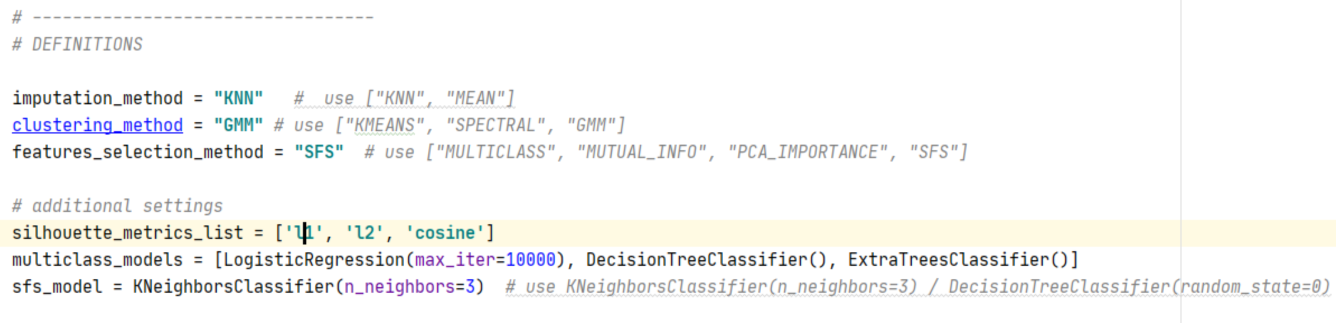
Wet section

Since this work was done in pairs, many independent methods were used.

Code organization

As before, we have built a sustainable code which reproduces the results over different imputation, clustering, and feature importance methods. For each chosen configuration, the user will receive all the relevant outputs, and the products of calculation, graphs, etc.

The configuration is defined inside the **main.py** file. **App_functions.py** file contains all the other procedures. Those are the supported and tested definitions (which you can change and run the code!):



```
# ----------------------------------
# DEFINITIONS

imputation_method = "KNN"   #  use ["KNN", "MEAN"]
clustering_method = "GMM" # use ["KMEANS", "SPECTRAL", "GMM"]
features_selection_method = "SFS"  # use ["MULTICLASS", "MUTUAL_INFO", "PCA_IMPORTANCE", "SFS"]

# additional settings
silhouette_metrics_list = ['l1', 'l2', 'cosine']
multiclass_models = [LogisticRegression(max_iter=10000), DecisionTreeClassifier(), ExtraTreesClassifier()]
sfs_model = KNeighborsClassifier(n_neighbors=3)  # use KNeighborsClassifier(n_neighbors=3) / DecisionTreeClassifier(random_state=0)
```

For the convenience, we have attached the “output” folder with the results for ALL the configurations.

Running each configuration will produce the required **selected_proteins.txt** and **clusters.csv** files.

The files attached here are for the configuration that is shown above – those are our chosen features.

The folder “notebooks” contains the jupyter notebooks we used while doing this homework to verify our results.

The outputs you receive when running the code:

1. Outputs/protein_fixed_<IMPUTATION_METHOD>.csv – fixed db with chosen imputation
2. outputs/<CLUSTERING_METHOD>/ 2D_PCA_Scatter.png – scatter plot on PCA 2 components
3. outputs/<CLUSTERING_METHOD>/3D_PCA_Scatter.png – scatter plot on PCA 3 components
4. outputs/<CLUSTERING_METHOD>/scores.txt – scores of this clustering method
5. outputs/<CLUSTERING_METHOD>/Silhouette_graph.png – silhouette scores graph for each sample
6. outputs/<CLUSTERING_METHOD>/<FEATURES_SELECTION_METHOD>/mutations_to_treat.txt – list of 3 mutations according to current indexation of the clusters to treat
7. outputs/<CLUSTERING_METHOD>/<FEATURES_SELECTION_METHOD>/* - all the other outputs for the selected features selection algorithm

IMPORTANT NOTE:

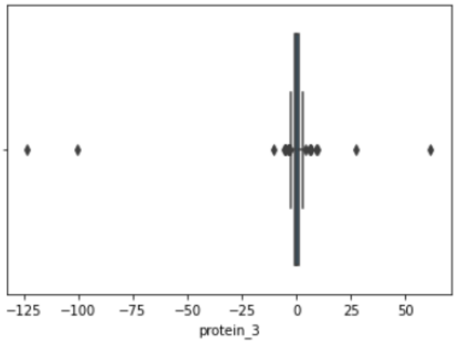
- **since for every run, the index of the mutations differs, the list of the CURRENT mutations appears in the outputs/<CLUSTERING_METHOD>/<FEATURES_SELECTION_METHOD>/ mutations_to_treat.txt. The mutations that appear in this report may not be consistent with the output that you get when running this code**

Data Preprocessing

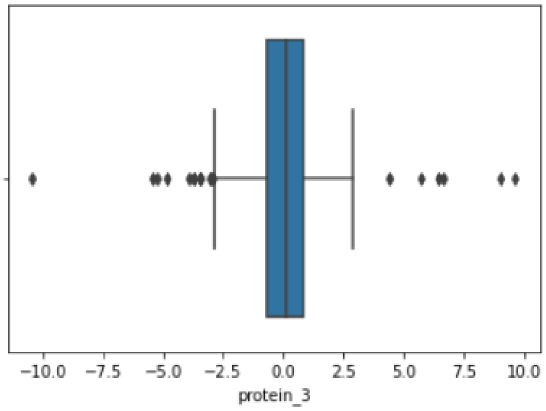
We have noticed that data has a lot of outliers and a lot of missing data. So the following steps were taken to deal with it, without losing much information (without simply dropping rows with missing data or outliers):

1. For each feature, outliers were detected using the z-score equal 2.5 (empirical number)
2. The outliers and the missing values were replaced with KNN Imputation

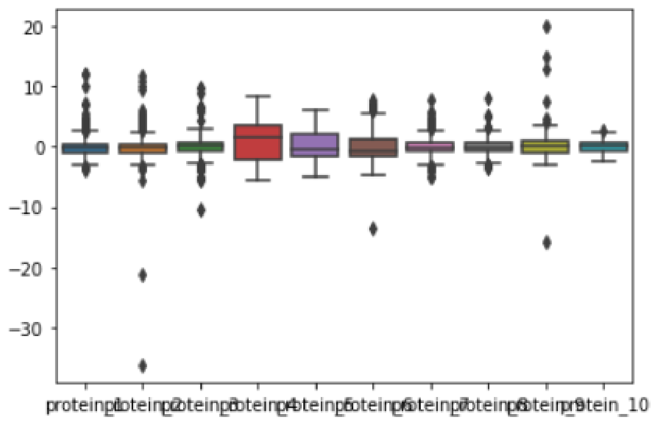
We can see the results visually. Box plot of the distribution of one of the features before the preprocessing was applied:



And after:



And the distribution of all the features (after):


20
10
0
−10
−20
−30
protein_1
protein_2
protein_3
protein_4
protein_5
protein_6
protein_7
protein_8
protein_9
protein_10

Addressing the ML tasks

We have examined the existing Clustering ML approached, chose 3 models we want to use based on their diversified approached and the fact that all of them were taught in the lectures. We evaluate the performance of each clustering method using Silhouette metrics. Later, we use different methods to choose the most important features.

This is the short examination:

Pros/Cons of various Clustering Methods

K Means Clustering

Pros

- Easy to implement.
- Guaranteed convergence.
- Scales to large datasets

Cons

- a negligent edge of each cluster, because the priorities are set on the center of the cluster, not on its borders. (Because optimizing k-means equivalent to optimizing MLE for mixed Gaussian model with covariance matrix $\sum = Id$)
- an inability to create a structure of a dataset with objects that can be classified to multiple clusters in equal measure.
- a need to guess the optimal *k* number, or a need to make preliminary calculations to specify this gauge.
- NP-hard.
- Handling of outliers.
- Requires initialization of centers.
- Difficulty clustering data of varying densities and sizes
- Curse of dimensionality- with increasing amount of features the distances between points obtain a lower ratio of std. variation to mean.

Mixed Gaussians Model

Pros

- Unlike the centroid-based models, the EM algorithm allows the points to classify for two or more clusters – it simply presents you the possibility of each event, using which you can conduct further analysis.
- The borders of each cluster compose ellipsoids of different measures unlike k-means, where the cluster is visually represented as a circle.

Cons

- The algorithm simply would not work for datasets where objects do not follow the Gaussian distribution. It is more applicable to theoretical problems rather than the actual measurements or observations.
- Insufficient data leads to overfitting and difficulty estimating the covariance matrices which leads to divergence

Spectral Clustering

Pros

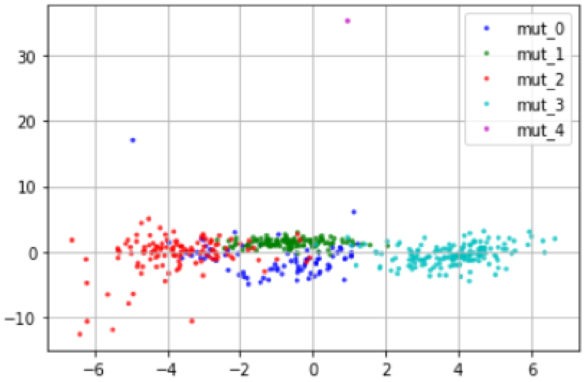
- Can recognize clusters that do not have a clear blob-shape, such as concentric rings.
- Polynomial time optimizable for fixed kfor cost function G_cut (defined in lecture)
- Avoids curse of dimesionality- projects data onto lower dimensional space and then perform clustering using k-means (or other method)

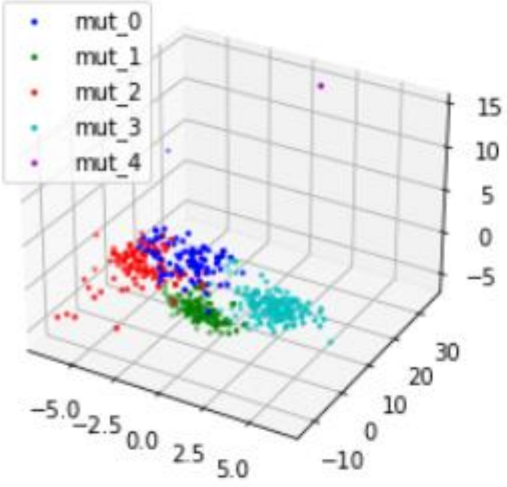
Cons

- tends to produce small clusters
- NP hard for cost function G_cost_cut for k=2 (defined in lecture)

