Recognizing Spoken Digits

Jay Esemudje

ECE 480

Table of Contents

Project Summary

This goal of this project is to explore Gaussian Mixture Models for the distributions of cepstral coefficients of spoken Arabic digits. It investigates the impact that certain modeling choices will have on the maximum likelihood classification of these digits. Personally, this project is interesting because it has allowed me to expand my understanding of GMMs and the way their prediction capabilities vary depending on parameter estimation methods and covariance restrictions. Being able to apply them in a practical way was extremely insightful.

This project is useful because it has real-world applications in domains like automated telephone systems, banking, and accessibility. Moreover, the focus on Arabic contributes a layer of importance because Arabic is a widely spoken language that presents distinct challenges compared to other languages such as English.

Beyond speech recognition, this framework could be extremely useful in environmental research, specifically animal cry classification. In my Rainforest Engineering class, one of the projects we're working on is the use of ML to identify birds via their cries. The ideas and modules I've been able to explore via this project are extremely relevant to that investigation.

Introduction: Phonemes

Phonemes are the smallest distinct units of sound that can be distinguished among words in a particular language. They are the building blocks of a phonetic representation of speech and are not synonymous to letters or syllables. For example, the word “Table” has two syllables and 5 letters but it has 4 phonemes. This is shown below:

Syllables: ta-ble

Phonemes: /t/-/eɪ/-/b/-/l/

These units of sound are captured in a given speech signal’s frequency spectrum.

Introduction: Cepstral Coefficients

Cepstral coefficients represent speech signals in a compact way by modeling their spectral envelopes. They are the set of values that represent the spectral envelope of a signal, obtained by taking the inverse Fourier transform of the logarithm of the signal's power spectrum.

Mel-Frequency Cepstral Coefficients (MFCCs) are specifically designed to mimic the human auditory system. They use the Mel scale to capture frequencies as perceived by humans and emphasize the spectral characteristics that are most important for distinguishing phonemes.

Machine learning models map these MFCCs to phonemes by recognizing patterns associated with specific phonemes. This allows the models to predict digits from MFCC data. However, this mapping is probabilistic because cepstral features don't directly correspond to phonemes.

Introduction: Gaussian Mixture Models

Gaussian Mixture Models are probabilistic models used to represent the underlying distribution of data as a combination of multiple Gaussian (normal) distributions. They are commonly used in unsupervised learning, clustering, and density estimation. They can perform classification on observations with softer boundaries i.e. overlap between components.

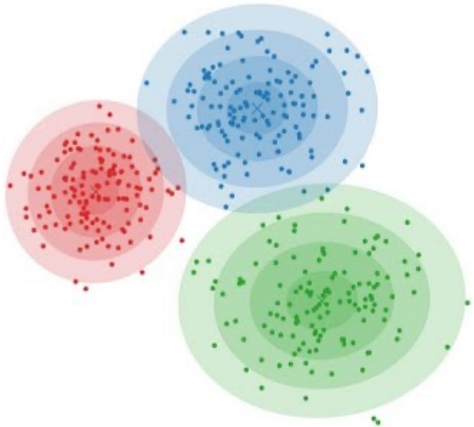
The probability density function of a GMM is given as:

$$\sum_{k=1}^{K} \pi_k N(x_n | \mu_k, \Sigma_k)$$

Where π_k, μ_k and Σ_k are the mixing probability, mean and covariance of the kth component respectively.

These parameters can be initialized either by Expectation Maximization or K-Means. Both initialization methods will be explored in this project.

Gaussian Mixture Models (GMMs) are particularly useful for this project because the boundaries between spoken digits are not always distinct. The digits 6 and 7 share **/s/-/ɪ/** phonemes. Furthermore, as speech signals are typically not linear, GMMS are more capable of approximating distributions than more simple models.



Introduction: Arabic Digits

As stated earlier, the MFCCs that will be used for classification are related to the phonemes of the digits. Here is a breakdown of phonemes for the digits, 0 -9, spoken in Arabic.

Number	Arabic	Arabic Translation	Phonemes	Phoneme Count
0	صفر	**ṣifr**	s-i-f-r	4
1	واحد	**wāḥid**	w-a–h-i-t	5
2	اثنان	**ithnān**	i-th-n-a-n	5
3	ثلاثة	**thalātha**	th-eh-l-eh-th-eh	6
4	أربعة	**arbaʿa**	a-r-b-ah	4
5	خمسة	**khamsa**	h-a-m-s-eh	5
6	ستة	**sitta**	s-i –t-eh	4
7	سبعة	**sabʿa**	s-a –b-ah	4
8	ثمانية	**thamāniya**	Th-e-m-e-n-i-y-eh	8
9	تسعة	**tisʿa**	T-i-s-ah	4

Dataset Breakdown

The dataset for this project contains time series of MFCCs representing spoken Arabic digits. It includes data from 44 male and 44 female native Arabic speakers. The data is divided into two text files: Train_Arabic_Digit.txt and Test_Arabic_Digit.txt.

