Final Year Project Report

Evaluating Clustering Approaches in Proteomic Data

Conall Doherty

A thesis submitted in part fulfilment of the degree of

BSc. (Hons.) in Computer Science with Data Science

Supervisor: Colm Ryan

UCD School of Computer Science

University College Dublin

April 2019

Project Specification

The bulk of the work in our cells is performed by proteins. Proteins are dynamically regulated such that they are more or less abundant in specific conditions or specific cell types, e.g. some proteins may have higher expression in kidney cells than liver cells. In the last few years it has become possible to quantify the proteome (the set of all expressed proteins) in large numbers of samples, e.g. Mertins et al measured the protein expression levels of 10,000 proteins in 77 breast tumours. Comparing the expression patterns of different proteins in these datasets can be very informative - often pairs of proteins that function together have very similar expression patterns (they are both up or down regulated in the same samples). Unsupervised machine learning approaches (clustering) can be used to identify groups of proteins (clusters) with similar expression profiles (see the attached image for a cluster of proteins with similar expression profiles). These clusters frequently contain significant overlap with known protein complexes or pathways - groups of proteins known from the literature to work together to carry out some common functionality (e.g. replicating DNA, manufacturing proteins). Identifying these pathways and complexes can help us understand what function individual proteins perform and how they cooperate with other proteins to perform this function.

Large scale human proteomics datasets (covering 50+ samples) have only become available in the last couple of years and consequently it has yet to be established how well different clustering approaches perform on proteomics data. The goal of this project is to establish how best to cluster proteomics datasets for the purpose of identifying potential protein complexes or pathways. Numerous clustering approaches have been developed for the related problem of clustering gene expression datasets providing a set of methods to evaluate.

Core:

- Evaluate different similarity / distance metrics (e.g. Pearson's correlation / Spearman's correlation / Euclidean distance) for their ability to identify functionally related protein pairs from protein expression patterns.

- Develop a framework for assessing the quality of a given clustering using the 'gold standard' validation sets

- Evaluate different clustering approaches for their ability to identify meaningful clusters from proteomics data

Advanced:

- Evaluate the impact of missing data (a major problem in proteomics datasets) upon clustering performance
- Develop a documented Python / R package for clustering proteomics data & evaluating the resulting clusters

Abstract

Often times proteins with similar expression profiles will work together to perform a function in what is called a protein complex. Thanks to recent advancements in mass spectrometry there has been an influx of proteomic data that we can now analyse. By clustering the expression profiles in this data, we could potentially identify unknown complexes. This paper shows how hierarchical clustering successfully outperform K-means clustering and how a Euclidean distance measures performs the best when trying to identify complexes in expression data.

Table of Contents

1 Introduction

In this project my aim is to apply multiple different clustering methods to our dataset, in order to identify groups of proteins with similar expression patterns. These clusters could indicate to potential proteins complexes or pathways. Identifying these can help us understand what function individual proteins perform and how they cooperate with other proteins to perform this function. We will experiment with different clustering methods and evaluate each method using a "gold standard" validation set.

For this project I will use two datasets, one containing the protein expression that we will cluster and the other which contains what proteins are in each complex, which we will use for evaluating our clustering method. We will call our validation dataset the gold standard. Each clustering method I use will be evaluated on its ability to find complexes given a set of protein expressions. We will also test how each method performs under different parameters such as the methods similarity metric, number of clusters, linkage methods etc.

This paper outlines my research, approach, design/implementation and my results. While my aim in this project is to cluster protein expression, a lot of my research is based around gene expression clustering. The reason for this is that gene expression is heavily similar to protein expressions as well as there being vast amounts more research in this area already due to it being around for a longer period.

Genes are made up segments of DNA. They are found inside cells and contain blueprints which tell that cell what to do. Central dogma explains the process were the genes create functional products such as proteins, based on the information contained in its DNA.

Gene expression is the process where information contained in a cell's DNA is used to make functional products. This is a tightly regulated procedure that allows for the cell to respond to the ever-changing environment. Differences in the gene expression determine the function of that cell. This carefully orchestrated procedure is regulated at several stages, and mis regulation can often result in diseases such as cancer.

Understanding the genomic sequence relevance and roles in the biological process has been a technical and conceptual challenge for current researchers for many years. Systematic and quantitative analysis of gene expressions is playing a valuable tool in our ability to distinguish between different types of cells. The analysing of gene expressions has played a critical role in the understanding of the structures, functions and the control of biological processes and disease processes

However, measuring expressions at a protein level has potentially more informative information. Proteins act as a major component in the biological process, they are macromolecules that do most of the work in the cell and are as diversified as the functions they handle. They contain multiple dimensions of data that collectively indicate its actual functional state. Information such as the abundance, state of modification, subcellular location, three-dimensional structure and its associations to other biomolecules can be measured now. It has only been recently that methods for extracting protein expressions has come in to place due to advances in proteome analyses by mass spectrometry [5].

By comparing the expression patterns of proteins across tumour profiles we may be able to find pairs of proteins which share a lot of similarities. Proteins that share similarities in their expression patterns can be grouped together. These groups could indicate towards a protein complex or pathway. Protein complexes are groups of proteins that work together to perform some kind of function.

For finding groups of protein expressions that share similar traits we will use an unsupervised machine learning technique called clustering. Clustering has proven to be useful in helping us comprehend data that otherwise would have been difficult to comprehend due to the size, vagueness and noise of the data [9]. The use of clustering has played a pivotal part in the deciphering and understanding of hidden patterns in gene expressions

The 2 main unsupervised clustering methods that we will be working with are k-means and hierarchical clustering. These 2 clustering methods have been used extensively in the area of gene expression analysis and it's interesting to see how they perform on protein expression data as well.

We will experiment with different similarity metrics and linkage methods to see how results differ. The similarity metrics we will be using are Euclidian distance, Pearson's correlation coefficient and Spearmen correlation coefficient.

In this project I will be clustering using Proteomic datasets that contain expression profiles for different tumour profiles. These datasets are publicly available from the cancer genome atlas or similar consortia. They are reasonably sized and well structured. However, there are unique challenges of analysing these datasets due to the large amounts of missing data.

