**CIS-660 Lab 1**

**Part 1 and Part 2**

**Submitted By:**

A person standing in front of a bridge

Description automatically generated with medium confidence

**Name: Mohamed Gani Mohamed Sulthan**

**CSU ID: 2811619**

**Grail Id: momohame@grail.eecs.csuohio.edu**

**Master’s in computer science**

**Part 1: Feature Selection:**

1. **Examine each attribute in the data file (vTargetMailCustomer.csv ) to select a set of features (Attributes) that would affect to predict future bike buyers.**

**Solution:**

I have clearly examined the vTargetMailCustomer.csv file to select the attributes that would affect to predict the feature bike buyers.

1. **Remove all the unnecessary attributes from the data file VTargetBuyerMailList.**

**Keys:**

GeographyKey, CustomerAlternateKey, Title, FirstName, MiddleName, LastName, NameStyle, BirthDate, Suffix, EmailAddress, SpanishEducation, FrenchEducation, SpanishOccupation, FrenchOccupation, AddressLine1, AddressLine2, Phone, DateFirstPurchase

1. **Create a new file VTargetBuyers with the selected Features only. Include ID and Class (BikeBuyer)**

**Solution:**

* Created a file with the selected Features and ID and Class (BikeBuyer).
* The attributes I selected here are relevant to predict the vTargetBuyers.

**Selected Attributes:** CustomerKey, GeographyKey, MaritalStatus, Gender, YearlyIncome, TotalChildren, NumberChildrenAtHome, EnglishEducation, EnglishOccupation, HouseOwnerFlag, NumberCarsOwned, CommuteDistance, Region, Age, BikeBuyer

Graphical user interface, application, table, Excel

Description automatically generated

1. **Determine data properties and data type for each selected feature**

**1) whether it is Discrete, or Continuous, and**

**2) whether it is Nominal, Ordinal, Interval, or Ratio for each attribute in your selection to determine correct preprocessing methods to transform input data correctly for Similarity Measures in Part3 of this Lab1 or Classification task later.**

**Discrete or Continuous**

CustomerKey -- Discrete

GeographyKey -- Discrete

MaritalStatus -- Discrete

Gender -- Discrete

YearlyIncome -- Continuous

TotalChildren -- Continuous

NumberChildrenAtHome -- Continuous

EnglishEducation -- Continuous

EnglishOccupation -- Continuous

HouseOwnerFlag -- Discrete

NumberCarsOwned -- Continuous

CommuteDistance -- Continuous

Region -- Continuous

Age -- Continuous

BikeBuyer -- Continuous

**Nominal, Ordinal, Interval or Ratio**

CustomerKey -- Nominal

GeographyKey -- Nominal

MaritalStatus -- Nominal

Gender -- Nominal

YearlyIncome -- Ratio

TotalChildren -- Ordinal

NumberChildrenAtHome -- Ordinal

EnglishEducation -- Ordinal

EnglishOccupation -- Ordinal

HouseOwnerFlag -- Ratio

NumberCarsOwned -- Ratio

CommuteDistance -- Interval

Region -- Nominal

Age -- Ratio

BikeBuyer -- Nominal

**Part 2: Data Preprocessing and Transformation**

For each selected feature in Part1,

1. For each feature, determine which data preprocessing methods and transformation techniques listed below should be done in which sequence depending on the data type and properties of each feature (attribute) that you identified in Part 1 to calculate dissimilarity/similarity distances between two objects in Part 3.

2. Perform the data preprocessing/transforming tasks for each feature with some of these transformation methods.

* Binarization (One Hot Encoding) for each nominal (discrete/discretized without ordering) attribute
* Discretization for a Continuous attribute or Numeric Data with too many different values if necessary.
* Normalization or Standardization method for each interval/ordinal/discretized attribute with ordering or any ratio attribute
* Handling Null values
* Identify Outliers with 1.5IQR Method to Replace
* Perform Random Sampling, Bootstrap Sampling with Replacement
* Mean/Variance/Standard Deviation for Ordinal Numeric attributes
* Normalization or Standardization
* Discretization (Binning/Histogram) on Continuous attributes or Categorical
* Attributes with too many different values
* Get Median of the Grouped data (those attributes that were transformed into
* Histograms)
* Binarization (One Hot Encoding)

**Jupyter notebook execution:**

Required imports to do the preprocessing techniques

# import required modules

import pandas as pd

# reading csv file

dataFrame = pd.read\_csv(r"C:\Users\Gani\Downloads\vTargetBuyers.csv")

The above imports and reading files are used to find the null values and all the preprocessing techniques

1. **Selected Attributes:**

* For the selected attributes, we are doing the null values check, whether in any of the columns contains null values or any data missing in the cell.
* This was implemented by the isnull().sum() function in the python.
* Below is the screenshot for finding null checks

Graphical user interface, text, application, email

Description automatically generated

**There are no null values or any missing values in the selected attribute list.**

So, we can move the next step and started working on the preprocessing steps.

1. **Perform the data preprocessing**

**Converting categorical data into binary data**

**HouseownerFlag and BikeBuyer Attributes:**

It is already in Binarization state 0 and 1. So we don’t need to convert anything to process at predicting in future.

**Binarization:**

* In this binarization, we are converting the default two values to 0 and 1
* For instance Single and Married to 0 and 1 Male and Female to 1 and 0

**Binarization attributes**

**MaritalStatus** and **Gender**

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

**Converting Binarization:**

* Here I’m trying to convert some of the attributes to Binarization format to easily use in the future.
* **Assigning the values if NumberCarsOwned** and **TotalChildren** and **NumberChildrenAtHome**. I just making sure if Cars, Children and ChildrenAtHome are greater than or equal to one, I’m making it as 1 and 0 as 0.