Clustering using KMeans

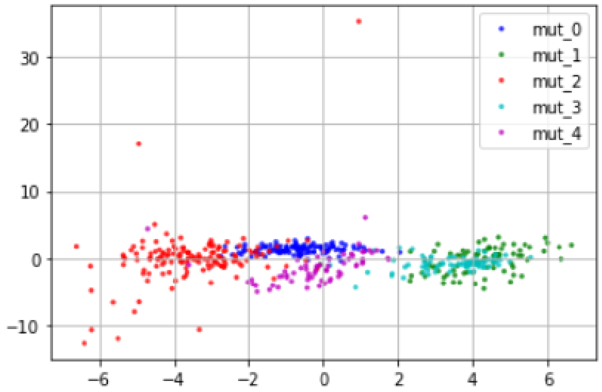
Using the first method, we cluster the data into 5 clusters. To observe the results visually (we will use this method a lot) we use the PCA, with 2 and 3 components. PCA makes each sample to contain only 2 or 3 PC, which retain the biggest amount of information from all the features, and allows us to plot them on a 2D, 3D graphs. Using the clusters labels on those samples we obtain:

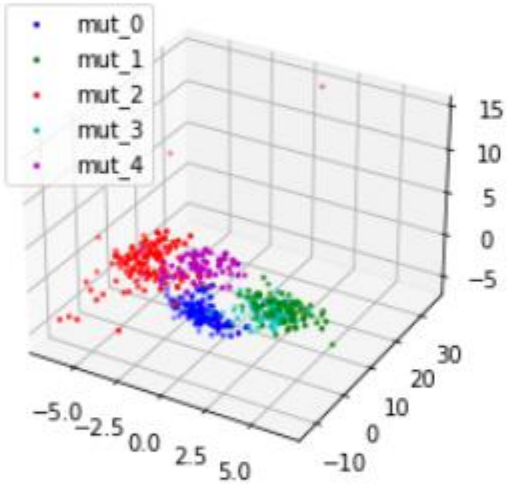




Clustering using Spectral Clustering

Similarly, using Spectral Clustering (We use nearest neighbor as criteria for Affinity Matrix), we obtain:

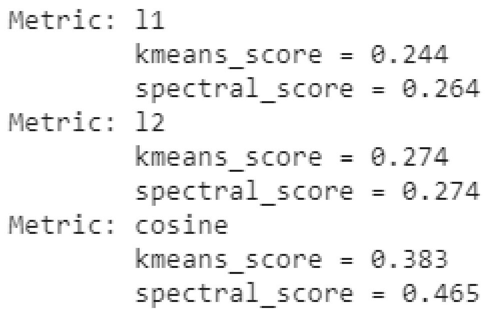




The colors are different, but we can see that in general the clusters are very similar, and visually we see indeed different group in both KMeans and Spectral Clustering algorithms.

Comparing between KMeans and Spectral Clustering

We use the silhouette score to compare between the two. Silhouette measures how similar are the object within the same class and how different they are between different classes. We use different distance metrics, such as L1, L2, or cosine (measures ‘angle’ between feature vectors projections). Results:



```
Metric: l1
        kmeans_score = 0.244
        spectral_score = 0.264
Metric: l2
        kmeans_score = 0.274
        spectral_score = 0.274
Metric: cosine
        kmeans_score = 0.383
        spectral_score = 0.465
```

The cosine score seems to be much higher for the spectral. Other metrics seem to be similar

Davies-Bouldin Score

Explanation:

The score is defined as the average similarity measure of each cluster with its most similar cluster, where simi arity is the ratio of within-cluster distances to between-cluster distances. Thus, clusters which are farther apart and less dispersed will result in a better score.

The minimum score is zero, with lower values indicating better clustering.

Result for Spectral Clustering: 1.54

Choosing the most dominant features

Several methods were used for this task.

Intra-class Feature Similarity

NOTE – we didn't use this method out of bad performance.

The base assumption was that the for a specific class and a specific feature, the distribution of the feature (if it's an important feature for this class) will be close to the mean value of this feature. E.G for mutation no. 2, the protein 4 levels is important – so this feature levels for this class will be similar.

We have used the similarity function:

$$similarity\ (C, F, S) = \exp\left(-\frac{distance}{\sigma^2}\right)$$

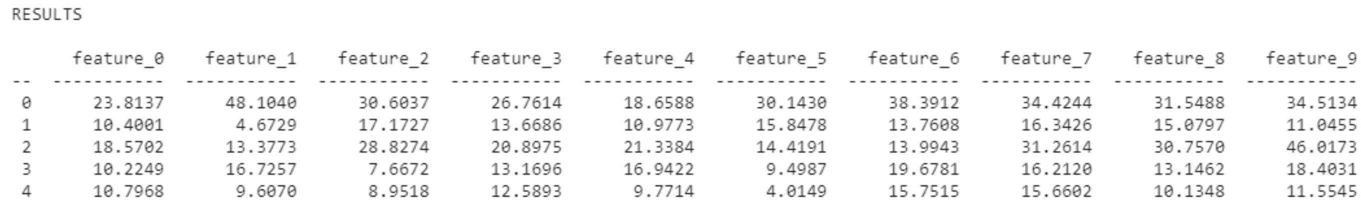
$$distance\ = \sqrt{\mu_{C,F}^2\ - f_{S,F}^2}$$

Where:

$$\begin{aligned}
&C - specific\ class\\
&F - specific\ feature\\
&S - specific\ sample\\
&\mu_{C,F} - mean\ value\ of\ feature\ F\ in\ class\ C\\
&f_{S,F} - feature\ value\ of\ sample\ S\ and\ feature\ F
\end{aligned}$$

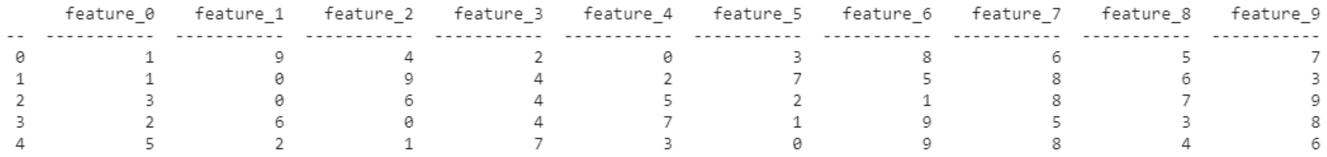
Results:

By using the σ value of 0.5, those are the results:



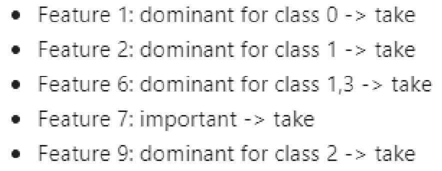
```
RESULTS

      feature_0    feature_1    feature_2    feature_3    feature_4    feature_5    feature_6    feature_7    feature_8    feature_9
--  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------
 0      23.8137      48.1040      30.6037      26.7614      18.6588      30.1430      38.3912      34.4244      31.5488      34.5134
 1      10.4001       4.6729      17.1727      13.6686      10.9773      15.8478      13.7608      16.3426      15.0797      11.0455
 2      18.5702      13.3773      28.8274      20.8975      21.3384      14.4191      13.9943      31.2614      30.7570      46.0173
 3      10.2249      16.7257       7.6672      13.1696      16.9422       9.4987      19.6781      16.2120      13.1462      18.4031
 4      10.7968       9.6070       8.9518      12.5893       9.7714       4.0149      15.7515      15.6602      10.1348      11.5545
```

For each class, we reorder the features according to their similarity:



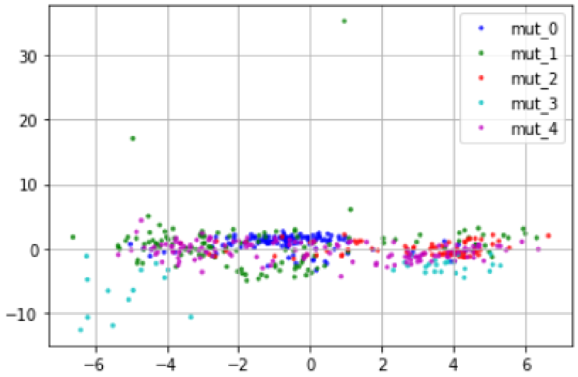
```
      feature_0    feature_1    feature_2    feature_3    feature_4    feature_5    feature_6    feature_7    feature_8    feature_9
--  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------
0             1            9            4            2            0            3            8            6            5            7
1             1            0            9            4            2            7            5            8            6            3
2             3            0            6            4            5            2            1            8            7            9
3             2            6            0            4            7            1            9            5            3            8
4             5            2            1            7            3            0            9            8            4            6
```

Analyzing:



- Feature 1: dominant for class 0 -> take
- Feature 2: dominant for class 1 -> take
- Feature 6: dominant for class 1,3 -> take
- Feature 7: important -> take
- Feature 9: dominant for class 2 -> take

To validate the results, we leave only those features, and try to cluster the samples again. We get the following visualization:



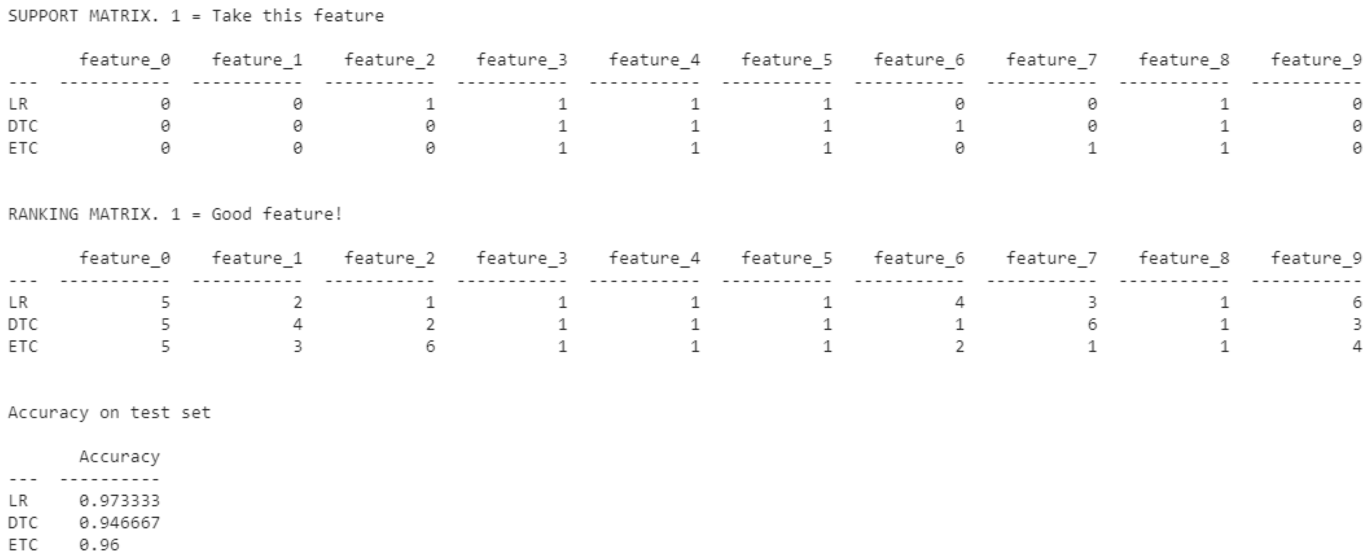
Which looks bad. From here, we have left this method out and didn't use its results further. It didn't work, perhaps out of erroneous assumptions.

