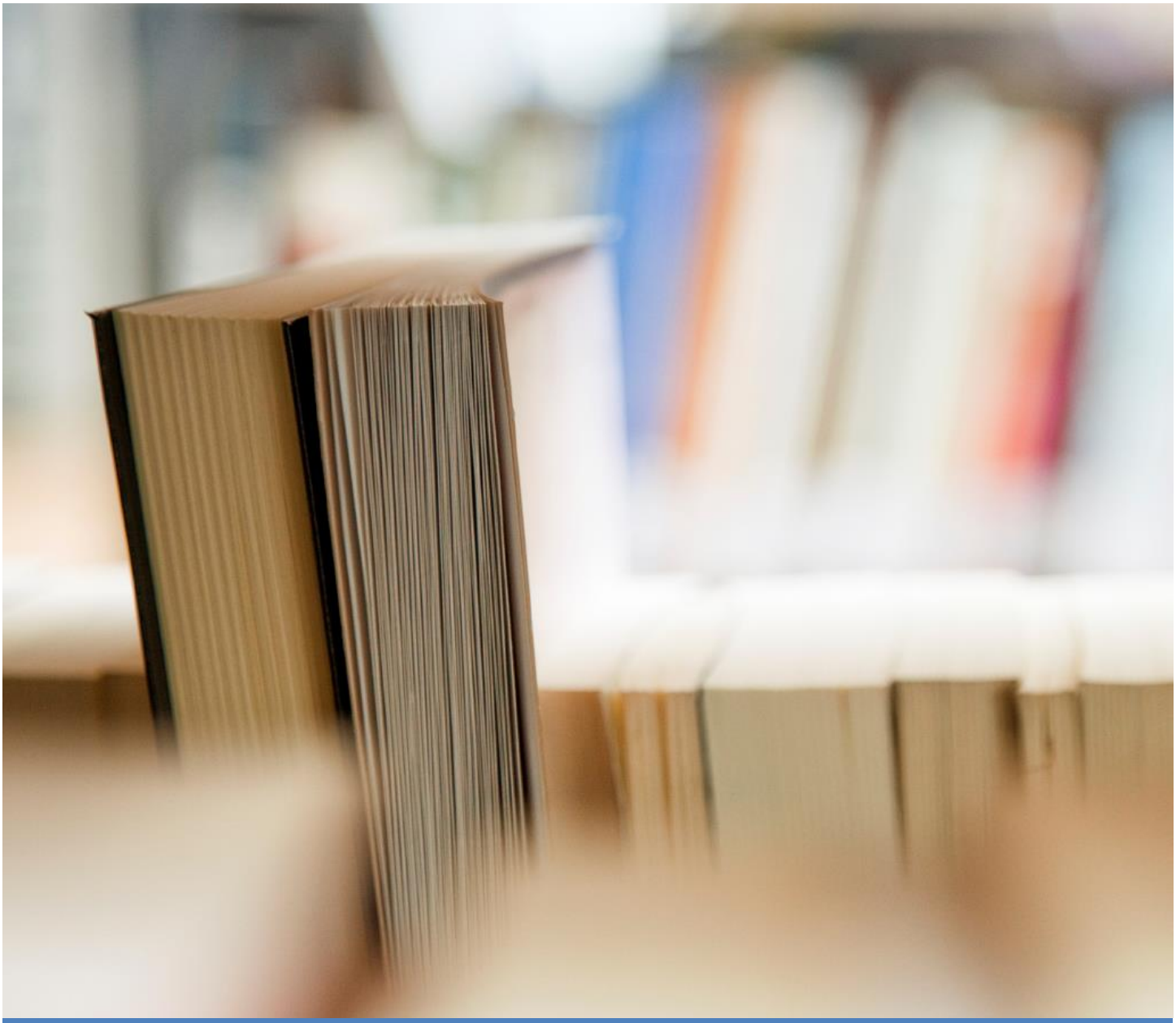


Data mining

Customer segmentation and claim prediction: Project report

Aditya Rishi

6/27/21

Great Learning

Machine Learning as problem solver for the businesses

I N D E X

Tables

Charts

Problem 1: Clustering

Executive summary

A leading bank wants to develop a customer segmentation to give promotional offers to its customers. They collected a sample that summarizes the activities of users during the past few months. You are given the task to identify the segments based on credit card usage.

Introduction

A market segmentation strategy, a way of pushing products and services, is important to banks and financial institutions, and is of two fundamental kinds: They can try to sell to everyone (mass-marketing) or focus their efforts on target segments (segmentation-marketing).

In most cases, mass-marketing is expensive and ineffective. For this reason, a market segmentation study should be a key component in any company's overall strategic plan.

Typically, banks tend to acquire new customers at huge costs rather than leveraging their existing customer base. A bank's customers leave behind a large footprint in terms of the transactions they perform, which can be analyzed to understand their behavior pattern which may be leveraged for selling new products. Segmentation models based on elementary classification schemes use 'spending' as the main demographic variable to cluster or segment their customer base.

Consumer banking preferences and behaviors continue to change based on technological advances, demographics and social factors. Retail bank executives seek to drive revenue generation by growing share of wallet from existing customer relationships. While customer acquisition is still important, greater emphasis is being placed on targeting current customer segments more accurately. The goal is to identify and capitalize on opportunities for revenue growth through cross selling and up selling. Meeting this priority requires a deep understanding of what customers need, want, expect and value in a financial institution.

A short story about segmentation

A client bank engaged BSG consultants to assess its private bank, which was underperforming. BSG worked closely with the bank's marketing group to analyze its customer base against criteria for wealth management and private banking segments. The analysis determined that the bank was over-penetrated in the mass affluent segment, but did not have a scheme in place to properly serve this segment. While the bank had a well-developed overall segmentation scheme, it was not designed to be used for managing the credit card business. BSG's wealth management experts assisted in developing and implementing a more effective mass market service platform.

Card trick: For the customers

Once you consent your willingness to be part of customer categorization, you will always be part of one or the other customer category depending upon your relationship value to be assessed on a half yearly basis. You will be upgraded or downgraded to other categories automatically based on whether your relationship value with the bank increases or decreases. It will be assessed on a yearly basis. You will earn benefits based on your eligible category during any period. You will be communicated through SMS, e-mail or registered number when these are upgraded or downgraded.

About clustering methods: K-means, a mean formula

Hierarchical clustering and k-means method can help customer segmentation plans a great deal. The k-means algorithm is an algorithm used commonly for clustering points. While this algorithm works quite well in practice, there are two aspects of this algorithm that are hard to grasp theoretically. First, it has been hard to prove any meaningful upper bound on the running time of this algorithm. In practice, k-means takes a sublinear number of iterations to converge for real datasets.

Smoothed analysis has given polynomial-time bounds to this problem, but even these bounds are much higher than what has been observed empirically. Secondly, it has been hard to quantify the accuracy of the solution that k-means converges to. Although k-means has guaranteed convergence because each step of the algorithm performs coordinate descent on the k-means objective, the algorithm rarely converges to the exact optimal solution because it mostly gets stuck at local minima. We were interested to see if the performance of k-means correlated with notions of stability we had discussed in class. Against our predictions, k-means did well through all the data. As an experiment beyond the scope of this project, we even pitted it against density-based DBSCAN and other clustering algorithms, and the little k-means 3 model beat them all. We'll mention that in brief, later. As far as time efficiency goes, k-means was far superior, even in the worst case. Since our dataset for it was not so large, even if k-means was run 25 different times, it would still be superior in time efficiency.

Data dictionary for market segmentation:

1. spending: Amount spent by the customer per month (in 1000s)

2. advance_payments: Amount paid by the customer in advance by cash (in 100s)

3. probability_of_full_payment: Probability of payment done in full by the customer to the bank

4. current_balance: Balance amount left in the account to make purchases (in 1000s)

5. credit_limit: Limit of the amount in credit card (10000s)

6. min_payment_amt : minimum paid by the customer while making payments for purchases made monthly (in 100s)

7. max_spent_in_single_shopping: Maximum amount spent in one purchase (in 1000s)

1.1 Read the data and do exploratory data analysis. Describe the data briefly.

spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
19.94	16.92	0.8752	6.675	3.763	3.252	6.550
15.99	14.89	0.9064	5.363	3.582	3.336	5.144
18.95	16.42	0.8829	6.248	3.755	3.368	6.148
10.83	12.96	0.8099	5.278	2.641	5.182	5.185
17.99	15.86	0.8992	5.890	3.694	2.068	5.837

Tab 1: The bank's credit card marketing dataset

The dataset loaded correctly and has 210 rows and 7 columns or variables and these seem to be all numeric.

	count	mean	std	min	25%	50%	75%	max
spending	210.0	14.847524	2.909699	10.5900	12.27000	14.35500	17.305000	21.1800
advance_payments	210.0	14.559286	1.305959	12.4100	13.45000	14.32000	15.715000	17.2500
probability_of_full_payment	210.0	0.870999	0.023629	0.8081	0.85690	0.87345	0.887775	0.9183
current_balance	210.0	5.628533	0.443063	4.8990	5.26225	5.52350	5.979750	6.6750
credit_limit	210.0	3.258605	0.377714	2.6300	2.94400	3.23700	3.561750	4.0330
min_payment_amt	210.0	3.700201	1.503557	0.7651	2.56150	3.59900	4.768750	8.4560
max_spent_in_single_shopping	210.0	5.408071	0.491480	4.5190	5.04500	5.22300	5.877000	6.5500

Tab 2: Data summary

There's not a major difference #between the 75th percentile and the maximum value for all the variables. Nevertheless, we will check for outliers by making box plots.

 # Are there any missing values?

```
spending                      0
advance_payments              0
probability_of_full_payment   0
current_balance               0
credit_limit                  0
min_payment_amt               0
max_spent_in_single_shopping  0
dtype: int64
```

There are no missing values

Data types

```
RangeIndex: 210 entries, 0 to 209
Data columns (total 7 columns):
 #   Column                        Non-Null Count  Dtype
---  ------                        --------------  -----
 0   spending                      210 non-null    float64
 1   advance_payments              210 non-null    float64
 2   probability_of_full_payment   210 non-null    float64
 3   current_balance               210 non-null    float64
 4   credit_limit                  210 non-null    float64
 5   min_payment_amt               210 non-null    float64
 6   max_spent_in_single_shopping  210 non-null    float64
dtypes: float64(7)
memory usage: 11.6 KB
```

Tab 3: Data info

All the seven variables are numeric, of data type float. This meets the basic requirement for the clustering algorithms that we'd like use for customer segmentation.

Are there any duplicates?

Number of duplicate rows = 0

There are no duplicates or missing values.

Checking for outliers

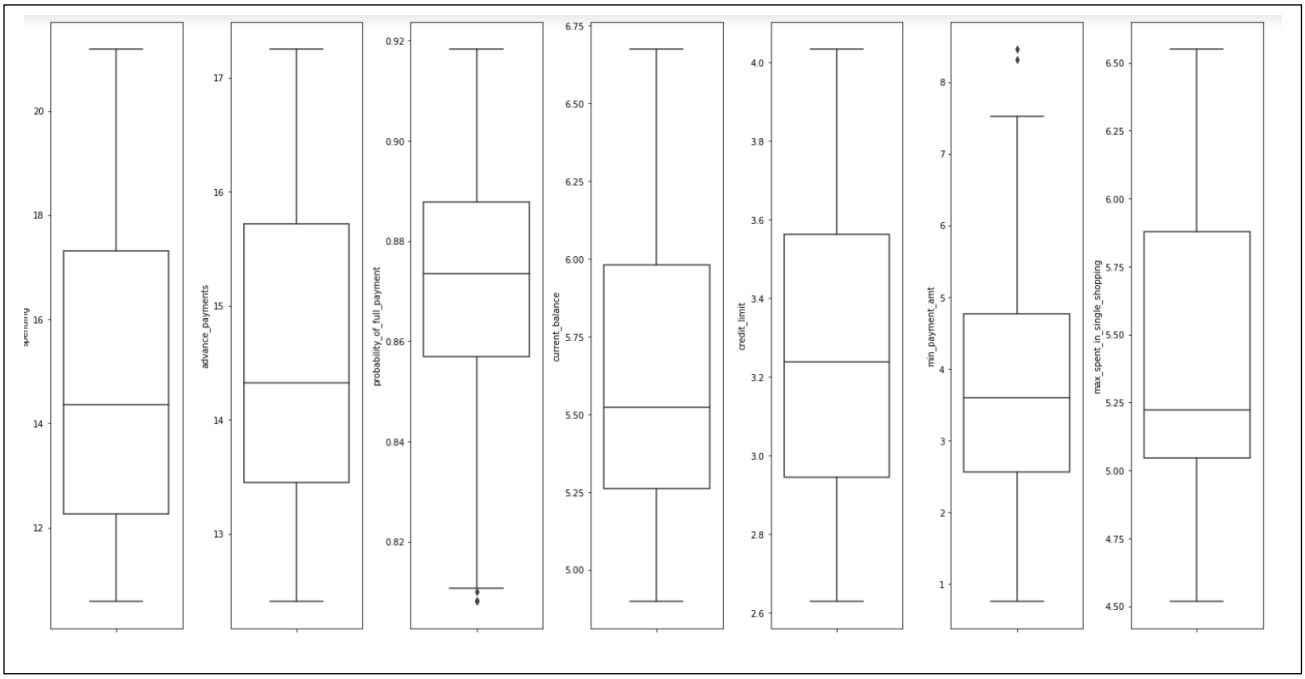


Fig 1: Boxplots for all variables

Second look after outlier treatment.

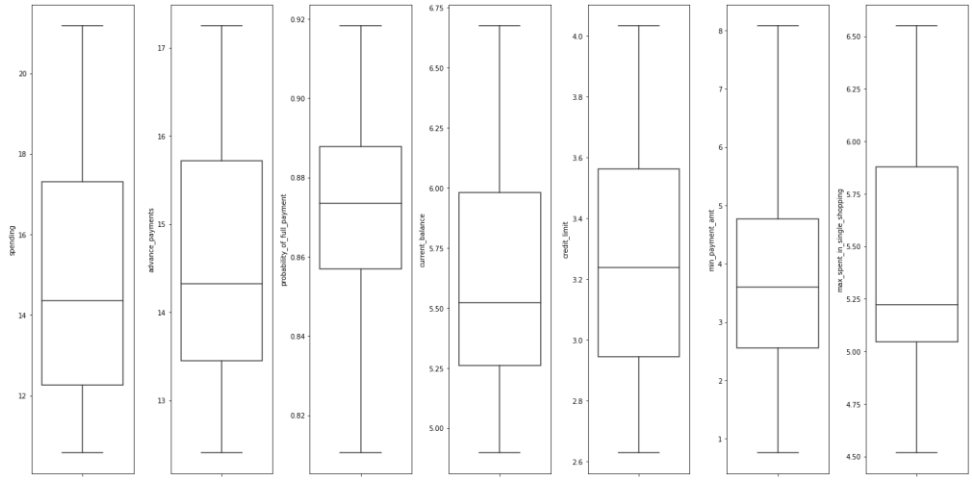


Fig 2: Boxplots after outlier removal

All outliers removed. These were in the lower range of the variable 'probability_of_full_payment' and the upper range of 'min_payment_amt'.

The outliers are treated with the minimum and maximum value of inter-quartile range or IQR.
The outliers beyond the third quartile or Q3 are replaced with the upper limit.
The outliers below the first quartile or Q1 are replaced with the lower limit.

For clustering, only the numeric columns are used. The outliers are points that do not meet distance and minimum samples requirements to be recognised as a cluster The dataset has a very small number of outliers, which we can replace with the upper or the lower limit.

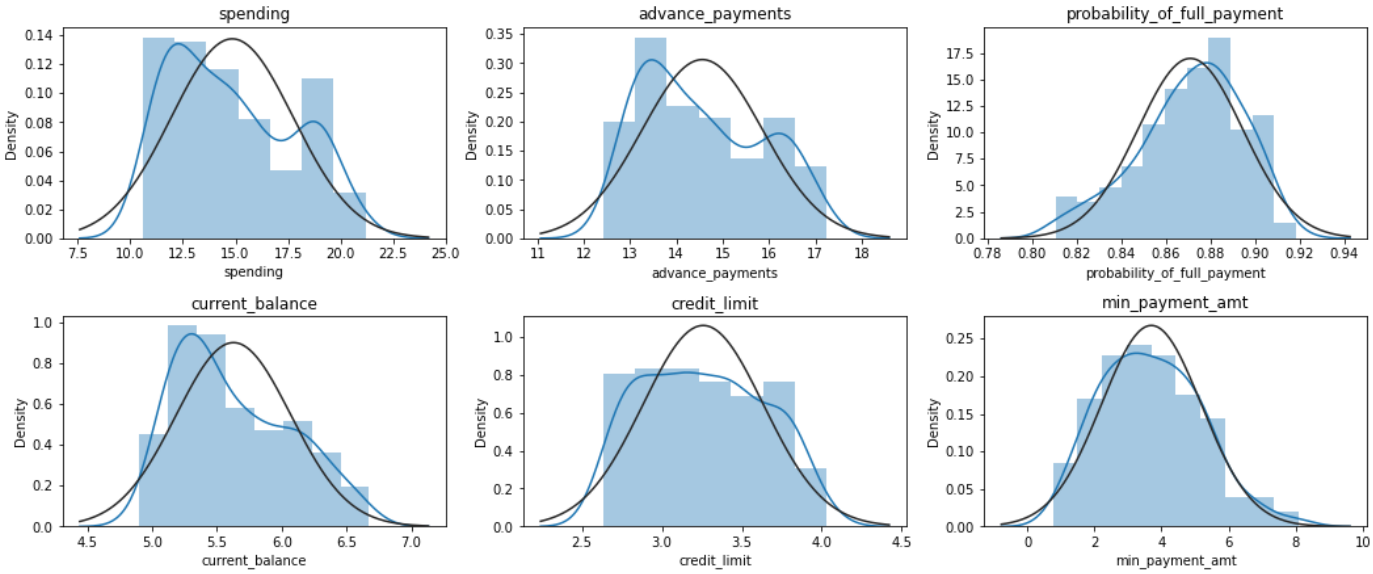


Fig 3: Distplots for all variables, showing also the Kernel Density Estimate line

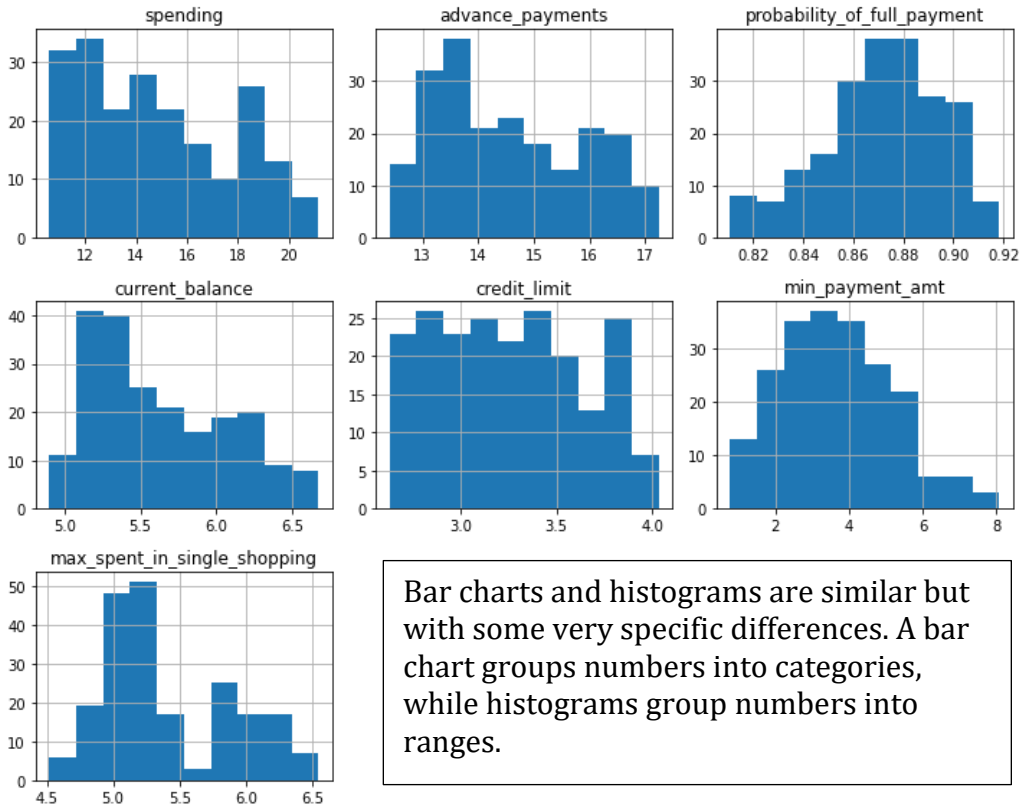
None of the variables has perfect normal distribution. Most of the variabels are right skewed, while 'probability_of_full_payment' is left-skewed. In a left skewed distribution, the mean is less than the median. In a right skewed distribution, the mean is greater than the median. In a symmetrical or normal distribution, the mean, median, and mode are all equal. Symmetrical distributions are indicated by the black curve for reference, while the actual distribution curves are in blue. A distribution is left skewed if it has a "tail" on the left side of the distribution. A distribution is right skewed if it has a "tail" on the right side of the distribution.

Further numeric EDA

#Frequency variables in charts.

Fig 4: Frequency of variables

We start by looking at the horizontal or x-axis of these histograms to see how the data in each variable is grouped. Then, we look at the vertical or y-axis to see how frequently that data occurs.



Bar charts and histograms are similar but with some very specific differences. A bar chart groups numbers into categories, while histograms group numbers into ranges.

Histogram analysis

Histograms are used to show the results of continuous data, generally. The spending range of up to 12k (rupees/dollars, or anything) has the highest number of customers (32+34=66). The highest number of customers for any advance payment range is 38 for between 13k and 14k. There's nearabout 88% probability of full payment for 78 customers (39+39), which is the highest. 82 customers have a current balance between 5.1k and 5.4k. The credit limit is most uniform across ranges, except when it touches 40k. Understandable, as higher credit limit is only for a few. 37 customers (the most for any range) have minimum payment amount around 3. 100 customers (the most for any range) have spent between 5 and 5.3 on their biggest single shopping trips. Maybe they charged higher-value products on credit card for which cash payment isn't the preferred mode of transaction.

A look at spending

Spending density is the maximum for 10 to 15 cost range and is now. The wave picks up around 7 and dies out after touching the shores of 23. A kernel density estimate (KDE) plot is a method for visualizing the distribution of observations in a dataset. This is analogous to a histogram. KDE represents the data using a continuous probability density curve in one or more dimensions. It depicts the probability density at different values in a

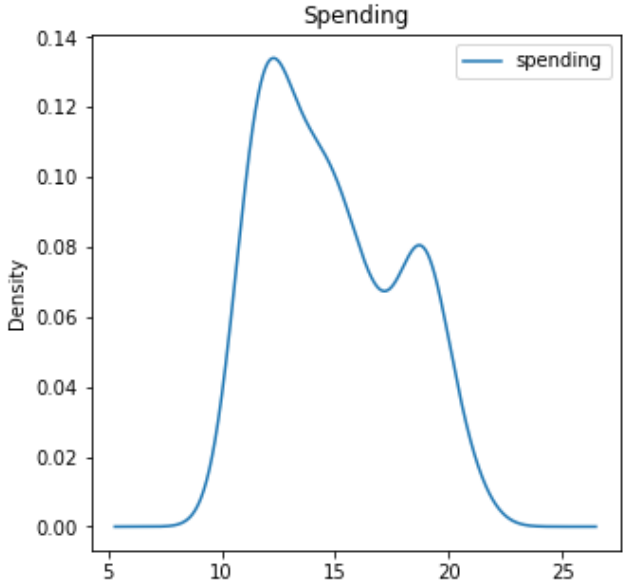


Fig 5: KDE plot for spending

continuous variable. The density curve for spending in this case has two distinct peaks, indicating that the distribution is bimodal.

Multivariate density plot.

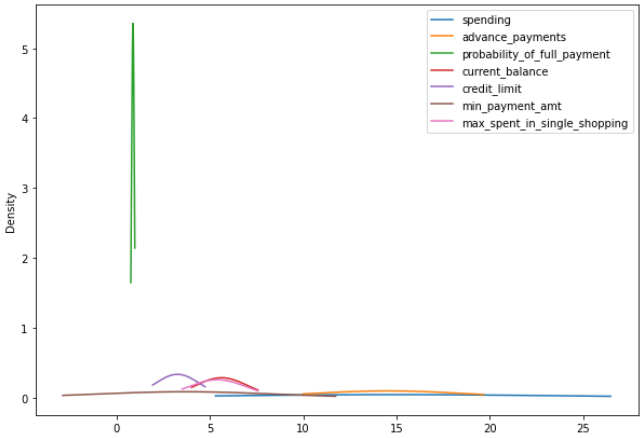


Fig 6: Multivariate density plot

The KDE algorithm takes a parameter, bandwidth, that affects how “smooth” the resulting curve is.

The KDE is calculated by weighting the distances of all the data points we've seen for each location on the blue line. If we've seen more points nearby, the estimate is higher, indicating that probability of seeing a point at that location. The density curve for probability of full payment has very low bandwidth and very high amplitude (almost 5 or 6 times). This means the probability of seeing a data point at that location is very slim. Not many people will ever make full payment. True, as there is never 100% surety of it.

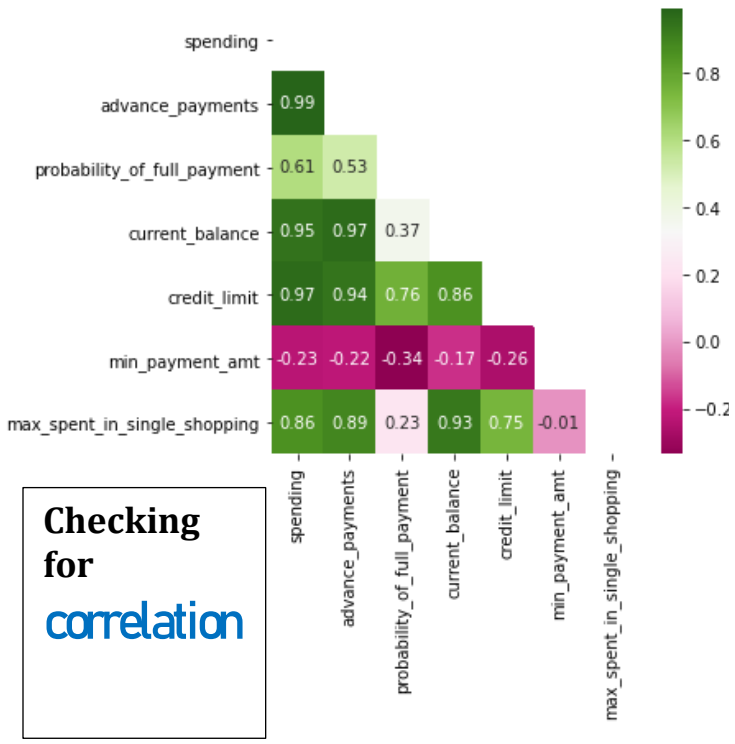
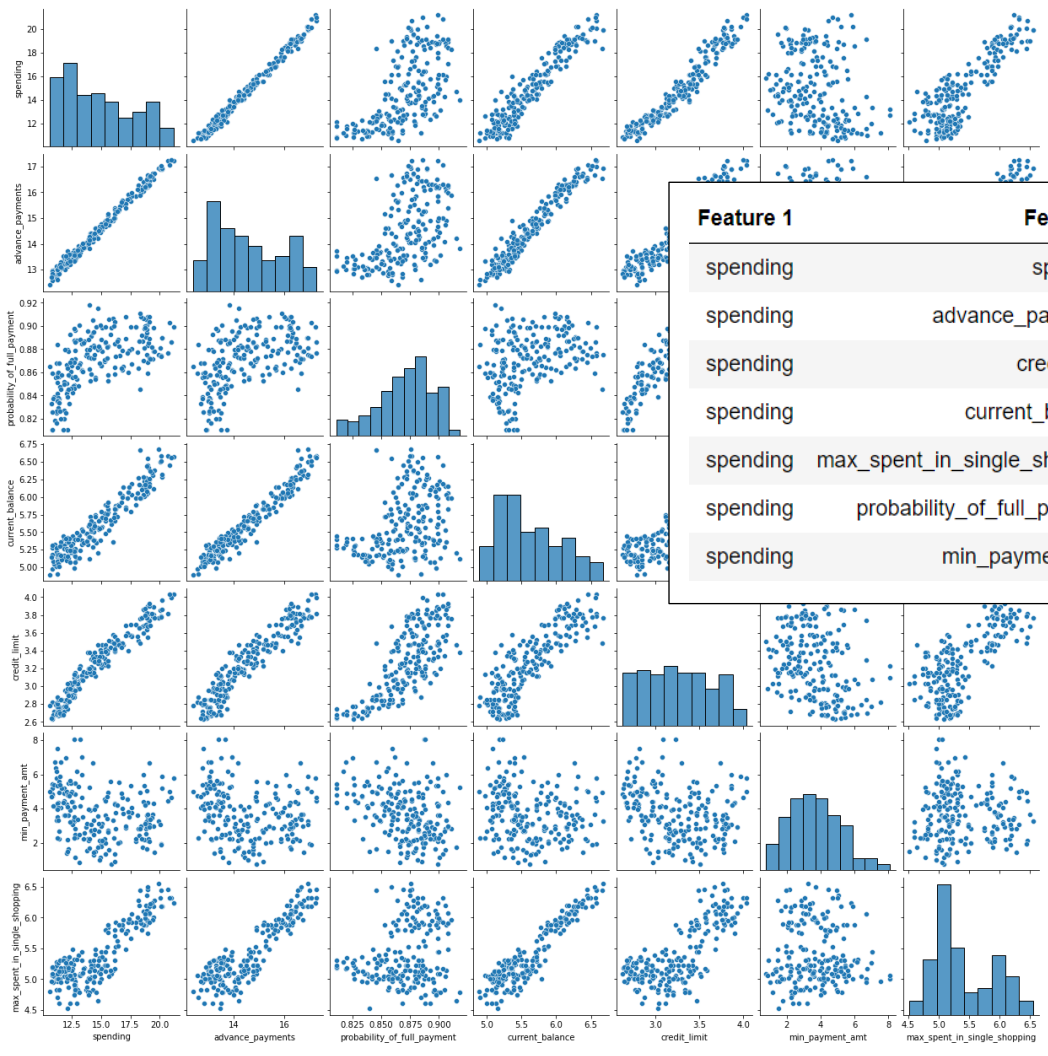


Fig7: Heatmap to check correlations

Advance payments have a high correlation with spending, credit balance and credit limit. Maximum spending in single shopping has a high correlation with current balance. Minimum payment amount has a negative correlation with spending, probability of full payment, current balance, credit limit and advance payments (Notice the purple strip).



Pairwise relationships.

Fig 8: Pairplot

Feature 1	Feature 2	Correlation Coefficient
spending	spending	1.000000
spending	advance_payments	0.925334
spending	credit_limit	0.856237
spending	current_balance	0.777859
spending	max_spent_in_single_shopping	0.575000
spending	probability_of_full_payment	0.450314
spending	min_payment_amt	-0.181093

There are quite a few instances of strong, linear correlation between variables, such as between spending and advance payments; advance payment and current balance; spending and credit limit; current balance and maximum spent in single shopping etc.

