Data Mining: case study

Agenda

Introduction

Business Understanding

Data Understanding

Data Preparation

Clustering analysis

Association rules

Business Understanding

Business objective

We want to analyze water meters located in NYC in during 2012-2020. Our goal is to understand patterns related to them in order to, possibly enter this segment of market.

Data mining goals

1) Discover segments in meter data
2) Analyze the semantic of these segment, trying to find interesting patterns

Data Understanding

Data collection and quality

1) Data has been collected from a **Kaggle** dataset that stored the [2012-2020] version of the water consumption dataset of NYC Open data
2) The dataset presented 24 fields, each one of them is explained in the file dict.xlsx
3) The real problem was about **completeness**: there were missing values for meters numbers and their location
4) Another problem was that the **unit of analysis** of the dataset was a specific **water consumption**, and **not the specific meter** we want to analyze.
5) An early exploration showed many redundancies and unbalanced categorical features, as long as many of them representing ids *(useless)*
6) When Boroughs = FHA after a brief search I discovered that related developments are part of QUEENS administration but under control of FHA

Data Preparation

Pre-processing strategy

1) It’s obvious that if we want to analyze meters, we **must aggregate data by its id**
2) For every different meters we want to get overall:
    1) Consumptions
    2) Charges
    3) Total number of days in which the meter is active
    4) First and last service date
    5) And others

Pre-processing strategy

It is not possible to aggregate if missing features aren't handled. Most of them like TDS #, and rate class are handled with a simple imputation.

Imputing meter number and location was a bit difficult. Exploring I understood that **every different meter should be placed on a different location** that could be obviously served by different meters.

- I used this relationship to impute meters.
- **Assumpion**: since there is no other way to be sure about how many different locations are marked with a NaN, we will assume that in this cases a single building instance is associated with a single meter.

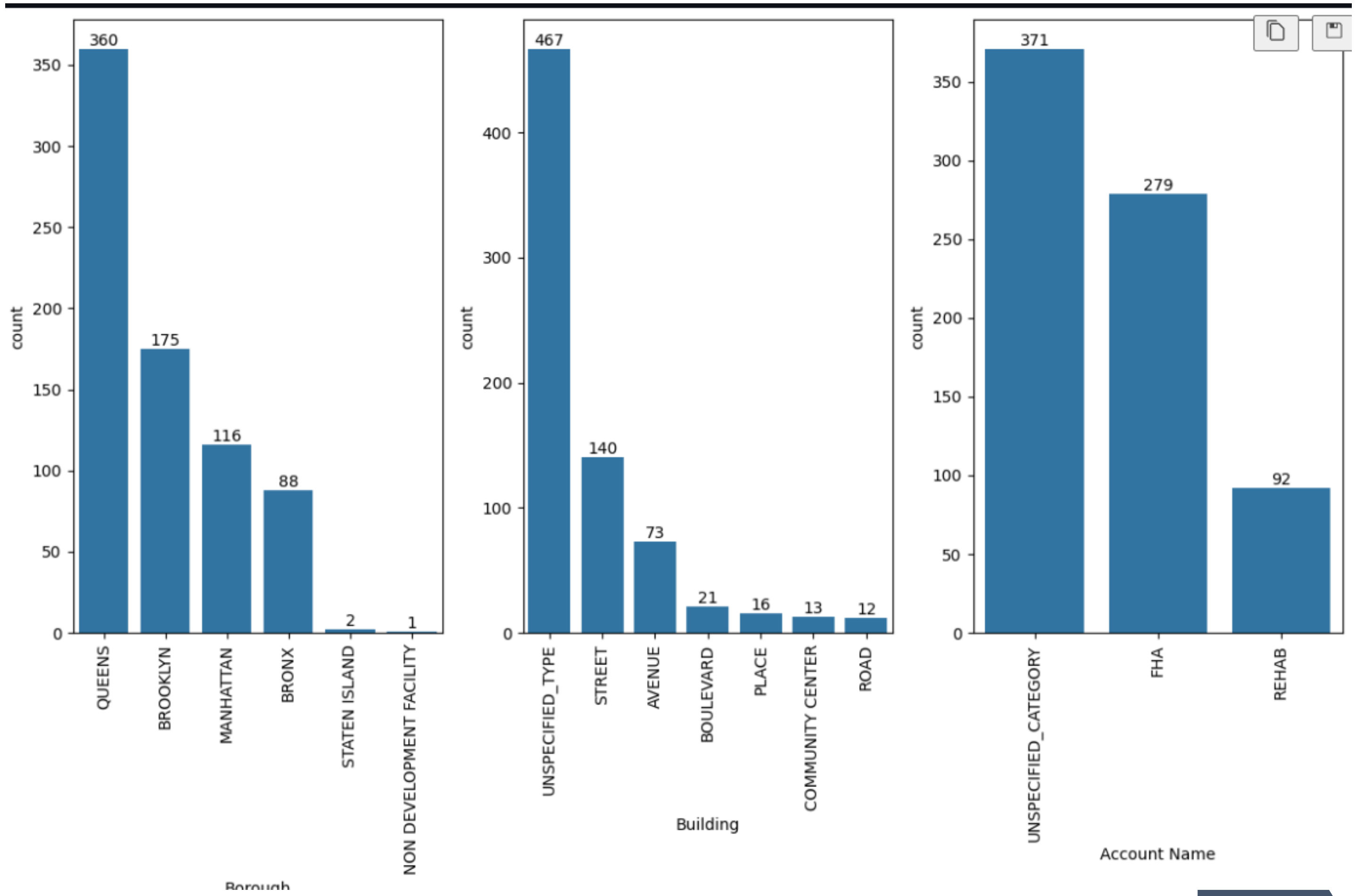
Pre-processing strategy: features

- A good number of features like **AMP #**, **EDP** were removed since TDS # was already enough to identify a development;
- Other features with loads of category with low value counts (with a single dominant category) have been turned to Boolean
- Charges are divided in Current, Water&Sewer and Others, where Current is the sum of the last two. I removed the second one, using Others as a Boolean indicator (since there were not much values)

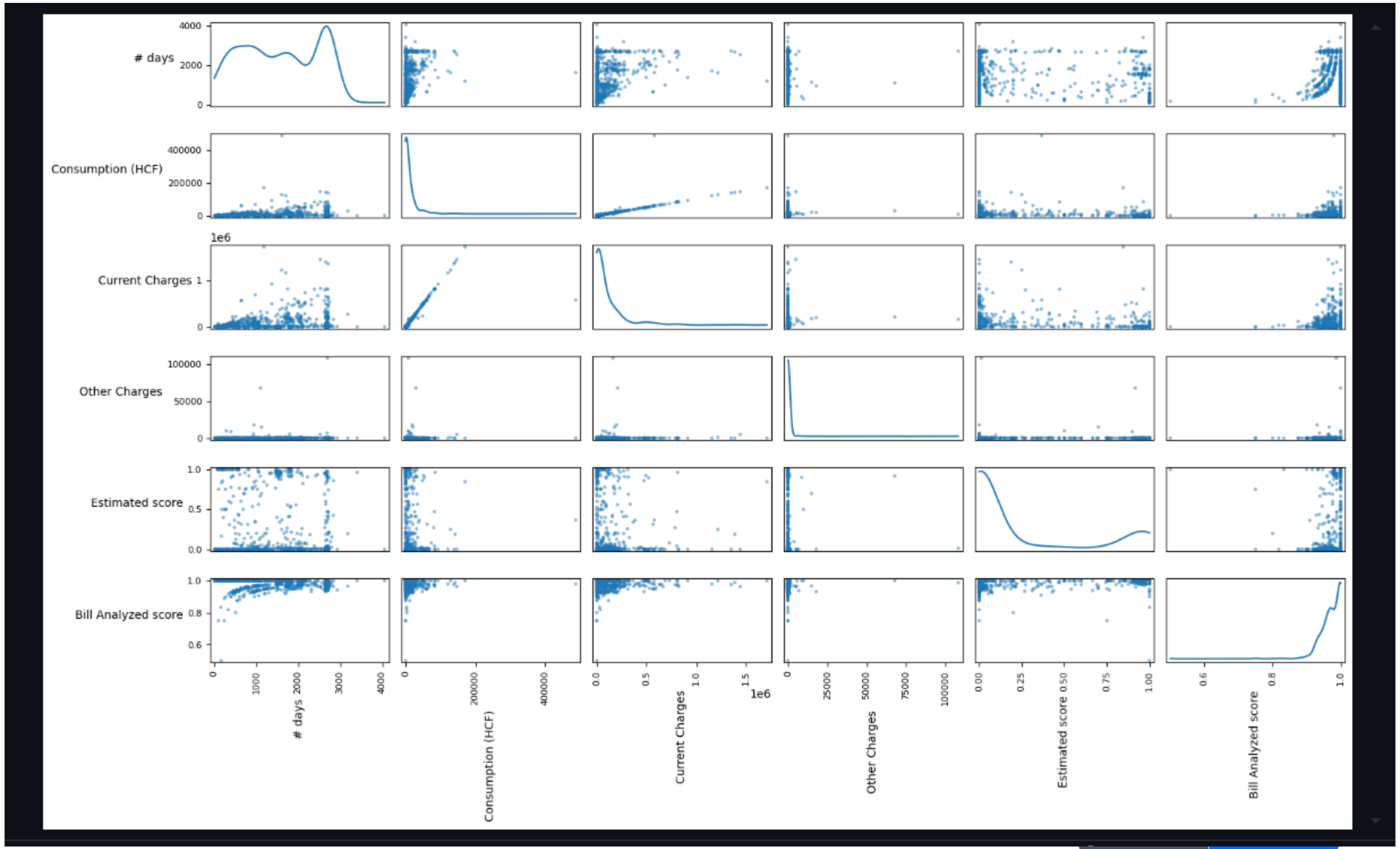
Pre-processing strategy: pipeline

- Using the base dataset impute missing features
- Aggregate the dataset by meter number
- Engineer **new features** for the aggregated dataset (strategy are reported on the pre-processing tab of dict.xlsx):
    - Estimated score = n_estimated / n_readings
    - Bill analyzed score = n_analysis / n_readings
    - Building = TDS# +Location
    - Location category
    - Development type

After the aggregation...



After the aggregation...



Data encoding for clustering

- For clustering purposes the data needs to be numeric. Since **One hot encoding would have resulted in a very sparse representation** i opted for Count Encoding from *feature_engine* library.
- How to treat dates?
    - Divide each date field in day month year (derive three features)
    - Apply a CyclicEncoding on month and day to preserve their structure
    - The last step will derive two features for day and two features for month based on *sin* and *cos* transformations

Data encoding for association rules

- To discover frequent pattern every instance on the original pre processed dataset needs to be One Hot Encoded (categorical)
- For numerical features i defined a number of bins in **equal frequency**. I decided 5 bins for *consumptions* and *charges* features, and 3 bins for *#_days*
- Only the scores could not be handled this way. So i used the classic equal-width tecnique with Friedman rule to decide the correct number of bins
- After binning i applied the classic One hot encoding

Clustering

Outliers?

