**HOW DO PREDICTORS FACTOR INTO CLUSTER-BASED FORECASTING?**

**OFM3 — OFM3 TASK 1: CLUSTERING TECHNIQUES**

**DATA MINING II — D212**

**PRFA — OFM**

**STUDENT NAME: IBRAHIM SULEIMAN**

**DATA ANALYTICS**

**STUDENT NUMBER: 001429984**

**COLLEGE OF INFORMATION TECHNOLOGY, WESTERN GOVERNORS UNIVERSITY**

**NAME OF INSTRUCTOR: DANIEL SMITH**

**OCTOBER 26th, 2022**

**Part 1: Research Question**

Section A: Description of the Report

Section A1: How do predictors factor into cluster-based forecasting?

This is relevant to a real-world organizational situation and would be answer using **K-mean** clustering technique.

k-means- In an unlabeled multidimensional dataset, the K-means algorithm searches for a predetermined number of clusters; it comes to this conclusion by providing a clear explanation of how an optimum cluster might be expressed. Clustering results in new knowledge being discovered. Unsupervised learning includes a branch called clustering (Neelam Tyagi, 2020).

Section A2: Goal of the data analysis

The objective of the data analysis is to utilize the algorithm to locate relationships among data, with the number of groups indicated by the variable K, and to categorize data points into clusters even when there is limited information available about the data with the quest in knowing how predictors factor into cluster-based forecasting using the churn dataset ([Andrea Trevino](https://blogs.oracle.com/authors/andrea-trevino), 2016).

Part II: Technique Justification

Section B1: How K-Means Clustering Technique Analyzes Churn Dataset

* In terms of interpretation and conclusion, it appears to be extremely easy in using k-mean to analyze the churn dataset focusing on the predicators.
* K-means performed more quickly than hierarchical clustering for a high number of variables in the churn dataset.
* It appears that an instance could change the cluster while recalculating the cluster center.
* Compact clusters are reformed by K-means.
* Numerical data that is not labeled was analyzed
* Additionally, it produces the greatest results when the datasets are well different from one another and is was quick, reliable, and simple to comprehend.

Section B2: Assumption

Two presumptions about the clusters are taken into account by the K-Means clustering method: first, that the clusters are spherical, and second, that the clusters are of equal size. When the program processes the data and creates clusters, the spherical assumption aids in separating the clusters.

Section B3: Justification of Chosen packages or Libraries

* The necessary programs and libraries for the k-means clustering are listed below, along with information on how they will help with the analysis:
* Pandas are a common import for projects in machine learning. It offers tools for parsing and scoring data, in addition to ways of accessing and displaying data.

- Numpy is a popular import for applications in machine learning. An approach can improve reading and visualizing data, in addition to statistical tools for information processing as well as evaluation.

* The standard visualization import is Matplotlib. The capabilities for visualizing reports and data points in this package are more powerful.
* Graphs, charts, and matrices from Seaborn are illustrative and intuitive to the eye.

- Scikit-learn offers strategies and justifications for dividing, training, testing, and fitting data. Additionally, this package includes justifications for categorizing and forecasting data as well as applying metrics to models (Michael Galarnyk,2018).

Part III: Data Preparation

Section C1: Data Preprocessing goal

For the categorization to function properly, the variables that are present (Yes or No) would be transformed to (0/1). Furthermore, Data preprocessing, also known as data preparation, is a data mining approach that turns unstructured data into a form that machine learning algorithms can use. Real-world data is frequently inaccurate (contains outliers, duplicates, and errors), incomplete (some values are missed), and it may be kept in many locations and formats. Handling these problems is the responsibility of data preprocessing. ([Evgeniy Ryzhkov](https://medium.com/@evgen.ryzhkov?source=post_page-----b755426f9932--------------------------------), 2020).

