Apoorva Singh| 40428

IAI | AGH | WEAiiB

Introduction to AI: Clustering data using Scikit-learn

Report 3

Contents

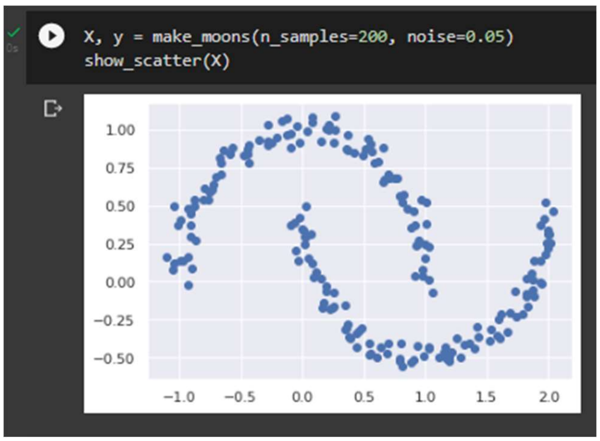
Exercise 1:

Summary of findings:

For both Moons and Circles algorithm the DBSCAN with an eps of 2 and 3 respectively worked best.

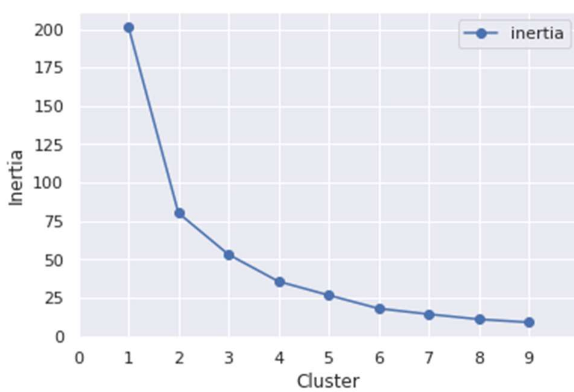
Moons:

Default graph:

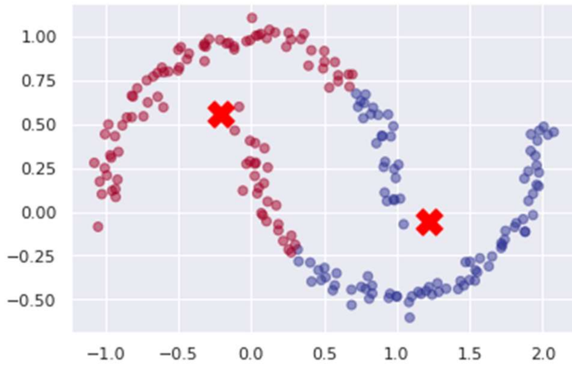


Kmeans:

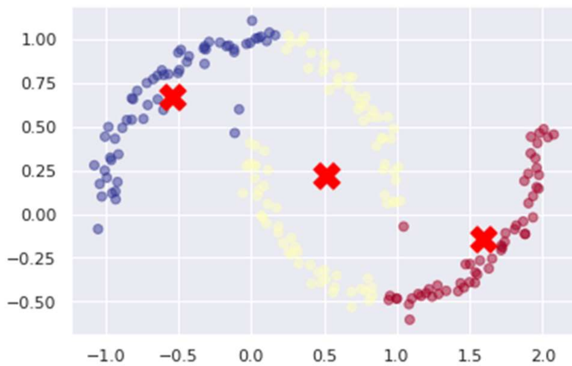
Generating the elbow graph:



We can see the elbow lies at **point 2.** Point 3 is not as optimized, however still demonstrated below:



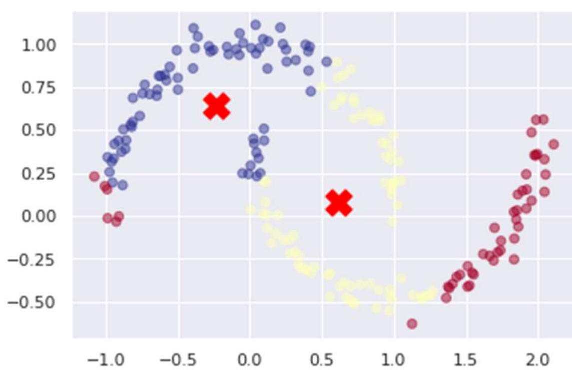
For k=2 inertia is 80.14738901867013



For k=3 inertia is 53.19848723408633

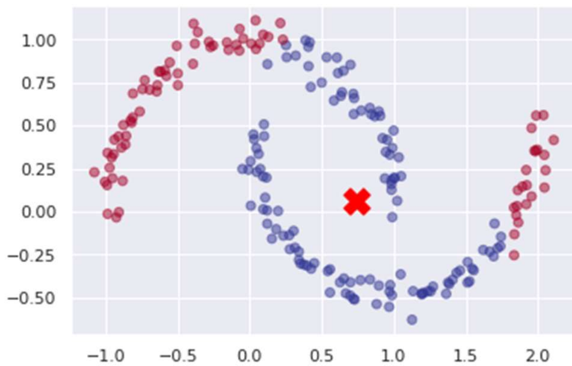
Mean Shift:

