Project Data Analysis

Replication of Research paper entitled *"Molecular markers of early Parkinson's disease based on gene expression in blood"*

PREPARED FOR

Dr. Suhila Sawesi, Ph.D., MPharm, BPharm

sawesis@gvsu.edu

PREPARED BY

Rashmita Vaggu

Meher Nivedita Avdut

Muttaki I. Bismoy

Paper Reference

Scherzer, C. R., Eklund, A. C., Morse, L. J., Liao, Z., Locascio, J. J., Fefer, D., Schwarzschild, M. A., Schlossmacher, M. G., Hauser, M. A., Vance, J. M., Sudarsky, L. R., Standaert, D. G., Growdon, J. H., Jensen, R. V., & Gullans, S. R. (2007) Molecular markers of early Parkinson's disease based on gene expression in the blood *Proceedings of the National Academy of Sciences of the United States of America*, *104*(3), 955–960. https://doi.org/10.1073/pnas.0610204104

Dataset Link

Link of the dataset that this paper used can be found here

Analysis Portion

In our analysis, we aimed to replicate the heatmaps of gene expression data, albeit with certain limitations. Due to constraints related to computational power, we were only able to utilize 53 genes out of the total 105 available in the dataset. This reduction in the number of genes analyzed, while regrettable, was necessary to ensure the feasibility of our computational efforts.

Partial Analysis and Explanation

In the course of our replication, we set out to replicate not only the heatmaps but also the ROC curves and other relevant graphs mentioned in the paper. The paper lacked clarity and had metadata that was not mentioned in the website which presented its own set of challenges. Nevertheless, we were fortunate to possess a vast dataset containing gene expression values. Based on this useful resource, our main goal

was to make heatmaps from the gene data and identify the marker genes, which would help us fully understand the underlying gene expression patterns.

For our second aim, we considered the importance of the eight marker genes and the role they play in determining PD. Therefore, we used Excel to preprocess and make the data better for analysis of these marker genes using Python. We were successful in identifying these marker genes. But given the quantity of this huge dataset and the limited computational power, we were unable to visualize the importance of these marker genes.

Despite the limitations we encountered, the extensive gene dataset at our disposal allowed us to conduct a robust analysis that will undoubtedly provide valuable insights into gene expression patterns and their potential implications. The new information, along with the heatmaps that have been replicated and the importance of marker genes, will help us understand the topic better and make our research more complete.

Experiment and Statistical Model Description

Experiment Description:

The original study aimed to identify molecular markers of early Parkinson's disease using gene expression data obtained from blood samples. The experiment involved collecting blood samples from individuals diagnosed with Parkinson's disease at an early stage and comparing them to samples from healthy control subjects. Key steps in the experiment likely included:

Sample Collection: Blood samples were collected from both early-stage Parkinson's disease patients and healthy control subjects. These samples served as the primary source of gene expression data.

Gene Expression Profiling: To find out the gene expression profiles of the blood samples that were taken, advanced molecular biology techniques were used. These included microarray analysis and RNA sequencing. This step allowed researchers to measure the expression levels of a wide range of genes in each sample.

Data Preprocessing: The raw gene expression data were likely subjected to preprocessing steps to remove noise, normalize the data, and ensure data quality. Preprocessing steps are essential for obtaining reliable results from gene expression studies.

Differential Expression Analysis: Statistical methods were applied to identify genes that showed differential expression between the Parkinson's disease group and the control group. Genes that exhibited significant differences in expression levels were considered potential molecular markers of the disease.

Statistical Model Description:

The statistical analysis in the original study likely involved the following components:

T-Tests or ANOVA: To identify differentially expressed genes, standard statistical tests such as t-tests or analysis of variance (ANOVA) may have been used. These tests help determine if the expression levels of individual genes significantly differ between the Parkinson's disease group and the control group.

Multiple Testing Correction: Since researchers were looking at a lot of genes at once, they probably used multiple testing correction methods, such as the Bonferroni correction or the False Discovery Rate (FDR) correction, to account for the higher chance of false positives.

Machine Learning Models: In addition to traditional statistical tests, machine learning models may have been employed to develop predictive models or to identify patterns in the gene expression data. Techniques such as logistic regression, random forests, or support vector machines could have been utilized for this purpose.

