Universitat Rovira i Virgili
Escola Tècnica Superior d'Enginyeria

Modelling Parkinson's Disease and API Integration

Master in Health Data Science — MHEDAS

Subject: Scientific Programming

Author: Annabel Margalef Burgos, Roger Puig I Arxer, Ravneet-Rahul Sandhu Singh, Scarlett Thomas

Date: January 25, 2025

UNIVERSITAT DE VIC
UNIVERSITAT CENTRAL
DE CATALUNYA

Universitat de Girona

UGA
Université
Grenoble Alpes

UNIVERSITAT DE BARCELONA

Universitat de Lleida

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH

UNIVERSITAT ROVIRA i VIRGILI

BIOINFORMATICS
BARCELONA

UAB
Universitat Autònoma
de Barcelona

eit Health

Co-funded by the
European Union

Table of Contents

1. **Introduction** 1
2. **Methodology** 2
3. **Results** 3
4. **Conclusions** 5

References 6

1. Introduction

Parkinson's disease (PD) is a progressive neurodegenerative disorder that affected over 8.5 million people globally in 2019 [1], with numbers increasing due to aging populations and improved diagnostics. Around 90% of individuals with PD experience speech impairment [2], primarily hypokinetic dysarthria, which involves reduced vocal loudness, monotone speech, imprecise articulation, and fluctuating speech rates. Detecting PD early through speech analysis can significantly improve access to treatment and may help slow disease progression.

Various machine learning methods have been applied to PD detection using speech features. Support Vector Machines (SVM) classify patients based on acoustic features like Mel Frequency Cepstral Coefficients (MFCC) and AutoEncoder-derived patterns [3]. Deep Neural Networks (DNN) capture complex patterns in speech data for accurate classification [4], while Random Forests (RF) help identify key speech features and assist in distinguishing PD-related symptoms [5].

Building upon these approaches, this project investigates the potential of K-Nearest Neighbors (KNN) for PD detection. Three distinct KNN models will be developed: one trained on cleaned data, another on aggregated data, and a third on normalized data. As a simple and non-parametric algorithm, KNN makes minimal assumptions about the underlying data distribution, which makes it particularly well-suited for smaller datasets and multi-class classification tasks. Furthermore, an API will be implemented to allow users to input their own values and test the models interactively, providing a practical and user-friendly tool for PD prediction. **Figure 1.1** illustrates the project timeline and task distribution, covering the period from 8/01/2025 to 19/01/2025.

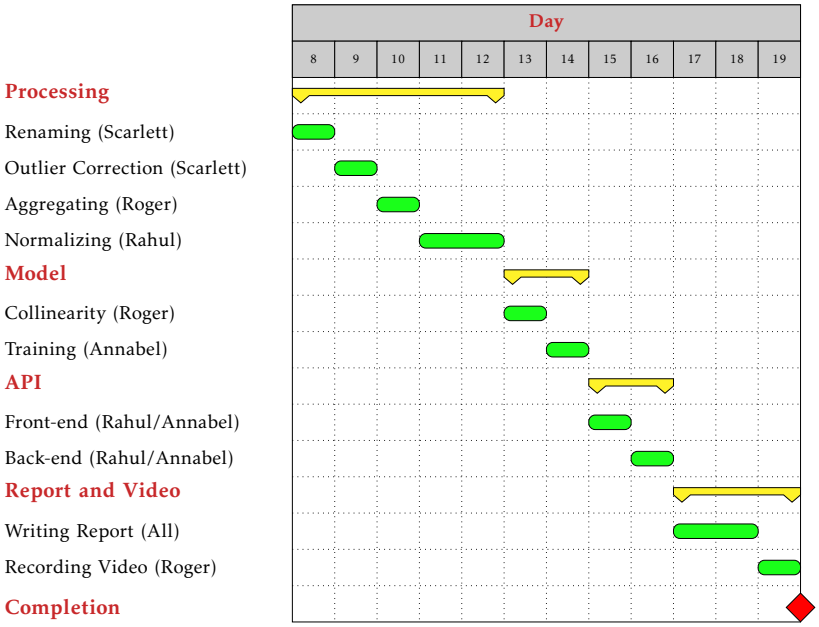

Figure 1.1: Gantt chart of the project.

