

Data report on the methylation of cell-free DNA

Daniel, Thomas & Hans-Henrik

May 17, 2024

Contents

1	Quick overview of repo	2
2	Introduction	2
3	The data	3
4	Normalizing the data	4
4.1	Normalizing with the background	4
4.2	PCA	4
5	Data exploration	5
5.1	Even and Odd	6
6	Initial model	9
6.1	Methylation classification model	9
6.2	Tumor classification	9
6.3	ROC curve	9
7	The straight model	11
8	Conclusion	12

1 Quick overview of repo

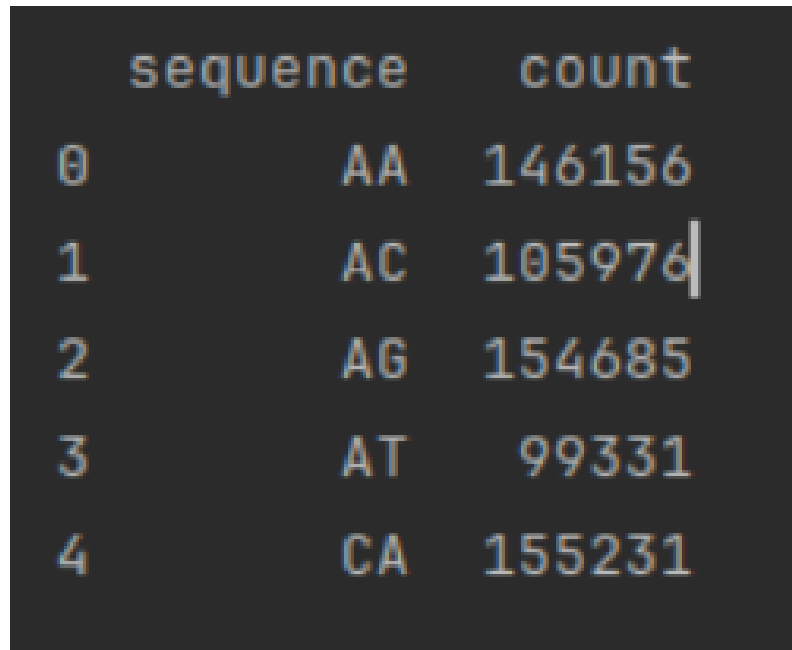
This is a quick overview of the structure of the repo. We have divided everything into various folders, we highly recommend reading the introduction here in order to understand the purpose of the project. Starting from the top we have the "figures" folder. This and certain other folders contain 3 subfolders for the various areas of our project, each of these subfolders contain more folders in which various figures and tables relating to the project can be found. The "Model" folder contains joblibs of our models created by the scripts and notebooks in this repo. The "notebooks" folder contains jupyter notebooks as well as R markdowns for various processes going from creating to models to organizing our data. This is closely related to the "Scripts" folder this contains scripts of various kinds, do note that while some notebooks do have script counterparts not all do, and not all scripts have a notebook. The data is split across 3 folders. The zip files are found in "zip_files" we extracted this into the "raw_data" folder. Which also includes combined tables. The "processed_data" is the data after being normalized, this is primarily the data we used in our various processes. The "tex" folder contains all of our LaTeX related items.

2 Introduction

In this project we will be looking at attempting to implement a model to predict, based on data from blood analysis, whether a given person potentially has cancer. The method in which we will be doing this, is by looking at methylized/unmethylized cell-free dna in peoples blood, specifically the fragmentation patterns. This is preferable to other methods, since it is both less invasive, and less expensive.

The data we will be looking at in this project, has been provided to us by Søren Bessenbacher, and contains information regarding fragmentation patterns for approximately 230 people diagnosed with cancer, and 230 people as a control group.

3 The data



	sequence	count
0	AA	146156
1	AC	105976
2	AG	154685
3	AT	99331
4	CA	155231

Figure 1: An example of the data

In the first column we have the k-mer, in this specific figure it is a 2-mer, and in the second column we have the count of that specific sequence. In addition to the control and the test samples, we also have a background file, which details the total number of combinations of sequences.