Multiclass Classification

Another way to choose the best features is to train classifier on the labels that were created during the clusterin. We will train some classifier and detect the most dominant features it used for the classification. We will use the labels from the Spectral Clustering with 10 features.

First we use the RFE - Recursive Feature Elimination. It works by recursively removing attributes and building a model on those attributes that remain. It uses the model accuracy to identify which attributes (and combination of attributes) contribute the most to predicting the target attribute.

To diversify, we use 3 classifier models – Logistic Regression, DecisionTreeClassifier and ExtraTreesClassifier. Those are the results:

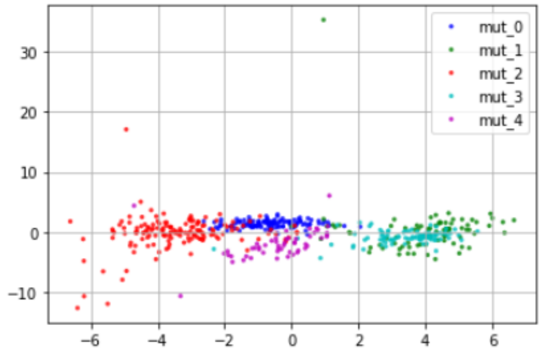


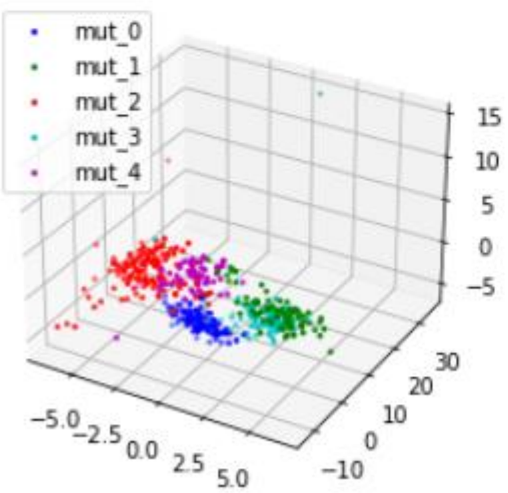
```
SUPPORT MATRIX. 1 = Take this feature

       feature_0    feature_1    feature_2    feature_3    feature_4    feature_5    feature_6    feature_7    feature_8    feature_9
---  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------
LR             0            0            1            1            1            1            0            0            1            0
DTC            0            0            0            1            1            1            1            0            1            0
ETC            0            0            0            1            1            1            0            1            1            0

RANKING MATRIX. 1 = Good feature!

       feature_0    feature_1    feature_2    feature_3    feature_4    feature_5    feature_6    feature_7    feature_8    feature_9
---  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------
LR             5            2            1            1            1            1            4            3            1            6
DTC            5            4            2            1            1            1            1            6            1            3
ETC            5            3            6            1            1            1            2            1            1            4

Accuracy on test set

       Accuracy
---  ----------
LR     0.973333
DTC    0.946667
ETC    0.96
```

Features 3,4,5,7 agreed on all classifiers to be best features. Feature 6 is good feature for LR, which has highest accuracy score.

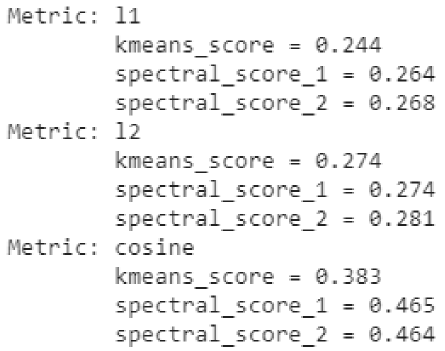
So from this analysis, best features are 3, 4, 5, 6, 7.

We make the visualization test again – I make clustering **using only those 5 chosen features**. The results are:





And it looks really good. So those features really are most important and contain most information for classification. But visualization is not enough, we also use silhouette score in this case. Spectral_score_2 applies to clustering using Specrtal Clustering, but only using 5 selected features:

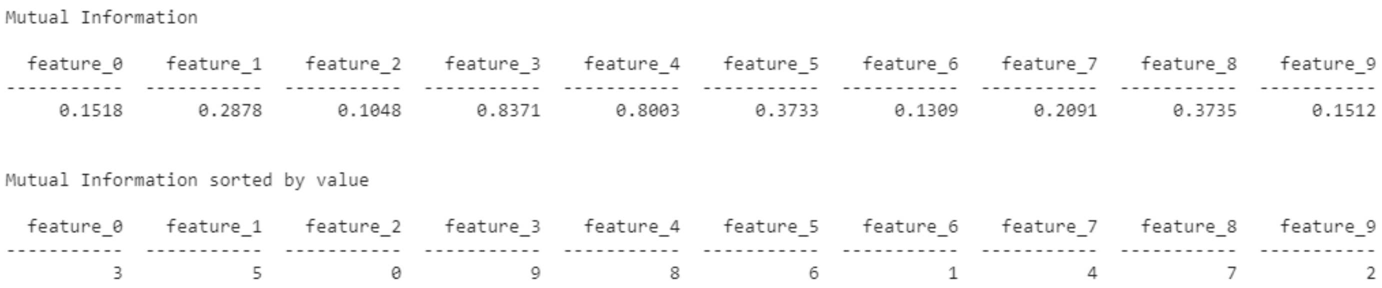


```
Metric: l1
        kmeans_score = 0.244
        spectral_score_1 = 0.264
        spectral_score_2 = 0.268
Metric: l2
        kmeans_score = 0.274
        spectral_score_1 = 0.274
        spectral_score_2 = 0.281
Metric: cosine
        kmeans_score = 0.383
        spectral_score_1 = 0.465
        spectral_score_2 = 0.464
```

We can see that the metrics didn’t get worse, and even improved a bit. This can be explained by removing features which did adversary effect on the clustering in the beginning.

Mutual Information Score

To validate the previous method, we use the Mutual Score between the feature values and the classification. Those are the results:

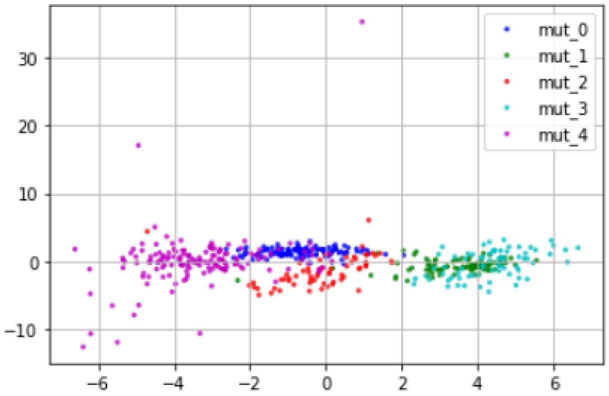


```
Mutual Information

  feature_0    feature_1    feature_2    feature_3    feature_4    feature_5    feature_6    feature_7    feature_8    feature_9
-----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------
     0.1518       0.2878       0.1048       0.8371       0.8003       0.3733       0.1309       0.2091       0.3735       0.1512

Mutual Information sorted by value

  feature_0    feature_1    feature_2    feature_3    feature_4    feature_5    feature_6    feature_7    feature_8    feature_9
-----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------  -----------
          3            5            0            9            8            6            1            4            7            2
```

best_features_list:
[1, 5, 8, 4, 3]

Which are mostly (3/5) similar to the Features found by the RFE (3, 4, 5, 7, 8).

By using those features from MI, we can also see quiet a good classification visually:



Best Features Spectral Clustering

The most important features were chosen as the output of the RFE method, since it used various models, and used various metrics.

Best features:

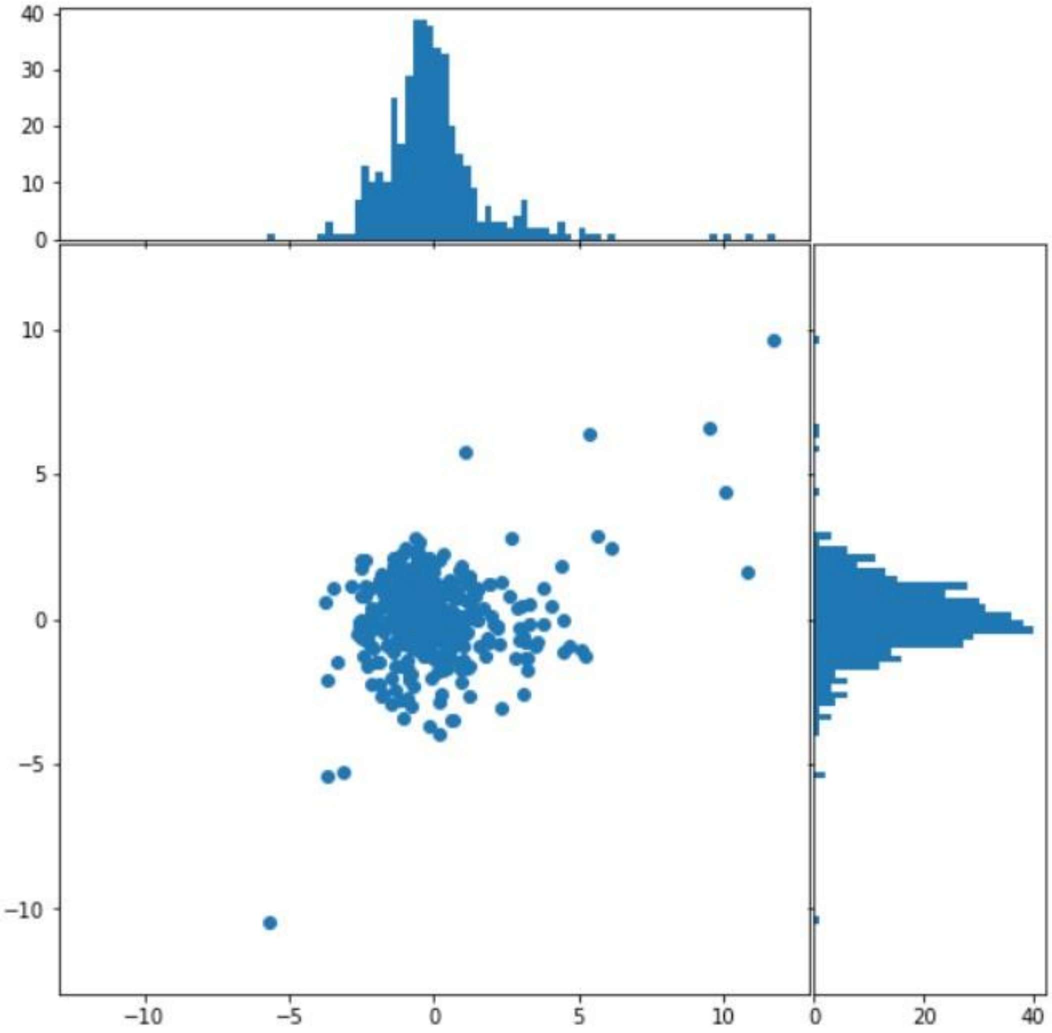
(3, 4, 5, 7, 8).

