical analysis processes. This export button, located in the upper right of the dashboard, will export the entirety of the data in an XLSX format. Further, if the user would like to preview the data prior to the export, a table format is available at the bottom of the dashboard.

Screen Definitions and Layouts

The project uses a Python dashboard package named Dash. This allows the project results to be displayed, reviewed, and exported by the user with minimal training. As the purpose of this dashboard product is to display the results of the analytical pipeline, there are to be no analytical elements included in the GUI. This includes the option to load data, perform data cleaning, EDA, or modify the built models. These have been excluded to simplify the dashboard GUI overall and match our audiences' expectations. The resulting data, however, will be displayed in various formats. For example, the marketing campaign success will be presented in a bar chart format whereas the goods purchased under each marketing campaign will be displayed in a pie chart. Another example will be the demographic information. The resulting demographics, or ideal customer profile, will be written to a table format, which will be available for download in a .XLSX format. A breakdown of the GUI elements follows:

Element	Purpose	Feature
Button [Export]	This button allows the user to compile and download a XLSX from the data table shown.	Generating reports

Bar plots	This provides a visualization option to review the EDA results.	Displaying data in various formats
Pie charts	This provides a visualization option to review the EDA results.	Displaying data in various formats

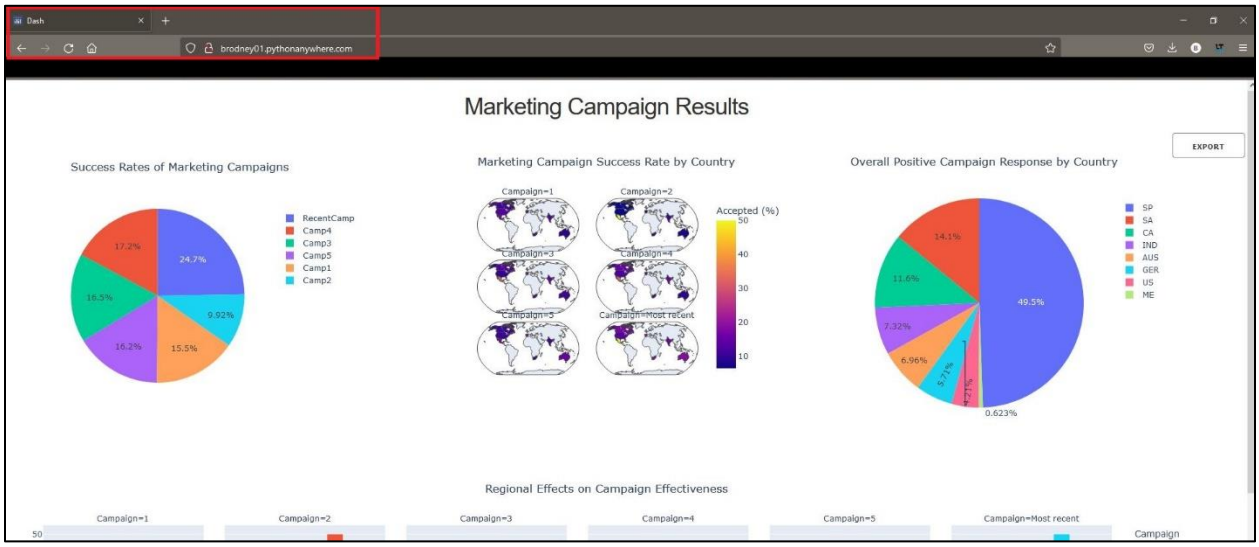


Figure 5. Dash dashboard concept

Security

While the project data is not expected to contain any PII, it does track the effectiveness of marketing campaigns for the customer base and therefore should be treated as confidential. The stakeholder's expectation is to host the dashboard on the company intranet, so all appropriate permissions and resources will need to be provided to the company IT prior to the full launch. Data security may be at risk when working with data from several sources with multiple users' access. When dealing with sensitive or secret data, data security is very crucial, so it is imperative that it is located, cataloged, and subjected to privacy rules. Implementing security controls and usage regulations on the data pipelines is a no brainer, so data governance is

expected to be in place (Munappy et al., 2020). It includes the actions people must take, the processes they must follow, and the technology that supports them throughout the data life cycle.

Model Pipeline Design

Design Planning Summary

The purpose of this project is to perform statistical and exploratory analysis on recent marketing campaign data to identify the levels of effectiveness. The goal is to analyze the dataset to understand the effectiveness of global marketing campaigns and propose recommendations that will improve future operations. The project is being undertaken to illustrate how, with the appropriate exploratory and statistical analysis, marketing campaigns can be assessed and potentially replicated to ensure a company's continued marketing success. Additionally, several objectives have been set to further flesh out the marketing portfolio including the creation of a customer profile, and gauging which products are performing best. Without knowing the appropriate customer segment, product purchase history, marketing channel, or success rate of previous marketing campaigns, an organization would be potentially spending millions of dollars targeting the wrong customer base. The goal of this report is to produce insights that can be used to propose data-backed recommendations for improvements on the next marketing initiative. The following steps would break down the process as it would be expected in this use case.

Overview of Design Concepts

The following information is a concept breakdown of the proposed project. The corresponding process chart may be found in Appendix A, Figure A1:

1. Determine the business objective
    a. Without determining what the businesses end goal for the data analysis is, there is no direction and, by default, no project.

2. Develop hypotheses
    a. Using the identified business objectives, the researcher can develop a hypothesis or hypotheses in accordance with quantitative research design.
3. Begin data collection
    a. The data collection process will query and store primary data, as the project is focused on establishing the effectiveness of an internal workflow. Important variables will be earmarked as to move into the next process.
4. Clean collected data
    a. Formatting and nomenclature will be harmonized. Redundant variables, outliers, and null values will be removed. Variables of interest are identified and confirmed while supporting variables will be transformed for ease of use.
5. Perform EDA
    a. Exploratory data analysis will commence to highlight any relationships between the variables. EDA will identify which characteristics might benefit the model, which features may be removed, and how missing values and correlations will be addressed.
6. Build regression models
    a. Data will be split into separate training and testing data sets. The initial model will be built following a regression methodology, where it will be trained with the respective data. This provides a building block for the validation step.
7. Validate the models

   a. Following the building of the models, they will be tested for predictive qualities. Additionally, the models will have the performance measured using statistical methods include MSE, MAE, RMSE, and confusion matrices, to name a few.

8. Test the hypotheses
   a. The resulting model information will provide the researcher enough information to accept or reject the null hypotheses defined previously.
9. Report the outcome
   a. Once all previous steps have been completed and the resulting data has been compiled, it will be assembled into a final report that will be presented to the project stakeholders.

Detailed Model Pipeline Design

Overview

The data set will be exported from a corporate data lake, where the original sample set is hosted for marketing analysis exercises. This file will then be hosted on a public site, where it may be used for replication or practice efforts. After the file has been loaded, simple formatting and nomenclature harmonization will take place prior to moving into an exploratory phase. Exploratory processes will commence, using correlation matrices, boxplots, and heatmaps to visualize any patterns or features of interest. The modeling of the data will be constructed following machine learning and regression methods, using a combination of classification and predictive models. Following the results of the models, a dashboard built with Plotly Dash, will be used to display the noted visualizations with the capability of exporting data. In addition to a PDF report containing the analysis results, the dashboard product includes the ability to export

the raw data file used for the visualization and subsequent statistical analysis processes. A high-level pipeline may be found in Figure A2.

Data Sources

The data set used for this project is a sample file, hosted on a corporate data lake. The data set contains no PII and is designed for the skill testing of new marketing analysts within the organization. As the file is listed as public view and does not violate corporate policies for sharing, the sample file has been exported and hosted on the GitHub repository for project duplication.

Dataset Types and Formatting

The dataset represents historical global marketing campaign results, in addition to general user demographics and purchase histories. The dataset type is written to a CSV file, containing 28 variables and 2556 observations. The observations are a combination of quantitative, date, and text, following a logical formatting throughout the dataset. For example, education levels are written out (i.e., Associate, PhD, etc.), timestamps represent the account creation, and income or product purchase histories are numerically formatted. The full list can be found in Figure A3.