- After the aggregations the number of instances was quite low: 742. I decided to keep outliers, to not reduce even further instances' number
- Instead i opted for scikit **RobustScaler** for the **heavy tailed** numerical distributions

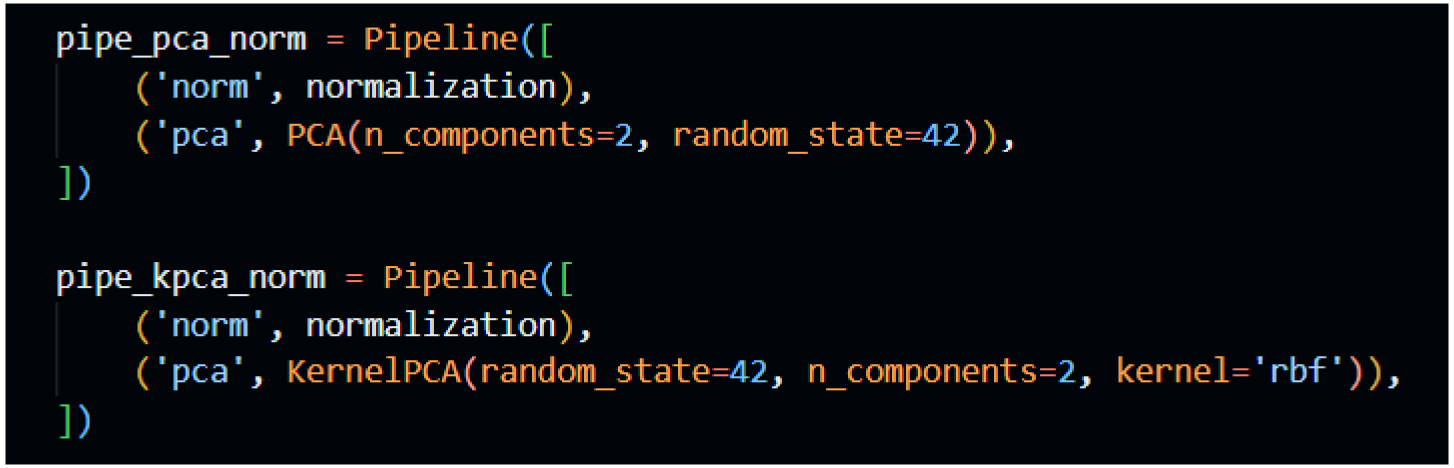
Clustering types

- I tried with different configurations:
  - **K-Means** with and without dimensionality reduction
  - **Agglomerative single linkage** and **ward linkage** with and without dimensionality reduction
  - **DBSCAN** with non-linear dimensionality reduction

Dimensionality reduction

- When applied clustering with dimensionality reduction i applied two tecniques:
  - **PCA** (n_components = 2) => it explain over 99% of the data variance
  - **Kernel PCA** (n_components = 2, kernel = rbf, gamma=1/n_features) => better clustering results

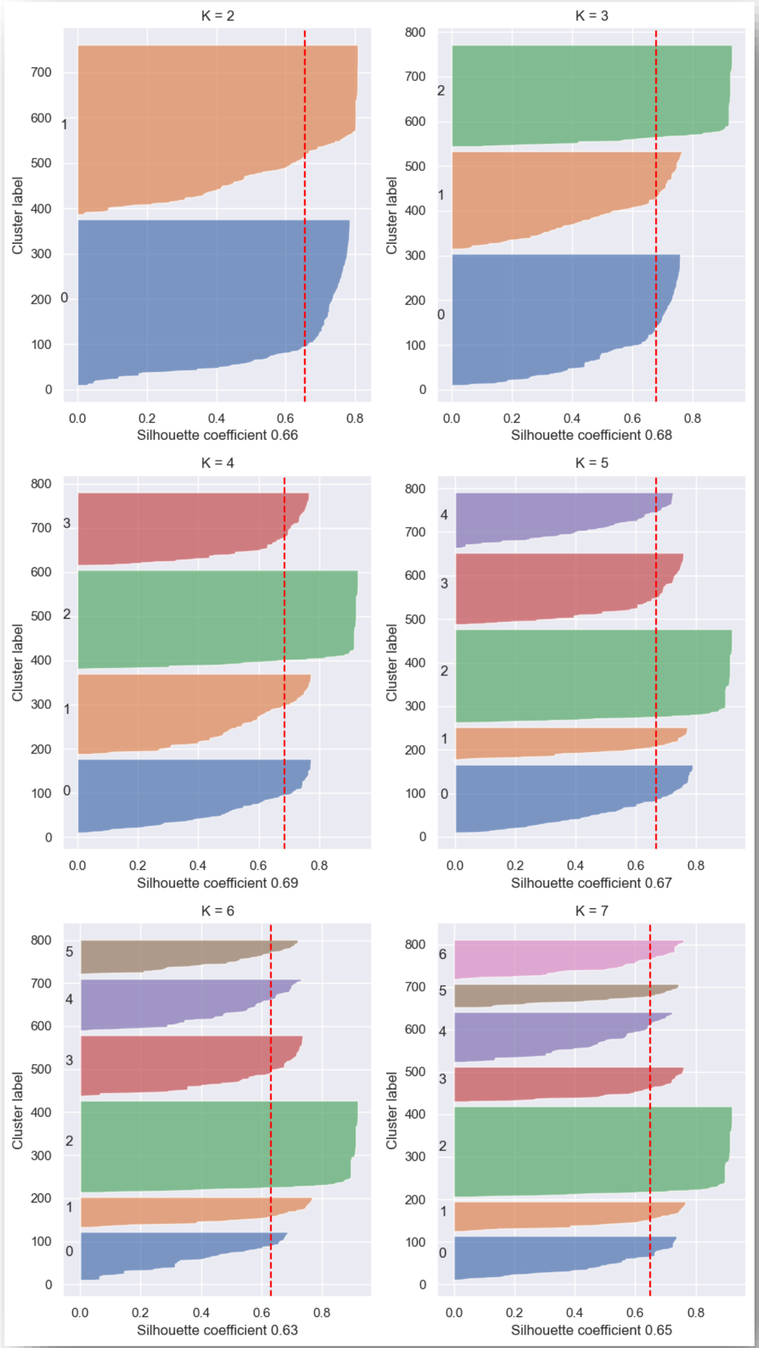
Pipelines



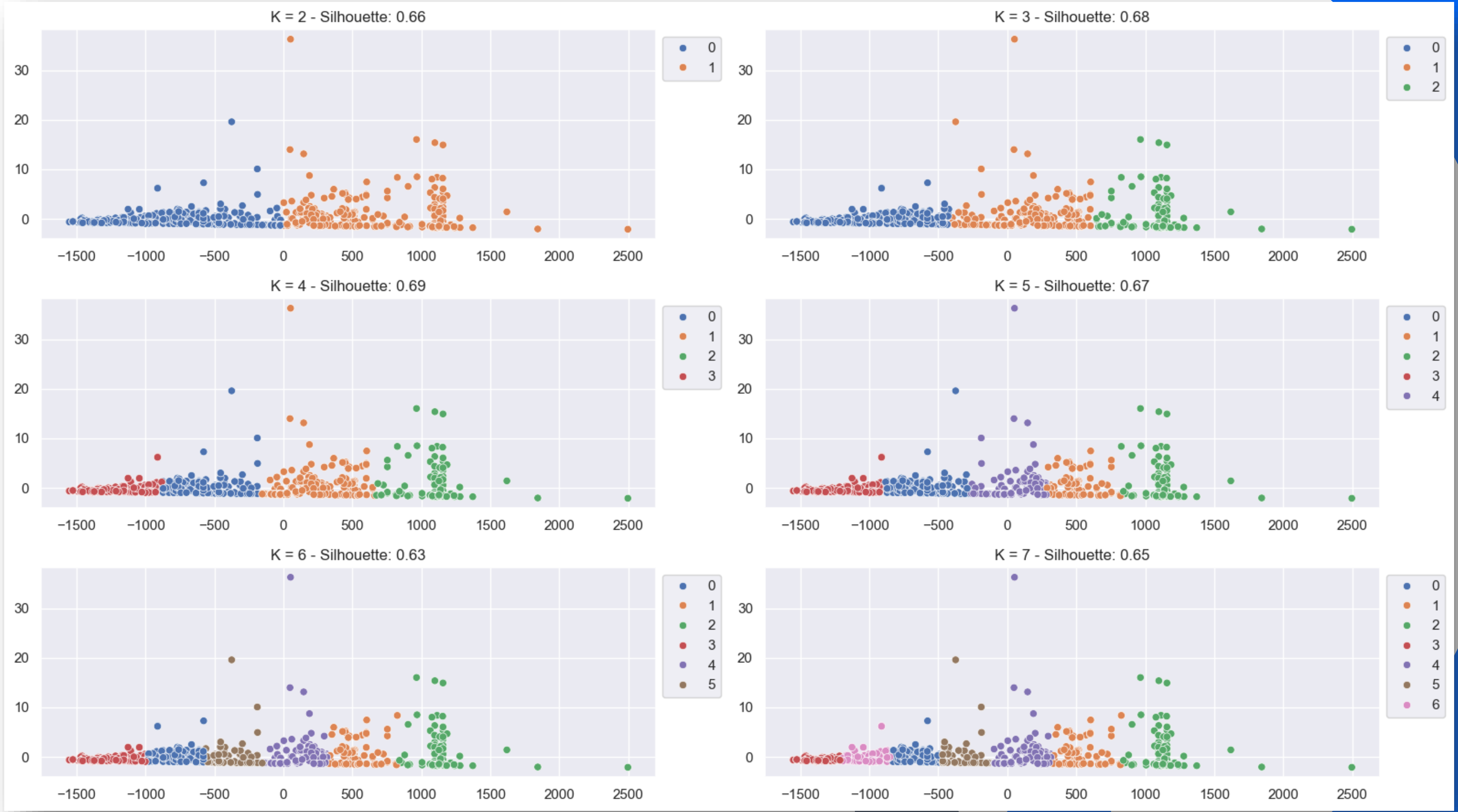
```
pipe_pca_norm = Pipeline([
    ('norm', normalization),
    ('pca', PCA(n_components=2, random_state=42)),
])

pipe_kpca_norm = Pipeline([
    ('norm', normalization),
    ('pca', KernelPCA(random_state=42, n_components=2, kernel='rbf')),
])
```

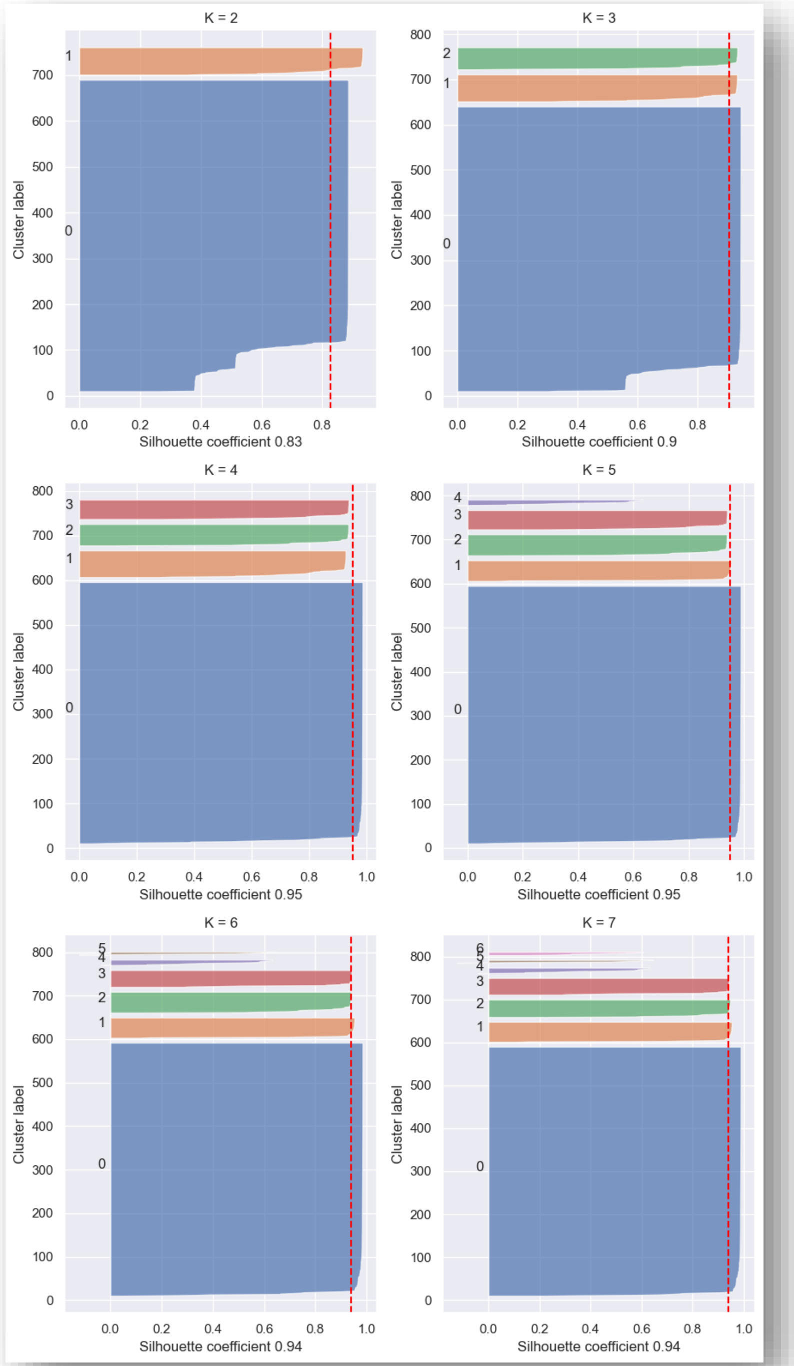
K Means PCA


K = 2
Silhouette coefficient 0.66
K = 3
Silhouette coefficient 0.68
K = 4
Silhouette coefficient 0.69
K = 5
Silhouette coefficient 0.67
K = 6
Silhouette coefficient 0.63
K = 7
Silhouette coefficient 0.65
Cluster label