Cross-validation: To assess the performance and generalizability of the statistical models, cross-validation techniques may have been employed. This involves splitting the data into training and testing sets to evaluate the model's ability to predict Parkinson's disease based on gene expression patterns.

Overall, the original study probably used a mix of statistical and computer methods to find molecular signs of early Parkinson's disease by looking at gene expression data from blood samples. These markers may have the potential to assist in early diagnosis or provide insights into the underlying mechanisms of the disease.

Tests Conducted

T-test

For identifying the marker genes, we used the popular t-test in Python. We aimed at identifying genetic expression differences between individuals with Parkinson's disease (PD) and healthy controls. t-tests are used to compare the levels of gene expression for each gene between the PD and control groups. This is what the code does for the most part. For each gene, the code extracts the expression values for PD patients and control subjects. It then applies the ttest_ind function from the scipy.stats package to calculate the t-statistic and corresponding p-value for the difference in gene expression between the two groups.

The t-test formula is as follows:

$t = (x1 - x2) / sqrt(s1^2/n1 + s2^2/n2)$

where:

t is the t-statistic

- x1 is the mean of the first group
- x2 is the mean of the second group
- s1 is the standard deviation of the first group
- n1 is the number of observations in the first group
- s2 is the standard deviation of the second group
- n2 is the number of observations in the second group

The p-values are stored in a list for further analysis. The code concludes by aligning the outcome variable (y) with the cleaned data and converting it to binary format. This preparation makes the data ready for further analysis using machine learning algorithms like logistic regression.

Correcting for Multiple Testing

The code begins by correcting for multiple testing using the Benjamini-Hochberg (BH) procedure, implemented in the multipletests function from the statsmodels.stats.multitest package. This step is crucial as performing multiple t-tests increases the likelihood of false positives, where genes are deemed statistically significant due to chance rather than actual differences in gene expression. The BH procedure manages the false discovery rate (FDR), which makes sure that the genes that are statistically significant are not likely to be false positives.

Test Results

Interpreting T-test and Multiple Testing Results:

A smaller p-value indicates stronger evidence against the null hypothesis, suggesting that the observed difference in gene expression is likely due to a real effect rather than chance. A common p-value threshold for statistical significance is 0.05. A p-value below 0.05 means that there is a statistically significant difference in gene expression between the two groups at the 95% confidence level.

After adjusting the p-values for multiple tests, the code selects the top 8 genes with the smallest adjusted p-values. This selection process prioritizes genes with the strongest evidence of differential expression between Parkinson's disease patients and healthy controls. These genes are considered the most promising candidates for further investigation and potential use in diagnostic or prognostic models. The identifier names corresponding to these are:

CEACAM4, CLTB, FPR3, SYNCRIP, PI4KB, PGF, VDR, and ASB4.

Heatmap:

The visual representations presented seem to be associated with heatmaps and tables that aggregate the values of gene expression datasets. The images appear to be related to this information.