Data Cleaning Procedures

Once the data has been loaded, it will need to be reviewed to ensure correct formatting and remove special characters, if required. To begin correcting the formatting, the column names will be checked and scrubbed of any extra whitespace [Figure A4]. This will harmonize the column headers which prevents errors by entering the incorrect values (i.e., " *Income*" fixed to "*Income*"). Continuing with the *Income* example, it was detected as an object type and included special characters [Figure A5]. This will be cleaned by removing the special characters and

correcting the format from object to float, as it is a numerical value. No other indications were noted that needed correction but there is room for variable transformations.

Data Exploration

The initial data exploration will be conducted by first determining if any null or empty values exist within the data set, and if identified, the values will be imputed with a median value to lessen the effects of outliers. In addition to running a null check, boxplots, histograms, heatmaps, and correlation matrices will be used for the initial EDA to further identify any outliers or correlated variables [Figure A6- A9]. The consolidation of variables of interest will be considered, as the engineering variables have the potential to provide further insight into any correlations or features of interest. Further EDA will be performed as necessary, including frequency checks, as they pertain to the set objectives of the project.

Data Model

OLS regression methods in addition to logistic regression models are used to model the data. Prior to running the OLS regression, we enact one-hot encoding of the categorial features using *sklearn.preprocessing.OneHotEncoder*. The cat types are assigned values and merged into the remaining numerical data using Pandas concat or *pd.concat* (pandas, 2022). We identified a few features that we wanted to run a linear regression against, however, the process is the same. First, we isolate the X and Y variables and split the data set into training and test data, using *sklearn.model_selection.test_train_split.* The data is split 70/30, leaving 30% of the data to be validated using RMSE. The model is built using *sklearn.linear_model.LinearRegression(),* fitted using the training data, and followed up with a prediction run using the X_test data. The OLS models are evaluated using RMSE in Numpy, as *np.sqrt(mean_squared_error(y_test,pred)).* The logistic regression model focused on the success of the campaigns by country and uses GLM to

produce the logistic regression p-values. Importing *smf* from *statsmodels* package allows us to build a GLM and calculate logistic regression supported by a bionomial family *model = smf.glm(formula = formula, data=data, family=sm.genmod.families.Binomial()).* The results give us a chisquare value that is used to calculate the p-values.

Methodology

As the choice algorithms will primarily be OLS regression and logistic regression, certain assumptions will need to be taken into consideration. One key assumption is there is no multicollinearity present between the variables, or plainly, the variables do not have a high level of correlation between them (Korstanje, 2022). This can be verified by reviewing the correlation matrix being deployed in the EDA phase of the project or running a VIF report to isolate and remove variable with a score > 10. An additional assumption specific to logistic regression is that there are no influential outliers. To verify this assumption, Cook's Distance can be utilized to flag any value with a CD > 1 (Thieme, 2022). Once the model results have been compiled, they will need to be validated. The goal is to utilize MAE, MSE, and RMSE metrics to evaluate the performance of the regression model. Further, a classification report function, found within the *sklearn* package, can generate an "all-in-one" readout for the classification problem. This includes metrics on the f1 score, precision, recall, averages, and accuracy, which can be further supported by generating a confusion matrix (Buitinck et al., 2013).

Configuration Changes

There are no configuration changes required to run this analysis. The data set is small in size (<500KB) and the code framework itself is not demanding of performance.

Security

While the project data is not expected to contain any PII, it does track the effectiveness of marketing campaigns for the customer base and therefore should be treated as confidential. The stakeholder's expectation is to host the dashboard on the company intranet, so all appropriate permissions and resources will need to be provided to the company IT prior to the full launch. Data security may be at risk when working with data from several sources with multiple users' access. When dealing with sensitive or secret data, data security is very crucial, so it is imperative that it is located, cataloged, and subjected to privacy rules. Implementing security controls and usage regulations on the data pipelines is a no brainer, so data governance is expected to be in place (Munappy et al., 2020). It includes the actions people must take, the processes they must follow, and the technology that supports them throughout the data life cycle.

Hardware and Software Technologies

Hardware and Software Technologies
1 – Windows 8.1 or later, Mac OS X 10.x or later, Linux-based OS
2 – 16GB RAM or more, 2GHz processor or higher
3 – Python programming language
4 – Python IDE (Spyder, Visual Code, PyCharm, etc.)

Implementation

Functional Requirements

Requirement	**Function**
Load dataset	*pd.read_csv()*
Check data types, values, and counts	*data.info()*
Read first five lines of data	*data.head()*

Clean column formatting	*columns.str.replace()*
Check for null values	*data.isna().sum()*
Remove special characters	*data[].str.replace()*
Change object type to float	*data.[].str.replace().astype('float')*
Remove outliers	*data[].reset_index(drop=True)*
Check for multicollinearity	*data.corr() or sns.clustermap()*
Produce variety of plots for numerical data	*sns.pairplot()*
Produce scatterplot	*sns.lmplot()*
Consolidate variables	*[col for col in data.columns if "xxx" in col]*
Encode categorial data	*sklearn.preprocessing.OneHotEncoder*
Split data into training and test sets	*sklearn.model_selection.test_train_split*
Build OLS model	*sklearn.linear_model.LinearRegression()*
Evaluate model using RMSE	*np.sqrt(mean_squared_error(y_test,pred))*
Build GLM model	*model = smf.glm(formula = formula, data=data, family=sm.genmod.families.Binomial()).*
Calculate percentage and create new dataframe	*new_data = pd.DataFrame(data[[]].mean()*100,columns=['Percent]).reset_index()*
Visualize campaign acceptance percent	*sns.barplot()*
Produce averages table and create new dataframe	*table = pd.DataFrame(round(data.drop(columns=ts_col+pur_col+cmp_col).mean(), 1)*
Create visualizations for Dash product	*Px.chloropeth(), px.pie(). Px.bar()*

Import CSS sheet for Dash	*external_stylesheets = ['https://codepen.io/chriddyp/pen/bWLwgP.css']*
Build Dash framework	*dash.Dash()*
Compile visualizations for Dash	*dcc.graph(id, figure, className)*
Build table for Dash	*dash_table.DataTable()*
Set header and layouts	*html.Div(), html.H2()*
Execute Dash product in sandbox	*if __name__ == '__main__':* *app.run_server(debug=False)*

Source Code

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import plotly.express as px
from sklearn.cluster import KMeans
from sklearn.metrics import accuracy_score, classification_report, confusion_
matrix, roc_auc_score, roc_curve
from sklearn import preprocessing
from sklearn.model_selection import train_test_split, cross_val_score
import warnings
warnings.filterwarnings('ignore')

import plotly.graph_objects as go

import dash
import dash_table
import dash_core_components as dcc
import dash_html_components as html
from dash.dependencies import Input, Output


sns.set_palette('icefire')
sns.set_style('darkgrid')
sns.set_context('paper')
```

Loading & Preprocessing Data

```
# Load dataset
data = pd.read_csv('C:/Users/BR/Desktop/mktcmpdata.csv')
```

```
print(data.info())
data.head()

# Clean any whitespace from column names
data.columns = data.columns.str.replace(' ', '')
data.head()

# Plot columns to check for outliers
dataplot = data.drop(columns=['Cust_ID', 'Camp1', 'Camp2',
                              'Camp3', 'Camp4', 'Camp5',
                              'RecentCamp', 'Complaints']).select_dtypes(incl
ude=np.number)
dataplot.plot(subplots=True, layout=(4,4), kind='box', figsize=(12,14),
              patch_artist=True)
plt.subplots_adjust(wspace=0.5)

# Removing erronous entries from Year of Birth [YOB] (<= 1900)
data = data[data['YOB'] > 1900].reset_index(drop=True)

plt.figure(figsize=(3,4))
data['YOB'].plot(kind='box', patch_artist=True)

# Checking for null values
data.isna().sum()
```

Income shows 30 null values. Prior to plotting to determine the imputation strategy, we will need to convert from object to float-type and remove the special character.

```
# Cleaning the Income feature

data['Income'] = data['Income'].str.replace('$', '')
data['Income'] = data['Income'].str.replace(',', '').astype('float')

plt.figure(figsize=(8,4))
sns.distplot(data['Income'], kde=False, hist=True)
plt.title('Income distribution', size=16)
plt.ylabel('count');
```

A couple of outliers were noted but nothing major. We will impute the null values with the median to reduce the effects of the outliers.