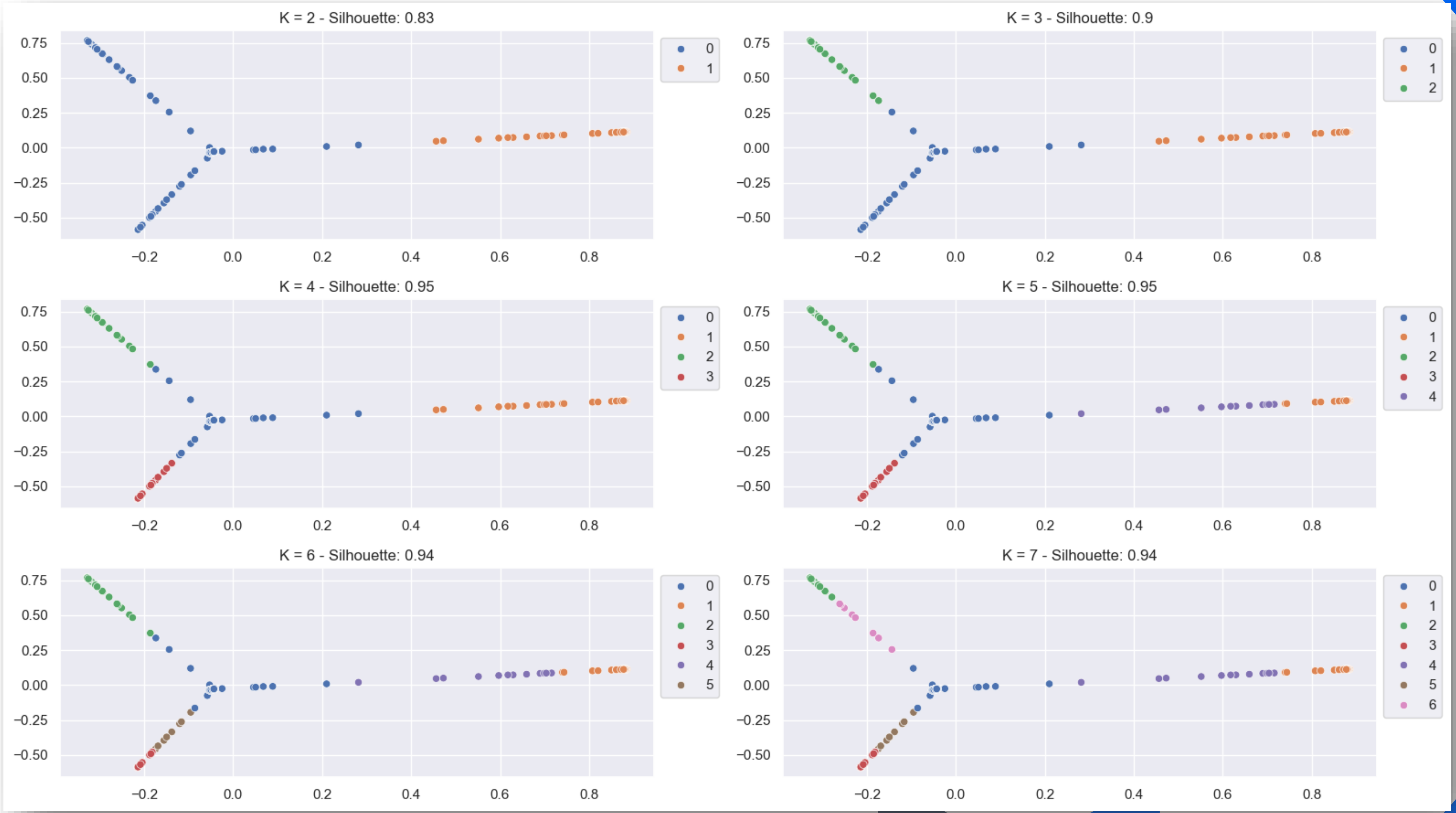
K Means PCA


K = 2 - Silhouette: 0.66
K = 3 - Silhouette: 0.68
K = 4 - Silhouette: 0.69
K = 5 - Silhouette: 0.67
K = 6 - Silhouette: 0.63
K = 7 - Silhouette: 0.65
0
1
2
3
4
5
6
−1500
−1000
−500
0
500
1000
1500
2000
2500
0
10
20
30

K Means Kernel PCA



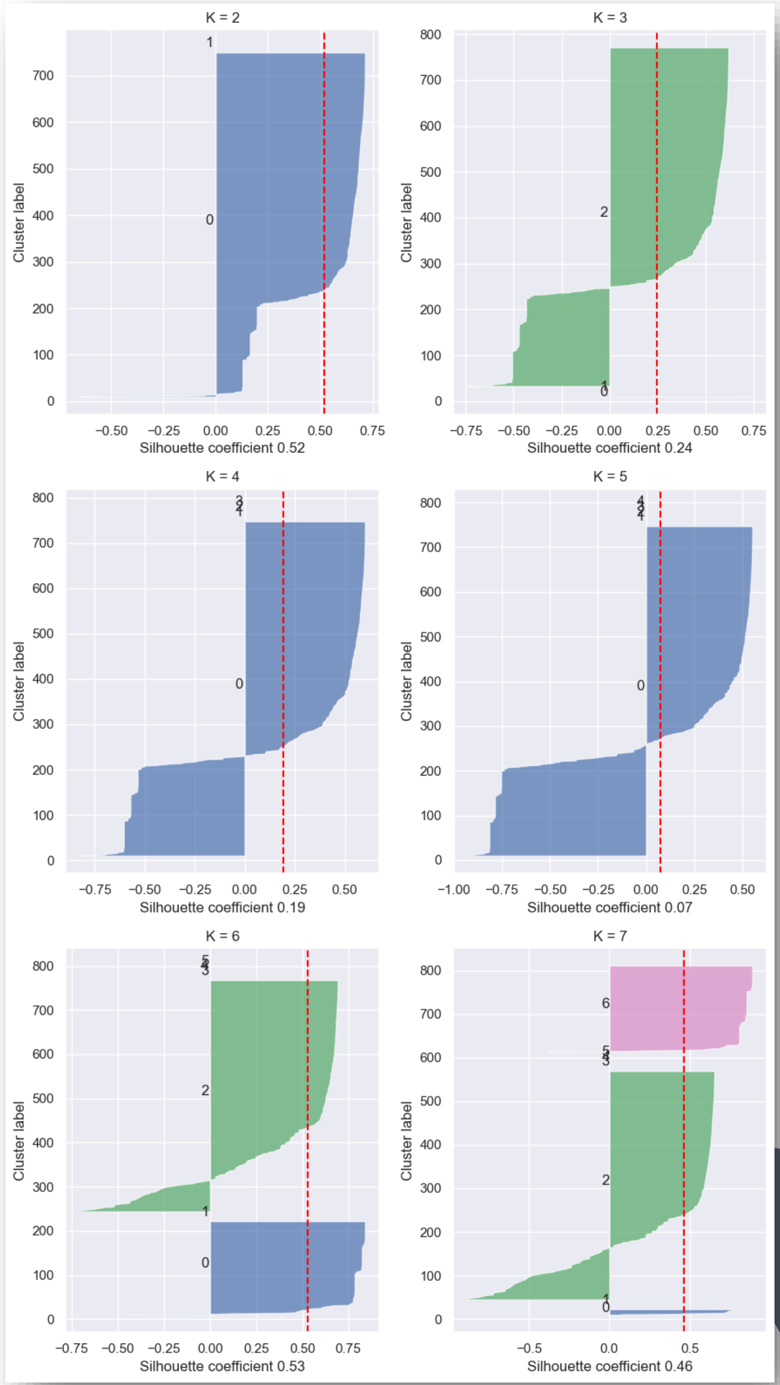
K Means Kernel PCA



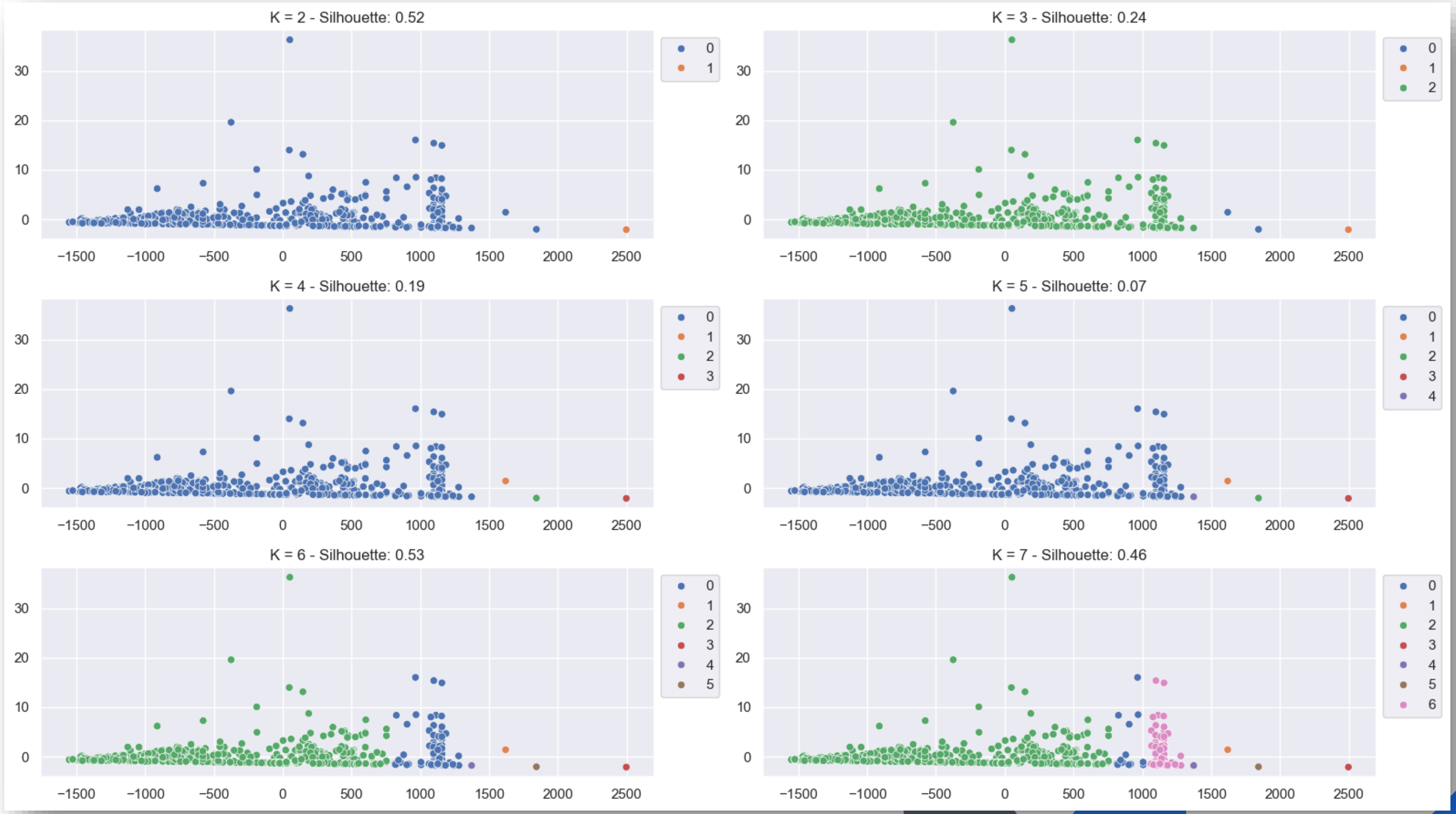
Comments

1) Kernel PCA looks to better separate data points, but visually inspecting clusters it looks like there is some kind of mixing in the “clouds”
2) Applying K-Means with plain PCA, results in more **balanced** clusters, that look likes subgroup of a bigger single one

