KPMG_VI_Task1

June 10, 2023

1 KPMG VIRTUAL INTERNSHIP PROJECT

1.1 Task 1: Data Quality Assessment

Assessment of data quality and completeness in preparation for analysis

1.1.1 Company Name: Sprocket Central Pty Ltd

Import Required Libraries

[1]:
```
import pandas as pd
```

[11]:
```
#Reading the data
data=pd.read_excel('KPMG_VI.xlsx')
```

[14]:
```
#Accessing the sheets in the excel file
Transactions = pd.read_excel(r"C:\Users\lenovo\Downloads\KPMG VI\KPMG_VI.
 ↪xlsx",sheet_name= 'Transactions')
NewCustomerList = pd.read_excel(r"C:\Users\lenovo\Downloads\KPMG VI\KPMG_VI.
 ↪xlsx",sheet_name= 'NewCustomerList')
CustomerDemographic = pd.read_excel(r"C:\Users\lenovo\Downloads\KPMG VI\KPMG_VI.
 ↪xlsx",sheet_name= 'CustomerDemographic')
CustomerAddress = pd.read_excel(r"C:\Users\lenovo\Downloads\KPMG VI\KPMG_VI.
 ↪xlsx", sheet_name='CustomerAddress')
```

```
C:\Users\lenovo\AppData\Local\Temp\ipykernel_15424\2069347996.py:3:
FutureWarning: Inferring datetime64[ns] from data containing strings is
deprecated and will be removed in a future version. To retain the old behavior
explicitly pass Series(data, dtype=datetime64[ns])
  NewCustomerList = pd.read_excel(r"C:\Users\lenovo\Downloads\KPMG
VI\KPMG_VI.xlsx",sheet_name= 'NewCustomerList')
C:\Users\lenovo\AppData\Local\Temp\ipykernel_15424\2069347996.py:4:
FutureWarning: Inferring datetime64[ns] from data containing strings is
deprecated and will be removed in a future version. To retain the old behavior
explicitly pass Series(data, dtype=datetime64[ns])
  CustomerDemographic = pd.read_excel(r"C:\Users\lenovo\Downloads\KPMG
VI\KPMG_VI.xlsx",sheet_name= 'CustomerDemographic')
```

1.1.2 Transaction Dataset

[16]:
```
Transactions.head()
```

[16]:

	transaction_id	product_id	customer_id	transaction_date	online_order	order_status	brand	product_line	product_class	product_size	list_price	standard_cost	product_first_sold_date
0	1	2	2950	2017-02-25	0.0	Approved	Solex	Standard	medium	medium	71.49	53.62	41245.0
1	2	3	3120	2017-05-21	1.0	Approved	Trek Bicycles	Standard	medium	large	2091.47	388.92	41701.0
2	3	37	402	2017-10-16	0.0	Approved	OHM Cycles	Standard	low	medium	1793.43	248.82	36361.0
3	4	88	3135	2017-08-31	0.0	Approved	Norco Bicycles	Standard	medium	medium	1198.46	381.10	36145.0
4	5	78	787	2017-10-01	1.0	Approved	Giant Bicycles	Standard	medium	large	1765.30	709.48	42226.0

[17]:
```
Transactions.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 13 columns):
 #   Column                   Non-Null Count  Dtype
---  ------                   --------------  -----
 0   transaction_id           20000 non-null  int64
 1   product_id               20000 non-null  int64
 2   customer_id              20000 non-null  int64
 3   transaction_date         20000 non-null  datetime64[ns]
 4   online_order             19640 non-null  float64
 5   order_status             20000 non-null  object
 6   brand                    19803 non-null  object
 7   product_line             19803 non-null  object
 8   product_class            19803 non-null  object
 9   product_size             19803 non-null  object
 10  list_price               20000 non-null  float64
 11  standard_cost            19803 non-null  float64
 12  product_first_sold_date  19803 non-null  float64
dtypes: datetime64[ns](1), float64(4), int64(3), object(5)
memory usage: 2.0+ MB
```

[18]:
```
# Checking the shape
Transactions.shape
```

[18]:
```
(20000, 13)
```

[19]:
```
#Checking for null values
Transactions.isnull().sum()
```

[19]:
```
transaction_id               0
product_id                   0
customer_id                  0
transaction_date             0
online_order               360
order_status                 0
brand                      197
product_line               197
product_class              197
product_size               197
list_price                   0
standard_cost              197
product_first_sold_date    197
dtype: int64
```

There are missing values in 7 columns. According to the need, the missing values in the columns can be dropped or treated.

