01_customer_segmentation

September 23, 2019

1 Customer Segmentation

1.1 Cohort Analyse

[87]:
```
import numpy as np
import pandas as pd
import os
import datetime as dt
import seaborn as sns
import matplotlib.pyplot as plt
```

1.1.1 Dataset

UCI Maschine Learning Repository

[88]:
```
print(os.getcwd())
#online = pd.read_excel('../data/Online Retail.xlsx')
```

```
/home/hans/python_codes/DataCamp/customer_segmentation
```

- Top 5 rows of data

[89]:
```
online.head(5)
```

[89]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	InvoiceMonth
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom	2010-12-01
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	2010-12-01
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom	2010-12-01
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	2010-12-01
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	2010-12-01

```
  CohortMonth  CohortIndex
0  2010-12-01          1.0
1  2010-12-01          1.0
2  2010-12-01          1.0
3  2010-12-01          1.0
4  2010-12-01          1.0
```

1.1.2 Assign acquisition month cohort

[90]:
```
def get_month(x): return dt.datetime(x.year, x.month, 1)

online['InvoiceMonth'] = online['InvoiceDate'].apply(get_month)
grouping = online.groupby('CustomerID')['InvoiceMonth']
online['CohortMonth'] = grouping.transform('min')
online.head()
```

[90]:
```
  InvoiceNo StockCode                          Description  Quantity  \
0    536365    85123A   WHITE HANGING HEART T-LIGHT HOLDER         6
1    536365     71053                  WHITE METAL LANTERN         6
2    536365    84406B       CREAM CUPID HEARTS COAT HANGER         8
3    536365    84029G  KNITTED UNION FLAG HOT WATER BOTTLE         6
4    536365    84029E       RED WOOLLY HOTTIE WHITE HEART.         6

          InvoiceDate  UnitPrice  CustomerID         Country InvoiceMonth  \
0 2010-12-01 08:26:00       2.55     17850.0  United Kingdom   2010-12-01
1 2010-12-01 08:26:00       3.39     17850.0  United Kingdom   2010-12-01
2 2010-12-01 08:26:00       2.75     17850.0  United Kingdom   2010-12-01
3 2010-12-01 08:26:00       3.39     17850.0  United Kingdom   2010-12-01
4 2010-12-01 08:26:00       3.39     17850.0  United Kingdom   2010-12-01

  CohortMonth  CohortIndex
0  2010-12-01          1.0
1  2010-12-01          1.0
2  2010-12-01          1.0
3  2010-12-01          1.0
4  2010-12-01          1.0
```

1.1.3 Extract integer values from data

Define function to extract year , month and day integer values. We will use it throughout the course.

[91]:
```
def get_date_int(df, column):
    year = df[column].dt.year
```

```
    month = df[column].dt.month
    day = df[column].dt.day
    return year, month, day
```

1.1.4 Assign time offset value

[92]:
```
invoice_year, invoice_month, _ = get_date_int(online, 'InvoiceMonth')
cohort_year, cohort_month, _ = get_date_int(online, 'CohortMonth')

years_diff = invoice_year - cohort_year
months_diff = invoice_month - cohort_month

online['CohortIndex'] = years_diff * 12 + months_diff + 1
online.head(5)
```

[92]:

	InvoiceNo	StockCode	Description	Quantity	InvoiceDate	UnitPrice	CustomerID	Country	InvoiceMonth	CohortMonth	CohortIndex
0	536365	85123A	WHITE HANGING HEART T-LIGHT HOLDER	6	2010-12-01 08:26:00	2.55	17850.0	United Kingdom	2010-12-01	2010-12-01	1.0
1	536365	71053	WHITE METAL LANTERN	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	2010-12-01	2010-12-01	1.0
2	536365	84406B	CREAM CUPID HEARTS COAT HANGER	8	2010-12-01 08:26:00	2.75	17850.0	United Kingdom	2010-12-01	2010-12-01	1.0
3	536365	84029G	KNITTED UNION FLAG HOT WATER BOTTLE	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	2010-12-01	2010-12-01	1.0
4	536365	84029E	RED WOOLLY HOTTIE WHITE HEART.	6	2010-12-01 08:26:00	3.39	17850.0	United Kingdom	2010-12-01	2010-12-01	1.0

