Classifying Spoken Arabic Digits

Ashley Hong

BACKGROUND

Applications of Speech Recognition

From Internet of Things (IoT) data to mobile data to health data to social media data, etc., machine learning models can access and aggregate data from nearly any dataset to create inferences and draw insights[1]. The increase in readily accessible data has led applications of Machine Learning to boom in recent years with significant advancements in many related topics such as speech recognition models. The ability to classify spoken words correctly and efficiently has the potential to free the world of the constraints of language barriers. This can facilitate collaboration between all parts of the world, increase the sense of global community, and increase accessibility across various domains.

The potential applications of speech recognition are vast, and one important use case is in healthcare. For those that are lucky enough to live a region where their language is the primary spoken language, this may be a distant after thought, but for others it can be difficult, even impossible, to properly communicate their health issues to doctors or understand the doctor's diagnoses. It is clear where speech recognition tools might fit into the solution for overcoming these language barriers. In one case, researchers performed a crossover study between 12 French speaking doctors at Geneva University Hospitals and 2 Arabic speaking patients to examine the efficacy of a speech-enabled fixed-phrase translator known as BabelDr[2]. It performed well enough that all doctors were able to reach the correct diagnosis for both patients, who were only given a pre-written set of symptoms. However, there are still concerns about the reliability and data confidentiality of machine speech translators, so there is more work to be done in the use of speech recognition for highly sensitive applications.

This speech recognition project uses Bayesian Modeling and Maximum Likelihood classification, which are both common machine learning approaches that can be applied to many scenarios. One such example that interests me is the use of Bayesian Networks in diagnosing lung cancer[3]. Currently, the primary way to diagnose lung cancer is through CT screens, which are expensive and occasionally unreliable. Bayesian Networks can be used as a supplemental tool that increases the accuracy of diagnosis and minimizes additional costs incurred by CT screen errors.

[1]Sarker, I.H. Machine Learning: Algorithms, Real-World Applications and Research Directions. SN COMPUT. SCI. 2, 160 (2021). https://doi.org/10.1007/s42979-021-00592-x

[2]Spechbach H, Gerlach J, Mazouri Karker S, Tsourakis N, Combescure C, Bouillon P A Speech-Enabled Fixed-Phrase Translator for Emergency Settings: Crossover Study

[3]M. Admane and S. Patil, "Modeling Lung Cancer Diagnosis using Bayesian Network Inference," 2022 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON), Bangalore, India, 2022, pp. 1-6, doi: 10.1109/SMARTGENCON56628.2022.10084230.

Key Terminology

Before reading further, it is fundamental to understand a couple terms that are used to describe the data: Mel-Frequency Cepstral Coefficients (MFCCs) and analysis windows.

1. Each utterance of a digit is represented by a matrix of MFCCs. Each row of MFCCs is considered an analysis window. Typically, a long, continuous audio signal is broken down into short-duration segments known as analysis windows such that each frame can be processed individually.

2. MFCCs hold information of various features that can be used to characterize audio signals, which makes it a great data type for speech recognition. To get MFCCs from a signal, the linear frequency scale is first transformed to the Mel-Frequency scale, which approximates how humans perceive pitch. Then, the coefficients can be extracted from each analysis window by performing cepstral analysis[1].

[1]A. V. Oppenheim and R. W. Schafer, "From frequency to quefrency: a history of the cepstrum," in IEEE Signal Processing Magazine, vol. 21, no. 5, pp. 95-106, Sept. 2004, doi: 10.1109/MSP.2004.1328092.

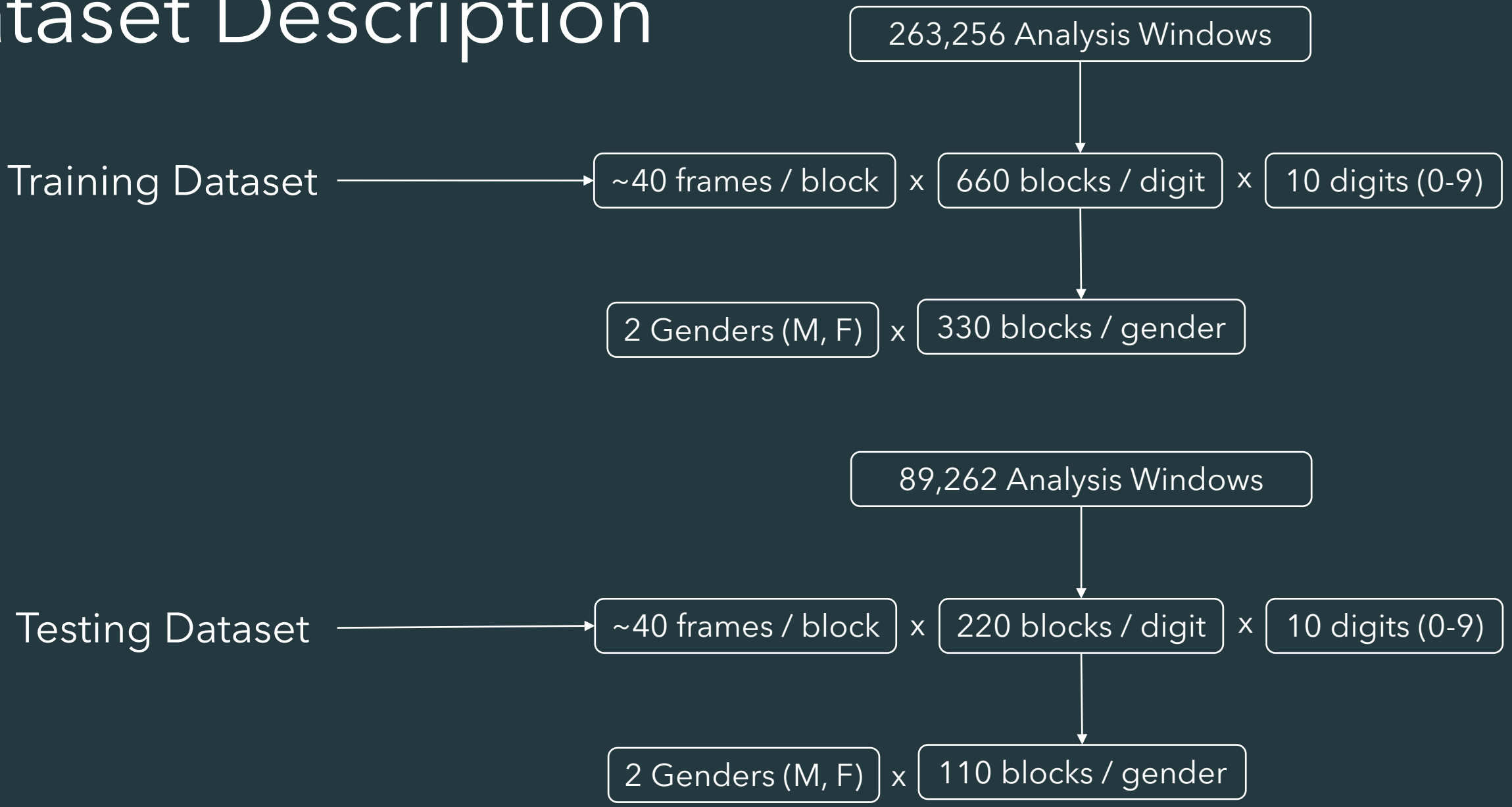
Problem Description

The goal of this project is to create a model that can classify spoken instances of the Arabic digits 0-9 from diverse sources. While it is unlikely that the model will be perfect, it should be greater than 10% accuracy, which is the approximate performance of a model that completely guesses the classification of an utterance. A personal target is to achieve at least 80% average accuracy. A training and testing dataset in the form of Mel-Frequency Coefficients (MFCCs) were provided through the UCI Machine Learning Repository[1]. For the purposes of this project, the model is represented as a Gaussian Mixture Model (GMM) for each digit given the simplicity of grouping MFCCs into distinct sounds, or phonemes. A phoneme is a distinct sound that occurs when speaking a digit out loud. The phonetic pronunciation of each of the digits used in this model are:

0: sifir	**2: ithnayn**	**4: araba'a**	**6: sittah**	**8: thamanieh**
1: wahad	**3: thalatha**	**5: khamsa**	**7: seb'a**	**9: tis'ah**

[1]Bedda,Mouldi and Hammami,Nacereddine. (2010). Spoken Arabic Digit. UCI Machine Learning Repository. https://doi.org/10.24432/C52C9Q.

Dataset Description



MODEL EXPLORATION

Aside: Accuracy and Precision

To measure the performance of various hyperparameters and evaluate the results, the primary measurements were accuracy and precision. The following are the equations for calculating both and a graphic that displays how the variables in the equations relate to the confusion matrices.

Accuracy= $\frac{TP+TN}{TP+FN+TN+FP}$

Precision= $\frac{TP}{TP+FP}$

Where…

- TP = True positives
- TN = True Negatives
- FN = False Negatives
- FP = False Positives

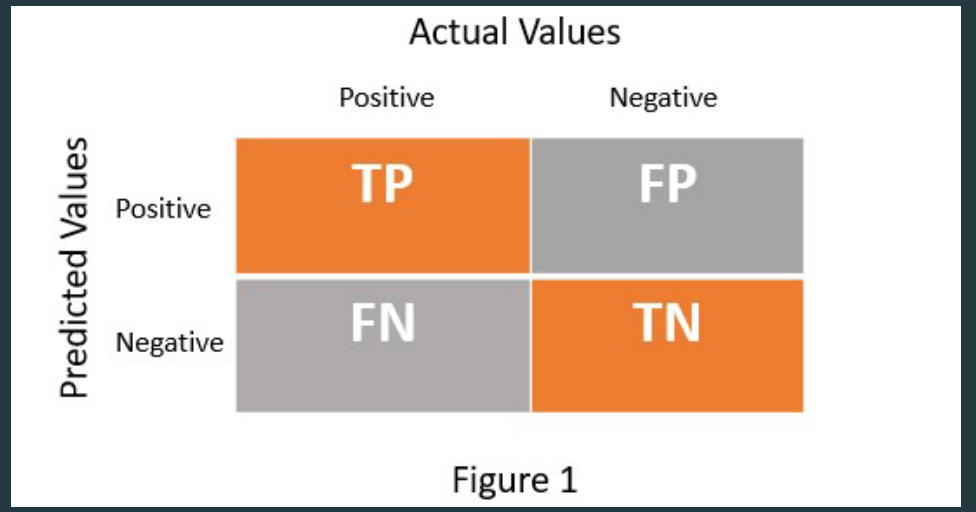


Figure: Depiction of true positive, true negative, false positive, and false negative in a table that resembles a confusion matrix.[1]

[1]Khanna, M. (2023, July 19). *Classification problem: Relation between sensitivity, specificity and accuracy*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/06/classification-problem-relation-between-sensitivity-specificity-and-accuracy/

Gaussian Mixture Model (GMM)

The chosen model for this project is the GMM. It uses multiple multivariate normal density functions to represent data. The advantage of combining more than one normal density function is in the case where there are many data points that vary in how related they are to each other. For example, in this project, all the MFCCs together appear to have no pattern, so a single GMM would be uninformative. But when examined in smaller groups, the MFCCs appear to cluster according to phoneme. This is where having multiple normal density functions becomes useful. The formula for a GMM is as follows[1]:

$$f(x_n \mid \mu, \Sigma, \pi) = \sum_{k=1}^{K} \pi_k \mathcal{N}(x_n \mid \mu_k, \Sigma_k)$$

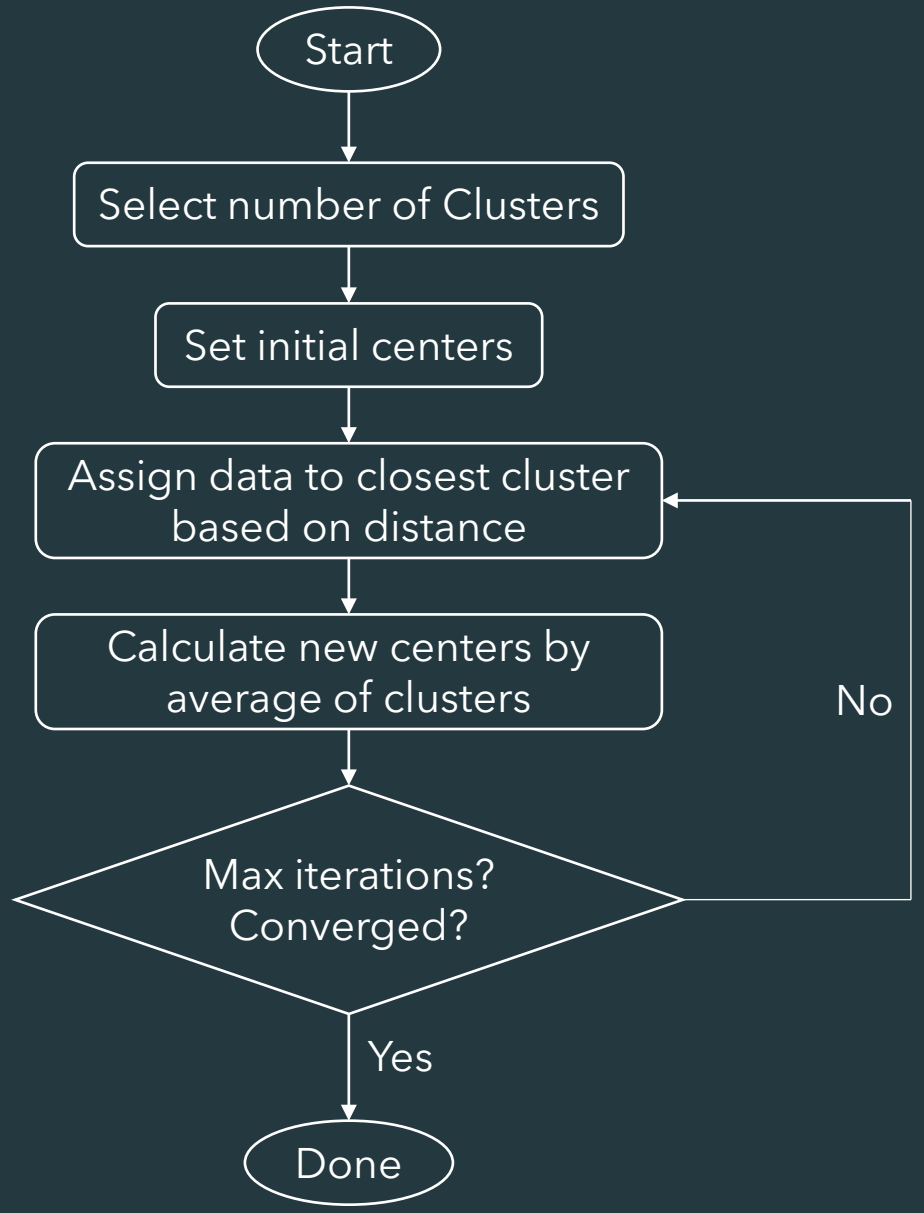
Where…

- K = number of mixture components
- $\mathcal{N}$ = Gaussian Model
- μ = vector of means of each Gaussian Model
- π = vector probabilities of each Gaussian Model
- Σ = vector of covariances of each Gaussian Model
- x_n = data

[2](Heard, N. (2021). Clustering and Latent Factor Models. In: An Introduction to Bayesian Inference, Methods and Computation. Springer, Cham. https://doi.org/10.1007/978-3-030-82808-0_11

K-Means Overview

K-Means is a fundamental clustering algorithm that can create hard groupings of data into a given number of clusters, n. In each iteration of K-Means, the distance between each data point and each centroid is calculated, and the minimum distance becomes the clustering result. In each iteration, the algorithm calculates the i^{th} new center m via $m_i^{new} = \frac{1}{N_i}\sum_{n \in N} x^n$ where N_i is the set of points in the cluster[2]. To converge on the ideal clusters, the algorithm seeks to reduce the sum of the distances between every point and its assigned center. A n-component GMM can be estimated based on the clustering results.

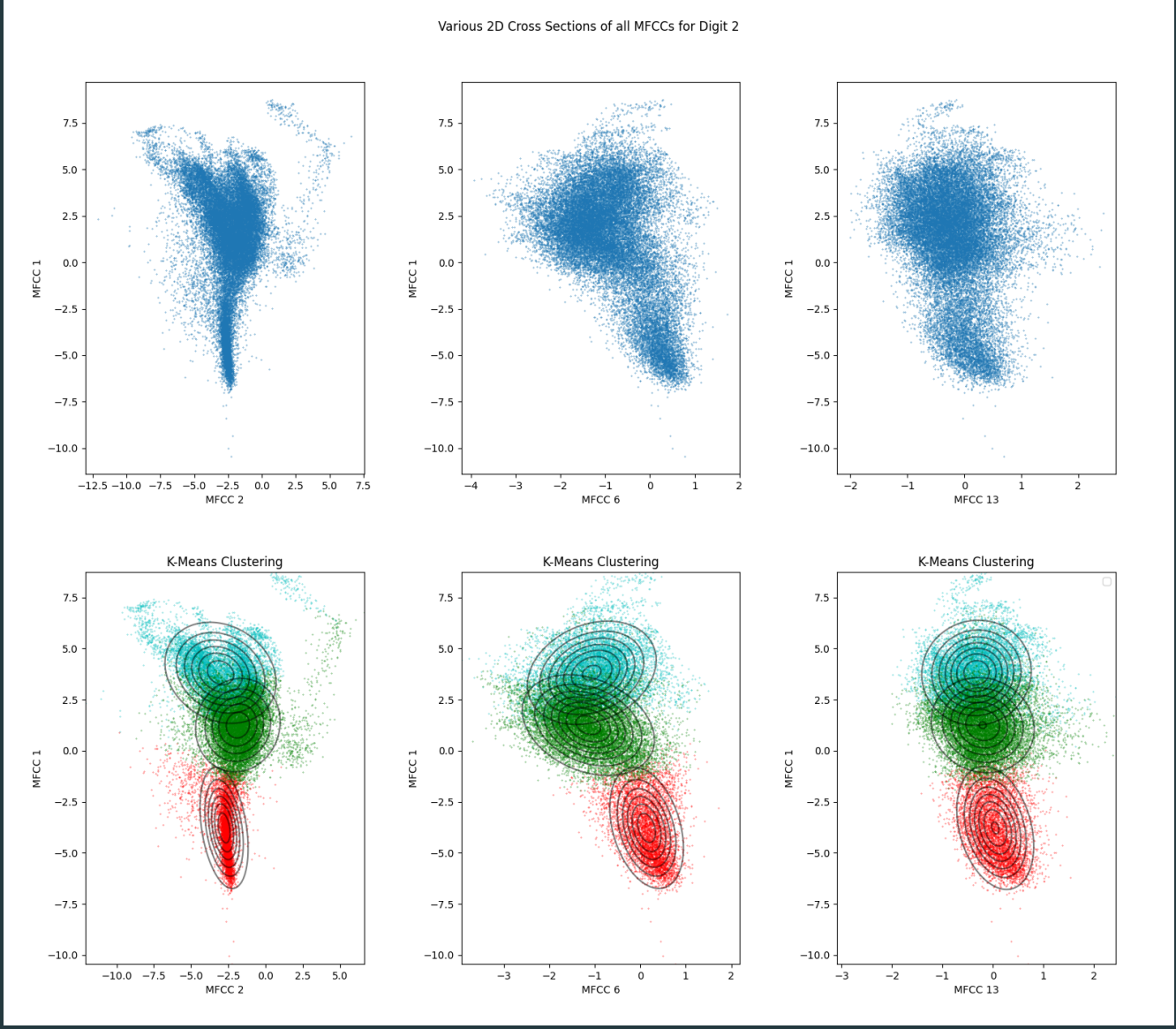


K-Means Flowchart[1]

[1]Senarathna, Sisitha & Hemapala, K T M U. (2020). Optimized Adaptive Overcurrent Protection Using Hybridized Nature-Inspired Algorithm and Clustering in Microgrids. Energies. 13. 3324. 10.3390/en13133324.

