**Introduction**

Chromosome 21 has an extra copy, which leads to the genetic condition known as Down syndrome (DS). Learning and memory issues are regularly reported, and cognitive deficiencies are a prevalent aspect of DS. The data collection focused at how context fear conditioning affects the levels of protein expression in the cerebral cortex of control and DS mice, how associative learning is measured, and how the medication memantine affects trisomic mice's capacity to learn again.

The goal of this study is to use computational methods and algorithms to evaluate protein expression data in order to identify proteins that are differentially expressed.

The objectives of the study are:

Cluster Analysis: To apply clustering algorithms, such as k-means, k-medoids, DBSCAN, and OPTICS. The aim is to identify inherent patterns and structures within the data. This analysis can help in grouping similar measurements together, providing insights into the relationships between different proteins.

Class Prediction: To build classification models using decision trees and random forests. The aim is to predict the class of a mouse based on its protein expression profile. This analysis can provide insights into the relationships between protein expression and mouse characteristics, enabling the identification of key features that contribute to class differentiation.

To find Biomarkers related to Down Syndrome: To utilize feature importance techniques provided by the decision tree and random forest models. These techniques measure the relative importance of each feature (protein) in the classification process. The importance values are typically based on metrics such as Gini impurity or information gain, which assess the extent to which a feature contributes to the purity or predictive power of the model. Higher importance scores indicate that a protein has a stronger influence on determining the class of a mouse.

Understanding the importance of specific proteins in distinguishing between control mice and trisomic mice, as well as the effects of different treatments and behaviors, can provide valuable insights into the molecular mechanisms underlying Down syndrome and the impact of interventions such as memantine.

**Data Description:**

The Mice Protein Expression Data Set is a collection of data related to the expression levels of 77 proteins measured in the cerebral cortex of mice. There are two groups of mice: control mice and trisomic mice (mice with Down syndrome). It contains a total of 1,080 instances. There are 38 control mice and 34 trisomic mice, resulting in a total of 72 mice included in the study. For each mouse, 15 measurements were taken for each of the 77 proteins, resulting in a total of 1,080 measurements per protein.

There are eight classes in which the mice are divided based on genotype, behavior, and treatment. The genotype can be either control (c) or trisomy (t). The behavior of the mice can be either context-shock (CS) or shock-context (SC), indicating whether they were stimulated to learn before the task or not. Additionally, the mice were either injected with saline (s) or memantine (m), a drug used to assess its effect on the learning ability of trisomic mice.

The classes are defined as follows:

c-CS-s: Control mice, stimulated to learn, injected with saline (9 mice)

c-CS-m: Control mice, stimulated to learn, injected with memantine (10 mice)

c-SC-s: Control mice, not stimulated to learn, injected with saline (9 mice)

c-SC-m: Control mice, not stimulated to learn, injected with memantine (10 mice)

t-CS-s: Trisomy mice, stimulated to learn, injected with saline (7 mice)

t-CS-m: Trisomy mice, stimulated to learn, injected with memantine (9 mice)

t-SC-s: Trisomy mice, not stimulated to learn, injected with saline (9 mice)

t-SC-m: Trisomy mice, not stimulated to learn, injected with memantine (9 mice)

**Methodology:**

Following are the steps of analysis:

Data Exploration

Clean the data

Data Standardization and PCA

Implementing K-Means, K-Medoids, DBSCAN and OPTICS Clustering algorithms

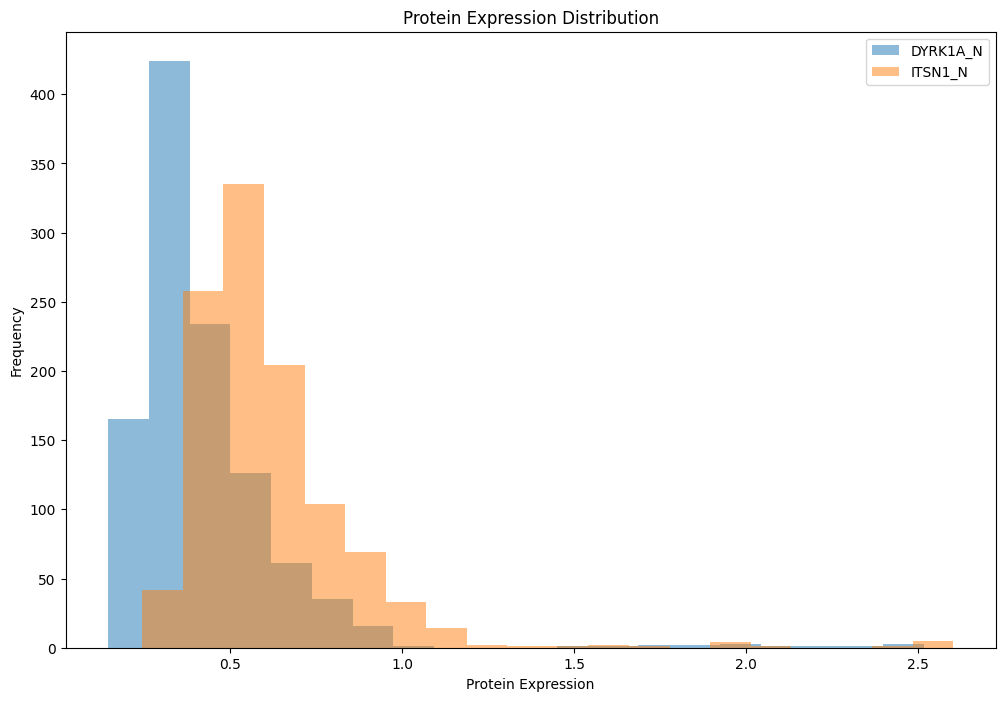
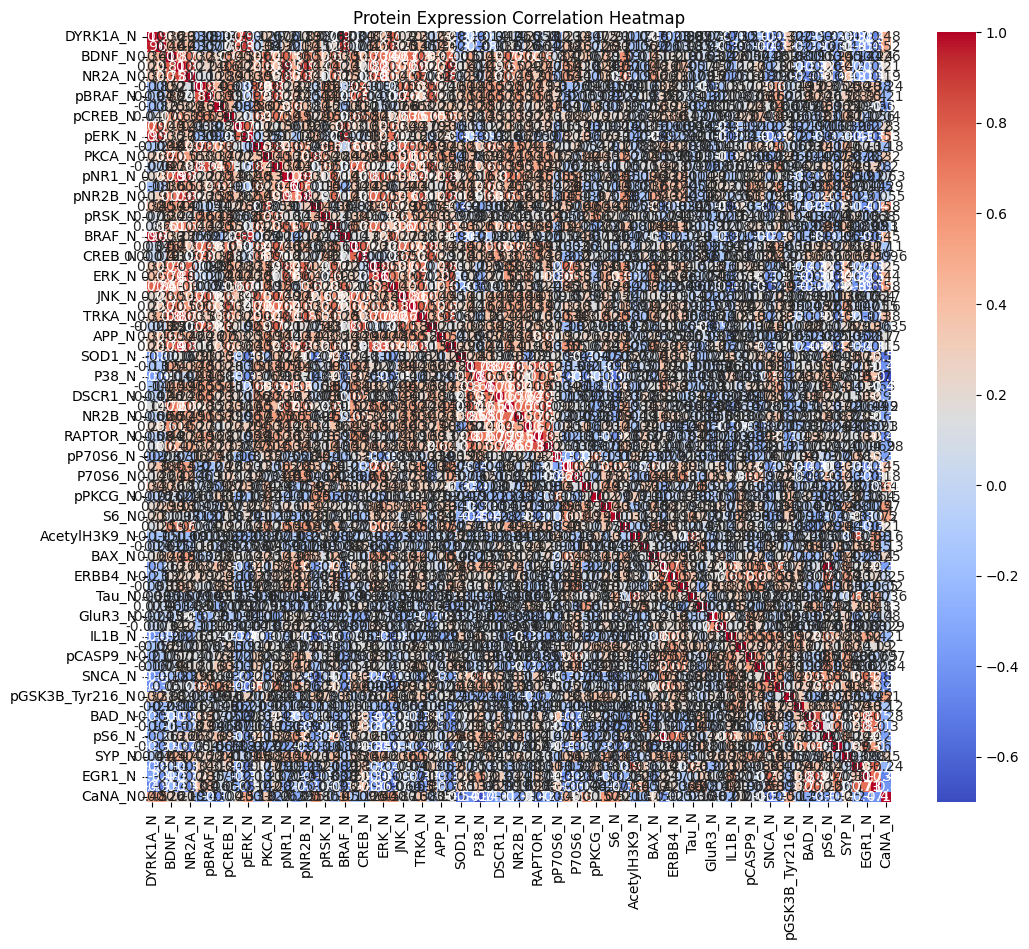
Implementing Decision Tree and Random Forest