2 Background Research

2.1 Proteins

Protein is an important component of every cell in the body. Proteins carry out most of the bodies function and they make up the bulk of its structure from hair and finger nails to bone and muscles. In fact, nearly everything in our body is made from or by proteins. The instructions for building proteins are found in segments of DNA called genes. The building of proteins from genes can be explained by the central dogma process. Central Dogma is the process in which genetic information flows from DNA to RNA in order to make functional products such as proteins. DNA contains the blueprints needed to make all of our proteins, and RNA acts a messenger that carries this information to the ribosomes. The ribosome serves as a factory within the cells where the information is translated from code into a functional product such as proteins.

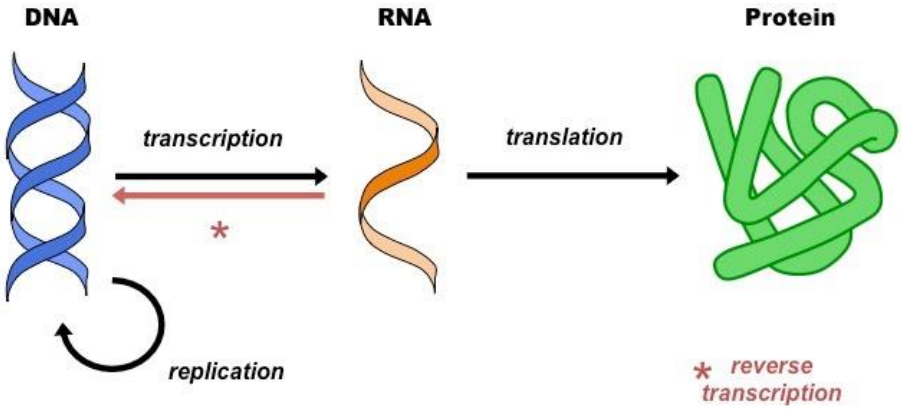


Figure 1. Central Dogma process

Proteins are made of long chains of building blocks called amino acids. There are 20 different types of amino acids that can be combined to make a protein. Put together in different combinations amino acids form proteins with a broad variety of shapes. This diversity of form gives proteins there many functions. Proteins differ from each other according to the type, number and sequence of amino acids that make up the polypeptide backbone. Hence, they have different molecular structures, nutritional attributes and physiochemical properties [6]. Our cells are constantly at work producing the thousands of different proteins that keep our bodies functioning.

2.2 Clustering

Clustering is an unsupervised machine learning technique used to find similarities in data. When we analyse clusters, we are trying to divide data into groups. We may be trying to capture meaning of the data or prepare the data for use for some other process. Clusters can be formed in different ways depending on the nature of the data and the needs of the users. Ideally objects within the same cluster are related to each other and dissimilar to objects in other cluster, the more this is true the better the clusters. This means there are many different ways of clustering the same data.

In bioinformatics, clustering has often been a go-to method in gene expressions analysis. For scientists deciphering the hidden patterns in gene expression data is beneficial in uncovering the natural structure present in expression data, understanding the function of genes and understanding gene regulation.

It has only been in the last few years that it has become possible to quantify the proteome in large number of samples. E.g Mertins et al measured the protein expression levels of 10,000 proteins in 77 breast tumours [7]. Tremendous progress has been made with protein analyses, achieving throughput and comprehensiveness so far only seen in the field of genomics. The resultant mass flood of proteomic data possesses great analytical challenges for researchers.

Protein expression contains vital information needed to understand the biological process that takes place in a particular organism in relation to its environment. Challenges exists for the interpreting and understanding of proteomic data. This is due to the large volume of data and the complexity of the biological network that exists. This data also contains indefiniteness, inaccuracies, and noise. Therefore, clustering techniques are essential in order to successfully analyse this data to unfold natural structures and understand complex patterns in the underlying data.

2.3 Clustering Methods

Clustering is being used in a wide variety of industries and fields of study. It is commonly used in areas of machine learning, data mining, data compression, pattern recognition and bioinformatics. There are many different types of clustering methods each with their own pros and cons. However, according to Pirim et al [1], no clustering algorithm exists with the best performance for all clustering problems. The most commonly used clustering methods used for genome expression data analysis include Hierarchical clustering [3] and K-means clustering [4]. These algorithms are simple to implement and are visually appealing, but their performances can be sensitive to noisy data.

2.3.1 Hierarchical Clustering

Hierarchical clustering is an unsupervised machine learning method that you can use to predict subgroups based on the difference between data points and their nearest neighbour. Cluster are represented by a hierarchy with a predetermined ordering from top to bottom. There are two main types of Hierarchical clustering, divisive and agglomerative.

Agglomerative Clustering:

Agglomerative clustering is a commonly used type of hierarchical clustering used to group observations based on their similarity. This algorithm has a bottom up approach where we start by putting every observation into its own cluster. We then compute the similarity between each and every cluster and merge the 2 most similar clusters. This procedure is iterated until all the observations are merged into one cluster. This clustering method uses a user specified similarity

metric to measure the distance between clusters and a linkage method which specifies the points within the cluster we measure from.

Given:
A set X of objects $\{x_1,...,x_n\}$
A distance function $dist(c_1,c_2)$
for $i = 1$ to n
    $c_i = \{x_i\}$
end for
$C = \{c_1,...,c_n\}$
$l = n+1$
while C.size > 1 **do**
- (c_{min1},c_{min2}) = minimum $dist(c_i,c_j)$ for all c_i,c_j in C
- remove c_{min1} and c_{min2} from C
- add $\{c_{min1},c_{min2}\}$ to C
- $l = l + 1$

end while

Figure 2. Pseudocode for agglomerative clustering [8]

Decisive Clustering:

Decisive clustering involves grouping all observations into a single cluster. At each step in the iteration we partition the least heterogeneous cluster into 2 clusters. This process continues recursively until there is only a single observation per cluster. This type of clustering is conceptionally more complex than that of agglomerative clustering. Divisive clustering works well when trying to identify large cluster, but agglomerative clustering performs better at identifying small clusters. [17]

Dendrograms:

We can represent the hierarchy of our clusters as a dendrogram. Dendrograms are useful for working out the best way of allocating observations to clusters. In a dendrogram the height at which any two observations are linked together indicates the order in which the clusters were formed. Sometimes in dendrograms the height could also reflect the similarity between clusters. Dendrograms are made up of many levels where the higher the level the fewer the clusters. It is up to the user to decide what level to cut the dendrogram with a horizontal line.

