**FINAL REPORT & ANALYSIS OF DATA SCIENCE PROJECT: PROTEIN EXPRESSION ANALYSIS IN MICE**

**Executive Summary**

**Overview**

This project focuses on analyzing protein expression levels in the cerebral cortex of mice to classify them based on genotype, behavior, and treatment.The aim is to identify subsets of proteins that can discriminate between these classes and to understand the biological mechanisms underlying learning and memory, particularly in the context of Down syndrome.

**Objectives:**

**Classify Mice Based on Protein Expression**

Develop a machine learning model to classify mice into one of eight categories based on the expression levels of 77 proteins. These categories are defined by a combination of genotype (control or trisomic), behavior (stimulated to learn or not), and treatment (saline or memantine).

**Identify Key Discriminant Proteins**

Utilize feature selection techniques to determine which proteins or protein modifications are most critical for distinguishing between the different classes. Identifying these key proteins can provide insights into the biological mechanisms underlying learning and memory in Down syndrome.

**Evaluate the Impact of Genotype, Behavior, and Treatment**

Analyze the influence of genotype (control vs. trisomic), behavior (context-shock vs. shock-context), and treatment (saline vs. memantine) on protein expression levels. This includes assessing how these factors affect associative learning and the potential therapeutic effects of memantine in trisomic mice.

**Methodologies:**

**Data source:** Real-time data set was provided focusing on protein expression level and impact of genotype, behavior and treatment on mice.

**Tools and Technologies**

* **Programming Language**: Python

**Libraries**:

* **Pandas**: For data manipulation and analysis.
* **NumPy**: For numerical computations.
* **Scikit-learn**: For machine learning algorithms and model evaluation.
* **Matplotlib** and **Seaborn**: For data visualization.
* **IDE**: Jupyter Notebook or any other Python IDE

**Key Findings**

**Introduction**

**Background**

Down Syndrome (DS) is a genetic disorder caused by an extra copy of chromosome 21, leading to intellectual disabilities and developmental delays. Associative learning, crucial for adapting to the environment, is often impaired in individuals with DS. Protein expression studies in mouse models (Ts65Dn mice) help uncover the molecular mechanisms underlying these cognitive deficits. These studies measure protein levels in the brain, providing insights into the biological processes affected by DS. Using mice models, which share genetic and biological similarities with humans, researchers can identify key proteins and pathways that are altered in DS. This knowledge aids in understanding the condition and evaluating potential treatments, such as memantine, to improve cognitive function and quality of life for those with DS.

**Problem Statement and Objectives**

The objective of this project is to analyze protein expression levels in the cerebral cortex of mice to classify them into distinct categories based on their genotype, behavior, and treatment. The project specifically aims to identify subsets of proteins that can distinguish these categories and to elucidate the biological mechanisms underlying learning and memory, with a particular focus on Down syndrome.

**Dataset Description**

**Dataset Characteristics**

The dataset contains protein expression levels measured in the nuclear fraction of the cerebral cortex in mice. It includes both control and trisomic (Down syndrome) mice subjected to a context fear conditioning task.

* **Type:** Multivariate
* **Subject Area:** Biology
* **Associated Tasks:** Classification, Clustering
* **Feature Type:** Real
* **Instances:** 1080
* **Features:** 80

**Experimental Design**

The dataset is divided as follows:

* **Control Mice:**
  + c-CS-s: Control, Stimulated, Saline (9 mice)
  + c-CS-m: Control, Stimulated, Memantine (10 mice)
  + c-SC-s: Control, Not Stimulated, Saline (9 mice)
  + c-SC-m: Control, Not Stimulated, Memantine (10 mice)
* **Trisomic Mice:**
  + t-CS-s: Trisomic, Stimulated, Saline (7 mice)
  + t-CS-m: Trisomic, Stimulated, Memantine (9 mice)
  + t-SC-s: Trisomic, Not Stimulated, Saline (9 mice)
  + t-SC-m: Trisomic, Not Stimulated, Memantine (9 mice)

**Data Breakdown**

* **Control Mice:** 38 mice, 570 measurements (15 measurements per protein per mouse)
* **Trisomic Mice:** 34 mice, 510 measurements (15 measurements per protein per mouse)
* **Total Measurements:** 1080 measurements per protein

**Features**

* **Protein Expression Levels:** 77 proteins/protein modifications.
* **Additional Features:** Mouse ID, Genotype, Treatment.

