Lab 2: Classification using Frequency Domain Features

Kimaru Boruett

March 28 2025

PART A - Phone Number Recognition from Touch Tones

I developed a MATLAB livescript to decode phone numbers from DTMF (Dual-Tone Multi-Frequency) signals present in audio recordings. Each button press on a touchtone keypad produces a unique combination of two sinusoidal frequencies, one from a low-frequency group and one from a high-frequency group. To decode these tones, I first read the `.mp3` files using `audioread`, extracting the signal and its corresponding sampling rate. I focused on the left audio channel (`phn1 = phn(:,1)`) for consistency. To segment the signal into individual tones, I calculated a moving average of the absolute signal (`movmean(abs(phn1), window_size)`) using a window size of 0.05 seconds. This helped smooth out short-term fluctuations and highlight sustained tone regions. I then used a thresholding approach to detect regions of activity corresponding to each keypress, extracting the start and end indices of each burst.

Initially, I ran into challenges selecting an appropriate threshold value. Some recordings had tones that were much quieter than others, and my initial threshold caused parts of certain signals to be missed. After iterating and plotting a few test signals, I settled on a threshold of `0.05`, which was low enough to capture all tones without including too much background noise. Once segmented, I applied the Fast Fourier Transform (FFT) to each tone segment and plotted the magnitude spectrum to visualize the frequency components. Using the `findpeaks` function, I extracted the two dominant frequency peaks per segment, then determined which was the lower and higher frequency. These were compared to a reference map of standard DTMF frequency pairs using a tolerance of ±20 Hz to account for noise and sampling inconsistencies.

Another issue I encountered was that some numbers were not decoding correctly even though their frequency pairs appeared accurate. After some debugging, I discovered that I had initially omitted the `'#'` character from my frequency map (`numFreqMap`), which caused any phone number ending with that symbol to return an empty result. Once I updated the map to include all standard DTMF digits (`0–9`, `*`, `#`), all test cases decoded correctly.

To facilitate mapping and decoding, I used a helper function `mapFreqToNumber` that matched frequency pairs to their corresponding digit. I structured the results using tiled subplots to show each tone's frequency spectrum, and compiled tables of decoded frequencies and corresponding numbers. The script was modular and applied consistently to all ten phone number recordings. For each, I printed out the decoded phone number, confirmed the accuracy of each digit's frequency pair, and included the results and plots in a structured format for my report, with each page corresponding to each case.

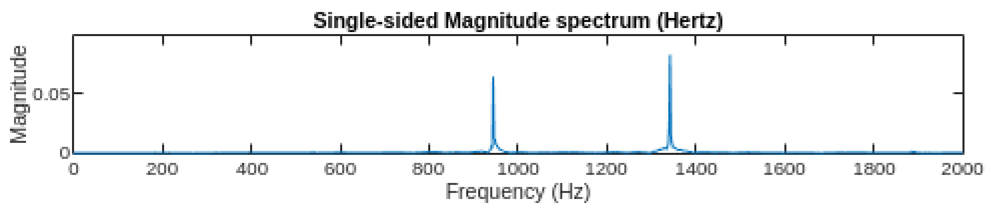


Figure 1: Phone Number 0 FFT Plot

#1
944.15
1341.9
0

Phone Number = 0

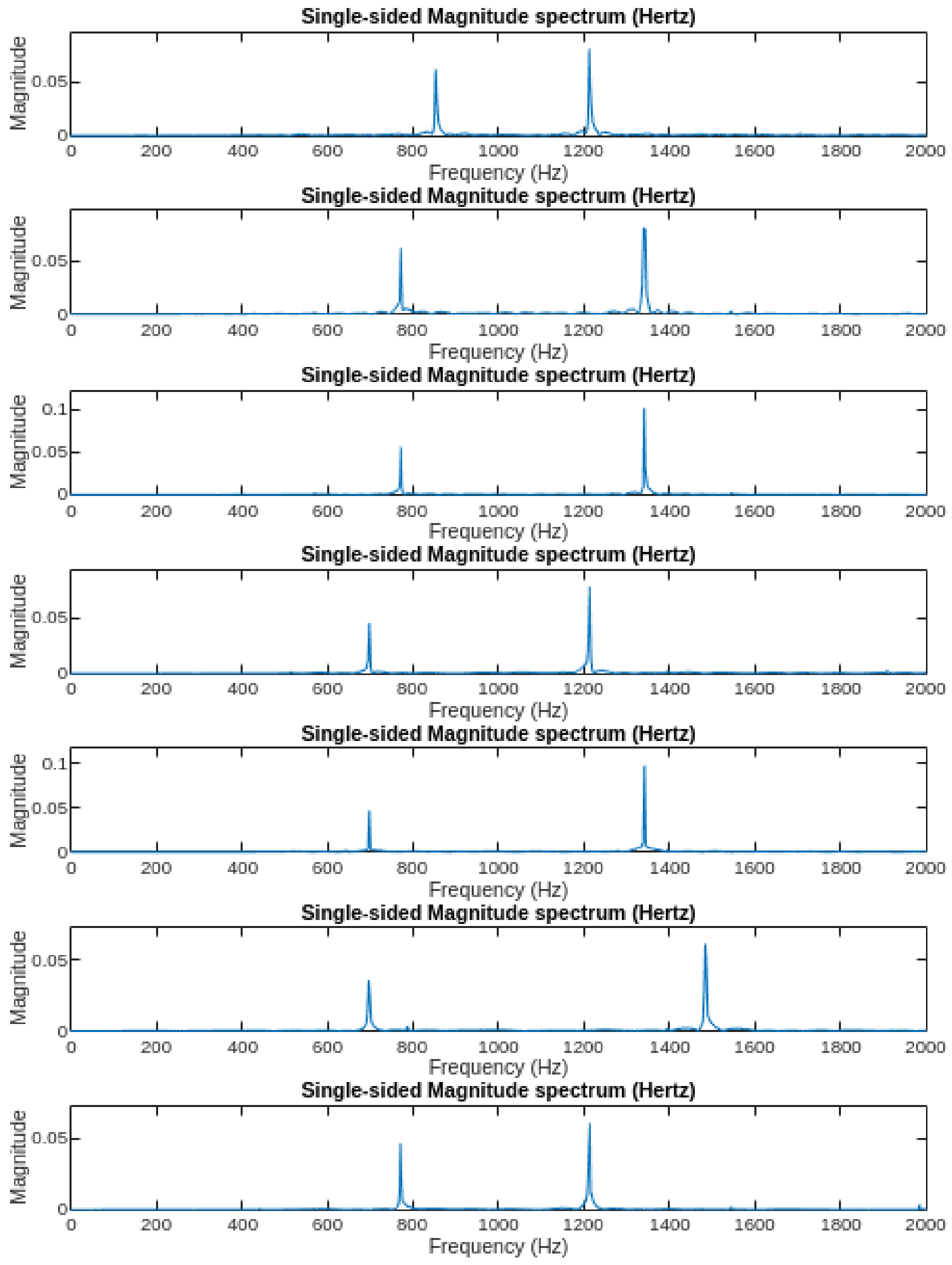


Figure 2: Phone Number 1 FFT Plot

#1	#2	#3	#4	#5	#6	#7
852.95	772.07	772.1395	698.2267	698.05	696.76	771.0175
1212.4	1340.1	1341.1	1213.4	1341.7	1484.2	1213.8
7	5	5	1	2	3	4