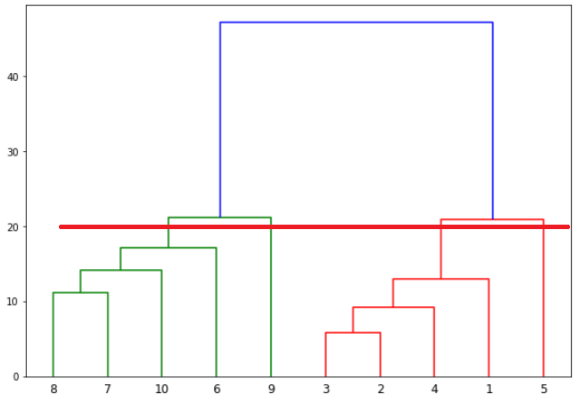


Figure 3. Example of a dendrogram

2.3.2 K-Means

K-Means is probably the most commonly used clustering algorithm. K-means clustering is fundamentally different from hierarchical clustering in that it is a form of partitional clustering where it simply divides a set of objects into non-overlapping clusters so that each object is in exactly one cluster.

Figure 4. Example of results after using k-means

K-mean clustering works by partitioning X objects into K clusters where every object belongs to the cluster with the nearest mean. Its starts by selecting from the data K random objects that will act as the centroids for the initial clusters. Objects are then assigned to the nearest cluster centroid. A new centroid is calculated for each cluster by taking the mean of all the objects in that cluster. We repeat this process until no objects change cluster.

Algorithm 1: K-means algorithm pseudo code

input : X (instance set), K (number of clusters)
output: Clusters $C = \{C_1, C_2, ..., C_k, ..., C_K\}$

while *termination is not satisfied* **do**
  foreach *instance* x_p **do**
    Calculate the distance of x_p to each cluster centroid by Eq. 1.
    Assign x_p to the closest centroid.
  end
  Update centroids by averaging instances within each cluster.
end

Figure 5. Pseudocode for k-means

K-means clustering is a somewhat efficient method. However, it requires a priori knowledge of the number of clusters and it sensitive to initialisation and often can terminate on a local optimum. Unfortunately, there is no method that guarantees to find the optimal number of clusters.

One way that we can find optimum number for K is to use hierarchical clustering to plot a heat map and look for clusters by eye. Another common approach is to plot an elbow chart to

determine the best value for K. An elbow chart is just a line chart where every K is plotted against some scoring value usually some measure of variance. The optimal K is decided by choosing the point on the chart were the line begins to level off.

2.4 Similarity Metrics

It is commonly accepted that proteins with similar expression profiles are functionally related. However, there are many ways one can measure the similarity of expression profiles, and it is not clear what is the most effective one. Moreover, so far, no clear distinction has been made as for the type of the functional link between proteins as suggested by microarray data. Similarly expressed proteins can be part of the same complex as interacting partners; they can participate in the same pathway without interacting directly; they can perform similar functions; or they can simply have similar regulatory sequences.

2.4.1 Euclidean Distance

The Euclidean distance is one of the most commonly used distance metrics. The Euclidean distance between two points, *p* and *q* with *n* dimensions is defined by the square root of the sum of the squares of the differences between all corresponding coordinates in p and q.

$$d(\mathbf{p}, \mathbf{q}) = \sqrt{\sum_{i=1}^{n} (q_i - p_i)^2}$$

Figure 6. Euclidean Distance Formula

The Euclidean distance is usually computed from raw data and not standardized data. With this metric the distance between any two objects is not affected by the addition of new objects. It should be noted that Euclidian distance is sensitive to scaling and difference in average expression levels [14].

2.4.2 Pearson's Correlation Coefficient

The Pearson's correlation coefficient is a widely used statistical score to measure the statistical relationship, or association, between two continuous variables. It is particularly used in the cases of hierarchical clustering applied to microarray or RNA-Seq data.

The Pearson's correlation coefficient can be computed by dividing the covariance of the two points X and Y by the product of their standard deviations.

$$r_{XY} = \frac{\sum_{i=1}^{n}(X_i - \overline{X})(Y_i - \overline{Y})}{\sqrt{\sum_{i=1}^{n}(X_i - \overline{X})^2}\sqrt{\sum_{i=1}^{n}(Y_i - \overline{Y})^2}}$$

Figure 7. Pearson's Correlation Coefficient

Pearson's correlation coefficient returns a value between -1 and 1 where 1 indicates a strong positive correlation, -1 indicates a strong negative correlation and a 0 result indicates no correlation.

2.4.3 Spearmen Correlation

Spearman's correlation coefficient can also be particularly useful to assess the similarity of two given expression profile. While this rank-based measure is more robust to outliers than Pearson's correlation coefficient, it is also less sensitive. While Pearson's correlation coefficient is constructed based on raw values, Spearman's correlation coefficient will assess the concordance between the ranks of these values.

In this formula d is the difference between ranks of corresponding values and n is the number of observations.

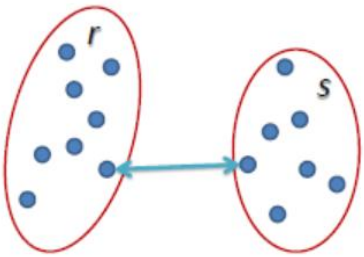
$$\rho = 1 - \frac{6\sum d_i^2}{n(n^2 - 1)}$$

Figure 8. Spearman's Correlation Coefficient

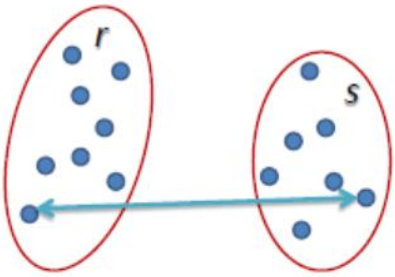
Just like with Pearson's correlation coefficient, Spearman's correlation coefficient also returns a value between -1 and 1, where 1 indicates a strong positive relation, -1 indicates a strong negative relationship and 0 indicates no relationship at all

2.5 Linkage Methods

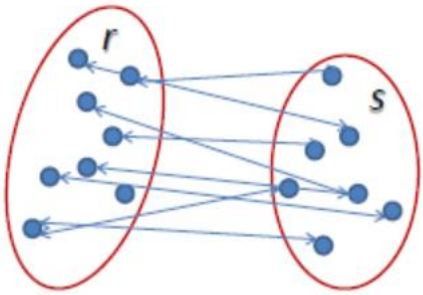
After selecting our distance metric for our hierarchical clustering, we will have to choose from where in the cluster should the distance be measured from. There are 3 main linkage methods:



Single Linkage:
In single linkage clustering we measure the shortest distance between two points in each cluster



Complete Linkage:
In complete linkage clustering we measure the distance between the two furthest points between two clusters.



