HW 4

Itay waisman – 311177562

Vlad Keel - 312866775

Wet Part – explanation

Data Preparation

We started by preparing the data for usage. We preformed the following tasks on the data:

1. Outlier Detection – we used the z-score technique, assuming the features are normally distributed. We tagged each sample's feature as an outlier value if it were more than one std step from the mean (maximal z-score of 1). We than clipped the value of that sample to be exactly equal to z-score of 1.
2. Imputation – we imputed the empty values using our built Imputer class from previous exercises. The imputer is using the IterativeImputer object from sklearn, which implements a multivariate imputation method, which helps us by not harming the connections between features.

Models

We decided to test 3 models to cluster the data:

1. KMEANS – we used the KMeans class from sklearn, and set n_components=5, in order to detect the 5 different mutations. We set max_iter=500, and n_init=30 which controls the number of times to run the algorithm with different starting conditions. This is done in order to avoid a local optimum point.
2. Spectral Clustering – we used the SpectralClustering class from skelarn, with n_clusters=5, to find the wanted number of mutations. We set n_components=6, as we discovered anything below that value returned less than 5 clusters. We played around with the value of gamma, but seen little evidence this changed the results.
3. GMM – we used the GaussianMixture class from sklearn, to try a softer version of the KMEANS model. Again we set n_components=5. We played around with optimizing the parameters using GridSearchCV (the score function is the model's internal scoring), but the improvements were minor, and as we see later the KMEANS model outperformed any GMM model we tried.

Model Evaluation

We compared the 3 different models using 2 methods: visualizations and scoring functions. Both convinced us that the best model is the KMEANS model.

Scoring

After some research, we discovered the common scoring functions. We chose to focus on two of these functions: davies boulding score and the silhouette score.

The silhouette score measures the average difference between each sample's cluster mean and the closest cluster mean, normalized by the cluster size. This measure is simple to compute and can be intuitively understood as a separation measure for the whole sample set.

The davies boulding score can be explained as measuring how dispersed each cluster is, and we would like each cluster to be as “tight“ as possible, thus a lower score is better.

We got the following scores:

```
KMEANS davies_boulding: 1.3126485827854544
SPECTRAL davies_boulding: 14.034399757310249
GNN davies_boulding: 2.1626187113991966
_____________________________________
KMEANS silhouette_score: 0.26111124912275385
SPECTRAL silhouette_score: -0.14813316429738 31
GNN silhouette_score: 0.1981922203259694
```

For the davies boulding score, lower is better. For the silhouette score, higher is better.

Clearly, the KMEANS provided better results for each scoring method, and we decided to select it as our model.

