פרויקט מבוא למדעי הנתונים

יובל גולדנשטיין - 311552368
ישי שפירא - 203016217
נדב שטרן - 203016100

Clustering
Link jupyter notebook for part 2:
https://colab.research.google.com/drive/1tuFFKBPyWnB-EXEyXntzggUJrgkPwjNO?usp=sharing

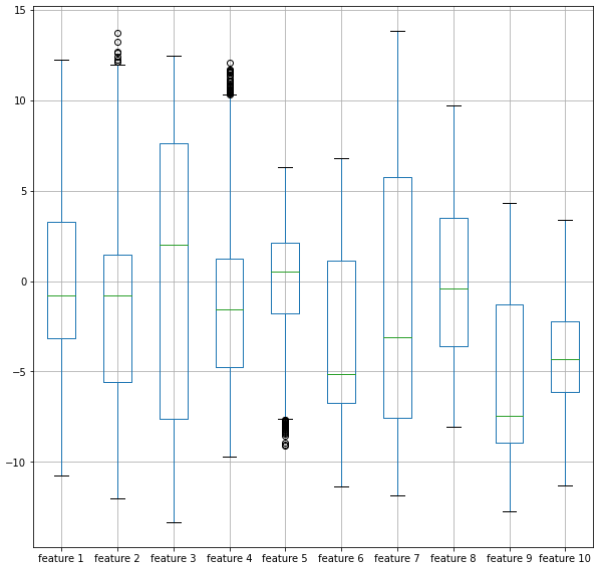
EDA

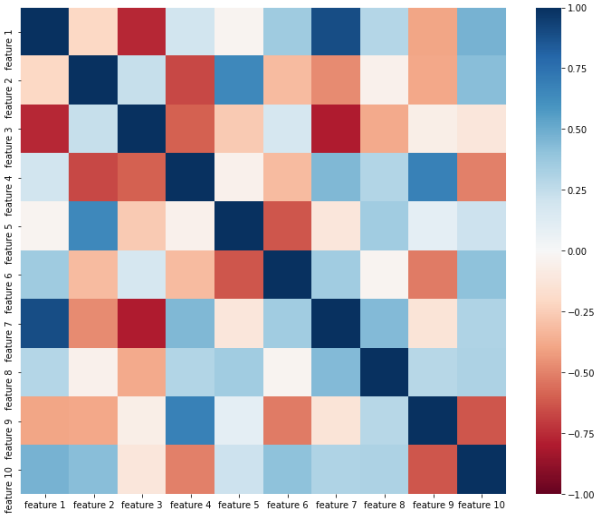
Dataset info:

RangeIndex: 2500 entries, 0 to 2499
Data columns (total 10 columns):

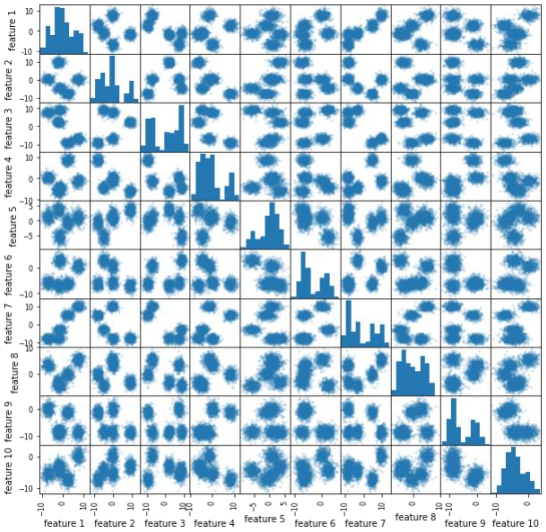
#	Column	Non-Null Count	Dtype
0	feature 1	2500 non-null	float64
1	feature 2	2500 non-null	float64
2	feature 3	2500 non-null	float64
3	feature 4	2500 non-null	float64
4	feature 5	2500 non-null	float64
5	feature 6	2500 non-null	float64
6	feature 7	2500 non-null	float64
7	feature 8	2500 non-null	float64
8	feature 9	2500 non-null	float64
9	feature 10	2500 non-null	float64

dtypes: float64(10)



