# Mouse Classification

Prepared By: Rutvi Macwan

Date: 10/06/2020

## Table of Contents

## Abstract

This project highly focuses on 'data modelling' task, one of the core steps in the field of data science. The data required for this project has been collected from the UCI Machine Learning Repository. This project follows the typical data science process starting with setting the research goal, acquiring the data, data preparation, data exploration and data modelling. As per the requirement of this project, two different classification models have been built while completing the data modelling process. Both models are compared with respect to their accuracy and performance. This project is developed using Anaconda with Python3 in Jupyter notebook IDE.

## Introduction

Goal of this project is to identify a mouse type based on the expression level of various proteins/protein modifications.

The methodology is divided into 3 major parts: Retrieving and preprocessing the data, Data Exploration and Data Modelling.

In the first part, the data retrieved from UCI repository is loaded into Jupyter notebook IDE. This data has been prepared to explore further by checking and correcting for any typo errors, imputing null values, detecting outliers and casting the data type of categorical variables. Sanity check is also performed to check for any impossible values.

In the second part, various visualization graphs with some descriptive statistics have been used to explore the data. Various columns are explored to check their frequency distribution. Multivariate visualization is also performed to check for any relations between two variables.

In the final part, two classification models are built and evaluated using cross-validation technique. Data encoding and scaling has been performed before the data can be fed to these classifiers. Moreover, both models are compared onto their accuracies to determine the model that is more suitable.

## Data set Introduction

The data set used for this project is available on UCI Machine Learning Repository as 'Mice Protein Expression Data Set'. It contains total 82 attributes including 77 types of proteins/protein modifications. 38 control mice and 34 mice with down syndrome were examined for this data collection.

Each observed mouse is assigned a Mouse ID. Protein names are followed by '_N', indicating nuclear fraction of that protein. Data set contains other categorical features including Genotype (control or trisomy), behavior (context-shock or shock-context), treatment (memantine or saline) and a target class of a mouse.

Eight target classes of mice are described based on their genotype, behavior and treatment.

## Retrieving and Preprocessing Data

The dataset was originally available as an excel file that was converted to a csv format explicitly. Later, it was loaded into Jupyter notebook and saved in a data frame named as 'mice_data'.

## Data Preprocessing

### 1. Extra whitespaces, Typo errors and Datatype casting

As we have four categorical features containing string values, I have converted these string literals to lowercase letters and removed extra spaces, if any. In the next step, I have performed a check to look for any typo errors. In the 'Genotype' variable, trisomic class is spelled as 'ts65dn'. I have changed it for readability purpose. Moreover, all the four categorical features have their data type predefined as 'object'. Therefore, I have casted their types to 'categorical'.

### 2. Handling Null values and Outliers

For this dataset, there are no missing values in categorical features. However, there exists some missing values for protein information. To handle this scenario, I have imputed null values with mean of their respective column.

Outliers are also detected in the data. I have used values of mean and standard deviation to detect outliers in the data. However, I have kept the outliers untouched to maintain the variability of the data.

### 3. Sanity checks

I have used summary tables to look for any impossible values. As per the summary table, there are no such values in the data.

| | DYRK1A_N | ITSN1_N | BDNF_N | NR1_N | NR2A_N | pAKT_N | pBRAF_N | pCAMKII_N | pCREB_N |
|---|---|---|---|---|---|---|---|---|---|
| **count** | 1080.000000 | 1080.000000 | 1080.000000 | 1080.000000 | 1080.000000 | 1080.000000 | 1080.000000 | 1080.000000 | 1080.000000 |
| **mean** | 0.425810 | 0.617102 | 0.319088 | 2.297269 | 3.843934 | 0.233168 | 0.181846 | 3.537109 | 0.212574 |
| **std** | 0.249015 | 0.251290 | 0.049314 | 0.346810 | 0.931802 | 0.041577 | 0.027004 | 1.293368 | 0.032542 |
| **min** | 0.145327 | 0.245359 | 0.115181 | 1.330831 | 1.737540 | 0.063236 | 0.064043 | 1.343998 | 0.112812 |
| **25%** | 0.288163 | 0.473669 | 0.287650 | 2.059152 | 3.160287 | 0.205821 | 0.164619 | 2.479861 | 0.190828 |
| **50%** | 0.366540 | 0.566365 | 0.316703 | 2.297269 | 3.763306 | 0.231246 | 0.182270 | 3.329624 | 0.210681 |
| **75%** | 0.487574 | 0.697500 | 0.348039 | 2.528035 | 4.425107 | 0.257225 | 0.197226 | 4.480652 | 0.234558 |
| **max** | 2.516367 | 2.602662 | 0.497160 | 3.757641 | 8.482553 | 0.539050 | 0.317066 | 7.464070 | 0.306247 |