```
data['Income'] = data['Income'].fillna(data['Income'].median())
```

Transformation of Features

- The total number of dependents in the home ('Children') can be engineered from the sum of 'Kidhome' and 'Teenhome'
- The year of becoming a customer ('Account_Year') can be engineered from 'Orig_Customer'
- The total amount spent ('Total_Spent') can be engineered from the sum of all features containing the keyword 'Amt'

- The total purchases ('Total_Purchases') can be engineered from the sum of all features containing the keyword 'Purchase'
- The total number of campains accepted ('Total_Campaign') can be engineered from the sum of all features containing the keywords 'Camp'.

```
# Variable consolidation

ts_col = [col for col in data.columns if "Amt" in col]
pur_col = [col for col in data.columns if "Purchase" in col]
cmp_col = [col for col in data.columns if "Camp" in col]

data['Children'] = data['Kidhome'] + data['Teenhome']
data['Account_Year'] = pd.DatetimeIndex(data['Orig_Customer']).year
data['Total_Spent'] = data[ts_col].sum(axis=1)
data['Total_Purchases'] = data[pur_col].sum(axis=1)
data['Total_Campaign'] = data[cmp_col].sum(axis=1)

data[['Cust_ID', 'Children', 'Account_Year', 'Total_Spent', 'Total_Purchases'
, 'Total_Campaign']].head()
```

Correlation/Patterns of Interest

```
# Checking for correlation

corr = data.drop(columns='Cust_ID').select_dtypes(include=np.number).corr(met
hod = 'kendall')

sns.clustermap(corr, cbar_pos=(-0.05, 0.8, 0.05, 0.18), cmap='icefire', cente
r=0)

sns.lmplot(x='Income', y='Total_Spent', data=data[data['Income'] < 300000]);

plt.figure(figsize=(4,4))
sns.boxplot(x='Children', y='Total_Spent', data=data);

plt.figure(figsize=(4,4))
sns.displot(x='Children', y='CouponPurchase', data=data);
```

The above scatterplot shows the effects of income on total spending. The income was limited to < 300000 to remove outliers.

Data Visualizations

Highlighting the success rate of each marketing campaign. Based on the results, the most recent campaign (RecentCamp) was the most successful.

```
# Calculate success rate (percent accepted)
cam_success = pd.DataFrame(data[['Camp1', 'Camp2', 'Camp3', 'Camp4', 'Camp5',
'RecentCamp']].mean()*100,
                           columns=['Percent']).reset_index()
```

```
# Plot
sns.barplot(x='Percent', y='index', data=cam_success.sort_values('Percent'))
plt.xlabel('Accepted (%)')
plt.ylabel('Campaign')
plt.title('Marketing campaign success rate', size=16);
```

Determining which products performed the best. Nearly $300 was spent on meat products (AmtMeatProducts) with alcohol products (AmtAlcoholPurchases) at ~$165.

```
spending = pd.DataFrame(round(data[ts_col].mean(), 1), columns=['Average']).s
ort_values(by='Average').reset_index()

# Plot
ax = sns.barplot(x='Average', y='index', data=spending)
plt.ylabel('Amount spent on...')

# Add text labels for each bar value
for p,q in zip(ax.patches, spending['Average']):
    ax.text(x=q+40,
            y=p.get_y()+0.5,
            s=q,
            ha="center") ;
```

Highlighting the average customer profile. Based on the demographics of the customer base, the average customer is:

- Born in 1969
- Opened their account in 2017
- Has an annual income of ~$52K
- Has at least 1 child at home

```
# Average customer demographics
demographics = pd.DataFrame(round(data.drop(columns=ts_col+pur_col+cmp_col).m
ean(), 1), columns=['Average']).reindex([
    'YOB', 'Account_Year', 'Income', 'Children', 'Kidhome', 'Teenhome'])

demographics
```

Statistical Analysis

Determining if there is any significant relationship between geographic location and the marketing campaign success.

- Findings:
  - The campaign acceptance rates are low overall
  - The campaign with the highest overall acceptance rate is the most recent campaign (column name: `RecentCamp`)
  - The country with the highest acceptance rate in any campaign is Mexico

```
cam_success = pd.DataFrame(data[['Camp1', 'Camp2', 'Camp3', 'Camp4', 'Camp5',
'RecentCamp']].mean()*100,
                           columns=['Percent']).reset_index()

data['Country_code'] = data['Country'].replace({'SP': 'ESP', 'CA': 'CAN', 'US
': 'USA', 'SA': 'ZAF', 'ME': 'MEX'})

# success of campaigns by country code
df_cam = data[['Country_code', 'Camp1', 'Camp2', 'Camp3', 'Camp4', 'Camp5', '
RecentCamp']].melt(
    id_vars='Country_code', var_name='Campaign', value_name='Accepted (%)')
df_cam = pd.DataFrame(df_cam.groupby(['Country_code', 'Campaign'])['Accepted
(%)'].mean()*100).reset_index(drop=False)

# rename the campaign variables so they're easier to interpret
df_cam['Campaign'] = df_cam['Campaign'].replace({'Camp1': '1',
                                                 'Camp2': '2',
                                                 'Camp3': '3',
                                                 'Camp4': '4',
                                                 'Camp5': '5',
                                                  'RecentCamp': 'Most recent'
                                                 })

# Average product sales

spending = pd.DataFrame(round(data[ts_col].mean(), 1), columns=['Average']).s
ort_values(by='Average').reset_index()



fig1 = px.bar(spending, x='Average', y='index', color = 'Average',

              title='Average $ spent on...',

               height = 500)

fig1.update(layout=dict(title=dict(x=0.5)))

# choropleth plot
fig2 = px.choropleth(df_cam, locationmode='ISO-3', color='Accepted (%)', face
t_col='Campaign', facet_col_wrap=2,
                     facet_row_spacing=0.05, facet_col_spacing=0.01, width=700
,
                     locations='Country_code', projection='natural earth', tit
le='Marketing Campaign Success Rate by Country'
                    )
fig2.show()

# Performing logistic regression p-values for campaign acceptance ~ country u
sing GLM
import statsmodels.formula.api as smf
import statsmodels as sm
```

```
from scipy import stats

data_cam_ww = data[['Country', 'Camp1', 'Camp2', 'Camp3', 'Camp4', 'Camp5', '
RecentCamp']]

stat_results = []

# GLM
for col in data_cam_ww.drop(columns='Country').columns:
    this_data = data_cam_ww[['Country', col]]

    formula = col+'~Country'

    # logistic regression (family=binomial)
    model = smf.glm(formula = formula, data=this_data, family=sm.genmod.famil
ies.Binomial())
    result = model.fit()

    chisq = result.pearson_chi2
    pval = stats.distributions.chi2.sf(chisq , 7)

    stat_results.append(pval)

    print(result.summary())

# Checking results
print("\nChisq p-values: ", stat_results)

# Merging original country codes into dataset
countries = data[['Country', 'Country_code']].drop_duplicates().reset_index(d
rop=True)
df_cam2 = df_cam.merge(countries, how='left', on='Country_code')
df_cam2.head()

# Graphing results
fig3 = px.bar(df_cam2, x='Country', y='Accepted (%)', facet_col='Campaign', c
olor = 'Campaign',
title='Regional Effects on Campaign Effectiveness')
fig3
```

Performed logistic regression for Campaign Accepted by Country, reporting Chisq p-value for overall model.

- Findings: The regional differences in marketing campaign success are statistically significant.

Dash (Data Product Prep)

```
chart1 = px.pie(cam_success, values='Percent', names='index',
                title='Success Rates of Marketing Campaigns',
```

```
                height = 500)
chart1.update(layout=dict(title=dict(x=0.5)))

chart2 = px.pie(data, values='Total_Campaign', names='Country',
               title='Overall Positive Campaign Response by Country',
                height = 500)
chart2.update(layout=dict(title=dict(x=0.5)))
```

Dash product

```
external_stylesheets = ['https://codepen.io/chriddyp/pen/bWLwgP.css']

app = dash.Dash(__name__, external_stylesheets=external_stylesheets)

fig1 = dcc.Graph(
        id='fig1',
        figure=fig1,
        className="four columns"
    )
chart1 = dcc.Graph(
        id='chart1',
        figure=chart1,
        className="four columns"
    )
chart2 = dcc.Graph(
        id='chart2',
        figure=chart2,
        className="four columns"
    )
fig3 = dcc.Graph(
        id='fig3',
        figure=fig3,
        className="twelve columns"
    )

table = dash_table.DataTable(
    id="table",
    columns=[{"name": i, "id": i} for i in data.columns],
    data=data.to_dict("records"),
    page_size=20,
    style_table={'height': '300px', 'overflowY': 'auto'},
    export_format='xlsx',
    export_headers='display',
    merge_duplicate_headers=True
)
# setup the header
header = html.H2(children="Marketing Campaign Results")