Number of clusters: 2




```
bandwidth = estimate_bandwidth(X, quantile=.4, n_samples=1000)
```

quantile at 4 gives similar result to kmeans at 2 clusteres

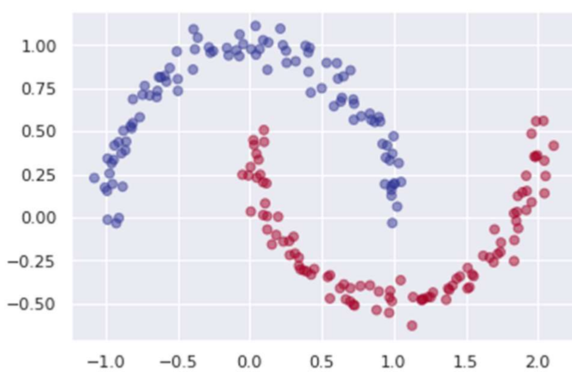


DBSCAN:

DBSCAN gave by far the best result with eps being a value of 0.2

```
#eps for DBSCAN,
dbscan = DBSCAN(eps=0.2)
y_pred = dbscan.fit_predict(X)
print('Number of clusters:', len(set(y_pred))-(1 if -1 in y_pred else 0))
print('Number of outliers:', list(y_pred).count(-1))
show_scatter(X, y_pred)
```

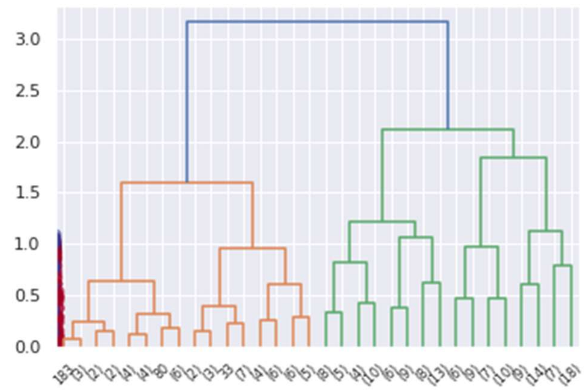
```
Number of clusters: 2
Number of outliers: 0
```

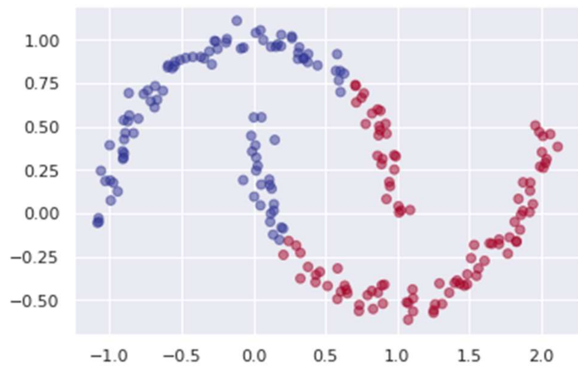


Agglomerative Clustering:

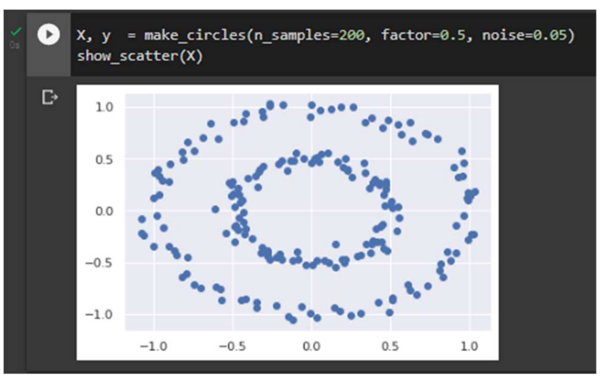
```
#distance_threshold, affinity or linkage for AgglomerativeClustering.
ac = AgglomerativeClustering(n_clusters=None, distance_threshold=3,
                             affinity='euclidean', linkage='complete')
```

Having a distance threshold of 3 gives 2 clusters and the below diagram, anything more than three gives a single cluster and anything less than 3 gives up to 8 clusters.