Finding important biomarkers of the Down Syndrome to classify class of mice

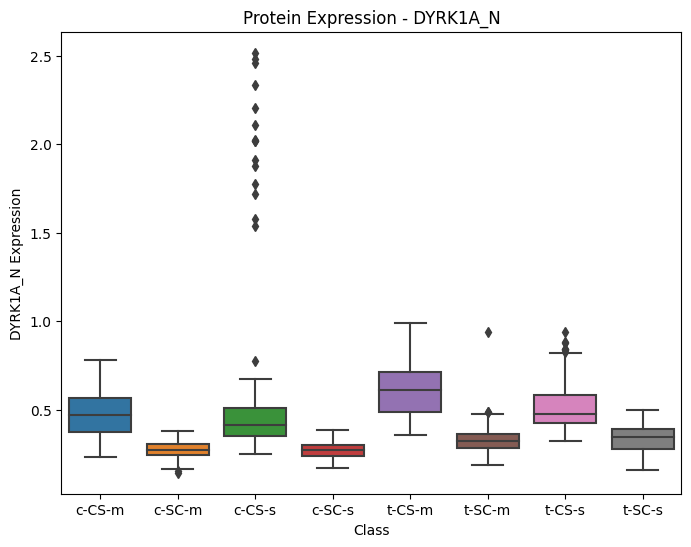
**Descriptive Analysis:**

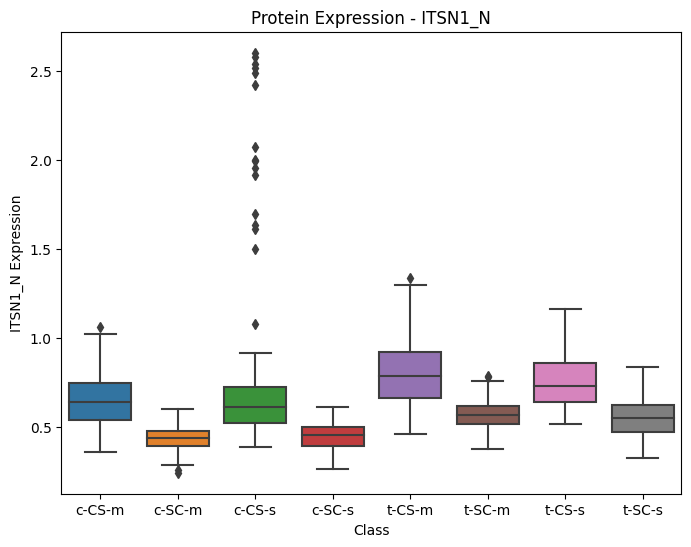
The 77 proteins are as follows:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **77 Proteins:** | |  |  |  |
| DYRK1A\_N | P38\_N | BAD\_N | ELK\_N | RRP1\_N |
| ITSN1\_N | pMTOR\_N | BCL2\_N | ERK\_N | BAX\_N |
| BDNF\_N | DSCR1\_N | pS6\_N | GSK3B\_N | ARC\_N |
| NR1\_N | AMPKA\_N | pCFOS\_N | JNK\_N | ERBB4\_N |
| NR2A\_N | NR2B\_N | SYP\_N | MEK\_N | nNOS\_N |
| pAKT\_N | pNUMB\_N | H3AcK18\_N | TRKA\_N | SNCA\_N |
| pBRAF\_N | RAPTOR\_N | EGR1\_N | RSK\_N | Ubiquitin\_N |
| pCAMKII\_N | TIAM1\_N | H3MeK4\_N | APP\_N | pGSK3B\_Tyr216\_N |
| pCREB\_N | pP70S6\_N | CaNA\_N | Bcatenin\_N | SHH\_N |
| pELK\_N | NUMB\_N | Tau\_N | SOD1\_N |  |
| pERK\_N | P70S6\_N | GFAP\_N | MTOR\_N |  |
| pJNK\_N | pGSK3B\_N | GluR3\_N | pPKCAB\_N |  |
| PKCA\_N | pPKCG\_N | GluR4\_N | pRSK\_N |  |
| pMEK\_N | CDK5\_N | IL1B\_N | AKT\_N |  |
| pNR1\_N | S6\_N | P3525\_N | BRAF\_N |  |
| pNR2A\_N | ADARB1\_N | pCASP9\_N | CAMKII\_N |  |
| pNR2B\_N | AcetylH3K9\_N | PSD95\_N | CREB\_N |  |

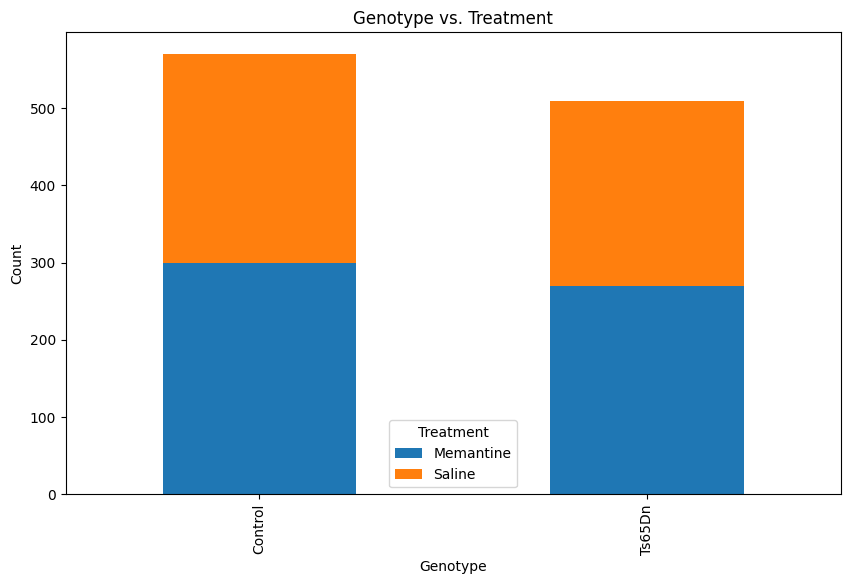
Following are the graphs to visualize the data used in this research: 、