1.1.5 Count monthly active customers from each cohort

[93]:
```
grouping = online.groupby(['CohortMonth', 'CohortIndex'])
cohort_data = grouping['CustomerID'].apply(pd.Series.nunique)
cohort_data = cohort_data.reset_index()
cohort_counts = cohort_data.pivot(index='CohortMonth',
    columns='CohortIndex',
    values='CustomerID')
```

1.2 Calculate cohort metrics

1.2.1 Customer retention: cohort_counts table

- How many customers originally in each cohort? (Column 1)
- How many of them were active in following months? (Column 2-13)

[94]:
```
cohort_counts.head()
```

[94]:
```
CohortIndex   1.0    2.0    3.0    4.0    5.0    6.0    7.0    8.0    9.0   \
CohortMonth
2010-12-01   948.0  362.0  317.0  367.0  341.0  376.0  360.0  336.0  336.0
2011-01-01   421.0  101.0  119.0  102.0  138.0  126.0  110.0  108.0  131.0
2011-02-01   380.0   94.0   73.0  106.0  102.0   94.0   97.0  107.0   98.0
2011-03-01   440.0   84.0  112.0   96.0  102.0   78.0  116.0  105.0  127.0
2011-04-01   299.0   68.0   66.0   63.0   62.0   71.0   69.0   78.0   25.0

CohortIndex   10.0   11.0   12.0   13.0
CohortMonth
2010-12-01   374.0  354.0  474.0  260.0
2011-01-01   146.0  155.0   63.0    NaN
2011-02-01   119.0   35.0    NaN    NaN
2011-03-01    39.0    NaN    NaN    NaN
2011-04-01     NaN    NaN    NaN    NaN
```

1.2.2 Calculate Retention rate

- Store the first column as cohort_sizes

[95]:
```
cohort_sizes = cohort_counts.iloc[:,0]
```

- Divide all values in the cohort_counts table by cohort_sizes

[96]:
```
retention = cohort_counts.divide(cohort_sizes, axis=0)
```

- Review the retention table

Retention table

[97]:
```
retention.round(3) * 100
```

[97]:
```
CohortIndex   1.0   2.0   3.0   4.0   5.0   6.0   7.0   8.0   9.0   10.0  \
CohortMonth
2010-12-01   100.0  38.2  33.4  38.7  36.0  39.7  38.0  35.4  35.4  39.5
2011-01-01   100.0  24.0  28.3  24.2  32.8  29.9  26.1  25.7  31.1  34.7
2011-02-01   100.0  24.7  19.2  27.9  26.8  24.7  25.5  28.2  25.8  31.3
2011-03-01   100.0  19.1  25.5  21.8  23.2  17.7  26.4  23.9  28.9   8.9
```

```
2011-04-01   100.0  22.7  22.1  21.1  20.7  23.7  23.1  26.1   8.4   NaN
2011-05-01   100.0  23.7  17.2  17.2  21.5  24.4  26.5  10.4   NaN   NaN
2011-06-01   100.0  20.9  18.7  27.2  24.7  33.6  10.2   NaN   NaN   NaN
2011-07-01   100.0  20.9  20.4  23.0  27.2  11.5   NaN   NaN   NaN   NaN
2011-08-01   100.0  25.1  25.1  25.1  13.8   NaN   NaN   NaN   NaN   NaN
2011-09-01   100.0  29.9  32.6  12.1   NaN   NaN   NaN   NaN   NaN   NaN
2011-10-01   100.0  26.4  13.1   NaN   NaN   NaN   NaN   NaN   NaN   NaN
2011-11-01   100.0  13.4   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN
2011-12-01   100.0   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN
```

```
CohortIndex  11.0  12.0  13.0
CohortMonth
2010-12-01   37.3  50.0  27.4
2011-01-01   36.8  15.0   NaN
2011-02-01    9.2   NaN   NaN
2011-03-01    NaN   NaN   NaN
2011-04-01    NaN   NaN   NaN
2011-05-01    NaN   NaN   NaN
2011-06-01    NaN   NaN   NaN
2011-07-01    NaN   NaN   NaN
2011-08-01    NaN   NaN   NaN
2011-09-01    NaN   NaN   NaN
2011-10-01    NaN   NaN   NaN
2011-11-01    NaN   NaN   NaN
2011-12-01    NaN   NaN   NaN
```

