Denison University

Data Analytics Program

DA 401: Senior Seminar in Data Analytics

Regression Model in Shaping the Voice Communication Control

Researcher: Minh Nguyen

Supervisor: Dr. Sarah Supp

Table of Contents

1. Abstract

"Talking while driving: is not a good idea. Texting while driving: is downright dangerous and talking while flying: is essential" (Volpe National Transportation Systems Center, 2022). The system primarily faces challenges with sound disruption and communication between air traffic controllers and pilots. Therefore, this study aims to integrate AI tools with sound disruption and text-driven communication systems to identify voice disturbances and guarantee consistent international messaging (Volpe National Transportation Systems Center, 2022). I specifically focused on how the integration of AI technology can enhance the efficiency and reliability of the Voice Communication System (VCS). Background noise in audio recordings can degrade the sound quality in the VCS. This study investigates using logistic regression to classify audio files as containing background noise or being clean recordings.

The signal-to-noise ratio (SNR) is extracted from the sound files. This feature is used to train a logistic regression model to predict if an audio file contains background noise or not. For the research question, can the logistic regression model detect whether the sound files contain background noise or clean noise? Using logistic regression to detect the clarity of the conversation between the controllers and the pilots, I can gain meaningful insights into how machine learning has shaped and enhanced voice control systems. The hypothesis is that by extracting SNR and training a logistic regression, the model can distinguish between audio with background noise and clean audio without noise. The audio feature is expected to differ systematically between noisy and clean files in ways that the logistic regression model can learn for classification. This hypothesis can be tested by training the model on a dataset of audio files labeled as containing noise or being clean, and then evaluating its performance on a held-out test set.

The metrics (**Accuracy**, **Precision**, **Recall**, and **F1-score**) of the logistic regression model are evaluated, on a test set of audio recordings with and without background noise, to classify model performance. By applying the logistic regression model that utilizes the Dicom Corporation Dataset. The analysis process is implemented in Python (v. 3.8.8), Librosa (v. 0.10.1), Soundfile (v. 0.12.1), Matplotlib (v. 3.3.4), Numpy (v. 1.20.1), Scikit-learn (v. 0.24.1), Scipy (v. 1.6.2). The open-source web application "Jupyter Notebook" (v. 6.3.0) and desktop graphical user interface "Anaconda Navigator" (v. 2.0.3) provide an environment and packages for Python.

Keywords' explanation:

- <u>Logistic Regression Model</u>: Logistic regression is a supervised machine learning technique used for binary classification tasks where the outcome is categorical with two options (Belyadi & Haghighat, 2021). Logistic regression utilizes a specific logistic mathematical function to model a dependent variable that can only take on values of 0 or 1, representing the two classes (Belyadi & Haghighat, 2021). This restricts the predicted probabilities generated by the model to fall between 0 and 1, rather than allowing a continuous range of values from negative infinity to positive infinity (Belyadi & Haghighat, 2021).
- <u>Classification metrics</u>
    - **Accuracy:** shows how often a classification ML model is correct overall ("Accuracy vs. precision vs. recall,"n.d.).
    - **Precision:** shows how often an ML model is correct when predicting the target class ("Accuracy vs. precision vs. recall,"n.d.).

- **Recall:** shows whether an ML model can find all objects of the target class ("Accuracy vs. precision vs. recall,"n.d.).
- **F1:** This score integrates precision and recall into a single metric to gain a better understanding of model performance (Sharma, 2023).

2. Introduction

(a) General Topic

A Voice Communication System (VCS) is essential in the aviation communication framework, allowing pilots, air traffic managers, and ground staff to converse. VCS ensures clear and efficient voice dialogues, promoting the secure and smooth functioning of air traffic management (Durgut,2023). Additionally, it is specifically developed for civil and military air traffic control such as ACC and APP control centers, ATC towers, simulators, and backup/emergency (Indra Air Communication, n.d.). VCS includes microphones, headsets, radios, and other integrated equipment. These components enable clear in-flight communication despite cockpit noise, vibrations, and other challenges (Durgut, 2023). This study aims to integrate machine learning with sound disruption to identify voice disturbances in the aviation industry (Volpe National Transportation Systems Center, 2022).

(b) Literature Review

Voice communications have traditionally been simple to intercept and monitor. When the digital cell and wireless phones arrived, there was a momentary window in which monitoring voice communications across these digital connections was difficult. There was a final report about the analysis of voice communication in a simulated approach control environment by O. Veronika Prinzo in the Civil Aeromedical Institute Federal Aviation Administration (Prinzo, 1998).

Prinzo analyzed air traffic control communications and found a high rate of irregularities (Prinzo, 1998). Specifically, 40% of controller statements and 59% of pilot statements contained some type of ambiguity or confusion (Prinzo, 1998). Over 90% of these issues occurred when giving instructions, advisories, or addressing others (Prinzo, 1998). While not directly hazardous, these problems impact efficiency.

Morrison & Wright (1989) and Morrow, Rodvold, & Lee (1994) suggest workload contributes to miscommunications. As traffic, congestion, and message length increase, so do errors. Three vocal qualities in particular reflect rising controller workload: pitch, volume, and speech rate (Prinzo, 1998). For instance, Griffin & Williams (1987) found emotional stress and task complexity manifest in higher-pitched, louder, faster speech. Furthermore, Brenner et al. (1994) report mental workload impacts language production similarly. As demands rise, controllers display signals of cognitive overload through their voice patterns. Taken together, this indicates issues like excessive traffic load not only directly reduce air traffic performance, but also introduce communication issues that further degrade efficiency and safety margins. Managing controller responsibilities is key to optimizing critical voice interactions between towers and aircraft.