2. Methodology

The dataset includes voice measurements from 31 individuals, 23 with Parkinson's disease, across 195 recordings. The recordings are labeled as 0 for healthy and 1 for Parkinson's. Features include: absJitter, apq, apq3, apq5, avFF, dda, dbShimmer, ddp, D2, DFA, HNR, lShimmer, maxFF, minFF, NHR, percJitter, PPE, ppq, rap, RPDE, spread1, and spread2.

We began with an exploratory analysis and found no missing values, NaNs, or duplicates. Preprocessing included: renaming columns using `rename()` for consistency; removing outliers with `outliers()` based on the IQR method ($Q3 - Q1$), replacing them with the mean per subject to preserve within-subject consistency; aggregating trials per subject using `aggregate()` to compute mean feature values; and applying Min-Max scaling via `normalize()`.

For the KNN model, we conducted dimensionality reduction, focusing on parameters related to fundamental frequency, jitter, and shimmer. The `collinearity()` function calculated correlations within each feature group and identified the least correlated pairs. To retain a feature, we compared its correlation with all other features. A scoring system assigned higher scores to features with lower overall correlation. Features with higher scores were kept, while others were removed.

In **Figure 2.1**, we show correlations for fundamental frequency features. As you can see, since these metrics are relative to the same measure, it makes sense that they show some sort of correlation between them. The final selected features to train the model were: absJitter, apq, D2, DFA, HNR, maxFF, NHR, PPE, RPDE, spread1, spread2.

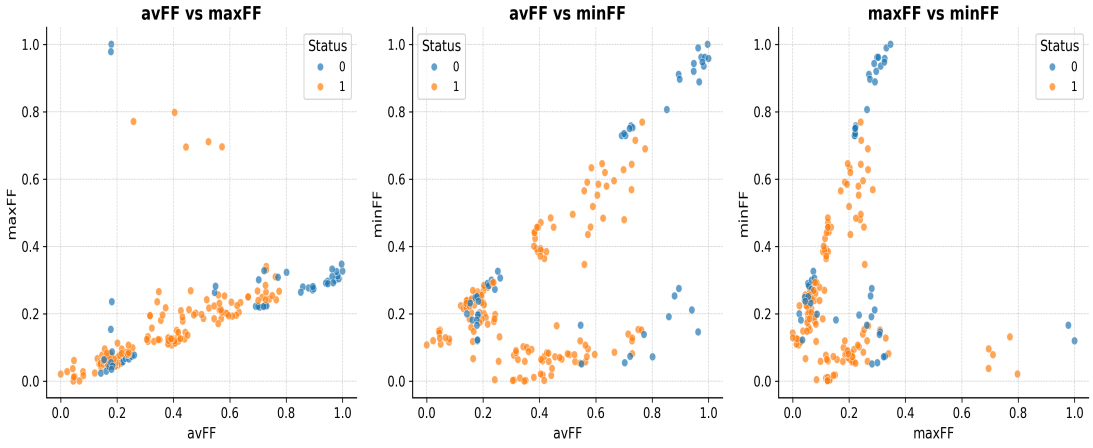

Figure 2.1: Correlation among fundamental frequency-related parameters.

For validation, the data was split into 70% training and 30% testing. To optimize the hyperparameter k, we selected the value where testing and training accuracy were similar. The intersection of these points was found using linear interpolation in the `best_k()` function.

3. Results

In **Figure 3.1**, we compare training and testing accuracy of the KNN model across three preprocessing methods: raw (`df_clean`), averaged (`df_avg`), and normalized (`df_norm`). For the raw model, optimal $k \approx 12$ balances training and testing accuracy. The averaged model shows flat trends, likely due to information loss from aggregation. The normalized model performs best, with optimal $k \approx 4$, offering balanced and stable accuracy.

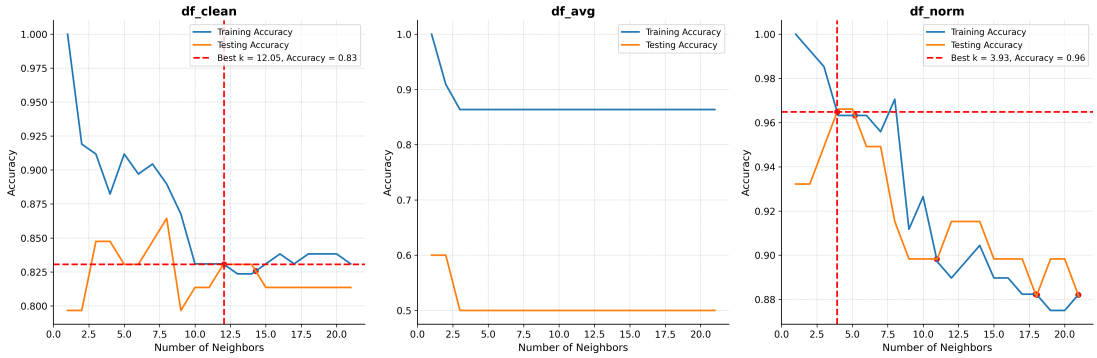

Figure 3.1: Optimal k determination using testing and training accuracy.