# setup to rows, graph 1-3 in the first row, and graph 4 in the second:
row1 = html.Div(children=[chart1, fig1])
```

```
row2 = html.Div(children=[chart2, fig3])

# setup & apply the layout
layout = html.Div(children=[header, row1, row2, table], style={"text-align":
"center"})
app.layout = layout

if __name__ == '__main__':
    app.run_server(debug=False)
```

Implementation Plan

Overview

The dashboard data product comes with a simple purpose: To drive efficiency and accountability by providing on-demand visibility and up-to-date information. A dashboard isn't merely a "report card" used by corporations to track KPIs, but a formidable tool that provides the scalability and illustrative power to tell a story of how a business is faring (Orlovskyi & Kopp, 2020). The dashboard is essentially the Swiss-army knife of business intelligence, providing organizations all the relevant tools for success where each option has its own function to be used alone or in combination with others.

Following the compilation of the source code, the completed Python file needs to be uploaded to the selected cloud service, in this case PythonAnywhere. PythonAnywhere, which is owned by Anaconda, is a web-based IDE and cloud environment that provides fully preconfigured Python environments, vastly reducing the number of libraries that need to be bash installed by the user in this new service (PythonAnywhere, 2022). The company offers several plans, including a free option that while offers less features, is a perfect fit to host this data product. After the file has been uploaded, the web hosted parameters need to be configured based off the PythonAnywhere guidelines. Once these parameters have been finalized, the data product can be tested for connectively and is automatically hosted for a period of 3 months.

Assumptions, Dependencies, Constraints

Assumptions:

- A cloud service platform which offers a free option will have restricted features.
    - This is always a possibility with any SaaS offering, but through proper vetting and prioritization of features, we can identify the appropriate service that fits the needs of the data product.
- The initial configuration of the cloud service platform could be complex.

Dependencies:

- Before the dashboard can be assembled, the supporting figures and charts need to be defined.
- Prior to the dashboard going live, it must be uploaded to the cloud service platform and pass a series of connectivity tests.

Constraints:

- As there is no approved budget for this project, free or open-source technologies must be utilized.
- The free PythonAnywhere account limits the user to hosting one [1] web application, with restricted bandwidth allowances and no Jupyter notebook support (PythonAnywhere, 2022).
- The dashboard can only be hosted for 30 days unless a paid account is used.

Operational Readiness

The initial plan was to host the data product on a Git-powered platform. However, due to complexity requirements in the initial configuration, this was scrapped, and a more linear

platform was selected. This deviation set the release timeframe back albeit by a small margin, but time lost is time lost. After the platform was identified, the data file would need to be converted to a usable format and tested to ensure that the integrity was not compromised. Platform simplicity, pricing structures, and code viability are the hallmarks of ensuring this products successful deployment.

Data Conversion

The original data product file was produced in a Jupyter Notebook format (.ipynb), but in order to be properly hosted and executed, it must be converted into a Python format (.py). This is accomplished by exporting a copy of the Jupyter file from the IDE into the Python format and reviewing for any potential translations issues that may have occurred (PythonAnywhere, 2022). Following the confirmation that the data file integrity has not been altered, a directory will need to be created to house the application file, in addition to the supporting raw data file. Once uploaded to its corresponding directory, the application file can be tested in a sandbox environment, allowing for traffic metrics and initialization procedures to be tested prior to moving into production.

Phased Rollout

Phase 1 – Account Creation

1. Create an account on the cloud service website
2. Select Web and choose the option to create a new web application.
3. Follow the prompts, selecting Flask as the framework and the version of Python used to develop the data product (Scola, 2022).

Phase 2 – Data Upload

1. Navigate to the file directory within the cloud service.

2. Create a new directory for the data product
3. Create a subdirectory for all supporting files (raw data, CSS, etc.)
4. Upload the files to the corresponding directories
    a. Prior to uploading the application file, ensure it is named “app.py”

Phase 3 – App Configuration

1. Enter the Web tab and navigate to the WGSI config file
2. Modify the line that reads *from flask_app import app as application* to *from app import app¶ application = app.server*.
3. Save the changes
4. Refresh the data product

Phase 4 – Confirm Functionality

1. Follow the URL link to the application page
2. Once the dashboard loads, return to the main screen and check the traffic count. If the traffic indicator changes, the dashboard has been successfully tested.

Phase 5 – Deploy to Users

1. Prior to distribution, host an online Q&A/training session for the users on the upcoming dashboard.
2. Once completed, using the recommended communication medium, distribute the URL to the identified stakeholders.
3. Monitor web traffic to confirm the dashboard is receiving hits.
4. Address all feedback from the users promptly, ensuring that the dashboard remains online and unimpeded.

Support

To best support the stakeholders, current, and prospective users of the dashboard product, an online training will need to commence prior to providing access. This will ensure that the user base has ample time to review the product, provide feedback, and if necessary, ask questions. As the dashboard itself is quite simplified, a training timeframe of 30 minutes would suffice but this block may be extended if needed. By hosting an online session, rather than an in-person presentation, the stability of the user's internet connection will indirectly test as they will be accessing the dashboard product throughout the training (Becerra et al., 2021). Following the training period, a general support email will be provided in the event stakeholders and users have additional questions, comments, or concerns.

Release Planning

Prior to launch, the dashboard product will be tested one final time to ensure nothing has changed to the cloud service or supporting framework. During the training session and following its conclusion, web traffic and product bandwidth will be monitored closely. If the data product begins to show signs of latency, the responsible team will begin analyzing the root of the issue. If a maximum user load is identified and cannot be circumvented by booting idle users, the hosting account will need to be upgraded to allocate additional resources (Carlshamre, 2002).

Application Functionality and Execution

The finalized dashboard product has been uploaded and may be accessed at http://brodney01.pythonanywhere.com/ , which is in a constant hosting state. The data product was confirmed to be able to handle simultaneous users from both desktop and mobile origins, in addition to generating a report of the final data set.

Performance Analysis

Module Test Cases

Test case name: TC01 – Data Load				
Priority: Critical				