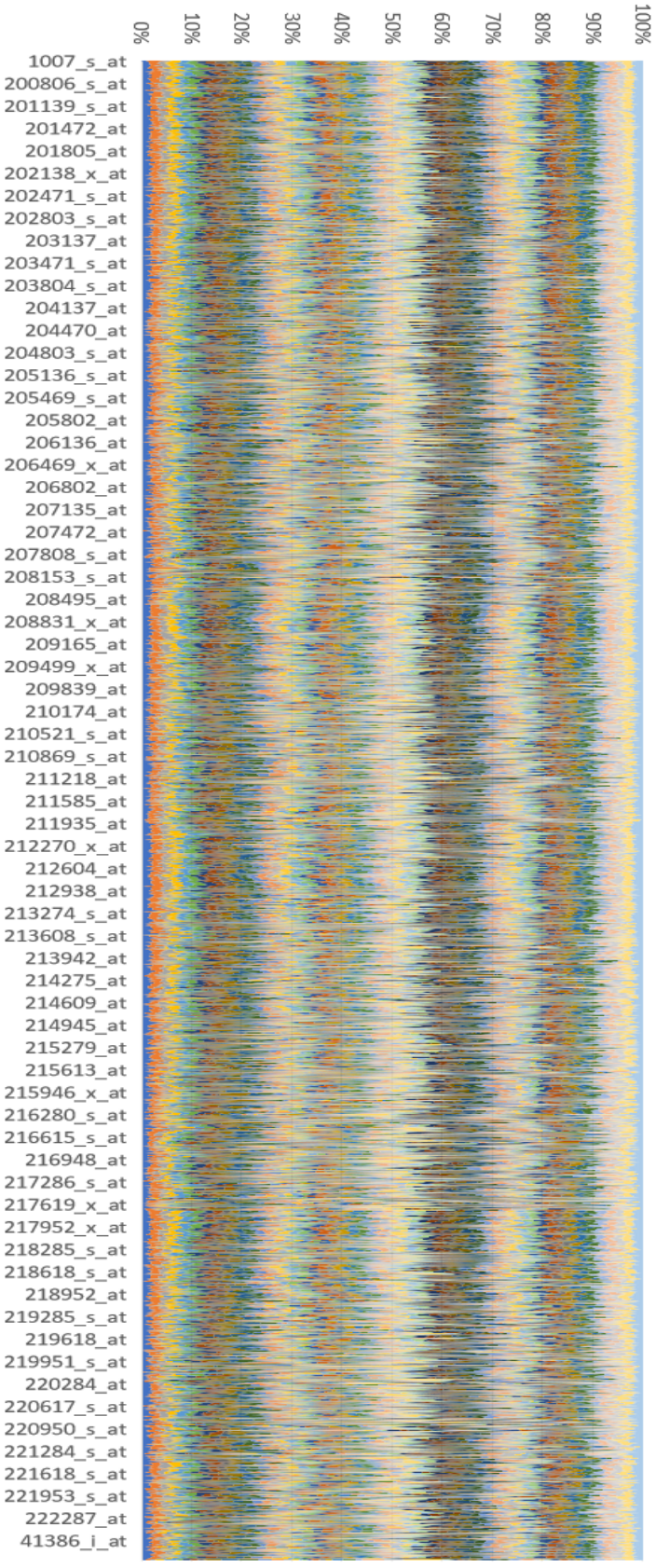
The first image is a **heatmap**, which is a data visualization technique used to represent the levels of gene expression across various conditions or samples. Here's a breakdown of what you're seeing:

Vertical Axis (Rows): Each row represents a different gene. Probe set IDs are unique numbers for certain sequences of DNA or RNA that are used to measure how much a gene is expressed. We use these IDs to talk about genes (for example, "200706_s_at").

Horizontal Axis (Columns): These likely represent different experimental samples or conditions, with the percentages possibly indicating different time points, treatment concentrations, or patient groups.

Color Scale: The colors in the heatmap correspond to the expression level of each gene in each sample or condition. Typically, colors towards one end of the spectrum (like red) indicate higher expression, and colors towards the other end (like blue) indicate lower expression. In this heatmap, it seems that a range of colors from blue, yellow, to red are used, which could represent a gradient from low to high expression levels.

The second image is a list of values, specifically the **sum of GSM values**. "GSM" stands for "Geo Sample," which is a term from the GEO (Gene Expression Omnibus) database that refers to individual samples for which gene expression is measured.



Values: Each value in the list represents the sum of the expression levels of all genes within a particular sample (GSM ID). This might be used to get an overall sense of the gene activity within each sample or to normalize the data.

Researchers can figure out which genes are differentially expressed under different conditions by using these kinds of visualizations and summaries along with statistical tests. This may help us understand diseases like Parkinson's, as the title of the paper suggests. The

heatmap provides a quick visual comparison across all genes and samples, while the summed values might be a step in preprocessing the data for further analysis.