Section C2: Initial Dataset Variables

Continuous Variables

|  |
| --- |
| 'CaseOrder', 'Zip', 'Lat', 'Lng', 'Population', 'Children', 'Age', 'Income''Outage\_sec\_perweek', 'Email', 'Contacts', 'Yearly\_equip\_failure', , 'Tenure', 'MonthlyCharge', 'Bandwidth\_GB\_Year', 'Item1', 'Item2', 'Item3', 'Item4', 'Item5', 'Item6', 'Item7', 'Item8'] |

Categorical Variables

|  |
| --- |
| 'Customer\_id', 'Interaction', 'UID', 'City', 'State', 'County', 'Area', 'TimeZone', 'Job', , 'Marital', 'Gender', 'Churn', 'Techie', 'Contract', 'Port\_modem', 'Tablet', 'InternetService', 'Phone', 'Multiple', 'OnlineSecurity', 'OnlineBackup', 'DeviceProtection', 'TechSupport', 'StreamingTV', 'StreamingMovies', 'PaperlessBilling', 'PaymentMethod' |

|  |
| --- |
| <class 'pandas.core.frame.DataFrame'>  RangeIndex: 10000 entries, 0 to 9999  Data columns (total 50 columns):  # Column Non-Null Count Dtype  --- ------ -------------- -----  0 CaseOrder 10000 non-null int64  1 Customer\_id 10000 non-null object  2 Interaction 10000 non-null object  3 UID 10000 non-null object  4 City 10000 non-null object  5 State 10000 non-null object  6 County 10000 non-null object  7 Zip 10000 non-null int64  8 Lat 10000 non-null float64  9 Lng 10000 non-null float64  10 Population 10000 non-null int64  11 Area 10000 non-null object  12 TimeZone 10000 non-null object  13 Job 10000 non-null object  14 Children 10000 non-null int64  15 Age 10000 non-null int64  16 Income 10000 non-null float64  17 Marital 10000 non-null object  18 Gender 10000 non-null object  19 Churn 10000 non-null object  20 Outage\_sec\_perweek 10000 non-null float64  21 Email 10000 non-null int64  22 Contacts 10000 non-null int64  23 Yearly\_equip\_failure 10000 non-null int64  24 Techie 10000 non-null object  25 Contract 10000 non-null object  26 Port\_modem 10000 non-null object  27 Tablet 10000 non-null object  28 InternetService 10000 non-null object  29 Phone 10000 non-null object  30 Multiple 10000 non-null object  31 OnlineSecurity 10000 non-null object  32 OnlineBackup 10000 non-null object  33 DeviceProtection 10000 non-null object  34 TechSupport 10000 non-null object  35 StreamingTV 10000 non-null object  36 StreamingMovies 10000 non-null object  37 PaperlessBilling 10000 non-null object  38 PaymentMethod 10000 non-null object  39 Tenure 10000 non-null float64  40 MonthlyCharge 10000 non-null float64  41 Bandwidth\_GB\_Year 10000 non-null float64  42 Item1 10000 non-null int64  43 Item2 10000 non-null int64  44 Item3 10000 non-null int64  45 Item4 10000 non-null int64  46 Item5 10000 non-null int64  47 Item6 10000 non-null int64  48 Item7 10000 non-null int64  49 Item8 10000 non-null int64  dtypes: float64(7), int64(16), object(27)  memory usage: 3.8+ MB |

Section C3: Data Preparation Steps and Codes Used

Prior to conducting the analysis, the data must be available. The first step is to ensure that there are no blank columns in any of the columns. Making sure there are no duplicates of any of the data in the columns should come next. In addition, we want to ensure that there are no duplicate columns or rows, so we'll verify that as well (False).

The dataset includes certain variables that were found to be worthless for the logistic analysis, such as customer demographics that cannot be changed and are connected to the interaction and location of the consumer, thus those columns should be deleted.

As a result, working with the data is now simpler. The categorical variables must be translated to numerical values before any (yes or no) or other category alternatives may be used. In order to give a clearer understanding and determination of applicable factors, the survey columns also need to be renamed (Peter Grant, 2019).