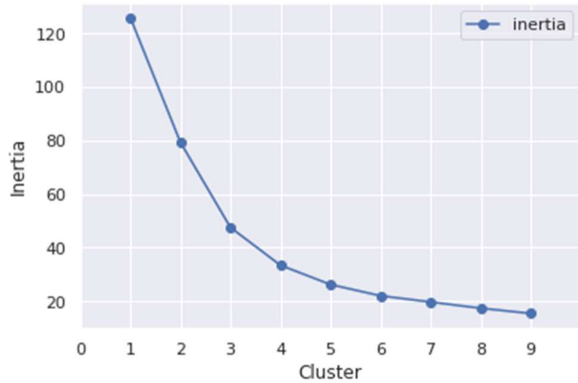


Circles:

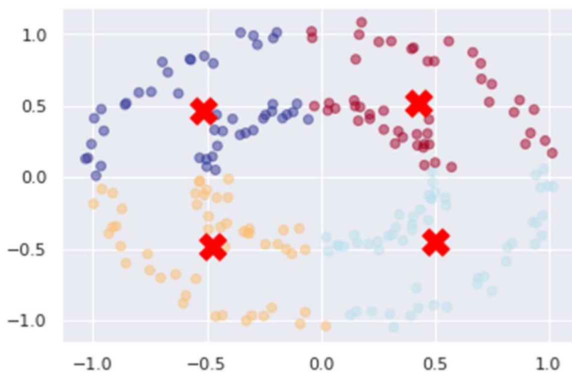


Kmeans:

Generating the elbow graph:



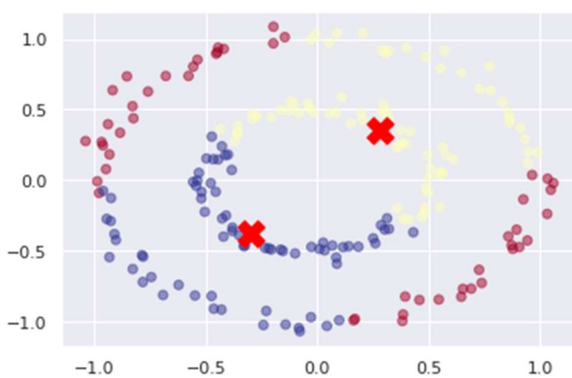
We can see the elbow lies at **point 3 or 4.**



For k=4 inertia is 33.354594398334626

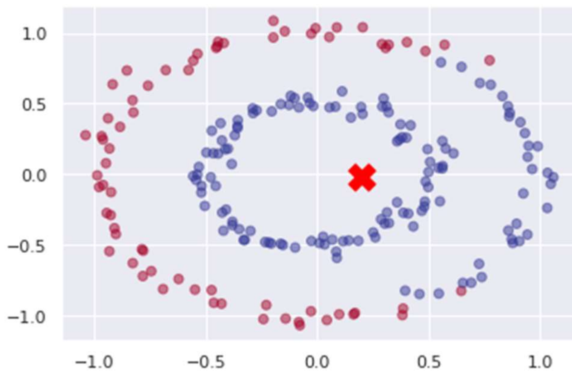
Mean Shift:

Number of clusters: 2




```
bandwidth = estimate_bandwidth(X, quantile=.4, n_samples=1000)
ms = MeanShift(cluster_all=False, bandwidth=bandwidth)
```

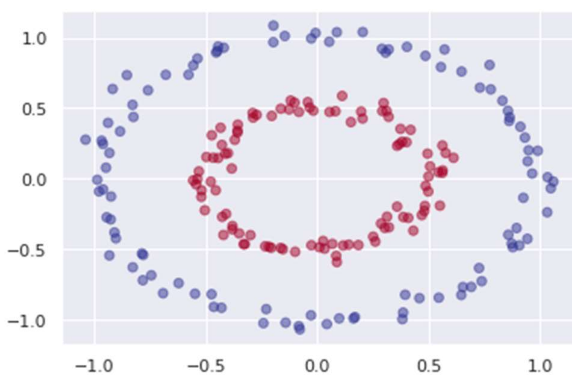
Better result than k means, thus far most satisfactory



DBSCAN:

DBSCAN gave by far the best result with eps being a value of 0.3, and no outliers

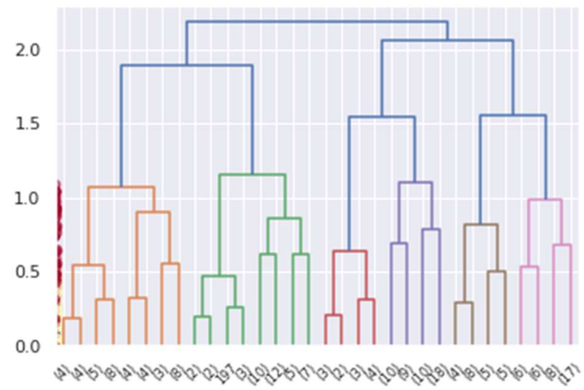
```python
dbscan = DBSCAN(eps=0.3)
y_pred = dbscan.fit_predict(X)
print('Number of clusters:', len(set(y_pred))-(1 if -1 in y_pred else 0))
print('Number of outliers:', list(y_pred).count(-1))
show_scatter(X, y_pred)
```