[2](Heard, N. (2021). Clustering and Latent Factor Models. In: An Introduction to Bayesian Inference, Methods and Computation. Springer, Cham. https://doi.org/10.1007/978-3-030-82808-0_11

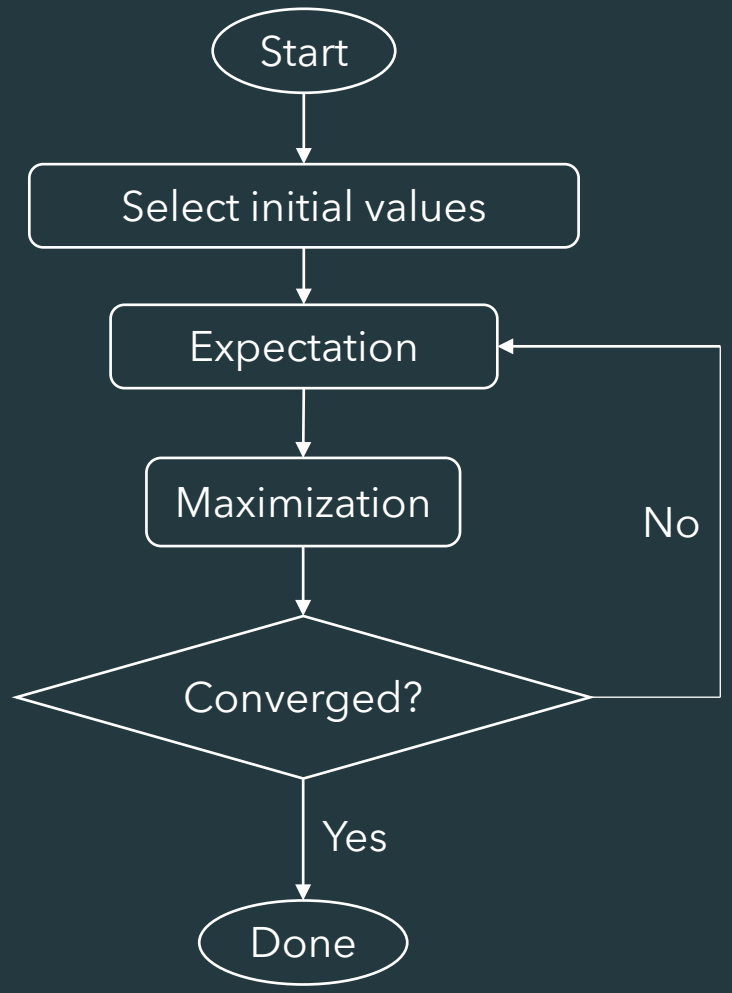
K-Means: Varying 2D Cross Sections



Although these plots do not inform the model parameters as well as other visualizations, it's interesting to observe how the K-Means Distinct Full GMM looks in different dimensions. The most obvious clusters in the top row occur at the bottom of the (1, 6) and (1, 13) cross section. Unsurprisingly, this group of points become the red cluster and their corresponding GMMs look very similar. Visually, the GMM appears to represent each cross section equally well, which indicates fair treatment of all dimensions in estimating the GMM.

Expectation Maximization (EM) Overview

The EM algorithm estimates GMMs by finding the latent variables that maximizes the likelihood of observing data. For this project, latent variables are the GMM parameters that can be inferred from observations. In each iteration of EM, it assigns a responsibility of each cluster over a given point, which can be used to recalculate the latent variables for the next iteration. EM converges when the latent variables stop changing or likelihood stabilizes. Log-likelihood is defined as $\boldsymbol{max}(\sum_{n=1}^{N} \boldsymbol{ln}(\sum_{k=1}^{K} \boldsymbol{\pi}_k \boldsymbol{\mathcal{N}}(\boldsymbol{x}_n \mid \boldsymbol{\mu}_k, \boldsymbol{\Sigma}_k))$ where $\boldsymbol{K}$ is the number of mixture components, $\boldsymbol{\mathcal{N}}$ is the Gaussian model, $\boldsymbol{N}$ is the observations, $\boldsymbol{\mu}$ is a vector of means of each Gaussian Model, $\boldsymbol{\pi}$ is a vector probabilities of each Gaussian Model, and $\boldsymbol{\Sigma}$ is a vector of covariances of each Gaussian Model[2]. The close relationship between GMMs and EM is clear here as the equation for log likelihood in EM resembles the equation for GMMs.

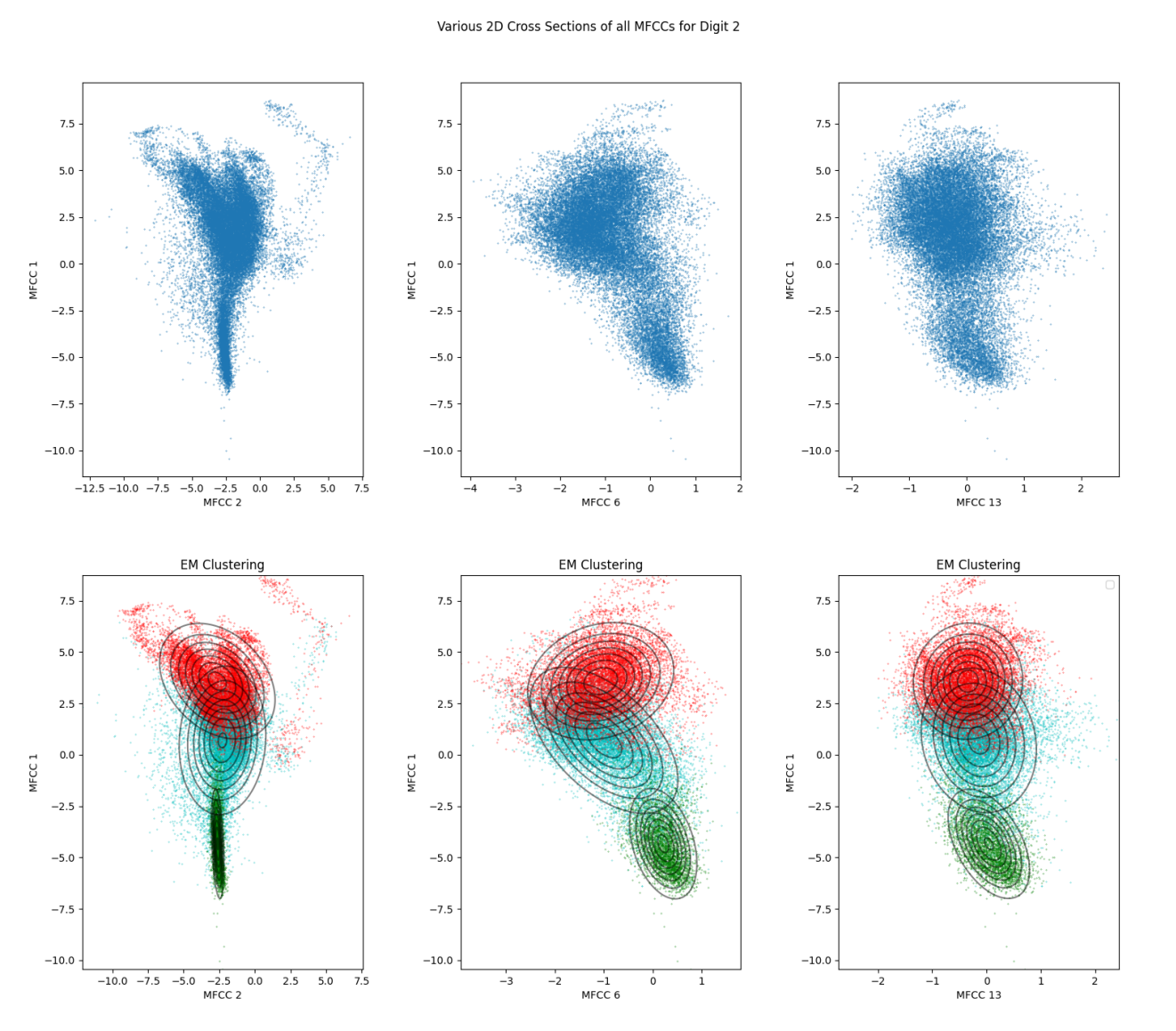


EM Flowchart[1]

[1]Silva J, Vaz P, Martins P, Ferreira L. Reliability Estimation Using EM Algorithm with Censored Data: A Case Study on Centrifugal Pumps in an Oil Refinery. Applied Sciences. 2023; 13(13):7736. https://doi.org/10.3390/app13137736

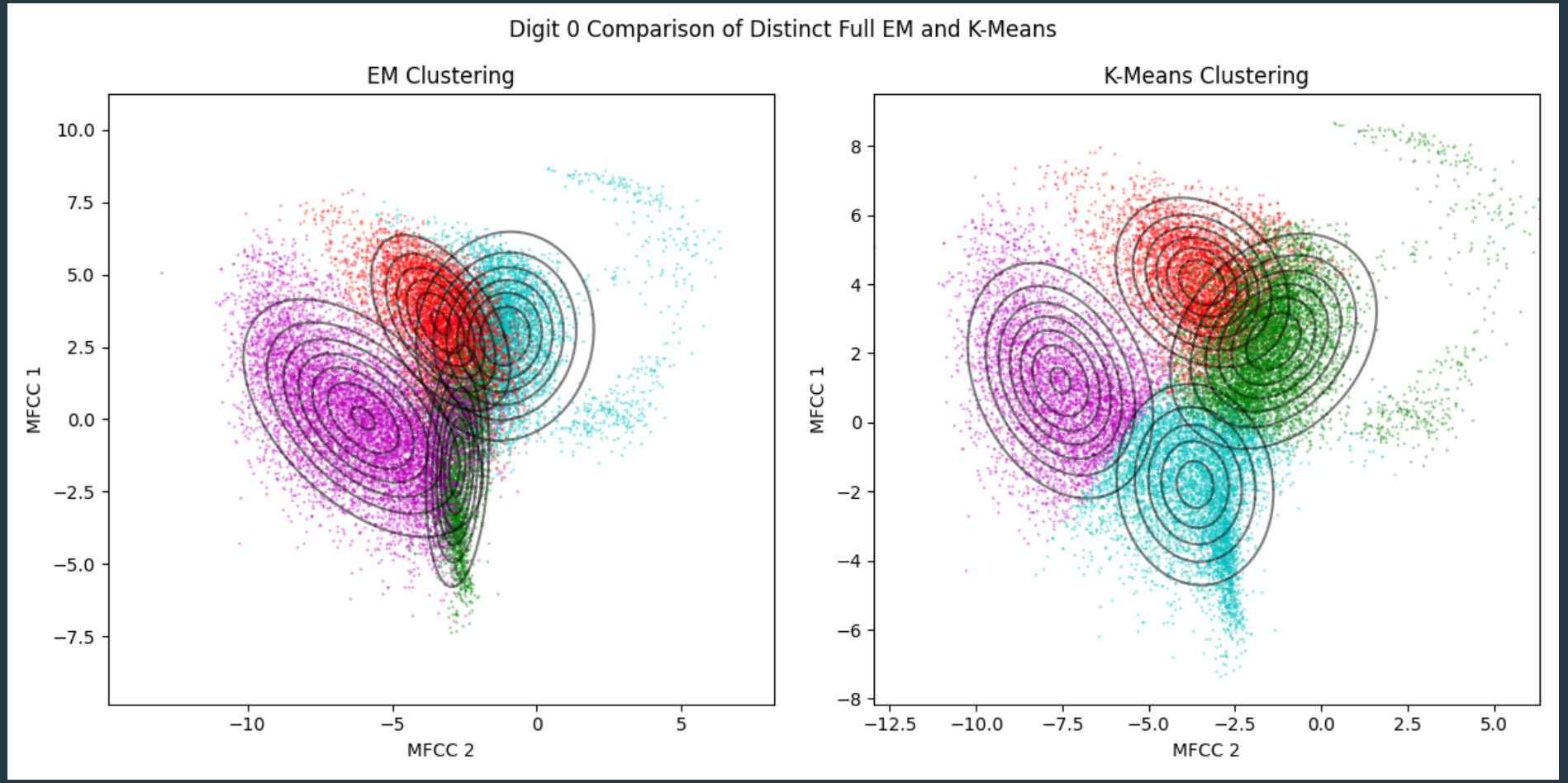
[2](Heard, N. (2021). Clustering and Latent Factor Models. In: An Introduction to Bayesian Inference, Methods and Computation. Springer, Cham. https://doi.org/10.1007/978-3-030-82808-0_11

EM: Varying 2D Cross Sections



This plot gives a closer look into how the EM Distinct Full GMM looks in different dimensions. An interesting difference between the K-Means clustering and EM is that EM appears to make more unique clusters. Nearly every contour looks different from the corresponding contour in other dimensions, which was not true of the K-Means cross sections. This displays the naturally flexible nature of using EM compared to K-Means.

K-Means vs. EM: Distinct Full Covariance



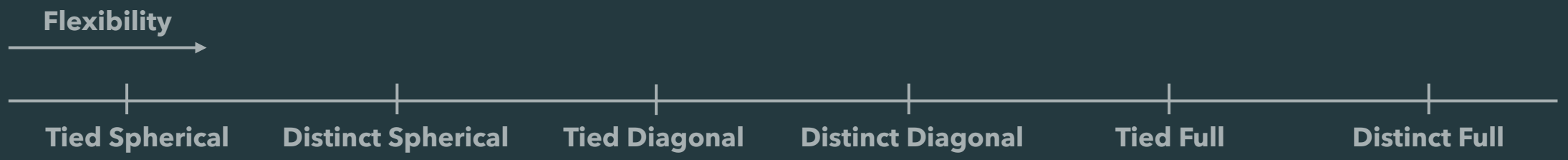
The green cluster in the EM clustering algorithm fits the tail of observations in digit 0 much better than the corresponding blue cluster in K-Means. The trade-off is having a larger purple cluster, but the overall fit of the EM contours look better. This indicates that EM is superior at taking full advantage of the flexibility of the covariance constraints while creating clusters. This aligns with the math behind EM, since it accounts for covariance in each iteration.

K-Means vs. EM: Tied Spherical Covariance

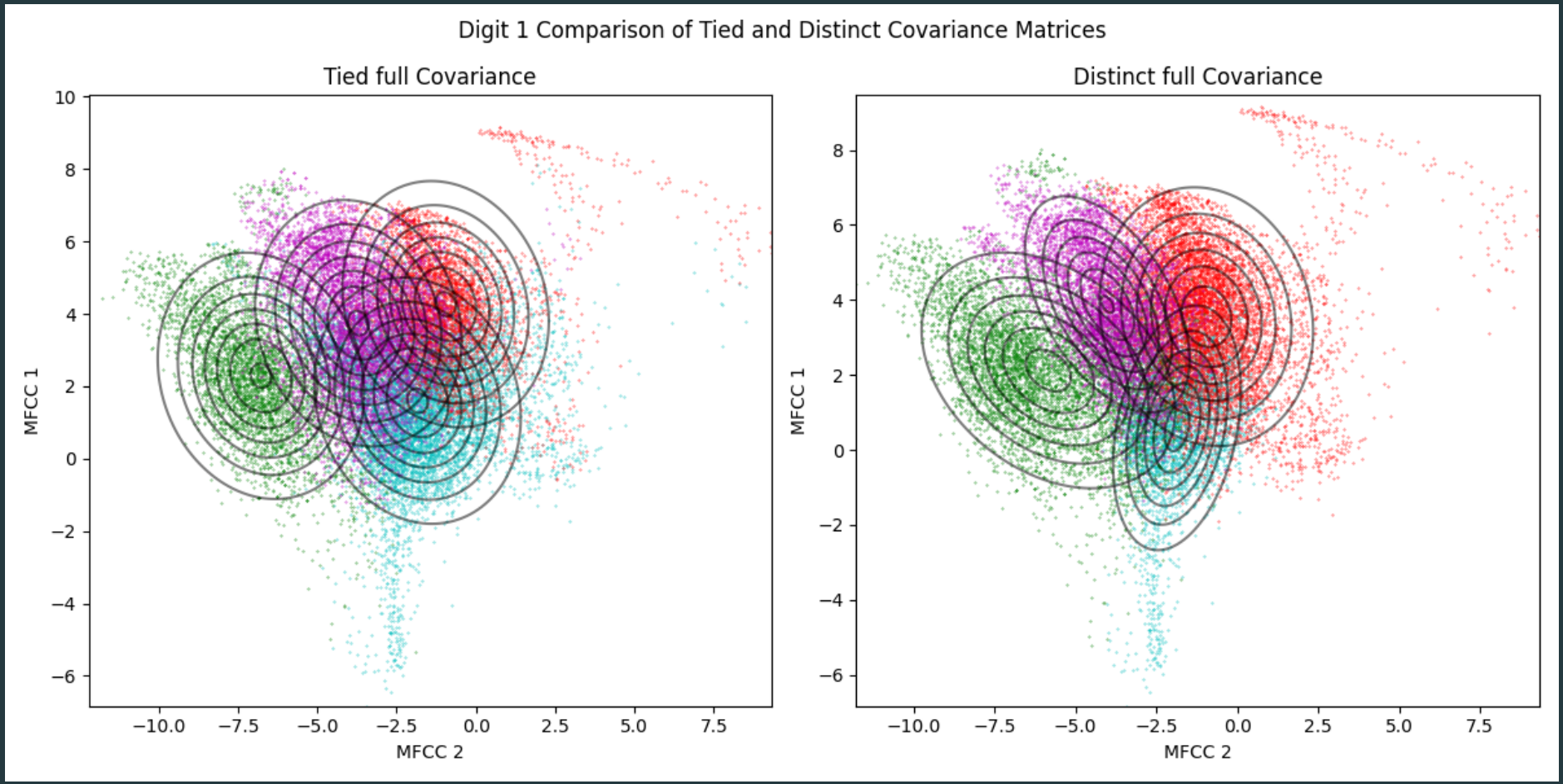
When comparing tied spherical covariance GMMs, EM and K-Means perform very similarly. The corresponding clusters in the left and right plots appear to have very similar boundaries, shapes, and centers. Just by comparing the cross sections of the clustering result from the previous slide and this slide, the optimal covariance constraint for the digit recognition model should be closer in flexibility to the distinct full covariance.

Influence of Covariance Constraints

Tied covariance implies creating a covariance with all cluster data decentralized. Distinct implies creating an individual covariance for each cluster. Spherical covariance makes a covariance with spherical shape by taking the variance of all the dimensions at once and using it as the variance of each dimension. Diagonal covariance finds the independent variation along each axis, but no covariation between axes. This is done by taking the variance of each dimension and zero-ing out the rest of the matrix. Full covariance implies that there are no constraints on the shape, which is equivalent to finding the variance of each dimension as well as the covariance between all dimensions.