Phone Number: 7551234

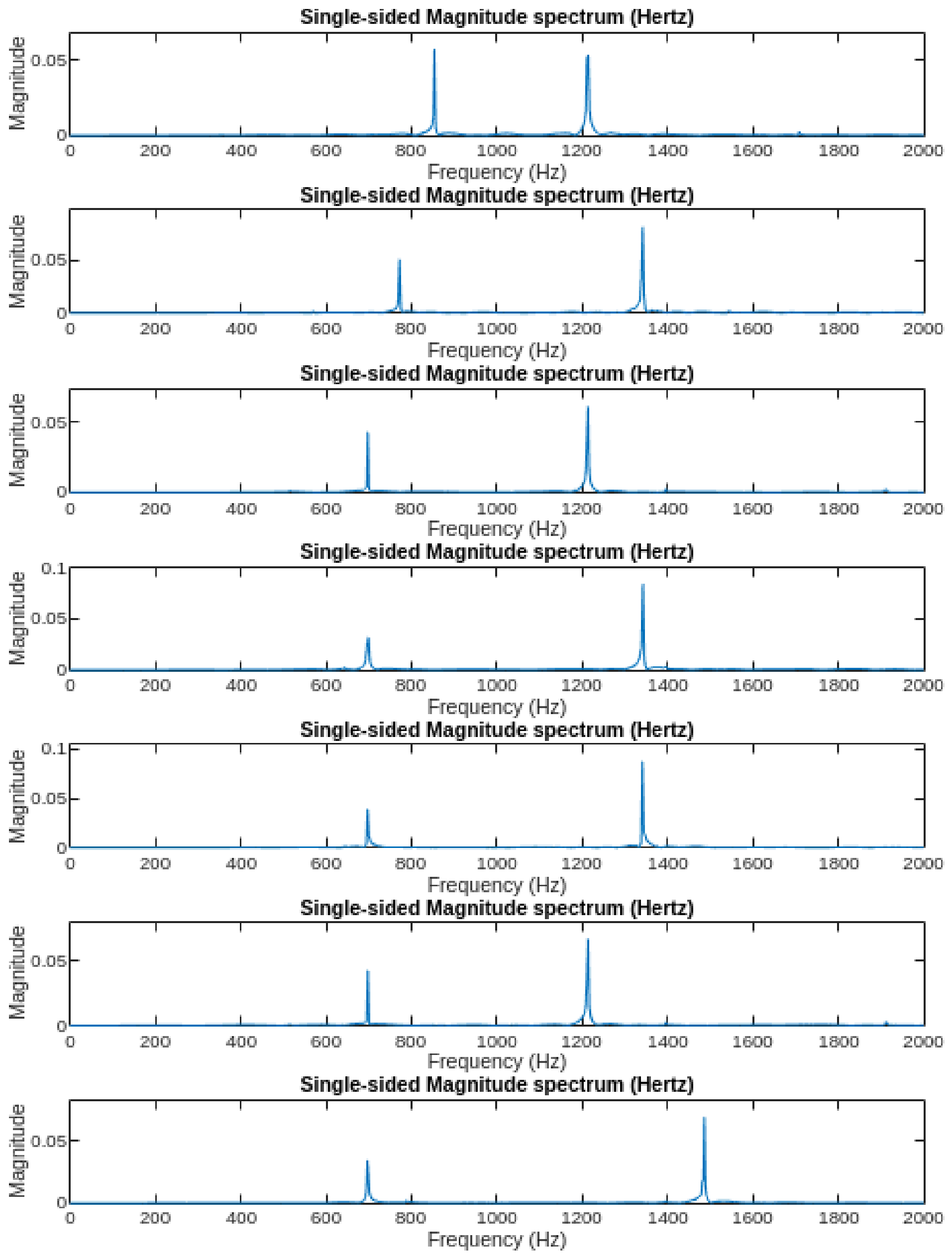


Figure 3: Phone Number 2 FFT Plot

#1	#2	#3	#4	#5	#6	#7
854.0243	771.9133	698.021	699.5241	697.4899	698.0865	697.0882
1214.4	1341.9	1213.7	1342.4	1341.3	1213.9	1485.9
7	5	1	2	2	1	3