Average Linkage:
In average linkage the we measure the distance between each point in one cluster to every point in the other cluster and then take the overall average of all the distances.

As with distance metrics, the decision on what linkage method to use is dependent on the theoretical considerations from the domain of application. As for example, it has been observed that for hierarchical clustering of gene expressions, single-linkage and average-linkage tend to perform more poorly [15].

2.6 Evaluating the Cluster

A number of clustering methods have been proposed for expression data. However, the resultant clusters gathered by these different clustering methods could have vastly different arrangements. Because clustering is an unsupervised machine learning method, we have no labels for which the model can provide significance of the resulting clusters. Common evaluation of these type of clusters in gene expression is often accessed by visual inspection, but this requires a priori biological knowledge. Whether the clusters actually correspond to the real structure of the data is somewhat less frequently considered. Furthermore, most clustering algorithms return clusters even in the absence of actual structure, generating results without significance.

An objective method for evaluating clustering's should be independent of the evaluator and reliable concerning its judgement about the quality of the clustering's. There are two main categories of evaluation approaches, external and internal**.**

Internal validation evaluates clusters based on information in the data, such as the characteristics, compactness or separation of the clusters. Clusters that compact and separated well from each other will generally get a good evaluation score. Examples of validation measures are:

- **Connectivity**
  This measure indicates the degree of which items that are found in the same cluster are also determined by their k-nearest neighbours.
- **Silhouette Width**
  This index defines the measure of how similar an item is to its own cluster compared to other clusters.
- **Dunn Index**
  This index measures the ratio of the shortest distance between items not inside the same cluster to the biggest intra-cluster distance.

In external evaluation we use a gold standard/validation dataset to measure the accuracy of our clusters. The gold standard dataset consists of pre-classified items, and these sets are often created by humans. This type of evaluation compares how close the clusters are to pre-determined benchmark classes. Some of the measures used for testing the quality of the clusters using external criteria are:

- **Rand Index**
  The Rand index computes how similar the clusters are to the benchmark classifications. One can also view it as a measure of the percentage of correct decisions made by the algorithm.

$$RI = \frac{TP + TN}{TP + FP + FN + TN}$$

  TP is the number of true positives, TN is the number of true negatives, FP is the false positive and FN is the false negative. In this equation the false positives and false negatives are equally weight which could be undesirable for our clustering application. In that case we would try the F-measure.
- **F-Measure**
  The F-Measure or F1 Score is the measure of tests accuracy and is defined as the weight harmonic mean of the precision and the recall. An F-measure may be a better measure to use if we need to seek a balance between Precision and Recall

$$\text{Precision} = \frac{TP}{TP+FP}$$

$$\text{Recall} = \frac{TP}{TP+FN}$$

$$F_1 = 2 \cdot \frac{precision \cdot recall}{precision + recall}$$

- **Jaccard Index**
  The Jaccard index is a similarity measure used for comparing members between sets in order to see which members are shared between them and which members are not. The Jaccard index is calculated by taking the sum of the intersections between the sets and dividing it by the union of the two sets, where set A is single cluster and set B is a single complex

$$J(A,B) = \frac{|A \cap B|}{|A \cup B|}$$

2.7 Tools

Two of the most popular programming languages used today in data science are R and Python. Both these programming languages are open source and have a vast selection of machine learning and statistical tools. R mainly focuses on statistical analysis while Python offers a broader range of uses cases.

R has been around for over 2 decades. It has become a popular language to use amongst academics and statisticians. It contains the largest ecosystem for data analysis with over 12000 packages available. R is a procedural language that works by breaking down tasks into series of steps. This is useful because it makes it relatively easy to follow along with complex operations, though this often at cost of code readability.

Python has been around since 1989. It was created with philosophy that emphasised efficiency and code readability. Unlike R, Python wasn't created with the intended focus for data science, instead its focus was on being a general-purpose programming language. Data science only ever played a small portion within Pythons' ecosystem. Despite all this Python has become a major tool for data scientists across all industries due to its replicability and accessibility over R.

Python has a number of powerful libraries that are capable of performing many of the same operations as what can be seen in R.

2.7.1 Pandas

Pandas is one of the most popular libraries in python. It is one of the most used and preferred tools to use when it comes to data wrangling and manipulation. It is fast, flexible and well structured. Its aim is to be the fundamental high-level building block for doing practical, real world data analysis in Python. Pandas is well suited for many kinds of data such as tabular data, ordered and unordered data, matrix data and any other form of observational data sets [10].

2.7.2 SciKit-Learn

When it comes to machine learning in Python, Scikit-Learn is the tool of choice. It offers a wide selection of supervised and unsupervised learning algorithms and is by far the simplest and tidiest machine library to use.

Scikit-Learn was designed with a software engineering mindset. It contains a large selection of algorithms for different types machine learning. Its powerful API is robust and easy to use which makes it such a perfect tool to use for all stages of any machine learning projects. [11]

2.7.3 Matplotlib

Matplotlib is a popular visualisation library used for plotting 2d arrays. It is easy to use and only takes a few lines of code to make quality charts. Matplotlib offers a wide selection of options such as bar, line and scatter plots that make creating informative and aesthetic visualisation possible.

3 The Data

3.1 The Proteomics Dataset

The Proteomic dataset labelled "CPTAC_BC_SupplementaryTable03.xlsx" will be the dataset that we carry out our clustering on. It was provided to me by my project supervisor.

This dataset contains proteome profiles from 77 breast tumour samples. Roughly ~12,000 Protein expression values where collected from the samples, with missing values present where the protein in a given sample couldn't be quantified. Samples were collected by the work of Phillip Mertins and the Clinical Proteomic Tumour Analysis Consortium (NCI/NIH), who used quantitative mass-spectrometry based proteomic analysis to identify these high-quality protein expressions data.

The important content of CPTAC_BC_SupplementaryTable03.xlsx file:

- RefSeq accession number: RefSeq protein ID (each protein has a unique ID in a RefSeq database)
- Gene name: a name of that protein
- Columns 12-94: log2 iTRAQ ratios for each sample (protein expression data, most important), three last columns are from healthy individual

The protein expression levels in the dataset were already normalised by the creator of the dataset. They attempted to identify the unregulated proteins and phosphosites, and centered the distribution of the log ratios around zero.

For our experiment we will focus on using just the expression values for our clustering algorithms.