Codes

|  |
| --- |
| import numpy as np  import pandas as pd  from sklearn import linear\_model  import matplotlib.pyplot as plt  import seaborn as sns  %matplotlib inline  pd.set\_option('display.max\_columns', None)  import pylab  import sklearn.cluster as cluster  import sklearn.metrics as metrics  from pylab import rcParams  import statsmodels.api as sm  import statistics  from scipy import stats  import sklearn  from sklearn import preprocessing  import numpy as np # linear algebra  import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)  import matplotlib.pyplot as plt # for data visualization  import seaborn as sns # for statistical data visualization  %matplotlib inline  import warnings  warnings.filterwarnings('ignore')  df = pd.read\_csv("churn\_clean.csv")  df.dropna()  print(df.shape)  print(list(df.columns))  df.head()  df.info()  #check for missing data  df.isna().any()  df.fillna(df.mean(), inplace=True)  df.isna()  df.nunique()  # dictating outliers  boxplot=sns.boxplot(x='Income',data=df)  # Dropping outliers systematically  outlierFilter=df['Income'] < 65000  df = df[outlierFilter]  boxplot=sns.boxplot(x='Income',data=df)  boxplot=sns.boxplot(x='MonthlyCharge',data=df)  boxplot=sns.boxplot(x='Age',data=df)  #create scatterplots to look for correlations  #create scatterplots to look for correlations  sns.scatterplot(x=df['MonthlyCharge'],y=df['Churn'],color='blue')  plt.show();  sns.scatterplot(x=df['Income'],y=df['Churn'],color='blue')  plt.show();  sns.scatterplot(x=df['Income'],y=df['MonthlyCharge'],color='blue')  plt.show();  sns.scatterplot(x=df['Children'],y=df['Age'],color='blue')  plt.show();  sns.scatterplot(x=df['TechSupport'],y=df['Age'],color='blue')  plt.show();  sns.scatterplot(x=df['TechSupport'],y=df['Churn'],color='blue')  plt.show();  sns.scatterplot(x=df['Children'],y=df['Churn'],color='blue')  plt.show();  #check for duplicate data in columns  df[df.duplicated()]  # check if any cols are duplicated - Looking for False  df.columns.duplicated().any()  # check if any rows are duplicated - looking for False  df.duplicated().any()  # The dropping of demographic data  df = df.drop(['CaseOrder','Customer\_id','Marital','Gender','Contract','InternetService','PaymentMethod','Email','Techie','Port\_modem','Phone','OnlineBackup','PaperlessBilling','Tenure','Bandwidth\_GB\_Year','Job','Interaction','UID','City','State','County','Zip','Lat','Lng','Population','Area','TimeZone','Job','Outage\_sec\_perweek','Email','Contacts','Yearly\_equip\_failure','Techie','Contract','Port\_modem','Tablet'], axis=1)  # Lets verify columns were dropped  df.head()  #The overview of descriptive statistics  df.describe()  #rename survey columns for easier identification  df.rename(columns={'Item1':'Timely response','Item2':'Timely fixes','Item3':'Timely replacements','Item4':'Reliability','Item5':'Options','Item6':'Respectful response','Item7':'Courteous exchange','Item8':'Evidence of active listening'},inplace=True)  #verify columns were renamed correctly  df.head()  #change yes/no to 1/0  df = df.replace(to\_replace = ['Yes','No'],value = [1,0])  #Lets ensure values were changed  df.head()  print(list(df.columns))  #create histograms of both categorical and continuous variables  df[["Children", "Age", "Income", "Churn", "Multiple", "OnlineSecurity", "DeviceProtection", "TechSupport", "StreamingTV", "StreamingMovies", "MonthlyCharge", "Timely response", "Timely fixes", "Timely replacements", "Reliability", "Options", "Respectful response", "Courteous exchange", "Evidence of active listening"]].hist()  plt.savefig('Churn\_plot,jpg')  plt.show()  #lets create scatterplots for numeric variables to view distributions and look for relationships  Churn\_numeric=df[['Children','Age','Income','MonthlyCharge','Respectful response','Courteous exchange','Evidence of active listening']]  pd.plotting.scatter\_matrix(Churn\_numeric,figsize=[15,15]);  df  #export prepared dataset  df.to\_csv('prepared\_d212task1.csv', index = False) |