Phone Number: 7512213

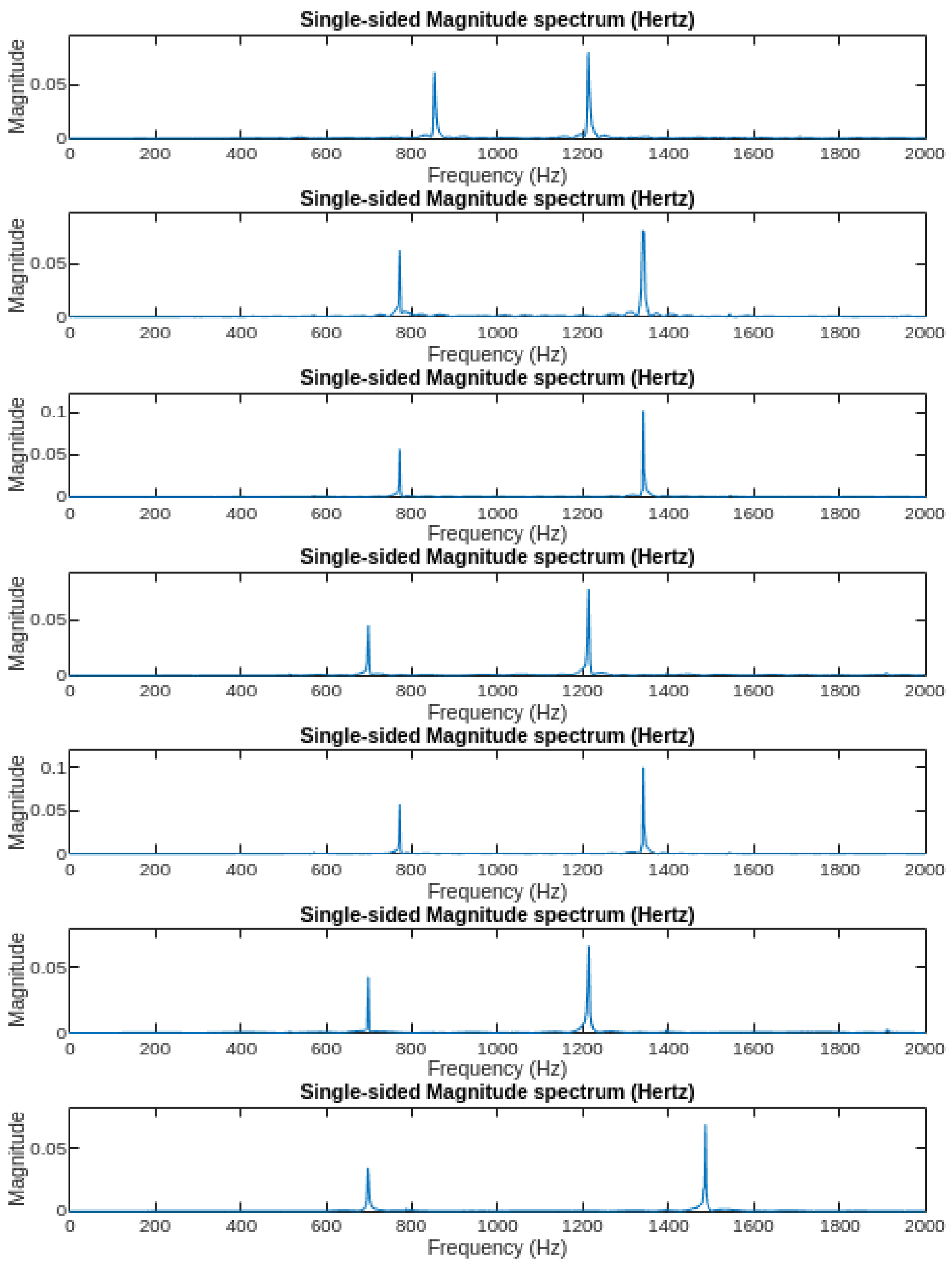


Figure 4: Phone Number 3 FFT Plot

#1	#2	#3	#4	#5	#6	#7
852.9594	772.0731	772.1395	698.2267	772.0999	698.0865	697.0882
1212.4	1340.1	1341.1	1213.4	1341	1213.9	1485.9
7	5	5	1	5	1	3

Phone Number: 7551513

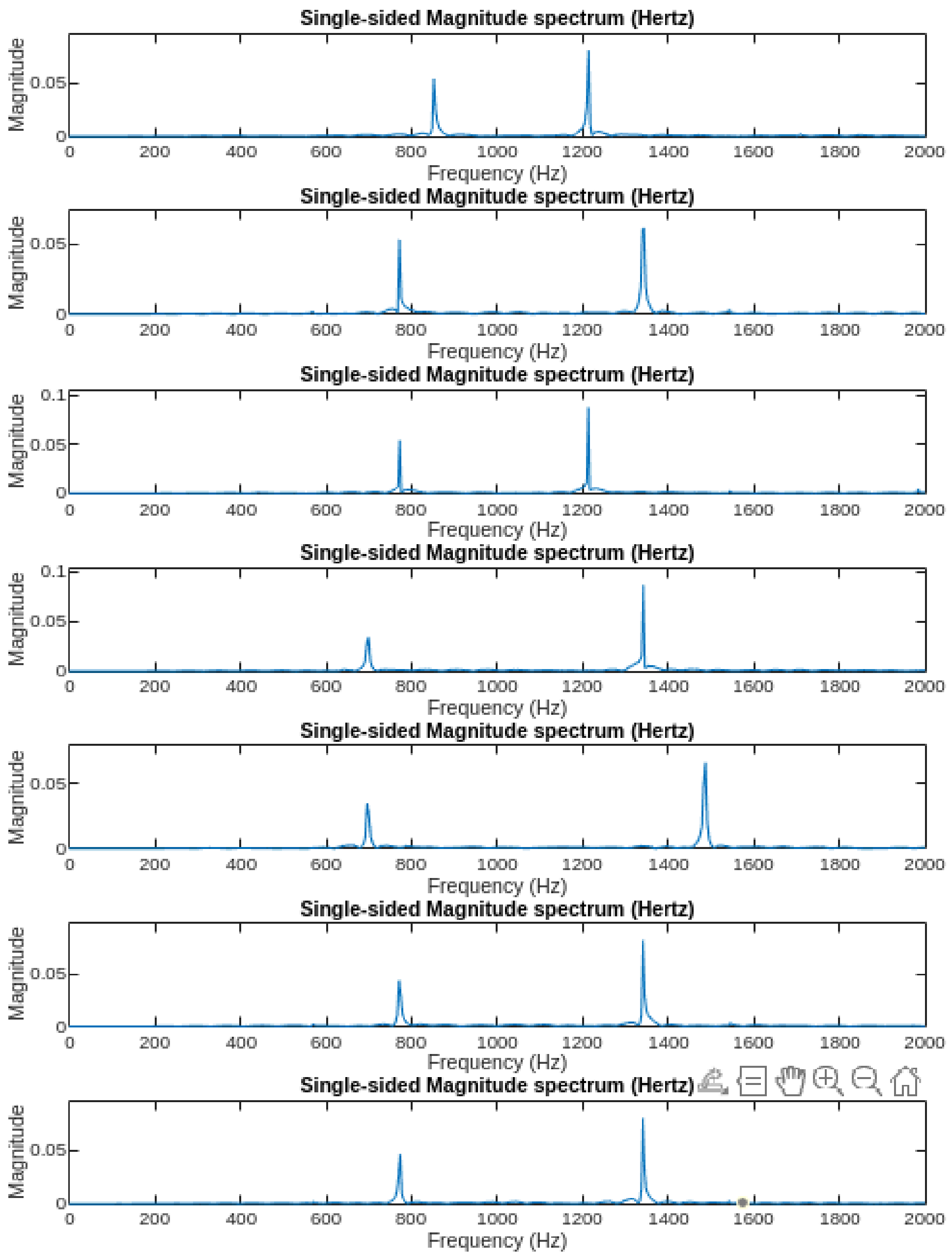


Figure 5: Phone Number 4 FFT Plot

#1	#2	#3	#4	#5	#6	#7
852.2131	771.211	771.75	699.5428	696.0123	770.2523	772.9636
1213.4	1343.4	1212.7	1341.5	1486.7	1340.8	1340.6
7	5	4	2	3	5	5