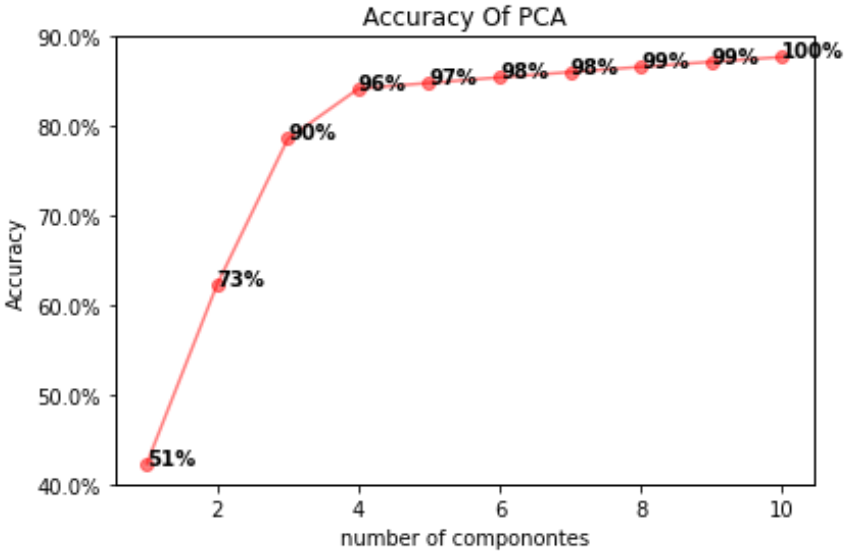


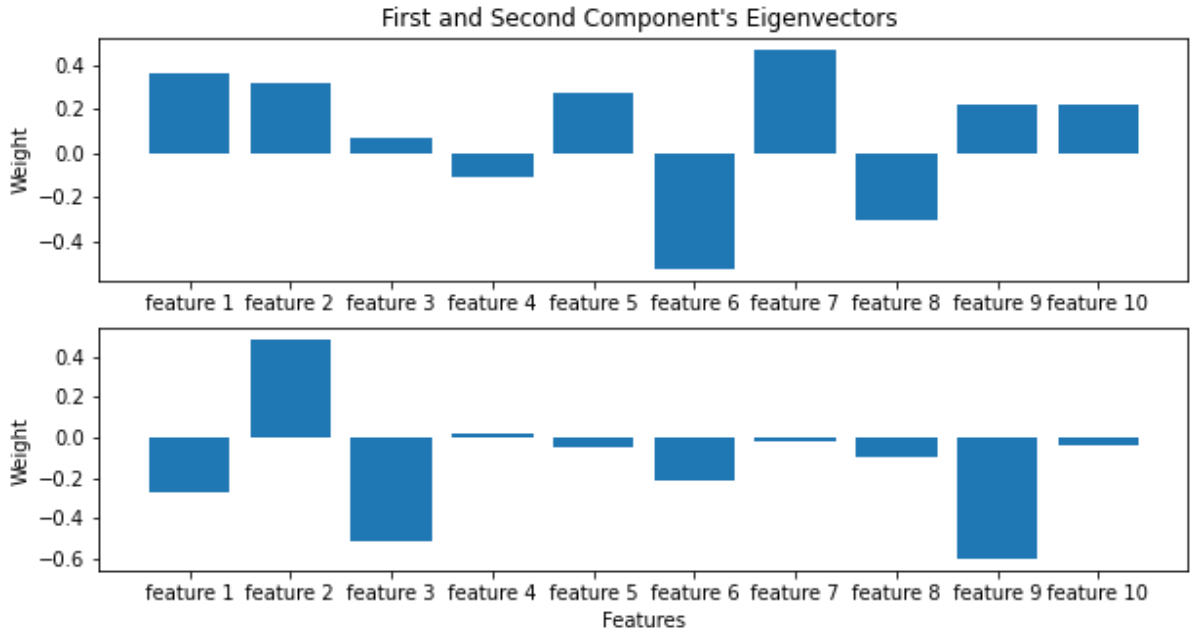
We can see the data has correlations and there is a good base for clustering:



PCA on orig. dataset

Principal Component Analysis (PCA) is a linear dimensionality reduction technique that can be utilized for extracting information from a high-dimensional space by projecting it into a lower-dimensional sub-space. It tries to preserve the essential parts that have more variation of the data and remove the non-essential parts with fewer variation.





We will use the PCA's results to visualize the k means model and DBSCAN model's outputs on the original dataset.