[20]:
```
#Checking for duplicate values
Transactions.duplicated().sum()
```

[20]:
```
0
```

There is no duplicate values in the Transaction dataset. All the values in the data are unique.

1.1.3 Checking all columns in the dataset

[22]:
```
Transactions.columns
```

[22]:
```
Index(['transaction_id', 'product_id', 'customer_id', 'transaction_date',
       'online_order', 'order_status', 'brand', 'product_line',
       'product_class', 'product_size', 'list_price', 'standard_cost',
       'product_first_sold_date'],
      dtype='object')
```

Reviewing critical columns to extract valuable insights.

[23]:
```
Transactions['order_status'].value_counts()
```

[23]:
```
Approved     19821
Cancelled      179
Name: order_status, dtype: int64
```

[24]:
```
Transactions['brand'].value_counts()
```

[24]:
```
Solex             4253
Giant Bicycles    3312
WeareA2B          3295
OHM Cycles        3043
Trek Bicycles     2990
Norco Bicycles    2910
Name: brand, dtype: int64
```

[25]:
```
Transactions['product_line'].value_counts()
```

[25]:
```
Standard    14176
Road         3970
Touring      1234
Mountain      423
Name: product_line, dtype: int64
```

[26]:
```
Transactions['product_class'].value_counts()
```

[26]:
```
medium    13826
high       3013
low        2964
Name: product_class, dtype: int64
```

[27]:
```
Transactions['product_size'].value_counts()
```

[27]:
```
medium    12990
large      3976
small      2837
Name: product_size, dtype: int64
```

[28]:
```
Transactions['product_first_sold_date'].value_counts()
```

[28]:
```
33879.0    234
41064.0    229
37823.0    227
39880.0    222
38216.0    220
           …
41848.0    169
42404.0    168
41922.0    166
```

```
37659.0    163
34586.0    162
Name: product_first_sold_date, Length: 100, dtype: int64
```

The product_first_sold_date column is not in the format of date.

[30]:
```
# Converting the product_first_sold_date column from integer to datetime
Transactions['product_first_sold_date'] = pd.
 ↪to_datetime(Transactions['product_first_sold_date'], unit='s')
Transactions['product_first_sold_date'].head(20)
```

[30]:
```
0    1970-01-01 11:27:25
1    1970-01-01 11:35:01
2    1970-01-01 10:06:01
3    1970-01-01 10:02:25
4    1970-01-01 11:43:46
5    1970-01-01 10:50:31
6    1970-01-01 09:29:25
7    1970-01-01 11:05:15
8    1970-01-01 09:17:35
9    1970-01-01 10:36:56
10   1970-01-01 11:19:44
11   1970-01-01 11:42:52
12   1970-01-01 09:35:27
13   1970-01-01 09:36:26
14   1970-01-01 10:36:33
15   1970-01-01 10:31:13
16   1970-01-01 10:36:46
17   1970-01-01 09:24:48
18   1970-01-01 11:05:15
19   1970-01-01 10:22:17
Name: product_first_sold_date, dtype: datetime64[ns]
```

Identified an issue with the product_first_sold_date column, as it inaccurately represents orders being placed on the same date but at different times. Determined that this column lacks meaningful information.