Section C4: Copy of the Cleaned Dataset

The cleaned dataset was provided as csv file in the submission named;

We exported our prepared dataset as

df.to\_csv('prepared\_d212task1.csv', index = False)

Part IV: Analysis

Section D1: Description and Screenshots of the Analysis Technique

Table

Description automatically generated

Table

Description automatically generated

Table

Description automatically generated

The below gives us an illustration of how we were able to declare the target variable and the feature vector. Also we checked to see if category variables were transformed to integers.

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, table

Description automatically generated with medium confidence

Graphical user interface

Description automatically generated with medium confidence

Table

Description automatically generated

Table

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Table

Description automatically generated

Table

Description automatically generated

Feature scaling in machine learning is one of the most important phases in the preprocessing of data before building a machine learning model. A machine learning model's strength can be changed through scaling, from poor to better.

Normalization and standardization are the two methods of feature scaling that are most frequently used. When we want to confine our data to a range between two numbers, usually between [0,1] and [-1,1], we employ normalization. Standardization renders our data unitless by transforming it to have a mean of 0 and a variation of 1.

Graphical user interface, text, application

Description automatically generated

Text

Description automatically generated

By attempting to divide samples into n groups with equal variances and minimizing an indicator referred to as inertia, or within-cluster sum-of-squares, the K-Means algorithm clusters data. A measure of how internally cohesive clusters are can be found in inertia, also known as the within-cluster sum of squares criterion.

A group of N samples is divided into K disjoint clusters via the k-means algorithm, and each cluster's mean j is used to describe its samples. The cluster centroids are another name for the meanings.

The within-cluster sum of squared criterion, often known as inertia, is the goal of the K-means algorithm, which selects centroids.

Graphical user interface, text, application, email

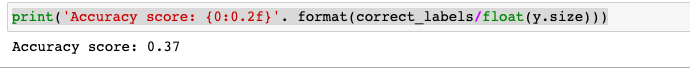
Description automatically generated

The better the model fits, the less inertia there is in the model. The model has extremely high inertia, which is above average, as can be shown. Therefore, it doesn't seem like this model fits 100% the data the best.

Graphical user interface, text

Description automatically generated

We also analyze the model's weak classification quality.



Our unsupervised model had a classification accuracy of 37% below average.

Graphical user interface

Description automatically generated with medium confidence

To determine the ideal number of clusters, we used the elbow method.

We can observe from the plot above that there is a bend at k=2. Therefore, k=2 can be thought of as an appropriate cluster size for grouping these data. But as we have shown, with k=2, we were able to attain a classification accuracy of 37%. For ease, we would rewrite the necessary code using k=2.

Graphical user interface, text, application, email

Description automatically generated

As a result, the classification accuracy of our unsupervised model was just 37%. We would use various cluster counts to test the model's accuracy.

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

Graphical user interface, text, application, email

Description automatically generated

With K =3, K =4, and K =5, we were able to get a substantially lower accuracy of 0.13%, 0.13%, and 0.15%, respectively.