3.2 Gold Standard

A Gold Standard dataset has been provided which will be used when evaluating the clustering method. The Gold Standard dataset contains information regarding the protein complexes and their functions. It includes helpful information such as the protein complex function, localization, subunit composition, literature references and more. This dataset includes 4274 mammalian protein complexes which are built from 4473 different genes. All information in this dataset was created by CORUM[16] who collected this information manually from individual experiments published in scientific articles. The data from high-throughput experiments were excluded.

4 Data Preparation

4.1 Set Up

For loading this data into our python jupyter notebook we used the pandas library. The pandas library has a number of useful functions which made reading in the datasets extremely easy. We just need to specify the file we want to read and the relevant delimitator that separate the data, in our case we used the default. For the proteomic dataset we only read data from sheet 3 of that excel sheet. These datasets are loaded into what pandas calls a dataframe. A dataframe is a data structure with a similar format to a spreadsheet, everything is represented by rows and columns.

4.2 Missing Data

Before we can carry out any clustering or evaluation we must first deal with any missing data in the dataset. One major problem in the proteomic dataset is that we are missing the gene names for some of our proteins. Without these we will have no way to compare the clusters to the validation set. For this case we must remove any entries that have a 'Nan' value in the 'geneName' field. Dropping these rows leaves us with ~10,000 rows.

In our proteomic dataset there are proteins whose gene name doesn't exist within our gold standard dataset. This means that these proteins are guaranteed to come out with a 0 score if clustered together or otherwise lower the score of other valid clusters. So, to deal with this I removed all proteins whose gene name doesn't appear in the gold standard dataset. This left us with only ~3000 rows.

There are also missing values wherever the protein in a given sample couldn't be quantified. This is a problem for our analysis as most clustering algorithms can't function with missing data. In our dataset ~600 of the proteins are missing at least one expression value which leaves us with ~2400 proteins for our clustering algorithm. Because of this we will create two datasets:

- Primary dataset: Contains proteomic data for ~2400 proteins, which is used for evaluating different clustering methods and similarity metrics.
- Secondary dataset: Contains proteomic data for ~2700 proteins where proteins with >20% of missing data were removed and any missing values left were filled with that samples mean. This dataset will only be used to highlight the effect that missing data has on our clustering performance.
- Tertiary dataset: This dataset contains ~10000 proteins, where the only proteins removed were the ones missing more than 20% of their expression levels. This dataset can't be evaluated to the gold standard due to missing gene names, but it can be used to create heatmaps to visualise the clusters.

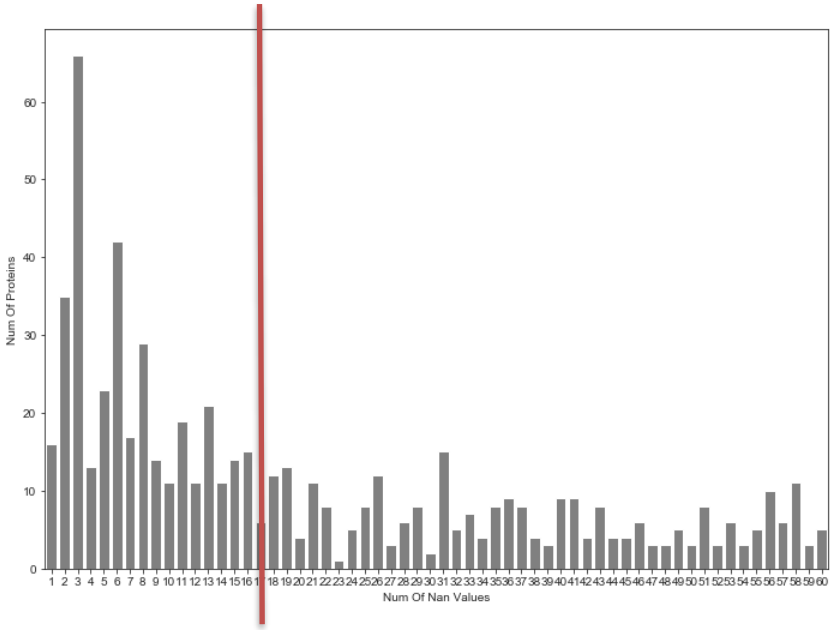


Figure 9. Number of nan values missing with line to mark 20% cut off point

4.3 Label Matching

When we perform our evaluation, we will need to be able to compare the clusters we created with the complexes found in the Gold Standard dataset. This involves having to find the set of gene names in each cluster and the set of gene names in each protein complex. When we perform our clustering, we are only given an array with the cluster labels of each protein index. We need to match these labels to the original data set so that we get the gene name in each cluster. After identifying the proteins in each cluster, we create a set using the values in the 'geneName' field. Likewise we create a set of gene names from our Gold Standard dataset by parsing the 'subunits(Gene name)' field. Once we have our sets, they can be passed to the evaluation function.

5 Implementation of Evaluation Metric

By implementing an evaluation metric, we will be able to assess our clustering algorithms performance and the effect different parameters have on it. For this project we were given a validation/gold standard dataset which we can compare the clusters to. This means that we can avoid using any internal criteria measures that focus on scoring based on high intra-cluster similarity (items in a cluster are similar) and low inter-cluster similarity (items in different clusters are dissimilar) and focus just on the external criteria measures that scores clusters on their similarity to a validation dataset.

5.1 Jaccard Index

The Jaccard index is a similarity measure used for comparing members between sets in order to see which members are shared between them and which members are not. It gives a score in the

range of 0 to 1 were the closer to one the more similar the two sets are and the closer to 0 the less similar. It should be noted that this measure is sensitive to small samples sizes.

For scoring my clustering algorithm I created a function called jaccard_Eval() which takes in two parameters, the clusters we want to evaluate and the validation set we compare them to. The function then returns a score between 0 and 1 which tells us how accurate our clustering algorithm is. The function works by iterating through our clusters C and comparing them against every complex in the Validation set V. Using pythons set function we get the intersection and union set of the cluster and complex. We then divide the number of elements in the intersection set by the number of elements in the union set. The complex that has the strongest similarity match to that cluster is recorded and stored in the scores array. Any clusters that get a 0 score were removed from this array. This reason behind this is that our validation set won't have every protein from the proteomics dataset. Therefore, any clusters made up of only proteins that are not in the validation set will always score 0 which would negatively affect the overall clustering evaluation score. The function finishes by returning the mean score of all the clusters we evaluated. We use this score as a measure for the clustering methods ability to find protein complexes from expression data.