The growing global air travel market has sparked high demand for new commercial aircraft and airport infrastructure construction. India and China have seen particularly rapid air traffic growth due to their massive populations, exemplified by India's recent announcement to build 100 additional airports by 2024 for expanded connectivity ("Voice Communication Control System Market Report,"n.d.). More airports globally will require upgraded air traffic management systems to enable efficient operations. Voice communication control systems (VCCS) play an integral role in various defense applications like military communications and aircraft control

("Voice Communication Control System Market Report,"n.d.). As military organizations worldwide invest in advanced VCCS to ensure safer, more effective operations, steady growth is expected in the military domain as well. Overall, rising air transportation and a push towards modernization in both commercial and defense aviation are driving the voice communication system market across sectors.

3. Methods

(a) Load and trim the sound files

The script is designed to handle audio file processing, specifically trimming the duration of audio files to a predetermined length. The *"file_list"* contains paths to four different audio files (HF_ANALOG.wav, VHF_ANALOG.wav, VHF_VOIP.wav, and VOIP telephone.wav) are mentioned in **Table 1**. The function *"cut_to_duration"* is the core of this script. It accepts the path to an audio file, an output file path, and an optional duration parameter in seconds (defaulting to 10 seconds). This flexibility allows users to specify how long the output file should be, which is crucial for tasks like creating consistent datasets for machine learning or simplifying audio data for analysis. By standardizing the length of all sound files to the first 10 seconds, it becomes easier to compare and analyze them. This uniformity is crucial for many algorithms in machine learning and audio processing, where having a consistent input size can significantly improve the effectiveness and accuracy of the models. Trimming to the first 10-second window can focus on the sound noise of the audio because the rest parts of the radio most likely contains no sound.

Table 1: Voice recording from the VCS system

Recording files	Description
HF_ANALOG.wav	● WAV: The digital format that preserves all of the original audio information (lossless audio format) ● Utilized analog radio to communicate voice signals using High-Frequency (HF) wave ● 10 seconds duration ● 160 KB size
VHF_ANALOG.wav	● WAV: The digital format that preserves all of the original audio information (lossless audio format) ● Utilized analog radio to communicate voice signals using Very-High-Frequency (VHF) wave ● 10 seconds duration ● 160 KB size
VHF_VOIP.wav	● WAV: The digital format that preserves all of the original audio information (lossless audio format) ● Utilized VoIP radio to communicate voice signals using Very-High-Frequency (VHF) wave ● 10 seconds duration ● 160 KB size
VOIP telephone.wav	● WAV: The digital format that preserves all of the original audio information (lossless audio format) ● Utilized broadband Internet connection instead of a regular (or analog) radio ● 10 seconds duration ● 160 KB size

Analog radio: Analog radios have communicated messages since the 1930s for military and business use. These radios commonly employ VHF or UHF waves to transmit voice data to receiving radios, which decode the signals into audio (Analog vs Digital Radios," n.d.).

VoIP radio: Originally designed as a telephone replacement, VoIP allowed large service providers to connect many users for voice communications on demand. They integrate radio networks easily with existing phone and voice systems (“What is (RoIP) Radio over IP and (VoIP) Voice over IP?” 2014)

High-frequency: Cover far greater distances due to ionospheric refraction, but HF radio signals are prevented from atmospheric conditions such as geomagnetic storms or solar flares that radio users cannot control ("Tactical HF vs VHF radio," n.d.). HF communications are best suited for long-distance communication between ground operators and base stations ("Tactical HF vs VHF radio," n.d.)

Very-high-frequency: Travel invisibly over land, which makes these signals ideal for short-distance land communications and indoor applications (“What’s the difference between HF, VHF, and UHF?”n.d.)

The script in “Coding Results.ipynb” utilizes the ***librosa*** library for audio loading and the ***soundfile*** library for writing the processed audio back to the file system. The process begins with loading the audio file using “*librosa.load*”, which returns the audio *time series (y)* and its *sampling rate (sr)*. The desired audio length is then calculated in samples (not in seconds), considering the file's sampling rate. This conversion ensures precision in the trimming process. The script checks if the loaded audio is longer than the desired length; if so, it truncates the audio to the specified duration. The script concludes with a loop that iterates through each file in the “*file_list*”. For each file, it calls “*cut_to_duration*”, effectively automating the process of truncating each file in the list. This loop signifies the script's practicality in handling multiple files efficiently.

(b) Calculate signal-to-ratio (SNR) over time

Signal-to-noise ratio (SNR) is the dimensionless ratio of the signal power to the noise power contained in a recording. Abbreviated SNR by engineers and scientists, the signal-to-noise ratio parameterizes the performance of optimal signal processing systems when the noise is Gaussian - not specified for any given instant of time (Johnson, 2006).

b1. Measure the signal-to-ratio

- Calculate the signal-to-noise ratio (SNR):

$$SNR = \left(\frac{A_{signal}}{A_{noise}}\right)^2 = \frac{P_{signal}}{P_{noise}}$$

whereas, A is a Root Mean Square (RMS) amplitude.