Hued pairplots

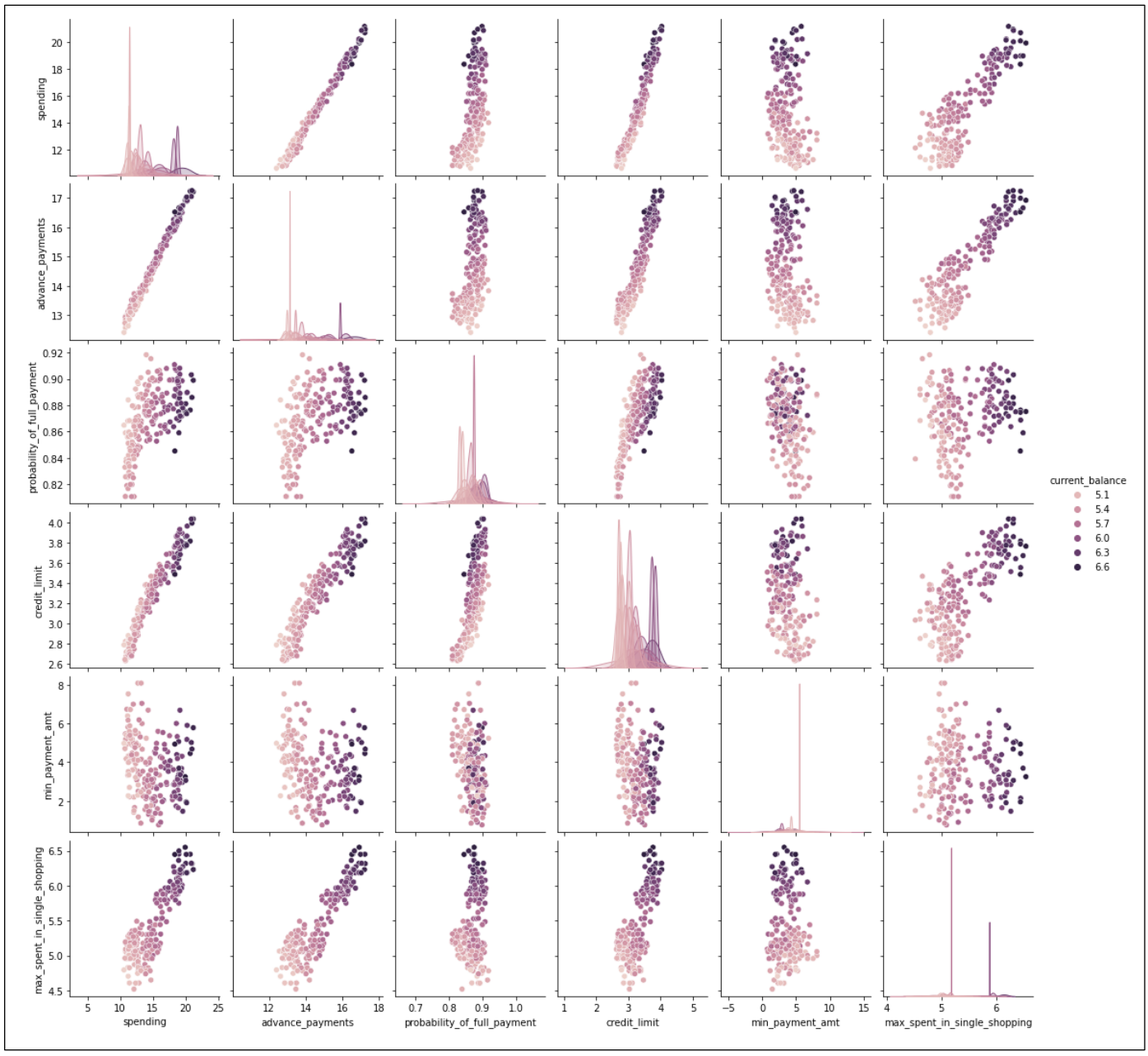


Fig 9: Hued pairplot for current balance

Current balance has six distinct groups. See legend

Fig 10: Hued pairplot for spending

Spending has five distinct groups. See legend.

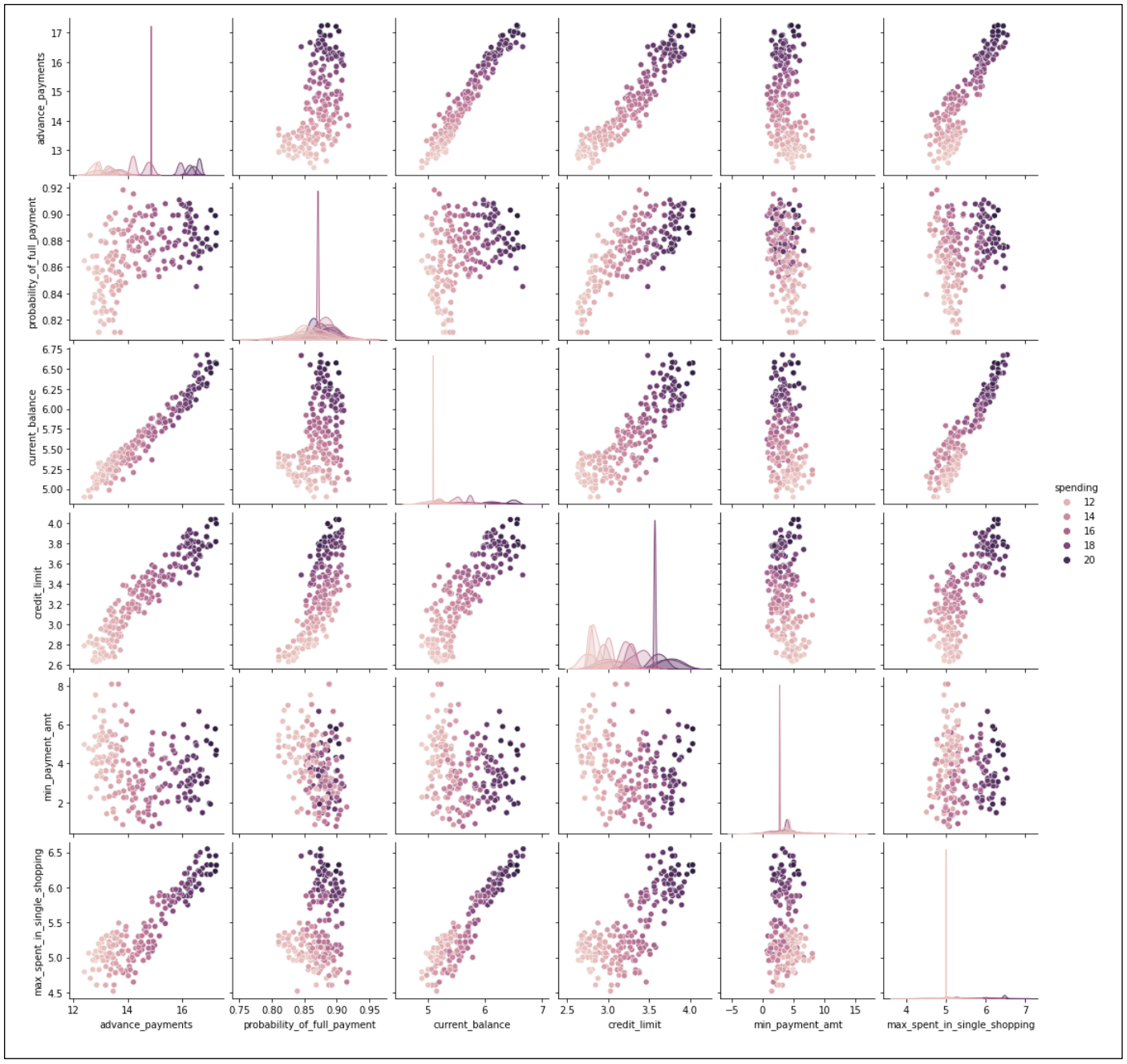


Fig 11: Hued pairplot for maximum spend in single shopping

Maximum spend in single shopping has five distinct groups. See legend.

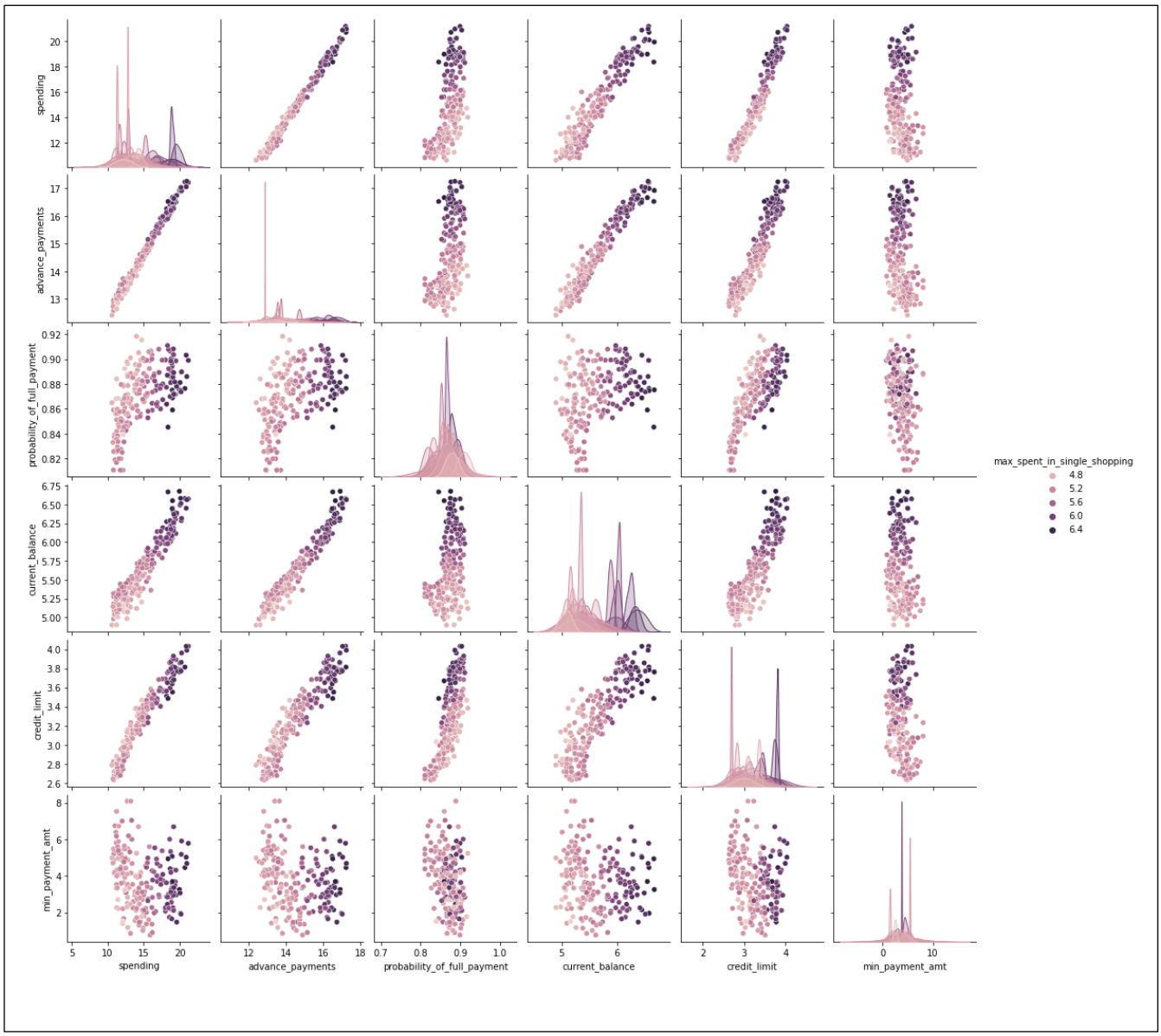


Fig 12: Advance payment. 5 groups

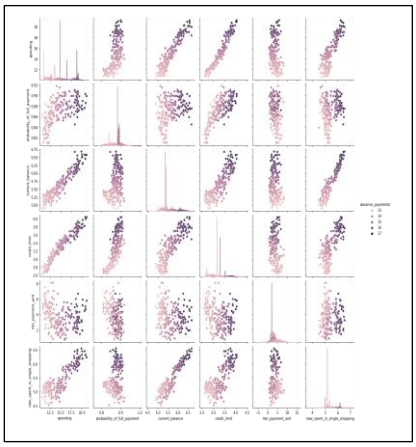


Fig 13: Credit limit 6 groups

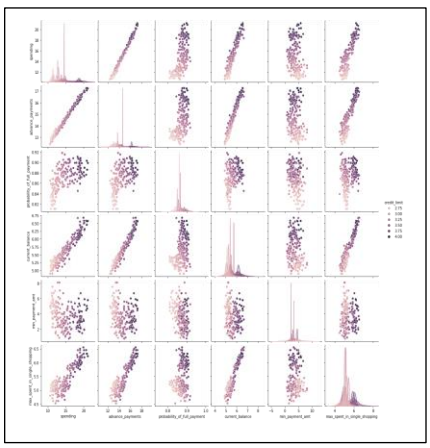
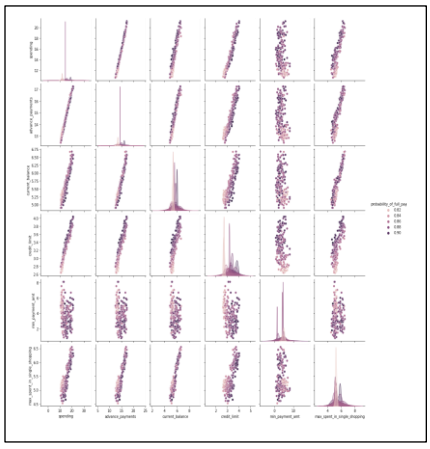


Fig 13.1: Probability of full payment. 5 groups



Variable description

```
Description of spending
----------------------------------------
count    210.000000
mean      14.847524
std        2.909699
min       10.590000
25%       12.270000
50%       14.355000
75%       17.305000
max       21.180000
Name: spending, dtype: float64
```

```
Description of advance_payments
----------------------------------------
----------------------------------------
count    210.000000
mean      14.559286
std        1.305959
min       12.410000
25%       13.450000
50%       14.320000
75%       15.715000
max       17.250000
Name: advance_payments, dtype: float64

Description of
probability_of_full_payment
----------------------------------------
----------------------------------------
count    210.000000
mean       0.871025
std        0.023560
min        0.810588
25%        0.856900
50%        0.873450
75%        0.887775
max        0.918300
Name: probability_of_full_payment,
dtype: float64
```

Tab 4: Five-number summaries for each variable

```
Description of current_balance
--------------------------------
count    210.000000
mean       5.628533
std        0.443063
min        4.899000
25%        5.262250
50%        5.523500
75%        5.979750
max        6.675000
Name: current_balance,
dtype: float64

Description of credit_limit
--------------------------------
count    210.000000
mean       3.258605
std        0.377714
min        2.630000
25%        2.944000
50%        3.237000
75%        3.561750
max        4.033000
Name: credit_limit,
dtype: float64

Description of min_payment_amt
--------------------------------
count    210.000000
mean       3.697288
std        1.494689
min        0.765100
25%        2.561500
50%        3.599000
75%        4.768750
max        8.079625
Name: min_payment_amt,
dtype: float64

Description of
max_spent_in_single_shopping
--------------------------------
count    210.000000
mean       5.408071
std        0.491480
min        4.519000
25%        5.045000
50%        5.223000
75%        5.877000
max        6.550000
Name:
max_spent_in_single_shopping,
dtype: float64
```

Checking columns

Index(['spending', 'advance_payments', 'probability_of_full_payment', 'current_balance', 'credit_limit', 'min_payment_amt', 'max_spent_in_single_shopping'], dtype='object')

spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
1.754355	1.811968	0.177628	2.367533	1.338579	-0.298625	2.328998
0.393582	0.253840	1.505071	-0.600744	0.858236	-0.242292	-0.538582
1.413300	1.428192	0.505234	1.401485	1.317348	-0.220832	1.509107
-1.384034	-1.227533	-2.571391	-0.793049	-1.639017	0.995699	-0.454961
1.082581	0.998364	1.198738	0.591544	1.155464	-1.092656	0.874813

Tab 5: Scaled dataset, as evidence below by checking the mean and the standard deviation

	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
count	2.100000e+02	2.100000e+02	2.100000e+02	2.100000e+02	2.100000e+02	2.100000e+02	2.100000e+02
mean	9.148766e-16	1.097006e-16	1.642601e-15	-1.089076e-16	-2.994298e-16	1.512018e-16	-1.935489e-15
std	1.002389e+00	1.002389e+00	1.002389e+00	1.002389e+00	1.002389e+00	1.002389e+00	1.002389e+00
min	-1.466714e+00	-1.649686e+00	-2.571391e+00	-1.650501e+00	-1.668209e+00	-1.966425e+00	-1.813288e+00
25%	-8.879552e-01	-8.514330e-01	-6.009681e-01	-8.286816e-01	-8.349072e-01	-7.616981e-01	-7.404953e-01
50%	-1.696741e-01	-1.836639e-01	1.031721e-01	-2.376280e-01	-5.733534e-02	-6.591519e-02	-3.774588e-01
75%	8.465989e-01	8.870693e-01	7.126469e-01	7.945947e-01	8.044956e-01	7.185591e-01	9.563941e-01
max	2.181534e+00	2.065260e+00	2.011371e+00	2.367533e+00	2.055112e+00	2.938945e+00	2.328998e+00

1.2 Do you think scaling is necessary for clustering in this case? Justify.

We do have to use scaling, since hierarichal clustering and Kmeans both use distance-based algorithms for computation. Moreover, the data also has a few variables on different scales. For example, probability of full payment is on a scale of 10^(-4), while spending is on a hundredth scale. Therefore, it become necessary to scale the data. We used z-scaling, because the variance between the columns is more or less the same.

Checking for outliers again

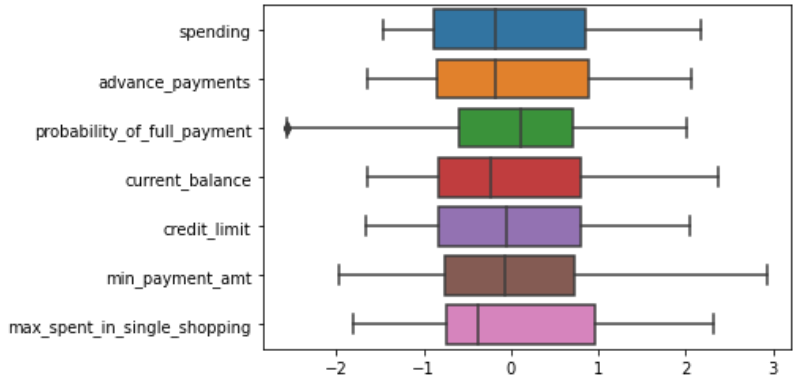


Fig 14: Boxplot of scaled dataset

Negligible outliers

1.3 Apply hierarchical clustering to scaled data. Identify the number of optimum clusters using Dendrogram and briefly describe them.

Hierarchical Clustering

Get the count of unique values for data

spending	193
advance_payments	170
probability_of_full_payment	184
current_balance	188
credit_limit	184
min_payment_amt	206
max_spent_in_single_shopping	148

dtype: int64

Creating a dendrogram

Choosing a clustering method

We make a silhouette score list in order to find an appropriate linkage method out of 'ward, 'average', and 'complete' for hierarchical clustering

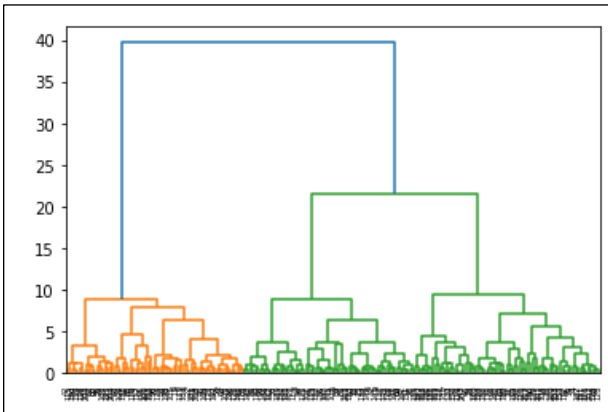


Fig 15: First dendrogram

	cluster	sil_score	linkage_method	number_of_clusters
0	2	0.461158	ward	2
1	2	0.441189	average	2
2	2	0.397924	complete	2
4	4	0.352612	average	4
3	4	0.301176	ward	4
7	6	0.293249	average	6
10	8	0.282421	average	8
13	10	0.253262	average	10
14	10	0.247572	complete	10
11	8	0.244273	complete	8
9	8	0.233347	ward	8
18	14	0.232082	ward	14
15	12	0.229234	ward	12
19	14	0.226325	average	14
17	12	0.223036	complete	12
12	10	0.222681	ward	10
24	18	0.220991	ward	18
8	6	0.215490	complete	6
21	16	0.215246	ward	16
6	6	0.215121	ward	6
5	4	0.207519	complete	4
22	16	0.205779	average	16
25	18	0.204940	average	18
16	12	0.197908	average	12
26	18	0.196368	complete	18
20	14	0.192355	complete	14
23	16	0.191458	complete	16

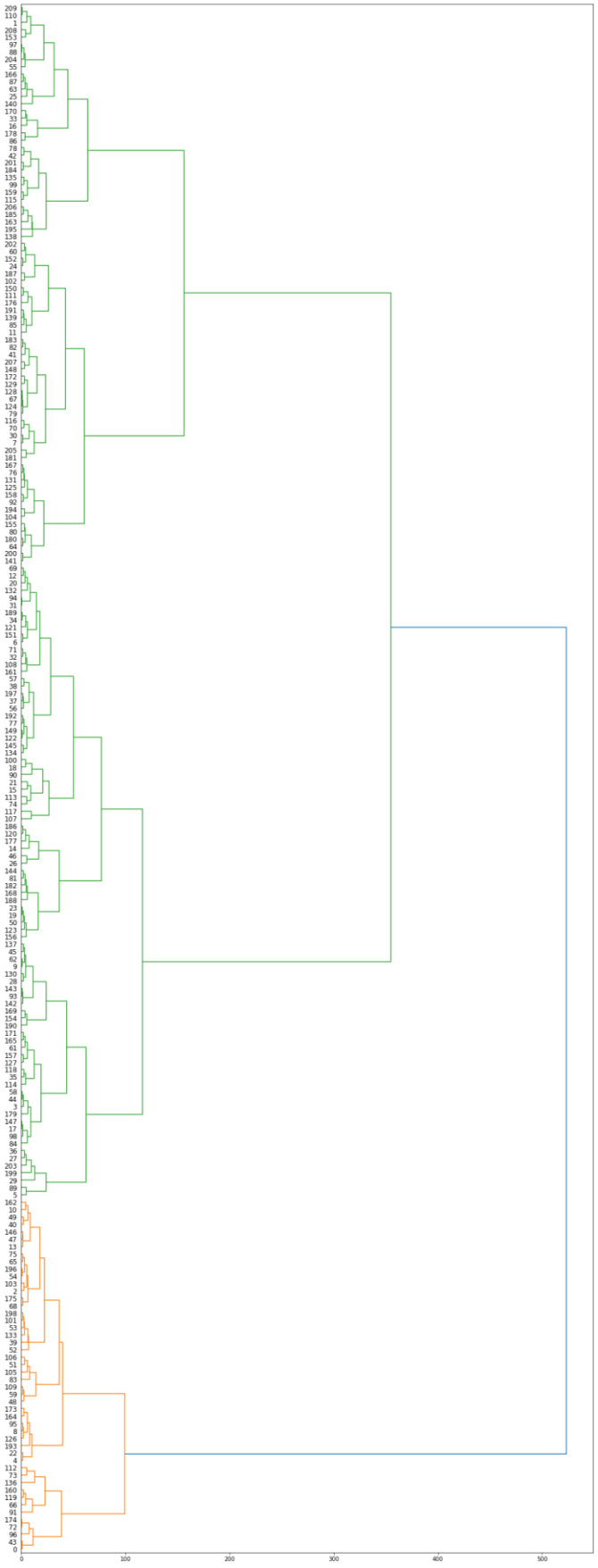
Tab 6: Silhouette score list

We choose the ward method on the basis of the highest silhouette score of 0.461158. Typically, mean silhouette over 0.6 is considered a "good" clustering solution but, for now, this 0.46 is our best option.

Checking columns

Index(['spending', 'advance_payments', 'probability_of_full_payment', 'current_balance', 'credit_limit', 'min_payment_amt', 'max_spent_in_single_shopping'], dtype='object')

Fig 16: Big dendrogram



Distance matrix

[[0. 4.84379711 1.26000488 ... 7.0105177 4.63461196 4.8895951]

[4.84379711 0. 3.59516985 ... 2.67724004 1.17640255 1.30574679]

[1.26000488 3.59516985 0. ... 5.858194 3.4773039 3.74469185]

...

[7.0105177 2.67724004 5.858194 ... 0. 2.39361362 2.28220521]

[4.63461196 1.17640255 3.4773039 ... 2.39361362 0. 0.87365342]

[4.8895951 1.30574679 3.74469185 ... 2.28220521 0.87365342 0.]]

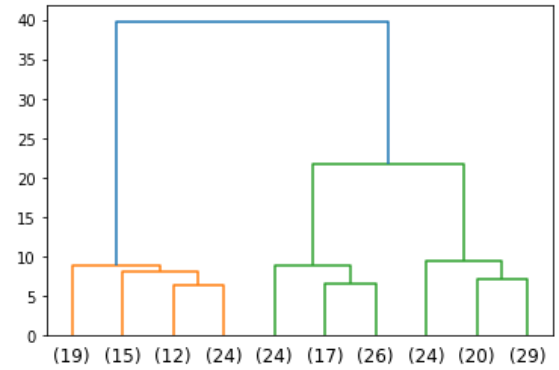


Fig 17: Truncated dendrogram

Where to cut this dendrogram? There is no one answer to this. The dendrogram suggests an optimal number (2) based on the colors that it assigns to clusters. At the same time, based on the distance criteria, we can form a different impression on what's optimal. Looking at different metrics and using the knowledge of the data and the clusters, we must make sure that the clusters are heterogeneous or diverse. That way we can optimize the same. If we lower the distance bar, it will cut three lines instead for two, and three might then be the optimal number of clusters. We'll do one thing: make both 2-cluster and 3-cluster models and then choose one that is less costly to the business and more heterogeneous.

Using fcluster module to create clusters

Set criterion as distance, then create 2 clusters, and then store the result in another object called 'clusters'.

Method= wardlink
Criterion=distance
Distance=25

Clusters

array([1, 2, 1, 2, 1, 2, 2, 2, 1, 2, 1, 2, 2, 1, 2, 2, 2, 2, 2, 2, 2,
1, 2, 2, 1, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 1, 1, 2, 1, 1,
2, 2, 2, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 2, 2, 1, 2, 2, 2, 2, 2, 1,
1, 2, 1, 2, 2, 2, 1, 1, 2, 1, 2, 2, 1, 2, 2, 2, 2, 1, 2, 2, 2, 1,
1, 2, 2, 1, 2, 2, 2, 1, 1, 1, 2, 1, 2, 1, 2, 1, 2, 1, 1, 2, 2, 1,
2, 2, 1, 2, 2, 1, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2,
2, 1, 2, 1, 1, 2, 1, 2, 2, 2, 2, 2, 2, 2, 1, 2, 2, 2, 2, 2, 2, 2,
2, 2, 2, 2, 2, 1, 1, 2, 1, 1, 1, 2, 1, 2, 2, 2, 2, 2, 2, 1, 1, 1,
2, 2, 1, 2, 2, 2, 2, 2, 1, 1, 2, 2, 2, 2, 2, 2, 2, 1, 2, 1, 1, 2,
1, 2, 2, 1, 2, 2, 1, 2, 1, 2, 1, 2], dtype=int32)

Cluster frequency

1 70
2 140

Cluster 1 has 70 records, while cluster 2 has 140 out of 210.

Appending clusters to original database

spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	clusters
19.94	16.92	0.875200	6.675	3.763	3.252	6.550	1
15.99	14.89	0.906400	5.363	3.582	3.336	5.144	2
18.95	16.42	0.882900	6.248	3.755	3.368	6.148	1
10.83	12.96	0.810588	5.278	2.641	5.182	5.185	2
17.99	15.86	0.899200	5.890	3.694	2.068	5.837	1

Tab 7: Data sets with clusters added

Cluster profiles

clusters	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	Freq
1	18.371429	16.145429	0.884400	6.158171	3.684629	3.639157	6.017371	70
2	13.085571	13.766214	0.864338	5.363714	3.045593	3.726353	5.103421	140

Tab 8: Data set pivoted on clusters and with frequency added

Cluster 1: High spenders with more advance payment and higher average maximum spend in single shopping

Cluster 2: Low spenders with less advance payment and lower average maximum spend in single shopping

Using Agglomerative Clustering

Affinity=Euclidean
Linkage=Ward

Clusters

[1 0 1 0 1 0 0 0 1 0 1 0 0 1 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 1 1 0 1 1 0 0 0 1 1 1 0 1 1 1 1 1 0 0 0 1 0 0 0 0 0 1 1 0 1 0 0 0 1 1 0 1 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 0 0 1 0 0 1 0 0 0 1 1 1 0 1 0 1 0 1 0 1 1 0 0 1 0 0 1 0 0 0 0 1 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 1 1 0 1 0 0 0 0 0 0 0 1 0 0 0 1 0 0 0 1 0 0 0 0 0 0 0 1 1 0 1 1 1 0 0 0 0 0 0 0 0 1 1 1 0 0 0 0 0 0 0 0 0 1 0 0 0 0 0 0 0 1 0 1 1 0 1 0 0 0 0 0 1 0 1 0 0 0]

Columns

Index(['spending',
advance_payments',
probability_of_full_payment',
'current_balance', 'credit_limit',
'min_payment_amt',
'max_spent_in_single_shopping',
'clusters', 'Agglo_CLusters'],
dtype='object')

Agglo_CLusters	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	Freq
0	13.232517	13.830612	0.865492	5.385932	3.065946	3.713520	5.128265	147
1	18.615873	16.259524	0.883937	6.194603	3.708143	3.659413	6.060952	63

Tab 9: Dataset pivoted on agglomerative clusters

	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
count	70.000000	70.000000	70.000000	70.000000	70.000000	70.000000	70.000000
mean	18.371429	16.145429	0.884400	6.158171	3.684629	3.639157	6.017371
std	1.381233	0.599277	0.014767	0.245926	0.174909	1.208271	0.251132
min	15.380000	14.860000	0.845200	5.709000	3.268000	1.472000	5.443000
25%	17.330000	15.737500	0.874700	5.979250	3.554250	2.845500	5.877000
50%	18.720000	16.210000	0.883950	6.148500	3.693500	3.629000	5.981500
75%	19.137500	16.557500	0.898225	6.312000	3.804750	4.459250	6.187750
max	21.180000	17.250000	0.910800	6.675000	4.033000	6.682000	6.550000

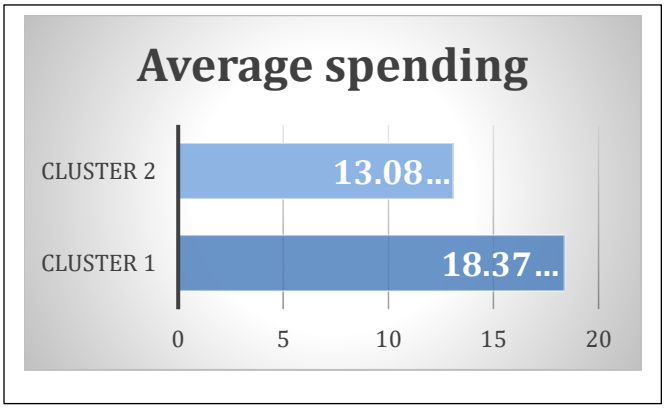
Tab 10: Data summary for cluster 1

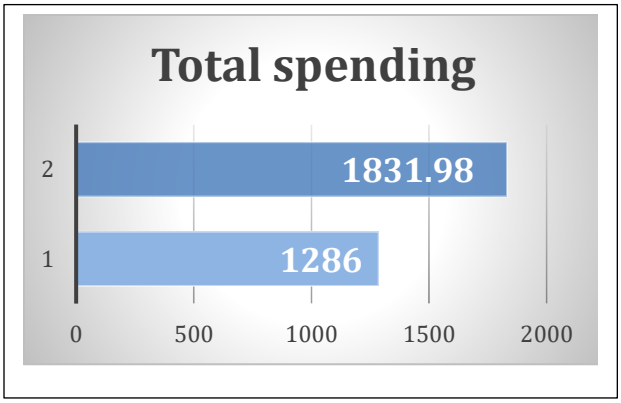
	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
count	140.000000	140.000000	140.000000	140.000000	140.000000	140.000000	140.000000
mean	13.085571	13.766214	0.864338	5.363714	3.045593	3.726353	5.103421
std	1.550003	0.696916	0.024315	0.230740	0.249454	1.622319	0.226834
min	10.590000	12.410000	0.810588	4.899000	2.630000	0.765100	4.519000
25%	11.817500	13.207500	0.848075	5.179000	2.835250	2.461750	5.000000
50%	12.770000	13.665000	0.865800	5.351000	3.037000	3.597500	5.091500
75%	14.347500	14.305000	0.882075	5.521750	3.234500	4.879250	5.222500
max	16.630000	15.460000	0.918300	6.053000	3.582000	8.079625	5.879000