K Means

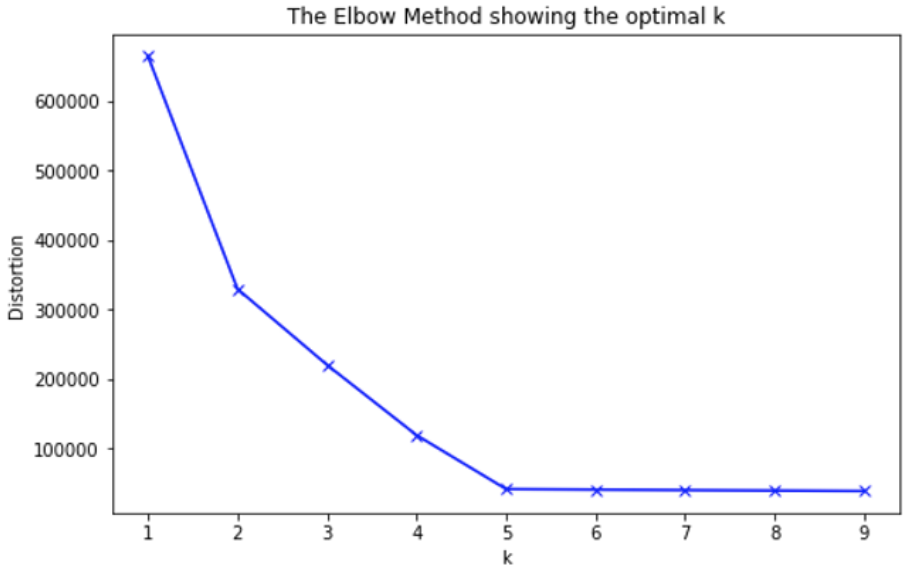
K-Means is an unsupervised machine learning algorithm that groups data into K number of clusters.

Number of clusters

The number of clusters is user-defined and the algorithm will try to group the data even if this number is not optimal for the specific case. Therefore we have to come up with a technique that somehow will help us decide how many clusters we should use for the K-Means model.

Elbow Method

The Elbow method is a very popular technique and the idea is to run k-means clustering for a range of clusters k (let's say from 1 to 10) and for each value, we are calculating the sum of squared distances from each point to its assigned center (distortions).



We can observe that the "elbow" is the number 5 which is optimal for this case. Now we can run a K-Means using as n_clusters the number 5.