*Figure 1: Summary table*

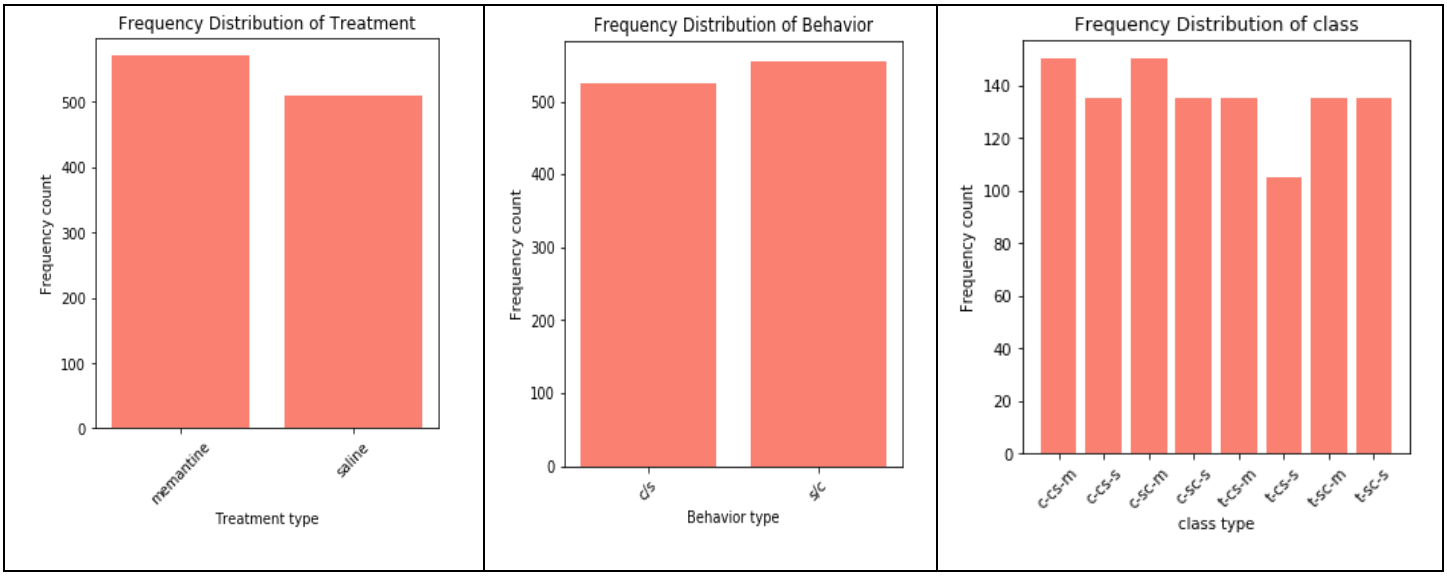
## Data Exploration

I have used univariate and multivariate visualizations to explore the unknown relationships in our data set.

### Univariate Visualizations

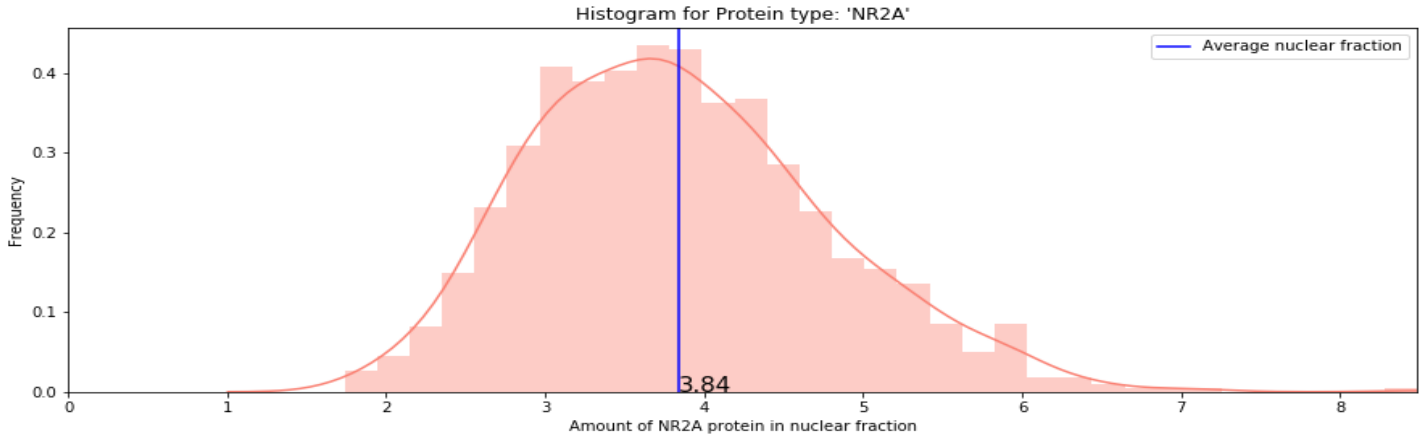
#### 1. Treatment, Behavior and Class

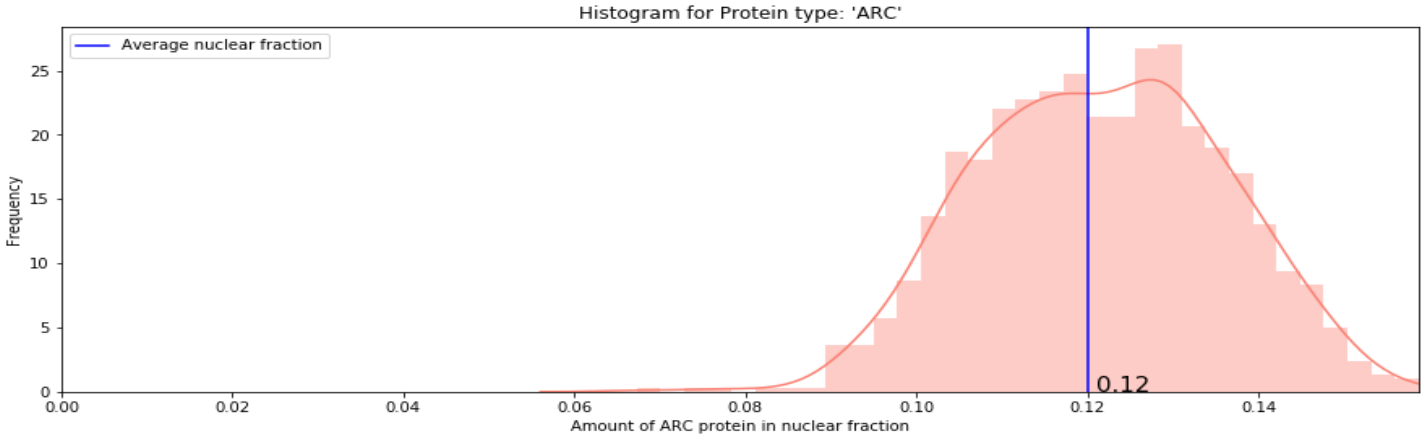
Exploring the categorical data and plotting their frequency distribution in a form of bar chart shows that there are more mice that have been treated with Memantine drug than Saline. The number of mice stimulated to learn is lower than those who are not stimulated to learn. Moreover, the frequency distribution of target class shows that the selected dataset is balanced in nature.