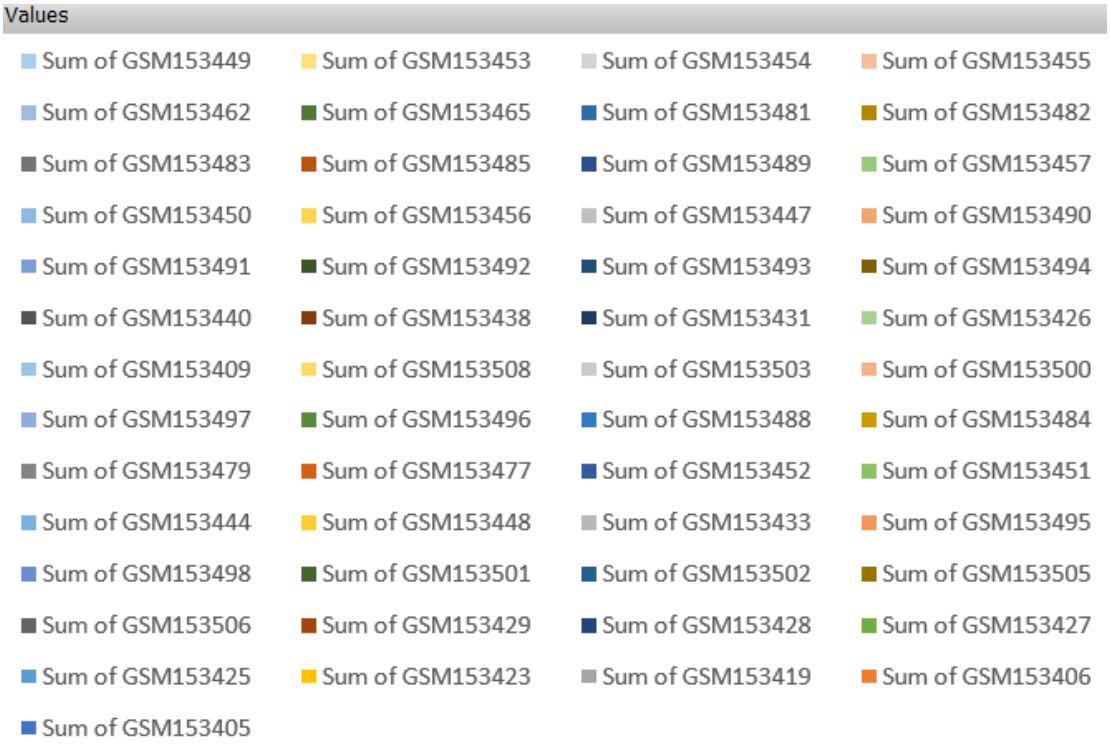


Fig. GSM Values selected for the heatmap

Comparison with the original results

Identifying marker genes:

Comparing the results of the marker genes obtained by the authors of the original paper to the marker genes we identified, there were only four that matched the genes from the paper. They are CEACAM4, FPR3, PGF, and VDR. This might be due to the difference in processes used for identifying the genes in the original paper vs. the replication paper. The original paper talked about how these marker genes were found by looking at changes in gene expression in the blood of people with early-stage Parkinson's disease. RNA from whole blood samples of Parkinson's patients and controls of the same age was analyzed using microarrays with 22,283 oligonucleotide probe sets. The genes were rank-ordered based on their correlation coefficient with Parkinson's disease. Which is more of a biological technique than a machine learning analysis.

In our replication, we used a t-test to find the p-values, adjusted them using multiple testing, and then prepared them for machine learning analysis. Therefore, we would like to highlight the observation that different processes may lead to some differences in the results.

Heatmap:

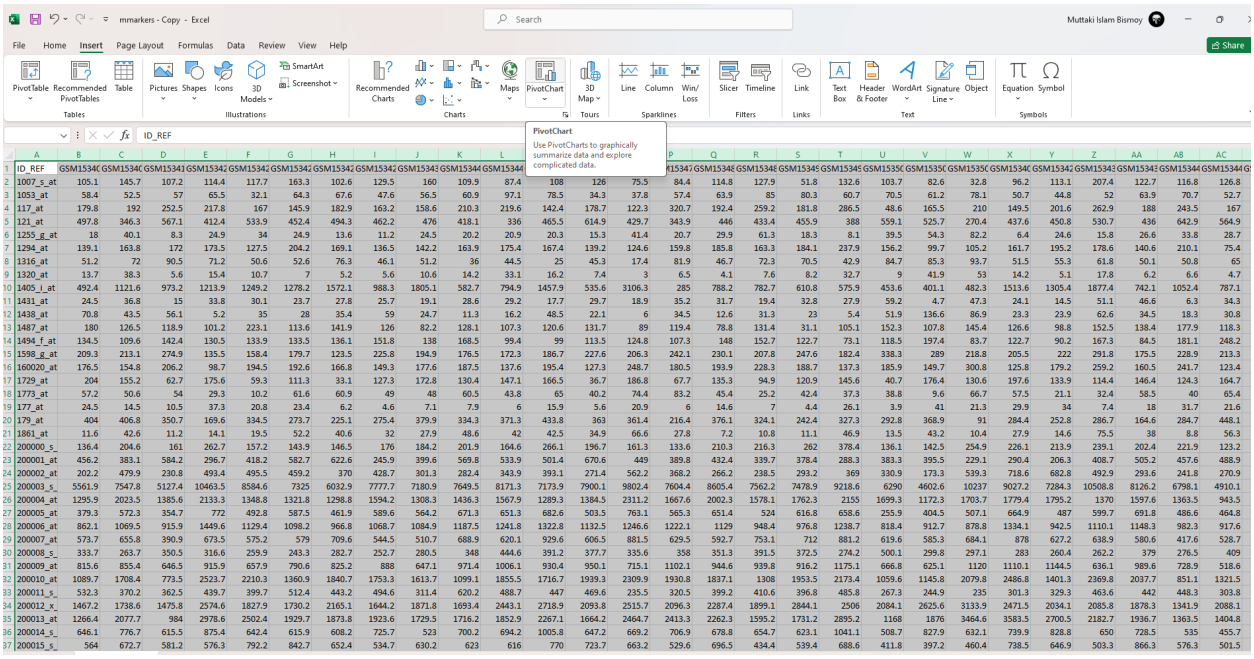
The heatmap that we generated were based on the tests conducted. This was an additional chart prepared to get an in-depth understanding of the data and gene expressions present in the dataset. As the dataset is large, we used 53 genes instead of 105. The heatmap gives us knowledge about the density of the gene values.