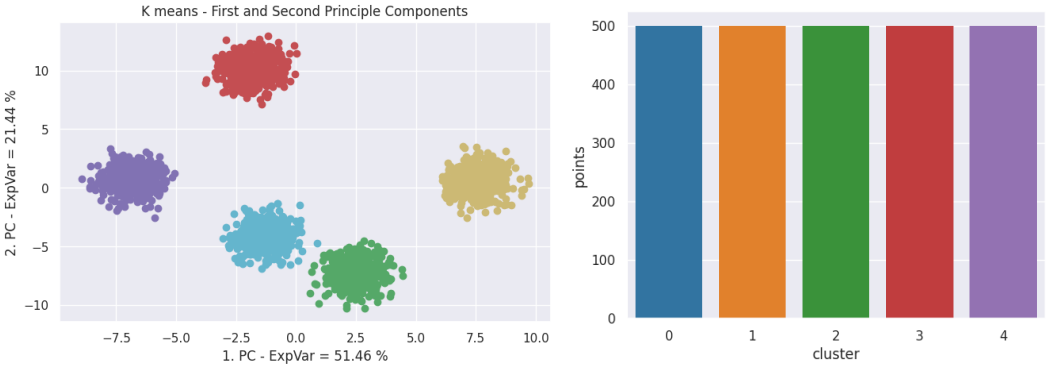
Running The K Means Model

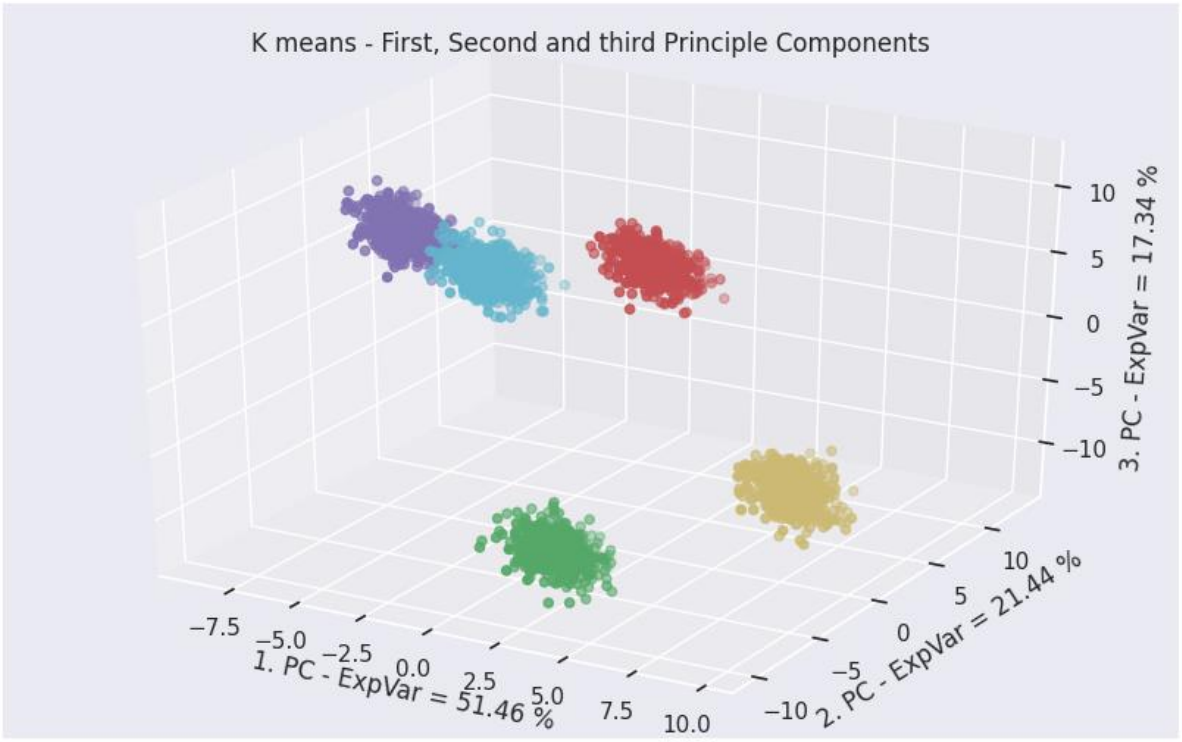
We used scipy package and kmeans2. The algorithm attempts to minimize the Euclidean distance between observations and centroids. initialization methods are included.

Model Parameters:

- **k = 5**
  The number of clusters to form as well as the number of centroids to generate
- **Iter = 100**
  Number of iterations of the k-means algorithm to run
- **Minit = random**

Method for initialization. 'random': generate k centroids from a Gaussian with mean and variance estimated from the data



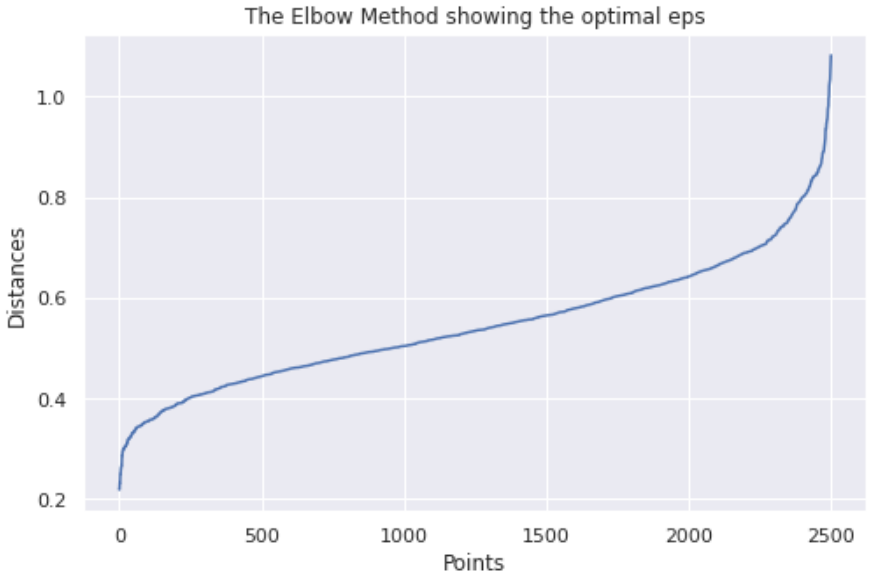


Silhouette Coefficient for k-means: 0.621

DBSCAN

DBSCAN - Density-Based Spatial Clustering of Applications with Noise. Finds core samples of high density and expands clusters from them. Good for data which contains clusters of similar density.