4 Normalizing the data

The raw data with just the counts, would likely not work, thus we want to normalize the data. The way in which we do this is by taking the sum of all the counts in a given file, and then dividing each count cell by this sum, thus giving us a ratio of the data. The way in which we've programmed this can be found in the *scripts/* folder of this repository, called *normalize_data.sh*. This script does as previously mentioned and writes the ratios into a new file, which can be found in the folder *processed_data/normalized_data*. We then combine all these matrices into a single matrix with a R-script, which can also be found in the *scripts/*. The new file can be found in the the *processed_data/combined_data/*.

4.1 Normalizing with the background

In order to see the ratios of the samples compared to the potential ratios of the region in question, one can also normalize with the background file found in each k-mer folder. The way in which we approached this, was as previous by taking the sum of each count column, and then in addition we divide each cell in each sample with their respective background ratio. We did this with the *normalize_data_with_background.sh* file, which can be found in the *scripts/* folder.

4.2 PCA

To make the data approachable, it would be desirable to transform the data into a smaller dimension, we do this by using *Principal Component Analysis*. The way in which do this is by using the function from *sklearn* called PCA. An example could be:

```
1 from sklearn.decomposition import PCA
2 ...
3 def pca_fit(data):
4     pca = PCA(n_components=2)
5     pca.fit(data)
6     data_pca = pca.transform(data)
7
8     return data_pca
9 ...
```

5 Data exploration

Now that the data has been normalized, we thought it natural to explore the data, and see if we could find something. The first thing we thought of was using *clustering*. We thought of using *k-means clustering*, and again we used the function *sklearn*. One can see an example of how we implemented this:

```
1 from sklearn.cluster import KMeans
2 ...
3 def kmeans_cluster(data, n_clusters):
4     kmeans = KMeans(n_clusters=n_clusters)
5     kmeans.fit(data)
6     y_kmeans = kmeans.predict(data)
7     plt.scatter(data[:, 0], data[:, 1], c=y_kmeans, s=50, cmap='viridis')
8     centers = kmeans.cluster_centers_
9     plt.scatter(centers[:, 0], centers[:, 1], c='black', s=200, alpha=0.5)
10 ...
```

One can also see an example of the resulting plot from taking the *combined_2mers* from the folder *processed_data/combined_data/with_background* here:

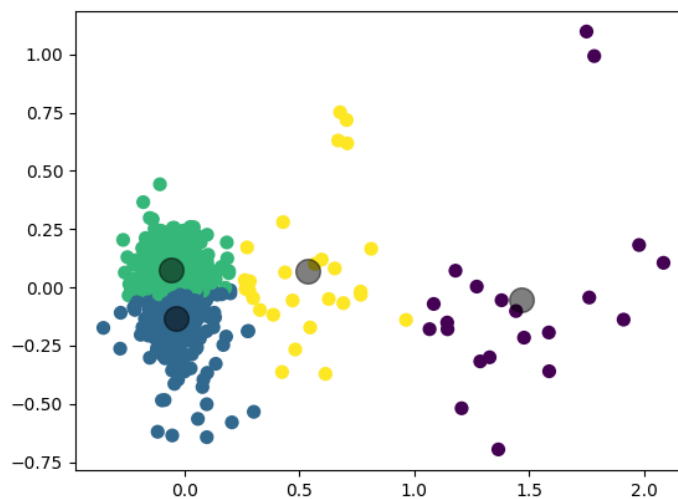


Figure 2: A k-means clustering of the combined 2-mers file

Since we have; methylated/unmethylated, cancer/healthy. Which would result in four groupings of the data. We chose to make 4 clusters of the data.

The figure above is for our data normalized with the background, we have also looked into how the clusters look just normalizing each sample individually by looking at the ratios within the sample.