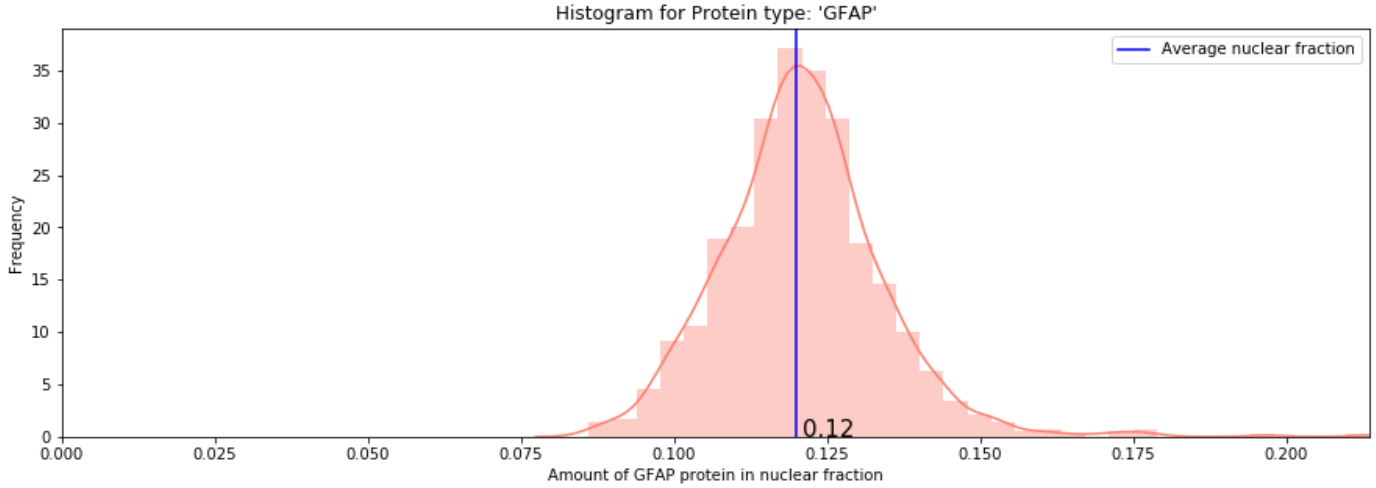


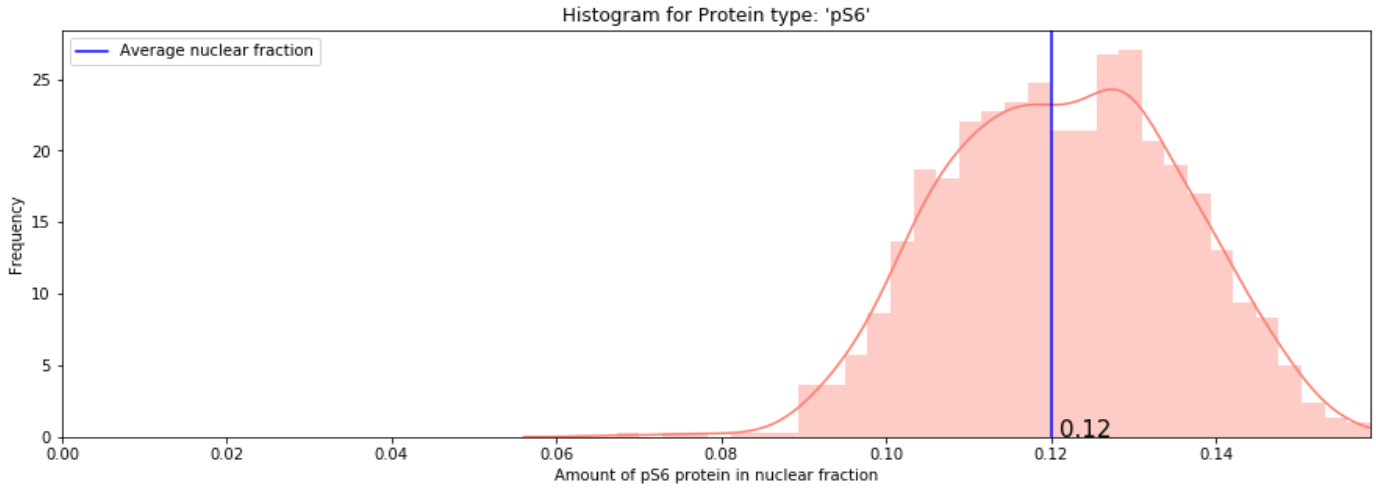

## 2. 77 Proteins / Protein Modifications

The columns containing protein information are all numerical. Therefore, I have plotted them as a histogram to check their distribution properties. From the graphs, it can be observed that examined mice have shown highest average amount of 'NR2A' protein (3.84 nuclear fraction) and lowest average amount of 'ARC', 'GFAP' and 'pS6' proteins (0.12 nuclear fraction).