Code and Lab Report

Heatmap

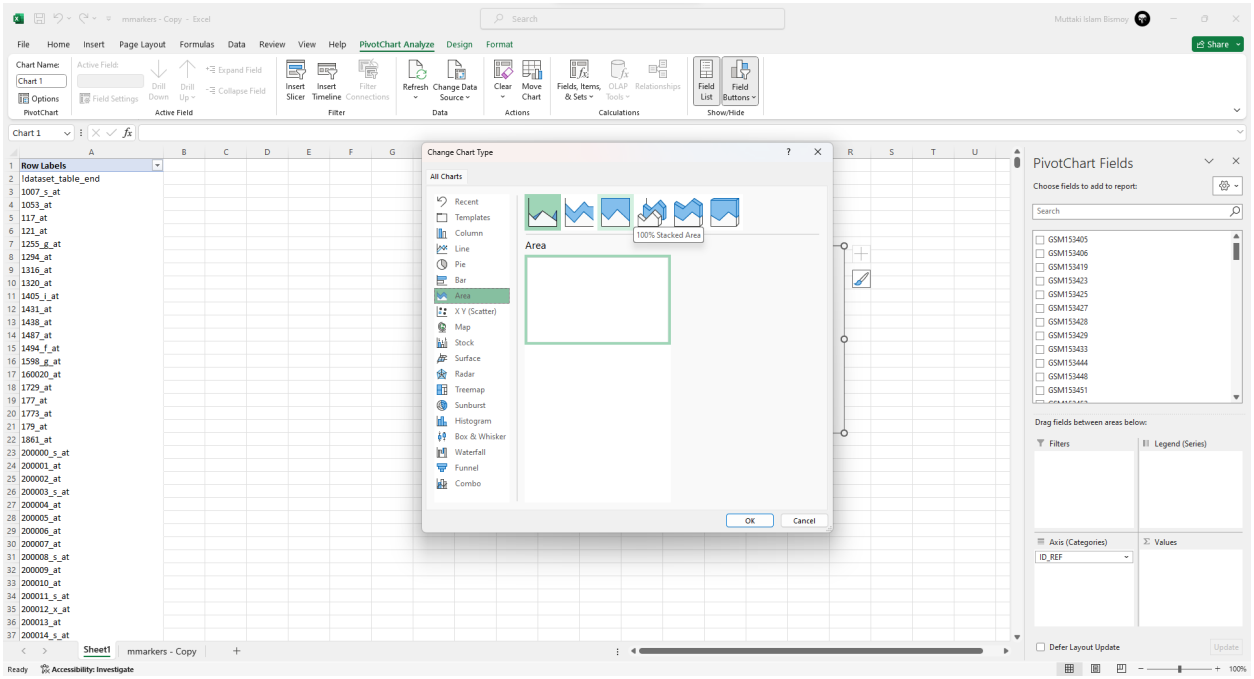
The images that have been uploaded show the process of creating a heatmap in Microsoft Excel from gene expression data. Here is a summary description of the steps taken based on what is visible in the screenshots:

Data Preparation: The raw data consisting of gene expression values (presumably from a microarray

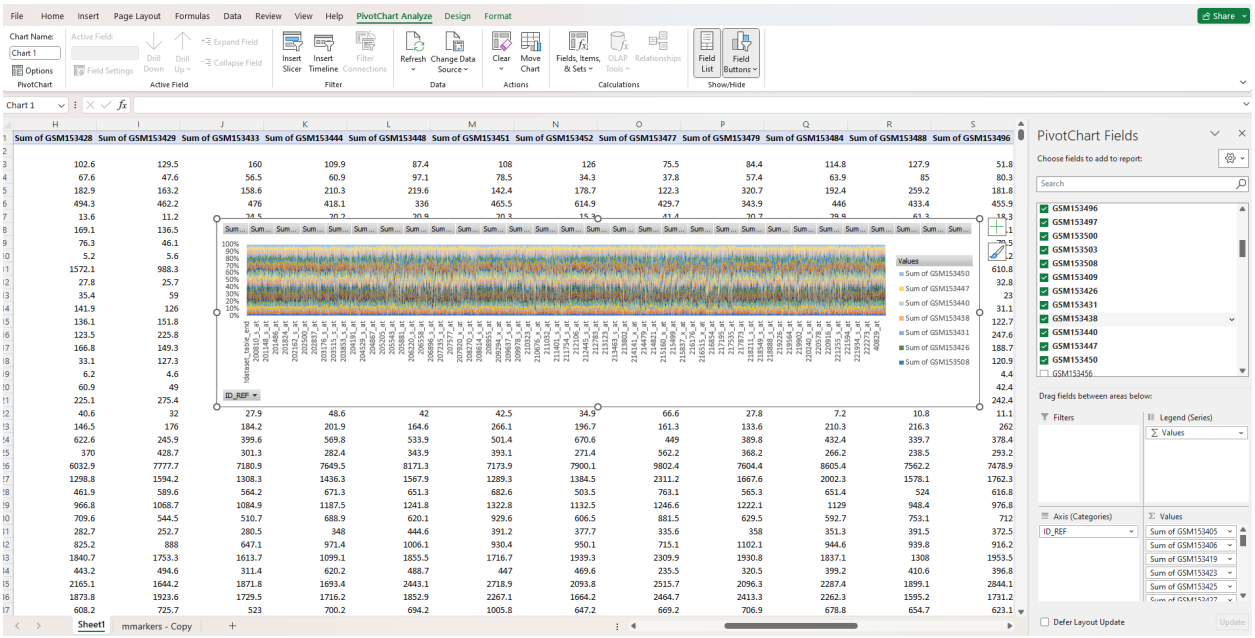


experiment or similar high-throughput technique) is organized in an Excel spreadsheet. Each column's "GSM" numbers denote a different sample or experimental condition, and each row corresponds to a different gene with an "ID_REF" designation.

Data Aggregation: In one of the images, we see the use of a PivotTable to aggregate the data. PivotTables in Excel are used to summarize, analyze, explore, and present summary data. It looks like the sums of the expression values for each GSM sample are being worked out here. This could be a step toward normalization or a view of the data as a whole.



Heatmap Generation: The final image shows the creation of a heatmap using the 'Conditional Formatting' feature of Excel, which is applied to the table of gene expression data. Conditional formatting allows you to apply a color scale to the data where the color intensity reflects the magnitude of the value — a common practice for visualizing expression levels.



Statistical analysis

The statistical analysis was done in Python using Google collaboration. The following is the summary of the code and steps involved.

Data Preparation: The dataset consists of 109 columns and 22,283 rows. The columns consist of gene samples from people with different health conditions, i.e., healthy controls and Parkinson's disease. The unnecessary columns like gene location and description were deleted using Excel, and the file was converted into .csv.

Data preprocessing: The data was then checked for any missing values using Excel, and it was found that there were no missing values for the required data.

Importing data to python: The data was imported into python along with all the necessary libraries, such as pandas, Numpy, etc.