Function: jaccard_eval()

Input: X (Cluster set), V (Validation set)

Output: Average Jaccard index score across all clusters

Foreach instance x_p in X **do:**

Foreach instance v_q in V **do:**

Calculate max score for x_p using length($x_p \cap v_q$) / length($x_p \cup v_q$)

End

End

Remove values if score = 0.0

Return mean of scores

Figure 10. Pseudocode of Jaccard evaluation function

6 Implementation Clustering Methods

Part of my core objectives is to evaluate different clustering approaches for their ability to identify meaningful clusters from proteomics data. Due to the similarities between gene expressions and protein expression I thought It would be best to try clustering methods that are commonly successful in gene expression clustering experiments. From my research I decided that I should use K-means and Hierarchical clustering for this experiment. All the tools I need for this experiment are all found in libraries in python.

6.1 K-Means

My decision to use K-means was based on multiple factors. K-means is a very popular clustering method that is used in a vast number of fields including the bio-informatic field who use it very

often in the case of clustering gene expression data. It's success there has played a major decision for me choosing it. Other reasons that influenced my choice were:

- It isn't complex to understand
- Its fast
- Implementation is simple and widely supported
- Always yields a result

6.1.1 Implementation

I used the machine learning library scikit-learn for my clustering. This library contains the function Kmeans() which I use for clustering the protein expressions. The Kmeans function takes a number of different parameters:

- n_clusters – The number of clusters (K) and centroids to generate
- random_state - Chooses random number generation for centroid initialization

The random_state value dictates the starting point for the centroids. By setting this parameter to none the kmeans function will use a random number generator to select the centroids starting points every time it is ran.

Once we create our kmeans cluster instance we need to pass the data into the fit() function in order to begin the clustering on that data.

We run the cluster_labels() function we made earlier on our clusters so that we can get the set of gene names in each cluster. Once we have the set, we can pass it on to the evaluation function which will give us a score.

6.2 Hierarchical Clustering

I chose hierarchical clustering because of its popularity in gene expression clustering and it is easy to understand. Also, hierarchical clustering allows me to try out a number of different parameters for similarity metrics and linkage methods. The Hierarchical clustering method that I will be implementing is the agglomerative hierarchical clustering due to its availability in python libraries.

For clustering the protein expressions I will be using the AgglomerativeClustering() function found in the sklearn library. This clustering function take three parameters:

- n_clusters – The number of clusters
- affinity – The similarity metric to use for measuring distance between point
- linkage – The linkage method to use

Once we create the clustering method instance we then call the .fit() function on it and pass in the data to be clustered.

The clusters are passed to cluster_labels() function in order to get the gene names in each cluster then the evaluation function is called to give us a score.

To visualize the results, we create a dendrogram. For creating the dendrogram we use the scipy library. This library contains a function dendrogram() which plots the dendrogram for the protein expressions.

A further visualisation used involve creating a heatmap of our clusters. For this we use the clustermap() function in the seaborn library.

The cophenetic correlation coefficient of the hierarchical clustering compares the pairwise distances of all the samples to those implied by the hierarchical clustering. The cophenet() function in the scipy library is used to get this score. The closer the score is to 1, the better the clustering is at preserving the original distances.

6.3 Parameter Tuning

For the clustering algorithms being used we needed to decide on how many clusters K to use. The choice for K can have a huge influence over the success of our clustering algorithm.

The elbow method can be used for finding the optimal value for K. This method works by iterating through a range of values for K and plotting a score for each value. The point where improvements in score begins to level off is considered the best value for k.

My implementation of this involve iterating through a range of values from 1 to 1000 in increments of 10. I would then apply K-means clustering to a sample of the data set that is to be clustered. My decision to use a sample was so that both the process would run faster, and it meant that we wouldn't be picking an overly tailored value for that dataset. At each iteration the Jaccard score was calculated and recorded for that set of clusters. Once all values for k have been measured, they are then plotted using the .plot() function in the matplotlib library.

From the implementation of the elbow method we can see that our performance on the evaluator rises quickly as we increase the value for k initially before slowing down. The optimal k is found at the elbow point of this curve where the score begins to level out. From my inspection of figure 11 I decided that the best value for k is 250.

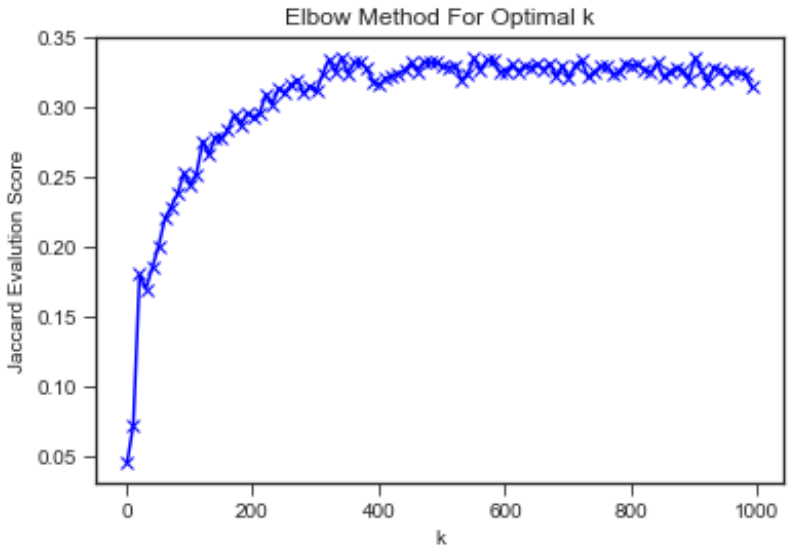


Figure 11. Parameter tuning for optimal k

7 Testing/Evaluation

7.1 Hierarchical clustering results

The agglomerative hierarchical clustering method was ran with multiple linkage methods and similarity metrics in order to determine the best clustering parameters for proteomic data. Euclidean distance outperformed all other similarity metric tested. It had a top score of 0.353 from the evaluation function. The Spearman measure had the most unfavourable result with the lowest score amongst all other metrics. Single linkage worked the best out of all other linkage method for every similarity measure.