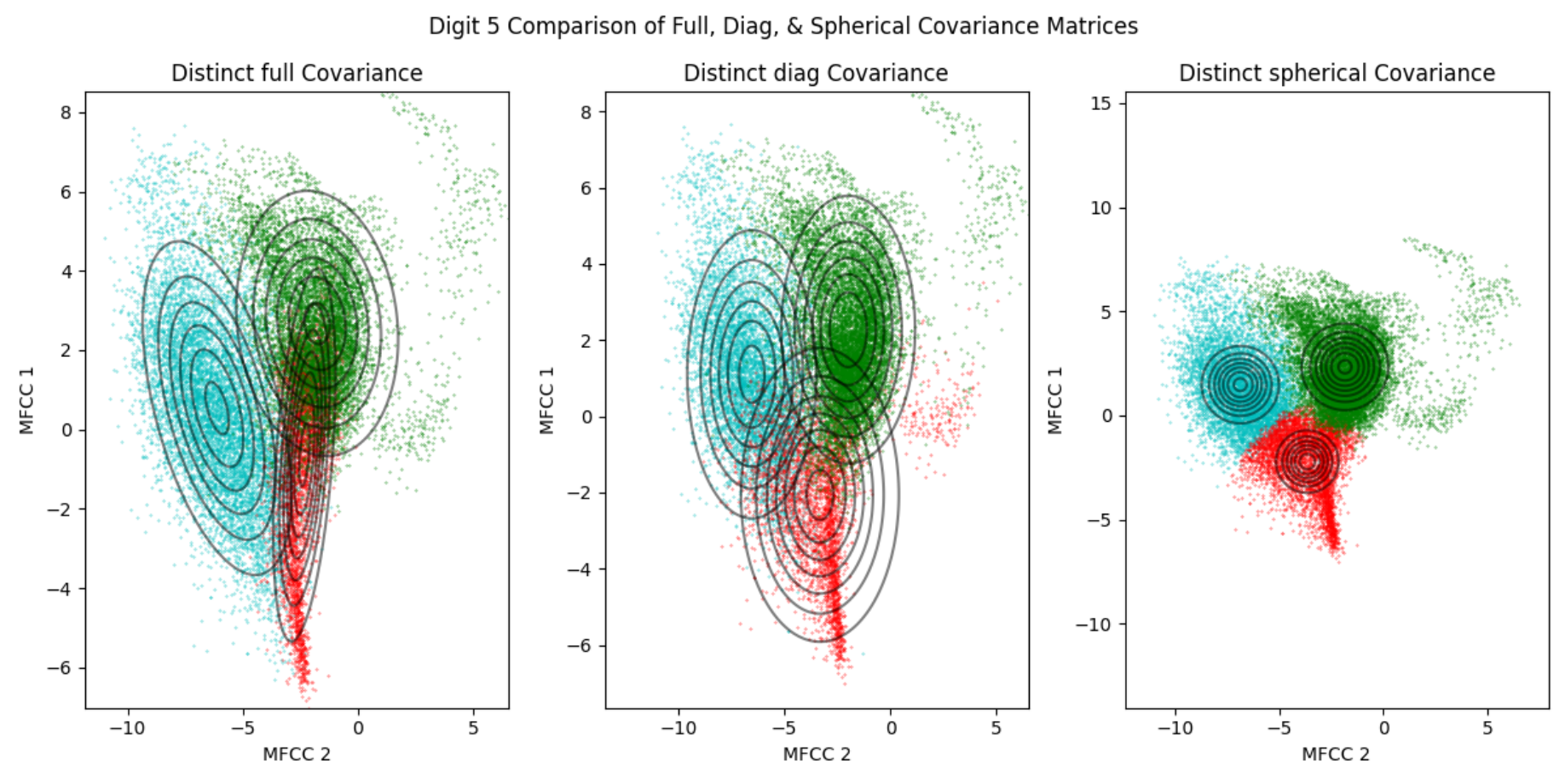


Visualization of Tied vs. Full



Distinct covariance fits the shape of the data better, especially for the blue tail of observations. Also, the GMMs are less spread out overall. For this model, it seems like it would be better to use distinct since each cluster of phonemes appear to have unique shapes. In this example, the blue cluster is much longer along the MFCC 1 dimension, which is a nuance that is lost by tying all the data together.

Visualization of Full, Diagonal, & Spherical



Distinct spherical covariance is the worst by far, so it is not a covariance constraint that would be suitable for the speech recognition model. Full does the best at capturing the variance of the red cluster along the MFCC 1 dimension, but for the blue and green clusters, the diagonal covariance is very comparable. Based off these plots, the full covariance would be the best option, but it would be interesting to explore the diagonal covariance as well.

SELECTING HYPERPARAMETERS

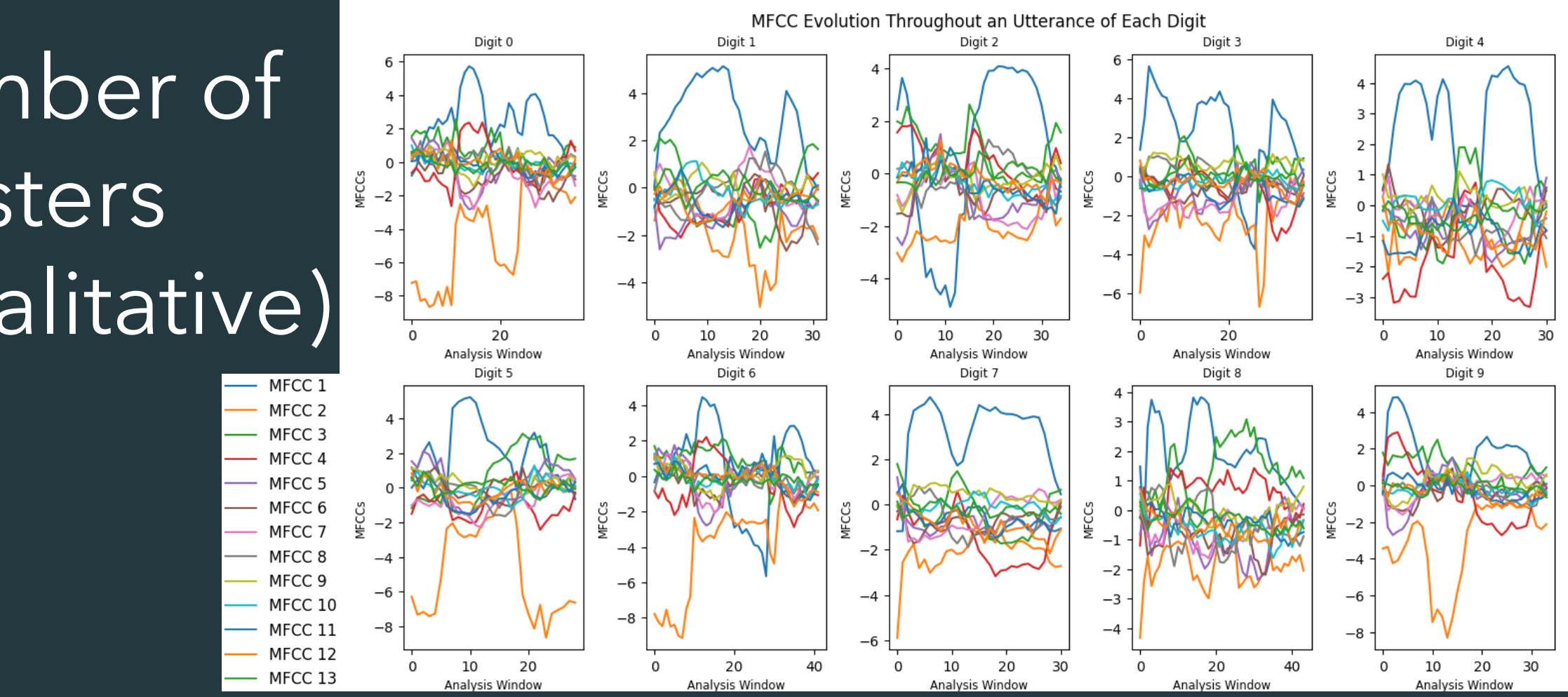
Per Digit Model Choices vs. Global Model Choices

While designing this model, there was a choice between choosing MFCCs, covariance constraints, and clustering algorithms per digit or choosing those hyperparameters for all the digits combined. The reason for choosing to create the model using all data is due to two reasons:

1) Computational Efficiency: It already takes a long time to analyze hyperparameters one time for all the digits. When measured with the time package, it take close to 30 seconds to train a GMM and classify the test data. To do this on each digit would take significantly longer.
2) Overfitting: After attempting to select covariance constraints and clustering algorithms for each digit, the resulting confusion matrix was extremely inaccurate. The model must have fit the training data so closely that variations introduced by the testing data did not fit in well.

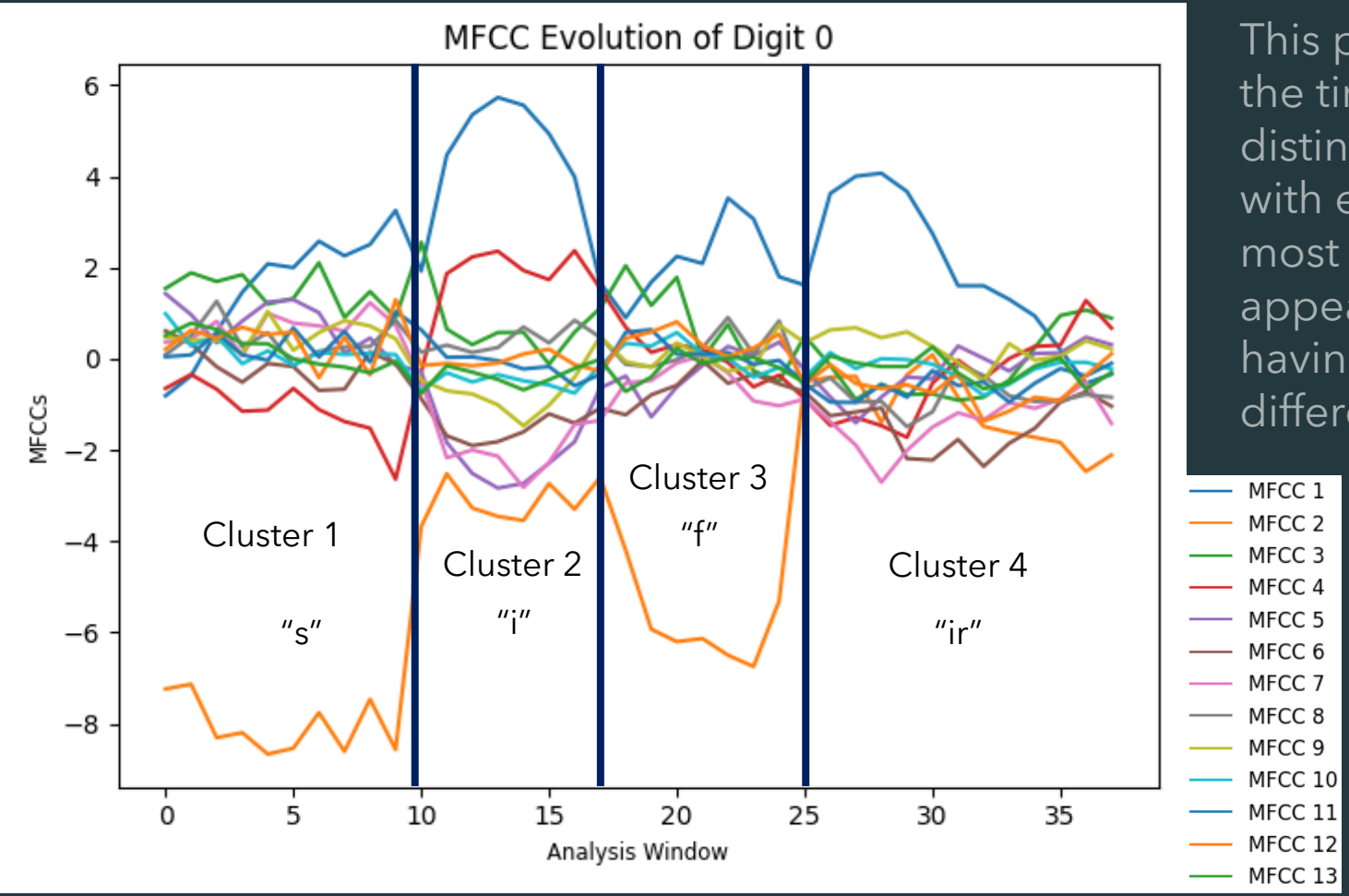
The only parameter that was assigned per digit was the number of clusters because it is based on the number of phonemes in an Arabic digit. This can be easily differentiated between digits just by listening to pronunciation videos, sound it out loud, or even qualitatively/quantitatively through plots.

Number of Clusters (Qualitative)



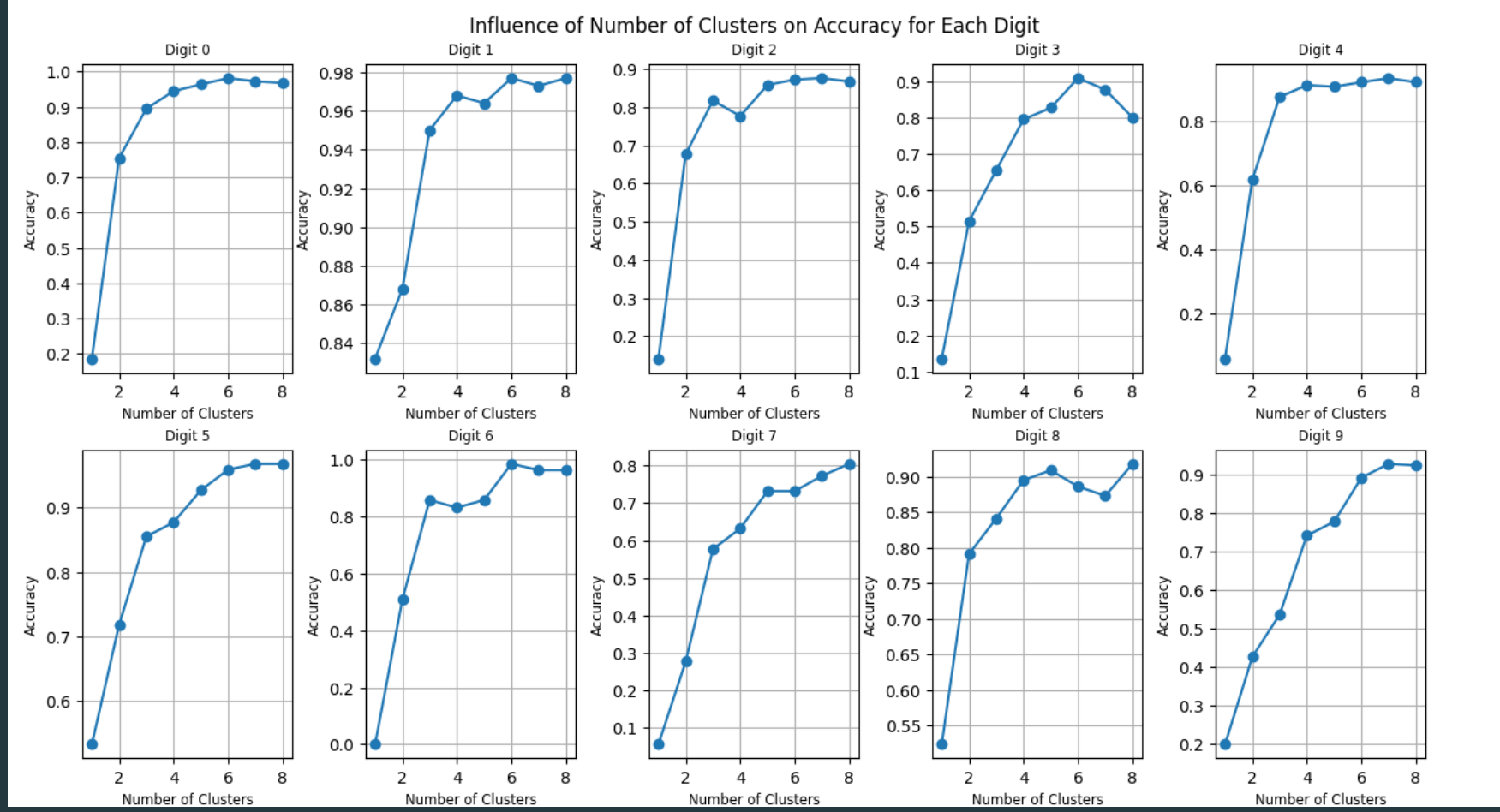
This is a plot of all 13 MFCC columns throughout analysis windows for an utterance of each digit. It can be used to qualitatively determine how many phonemes are in each digit, which informs the number of clusters to use. The MFCCs display distinct phases, each of which represents a different phoneme. For example, the MFCC 2 in digit 0 displays ~4 distinct phases for each phoneme, which is an indication that the GMM for digit 0 should have 4 clusters. Another interesting takeaway from these plots is that some MFCCs show distinct phases better than others, which could be an indication of which MFCCs captures the most variation of an utterance and would be more important for informing the GMM for a given digit. An example of this is that MFCC 1 shows the most distinct phonemes (~3) for digit 2, whereas the other MFCCs for digit 2 display a lot of noise that makes it difficult to split it into phases.

Interpretation of MFCC Timeseries Plot



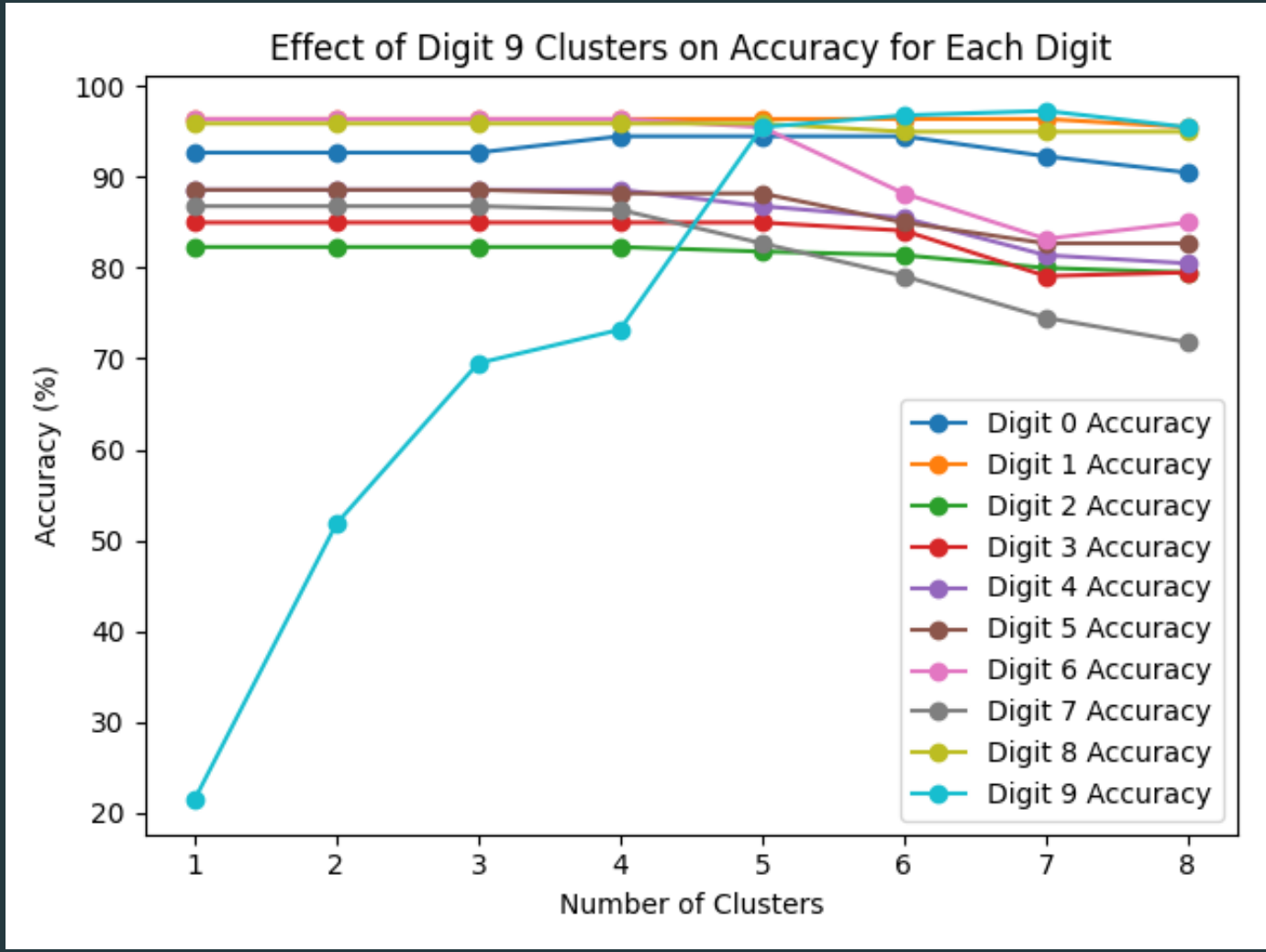
This plot provides a closer look of how to interpret the timeseries plots. MFCCs 1 and 2 display the distinct shifts the best, and they also align very well with each other, which confirms that this is the most likely division of clusters. Even MFCC 4 appears to follow these cluster divisions despite having lots of noise. Each cluster represents a different distinct sound, or phoneme.