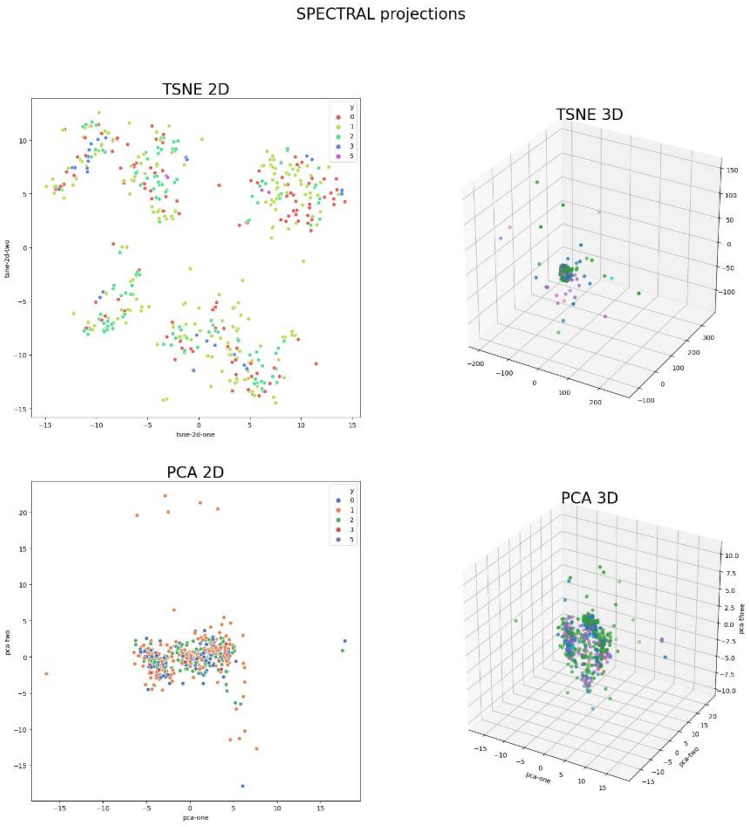
Visualization

We wanted to visualize the clusters on a graph to see which model better differentiate the different mutations. In order to do so, we needed to reduce the dimension of the data, as we had 10 features, but needed only 2 or 3 to plot a graph. To do so, we used 2 methods:

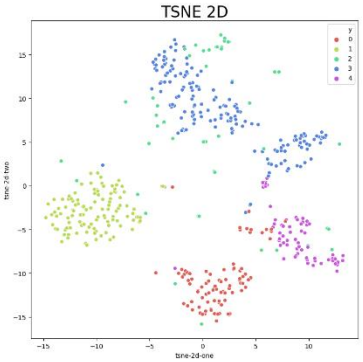
1. PCA – reducing the data to 3 dimensions using the PCA algorithm (provided in sklearn), allowed us to plot the data easily, both in 2d and 3d.
2. TSNE – another dimension reduction method, which seemed to be faster and provided better separation of the data when we plotted it on a 2d or 3d graph.

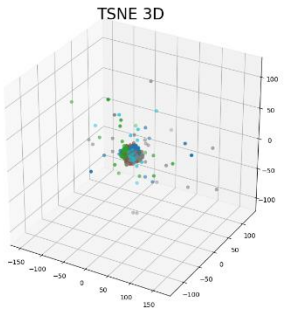
The 2 best separations we see are using the KMEANS and GMM model, while the KMEANS looks slightly better when projected in 3D.

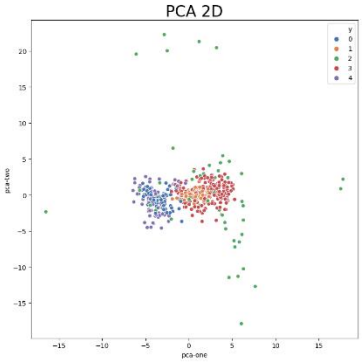
Following are the 4 plots using each reduction method (PCA and TSNE), each in 2d and in 3d. This was computed for each of the 3 models we evaluated.