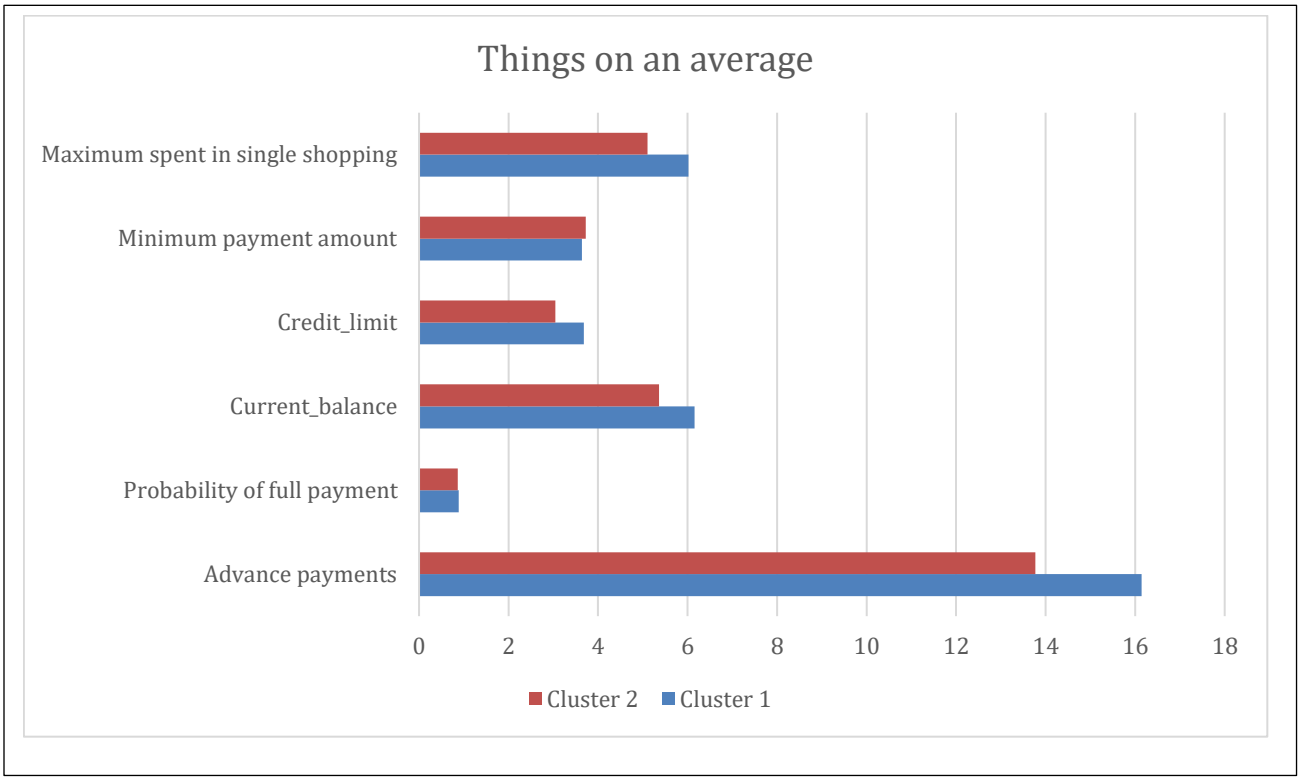
Tab 11: Data summary for cluster 2

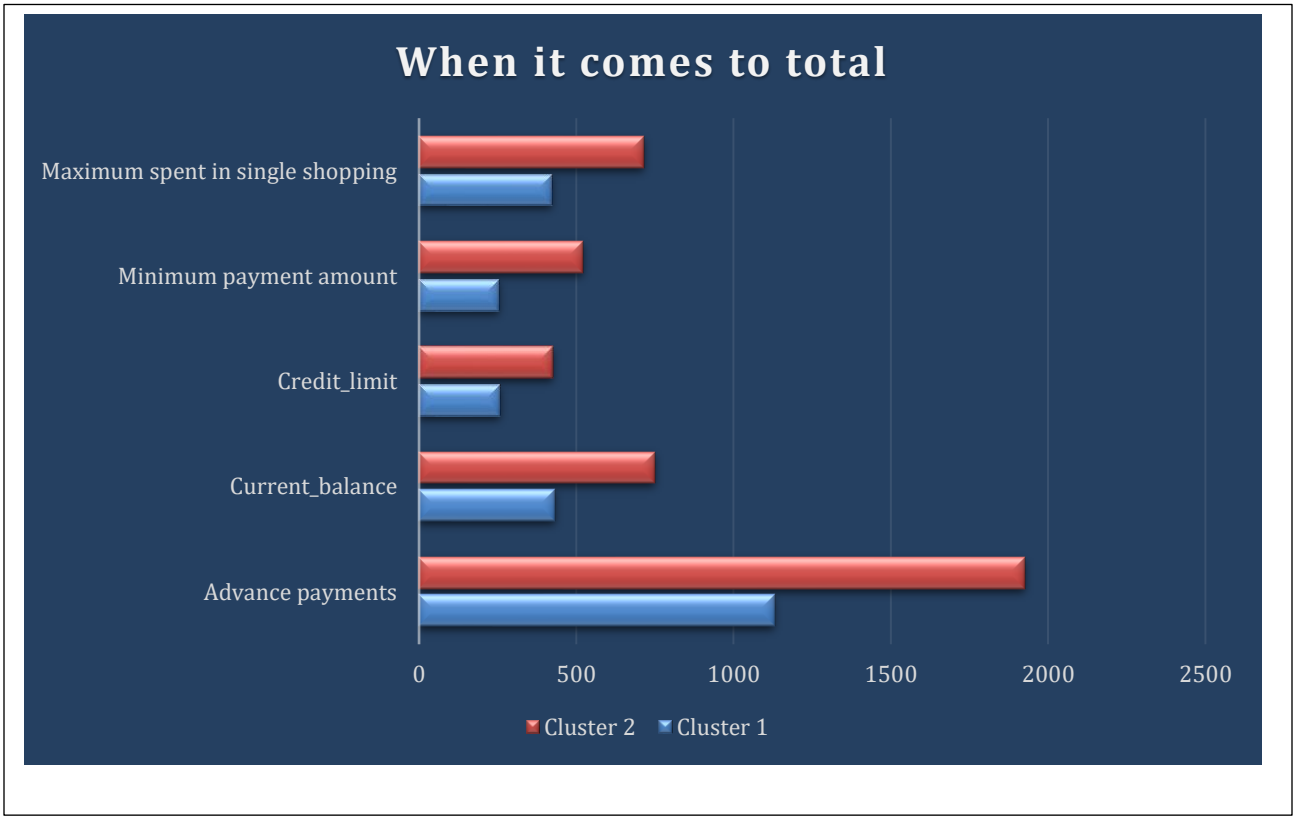
Cluster comparison

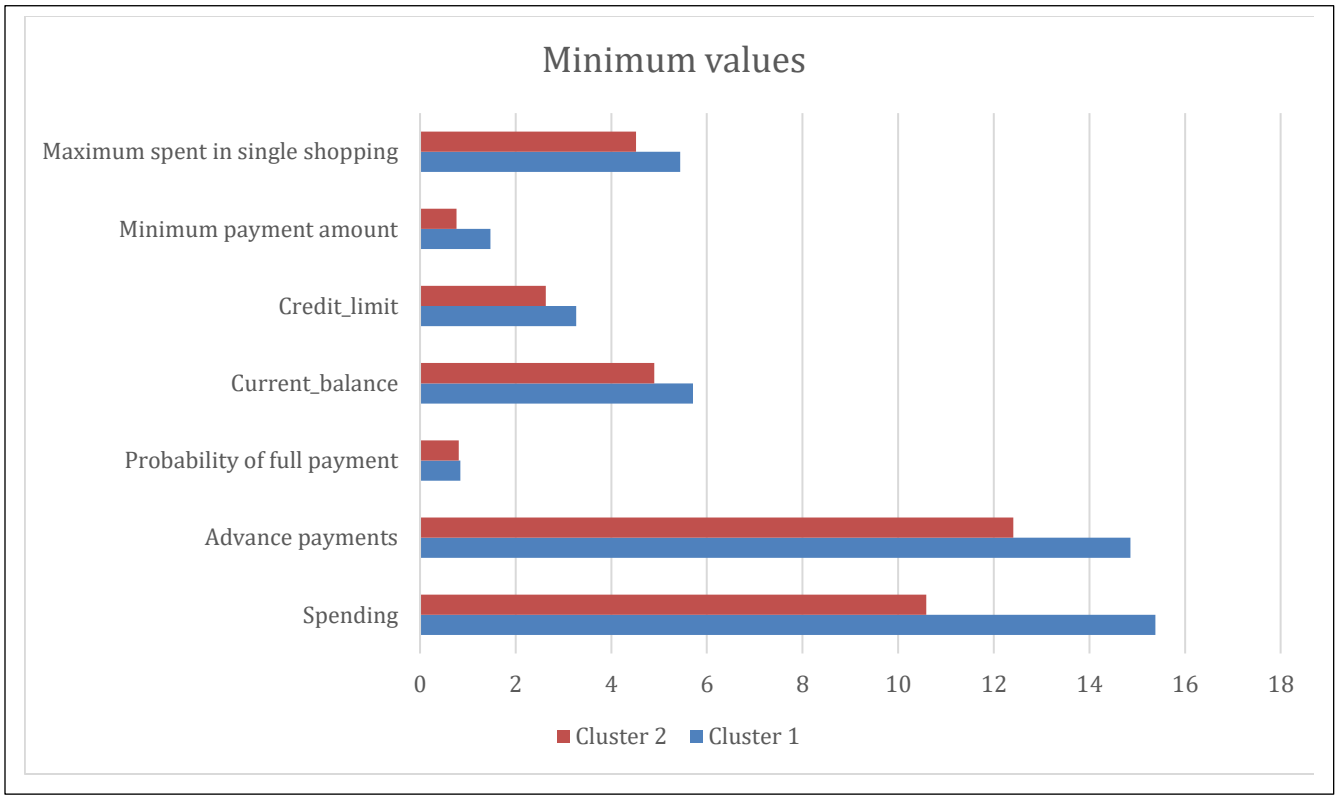
	Spending	Advance payments	Probability of full payment	Current_balance	Credit_limit	Minimum payment amount	Maximum spent in single shopping
	Cluster 1	(70 records)					
Average	18.37142857	16.14542857	0.8844	6.158171429	3.684628571	3.639157143	6.017371429
Sum	1286	1130.18	NA	431.072	257.924	254.741	421.216
Min	15.38	14.86	0.8452	5.709	3.268	1.472	5.443
Max	21.18	17.25	0.9108	6.675	4.033	6.682	6.55
	Cluster 2	(140 records)					
Average	13.08557143	13.76621429	0.864337589	5.363714286	3.045592857	3.726353214	5.103421429
Sum	1831.98	1927.27	NA	750.92	426.383	521.68945	714.479
Min	10.59	12.41	0.8105875	4.899	2.63	0.7651	4.519
Max	16.63	15.46	0.9183	6.053	3.582	8.079625	5.879

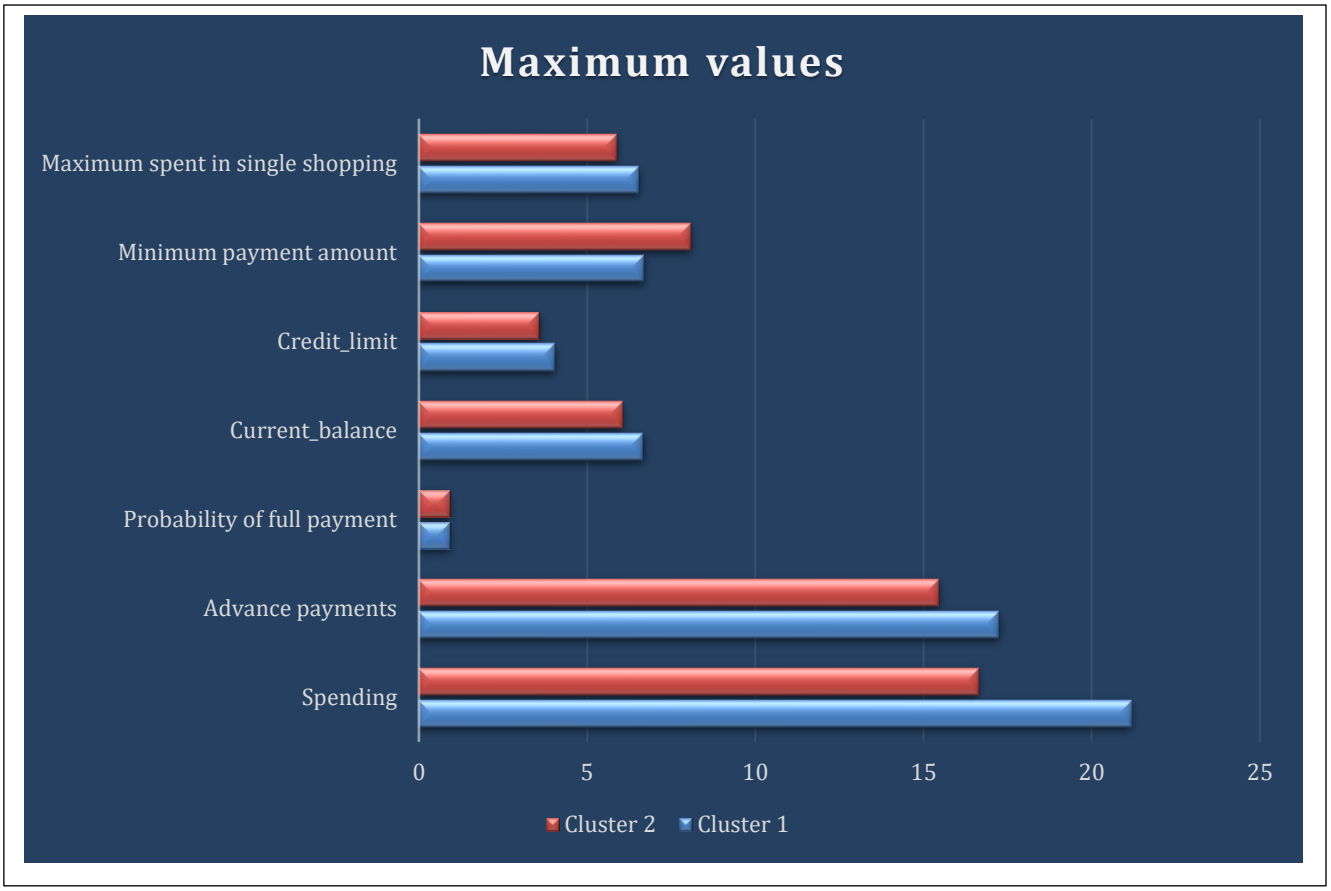


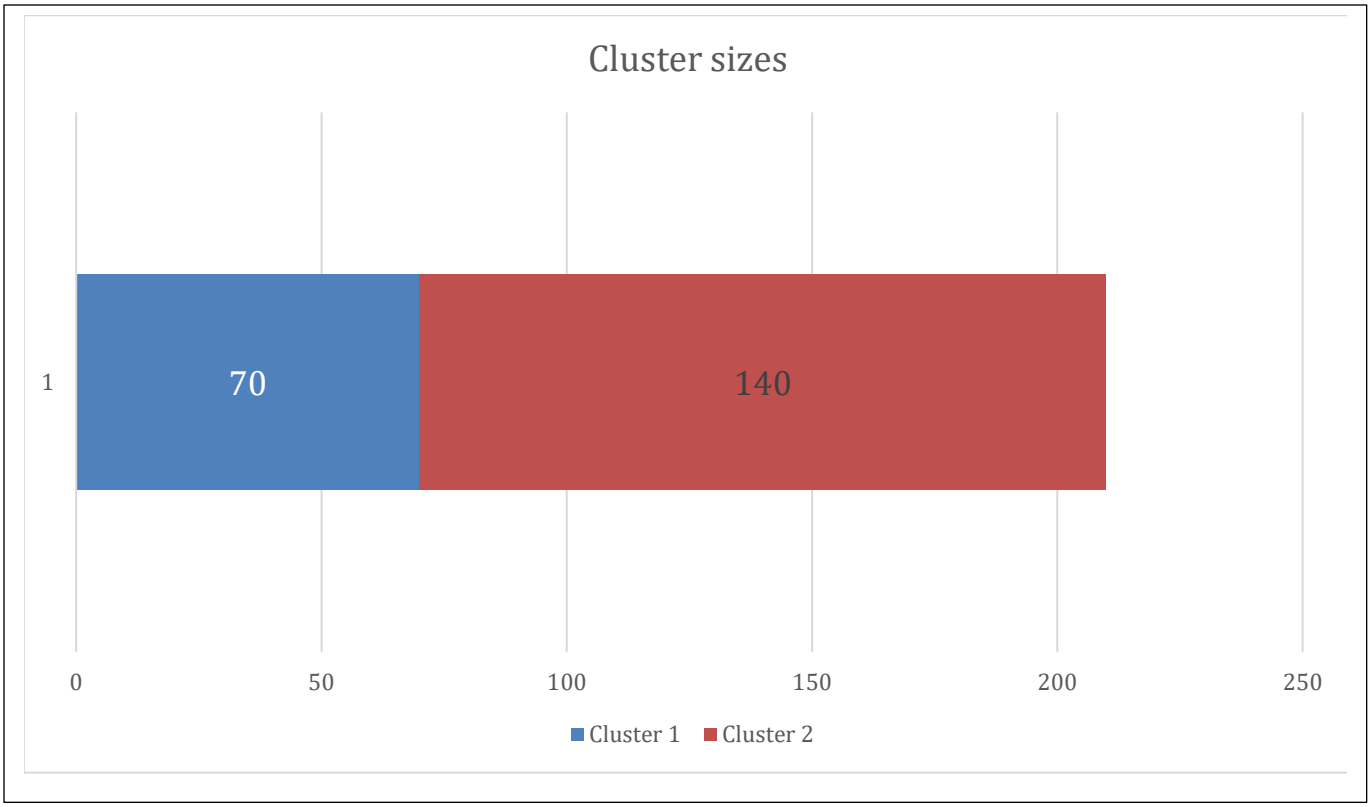



Things on an average
Maximum spent in single shopping
Minimum payment amount
Credit_limit
Current_balance
Probability of full payment
Advance payments
0
2
4
6
8
10
12
14
16
18
Cluster 2
Cluster 1


When it comes to total
Maximum spent in single shopping
Minimum payment amount
Credit_limit
Current_balance
Advance payments
0
500
1000
1500
2000
2500
Cluster 2
Cluster 1


Minimum values
Maximum spent in single shopping
Minimum payment amount
Credit_limit
Current_balance
Probability of full payment
Advance payments
Spending
0
2
4
6
8
10
12
14
16
18
Cluster 2
Cluster 1


Maximum values
Maximum spent in single shopping
Minimum payment amount
Credit_limit
Current_balance
Probability of full payment
Advance payments
Spending
0
5
10
15
20
25
Cluster 2
Cluster 1



Recommendations

Outcomes

1. Like said before, customers from cluster 1 are better at advance payments than those in cluster 2.

2. Examination of the mean and standard deviation shows that the customers in cluster 1 have a higher spending habit than those in cluster 2.

3. Cluster 1 spends more in single shopping, yet both clusters have only a marginal difference in credit limit.

4. Their probabilities of making full payment are almost the same.

Suggestions

1. Since customers in cluster 2 lag in advance payment, the bank can introduce schemes such as doubling the reward points of whoever pays in advance.

2. To increase the minimum payment amount, the bank can introduce some cashback offers of up to a certain amount's worth of minimum purchase.

3. Since the customers in cluster 1 have a better record, the bank can consider them its premium clients and think of introducing cashback offers for them to encourage more on spending.

4. It'll be fair and good to increase the credit limit of premium clients to separate them from normal customers. Also, because there is a strong correlation between spending and credit limit. We can increase the credit limit for both category of customer segments, while giving the higher spenders a lot more freedom.

We saved cluster profiles in a csv file and made these Excel graphics based on its output.

Checking out the 3-cluster model

Method=wardlink
Criterion='distance
Distance=20

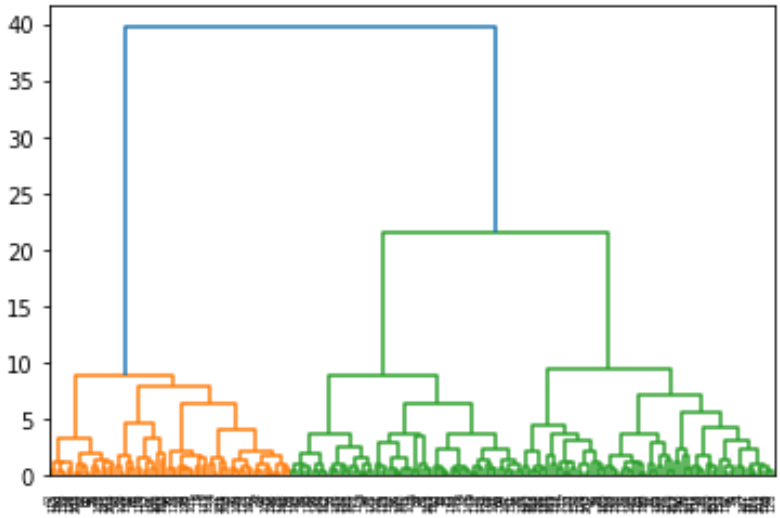


Fig 18: Dendrogram

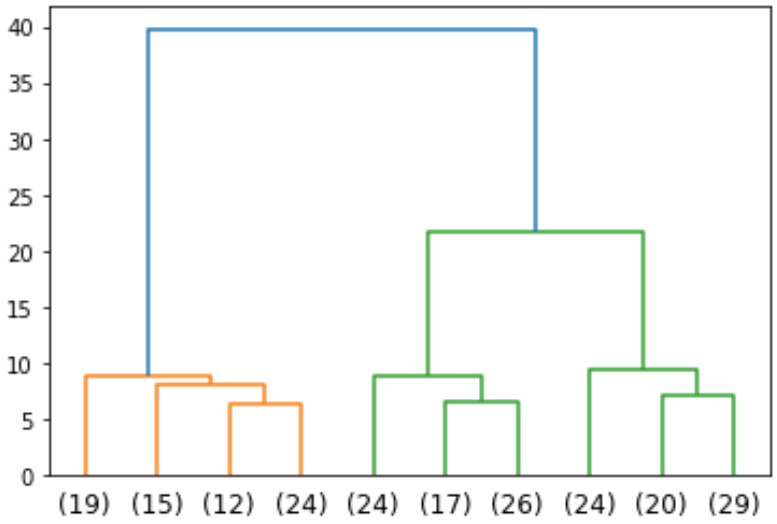


Fig 19: Truncated dendrogram

Clusters

array([1, 3, 1, 2, 1, 2, 2, 3, 1, 2, 1, 3, 2, 1, 3, 2, 3, 2, 3, 2, 2, 2,

1, 2, 3, 1, 3, 2, 2, 2, 3, 2, 2, 3, 2, 2, 2, 2, 2, 1, 1, 3, 1, 1,

2, 2, 3, 1, 1, 1, 2, 1, 1, 1, 1, 1, 2, 2, 2, 1, 3, 2, 2, 3, 3, 1,

1, 3, 1, 2, 3, 2, 1, 1, 2, 1, 3, 2, 1, 3, 3, 3, 3, 1, 2, 3, 3, 1,

1, 2, 3, 1, 3, 2, 2, 1, 1, 1, 2, 1, 2, 1, 3, 1, 3, 1, 1, 2, 2, 1,

3, 3, 1, 2, 2, 1, 3, 3, 2, 1, 3, 2, 2, 2, 2, 3, 3, 1, 2, 3, 3, 2, 3,

3, 1, 2, 1, 1, 2, 1, 3, 3, 3, 2, 2, 3, 2, 1, 2, 3, 2, 3, 2, 3, 3,

3, 3, 3, 2, 3, 1, 1, 2, 1, 1, 1, 2, 1, 3, 3, 3, 3, 2, 3, 1, 1, 1,

3, 3, 1, 2, 3, 3, 3, 3, 1, 1, 3, 3, 3, 2, 3, 3, 2, 1, 3, 1, 1, 2,

1, 2, 3, 1, 3, 2, 1, 3, 1, 3, 1, 3], dtype=int32)

Dataset with appended 3 clusters

spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	clusters
19.94	16.92	0.875200	6.675	3.763	3.252	6.550	1
15.99	14.89	0.906400	5.363	3.582	3.336	5.144	3
18.95	16.42	0.882900	6.248	3.755	3.368	6.148	1
10.83	12.96	0.810588	5.278	2.641	5.182	5.185	2
17.99	15.86	0.899200	5.890	3.694	2.068	5.837	1

Tab 12: Dataset with appended 3 clusters

clusters	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	Freq
1	18.371429	16.145429	0.884400	6.158171	3.684629	3.639157	6.017371	70
2	11.872388	13.257015	0.848155	5.238940	2.848537	4.940302	5.122209	67
3	14.199041	14.233562	0.879190	5.478233	3.226452	2.612181	5.086178	73

Tab 13: Dataset pivoted on three clusters, with appended frequencies

Cluster profile

Cluster 1: **High spenders** with high advance payments

Cluster 2: **Low spenders** with high minimum payment amount and low credit limit

Cluster 3: **Medium spenders** with advance payment level 14 and least minimum payment

Cluster sizes

1 70
2 67
3 73

Using Agglomerative Clustering

Clusters=3
Affinity=Euclidean
Linkage=Ward

Checking columns

```
Index(['spending',
'advance_payments',
'probability_of_full_payment',
'current_balance', 'credit_limit',
'min_payment_amt',
'max_spent_in_single_shopping',
'Agglo_CLusters3'], dtype='object')
```

Agglomerative clusters

[0 1 0 2 0 2 2 1 0 2 0 1 2 0 1 2 1 2 1 2 2 2 0 2 1 0 1 2 2 2 1 2 2 1 2 2 2
2 2 0 0 1 0 0 2 2 1 0 0 0 2 0 0 0 0 0 2 2 2 0 1 2 2 1 1 0 0 1 0 2 1 2 0 0
2 0 1 2 0 1 1 1 1 0 2 1 1 0 0 2 2 0 1 2 2 0 0 0 2 0 2 0 1 0 1 0 0 2 2 0 1
1 0 2 2 0 1 2 2 0 1 2 2 2 1 1 0 2 1 1 2 1 1 0 2 0 0 2 0 1 1 1 2 2 1 2 0 2
1 2 1 2 1 1 1 1 1 2 1 0 0 2 0 0 0 2 0 1 1 1 1 2 1 0 0 0 1 1 0 2 1 1 1 1 0
0 1 1 1 2 1 1 2 0 1 0 0 2 0 2 1 0 1 2 0 1 0 1 0 1]

Appended 3 agglomerative clusters

Agglo_CLusters3	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
0	18.371429	16.145429	0.884400	6.158171	3.684629	3.639157	6.017371
1	14.200845	14.240141	0.878425	5.484437	3.223239	2.517890	5.091676
2	11.937971	13.278551	0.849841	5.239493	2.862797	4.969844	5.115507

Agglo_CLusters3	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
0	1286.00	1130.18	61.908000	431.072	257.924	254.74100	421.216
1	1008.26	1011.05	62.368200	389.395	228.850	178.77020	361.509
2	823.72	916.22	58.639063	361.525	197.533	342.91925	352.970

Agglo_CLusters3	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
0	21.18	17.25	0.9108	6.675	4.033	6.682000	6.550
1	16.63	15.46	0.9153	6.053	3.582	5.593000	5.879
2	14.28	14.17	0.9183	5.541	3.383	8.079625	5.491

Agglo_CLusters3	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
0	15.38	14.86	0.845200	5.709	3.268	1.4720	5.443
1	11.23	12.63	0.833500	4.902	2.719	0.7651	4.519
2	10.59	12.41	0.810588	4.899	2.630	3.0820	4.781

Tab 13,14,15,16: Dataset pivoted on 3 agglomerative clusters (versions: mean, sum, maximum, and minimum)

Recommendations

Beyond what has been proposed for the two categories before, here's something more we suggest:-

1. **Colour scheme:**

Using a simple colour scheme, the bank can introduce the gold, silver, and platinum cards for the three classes of customers, with a varying degree of benefits and special privileges such as extra points. Add higher limits, fees, reward points, and cache to higher cards.

2. **Loans:**

These cards can come with pre-approved loans (for housing, car, education, and retail), say silver for Rs 25 lakh, gold for Rs 25 lakh to Rs 50 lakh, and platinum for Rs 50 lakh to Rs 1 crore.

3. Rating:

The bank can start card-based credit rating to encourage more full payments.

4. Show time:

Run a scheme that allows card holders access to coveted show tickets before they are available to the general public. That'll encourage more spending across categories.

5. Link it to cash:

The platinum card, which, in addition to quadrupling the gold card fees, could offer cash availability anywhere around the world without the need for a cheque, besides hotel upgrades when available and other travel and concierge services.

6. Top class:

Create a high-quality magazine aimed solely at the top segment (which will also make big money on the advertising on account of being desired by high-end vendors as one of the very few magazines targeted fully to their target audience. The vendors will minimize wasted advertising.

7. Let's play:

Offers exclusive performances and evenings with different artists, authors, actors etc. Top spenders should get IPL, Euro Cup, World Cup box tickets for sense of exclusivity and greater following.

8. Boss card:

Introduce high-end credit card differentiation. Beyond silver, gold, or platinum, call it "the Boss card" or something, so exclusive you shouldn't be able to apply for it. It's a "by invitation only" product, and the invitations are extended only to the most credit-worthy customers, who spend at least a certain high amount of money using their card annually. Put a price on it and the card should be hand delivered in a designer box. Exclusive customers seek exclusive items and experiences.

9. Reward:

Reward the customers for the length and size of business. This will encourage them to stay with the bank, which will save it the cost of acquiring new customers. Instead of validity date, the cards could say: "Member since".

10. Use digital age:

Allow customers to transfer money via smartphone using their card number at any time using. This will encourage more middle spenders towards cashless transaction.

11. A card they can bank on:

To attract low spenders towards using their credit card more often or exploring more of its applications, add traditional banking features such as account opening, money transfer, bill payment, and retail transactions, since this class has to queue up for most of these services, otherwise. The bank can also link it with subsidies.

13. Hacker-proof:

For high spenders at least, replace magnetic strip on the card with biometric authentication to make their accounts and transactions safer, as they will use biometric scans in place of PIN or password. Remember, their accounts are bank's cash cow.

Suggested model out of the two

Go with 3 clusters

Since cluster analysis is essentially an exploratory approach, the interpretation of the resulting hierarchical structure is context-dependent and often several solutions are equally good from a theoretical point of view.

Even though cheaper, a generic credit card scheme tailored to the average will have the lowest response rate. A scheme that is tailored to each individual will have the highest cost. Three schemes tailored to three segments will be somewhere in between. This is the revenue side.

The silhouette score is best for 2 clusters but it's still far below 0.6. The score is only 0.46 or so, so, one the **clusters are not all that well separated**, and two**, the chances of calling it right are less than 50%**.

To maximize profit, the bank would want to keep segmenting until the marginal revenue from segmenting equals the marginal cost of segmenting. In this case study, our suggestion to the bank is to use three segments to maximize profit. The cost of losing a customer segment can be huge.

1.4 Apply K-Means clustering on scaled data and determine optimum clusters. Apply elbow curve and silhouette score.

KMeans

K-means clustering is an unsupervised learning algorithm with a goal to find groups or assign the data points to clusters on the basis of their similarity. Which means the points in same cluster are similar to each other and dissimilar to those in different clusters. It was developed by researcher James Macqueen in 1967. Here K means the number of clusters. In order to find an appropriate number of clusters, the elbow method will be used. In this method for this case, the inertia for a number of clusters between 2 and 10 will be calculated. The rule is to choose the number of clusters where you see a kink or "an elbow" in the graph.

Calculating Within Cluster Sum of Squares or WSS for 10 values of K

[1469.999999999999,

659.14740095485,

430.298481751223,

370.6909292210199,

327.8379162768708,

294.70740950686735,

262.5742242335426,

240.20718392679387,

222.41512322267613,

205.2353741836641]

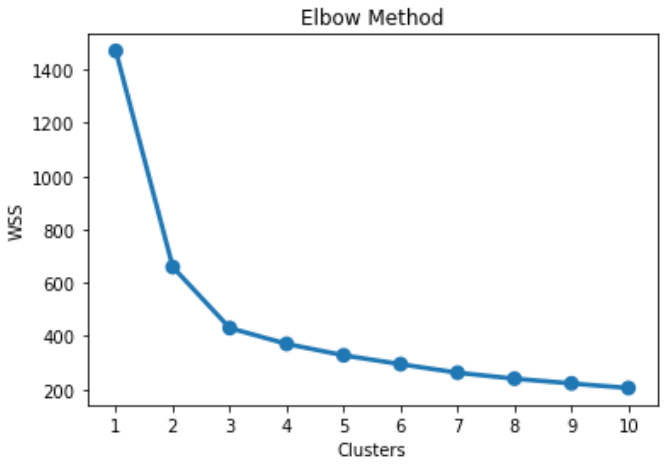


Fig 18.1: Line plot for the elbow method. WSS for 10 values of K are plotted on it.

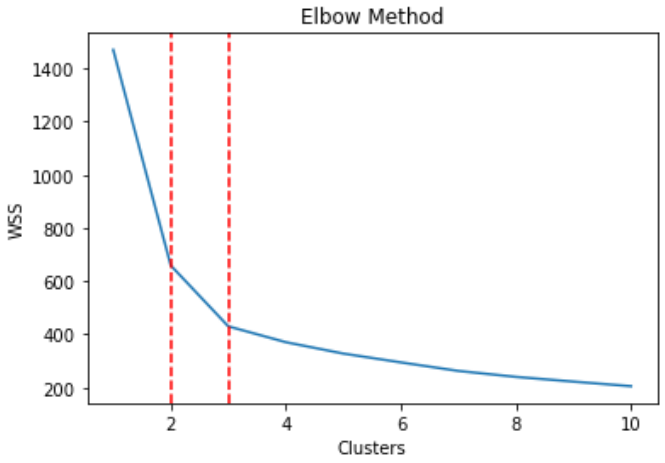


Fig 19.1: Double elbow marked in red.

Silhouette scores for different clusters

Silhouette score for 2 cluster k-means: 0.466
Silhouette score for 3 cluster k-means: 0.401
Silhouette score for 4 cluster k-means: 0.335
Silhouette score for 5 cluster k-means: 0.281
Silhouette score for 6 cluster k-means: 0.277
Silhouette score for 7 cluster k-means: 0.276
Silhouette score for 8 cluster k-means: 0.265
Silhouette score for 9 cluster k-means: 0.272
Silhouette score for 10 cluster k-means: 0.260

We have a double elbow, at 2 and 3 clusters, marked with vertical red lines. The other clusters have very little inertia.

Looking into the cluster sizes across both algorithm

Cluster	KM2_size	KM3_size
0	77.0	71
1	133.0	72
2	NaN	67

Tab 14: K-means 2 and K-means 3 cluster size comparison.