Number of Clusters (Quantitative)



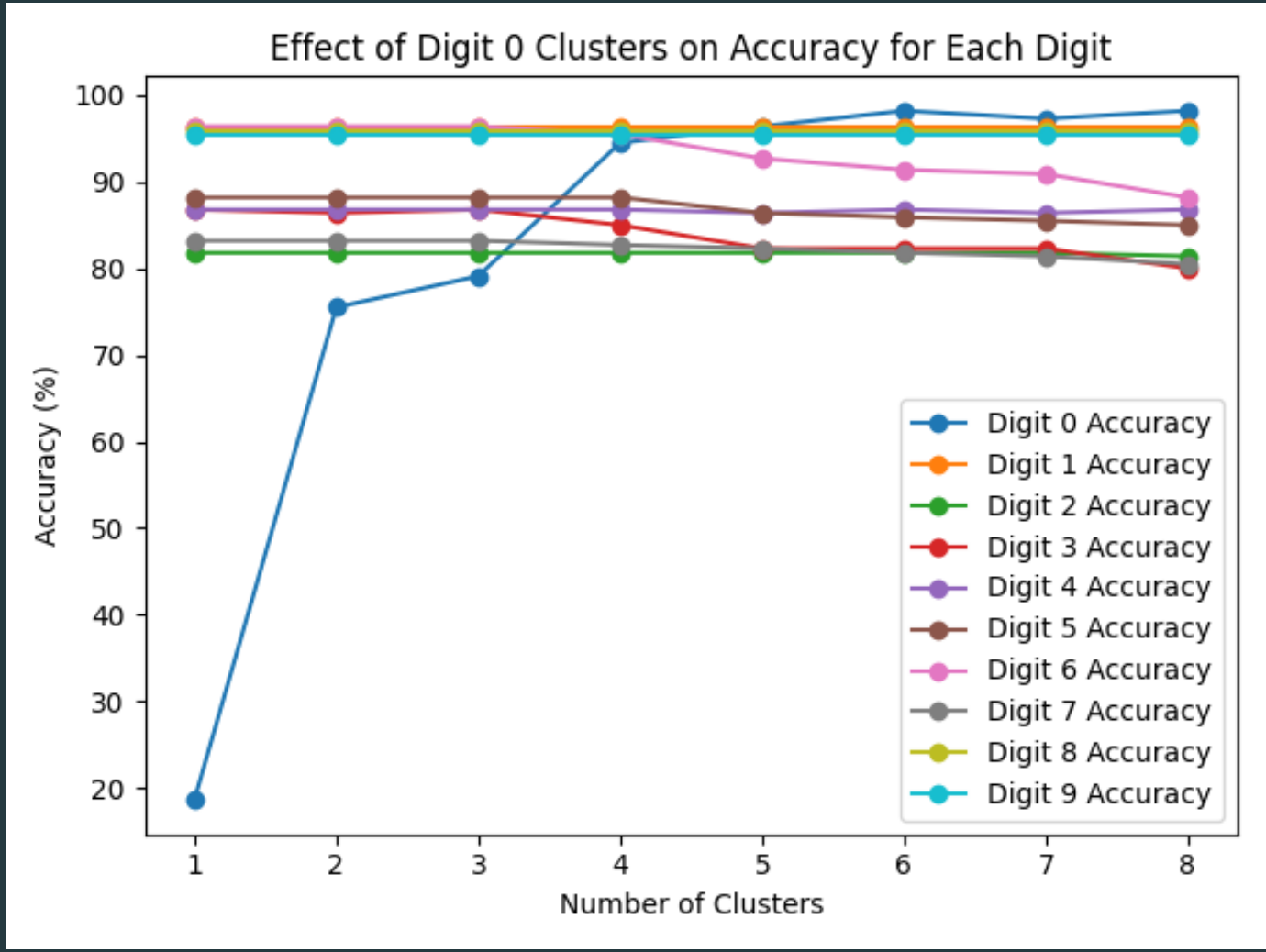
The qualitative analysis of the previous graphs informed a preliminary mapping of each digit to their number of phonemes. These plots are meant to build off those initial values to quantitatively determine approximately how many phonemes each digit has. In each plot, all other hyperparameters are set to the same values, and the only parameter that is varied is the number of clusters for that digit. Each plot has a knee that occurs when the rate of increase in accuracy slows down. For digits 0 and 1, it clearly occurs at ~4 clusters, whereas other digits such as 9 appear more ambiguous. A Bayesian approach can be taken to determine how many phonemes to use for each digit. If the quantitative plot is clean, then the number of clusters parameter is mostly informed by these plots. However, if the quantitative graph does not display clear trends, then it would be useful to find the approximate knee in these plots and cross check them against the qualitative graphs.

Effect of Changing # of Clusters: Digit 9



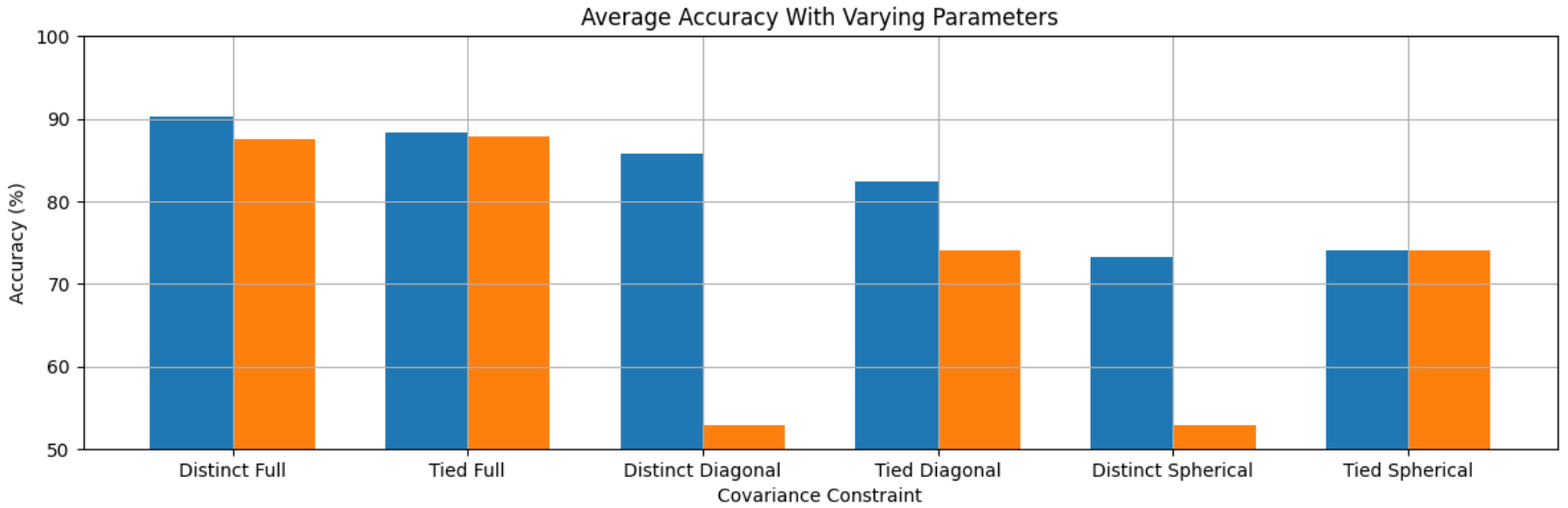
To explore the ambiguity in determining the optimal number of clusters for digit 9, this plot displays the global effect for changing its cluster number. It seems that the rest of the digits are quite sensitive to the number of clusters used for digit 9. Although digit 9's accuracy has an upward trend in accuracy with more clusters, the rest of the digits start to lose accuracy past 4 clusters. This indicates that those digits are mistakenly being classified as a 9. This is an interesting trend, and it speaks to how certain digits may be more closely related to each other. Number 7 is the most sensitive to 9's number of clusters, so its possible that their phonemes are more similar, and can only be differentiated by a temporal factor that is not accounted for in this model.

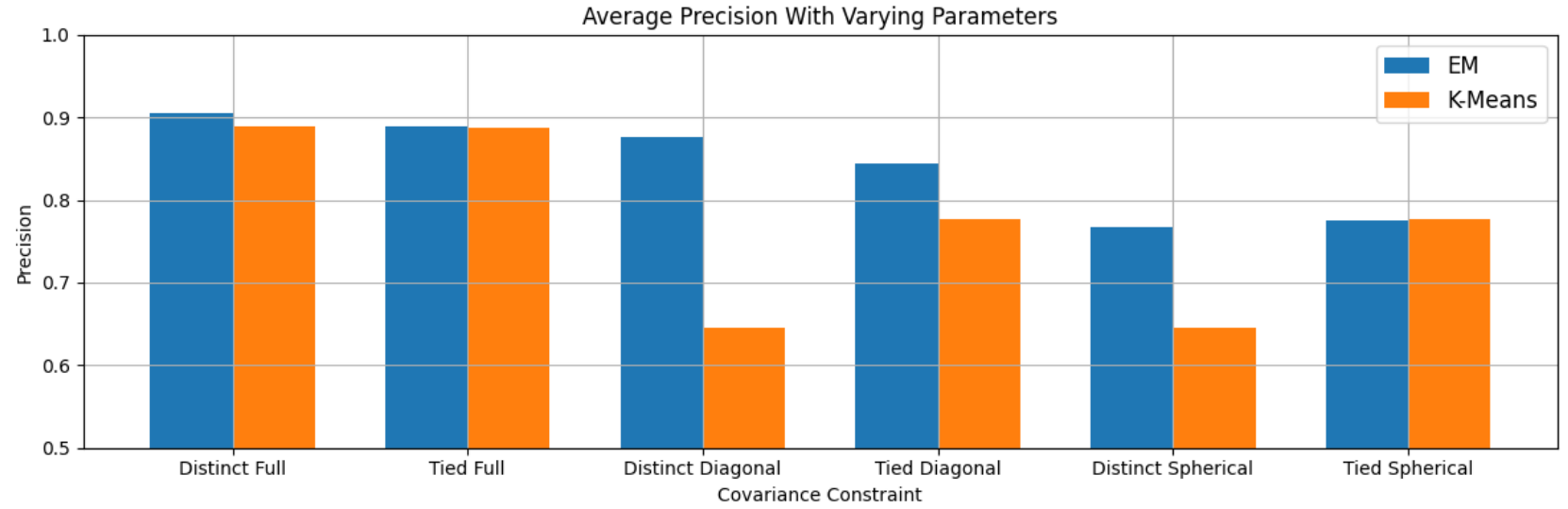
Effect of Changing # of Clusters: Digit 0



Since the digit 0 had the least ambiguity in number of clusters, it is interesting to explore the effect of 0's number of clusters parameter affects the rest of the digits. Where many digits were highly sensitive to 9's corresponding parameter, this plot shows the opposite. The other digits appear to experience minimal change in accuracy when digit 0's cluster number increases. This indicates that digit 0's data may be more isolated on average from the rest of the digits' data when compared to digit 9. Still, it is necessary to be careful about selecting 0's parameters since it negatively impacts accuracy for digits like 6 and 7.

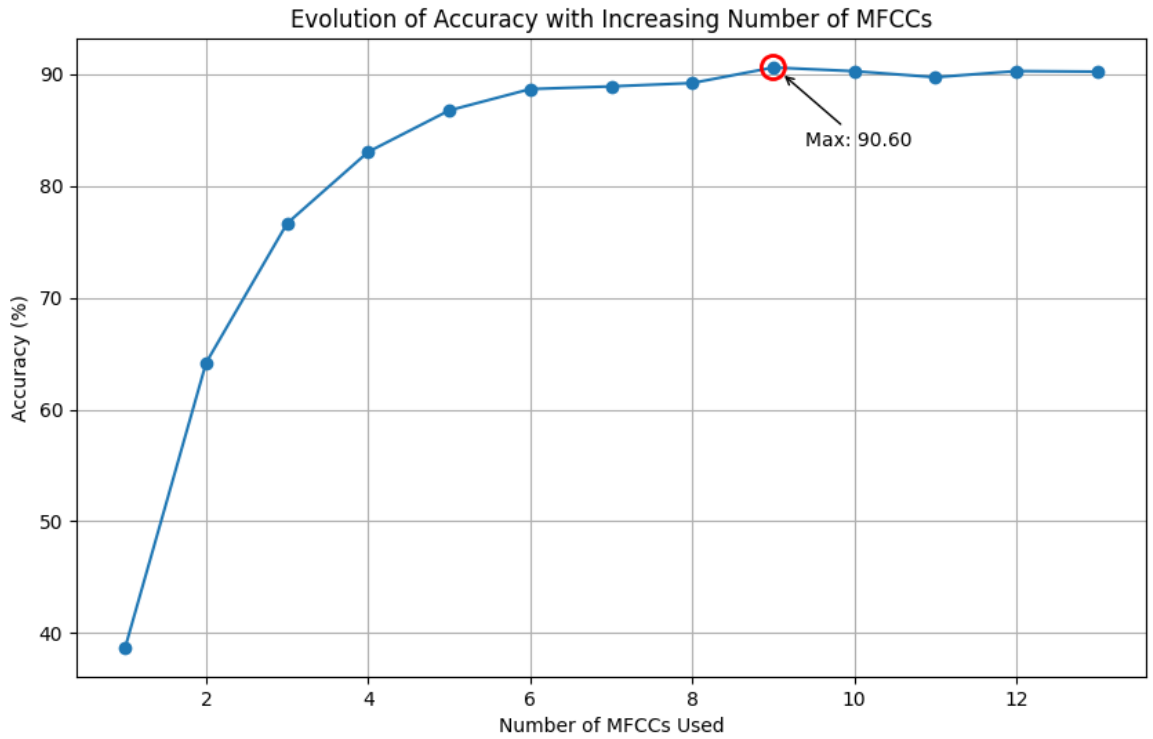
Covariances Constraints





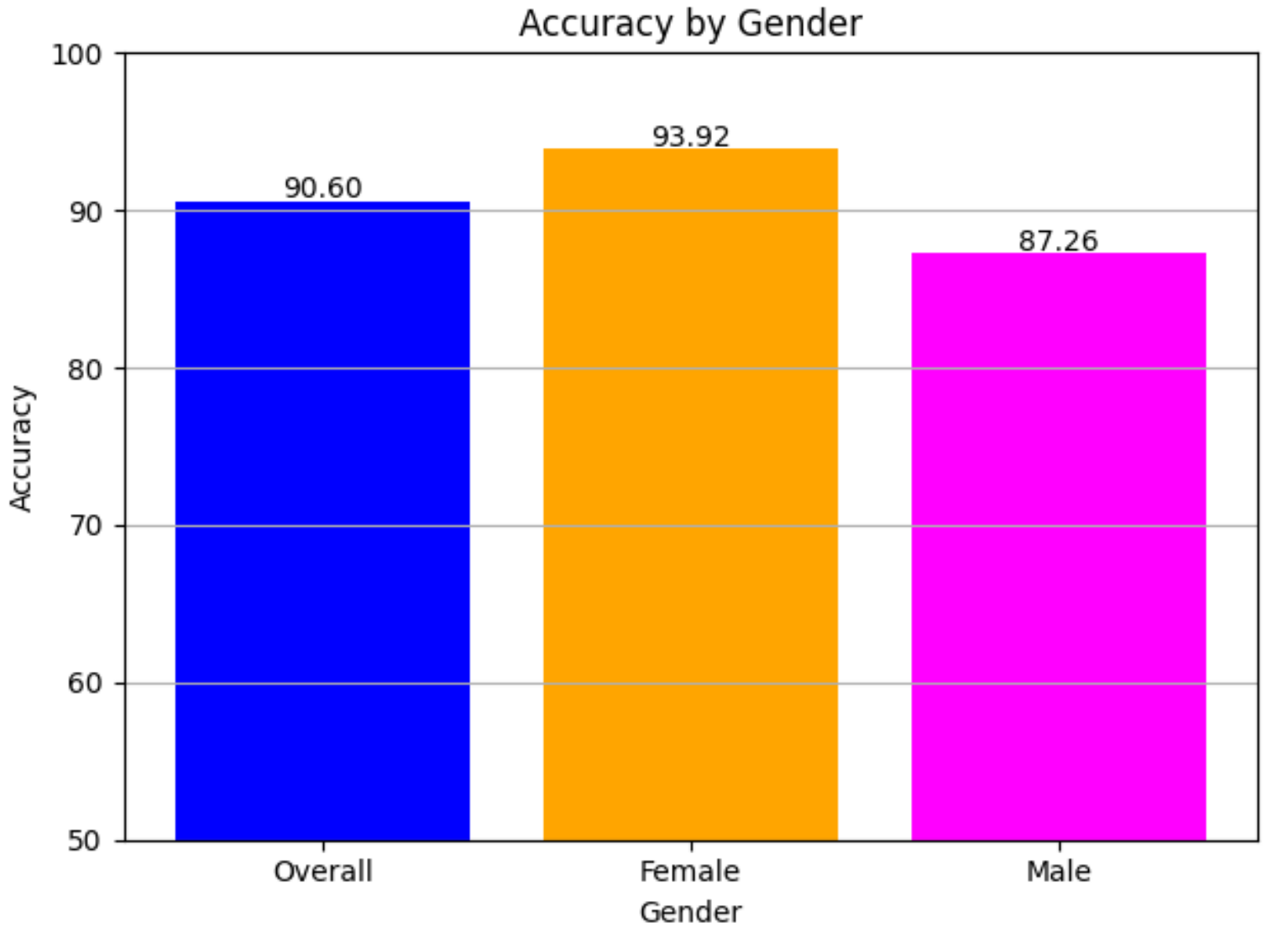
This plot displays the average accuracy and precision of the confusion matrix generated with variable parameters. These results align with many plots from previous slides. K-Means and EM not only qualitatively appear to have the same performance for tied spherical, but also quantitatively. Aside from that case, EM is better than K-Means every time. This indicates that EM is a better clustering algorithm for this model. Of the covariance constraints, distinct full is the best overall, which indicates that flexibility is important for this model. However, tied full and distinct diagonal are other good options.