Importing pandas and reading the dataset

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.metrics import silhouette_score
data = pd.read_csv("/content/mmarkers.csv",index_col=0)
data = data.T
```

ID_REF	is_PD	1007_s_at	1053_at	117_at	121_at	1255_g_at	1294_at	1316_at	1320_at	1405_i_at
GSM153405	healthy control	105.1	58.4	179.8	497.8	18	139.1	51.2	13.7	492.4
GSM153406	healthy control	145.7	52.5	192	346.3	40.1	163.8	72	38.3	1121.6
GSM153419	healthy control	107.2	57	252.5	567.1	8.3	172	90.5	5.6	973.2
GSM153423	healthy control	114.4	65.5	217.8	412.4	24.9	173.5	71.2	15.4	1213.9
GSM153425	healthy control	117.7	32.1	167	533.9	34	127.5	50.6	10.7	1249.2

Statistical testing: We used the preprocessed data to perform a t-test. In this step, it was required to change the values of the labels from categorical tboolean,an such as, healthy control = 0, PD = 1. We then used ttest_ind() function on pd samples and control samples and calculated the p-values.

Performing T-test and FDR

```
from scipy.stats import ttest_ind
from statsmodels.stats.multitest import multipletests
from sklearn.feature_selection import SelectKBest, f_classif
from sklearn.model_selection import cross_val_score
from sklearn.linear_model import LogisticRegression
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
X = data.drop('is_PD', axis=1)  # gene expression data
y = data['is_PD']  # PD vs control labels
# Convert all columns to numeric and drop those that cannot be converted
X_numeric = X.apply(pd.to_numeric, errors='coerce')

# Drop all columns that have only NaN values (which indicates non-numeric data that couldn't be converted)
X_numeric = X_numeric.dropna(axis=1, how='all')

# Now, also drop rows with NaN values
X_numeric = X_numeric.dropna()

# Make sure y is aligned with X_numeric
y_aligned = y.loc[X_numeric.index]
# Convert 'y' to binary format and align it with the cleaned X
y_binary = y_aligned.map({'Parkinson\'s disease': 1, 'healthy control': 0}).dropna()
# Perform t-tests
p_values = []
for gene in X_numeric.columns:
    pd_samples = X_numeric[y_binary == 1][gene]
    control_samples = X_numeric[y_binary == 0][gene]
    t_stat, p_val = ttest_ind(pd_samples, control_samples)
    p_values.append(p_val)
```

In the next step, for multiple testing, we used FDR to calculate the adjusted p-values. The smallest p-values that are less than 0.05 are then stored in the variable,which represent the 8 marker genes.

```
# Correct for multiple testing (e.g., FDR)
_, adjusted_p_values, _, _ = multipletests(p_values, alpha=0.05, method='fdr_bh')

# Feature selection (Select top 8 genes)
selected_genes = pd.Series(adjusted_p_values, index=X.columns).nsmallest(8).index

# Now use these genes to create a machine learning model
# For example, using logistic regression with cross-validation
X_selected = X[selected_genes]
pipeline = make_pipeline(StandardScaler(), LogisticRegression(max_iter=10000))
```

```
38] print(p_values)
```

```
00783109, 0.2185182323473658, 0.2624612786038387, 0.5471003466520341, 0.03051674644374021, 0.409133
```

```
print(f"Selected genes: {selected_genes}")
```

```
Selected genes: Index(['207205_at', '211043_s_at', '214560_at', '217832_at', '206139_at',
       '215179_x_at', '204255_s_at', '208481_at'],
      dtype='object', name='ID_REF')
```

Potential analysis:

Using the above analysis, the next step would be to calculate the risk scores from the data. But due to limited time and the necessity of dealing with a huge dataset with 22,283 rows, it was not possible to continue with the analysis. Given more time, we could calculate the risk scores, which can be used to generate charts such as the ROC curve. We can also test the machine learning model and can use it to make predictions.

Team Contributions

Muttaki Bismoy:
Data preparation - Prepared the data to plot the heatmaps using excel.
Data aggregation- Aggregated the data used to plot the heatmaps.
Heatmap - Plotted the heatmap in excel to get an understanding of the genes.

Rashmita Vaggu:
Data Preparation: Prepared processed and cleaned data for statistical analysis in python.
T-testing - Used T-test to calculate p-values used to find the marker genes.
Multiple testing- Used FDR to perform multiple testing and calculate adjusted p-values
ML Model- Partially designed Machine learning model which can be further used for analysis.

Meher Nivedita Avdut:
Information collection- Gathered all the information required for data analysis from the research paper.
Generating Idea- Creatively analyzed and came up with solutions needed for analysis and documentation.
Documenting - Made a detail documentation by adding important sections following the rubric.