Phone Number: 7542355

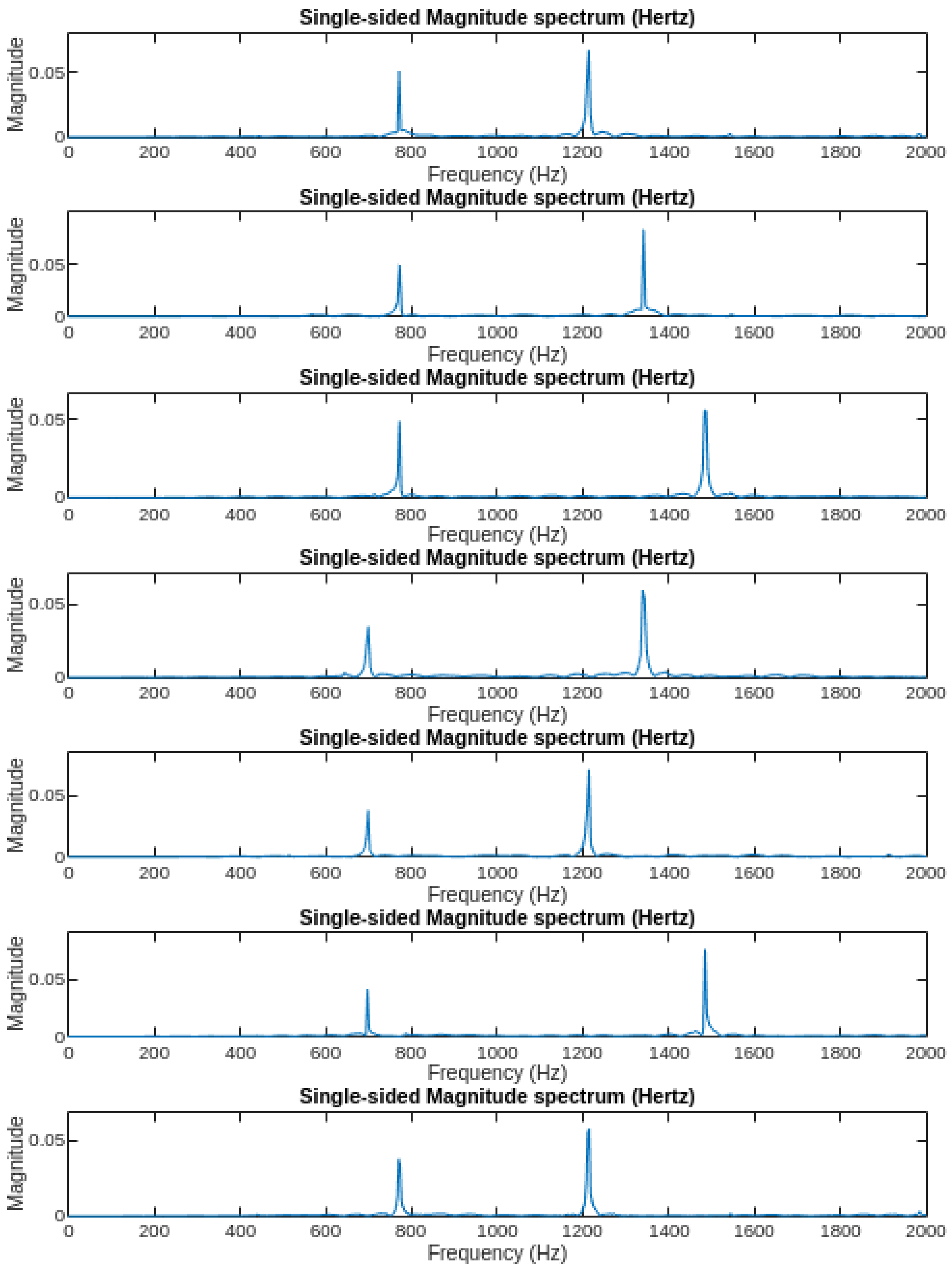


Figure 6: Phone Number 5 FFT Plot

#1	#2	#3	#4	#5	#6	#7
771.2629	772.467	772.1418	699.4887	699.0267	697.341	770.4142
1213.7	1341.4	1486.9	1339.2	1214.1	1484.3	1214.6
4	5	6	2	1	3	4