Clustering using Gaussian Mixture Models

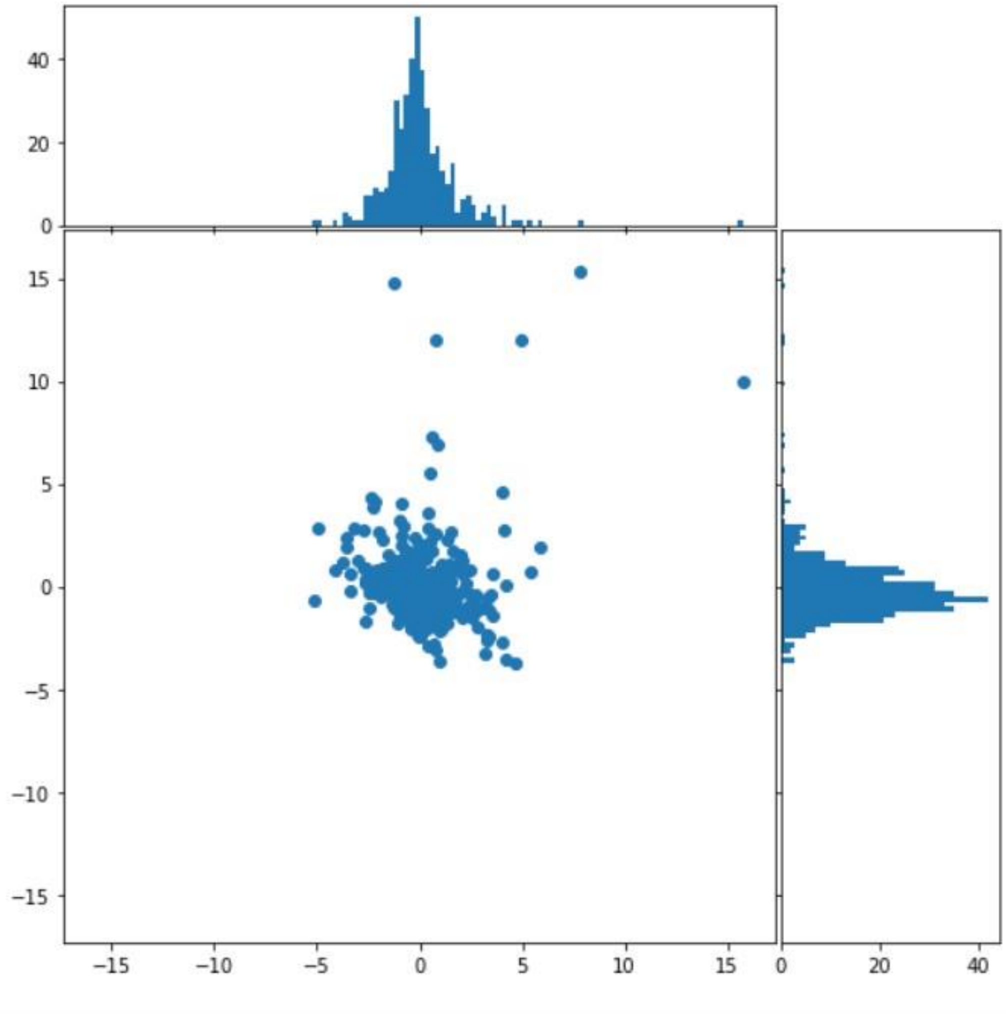
Motivation

Visualizing the data and using expert knowledge of Gaussian distributions, the features are closely matching a Gaussian distribution over 10 dim-space, as seen below:

Feature 3 vs Feature 2:



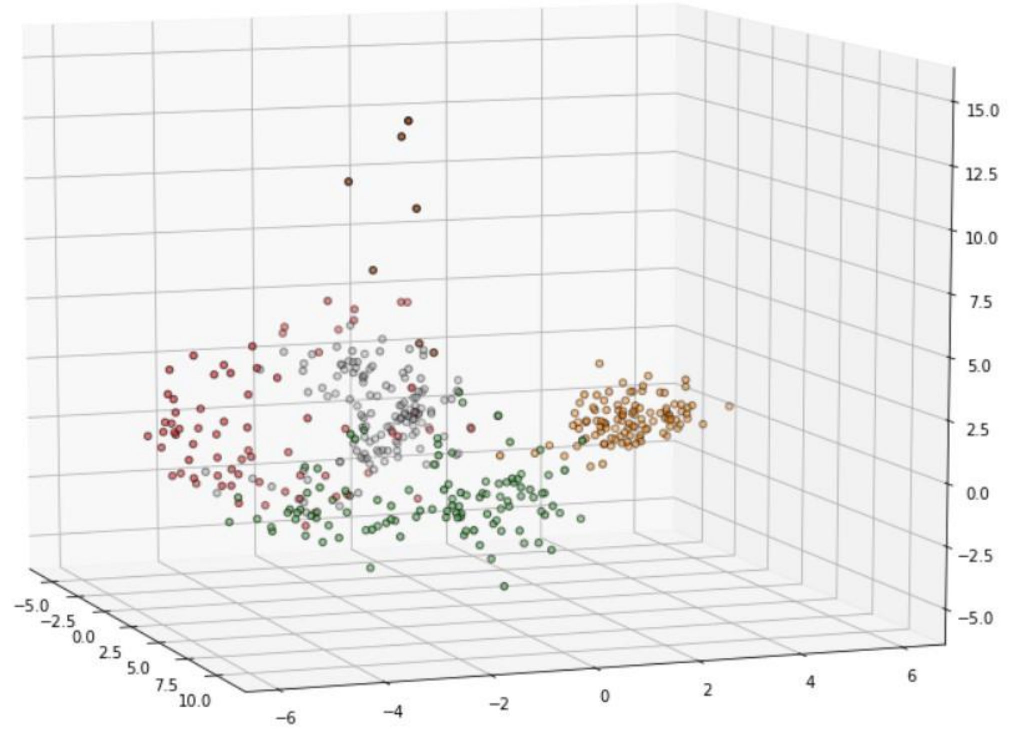
Feature 7 vs Feature 9:

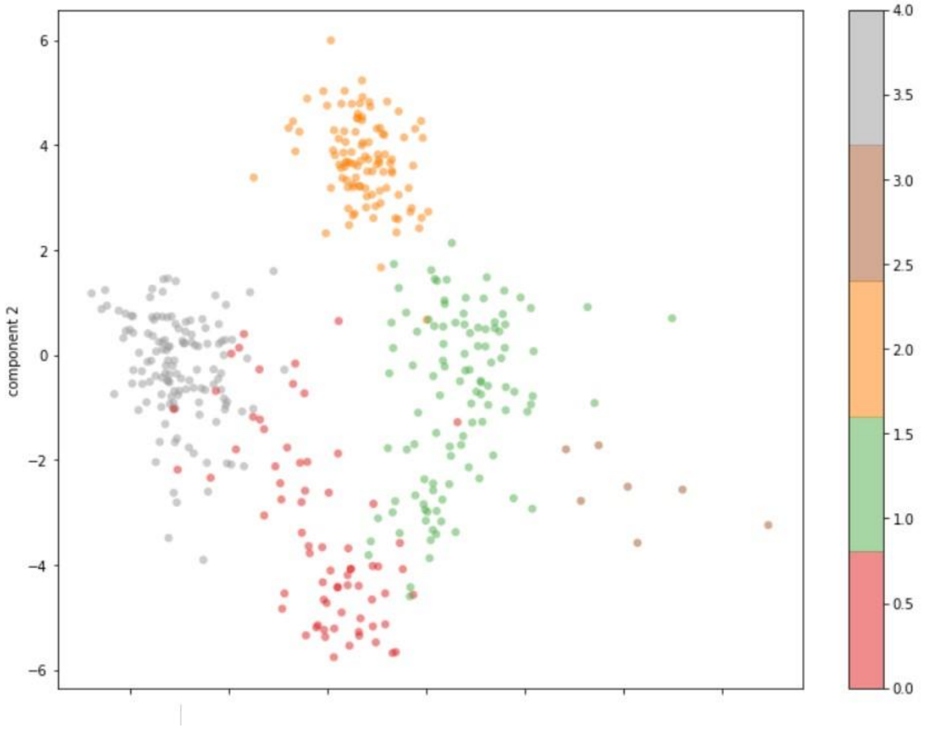


Taking into account an advantage of Gaussian Mixture Model mentioned in a prior section of this report, which was that GMM works well when the data is sampled from a Gaussian distribution, we evaluated this was a meaningful clustering method to consider and implement.