Greedy Algorithm to Filter MFCCs



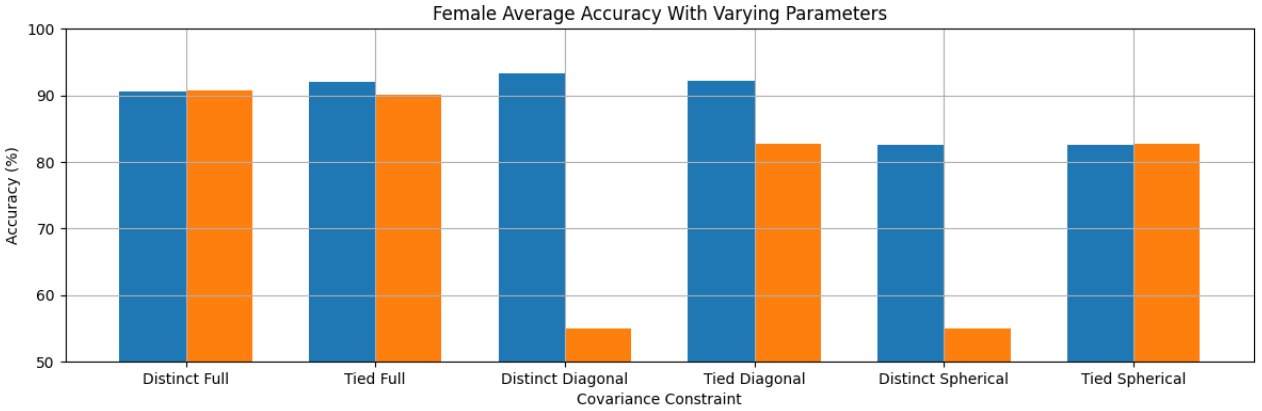
Some MFCCs propagate more information about phonemes than others. This was observed in the "Number of Clusters (Qualitative)" slide, which clearly showed that some MFCCs have more distinct phases than others. To quantitatively analyze which MFCCs truly inform phoneme clusters more, every possible combination of would need to be tested. However, this is computationally unrealistic, so a greedy approach is a suitable alternative. It iterates 13 times and selects the MFCC that optimizes accuracy of the existing list of ideal MFCCs at each step. According to the plot, the number of MFCCs that maximized accuracy was 9. The exact list of MFCCs is [1, 2, 4, 5, 6, 7, 8, 10, 12]. This was determined using EM, distinct full covariance, and the previously determined number of phoneme clusters per digit.

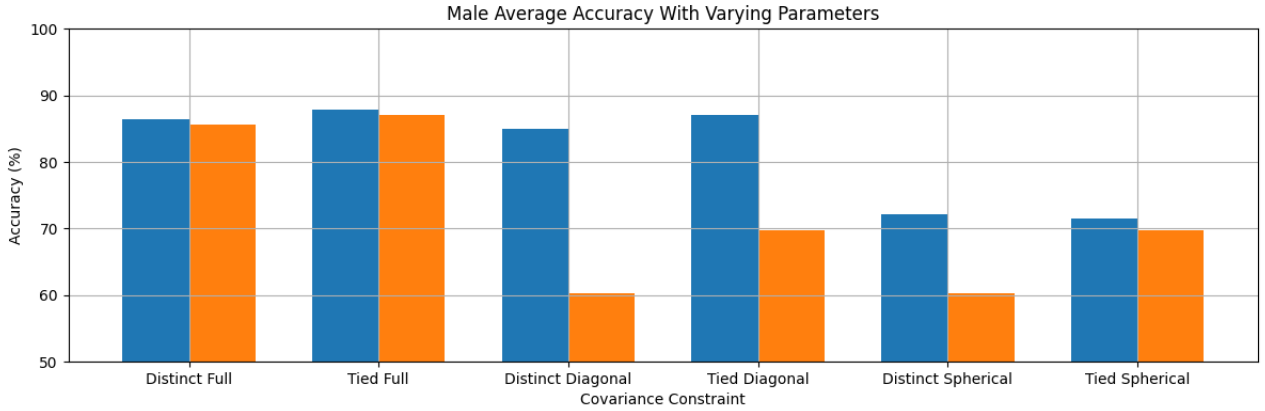
Effect of Testing by Gender

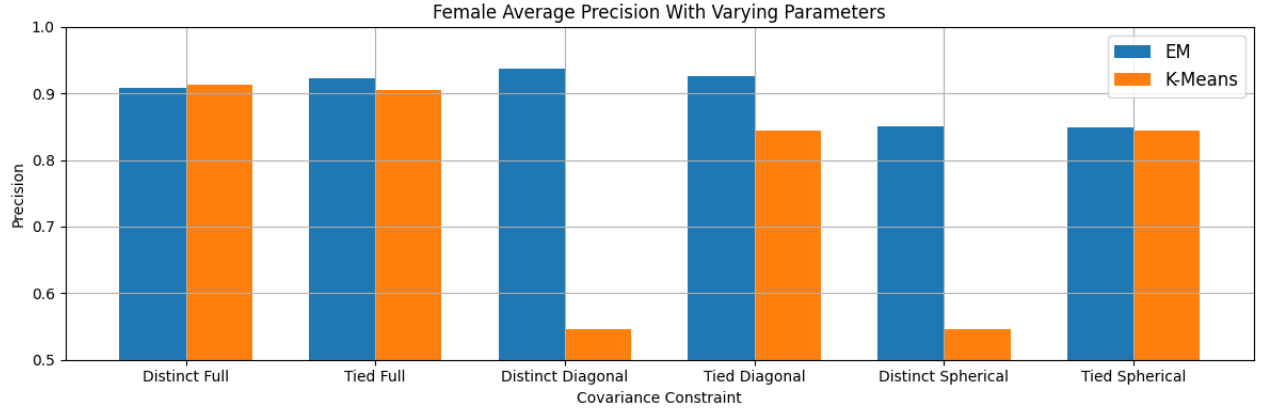


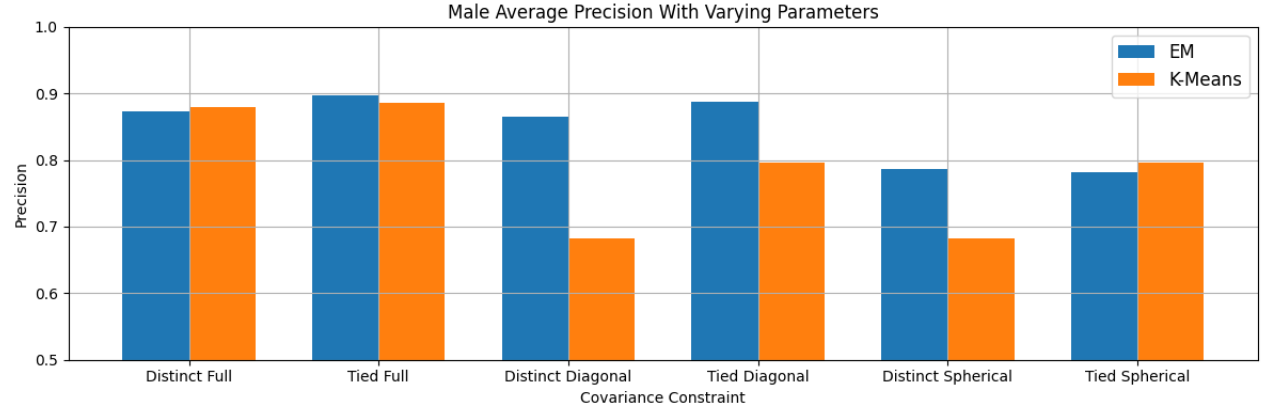
The plot is generated by training the GMMs with all the data but tested with data separated by genders. The female utterances were classified more accurately than male utterances, which indicates that the overall model more closely fits the female utterances. To decrease the gap between male and female accuracies, it would be useful to explore the option of training the GMMs with male and female data separated. The following slides will explore this path.

Covariances Constraints by Gender



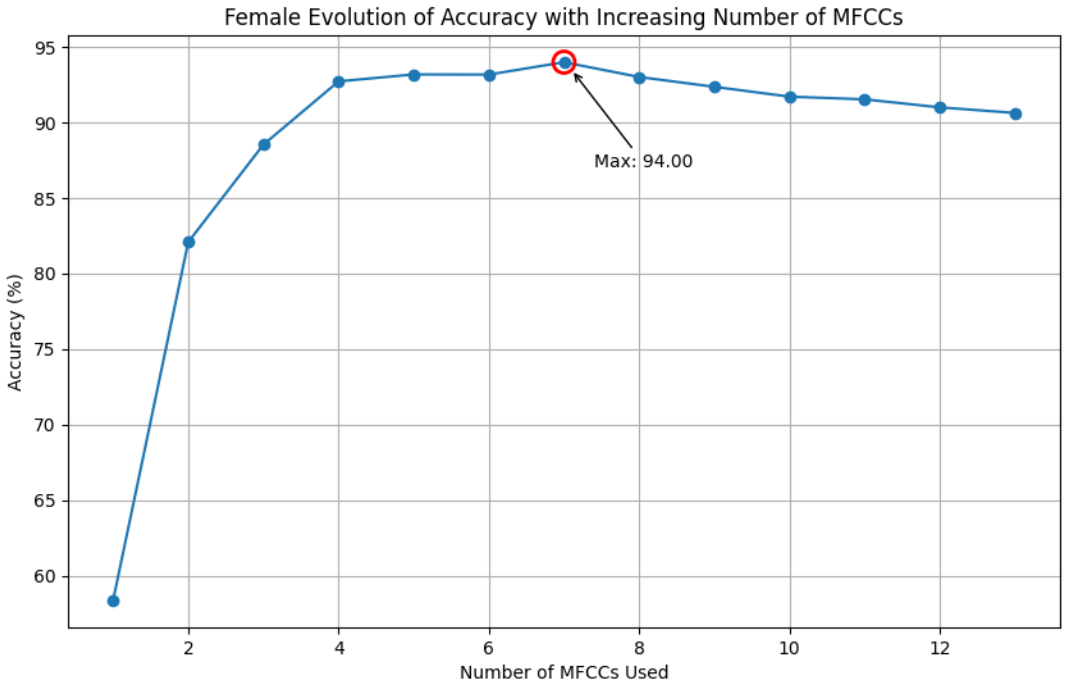


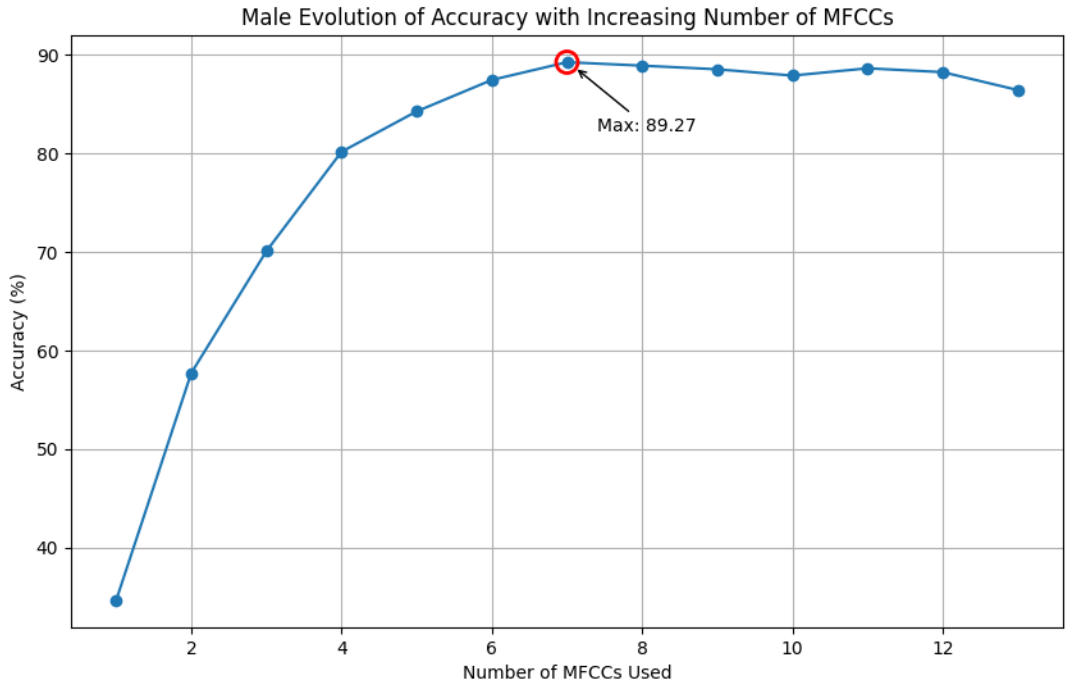




These plots display the performance of varying parameters if the training and testing is separated by gender. Still, using EM performs better in terms of accuracy and precision than K-Means. While both genders experience an increase in accuracy and precision, it seems that the female data had a larger increase. Also, the optimal covariance for classifying female utterances is distinct diagonal and tied full for male utterances. The change in covariance constraints shows promise, but it is possible that assigning covariance constraints by gender causes overfitting and negatively affects the classification results.

Greedy Algorithm to Filter MFCCs by Gender





When the GMMs are trained and tested using female utterances, the accuracy peaks at 94%, which is a significant improvement from the GMMs trained with all data. It seems that for the female utterances, 7 MFCCs are more important than the others, and these are MFCCs [1, 2, 4, 6, 7, 10, 12]. For the male model, the accuracy peaks at 89.27%, which is still worse than female classification. The male data also had 7 important MFCCs, which were [3, 4, 6, 7, 8, 10, 11]. Based on both plots, it seems possible that the male data was lower quality since the male model consistently performs worse than in accuracy than the female model. Changing the MFCCs shows promise, but it is possible that it will cause overfitting on the training data, thus decreasing the classification accuracy.

MAXIMUM LIKELIHOOD CLASSIFICATION

Likelihood Equation

The key to automatically recognizing speech is the likelihood equation. This formula computes the likelihood that an utterance with N analysis frames came from a given model. In this case, each digit is represented by a Gaussian Mixture Model, so the likelihood is represented as[1]:

$$p(X \mid \mu_d, \Sigma_d, \Pi_d) = \prod_{n=1}^{N} \sum_{m=1}^{M} \pi_{m,d} p(x_n \mid \mu_{m,d}, \Sigma_{m,d})$$

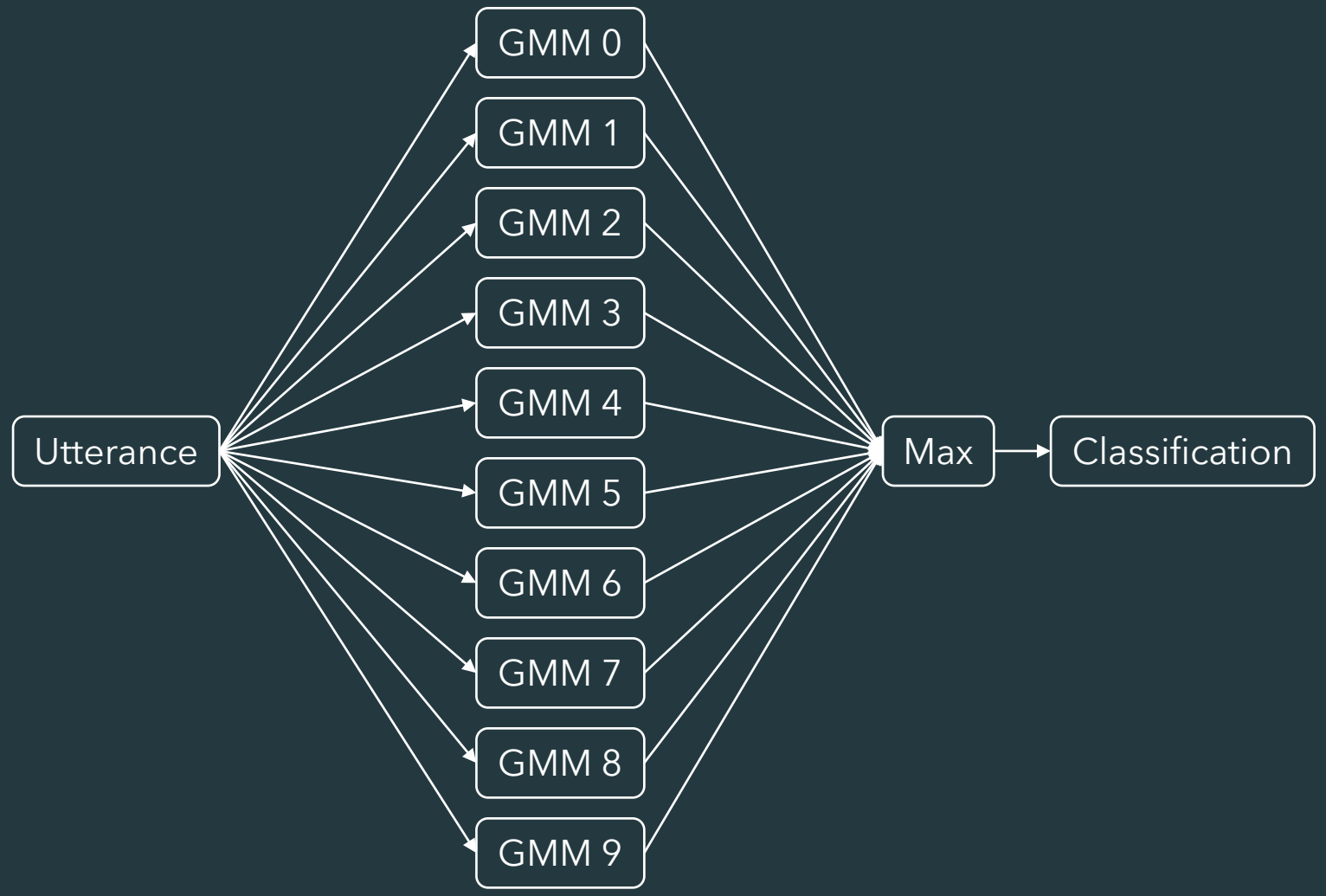
Where…

- X is the block of MFCCs for an utterance
- N is the number of frames in X
- M is the number of components in the GMM
- d is the digit x_n
- μ_d is the set of means of the GMM
- Σ_d is the set of covariances of the GMM
- Π_d is the set of probabilities of each Gaussian Model
- x_n is an analysis frame in X
- $\pi_{m,d}$ is the probability of the mth component
- $\mu_{m,d}$ is the mean of the mth component
- $\Sigma_{m,d}$ is the covariance of the mth component

