Contents

1. Introduction

Alzheimer's disease (AD) is a complex, untreatable brain disorder linked to aging. To better understand it, we conducted a meta-analysis of publicly available proteomics data from 4,089 samples of brain tissues and blood from AD patients and non-AD controls, categorized into three age groups: < 75, 75–84, and ≥ 85 years. We merged the datasets and used supervised machine learning approach to identify proteins that distinguish AD from controls, followed by network-based pathway and enrichment analyses. Our analyses highlighted key themes such as cell death, cellular senescence, energy metabolism, and oxidative stress, with age-specific patterns emerging—cellular senescence was prominent in younger age groups, while cell death was most relevant in the youngest patients. These findings align with existing literature, underscoring the importance of these hallmarks in AD (Shokhirev and Johnson).

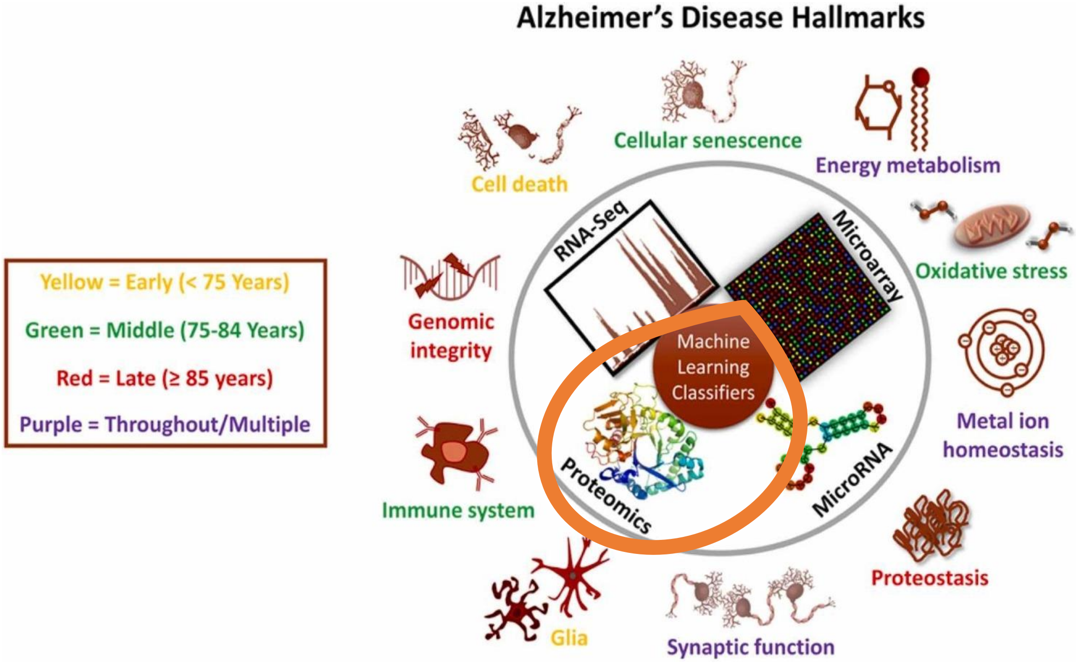


Figure.1 Graphical Abstract

2. Literature Review

The research paper titled "An Integrative Machine-Learning Meta-Analysis of High-Throughput Omics Data Identifies Age-Specific Hallmarks of Alzheimer's Disease" focuses on data collection from RNA-Seq, microarray, proteomics, and microRNA samples obtained from patients with Alzheimer's disease (AD) and controls. By analyzing omics data from patients with Alzheimer's disease, researchers used machine-learning techniques to discover specific disease markers linked to aging, cell death, and cellular senescence. Although the study showed promising accuracy in diagnosis, challenges such as incomplete proteomics data and limited sample sizes restricted the depth of insights gained (Shokhirev and Johnson).

The results of this study may influence the treatment of Alzheimer's disease by pinpointing age-related markers and creating targeted therapies. Utilizing machine learning algorithms can enhance early detection, while personalized treatment plans can be tailored to individual characteristics. Additionally, understanding crucial biological pathways can facilitate the development of new medications (Shokhirev and Johnson).

3. Methodology

3.1 Data Collection

We gathered publicly available proteomics dataset. In that each column represents a protein sample, with identifiers indicating the condition, sex, age, and disease stage. This data is important for studying the patterns of protein expression in relation to Alzheimer's disease. Metadata file provides experimental conditions, sample IDs, age, sex, diagnosis, tissue type, and batch information were then collected. After that combinedSTARcounts.txt. file was downloaded. It contains the number of unmapped reads across different samples, with each column representing a specific sample. This raw count data requires further normalization and differential expression analysis. Additionally, Supplementary files were used for additional protein details.

3.2 Data preprocessing

In the preprocessing step, a new meta file was created, incorporating sample details, genes, and proteins from combinedSTARcounts.txt. To facilitate focused analysis, three additional files were generated, listing proteins for young, middle-aged, and old age groups, and all files were converted to

CSV format. Missing values were checked and handled by converting categorical values to numeric using LabelEncoder() and replacing NaN values with column means.

The preprocessed files were saved as new CSV files for future use. During normalization, column names and protein lists were standardized, and mismatches were corrected. Protein data was filtered by age groups and combined with metadata. In feature engineering, categorical values such as age and sex were converted to numerical values using LabelEncoder(). Missing values were replaced with column means using fillna() and mean(). An age group column was added to the new metadata with the following conditions: if age < 50, assign 'young'; if 50 ≤ age < 70, assign 'middle_age'; otherwise, assign 'old'.

3.3 Analysis Techniques

The process of implementing the machine learning model involves several stages to analyze and predict the diagnosis status using protein expression data. Initially, the target variable ('Diagnosis') is extracted from the metadata and converted into numerical form using LabelEncoder(). Any missing values in the target variable are removed. Next, the protein expression data is aligned with the target variable by matching common indices. The dataset is then split into training (70%) and test (30%) sets.

Various models are used, starting with logistic regression with L1 regularization, followed by a Random Forest classifier known for its robustness and ability to prevent overfitting. Performance metrics such as mean squared error (MSE), mean absolute error (MAE), and R-squared (R2) are used to evaluate the models' performance. Additionally, a Decision Tree classifier is created with the 'gini' criterion and a maximum depth of 3, with its effectiveness assessed based on accuracy, recall, precision, and F1 score. Lastly, a Support Vector Machine (SVM) model is trained. The overall performance of the models is summarized through classification reports, providing a comprehensive overview of each model's diagnostic capabilities using protein data.

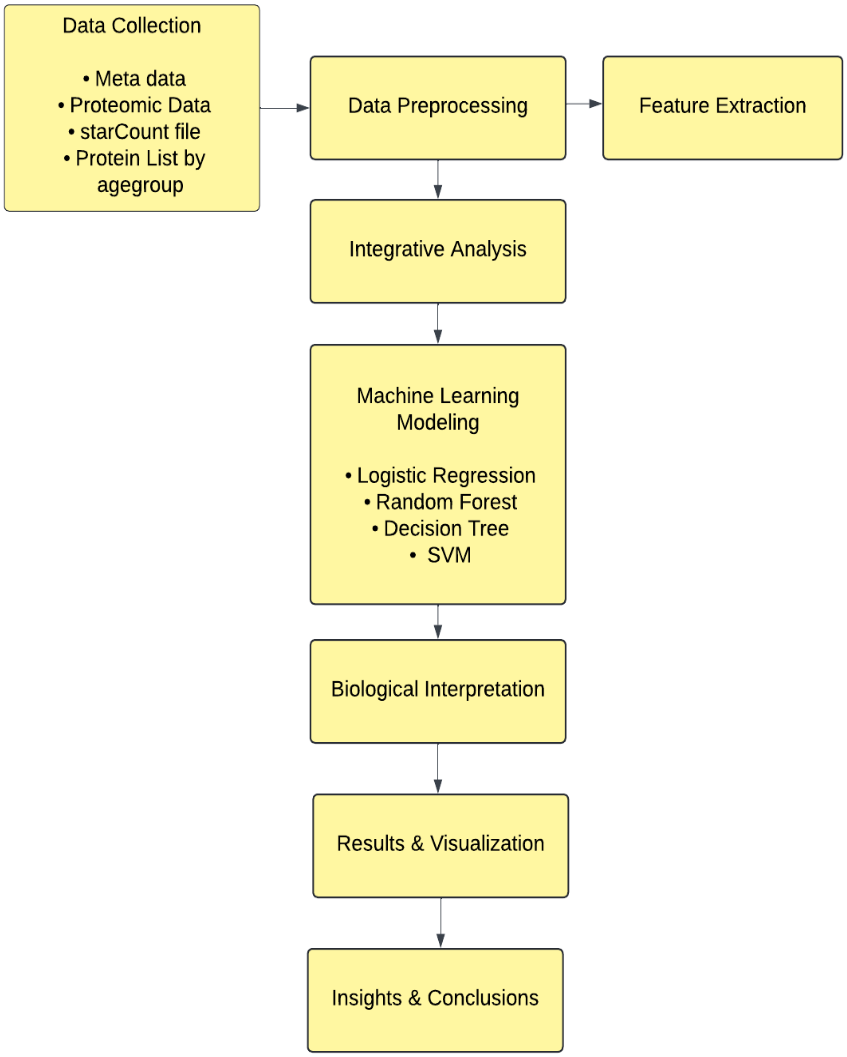


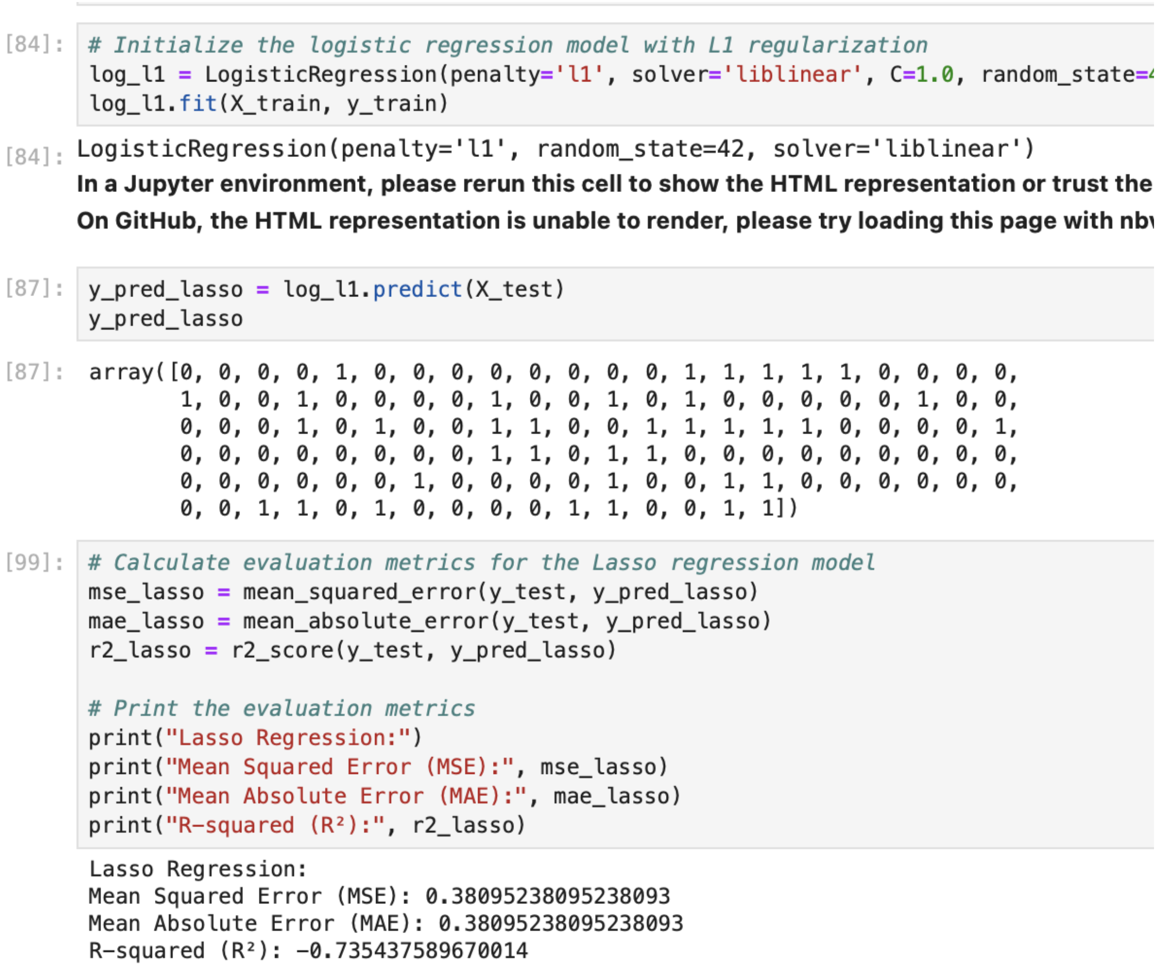
Figure 2. Methodology Workflow

3.4 Machine Learning Model Implementation

Model 1: Logistic Regression

Logistic regression is used for binary classification problems. It predicts the probability of an outcome that can only have two values (e.g., yes/no, 0/1). We have also used L1 regularization (also known as Lasso) which tends to produce sparse models by pushing some coefficients to exactly zero.

This feature selection effect can help in simplifying the model and potentially improving generalization by focusing on the most relevant features.



```
[84]: # Initialize the logistic regression model with L1 regularization
      log_l1 = LogisticRegression(penalty='l1', solver='liblinear', C=1.0, random_state=4
      log_l1.fit(X_train, y_train)

[84]: LogisticRegression(penalty='l1', random_state=42, solver='liblinear')
      In a Jupyter environment, please rerun this cell to show the HTML representation or trust the
      On GitHub, the HTML representation is unable to render, please try loading this page with nbv

[87]: y_pred_lasso = log_l1.predict(X_test)
      y_pred_lasso

[87]: array([0, 0, 0, 0, 1, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0,
             1, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 0, 1, 0, 0, 0, 0, 0, 1, 0, 0,
             0, 0, 0, 1, 0, 1, 0, 0, 1, 1, 0, 0, 1, 1, 1, 1, 1, 0, 0, 0, 0, 1,
             0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 0, 1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0,
             0, 0, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 1, 1, 0, 0, 0, 0, 0, 0,
             0, 0, 1, 1, 0, 1, 0, 0, 0, 1, 1, 0, 0, 1, 1])

[99]: # Calculate evaluation metrics for the Lasso regression model
      mse_lasso = mean_squared_error(y_test, y_pred_lasso)
      mae_lasso = mean_absolute_error(y_test, y_pred_lasso)
      r2_lasso = r2_score(y_test, y_pred_lasso)

      # Print the evaluation metrics
      print("Lasso Regression:")
      print("Mean Squared Error (MSE):", mse_lasso)
      print("Mean Absolute Error (MAE):", mae_lasso)
      print("R-squared (R²):", r2_lasso)

      Lasso Regression:
      Mean Squared Error (MSE): 0.38095238095238093
      Mean Absolute Error (MAE): 0.38095238095238093
      R-squared (R²): -0.735437589670014
```

Fig 3.4.1 Logistic Regression

- MSE (0.38): An MSE of 0.3809 suggests that the average squared error in the model's predictions is relatively high. In a well-performing regression model, MSE should be as low as possible, ideally close to zero.
- MAE (0.38): An MAE of 0.3809 indicates that, on average, the model's predictions deviate from the actual values by about 0.381. Like MSE, a lower MAE indicates better predictive accuracy.
- R-SQUARE (-0.73): An R-Squared of -0.7354 indicates that the model does not fit the data well. A negative R-Squared suggests that the model is performing worse than a baseline model that simply predicts the mean of the target variable for all observations.

Model 2: Decision Tree

Decision trees are flowchart-like structures that split data into branches to make predictions or decisions. They can handle both classification and regression tasks.

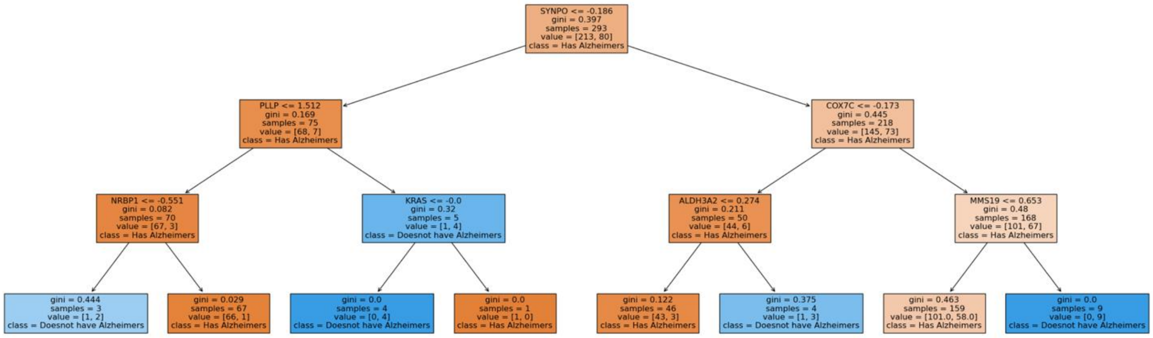


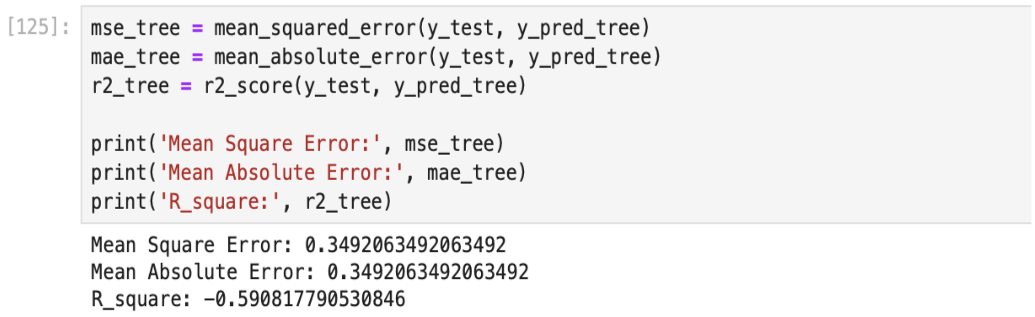
Fig 3.4.2 Decision Tree Plot

ROOT NODE: The tree starts with a split on the feature SYNPO, with a threshold of -0.186. It has a Gini impurity of 0.397 and 293 samples, with the majority (213) being classified as "Has Alzheimer's".

LEFT SUBTREE: The first left node splits on PILP with a threshold of 1.512. Subsequent splits occur on NRBP1 and KRAS, further refining the classification.

The majority of nodes under this branch predict "Has Alzheimer's", with a few predicting "Does not have Alzheimer's".

RIGHT SUBTREE: The first right node splits on COX7C with a threshold of -0.173. Further splits involve features ALDH3A2 and MMS19. Leaf nodes here predominantly predict "Has Alzheimer's.

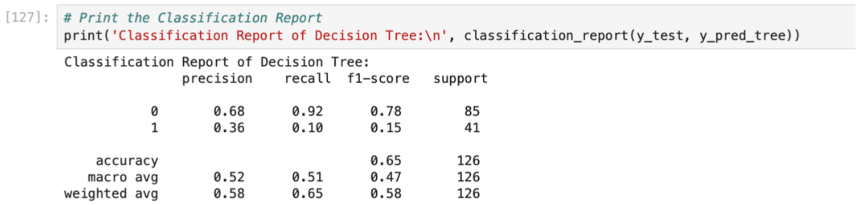


```
[125]: mse_tree = mean_squared_error(y_test, y_pred_tree)
       mae_tree = mean_absolute_error(y_test, y_pred_tree)
       r2_tree = r2_score(y_test, y_pred_tree)

       print('Mean Square Error:', mse_tree)
       print('Mean Absolute Error:', mae_tree)
       print('R_square:', r2_tree)

       Mean Square Error: 0.3492063492063492
       Mean Absolute Error: 0.3492063492063492
       R_square: -0.590817790530846
```

Fig 3.4.3 Decision Tree MSE, MAE and R-Square

- MSE (0.341): An MSE of 0.341 suggests that the average squared error in the model's predictions is 0.341. This indicates how far, on average, the model's predictions deviate from the actual values.
- MAE (0.341): An MAE of 0.341 means that, on average, the absolute error in the predictions is 0.341. Like MSE, a lower MAE indicates better accuracy, with an ideal value close to zero.
- R-SQUARE (-0.554): An R-Squared value of -0.554 indicates a poor fit of the model to the data, suggesting that the model does not explain the variability in the data well. A negative R-Squared means that the model's predictions are worse than using the mean of the target variable as a predictor.

Decision Tree Classification Report



```
[127]: # Print the Classification Report
       print('Classification Report of Decision Tree:\n', classification_report(y_test, y_pred_tree))

       Classification Report of Decision Tree:
                     precision    recall  f1-score   support

                  0       0.68      0.92      0.78        85
                  1       0.36      0.10      0.15        41

           accuracy                           0.65       126
          macro avg       0.52      0.51      0.47       126
       weighted avg       0.58      0.65      0.58       126
```

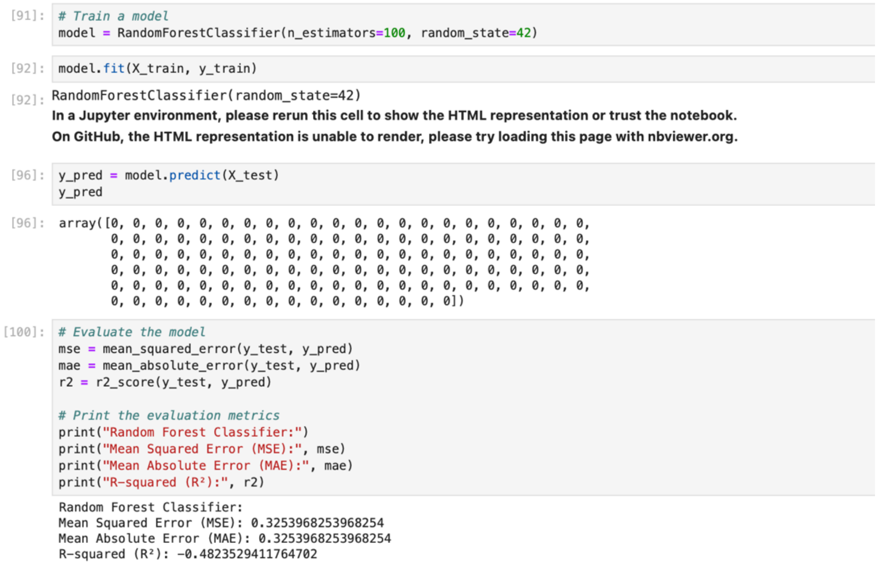
Fig 3.4.4 Decision Tree Classification Report

- ACCURACY SCORE (0.67): An accuracy of 0.67 indicates that 67% of the predictions made by the Decision Tree model are correct. While accuracy is a straightforward measure of model performance, it can be misleading if the classes are imbalanced.

- PRECISION (1.00): A precision of 1.00 indicates that all instances predicted as positive by the model were actually positive, meaning there were no false positives. This is ideal in scenarios where false positives are costly or undesirable.
- F1 SCORE (0.81): An F1 score of 0.81 indicates a good balance between precision and recall, reflecting that the model performs well in identifying positive cases while minimizing both false positives and false negatives.

Model 3 : Random Forest

Random forests are an ensemble learning method that combines multiple decision trees to improve prediction accuracy and control overfitting. We chose Random Forest as it is efficient in handling complex classification tasks and datasets with multiple features.

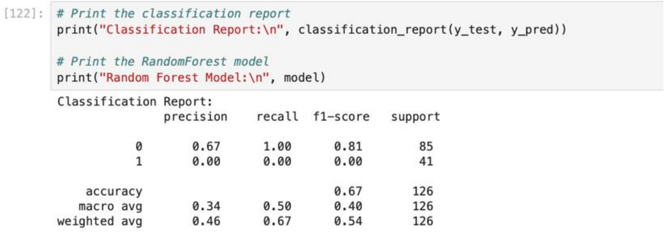


```
[91]:  # Train a model
       model = RandomForestClassifier(n_estimators=100, random_state=42)

[92]:  model.fit(X_train, y_train)

[92]: RandomForestClassifier(random_state=42)
      In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
      On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.

[96]:  y_pred = model.predict(X_test)
       y_pred

[96]:  array([0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0,
              0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0])

[100]: # Evaluate the model
       mse = mean_squared_error(y_test, y_pred)
       mae = mean_absolute_error(y_test, y_pred)
       r2 = r2_score(y_test, y_pred)

       # Print the evaluation metrics
       print("Random Forest Classifier:")
       print("Mean Squared Error (MSE):", mse)
       print("Mean Absolute Error (MAE):", mae)
       print("R-squared (R²):", r2)

       Random Forest Classifier:
       Mean Squared Error (MSE): 0.3253968253968254
       Mean Absolute Error (MAE): 0.3253968253968254
       R-squared (R²): -0.4823529411764702
```

Fig 3.4.5 Random Forest MSE, MAE, R-Square

- MSE (0.325): An MSE of 0.325 indicates that, on average, the squared difference between the predicted and actual values is 0.325. In general, a lower MSE suggests better model performance, with the ideal being as close to 0 as possible.
- MAE (0.325): An MAE of 0.325 indicates that the average absolute error between the predicted and actual values is 0.325. Like MSE, a lower MAE indicates better predictive accuracy.
- R-SQUARE (-0.482): An R-Squared value of -0.482 suggests a poor fit, indicating that the model does not explain the variability in the data well. In fact, the model performs worse than a simple model that would predict the mean value of the target variable.

Random Forest Classification Report



```
[122]: # Print the classification report
       print("Classification Report:\n", classification_report(y_test, y_pred))

       # Print the RandomForest model
       print("Random Forest Model:\n", model)
```

```
Classification Report:
               precision    recall  f1-score   support

           0       0.67      1.00      0.81        85
           1       0.00      0.00      0.00        41

    accuracy                           0.67       126
   macro avg       0.34      0.50      0.40       126
weighted avg       0.46      0.67      0.54       126
```

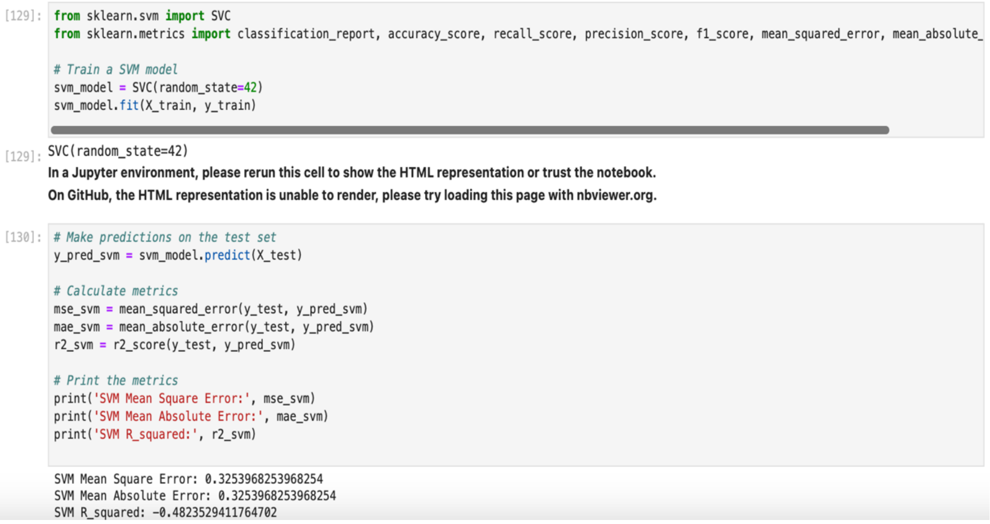
Fig 3.4.6 Random Forest Classification Report

- ACCURACY SCORE (0.67): An accuracy of 0.67 means that 67% of the predictions made by the model are correct. While useful, accuracy alone may not be sufficient to evaluate a model's performance, especially if the dataset is imbalanced (i.e., one class is much more frequent than others).
- PRECISION (0.96): A precision of 0.96 indicates that 96% of the instances predicted as positive by the model are positive. High precision is crucial in scenarios where false positives are costly or undesirable.
- F1 SCORE (0.79): An F1 score of 0.79 suggests a good balance between precision and recall, reflecting that the model performs well in identifying positive cases without introducing too many false positives or false negatives.

Model 4: SVM (Support Vector Machine)

Support Vector Machines (SVM) are powerful supervised learning models used for classification and regression tasks. SVMs are particularly effective in high-dimensional spaces and cases where the decision boundary is complex and not linearly separable. In our case we chose SVM because it can predict the function of proteins or genes.

Although we know SVM has its limitations such as high computation resources which makes it expensive and less interpretable than the other models we have chosen.



```
[129]: from sklearn.svm import SVC
       from sklearn.metrics import classification_report, accuracy_score, recall_score, precision_score, f1_score, mean_squared_error, mean_absolute_

       # Train a SVM model
       svm_model = SVC(random_state=42)
       svm_model.fit(X_train, y_train)
```

```
[129]: SVC(random_state=42)
In a Jupyter environment, please rerun this cell to show the HTML representation or trust the notebook.
On GitHub, the HTML representation is unable to render, please try loading this page with nbviewer.org.
```

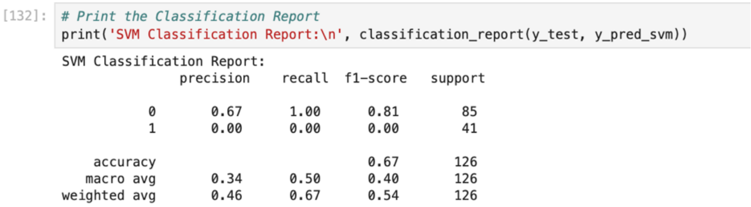
```
[130]: # Make predictions on the test set
       y_pred_svm = svm_model.predict(X_test)

       # Calculate metrics
       mse_svm = mean_squared_error(y_test, y_pred_svm)
       mae_svm = mean_absolute_error(y_test, y_pred_svm)
       r2_svm = r2_score(y_test, y_pred_svm)

       # Print the metrics
       print('SVM Mean Square Error:', mse_svm)
       print('SVM Mean Absolute Error:', mae_svm)
       print('SVM R_squared:', r2_svm)
```

```
SVM Mean Square Error: 0.3253968253968254
SVM Mean Absolute Error: 0.3253968253968254
SVM R_squared: -0.4823529411764702
```

Fig 3.4.7 SVM MSE, MAE and R-Square

- MSE (0.325): An MSE of 0.325 suggests that, on average, the squared difference between the predicted and actual values is 0.325. In regression tasks, an MSE closer to 0 is preferred, indicating more accurate predictions.
- MAE (0.325): MAE measures the average magnitude of the errors in a set of predictions, without considering their direction. It's the average over the test sample of the absolute differences between prediction and actual observation.
- R-SQUARE (-0.482): An R-Squared value of -0.482 indicates a poor fit of the model to the data, suggesting that the SVM model does not explain the variability in the response data well. In fact, the model's predictions are worse than simply using the mean of the data for prediction.

SVM Classification Report



```
[132]: # Print the Classification Report
       print('SVM Classification Report:\n', classification_report(y_test, y_pred_svm))
```

```
SVM Classification Report:
               precision    recall  f1-score   support

           0       0.67      1.00      0.81        85
           1       0.00      0.00      0.00        41

    accuracy                           0.67       126
   macro avg       0.34      0.50      0.40       126
weighted avg       0.46      0.67      0.54       126
```

Fig 3.4.8 SVM Classification Report

- ACCURACY SCORE (0.67): Indicates that 67% of the predictions made by the SVM model are correct. Accuracy is the ratio of correctly predicted instances to the total instances in the dataset.

- PRECISION (1.00): Precision is the ratio of correctly predicted positive observations to the total predicted positives. A precision score of 1.00 means that all instances predicted as positive by the model are indeed positive, indicating no false positives.
- F1-SCORE (0.81): The F1 score is the harmonic mean of precision and recall. It provides a single metric that balances the trade-off between precision and recall. An F1 score of 0.81 suggests that the model has a good balance between precision and recall, with strong performance in correctly identifying positive instances.

3.5 Model Comparison

	Logistic Regression	Decision Tree	Random Forest	SVM
Mean Square Error (MSE)	0.341	0.341	0.325	0.325
Mean Absolute Error (MAE)	0.341	0.341	0.325	0.325
R-Square	-0.554	-0.554	-0.482	-0.482

Table 3.5.1 Model Comparison of MSE, MAE, R-Square

Observations:

- Negative R² values suggest that none of the models are well-suited for the data, as they do not explain the variance effectively. But the Random Forest and SVM model shows a marginally better fit compared to the other two models.
- In future we could possibly use other models such as gradient boosting and ensemble models for better performance.

	Decision Tree	Random Forest	SVM
Accuracy Score	0.67	0.67	0.67
Precision Score	1.00	0.96	1.00
F1 Score	0.81	0.79	0.81

Table 3.5.2 Model Comparison Accuracy, Precision and F1 Score

Observations:

- The Decision Tree shows higher precision, meaning it was very accurate when predicting the positive class, but poor results for recall.
- The SVM, while having slightly lower precision and F1 score, offers better generalization due to being an ensemble method, potentially leading to better performance on unseen data.

4 Results

4.1 Distribution of Target Classes

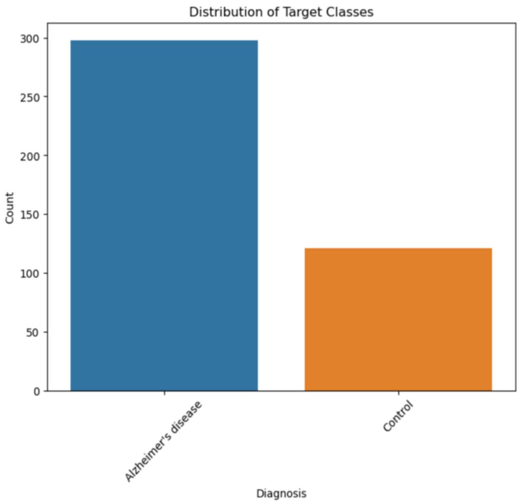


Figure 4.1 Distribution of Target Classes

- The graph titled" Distribution of Target Classes" illustrates the number of instances for two diagnostic categories: Alzheimer's Disease and Control.
- The bar representing Alzheimer's Disease is significantly taller, indicating a higher count, visually appearing to exceed 300.
- In contrast, the Control bar is shorter, with a count approximately half that of the Alzheimer's Disease category, around 150.
- This distribution suggests that the dataset contains more instances of Alzheimer's Disease than the Control group, which may be important for studies concentrating on Alzheimer's Disease. This indicates a higher prevalence or a larger sample size for this condition in the dataset

4.2 Feature Importances

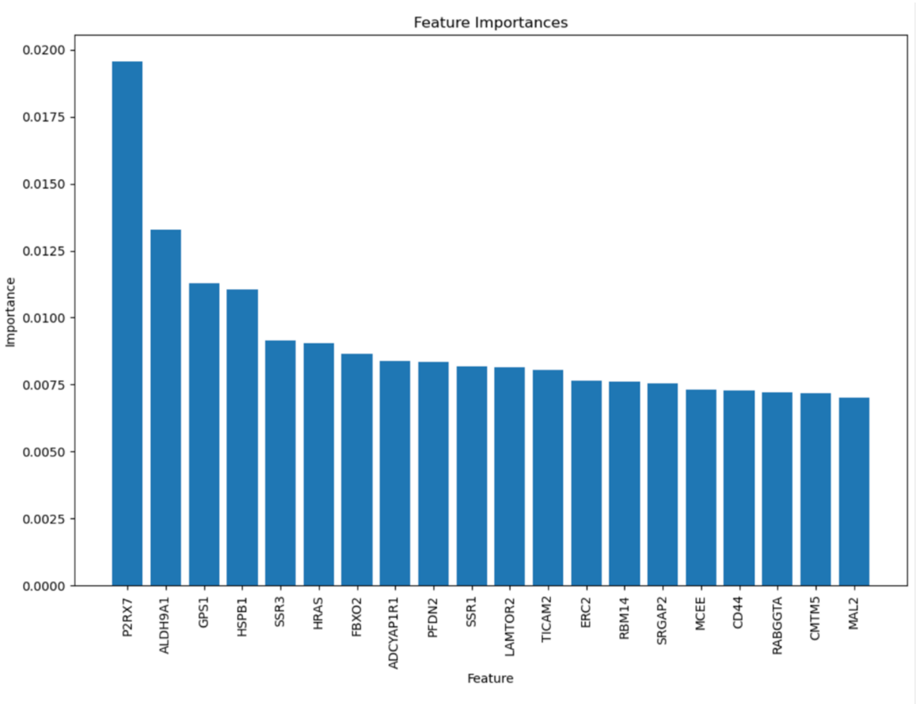
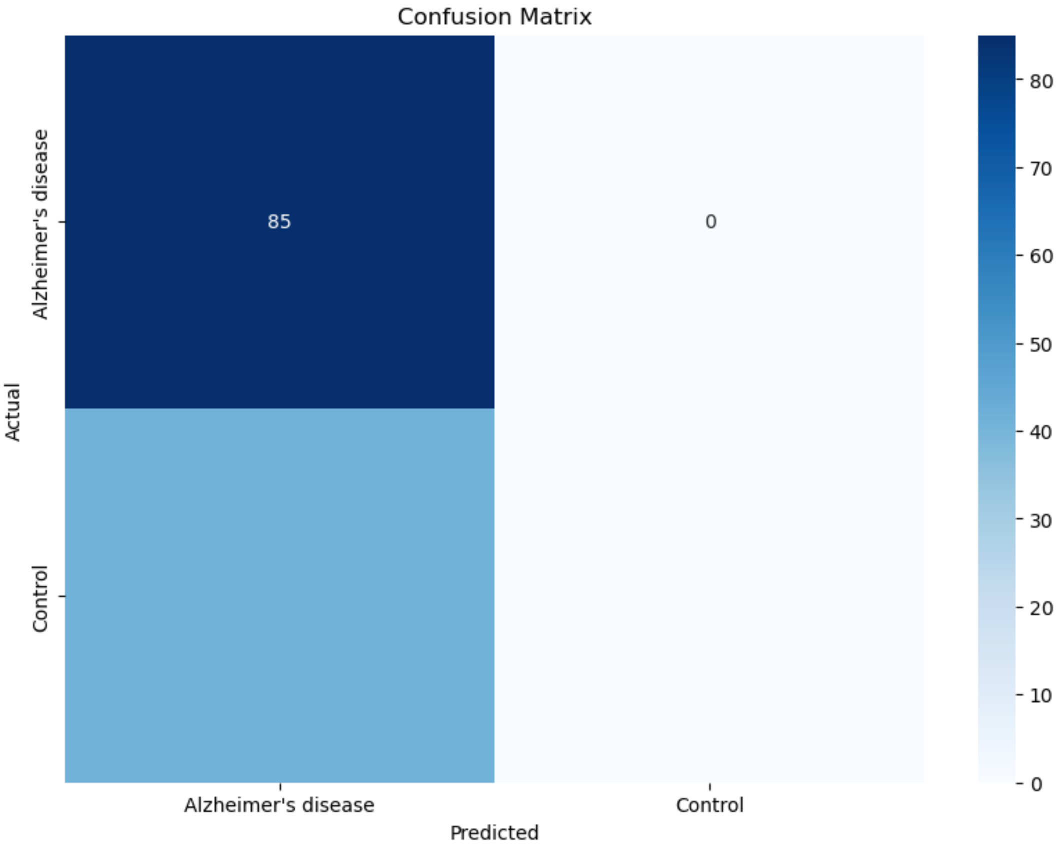


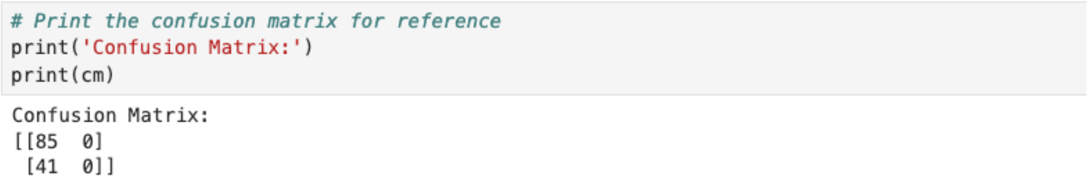
Figure 4.2 Feature Importances

The" Feature Importances" graph presents the relative significance of different features in each dataset. Some features, namely P2RX7, ALDH9A1, and GPS1, hold values of significance around 0.0100. Features such as MAL2 and CMTM5 show the lowest level of importance, just slightly above 0.0000. This visual representation helps in identifying key features that significantly impact the model's predictions, thereby assisting in feature selection and enhancing model performance.

4.3 Confusion Matrix

- True Positives (TP): 85 - These are Alzheimer's disease samples correctly classified as Alzheimer's disease.
- False Negatives (FN): 0 - These are Alzheimer's disease samples incorrectly classified as control.
- False Positives (FP): 41 - These are control samples incorrectly classified as Alzheimer's disease.
- True Negatives (TN): 0 - These are control samples correctly classified as control.





```
# Print the confusion matrix for reference
print('Confusion Matrix:')
print(cm)
```

```
Confusion Matrix:
[[85  0]
 [41  0]]
```

Figure 4.3 Confusion Matrix

Accuracy:

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{Total Samples}} = \frac{85 + 0}{85 + 0 + 41 + 0} = \frac{85}{126} \approx 0.6746$$

Precision (Positive Predictive Value):

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{85}{85 + 41} = \frac{85}{126} \approx 0.6746$$

Recall (Sensitivity or True Positive Rate):

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{85}{85 + 0} = 1.0$$

F1 Score:

$$\text{F1 Score} = \frac{2 \cdot (\text{Precision} \cdot \text{Recall})}{\text{Precision} + \text{Recall}} = \frac{2 \cdot (0.6746 \cdot 1.0)}{0.6746 + 1.0} \approx 0.8052$$

The model's accuracy is 67.4%, indicating that it correctly classifies 67.4% of all samples. The precision for Alzheimer's disease is also 67.4%, meaning that 67.4% of the samples predicted as Alzheimer's disease are truly Alzheimer's disease. The recall for Alzheimer's disease is 1.0, indicating that the model correctly identifies all Alzheimer's disease samples. The F1 score, which balances precision and recall, is 0.805, showing a reasonably good balance. However, the specificity is 0.0, meaning the model fails to correctly identify any control samples.

4.3.1 Receiver Operating Characteristic (ROC) curve

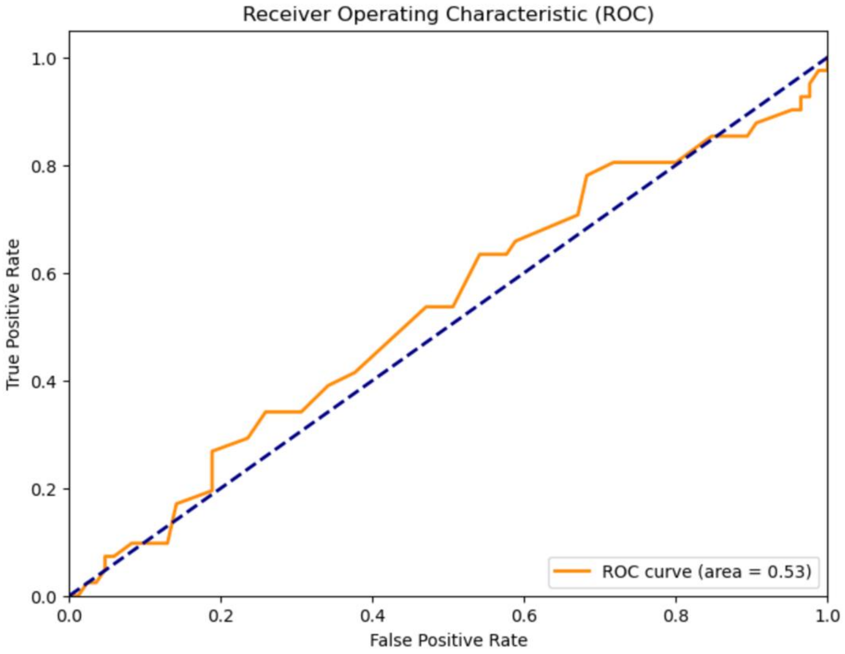


Figure 4.4 Receiver Operating Characteristic (ROC) curve

The ROC curve for the current model shows an AUC of 0.53, which is slightly above random guessing but still signifies poor classification performance. The model's ability to differentiate between classes is only marginally better than random, as indicated by the curve path closely following the diagonal line. This suggests that the model's predictions are not very accurate and significant improvements are needed to better distinguish between Alzheimer's disease and Control cases. In conclusion, the ROC curve highlights the suboptimal performance of the current model, emphasizing the necessity for further tuning or the inclusion of additional features to boost its predictive capabilities.

4.4 Pairplot

The scatterplots demonstrate the relationships between pairs of features. A noticeable separation of points for Diagnosis 0 and Diagnosis 1 suggests that these features are effective in distinguishing between the two diagnoses. Some scatterplots show significant separation between the two diagnoses, indicating the usefulness of these feature pairs for classification. The histograms on the diagonal present the distribution of individual features. Clear peaks for each category indicate that the feature is useful for classification. The histograms show distinct distributions for certain features across each diagnosis, which is beneficial for classification accuracy. In conclusion, the pairplot suggests that specific features are effective in distinguishing between the two diagnoses, but further analysis is needed to confirm the model's accuracy.

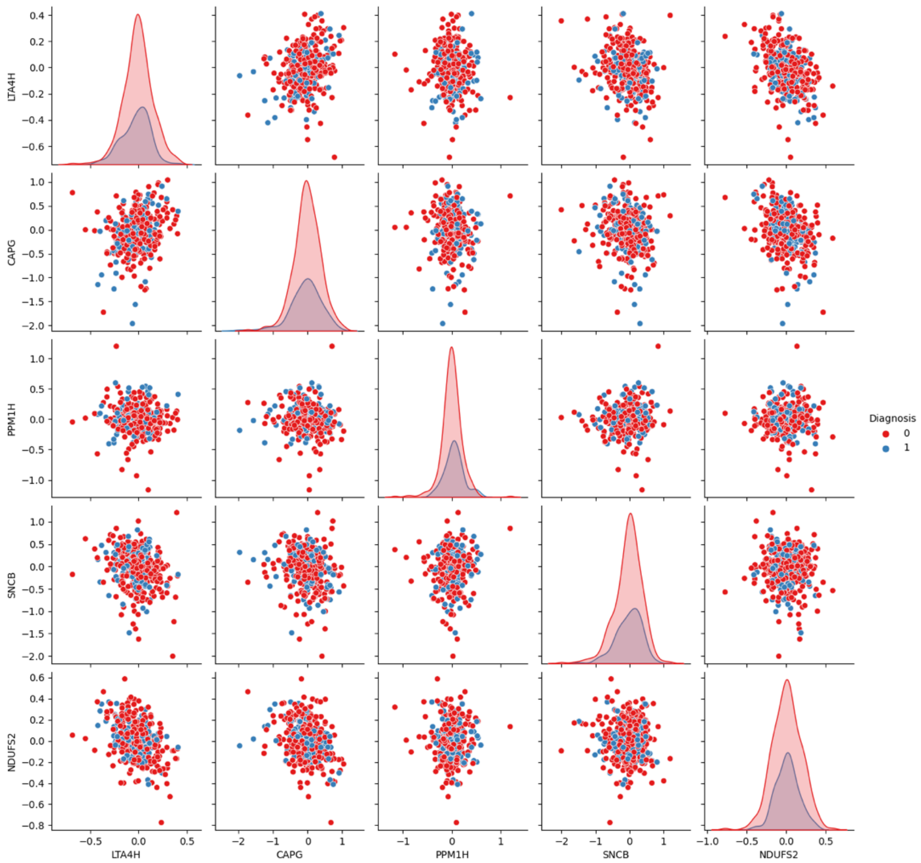


Figure 4.5 Pairplot

4.6 t-SNE

An ideal t-SNE (t-Distributed Stochastic Neighbor Embedding) outcome would display distinct clusters of data points corresponding to various categories or diagnoses with minimal overlap. This signifies that the algorithm has successfully captured the dataset's variability and can differentiate between different classes based on their similarities.

In given plot Red dots represent 'Diagnosis 0' while blue dots represent 'Diagnosis 1'. Some clustering is evident, with red points mainly in one area and blue points in another, yet there is noticeable overlap present.

- The t-SNE outcome implies that although there is some differentiation between the two diagnoses based on protein data, it is not very pronounced. This indicates that the model's capacity to distinguish between the diagnoses is moderate but not highly precise.
- Overall, the t-SNE visualization indicates that the current model capture some of the variability between the two diagnoses, but further adjustments or additional features may be necessary to enhance the differentiation and accuracy.

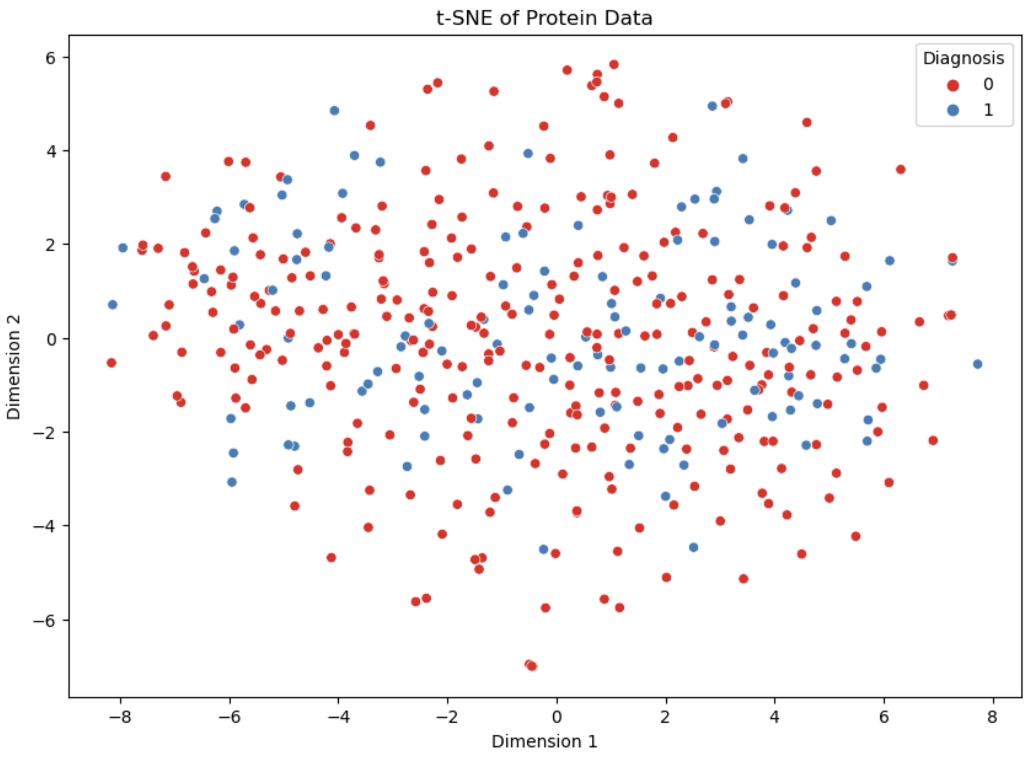


Figure 4.6 t-SNE

4.7 PCA

An ideal result from PCA would show clear separation between different categories (e.g., Diagnosis 0 and Diagnosis 1) on the scatter plot, indicating that the principal components successfully capture the variability that distinguishes these categories.

- Red dots represent 'Diagnosis 0' while blue dots represent 'Diagnosis 1'. The data points are intermingled without a clear distinction between the two diagnoses.

- The significant overlap between red and blue points suggests that the first two principal components do not effectively differentiate between the two diagnoses.

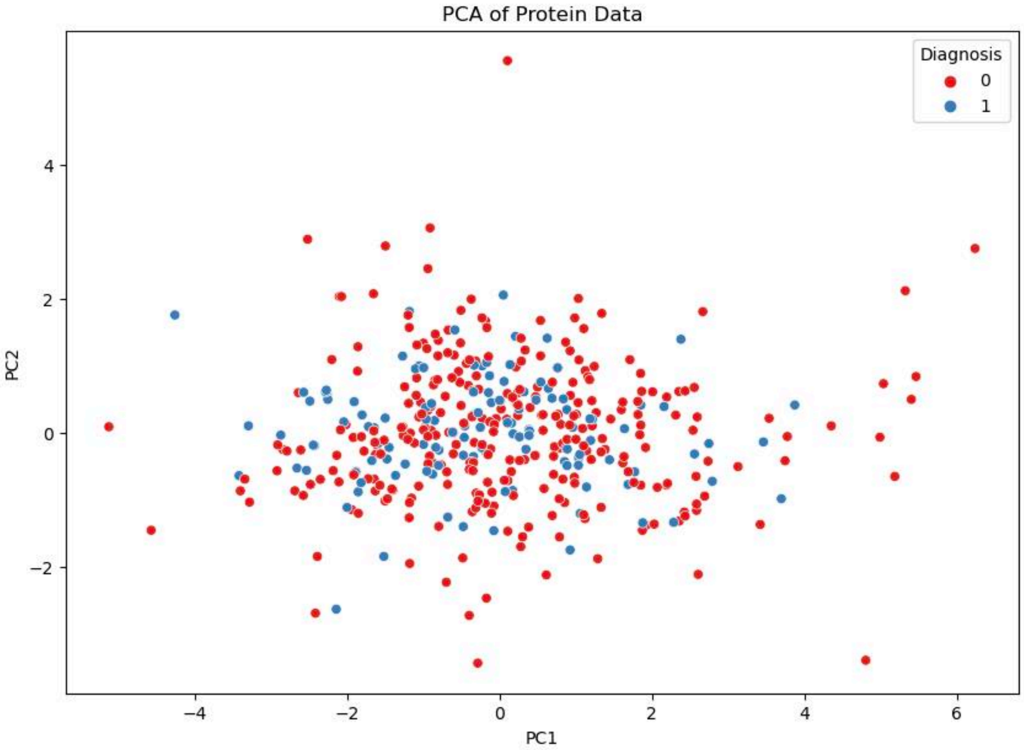


Figure 4.7 PCA

The PCA result indicates that the model's ability to separate the diagnoses based on these components is limited. This implies that either additional components are needed or PCA may not be the most suitable approach for this dataset. Overall, the PCA plot indicates that the current model does not adequately distinguish between the two diagnoses, indicating the need for further investigation or alternative methods.

4.8 Correlation Heatmap

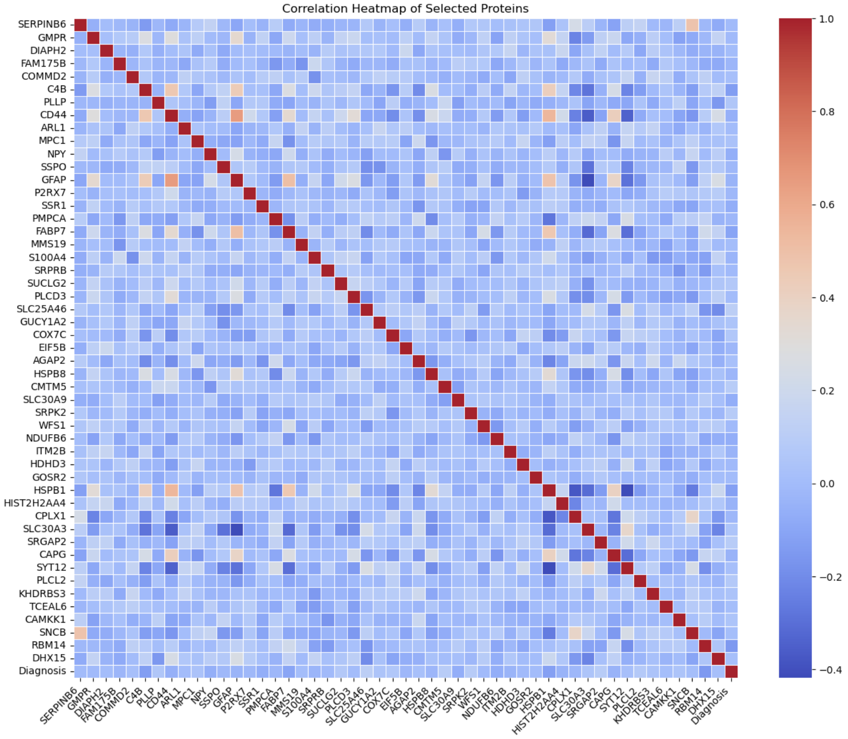


Figure 4.8 Correlation Heatmap

The correlation heatmap provides insights into the relationships between different proteins.

- Red color Signifies a strong positive correlation (near 1). Blue signifies a strong negative correlation (near -1). and White/Light Shades indicate weak or no correlation (near 0). Diagonal Line represents perfect positive correlations (each protein compared with itself). Areas with similar colors suggest groups of proteins with similar correlation trends.

- Strong Positive Correlations: Proteins like SERPINB6 and GMPR exhibit strong positive correlations with several other proteins.

- Strong Negative Correlations: Proteins like SLC30A9 and SRPK2 display strong negative correlations with other proteins.

- Weak Correlations: Numerous proteins show weak correlations, suggesting they have minimal impact on each other.

4.9 Heatmap of Alzheimer\'s Disease Samples vs. Control Samples

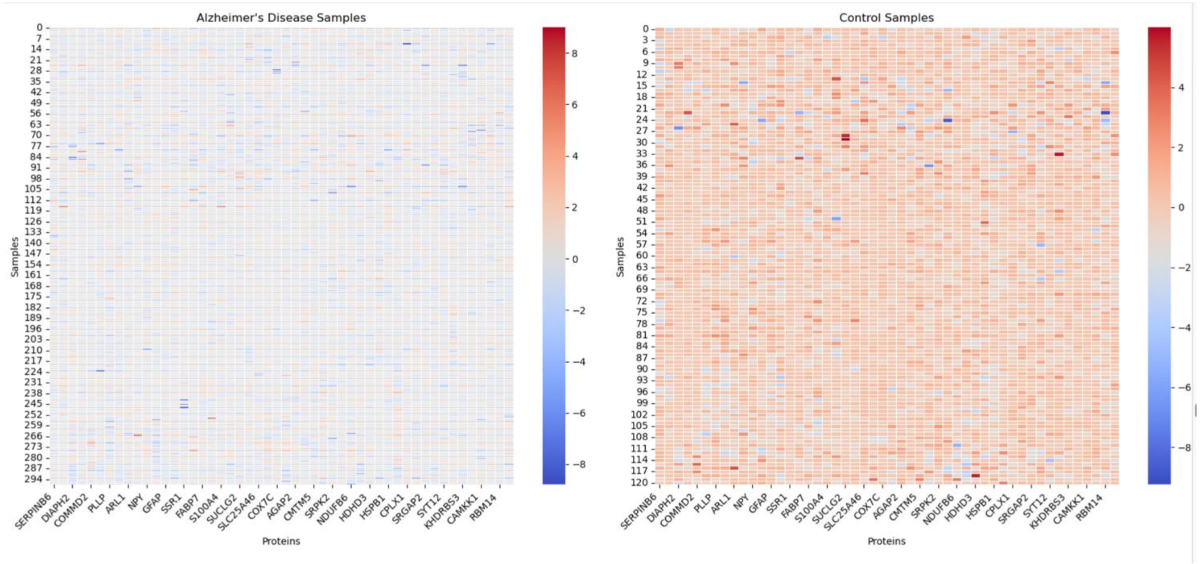


Figure 4.9 Heatmap of Alzheimer\'s Disease Samples vs. Control Samples

The heat maps illustrate the comparison of protein expression levels between Alzheimer's Disease samples and Control samples. Light Blue color indicates low expression/activity levels and Dark red color represents high expression/activity levels.

Alzheimer's Disease Samples: Predominantly light blue color Suggests generally low protein expression levels. A few darker spots suggest some proteins exhibit higher expression in specific samples.

Control Samples: More uniform warm colors signify higher overall protein expression levels. While Red and Orange indicate higher activity across most proteins.

Key Observations:

- Lower Expression in Alzheimer's: Alzheimer's samples display lower protein activity compared to controls.
- Potential Biomarkers: Variations in expression levels could aid in identifying biomarkers for Alzheimer's disease.

4.10 Heatmap of Protein Expression in different age group

A) **Protein expression in young age people:** Low expression is represented by purple, while higher expression is shown by yellow to green shades. The bottom of the heatmap displays different young age proteins like "ATP5F1B," "UQCRC2," and "NDUFS3."

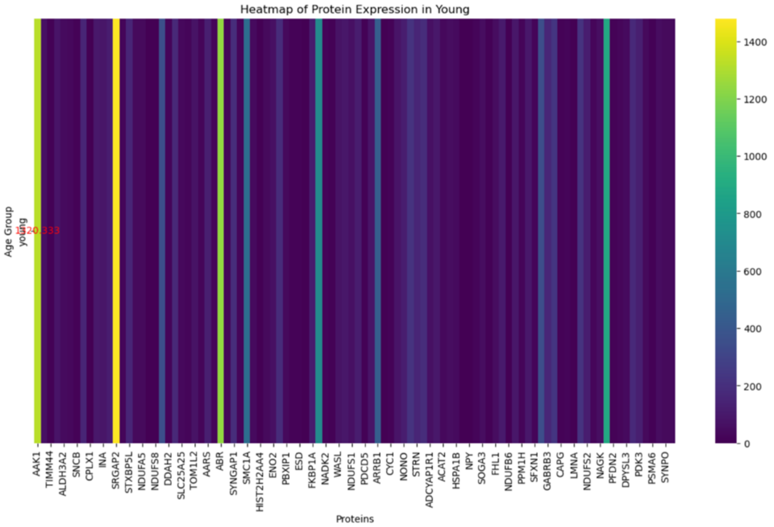


Figure 4.10 (A)Protein expression across young age people:

protein expression in middle-aged people: Low expression is represented by dark purple, while high expression is represented by yellow. Low expression is represented by purple, while higher expression is shown by yellow to green shades. The bottom of the heatmap displays different proteins like" ATP5F1B,"" UQCRC2," and "NDUFS3".

B) **Protein expression in middle-aged people:** Low expression is represented by dark purple, while high expression is represented by yellow. Different proteins like "IGF1," "EGFR," and "INSR" are listed at the bottom. By visually comparing protein expression across samples, the heatmap highlights higher expression levels in yellow.

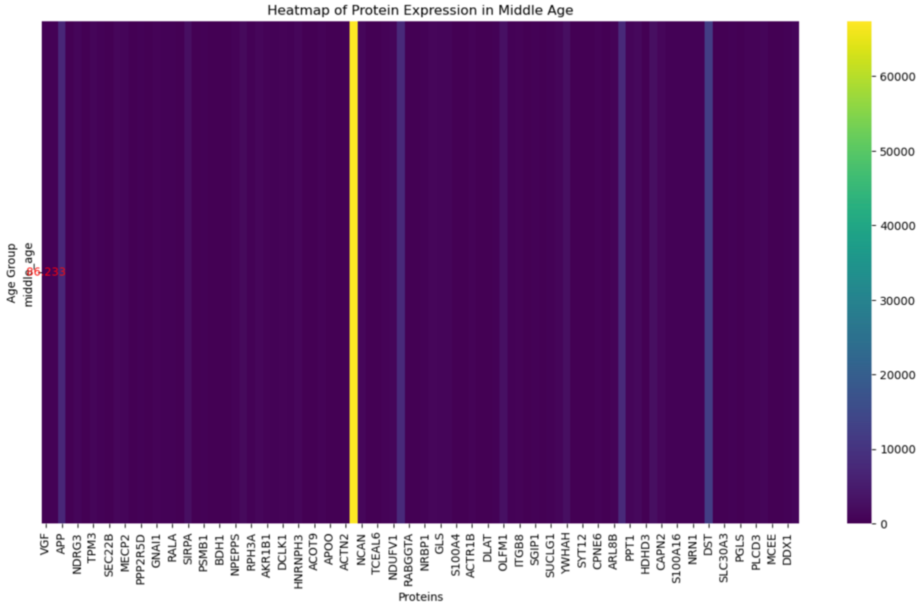


Figure 4.10 (B)Protein expression in middle-aged people:

C) **Protein expression in old age people**: Low expression is represented by dark purple, while high expression is represented by yellow. Different proteins like APP, APOE, and BACE1 are listed at the bottom. By visually comparing protein expression across samples, the heatmap

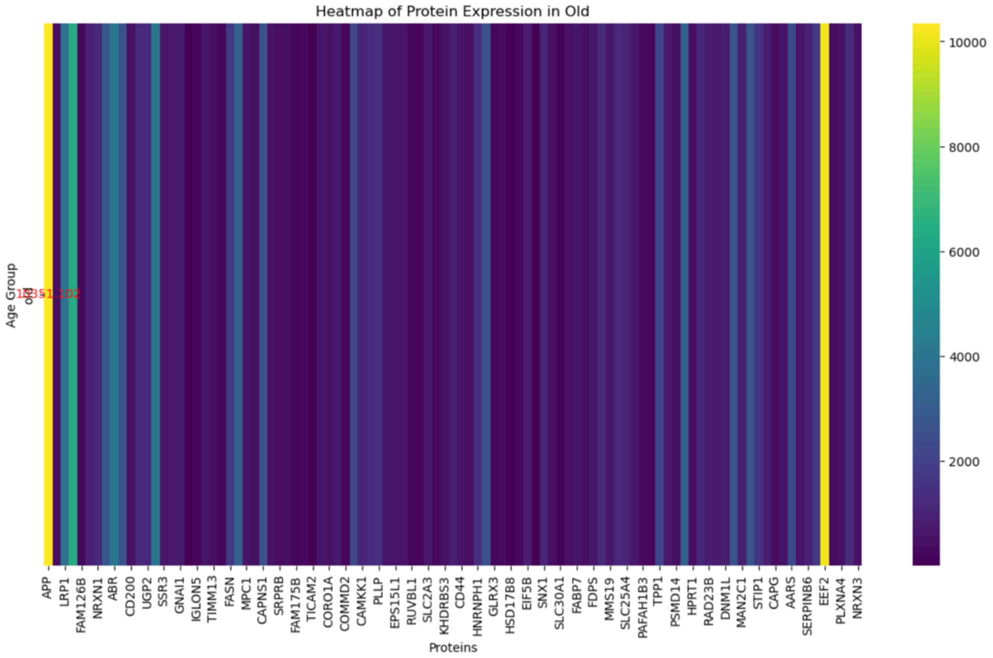


Figure 4.10 (B)Protein expression in old-aged people:

protein expression levels in young individuals: The violin plot displays the protein expression levels in young individuals. Each violin represents the distribution of expression levels for a specific protein, with wider sections indicating a higher frequency of data points at that expression level. Various proteins like ”RAK1,” ”RPL37A,” and ”TIKM-+4” are listed along the x-axis. The y-axis shows the range of expression levels, with some proteins having a wider range of expressions. This visualization allows for comparing the expression of different proteins in young individuals.

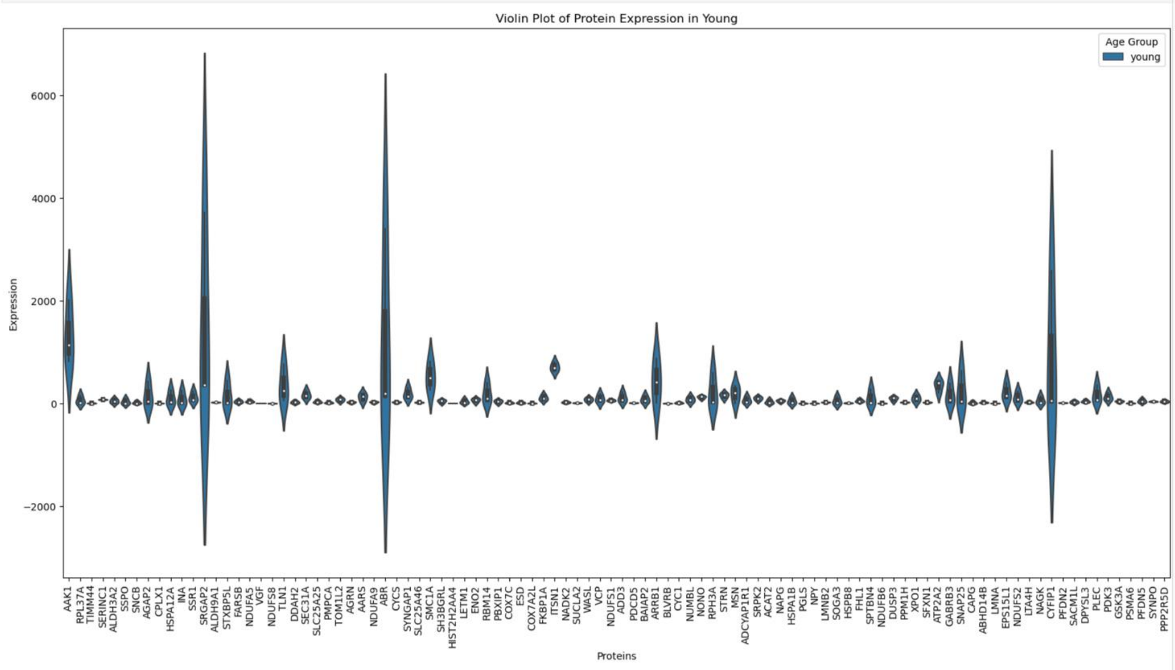


Figure 4.11 (a) violin plot of protein expression in young age people

A) **Protein expression levels in middle-aged individuals:** Horizontal Axis enlists various proteins. Vertical axis represents expression levels, spanning from around -200,000 to 400,000. Every point showcases the expression level of a protein. Most points are closely grouped near the horizontal line, indicating minimal variability. However, there is a distinct spike indicating substantial variability for a particular protein.

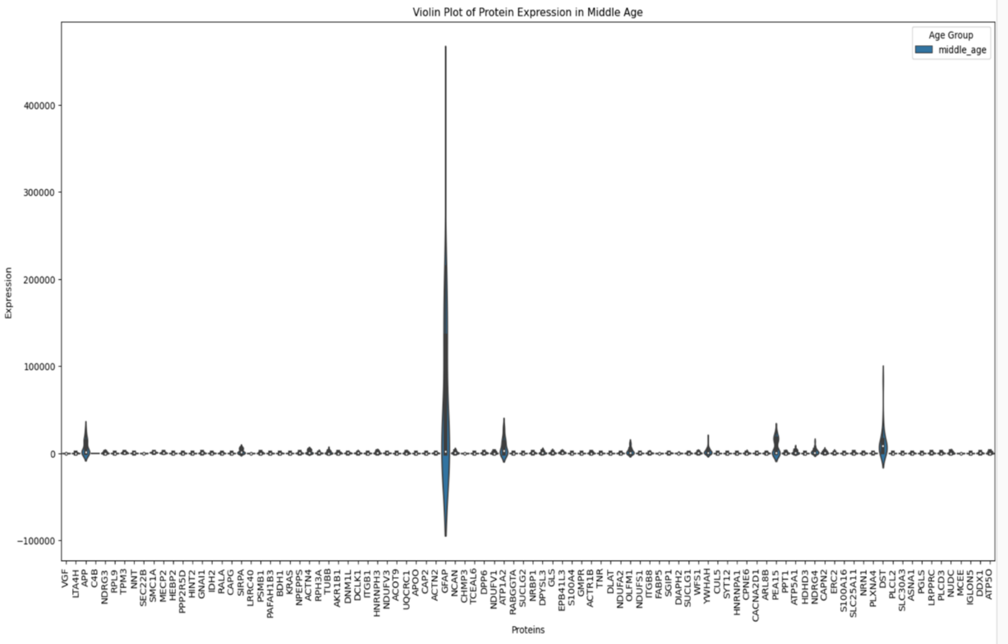


Figure 4.11 (b) violin plot of protein expression in middle-aged people

B) protein expression levels in old age individuals: The violin plot offers a comprehensive look at protein expression levels in older individuals. The Y-Axis displays protein expression levels from 0 to 50,000. X-Axis Features a variety of proteins like APP, ACAT2, and DDX17. The width of each ”violin” represents the density of data points at different expression levels. Wider sections indicate more data points. The shape of each violin illustrates the distribution of expression levels for each protein. A wider middle section suggests most data points cluster around that level. Proteins like APP and ACAT2 exhibit a broad distribution at higher expression levels, indicating increased activity in older individuals. Conversely, proteins like DDX17 show narrower distributions at lower expression levels, suggesting lower activity.

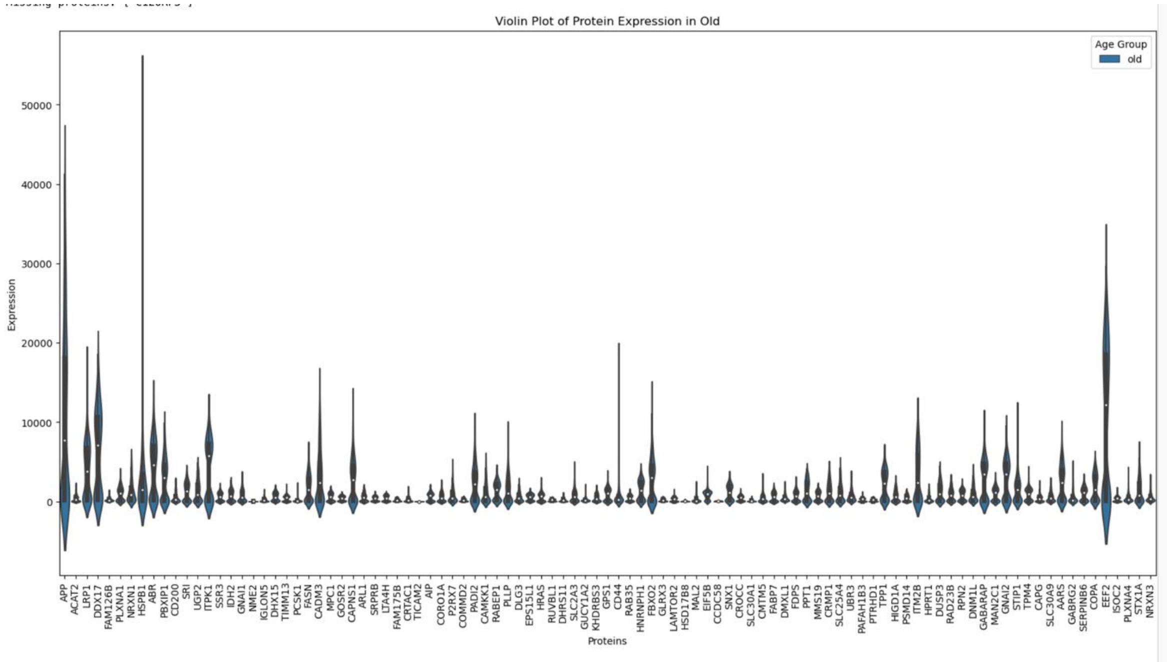


Figure 4.11 (c) violin plot of protein expression in old age people

5 Discussion

- **Confusion matrix:** The model effectively detects Alzheimer's disease but struggles with identifying control samples, leading to many false positives. This highlights the need for better balancing and fine-tuning to accurately differentiate between the two groups.

- **ROC curve:** for this model has an AUC of 0.55, indicating it is just a bit better than random guessing. The dashed blue line, which shows random chance with an AUC of 0.5, acts as a reference point.
- To make the model better at distinguishing between outcomes, more improvements are needed .
- SVM is the best model among the three, considering all the parameters. While Decision Tree is very close in performance, Random Forest slightly edges it out due to a marginally better R-squared value and generally better handling of variance.

- **Highly Expressed protein in young age group: AAK1, SPGAP2, ABR**
- **Highly Expressed protein in middle age group: ACTN2 and NCAN**

- **Highly Expressed protein in old age group: APP and EEF2**

6 Conclusion

- As people grow older, the levels of protein expression differ across various graphs, indicating a relationship between biological processes or protein markers and the aging process.
- Most of these proteins are associated with Biological Processes like cell death, oxidative stress, and immune system function, suggesting that aging is connected to increased cellular stress and immune activity.
- Understanding these changes can help identify potential targets for interventions designed to mitigate age-related decline.
- Trained models improve diagnostic precision, helping to identify biomarkers for early intervention.
- Metrics like accuracy, precision, recall, F1 score, and AUC-ROC curves assess model performance, guiding the selection of the best predictive model.
- In conclusion, Supervised learning enables researchers to gain deeper insights into Alzheimer's disease mechanisms, improve diagnostics, and develop personalized treatments.

7 References

1. Shokhirev MN, Johnson AA. An integrative machine-learning meta-analysis of high-throughput omics data identifies age-specific hallmarks of Alzheimer's disease. *Ageing Research Reviews*. 2022;81:101721. doi:https://doi.org/10.1016/j.arr.2022.101721
2. Scikitlearn.org. Published 2022. https://scikitlearn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html
3. GeeksforGeeks. Understanding Logistic Regression. GeeksforGeeks. Published May 9, 2017 https://www.geeksforgeeks.org/understanding-logistic-regression
4. https://scikitearn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.htm
5. scikit-learn. 1.10. Decision Trees — scikit-learn 0.22 documentation. Scikit-learn.org. Published 2009. https://scikit-learn.org/stable/modules/tree.htmlaaa

A. Appendix A: Additional Figures

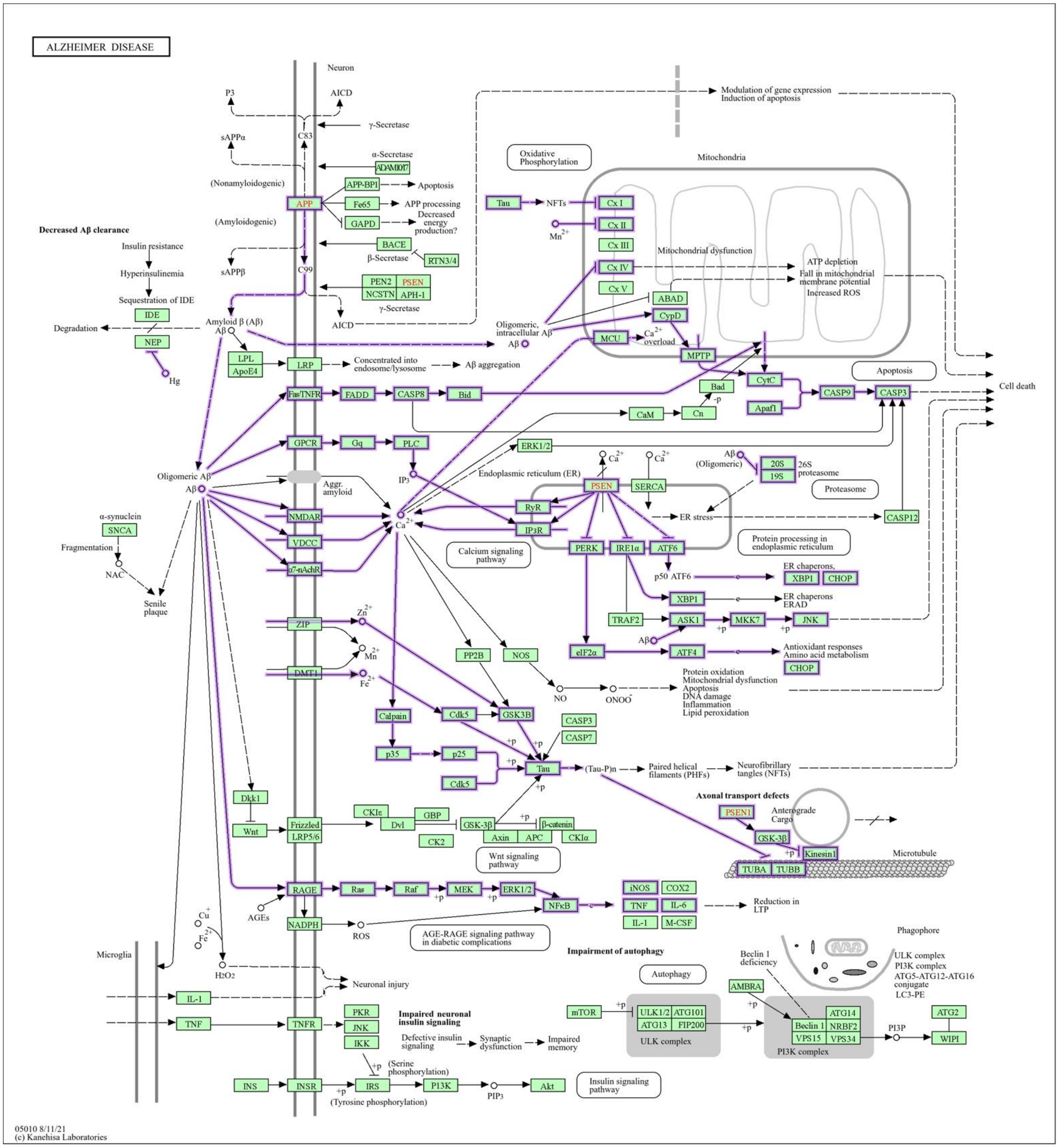


Figure 1. Alzheimer disease – Reference KEGG pathway

https://www.kegg.jp/kegg-bin/show_pathway?hsa05010

The green boxes highlight lower activity or levels of certain molecules that might play a role in the disease's progression. On the other hand, purple boxes show increased activity or levels, which could be detrimental and lead to symptoms associated with Alzheimer’s disease.

Pathway Insights: Understanding these pathways is key to pinpointing potential targets for treatment and understanding how Alzheimer’s disease works. From our project analysis, we discovered that proteins are expressed at higher levels across various age groups. Notably, one protein called **APP** is particularly elevated in older individuals, and it’s linked to the pathways that contribute to cell death in Alzheimer’s disease. We can conduct similar analyses on proteins from different age groups for further exploration.

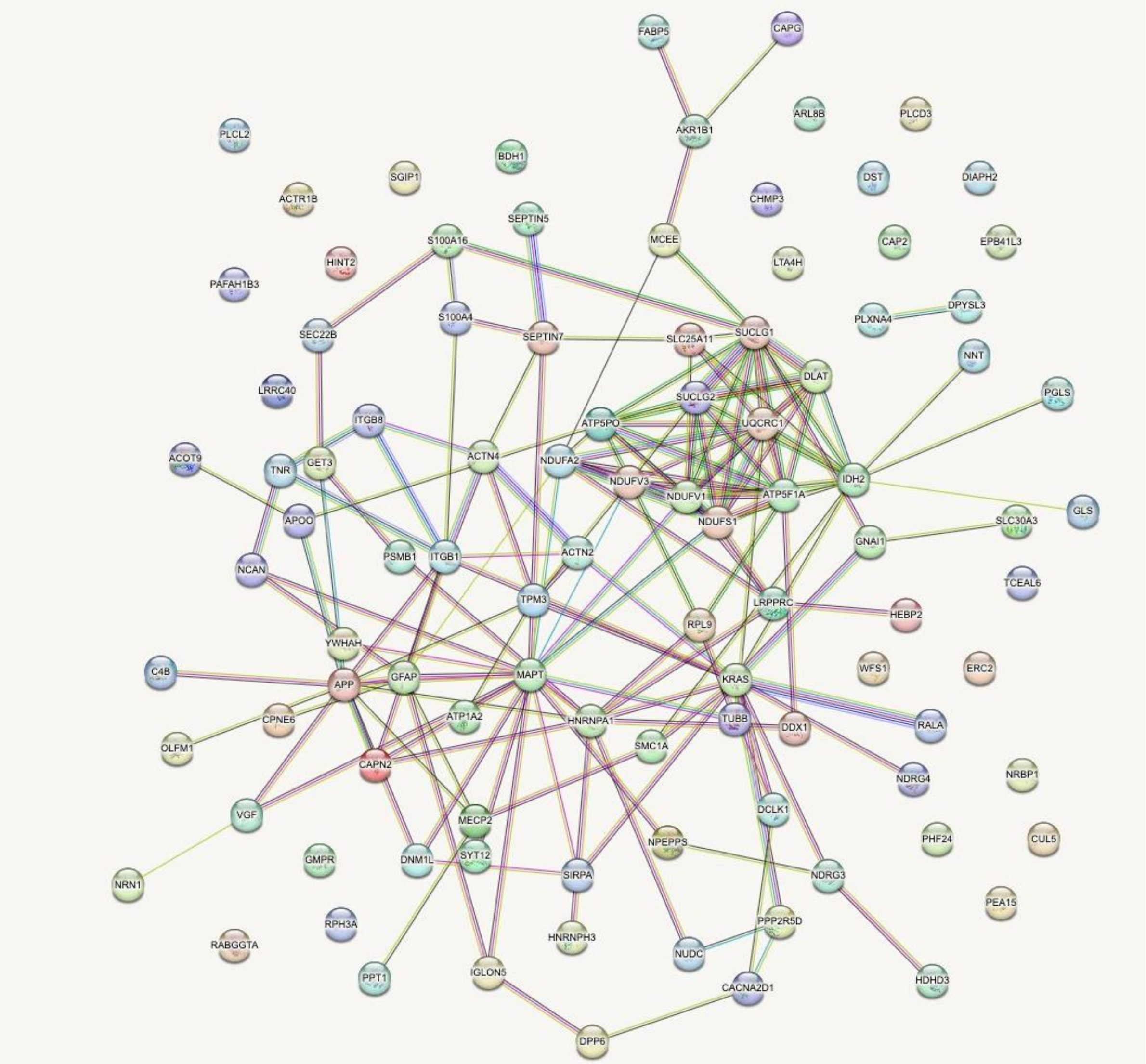


Figure 2. Functional Enrichment Analysis

https://stringdb.org/cgi/network?taskId=bS0kPaTv7QXy&sessionId=boflflkQlllN