Section D2: The Codes Used To Perform The Clustering Technique

|  |
| --- |
| import numpy as np  import pandas as pd  from sklearn import linear\_model  import matplotlib.pyplot as plt  import seaborn as sns  %matplotlib inline  pd.set\_option('display.max\_columns', None)  import pylab  import sklearn.cluster as cluster  import sklearn.metrics as metrics  from pylab import rcParams  import statsmodels.api as sm  import statistics  from scipy import stats  import sklearn  from sklearn import preprocessing  import numpy as np # linear algebra  import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)  import matplotlib.pyplot as plt # for data visualization  import seaborn as sns # for statistical data visualization  %matplotlib inline  import warnings  warnings.filterwarnings('ignore')  df = pd.read\_csv("churn\_clean.csv")  df.dropna()  print(df.shape)  print(list(df.columns))  df.head()  df.info()  #check for missing data  df.isna().any()  df.fillna(df.mean(), inplace=True)  df.isna()  df.nunique()  # dictating outliers  boxplot=sns.boxplot(x='Income',data=df)  # Dropping outliers systematically  outlierFilter=df['Income'] < 65000  df = df[outlierFilter]  boxplot=sns.boxplot(x='Income',data=df)  boxplot=sns.boxplot(x='MonthlyCharge',data=df)  boxplot=sns.boxplot(x='Age',data=df)  #create scatterplots to look for correlations  sns.scatterplot(x=df['MonthlyCharge'],y=df['Churn'],color='blue')  plt.show();  sns.scatterplot(x=df['Income'],y=df['Churn'],color='blue')  plt.show();  sns.scatterplot(x=df['Income'],y=df['MonthlyCharge'],color='blue')  plt.show();  sns.scatterplot(x=df['Children'],y=df['Age'],color='blue')  plt.show();  sns.scatterplot(x=df['TechSupport'],y=df['Age'],color='blue')  plt.show();  sns.scatterplot(x=df['TechSupport'],y=df['Churn'],color='blue')  plt.show();  sns.scatterplot(x=df['Children'],y=df['Churn'],color='blue')  plt.show();  #check for duplicate data in columns  df[df.duplicated()]  # check if any cols are duplicated - Looking for False  df.columns.duplicated().any()  # check if any rows are duplicated - looking for False  df.duplicated().any()  # The dropping of demographic data  df = df.drop(['CaseOrder','Customer\_id','Marital','Gender','Contract','InternetService','PaymentMethod','Email','Techie','Port\_modem','Phone','OnlineBackup','PaperlessBilling','Tenure','Bandwidth\_GB\_Year','Job','Interaction','UID','City','State','County','Zip','Lat','Lng','Population','Area','TimeZone','Job','Outage\_sec\_perweek','Email','Contacts','Yearly\_equip\_failure','Techie','Contract','Port\_modem','Tablet'], axis=1)  # Lets verify columns were dropped  df.head()  #The overview of descriptive statistics  df.describe()  #rename survey columns for easier identification  df.rename(columns={'Item1':'Timely response','Item2':'Timely fixes','Item3':'Timely replacements','Item4':'Reliability','Item5':'Options','Item6':'Respectful response','Item7':'Courteous exchange','Item8':'Evidence of active listening'},inplace=True)  #verify columns were renamed correctly  df.head()  #change yes/no to 1/0  df = df.replace(to\_replace = ['Yes','No'],value = [1,0])  #Lets ensure values were changed  df.head()  print(list(df.columns))  #create histograms of both categorical and continuous variables  df[["Children", "Age", "Income", "Churn", "Multiple", "OnlineSecurity", "DeviceProtection", "TechSupport", "StreamingTV", "StreamingMovies", "MonthlyCharge", "Timely response", "Timely fixes", "Timely replacements", "Reliability", "Options", "Respectful response", "Courteous exchange", "Evidence of active listening"]].hist()  plt.savefig('Churn\_plot,jpg')  plt.show()  #lets create scatterplots for numeric variables to view distributions and look for relationships  Churn\_numeric=df[['Children','Age','Income','MonthlyCharge','Respectful response','Courteous exchange','Evidence of active listening']]  pd.plotting.scatter\_matrix(Churn\_numeric,figsize=[15,15]);  