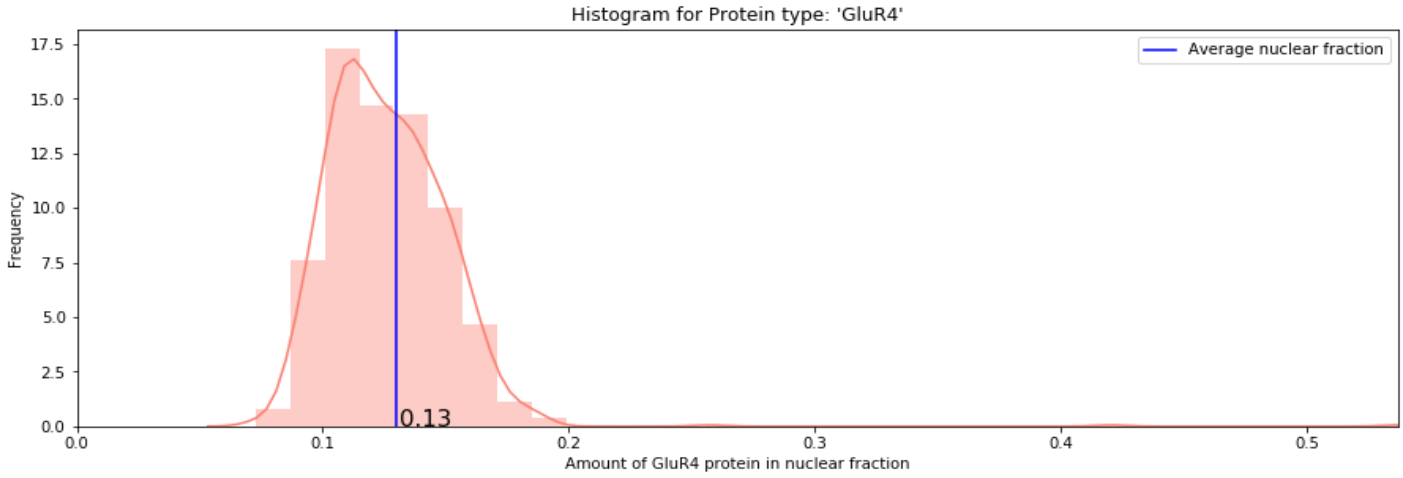



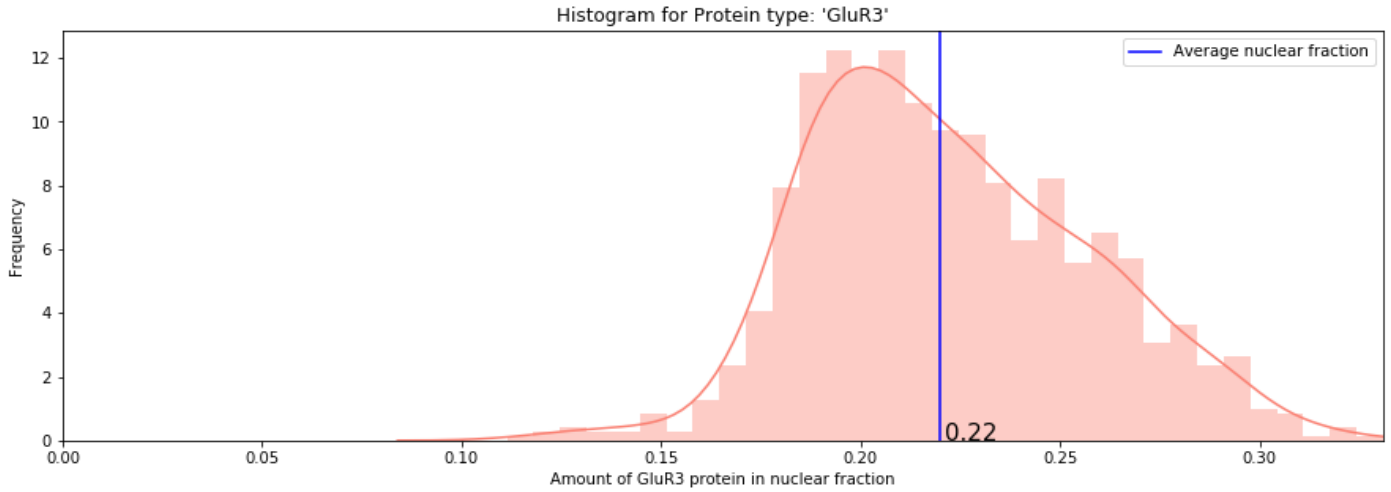


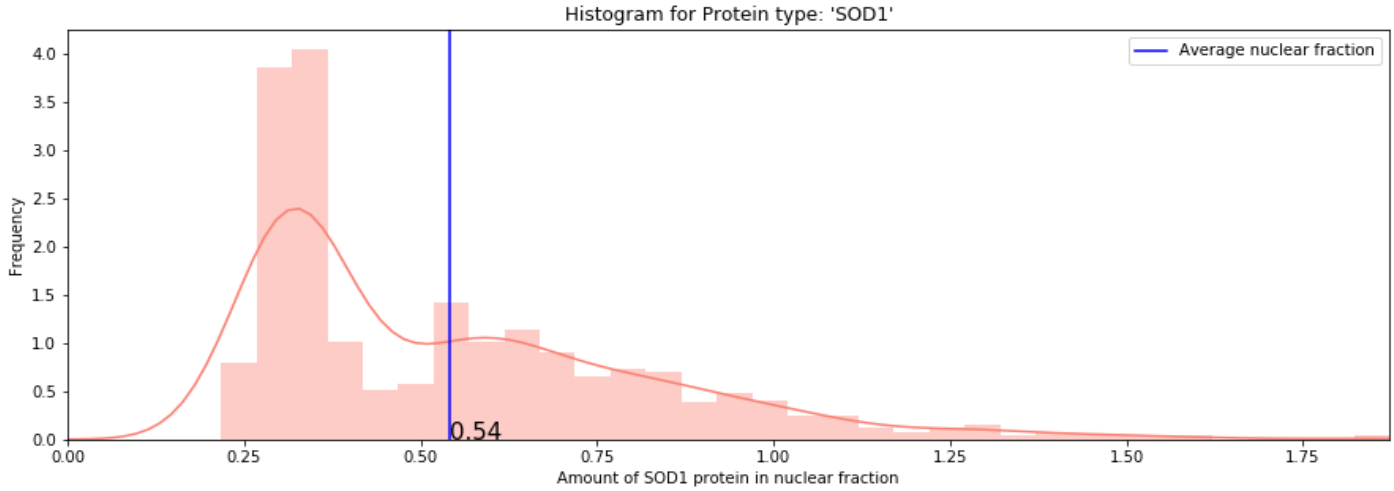




Moreover, the mice showed two different levels of Glutamate receptor proteins (GluR3 and GluR4). The graphs show that mice carried higher amount of 'GluR3' (0.22 nuclear fraction) than 'GluR4' (0.13 nuclear fraction).

The distribution shape of 'SOD1' protein is different than other plots showing a fine right skewness in the data. 'SOD1' is highly variant in nature showing average amount of 0.54 nuclear fraction in the data.