**Data Preprocessing**

Data preprocessing is the process of transforming raw data into a useful, understandable format, resolving issues like inconsistent formatting, human errors, and incompleteness. It's crucial for data mining and machine learning projects, affecting the success and performance of machine learning models by making data completer and more efficient for analysis.

* **Handling Missing Values:** We identified and managed missing or incomplete data. Various strategies were used to remove records with missing values, filling them with mean/median values, or using more advanced techniques like k-nearest neighbors (KNN) imputation, SimpleImputer.
* **Normalization/Scaling:** The data was adjusted to a common scale without altering differences in the ranges of values. Various techniques include min-max scaling (rescaling data to a range, often 0 to 1) and standardization (transforming data to have a mean of 0 and a standard deviation of 1). Libraries like sklearn was used to import preprocessing.
* **Encoding Categorical Variables**: Categorical data was converted into a numerical format that it could be used for machine learning algorithms. Library like LabelEncoder was imported from sklearn to perform label encoding and one-hot encoding.

**Exploratory Data Analysis (EDA)**

EDA is used to discover patterns, spot anomalies, test hypotheses, and check assumptions with the help of summary statistics and graphical representations.

* **Summary statistics:**

Libraries like pandas and numpy were used to calculate and examine mean, median, standard deviation, etc. for each protein to understand the distribution and variability of the data.

* **Visualizations of data distribution:**

Libraries like matplotlib and seaborn were used to visualize the data in the form of histograms (for numerical columns), pie charts (distribution of mice classes), bar plot (for categorical variables & distribution of target variables), heatmap (protein expression level by classes & for correlation matrix of protein expression), box plot (protein expression levels & for outlier detection & for numerical variables by target variables). These visualizations helped in providing valuable insights and identifying any patterns or anomalies in the data.

* **Correlation analysis:**

Libraries like panda, matplotlib and seaborn were used to analyze the correlations between different proteins to understand their relationships. Correlations were visually represented in the form of heatmap.

**Feature Selection**

* **Techniques used (correlation analysis, mutual information, feature importance from models):**
* **Results of feature selection and key proteins identified.**

**Model Training and Evaluation:**

Model training involves selecting appropriate machine learning algorithms, splitting the dataset into training and testing sets, tuning hyperparameters, and training the models. In this project, several models were evaluated to find the best-performing one for classifying mice based on protein expression levels.

* **Description of models used (Random Forest, SVM, Neural Networks):**

1. **Random Forest:** A robust ensemble learning method that builds multiple decision trees and merges them to get a more accurate and stable prediction.
2. **Support Vector Machine (SVM):** A powerful classifier that finds the optimal hyperplane which best separates the data into different classes.
3. **Neural Networks:** A deep learning model that can capture complex relationships in the data through multiple layers of neurons.

* **Model training process, including data splitting and hyperparameter tuning:**

Various libraries were imported from scikit-learn that included LogisticRegression, SVC, DecisionTreeClassifier, KNeighborsClassifier, RandomForestClassifier, AdaBoostClassifier, BaggingClassifier, ExtraTreesClassifier, GradientBoostingClassifier and XGBClassifier was imported from xgboost were used to train the model to get maximum accuracy and precision.

**Data Splitting:**

The dataset was split into training and testing sets using an 80:20 ratio to ensure sufficient data for both model training and evaluation.

**Hyperparameter Tuning**:

 Grid Search and Random Search methods were employed to find the best hyperparameters for each model.

 Cross-validation (5-fold) was used during hyperparameter tuning to ensure the model generalizes well to unseen data.