Labeling

Choosing 5 components, or distributions, because of assignment of clustering 5 virus strains, and projecting onto 3-dim and 2-dim space using PCA we got:

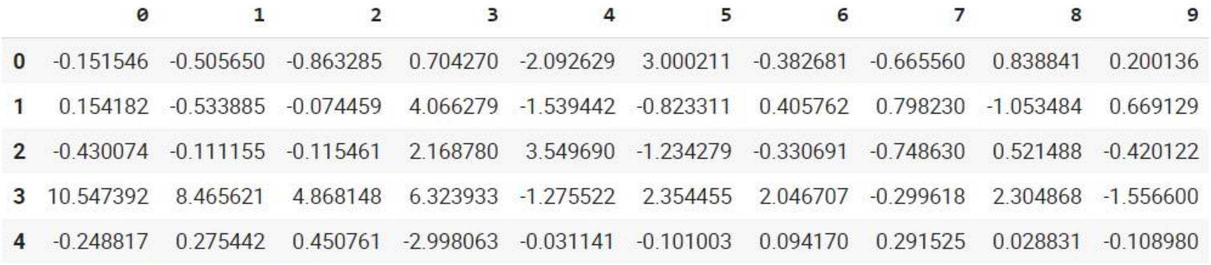




Cluster Analysis

1. Centroids

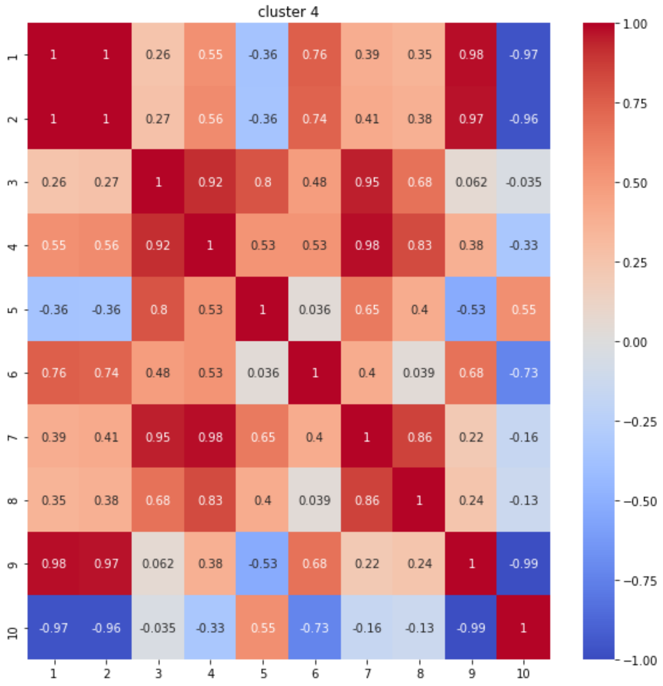
The centroids of each cluster are shown below:

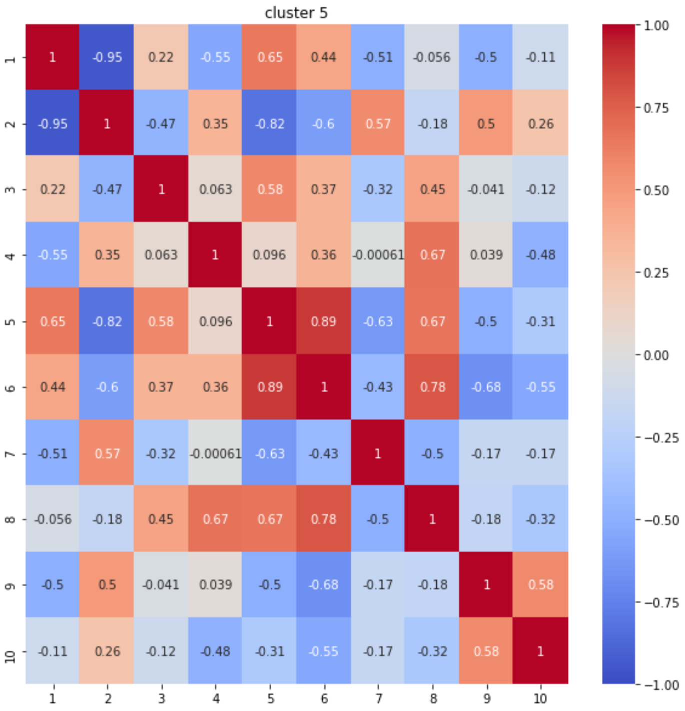


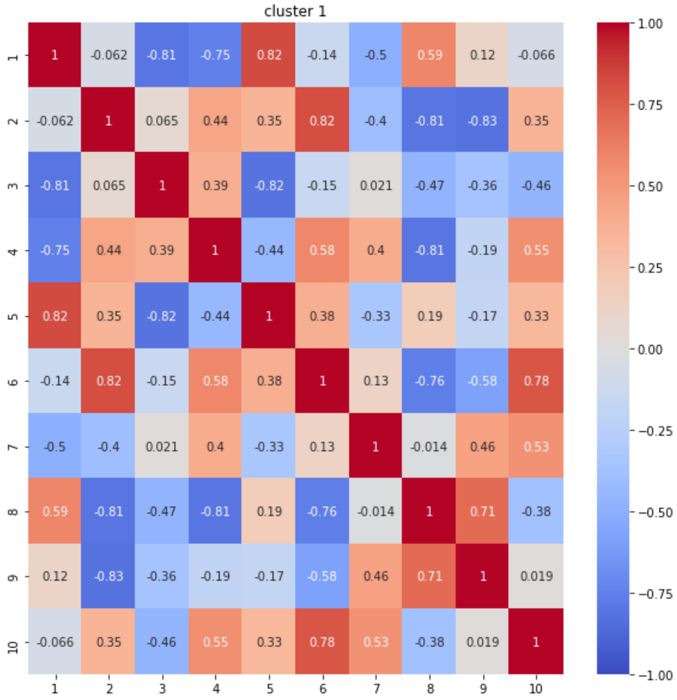
	0	1	2	3	4	5	6	7	8	9
0	-0.151546	-0.505650	-0.863285	0.704270	-2.092629	3.000211	-0.382681	-0.665560	0.838841	0.200136
1	0.154182	-0.533885	-0.074459	4.066279	-1.539442	-0.823311	0.405762	0.798230	-1.053484	0.669129
2	-0.430074	-0.111155	-0.115461	2.168780	3.549690	-1.234279	-0.330691	-0.748630	0.521488	-0.420122
3	10.547392	8.465621	4.868148	6.323933	-1.275522	2.354455	2.046707	-0.299618	2.304868	-1.556600
4	-0.248817	0.275442	0.450761	-2.998063	-0.031141	-0.101003	0.094170	0.291525	0.028831	-0.108980

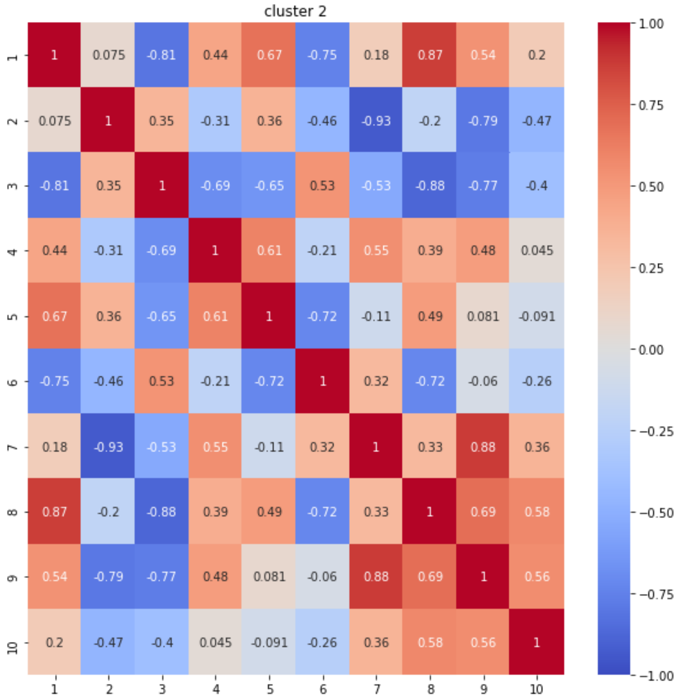
Where each row, i, is a centroid of the i-1 cluster.