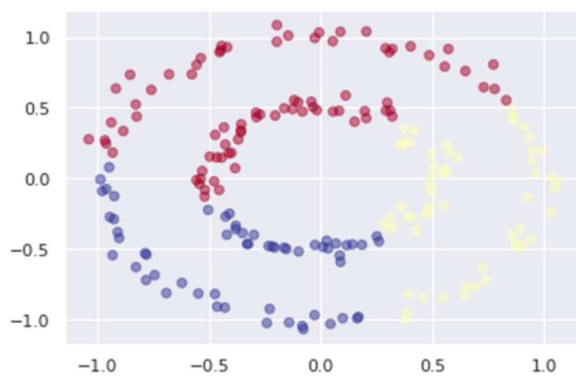


Agglomerative Clustering:

```python
ac = AgglomerativeClustering(n_clusters=None, distance_threshold=2,
                             affinity='euclidean', linkage='complete')
```

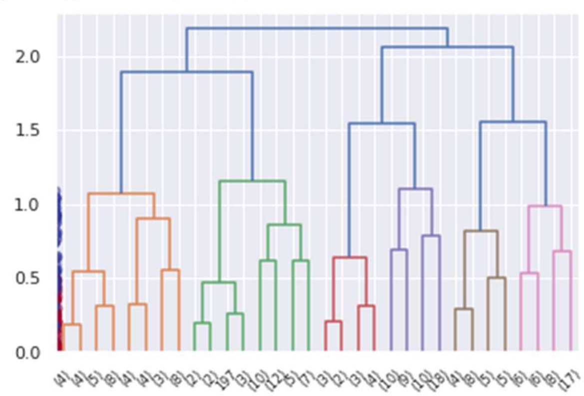
Having a distance threshold of 2 gives 3 clusters and the below diagram, anything more than two gives a single cluster and anything less than 2 gives up to 9 clusters.

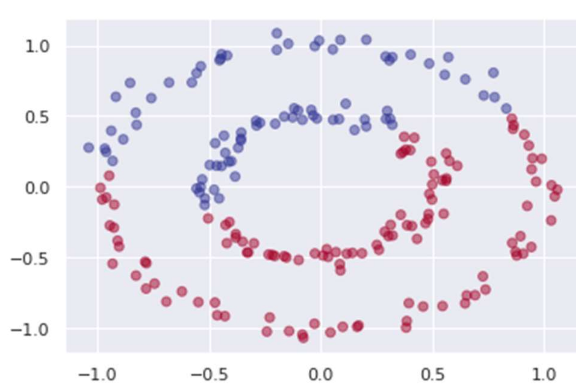




The closest you can get is with 2.1 and 2 clusters, the result though leaves much to be desired.

```
ac = AgglomerativeClustering(n_clusters=None, distance_threshold=2.1,
                             affinity='euclidean', linkage='complete')
y_pred = ac.fit_predict(X)
```





Exercise 2:

```
from sklearn.datasets import load_sample_image
flower = load_sample_image("flower.jpg")
ax = plt.axes(xticks=[], yticks=[])
ax.imshow(flower);
```

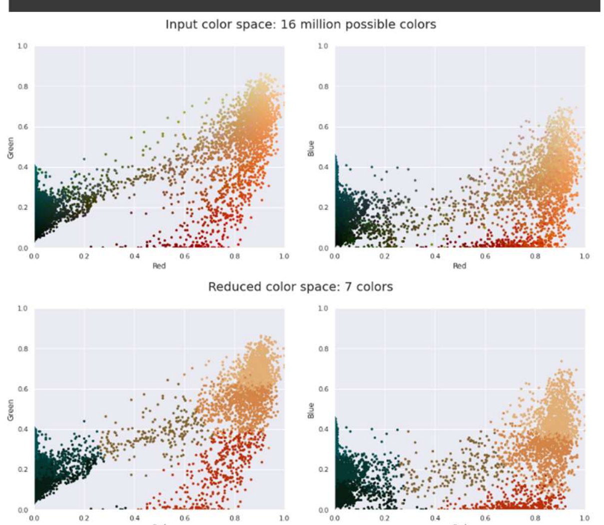
Initially generated Image:

```
flower.shape
```

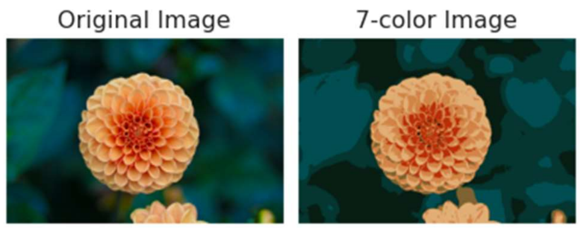
```
(427, 640, 3)
```

```
[156] data = flower / 255.0 # use 0...1 scale
      data = data.reshape(427 * 640, 3)
      data.shape
```

```
(273280, 3)
```



Final Output (7 color Image):



Exercise 3:

Task 1 Which column from the mergedcustomers.csv dataset strongly affects cluster partitioning? Justify.