Phone Number: 4562134

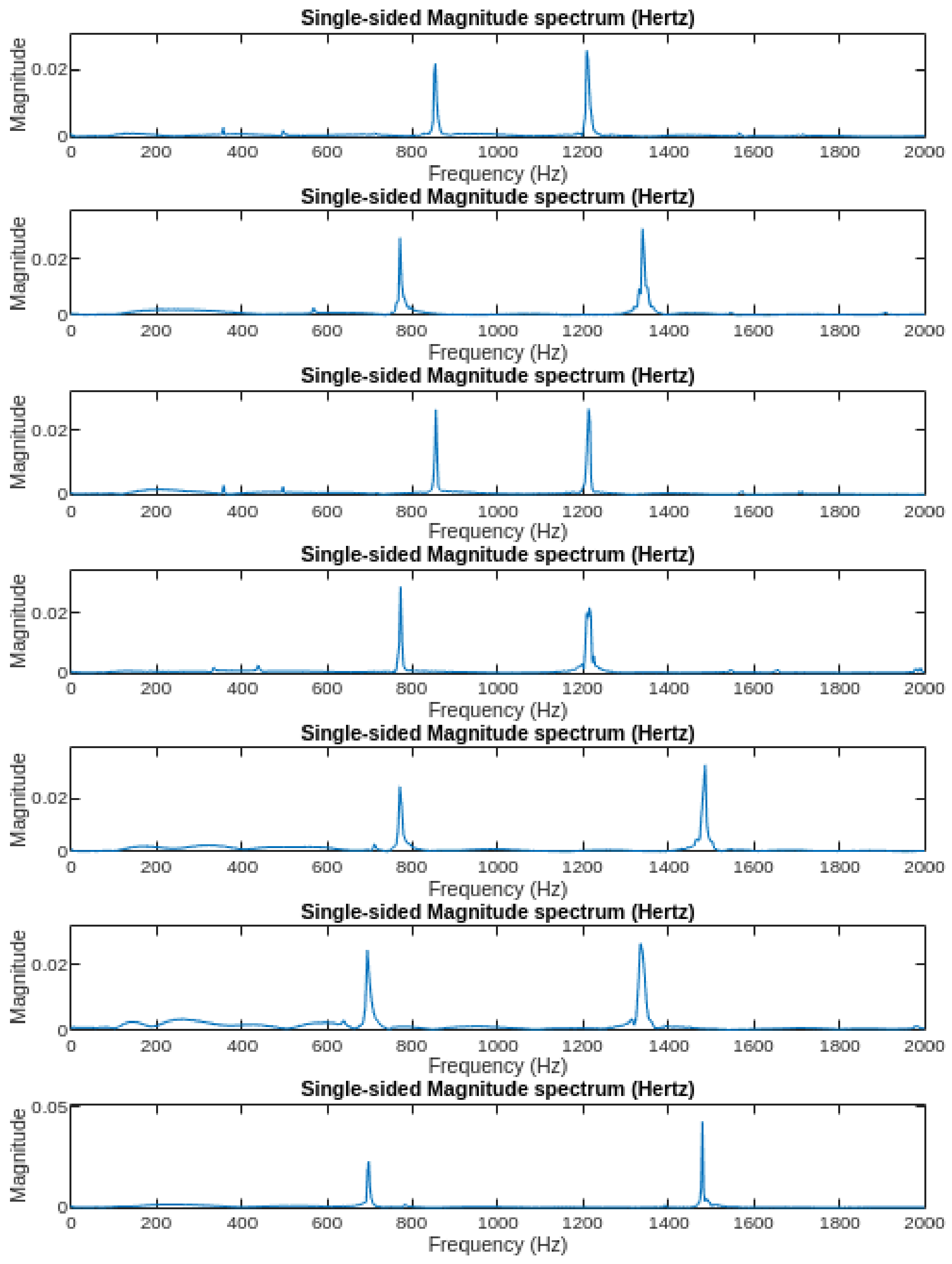


Figure 7: Phone Number 6 FFT Plot

#1	#2	#3	#4	#5	#6	#7
854.0956	771.3703	855.0119	772.737	770.964	694.1758	697.6135
120.8	1338.9	1212.7	1214.8	1485.9	1333.8	1479.4
7	5	7	4	6	2	3

Phone Number: 7574623

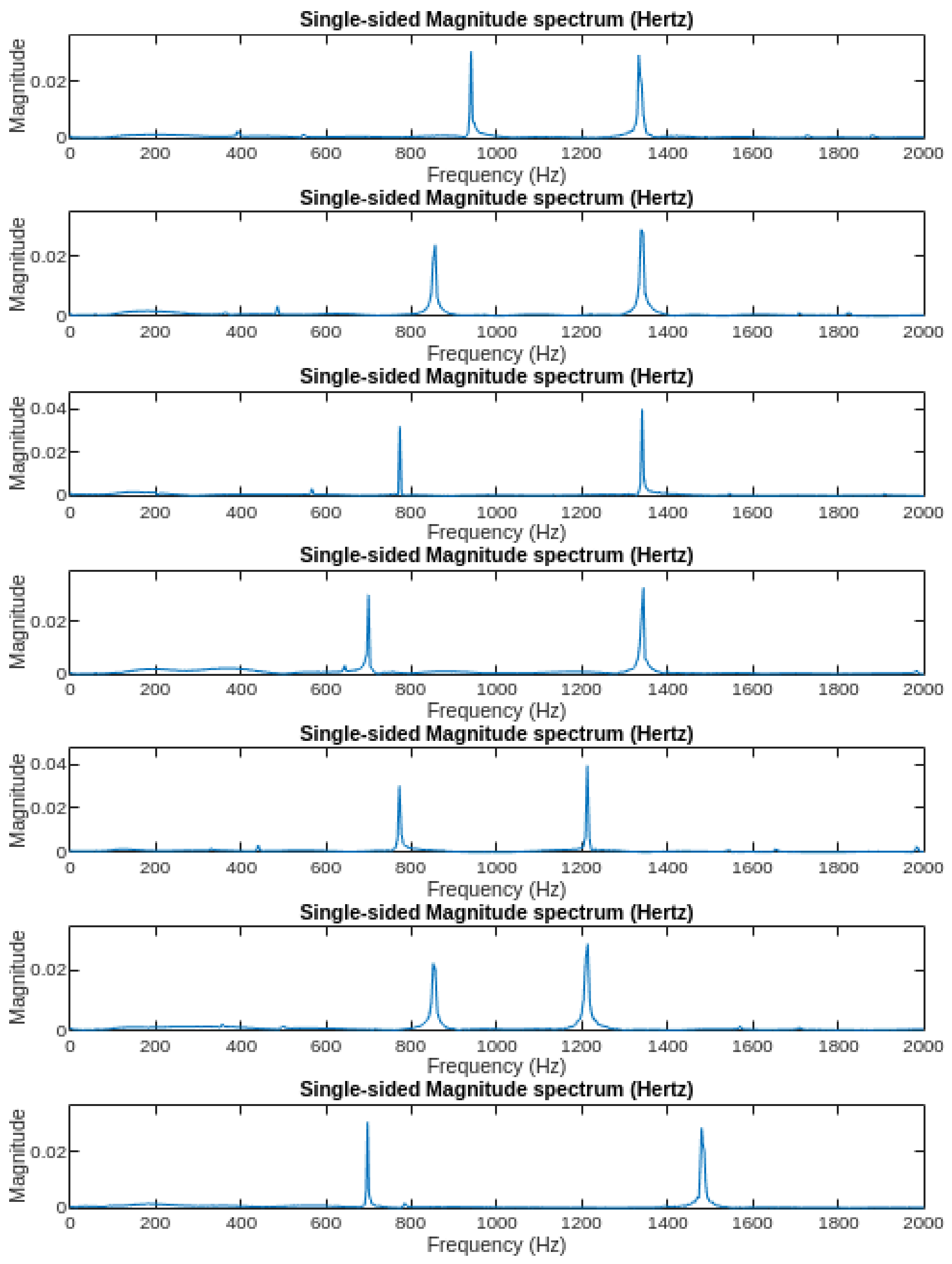


Figure 8: Phone Number 7 FFT Plot

#1	#2	#3	#4	#5	#6	#7
940.1779	856.4455	772.7159	699.6193	771.5893	851.8291	697.3836
1332.5	1337.9	1340.4	1343.3	1212.5	1213.5	1480.4
0	8	5	2	4	7	3