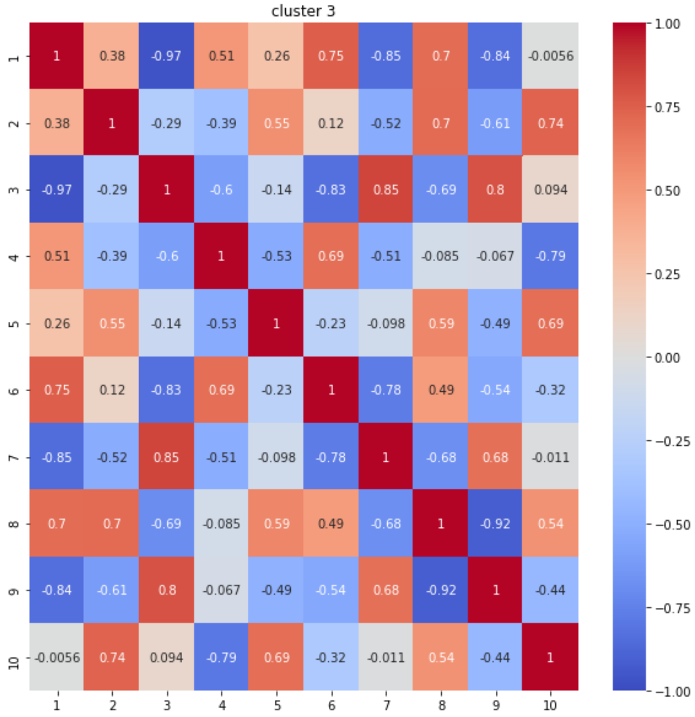
2. Covariance Matrices of Each Cluster





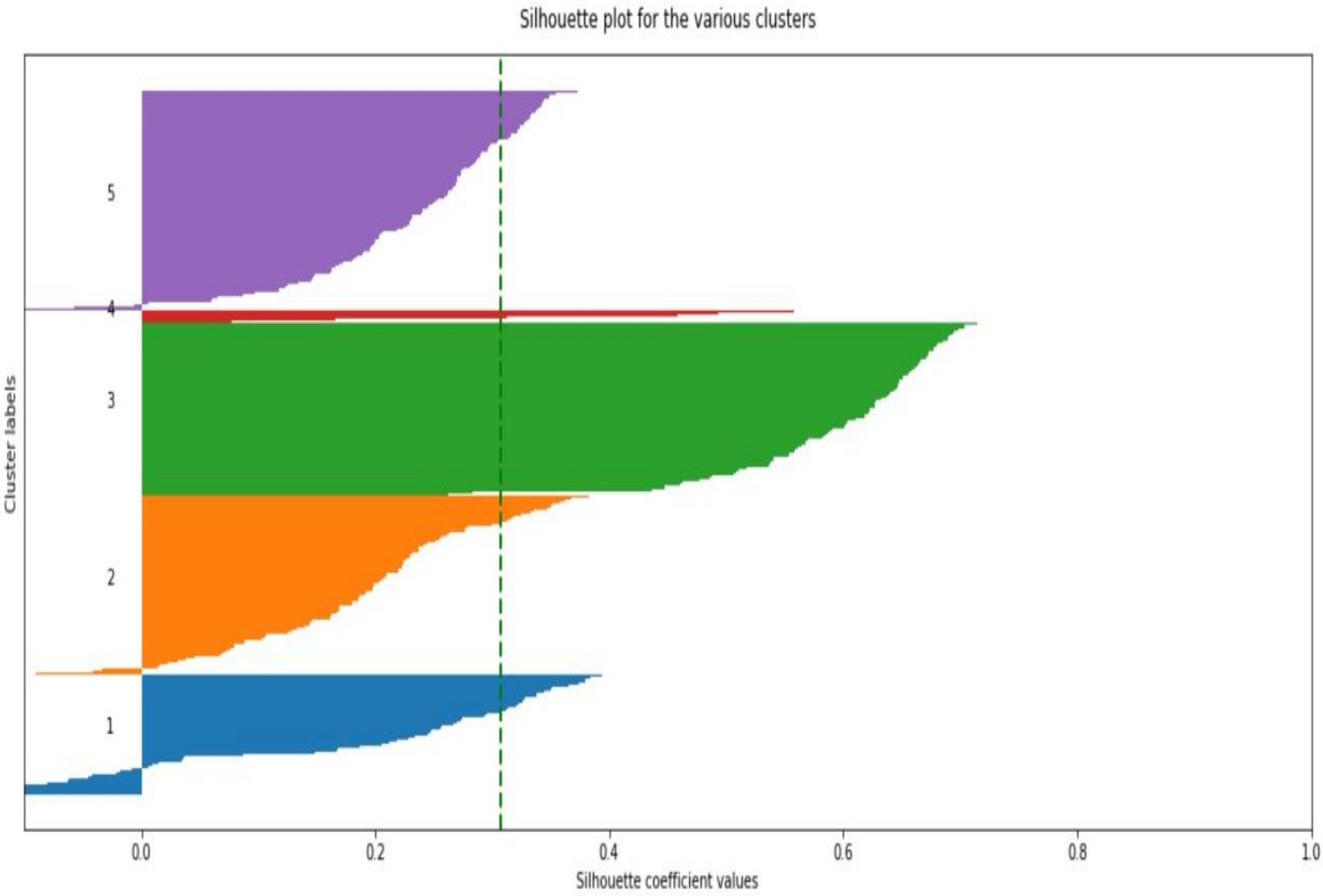

cluster 1
1 -0.062 -0.81 -0.75 0.82 -0.14 -0.5 0.59 0.12 -0.066
-0.062 1 0.065 0.44 0.35 0.82 -0.4 -0.81 -0.83 0.35
-0.81 0.065 1 0.39 -0.82 -0.15 0.021 -0.47 -0.36 -0.46
-0.75 0.44 0.39 1 -0.44 0.58 0.4 -0.81 -0.19 0.55
0.82 0.35 -0.82 -0.44 1 0.38 -0.33 0.19 -0.17 0.33
-0.14 0.82 -0.15 0.58 0.38 1 0.13 -0.76 -0.58 0.78
-0.5 -0.4 0.021 0.4 -0.33 0.13 1 -0.014 0.46 0.53
0.59 -0.81 -0.47 -0.81 0.19 -0.76 -0.014 1 0.71 -0.38
0.12 -0.83 -0.36 -0.19 -0.17 -0.58 0.46 0.71 1 0.019
-0.066 0.35 -0.46 0.55 0.33 0.78 0.53 -0.38 0.019 1
1 2 3 4 5 6 7 8 9 10
1.00 0.75 0.50 0.25 0.00 −0.25 −0.50 −0.75 −1.00


cluster 2
1 0.075 -0.81 0.44 0.67 -0.75 0.18 0.87 0.54 0.2
0.075 1 0.35 -0.31 0.36 -0.46 -0.93 -0.2 -0.79 -0.47
-0.81 0.35 1 -0.69 -0.65 0.53 -0.53 -0.88 -0.77 -0.4
0.44 -0.31 -0.69 1 0.61 -0.21 0.55 0.39 0.48 0.045
0.67 0.36 -0.65 0.61 1 -0.72 -0.11 0.49 0.081 -0.091
-0.75 -0.46 0.53 -0.21 -0.72 1 0.32 -0.72 -0.06 -0.26
0.18 -0.93 -0.53 0.55 -0.11 0.32 1 0.33 0.88 0.36
0.87 -0.2 -0.88 0.39 0.49 -0.72 0.33 1 0.69 0.58
0.54 -0.79 -0.77 0.48 0.081 -0.06 0.88 0.69 1 0.56
0.2 -0.47 -0.4 0.045 -0.091 -0.26 0.36 0.58 0.56 1
1 2 3 4 5 6 7 8 9 10
1.00 0.75 0.50 0.25 0.00 −0.25 −0.50 −0.75 −1.00

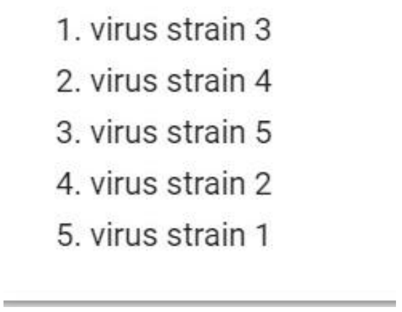

cluster 3
1 0.38 -0.97 0.51 0.26 0.75 -0.85 0.7 -0.84 -0.0056
0.38 1 -0.29 -0.39 0.55 0.12 -0.52 0.7 -0.61 0.74
-0.97 -0.29 1 -0.6 -0.14 -0.83 0.85 -0.69 0.8 0.094
0.51 -0.39 -0.6 1 -0.53 0.69 -0.51 -0.085 -0.067 -0.79
0.26 0.55 -0.14 -0.53 1 -0.23 -0.098 0.59 -0.49 0.69
0.75 0.12 -0.83 0.69 -0.23 1 -0.78 0.49 -0.54 -0.32
-0.85 -0.52 0.85 -0.51 -0.098 -0.78 1 -0.68 0.68 -0.011
0.7 0.7 -0.69 -0.085 0.59 0.49 -0.68 1 -0.92 0.54
-0.84 -0.61 0.8 -0.067 -0.49 -0.54 0.68 -0.92 1 -0.44
-0.0056 0.74 0.094 -0.79 0.69 -0.32 -0.011 0.54 -0.44 1
1 2 3 4 5 6 7 8 9 10
1.00 0.75 0.50 0.25 0.00 −0.25 −0.50 −0.75 −1.00

3. Silhouette Score

Plot



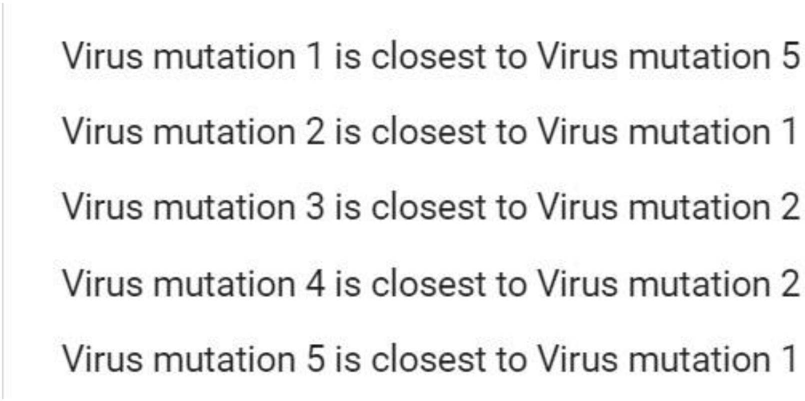
Ranking of Clusters by Silhouette score



1. virus strain 3
2. virus strain 4
3. virus strain 5
4. virus strain 2
5. virus strain 1

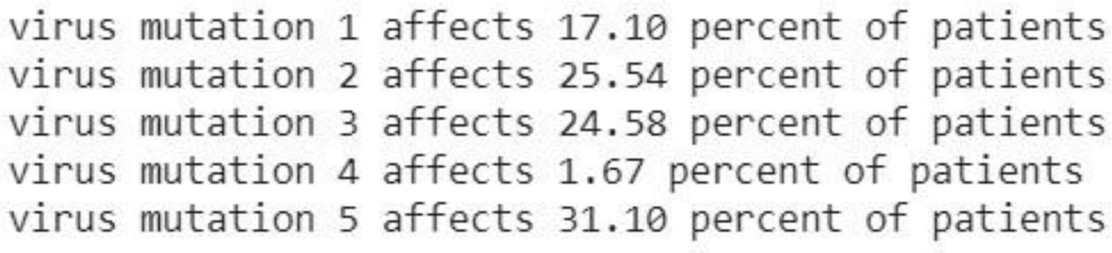
4. Nearest Mutant Virus

By computing pair-wise distances between centroids



```
Virus mutation 1 is closest to Virus mutation 5

Virus mutation 2 is closest to Virus mutation 1

Virus mutation 3 is closest to Virus mutation 2

Virus mutation 4 is closest to Virus mutation 2

Virus mutation 5 is closest to Virus mutation 1
```

5. Prevalence



```
virus mutation 1 affects 17.10 percent of patients
virus mutation 2 affects 25.54 percent of patients
virus mutation 3 affects 24.58 percent of patients
virus mutation 4 affects 1.67 percent of patients
virus mutation 5 affects 31.10 percent of patients
```

Model Evaluation-GMM

We used two internal scores, meaning assuming no prior knowledge of the true labels, silhouette score and Davies-Bouldin score.

Silhouette Score

Explanation:

he Silhouette Coefficient is calculated using the mean intra-cluster distance (a) and the mean nearest-cluster distance (b) for each sample. The Silhouette Coefficient for a sample is (b-a)/max(a,b)

Then take average over all samples in dataset.

Scores from -1 to 1 where the greater score the better cohesion and separation of the clusters.

Result: 0.307

Davies-Bouldin Score

Explanation:

The score is defined as the average similarity measure of each cluster with its most similar cluster, where simi arity is the ratio of within-cluster distances to between-cluster distances. Thus, clusters which are farther apart and less dispersed will result in a better score.