- By the definition of SNR in decibels:

$$SNR_{db} = 10 * log_{10}\left(\frac{P_{signal}}{P_{noise}}\right)$$

The *calculate_snr* function computes the signal-to-noise ratio (SNR) in decibels for a given signal. It begins by estimating the noise floor, the level below which values in the signal are considered noise. This is done by finding the percentile (default is 10th percentile) of the absolute value of the signal, identifying this as the noise floor. Then, the indices where the signal is less than this noise floor are marked as variable "*noise_idx*". Next, the function calculates the power of the signal and the noise. Signal power is the mean of the squared signal, and noise power is the mean of the squared signal values at the noise indices. If the noise power is zero, then it indicates no noise, and the function returns infinity and represents an infinite SNR. Otherwise, the SNR is calculated as the ratio of signal power to noise power. This ratio is then converted to decibels. This logarithmic conversion provides a more intuitive and widely used representation of SNR.

b2. Measure the time window

The data windowing technique is a useful approach for applying machine learning to time series data (Zabaleta, 2023). This method involves dividing the time series into smaller consecutive segments made up of an input sequence and an output sequence.

- Input: data sequence with a certain number of data points (past time steps)
- Output: the corresponding next data points in the sequence (future time steps)

The *calculate_snr_over_time* function evaluates the signal-to-noise ratio (SNR) of an audio signal over time, using a windowing approach to analyze discrete segments of the signal. Initially, the audio data is read from a WAV file and converted to a mono signal if it's stereo. The data is then normalized to ensure consistent amplitude levels. The function employs a Hanning window for each segment or frame of the signal to smooth the values ("numpy.hanning," n.d.). A key aspect of this function is the calculation of SNR values over different time windows of the signal. The frame length and overlap are configurable parameters that dictate the size of each segment and the amount of overlap between consecutive segments. The function iterates over these frames, calculates the SNR for each using a separate *calculate_snr* function, and stores these values. Additionally, the function generates a time vector that correlates these SNR values to specific times in the audio file.

(c) Logistic Regression Model

The logistic regression model is utilized to train 70 different signal-to-noise ratios (SNR), from 1 to 70, and determine which one yields the highest accuracy. For example, *i* equals 1 means that any SNR from the time windows must be greater than 1. These validated SNRs (all SNR > *i*) will be put into the training set, this is called the constraints. The *i* with the highest accuracy will be selected to run the testing set and evaluate the model's performance.

Figure 1, **Figure 2**, **Figure 3**, and **Figure 4** illustrate SNR (in dB) over time and visualize how the signal-to-noise ratio varies throughout each audio recording.

Figure 1: HF_ANALOG (Noise file)

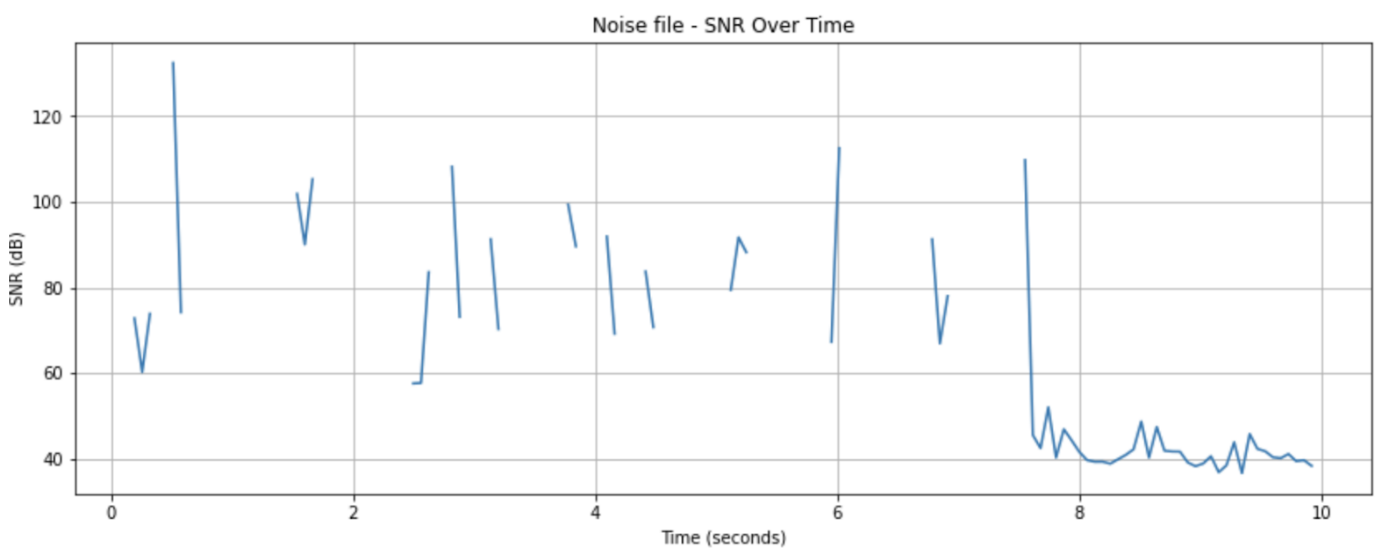

Figure 1: SNR in decibels over time windows in noise file

Figure 2: VHF_ANALOG (Noise file)