Module: Data load				
Test objective: Load dataset of interest using pandas read_csv() function.				
Step	**Test Detail**	**Expected Results**	**Actual Results**	**Pass/Fail**
1	Start Windows 10 machine	File imports successfully, reading in 2556 entries in 28 columns.	As expected.	Pass
2	Open Visual Studio Code			
3	Import Pandas library as pd			
4	Execute *pd.read_csv()* function			
5	Confirm data loaded with *data.info()* function			

Test case name: TC02 – Pre-Train Null Check				
Priority: Medium				
Module: EDA/Preprocessing				
Test objective: Perform a null check to determine if any values contain null entries.				
Step	**Test Detail**	**Expected Results**	**Actual Results**	**Pass/Fail**
1	Start Windows 10 machine	The file will not contain any null values.	The *income* feature contained 30 null values.	Pass*
2	Open Visual Studio Code			
3	Execute *data.isna().sum()* function			

Test case name: TC03 – File Export				
Priority: High				
Module: Dashboard application (results export)				
Test objective: Export model dataset as XLSX using button functionality				
Step	**Test Detail**	**Expected Results**	**Actual Results**	**Pass/Fail**

1	Go to site https://brodney01.pythonanywhere.com/	File downloads successfully as XLSX. Contents are true to data table.	As expected.	Pass
2	Click Export			
3	Open File			

Test case name: TC04 – Dashboard Browser Compatibility				
Priority: Medium				
Module: Dashboard application (UI compatibility)				
Test objective: Perform a browser compatibility check on the dashboard application.				
Step	**Test Detail**	**Expected Results**	**Actual Results**	**Pass/Fail**
1	Open Chrome browser	The dashboard layout and functionality remain unchanged in different browser environments.	As expected.	Pass
2	Go to site https://brodney01.pythonanywhere.com/			
3	Confirm layout remains unchanged			
4	Confirm export button functions			
5	Open FireFox browser			
6	Repeat steps 2-4			
7	Open Edge browser			
8	Repeat steps 2-4			

Requirements Testing

Component: Code to compile in < 60 seconds		
Name of Developer: Brian Rodney		
Name of Reviewer: Brian Rodney		
Checklist		
Type	**Pass**	**Comments**
Performance	Passed	Code compiled in 7.8 seconds.

Component: Export results report from dashboard
Name of Developer: Brian Rodney
Name of Reviewer: Brian Rodney

Checklist		
Type	**Pass**	**Comments**
Performance	Passed	The file downloaded with no errors and contains the expected information.
Functionality	Passed	Export button from dashboard generates a file without error.

System Testing

Test #	Test Description	Expected Result	Actual Result
1	Dashboard stress testing	The dashboard website should be able to sustain 100+ concurrent user requests.	The average response time was 0.003 seconds. Based off this, the dashboard can handle 3000+ requests before slowdown.
2	Device compatibility	The dashboard should automatically resize for mobile and desktop devices.	The dashboard is optimized and resizes appropriately to the device accessing it.
3	Website speed	The dashboard should load in < 5 seconds	The dashboard site loads < 5 seconds. However, this is subjective to ISP speed.
4	Code compilation	Code should compile in < 60 seconds	The source code compiles in < 10 seconds.

Evaluation

Project Requirement Results

Requirement	Outcome	Justification
Build a basic customer profile from the data available.	Met	The customer profile was constructed using descriptive statistics. The resulting demographic profile is: - Born in 1969 - Opened their account in 2017 - Has an annual income of ~$52K - Has at least 1 child at home

Identify which marketing campaign is the most successful.	Met	Based on the results of descriptive statistics, the "RecentCamp" was the most successful marketing campaign with a ~25% response rate.
Determine if a marketing campaigns success is influenced by a geographic region.	Met	Using GLM Logistic Regression, calculating for the ChiSq P-Value, the regional differences in marketing campaigns are statistically significant.
Identify which products are performing at higher levels.	Met	Statistical analysis places meat products as the highest performing product line. Nearly $300 was spent on meat products (AmtMeatProducts) with alcohol products (AmtAlcoholPurchases) at ~$165.

Planned Improvements

In an effort to continually update the dashboard product, a handful of key features have been identified that are currently missing or scoped down in their respective function. One improvement to note will be the inclusion of a public facing data load option. This would allow for the stakeholders to load future marketing datasets and view the resulting EDA and analysis without the need for a new dashboard to be created. An additional function will be to further enhance the output report, including fields that were otherwise omitted in the current revision, such as cluster groupings and confidence rates.

Project Presentation

The project presentation may be found on YouTube at: https://youtu.be/UV6VJ31Yxbg

The project repository is hosted on Github at: https://github.com/brodney01/GCUCapstone

User Manual

Dash Data Product User Guide

Dashboard Training & Documentation December 2022

Software Version 1.0

Document Revision 12/21/2022

Prepared by Brian Rodney

First issuance, 2022.

Cloud Host:
PythonAnywhere

https://brodney01.pythonanywhere.com/

Preface

This document is a user guide for the Dash dashboard and may act as a reference guide for the different fields and functionality built into the dashboard. Dashboards are designed to provide the users with visualized results of projects, KPIs, financial goals, etc. The views are flexible and allow you, the user, the ability to export the dataset directly for further analysis.

TABLE OF CONTENTS

PREFACE **46**

GENERAL INFORMATION **48**

SYSTEM SUMMARY **48**

USING THE SYSTEM **49**

ACCESSING THE DASHBOARD **49**

NAVIGATING THE DASHBOARD **49**

EXPORTING DATA **49**

TROUBLESHOOTING **49**

FAQ (FREQUENTLY ASKED QUESTIONS) **50**

HELP AND CONTACT DETAILS **50**

GLOSSARY **50**

General Information

The dashboard comes with a simple purpose: To drive efficiency and accountability by providing on-demand visibility and up-to-date information. A dashboard isn't merely a "report card" used by corporations to track KPIs, but a formidable tool that provides the scalability and illustrative power to tell a story of how a business is faring. The dashboard is essentially the Swiss-army knife of business intelligence, providing organizations all the relevant tools for success where each option has its own function to be used alone or in combination with others.

System Summary

To best achieve the implementation of a dashboard, several factors need to be taken into consideration: price, timeline, and accessibility. After much consideration, it was determined that Dash would be the best choice for the dashboard technology. Dash is an open-source, user interface library for creating analytical web applications. As Dash products are executed in web browsers, they are by default cross-compatible with mobile devices, allowing users to access dashboards on the go. Further, Dash does not require extensive knowledge in Python or HTML to build a strong dashboard UI and instead uses several supporting component libraries to allow for a building block type assembly. Dash also provides support for other programming languages including R, Julia, and F#. The designed product hosts the results of historical and current marketing campaigns, providing the users insights on the success rates and countries of interest that participated in these campaigns.

Using the System