Task 2 What is the best parameter to represent the actual values in your opinion when modifying the value of distance_threshold to 'manhattan' or 'euclidean' , include affinity.

My analysis:

I believe that **PROFIT_YTD** strongly affects cluster partitioning. The reason being that throughout all the clustering methods, Kmeans, Meanshift, DBSCAN and Agglomerative Clustering (both Euclidean and Manhattan), the one constant that seemed to divide the group was PROFIT_YTD. The age, Days since last trade, Total units traded, Days since last login, were all tried out with the same result, the graph always divided in the Profit column and seemingly random otherwise.

Below I have given 5 test cases for different column axis to see impact on grouping, and then further given visual proof of the different cluster partitioning methods with all 5 test cases to show the result and how I reached my conclusion.

Test Cases:

CASE 1

Data is divided by Profit.

```
x_name = 'AGE'
y_name = 'PROFIT_YTD'
y_name = 'DAYSSINCELASTTRADE'
```

CASE 2

Data is divided by Profit.

```
x_name = 'AGE'
y_name = 'PROFIT_YTD'
z_name = 'TOTALUNITSTRADED'
```

CASE 3

Data is divided by Profit.

```
x_name = 'AGE'
y_name = 'PROFIT_YTD'
z_name ='DAYSSINCELASTLOGIN'
```

CASE 4

Data is divided at random, division halfway with days since last trade. Not properly grouped.

```
x_name = 'AGE'
y_name = 'DAYSSINCELASTTRADE'
z_name ='DAYSSINCELASTLOGIN'
```

CASE 5

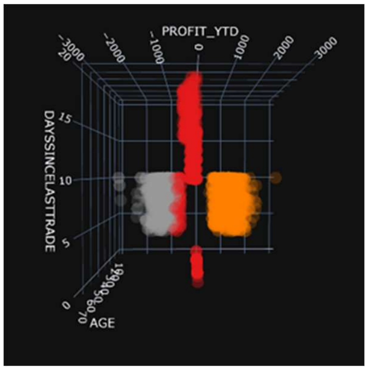
Data is divided at random, division halfway with days since last trade. Not properly grouped.

```
x_name = 'TOTALUNITSTRADED'
y_name = 'DAYSSINCELASTTRADE'
z_name ='DAYSSINCELASTLOGIN'
```

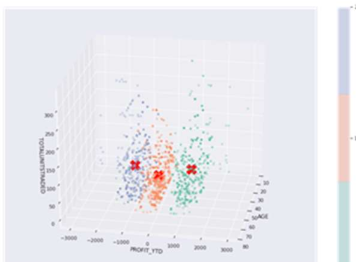
Analysis Proof by Clustering Methods:

Kmeans with 3 clusters:

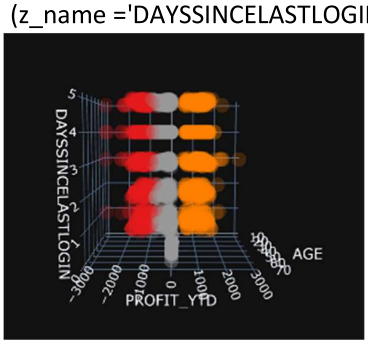
1. (z_name = 'DAYSSINCELASTTRADE')



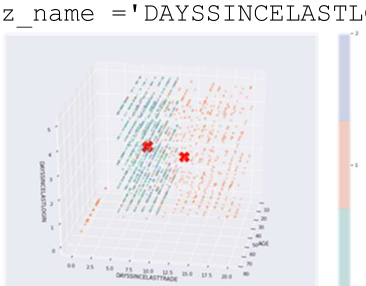
2. (z_name = 'TOTALUNITSTRADED')



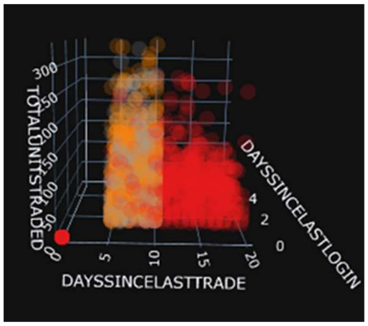
3. (z_name ='DAYSSINCELASTLOGIN')



4.
```
x_name = 'AGE'
y_name = 'DAYSSINCELASTTRADE'
z_name ='DAYSSINCELASTLOGIN'
```