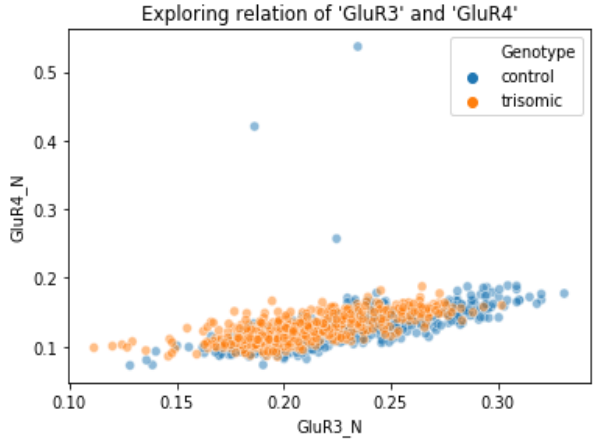

Exploring all the protein attributes show that the protein with lowest frequency has highest nuclear fraction amount, and the one with highest frequency showed the lowest nuclear fraction amount.

## Multivariate Visualizations

I have performed multivariate visualization between 10 pairs of attributes to explore any interesting relationships with the help of some plausible hypotheses as shown below:

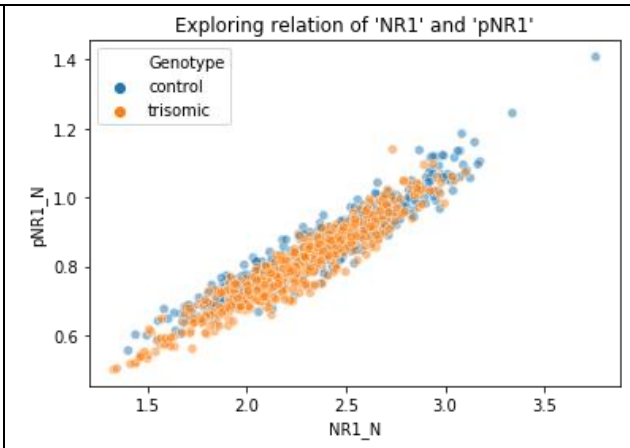
1. **There exists a linear relationship between 'GluR3' and 'GluR4' .**

- As 'GluR3' and 'GluR4' are two different levels of Glutamate Protein Receptors, it can be assumed that they may be some modifications of original Glutamate protein and can have some hidden relationship between the two.
- Plotting them as a scatter plot showed there exists a linear relationship between these two.
- Hence, providing an evidence that above stated hypothesis is true.



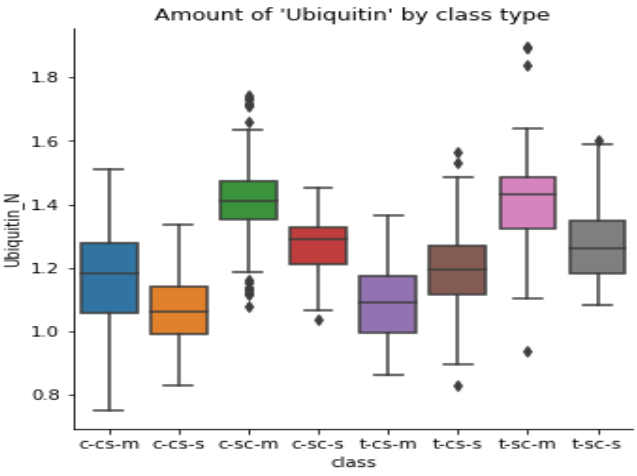
2. **There exists a linear relationship between 'NR1' and 'pNR1'.**

- Similar to above hypothesis there are two other proteins 'NR1' and 'pNR1' belonging to same family.
- Plotting these two types of proteins as shown in figure, confirms that there is a linear relationship between the two.
- Thus, providing an evidence that above hypothesis is true.



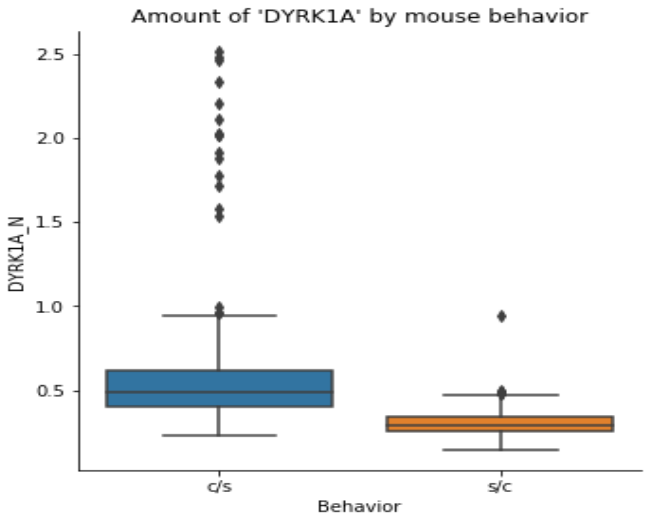

3. **Which subclass of mouse has highest protein amount of 'Ubiquitin'?**

- This hypothesis is to investigate the type of mouse showing higher amount of 'Ubiquitin' protein.
- The boxplot drawn to explore this hypothesis shows that mouse type 't-sc-m' carries the highest level of 'Ubiquitin'.
- Closely examining this boxplot shows that mice which are not stimulated to learn showed higher level of 'Ubiquitin' protein.