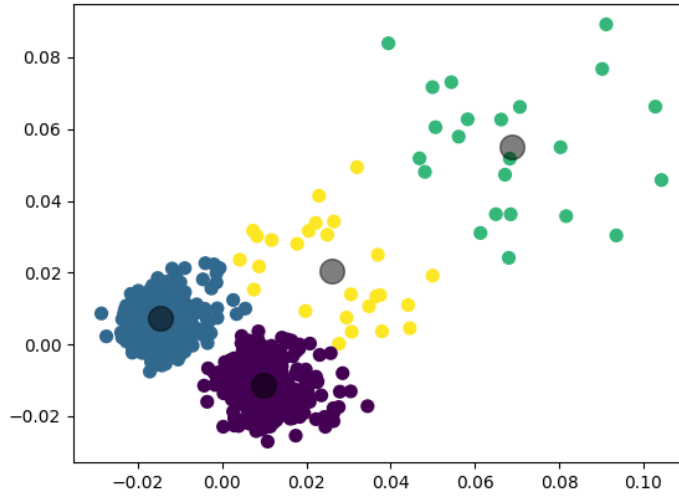


Figure 3: A k-means clustering of the combined 2-mers file without proper normalization

As you can see the divide between clusters is somewhat more clear here, however the eventual goal is distinguishing our cases from controls is more difficult using this form of normalization.

5.1 Even and Odd

In order to test our normalization we had the data split into only taking the even/odd lines from each sample, therefore meaning that there should be a noticeable difference between odds and evens despite both being either methylated or unmethylated. To make this process more clear we focused purely on the controls.

Unlike before where we were primarily interested in just seeing whether these clusters just existed, now we are interested in if we are getting 2 distinct clusters for our 4 groups.

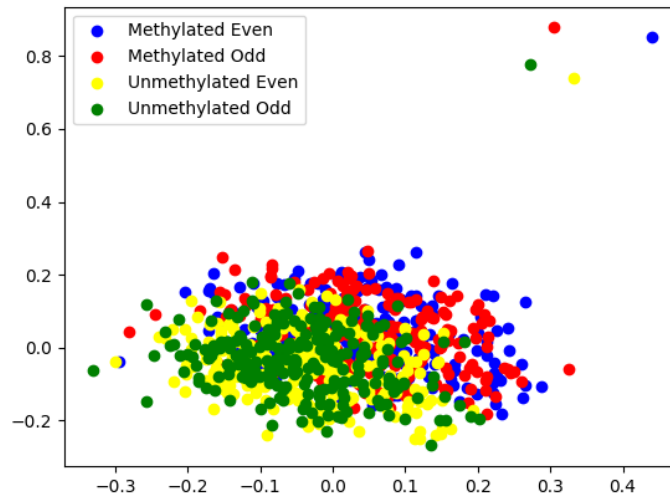


Figure 4: A PCA plot of the 4 groups

Here we are getting 1 large cluster although we are seeing that we are getting close to having 2 separate clusters. There is however 1 very clear outlier that may be affecting the PCA since one of the 2 components we are plotting may just be the variation caused by this outlier. Therefore we removed it and did the same plot again

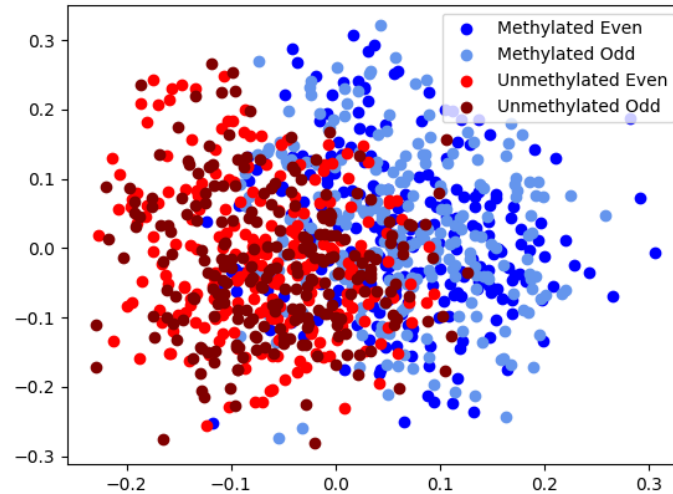


Figure 5: A PCA plot of the 4 groups with the outlier removed

As you can see this made the expected 2 clusters more distinct however there is still a large amount of overlap. We were not able to improve on this using only 2 principal components, although as we will detail later on we were able to clearly distinguish the 2 clusters using more principal components. For the sake of clarity we also looked into what this plot would look like without any form of normalization and just looking at the raw data.