Accessing the Dashboard

To use the dashboard, you must first access the dashboard via a web browser. Open the web browser or mobile device of choice and enter https://brodney01.pythonanywhere.com/ into the search bar and hit Enter. This will take you directly to the dashboard.

Navigating the Dashboard

To navigate the dashboard, simply use the slider bar on the web browser to move the contents up or down.

Exporting Data

Exporting data is simple! Simple navigate to the 

button in the top right corner of the dashboard and click it. A copy of the finalized data will be downloaded automatically in an XLXS format.

Troubleshooting

- I can't access the dashboard. It doesn't load!
    - Please check your internet connection. If your internet connection is stable, double-check the URL and ensure you are entering it correctly.
- The XLSX downloads but cannot be found.
    - Confirm the default download path set by your browser. The default folder is typically "Downloads"
- The dashboard takes a long time to load or crashes.
    - This may be due to high traffic. Please wait a few minutes and try again.
        - If the issue persists, please contact the helpdesk (details in the help page)

FAQ (Frequently Asked Questions)

- Why can't I zoom in or out on the visualizations?
    - This is a known bug. We are working on repairing the functionality of the interactive graphics.
- Can I access this dashboard on my mobile device?
    - Yes! The dashboard is optimized for both desktop and mobile devices.
- Will the visualizations change over time, or will we expect to see the same data every time the dashboard loads?
    - We have ensured that the data is reproducible by enabling a function within the source code. You will always see the same results.
- Can I get a copy of the source code?
    - Please contact our helpdesk so we may determine if the request meets the use case requirements.

Help and Contact Details

If you have any questions or concerns, please contact us at helpdesk@brodney.gcu.edu. The support team is available 9am – 5pm EST, Mon-Fri.

Glossary

- ***Bug*** - A **software bug** is an error, flaw or fault in the design, development, or operation of computer **software** that causes it to produce an incorrect or unexpected result, or to behave in unintended ways.
- ***Dashboard*** - A **dashboard** is a way of displaying various types of visual data in one place.
- ***Export*** - To send (data) from one program to another.

- ***Source Code*** - **Source code** is a human-readable text written in a specific programming language. (Not to be confused with the 2011 Jake Gyllenhaal movie)
- ***Visualization*** - **Data visualization** is the representation of **data** through use of common graphics, such as charts, plots, infographics, and even animations.

System Administration Guide

Dash Data Product System Admin Guide

Dashboard Training & Documentation December 2022

Software Version 1.0

Document Revision 12/21/2022

Prepared by Brian Rodney

First issuance, 2022.

Cloud Host:
PythonAnywhere

https://brodney01.pythonanywhere.com/

Preface

This document is a system admin guide for the Dash dashboard and may act as a reference guide for the different fields and functionality built into the dashboard. Dashboards are designed to provide the users with visualized results of projects, KPIs, financial goals, etc. The views are static and allow the system admin to adjust the visualizations.

TABLE OF CONTENTS

SYSTEM ADMINISTRATION GUIDE 52

PREFACE 54

SYSTEM OVERVIEW 55

SYSTEM CONFIGURATION 55

SYSTEM MAINTENANCE 55

SECURITY RELATED PROCESSES 56

System Overview

Dash is an open-source, user interface library for creating analytical web applications. As Dash products are executed in web browsers, they are by default cross-compatible with mobile devices, allowing users to access dashboards on the go. Dash does not require extensive knowledge in Python or HTML to build a strong dashboard UI and instead uses several supporting component libraries to allow for a building block type assembly. Dash also provides support for other programming languages including R, Julia, and F#. The designed product hosts the results of historical and current marketing campaigns, providing the users insights on the success rates and countries of interest that participated in these campaigns. The backend code is hosted by PythonAnywhere and can be downloaded by a system administrator with the proper credentials.

System Configuration

System Maintenance

The hosting platform, PythonAnywhere, maintains their servers automatically. This reduces the need for the system administrator to perform any patching or security rework. However, due to budget constraints this platform has only allotted a host time of three (3) months, after which the

dashboard will be pulled down. The system administrator should investigate other avenues to host the dashboard product or request a budget that covers a paid subscription to the current hosting service.

Security Related Processes

While the dashboard data is not expected to contain any PII, it does track the effectiveness of marketing campaigns for the customer base and therefore should be treated as confidential. The dashboard is hosted on PythonAnywhere, which is powered by AWS, so the hosting service s responsible for maintaining the overall security of the files. The user account utilized to host the dashboard has been set up with an alphanumeric password, which aligns with the current complexity requirements by NIST. The web application has been configured to only run-on HTTPS, which alleviates encryption concerns. While access to the dashboard product UI is open to all users that have the direct link, access to the file repository is strictly limited to only the system administrator and the creator.

Appendix A – Figures

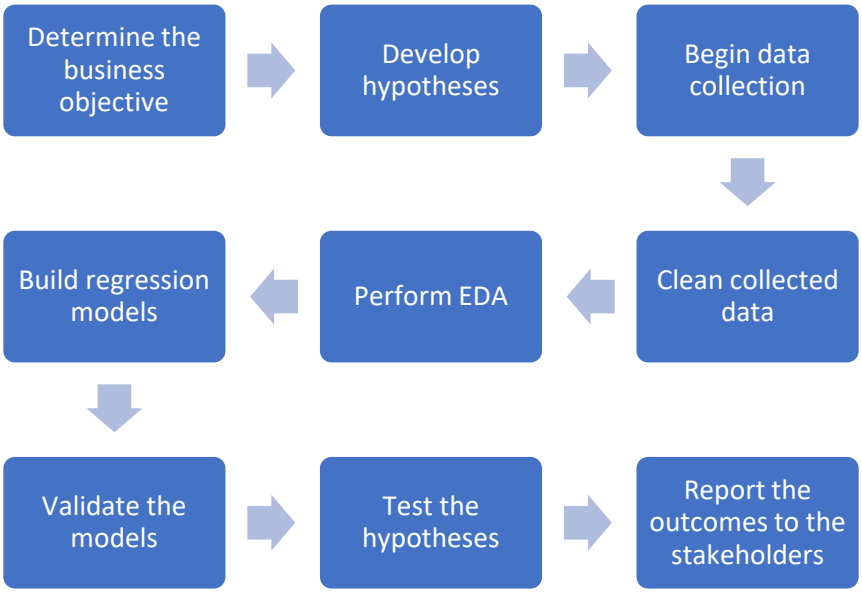


Figure A1. Process flow chart

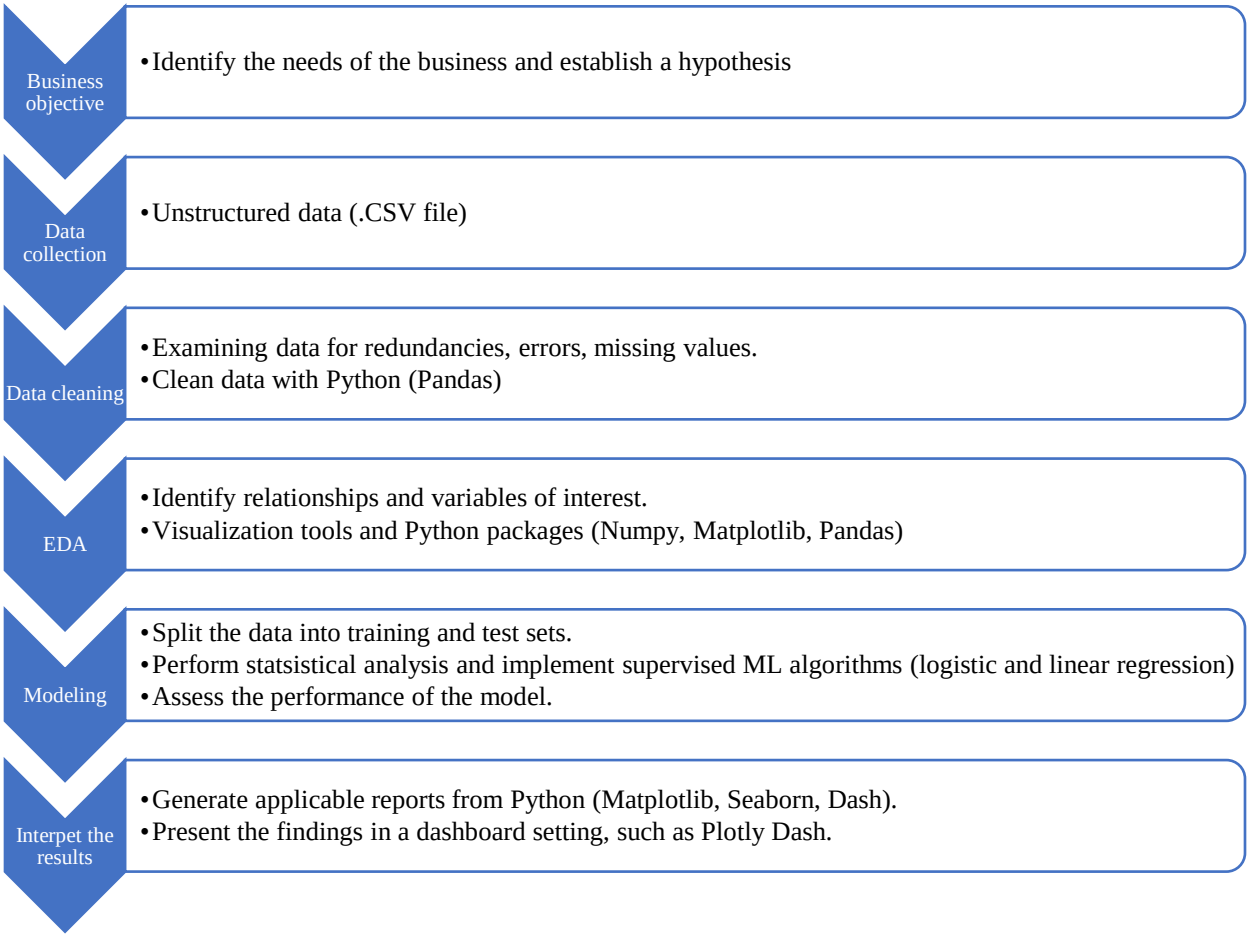


Figure A2. Data model pipeline

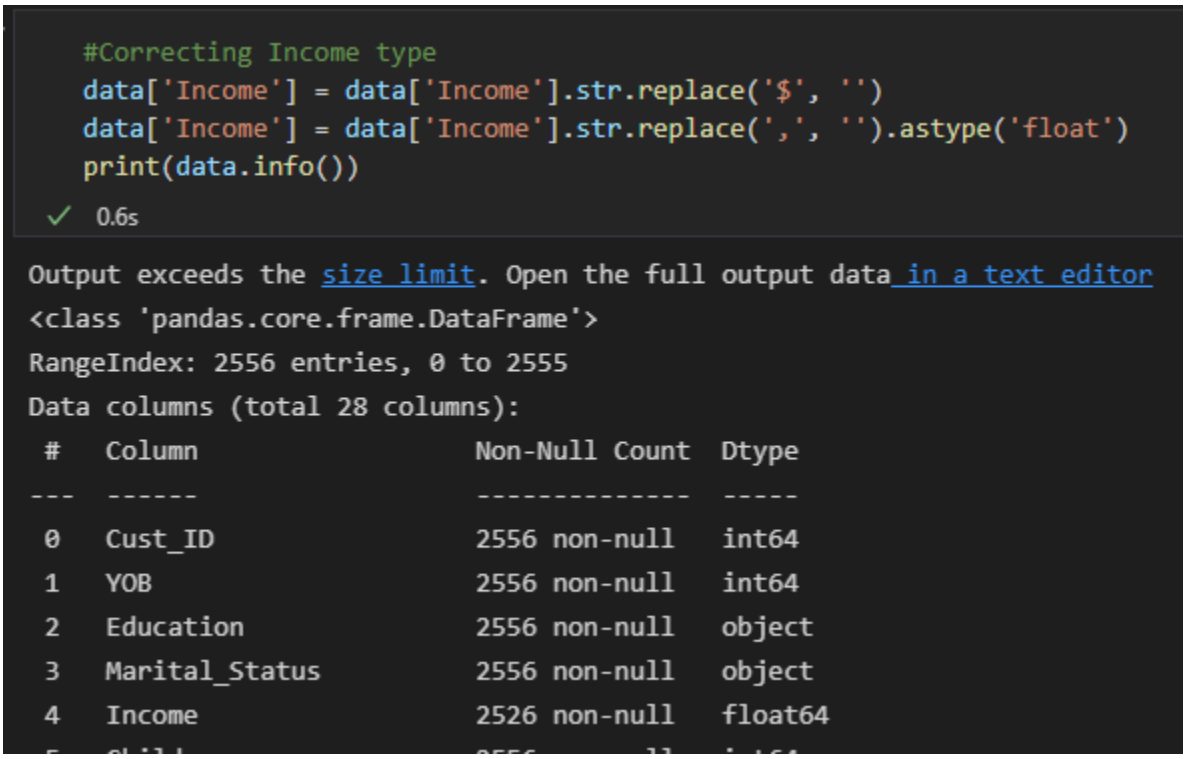
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2556 entries, 0 to 2555
Data columns (total 28 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Cust_ID                2556 non-null   int64
 1   YOB                    2556 non-null   int64
 2   Education              2556 non-null   object
 3   Marital_Status         2556 non-null   object
 4   Income                 2526 non-null   object
 5   Children               2556 non-null   int64
 6   18Plus_Dependents      2556 non-null   int64
 7   Country                2556 non-null   object
 8   Orig_Customer          2556 non-null   object
 9   Last Activity          2556 non-null   int64
 10  AmtMeatProducts        2556 non-null   int64
 11  AmtVegetables          2556 non-null   int64
 12  AmtAlcoholPurchases    2556 non-null   int64
 13  AmtClothingPurchases   2556 non-null   int64
 14  AmtCosmeticPurchases   2556 non-null   int64
 15  AmtElectronicPurchases 2556 non-null   int64
 16  CouponPurchase         2556 non-null   int64
 17  OnlinePurchase         2556 non-null   int64
 18  PhonePurchases         2556 non-null   int64
 19  BMPurchase             2556 non-null   int64
 20  OnlineActivity         2556 non-null   int64
 21  Camp3                  2556 non-null   int64
 22  Camp4                  2556 non-null   int64
 23  Camp5                  2556 non-null   int64
 24  Camp1                  2556 non-null   int64
 25  Camp2                  2556 non-null   int64
 26  RecentCamp             2556 non-null   int64
 27  Complaints             2556 non-null   int64
dtypes: int64(23), object(5)
memory usage: 559.2+ KB
None