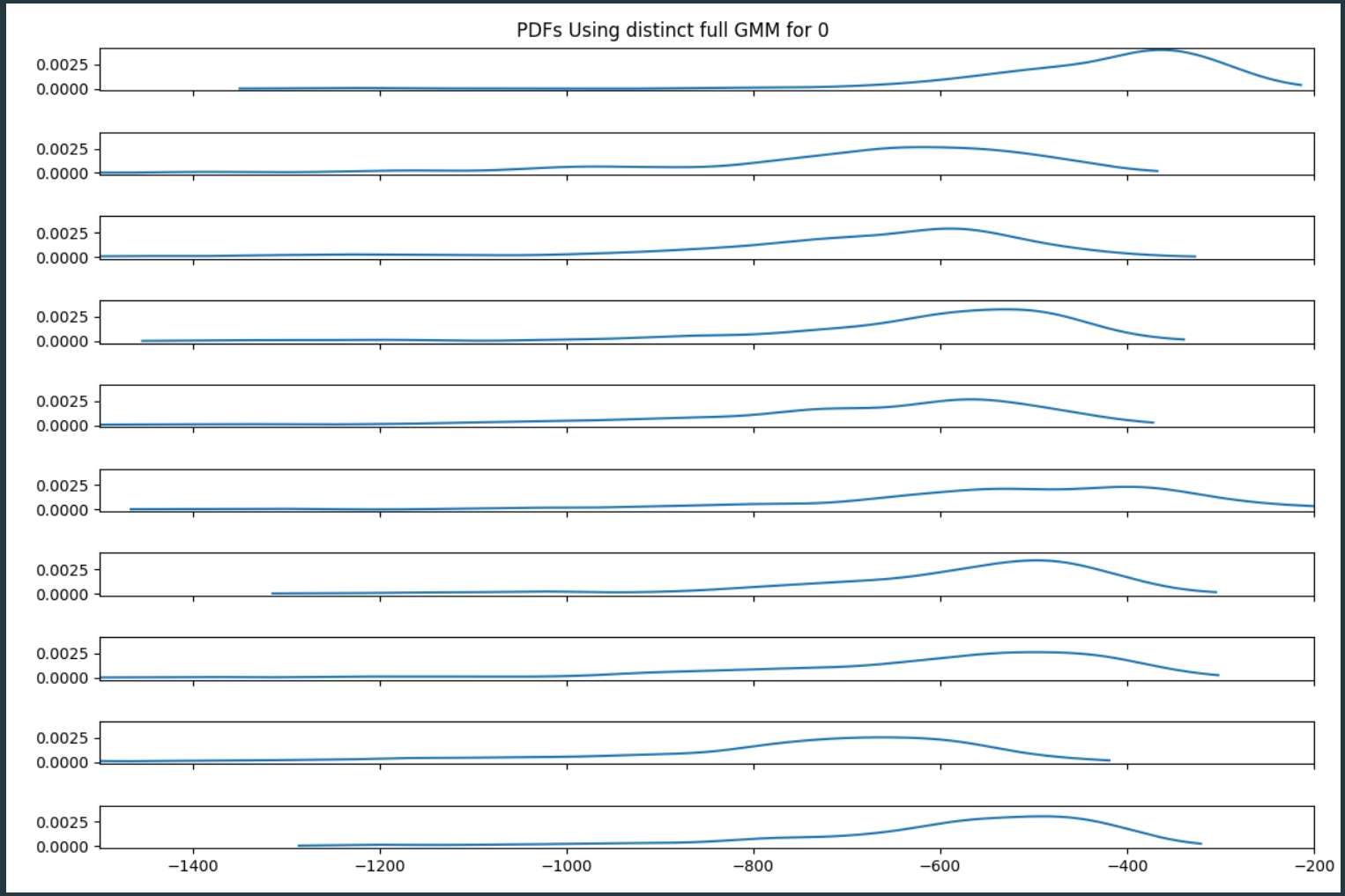
[1](Heard, N. (2021). Clustering and Latent Factor Models. In: An Introduction to Bayesian Inference, Methods and Computation. Springer, Cham. https://doi.org/10.1007/978-3-030-82808-0_11

Maximum Likelihood Classification (MLC)

Given a single test utterance, 10 likelihoods are calculated for each digit's GMM using the likelihood equation. The GMM that results in the largest likelihood is selected as the classification of the utterance. Since the true digit is embedded in the test data, the classification can be identified as correct or incorrect, which is useful for analyzing the performance of the speech classification model.



Interpretation of Likelihood PDF



From top to bottom, the graphs display the log-likelihood PDF of digits 0 through 9 as rendered by a Kernel Density Estimate. The x-axis is the log-likelihood of a digit belonging to an inputted GMM, and the y-axis is the probability density of the log likelihoods. This is a qualitative graph that shows what the likelihood equation produces, and how the MLC formula classifies digits. By nature of the log-scale, even though the x-values are on the wrong scale, they are still in the correct order relevant to each other. Therefore, the key feature to look for is how far to the right the peak of the PDF occurs, which means that it has a higher density at higher likelihoods. In this example, the GMM for 0 is passed in, and as expected, the PDF for 0 peaks furthest to the right.

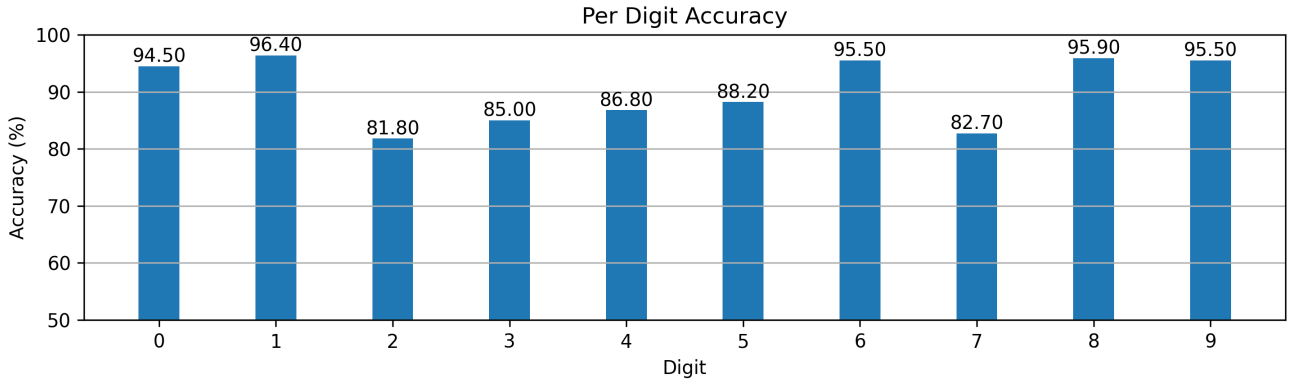
Challenges of Implementing MLC

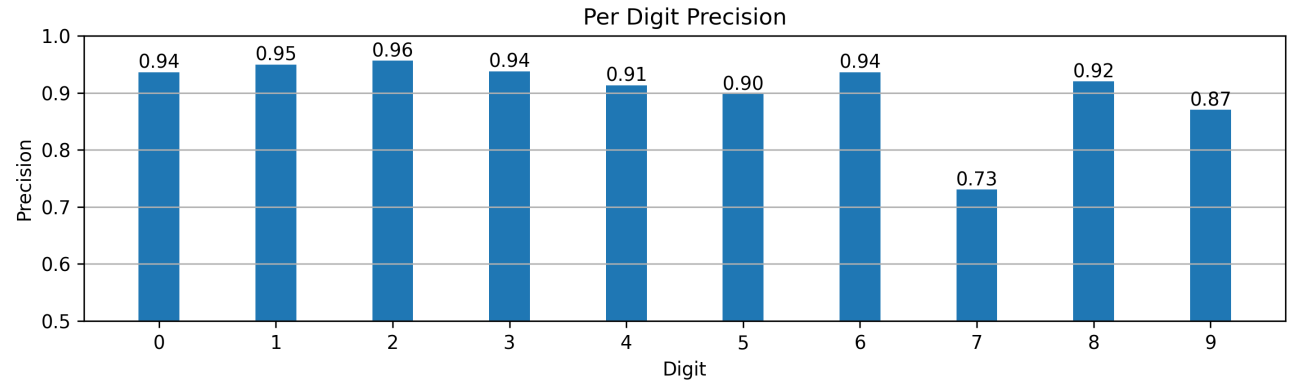
While Maximum Likelihood Classification is a powerful tool for assigning data to a category, it is imperative to ensure that the parameters of the model are input into the likelihood equation correctly. In this project, the model is thoroughly documented on the internet and the guidelines described the likelihood equation clearly, so it was much easier to transcribe that into code. However, if the problem required a different kind of model – which is a likely case in other applications – it would require much more tuning to ensure that the likelihood equation is correct.

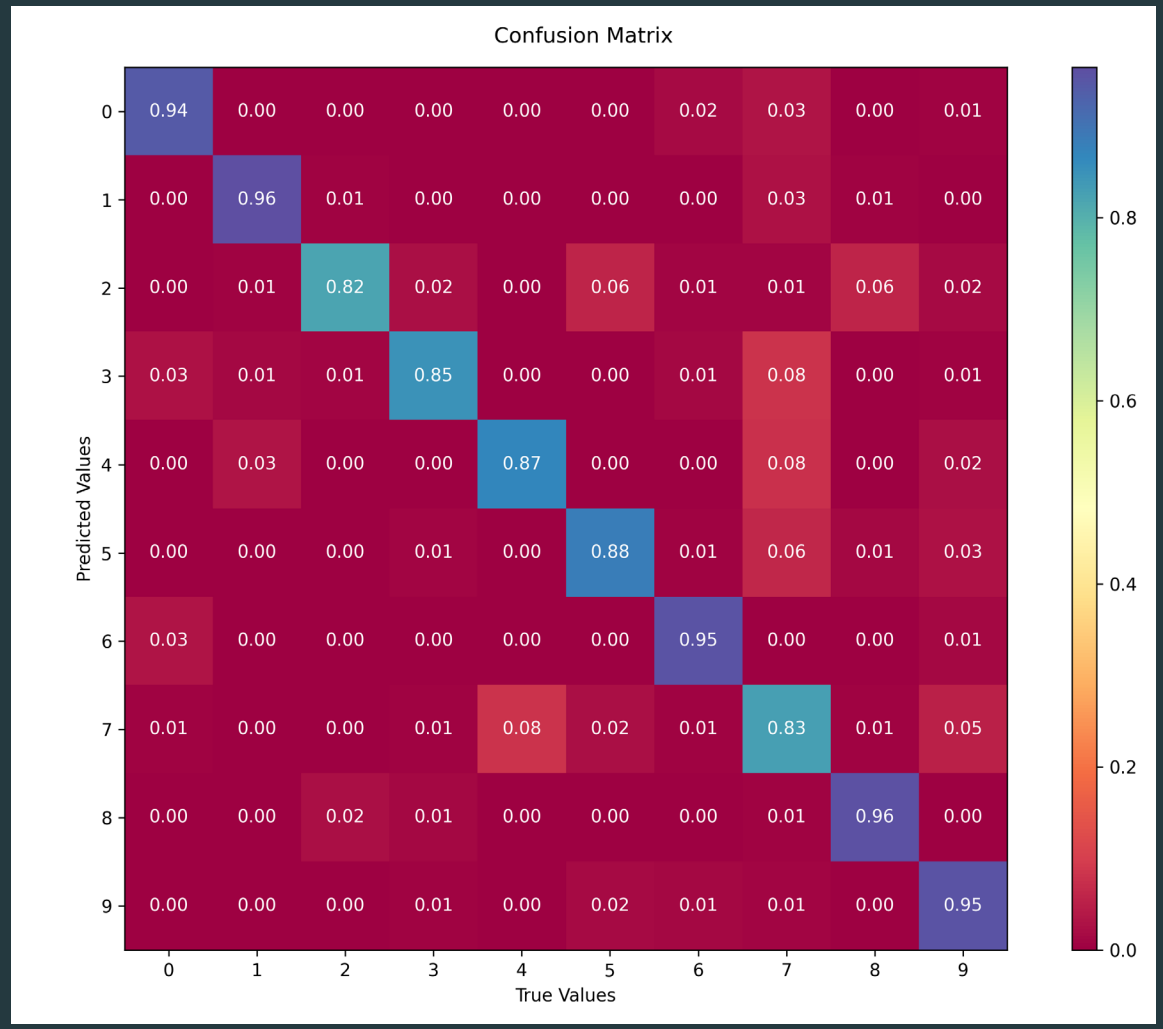
Still, while transcribing MLC to code, there were some issues along the way such as underflow. This was a simple fix that involved using a log sum instead of a product. Another challenge was ensuring that the inputs to the function were standardized across every possible hyperparameter change. For example, K-Means and EM output different values, so they each needed a helper function whose sole purpose was to reformat the data into a standardized object. Also, more data had to be recorded in the input dictionary as more hyperparameters were tested, which typically had effects on other parts of the code that could only be identified by running it and debugging errors. Overall, the MLC was very useful for the purpose of speech classification, but it was tedious to maintain as the code evolved into thousands of lines.

RESULTS

EM Performance





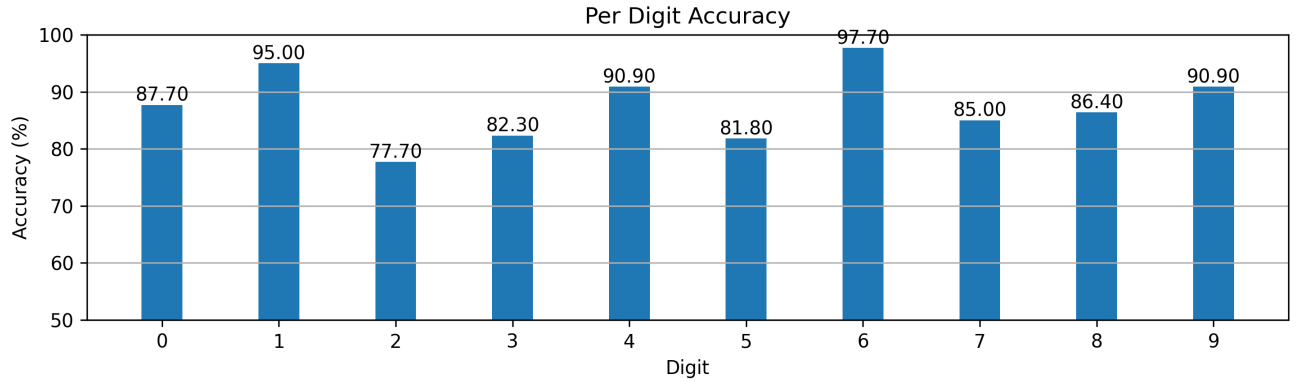


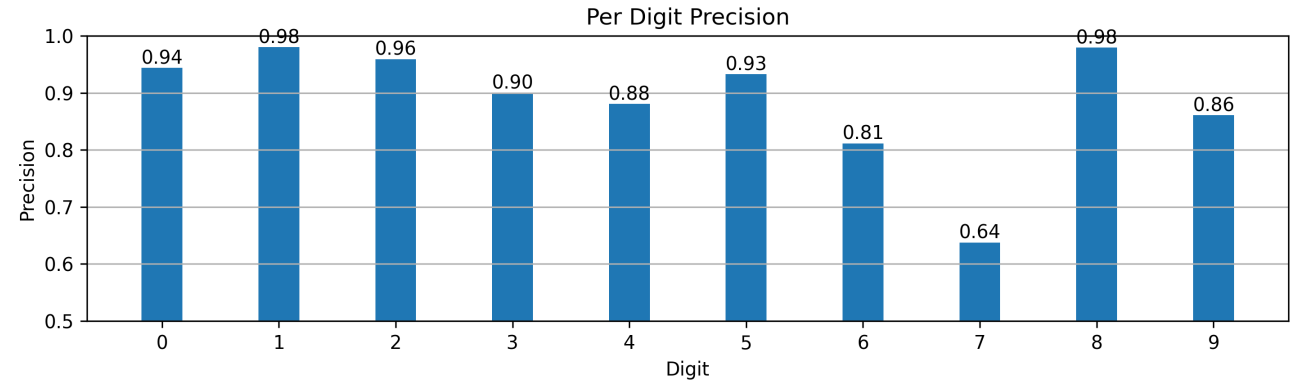
Avg Accuracy: 90.23%

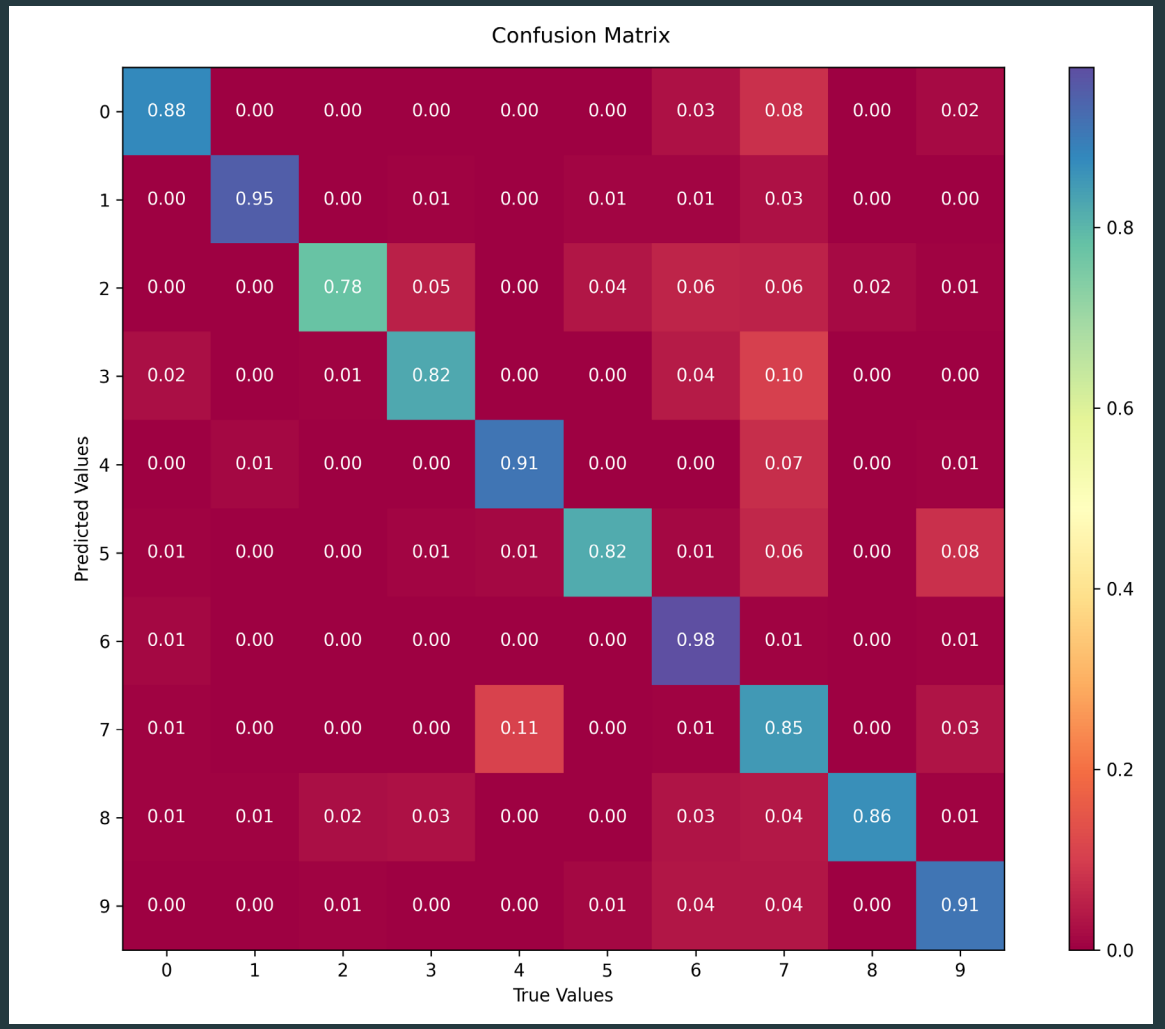
Avg Precision: 0.9051

Without dividing by gender or filtering for MFCCs, the resulting average accuracy and precision is good. The per digit accuracy is consistent, with the largest deficit being 15.6 percentage points. Precision is also mostly consistent, except for digit 7, which is performs much worse than the rest of the digits. This is because digits 3, 4, and 5 are misclassified as 7 at a high rate. Since the covariance constraint and clustering algorithm selection accounts for all digits at once, it is likely that digit 7 is an outlier from the rest. This would lead the overall optimal covariance constraints and clustering algorithm to perform very well for all digits except 7, which is exactly what is displayed in the precision bar graph. Digits 0, and 1 perform the best for both accuracy and precision, which indicates that the models are well balanced for bias and variance.