1.1.4 New Customer List Dataset

[31]:
```
NewCustomerList.head()
```

[31]:

	first_name	last_name	gender	past_3_years_bike_related_purchases
0	Chickie	Brister	Male	86
1	Morly	Genery	Male	69
2	Ardelis	Forrester	Female	10

```
3     Lucine      Stutt  Female                                    64
4    Melinda     Hadlee  Female                                    34

         DOB                  job_title job_industry_category  \
0 1957-07-12            General Manager         Manufacturing
1 1970-03-22         Structural Engineer              Property
2 1974-08-28      Senior Cost Accountant    Financial Services
3 1979-01-28  Account Representative III         Manufacturing
4 1965-09-21           Financial Analyst    Financial Services

      wealth_segment deceased_indicator owns_car  …  state    country  \
0      Mass Customer                  N      Yes  …    QLD  Australia
1      Mass Customer                  N       No  …    NSW  Australia
2  Affluent Customer                  N       No  …    VIC  Australia
3  Affluent Customer                  N      Yes  …    QLD  Australia
4  Affluent Customer                  N       No  …    NSW  Australia

   property_valuation Unnamed: 16 Unnamed: 17  Unnamed: 18  Unnamed: 19  \
0                   6        0.96       1.200        1.500      1.27500
1                  11        0.54       0.540        0.675      0.57375
2                   5        0.78       0.780        0.780      0.78000
3                   1        1.02       1.275        1.275      1.27500
4                   9        0.50       0.500        0.625      0.62500

   Unnamed: 20  Rank     Value
0            1     1  1.718750
1            1     1  1.718750
2            1     1  1.718750
3            4     4  1.703125
4            4     4  1.703125

[5 rows x 23 columns]
```

Identified five columns labeled as "Unnamed" that lack a specific purpose or identifiable content, rendering them irrelevant for analysis. So dropping these columns from the dataset.

[32]:
```
#Dropping the unnamed columns
NewCustomerList.drop(['Unnamed: 16', 'Unnamed: 17', 'Unnamed: 18',
       'Unnamed: 19', 'Unnamed: 20'], axis=1, inplace=True)
```

[33]:
```
NewCustomerList.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 18 columns):
 #   Column                               Non-Null Count  Dtype
```

```
---  ------                                 --------------  -----
 0   first_name                             1000 non-null   object
 1   last_name                              971 non-null    object
 2   gender                                 1000 non-null   object
 3   past_3_years_bike_related_purchases    1000 non-null   int64
 4   DOB                                    983 non-null    datetime64[ns]
 5   job_title                              894 non-null    object
 6   job_industry_category                  835 non-null    object
 7   wealth_segment                         1000 non-null   object
 8   deceased_indicator                     1000 non-null   object
 9   owns_car                               1000 non-null   object
 10  tenure                                 1000 non-null   int64
 11  address                                1000 non-null   object
 12  postcode                               1000 non-null   int64
 13  state                                  1000 non-null   object
 14  country                                1000 non-null   object
 15  property_valuation                     1000 non-null   int64
 16  Rank                                   1000 non-null   int64
 17  Value                                  1000 non-null   float64
dtypes: datetime64[ns](1), float64(1), int64(5), object(11)
memory usage: 140.8+ KB
```

[34]:
```
# Checking the shape of the dataset
NewCustomerList.shape
```

[34]:
```
(1000, 18)
```

[35]:
```
#Checking for null values
NewCustomerList.isnull().sum()
```

[35]:
```
first_name                             0
last_name                             29
gender                                 0
past_3_years_bike_related_purchases    0
DOB                                   17
job_title                            106
job_industry_category                165
wealth_segment                         0
deceased_indicator                     0
owns_car                               0
tenure                                 0
address                                0
postcode                               0
state                                  0
country                                0
property_valuation                     0
Rank                                   0
```

```
Value                                0
dtype: int64
```

There are missing values in 4 columns. These values can be treated according to the importance of the columns.

[36]:
```
#Checking for duplicate values
NewCustomerList.duplicated().sum()
```

[36]:
```
0
```

There is no duplicate values in the dataset.

[37]:
```
#Checking for uniquess of each column
NewCustomerList.nunique()
```

[37]:
```
first_name                           940
last_name                            961
gender                                 3
past_3_years_bike_related_purchases  100
DOB                                  958
job_title                            184
job_industry_category                  9
wealth_segment                         3
deceased_indicator                     1
owns_car                               2
tenure                                23
address                             1000
postcode                             522
state                                  3
country                                1
property_valuation                    12
Rank                                 324
Value                                324
dtype: int64
```

Reviewing critical columns to extract valuable insights.