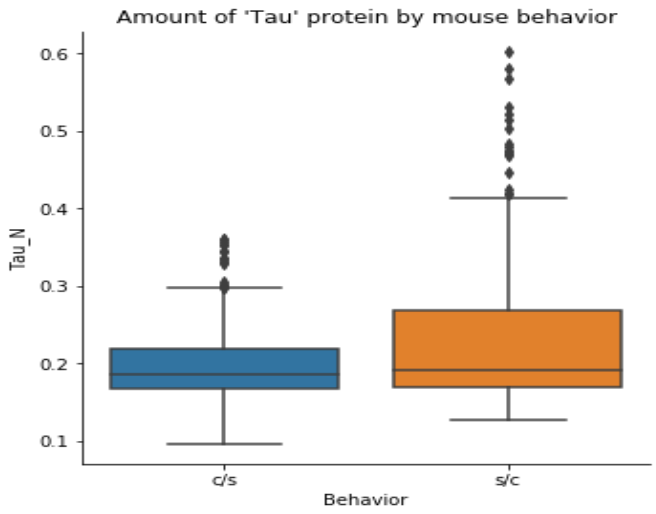

4. **Which type of behavior influences higher amount of 'DYRK1A' protein?**

- This hypothesis is to investigate the level of 'DYRK1A' protein by mouse behavior.
- Plotting this protein column by mouse behavior confirms that the mice which are stimulated to learn (context-shock or c/s) carry higher amount of protein than those who are not stimulated to learn (shock-context or s/c).



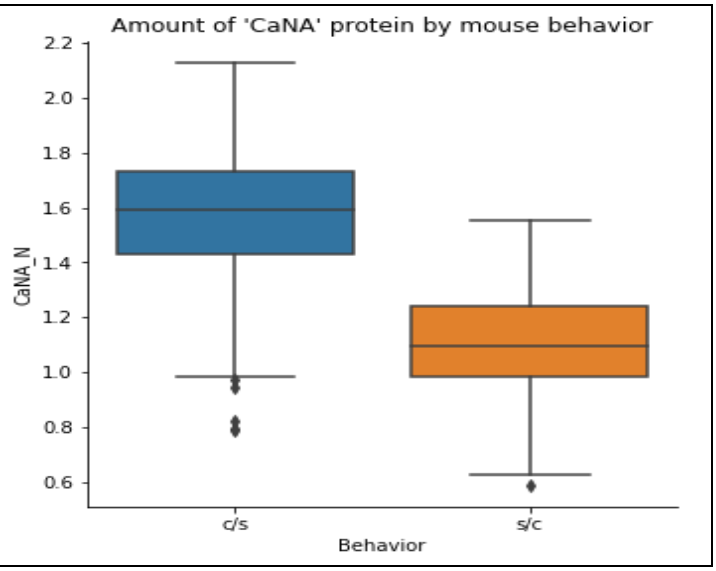
5. **Which type of behavior influences higher amount of 'Tau' protein?**

- This hypothesis is to investigate the level of 'Tau' protein by mouse behavior.
- Plotting this protein column by mouse behavior confirms that the mice that are not stimulated to learn carry higher amount of protein than those who are stimulated to learn.



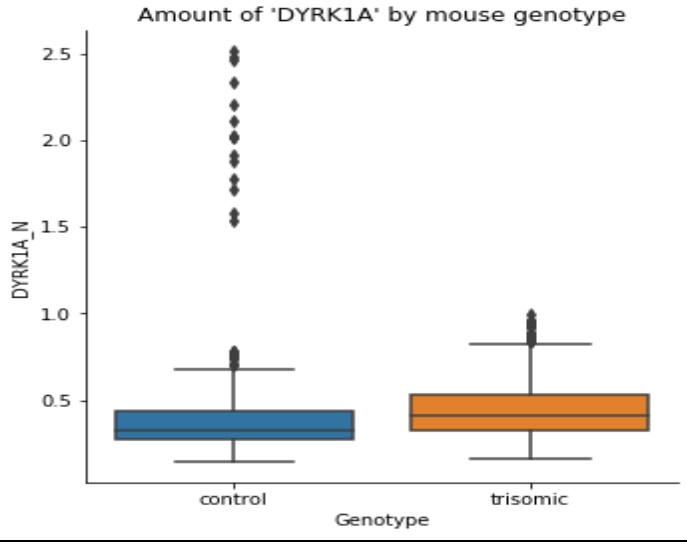

6. **Which type of behaviour influences higher amount of 'CaNA' protein?**

- The boxplot is drawn to confirm the level of 'CaNA' protein type with respect to mouse behavior.
- The plot confirms that the mice stimulated to learn showed higher amount of 'CaNA' protein



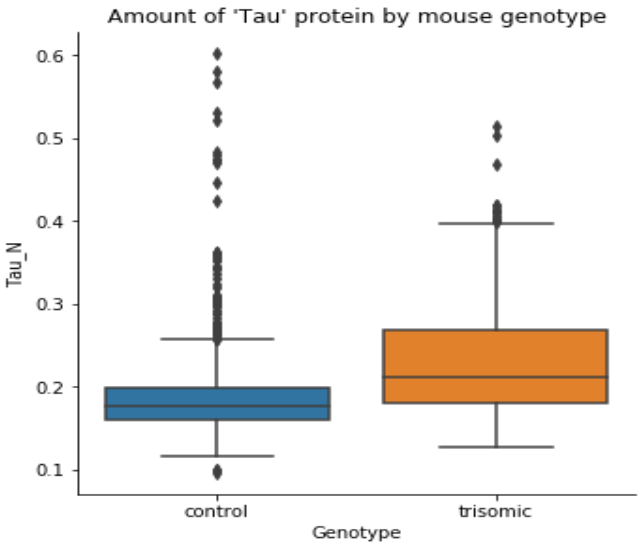

7. **Which genotype of mice carry higher amount of 'DYRK1A' protein?**

- This hypothesis is to investigate the mouse genotype (control/trisomy) that carries higher level of 'DYRK1A' protein.
- The boxplot shows that trisomic mice showed higher average amount of protein than control mice.