Finally, the last step is to integrate the whole process within an API. The backend of this system is implemented using FastAPI. Its primary functions include serving the frontend, managing model files, retrieving evaluation metrics, and handling prediction requests, as shown in **Figure 3.2**. The root endpoint (`/`) serves the `index.html` file, which provides the user interface for interaction. The system organizes models into datetime-labeled folders under `MODEL_DIR`, where each folder represents a distinct training session and contains model files (`.pkl`) and evaluation metrics stored in a `metrics.txt` file.

```
INFO:     Started server process [13222]
INFO:     Waiting for application startup.
INFO:     Application startup complete.
INFO:     Uvicorn running on http://0.0.0.0:8000 (Press CTRL+C to quit)
INFO:     127.0.0.1:53832 - "GET / HTTP/1.1" 200 OK
INFO:     127.0.0.1:53832 - "GET /datetimes/ HTTP/1.1" 200 OK
INFO:     127.0.0.1:53832 - "GET /models/?datetime_folder=2025-01-17_00-43-59 H
TTP/1.1" 200 OK
INFO:     127.0.0.1:53832 - "GET /metrics/?datetime_folder=2025-01-17_00-43-598
model_name=model_avg.pkl HTTP/1.1" 200 OK
INFO:     127.0.0.1:53832 - "GET /metrics/?datetime_folder=2025-01-17_00-43-598
model_name=model_norm.pkl HTTP/1.1" 200 OK
INFO:     127.0.0.1:53832 - "GET /metrics/?datetime_folder=2025-01-17_00-43-598
model_name=model_clean.pkl HTTP/1.1" 200 OK
INFO:     127.0.0.1:55354 - "POST /predict/ HTTP/1.1" 200 OK
```

Figure 3.2: Example of messages exchange between frontend and backend.

The `/datetimes/` endpoint retrieves all datetime-named folders, enabling users to explore different model versions. Once a folder is selected, the `/models/` endpoint lists available model files within it. The `/metrics/` endpoint fetches and displays the content of the `metrics.txt` file for the chosen folder, allowing users to view the performance metrics of the associated models. The prediction process is handled by the

`/predict/` endpoint, where the user provides feature inputs and selects a model. The backend then loads the specified model using the `pickle` module and if the selected model is `model_norm` it applies min-max normalization, using pre-computed max-min values from the normalized dataset. Predictions are generated by the model and returned as a JSON response, indicating whether the input data corresponds to a healthy or Parkinson's individual.

The frontend was built using HTML, CSS, and JavaScript. The **Figure 3.3** shows a preview of it. Users can select datetime folders and models through dynamically populated dropdown menus, which are updated by fetching data from the backend endpoints `/datetimes/` and `/models/`. Additionally, users can upload a JSON file containing feature values, and the uploaded data automatically populates the input fields, simplifying the data entry process. Metrics for the selected model are retrieved via the `/metrics/` endpoint and displayed directly on the page to provide insights into the model's performance.