K-Means Performance





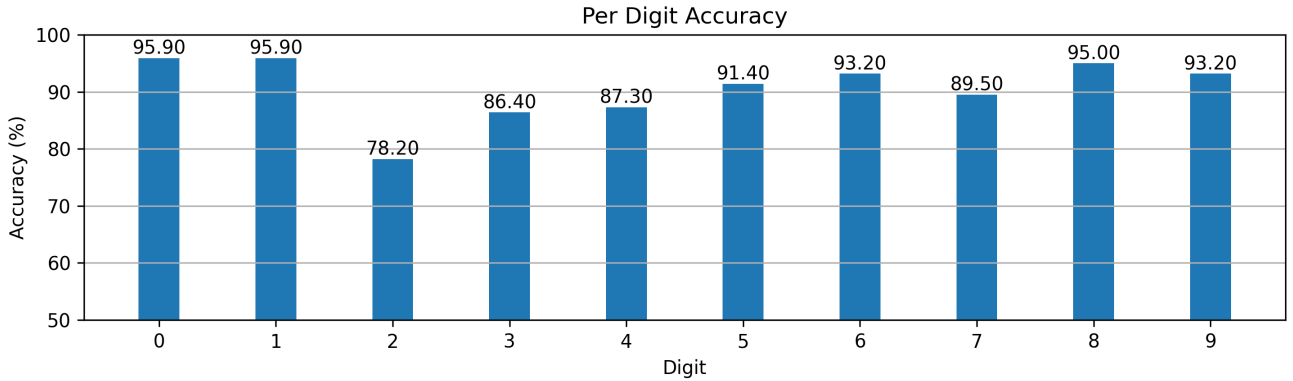


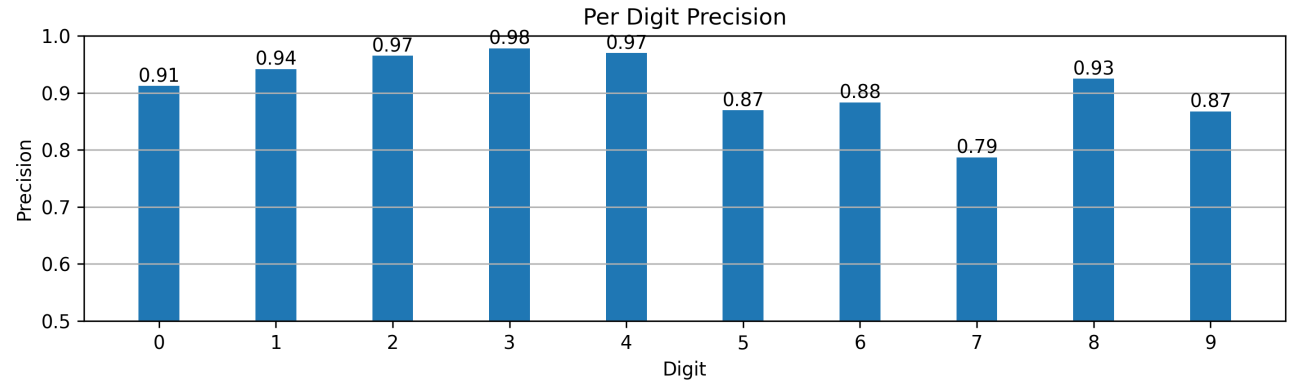
Avg Accuracy: 87.54%

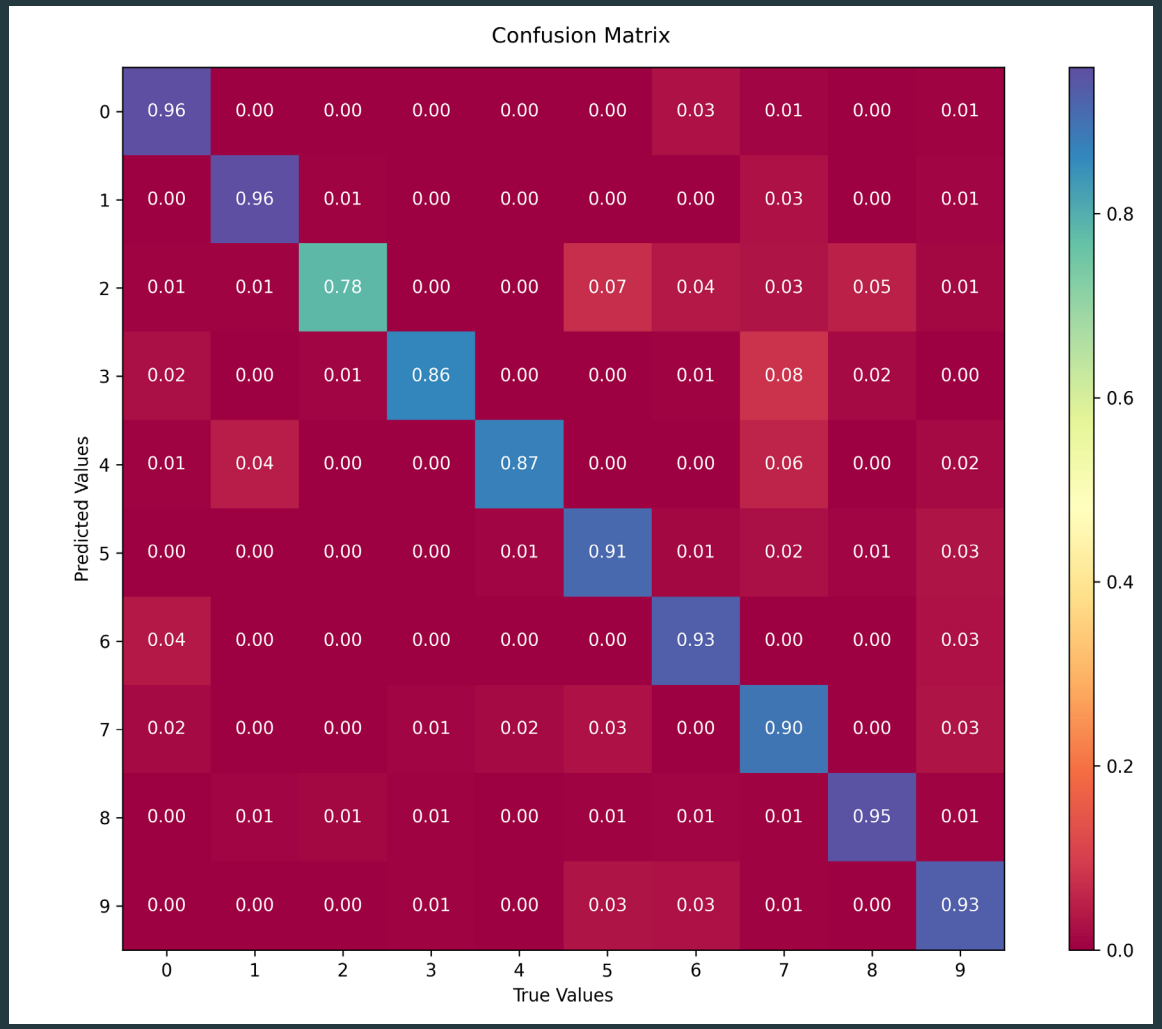
Avg Precision: 0.8887

Without dividing by gender or filtering for MFCCs, the resulting average accuracy and precision is good, but still not as good as EM. The per digit accuracy is still relatively consistent, but all the values are lower. For K-Means, the largest accuracy deficit is 20 percentage points. The precision plot displays similar trends to the EM precision plot, with digit 7 still being the point of concern. Again, a large source of precision loss is because digits 3, 4, and 5 are misclassified as 7 at a high rate, but other sources are sprouting from digits 0 and 2. It is interesting that 0's misclassification rate over doubled compared to EM since 0 has the same number of clusters as 7 in this model. This indicates that EM is better at creating tight, informative clusters under distinct full covariance constraints when compared to EM. Digit 1 still performs the best for accuracy and precision, which means that its GMM may not be as heavily influenced by the clustering algorithm. However, Digit 0 drops more in accuracy, so it is more dependent on the clustering algorithm.

Results of Tuning MFCCs





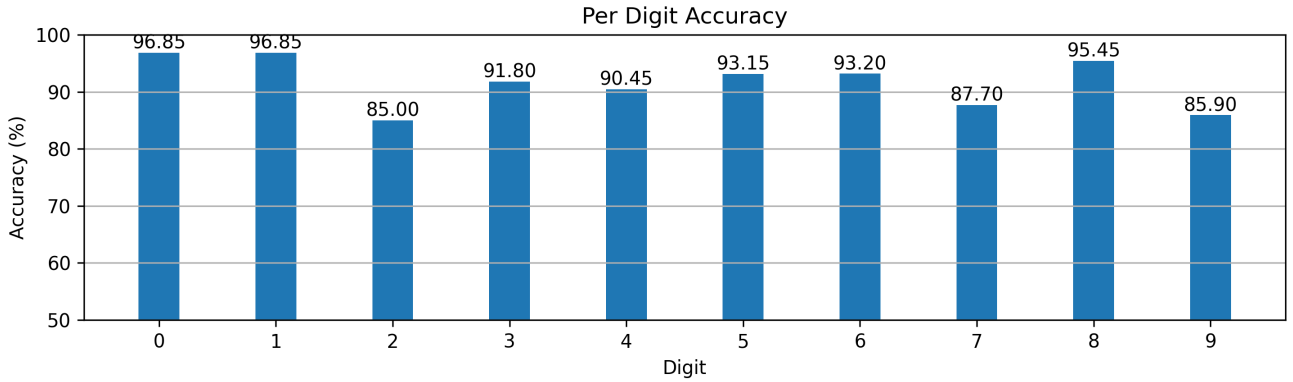


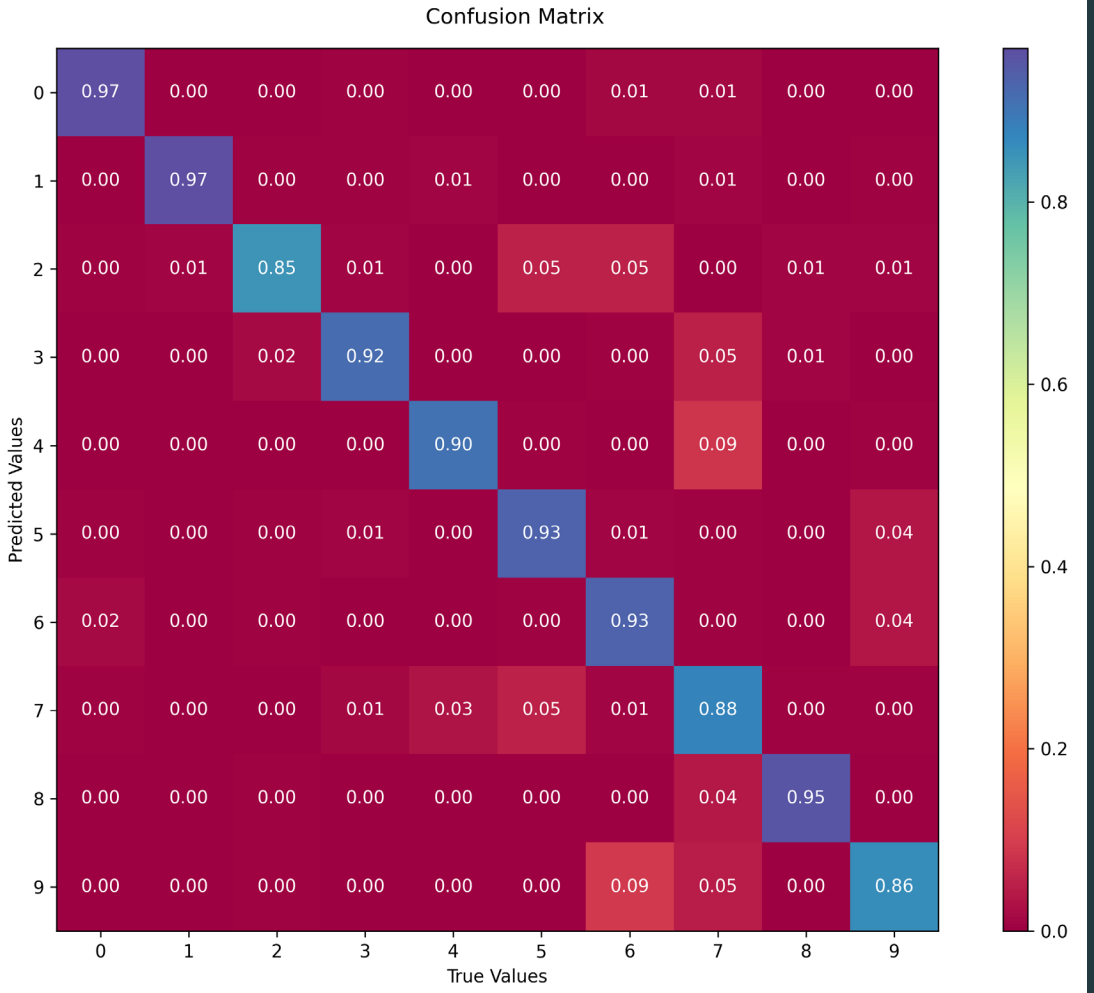
Avg Accuracy: 90.60%

Avg Precision: 0.9101

Without dividing by gender and with MFCC filtration, the resulting average accuracy and precision is much better than K-Means, but comparable to the EM result. An interesting result of filtering MFCCs is that the accuracy of digit 7 is up 6.8 percentage points, but accuracy of digit 2 loses 3.6 percentage points when compared to the EM slide. This is likely a trade-off that was made by the greedy MFCC selection algorithm. The algorithm chose to keep 9 of 13 MFCCs, and at least one of the removed MFCCs may have been more important to digit 2 than digit 7 but resulted in a lower net accuracy. As a result, other MFCCs that propagate higher quality information for digit 7 were kept instead.

Separating by Gender



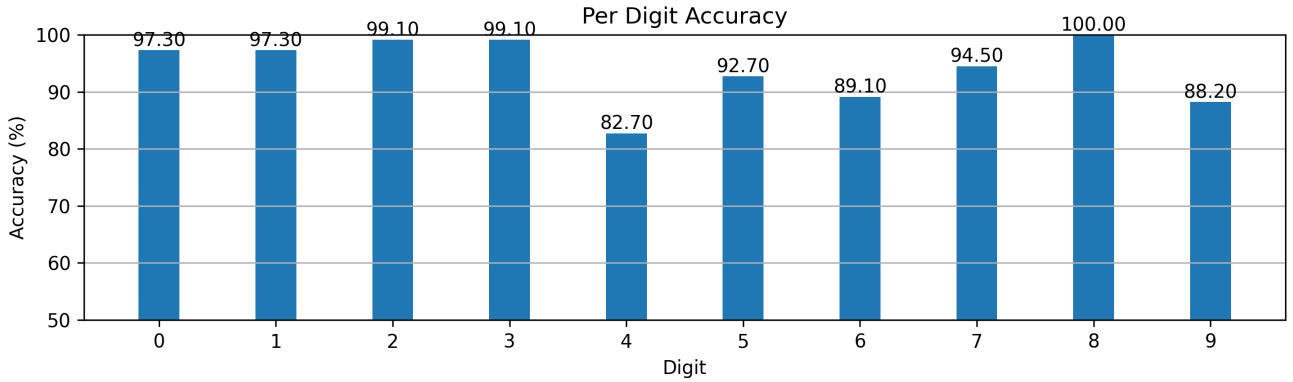


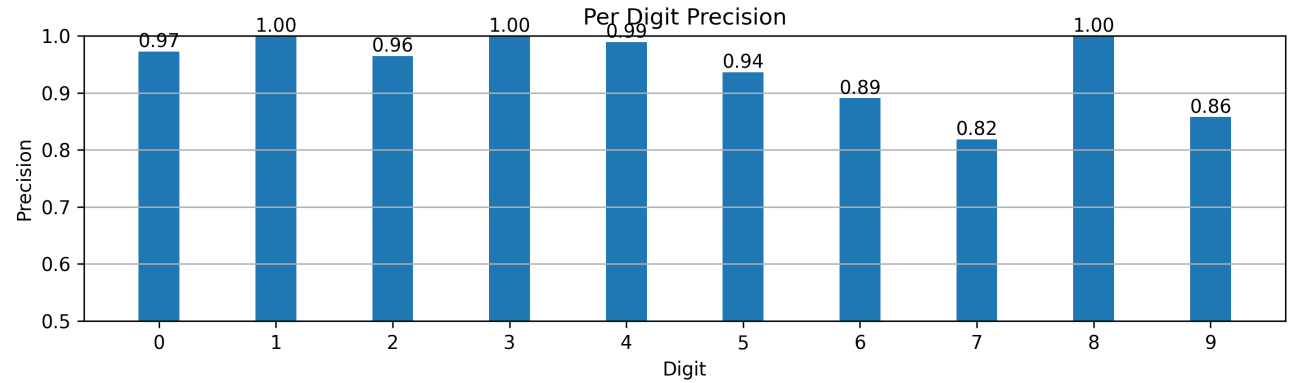
Avg Accuracy: 91.64%

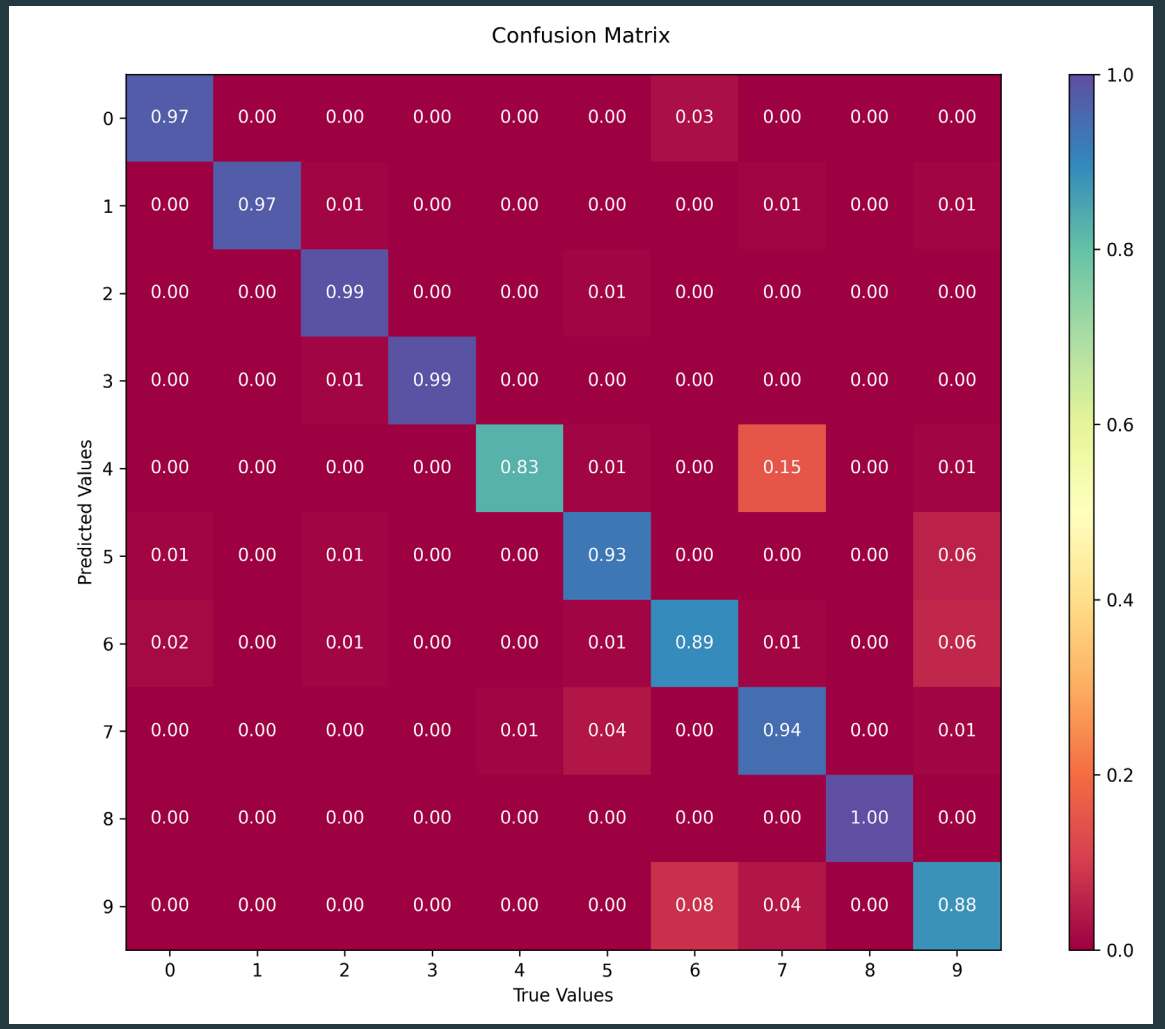
Avg Precision: 0.9205

When gender is factored into training and testing the model, the average accuracy and precision increases slightly. The issue of lower precision for digit 7 still exists, with digits 3, 4, and 9 being the largest sources of misclassification. This issue has persisted through every model change that has been presented, which implies that the problem may not be a direct result of the clustering algorithm, the MFCCs, or the covariance constraint. It must be that the model for digit 7 needs special customization separately from the rest of the digits. These plots have all been created under the assumption that the MFCCs, covariance constraints, and clustering algorithm are selected to optimize all digits combined, but it clearly does not represent the outlier that is digit 7 very well.

