Comprehensive Technical Test for Data Senior (IA) at Samay

Lung Sound Analysis for Respiratory Health

Objective:

The primary goal of this technical test is to design, develop, and optimize a machine-learning solution for detecting and classifying respiratory diseases. This will be achieved using a dataset of lung sounds recorded with an electronic stethoscope. The comprehensive IPython Notebook delivered should demonstrate the candidate's proficiency in analyzing sensor data for health applications, specifically in pulmonary diseases.

Dataset Description:

The evolution of stethoscope technology has facilitated the high-quality recording of lung sounds from healthy individuals and those with various pulmonary conditions. This dataset encompasses audio recordings from patients with seven different ailments, including asthma, heart failure, pneumonia, bronchitis, pleural effusion, lung fibrosis, and COPD, alongside normal breathing sounds. Recordings were taken from multiple positions on the chest, as determined by a specialist physician. Each sound was recorded thrice using different frequency filters to highlight specific bodily sounds. This valuable dataset supports the development of automated tools for diagnosing pulmonary diseases through lung sound analysis and can be extended to heart sound studies.

The dataset comprises audio recordings of lung sounds from 112 subjects, captured using an electronic stethoscope. It includes data from 35 healthy individuals and 77 subjects with respiratory diseases [1, 2, 3].

Content: The audio recordings have been filtered through Bell, Diaphragm, and Extended modes to ensure clarity and precision in sound quality.

Annotations: Each audio file is annotated with comprehensive details, including the type of lung sound, the disease diagnosis, the recording location on the subject's chest, and the subject's age and gender. This information is crucial for the analysis and classification tasks.

1. First part

In this part, the test will evaluate the candidate's ability to load a dataset into a Python environment (likely using libraries like Pandas) and perform preliminary analysis. Tasks may include reading data from various file formats, understanding the dataset's structure, and identifying initial trends or patterns.

A. Use url="https://prod-dcd-datasets-cache-zipfiles.s3.eu-west-1.amazonaws.com/jwyy9np4gv-3.zip" to download a ZIP archive, and read the content. Do an exploratory analysis of the extracted data. (We suggest creating a function to extract each filename, construct labels, and then build histograms or any tool that allows you to analyze the distributions and trends)

B. Generate a table data annotation, that contains demographic data of the patients, some of the columns mean:

- **Sound Type:** Inspiratory: I, Expiratory: E, Wheezes: W, Crackles: C, N: Normal
- **Location:** Posterior: P Lower: L Left: L Right R, UPPER: U, ANTERIOR: A, MIDDLE: M Create a Frequency Table with absolute and relative values of each of the variables Graph a boxplot of the age facet with diagnosis (if a patient have two diagnoses, the patient will count to the group two times)

2. Second part

This part focuses on signal processing and feature engineering, particularly in the context of audio or time-series data. Candidates may be asked to perform tasks like analyzing audio signals, visualizing data (e.g., waveforms), extracting relevant features (e.g., MFCC features), and explaining their significance.

A. Create a Python script that iterates through an array of WAV file paths, reads each file, and stores relevant information about each file in an array. This script will result in a list of dictionaries, where each dictionary contains detailed information about a WAV file, including its path, sample rate, audio data, and the data type of the audio data. This is useful for audio processing tasks where you must handle multiple audio files and require information about their sample rate and data type. Then, create a data frame that contains two columns to save data arrays and their corresponding label and do an Overall Statistical Analysis to get statistical information for each disease.

B. Write functions to get and plot the spectrogram, which will use samples with length=16000 (If necessary, use techniques like zero-padding), and use them to plot some samples for each one of the diseases. The main idea is you can explain the graph's meaning and look for insights by a visual inspection mode. If you wish, you can plot the time series waveform as well.

C. Create a Python function that extracts various audio features from audio data (E.g. duration, amplitude mean, amplitude median, amplitude standard deviation, amplitude maximum, amplitude minimum, zero crossing rate, energy, and so on; use your creativity and look for in literature).