1.2.3 Other metrics

[98]:
```
grouping = online.groupby(['CohortMonth', 'CohortIndex'])
cohort_data = grouping['Quantity'].mean()
cohort_data = cohort_data.reset_index()
average_quantity = cohort_data.pivot(index='CohortMonth',
columns='CohortIndex',
values='Quantity')
```

Average quantity for each cohort

[99]:
```
average_quantity.round(1)
```

[99]:
```
CohortIndex  1.0   2.0   3.0   4.0   5.0   6.0   7.0   8.0   9.0   10.0  11.0  \
CohortMonth
2010-12-01   11.0  14.6  15.0  14.8  12.9  14.3  15.2  14.8  16.7  16.7  17.3
2011-01-01   10.0  12.6  12.3  10.9  12.2  14.9  14.2  14.4  11.4   9.9   9.1
2011-02-01   10.8  12.1  18.6  12.0  11.1  11.4  13.3  12.4  10.3  11.9  12.6
2011-03-01    9.8   9.9  12.2   9.5  13.6  12.3  13.2  12.2  10.5   8.9   NaN
2011-04-01    9.8  10.1   9.4  11.6  11.5   8.2   9.7   9.3   7.3   NaN   NaN
```

```
2011-05-01   10.9   9.0  13.9  11.8  10.9   8.7  10.1   7.4   NaN   NaN   NaN
2011-06-01   10.3  13.7  10.5  13.3  10.2   9.8   9.3   NaN   NaN   NaN   NaN
2011-07-01    9.7  12.7   7.1   7.8   6.0   7.0   NaN   NaN   NaN   NaN   NaN
2011-08-01    9.9   6.0   5.3   6.0   7.0   NaN   NaN   NaN   NaN   NaN   NaN
2011-09-01   11.9   5.5   7.6   8.8   NaN   NaN   NaN   NaN   NaN   NaN   NaN
2011-10-01    8.4   6.9   8.0   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN
2011-11-01    8.7   9.3   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN
2011-12-01   14.5   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN   NaN
```

```
CohortIndex  12.0  13.0
CohortMonth
2010-12-01   12.8  14.8
2011-01-01    9.5   NaN
2011-02-01    NaN   NaN
2011-03-01    NaN   NaN
2011-04-01    NaN   NaN
2011-05-01    NaN   NaN
2011-06-01    NaN   NaN
2011-07-01    NaN   NaN
2011-08-01    NaN   NaN
2011-09-01    NaN   NaN
2011-10-01    NaN   NaN
2011-11-01    NaN   NaN
2011-12-01    NaN   NaN
```