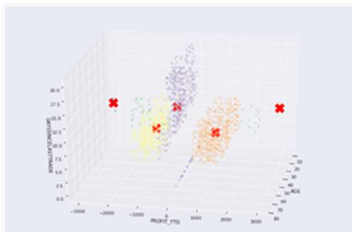


5.
```
x_name = 'TOTALUNITSTRADED'
y_name = 'DAYSSINCELASTTRADE'
z_name ='DAYSSINCELASTLOGIN'
```

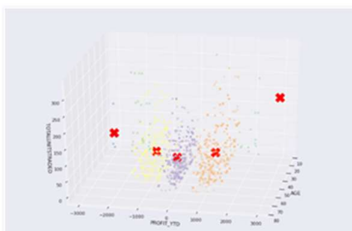


Meanshift with 5 clusters:

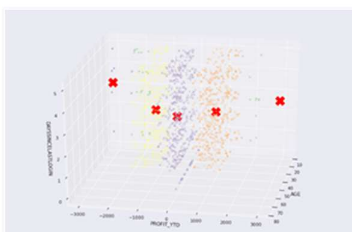
1. (z_name = 'DAYSSINCELASTTRADE')



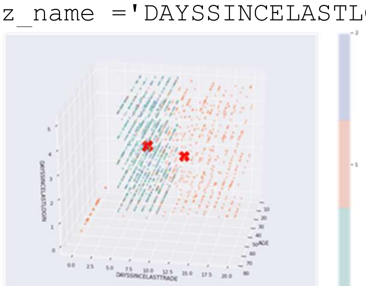
2. (z_name = 'TOTALUNITSTRADED')



3. (z_name ='DAYSSINCELASTLOGIN')



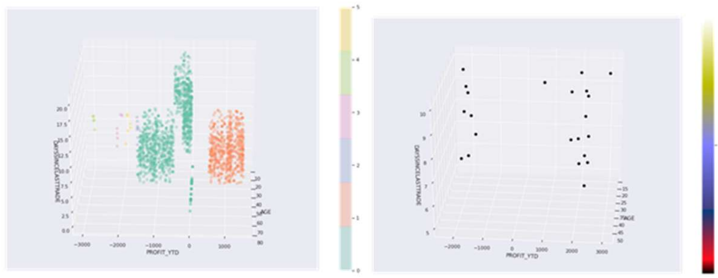
```
4. x_name  =  'AGE'
   y_name  =  'DAYSSINCELASTTRADE'
   z_name  ='DAYSSINCELASTLOGIN'
```



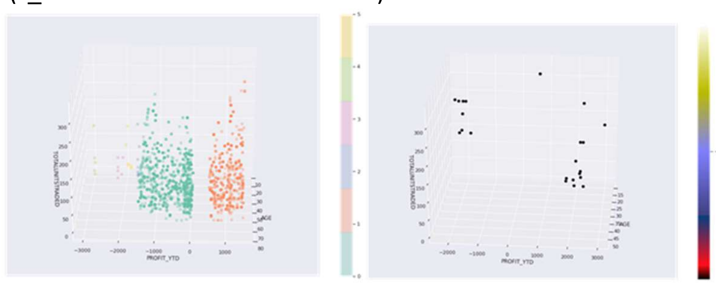
```
5. x_name  =  'TOTALUNITSTRADED'
   y_name  =  'DAYSSINCELASTTRADE'
   z_name  ='DAYSSINCELASTLOGIN'
```

DBSCAN and its outliers:

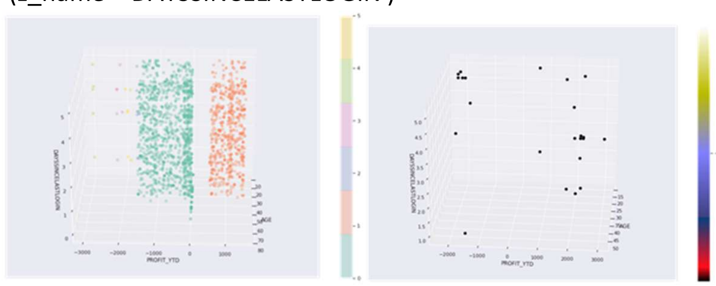
1. (z_name = 'DAYSSINCELASTTRADE')



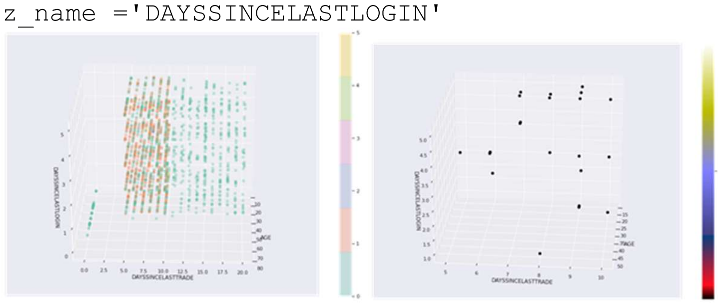
2. (z_name = 'TOTALUNITSTRADED')