df  #export prepared dataset  df.to\_csv('prepared\_d212task1.csv', index = False)  PERFORMING THE CLUSTERING TECHNIQUE:  import numpy as np  import pandas as pd  from sklearn import linear\_model  import matplotlib.pyplot as plt  import seaborn as sns  %matplotlib inline  pd.set\_option('display.max\_columns', None)  import pylab  from pylab import rcParams  import statsmodels.api as sm  import statistics  from scipy import stats  import sklearn  from sklearn import preprocessing  import numpy as np # linear algebra  import pandas as pd # data processing, CSV file I/O (e.g. pd.read\_csv)  import matplotlib.pyplot as plt # for data visualization  import seaborn as sns # for statistical data visualization  %matplotlib inline  import warnings  warnings.filterwarnings('ignore')  df = pd.read\_csv("prepared\_d212task1.csv")  df.dropna()  print(df.shape)  print(list(df.columns))  df.head()  df.info()  df.head()  #Declare feature vector and target variable  X = df  y = df['Churn']  # Convert categorical variable into integers verified  from sklearn.preprocessing import LabelEncoder  le = LabelEncoder()  X['Churn'] = le.fit\_transform(X['Churn'])  y = le.transform(y)  # View the summary of X  X.info()  X.head()  # Feature Scaling  cols = X.columns  from sklearn.preprocessing import MinMaxScaler  ms = MinMaxScaler()  X = ms.fit\_transform(X)  X = pd.DataFrame(X, columns=[cols])  X.head()  # K-Means model with two clusters  from sklearn.cluster import KMeans  kmeans = KMeans(n\_clusters=2, random\_state=0)  kmeans.fit(X)  # K-Means model parameters study  kmeans.cluster\_centers\_  kmeans.labels\_  kmeans.inertia\_  # Check quality of weak classification by the model  labels = kmeans.labels\_  # check how many of the samples were correctly labeled  correct\_labels = sum(y == labels)  print("Result: %d out of %d samples were correctly labeled." % (correct\_labels, y.size))  print('Accuracy score: {0:0.2f}'. format(correct\_labels/float(y.size)))  # Use elbow method to find optimal number of clusters  from sklearn.cluster import KMeans  cs = []  for i in range(1, 11):  kmeans = KMeans(n\_clusters = i, init = 'k-means++', max\_iter = 300, n\_init = 10, random\_state = 0)  kmeans.fit(X)  cs.append(kmeans.inertia\_)  plt.plot(range(1, 11), cs)  plt.title('The Elbow Method')  plt.xlabel('Number of clusters')  plt.ylabel('CS')  plt.show()  from sklearn.cluster import KMeans  kmeans = KMeans(n\_clusters=2,random\_state=0)  kmeans.fit(X)  labels = kmeans.labels\_  # check how many of the samples were correctly labeled  correct\_labels = sum(y == labels)  print("Result: %d out of %d samples were correctly labeled." % (correct\_labels, y.size))  print('Accuracy score: {0:0.2f}'. format(correct\_labels/float(y.size)))  # K-Means model with 3 different clusters  kmeans = KMeans(n\_clusters=3, random\_state=0)  kmeans.fit(X)  # check how many of the samples were correctly labeled  labels = kmeans.labels\_  correct\_labels = sum(y == labels)  print("Result: %d out of %d samples were correctly labeled." % (correct\_labels, y.size))  print('Accuracy score: {0:0.2f}'. format(correct\_labels/float(y.size)))  # K-Means model with 4 clusters  kmeans = KMeans(n\_clusters=4, random\_state=0)  kmeans.fit(X)  # check how many of the samples were correctly labeled  labels = kmeans.labels\_  correct\_labels = sum(y == labels)  print("Result: %d out of %d samples were correctly labeled." % (correct\_labels, y.size))  print('Accuracy score: {0:0.2f}'. format(correct\_labels/float(y.size)))  # K-Means model with 5 clusters  kmeans = KMeans(n\_clusters=5, random\_state=0)  kmeans.fit(X)  # check how many of the samples were correctly labeled  labels = kmeans.labels\_  correct\_labels = sum(y == labels)  print("Result: %d out of %d samples were correctly labeled." % (correct\_labels, y.size))  print('Accuracy score: {0:0.2f}'. format(correct\_labels/float(y.size))) |