* **Evaluation metrics and confusion matrix analysis:**

**Results and Discussion:**

* **Interpretation of the results.**
* **Identification of key discriminant proteins.**
* **Discussion on the impact of genotype, behavior, and treatment on protein expression.**
* **Biological significance and potential implications for Down syndrome research.**

**Conclusion:**

* **Summary of key findings.**
* **Limitations of the study.**
* **Recommendations for future research.**

**References:**

* <https://pandas.pydata.org/docs/>
* <https://seaborn.pydata.org/>
* <https://matplotlib.org/stable/api/pyplot_summary.html>
* <https://scikit-learn.org/0.21/documentation.html>
* <https://numpy.org/doc/>
* <https://www.v7labs.com/blog/confusion-matrix-guide>
* <https://www.geeksforgeeks.org/ml-label-encoding-of-datasets-in-python/>
* <https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html>
* <https://builtin.com/data-science/step-step-explanation-principal-component-analysis>

**Appendix:**

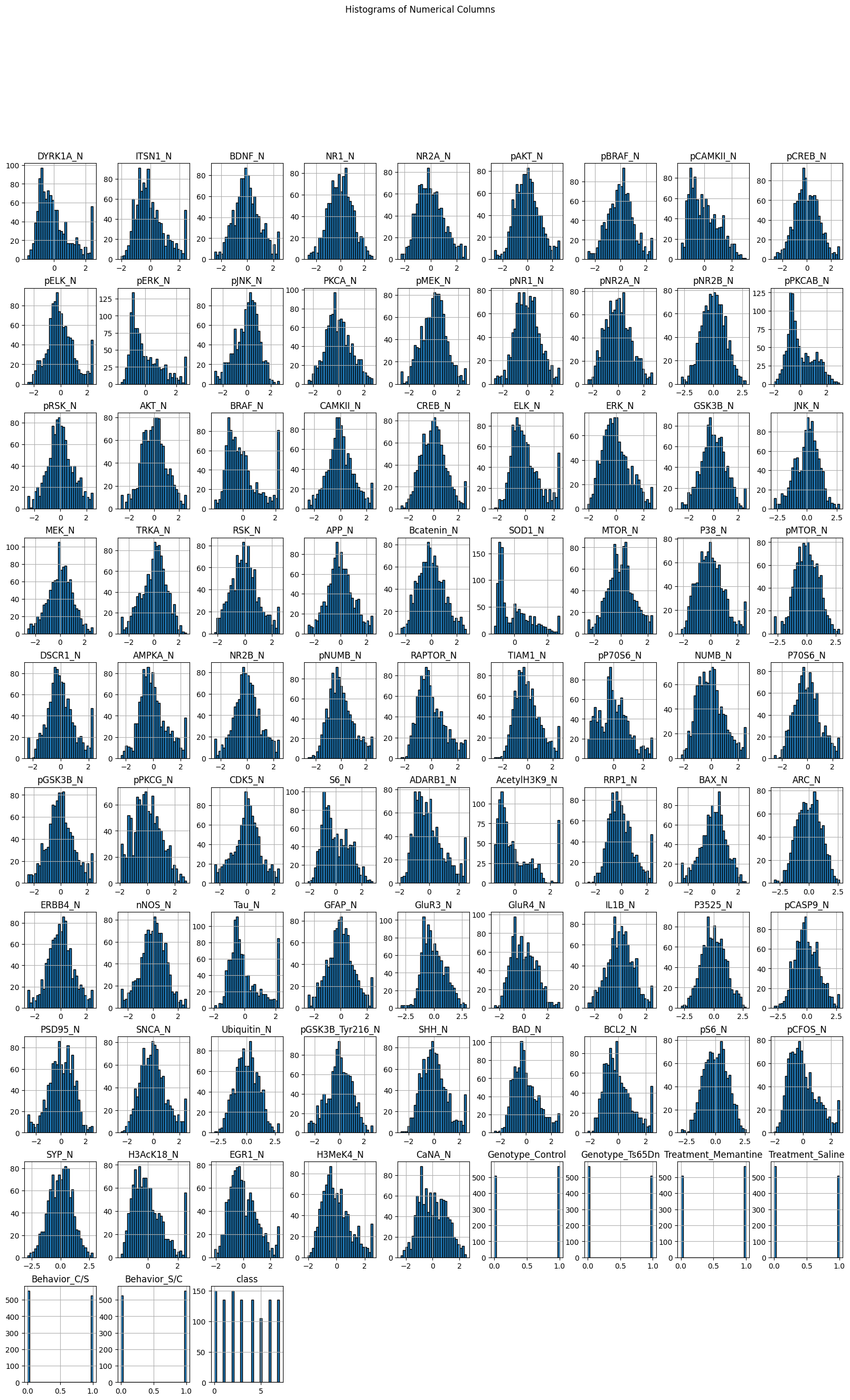
* **Additional charts, tables, or code snippets used in the analysis.**