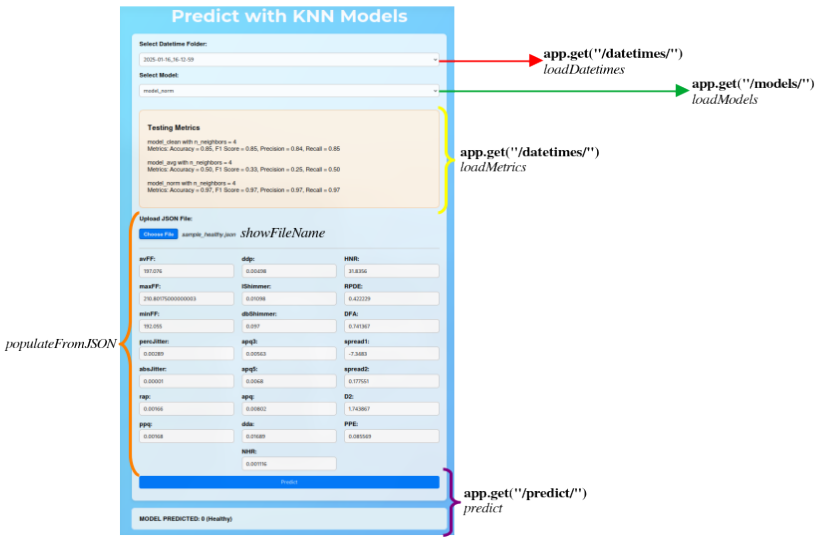

Figure 3.3: Frontend preview with annotated elements: functions related to the backend are shown in **bold**, while functions specific to the frontend are highlighted in *italics*.

The frontend handles form submissions by sending a POST request to the `/predict/` endpoint with the user-provided inputs and the selected model. The backend processes the request, and the prediction results are displayed directly on the webpage. JavaScript functions manage dynamic interactions, such as loading datetime folders, model files, and metrics, as well as validating uploaded JSON files and showing prediction outputs.

4. Conclusions

The purpose of this study was to detect Parkinson's disease (PD) using voice measurements and to evaluate the performance of three different preprocessing approaches with a KNN model: the raw preprocessed model, the averaged model, and the normalized model. Exploratory data analysis was performed to identify outliers, missing values, and inconsistencies in the raw data. The dataset was then preprocessed through renaming, outlier handling, aggregation, and normalization. Dimensionality reduction and correlation-based feature selection were applied to optimize model performance.

The results demonstrated that preprocessing methods significantly influence the accuracy and stability of the KNN model. Among the three approaches, the normalized model (`df_norm`) provided the best performance, achieving an optimal $k = 4$ with nearly 97% accuracy, compared to approximately 83% accuracy with the raw preprocessed model (`df_clean`). These findings underscore the importance of normalizing and scaling features to ensure compatibility and improve model performance. The averaged model (`df_avg`) failed to achieve convergence, likely due to insufficient data points and the resulting loss of critical information. This suggests that averaging may only be effective on substantially larger datasets.

The integration of the project workflow into an API enhanced model management, evaluation, and prediction capabilities. The user-friendly frontend of the API allows users to explore different model versions, upload data, and obtain predictions seamlessly. This implementation not only facilitates collaboration but also provides a foundation for real-world applications of the model.

However, the study has several limitations. The models were trained on a small dataset consisting of 31 individuals, which limits the generalizability of the results. Training on larger datasets would produce a more robust predictive model. Additionally, outlier handling, which replaced outlier values with within-subject non-outlier averages, may have excluded important Parkinson's-specific diagnostic information, potentially reducing the model's sensitivity. Dimensionality reduction and feature selection were based solely on correlation metrics, which capture only linear relationships and may overlook non-linear interactions between features. Finally, the model is designed to classify data into a binary outcome (Parkinson's or no Parkinson's), whereas predicting disease stages or progression would provide greater clinical value.

Future directions for this work include expanding the dataset to improve model robustness, integrating additional biomarkers (e.g., handwriting or gait analysis) alongside voice data, and enhancing the API to support real-time data input and predictions. These advancements would significantly increase the utility of the system, particularly in clinical settings, where early detection and personalized treatment of PD are critical.

References

[1] World Health Organization. Launch of WHO's Parkinson Disease Technical Brief. Accessed: 2025-01-15. URL: https://www.who.int/news/item/14-06-2022-launch-of-who-s-parkinson-disease-technical-brief.

[2] Indiana Parkinson Foundation. Speech and Swallowing Disorders in Parkinson Disease. Accessed: 2025-01-15. URL: https://www.indianaparkinson.org/speech-and-swallowing-disorders-in-parkinson-disease.

[3] Springer. Speech Processing for Early PD Diagnosis, 2023. Accessed: 2025-01-15. URL: https://link.springer.com/article/10.1007/s10772-023-10070-9.

[4] AIP Publishing. Parkinson's Disease Detection from Voice Using Artificial Intelligence, 2024. Accessed: 2025-01-15. URL: https://pubs.aip.org/aip/acp/article/3232/1/040010/3316679/Parkinson-s-disease-detection-from-voice-using.

[5] Springer. Machine Learning for PD Diagnosis Using Speech Analysis, 2023. Accessed: 2025-01-15. URL: https://link.springer.com/article/10.1007/s13278-022-00905-9.