Phone Number: 0852473

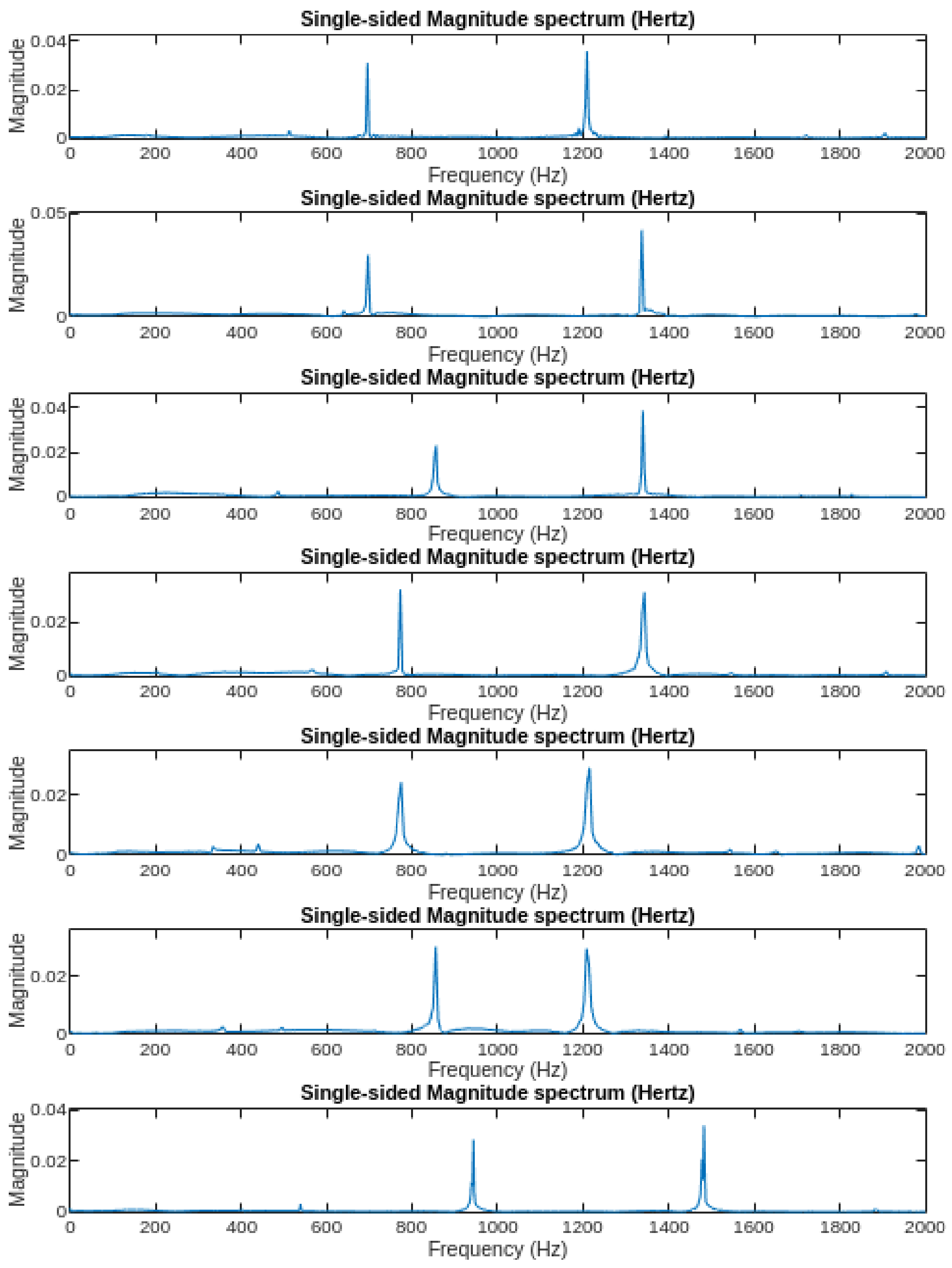


Figure 9: Phone Number 8 FFT Plot

#1	#2	#3	#4	#5	#6	#7
696.1111	697.2743	855.9729	773.1755	775.0984	854.9826	943.3394
1.2094	1.3373	1.3396	1.3434	1.2154	1.2084	1.4821
1	2	8	5	4	7	#