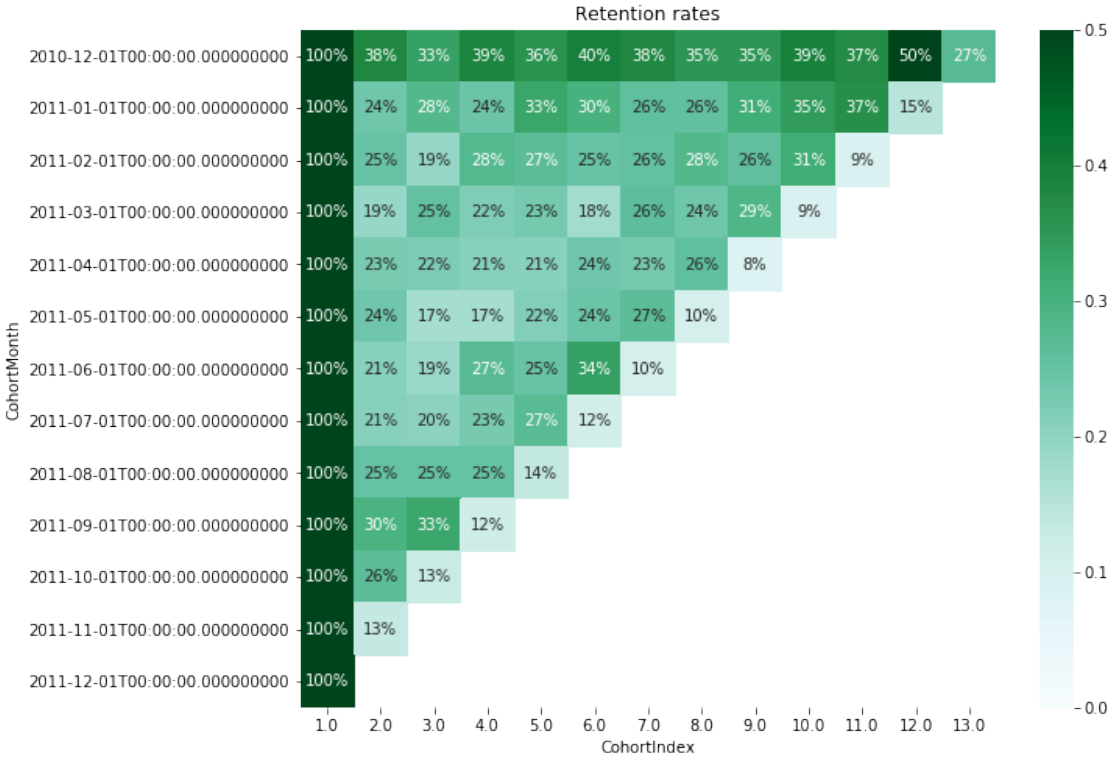
1.2.4 Cohort analysis visualization

- Build the heatmap

[100]:
```
import seaborn as sns
import matplotlib.pyplot as plt

plt.figure(figsize=(10, 8))
plt.title('Retention rates')
sns.heatmap(data = retention,
annot = True,
fmt = '.0%',
vmin = 0.0,
vmax = 0.5,
cmap = 'BuGn')

plt.show()
```


Retention rates
2010-12-01T00:00:00.000000000 100% 38% 33% 39% 36% 40% 38% 35% 35% 39% 37% 50% 27%
2011-01-01T00:00:00.000000000 100% 24% 28% 24% 33% 30% 26% 26% 31% 35% 37% 15%
2011-02-01T00:00:00.000000000 100% 25% 19% 28% 27% 25% 26% 28% 26% 31% 9%
2011-03-01T00:00:00.000000000 100% 19% 25% 22% 23% 18% 26% 24% 29% 9%
2011-04-01T00:00:00.000000000 100% 23% 22% 21% 21% 24% 23% 26% 8%
2011-05-01T00:00:00.000000000 100% 24% 17% 17% 22% 24% 27% 10%
2011-06-01T00:00:00.000000000 100% 21% 19% 27% 25% 34% 10%
2011-07-01T00:00:00.000000000 100% 21% 20% 23% 27% 12%
2011-08-01T00:00:00.000000000 100% 25% 25% 25% 14%
2011-09-01T00:00:00.000000000 100% 30% 33% 12%
2011-10-01T00:00:00.000000000 100% 26% 13%
2011-11-01T00:00:00.000000000 100% 13%
2011-12-01T00:00:00.000000000 100%
CohortMonth
1.0 2.0 3.0 4.0 5.0 6.0 7.0 8.0 9.0 10.0 11.0 12.0 13.0
CohortIndex
0.5 0.4 0.3 0.2 0.1 0.0

[ ]: