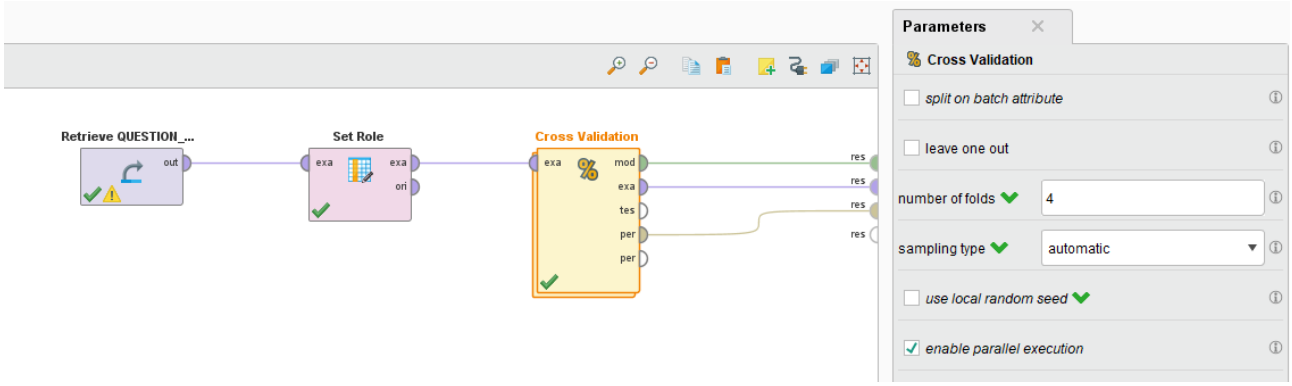
Question 1

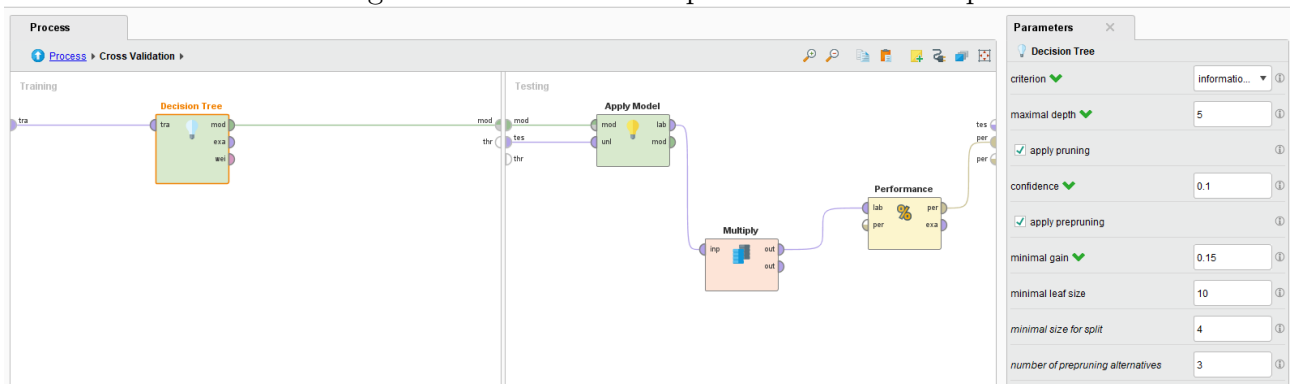
Question 2

(a)

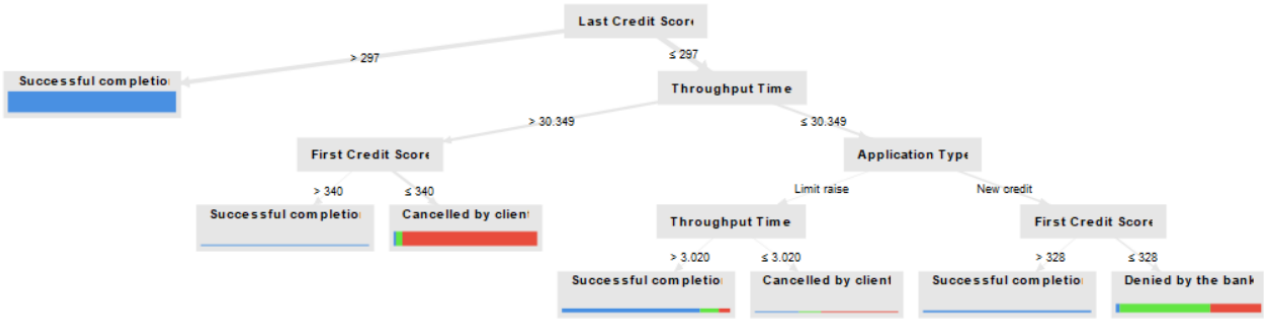
After importing the situation table from Celonis into RapidMiner we can design the decision tree learning process using the cross validation operator after setting the class variable:



We construct the training section of the cross validation using the decision tree operator in RapidMiner and also specifying the given parameters. In the testing section we simply apply the current test set to the given model and run a performance check upon it.



This process results in the following decision tree:



Here we can see that for all applications, where the applicants last credit score was above 297, the application was successfully completed.

We can also observe, that the very few applicants for new credits, those credit scores dropped significantly (First Credit Score > 329 and Last Credit Score ≤ 297) within a time less or equal to 30.349 days, all completed their application successfully.

Using the cross validation operator, we could also observe the following performance metrics:

Table View Plot View

accuracy: 90.16% +/- 0.80% (micro average: 90.16%)

	true Successful completion	true Denied by the bank	true Cancelled by client	class precision
pred. Successful completion	2600	40	30	97.38%
pred. Denied by the bank	23	524	296	62.16%
pred. Cancelled by client	35	66	1368	93.12%
class recall	97.82%	83.17%	80.76%	

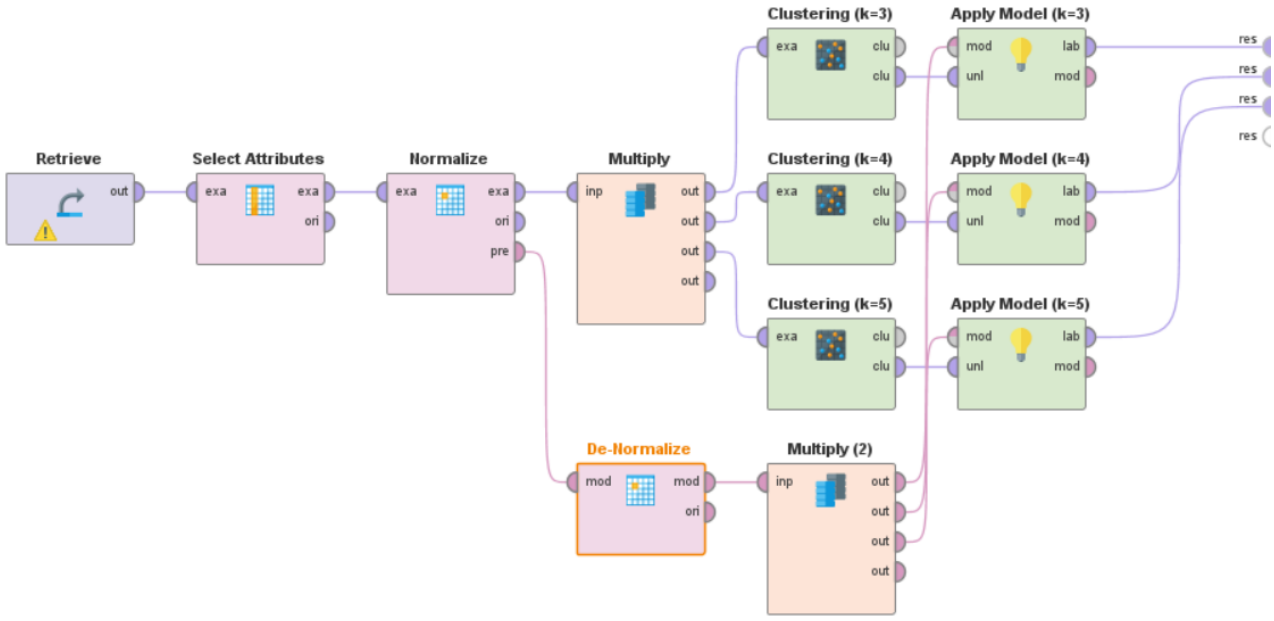
(b)

1. Most applicants with a last credit score of at most 297 cancelled their own application when it was already roughly over a month's time in progress. Perhaps the current application process needs to be revised to minimize throughput times.
2. 62% of applications, that was roughly under a months time in progress, by applicants with a last credit score of at most 297 for a new credit were rejected by the bank, if their first credit score was at most 328. If the bank seeks for more successful application completions, perhaps they should adjust their rejection criteria.

Question 3

(a)

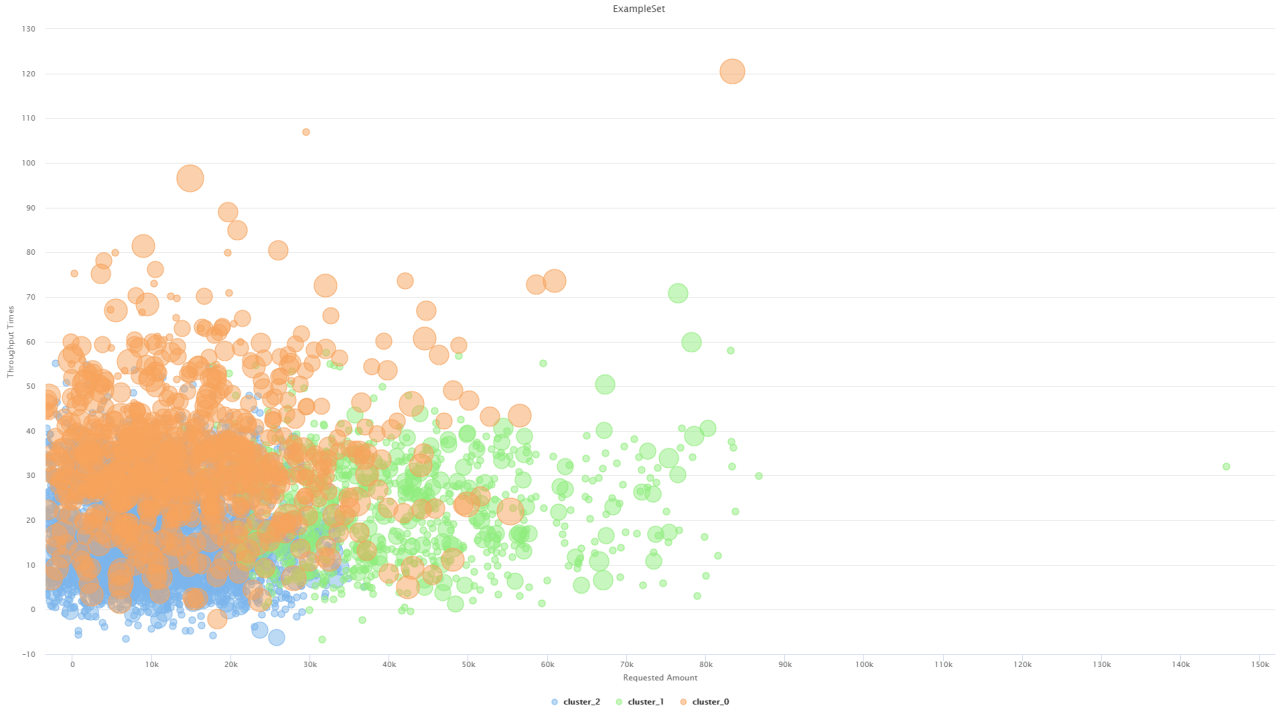
After selecting the three specified attributes, normalization by Z-transformation is done. The result is multiplied so it can be fed to all k-means clusterers at once. For each clustering we set the necessary 'k' and 'max runs' and select the 'add as label' checkbox so the cluster labels are added as a new column. To use the unscaled dataset we first reverse the Z-transformation by De-Normalizing and then apply it to the clustered data.

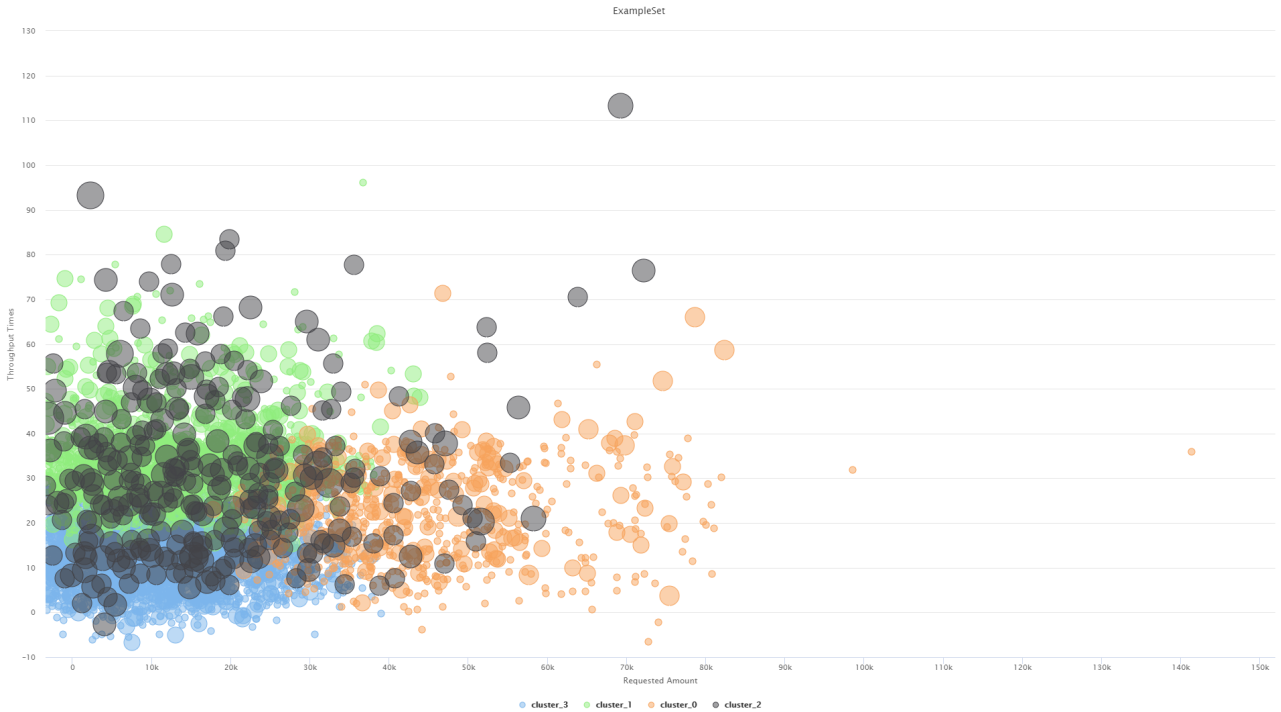


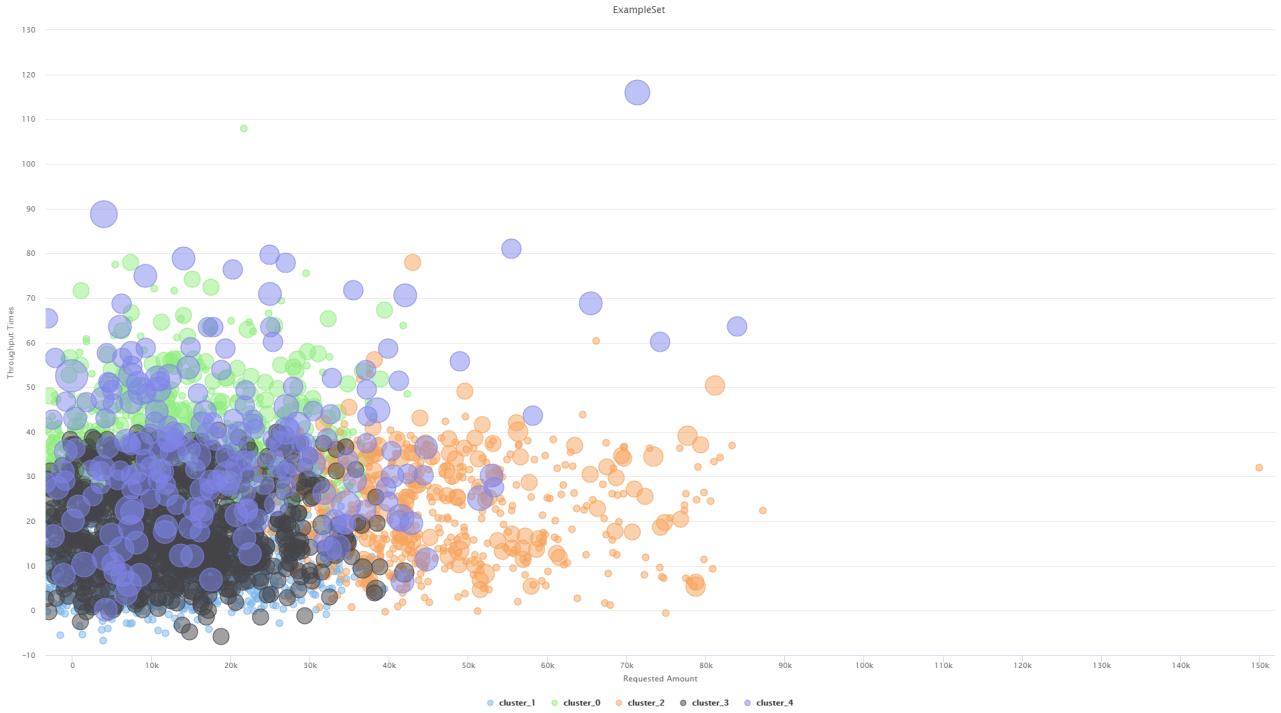
The following graphics show the clustering results for Number of Offers mapped against the Requested Amount. Over all $k \in \{3, 4, 5\}$ there are a few similar clusters:

- `Low Requested Amount (mostly 0-25k) and low Number of Offers (0-2.5)`. However this region is further split with higher k's.

- `Higher Requested Amount (mostly >25k) and low Number of Offers (mostly 0-2.5)`. This class is very similar over all k.
- `Higher Number of Offers (>2.5)`. Only k=3 also clusters in lower values because of a lack of options.







Attribute	cluster_0	cluster_1	cluster_2
Requested Amount	-0.082	1.688	-0.407
Number of Offers	1.622	-0.240	-0.330
Throughput Times	1.028	0.101	-0.273

Attribute	cluster_0	cluster_1	cluster_2	cluster_3
Requested Amount	1.971	-0.282	0.077	-0.351
Number of Offers	-0.150	-0.128	3.025	-0.249
Throughput Times	0.018	0.924	0.675	-0.772

Attribute	cluster_0	cluster_1	cluster_2	cluster_3	cluster_4
Requested Amount	-0.281	-0.342	2.063	-0.244	0.200
Number of Offers	-0.344	-0.508	-0.189	1.046	3.285
Throughput Times	0.989	-0.776	0.035	-0.203	1.142

Cluster Model

```
Cluster 0: 803 items
Cluster 1: 843 items
Cluster 2: 3336 items
Total number of items: 4982
```

Cluster Model

```
Cluster 0: 648 items
Cluster 1: 1713 items
Cluster 2: 296 items
Cluster 3: 2325 items
Total number of items: 4982
```

Cluster Model

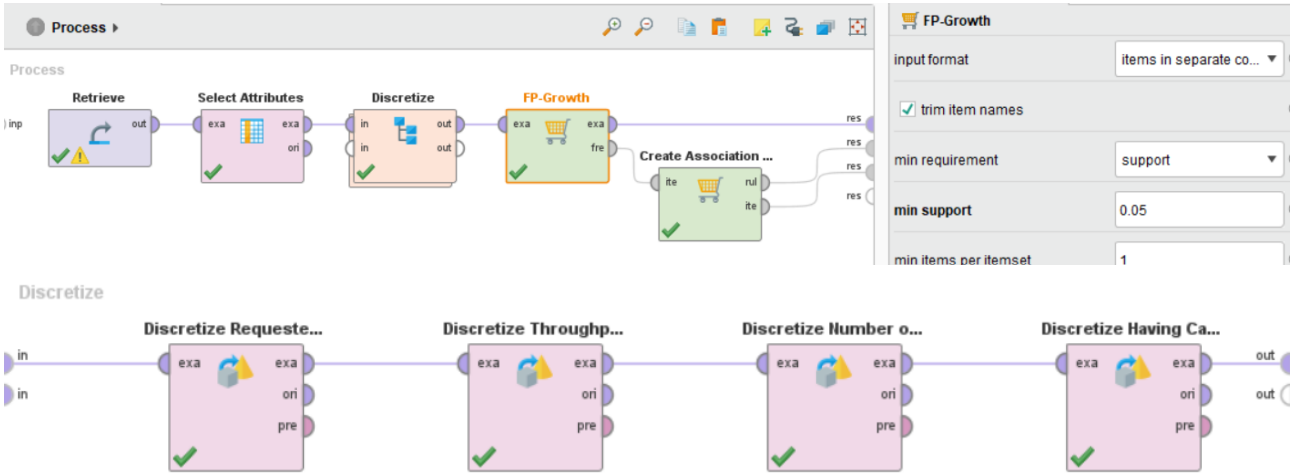
```
Cluster 0: 1415 items
Cluster 1: 1929 items
Cluster 2: 590 items
Cluster 3: 832 items
Cluster 4: 216 items
Total number of items: 4982
```

(b)

Question 4

(a)

The following screenshots show the process used for 4a). A minimum support count of 250 was specified in the task. At a dataset size of 4982 this equals a minimum support of $\frac{250}{4982} = 0.05$



158 association rules were found at a minimum confidence of 0.8, the top 10 in confidence can be seen below:

No.	Premises	Conclusion	Support	Confiden...	LaPlace	Gain	p-s	Lift
158	one offer, med duration, Cancelled by client, Car	not called	0.063	0.991	0.999	-0.064	0.005	1.082
157	one offer, med duration, med request, Cancelled b...	not called	0.071	0.983	0.999	-0.074	0.005	1.074
156	med duration, Cancelled by client, Home improve...	not called	0.058	0.983	0.999	-0.060	0.004	1.074
155	one offer, med duration, low request, Cancelled by...	not called	0.079	0.983	0.999	-0.082	0.005	1.074
154	med duration, med request, Cancelled by client	not called	0.089	0.982	0.999	-0.093	0.006	1.073
153	med duration, Cancelled by client, Car	not called	0.078	0.982	0.999	-0.081	0.005	1.073
152	one offer, med duration, Cancelled by client	not called	0.200	0.981	0.997	-0.208	0.013	1.072
151	med duration, Cancelled by client	not called	0.249	0.976	0.995	-0.261	0.015	1.066
150	med duration, Cancelled by client, high request	not called	0.063	0.972	0.998	-0.067	0.004	1.062
149	med duration, low request, Cancelled by client	not called	0.096	0.972	0.997	-0.102	0.006	1.062

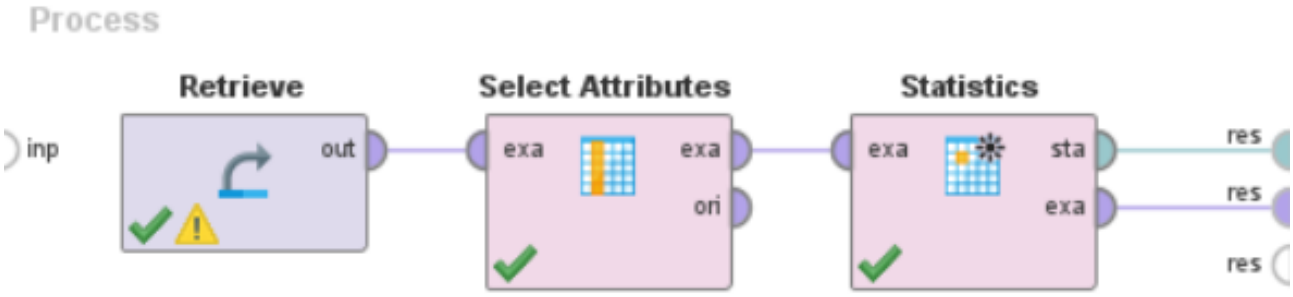
(b)

The conclusion with the highest confidence (0.991) states that if there is a car loan with one offer and a medium throughput time that gets cancelled by the client, the client likely doesn't call to complete the application

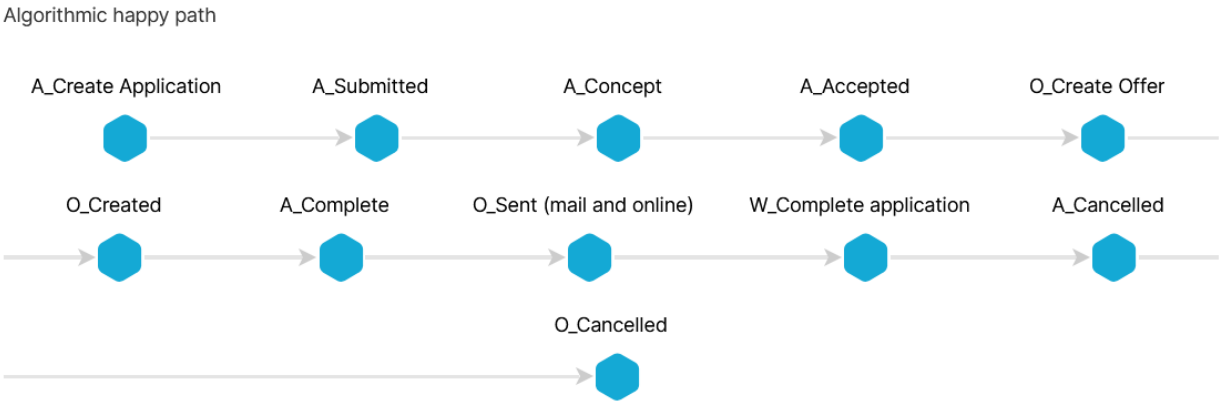
Question 5

(a)

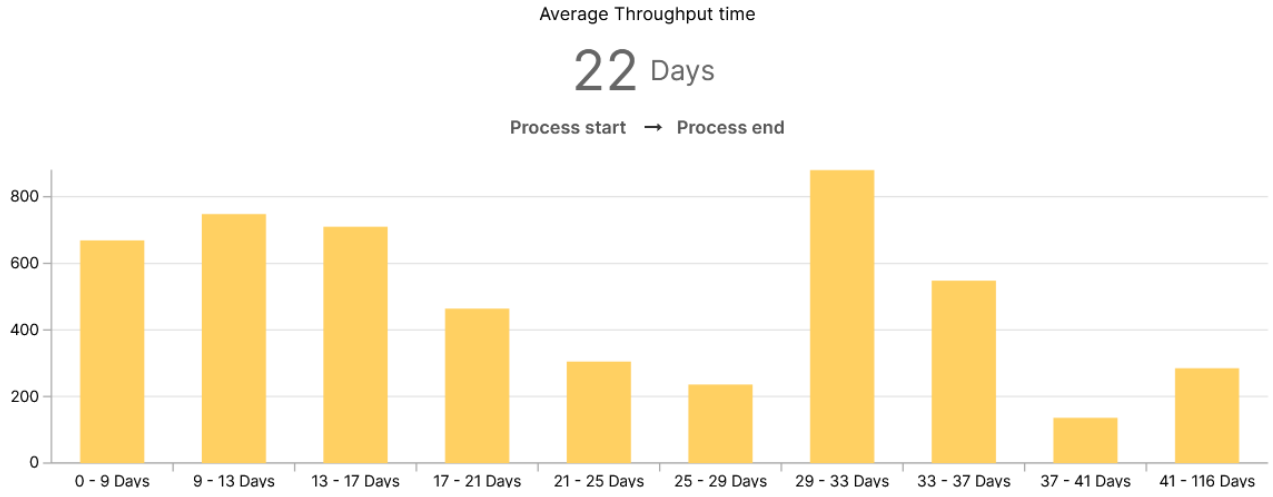
There are 4982 Applications with an average throughput time of 21.904 as determined using the following Process:



Using Celonis Process AI on our Dataset we learn that the most frequent variant (happy path) happens 320 times. This variant can be seen below.



The following graph shows the frequency distribution of the throughput times. As one can see they are initially quite high and eventually deteriorate in frequency until the 29-33 Day, where they spike again and after that deteriorate quickly again. The reason for the spike at 29-33 Days is probably that a new month begins/ends at this time. Therefore many applications will likely be terminated at this time for administrative reasons.



(b)

(c)

(d)