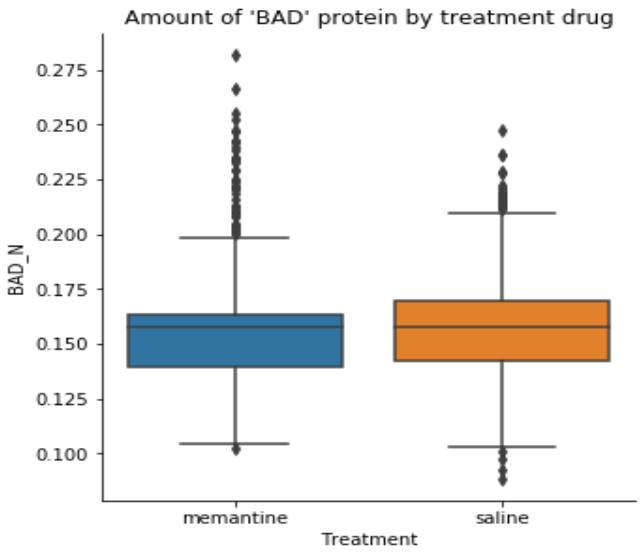
**8. Which genotype of mice carry higher amount of 'Tau' protein?**

- This hypothesis is to investigate the mouse genotype (control/trisomy) that carries higher level of 'Tau' protein.
- The boxplot shows that trisomic mice showed higher level of protein than control mice.



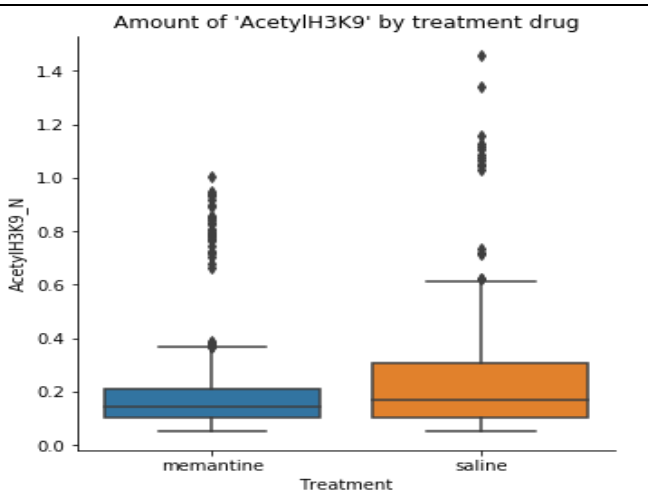

**9. Which treatment drug influences higher level of 'BAD' protein?**

- This hypothesis is to investigate the drug type that influences higher amount of 'BAD' protein within mice.
- The boxplot displays that the average amount of 'BAD' protein remains same for both the types of drugs.
- There exists many extreme values of 'BAD' protein within mice treated with memantine drug drawing the average to higher points and makes both the averages equal.
- Therefore, it does not show any strong evidence to determine the drug type that generates higher level of 'BAD' protein.




**10. Which treatment drug influences higher level of 'AcetylH3K9' protein?**

- This hypothesis is to investigate the drug type that influences higher amount of 'AcetylH3K9' protein within mice
- The boxplot shows that the mice treated with saline drug show higher amount of 'AcetylH3K9' protein than those who are treated with memantine.



## Data Modelling

Before we build the classifiers, it is important to convert the data in a format that is acceptable to models. There are four categorical variables containing string literals as a value in the dataset. Therefore, I have performed label encoding for these categorical columns to convert their values into numerical ones.

Next step is to divide the data into independent and dependent variables. I have not included the categorical variables 'Genotype', 'Treatment' and 'Behavior' into independent features, as the target variable 'class' is described based on these features. Moreover, as a dependent variable, only 'class' variable is considered.

Next, we need to scale and normalize the data as we already inferred from the histograms (in univariate visualizations) that some features are not normally distributed. I have used MinMaxScaler() because it reduces the effect of outliers while normalizing the data.

After partitioning the data into train and test sets, I will proceed towards model building process. Here, I have chosen to build two classifiers:

1. K Nearest Neighbors classifier (Similarity based)
2. Decision Tree classifier (Information based)

### Knn Classifier

KNN is an instance based learning algorithm that works on the assumption that similar data points are close to each other. To build the Knn classifier for our data set, I have tried different values of 'k' with default parameters and calculated their F1 scores to determine the best value of 'k'.

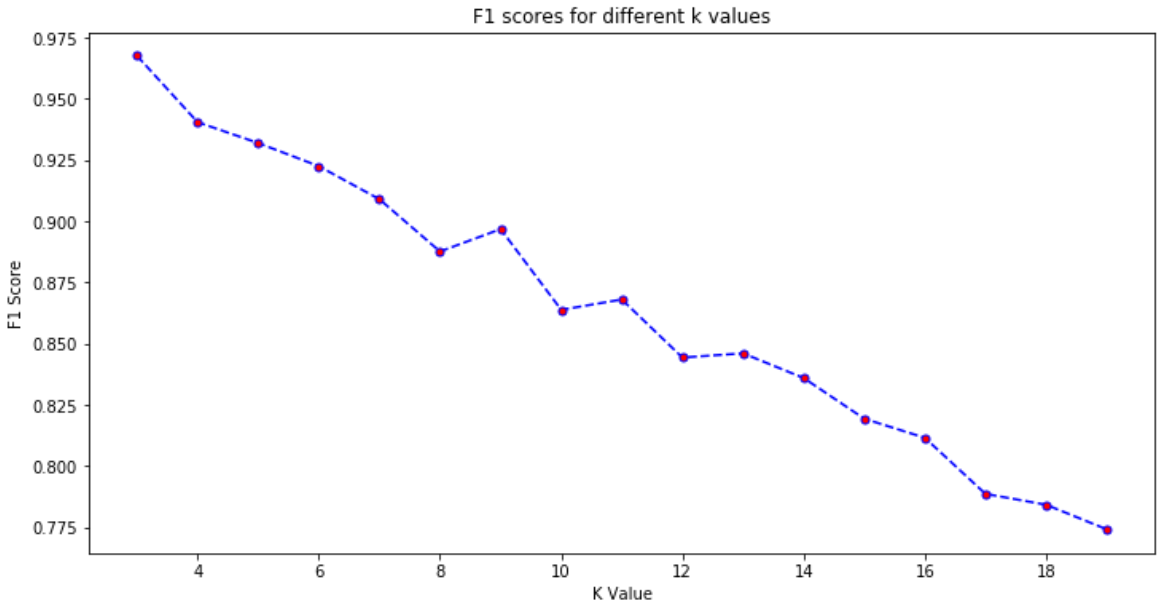



*Figure 2: Plotting F1 scores to find best 'k' value*

The F1 scores for varying values of 'k' are as shown in figure2. From the plot, it can be easily observed that k=3 produces the best result.

