

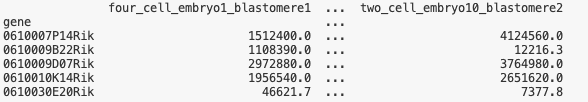
# Assignment 3

|  |  |  |
| --- | --- | --- |
| Date:  Course:  Name:  Student number: | 04/05/2025  MMG1344H  Maria Eleni Fafouti  1010799094 | |
|  | |  |
|  | |  |
|  | |  |

# Part 1

Question 1:

To begin with, I imported the data as a csv file using pandas. When inspecting the data, it looks like this:



We can see that each row is a gene and each column is a sample. There are 6812 genes and 40 samples in total.

Then, to standardize the data, I computed the means and the standard deviation across all rows. After subtracting the means from all data points and dividing by standard deviation, I obtained the following:



This shows that the mean values are very close to zero, while the standard deviation values equal to 1, which is what we were aiming for.

To implement the k-means algorithm as a python function, I was given a few parameters (k=4, using Euclidian distance) but I also had to think of a few key concepts: how many would the maximum iterations be and how large should the tolerance be. I picked 100 maximum iterations as a safe upper limit, as k-means tends to converge at about 20-30 iterations. To ensure that the algorithm does not keep iterating for no reason, I chose to set a convergence threshold at 1e^-4. This value is small enough to ensure precision and make the algorithm stop when the clusters are stable enough (which is usually well before the 100th iteration). Larger values than this one could be too permissive, making the algorithm stop too early and hence sacrifice precision for faster computation.

The algorithm takes a matrix as input, where the rows correspond to samples and columns correspond to features (genes, in this case). After that, the algorithm picks 4 (k=4) samples randomly to become the “centroids”, without allowing for a sample to be chosen twice. These centroids will evolve to be the centres of the clusters once the algorithm completes its run. The algorithm performs a specific sequence of tasks for 100 iterations: for each sample, it computes the Euclidian distances to all centroids and assigning the sample to the one where the distance is the smallest. After completing that for all samples, it proceeds to recompute the centroids. For each centroid, the new value will be the mean of all samples assigned to that specific cluster. After that, the algorithm checks how much the centroids moved i.e how different is the centroid mean now compared to the previous iteration. If that value is above the tolerance threshold, then the algorithm jumps into another loop to improve cluster precision. If the value is below the tolerance threshold, the iterations stop and an array containing the labels assigned to the samples is returned.

Below is a snippet showing the sample ids in each cluster after one run, using seed=42.

A screenshot of a computer code

AI-generated content may be incorrect.

Question 2:

To run the algorithm 10 times, using a different random seed each time, I wrote code that loops 10 times (from 0 to 9) and implements the kmeans function on the given data that I standardized for the previous part. To make sure that the initiation of centroids is different for each seed, I implemented a print statement, which confirmed the following:

A screenshot of a computer code

AI-generated content may be incorrect.

After completing all 10 runs, I saved the results in a matrix which I plotted on the heatmap below:

A chart of multiple colors

AI-generated content may be incorrect.

We can observe that cluster membership differs in each run, but overall, there is a pattern where the two-cell and four-cell embryos appear to be in different clusters. For example, in the first attempt, the four-cell embryo comprises cluster 4, while the two-cell embryo is divided among the rest of the clusters.

Note: The naming of clusters here starts from 1 while in the previous example it started from 0.

Question 3:

To identify the best clustering run, I defined a function that loops over all clusters of a specific run and computes the Within-Cluster Sum of Squares (WCSS). This is a measure of how close the data points in a cluster are arranged around the centroid. Here are the WCSS values computed for all the runs:

A screenshot of a computer

AI-generated content may be incorrect.

The smallest value out of all corresponds to the run with seed=7, which is the run I will be picking as the best one. Looking at the heatmap above, we can appreciate that seed=7 performs a nice division between the two-cell and four-cell embryos with clusters being composed of entirely 4-cell embryo or 2-cell embryo samples.

To compare gene expression within a cluster, we will use the raw data. The standardized data allowed us to avoid certain genes with very large absolute FPKM values dominating the Eucledian distances. Now that we have defined the clusters, we wish to compare whether genes are expressed higher or lower biologically. I defined a function for the upcoming analysis, which loops over all the clusters and for each cluster it compares the expression of every gene (6812 in total) within the cluster versus all other clusters (3 in our case). This comparison is done using a Mann-Whitney U test and the p values are recorded. Then, the Bonferroni correction is applied, to avoid false positive results i.e genes that were found significant by chance. This helps to adjust the p value, considering that we checked 6812 genes and performed 6812\*4 (k=4) tests. Then, we can compare this new value to the significance threshold (0.05) to decide if the effect of a specific gene is significant.

I implemented that function to the genes in the clustering result using seed=7, which was determined to be the best one (as explained above). Below is a short list of the top 5 significant genes per cluster, for the ones that do have significant genes.

A screenshot of a computer screen

AI-generated content may be incorrect.

Note: A full list has been saved as a .txt file and is included in the submission.