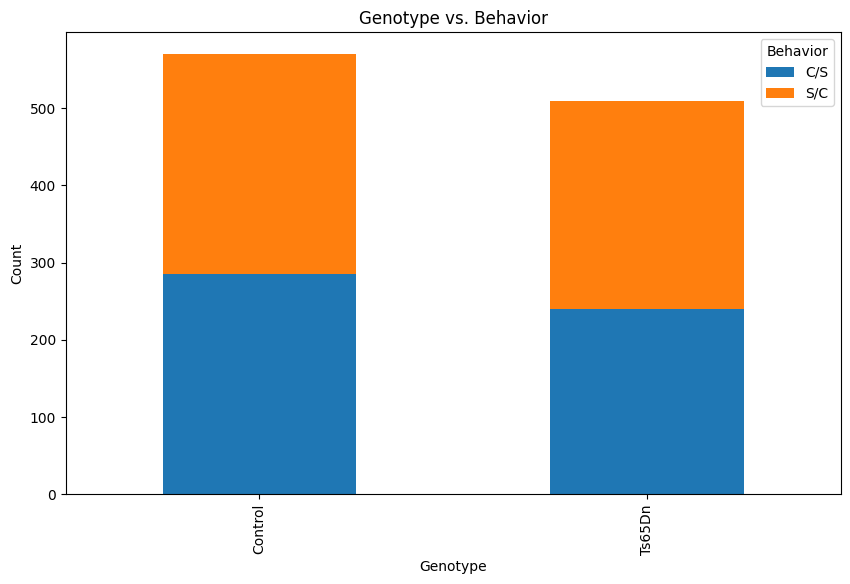
The above graphs show that different proteins have high correlation. Therefore, PCA technique can be used to reduce data complexity.



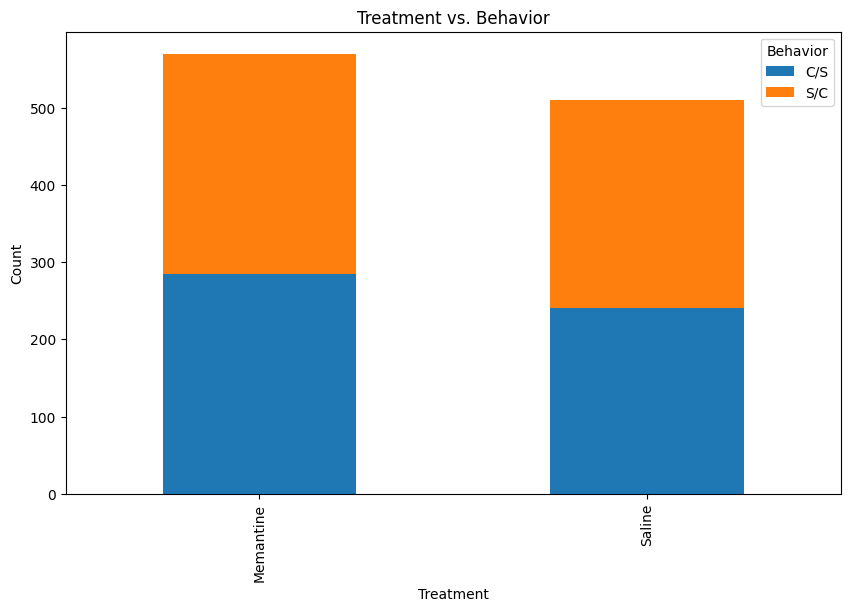


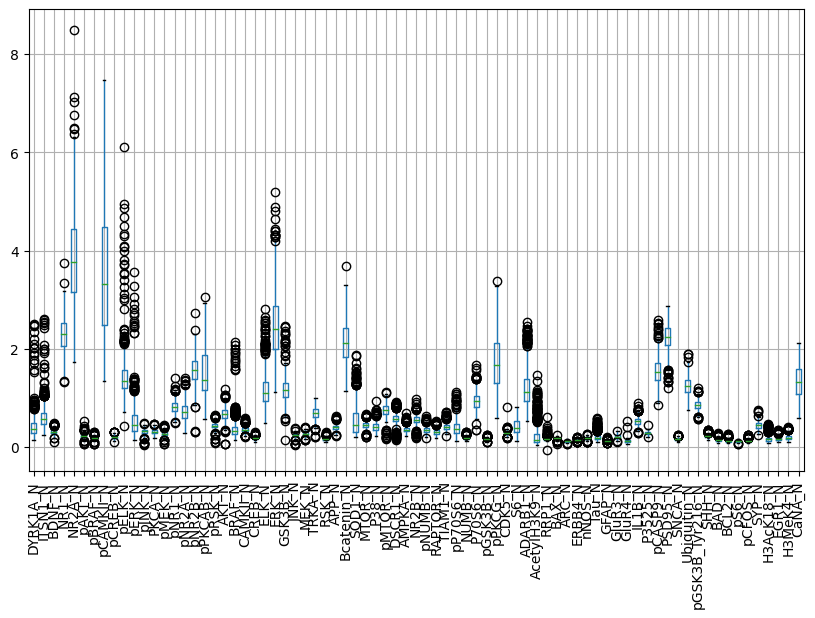
These box plots show that a protein has different levels in each of the 8 classes of mice.