Silhouette score is better for 2 clusters than for 3 clusters. But we should also look at the elbow method. To determine the optimal number of clusters, we have to select the value of k at the **"elbow",** i.e. the point after which the distortion/inertia starts decreasing in a linear fashion for the given data. **We conclude that the optimal number of clusters for the data is 3**. It also seems from the size comparison of the algorithms that KMeans model with 3 clusters has more evenly balanced observations. **We'll go with 3 clusters.**

KMeans with K=3

Clusters

array([[0, 2, 0, 1, 0, 1, 1, 2, 0, 1, 0, 2, 1, 0, 2, 1, 2, 1, 1, 1, 1, 1,
0, 1, 2, 0, 2, 1, 1, 1, 2, 1, 1, 2, 1, 1, 1, 1, 1, 0, 0, 2, 0, 0,
1, 1, 2, 0, 0, 0, 1, 0, 0, 0, 0, 0, 1, 1, 1, 0, 2, 1, 1, 2, 2, 0,
0, 2, 0, 1, 2, 1, 0, 0, 1, 0, 2, 1, 0, 2, 2, 2, 2, 0, 1, 2, 0, 2,
0, 1, 2, 0, 2, 1, 1, 0, 0, 0, 1, 0, 2, 0, 2, 0, 2, 0, 0, 1, 1, 0,
2, 2, 0, 1, 1, 0, 2, 2, 1, 0, 2, 1, 1, 1, 2, 2, 0, 1, 2, 2, 1, 2,
2, 0, 1, 0, 0, 1, 0, 2, 2, 2, 1, 1, 2, 1, 0, 1, 2, 1, 2, 1, 2, 2,
1, 2, 2, 1, 2, 0, 0, 1, 0, 0, 0, 1, 2, 2, 2, 1, 2, 1, 2, 0, 0, 0,
2, 1, 2, 1, 2, 2, 2, 2, 0, 0, 1, 2, 2, 1, 1, 2, 1, 0, 2, 0, 0, 1,
0, 1, 2, 0, 2, 1, 0, 2, 0, 2, 2, 2])

Appending clusters to the original dataset

spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
19.94	16.92	0.875200	6.675	3.763	3.252	6.550
15.99	14.89	0.906400	5.363	3.582	3.336	5.144
18.95	16.42	0.882900	6.248	3.755	3.368	6.148
10.83	12.96	0.810588	5.278	2.641	5.182	5.185
17.99	15.86	0.899200	5.890	3.694	2.068	5.837

Tab 15: Dataset with appended K-means 3 cluster
Output saved in .csv file and graphs made on the basis.

KM3_cluster
0
2
0
1
0

Cluster profiling

0 67
1 72
2 71

The three customer segments are similar to the ones detected by hierarchical clustering model. We'll describe these, later.

KM3_cluster	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
0	18.495373	16.203433	0.884210	6.175687	3.697537	3.632373	6.041701
1	11.856944	13.247778	0.848330	5.231750	2.849542	4.733892	5.101722
2	14.437887	14.337746	0.881597	5.514577	3.259225	2.707341	5.120803

Tab 16: Dataset pivoted on K-means 3 clusters, with appended frequencies.

freq
67
72
71

Model evaluation

Computing silhouette width¶

array([0.5732776 , 0.36556355, 0.63709249, 0.515595 , 0.36097201,
0.22152508, 0.47529542, 0.36025848, 0.51938329, 0.53443903,
0.46599399, 0.12839864, 0.39177784, 0.52379458, 0.11202082,
0.22512083, 0.33760956, 0.5018087, 0.03635503, 0.23801566,
0.36177434, 0.3693663, 0.43153403, 0.26364196, 0.47484293,
0.06663956, 0.27151643, 0.50414367, 0.55487254, 0.43479958, **...**

...

0.55091698, 0.45775599, 0.04751093, 0.08299646, 0.44235462,
0.48260857, 0.07809599, 0.27491244, 0.40433159, 0.24770267,
0.33999491, 0.04992993, 0.40361423, 0.36916642, 0.45685928,
0.00276854, 0.36816915, 0.49743358, 0.54713282, 0.48730846,
0.26508389, 0.59700311, 0.39850516, 0.61330409, 0.47290575,
0.52337193, 0.09672676, 0.51720179, 0.5116529, 0.0473538,
0.30803559, 0.26742336, 0.5059218, 0.25717369, 0.04206292])

Showing **silhouette score** once again. It's the average of the silhouette widths.

0.4008059221522216

Appending silhouette width to dataset

	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
0	19.94	16.92	0.875200	6.675	3.763	3.252	6.550
1	15.99	14.89	0.906400	5.363	3.582	3.336	5.144
2	18.95	16.42	0.882900	6.248	3.755	3.368	6.148
3	10.83	12.96	0.810588	5.278	2.641	5.182	5.185
4	17.99	15.86	0.899200	5.890	3.694	2.068	5.837

Tab 17: Dataset with appended silhouette width

sil_width
0.573278
0.365564
0.637092
0.515595
0.360972

Minimum silhouette width
0.0027685411286160638

Checking for any negative silhouette width

The minimum silhouette width is on the positive side of zero but not closer to 1, so there are clusters, even though not well separated. There is no record with negative silhouette score, which means that the classification was proper. The model is verified.

High spenders cluster description

	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
count	67.000000	67.000000	67.000000	67.000000	67.000000	67.000000	67.000000
mean	18.495373	16.203433	0.884210	6.175687	3.697537	3.632373	6.041701
std	1.277122	0.546439	0.014917	0.237807	0.166014	1.211052	0.229566
min	15.560000	14.890000	0.845200	5.718000	3.387000	1.472000	5.484000
25%	17.590000	15.855000	0.874650	6.011500	3.564500	2.848000	5.879000
50%	18.750000	16.230000	0.882900	6.153000	3.719000	3.619000	6.009000
75%	19.145000	16.580000	0.898050	6.328000	3.808000	4.421000	6.192500
max	21.180000	17.250000	0.910800	6.675000	4.033000	6.682000	6.550000

Tab 18: Data summary for high spenders

Low spenders cluster description

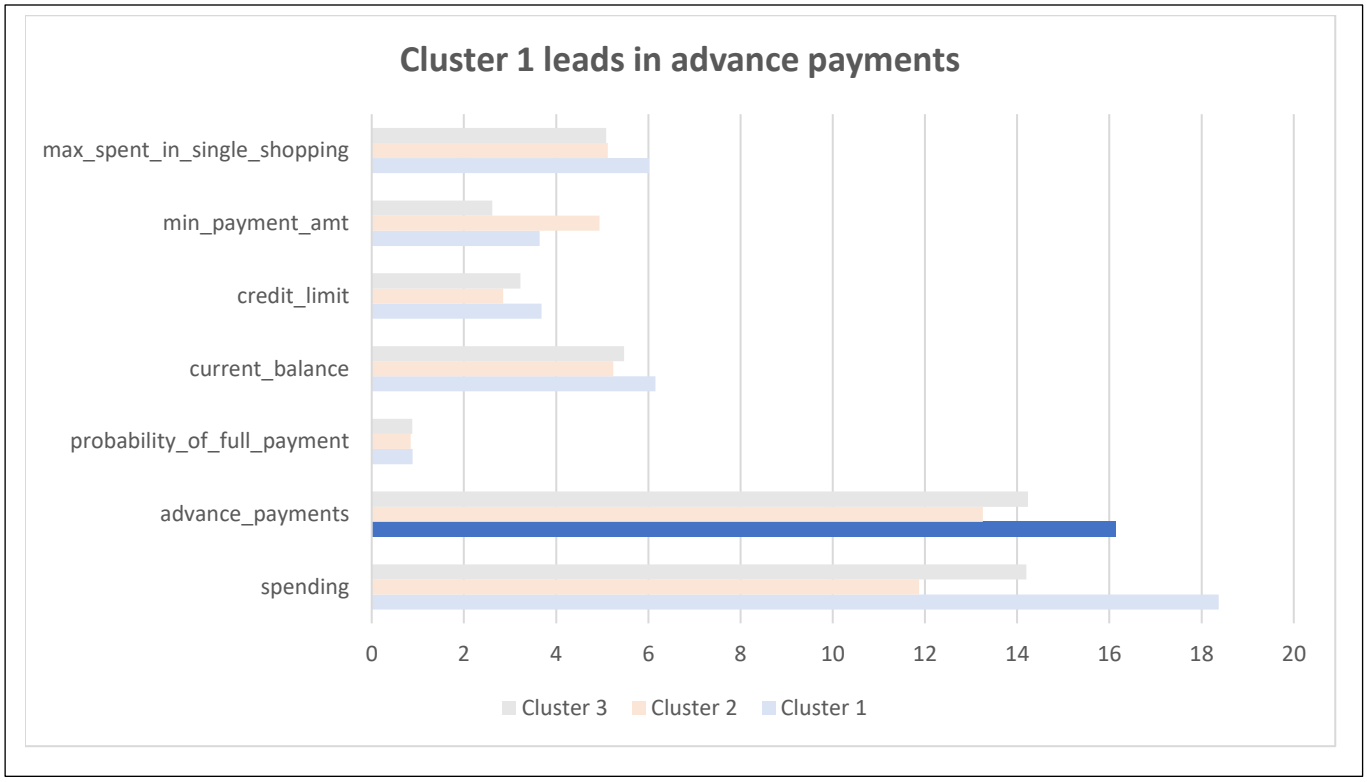
	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping
count	72.000000	72.000000	72.000000	72.000000	72.000000	72.000000	72.000000
mean	11.856944	13.247778	0.848330	5.231750	2.849542	4.733892	5.101722
std	0.714801	0.355208	0.019800	0.141795	0.138689	1.332248	0.184012
min	10.590000	12.410000	0.810588	4.899000	2.630000	1.502000	4.519000
25%	11.255000	12.992500	0.835000	5.139250	2.738500	4.032250	5.001000
50%	11.825000	13.250000	0.848600	5.225000	2.836500	4.799000	5.089000
75%	12.395000	13.482500	0.861475	5.337250	2.967000	5.463750	5.223500
max	13.340000	13.950000	0.888300	5.541000	3.232000	8.079625	5.491000

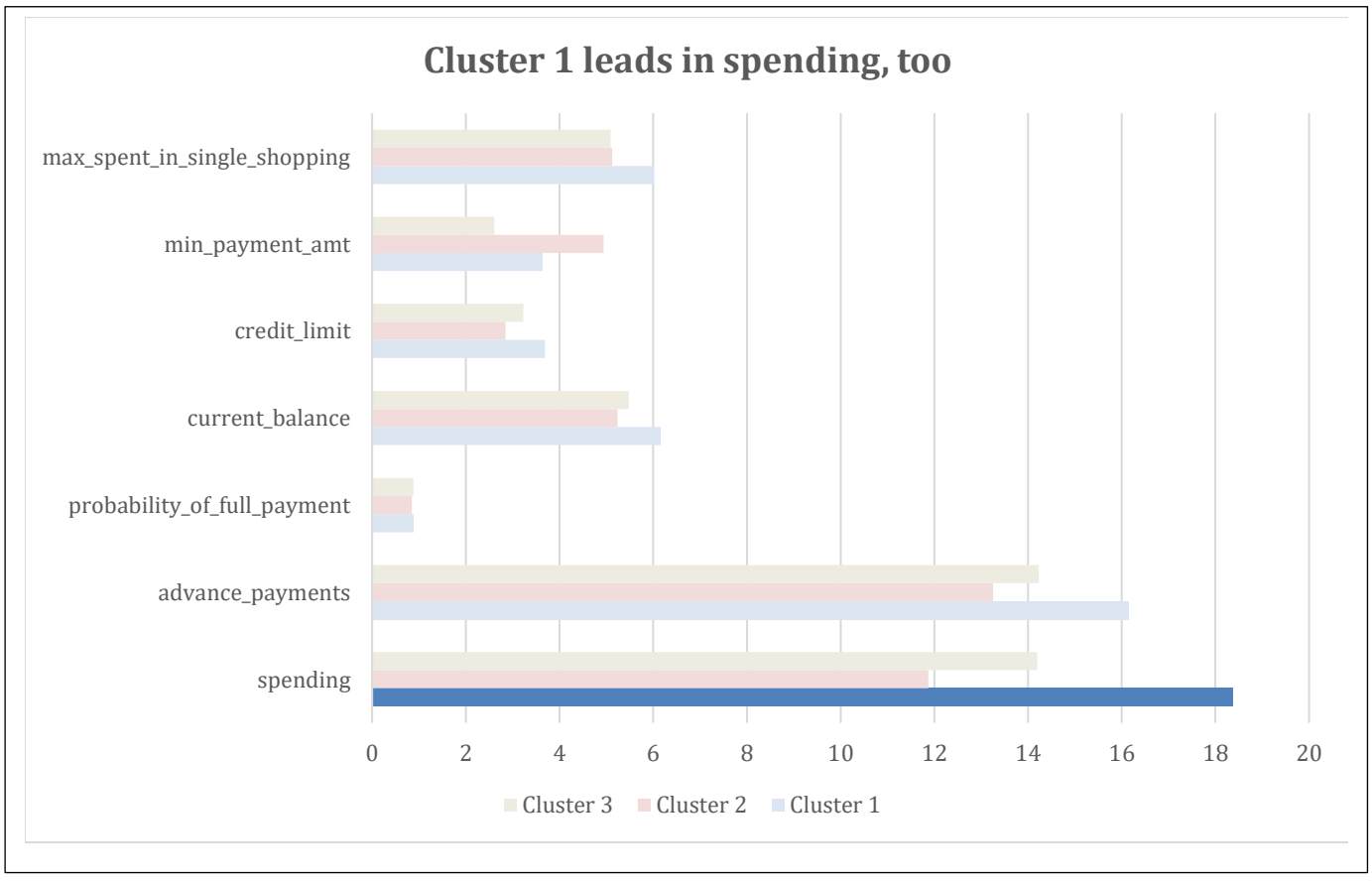
Tab 19: Data summary for low spenders

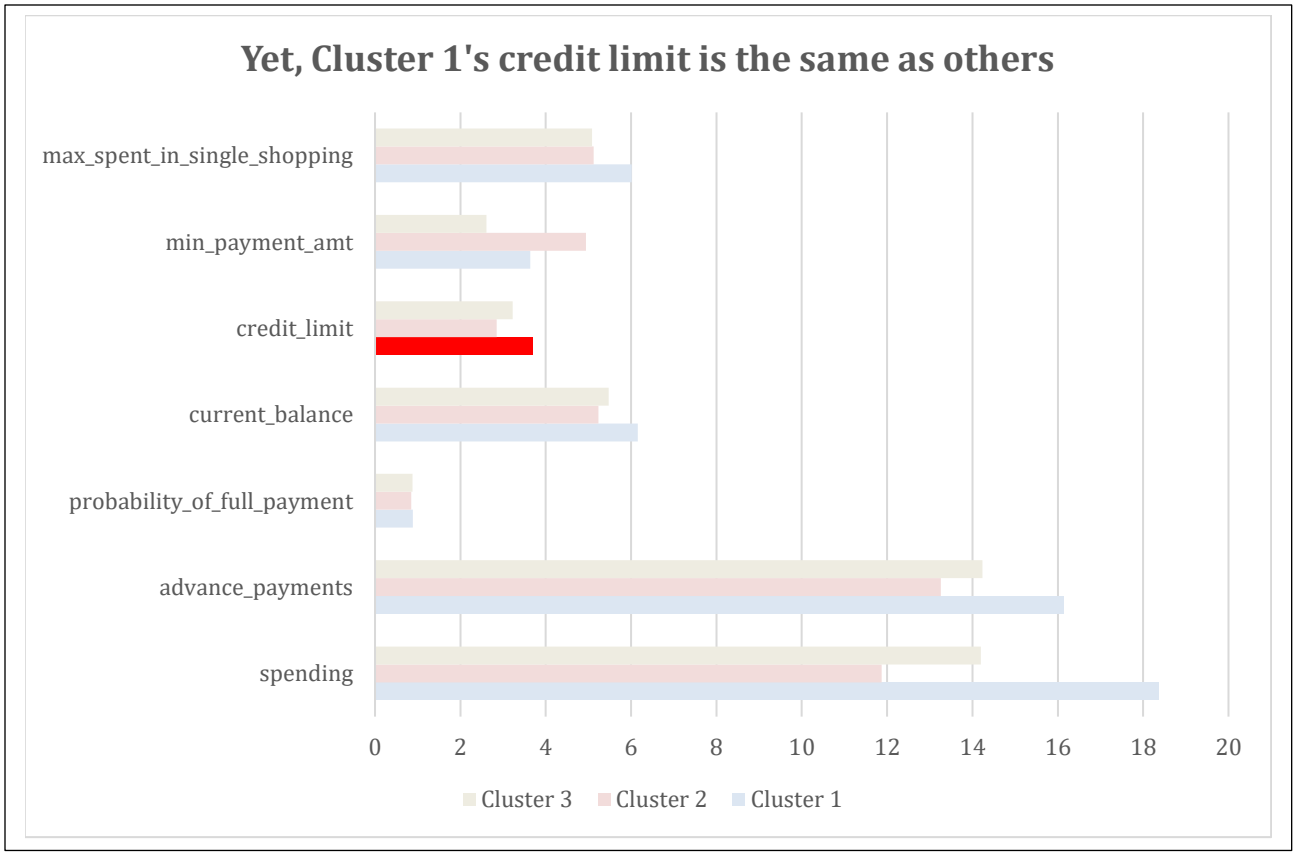
Medium spenders cluster description

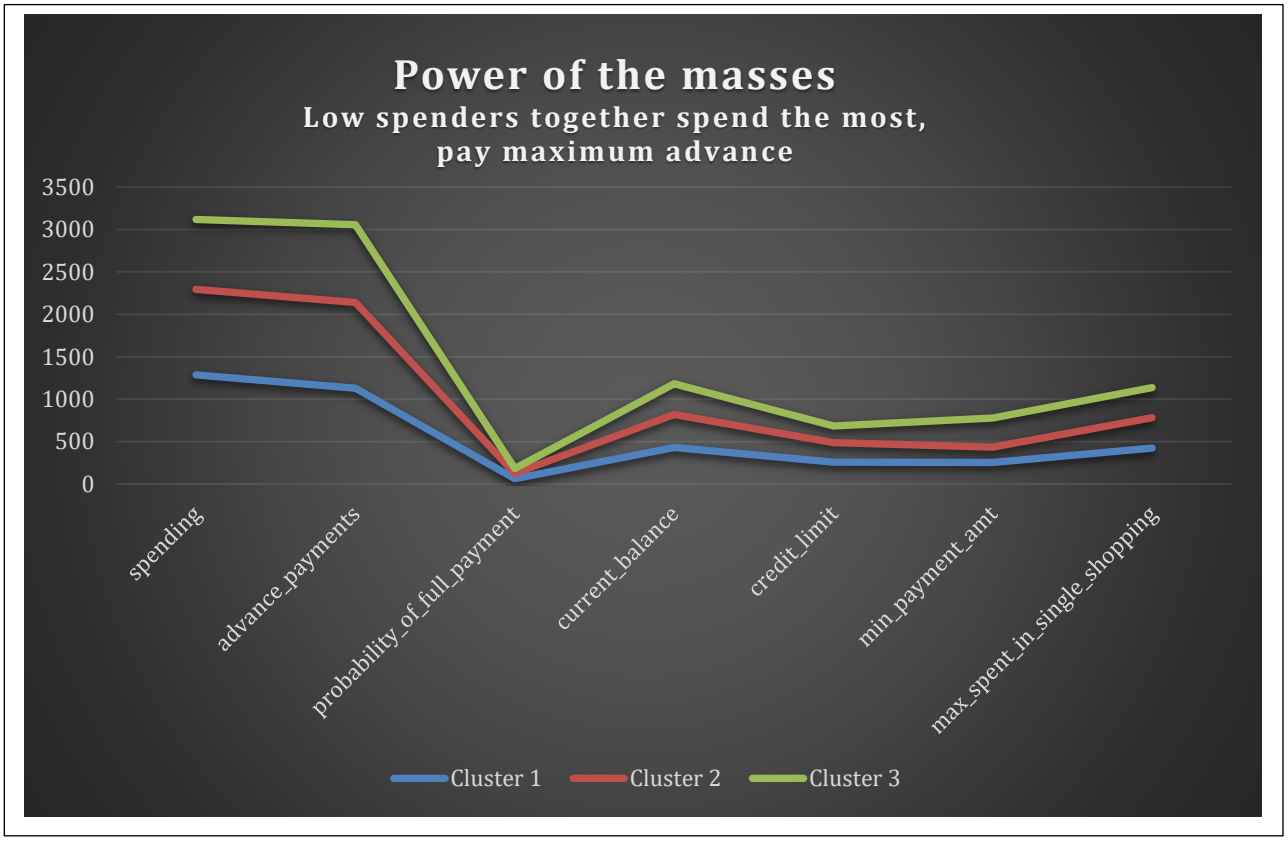
	spending	advance_payments	probability_of_full_payment	current_balance	credit_limit	min_payment_amt	max_spent_in_single_shopping	clusters	A
count	71.000000	71.000000	71.000000	71.000000	71.000000	71.000000	71.000000	71.000000	
mean	14.437887	14.337746	0.881597	5.514577	3.259225	2.707341	5.120803	2.873239	
std	1.056513	0.525706	0.015502	0.225266	0.154766	1.176440	0.269558	0.475983	
min	12.080000	13.150000	0.852700	4.984000	2.936000	0.765100	4.605000	1.000000	
25%	13.820000	14.030000	0.871300	5.380000	3.155000	1.951000	4.958500	3.000000	
50%	14.430000	14.390000	0.881900	5.541000	3.258000	2.640000	5.132000	3.000000	
75%	15.260000	14.760000	0.893350	5.689500	3.378000	3.332000	5.263500	3.000000	
max	16.440000	15.270000	0.918300	5.920000	3.582000	6.685000	5.879000	3.000000	

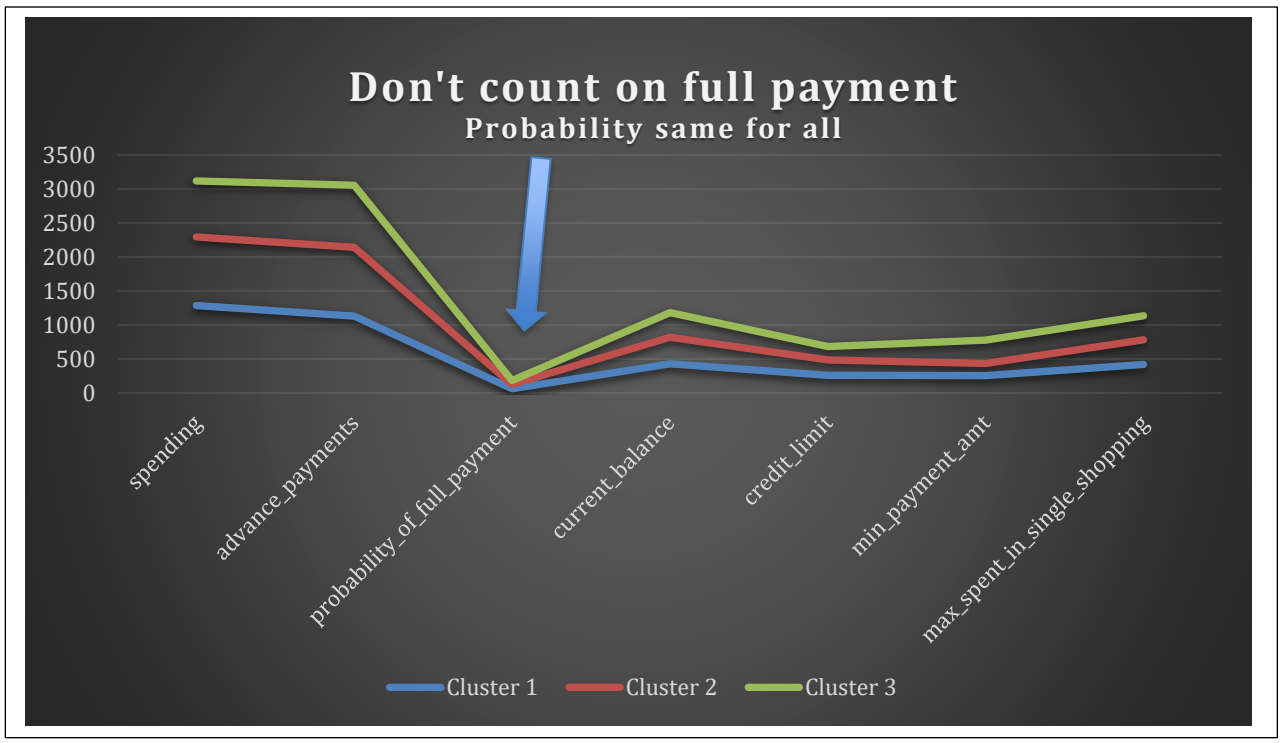
Tab 20: Data summary for high spenders

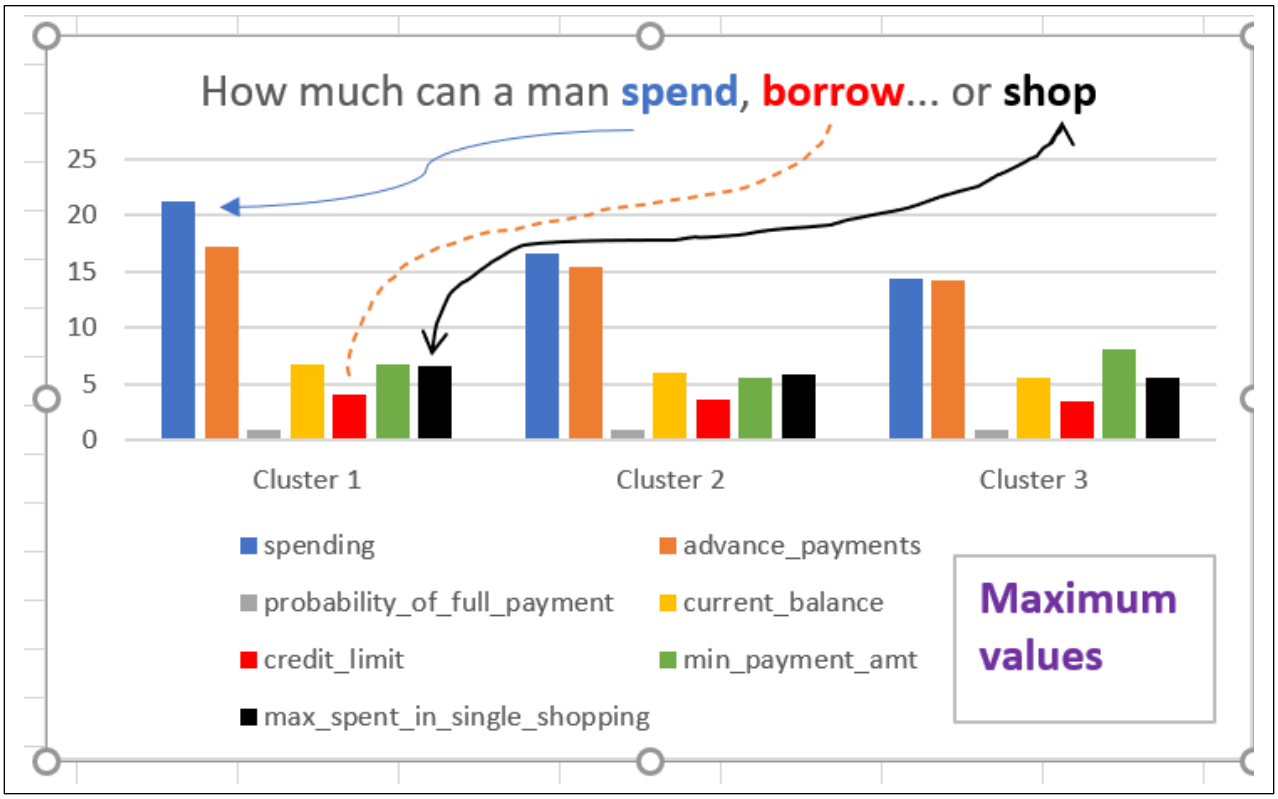

Cluster 1 leads in advance payments
max_spent_in_single_shopping
min_payment_amt
credit_limit
current_balance
probability_of_full_payment
advance_payments
spending
0
2
4
6
8
10
12
14
16
18
20
Cluster 3
Cluster 2
Cluster 1

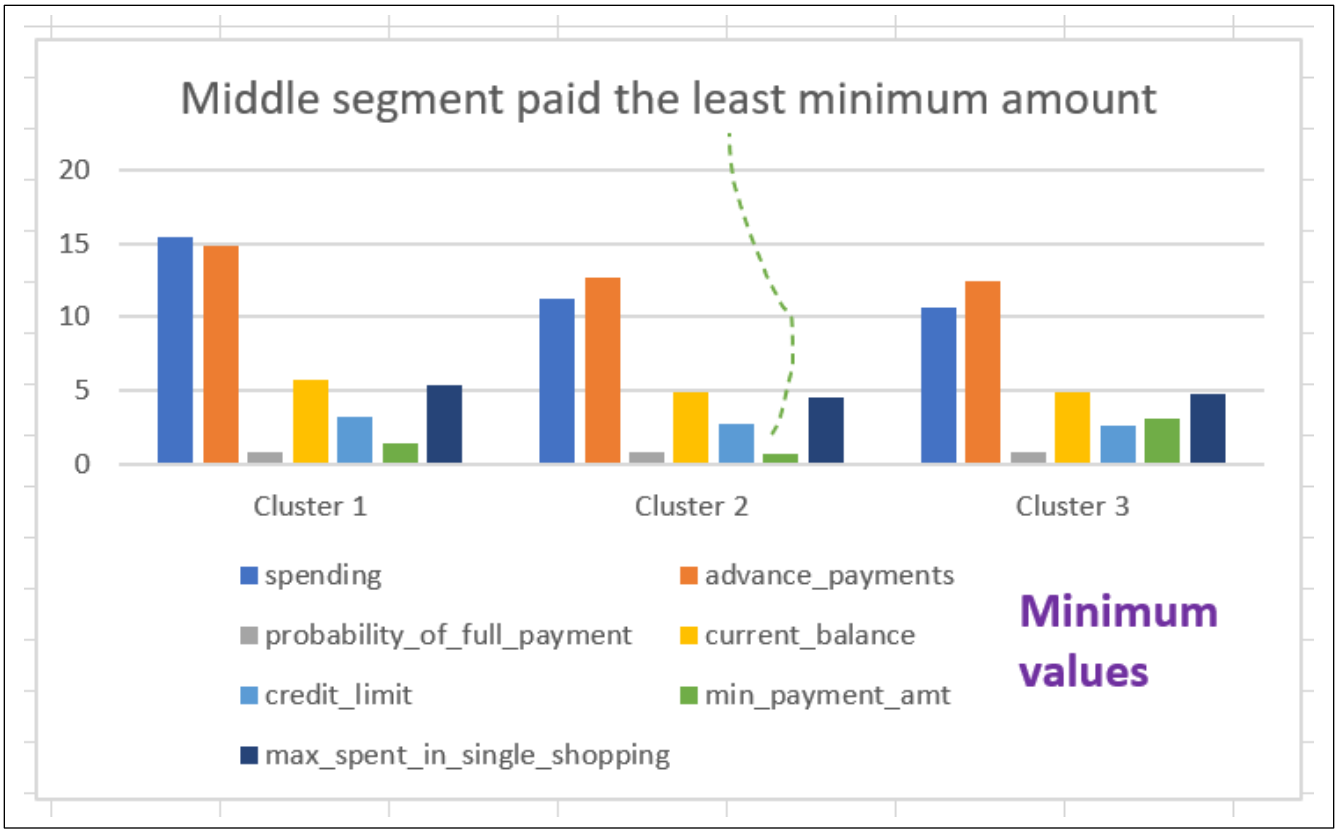

Cluster 1 leads in spending, too
max_spent_in_single_shopping
min_payment_amt
credit_limit
current_balance
probability_of_full_payment
advance_payments
spending
0
2
4
6
8
10
12
14
16
18
20
Cluster 3
Cluster 2
Cluster 1


Yet, Cluster 1's credit limit is the same as others
max_spent_in_single_shopping
min_payment_amt
credit_limit
current_balance
probability_of_full_payment
advance_payments
spending
0
2
4
6
8
10
12
14
16
18
20
Cluster 3
Cluster 2
Cluster 1


Power of the masses
Low spenders together spend the most,
pay maximum advance
3500
3000
2500
2000
1500
1000
500
0
spending
advance_payments
probability_of_full_payment
current_balance
credit_limit
min_payment_amt
max_spent_in_single_shopping
Cluster 1
Cluster 2
Cluster 3

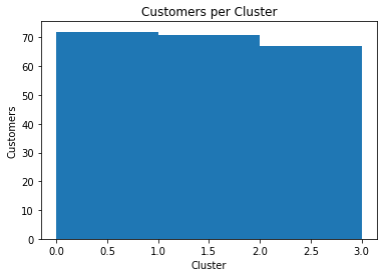

Don't count on full payment
Probability same for all
3500
3000
2500
2000
1500
1000
500
0
spending
advance_payments
probability_of_full_payment
current_balance
credit_limit
min_payment_amt
max_spent_in_single_shopping
Cluster 1
Cluster 2
Cluster 3


How much can a man spend, borrow... or shop
25
20
15
10
5
0
Cluster 1
Cluster 2
Cluster 3
spending
advance_payments
probability_of_full_payment
current_balance
credit_limit
min_payment_amt
max_spent_in_single_shopping
Maximum values



Visual representation of 3 clusters

Customers in each cluster



Variable histograms for each cluster

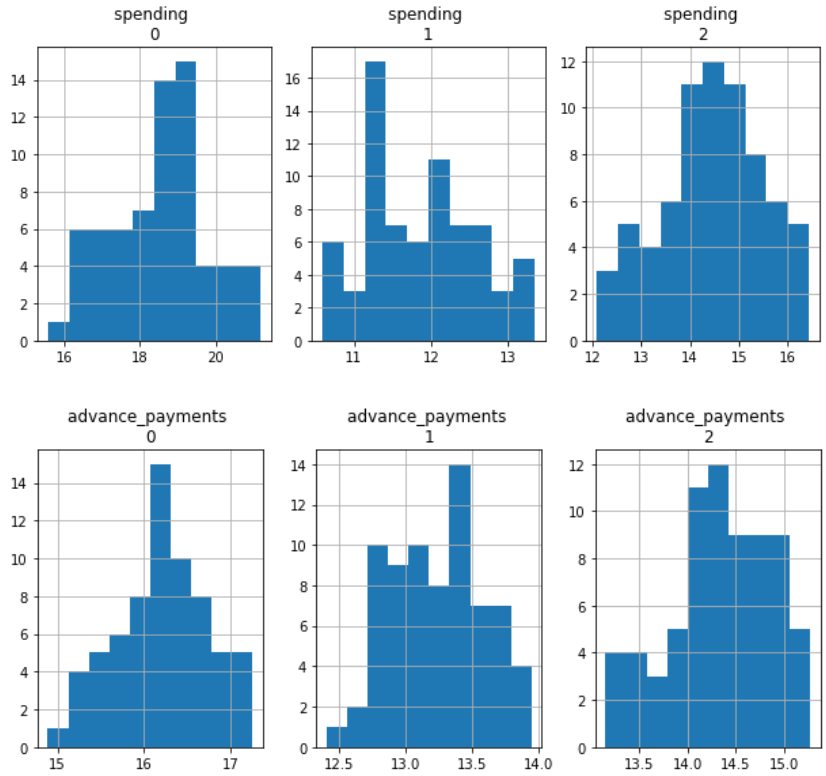
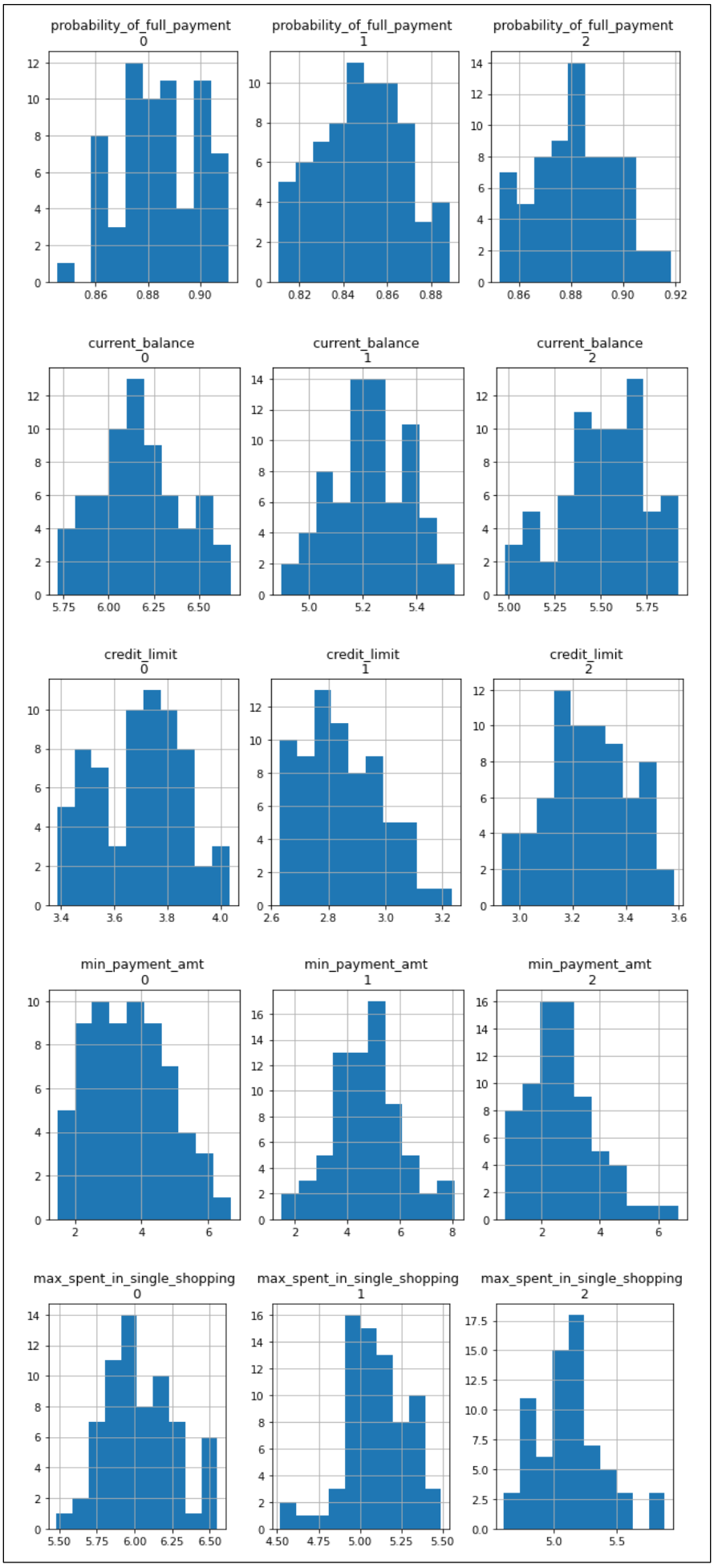


Fig 20: Variable histograms for each cluster


probability_of_full_payment
0
probability_of_full_payment
1
probability_of_full_payment
2
current_balance
0
current_balance
1
current_balance
2
credit_limit
0
credit_limit
1
credit_limit
2
min_payment_amt
0
min_payment_amt
1
min_payment_amt
2
max_spent_in_single_shopping
0
max_spent_in_single_shopping
1
max_spent_in_single_shopping
2

Scatter plot

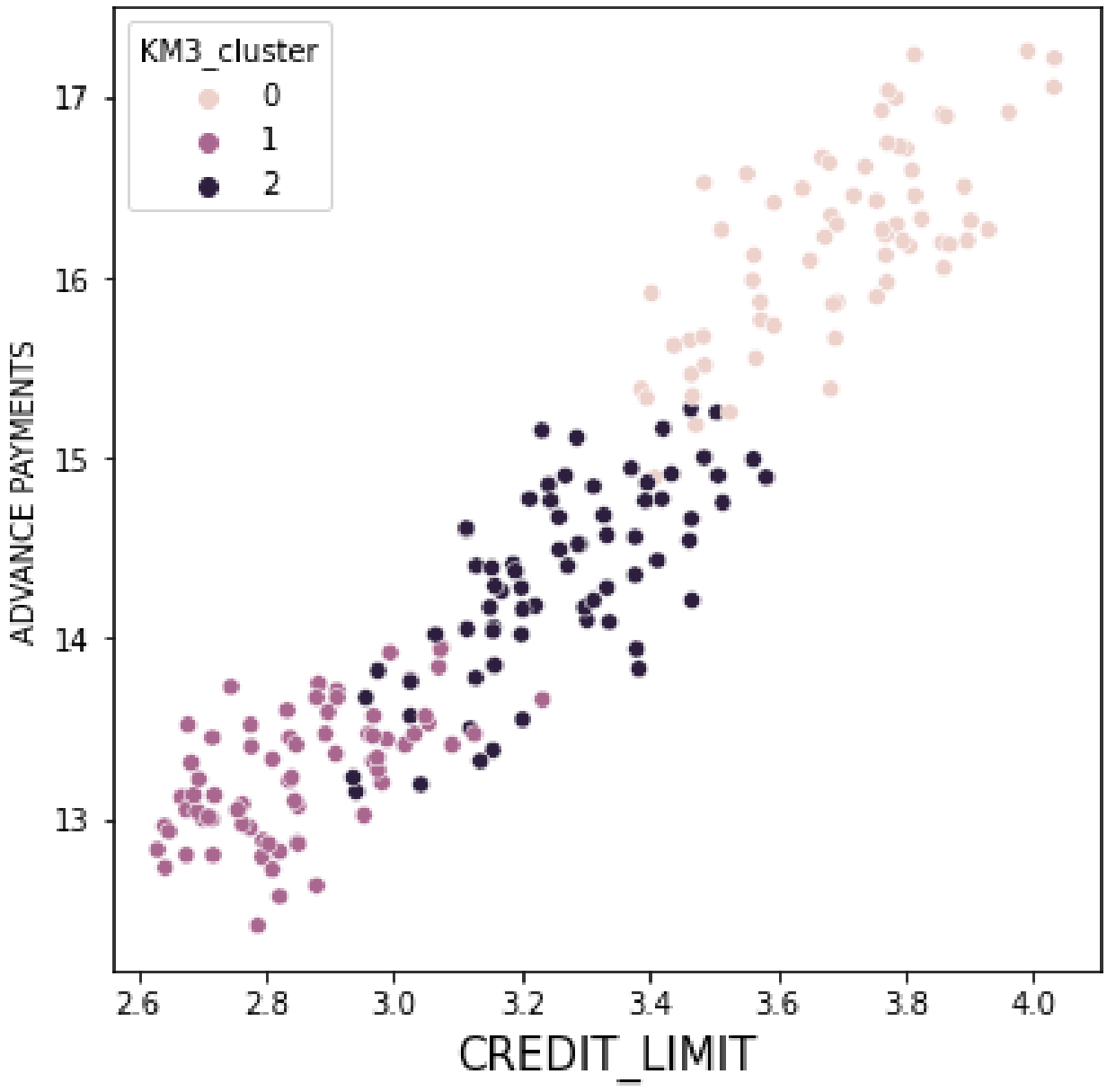


Fig 21: Scatterplot of credit limit and advance payments, hued on K-means 3 clusters

Pairplot hued on KM3 clusters.

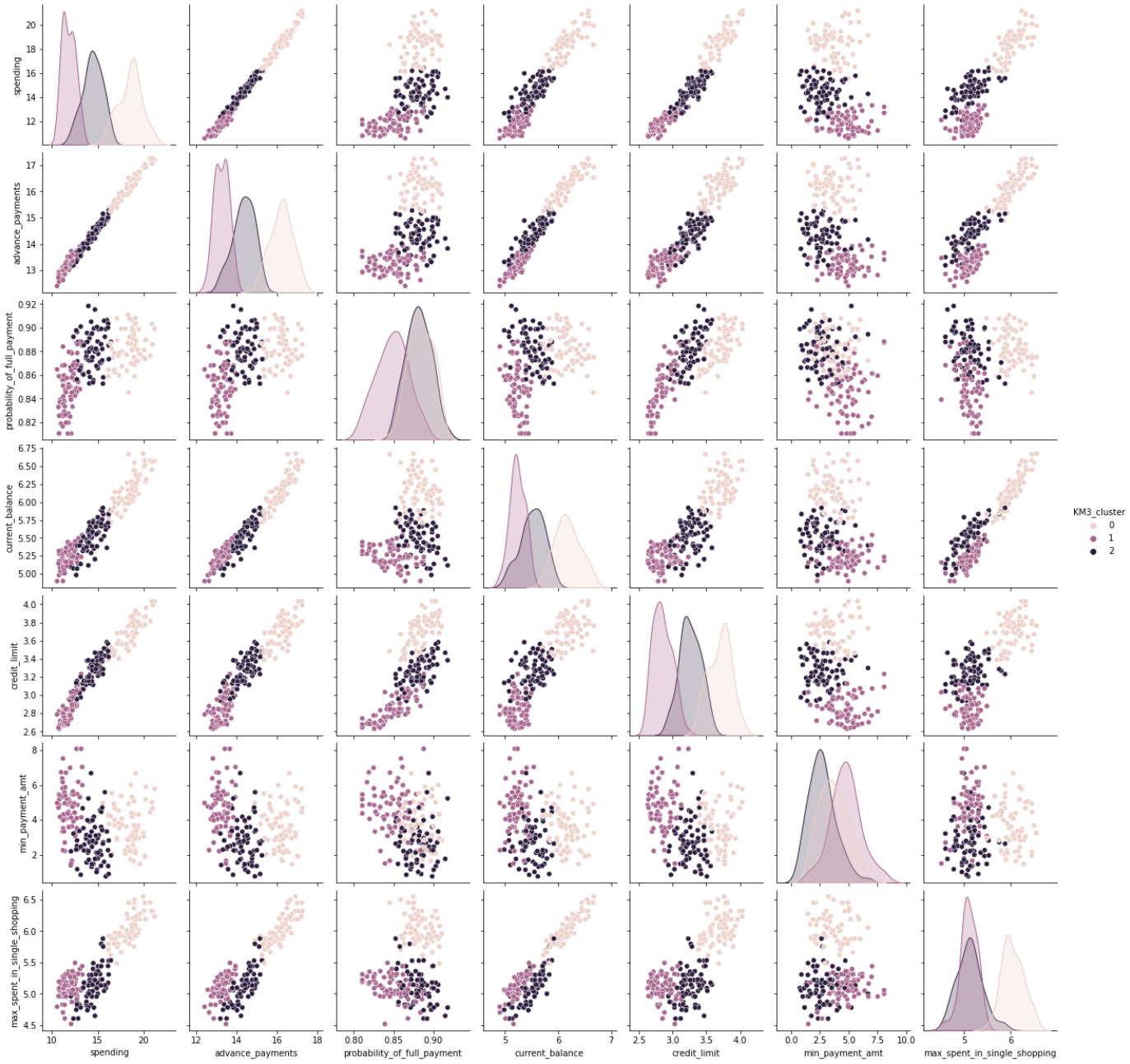


Fig 22: Pairplot hued on KM3 clusters. Quite different from the previous hued pairplots.

Different variables against advance payment

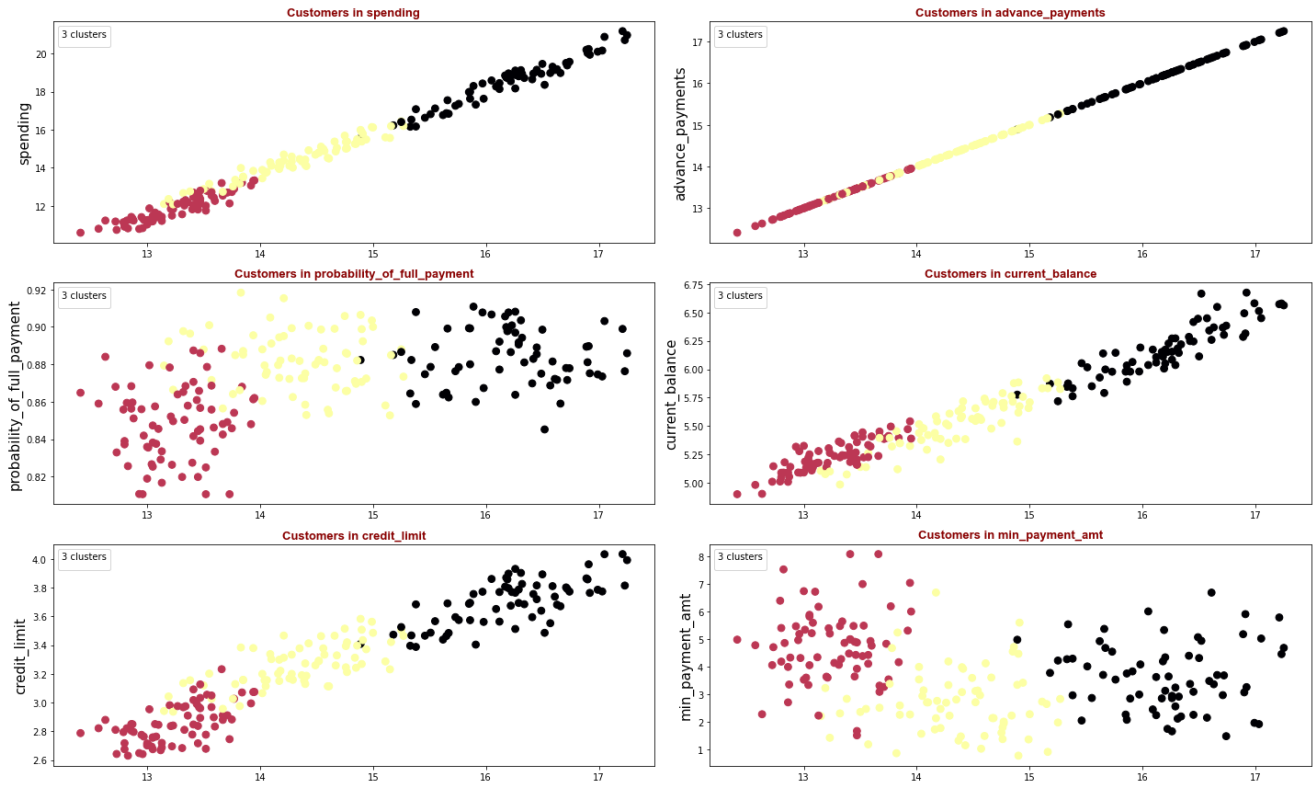


Fig 23: Scatter plots of different variables against advance payment.

Indicated: Customers with small, medium, and large spending, advance payments, and credit limit. Low spenders make a higher minimum payment. A customer segment indicated in black has credit limit way lower in comparison with their spending and advance payments. Probability of full payment is higher for the group marked in black.

Cluster details

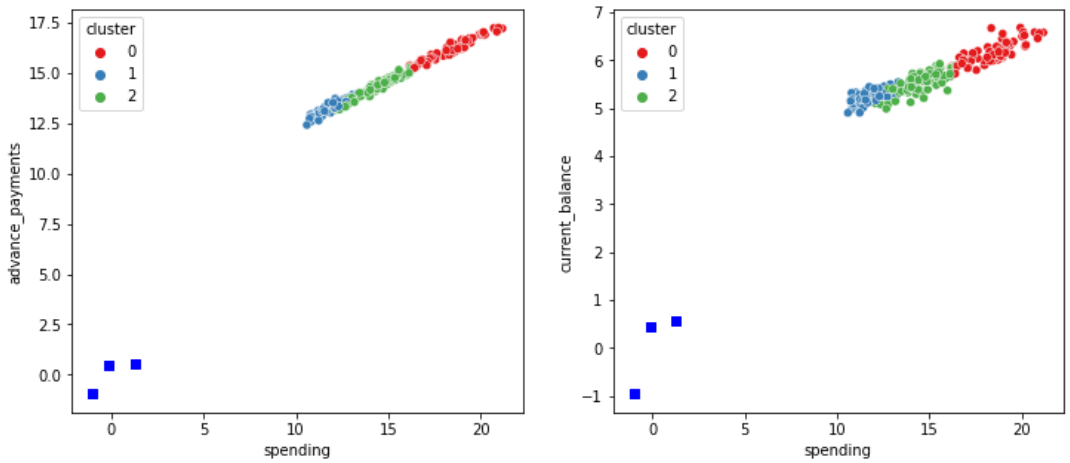


Fig 24: Spending against advance payments and current balance.

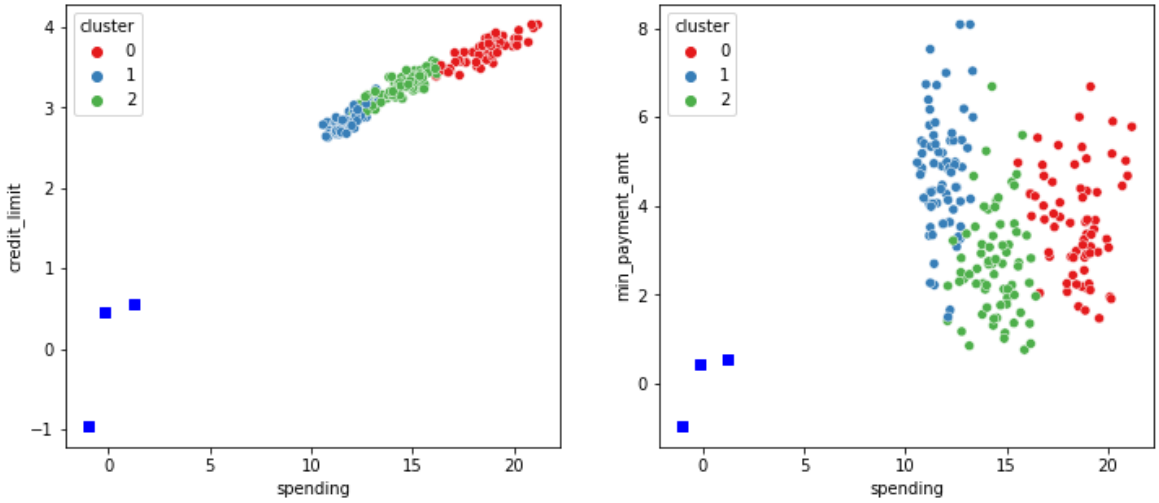


Fig 24: Spending against credit limit and minimum payment amount.

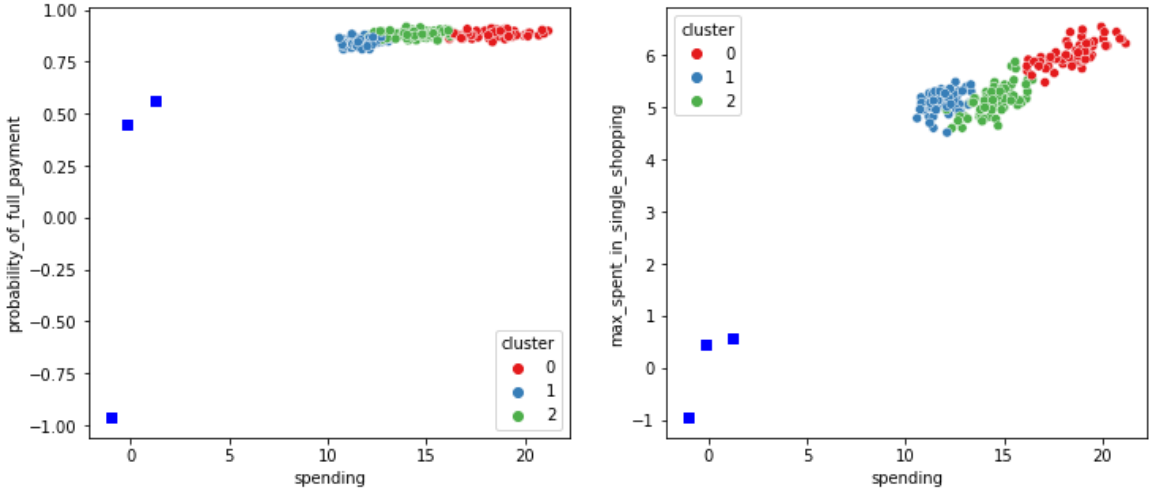


Fig 26: Spending against probability of full payment and maximum spent in single shopping.

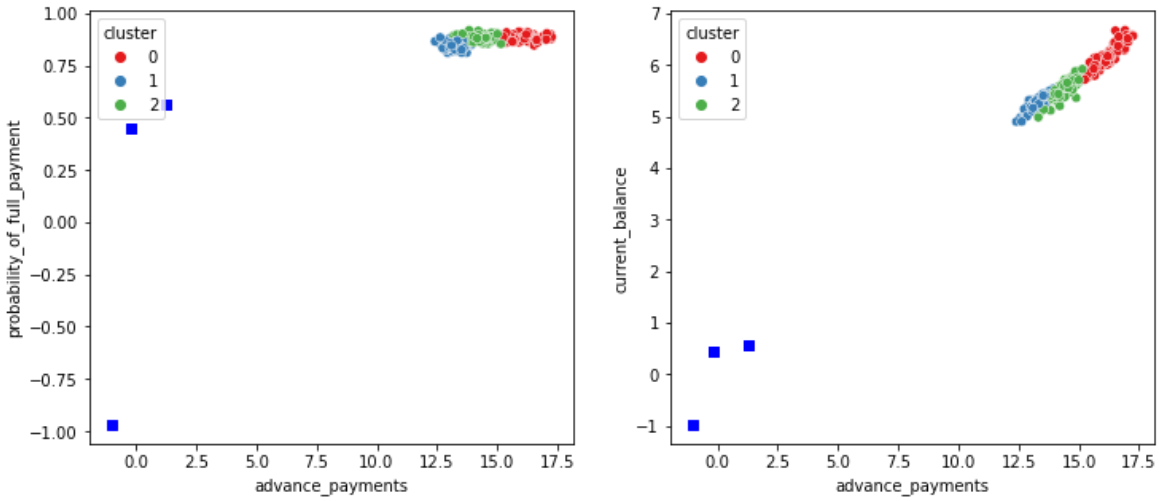


Fig 27: Advance payments against probability of full payment and current balance.

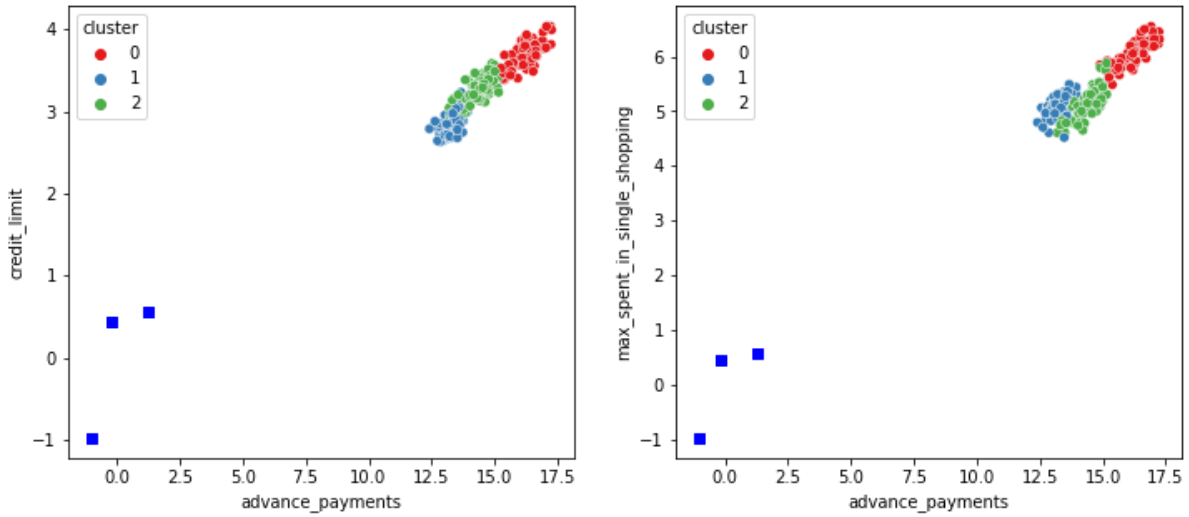


Fig 28: Advance payments against credit limit and maximum spent in single shopping.

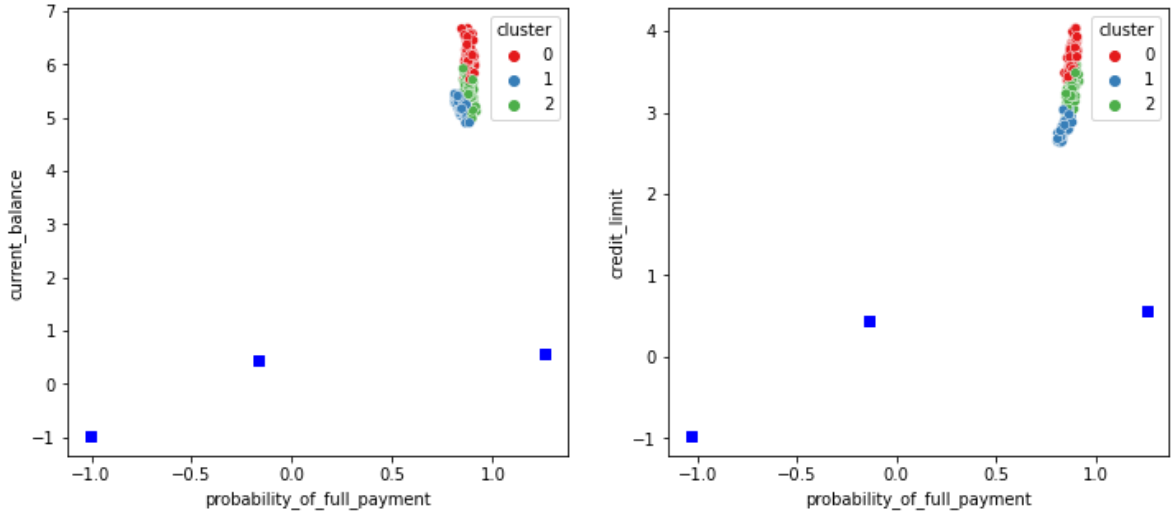


Fig 29: Probability of full payment against current balance and credit limit.

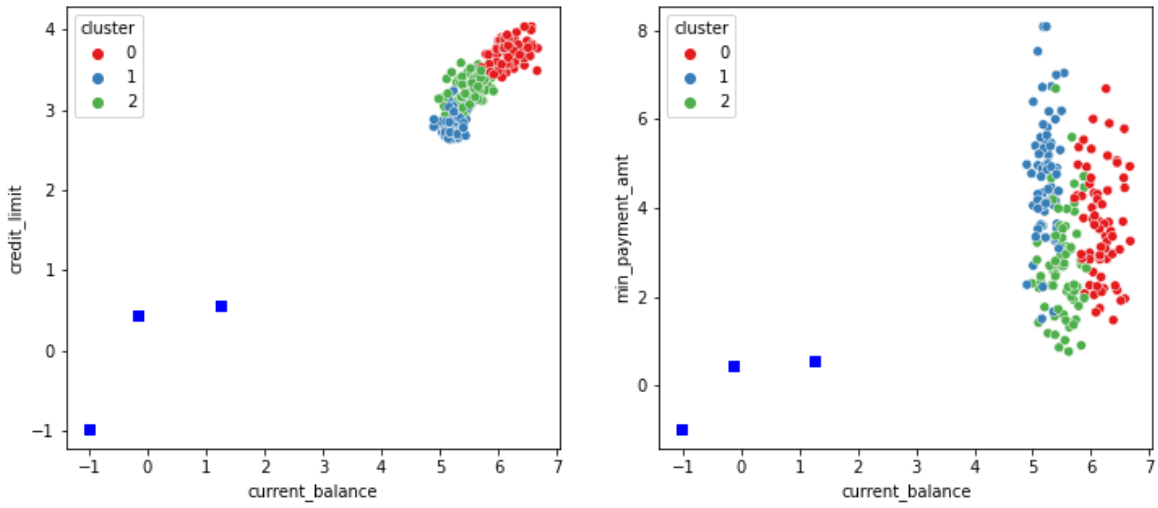


Fig 30: Current balance against credit limit and minimum payment amount.

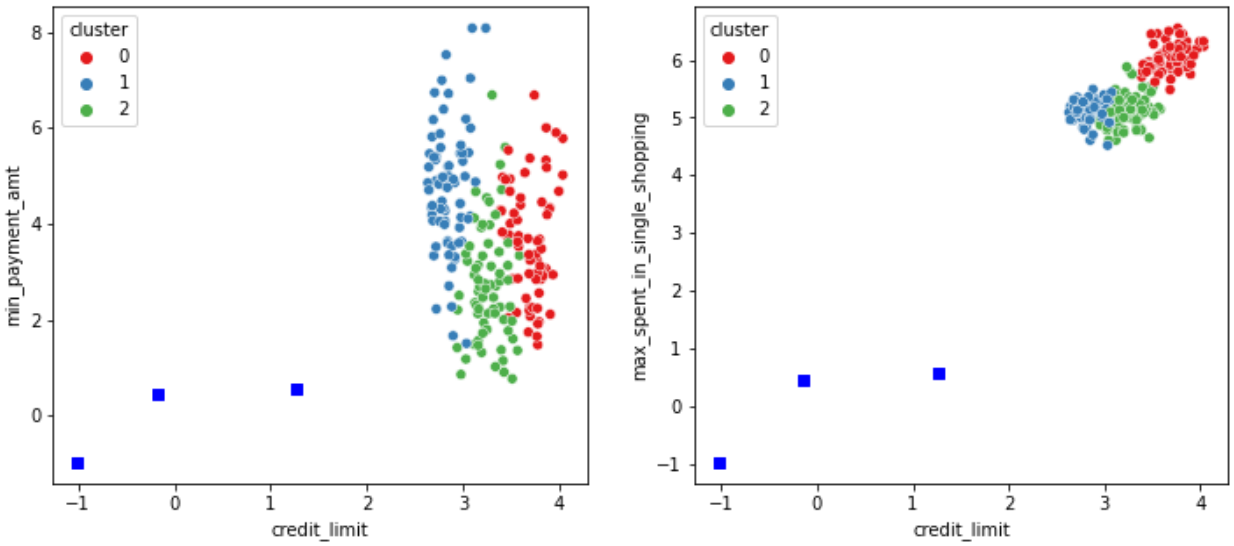


Fig 31: Credit limit against minimum payment amount and maximum spent in single shopping.