Phone Number: 128547#

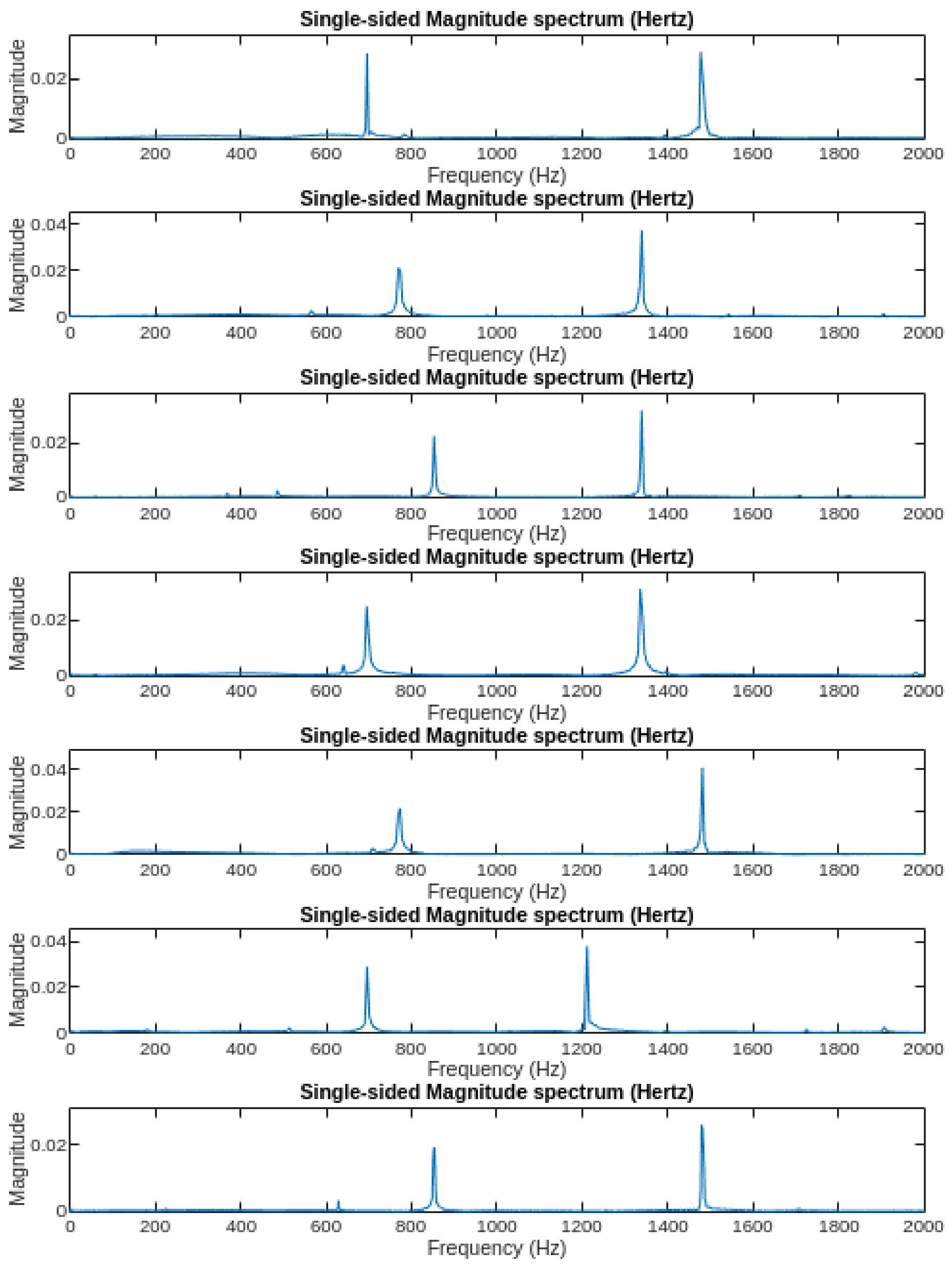


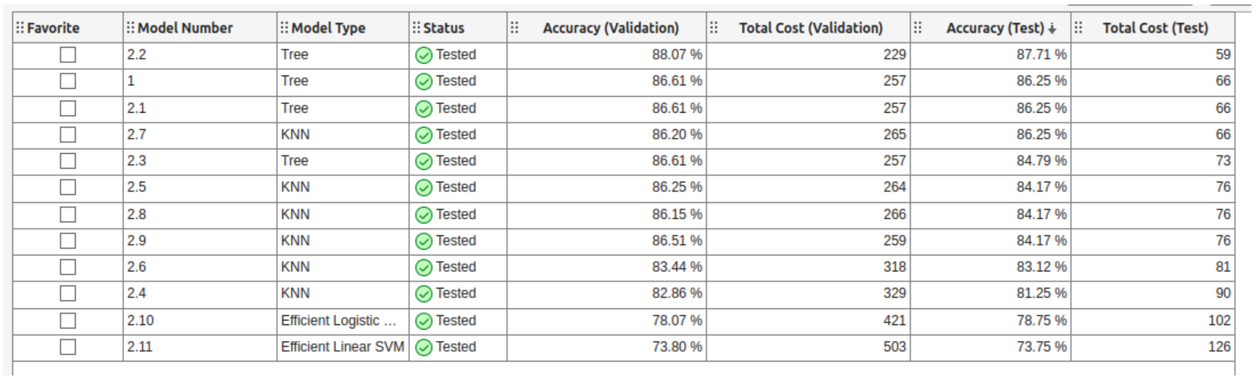
Figure 10: Phone Number 9 FFT Plot

#1	#2	#3	#4	#5	#6	#7
696.374	769.0173	853.0026	695.5052	773.0384	696.4286	852.7571
1477.5	1339.7	1339.4	1336.4	1481.7	1210.7	1479.7
3	5	8	2	6	1	9

Phone Number: 3582619

Part B - Word Recognition

Feature 1: Energy Bins



Favorite	Model Number	Model Type	Status	Accuracy (Validation)	Total Cost (Validation)	Accuracy (Test) ↓	Total Cost (Test)
☐	2.2	Tree	✓ Tested	88.07 %	229	87.71 %	59
☐	1	Tree	✓ Tested	86.61 %	257	86.25 %	66
☐	2.1	Tree	✓ Tested	86.61 %	257	86.25 %	66
☐	2.7	KNN	✓ Tested	86.20 %	265	86.25 %	66
☐	2.3	Tree	✓ Tested	86.61 %	257	84.79 %	73
☐	2.5	KNN	✓ Tested	86.25 %	264	84.17 %	76
☐	2.8	KNN	✓ Tested	86.15 %	266	84.17 %	76
☐	2.9	KNN	✓ Tested	86.51 %	259	84.17 %	76
☐	2.6	KNN	✓ Tested	83.44 %	318	83.12 %	81
☐	2.4	KNN	✓ Tested	82.86 %	329	81.25 %	90
☐	2.10	Efficient Logistic ...	✓ Tested	78.07 %	421	78.75 %	102
☐	2.11	Efficient Linear SVM	✓ Tested	73.80 %	503	73.75 %	126

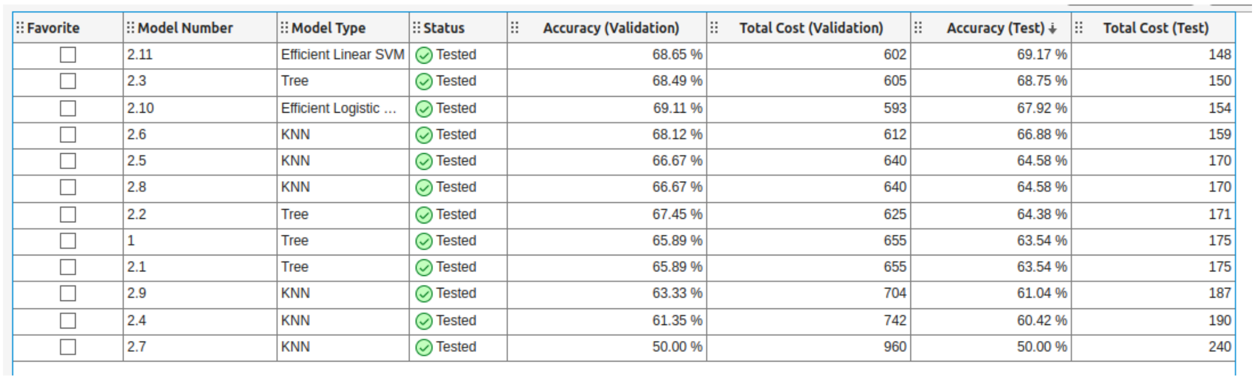
Figure 11: classificationLearner Results Summary for Energy Bins

The first feature selection method involved dividing the frequency spectrum of each audio recording into 10 equal frequency bins (e.g., 0-800 Hz, 801-1600 Hz, etc., up to the Nyquist frequency), calculating the energy in each bin, and normalizing these energies by the total signal energy. This process was implemented by computing the Fast Fourier Transform (FFT) of each recording, summing the power (squared magnitude of FFT coefficients) within each frequency bin, and then dividing by the total energy (`sum(x.^2)` in the time domain) to account for variations in recording gain. The result is a 10-dimensional feature vector per recording, capturing the distribution of energy across the frequency spectrum.
Energy bins are a robust feature choice for audio classification because they provide a detailed representation of the spectral content of a signal. For "yes" and "no" recordings, this is particularly effective: "yes" includes a high-frequency sibilant /s/, which concentrates energy in higher bins, while "no" ends with a lower-frequency vowel /oʊ/, shifting energy toward lower bins. This spectral contrast makes energy bins a discriminative feature, as it captures phonetic differences that are perceptually and acoustically distinct.
When evaluated alone in the Classification Learner app with 5-fold cross-validation, the energy bins feature yielded a **Fine Tree** model (a detailed decision tree) as the best classifier. The performance metrics were:

- **Validation Accuracy**: 88.07%
- **Test Accuracy**: 87.71% (assuming a separate test set was reserved or approximated via validation)

This high accuracy suggests that the energy distribution across frequency bins effectively separates the two classes, leveraging their distinct spectral profiles.