- Min_samples: The minimum number of neighbors a given point should have in order to be classified as a core point.
- eps: Two points are considered neighbors if the distance between the two points is below the threshold epsilon. we find a suitable value for epsilon by calculating the distance to the nearest n points for each point, sorting and plotting the results. Then we look to see where the change is most pronounced ('Elbow') and select that as epsilon.
- metric: The metric to use when calculating distance between instances in a feature array (i.e. euclidean distance).

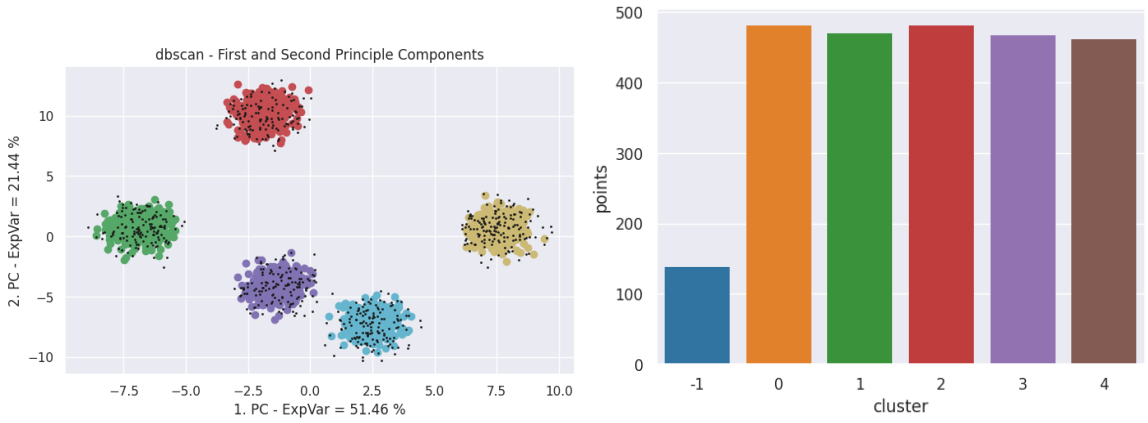


We train our model, selecting 0.8 for eps and setting min_samples to 10

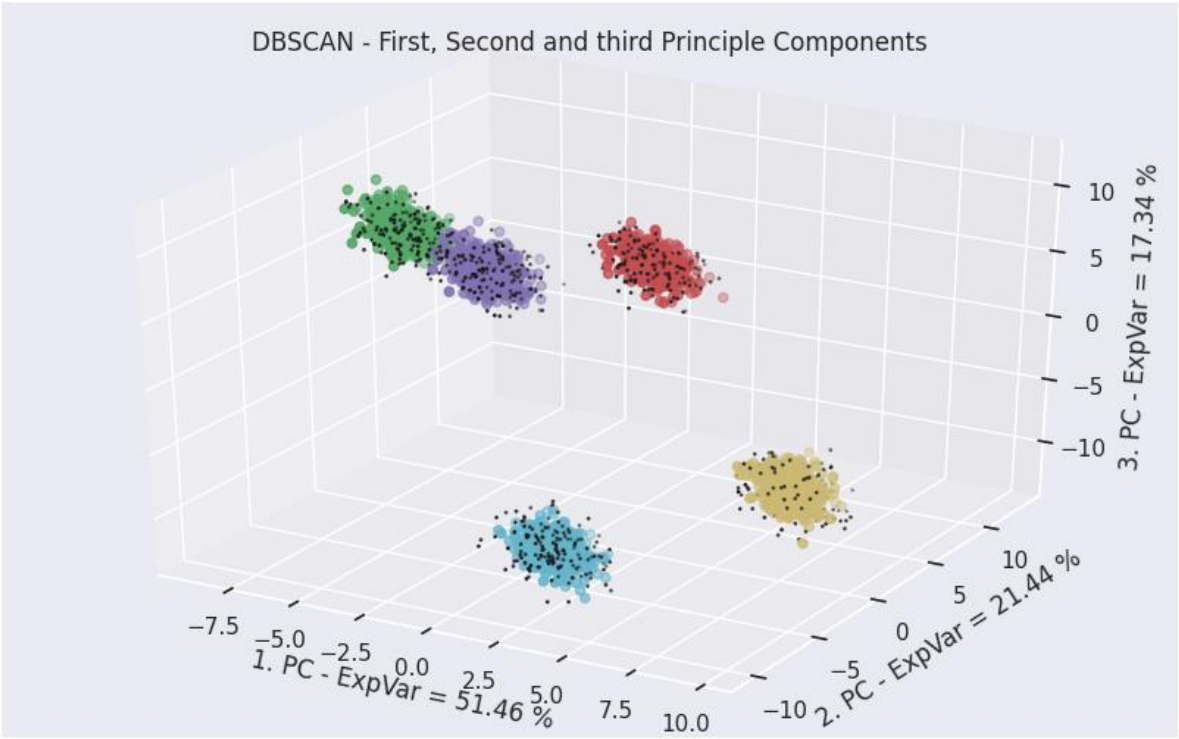
```
# Compute DBSCAN
db = DBSCAN(eps=0.7, min_samples=100).fit(X)
core_samples_mask = np.zeros_like(db.labels_, dtype=bool)
core_samples_mask[db.core_sample_indices_] = True
labels = db.labels_

# Number of clusters in labels, ignoring noise if present.
n_clusters_ = len(set(labels)) - (1 if -1 in labels else 0)
n_noise_ = list(labe¡ls).count(-1)
```