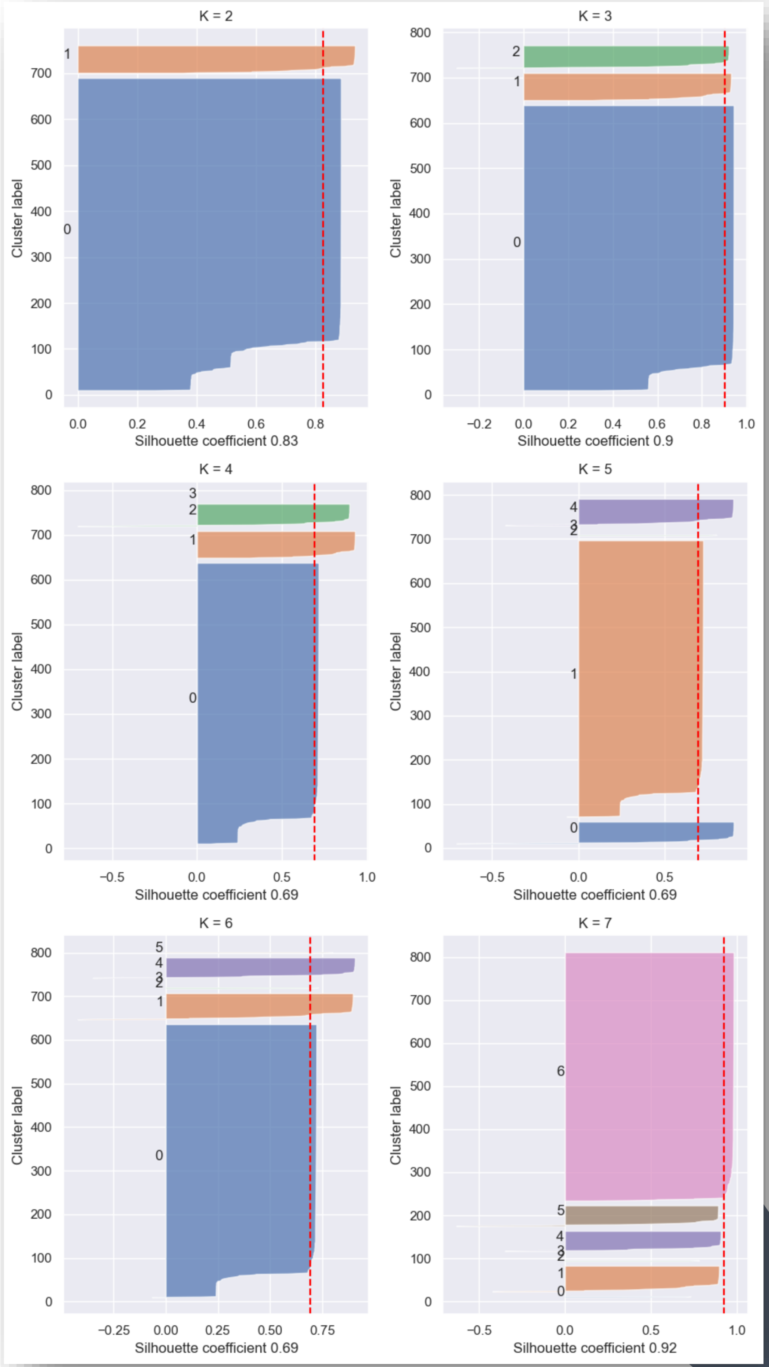
Agglomerative Single linkage PCA



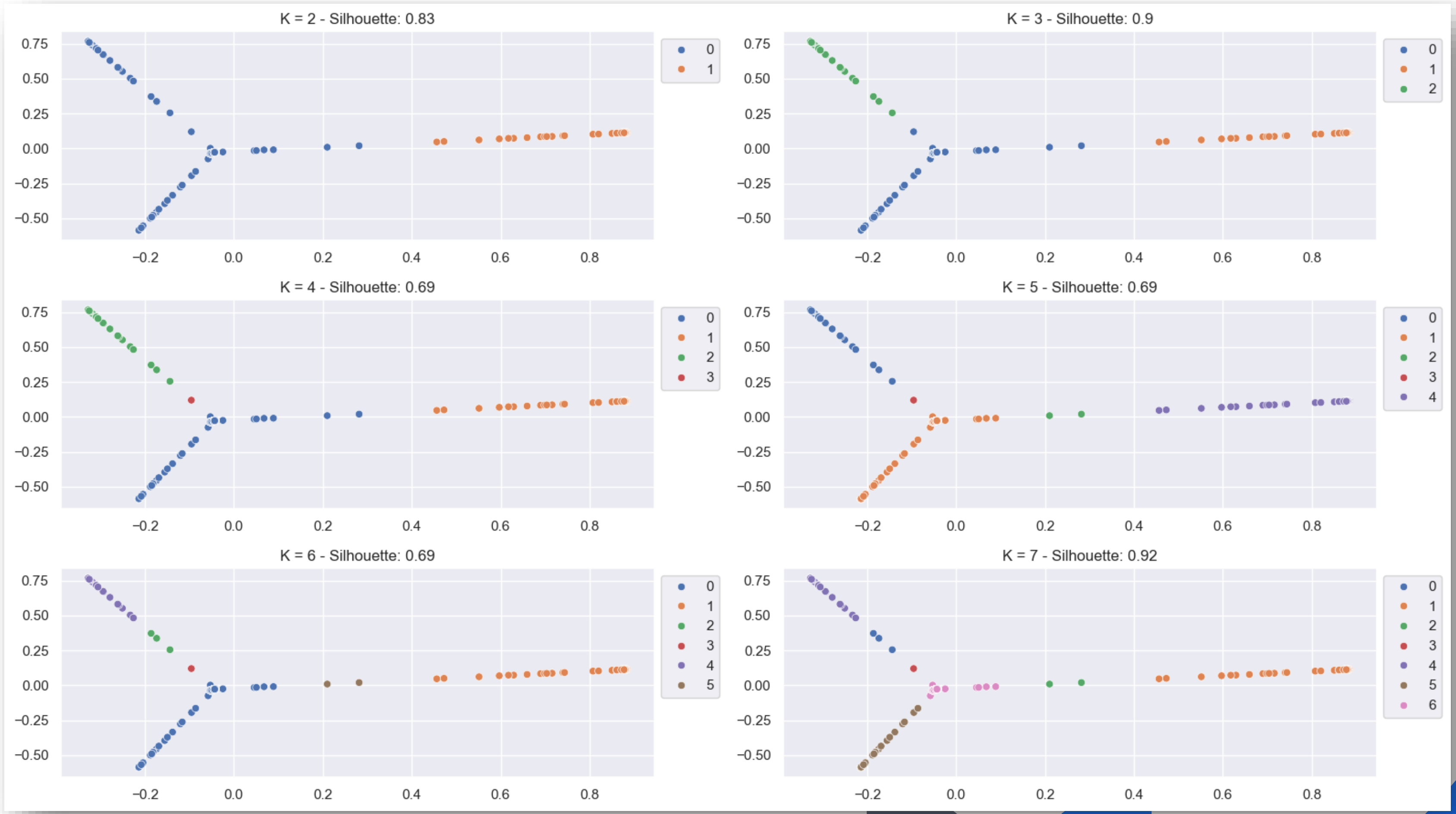
Agglomerative Single linkage PCA



Agglomerative Single linkage Kernel PCA



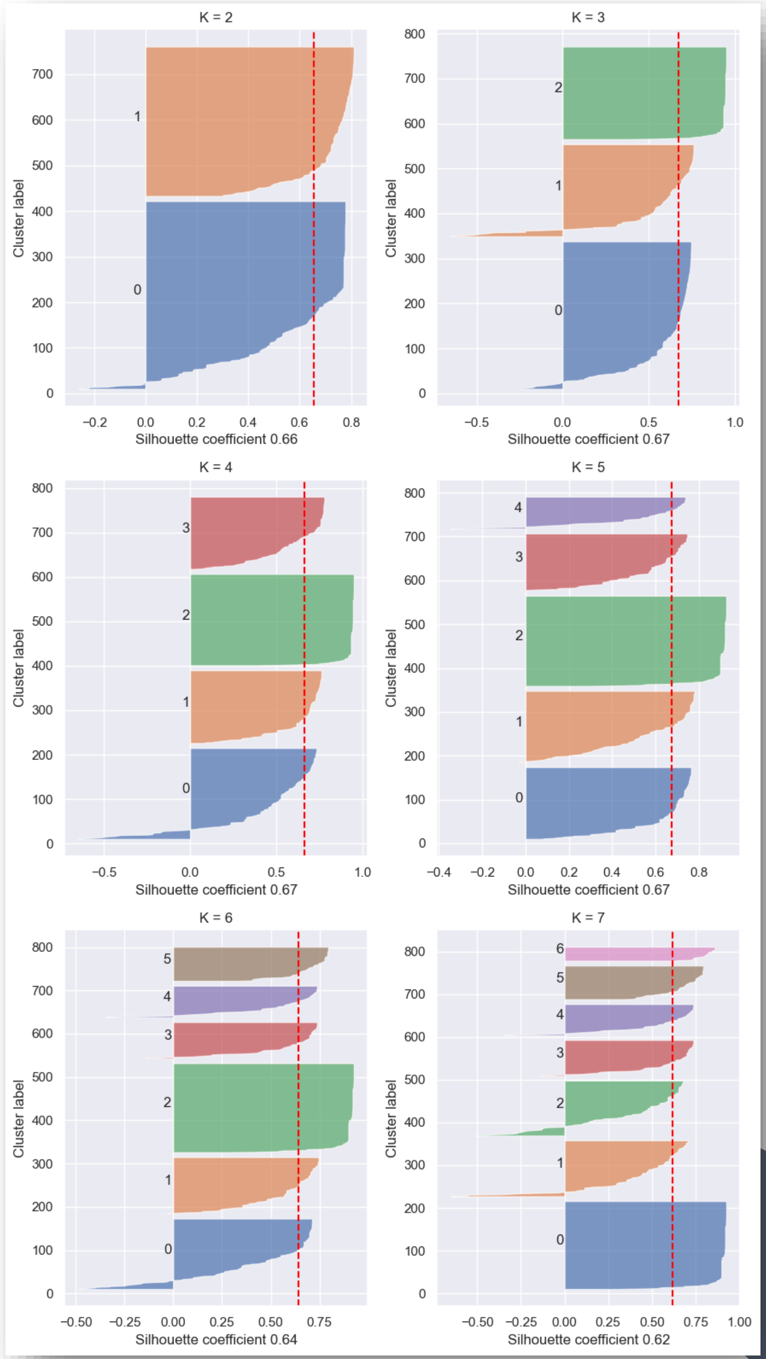
Agglomerative Single linkage Kernel PCA



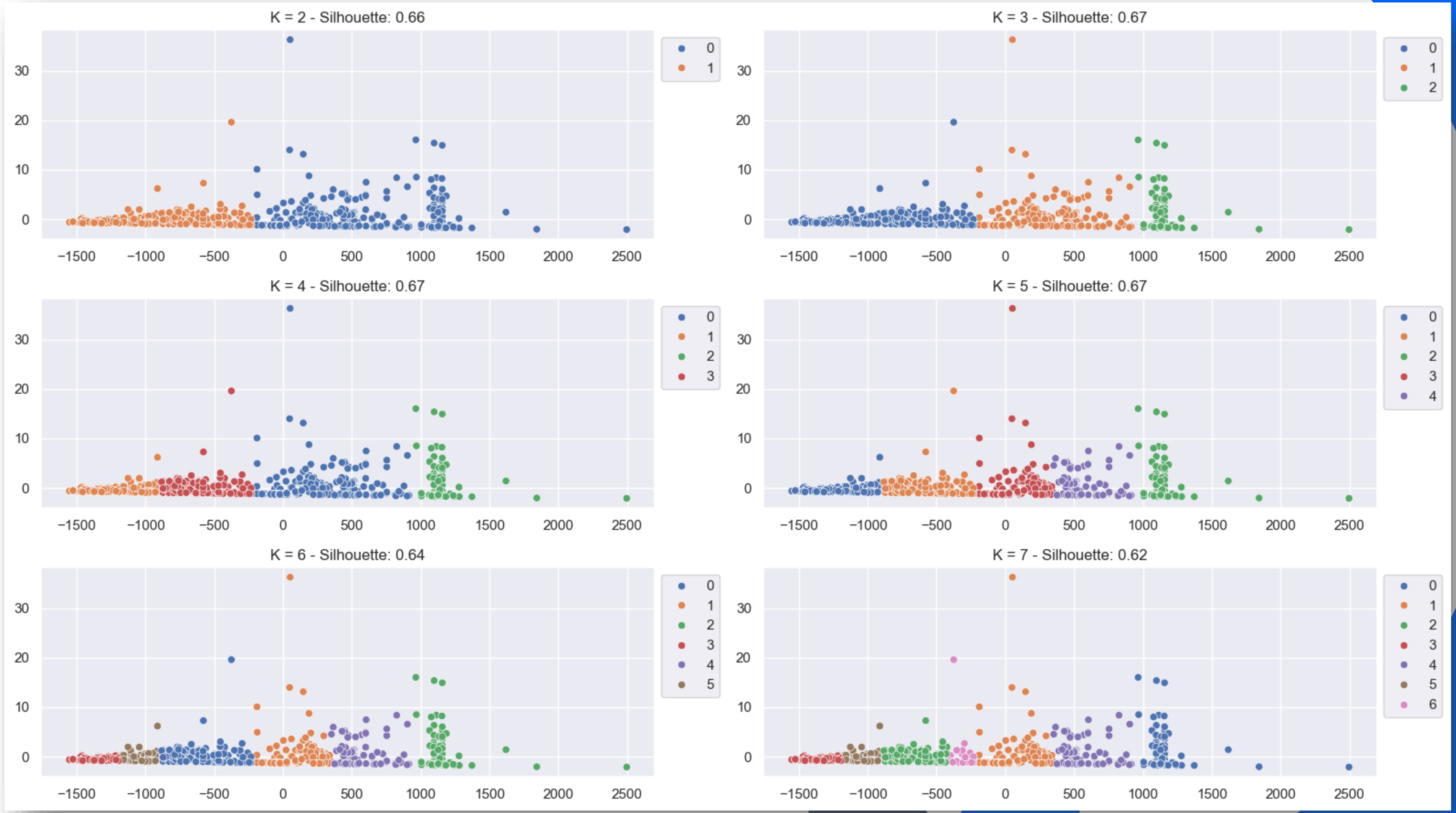
Comments

1) Single linkage performs very poorly
2) Data is noisy and single linkage is prone to ouliers, in fact creates very unbalanced clusters, and in PCA case it creates a single cluster very often