The minimum score is zero, with lower values indicating better clustering.

Result: 1.3

Feature Importance

Method 1 – PCA

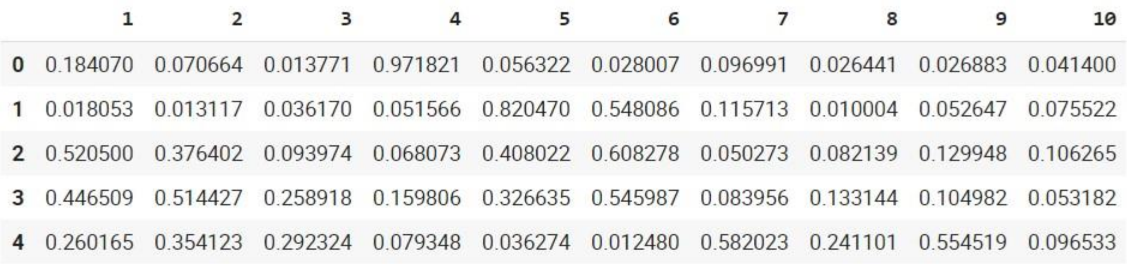
<u>Explanation</u>

Using PCA, we evaluate feature importance by the magnitude of the entries in the Principal Components (feature values) derived in PCA.

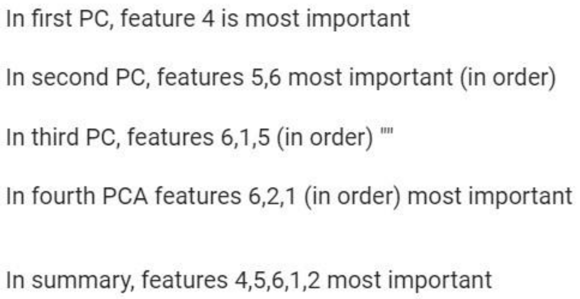
The larger the magnitude of a feature, the higher the importance of it. Ranking of importance done by rank (by magnitude of eigenvalue in SVD) of the Principal Component it belongs to and the magnitude of the feature in the Principal Component.

<u>Conclusions</u>

The Principal Components are:



	1	2	3	4	5	6	7	8	9	10
0	0.184070	0.070664	0.013771	0.971821	0.056322	0.028007	0.096991	0.026441	0.026883	0.041400
1	0.018053	0.013117	0.036170	0.051566	0.820470	0.548086	0.115713	0.010004	0.052647	0.075522
2	0.520500	0.376402	0.093974	0.068073	0.408022	0.608278	0.050273	0.082139	0.129948	0.106265
3	0.446509	0.514427	0.258918	0.159806	0.326635	0.545987	0.083956	0.133144	0.104982	0.053182
4	0.260165	0.354123	0.292324	0.079348	0.036274	0.012480	0.582023	0.241101	0.554519	0.096533



In first PC, feature 4 is most important

In second PC, features 5,6 most important (in order)

In third PC, features 6,1,5 (in order) ""

In fourth PCA features 6,2,1 (in order) most important

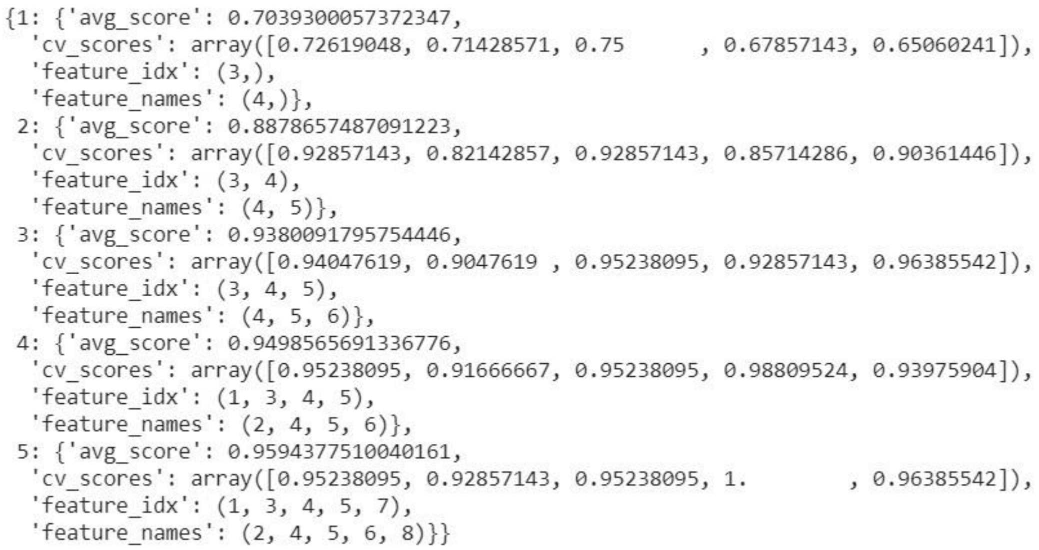
In summary, features 4,5,6,1,2 most important

Analysis based on Multiclass Classification

Method 2- Sequential Forward Selection

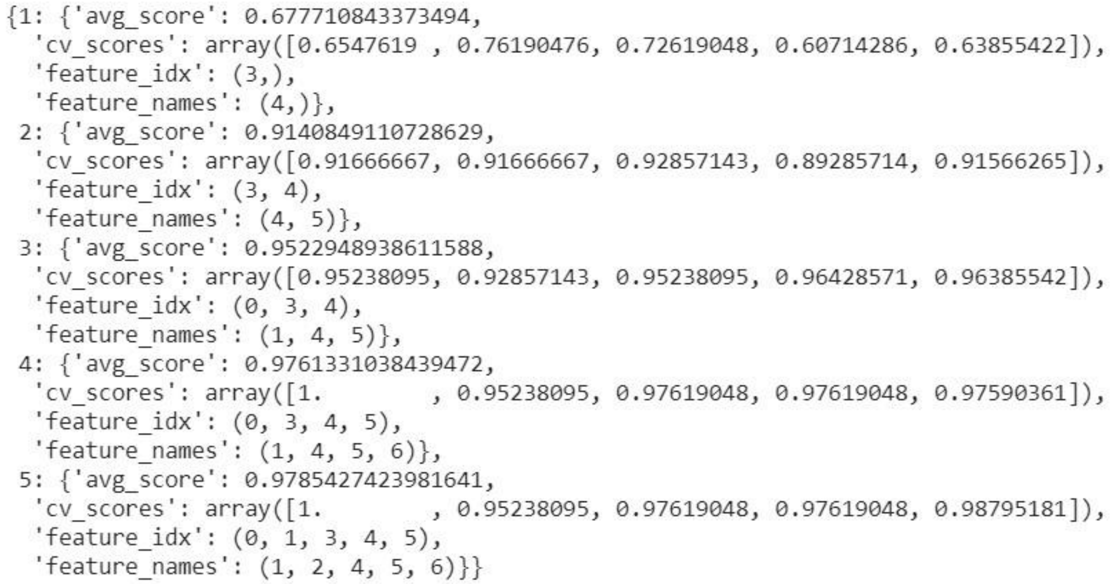
Train classifier on labeled data (labeled by GMM) and then use SFS to obtain 5 most important features, using Decision Tree Classifier and K-nearest Neighbors.

Below we show scores as the SFS progresses score of Decision Tree Classifier:



```
{1: {'avg_score': 0.7039300057372347,
  'cv_scores': array([0.72619048, 0.71428571, 0.75      , 0.67857143, 0.65060241]),
  'feature_idx': (3,),
  'feature_names': (4,)},
 2: {'avg_score': 0.8878657487091223,
  'cv_scores': array([0.92857143, 0.82142857, 0.92857143, 0.85714286, 0.90361446]),
  'feature_idx': (3, 4),
  'feature_names': (4, 5)},
 3: {'avg_score': 0.9380091795754446,
  'cv_scores': array([0.94047619, 0.9047619 , 0.95238095, 0.92857143, 0.96385542]),
  'feature_idx': (3, 4, 5),
  'feature_names': (4, 5, 6)},
 4: {'avg_score': 0.9498565691336776,
  'cv_scores': array([0.95238095, 0.91666667, 0.95238095, 0.98809524, 0.93975904]),
  'feature_idx': (1, 3, 4, 5),
  'feature_names': (2, 4, 5, 6)},
 5: {'avg_score': 0.9594377510040161,
  'cv_scores': array([0.95238095, 0.92857143, 0.95238095, 1.        , 0.96385542]),
  'feature_idx': (1, 3, 4, 5, 7),
  'feature_names': (2, 4, 5, 6, 8)}}
```

Below we show scores as the SFS progresses score of KNN:



```
{1: {'avg_score': 0.677710843373494,
  'cv_scores': array([0.6547619 , 0.76190476, 0.72619048, 0.60714286, 0.63855422]),
  'feature_idx': (3,),
  'feature_names': (4,)},
 2: {'avg_score': 0.9140849110728629,
  'cv_scores': array([0.91666667, 0.91666667, 0.92857143, 0.89285714, 0.91566265]),
  'feature_idx': (3, 4),
  'feature_names': (4, 5)},
 3: {'avg_score': 0.9522948938611588,
  'cv_scores': array([0.95238095, 0.92857143, 0.95238095, 0.96428571, 0.96385542]),
  'feature_idx': (0, 3, 4),
  'feature_names': (1, 4, 5)},
 4: {'avg_score': 0.9761331038439472,
  'cv_scores': array([1.        , 0.95238095, 0.97619048, 0.97619048, 0.97590361]),
  'feature_idx': (0, 3, 4, 5),
  'feature_names': (1, 4, 5, 6)},
 5: {'avg_score': 0.9785427423981641,
  'cv_scores': array([1.        , 0.95238095, 0.97619048, 0.97619048, 0.98795181]),
  'feature_idx': (0, 1, 3, 4, 5),
  'feature_names': (1, 2, 4, 5, 6)}}
```