Female Speakers Performance





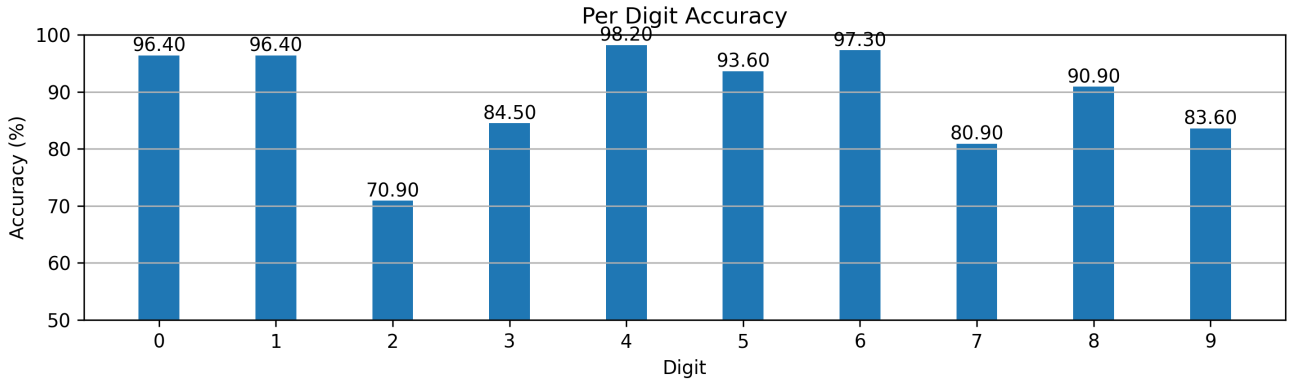


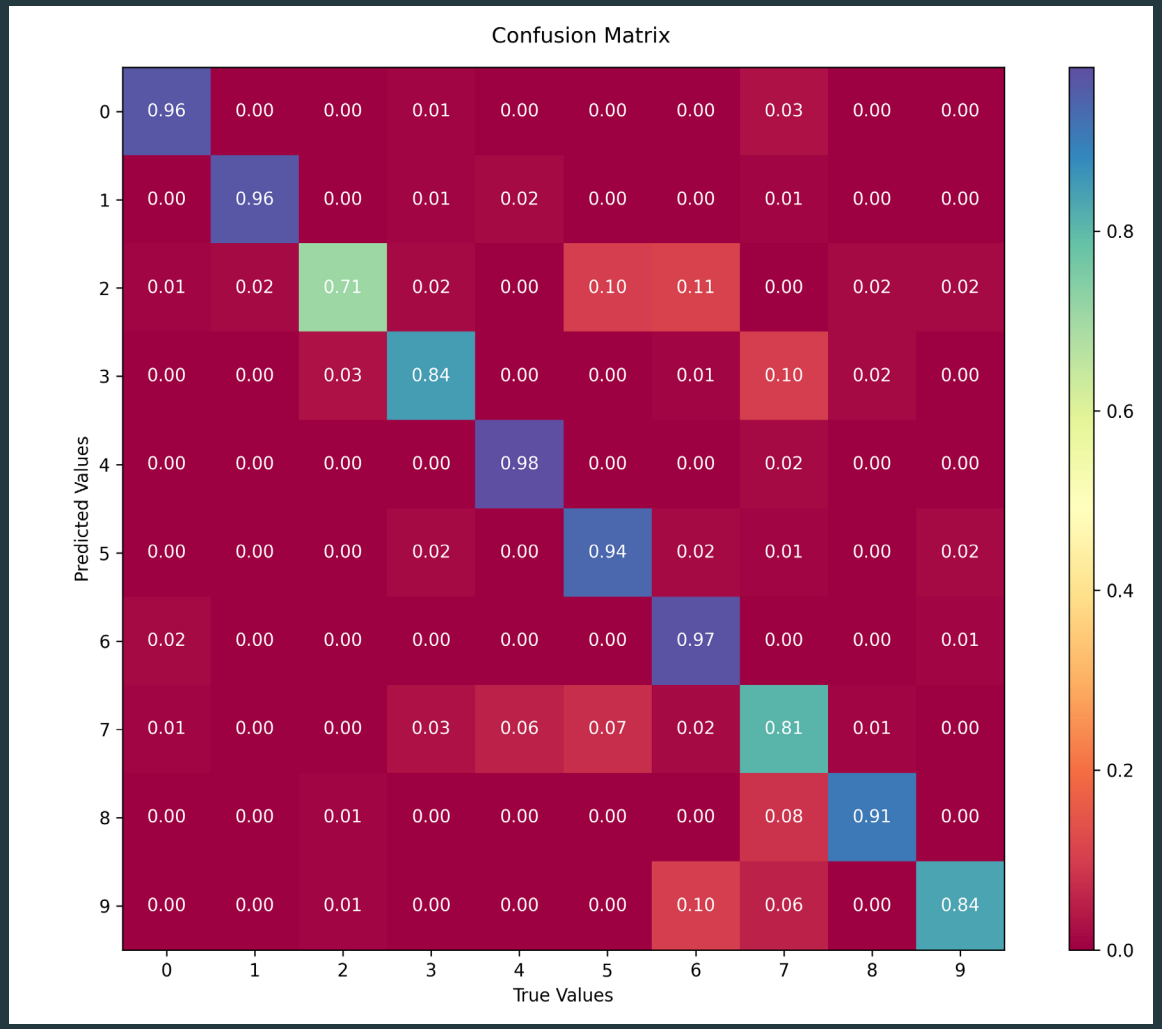
Avg Accuracy: 94%

Avg Precision: 0.9431

At a closer look of the female speaker classification results, it performs much better than the combined performance. Digits 0, 1, and 8 performed the best for females with a perfect accuracy and precision for digit 8. This could be because digit 8's ideal number of clusters is higher than most other digits aside from 9. The Number of Clusters (Quantitative) plots displayed that the effect of increasing cluster number on accuracy for each digit typically plateaued before hitting 5 clusters. Digit 4's low accuracy is primarily due to being mistakenly classified as a 7 15% of the time. Although the performance of female speakers is the best overall, it also contains the highest misclassification rate of all the other results in the section. To address the issue of low precision for digit 7, it would be useful to find the relationship between 4 and 7 since 7's consistently low precision is related to 4's consistently low accuracy.

Male Speakers





Avg Accuracy: 89.27%

Avg Precision: 0.8998

The average results are still good, but it still performs quite a bit worse than female classification. One possible explanation is that the data for male utterances is lower quality than the data for female utterances. Again, digit 7 performs very poorly in terms of accuracy and precision. A new observation of this plot is that the accuracy of digits 2 and 3 drop significantly compared to the corresponding female accuracy, whereas digit 4 improves a lot. This weakens my previous theory that the relationship between digits 4 and 7 could be the reason for 7's consistently low precision, but it could be that digit 4 is the source of the misclassification problem for females, whereas digit 3 is the source of the misclassification problem for males. It is possible that some digits' models need to be customized instead of using the same covariance algorithms, MFCCs, and clustering algorithms for everything.

CONCLUSIONS

Summary of Project Model and Results

Multiple modeling choices came into play with this project. Some of them were essential choices that directly impact the GMM, whereas others were simply extensions to try to break past 90% accuracy. To reiterate, these are all the modeling choices that were considered:

- K-Means vs. EM
- Number of clusters for each digit
- Covariance constraints
- Training and testing genders separately
- Filtering MFCCs

High Impact Modeling Choices

Some parameters had large impacts on the accuracy and precision of the model. The most impactful parameter was the number of clusters. Finding the right number of clusters for each digit could change its classification performance from 10% to over 90%. This was also one of the harder parameters to tune because changing one digit's number of clusters affected other digits as well. Digit 9's parameter in particularly appeared to have the largest effect on other digits, so finding the best number required a mix of evaluating the qualitative plots, quantitative plots, and running multiple trials. Aside from digit 9, tuning the number of clusters parameter was quite straightforward.

Another very impactful parameter was the covariance constraints. Very rigid constraints like spherical covariance performed significantly worse compared to the other covariance constraints whether it was using the combined data or stratifying by gender. This also would have changed the clustering algorithm selection for tied spherical since EM and K-Means performed very similarly.

On a related thread, the selection of EM vs. K-Means also had the potential to make a large impact depending on the covariance constraint. EM was usually always better, but the size of the performance deficit was related to the covariance type. For tied and distinct full covariance and tied spherical, the deficit was minimal. On the other hand, the performance of tied and distinct diagonal and distinct spherical was heavily dependent on the clustering algorithm. It is important to account for the clustering algorithm while selecting the covariance constraint, and vice versa.

Low Impact Modeling Choices

The two low impact modeling choices also served as extensions to the project. MFCC filtration was not necessary to achieve higher accuracy and precision, but it still improved the model slightly. Even without any MFCC filtration, the model was able to achieve 90% accuracy and .90 precision. However, breaking past 90% accuracy and .90 precision was extremely difficult through only modifying the clustering algorithm, covariance constraint, and number of clusters. The improvement through MFCC filtration shows that there are some MFCCs that confuse the speech recognition model more than it helps.

The other extension that helped push the model past 90% performance was separating by gender. The resulting overall accuracy and precision were 91.64% and 0.9205, which was a big break, but was not a large improvement in the grand scheme of the project. The key takeaway of exploring this model choice was comparing gender differences. It revealed that the female model scored much higher than the average overall performance, whereas the male model scored lower. This could be because the audio features that are embedded in MFCCs are of higher quality for women than men. According to Smorenburg, female speakers have higher mean pitch and formants, as well as a larger pitch range[1]. The MFCCs may be better at capturing female pitches, which would create a performance deficit between genders.

[1]Smorenburg, L., & Chen, A. (2020). The effect of female voice on verbal processing. Speech Communication, 122, 11–18. https://doi.org/10.1016/j.specom.2020.04.004

Specifying a Single System

Based on the extensive exploration of each hyperparameter and their results, this is a summary of the ideal selections.

- Covariance Constraint: Distinct Full
- Female MFCCs: [1, 2, 4, 6, 7, 10, 12]
- Male MFCCs: [3, 4, 6, 7, 8, 10, 11]
- Clustering Algorithm: Expectation Maximization
- Number of clusters:

Digit	0	1	2	3	4	5	6	7	8	9
# of Clusters	4	4	3	4	3	3	4	4	5	5

Despite the initial positive results of separating covariance constraints by gender, it ended up performing worse than using the overall covariance constraints. On the other hand, assigning MFCCs by gender does increase the accuracy of classification.

Specifying a Single System w/o Gender

It is important to note that the single system specified on the previous slide only works if the provided data includes information about the gender of each utterance. If that information is missing, then the following system would be the best:

- Covariance Constraint: Distinct Full
- Overall MFCCs: [1, 2, 4, 5, 6, 7, 8, 10, 12]
- Clustering Algorithm: Expectation Maximization
- Number of clusters:

Digit	0	1	2	3	4	5	6	7	8	9
# of Clusters	4	4	3	4	3	3	4	4	5	5

Evaluation of Selected System w/ Gender

The selected system resulted in the highest accuracy of classification and revealed important information regarding gender differences. It resulted in better overall performance for the female data when compared to the male data, which implies that there could be bias in the model or in the data itself. Although EM is computationally more intensive than K-Means, it is a great choice for speech recognition. It is very flexible and performs equivalent or better than K-Means for any covariance constraint. Therefore, if this model were to be expanded with further training data, EM would be the most robust to any newly introduced variations. This system performed especially well for digits 0, 1, and 8. There was a lower deficit between female and male performance for those digits when compared to the rest of the digits. Therefore, the models for 0, 1, and 8 have a good balance between bias and variance.

One area that the model was not able to fix was the misclassification of various numbers as 7, resulting in low precision for 7. Despite experimenting with different combinations of hyperparameters, 7 consistently performed the worst compared to all other digits. A potential solution to this issue is to tune digit 7's model separately from the rest of the digits as it is possible that digit 7 is simply an outlier. Since all the system's parameters except for cluster number was selected to optimize all digits at once, it loses information on the nuances of each utterance of 7. Another area of improvement is selecting hyperparameters and evaluating results based on more measurements than just accuracy and precision.

Personal Takeaways

One thing I am appreciative of is the built in check-ins for the project. After completing everything, I can see how it might have been easy to procrastinate the work and how difficult it would have been to finish it as a result. Aside from this, I liked the open-endedness of the project and the slide doc. There were no required visualizations, so I had lots of freedom in deciding how I wanted to present my information. Being able to present technical material in a digestible manner is a skill that will undoubtedly come in handy in future projects.

This project had many extensions to explore from MFCC filtration to gender stratification to analysis frame sampling. It would be interesting if I could go even further by analyzing the temporal relationship of the MFCCs.

REFERENCES

Slide Doc References

[1]Sarker, I.H. Machine Learning: Algorithms, Real-World Applications and Research Directions. SN COMPUT. SCI. 2, 160 (2021). https://doi.org/10.1007/s42979-021-00592-x

[2]Spechbach H, Gerlach J, Mazouri Karker S, Tsourakis N, Combescure C, Bouillon P A Speech-Enabled Fixed-Phrase Translator for Emergency Settings: Crossover Study

[3]M. Admane and S. Patil, "Modeling Lung Cancer Diagnosis using Bayesian Network Inference," 2022 International Conference on Smart Generation Computing, Communication and Networking (SMART GENCON), Bangalore, India, 2022, pp. 1-6, doi: 10.1109/SMARTGENCON56628.2022.10084230.

[4]Bedda,Mouldi and Hammami,Nacereddine. (2010). Spoken Arabic Digit. UCI Machine Learning Repository. https://doi.org/10.24432/C52C9Q.

[5]Khanna, M. (2023, July 19). *Classification problem: Relation between sensitivity, specificity and accuracy*. Analytics Vidhya. https://www.analyticsvidhya.com/blog/2021/06/classification-problem-relation-between-sensitivity-specificity-and-accuracy/

[6]Senarathna, Sisitha & Hemapala, K T M U. (2020). Optimized Adaptive Overcurrent Protection Using Hybridized Nature-Inspired Algorithm and Clustering in Microgrids. Energies. 13. 3324. 10.3390/en13133324.

[7](Heard, N. (2021). Clustering and Latent Factor Models. In: An Introduction to Bayesian Inference, Methods and Computation. Springer, Cham. https://doi.org/10.1007/978-3-030-82808-0_11

[8]Silva J, Vaz P, Martins P, Ferreira L. Reliability Estimation Using EM Algorithm with Censored Data: A Case Study on Centrifugal Pumps in an Oil Refinery. Applied Sciences. 2023; 13(13):7736. https://doi.org/10.3390/app13137736

[9]Smorenburg, L., & Chen, A. (2020). The effect of female voice on verbal processing. Speech Communication, 122, 11–18. https://doi.org/10.1016/j.specom.2020.04.004

Code References

Numpy: Harris, C.R., Millman, K.J., van der Walt, S.J. et al. Array programming with NumPy. Nature 585, 357–362 (2020). DOI: 10.1038/s41586-020-2649-2

Scipy: Pauli Virtanen, Ralf Gommers, Travis E. Oliphant, Matt Haberland, Tyler Reddy, David Cournapeau, Evgeni Burovski, Pearu Peterson, Warren Weckesser, Jonathan Bright, Stéfan J. van der Walt, Matthew Brett, Joshua Wilson, K. Jarrod Millman, Nikolay Mayorov, Andrew R. J. Nelson, Eric Jones, Robert Kern, Eric Larson, CJ Carey, İlhan Polat, Yu Feng, Eric W. Moore, Jake VanderPlas, Denis Laxalde, Josef Perktold, Robert Cimrman, Ian Henriksen, E.A. Quintero, Charles R Harris, Anne M. Archibald, Antônio H. Ribeiro, Fabian Pedregosa, Paul van Mulbregt, and SciPy 1.0 Contributors. (2020) SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. Nature Methods, 17(3), 261-272.

Matplotlib: J. D. Hunter, "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007

Pandas: McKinney, W., & others. (2010). Data structures for statistical computing in python. In Proceedings of the 9th Python in Science Conference (Vol. 445, pp. 51–56).

Scikit-learn (EM and K-Means): Pedregosa, F., Varoquaux, Ga"el, Gramfort, A., Michel, V., Thirion, B., Grisel, O., … others. (2011). Scikit-learn: Machine learning in Python. Journal of Machine Learning Research, 12(Oct), 2825–2830.

Python 3: Van Rossum, G., & Drake Jr, F. L. (1995). Python reference manual. Centrum voor Wiskunde en Informatica Amsterdam.

COLLABORATORS

Collaborators

Throughout the project, there were a few times where I ran into a wall with how to implement something. In those times, I worked with Jerry Worthy and Nolan Gelinas. Typically, if one of us had an implementation issue, someone else had already figured it out, so it was easy to help each other debug. Also, as we created our slide doc presentation, we would share ideas for how to visualize the results. More specifically, Jerry helped me figure out how to make a heat map on top of the confusion matrix, and Nolan gave me the idea of plotting each combination of K-Means vs. EM, Tied vs. Distinct, and Full vs. Diagonal vs. Spherical in a bar graph for side-by-side comparison. Aside from small parts of the code and visualization, I worked on this project independently. I also helped Ryan Devries throughout project check-ins by helping him debug his code and explaining what the check-ins were meant to accomplish.