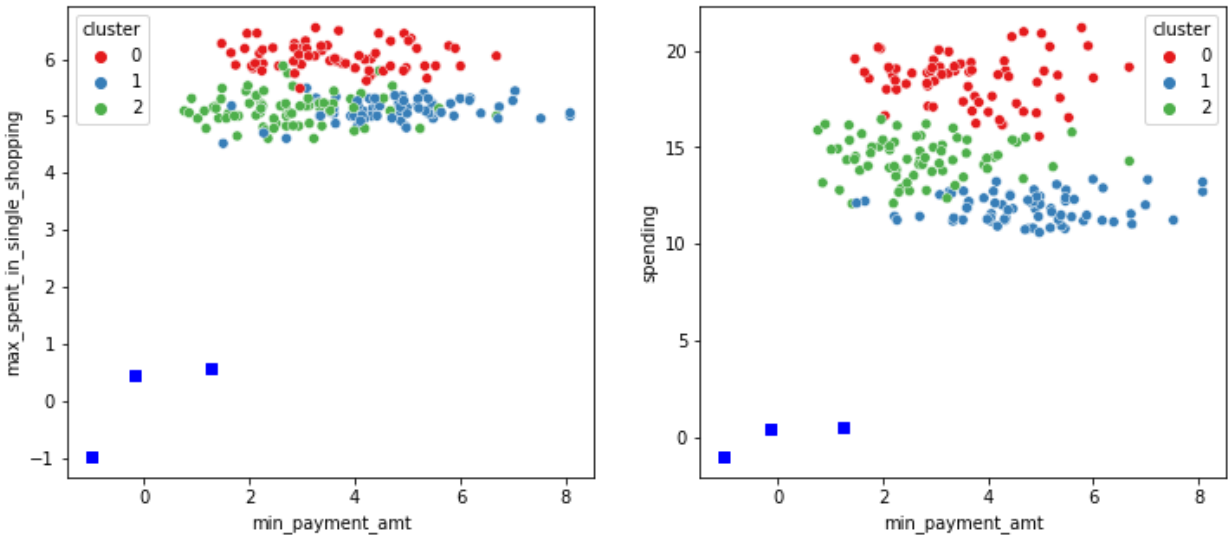


Fig 32: Minimum payment amount against maximum spent in single shopping and spending.

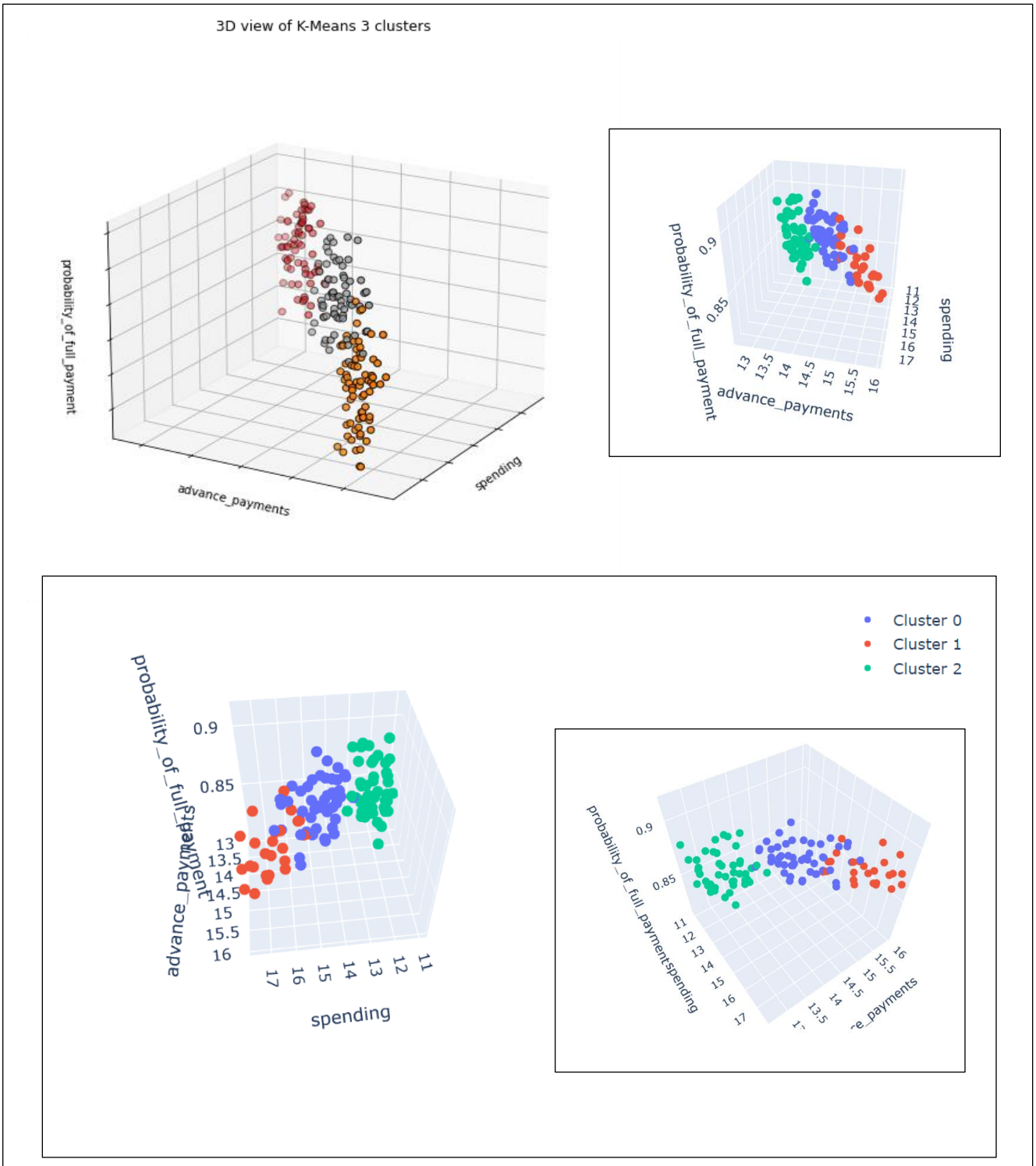


Fig 33,34,35,36: 3-D views of K-means 3 clusters

Conclusion

After executing various methods, we choose Kmeans, since the data does not include different small groups but group that are very similar to each other. For record sake, we did try out density-based algorithms but, one, it was out of the scope of this project, and two, we did not prefer to use those based on the results. Those algorithms, though, can help us seek out extreme customers in credit card fraud. This study aims customer segmentation by using customer behaviour.

Compering 2 different Kmeans models showed that we have a better understanding of customer segmentation by using the 3-cluster model.

Some of the outstanding results:

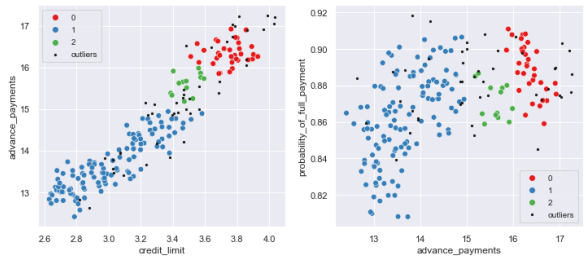
Cluster 1: This customer group indicates a small group of customers who are high spenders with high advance payment. We can also assume that this customer segment uses its credit card as a loan. They may want the bank to increase their credit limits so that they are able to keep their spending habits.

Cluster 2: These customers purchase frequently but use their credit cards for a small number of purchases.

Cluster 3: This segment points out new or low-spending customers with a lower credit limit and current balance. It is a similar customer segment as cluster 2 but with a lower average spending.

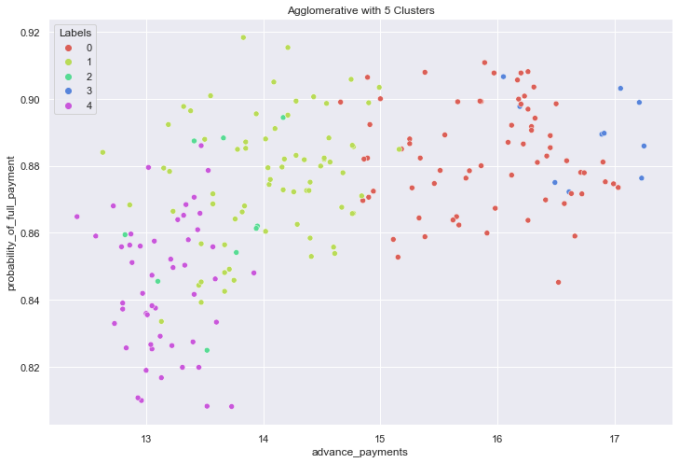
K-means in race against other clustering algorithms

Small experiments with data, for bonus points 😊

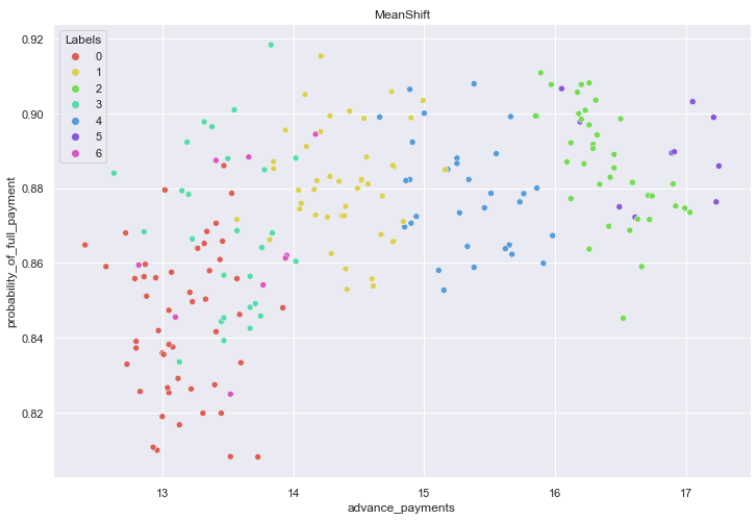


DBSCAN created 3 clusters plus the outliers cluster (-1).

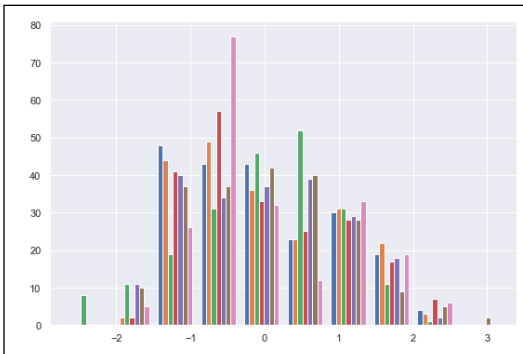
Sizes of clusters 0-2 vary significantly - one has 13 observations, while another has 118. There are 45 outliers.



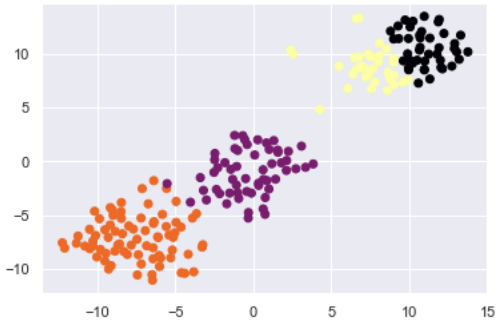
Agglomerative clustering picked 5 clusters



Mean Shift created 7 clusters



Spectral clustering: 7 clusters, mainly



T-SNE

And the winner is…

	Clustering Method	Silhouette Score
0	KMeans	0.334485
2	Hierarchical	0.275231
1	GMM	0.233004
3	Spectral	0.164301
4	DBSCAN	0.154762

Since this was an experiment and not part of the project, it was done with K-means 4 and K-means 5. DBSCAN works best on large data. This page is just for fun.

Problem 2: CART-RF-ANN

Editorial summary

An Insurance firm providing tour insurance is facing higher claim frequency. The management decides to collect data from the past few years. We are assigned to make a model that predicts the claim status and provides the management with recommendations. We are asked to use CART, RF & ANN, and compare the models' performances in train and test sets.

Introduction

Even before Covid-19, countries such as Britain had a staggering 10 times increase in customer inquiries, claims and complaints relating to their travel insurance. Whether it's about travel insurance, critical illness, health cover, business interruption, or another issue, customers are deluging insurers with queries over what they may or may not be covered for or to actually make a claim. An insurance claim is a formal request to an insurance company asking for a payment based on the terms of the insurance policy. The insurance company reviews the claim for its validity and then pays out to the insured or requesting party (on behalf of the insured) once approved. One of the biggest challenges is that all of this is happening at the same time as a huge spike in customer contact.

This is a predictive analysis problem under classification and CART (classification and regression trees), Random Forest, and Artificial Neural Network are all classification techniques.

Decision tree is a non-parametric supervised machine learning algorithm which is used mostly for classification and regression problems. In this scenario, we split the population or sample into two or more homogeneous sets (or subpopulations) based on the most significant splitter/differentiator in input variables.

If there is high non-linearity and complex relationship between dependent and independent variables, a tree-based model will outperform a classical regression method. And yet, if we need to build a model which is easy to explain to people, a decision tree model will always do better and be simpler to interpret than linear regression.

Decision tree can automatically handle missing values.

Decision tree is usually robust to outlier and can handle them automatically.

Training period is less as compared to Random Forest because it generates only one tree unlike forest of trees in the Random Forest.

Disadvantages of Decision Tree

Overfitting is the main problem of decision tree. It generally leads to overfitting of the data which ultimately leads to wrong prediction.

Due to the overfitting, there are very high chances of high variance in the output which leads to many errors in the final estimation and shows high in accuracy in the results.

Adding a new datapoint can lead to re-generation of the overall tree and all

nodes need to be recalculated and recreated.

Decision tree is not suitable for large datasets. If the dataset is large, then one single tree may grow complex and lead to overfit. So, in that case, we should use random forest instead of a single decision tree.

Important terminology related to Decision Tree

Root Node: It represents the entire population or sample and this further gets into two or more homogeneous sets.

Splitting: It is a process of dividing a node into two or more sub nodes.

Decision Node: When a sub-node splits into further sub nodes, then it is called the decision tree.

Leaf/Terminal Node: Nodes do not split is called leaf or terminal node.

Pruning: When we remove sub-nodes of a decision node, this process is called pruning.

Branch/Sub-Tree: A subsection of the entire tree is called branch/sub tree.

Parent and Child Node: A node, which is divided into sub nodes is called a parent node of sub nodes whereas sub nodes are the child of a parent node.

Then we come to Artificial Neural Network, which is a Machine Learning algorithm that is modelled roughly around what we know currently about how the human brain functions. ANN Models the relationship between a set of input signals and an output, similar to a biological brain response to stimuli from sensory inputs.

The brain uses a network of interconnected cells called neurons to provide learning capability, while ANN uses a network of artificial neurons or nodes to solve challenging learning problems. ANNs have the ability to learn and figure out how to perform their function on their own.

ANNs determine their function based only on sample inputs. ANN also has the ability to generalize and produce outputs for inputs it has not been taught how to deal with. Besides, it can be easily retrained to suit a new environment.

The problem at hand:- The objective will be to try out the three models and check which one has the strongest predictive power so as to call most correctly if a customer will file for claim.

Attribute Information:

1. Target: Claim Status (Claimed)

2. Code of tour firm (Agency_Code)

3. Type of tour insurance firms (Type)

4. Distribution channel of tour insurance agencies (Channel)

5. Name of the tour insurance products (Product)

6. Duration of the tour (Duration)

7. Destination of the tour (Destination)

8. Amount of sales of tour insurance policies (Sales)

9. The commission received for tour insurance firm (Commission)

10. Age of insured (Age)

** In the tour insurance dataset, the variable 'commission' is misspelt but not so grossly as to be misunderstood.

6 | Categorial variables
3,000 | Records, none missing
9 | Independant variables
1 | Target variable (Clamied)

	Age	Agency_Code	Type	Claimed	Commision	Channel	Duration	Sales	Product Name	Destination
0	48	C2B	Airlines	No	0.70	Online	7	2.51	Customised Plan	ASIA
1	36	EPX	Travel Agency	No	0.00	Online	34	20.00	Customised Plan	ASIA
2	39	CWT	Travel Agency	No	5.94	Online	3	9.90	Customised Plan	Americas
3	36	EPX	Travel Agency	No	0.00	Online	4	26.00	Cancellation Plan	ASIA
4	33	JZI	Airlines	No	6.30	Online	53	18.00	Bronze Plan	ASIA

Tab 1: The dataset

2.1 Data Ingestion: Read the dataset. Do the descriptive statistics and do null value condition check, write an inference on it.

Data above, details below

```
3,000 entries, 0 to 2,999
Total 10 columns
```

```
 #   Column         Non-Null Count  Dtype
---  ------         --------------  -----
 0   Age            3000 non-null   int64
 1   Agency_Code    3000 non-null   object
 2   Type           3000 non-null   object
 3   Claimed        3000 non-null   object
 4   Commision      3000 non-null   float64
 5   Channel        3000 non-null   object
 6   Duration       3000 non-null   int64
 7   Sales          3000 non-null   float64
 8   Product Name   3000 non-null   object
 9   Destination    3000 non-null   object
dtypes: float64(2), int64(2), object(6)
memory usage: 234.5+ KB
```

Tab 2: Data info

Observations on data

10 | Variables
4 | Numeric variables
(Age, Commision, Duration, Sales)

Check for missing values in any column

```
Age             0
Agency_Code     0
Type            0
Claimed         0
Commision       0
Channel         0
Duration        0
Sales           0
Product Name    0
Destination     0
dtype: int64
```

Observation on data: No missing value

Descriptive Statistics Summary

	count	mean	std	min	25%	50%	75%	max
Age	3000.0	38.091000	10.463518	8.0	32.0	36.00	42.000	84.00
Commision	3000.0	14.529203	25.481455	0.0	0.0	4.63	17.235	210.21
Duration	3000.0	70.001333	134.053313	-1.0	11.0	26.50	63.000	4580.00
Sales	3000.0	60.249913	70.733954	0.0	20.0	33.00	69.000	539.00

Tab 3: Descriptive stats

Initial descriptive analysis

	count	mean	std	min	25%	50%	75%	90%	max
Age	3000.0	38.091000	10.463518	8.0	32.0	36.00	42.000	53.000	84.00
Commision	3000.0	14.529203	25.481455	0.0	0.0	4.63	17.235	48.300	210.21
Duration	3000.0	70.001333	134.053313	-1.0	11.0	26.50	63.000	224.200	4580.00
Sales	3000.0	60.249913	70.733954	0.0	20.0	33.00	69.000	172.025	539.00

Tab 4: Initial descriptive analysis

Observations

Duration has negative value, while it's not possible, so it must be a wrong entry.

Age	Agency_Code	Type	Claimed	Commision	Channel	Duration	Sales	Product Name	Destination
48	C2B	Airlines	No	0.70	Online	7	2.51	Customised Plan	ASIA
36	EPX	Travel Agency	No	0.00	Online	34	20.00	Customised Plan	ASIA
39	CWT	Travel Agency	No	5.94	Online	3	9.90	Customised Plan	Americas
36	EPX	Travel Agency	No	0.00	Online	4	26.00	Cancellation Plan	ASIA
33	JZI	Airlines	No	6.30	Online	53	18.00	Bronze Plan	ASIA
45	JZI	Airlines	Yes	15.75	Online	8	45.00	Bronze Plan	ASIA
61	CWT	Travel Agency	No	35.64	Online	30	59.40	Customised Plan	Americas
36	EPX	Travel Agency	No	0.00	Online	16	80.00	Cancellation Plan	ASIA
36	EPX	Travel Agency	No	0.00	Online	19	14.00	Cancellation Plan	ASIA
36	EPX	Travel Agency	No	0.00	Online	42	43.00	Cancellation Plan	ASIA

Tab 4: First 10 rows

For sales and commision (which we think is commission misspelt), the mean and the median vary a lot.

Agency code EPX has a frequency of 1365.

The most preferred type seems to be travel agency. Customized plan is the most sought plan by customers

Asia seems to be most sought-after of all destinations.

We will further look at the distribution of dataset in the univariate and bivariate analysis.

```
First 10 records
```

Observation: All fine.

	Age	Agency_Code	Type	Claimed	Commision	Channel	Duration	Sales	Product Name	Destination
2990	51	EPX	Travel Agency	No	0.00	Online	2	20.00	Customised Plan	ASIA
2991	29	C2B	Airlines	Yes	48.30	Online	381	193.20	Silver Plan	ASIA
2992	28	CWT	Travel Agency	No	11.88	Online	389	19.80	Customised Plan	ASIA
2993	36	EPX	Travel Agency	No	0.00	Online	234	10.00	Cancellation Plan	ASIA
2994	27	C2B	Airlines	Yes	71.85	Online	416	287.40	Gold Plan	ASIA
2995	28	CWT	Travel Agency	Yes	166.53	Online	364	256.20	Gold Plan	Americas
2996	35	C2B	Airlines	No	13.50	Online	5	54.00	Gold Plan	ASIA
2997	36	EPX	Travel Agency	No	0.00	Online	54	28.00	Customised Plan	ASIA
2998	34	C2B	Airlines	Yes	7.64	Online	39	30.55	Bronze Plan	ASIA
2999	47	JZI	Airlines	No	11.55	Online	15	33.00	Bronze Plan	ASIA

Tab 5: Last 10 rows

```
Last 10 records
```

Observation: All fine here as well.

```
Summary of all the variables
```

Observation

```
Data dimensions
(3000, 10)

3,000 rows and 10 columns
```

	count	unique	top	freq	mean	std	min	25%	50%	75%	max
Age	3000.0	NaN	NaN	NaN	38.091	10.463518	8.0	32.0	36.0	42.0	84.0
Agency_Code	3000	4	EPX	1365	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Type	3000	2	Travel Agency	1837	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Claimed	3000	2	No	2076	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Commision	3000.0	NaN	NaN	NaN	14.529203	25.481455	0.0	0.0	4.63	17.235	210.21
Channel	3000	2	Online	2954	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Duration	3000.0	NaN	NaN	NaN	70.001333	134.053313	-1.0	11.0	26.5	63.0	4580.0
Sales	3000.0	NaN	NaN	NaN	60.249913	70.733954	0.0	20.0	33.0	69.0	539.0
Product Name	3000	5	Customised Plan	1136	NaN	NaN	NaN	NaN	NaN	NaN	NaN
Destination	3000	3	ASIA	2465	NaN	NaN	NaN	NaN	NaN	NaN	NaN

Tab 6: Summary of all the variables

For categorial code variables, maximun unique count is 5 (for Product Name)

Getting unique counts for all the categorical variables

```
AGENCY_CODE :  4
JZI      239
CWT      472
C2B      924
EPX     1365
Name: Agency_Code, dtype: int64


TYPE :  2
Airlines         1163
Travel Agency    1837
Name: Type, dtype: int64

CLAIMED :  2
Yes      924
No      2076
Name: Claimed, dtype: int64

CHANNEL :  2
Offline      46
Online     2954
Name: Channel, dtype: int64

PRODUCT NAME :  5
Gold Plan             109
Silver Plan           427
Bronze Plan           650
Cancellation Plan     678
Customised Plan      1136
Name: Product Name, dtype: int64

DESTINATION :  3
EUROPE            215
Americas          320
ASIA             2465
Name: Destination, dtype: int64
```

Check for duplicate data

```
Number of duplicate rows = 139
```

	Age	Agency_Code	Type	Claimed	Commision	Channel	Duration	Sales	Product Name	Destination
63	30	C2B	Airlines	Yes	15.0	Online	27	60.0	Bronze Plan	ASIA
329	36	EPX	Travel Agency	No	0.0	Online	5	20.0	Customised Plan	ASIA
407	36	EPX	Travel Agency	No	0.0	Online	11	19.0	Cancellation Plan	ASIA
411	35	EPX	Travel Agency	No	0.0	Online	2	20.0	Customised Plan	ASIA
422	36	EPX	Travel Agency	No	0.0	Online	5	20.0	Customised Plan	ASIA
...	...	...	...	...	...	...	...	...	...	...
2940	36	EPX	Travel Agency	No	0.0	Online	8	10.0	Cancellation Plan	ASIA

Tab 6: Duplicated rows

Not removing duplicates and here's why

Even though 139 records are suggested to be duplicate, these can be of different customers, since there is no customer ID or any unique identifier. So, we choose not to drop these.

Univariate Analysis

Age

```
Range of values:  76
```

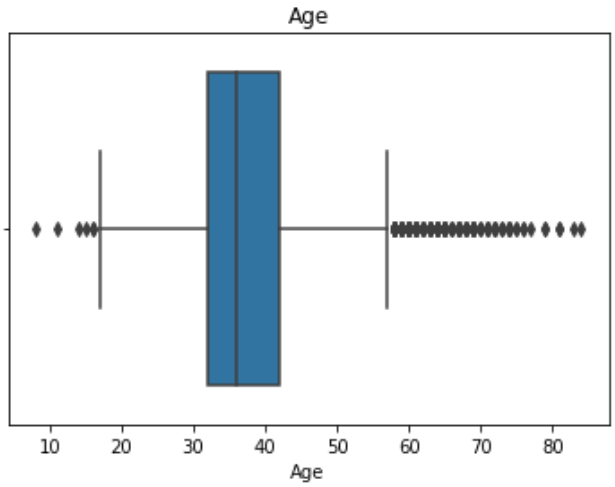


Fig 1: Boxplot for age

```
#Central values

Minimum Age:  8
Maximum Age:  84
Mean value:  38.091
Median value:  36.0
Standard deviation:  10.463518245377944
Null values:  False
```

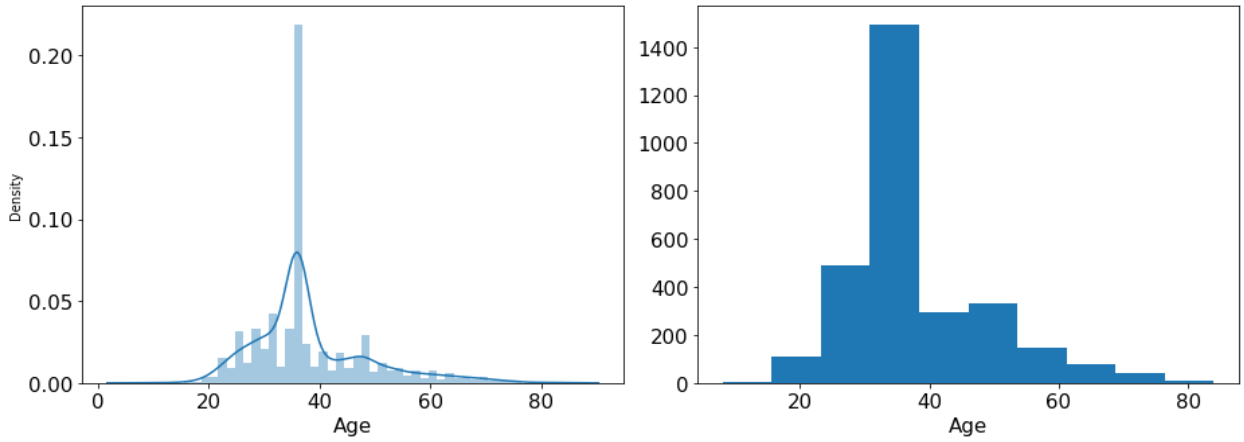


Fig 2: Distribution plot and histogram for age

```
#Quartiles

Age - 1st Quartile (Q1) is:  32.0
Age - 3st Quartile (Q3) is:  42.0
Interquartile range (IQR) of Age is
10.0

#Outlier detection from Interquarti
le range (IQR) in original data

Lower outliers in Age:  17.0
Upper outliers in Age:  57.0

Number of outliers in Age upper:  1
98
Number of outliers in Age lower:  6
% of Outlier in Age upper:  7 %
% of Outlier in Age lower:  0 %
```

Interpreting the graphs

The box plot of the age variable shows outliers. Spending is positively skewed - 1.149713 The distplot shows the distribution of data from 20 to 80. In the range of 30 to 40 is where the majority of the distribution lies.

Commission variable

```
Range of values:  210.21
```

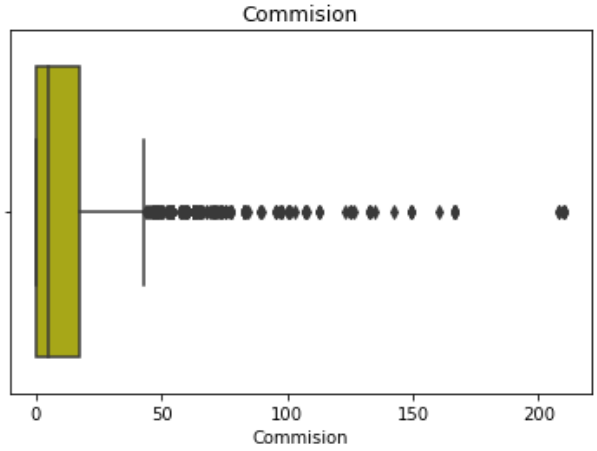


Fig 3: Boxplot for commission

```
#Central values

Minimum Commission:  0.0
Maximum Commission:  210.21
Mean value:  14.529203333333266
Median value:  4.63
Standard deviation:  25.48145450662
553
Null values:  False
```

```
#Quartiles

Commision - 1st Quartile (Q1) is:
0.0
Commision - 3st Quartile (Q3) is:
17.235
Interquartile range (IQR) of Commis
ion is 17.235

#Outlier detection from Interquarti
le range (IQR) in original data


Lower outliers in Commission:  -25.
8525
Upper outliers in Commision:  43.08
75

Number of outliers in Commission up
per :  362
Number of outliers in Commission lo
wer :  0
% of Outlier in Commission upper:
12 %
% of Outlier in Commission lower:
0 %
```

Interpreting the graphs

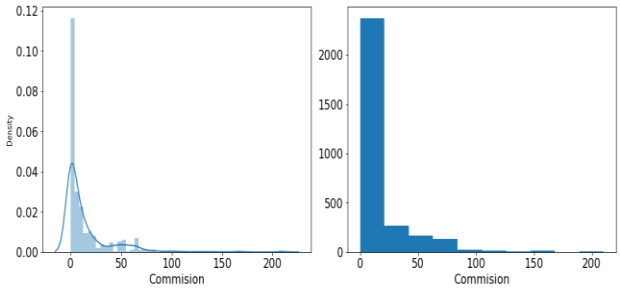


Fig 4: Distribution plot and histogram for commission

The boxplot of the commission variable shows outliers. Spending is positively skewed - 3.148858. The distplot shows the distribution of data from 0 to 30.

Duration

```
Range of values:  4,581
```

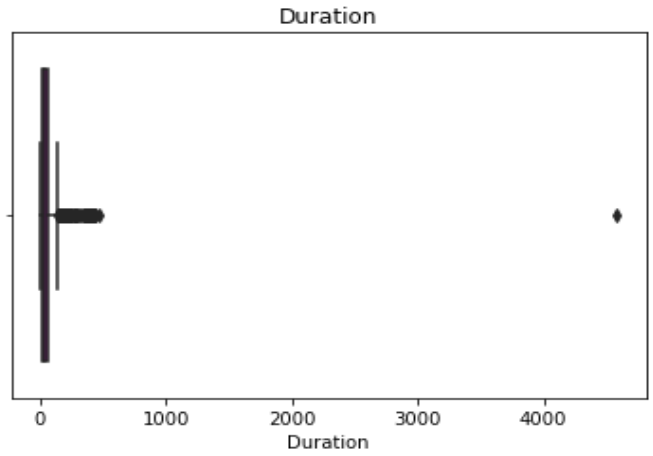


Fig 5: Boxplot for Duration

```
#Central values
Minimum Duration:  -1
Maximum Duration:  4580
Mean value:  70.00133333333333
Median value:  26.5
Standard deviation:  134.0533131325
3495
Null values:  False

# Quartiles

Duration - 1st Quartile (Q1) is:  1
1.0
Duration - 3st Quartile (Q3) is:  6
3.0
Interquartile range (IQR) of
Duration is  52.0

#Outlier detection from Interquarti
le range (IQR) in original data)

Lower outliers in Duration:  -67.0
Upper outliers in Duration:  141.0


Number of outliers in Duration
upper:  382
```

```
Number of outliers in Duration
lower:  0


Outliers in Duration upper: 13 %
Outliers in Duration lower:  0 %
```

Interpreting the graphs

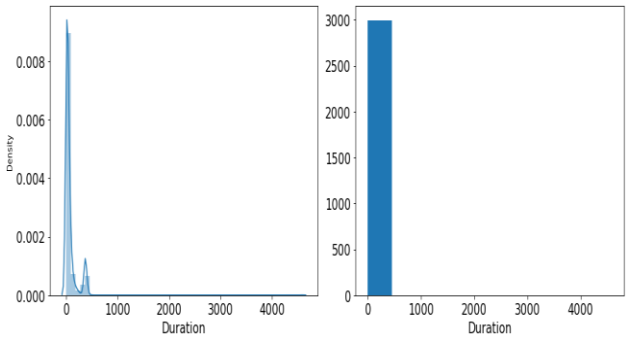


Fig 6: Distribution plot and histogram for duration

The box plot of the duration variable shows outliers. Spending is positively skewed - 13.784681 The dist plot shows the distribution of data from 0 to 100

Sales

```
Range of values:  539.0
```

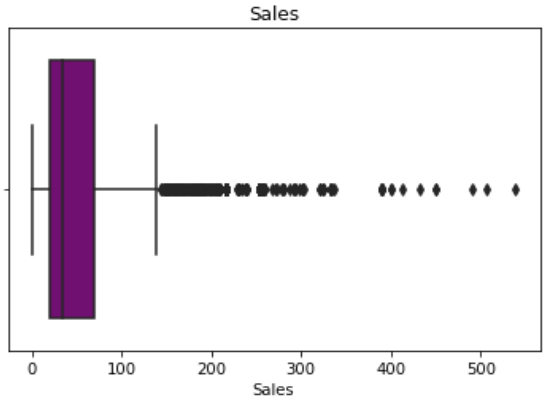


Fig 7: Boxplot for sales

```
#Central values
Minimum Sales:  0.0
Maximum Sales:  539.0
Mean value:  60.24991333333344
Median value:  33.0
Standard deviation:  70.73395353143
047
Null values:  False

#Quartiles
Sales - 1st Quartile (Q1) is:  20.0
Sales - 3st Quartile (Q3) is:  69.0
Interquartile range (IQR) of Sales
is  49.0

#Outlier detection from
Interquartile range (IQR) in
original data # IQR=Q3-Q1


L_outliers=Q1-1.5*(Q3-Q1)
U_outliers=Q3+1.5*(Q3-Q1)
Lower outliers in Sales:  -53.5
Upper outliers in Sales:  142.5

Number of outliers in Sales upper:
353
Number of outliers in Sales lower:
0
% of Outlier in Sales upper:  12 %
% of Outlier in Sales lower:  0 %
```

Interpreting the graphs

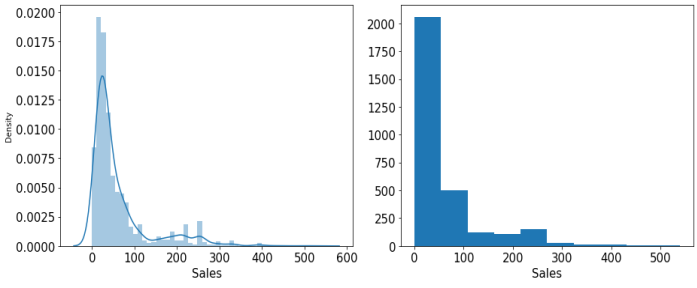


Fig 8: Distribution plot and histogram for sales

The boxplot of the sales variable shows outliers. Sales is positively skewed - 2.381148. The distplot shows the distribution of data from 0 to 300.