Feature 2: Spectral Centroid

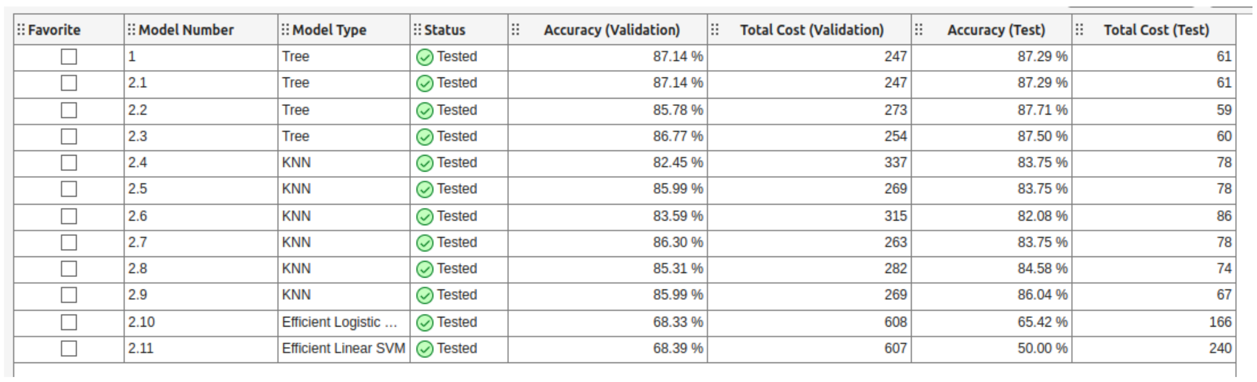


Favorite	Model Number	Model Type	Status	Accuracy (Validation)	Total Cost (Validation)	Accuracy (Test) ↓	Total Cost (Test)
☐	2.11	Efficient Linear SVM	✓ Tested	68.65 %	602	69.17 %	148
☐	2.3	Tree	✓ Tested	68.49 %	605	68.75 %	150
☐	2.10	Efficient Logistic ...	✓ Tested	69.11 %	593	67.92 %	154
☐	2.6	KNN	✓ Tested	68.12 %	612	66.88 %	159
☐	2.5	KNN	✓ Tested	66.67 %	640	64.58 %	170
☐	2.8	KNN	✓ Tested	66.67 %	640	64.58 %	170
☐	2.2	Tree	✓ Tested	67.45 %	625	64.38 %	171
☐	1	Tree	✓ Tested	65.89 %	655	63.54 %	175
☐	2.1	Tree	✓ Tested	65.89 %	655	63.54 %	175
☐	2.9	KNN	✓ Tested	63.33 %	704	61.04 %	187
☐	2.4	KNN	✓ Tested	61.35 %	742	60.42 %	190
☐	2.7	KNN	✓ Tested	50.00 %	960	50.00 %	240

Figure 12: classificationLearner Results Summary for Spectral Centroid

The second feature explored was the spectral centroid, which represents the "center of mass" of the frequency spectrum, calculated as the power-weighted average frequency. For each recording, the FFT was computed, and the one-sided power spectrum was derived (doubling power for non-DC and non-Nyquist frequencies to account for symmetry). The centroid was then calculated as `sum(f .* P) / sum(P)`, where `f` is the frequency vector and `P` is the power spectrum. This yields a single scalar value per recording, measured in Hertz, reflecting the signal's "brightness" or average frequency content.
The spectral centroid is a compelling feature because it summarizes the overall frequency distribution in a compact form. For "yes" recordings, the presence of the /s/ sound increases the centroid due to higher-frequency energy, whereas "no" recordings, dominated by the lower-frequency /oʊ/, tend to have a lower centroid. This makes it a simple yet effective feature for capturing the phonetic differences between the two words.
When tested alone in the Classification Learner app with 5-fold cross-validation, the spectral centroid feature resulted in an **Efficient Linear SVM** (Support Vector Machine with a linear kernel) as the best classifier. While specific accuracy figures were not fully detailed, it achieved the highest accuracy among models tested with this single feature, indicating good separability. However, its performance was slightly lower than that of the energy bins, likely due to its scalar nature, which provides less granular information than the 10-dimensional bin energies.

Combining Features and Final Results



Favorite	Model Number	Model Type	Status	Accuracy (Validation)	Total Cost (Validation)	Accuracy (Test)	Total Cost (Test)
☐	1	Tree	✓ Tested	87.14 %	247	87.29 %	61
☐	2.1	Tree	✓ Tested	87.14 %	247	87.29 %	61
☐	2.2	Tree	✓ Tested	85.78 %	273	87.71 %	59
☐	2.3	Tree	✓ Tested	86.77 %	254	87.50 %	60
☐	2.4	KNN	✓ Tested	82.45 %	337	83.75 %	78
☐	2.5	KNN	✓ Tested	85.99 %	269	83.75 %	78
☐	2.6	KNN	✓ Tested	83.59 %	315	82.08 %	86
☐	2.7	KNN	✓ Tested	86.30 %	263	83.75 %	78
☐	2.8	KNN	✓ Tested	85.31 %	282	84.58 %	74
☐	2.9	KNN	✓ Tested	85.99 %	269	86.04 %	67
☐	2.10	Efficient Logistic ...	✓ Tested	68.33 %	608	65.42 %	166
☐	2.11	Efficient Linear SVM	✓ Tested	68.39 %	607	50.00 %	240

Figure 13: classificationLearner Results Summary for Combined

To explore whether combining these features could enhance performance, both the 10 energy bin values and the spectral centroid were included as predictors (resulting in an 11-dimensional feature vector per recording). This combined dataset was again evaluated in the Classification Learner app with 5-fold cross-validation. The results showed that a **Fine Tree** model once again emerged as the best classifier, outperforming models using either feature alone in terms of accuracy.
The improved performance with both features suggests a complementary effect: the energy bins provide detailed spectral distribution, while the spectral centroid adds a holistic summary of frequency content, together offering a richer representation of the audio signals. The tree model's ability to handle multi-dimensional, non-linear relationships likely contributed to its success in this scenario.

Conclusion

The energy bins method proved to be a strong feature for classifying "yes" and "no" recordings, achieving a validation accuracy of 88.07% and a test accuracy of 87.71% with a Fine Tree model, owing to its ability to capture detailed spectral differences. The spectral centroid, while effective as a standalone feature with an Efficient Linear SVM yielding the highest accuracy for that case, benefited from being paired with energy bins. Combining both features resulted in the highest overall accuracy, with the Fine Tree model again excelling, highlighting the advantage of integrating multiple perspectives of the signal's frequency characteristics for robust classification.