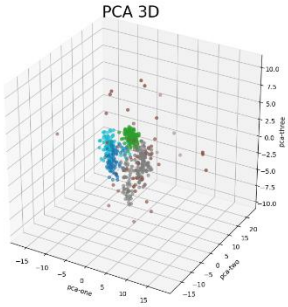


GMM projections

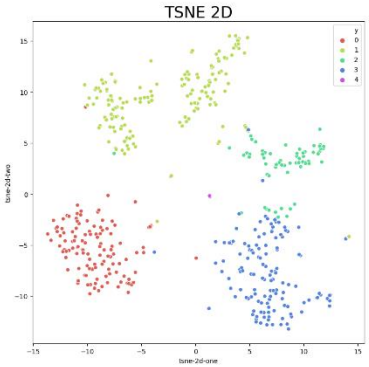


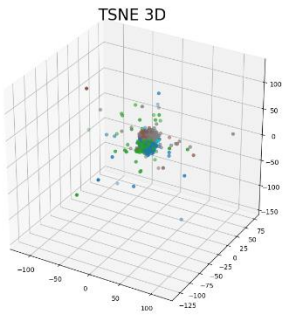


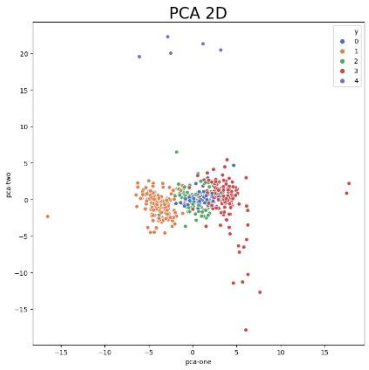


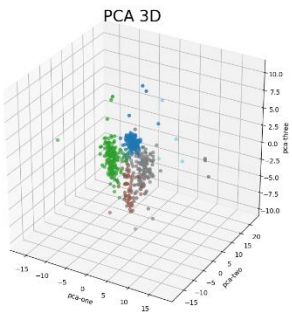


KMEANS projections









Most Important Features

Another task we had was to rank the features by their importance for better clustering overall.

We approached this problem using 4 techniques:

1. Feature score effect – by removing each feature and examining the silhouette score, we could see which features hurt the score the most. We ranked the features using this method and got the set: [7 10 1 9 3] (protein_7, protein_10 etc.). A problem with this approach is that it losses the affect of multiple features together where as a group they might improve the score more than each feature alone.
2. Feature set – we used a subset of the power set of the features, which contained tuples of 5 features each. We then proceded to check each and every tuple and ranked each tuple using both davies boulding and the silhouette scores. This method was extremely slow, and although the scores really improved, when visualizing the data we saw a phenomenon where most of the data was tagged in 2 clusters. This might happen as these 2 scores we chose target specific characteristics of the clusters, while neglecting others.
3. Random forest classifier – we fitted a random forest classifier over our data and predicted clusters from the KMeans model. We used the implementation provided in the sklearn library. This implementation exposes a member called 'feature_importance_' which allowed us to rank the feature. The 5 best features we got are: [4 5 6 8 9]
4. PCA analysis – we used the PCA algorithm, implemented in sklearn, to see what feature affected each resulted principle component. This way, for the first PC, we saw the protein_4 had the highest weight. Going over all 5 PC, we got the feature set: [4 2 6 5 9] – this is very similar to the random forest technique.

In the end we chose the set provided by the random forest method, which provided a slightly better silhouette score against the pca method, when fitting only on the resulted set.

Mutations Characteristics

We calculated 3 characteristics for each mutation we labeled:

1. Mutation Prevalence – we used the method provided by the pandas library to count the percentages each mutation accounts for from the whole population.
2. Mutations centroids – we used the NearestCentroid estimator from sklearn to calculate the centroids of each label (using all 10 features).
3. Each mutation nearest neighbor – we used the centroids to calculate each mutation nearest neighbor. This was done using the NearestNeighbors estimator from sklearn. We used the centroids as a measure for a mutation instead of the features directly in order to deal with outliers and account for the center of mass.

Choosing only 3 mutations

If we had to choose only 3 mutations to vaccinate, we would choose the 3 most common mutation we labeled, using the prevalence we calculated. This assuming the data was representative of the real world.

Another way to choose the mutations is to choose those which are most "spread" across the whole population.

מבוא למערכות לומדות - תרגיל 4

<u>שאלה 1</u>

יהיו $x_i \sim N\left(\mu, \sigma^2\right)$ לכל $i \in [1,m]$ משתנים מקריים בלתי תלויים

הראנו כי $\hat{\mu}_{MLE} = \overline{X} = \frac{1}{m}\sum_{i=1}^{m} x_i$. נראה כעת $\hat{\sigma^2}_{MLE} = \frac{1}{m}\sum_{i=1}^{m}\left(x_i - \hat{\mu}_{MLE}\right)^2$

פונק' הנראות המירבית

$$L\left(\mu, \sigma^2 | x_1, ..., x_m\right) = \prod_{i=1}^{m}\left(2\pi\sigma^2\right)^{-\frac{1}{2}} e^{-\frac{1}{2}\frac{(x_i-\mu)^2}{\sigma^2}} = \left(2\pi\sigma^2\right)^{-\frac{n}{2}} e^{-\frac{1}{2\sigma^2}\sum_{i=1}^{m}(x_i-\mu)^2}$$