D. Choose a frequency transformation (E.g. FFT, Mel-Frequency Cepstral Coefficients MFCC, or other popular feature extraction methods in speech and audio processing). They represent spectral properties of sound in a particularly effective way for machine learning applications, such as speech recognition and music analysis [4]. Extract features for normalized audio and save it for further study. (You can use librosa library to make the process easier).

3. Third Part

In this part you will use advanced machine learning techniques, exploring options like ensemble methods, decision trees, and neural networks. The candidate should demonstrate creativity and proficiency in choosing and applying the most effective modeling techniques for this specific application.

General considerations:

- To get the dataset, you should use the data frame extracted in section 2(A), normalize the length (all samples should have the same shape), and scale it.
- You could use an instance of OneHotEncoder from scikit-learn library or similar to encode categorical labels in your data frame. So far, your dataset consists of scaled features and encoded labels. The split should allocate 20% of the data to the test set, apply stratification based on the labels, and use a fixed random state for reproducibility.

A. **KNN algorithm**.
    a. Create a Python code set that sets up a grid of hyperparameters for the k-Nearest Neighbors (kNN) classifier. The grid should include different values for the number of neighbors to consider, ranging from 3 to 15, and two types of weighting schemes: 'uniform' and 'distance'. This grid will be used for hyperparameter tuning in machine learning models.
    b. Create a code snippet to instantiate a Grid Search object. This object is for tuning the hyperparameters of a K-Nearest Neighbors (KNN) classifier. Use the base estimator and search over a provided grid of hyperparameters.
    c. Write a code snippet to retrieve and display the best set of hyperparameters selected by GridSearch for a K-Nearest Neighbors (KNN) model.

d. Write a code snippet to evaluate the performance of a K-Nearest Neighbors (KNN) model that has been optimized using GridSearch. Use the best estimator obtained from the grid search to make predictions on the testing dataset.

e. Create a code snippet that generates a confusion matrix using scikit-learn or similar.

B. Artificial Neural Networks algorithm

a. Create a code snippet to train a neural network model using PyTorch, TensorFlow's Keras API, or a similar framework. Additionally, include a validation dataset composed of X_test and y_test for performance evaluation during training. Implement callbacks to monitor and save the model's progress and set the verbosity level to display progress updates.

b. Create a heatmap to display the confusion matrix

C. Machine Learning Algorithm

a. Develop a multiclass classifier using a classical technique(SVM, XGBoost, LGBM, etc.) and present the results.

Additional Considerations

-The submission must be made on a fully functional notebook so that the test reviewers can run it without making changes.

-Display clarity and effectiveness in presenting complex technical ideas to multidisciplinary teams and stakeholders.

-Ensure the code is well-structured, commented, and adheres to best practices in software development. (Optional: Evaluate the computational efficiency of the algorithms.)

-Summarize key findings, model performance, and potential impact on respiratory health diagnostics.

-Include a comprehensive bibliography of all sources referenced throughout the test.

-This test is designed to be solved in 6 hours of effective work. The candidate will have 36 hours to resolve the issue.

Literature References

1. Fraiwan, Mohammad; Fraiwan, Luay; Khassawneh, Basheer; Ibnian, Ali (2021), “A dataset of lung sounds recorded from the chest wall using an electronic stethoscope”, Mendeley Data, V3, doi: 10.17632/jwyy9np4gv.3.
2. Luay Fraiwan, Omnia Hassanin, Mohammad Fraiwan, Basheer Khassawneh, Ali M. Ibnian, Mohanad Alkhodari, "Automatic identification of respiratory diseases from stethoscopic lung sound signals using ensemble classifiers," Biocybernetics and Biomedical Engineering, Volume 41, Issue 1, 2021, Pages 1-14, ISSN 0208-5216, https://doi.org/10.1016/j.bbe.2020.11.003.
3. Kaggle Dataset: Lung Sound Dataset.
4. Md. Afzal Hossan, Sheeraz Memon, Mark A Gregory, “A Novel Approach for MFCC Feature Extraction ”, https://ieeexplore.ieee.org/abstract/document/5709752