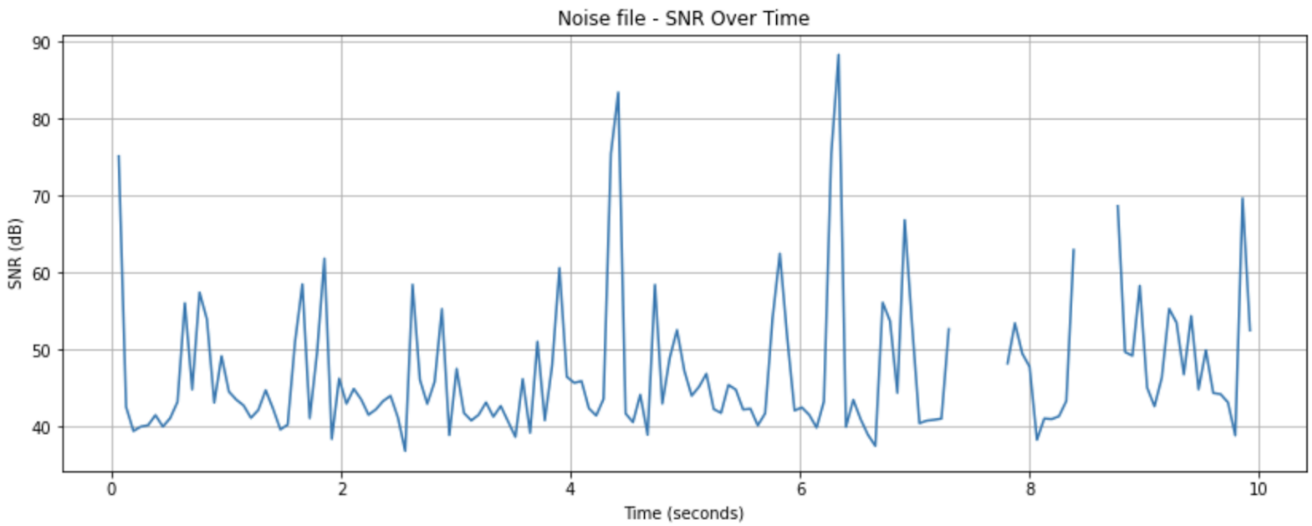

Figure 2: SNR in decibels over time windows in noise file

Figure 3: VHF_VOIP (Clean file)

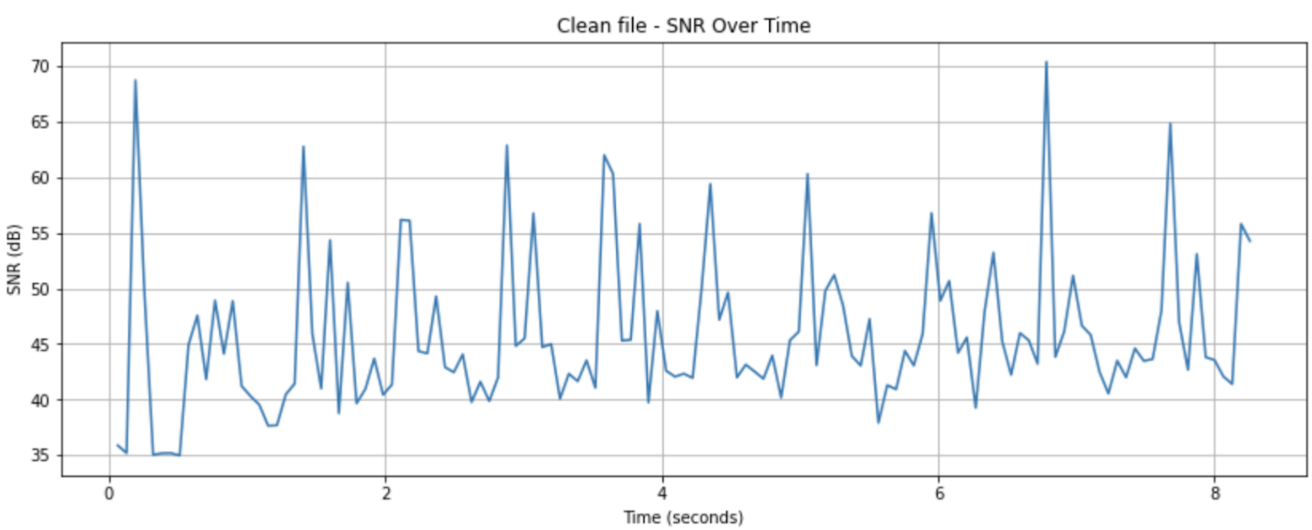

Figure 3: SNR in decibels over time windows in clean file

Figure 4: VOIP Telephone (Clean file)