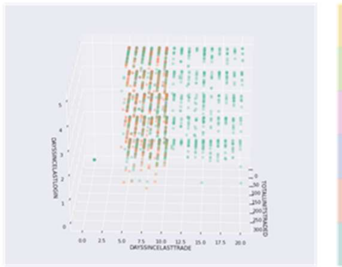
3. (z_name ='DAYSSINCELASTLOGIN')

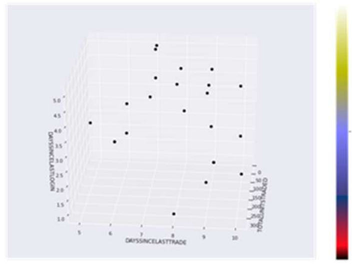


4.
```
x_name = 'AGE'
y_name = 'DAYSSINCELASTTRADE'
z_name ='DAYSSINCELASTLOGIN'
```



5.
```
x_name = 'TOTALUNITSTRADED'
y_name = 'DAYSSINCELASTTRADE'
z_name ='DAYSSINCELASTLOGIN'
```

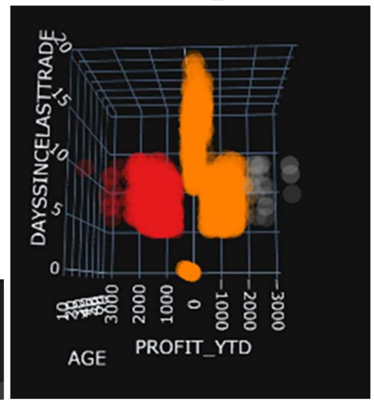




Agglomerative Clustering and its examples:

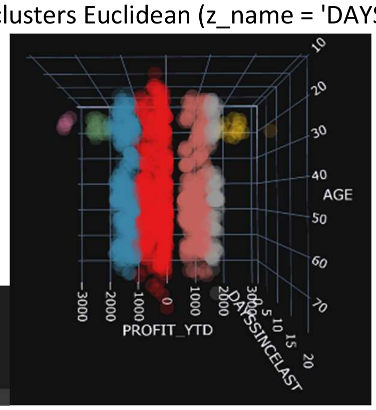
1. 2700 distance threshold for 3 clusters Euclidean (z_name = 'DAYSSINCELASTTRADE')

```
from sklearn.cluster import AgglomerativeClustering
ac = AgglomerativeClustering(n_clusters=None, distance_threshold=2700,
                             affinity='euclidean', linkage='complete')
predicted = ac.fit_predict(df_churn.values)

print('Number of clusters:', len(set(predicted)))

plot_dendrogram(ac, truncate_mode='level', p=3)

Number of clusters: 3
```

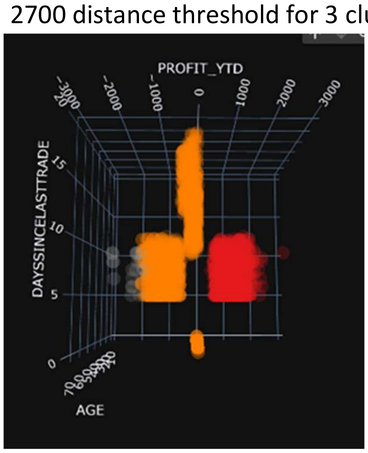


2. 1000 distance threshold for 8 clusters Euclidean (z_name = 'DAYSSINCELASTTRADE')

```
from sklearn.cluster import AgglomerativeClustering
ac = AgglomerativeClustering(n_clusters=None, distance_threshold=1000,
                             affinity='euclidean', linkage='complete')
predicted = ac.fit_predict(df_churn.values)

print('Number of clusters:', len(set(predicted)))

plot_dendrogram(ac, truncate_mode='level', p=3)

Number of clusters: 8
```

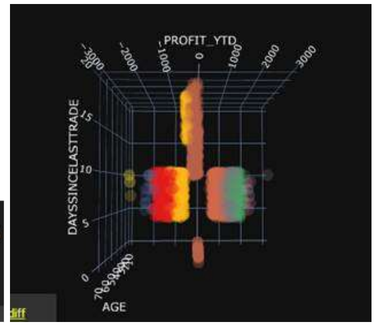


3. 2700 distance threshold for 3 clusters Manhattan (z_name = 'DAYSSINCELASTTRADE')