Part V: Data Summary and Implications

Section E1: Accuracy

The model has an extremely high inertia of 15196.10078690495, as we discovered. Therefore, it doesn't seem like this model fits the data the best. We have a subpar classification accuracy of 0.37% with our unsupervised model and k = 2.

We were able to get significantly lower accuracy of 0.13%, 0.13%, and 0.03% using K =3, K =4, and K =5.

Section E2: Results and Implications

We used K-Means Clustering, the most well-liked unsupervised clustering method, in this project.

We used the elbow approach and discovered that 2 is an acceptable number of clusters to use for this data. K is the number of clusters. As a result, we altered the value of k and found that the classification accuracy was significantly lower. As a result, we can say that k = 5 has fewer clusters.

K-Means model with 2 different clusters Result: 3083 out of 8361 samples were correctly labeled.

Accuracy score: 0.37.

K-Means model with 3 different clusters Result: 1068 out of 8361 samples were correctly labeled.

Accuracy score: 0.13.

K-Means model with 4 clusters Result: 1091 out of 8361 samples were correctly labeled.

Accuracy score: 0.13.

K-Means model with 5 clusters Result: 1276 out of 8361 samples were correctly labeled.

Accuracy score: 0.15.

Section E3: Limitation

K-means will not produce the desired clusters if the cluster sizes are varied since it attempts to divide the clusters equally. I don't believe that is always the case, though. On the dataset below, I used k-means with k-means++ initialization.

When there is a cluster overlap, even distribution is likely to be an issue. K-means will then attempt to define the boundary roughly halfway between the cluster centers. The most significant restriction is that the user must initially define k (the number of clusters). Lastly, only numerical data can be handled by k-means.

Section E4: Recommendation

Clusters were successfully produced from our random data set. Different colors were used to represent each data cluster. The same code may be used to create clusters on different types of data, and we can even modify the algorithm's number of clusters. Then, based on the elbow technique, we hypothesized that adding more centroids would enhance clustering. Therefore, we get better clusters with improved information gain after selecting more clusters.

The final code portion, which only works for two-dimensional clustering, should not be run if we are clustering in more than two dimensions. Utilizing the dimension reduction method, this code is usable. So, if you apply these strategies to reduce the dataset to two dimensions, you may use the last code section to depict the clusters.

Part V: Panopto Video presentation

Section F:

You can view the session using the following link:  
<https://wgu.hosted.panopto.com/Panopto/Pages/Viewer.aspx?id=a110abea-d474-4ffa-b989-af3d017bcde2>

Section G: Sources

Grant, P. (2019). Understanding Multiple Regression; The fundamental basis behind this commonly used algorithm.

Medium.<https://towardsdatascience.com/understanding-multiple-regression-249b16bde83e>

Evgeniy R.(2020). 5 Stages of Data Preprocessing for K-means clustering

Medium. <https://medium.com/@evgen.ryzhkov/5-stages-of-data-preprocessing-for-k-means-clustering-b755426f9932>

Michael G. (2018). Understanding Boxplots.

Medium. <https://towardsdatascience.com/understanding-boxplots-5e2df7bcbd51>

Mr D. (2018). K-Mean clustering for Wine Quality Data

<https://www.kaggle.com/code/digvijaysingh16/k-mean-clustering-for-wine-quality-data>

Neelam T. (2020). What is K-means Clustering in Machine Learning?

<https://www.analyticssteps.com/blogs/what-k-means-clustering-machine-learning>

Andrea T.(2016). Introduction to K-means Clustering

<https://blogs.oracle.com/ai-and-datascience/post/introduction-to-k-means-clustering>

How to Swap Between Colorblind and Colorful Dashboards. (2016, July 5). YouTube.

<https://www.youtube.com/watch?v=3iNl7KMK8pM>

Knaflic, C. N. (2015). Storytelling with data: A data visualization guide for business professionals. Wiley .(Chapter 1, Chapters 3-5, Chapters 7 - 8)

<https://ebookcentral.proquest.com/lib/westerngovernorsebooks/reader.action?docID=4187267&ppg=1>

Knaflic, C. N. (2015). storytelling with data: A data visualization guide for business professionals.

Wiley .

(Chapter 1, Chapters 3-5, Chapters 7 - 8)

<https://ebookcentral.proquest.com/lib/westerngovernors-ebooks/reader.action?docID=4187267&ppg=1>