Observation on outliers

There are outliers in all the variables but the sales and commission figures can be genuine. Besides, Random Forest and CART can handle the outliers. Outliers are also too many for us to be comfortable removing them without consulting the business. Hence, we will keep the data as it is and treat the outliers for the ANN model to compare the same after all the steps are done.

Categorical variables

Agency_Code

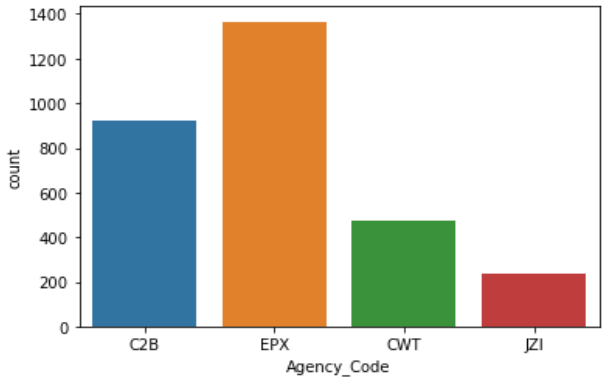


Fig 9: Countplot for agency code

```
# Boxplot
```

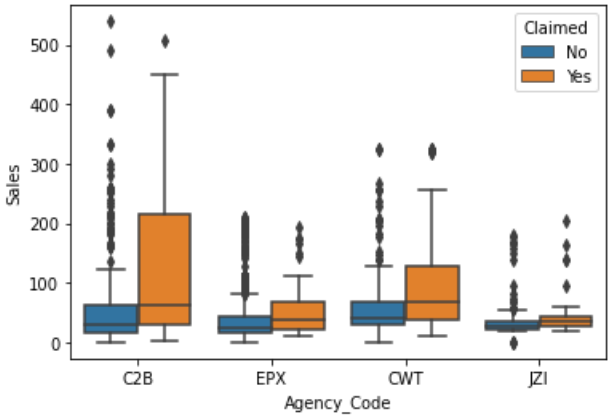


Fig 10: Boxplot for agency code

```
# Swarmpot
```

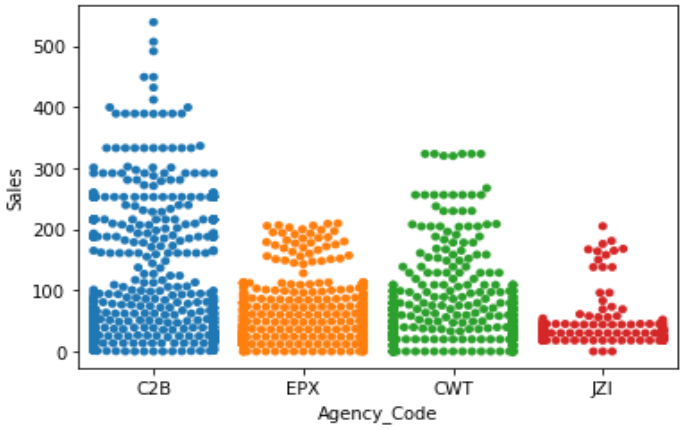


Fig 11: Swarmplot for agency code

```
# Combined Violin and Swarm plot
```

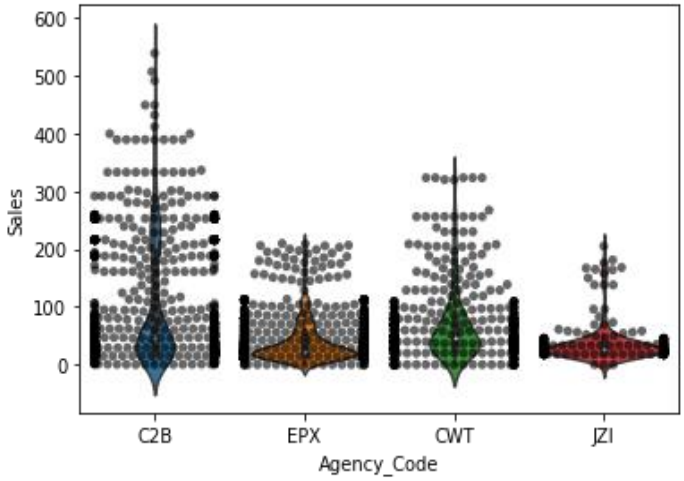


Fig 12: Combined swarm-and-violin plot for agency code

The distribution of the agency code, shows us EPX with maximum frequency. The boxplot shows the split of sales for different agencies and a comparison of their yes and no claims. The boxplot shows that C2B has more claims for any agency. The bimodal nature of the

distributions is much clearer in the swarmplot and their shape can almost be made out. Also, it plots all the data points. However, the thickness of the plotted points causes the formation of the strands extending out from each swarm, which make judging the true shape of the distributions difficult, so we combine it with the violin plot, which also tells us the density.

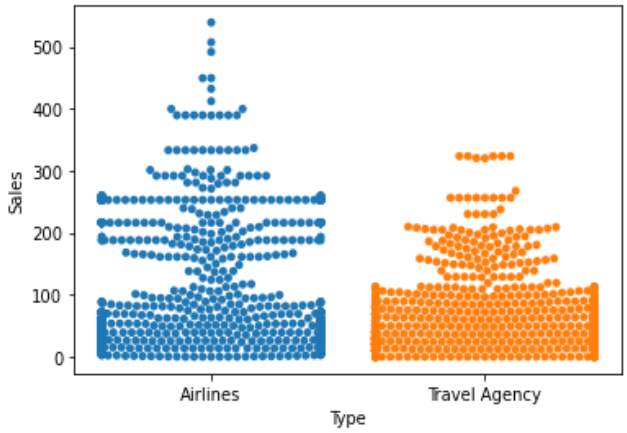


Fig 15: Swarmplot for type

Type

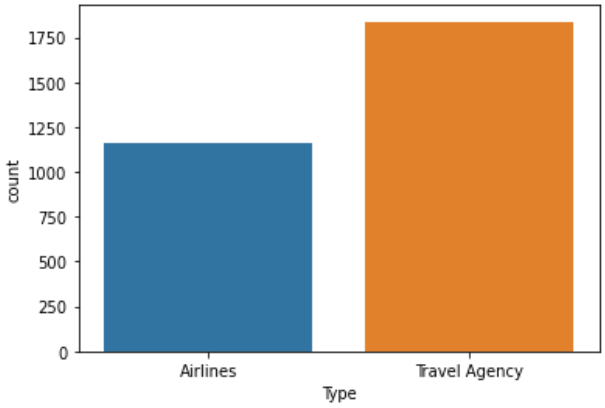


Fig 13: Countplot for type

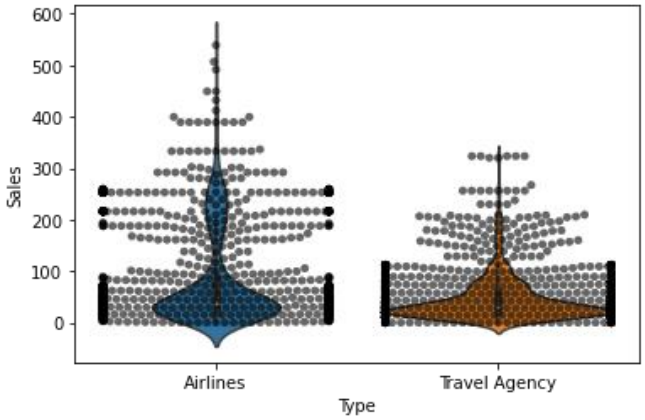


Fig 16: Combined swarm-and-violin plot for type

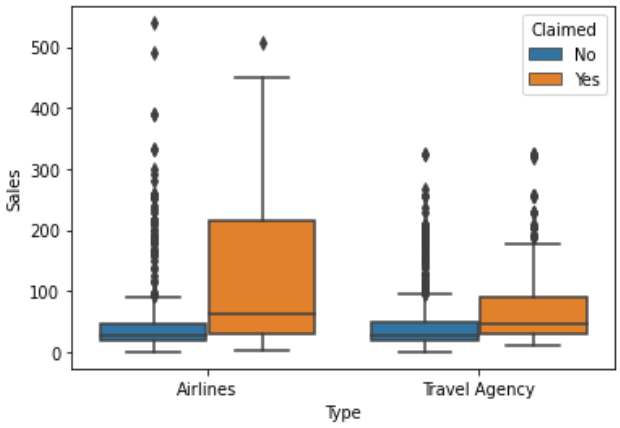


Fig 14: Boxplot for type

Even though fewer in numbers, the airlines seem to be outcompeting the travel agencies in sales, but they also have more claims. Travel agencies have more density, a fatter bulge at the bottom. Airlines seem to be making money at the top.

Channel

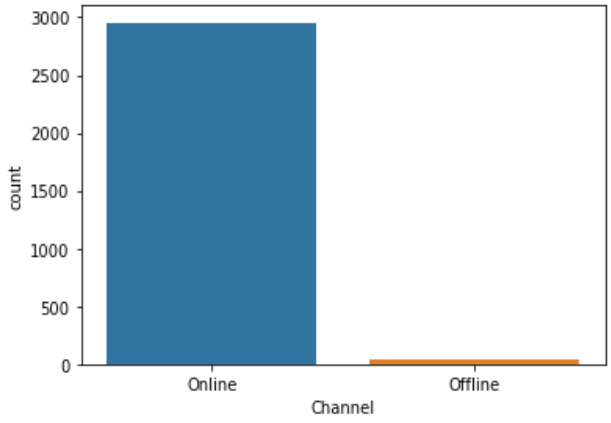


Fig 17: Countplot for channel

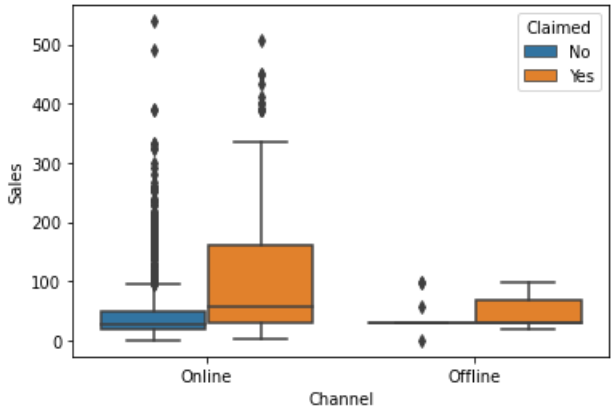


Fig 18: Boxplot for channel

By a huge majority, the customers have chosen the online over the offline channel for booking, but the online channel also has more claims. It shows how agencies have improved their booking portals and made it easier for the customer.

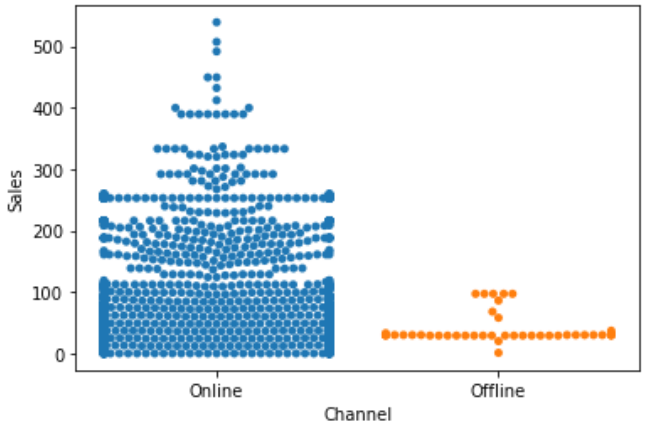


Fig 19: Swarmplot for channel

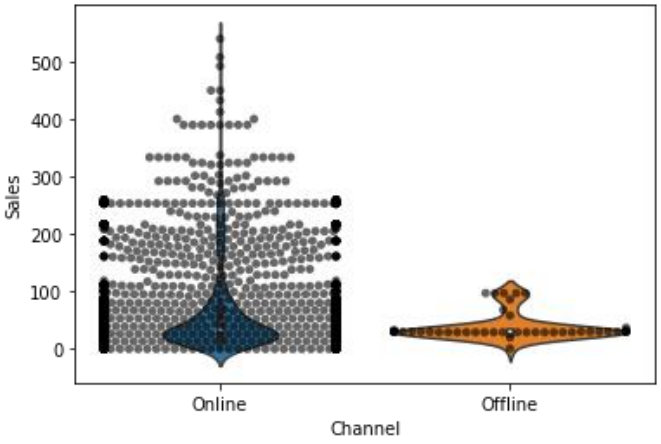
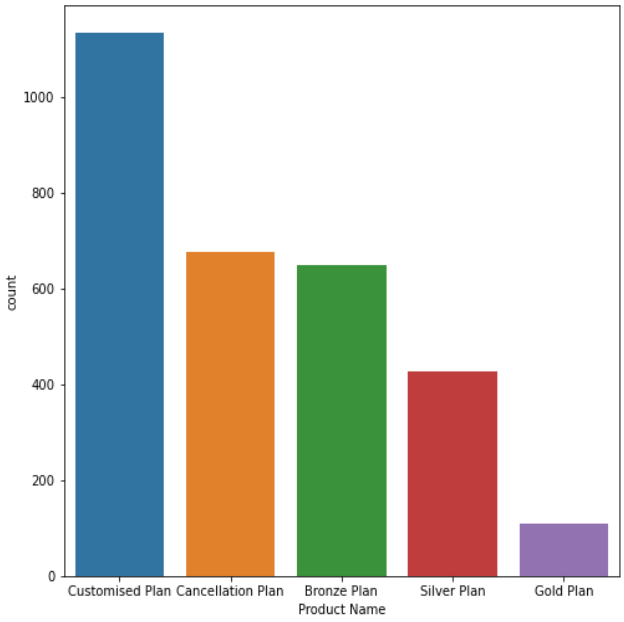
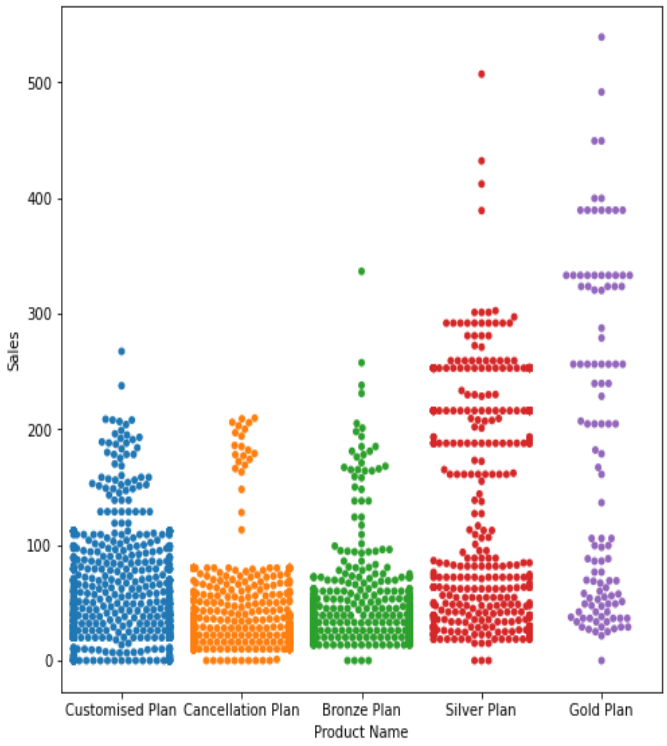
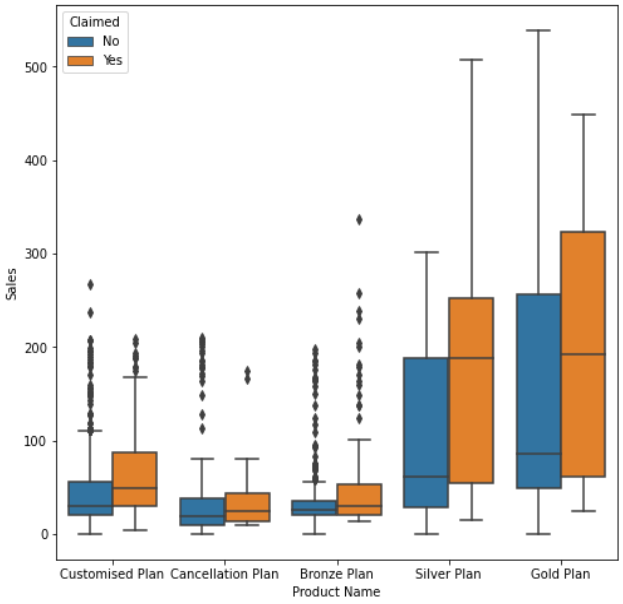


Fig 20: Swarm-and-violin plot for channel

Product Name







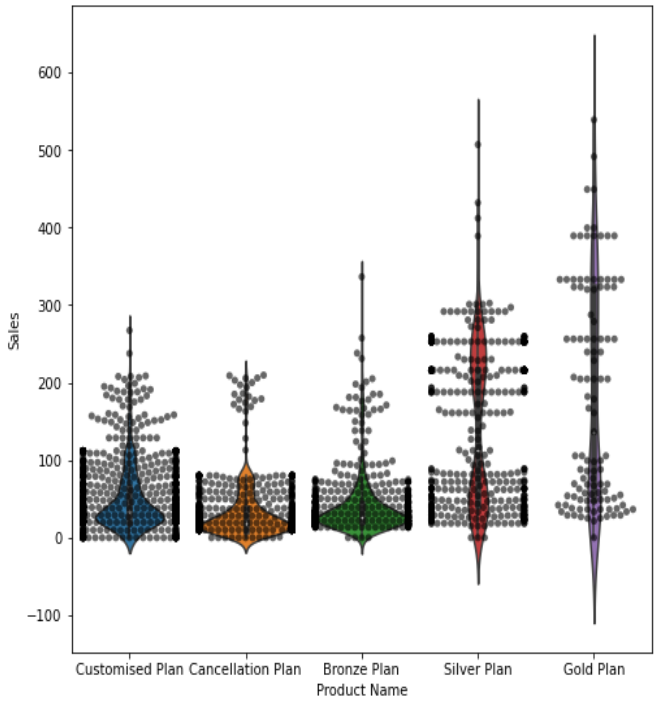
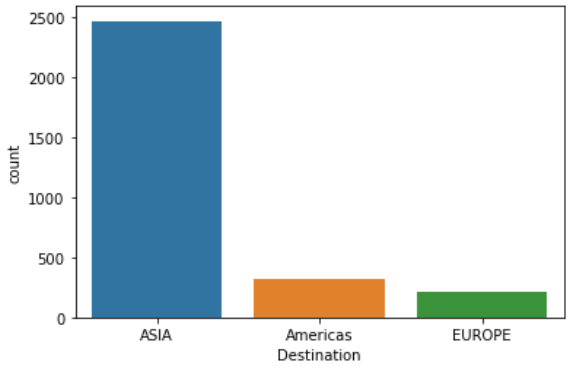
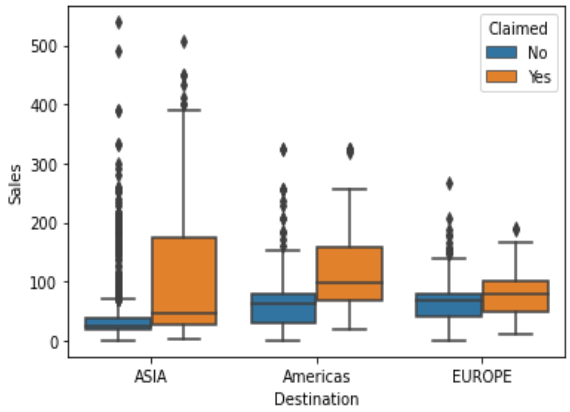


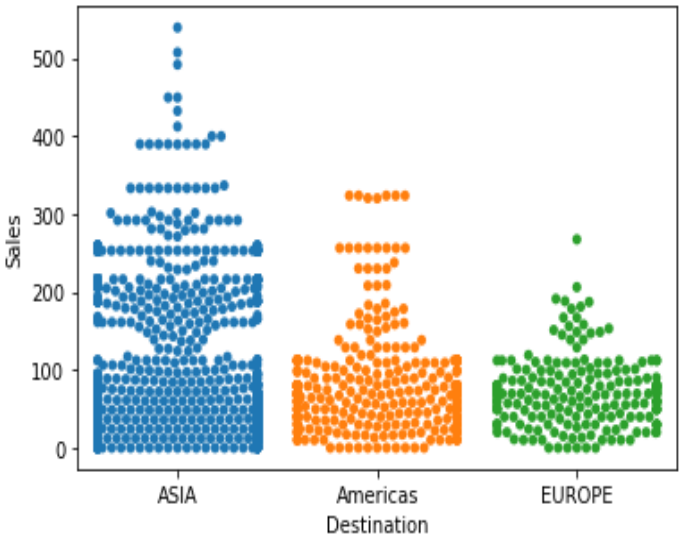
Fig 21, 22, 23, 24: Countplot, boxplot, swarmplot, and swarm-and-violin plot for channel

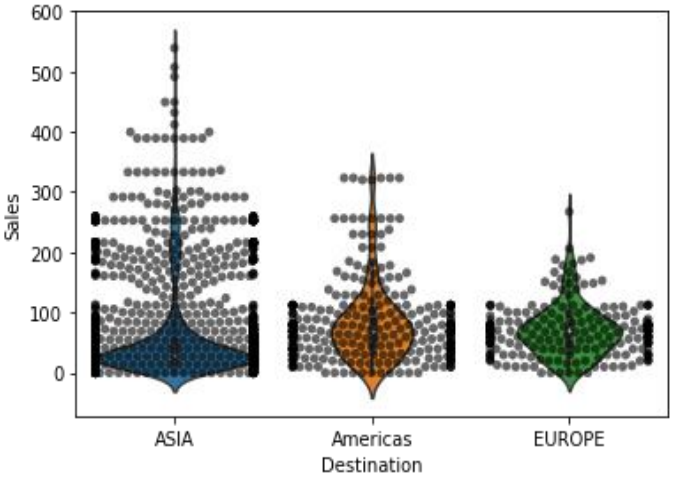
Customized plan seems to be the most liked by the customers and the most rewarding for the agencies, while both yes and no claims are the highest for the gold plan. This plan has also accounted for most sales, keeping the cash registers ringing.

Destination



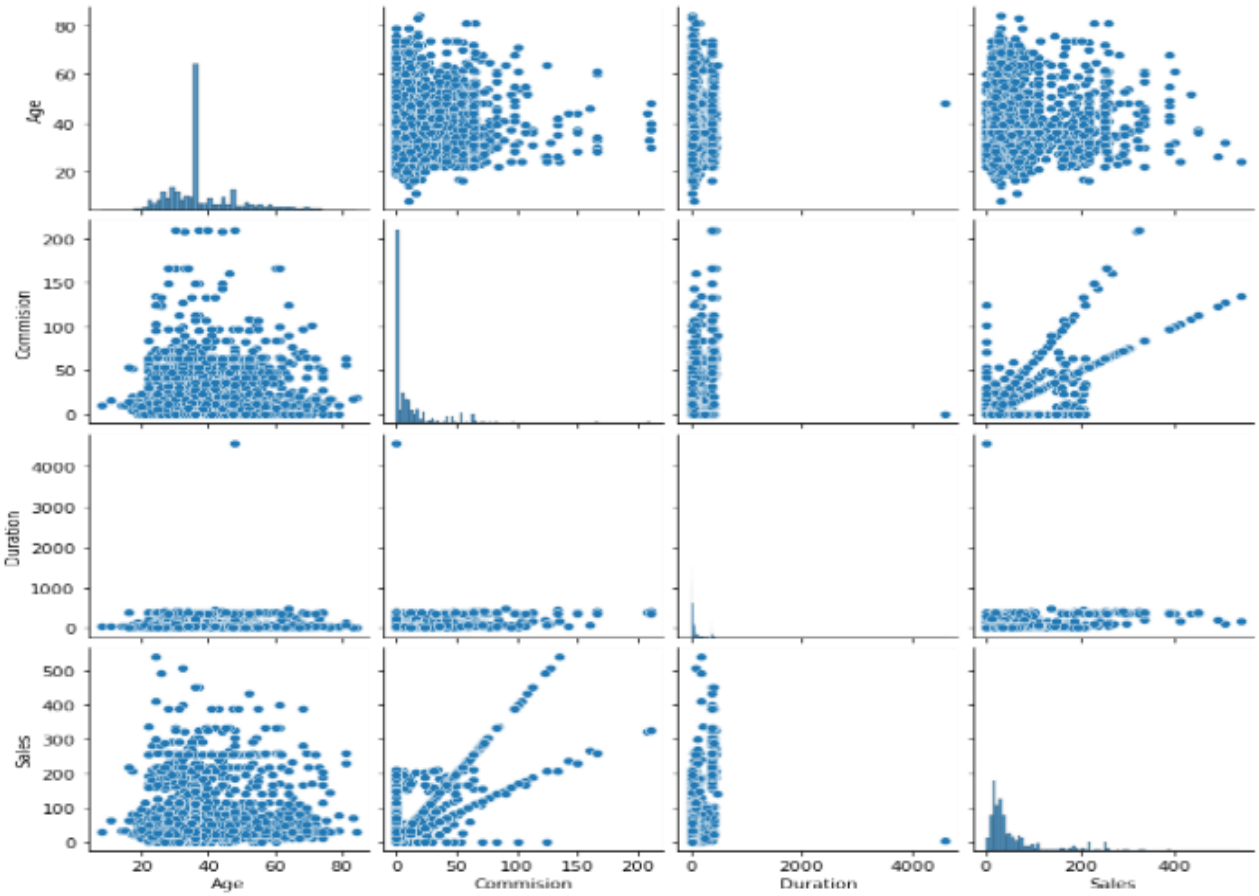






Most popular destination? Not developed Americas or Europe but a culturally richer and tropically warmer Asia, but it also accounts for most claims. Maybe the destination is good but the customer experience is not or maybe one-size-fit-all packages don't work. A local awareness programme can help encourage tour and travel.

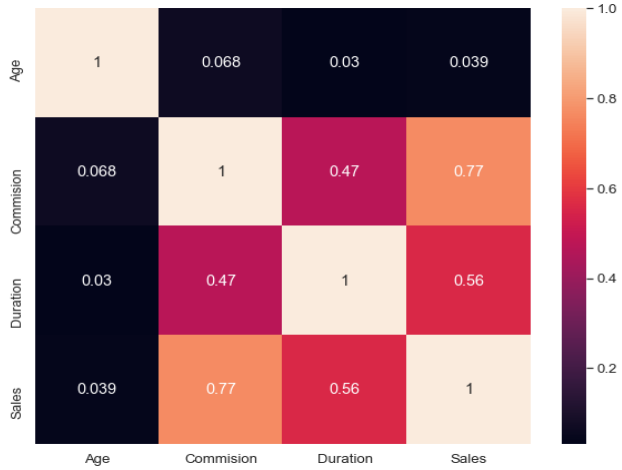
Fig 25, 26, 27, 28: Countplot, boxplot, swarmplot, and swarm-and-violin plot for destination



Checking pairwise distribution of the continuous variables

Not much of multi collinearity observed
No negative correlation. Only positive correlations seen.

	Feature 1	Feature 2	Correlation Coefficient
7	Claimed	Claimed	1.000000
21	Claimed	Commision	0.294172
23	Claimed	Sales	0.262421
27	Claimed	Product Name	0.197263
34	Claimed	Duration	0.146877
67	Claimed	Channel	-0.013646
77	Claimed	Destination	-0.038362
80	Claimed	Age	-0.062749
90	Claimed	Type	-0.328268
94	Claimed	Agency_Code	-0.386112



Figs 29 and 30: Pairplot and heatmap

Checking for Correlations

A heatmap with only continuous variables and checking correlation coefficients against 'claimed' shows that it correlates the most with commission and sales.

Converting all objects to categorical codes

```
feature: Agency_Code
['C2B', 'EPX', 'CWT', 'JZI']
Categories (4, object): ['C2B', 'CW
T', 'EPX', 'JZI']
[0 2 1 3]


feature: Type
['Airlines', 'Travel Agency']
Categories (2, object): ['Airlines'
, 'Travel Agency']
[0 1]


feature: Claimed
['No', 'Yes']
Categories (2, object): ['No', 'Yes
']
[0 1]


feature: Channel
['Online', 'Offline']
Categories (2, object): ['Offline',
'Online']
[1 0]


feature: Product Name
['Customised Plan', 'Cancellation P
lan', 'Bronze Plan', 'Silver Plan',
'Gold Plan']
Categories (5, object): ['Bronze Pl
an', 'Cancellation Plan', 'Customis
ed Plan', 'Gold Plan', 'Silver Plan
']
[2 1 0 4 3]


feature: Destination
['ASIA', 'Americas', 'EUROPE']
Categories (3, object): ['ASIA', 'A
mericas', 'EUROPE']
[0 1 2]
```

```
 #   Column         Dtype
---  ------         ------
 0   Age            int64
 1   Agency_Code    int8
 2   Type           int8
 3   Claimed        int8
 4   Commision      float64
 5   Channel        int8
 6   Duration       int64
 7   Sales          float64
 8   Product Name   int8
 9   Destination    int8
```

Age	Agency_Code	Type	Claimed	Commision
48	0	0	0	0.70
36	2	1	0	0.00
39	1	1	0	5.94
36	2	1	0	0.00
33	3	0	0	6.30

Channel	Duration	Sales	Product Name	Destination
1	7	2.51	2	0
1	34	20.00	2	0
1	3	9.90	2	1
1	4	26.00	1	0
1	53	18.00	0	0

Tab 7: Data converted and coded

Observation

Data types converted and coded successfully

Proportion of 1s and 0s

```
0    0.692
1    0.308
```

Seems a balanced dataset

2.2 Data Split: Split the data into test and train, build classification model CART, Random Forest, Artificial Neural Network

Extracting the target column into separate vectors for training set and test set

Age	Agency_Code	Type	Commision	Channel
48	0	0	0.70	1
36	2	1	0.00	1
39	1	1	5.94	1
36	2	1	0.00	1
33	3	0	6.30	1

Duration	Sales	Product Name	Destination
7	2.51	2	0
34	20.00	2	0
3	9.90	2	1
4	26.00	1	0
53	18.00	0	0

Tab 8: The dataset after extracting target column 'claimed'

```
# Prior to scaling
```

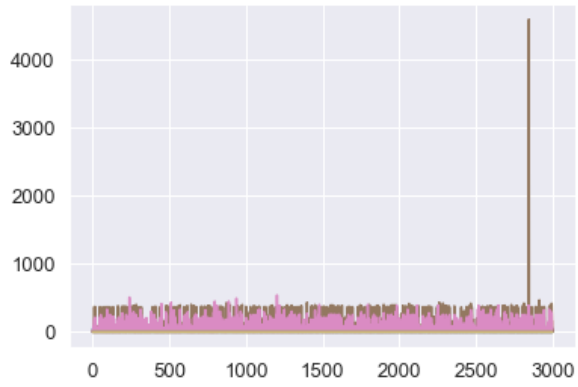


Fig 31: The dataset before scaling

```
# Scaling the attributes.
```

The data has a few variables on different scales, so it become necessary to scale the data. We used z-scaling, because the variance between the columns is more or less the same.