1. Histogram

df.hist(figsize=(20, 30), bins=30, edgecolor='black')

plt.suptitle('Histograms of Numerical Columns')

plt.show()



1. Boxplot

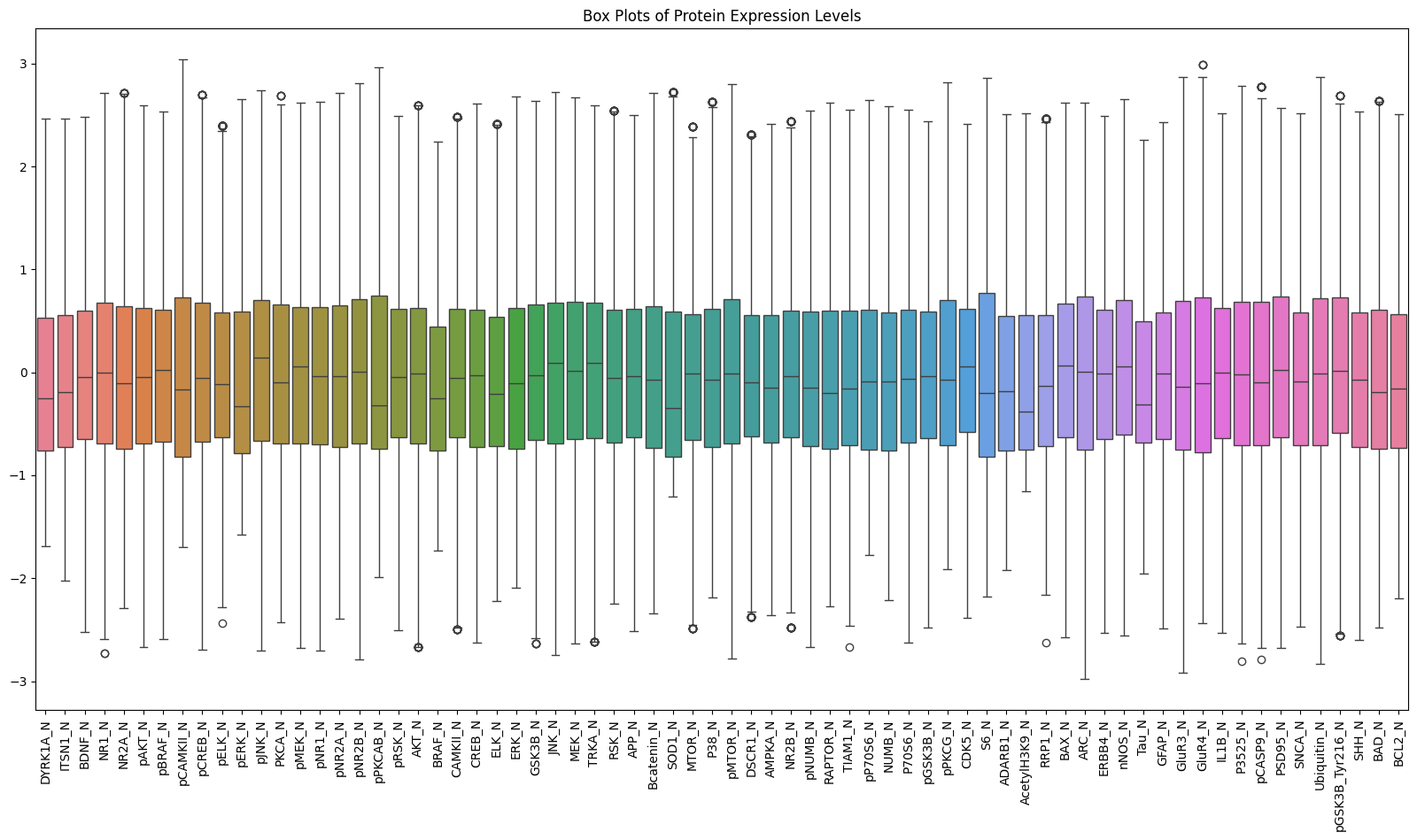
plt.figure(figsize=(20, 10))

sns.boxplot(data=df.iloc[:,:-15])

plt.xticks(rotation=90)

plt.title('Box Plots of Protein Expression Levels')

plt.show()



1. Piechart

unique\_classes = df['class'].unique()

class\_counts = df['class'].value\_counts()

class\_labels = ['Control, Stimulated, Memantine', 'Control, Stimulated, Saline',

'Control, Not Stimulated, Memantine', 'Control, Not Stimulated, Saline',

'Trisomic, Stimulated, Memantine', 'Trisomic, Stimulated, Saline',

'Trisomic, Not Stimulated, Memantine', 'Trisomic, Not Stimulated, Saline']

mapped\_class\_counts = [class\_counts.get(i, 0) for i in range(len(class\_labels))]

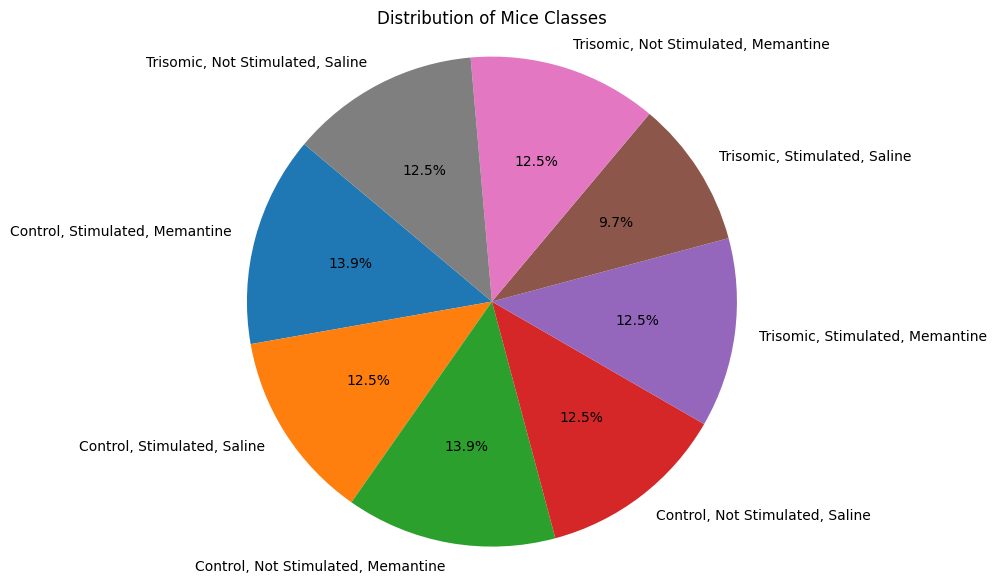
plt.figure(figsize=(10, 7))

plt.pie(mapped\_class\_counts, labels=class\_labels, autopct='%1.1f%%', startangle=140)

plt.title('Distribution of Mice Classes')

plt.axis('equal')

plt.show()



1. Barplot

categorical\_columns = ['Genotype\_Control', 'Genotype\_Ts65Dn', 'Treatment\_Memantine', 'Treatment\_Saline', 'Behavior\_C/S', 'Behavior\_S/C', 'class']

df[categorical\_columns] = df[categorical\_columns].astype('category')

categorical\_columns = df.select\_dtypes(include=['category']).columns

fig, axes = plt.subplots(len(categorical\_columns), 1, figsize=(15, 15))

fig.suptitle('Bar Plots of Categorical Variables', fontsize=20)