The model classified the densely populated areas. As we can see, all the dark blue points were categorized as noise.

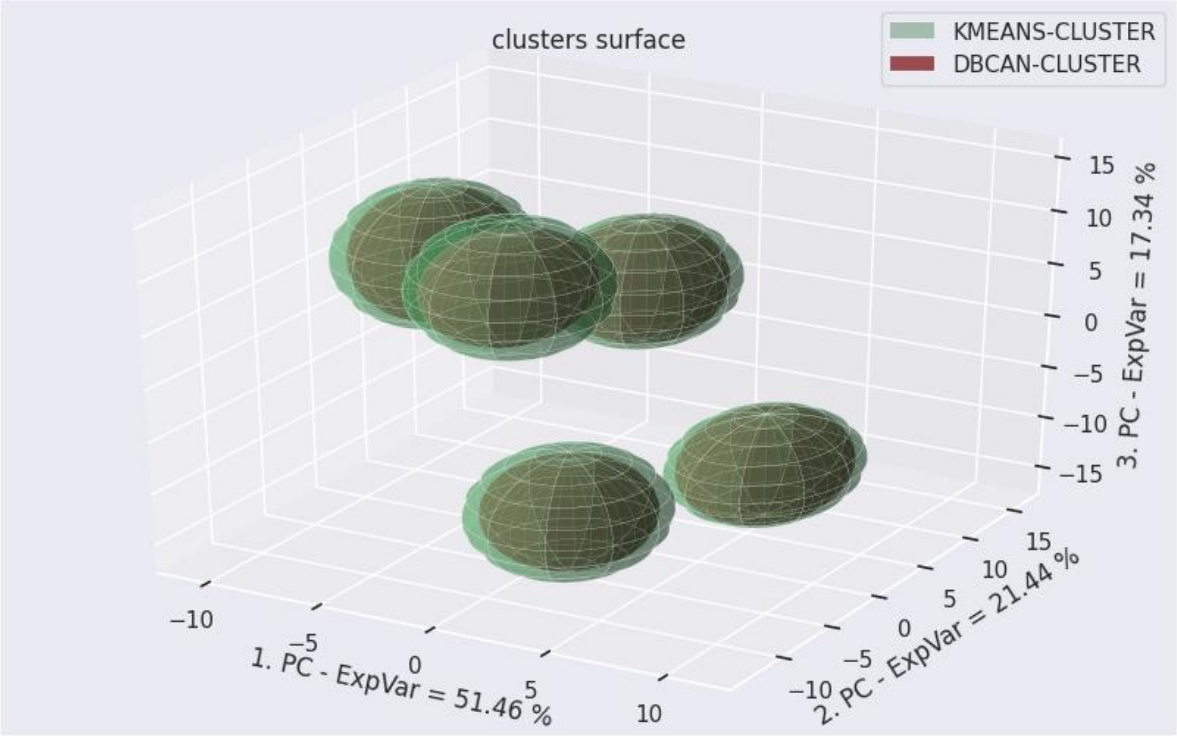


Estimated number of clusters: 5
Estimated number of noise points: 139



Analysis

The figure represents the surface of the 5 clusters in k-means model and for the DBSCAN model.



n_samples 2500, n_features 10

As can see in the plot, the K Means clusters have smaller volume compared to the dbscan's clusters due the noise points that were eliminated from the clusters.
Both of the models have similar centroids.

	time	**Silhouette Coefficient**	**Clusters**	**Noise Points**
K Means	0.06s	0.71	5	-
DBSCAN	0.23	0.62	5	139