Therefore, I have chosen the value of 'k' neighbors to be 3 while building the classifier. I have tuned other parameters e.g. weights to be uniform so that considered points get equal weightage while predicting the target class and a distance metric to be Euclidean distance. I have also used K-fold cross-validation to check the accuracy over each fold of the data.

The built classifier is 98.61% accurate.

### Decision Tree Classifier

Decision tree is an information-based algorithm that breaks down the dataset into smaller subsets while training and incrementally keeps building the tree. To build the decision tree classifier, I have used the criterion as 'entropy' to enable the information gain. A 'random' splitter chooses the best split point randomly. 'max_features' determines the maximum no. of features to take in consideration while classifying the data. Value of 'auto' decides the max_features to be sqrt(total no. of features). I have tuned other parameters e.g. max_depth to be 50 and min_samples_split to be 3 to prevent tree from overfitting. I have performed K-fold cross validation to check the accuracy for each fold of the data.

The built classifier is 81.48% accurate.

## Model validation & comparison

The trained classifiers can be validated based on various evaluation parameters such as accuracies, confusion matrix, F1 score, precision and recall values.

Knn and decision tree classifiers give accuracy of 0.98 and 0.81 respectively over test data set. The confusion matrices for both the classifiers are as shown in table1.

The knn classifier fails to classify hardly 3 classes correctly, while decision tree model is confused for many classes.

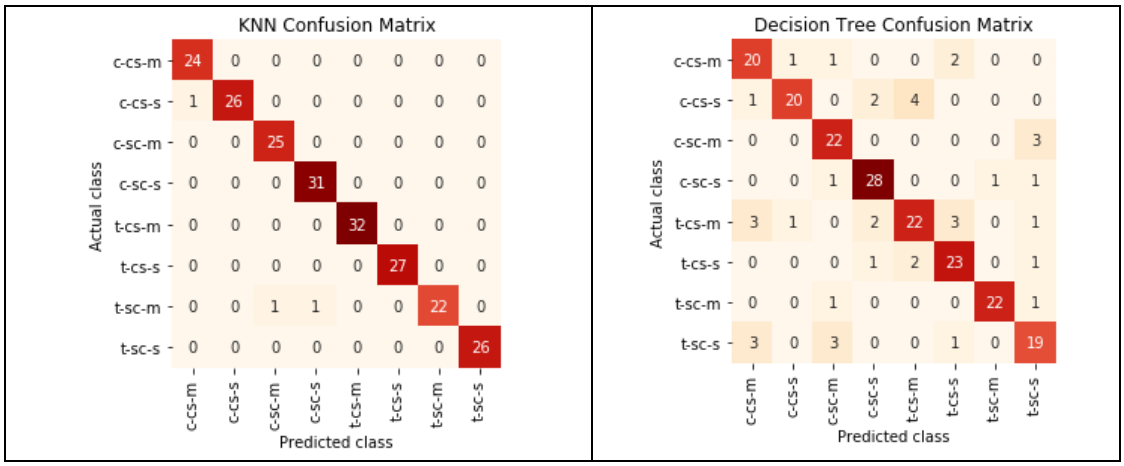



*Table 1: Confusion matrices*

The precision, recall and F1 score values for Knn classifier results to 0.99, while that for decision tree lies around 0.81 and 0.82. Moreover, for unseen validation data, Knn and decision tree classifiers give accuracy of 0.97 and 0.75 respectively.

Below table shows the comparison of both the classifiers:

| Classifier | Test set accuracy | Validation set accuracy | Precision | Recall | F1-score |
|---|---|---|---|---|---|
| KNN | 0.98 | 0.97 | 0.99 | 0.99 | 0.99 |
| Decision Tree | 0.81 | 0.75 | 0.82 | 0.81 | 0.81 |

*Table 2: Comparing classifiers*

However, the performance of decision tree is not poor and we have all numeric data points available, use of KNN for this classification problem is recommended. After careful consideration of all the evaluation parameters for both the classifiers, I would recommend knn classifier over decision tree classifier for this project.

## Limitations and Future Work

- Advanced parameter tuning was not performed while building the models as it was out of the scope for this project.
- Though we had a large number of features, we haven't performed any data reduction or feature selection techniques.
- However, in future this can be implemented to check whether it affects the model behaviour and improves the performance.

## References


- Higuera, C., Gardiner, K. and Cios, K., 2015. UCI Machine Learning Repository: Mice Protein Expression Data Set. [online] Archive.ics.uci.edu. Available at: <https://archive.ics.uci.edu/ml/datasets/Mice+Protein+Expression> [Accessed 1 June 2020].
- Seaborn.pydata.org. 2020. Plotting With Categorical Data — Seaborn 0.10.1 Documentation. [online] Available at: <https://seaborn.pydata.org/tutorial/categorical.html> [Accessed 4 June 2020].
- Jaroli, H., 2019. K-Nearest Neighbors (KNN) With Python. [online] DataScience+. Available at: <https://datascienceplus.com/k-nearest-neighbors-knn-with-python/> [Accessed 4 June 2020].
- pythonspot. 2015. K Nearest Neighbors. [online] Available at: <https://pythonspot.com/k-nearest-neighbors/> [Accessed 4 June 2020].
- Kevin's Blog. 2016. A Complete Guide To K-Nearest-Neighbors With Applications In Python And R. [online] Available at: <https://kevinzakka.github.io/2016/07/13/k-nearest-neighbor/> [Accessed 4 June 2020].
- Pant, A., 2019. Workflow Of A Machine Learning Project. [online] Medium. Available at: <https://towardsdatascience.com/workflow-of-a-machine-learning-project-ec1dba419b94> [Accessed 4 June 2020].
- Sehra, C., 2018. Decision Trees Explained Easily. [online] Medium. Available at: <https://medium.com/@chiragsehra42/decision-trees-explained-easily-28f23241248> [Accessed 4 June 2020].