Question 4:

After looking at the heatmap in the previous question, we can see that there are two types of clusters: two major and two minor ones. The major clusters are the ones containing more samples (in the heatmap above, that is cluster 1 -blue- and 2 -yellow-, which correspond to clusters 0 and 1 in the list of significant above). There is one major cluster per embryo i.e one for the 4-cell embryo and one for the 2-cell embryo. Then, there is also the minor clusters which contain a smaller number of samples. Again, there is one minor cluster per embryo type. In the significant genes list, we can see that those two minor clusters do not have any significant genes.

# Part 2

Question 1:

First, I had to unzip the folder and inspect it. Once opening it, I noticed that it was comprised of an input folder which contained the following:

A screenshot of a computer

AI-generated content may be incorrect.

I proceeded to load the mutations and control sample csvs onto my program. I then initiated an empty dataframe with columns for the chromosome, position, nucleotide, and the VAF for each control sample.

My program iterates over each mutation in the mutations.csv file and searches for it in each of the control sample csv file. To do this, each of those files is loaded and the code checks for the specific mutation. If it is found, the corresponding VAF value is extacted and if not the value 0 is assigned.

For each mutation, a new row is added to the dataframe until the program has looped over all the control sample files.

Note: The folder is saved in a csv file and will be submitted along with all files required for this assignment.

Question 2:

Here we wish to check if a variant that was observed in patients, was also observed at least once in the controls. If this is the case for a given variant, we are fitting an exponential distribution. This will help us model background noise and understand which variants might just appear by chance and are actually not related to the disease. Usually, random and low-frequency mutations present very low VAF which drops quickly as the frequency increases. This can be captured well with the exponential distribution, while it requires only 1 parameter to fully describe it: lamda. Hence, we can quantify the typical background noise for a variant and see how the patients VAF compares to that noise. If the value in patient is much higher, then the variant is likely real.

The output of this computation is saved as a .csv file as instructed.

Question 3:

Here, we wish to compare the patient-variant pairs at diagnosis and how they compare to the ones after treatment. We want to see if a given variant is significantly above the background level (which was modelled with the exponential distribution in the previous question). After treatment, we are expecting the VAF to decrease and approach background level and hence not be detected to be significantly high anymore.

First, I created a table showing the VAF for each patient, in the same way that I did for controls in question 1. Then, I merged the VAF per patient table with the rate fitting table we created earlier.

In the merged dataframe, for each patient VAF I computed the probability of seeing a VAF at least as large as the one observed, assuming the variant is background noise (and hence not truly present). I am saving the p-values as a .csv in the next question but here’s a preview of the result, before performing any corrections:

A screenshot of a computer

AI-generated content may be incorrect.

In general, high p-values indicate that the VAF is consistent with background noise, while low p-values indicate possible mutation signal.

Question 4:

After obtaining nominal p-values per patient-variant pair, we need to correct for multiple comparisons because the likelihood of false positives increases when testing multiple hypotheses at the same time. The Benjamini-Hochberg (BH) correction controls the False Discovery Rate (FDR). This method outputs q-values, which are essentially the FDR-adjusted p-values. Then we set two FDR thresholds: below or equal to 0.01 or below/equal to 0.05. After this, we are able to detect significant results while controlling for FDR. Here is a summary of the result:

Variants Significant at FDR ≤ 0.01: 28

Variants Significant at FDR ≤ 0.05: 33

A .csv file will be included with the submission.

Question 5:

To complete this task, I need to compare two variables: patient-variant pairs where the VAF is significantly above background after treatment and whether that patient-variant pair was observed at diagnosis. For the first variable, I need to look into the table we created in question 4, where we tabulated the q values showing whether a VAF is significantly higher than the background. For the second variable, I will look at the mutations table, to determine the status at diagnosis.

Using the information from both tables, I created two more columns in the question 4 table (with the FDR values): a Boolean for whether a VAF is significant and another Boolean for whether that variant was observed at diagnosis. Additionally, I included all the variant information from all patients, using a table created in one of the previous questions. I then proceeded to construct a contingency table and performed the fisher’s exact test.

A close-up of a white background

AI-generated content may be incorrect.

The Fisher’s test tells us if the observations are independent and we can see that the p-value it generates is very small, meaning that we reject the null hypothesis. This indicates a strong association between being observed at diagnosis and being significant after treatment.

In conclusion, the samples do not appear to have been randomly swapped because if they had been we would expect no association between diagnosis-time variants and post-treatment significance. Instead, the data shows enrichment of diagnosis-time variants that are significant post-treatment. The biological reason for this is that they are potentially persistent or clonal variants.

Question 6:

Applying the Benjamini-Hochberg procedure to adjust nominal p-values for multiple testing among variants detected at diagnosis was performed in a similar way as in question 4. At an FDR threshold of 0.05, 29 patient-variant pairs remained significantly above background; at FDR ≤ 0.01, fewer remained significant. The relevant .csv file was saved for submission.

Question 7:

None of the variants observed at diagnosis became non-significant after treatment, indicating no full treatment responses under the strict definition of this assignment. This means that 0% of patients showed a full molecular response by the criteria of diagnosed variants becoming statistically insignificant post-treatment.

This suggests persistent molecular traces of disease or limited sensitivity to treatment. While no full responses were observed, partial responses (e.g., VAF reductions) could still be explored in future analysis.

It is possible that the lack of full responses could reflect technical issues such as sample mixing or contamination, where reads from the diagnosis samples inadvertently introduced into post-treatment samples may have falsely preserved variant signal.