אזי log של פונק' הנראות

$$l\left(\mu, \sigma^2 | x_1, ..., x_m\right) = ln\left(\left(2\pi\sigma^2\right)^{-\frac{m}{2}} e^{-\frac{1}{2\sigma^2}\sum_{i=1}^{m}(x_i-\mu)^2}\right) =$$

$$= ln\left(\left(2\pi\sigma^2\right)^{-\frac{m}{2}}\right) + ln\left(e^{-\frac{1}{2\sigma^2}\sum_{i=1}^{m}(x_i-\mu)^2}\right) =$$

$$= -\frac{m}{2}ln\left(2\pi\sigma^2\right) - \frac{1}{2\sigma^2}\sum_{i=1}^{m}\left(x_i-\mu\right)^2 =$$

$$= -\frac{m}{2}ln\left(2\pi\right) - \frac{m}{2}ln\left(\sigma^2\right) - \frac{1}{2\sigma^2}\sum_{i=1}^{m}\left(x_i-\mu\right)^2$$

נגזור לפי σ^2

$$\frac{\partial}{\partial\sigma^2} l\left(\mu, \sigma^2 | x_1, ..., x_m\right) = -\frac{m}{2\sigma^2} - \frac{1}{2}\sum_{i=1}^{m}\left(x_i-\mu\right)^2 \cdot \left(-\frac{1}{(\sigma^2)^2}\right) =$$

$$= \frac{1}{2\sigma^2}\left(\frac{1}{\sigma^2}\sum_{i=1}^{m}\left(x_i-\mu\right)^2 - m\right)$$

נשווה לאפס ונמצא נק' מקסימום

$$\frac{1}{2\sigma^2}\left(\frac{1}{\sigma^2}\sum_{i=1}^{m}\left(x_i-\mu\right)^2 - m\right) = 0 \Longrightarrow$$

$$\Longrightarrow \frac{1}{2\sigma^2} = 0 \Longrightarrow \sigma_1^2 = 0 \text{ נפסל}$$

$$\Longrightarrow \frac{1}{\sigma^2}\sum_{i=1}^{m}\left(x_i-\mu\right)^2 - m = 0 \Longrightarrow \sigma^2 = \frac{1}{m}\sum_{i=1}^{m}\left(x_i-\mu\right)^2$$

ולכן אמד נראות מירבית לשונות $\hat{\sigma^2}_{MLE} = \frac{1}{m}\sum_{i=1}^{m}\left(x_i - \hat{\mu}_{MLE}\right)^2$

Question 2

Part a

Under i.i.d assumptions we can show that:

$$P(w|\mu=0,b) = \prod_{i=1}^{m} P(w_i|\mu=0,b) =$$

$$\prod_{i=1}^{m} \frac{1}{2b} exp(-\frac{|w_i|}{b}) =$$

$$(2b)^{-m} exp(-\frac{\sum_{i=1}^{m}|w_i|}{b})$$

Part b

The lasso regression problem can be written as follows:

$$\hat{w}_{LASSO} = argmin_w \ ||Xw-y||^2 + \lambda||w||_1$$

$$= argmin_w \ \sum_{i=1}^{m}(x_i^T w - y_i)^2 + \lambda||w||_1$$

We can show that:

$$p(\{(x_i,y_i\}_{i=1}^m|w,\mu=0,b) = \Pi_{i=1}^m p((x_i,y_i)|w) = \Pi_{i=1}^m p(y_i|x_i,w)$$

$$= \Pi_{i=1}^m \frac{1}{\sqrt{2\pi}} exp(-\frac{(x_i^T w - y_i)^2}{2})$$

$$= (2\pi)^{-\frac{m}{2}} exp(-\frac{1}{2}\sum_{i=1}^{m}(x_i^T w - y_i)^2)$$

Using our knowledge of the noise $\epsilon_i \sim \mathcal{N}(0,1)$

We can now show that $\hat{w}_{MAP} = \hat{w}_{LASSO}$ using bayes theroem:

$$\hat{w}_{MAP} \triangleq argmax_w \ p(w|\{(x_i,y_i)\}_{i=1}^m, \mu=0,b)$$

$$= argmax_w \ p(\{(x_i,y_i)\}_{i=1}^m|w,\mu=0,b)p(w|\mu=0,b)$$

$$= argmax_w \ ln(p(\{(x_i,y_i)\}_{i=1}^m|w,\mu=0,b)p(w|\mu=0,b))$$

$$= argmax_w \ -\frac{m}{2}\cdot ln(2\pi) - \frac{1}{2}\sum_{i=1}^{m}(x_i^T w - y_i)^2 - m\cdot ln(2b) - \frac{\sum_{i=1}^{m}|w_i|}{b}$$

$$= argmin_w \ \sum_{i=1}^{m}(x_i^T w - y_i)^2 + \frac{1}{b}\sum_{i=1}^{m}|w_i|$$

$$= argmin_w \ ||Xw-y||^2 + \lambda||w||_1$$

$$= \hat{w}_{LASSO}$$

Where the suitable parameter λ is $\frac{1}{b}$.

Part c

The intuition is that by looking at the figure we can see that the cdf is more "concentrated" around 0, than the normal-distributed case.
Translated to the regression (Least Squares) problem, using the above formulation, we can see that this

means that most weight w_i are probably close to 0, and fewer dominent weights are more distant from 0. This is exactly the sparsity we are talking about, as the lasso regressor fits a weights vector w, in which most of the weights are very close to 0, and only a few are different from 0.

שאלה 3

a. יהיו $x_i \sim Poisson(\lambda)$ לכל $i \in [1,n]$ מ"מ בלתי תלויים
פונק' ההתפלגות המצטברת $P(x_i|\lambda) = \frac{\lambda^{x_i}}{x_i!}e^{-\lambda}$. אזי פונק' נראות מירבית

$$L(\lambda|x_1,...,x_n) = \prod_{i=1}^{n} e^{-\lambda}\frac{1}{x_i!}\lambda^{x_i}$$

אזי log של פונק' הנראות

$$l(\lambda|x_1,...,x_n) = ln\left(\prod_{i=1}^{n} e^{-\lambda}\frac{1}{x_i!}\lambda^{x_i}\right) =$$
$$= \sum_{i=1}^{n} ln\left(e^{-\lambda}\frac{1}{x_i!}\lambda^{x_i}\right) = \sum_{i=1}^{n}\left[ln\left(e^{-\lambda}\right) - ln(x_i!) + ln\left(\lambda^{x_i}\right)\right] =$$
$$= \sum_{i=1}^{n} -\lambda - ln(x_i!) + x_i ln(\lambda) = -n\lambda - \sum_{i=1}^{n} ln(x_i!) + ln(\lambda)\sum_{i=1}^{n} x_i$$

נגזור ונמצא מקסימום

$$\frac{d}{d\lambda} l(\lambda|x_1,...,x_n) = -n + \frac{1}{\lambda}\sum_{i=1}^{n} x_i$$

נשווה לאפס

$$-n + \frac{1}{\lambda}\sum_{i=1}^{n} x_i = 0 \implies \lambda = \frac{1}{n}\sum_{i=1}^{n} x_i$$

b. i. נגדיר

$$Pr(x_i|\theta) = \sum_{j=1}^{K} P(x_i|c_j)P(c_j) = \sum_{j=1}^{K} \lambda_j e^{-\lambda_j x_i}\alpha_j$$

ii. 1. המידע השלם מורכב מקבוצת המשתנים המקריים
הנצפים x_i, והמשתנים המקריים החבויים $\{z_{i,j}\}$
כ"ש Z וקטור 0 פרט לרכיב אחד המתאים לאחת ההתפלגויות
בתערובת המתאים לדגימה x_i.
נראות המידע השלם $p(x,z|\lambda)$ נתונה ע"י

$$L_c(x_i,z_i|\theta) = log\left(\prod_i P(x_i,z_i|\theta)\right) =$$
$$= \sum_i log\left(P(x_i|z_i,\theta)P(z_i|\theta)\right) = \sum_i log\left(\lambda_{z_i} e^{-\lambda_{z_i}x_i}\alpha_{z_i}\right)$$