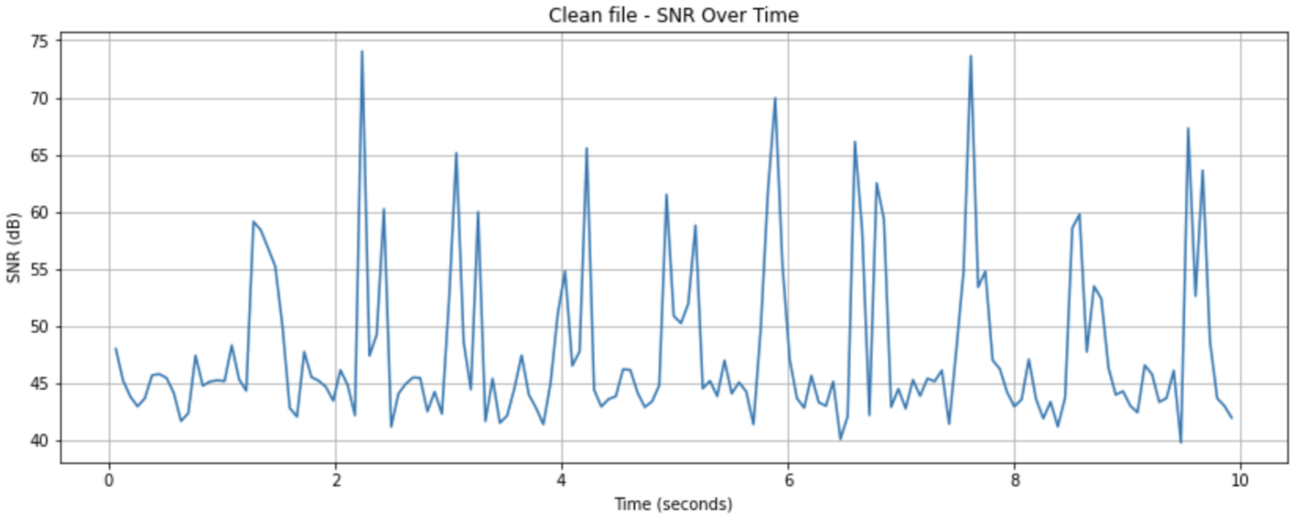

Figure 4: SNR in decibels over time windows in clean file

For the noise files “HF_ANALOG” and “VHF_ANALOG”, the SNR fluctuates dramatically over time. There are peaks where the signal strength temporarily exceeds the noise, followed by dips when noise dominates. This matches expectations for noisy radio communications. In contrast, the clean recordings of “VHF_VOIP” and “VOIP_telephone” maintain a relatively stable SNR over time. The SNR stays consistently high, only dropping during periods of silence. This indicates a lack of background noise. These plots demonstrate how SNR can distinguish clean from noisy audio. Clean recordings have uniformly high SNR over time, while noisy signals have more erratic SNR fluctuations.

c1. Training set

The provided code snippet is designed to evaluate the performance of a Logistic Regression model across 70 different signal-to-noise ratio (SNR) levels. It initializes an empty list “train_list” and a data frame “*model_df*” to store the results. The model is trained for each SNR level, from 1 to 70, using the “*print_dataframe*” and “*np_arr*” functions to prepare the training data (*X_train*, *Y_train)*. After fitting the model with the training data, the code records the SNR level and the model's score on the training set in “*i_list”*, which is then appended to train_list. Finally, *“train_list”* is converted into a DataFrame with columns “I” (for the SNR level) and “Accuracy Score” (for the model score), and the results are saved to an Excel file "training_set.xlsx". This approach systematically evaluates how the logistic regression model's performance varies with different levels of SNR in the training data.

c2. Testing set

This code snippet evaluates the performance of a Logistic Regression model on a testing set where the signal-to-noise ratio (SNR) is equal to or greater than 68. For the training set, *i* equal to 68 yielded the highest accuracy score. This indicated any SNR in decibels above 68 fitted the model the most to learn the underlying patterns in the data. **Figure 5**, **Figure 6**, **Figure 7**, and **Figure 8** filtered SNR over time plot shows the effect of applying a 68 dB SNR threshold. The testing set is prepared using the “*print_dataframe”* and “*np_arr*” functions, which likely transform the data into a format suitable for model training and prediction. After training the model on this testing set, the model's predictions are generated using “*model.predict”*. The “*classification_report*” from ***scikit-learn*** library is then used to print a detailed performance report, including **precision**, **recall**, **accuracy**, and **F1-score**. These metrics provide a comprehensive assessment of the model's performance, particularly its ability to correctly classify data in the high-SNR testing set.