[38]:
```
NewCustomerList.columns
```

[38]:
```
Index(['first_name', 'last_name', 'gender',
       'past_3_years_bike_related_purchases', 'DOB', 'job_title',
       'job_industry_category', 'wealth_segment', 'deceased_indicator',
       'owns_car', 'tenure', 'address', 'postcode', 'state', 'country',
       'property_valuation', 'Rank', 'Value'],
      dtype='object')
```

[39]:
```
NewCustomerList['gender'].value_counts()
```

[39]:
```
Female    513
Male      470
U          17
Name: gender, dtype: int64
```

The presence of 17 records with unknown gender introduces a minor impact on the overall gender distribution among the customers.

[40]:
```
NewCustomerList['job_industry_category'].value_counts()
```

[40]:
```
Financial Services    203
Manufacturing         199
Health                152
Retail                 78
Property               64
IT                     51
Entertainment          37
Argiculture            26
Telecommunications     25
Name: job_industry_category, dtype: int64
```

[41]:
```
NewCustomerList['wealth_segment'].value_counts()
```

[41]:
```
Mass Customer        508
High Net Worth       251
Affluent Customer    241
Name: wealth_segment, dtype: int64
```

[42]:
```
NewCustomerList['state'].value_counts()
```

[42]:
```
NSW    506
VIC    266
QLD    228
Name: state, dtype: int64
```

[43]:
```
NewCustomerList['owns_car'].value_counts()
```

[43]:
```
No     507
Yes    493
Name: owns_car, dtype: int64
```

[44]:
```
NewCustomerList['deceased_indicator'].value_counts()
```

[44]:
```
N    1000
Name: deceased_indicator, dtype: int64
```

1.1.5 Customer Demographic Dataset

[45]:
```
CustomerDemographic.head()
```

[45]:
```
   customer_id      first_name  last_name  gender  \
0            1         Laraine  Medendorp       F
1            2             Eli    Bockman    Male
2            3          Arlin     Dearle    Male
3            4          Talbot        NaN    Male
4            5  Sheila-kathryn     Calton  Female

   past_3_years_bike_related_purchases        DOB                job_title  \
0                                   93 1953-10-12     Executive Secretary
1                                   81 1980-12-16  Administrative Officer
2                                   61 1954-01-20      Recruiting Manager
3                                   33 1961-10-03                     NaN
4                                   56 1977-05-13           Senior Editor

  job_industry_category     wealth_segment deceased_indicator  \
0                Health      Mass Customer                  N
1    Financial Services      Mass Customer                  N
2              Property      Mass Customer                  N
3                    IT      Mass Customer                  N
4                   NaN  Affluent Customer                  N

                                           default owns_car  tenure
0                                              "'      Yes    11.0
1                  <script>alert('hi')</script>       Yes    16.0
2                            2018-02-01 00:00:00      Yes    15.0
3  () { _; } >_[$($())] { touch /tmp/blns.shellsh…       No     7.0
4                                             NIL      Yes     8.0
```

[46]:
```
CustomerDemographic.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 4000 entries, 0 to 3999
Data columns (total 13 columns):
 #   Column                               Non-Null Count  Dtype
---  ------                               --------------  -----
 0   customer_id                          4000 non-null   int64
 1   first_name                           4000 non-null   object
 2   last_name                            3875 non-null   object
 3   gender                               4000 non-null   object
 4   past_3_years_bike_related_purchases  4000 non-null   int64
 5   DOB                                  3913 non-null   datetime64[ns]
 6   job_title                            3494 non-null   object
 7   job_industry_category                3344 non-null   object
```

```
 8   wealth_segment                       4000 non-null   object
 9   deceased_indicator                   4000 non-null   object
 10  default                              3698 non-null   object
 11  owns_car                             4000 non-null   object
 12  tenure                               3913 non-null   float64
dtypes: datetime64[ns](1), float64(1), int64(2), object(9)
memory usage: 406.4+ KB
```

[47]:
```
# Checking null values
CustomerDemographic.isnull().sum()
```

[47]:
```
customer_id                            0
first_name                             0
last_name                            125
gender                                 0
past_3_years_bike_related_purchases    0
DOB                                   87
job_title                            506
job_industry_category                656
wealth_segment                         0
deceased_indicator                     0
default                              302
owns_car                               0
tenure                                87
dtype: int64
```

The dataset contains missing values in six columns. To handle this, a strategic approach was adopted by dropping the least significant columns and applying appropriate treatment methods to the essential columns.