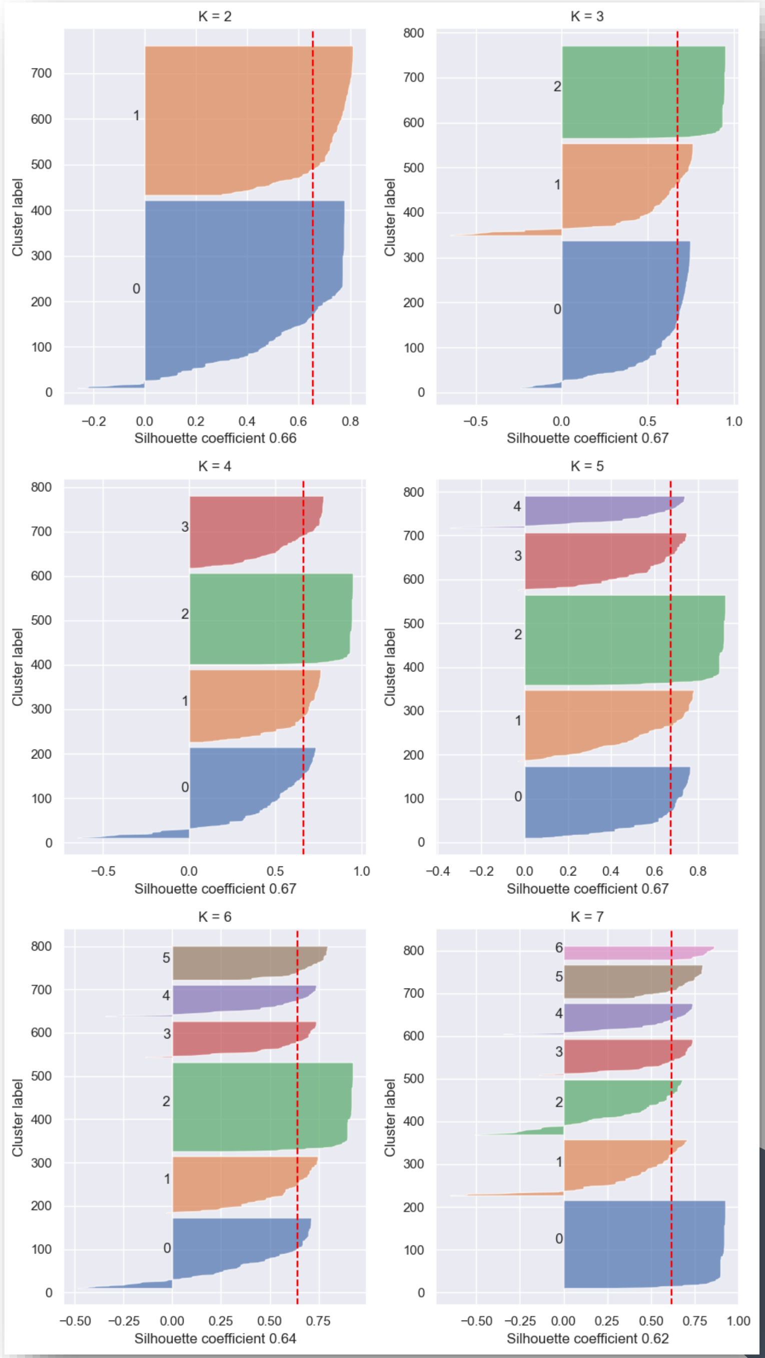
Agglomerative Ward linkage PCA



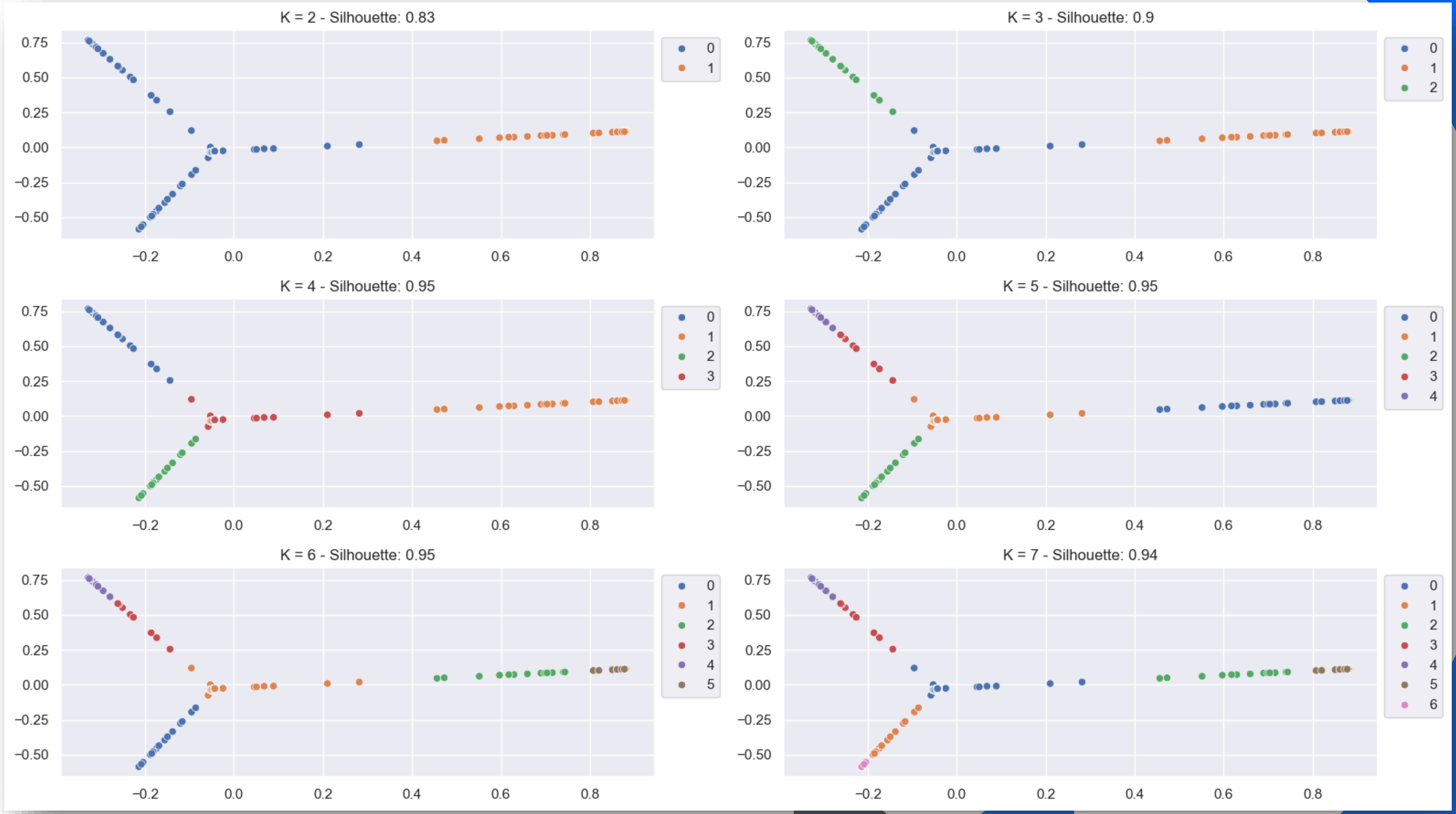
Agglomerative Ward linkage PCA



Agglomerative Ward linkage Kernel PCA



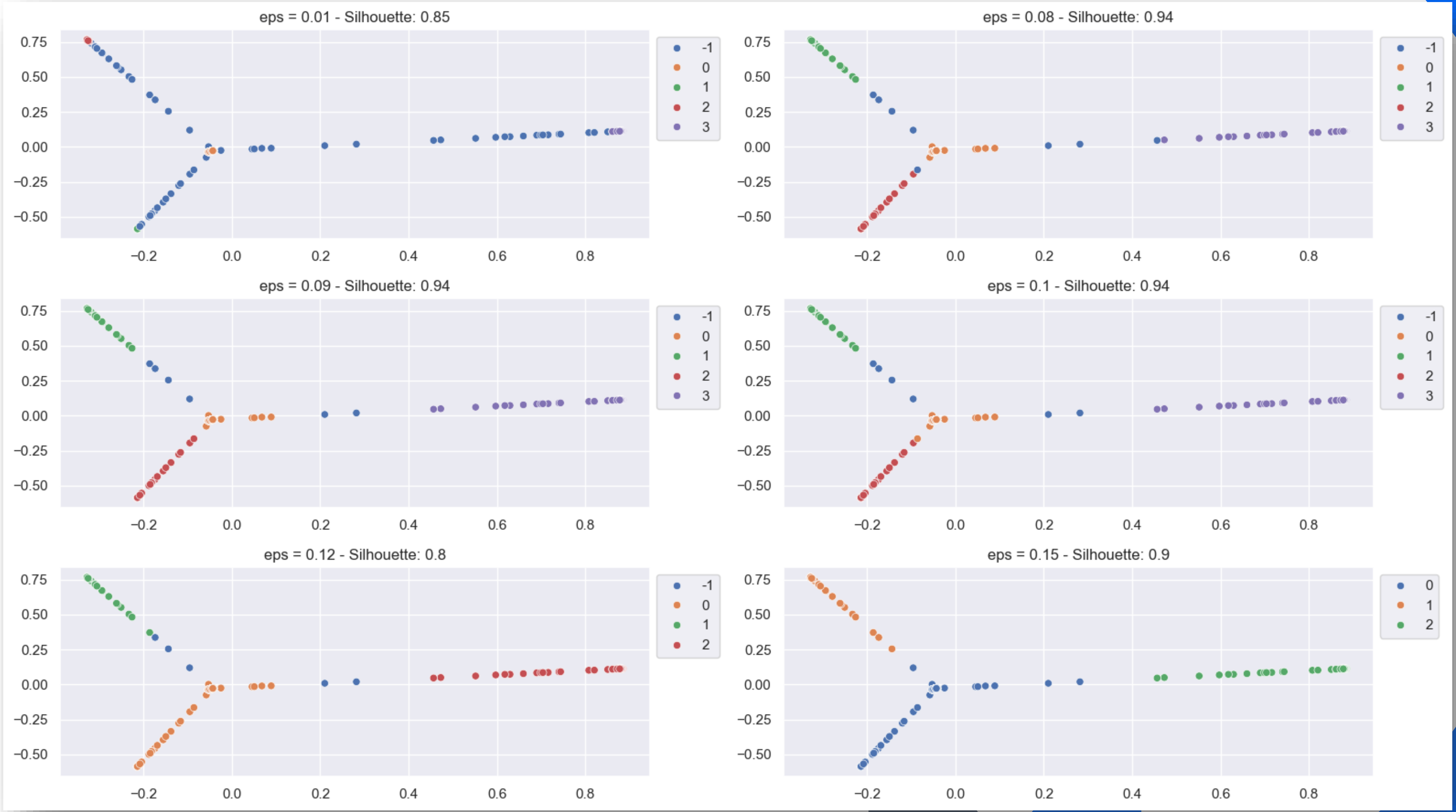
Agglomerative Ward linkage Kernel PCA



Comments

1) Agglomerative clustering with ward linkage seems more stable and provides good results with the kernel pca
2) Observing data points' plots and cosindering K=4, Ward clustering provides less mixing than 4-Means using the kernel pca approach, even though the silhoutte score for K = 4 is the the same in both K-Means and Ward the latter;
3) **Silhoutte coefficient** is an **intuitive internal validation criteria**, but like every other methods of the same category, is not good to compare different algorithms, since results will be biased towards a particular one. At least the absolute values gives some insights. But in this case for example a visual analysis were required to discriminate the best tecnique.*

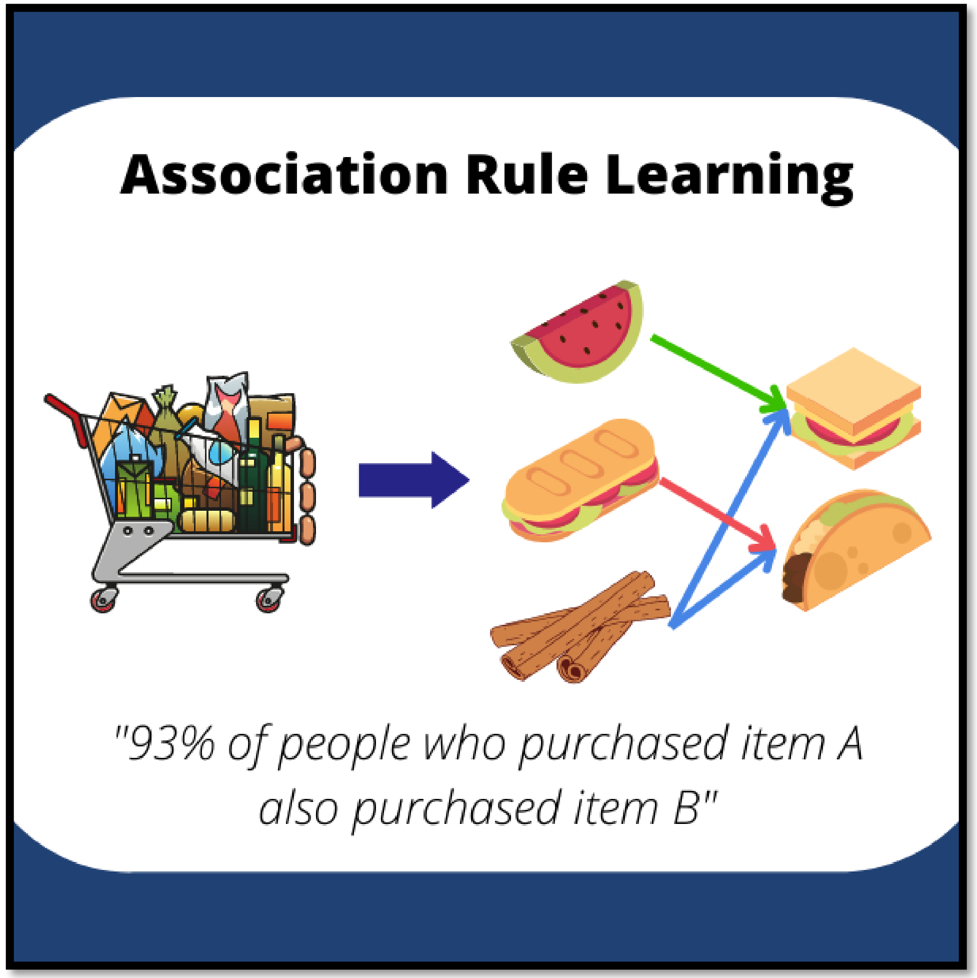
DBSCAN Kernel PCA



Comments

1) Applying DBSCAN on the kernel pca dataset, provides good results in terms of both silhoutte score and seprations. In particular with eps = 0.09. The advantage in using this tecnique is the fact that noise points are detected automatically. Since no outlier are removed, this property *could be really useful*.

Association rules


Association Rule Learning
"93% of people who purchased item A also purchased item B"