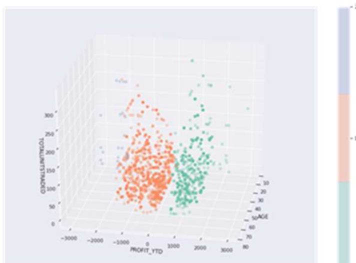


4. 1000 distance threshold for 10 clusters Manhattan (z_name = 'DAYSSINCELASTTRADE')

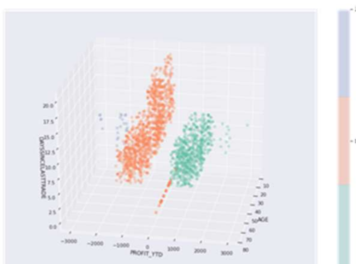
```
from sklearn.cluster import AgglomerativeClustering
ac = AgglomerativeClustering(n_clusters=None, distance_threshold=1000,
                             affinity='manhattan', linkage='complete')
predicted = ac.fit_predict(df_churn.values)

print('Number of clusters:', len(set(predicted)))

plot_dendrogram(ac, truncate_mode='level', p=3)

Number of clusters: 10
```



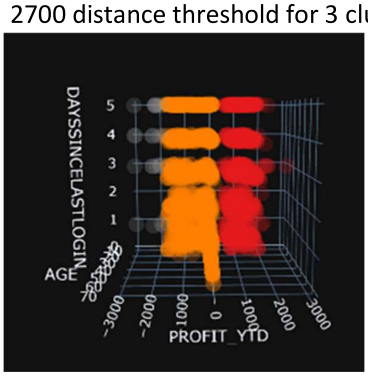
5. 2700 distance threshold for 3 clusters Euclidean (z_name = 'TOTALUNITSTRADED')



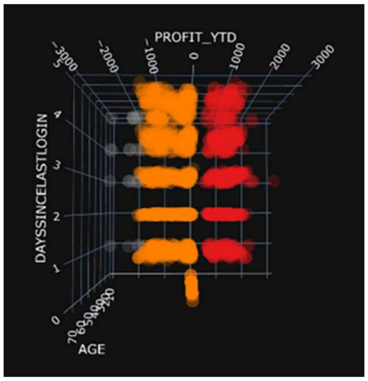
6. 2700 distance threshold for 3 clusters Manhattan (z_name = 'TOTALUNITSTRADED')



7. 2700 distance threshold for 3 clusters Euclidean (z_name ='DAYSSINCELASTLOGIN')

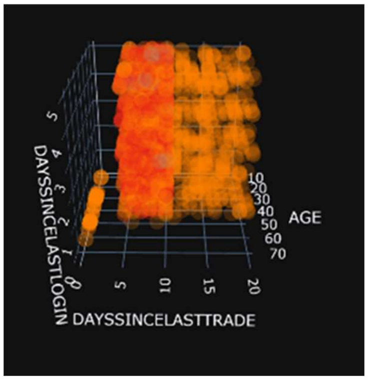


8. 2700 distance threshold for 3 clusters Manhattan (z_name ='DAYSSINCELASTLOGIN')

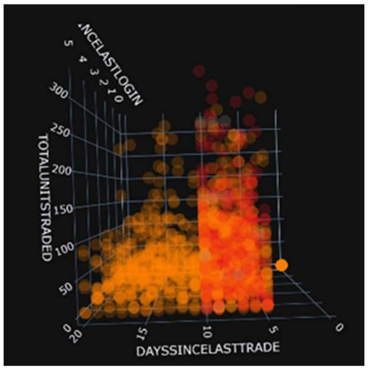


9. 2700 distance threshold for 3 clusters Manhattan

```
x_name = 'AGE'
y_name = 'DAYSSINCELASTTRADE'
z_name ='DAYSSINCELASTLOGIN'
```




```
x_name = 'TOTALUNITSTRADED'
y_name = 'DAYSSINCELASTTRADE'
z_name ='DAYSSINCELASTLOGIN'
```



Research Sources:

Original Collab:
https://colab.research.google.com/drive/1tM81sZl8yCmlKhkpre_xwuz1Mu2AdeWY?usp=sharing

Link to my own collab: https://colab.research.google.com/drive/12jSF0BQ2Embr_V24-gupI6GnEZpRhQx_#scrollTo=SmxVHTrScXo3

Link to some research sources:

https://towardsdatascience.com/k-means-clustering-with-scikit-learn-6b47a369a83c

https://towardsdatascience.com/breaking-down-the-agglomerative-clustering-process-1c367f74c7c2

https://towardsdatascience.com/ai-cluster-analysis-of-categorical-data-part-ii-47f3a13601a2

Feedback:

This report was quite enjoyable, easy to complete and informative, not as intensive as the report given by prof. Paweł Jemioło. Favorite part was the color compression task. All exercises were clear in requirements and there were not too many subtasks or optional tasks to complete. Working in collab overall again, a very positive experience. Thank you for the report tasks, presentation, and checking.