[48]:
```
#Checking duplicate values
CustomerDemographic.duplicated().sum()
```

[48]:
```
0
```

There are no duplicate values

Reviewing critical columns to extract valuable insights.

[49]:
```
CustomerDemographic.columns
```

[49]:
```
Index(['customer_id', 'first_name', 'last_name', 'gender',
       'past_3_years_bike_related_purchases', 'DOB', 'job_title',
       'job_industry_category', 'wealth_segment', 'deceased_indicator',
       'default', 'owns_car', 'tenure'],
      dtype='object')
```

[50]:
```
CustomerDemographic['gender'].value_counts()
```

[50]:
```
Female    2037
Male      1872
U           88
F            1
Femal        1
M            1
Name: gender, dtype: int64
```

There are 88 instances with unspecified gender and certain categories that require correct titling.

Renaming the Categories.

[51]:
```
CustomerDemographic['gender'] = CustomerDemographic['gender'].
↪replace('F','Female').replace('M','Male').replace('Femal','Female').
↪replace('U','Unspecified')
```

[52]:
```
CustomerDemographic['gender'].value_counts()
```

[52]:
```
Female         2039
Male           1873
Unspecified      88
Name: gender, dtype: int64
```

[53]:
```
CustomerDemographic['past_3_years_bike_related_purchases'].value_counts()
```

[53]:
```
16    56
19    56
67    54
20    54
2     50
      ..
8     28
95    27
85    27
86    27
92    24
Name: past_3_years_bike_related_purchases, Length: 100, dtype: int64
```

[54]:
```
CustomerDemographic['job_title'].value_counts()
```

[54]:
```
Business Systems Development Analyst    45
Tax Accountant                          44
Social Worker                           44
```

```
Internal Auditor                          42
Recruiting Manager                        41
                                          ..
Database Administrator I                   4
Health Coach I                             3
Health Coach III                           3
Research Assistant III                     3
Developer I                                1
Name: job_title, Length: 195, dtype: int64
```

[55]:
```
CustomerDemographic['job_industry_category'].value_counts()
```

[55]:
```
Manufacturing         799
Financial Services    774
Health                602
Retail                358
Property              267
IT                    223
Entertainment         136
Argiculture           113
Telecommunications     72
Name: job_industry_category, dtype: int64
```

[56]:
```
CustomerDemographic['wealth_segment'].value_counts()
```

[56]:
```
Mass Customer        2000
High Net Worth       1021
Affluent Customer     979
Name: wealth_segment, dtype: int64
```

[57]:
```
CustomerDemographic['deceased_indicator'].value_counts()
```

[57]:
```
N    3998
Y       2
Name: deceased_indicator, dtype: int64
```

[58]:
```
CustomerDemographic['default'].value_counts()
```

[58]:
```
100                                       113
1                                         112
-1                                        111
-100                                       99
Ù¡Ù¢Ù£                                      53
                                          ...
testâ testâ«                               31
/dev/null; touch /tmp/blns.fail ; echo     30
âªâªtestâª                                  29
```

```
ì¸ëë°í ë¥´                                  27
,ãã»:*:ã»ãâ( â» Ï â» )ãã»:*:ã»ãâ           25
Name: default, Length: 90, dtype: int64
```

The data values in the column are inconsistent, containing a mixture of integers and multiple characters, rendering the column insufficient and non-informative for analysis purposes. Hence dropping the column.