```

Figure A3. Variable list with Dtype

```
    # Clean any whitespace from column names
    data.columns = data.columns.str.replace(' ', '')
```

Figure A4. Whitespace removal code



```
#Correcting Income type
data['Income'] = data['Income'].str.replace('$', '')
data['Income'] = data['Income'].str.replace(',', '').astype('float')
print(data.info())
```

✓ 0.6s

```
Output exceeds the size limit. Open the full output data in a text editor
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2556 entries, 0 to 2555
Data columns (total 28 columns):
 #   Column                 Non-Null Count  Dtype
---  ------                 --------------  -----
 0   Cust_ID                2556 non-null   int64
 1   YOB                    2556 non-null   int64
 2   Education              2556 non-null   object
 3   Marital_Status         2556 non-null   object
 4   Income                 2526 non-null   float64
```

Figure A5. Correction of *Income* Dtype

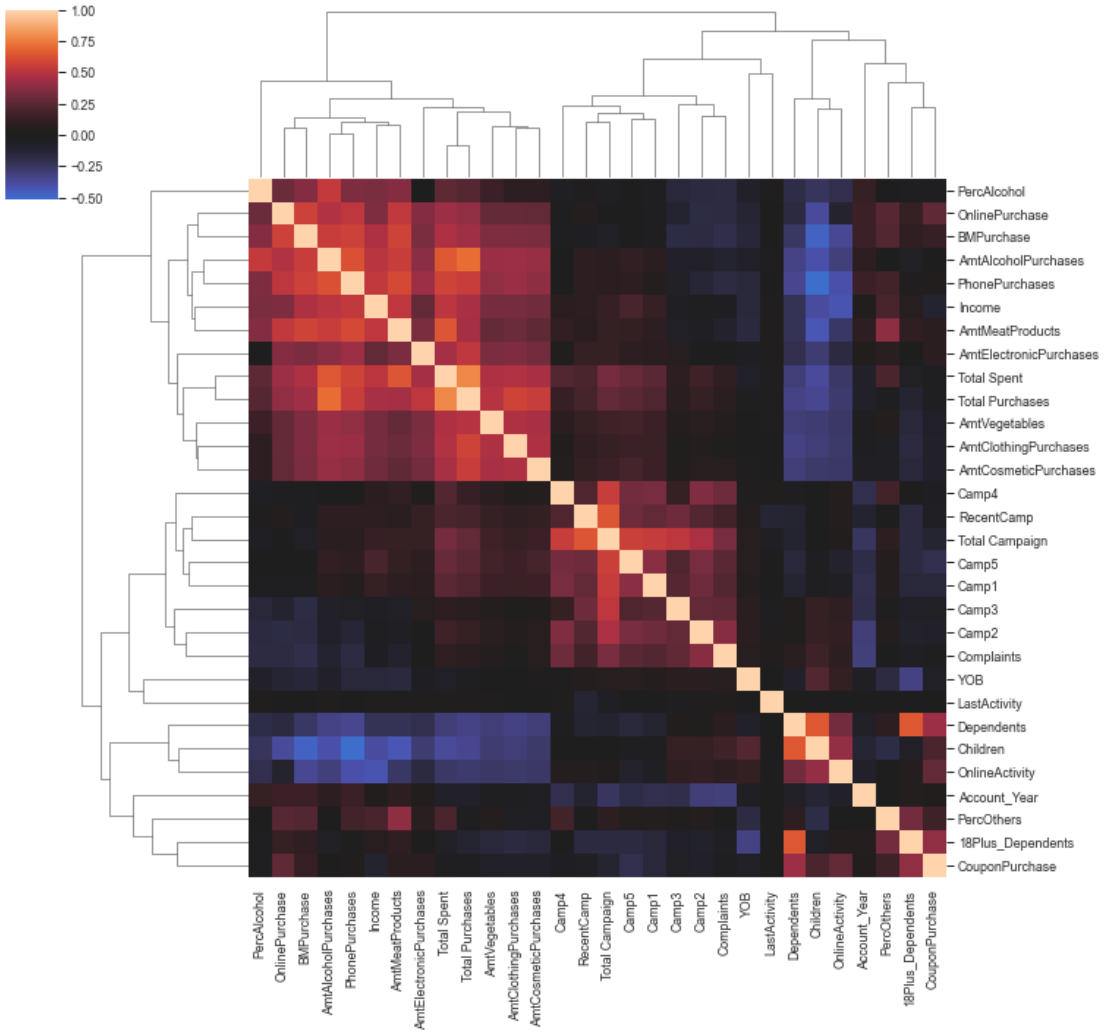


Figure A6. Correlation heatmap

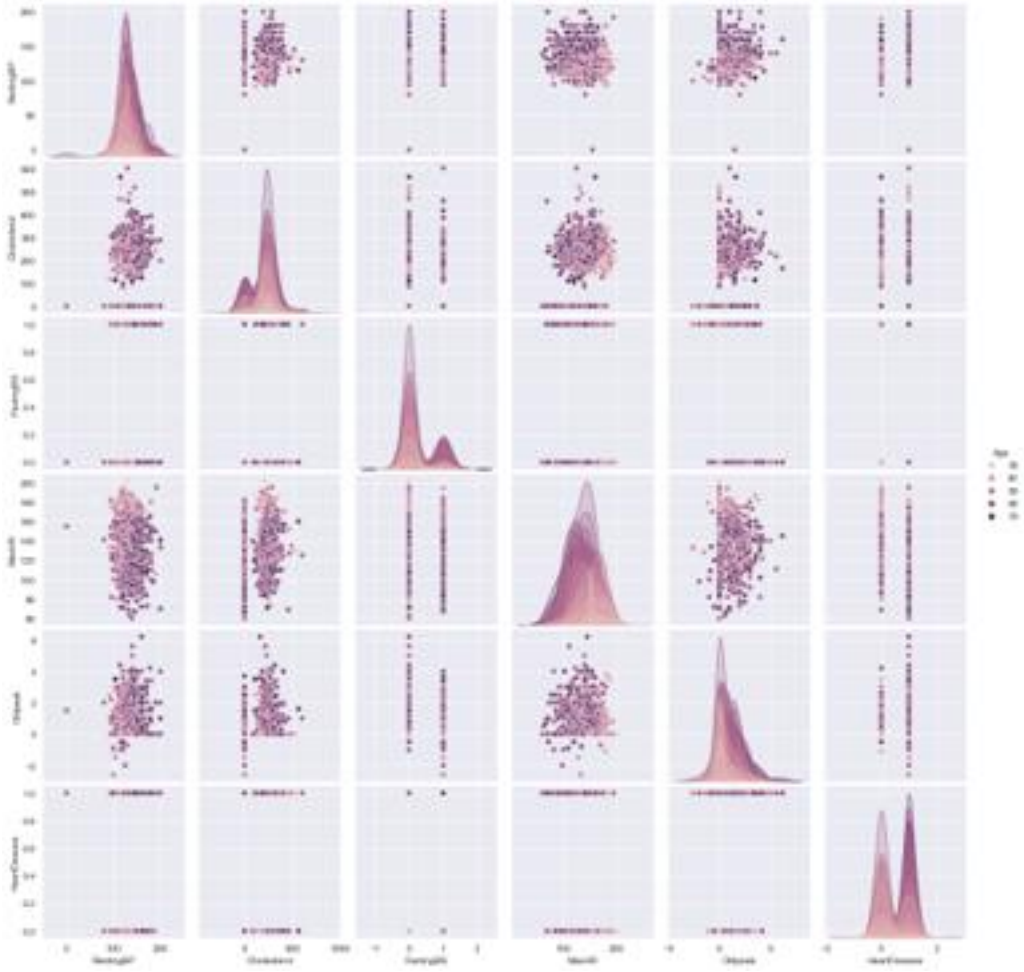


Figure A7. Seaborn scatterplot matrix

Figure A8. Plotly choropleth global heatmap

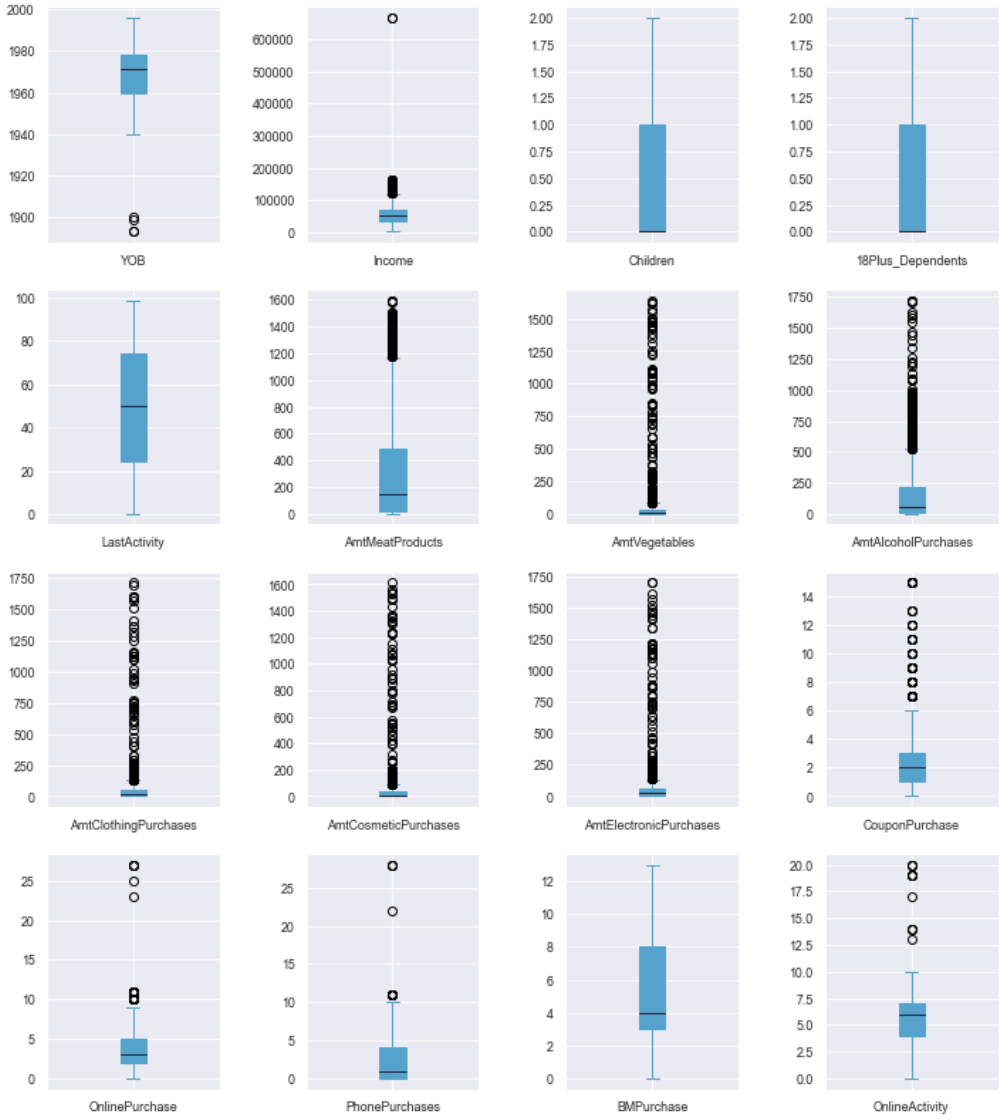


Figure A9. Boxplots for outlier check

Appendix B – References

Becerra, Z. M., Fereydooni, N., Kun, A. L., McKerral, A., Riener, A., Schartmüller, C., ... & Wintersberger, P. (2021). Interactive workshops in a pandemic: the real benefits of virtual spaces. *IEEE Pervasive Computing*, *20*(1), 35-39.

Buitinck, L., Louppe, G., Blondel, M., Pedregosa, F., Mueller, A., Grisel, O., ... & Varoquaux, G. (2013). API design for machine learning software: experiences from the scikit-learn project. *arXiv preprint arXiv:1309.0238*.

Carlshamre, P. (2002). Release planning in market-driven software product development: Provoking an understanding. *Requirements engineering*, *7*(3), 139-151.