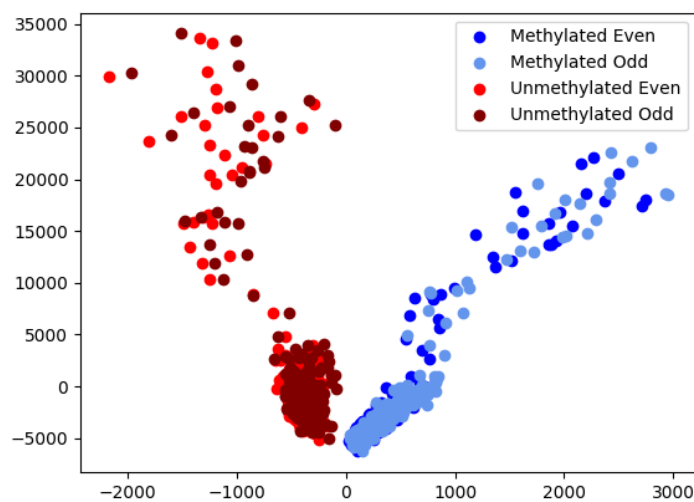


Figure 6: A PCA plot of the 4 groups with the outlier removed without normalization

Much like in our initial exploration we can see a more clear methylated/unmethylated

divide. However, it is becoming more apparent why we need to normalize as there is a large amount of variance in the values of our principal components, making a later distinguishment of controls against cases more difficult

6 Initial model

We will make 2 models where the output of one is the input of the other. The first of these models determines the probability of a sample being either methylated or unmethylated. The second model determines based on those probabilities whether it comes from someone with a tumor or without.

6.1 Methylation classification model

This is the first proper step in the creation of the model that should be able to predict whether a given person may have a tumor based on blood analysis.

This first step is the creation of a *binary classification model* to determine whether we are looking at a methylated or unmethylated region. We decided that a *logistic regression* using *L1 normalization (LASSO)* should be a good classifier. We used the same PCA data as in previous chapters to train and test the model. In addition we also used cross validation to minimize the chance of overfitting. We decided that training on even and testing on odd should be adequate. Using *SKlearn* we created the following code to create, train and test the model.

```
1 pipe = make_pipeline(LogisticRegression(penalty='l1', solver='liblinear',
2                                     max_iter=1000))
3     param_grid = {
4         'logisticregression__C': np.logspace(-3, 3, 1000) # Values for
5                                                         regularization parameter C
6     }
7     grid_search = GridSearchCV(pipe, param_grid, cv=10, n_jobs=-1)
8     grid_search.fit(X_train, y_train)
9
10
11     best_model = grid_search.best_estimator_
12
13     train_accuracy = best_model.score(X_train, y_train)
14     test_accuracy = best_model.score(X_test, y_test)
15     num_selected_variables = np.sum(best_model.named_steps['
                                     logisticregression'].coef_ != 0)
```

We end up with 100% accuracy in our prediction of methylated and unmethylated, while this is regularly a red flag we do not find it too surprising as even just with 2 principal components we could almost separate methylated and unmethylated into 2 distinct clusters. Here is a graph showing the values of the coefficients for the different principal components.

With a good classifier for methylated and unmethylated set up we moved on to the next model.