Figure 5: HF_ANALOG (Noise file) for SNR in decibels >= 68

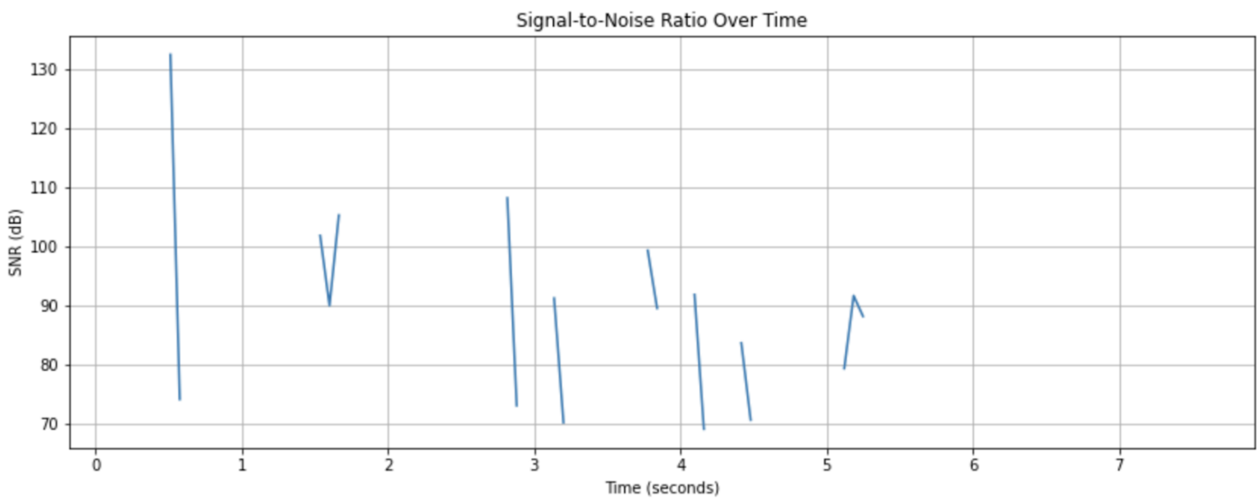

Figure 5: SNR in decibels over time windows in noise file

Figure 6: VHF_ANALOG (Noise file) for SNR in decibels >= 68

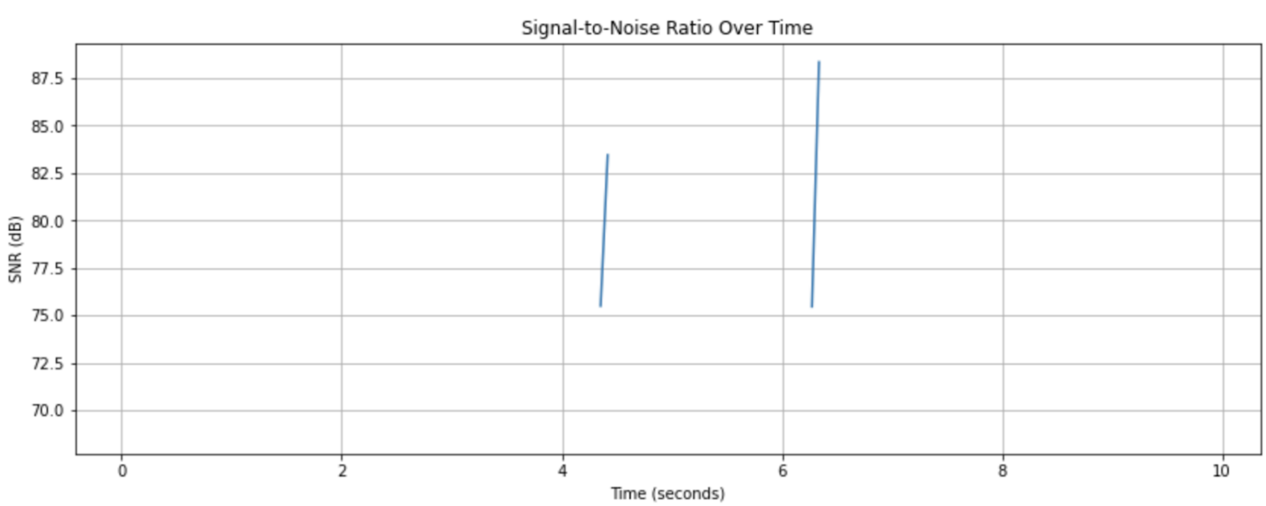

Figure 6: SNR in decibels over time windows in noise file

Figure 7: VHF_VOIP (Clean file) for SNR in decibels >= 68

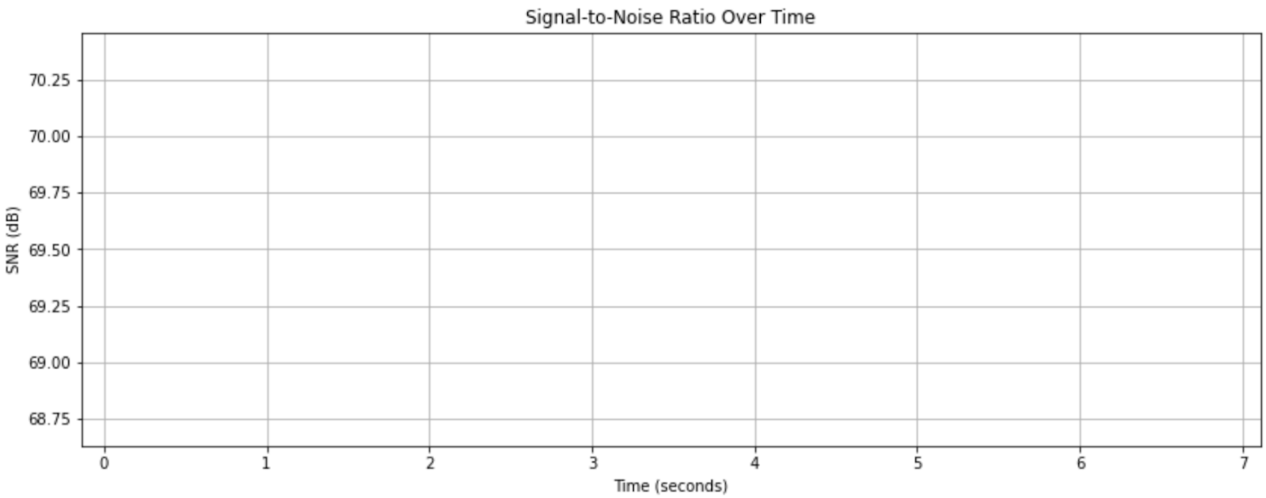

Figure 7: SNR in decibels over time windows in clean file

Figure 8: VOIP Telephone (Clean file) for SNR in decibels >= 68

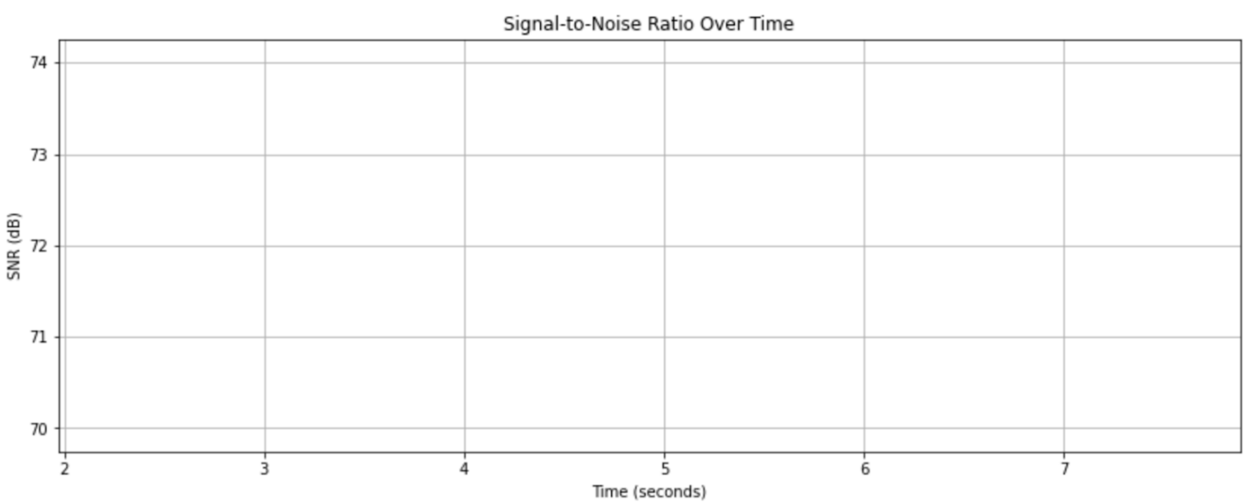

Figure 8: SNR in decibels over time windows in clean file

For the noisy signals, this removes most of the low SNR frames, highlighting the peaks where speech is present. For the clean signals, all of the frames are removed since SNR remains low throughout.

4. Results

Result 1: Model Accuracy in each constraint

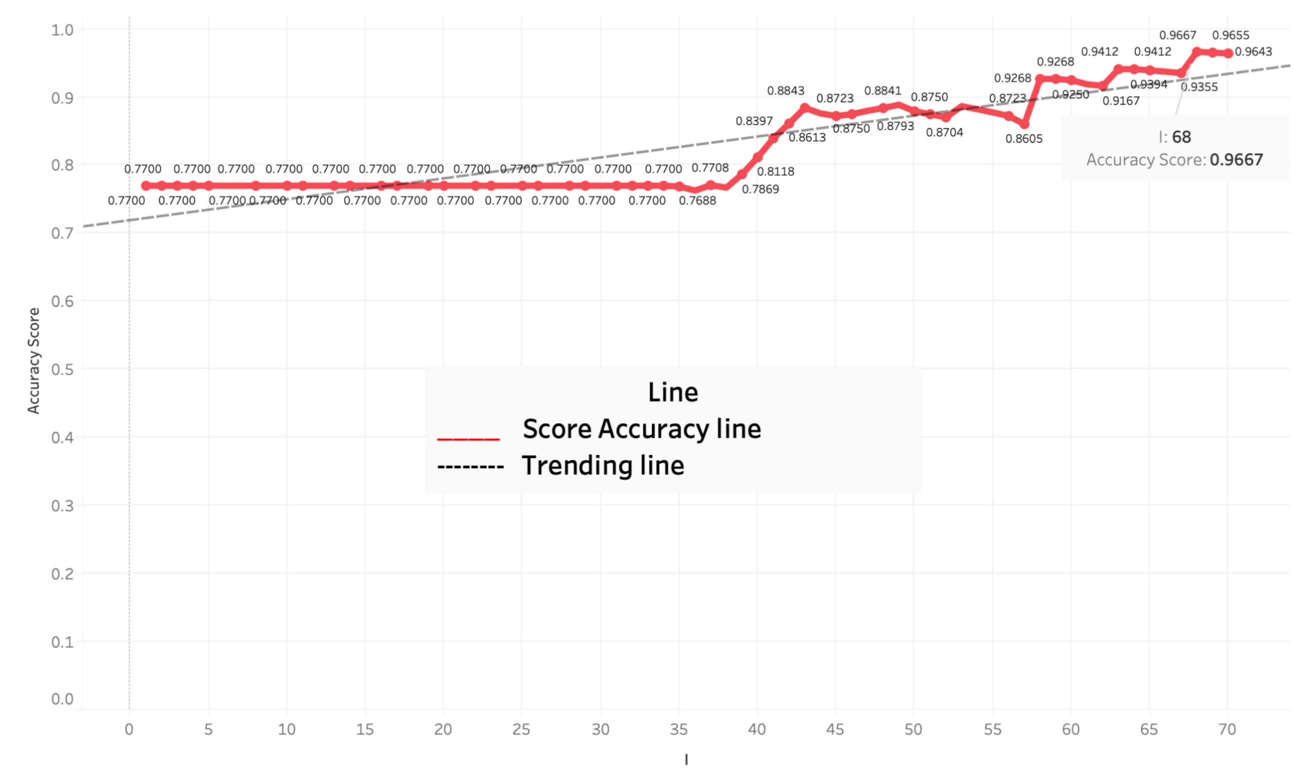

Figure 9: The line graph represents the model accuracy in each constraint

The line graph is performed from Tableau Public (v. 2022.2.0) based on the saved results Excel file "training_set.xlsx". After processing the audio data by truncating each sound clip to a duration of 10 seconds, the machine learning model demonstrated flawless performance. The reported accuracy score of 0.967 indicates that the model correctly classified most of the test samples. The logistic regression model was trained on audio frames from the HF_ANALOG (noise sound) and VHF_VOIP (clean sound) recordings. SNR values of ≥ 68 dB were extracted from these training recordings. The model effectively separated the noise and clean training examples using the SNR feature. Training accuracy plateaued at 96.7% for SNR thresholds of 68 dB and above. This indicates that a high SNR threshold captures the most informative training data for differentiating between noise and clean frames.