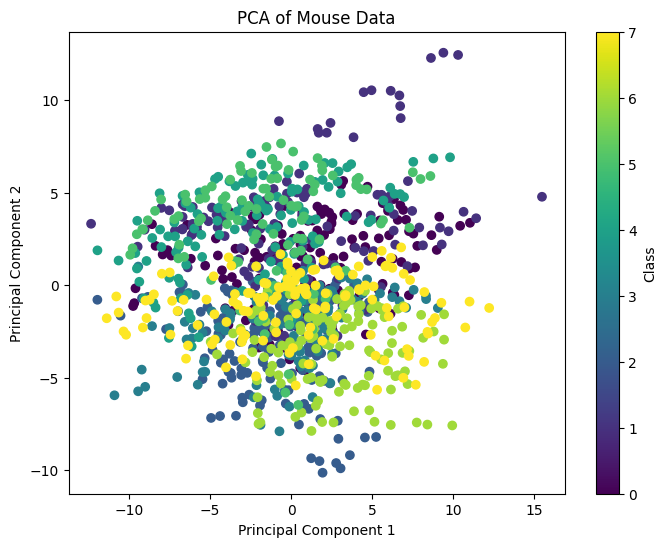
The above graphs show the distribution of mice based on the behaviour and genotype & treatment and genotype.





There are some outliers but they are important and not due to any data recording error. Hence these outliers are not being removed.

Following is the visualization of data after performing PCA on the data:



**Modeling:**

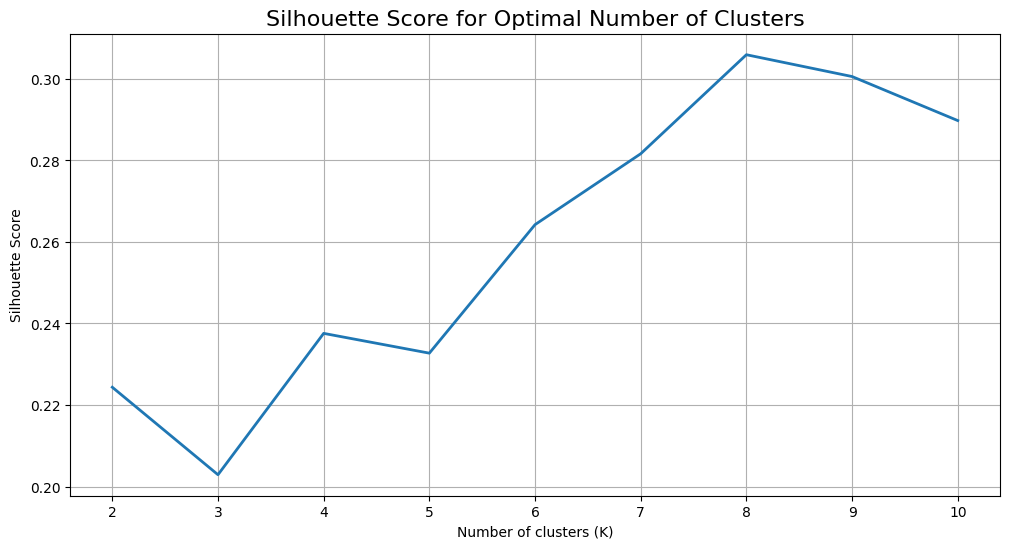
**K-Means:**

The K-means clustering algorithm is an unsupervised machine learning technique widely used for partitioning data into distinct groups based on their similarity. It aims to identify K clusters, where K is a predefined number, by minimizing the within-cluster sum of squared distances. K-means clustering is particularly suited for large datasets with continuous features, making it a suitable choice for analyzing mouse protein data.

The K-means algorithm can be summarized as follows:

Initialization is performed by randomly selecting K initial cluster centroids. Next, assignment takes place where each data point is assigned to the nearest centroid based on a distance metric, typically using Euclidean distance. Then, an update step is performed where the centroids are recalculated by computing the mean of all data points assigned to each cluster. The algorithm proceeds by iterating steps b and c until convergence is reached, typically defined by a threshold or when the assignment no longer changes significantly. Finally, the algorithm terminates when convergence is achieved, and the final centroids represent the clusters.

Following shows the optimal number of clusters:



Hyperparameter tuning is done to find the optimal number of clusters. The optimal is that number which maximizes the Silhouette Score.

Hyperparameter Tuning:

Silhouette\_score for clusters = 2 is: 0.2243608471859176

Silhouette\_score for clusters = 3 is: 0.2028958731510061

Silhouette\_score for clusters = 4 is: 0.23757483780390265

Silhouette\_score for clusters = 5 is: 0.23271972222694462

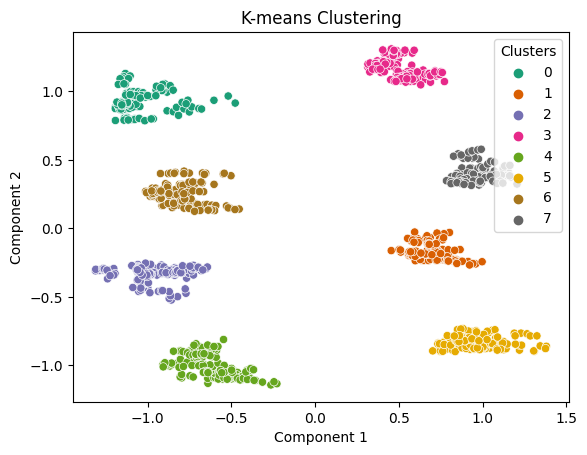
Silhouette\_score for clusters = 6 is: 0.2642494759142338

Silhouette\_score for clusters = 7 is: 0.2816428982124519

Silhouette\_score for clusters = 8 is: 0.30593938625627853

Silhouette\_score for clusters = 9 is: 0.30058711038567343

Silhouette\_score for clusters = 10 is: 0.2897836599421153



We can see that there are 8 clusters of mice. The class of any new mice can be predicted based on its relative position compared to the positions of these clusters.