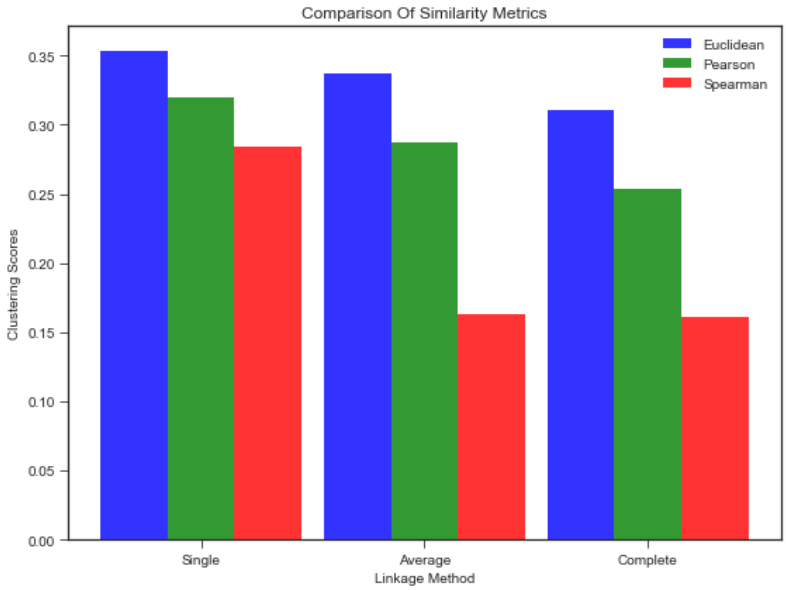


Figure 12. Results for individual hierarchical clustering parameters

The best hierarchical clustering method also has a cophenetic score of 0.89 which means it does a good job at preserving the pairwise distance to the original data.

7.2 K-means

Kmeans performed poorer than Hierarchical having only scored 0.31 in the evaluation function.

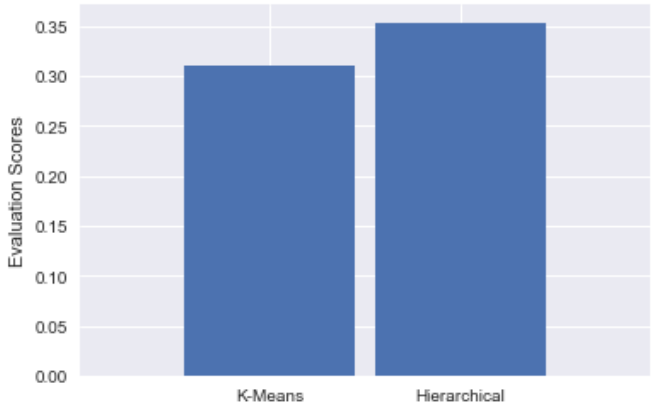


Figure 13. Cluster Method Comparison

7.3 Impact of Missing Data

Part of the objectives for this project was to examine the effect missing data has on clustering results. A dataset with the missing data included was created and put through the same clustering and evaluation approach as the primary dataset. Upon evaluation it was discovered that the missing data had very little impact on the performance of the clustering method having scored nearly identical results across both clustering methods.

Figure 14. Comparison of results between primary and secondary datasets

7.4 Visualisations of the Clusters

A heatmap is used to visualise the clusters that were formed by hierarchical clustering. On the y-axis are the protein indexes and the x-axis are the tumour samples. Areas with a large concentration of colour indicate to a high level of similarity amongst protein expression for those specific tumours. There are a number of possible protein complexes that can be seen from the proteomic dataset by using a heatmap. The most obvious one can be seen in the top right corner of figure 15

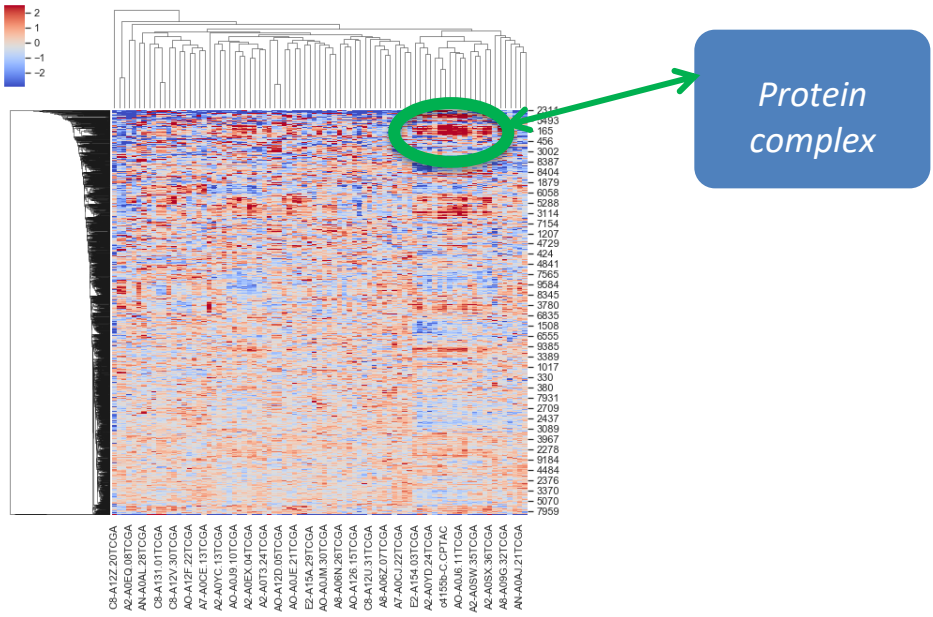


Figure 15. Hierarchical-cluster heatmap

Heatmaps were also done for versions of the dataset that contained missing values. Despite having a larger selection of data these heatmaps don't appear to have any obvious improvements in the quantity or quality of identified complexes.