Result 2: Metrics Analysis to evaluate model performance

	Precision	**Recall**	**F1-score**	**Support**
0	0.00	0.00	0.00	3
1	0.67	0.86	0.75	7
Accuracy	0.60		10	

The classes are the distinct categories that the model is trying to predict. These classes are often represented by numerical labels. In this case, there are two classes: class "0" and class "1", whereas class "0" represents "**clean sound** " and class "1" represent "**noise sound**". The results indicate that the model performs significantly better in classifying instances of class '1' than class '0'. For class '1', it has a precision of 0.67, meaning that 67% of instances predicted as class '1' are correct. The recall of 0.86 for class '1' indicates that the model correctly identifies 86% of actual class '1' instances. The F1-score, which balances precision and recall, is 0.75 for class '1', suggesting a reasonably good model performance for this class. However, for class '0', the precision, recall, and F1-score are all 0.00, indicating that the model fails to correctly identify any instance of class '0'. This could suggest a bias in the model or issues in the training data, especially if the class '0' is underrepresented or more complex to classify. The overall accuracy of the model is 0.60, meaning it correctly predicts the class for 60% of the testing set. However, this figure might be misleading given the poor performance for class '0'. This suggests that while the model is somewhat effective, its utility is limited by its inability to accurately classify one of the classes.

5. Discussions

This study demonstrated the potential for using signal-to-noise ratio (SNR) features and logistic regression to detect background noise in audio recordings. SNR time series were extracted from 10-second audio clips and used to train a model to classify noisy versus clean frames. A threshold of ≥ 68 dB SNR was most effective for modeling the differences between noise and clean training data, achieving 96.7% accuracy.

When applied to test data, the model had moderate success with 60% accuracy in identifying noisy and clean recordings. Performance was better for clean frame detection than noisy frame detection. This indicates additional discriminative features could help improve generalization. The model shows promising initial results but would need refinement before being applied in a real-world noise reduction system.

This work provides a solid foundation for using machine learning to leverage differences in signal-to-noise patterns to identify background noise in the VCS. The model will be able to distinguish between audio with background noise and clean audio without noise.

6. Conclusion

The results demonstrate the promising capabilities of the logistic regression model in distinguishing between clean and noisy audio recordings, with a decent accuracy of 60% on the test set. The poor recall of 0% for the noisy class indicates the model frequently misclassifies actual noisy recordings as clean. This skew towards false negatives reveals limitations in using SNR alone as the distinguishing feature. Background noise likely shares some acoustic qualities with speech, making noisy segments difficult to identify. Incorporating additional audio features could enhance the separation between the classes. Characteristics like spectral centroid, zero-crossing rate, spectral roll-off, and Mel frequency cepstral coefficients can also be tested for the logistic regression model instead of the SNR measurements.

7. Reference & Bibliography

Logistic regression. Logistic Regression - an overview | ScienceDirect Topics. (n.d.). https://www.sciencedirect.com/topics/computer-science/logistic-regression

Accuracy vs. precision vs. recall in machine learning: What's the difference? Evidently AI - Open-Source ML Monitoring and Observability. (n.d.). https://www.evidentlyai.com/classification-metrics/accuracy-precision-recall

Understanding and applying F1 score: A deep dive with hands-on coding. Arize AI. (2023, June 12). https://arize.com/blog-course/f1-score

Prinzo, O. V., & Civil Aeromedical Institute. (1998, May 1). *An analysis of voice communication in a simulated approach control environment: Final report*. Welcome to ROSA P. https://rosap.ntl.bts.gov/view/dot/21476

Voice Communication Control System Market Report, 2030. (n.d.). https://www.grandviewresearch.com/industry-analysis/voice-communication-control-system-market-report

Analog vs Digital Radios: Which one is best?. VEI Communications. (2023, May 31). https://www.veicomm.com/analog-vs-digital-radios-which-one-is-best/

What is (RoIP) radio over IP and (voip) voice over ip?. Tait Communications Blog What is RoIP Radio over IP and VoIP Voice over IP Comments. (n.d.). https://blog.taitcommunications.com/2014/10/28/what-is-roip-radio-over-ip-and-voip-voice-over-ip/

Johnson, D. H. (2006). Signal-to-noise ratio. *Scholarpedia*, *1*(12), 2088.

Signal-to-noise ratio. Signal-to-Noise Ratio - an overview | ScienceDirect Topics. (n.d.). https://www.sciencedirect.com/topics/engineering/signal-to-noise-ratio

Improving communications between pilots and air traffic controllers. Improving Communications Between Pilots and Air Traffic Controllers | V olpe National Transportation Systems Center. (n.d.). https://www.volpe.dot.gov/safety-management-and-human-factors/improving-communication-between-pilots-and-controllers

VE6EY, J. (2017, June 6). *Signal-noise ratio - the essence of Radio*. Making It Up. https://play.fallows.ca/wp/radio/ham-radio/signal-noise-ratio-essence-radio/

Voice over internet protocol (VoIP). Federal Communications Commission. (n.d.). https://www.fcc.gov/general/voice-over-internet-protocol-voip

Numpy.hanning#. numpy.hanning - NumPy v1.26 Manual. (n.d.). https://numpy.org/doc/stable/reference/generated/numpy.hanning.html#numpy-hanning

Github link:
"https://github.com/dangminh0912/Regression-Model-in-shaping-the-Voice-Communication-Control/tree/main"