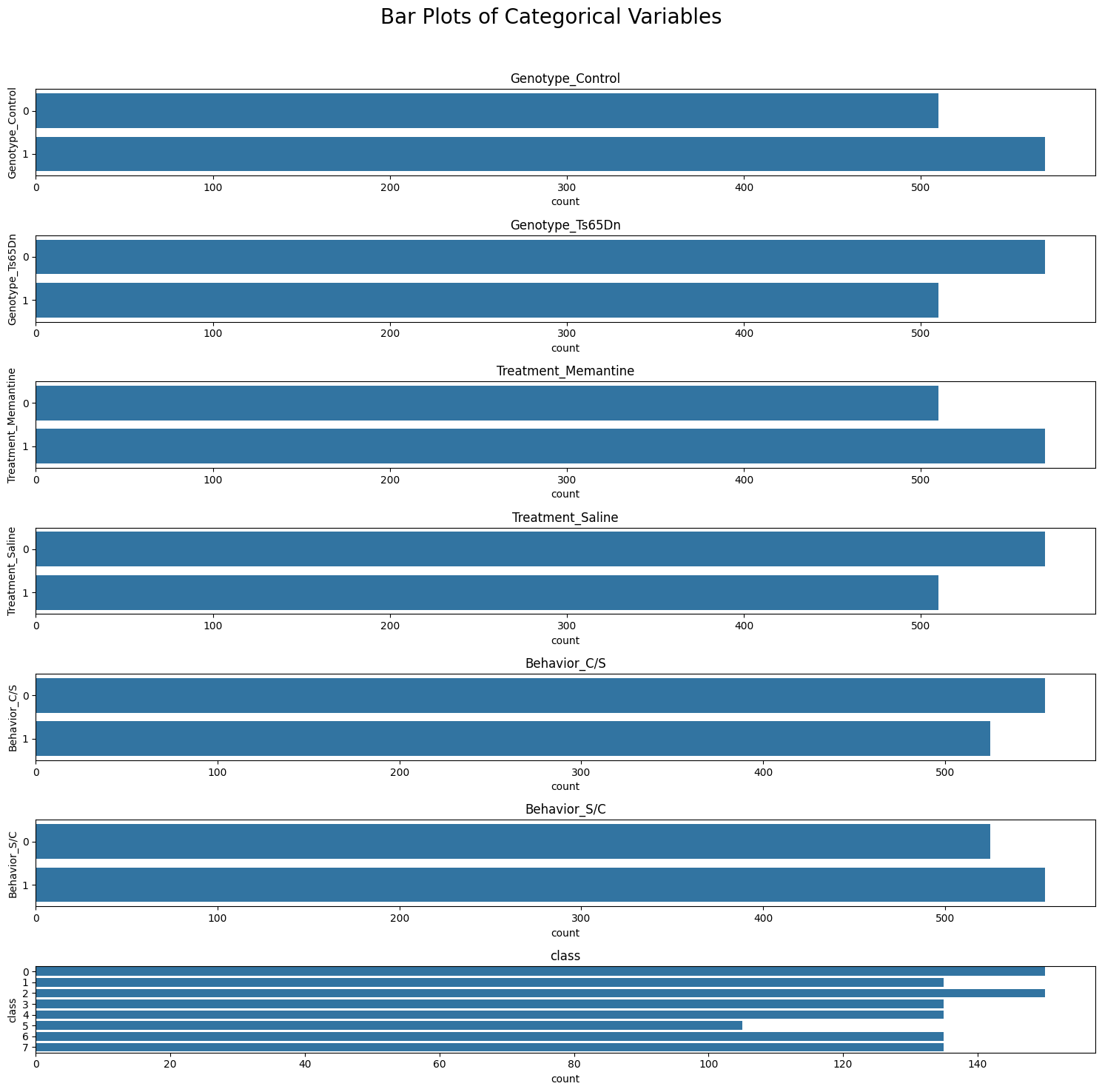
for i, col in enumerate(categorical\_columns):

sns.countplot(df[col], ax=axes[i])

axes[i].set\_title(col)

plt.tight\_layout(rect=[0, 0, 1, 0.96])

plt.show()



1. Heatmap

protein\_columns = df.columns[:77]

mean\_protein\_by\_class = df.groupby('class')[protein\_columns].mean()

plt.figure(figsize=(20, 14))

sns.heatmap(mean\_protein\_by\_class.T, cmap='viridis', annot=False, linewidths=0.5)

plt.title('Heatmap of Protein Expression Levels by Class')

plt.xlabel('Classes')

plt.ylabel('Proteins')

plt.xticks(ticks=range(len(mean\_protein\_by\_class.index)), labels=mean\_protein\_by\_class.index, ha='right')

plt.yticks(ticks=range(len(protein\_columns)), labels=protein\_columns)

plt.show()