**K-Medoids:**

The K-medoids clustering algorithm is a popular technique used for partitioning data into distinct clusters based on their similarities. Unlike the K-means algorithm, which uses centroids to represent clusters, K-medoids employs actual data points, known as medoids, as representatives of the clusters.

The K-medoids algorithm is as follows:

Initialization is performed by randomly selecting K data points as the initial medoids. Next, assignment takes place where each data point is assigned to the nearest medoid based on a distance metric, commonly using methods like Manhattan or Euclidean distance. Then, an update step is performed where, for each cluster, the total dissimilarity (distance) between the medoid and all other data points in the cluster is evaluated. The data point with the lowest total dissimilarity is selected as the new medoid for that cluster. The algorithm proceeds by iterating steps b and c until convergence is reached, typically defined by a threshold or when the assignment no longer changes significantly. Finally, the algorithm terminates when convergence is achieved, and the final medoids represent the clusters.

Hyperparameter tuning is done to find the optimal number of clusters. The optimal is that number which maximizes the Silhouette Score.

Hyperparameter Tuning:

Silhouette\_score for clusters = 2 is: 0.5363304963448586

Silhouette\_score for clusters = 3 is: 0.5389704012252952

Silhouette\_score for clusters = 4 is: 0.5710370909841883

Silhouette\_score for clusters = 5 is: 0.4608056868732069

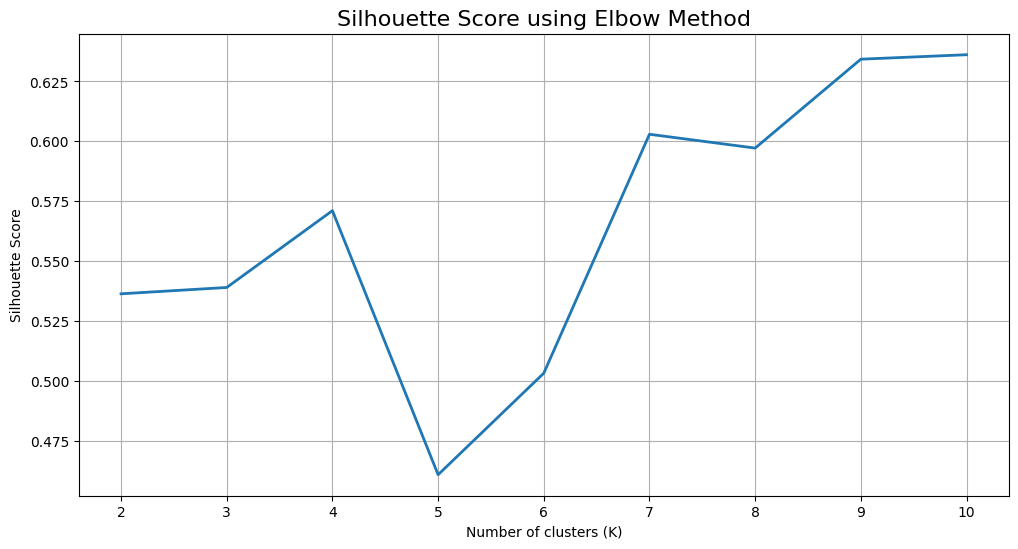
Silhouette\_score for clusters = 6 is: 0.5031903279052898

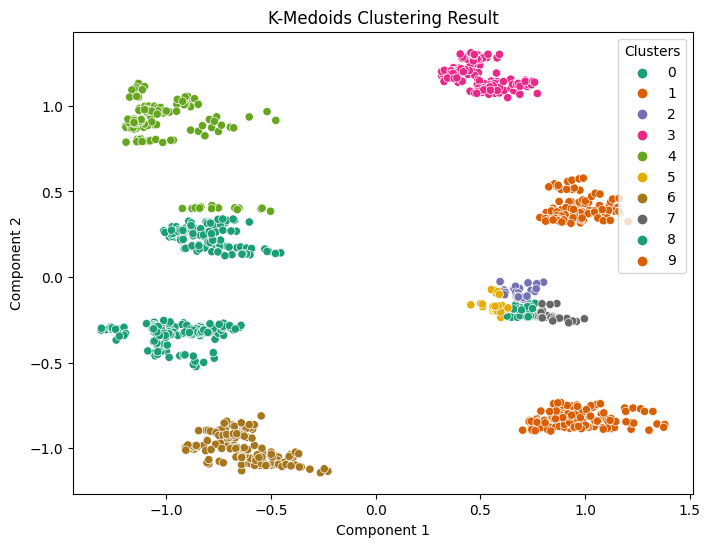
Silhouette\_score for clusters = 7 is: 0.6029256716241355

Silhouette\_score for clusters = 8 is: 0.5971616210801299

Silhouette\_score for clusters = 9 is: 0.6342663771192352

Silhouette\_score for clusters = 10 is: 0.6361326956432852





The K-Medoids model gives a different result. There are 10 optimal clusters of mice. This shows that K-Medoids is not the appropriate model for clustering mice data.