**Converting Binarization Attributes:**

**TotalChildren**

Graphical user interface, text, application, email

Description automatically generated

**NumberChildrenAtHome**

Graphical user interface, text, application, email

Description automatically generated

**NumberCarsOwned**

Graphical user interface, text, application, email

Description automatically generated

**One Hot Encoding:**

* + Here I’m doing one hot encoding for the attributes having more than one data, which I couldn’t do on Binarization.
  + I’m taking a data frame and using the get dummies to print in the separated columns
  + Then using cumsum function to add it in to the single column
  + So, I can use it in the future to predict on more accuracy

**One hot encoding attributes**

**EnglishEducation**

Graphical user interface, text, application

Description automatically generated

**EnglishOccupation**

Graphical user interface, text, application

Description automatically generated

**CommuteDistance**

Graphical user interface, text, application

Description automatically generated

**Region**

Graphical user interface, text, application

Description automatically generated

**Normalization:**

* Making the income as an easily readable value. I did Data normalization by using min max feature scaling.
* The min-max approach rescales the feature to a hard and fast range of [0,1] by subtracting the minimum value of the feature then dividing by the range. We can apply the min-max scaling in Pandas using the .min() and .max() methods.

**Normalization Attributes:**

**Yearly Income**

Graphical user interface, text, application

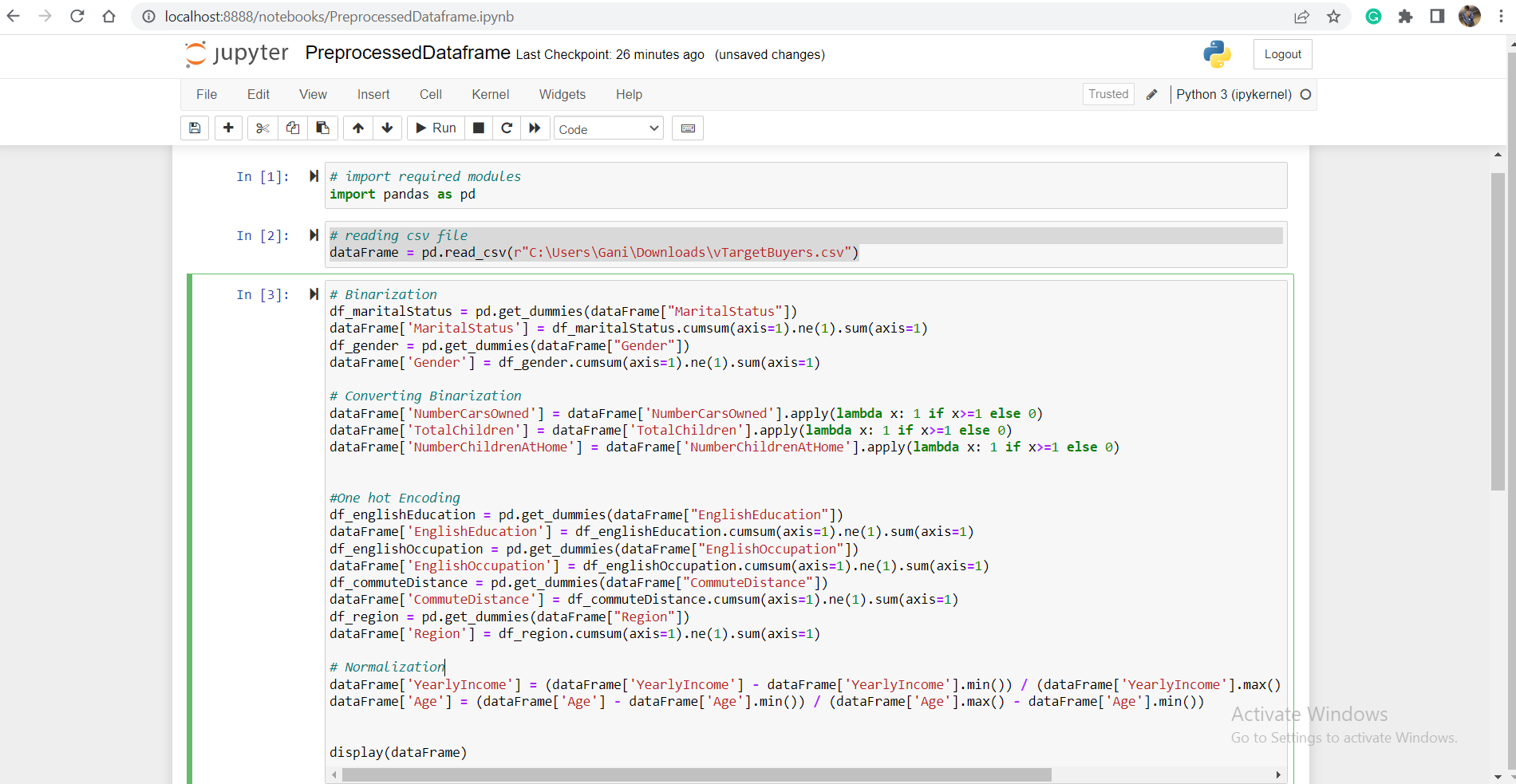
Description automatically generated

**Age**

Graphical user interface, text, application, email

Description automatically generated

I have made this report one by one for everything to show the clear steps of preprocessing. In the below, I have created a preprocessedDataframe.py file and added all the changes here.



Table

Description automatically generated

Table

Description automatically generated