[59]:
```
CustomerDemographic = CustomerDemographic.drop('default', axis=1)
```

[60]:
```
CustomerDemographic.head(5)
```

[60]:

	customer_id	first_name	last_name	gender	\
0	1	Laraine	Medendorp	Female	
1	2	Eli	Bockman	Male	
2	3	Arlin	Dearle	Male	
3	4	Talbot	NaN	Male	
4	5	Sheila-kathryn	Calton	Female	

	past_3_years_bike_related_purchases	DOB	job_title	\
0	93	1953-10-12	Executive Secretary	
1	81	1980-12-16	Administrative Officer	
2	61	1954-01-20	Recruiting Manager	
3	33	1961-10-03	NaN	
4	56	1977-05-13	Senior Editor	

	job_industry_category	wealth_segment	deceased_indicator	owns_car	tenure
0	Health	Mass Customer	N	Yes	11.0
1	Financial Services	Mass Customer	N	Yes	16.0
2	Property	Mass Customer	N	Yes	15.0
3	IT	Mass Customer	N	No	7.0
4	NaN	Affluent Customer	N	Yes	8.0

The 'default' column has been dropped.

[61]:
```
CustomerDemographic['owns_car'].value_counts()
```

[61]:
```
Yes    2024
No     1976
Name: owns_car, dtype: int64
```

[62]:
```
CustomerDemographic['tenure'].value_counts()
```

[62]:
```
7.0     235
5.0     228
11.0    221
10.0    218
```

```
16.0    215
8.0     211
18.0    208
12.0    202
9.0     200
14.0    200
6.0     192
13.0    191
4.0     191
17.0    182
15.0    179
1.0     166
3.0     160
19.0    159
2.0     150
20.0     96
22.0     55
21.0     54
Name: tenure, dtype: int64
```

1.1.6 Customer Address Dataset

[63]:
```
CustomerAddress.head()
```

[63]:
```
   customer_id              address  postcode            state    country  \
0            1   060 Morning Avenue      2016  New South Wales  Australia
1            2  6 Meadow Vale Court      2153  New South Wales  Australia
2            4   0 Holy Cross Court      4211              QLD  Australia
3            5  17979 Del Mar Point      2448  New South Wales  Australia
4            6     9 Oakridge Court      3216              VIC  Australia

   property_valuation
0                  10
1                  10
2                   9
3                   4
4                   9
```

[64]:
```
CustomerAddress.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3999 entries, 0 to 3998
Data columns (total 6 columns):
 #   Column              Non-Null Count  Dtype
---  ------              --------------  -----
 0   customer_id         3999 non-null   int64
```

```
 1   address              3999 non-null   object
 2   postcode             3999 non-null   int64
 3   state                3999 non-null   object
 4   country              3999 non-null   object
 5   property_valuation   3999 non-null   int64
dtypes: int64(3), object(3)
memory usage: 187.6+ KB
```

[65]:
```
CustomerAddress.isnull().sum()
```

[65]:
```
customer_id          0
address              0
postcode             0
state                0
country              0
property_valuation   0
dtype: int64
```

There are no null values which means the dataset quality is good.

[66]:
```
#Checking for duplicate values
CustomerAddress.duplicated().sum()
```

[66]:
```
0
```

[67]:
```
#Checking for uniqueness of each column
CustomerAddress.nunique()
```

[67]:
```
customer_id          3999
address              3996
postcode              873
state                   5
country                 1
property_valuation     12
dtype: int64
```

Reviewing critical columns to extract valuable insights.

[68]:
```
CustomerAddress.columns
```

[68]:
```
Index(['customer_id', 'address', 'postcode', 'state', 'country',
       'property_valuation'],
      dtype='object')
```

[69]:
```
CustomerAddress['postcode'].value_counts()
```

[69]:
```
2170    31
2155    30
2145    30
2153    29
3977    26
        ..
3808     1
3114     1
4721     1
4799     1
3089     1
Name: postcode, Length: 873, dtype: int64
```

[70]:
```
CustomerAddress['state'].value_counts()
```

[70]:
```
NSW                2054
VIC                 939
QLD                 838
New South Wales      86
Victoria             82
Name: state, dtype: int64
```

[71]:
```
CustomerAddress['country'].value_counts()
```

[71]:
```
Australia    3999
Name: country, dtype: int64
```

[72]:
```
CustomerAddress['property_valuation'].value_counts()
```

[72]:
```
9     647
8     646
10    577
7     493
11    281
6     238
5     225
4     214
12    195
3     186
1     154
2     143
Name: property_valuation, dtype: int64
```

All columns demonstrate consistent and accurate information throughout the dataset.

[ ]:
```

```