Why association rules

Note: I'll report the most interesting ones. Using VS code data wrangler tools other rules can be inspected. I will no

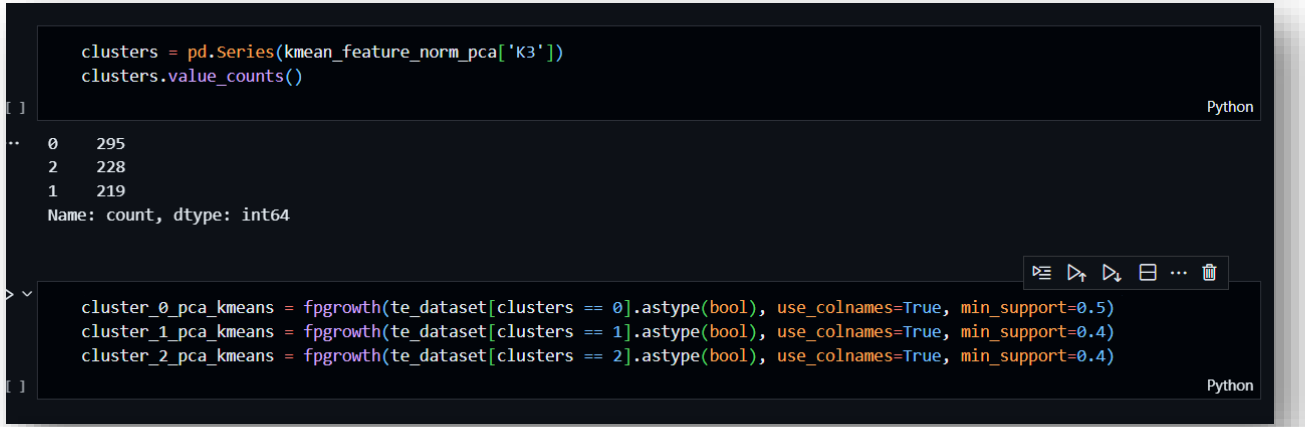
1) Understand cluster semantics
2) Trying to understand recurring and interesting patterns in our data

I'll use FPGROWTH as **frequent pattern mining** algorithm, lowering or increasing min_sup depending on how many (useful) patterns are retrieved.

The association rule will be extracted for the following cases:

1) 3-Means with PCA (here will be provided a very short description)
2) Agglomerative clustering with Ward linkage and with Kernel PCA

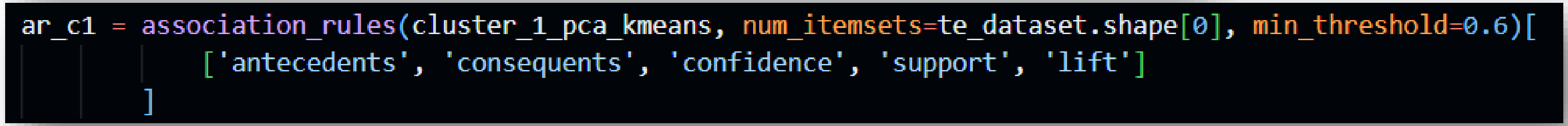
3 Means PCA



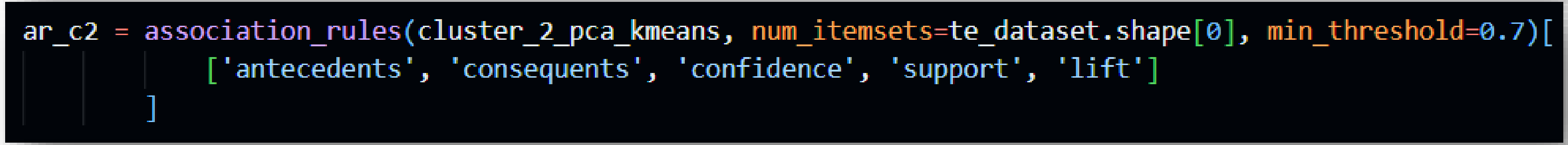
```
clusters = pd.Series(kmean_feature_norm_pca['K3'])
clusters.value_counts()
```

```
0    295
2    228
1    219
Name: count, dtype: int64
```

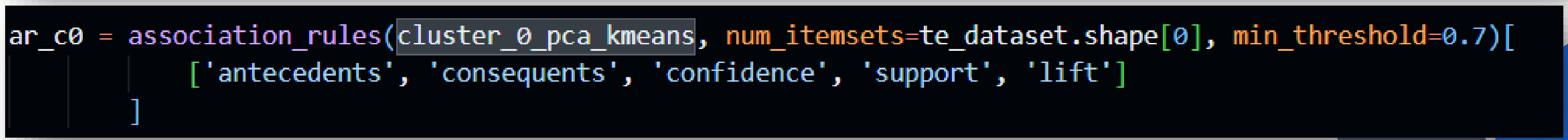
```
cluster_0_pca_kmeans = fpgrowth(te_dataset[clusters == 0].astype(bool), use_colnames=True, min_support=0.5)
cluster_1_pca_kmeans = fpgrowth(te_dataset[clusters == 1].astype(bool), use_colnames=True, min_support=0.4)
cluster_2_pca_kmeans = fpgrowth(te_dataset[clusters == 2].astype(bool), use_colnames=True, min_support=0.4)
```



```
ar_c1 = association_rules(cluster_1_pca_kmeans, num_itemsets=te_dataset.shape[0], min_threshold=0.6)[
            ['antecedents', 'consequents', 'confidence', 'support', 'lift']
        ]
```