Conclusion

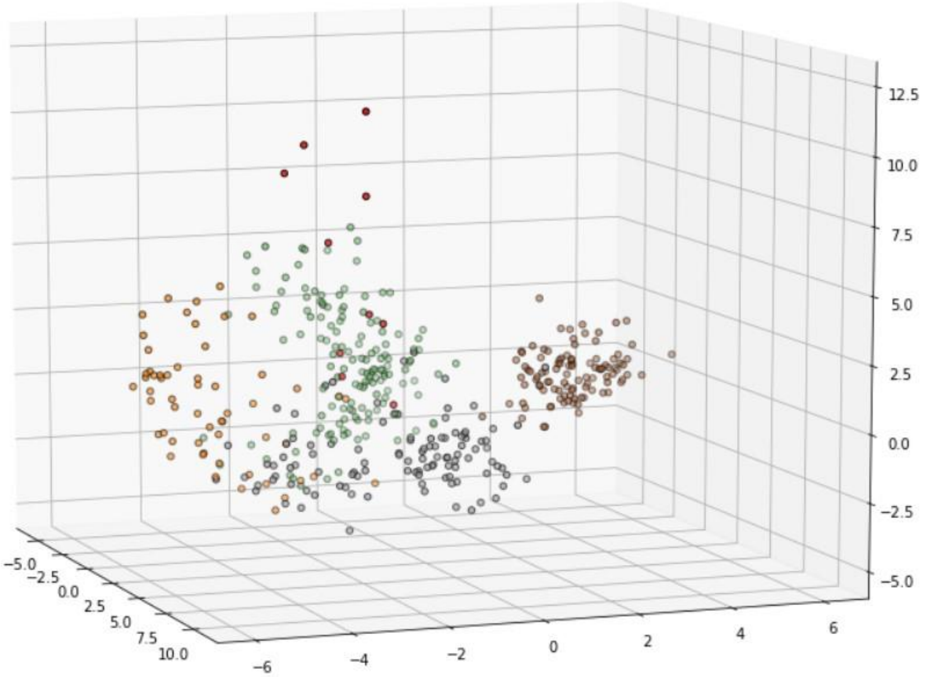
Tree Classifier : Features 2,4,5,6,8 , KNN : Features 1,2,4,5,6

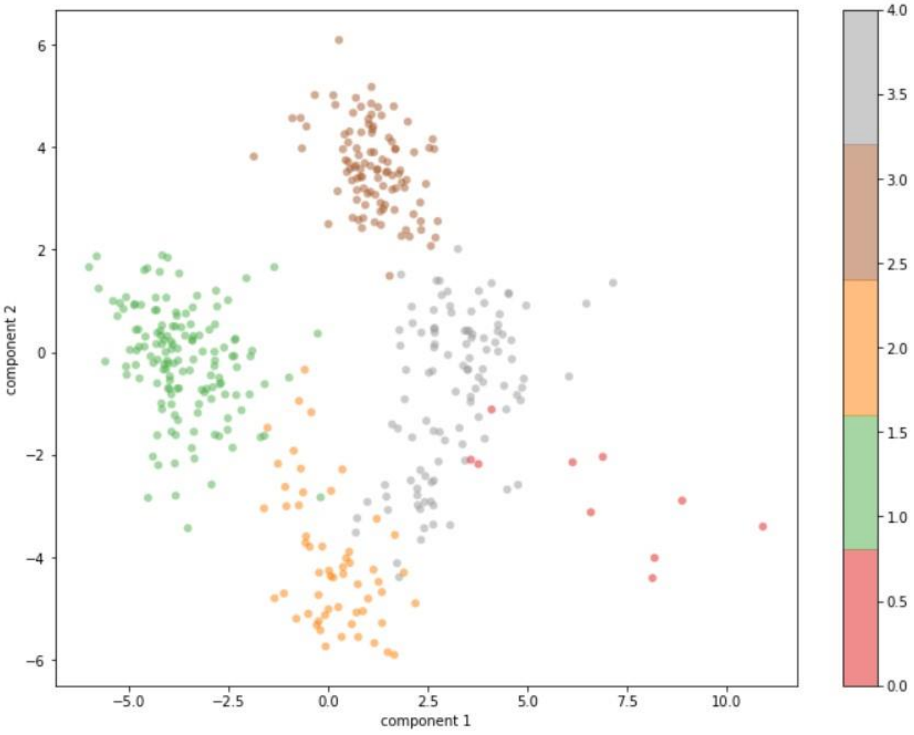
Feature Selection Conclusion

Most prevalent features in all methods are 1,2,4,5,6

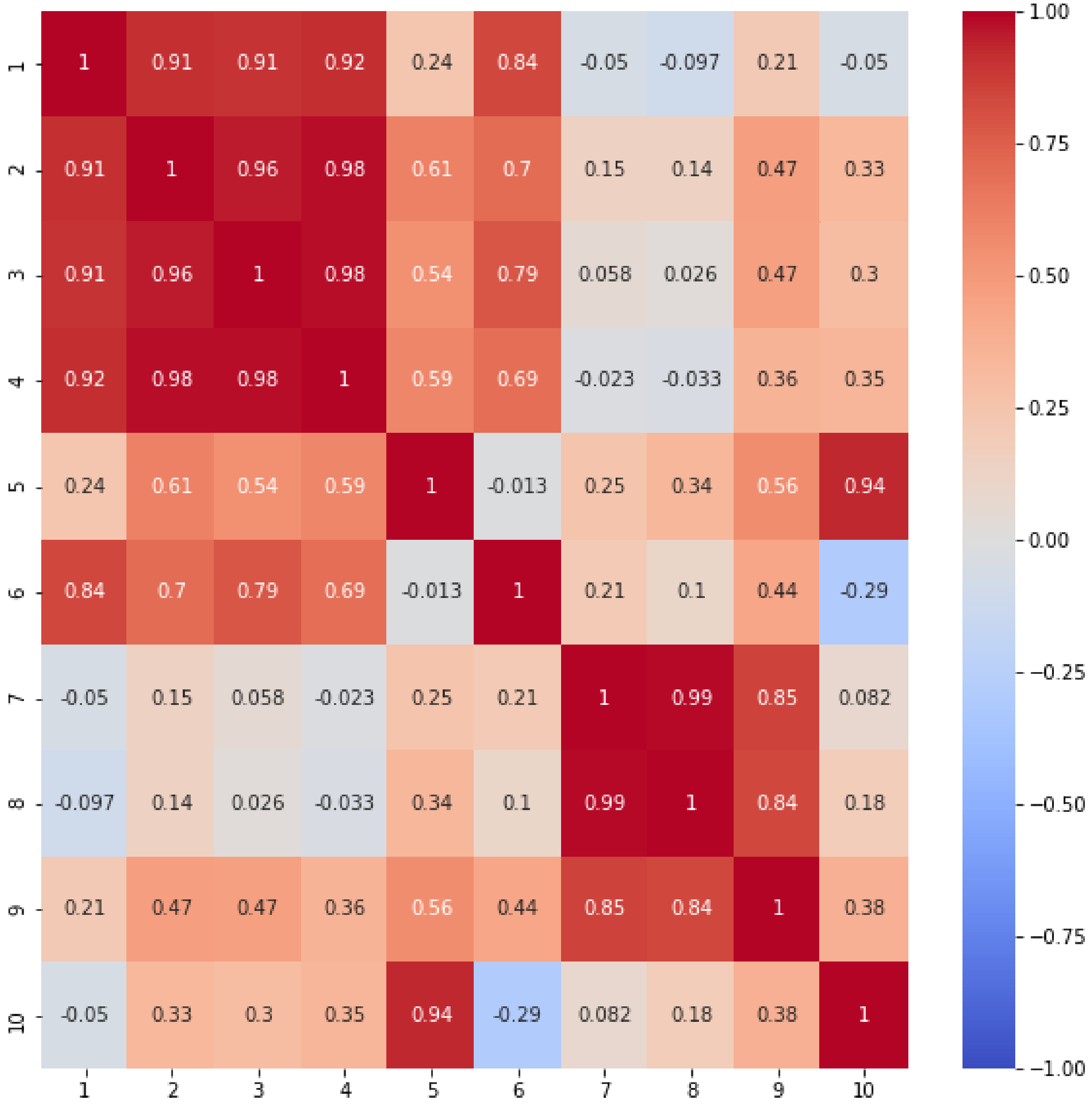
Therefore we chose them as most important.

Validating they are vital for clustering we fit a GMM on dataset with only these features and projected the clusters onto low dim space using PCA and got very cohesive and separated clusters, as desired:





Prior analysis using Spectral Clustering yielded features 7,8 of importance yet fitting a GMM using one covariance matrix for all clusters we obtain the following covariance matrix:

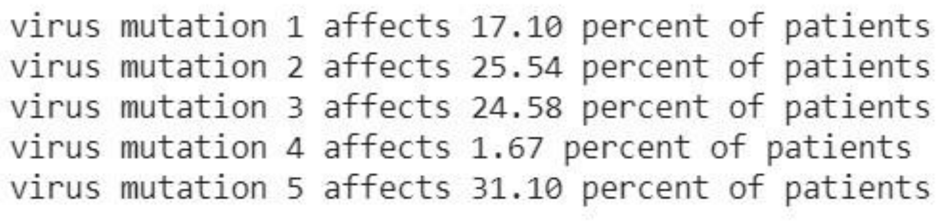


	1	2	3	4	5	6	7	8	9	10
1	1	0.91	0.91	0.92	0.24	0.84	-0.05	-0.097	0.21	-0.05
2	0.91	1	0.96	0.98	0.61	0.7	0.15	0.14	0.47	0.33
3	0.91	0.96	1	0.98	0.54	0.79	0.058	0.026	0.47	0.3
4	0.92	0.98	0.98	1	0.59	0.69	-0.023	-0.033	0.36	0.35
5	0.24	0.61	0.54	0.59	1	-0.013	0.25	0.34	0.56	0.94
6	0.84	0.7	0.79	0.69	-0.013	1	0.21	0.1	0.44	-0.29
7	-0.05	0.15	0.058	-0.023	0.25	0.21	1	0.99	0.85	0.082
8	-0.097	0.14	0.026	-0.033	0.34	0.1	0.99	1	0.84	0.18
9	0.21	0.47	0.47	0.36	0.56	0.44	0.85	0.84	1	0.38
10	-0.05	0.33	0.3	0.35	0.94	-0.29	0.082	0.18	0.38	1

Then features 6,7,8 are highly correlated then choosing feature 6 is sufficient.

Choosing the 3 mutations to treat

The best way will be to treat the 3 mutations which are the most prevalent in the dataset, according to labeling by Gaussian Mixture Models



```
virus mutation 1 affects 17.10 percent of patients
virus mutation 2 affects 25.54 percent of patients
virus mutation 3 affects 24.58 percent of patients
virus mutation 4 affects 1.67 percent of patients
virus mutation 5 affects 31.10 percent of patients
```

We choose to treat virus mutations 2,3,5.