Age	Agency_Code	Type	Commision	Channel
0.947162	-1.314358	-1.256796	-0.542807	0.124788
-0.199870	0.697928	0.795674	-0.570282	0.124788
0.086888	-0.308215	0.795674	-0.337133	0.124788
-0.199870	0.697928	0.795674	-0.570282	0.124788
-0.486629	1.704071	-1.256796	-0.323003	0.124788

Duration	Sales	Product Name	Destination
-0.470051	-0.816433	0.268835	-0.434646
-0.268605	-0.569127	0.268835	-0.434646
-0.499894	-0.711940	0.268835	1.303937
-0.492433	-0.484288	-0.525751	-0.434646
-0.126846	-0.597407	-1.320338	-0.434646

Tab 9: The scaled dataset

```
# After scaling
```

Fig 32: The dataset after scaling

Splitting data into training and test set and checking their dimensions

```
X_train (2100, 9)
X_test (900, 9)
train_labels (2100,)
test_labels (900,)
```

Building a Decision Tree Classifier

```
Parameters passed


'criterion': ['gini']
'max_depth': [10,20,30,50]
'min_samples_leaf': [50,100,150]
'min_samples_split': [150,300,450]

Output:

{'criterion': 'gini', 'max_depth':
10, 'min_samples_leaf': 50, 'min_sa
mples_split': 450}

Best grid:

DecisionTreeClassifier
(max_depth=10, min_samples_leaf=50,
min_samples_split=450,
random_state=1)

Second parameter set
'criterion': ['gini'],
'max_depth': [3, 5, 7, 10,12],
'min_samples_leaf': [20,30,40,50,60]
'min_samples_split': [150,300,450]


Output:

{'criterion': 'gini', 'max_depth':
5, 'min_samples_leaf': 20, 'min_sam
ples_split': 150}

Best grid:

DecisionTreeClassifier

(max_depth=5, min_samples_leaf=20,
min_samples_split=150,
random_state=1)

Third parameter set
'criterion': ['gini'],
'max_depth': [3.5,4.0,4.5, 5.0,5.5]
'min_samples_leaf': [40, 42, 44,46,
48,50,52,54],
'min_samples_split': [250, 270, 280
, 290, 300,310]

Output:

{'criterion': 'gini', 'max_depth':
3.5, 'min_samples_leaf': 44, 'min_s
amples_split': 250}

Best grid:

DecisionTreeClassifier(max_depth=3.
5, min_samples_leaf=44,
min_samples_split=250, random_state
=1)

Fourth parameter set
'criterion': ['gini'],
'max_depth': [4.85, 4.90,4.95, 5.0,
5.05,5.10,5.15],
'min_samples_leaf':  [40, 41,

42, 43, 44],
min_samples_split: [150, 175, 200,
210, 220, 230, 240, 250, 260, 270]

Output:

{'criterion': 'gini', 'max_depth':
4.85, 'min_samples_leaf': 44, 'min_
samples_split': 260}

Best grid:

DecisionTreeClassifier(max_depth=4.
85, min_samples_leaf=44,
min_samples_split=260,
random_state=1)
```

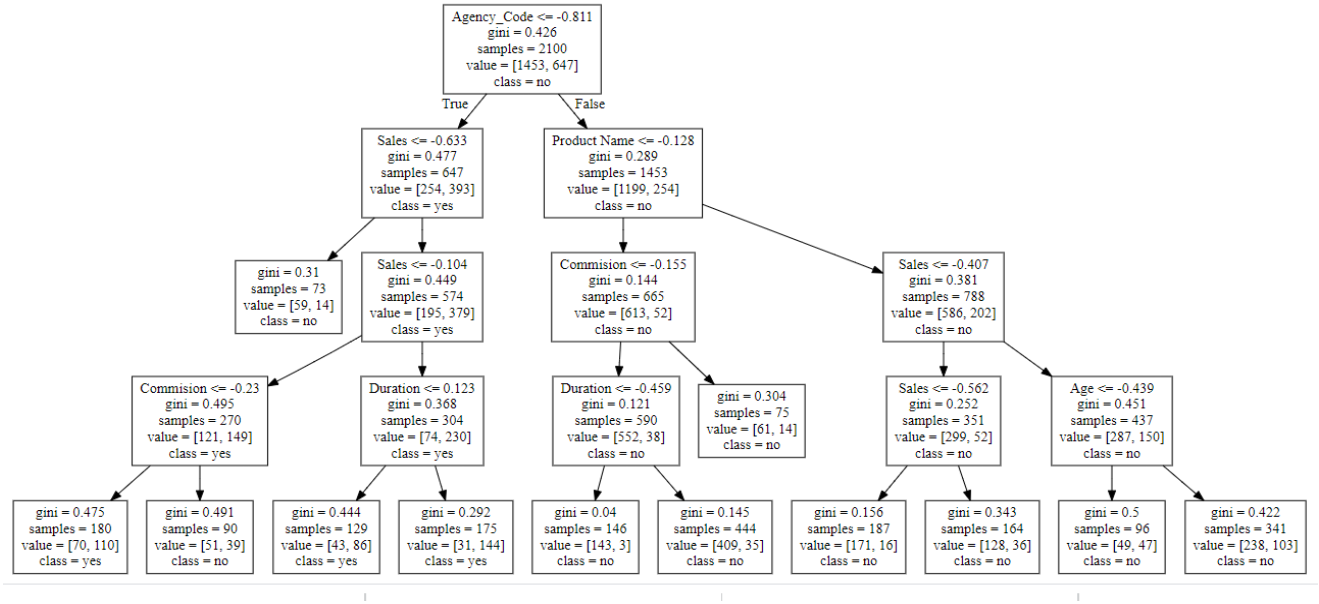


Fig 33: The decision tree

Generating Tree

A regularised (pruned) tree

The splitting criterion is agency code. A total of 2,100 agencies were divided 647: 1453 between sales and product name, respectively. The sales were further split 254 and 393, while product name was further split 1199 and 254. Age ncy code, sales, and product name are the most important features or variables, an analysis based on the following scores.

Variable Importance – DTCL

	Imp
Agency_Code	0.634112
Sales	0.220899
Product Name	0.086632
Commision	0.021881
Age	0.019940
Duration	0.016536
Type	0.000000
Channel	0.000000
Destination	0.000000

Predicting on Training and Test dataset

Getting the Predicted Classes and Probabilities

	0	1
0	0.697947	0.302053
1	0.979452	0.020548
2	0.921171	0.078829
3	0.510417	0.489583
4	0.921171	0.078829

These are the fractions of trees voting either class. The class probability of a single tree is the fraction of samples of the same class in a leaf. This tree predicts 2 to 30% chances of filing a claim. For example, 175 agencies among 304 with short-duration tours out of 574 with low sales are predicted to get requests for claims.

Building a Random Forest Classifier

```
Parameters set

'max_depth': [5,10,15],
'max_features': [4,5,6,7],
'min_samples_leaf': [10,50,70],
'min_samples_split': [30,50,70],
'n_estimators': [200, 250,300]

Output:

{'max_depth': [5, 10, 15],
'max_features': [4, 5, 6, 7],
'min_samples_leaf': [10, 50, 70],
'min_samples_split': [30, 50, 70],
'n_estimators': [200, 250, 300]})

best parameters

{'max_depth': 5,
 'max_features': 5,
 'min_samples_leaf': 10,
 'min_samples_split': 50,
 'n_estimators': 250}

Best grid

RandomForestClassifier
(max_depth=5, max_features=5,
min_samples_leaf=10,
min_samples_split=50,
n_estimators=250, random_state=1)

Second parameters set
'max_depth': [4,5,6],
'max_features': [2,3,4,5],
'min_samples_leaf': [8,9,11,15],
'min_samples_split': [46,50,55],
'n_estimators': [290,350,400]


Output:

{'max_depth': [4, 5, 6],
'max_features': [2, 3, 4, 5],
'min_samples_leaf': [8, 9, 11, 15],
'min_samples_split': [46, 50, 55],
'n_estimators': [290, 350, 400]})

best parameters

{'max_depth': 6,
 'max_features': 3,
 'min_samples_leaf': 8,
 'min_samples_split': 46,
 'n_estimators': 350}

Best grid

RandomForestClassifier
(max_depth=6, max_features=3,
min_samples_leaf=8,
min_samples_split=46,
n_estimators=350, random_state=1)
```

Predicting the Training and Testing data

Getting the Predicted Classes and Probabilities

	0	1
0	0.778010	0.221990
1	0.971910	0.028090
2	0.904401	0.095599
3	0.651398	0.348602
4	0.868406	0.131594

Variable Importance via RF

	Imp
Agency_Code	0.276015
Product Name	0.235583
Sales	0.152733
Commision	0.135997
Duration	0.077475
Type	0.071019
Age	0.039503
Destination	0.008971
Channel	0.002705

Random forest also shows the same three variables (Agency_Code, Product Name, and Sales to be most important features, in that order, in deciding if some customer will file a claim or not)

Building a Neural Network Classifier

```
Parameters set
'hidden_layer_sizes': [50,100,200],
'max_iter': [2500,3000,4000],
'solver': ['adam'],
'tol': [0.01],


Output:

'hidden_layer_sizes': [50, 100, 20]
'max_iter': [2500, 3000, 4000]
'solver': ['adam']
'tol': [0.01]})

Best parameters

{'hidden_layer_sizes': 200,
'max_iter': 2500, 'solver': 'adam',
'tol': 0.01}

Best grid

MLPClassifier
(hidden_layer_sizes=200, max_iter=2
500, random_state=1, tol=0.01)
```

Predicting the Training and Testing data

Getting the predicted classes and probabilities

```
          0         1
0  0.822676  0.177324
1  0.933407  0.066593
2  0.918772  0.081228
3  0.688933  0.311067
4  0.913425  0.086575
```

The Artificial Neural Network predicts 6 to 17% chance of being asked for claim.

2.3 Performance Metrics: Check the performance of Predictions on Train and Test sets using Accuracy, Confusion Matrix, Plot ROC curve and get ROC_AUC score for each model

CART - AUC and ROC for the training data

```
AUC: 0.823
This is area under the curve.
```

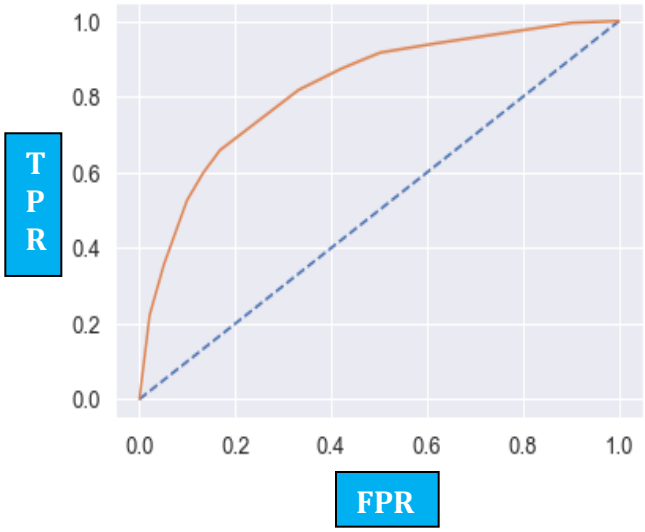


Fig 34: Receiver operating characteristic (ROC) curve for CART training data, plotting true positive rate (TPR) against false positive rate (FPR)

CART -AUC and ROC for the test data

```
AUC: 0.801
```

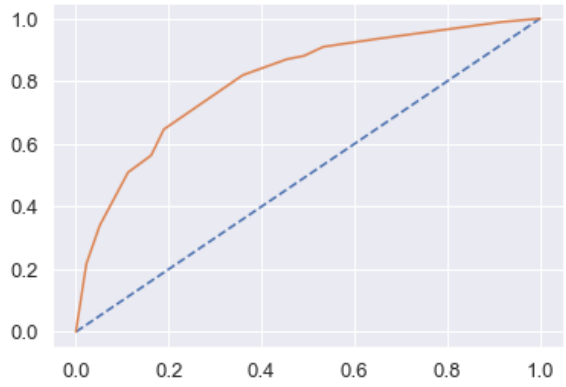


Fig 35: Receiver operating characteristic (ROC) curve for CART test data

CART Confusion Matrix and Classification Report for the training data

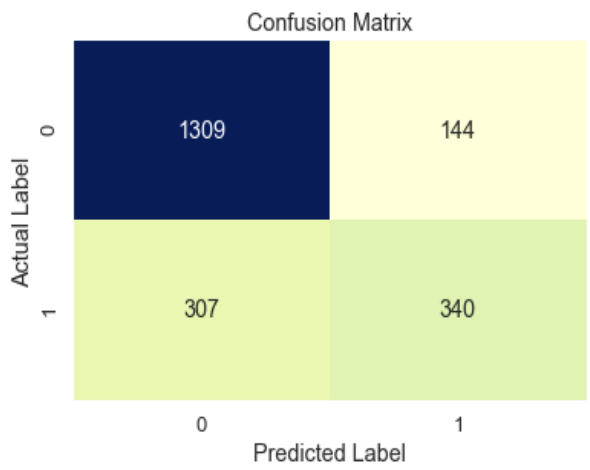


Fig 36: Confusion matrix for CART training data

CART Confusion Matrix and Classification Report for the testing data

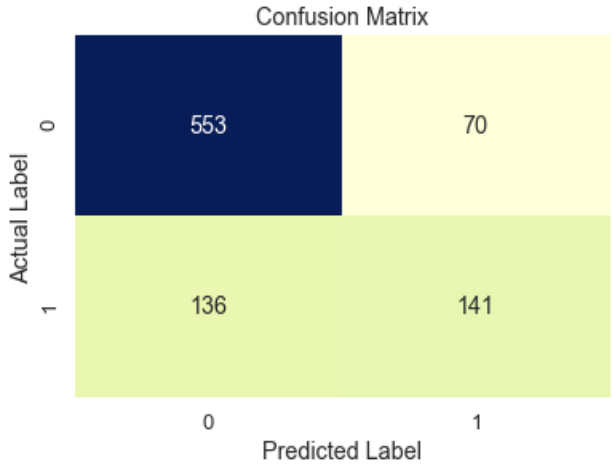


Fig 37: Confusion matrix for CART test data

#Train Data Accuracy

0.7852380952380953

Classification report:-

	precision	recall	f1-score	support
0	0.81	0.90	0.85	1453
1	0.70	0.53	0.60	647
accuracy			0.79	2100
macro avg	0.76	0.71	0.73	2100
weighted avg	0.78	0.79	0.78	2100

cart_train_precision 0.7
cart_train_recall 0.53
cart_train_f1 0.6

#Test Data Accuracy

0.7711111111111111

Classification report:-

	precision	recall	f1-score	support
0	0.80	0.89	0.84	623
1	0.67	0.51	0.58	277
accuracy			0.77	900
macro avg	0.74	0.70	0.71	900
weighted avg	0.76	0.77	0.76	900

cart_test_precision 0.67
cart_test_recall 0.51
cart_test_f1 0.58

Train Data

AUC: 82%
Accuracy: 79%
Precision: 70%
f1-Score: 60%

Test Data

AUC: 80%
Accuracy: 77%
Precision: 80%
f1-Score: 84%

Cart Conclusion

Training and Test set results are almost similar, and with the overall measures high, this is a good model. Agency_Code is the most important variable for predicting claim.

RF Model Performance Evaluation on Training data

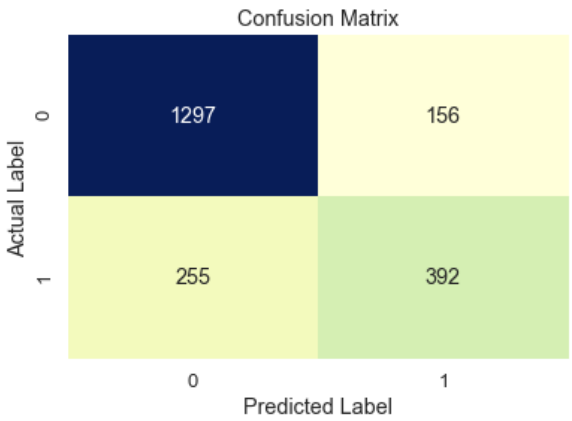


Fig 38: Confusion matrix for RF training data

RF training accuracy score

0.8042857142857143

Classification report:-

	precision	recall	f1-score	support
0	0.84	0.89	0.86	1453
1	0.72	0.61	0.66	647
accuracy			0.80	2100
macro avg	0.78	0.75	0.76	2100
weighted avg	0.80	0.80	0.80	2100

rf_train_precision 0.72
rf_train_recall 0.61
rf_train_f1 0.66

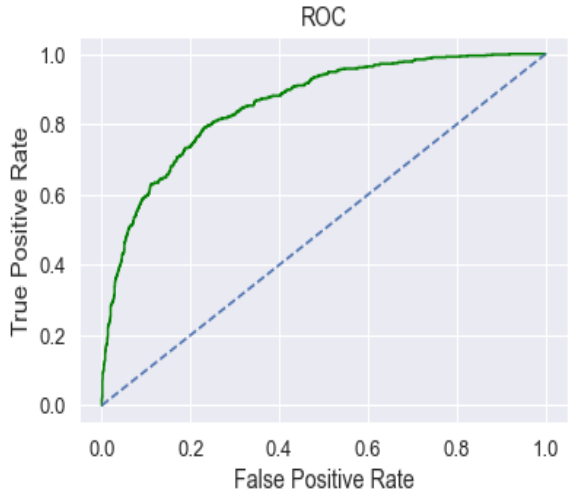


Fig 39: ROC curve for RF training data

Area under Curve is
0.8563713512840778

RF Model Performance Evaluation on Test data

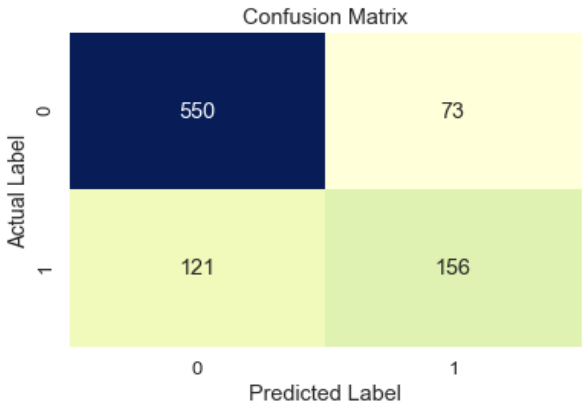


Fig 40: Confusion matrix for RF test data

RF test accuracy score

0.7844444444444445

Classification report:-

	precision	recall	f1-score	support
0	0.82	0.88	0.85	623
1	0.68	0.56	0.62	277
accuracy			0.78	900
macro avg	0.75	0.72	0.73	900
weighted avg	0.78	0.78	0.78	900

rf_test_precision 0.68
rf_test_recall 0.56
rf_test_f1 0.62

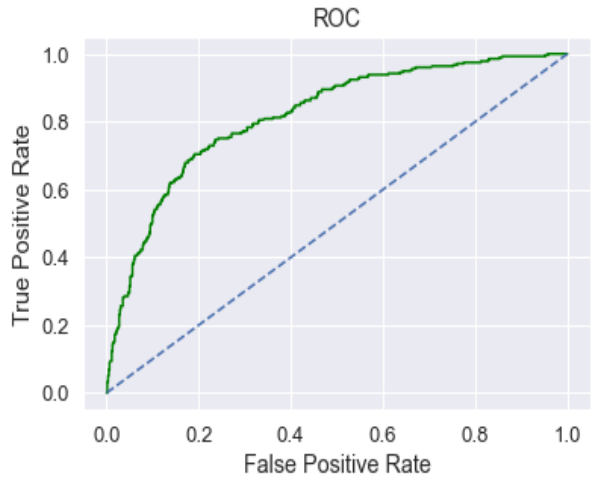


Fig 41: ROC curve for RF test data

Area under Curve is
0.8181994657271499

Random Forest Conclusion

Train Data

AUC: 86%
Accuracy: 80%
Precision: 72%
f1-Score: 66%

Test Data

AUC: 82%
Accuracy: 78%
Precision: 68%
f1-Score: 62

Training and Test set results are almost similar, and with the overall measures high, this is also a good model.

Agency_Code is again the most important variable for predicting claim.

NN Model Performance Evaluation on Training data

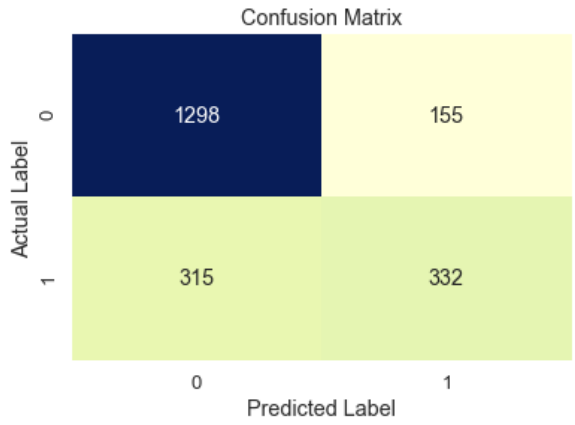


Fig 42: Confusion matrix for ANN training data

NN training data accuracy score

```
0.7761904761904762
```

```
Classification report:-
```

	precision	recall	f1-score	support
0	0.80	0.89	0.85	1453
1	0.68	0.51	0.59	647
accuracy			0.78	2100
macro avg	0.74	0.70	0.72	2100
weighted avg	0.77	0.78	0.77	2100

```
nn_train_precision  0.68
nn_train_recall  0.51
nn_train_f1  0.59
```

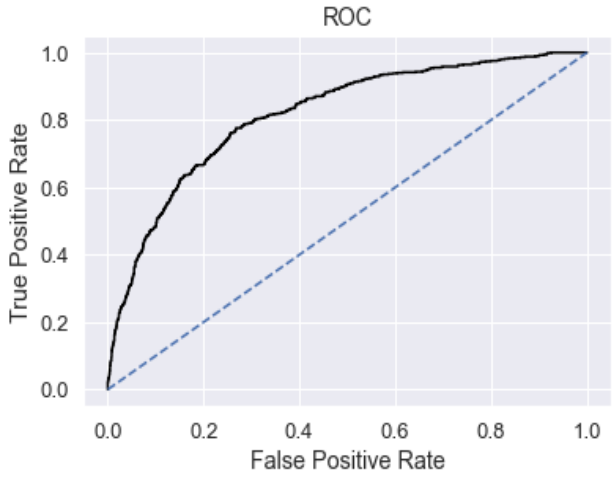


Fig 43: ROC curve for ANN training data

```
Area under Curve is
0.8166831721609928
```

NN Model Performance Evaluation on Test data

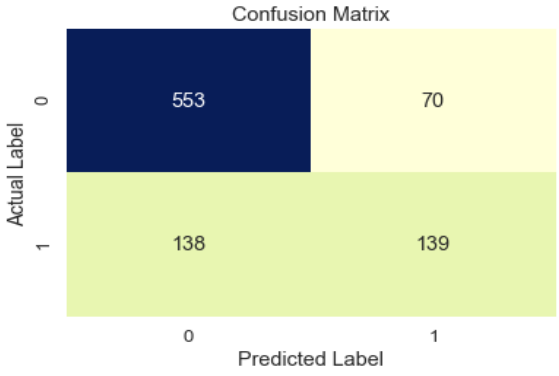


Fig 44: Confusion matrix for ANN test data

NN test accuracy score

0.7688888888888888

Classification report:-

	precision	recall	f1-score	support
0	0.80	0.89	0.84	623
1	0.67	0.50	0.57	277
accuracy			0.77	900
macro avg	0.73	0.69	0.71	900
weighted avg	0.76	0.77	0.76	900

nn_test_precision 0.67
nn_test_recall 0.5
nn_test_f1 0.57

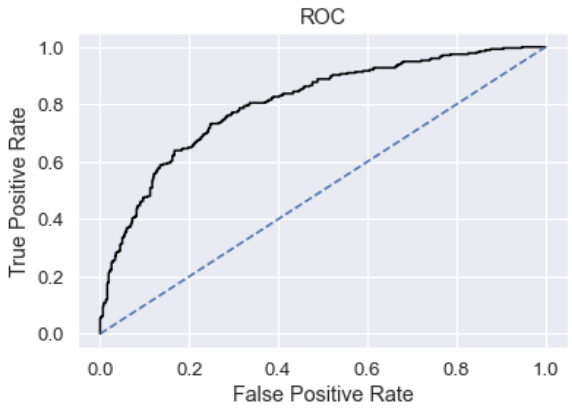


Fig 45: ROC curve for ANN test data

Area under Curve is
0.8044225275393896

Neural Network Conclusion

Train Data

AUC: 82%
Accuracy: 78%
Precision: 68%
f1-Score: 59

Test Data

AUC: 80%
Accuracy: 77%
Precision: 67%
f1-Score: 57%

Neural network conclusion

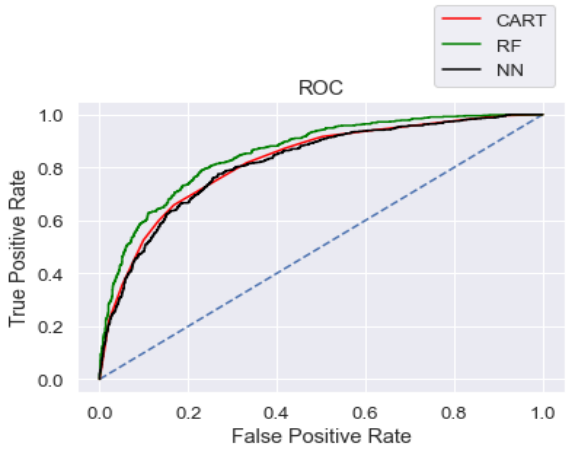
Training and test set results are almost similar, and with the overall measures high, this as well is a good model. Neural network is a 'black box' method that doesn't give you the most important variables. One doesn't know what happens in the hidden layers.

2.4 Final Model: Compare all the model and write an inference which model is best/optimized.

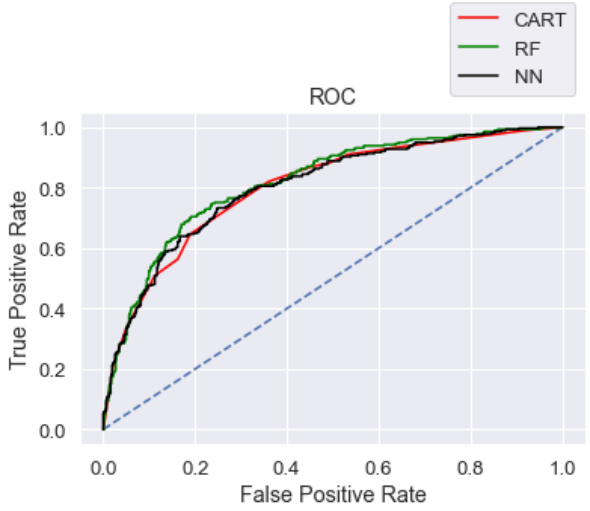
	CART Train	CART Test	Random Forest Train	Random Forest Test	Neural Network Train	Neural Network Test
Accuracy	0.79	0.77	0.80	0.78	0.78	0.77
AUC	0.82	0.80	0.86	0.82	0.82	0.80
Recall	0.53	0.51	0.61	0.56	0.51	0.50
Precision	0.70	0.67	0.72	0.68	0.68	0.67
F1 Score	0.60	0.58	0.66	0.62	0.59	0.57

Tab 10: Comparison of the training and test data of three different models

ROC Curve for the training data for three models



ROC Curve for the test data for three models



Conclusion

For predicting claim, we select the Random Forest model, as it has better accuracy, precsion, recall, and f1 score than the CART and Neural Network models.

2.5 Inference: Based on these predictions, what are the business insights and recommendations

The data shows that 90% of insurance is done via the online channel.

Streamlining online experiences benefitted customers, leading to an increase in conversions, which raised profits.

Almost all the offline business has a 'claim' associated. We must find out why.

Agency JZI must train its resources to pick up sales, since the agency is scraping the bottom of the business. It can run a promotional marketing campaign or evaluate if it needs to tie up with another agency.

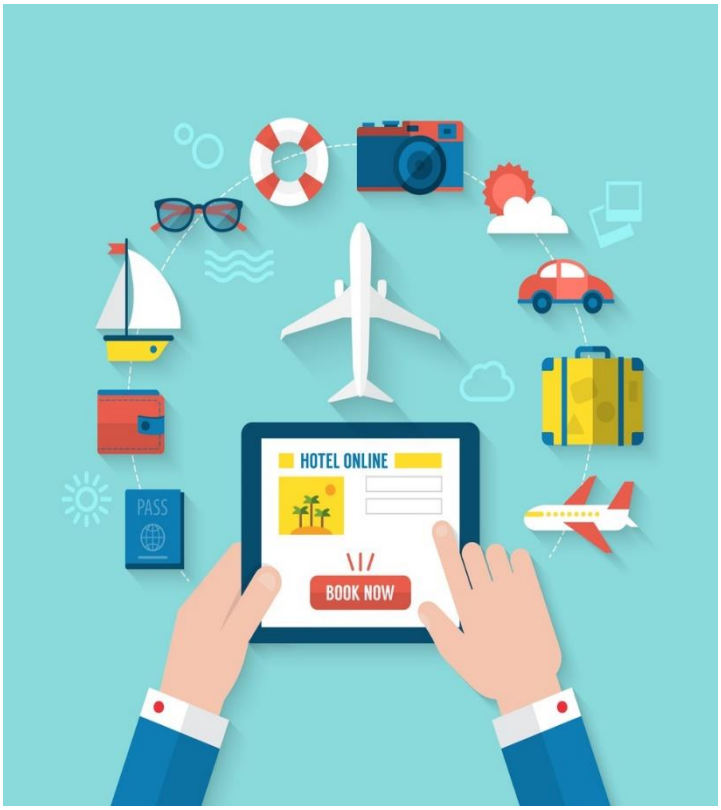
The RF model gives us 80% accuracy. Based on this claim data pattern, the agencies can cross sell insurance to the people who book airline tickets or holiday plans.

More sales happen via airlines than the agency. The claims as well are processed more at the airlines. We may have to do a deeper investigation of the workflow.

Key performance indicators (KPIs) of the insurance claims are:- Reduce claims cycle time. Increase customer satisfaction. Combat fraud. Optimize claims recovery. Reduce claim handling costs.

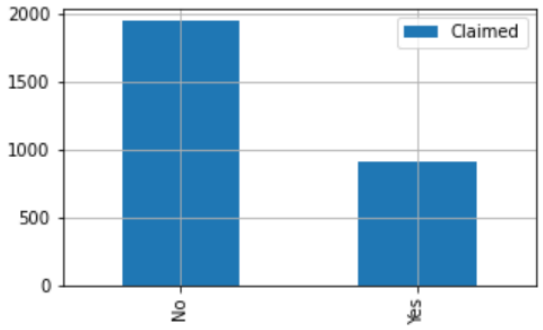
Insights gained from data and machine-learning-powered analytics can help the agencies insure more people, extend their products, and discover new risk-transfer solutions in areas such as non-damage business interruption and reputational damage.

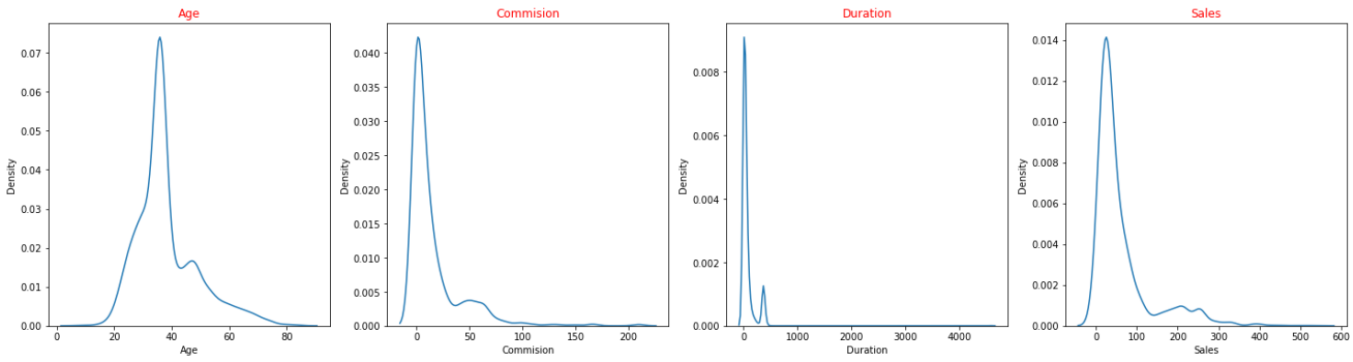
We recommended collecting more real-time, unstructured, and previous data, if possible, to be able to also study the impact of day of the incident, time, age group on claims and associate it with external information such as location, behavior patterns, weather information, and the type of airline and vehicle.



Last look at the target variable and some other features

Claims





Inference:

The volume of claims is worrisome indeed for the tour agencies.

20 to 40 is the age of travelling and that's where most customers are.
Low commission is on the higher side.

Short trips make tallest curves.

Sales between 0 and100 occur the most.