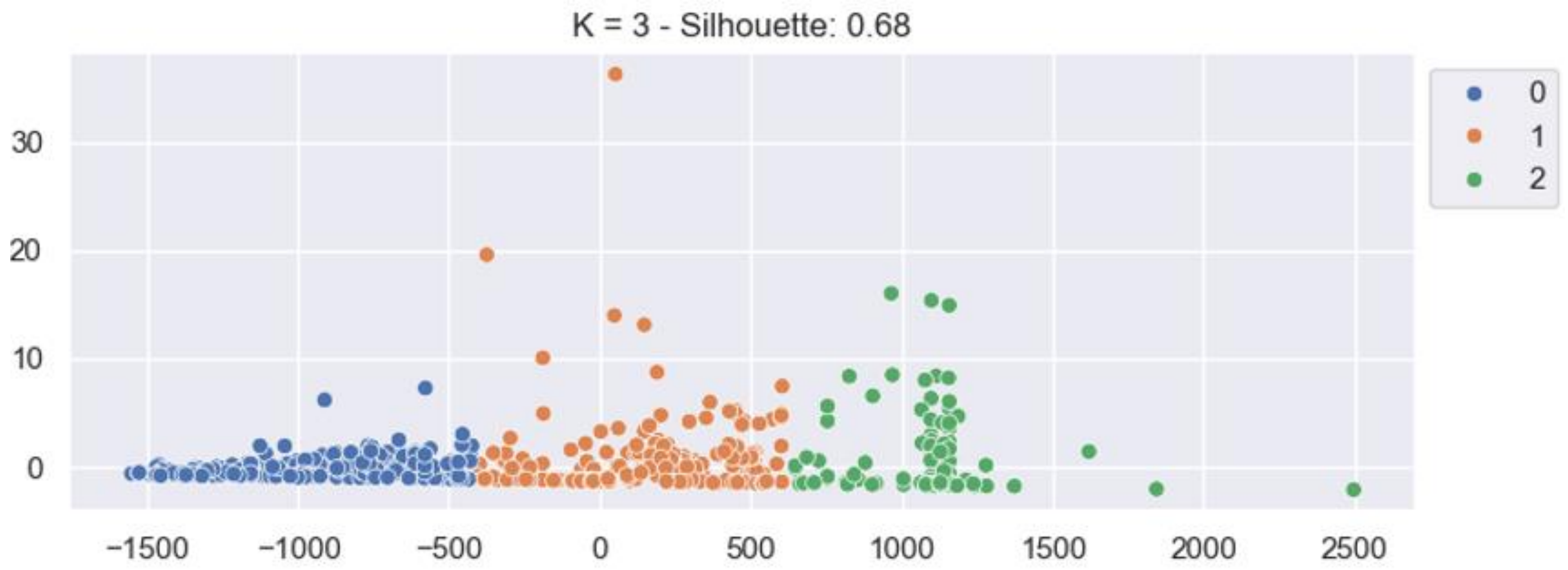


```
ar_c2 = association_rules(cluster_2_pca_kmeans, num_itemsets=te_dataset.shape[0], min_threshold=0.7)[
            ['antecedents', 'consequents', 'confidence', 'support', 'lift']
        ]
```



```
ar_c0 = association_rules(cluster_0_pca_kmeans, num_itemsets=te_dataset.shape[0], min_threshold=0.7)[
            ['antecedents', 'consequents', 'confidence', 'support', 'lift']
        ]
```

3 Means PCA

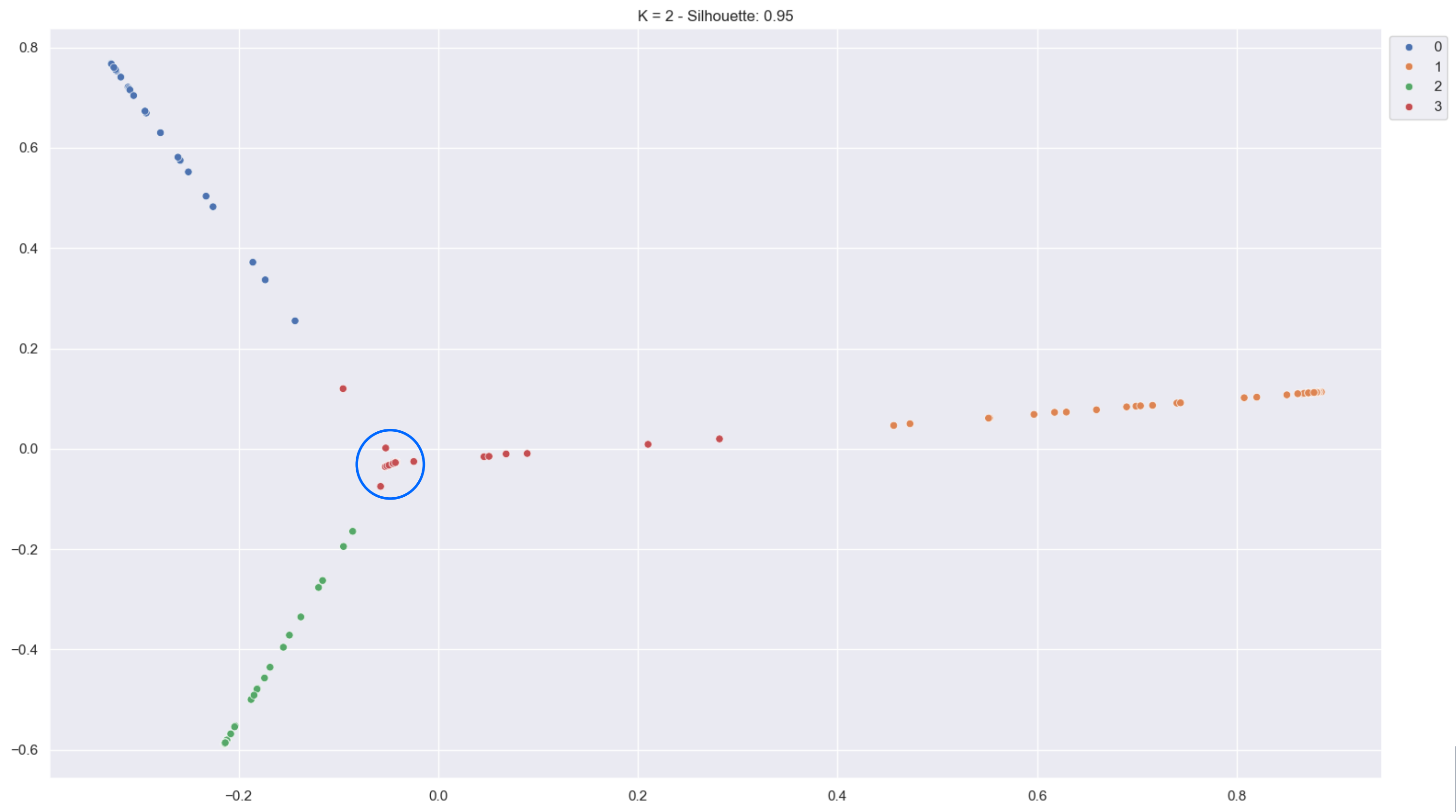

K = 3 - Silhouette: 0.68
30
20
10
0
−1500
−1000
−500
0
500
1000
1500
2000
2500
0
1
2

Comments

I won't show specific association rules, since the one found are very similar for every cluster.

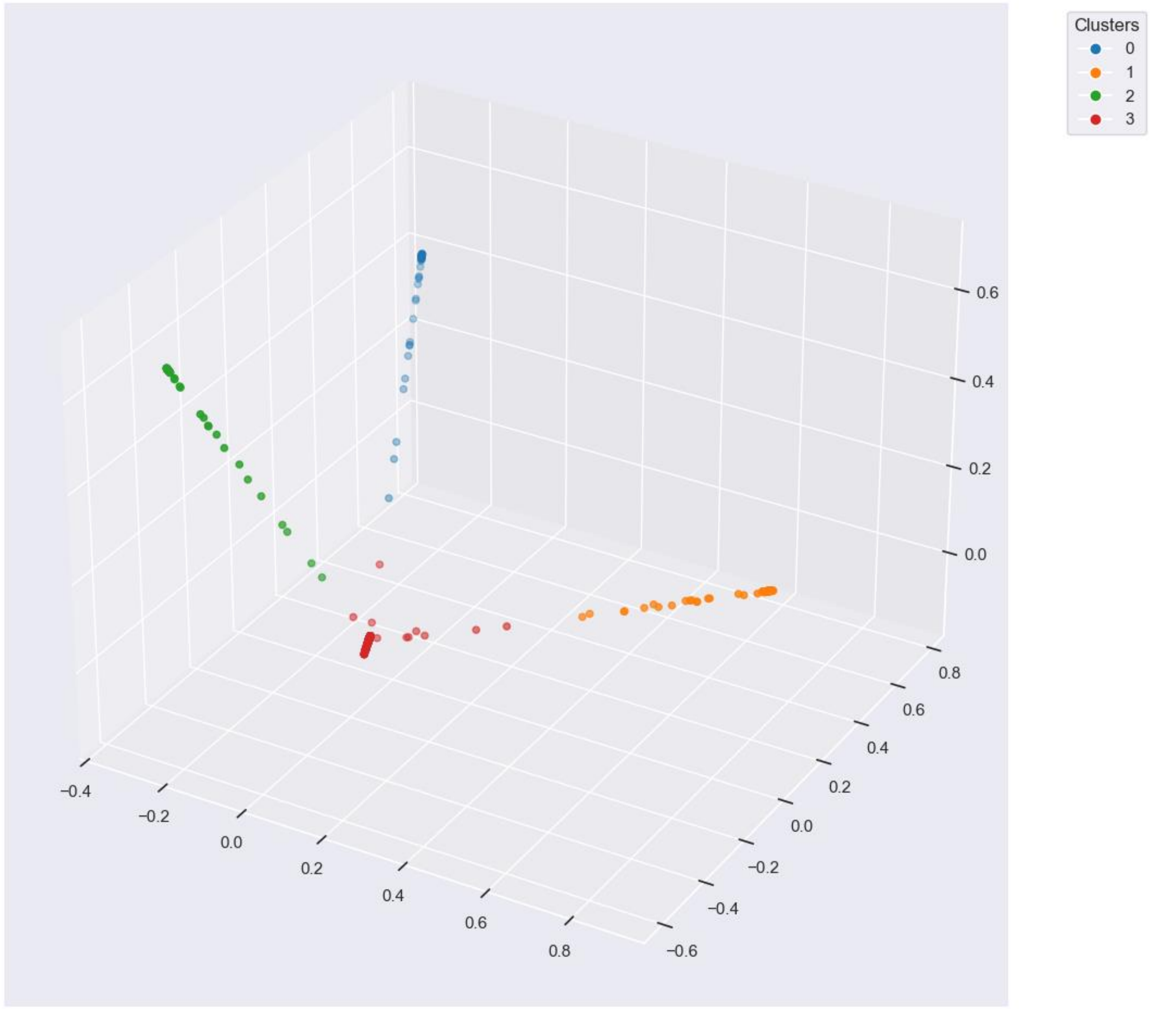
1. We have bucketized the day column in three equal frequency buckets
2. The first cluster contains data points related to the lowest possible number of day in which the meter is active…obviously the second cluster contains meter who registered consumptions for an average number of day (second bucket) and so on
3. Association rules are very similar, the only difference is related to this aspect

Agglomerative WARD linkage with KPCA (rbf kernel)