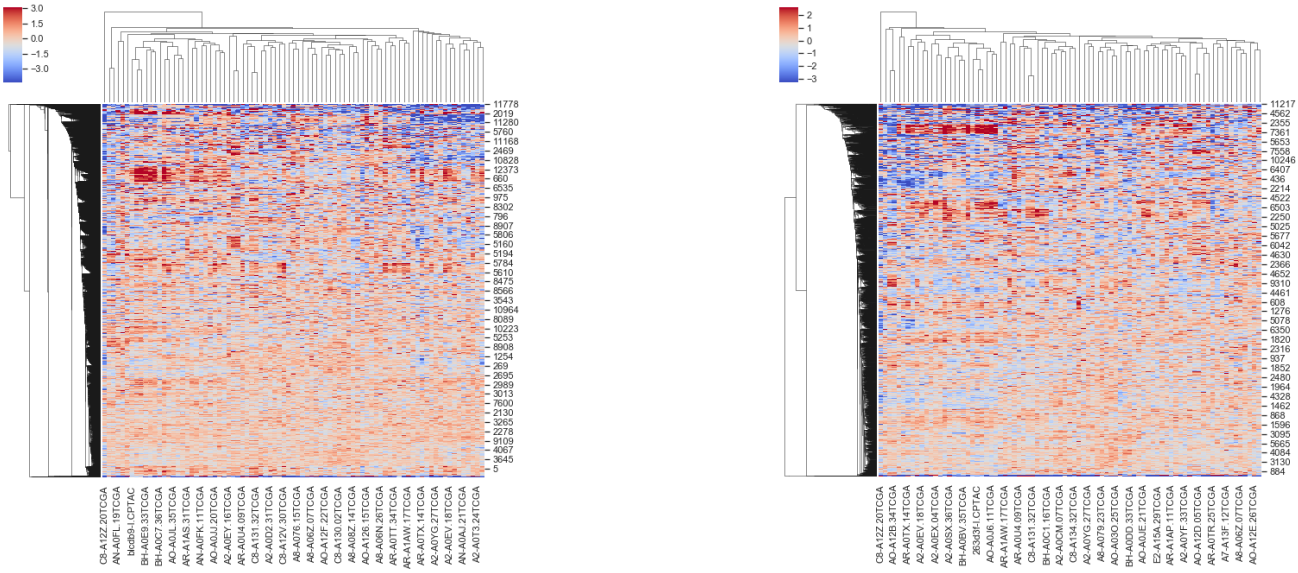


Figure 16. Left: Tertiary dataset
Right: Secondary dataset

8 Conclusion

Proteins are dynamically regulated such that they are more or less abundant in specific conditions or specific cell types. Only recently has it become possible to quantify the proteome (the set of all expressed proteins) in large numbers of samples. Comparing the expression patterns of different proteins in these datasets can be very informative - often pairs of proteins that function together have very similar expression patterns. For this project I was tasked to use unsupervised machine learning approaches such as clustering to identify groups of proteins with similar expression profiles and build an evaluation function to measure their similarity against a gold standard dataset.

In completion of this research project I have successfully completed all the core objectives and one of the two advanced objectives. The one uncompleted advanced objective involved developing a documented python package that can be used for clustering and evaluating proteomic data. Although the functions that perform the clustering and evaluation had been developed during this project certain features and implementations were still needed in order to make a fully functional package that others could use.

I was faced with many different challenges over the course of this research project. Overcoming these challenges required extensive research, critical thinking and collaboration with my supervisor. These challenges consisted of:

- Familiarizing myself with the topics of proteins and genes and their functions, due to much of the reading material during my research phase expected prior knowledge with biology which I did not have.
- Handling the missing data so that it would not interfere with the clustering methods or hamper the evaluation results.
- Finding a metric for evaluating the clusters that provides an accurate representation of how similar the cluster are to complexes found in the gold standard.
- Implementing the different similarity metrics, as two of the three similarity measure didn't have native support within the clustering library used.
- Finding a suitable number of clusters to use for the clustering algorithms.

Ultimately, I have achieved good results for the tasks I was set out to do. The key take home points from the evaluation are:

- Hierarchical clustering performs better than k-means when finding complexes in protein expressions.
- When using hierarchical clustering it is better to use a Euclidian distance measure than Pearson or Spearmen
- Single linkage outperforms complete and average linkage methods.
- Missing data has little impact on clustering performance

9 Future Work

- Create a more advanced evaluation metric. The metric used for this project was simple and sensitive to small sample sizes. A more advanced metric might weight scores giving priority to larger samples.
- Implement more clustering metrics. K-means and hierarchical clustering are only a few of the many clustering algorithm available. Other clustering methods may potentially be more successful in identifying complexes from protein expressions.
- Build an interactive dashboard that allows people to input expression data and see the known and potential complexes. A choice of options on cluster method and metrics to use as well as visualisations of result could be implemented.
- Create a database of identified potential complexes along with a score of how likely it is to be a complex so that researchers can identify it further.
- Find a better way to handle missing data rather than filling it with the mean or dropping it. A better method may involve filling nan values with estimates worked from similar proteins.

10 References

[1] Pirim H, Ekşioğlu B, Perkins AD, Yüceer Ç. Clustering of high throughput gene expression data. Comput Oper Res. 2012;39(12):3046–61.

[2] Daxin Jiang, Chun Tang, Aidong Zhang. Cluster Analysis for Gene Expression Data: A Survey. State University of New York at Buffalo

[3] Eisen MB, Spellman PT, Brown PO, Botstein D. Cluster analysis and display of genome-wide expression patterns. Proc Natl Acad Sci. 1998;95(25):14863–8.

[4] MacQueen J. Some methods for classification and analysis of multivariate observations; Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability; Oakland, CA, USA: University of California Press; 1967. pp. 281–97.

[5] Timothy J. Griffin, Ruedi Aebersold. Advances in Proteome Analysis by Mass Spectrometry. The Journal of Biological Chemistry. 276, 45497-45500. (2001)

[6] https://ghr.nlm.nih.gov/primer/howgeneswork/protein

[7] Philipp Mertins, Mani DR. Proteogenomics connects somatic mutations to signalling in breast cancer. Nature volume534, pages55–62 (02 June 2016)

[8] https://www.saedsayad.com/clustering_hierarchical.htm

[9] Oyelade J, Isewon I, Oladipupo F, Aromolaran O, Uwoghiren E, Ameh F, Achas M, Adebiyi E. Clustering algorithms: Their application to gene expression data. Bioinforma Biol Insights. 2016;10:38316.

[10] https://pypi.org/project/pandas/

[11] https://pypi.org/project/scikit-learn/

[12] https://matplotlib.org/

[13] Ziming Li, Maarten de Rijke. The Impact of Linkage Methods in Hierarchical Clustering for Active Learning to Rank. In Proceedings of SIGIR '17. (2017)

[14] Patrik D'haeseleer. How does gene expression clustering work? NATURE BIOTECHNOLOGY, 23, 12. (DECEMBER 2005)

[15] Gibbons FD, Roth FP. Judging the quality of gene expression-based clustering methods using gene annotation. Genome Res. 2002;12:1574–1581

[16] Giurgiu M, Reinhard J, Brauner B, Dunger-Kaltenbach I, Fobo G, Frishman G, Montrone C, Ruepp A. CORUM: the comprehensive resource of mammalian protein complexes-2019.

[17] https://nlp.stanford.edu/IR-book/html/htmledition/divisive-clustering-1.html