6.2 Tumor classification

In the previous model we trained the model on only healthy sample, now we try too fit all samples including the the tumor samples. The idea would be that we would have a lower certainty in our methylation classification when using a tumor sample, and that we should be able to find an ideal percentage certainty for a cutoff to do this we used ROC curves

6.3 ROC curve

We used standard SKlearn ROC curve functions to make the ROC curve, we specifically used the following setup

```
1 fpr, tpr, _ = metrics.roc_curve(y, y_scores)
2 AUC = metrics.roc_auc_score(y, y_scores)
```

This allowed us to create ROC curves like this

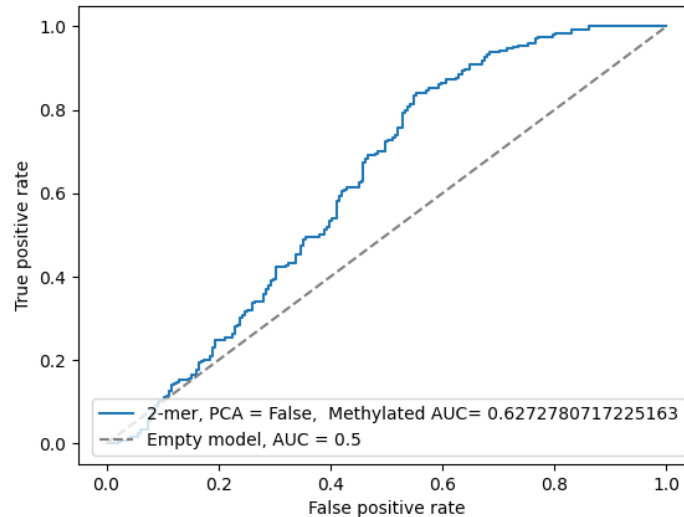


Figure 7: A ROC curve of the methylated 2-mers without PCA

The other models can be found in the figures folder however, they all have around 0.4 - 0.6 AUC. This is not ideal, though it does show that there is a possibility of potentially using a different method to predict cancer. Thus, we decided that maybe we should cut out the middle step of determining methylation and instead go straight to predicting cancer. This new model will be called the "Straight model".

7 The straight model

This model was generally the same as the initial model just with a few key differences were as follows. Instead of training and testing on half of the healthy respectively we were now able to train and test on half of all samples for all regions. Thereby increasing the data for the model. The other being that our new targets for training the model being cancer/noncancer instead of methylation. Using this we were able to create the following ROC curves using the same method as before giving us these results.

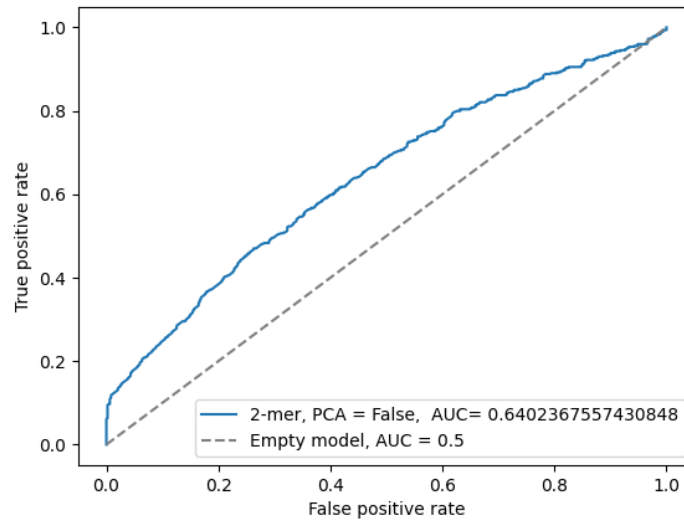


Figure 8: A ROC curve of the methylated 2-mers without PCA

As you can see not much of an improvement though it does seem to be more consistent than our previous method, giving us only AUC above 0.5.

8 Conclusion

Overall the straight model proved to be more effective though not by much, and we were not really able to get a high enough AUC for this to be a viable method for predicting cancer diagnosis. However, we have seen that there is a possibly a different model that is able to pick up the signals from the cancer samples better than the models we created. As such we would say that to some degree this project has been a success, and that more iteration in this space would potentially lead to a more accurate model.