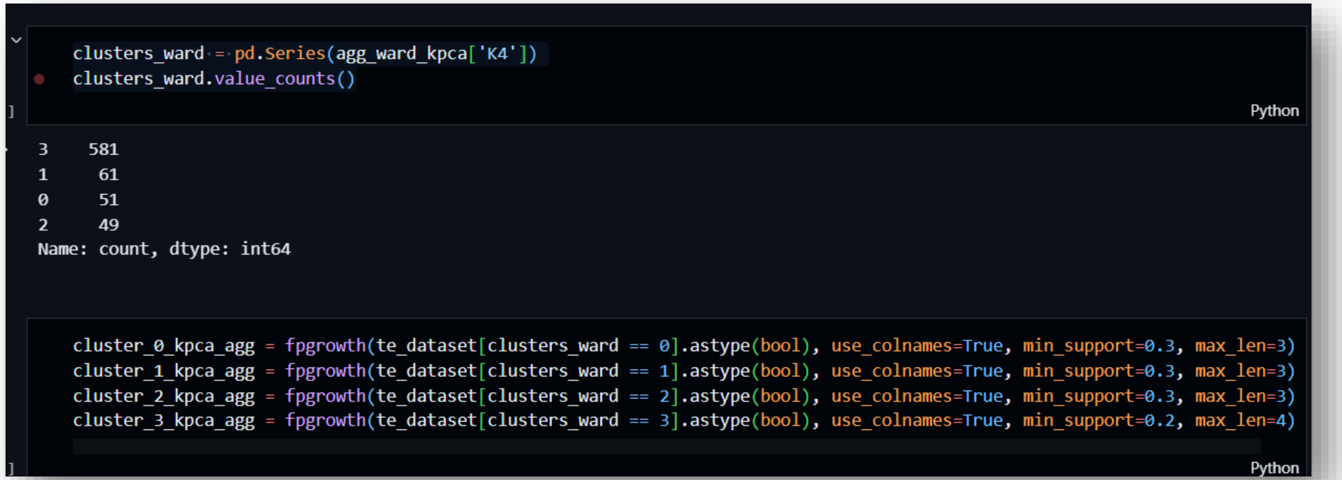
The circled cluster holds nearly 600 Data points over the 742 total data points

Agglomerative WARD linkage with KPCA (rbf kernel)



By visualising in 3D we can actually appreciate the density of the «central» cluster C3

Agglomerative WARD KPCA



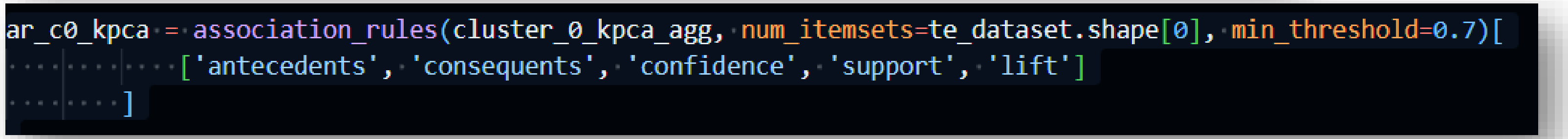
```python
clusters_ward = pd.Series(agg_ward_kpca['K4'])
clusters_ward.value_counts()
```

Python

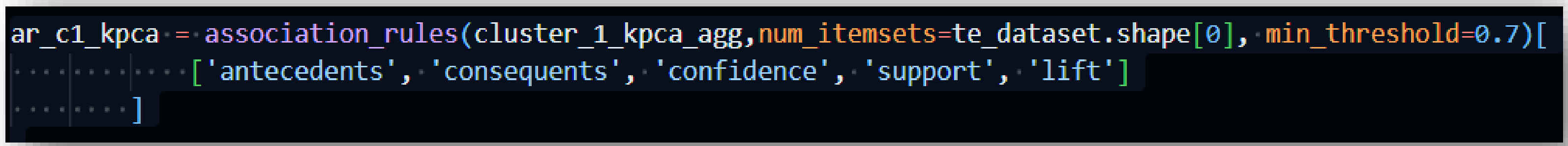
```
3    581
1     61
0     51
2     49
Name: count, dtype: int64
```

```python
cluster_0_kpca_agg = fpgrowth(te_dataset[clusters_ward == 0].astype(bool), use_colnames=True, min_support=0.3, max_len=3)
cluster_1_kpca_agg = fpgrowth(te_dataset[clusters_ward == 1].astype(bool), use_colnames=True, min_support=0.3, max_len=3)
cluster_2_kpca_agg = fpgrowth(te_dataset[clusters_ward == 2].astype(bool), use_colnames=True, min_support=0.3, max_len=3)
cluster_3_kpca_agg = fpgrowth(te_dataset[clusters_ward == 3].astype(bool), use_colnames=True, min_support=0.2, max_len=4)
```

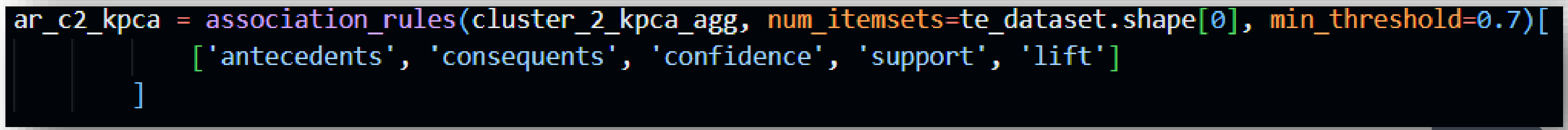
Python



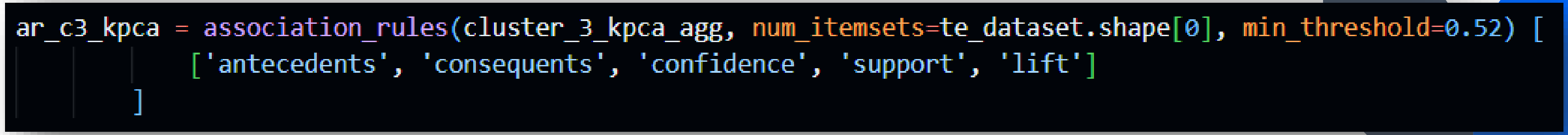
```python
ar_c0_kpca = association_rules(cluster_0_kpca_agg, num_itemsets=te_dataset.shape[0], min_threshold=0.7)[
            ['antecedents', 'consequents', 'confidence', 'support', 'lift']
        ]
```



```python
ar_c1_kpca = association_rules(cluster_1_kpca_agg,num_itemsets=te_dataset.shape[0], min_threshold=0.7)[
            ['antecedents', 'consequents', 'confidence', 'support', 'lift']
        ]
```



```python
ar_c2_kpca = association_rules(cluster_2_kpca_agg, num_itemsets=te_dataset.shape[0], min_threshold=0.7)[
            ['antecedents', 'consequents', 'confidence', 'support', 'lift']
        ]
```



```python
ar_c3_kpca = association_rules(cluster_3_kpca_agg, num_itemsets=te_dataset.shape[0], min_threshold=0.52) [
            ['antecedents', 'consequents', 'confidence', 'support', 'lift']
        ]
```

Agglomerative WARD linkage with KPCA (rbf kernel): cluster 0

Antecedent	Consequent	Confindence	Support
Borough_QUEENS, CONSUMPTION (1062, 5271]	IS_METER_AMR	90%	34%
Borough_QUEENS	IS_METER_AMR	93%	61%
current_charges_(9610.33, 46454.87]	consumption_(hcf)_(1061.6, 5270.8],is_federal	100%	48%
service_start_date_12/13/2012	#_days_(2077.0, 4057.0],service_end_date_10/22/2019	95%	75%
current_charges_(3040.262, 9610.33]	service_start_date_12/13/2012	90%	33%

Comments

- It looks like AMR meters are very common in Queens, and this cluster groups meter that registered a moderate consumption level
- A large subset of this clustered data seems to follow a period from early winter 2012 and early Autumn 2019

Agglomerative WARD linkage with KPCA (rbf kernel): cluster 1

Antecedent	Consequent	Confindence	Support
consumption_(hcf)_(242.8, 1061.6], current_charges_(3040.262, 9610.33]	is_meter_amr	100%	40%
is_not_only_water_sewer_charges	#_days_(2077.0, 4057.0], is_meter_amr	100%	40%
is_only_water_sewer_charges	#_days_(2077.0, 4057.0], is_meter_amr	97%	57%
service_start_date_11/14/2012 current_charges_(9610.33, 46454.87]	service_end_date_10/22/2019	96%	38%
service_end_date_10/22/2019, consumption_(hcf)_(1061.6, 5270.8]	current_charges_(9610.33, 46454.87]	100%	38%

Comments

Question: why did I reported a.r. based on consumptions?

- It looks **like clusters are grouped based on the time period and number of days of activation**
- While the cluster c0 mainly focus on a single category of consumption, this cluster is more “mixed”. Two groups are identified in this cluster:
  1. Meters with additional charges beyond water/sewer
  2. Basic water/sewer accounts, which are more common
- There are also rules that specify different consumptions and costs in the time period followed by the cluster, ***potentially corresponding to residential versus commercial/federal usage patterns***.

Note: cluster contains rules related to QUEENS like the previous one.

Agglomerative WARD linkage with KPCA (rbf kernel): cluster 2

Antecedent	Consequent	Confindence	Support
service_start_date_01/14/2013	service_end_date_10/22/2019, is_meter_amr	90%	75%
service_start_date_01/14/2013, bill_analyzed_score_(0.499, 0.953]	is_meter_amr	100%	30%
service_start_date_01/14/2013, bill_analyzed_score_(0.953, 0.969]	is_meter_amr	100%	30%

Comments

- Rules found are basically the same of cluster 0 but follow a different time period, in which analyzed consumption are in the same range of consumption (and charges) in cluster 0… (still in QUEENS)
- Graphically this cluster is not far away from the c0
- Like in the previous case there are two groups of meters here:
  - *The one that analyze an average number of bills*
  - *The one that analyze basically every bill* (this case is common to other clusters)

Agglomerative WARD linkage with KPCA (rbf kernel): cluster 3

Antecedent	Consequent	Confindence	Support
borough_BROOKLYN	is_meter_amr	82%	22%
borough_QUEENS	is_meter_amr	70%	30%
#_days_(5.999, 980.0]	is_only_water_sewer_charges, is_meter_amr	75%	35%
#_days_(980.0, 2077.0]	is_only_water_sewer_charges, is_meter_amr	60%	25%
consumption_(hcf)_(5270.8, 16346.2]	current_charges_(46454.87, 158822.24]	100%	38%
consumption_(hcf)_(16346.2, 487833.0]	current_charges_(158822.24, 1713233.95]	97%	20%
consumption_(hcf)_(5270.8, 16346.2], is_meter_amr	current_charges_(46454.87, 158822.24]	95%	20%

Comments

- Brooklyn and Queens are the most represented boroughs. Brooklyn has a stronger association with AMR meters (82%) than Queens (70%), but Queens represents a larger share of the cluster.
- This might suggest differences in AMR rollout strategies or infrastructure across boroughs.
- Very high consumption (and charges)... maybe meters are located in buildings used for some kind of commercial activity

Final comments

What did we have discovered?

- For each cluster we have identified how for a certain **consumption range** there is a certain **charges range**
- **AMR Meters Are Predominant**, but Distribution Varies by Borough
- AMR meters are often linked with federal funding for developments
- Groups in the data look like they are formed considering a specific time period or number of day in which the meter Is active
- Linear shaped cluster are characterized by a constant #_days interval
- The dense cluster is characterized by groups of different, and smaller, #_days interval but meter in this cluster appear to register high levels of water consumption

We have not discovered much from this dataset.

But at least we have some into that can be used to improve the analysis.

Possible improvements

- Improving data quality
  - Engineer new kind of features (e.g. seasons...)
- Apply different combinations of standardizations
  - This could lead to different groups
- Add more data...

Possible applications

Billing Optimization

Infrastructure management

Plan future maintenance

Thank you

Giacomo Detomaso

giacomodetomaso24@gmail.com