ומהסתברות שלמה

$$L(x_i|\theta) = log\left(\prod_i P(x_i|\theta)\right) =$$
$$\sum_i log\left(\sum_{j=1}^{K} P(x_i|z_i=j,\theta)P(z_i=j|\theta)\right) = \sum_i log\left(\sum_{j=1}^{K}\lambda_j e^{-\lambda_j x_i}\alpha_j\right)$$

בגלל הסכום בתוך הlog מביאה למשוואות סתומות, לכן קשה
לעבוד עם פונק' הנראות הנצפית ולכן נעבוד עם נראות המידע השלם.

2. בשלב זה נשתמש בערך נתון λ, α ונחשב:

$$P(z_i=j|x_i,\theta) = \frac{P(x_i|z_i=j,\theta)\alpha_j}{\sum_{l=1}^{K} P(x_i|z_i=l)\alpha_l}$$

ומתקיים $P(x_i|z_i=j,\theta) = \lambda_j e^{\lambda_j x_i}$ וגם $\alpha_j = P(z_i=j|\theta,x_i)$

3. נגדיר את F פונק' התוחלת של log הנראות של המידע הנצפה

$$F(Q,\theta) = E\left(\sum_i log\left(P(x_i,z_i|\theta)\right)\right) =$$
$$= \sum_i \sum_j P(z_i=j|x_i,\theta) log\left(P(x_i,z_i=j)\right) =$$
$$= \sum_i \sum_j Q_{i,j} log\left(P(x_i,z_i=j|\theta)\right) =$$
$$= \sum_i \sum_j Q_{i,j}\left(log\left(P(z_i=j)\right) + log\left(P(x_i|z_i=j)\right)\right) =$$
$$= \sum_i \sum_j Q_{i,j}\left(log(\alpha_j) + log\left(\lambda_j e^{-\lambda_j x_i}\right)\right) =$$
$$= \sum_i \sum_j Q_{i,j}\left(log(\alpha_j) + log(\lambda_j) - \lambda_j x_i\right) =$$

כאשר $Q_{i,j} = P\left(z_i = j | x_i, \theta\right)$

4. הפרמטרים שמחושבים בשלב הם λ, α.

כדי למצוא λ נגזור את F ונשווה ל0:

$\frac{\partial}{\partial \lambda_j} \sum_i \sum_j Q_{i,j} \left(log\left(\alpha_j\right) + log\left(\lambda_j\right) - \lambda_j x_i\right) =$

$= \sum_i Q_{i,j} \left(\frac{1}{\lambda_j} - x_i\right) = 0 \Longrightarrow$

$\Rightarrow \hat{\lambda_j} = \frac{\sum_i Q_{i,j}}{\sum_i Q_{i,j} x_i}$

כדי למצוא α נזכור שצקיך להתקיים $\sum_j \alpha_j = 1$ ולכן

$\frac{\partial}{\partial \alpha_j} \sum_i \sum_j Q_{i,j} \left(log\left(\alpha_j\right) + log\left(\lambda_j\right) - \lambda_j x_i\right) + \lambda \left(1 - \sum_j \alpha_j\right) =$

$= \sum_i \frac{Q_{i,j}}{\alpha_j} - \lambda = 0 \Longrightarrow$

$\Rightarrow \sum_i Q_{i,j} = \lambda \alpha_j$

נסכום על j

$\sum_j \sum_i Q_{i,j} = \lambda \sum_j \alpha_j \Longrightarrow \lambda = n$

נציב חזרה ונקבל

$\sum_i \frac{Q_{i,j}}{\alpha_j} - n = 0 \Longrightarrow$

$\hat{\alpha_j} = \frac{\sum_i Q_{i,j}}{n}$