**DBSCAN:**

DBSCAN is a density-based algorithm that identifies clusters based on the density of data points in their local neighborhoods.

The DBSCAN algorithm is as follows:

Parameter selection is performed to determine two parameters: epsilon (ε), which defines the maximum distance between neighboring points, and minPts, the minimum number of points within ε to consider a core point. Next, core points are identified as data points with at least minPts neighbors within ε. Density reachability is used to determine if a point can be reached from another point based on the ε and minPts parameters. Then, cluster formation takes place by connecting core points and their density-reachable neighbors, creating clusters. Points that do not meet the density requirements are labeled as outliers or noise. The algorithm then expands the clusters by including density-reachable points, recursively exploring their neighborhoods. Finally, the algorithm terminates when all data points have been assigned to clusters or marked as noise.

DBSCAN is sensitive to the choice of ε and minPts parameters, requiring careful selection for optimal results. That’s why we use Hyperparameter tuning.

DBSCAN can aid in identifying differentially expressed proteins by grouping together data points with similar expression patterns. It allows to explore clusters of proteins that exhibit coordinated expression changes in response to experimental conditions. DBSCAN helps detect outliers or noise proteins that do not fit into any specific cluster, potentially revealing unique or rare proteins of interest.

Hyperparameter tuning results are shown as follows:

Hyperparameter Tuning:

Best hyperparameters:

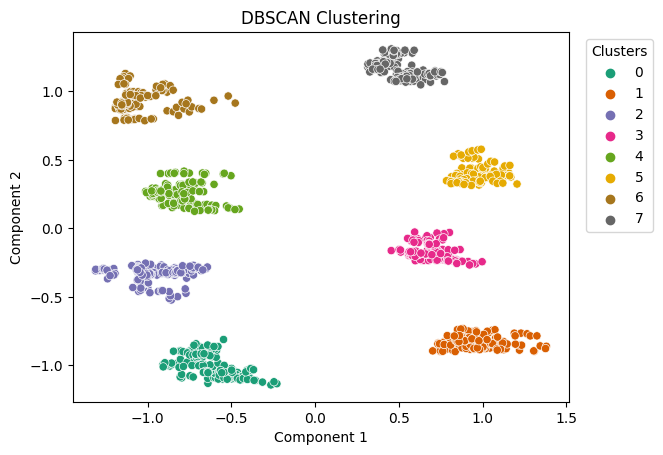
Epsilon: 0.12

Min\_samples: 3

Estimated number of clusters: 8

Estimated number of noise points: 0

Silhouette Coefficient for DBSCAN model: 0.7485061641990243



DBSCAN shows there are 8 clusters. It’s Silhouette Coefficient is higher than that of K-Means, which means DBSCAN is better than K-Means.

**OPTICS:**

The OPTICS algorithm is a density-based method that identifies clusters based on the local density of data points. It provides a flexible approach for capturing clusters of varying densities and extracting hierarchical relationships.

The OPTICS algorithm is as follows:

Firstly, parameter selection is performed to determine the radius of the neighborhood (epsilon) and the minimum number of points (minPts) required to form a cluster. Next, the core distance is computed for each point, which represents the distance to the minPts-th nearest neighbor within epsilon. Subsequently, the reachability distance is calculated for each point, measuring the distance to other points relative to their core distances. Then, a cluster extraction process takes place, where the reachability distances are used to build an ordering of points based on their density-based connectivity. Clusters can be extracted by applying a density threshold or using a clustering algorithm such as DBSCAN on the ordering.

OPTICS offers several advantages for clustering data. It can capture clusters of varying densities, making it suitable for datasets with heterogeneous density distributions. OPTICS also provides a hierarchical representation of clusters, allowing us to explore different levels of granularity within the data.

Similarly with OPTICS, the determination of appropriate values for epsilon and minPts requires careful parameter selection. That’s why we use Hyperparameter tuning.

The Hyperparameter tuning results are:

Best hyperparameters:

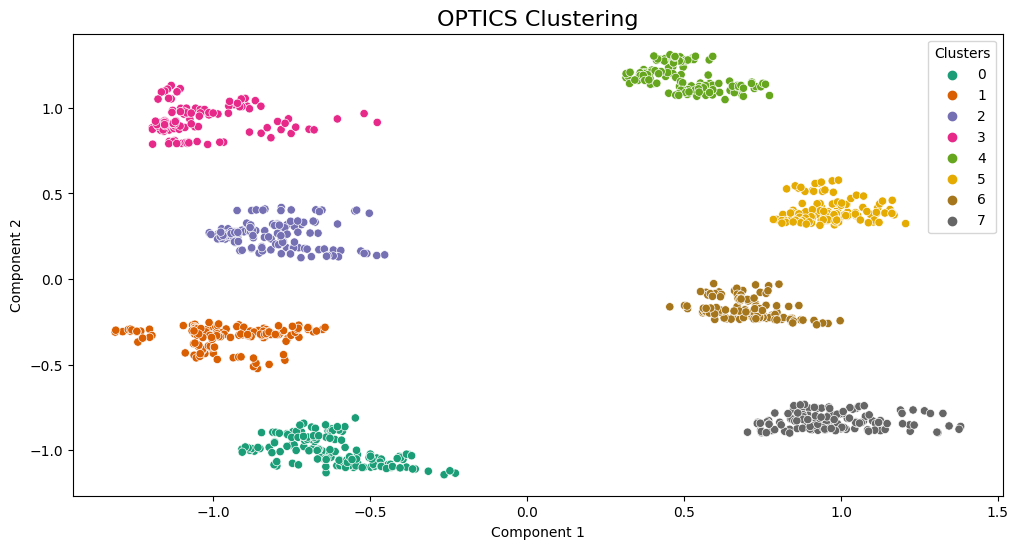
Min\_samples: 30

Xi: 0.01

Min\_cluster\_size: 0.07

Silhouette Coefficient for OPTICS model: 0.7485061641990243

Estimated number of clusters using OPTICS is: 8



We can see that OPTICS performs exactly same as DBSCAN. Both have same value of Silhouette Coefficient.

**Decision Tree Classification:**

The decision tree classification algorithm is a popular and interpretable technique for solving classification problems.

The decision tree classification algorithm is as follows:

Feature selection is performed to identify the most informative features for classification. This is done by using metrics such as information gain, Gini index, or entropy. Next, the algorithm constructs a tree by recursively partitioning the data into homogeneous subsets based on the selected features. At each node of the tree, the best attribute to split is determined using a chosen evaluation metric. The algorithm then assigns a class label to each leaf node based on the majority class in the corresponding subset of data. Optionally, tree pruning can be applied to reduce overfitting by removing unnecessary branches or merging nodes. Finally, the trained decision tree is used for classification, allowing new instances with unknown class labels to be categorized based on their feature values.

Decision trees are interpretable, providing insights into the classification process and feature importance ranking. They handle both categorical and numerical features and can handle high-dimensional datasets. Decision trees can capture non-linear relationships between features, allowing for more accurate predictions.

By training a decision tree classifier on labeled protein data, we can predict the function of unlabeled proteins based on their features. Decision trees can provide insights into the most important features driving the classification, helping identify key factors contributing to protein function or disease association.

There is an accuracy of 88% in classifying mice into the correct classes.

**Random Forest Classification:**

Due to its efficiency in classification and regression problems, Random Forest is a well-known ensemble machine learning technique utilised in many different disciplines. The final prediction is generated by a voting or averaging method, and it is a mixture of decision trees where many decision trees are trained on various subsets of the data.

Each decision tree in the algorithm's collection is trained using a different random subset of the original data. The data is randomly sampled with replacement to produce the random subsets, sometimes referred to as bootstrap samples. The decision trees capture distinct facets of the underlying patterns and minimize overfitting by utilizing different subsets of the data for training.

In addition to sampling the data, Random Forest introduces randomness in the feature selection process. At each node of the decision tree, instead of considering all features, a random subset of features is considered for splitting. This introduces diversity among the trees and helps in reducing the correlation between them. To make predictions, the trained decision trees in the Random Forest collectively vote or average their predictions.

The capacity of Random Forest can handle high-dimensional data and prevent overfitting is one of its main features. Random Forest lessens variation and produces reliable forecasts by mixing numerous decision trees. Additionally, it offers a measure of feature importance that can be used to comprehend the significance of various factors in the prediction job.

Random Forest is renowned for its resilience and adaptability. Numerous data kinds, such as category and numerical variables, may be handled by it. It can accommodate missing data via imputed values based on other variables and is less sensitive to outliers. Additionally, it effectively handles huge datasets thanks to parallel training.

The classification of mice into the appropriate groups has a 99% accuracy rate.

**Conclusion:**

DBSCAN and OPTICS are the best models to cluster the mice protein expression data. Random Forest clustering algorithm is the best model to classify the mice based on protein expression.

The following are the most important biomarkers to identify mice into correct classes, solely based on their all of their 77 protein’s values:

* SOD1\_N (Superoxide dismutase 1)
* pERK\_N (Phosphorylated extracellular signal-regulated kinase)
* CaNA\_N (Calcineurin A)
* pPKCG\_N (Phosphorylated protein kinase C gamma)
* pS6\_N (Phosphorylated S6 ribosomal protein)
* APP\_N (Amyloid precursor protein)
* Ubiquitin\_N
* ITSN1\_N (Intersectin-1)
* DYRK1A\_N (Dual-specificity tyrosine-phosphorylation-regulated kinase 1A)
* pCAMKII\_N (Phosphorylated calcium/calmodulin-dependent protein kinase II)
* pPKCAB\_N (Phosphorylated protein kinase C alpha/beta)
* pNUMB\_N (Phosphorylated NUMB)
* ARC\_N (Activity-regulated cytoskeleton-associated protein)
* Tau\_N (Tau protein)
* pP70S6\_N (Phosphorylated ribosomal protein S6)
* S6\_N (Ribosomal protein S6)
* BRAF\_N (B-Raf proto-oncogene)
* AKT\_N (Protein kinase B)
* P38\_N (p38 mitogen-activated protein kinase)

So, if we have a new and unknown mouse about which we don’t know, whether it has Down Syndrome and whether it is being subjected to any treatment, then we can use Random Forest to accurately determine the genotype, control/down, behaviour and treatment.