Hair Jr, J. F., & Sarstedt, M. (2021). Data, measurement, and causal inferences in machine learning: opportunities and challenges for marketing. *Journal of Marketing Theory and Practice*, *29(*1), 65-77.

Korstanje, J. (2022, January 6). *Assumptions of Linear Regression | Towards Data Science*. Medium. Retrieved from https://towardsdatascience.com/assumptions-of-linear-regression-fdb71ebeaa8b

Munappy, A. R., Bosch, J., & Olsson, H. H. (2020, November). Data pipeline management in practice: Challenges and opportunities. In *International Conference on Product-Focused Software Process Improvement* (pp. 168-184). Springer, Cham.

Orlovskyi, D. L., & Kopp, A. M. (2020). *A business intelligence dashboard design approach to improve data analytics and decision making* (Doctoral dissertation, Stylos).

Pepall, L., & Richards, D. (2021). Targeted Value-Enhancing Advertising and Price Competition. *Review of Industrial Organization, 59*(3), 443-459.

Plotly. (2022). *Plotly Open Source Graphing Libraries*. https://plotly.com/graphing-libraries/

plotly. (2017, June 21). *Introducing Dash - Plotly*. Medium. https://medium.com/plotly/introducing-dash-5ecf7191b503

PythonAnywhere (2022). *Host, run, and code Python in the cloud: PythonAnywhere*. Retrieved from https://www.pythonanywhere.com/

Scola, E. (2022, March 30). *The Easiest Way to Deploy Your Dash App for Free - Towards Data Science*. Medium. Retrieved from https://towardsdatascience.com/the-easiest-way-to-deploy-your-dash-app-for-free-f92c575bb69e

Thieme, C. (2022, March 30). *Identifying Outliers in Linear Regression — Cook's Distance*. Medium. Retrieved from https://towardsdatascience.com/identifying-outliers-in-linear-regression-cooks-distance-9e212e9136a

Waskom, M. (2012). *seaborn: statistical data visualization — seaborn 0.11.2 documentation*. Seaborn. https://seaborn.pydata.org/index.html