Each line in these files contains 13 MFCC coefficients in ascending order, separated by spaces, corresponding to a single analysis frame. Lines are grouped into blocks, where each block consists of 4–93 lines separated by blank lines. Each block corresponds to one speech utterance of an Arabic digit, and multiple consecutive blocks represent the utterances of the same digit.

In Train_Arabic_Digit.txt, there are 660 blocks for each spoken digit. The first 330 blocks are from male speakers, and the next 330 are from female speakers. For example:

- Blocks 1–660 represent digit "0" (10 utterances from 66 speakers, split evenly between male and female).
- Blocks 661–1320 represent digit "1" (10 utterances from the same 66 speakers).

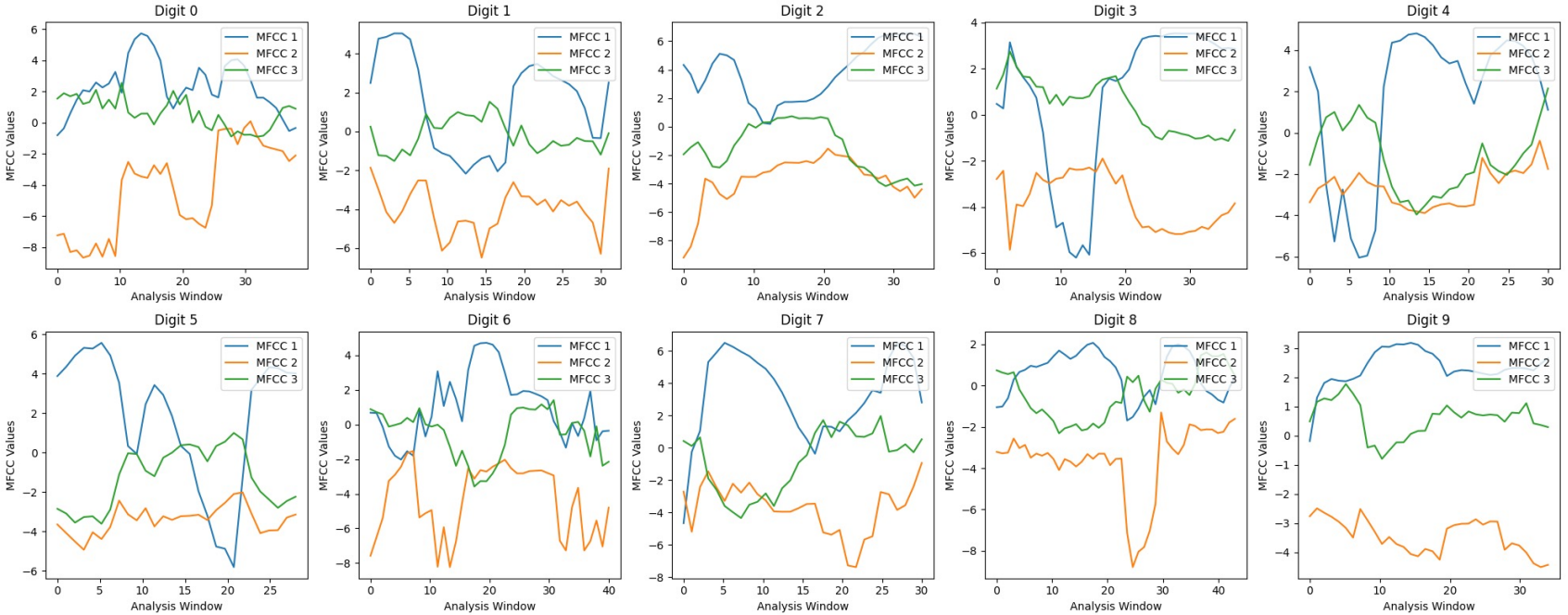
In Test_Arabic_Digit.txt, there are 220 blocks for each digit. The first 110 blocks are from male speakers, and the next 110 are from female speakers. For example:

- Blocks 1–220 represent digit "0" (10 utterances from 22 speakers, 11 male and 11 female).
- Blocks 221–440 represent digit "1" (10 utterances from the same 22 speakers).

Note that the speakers in the test dataset are different from those in the training dataset.

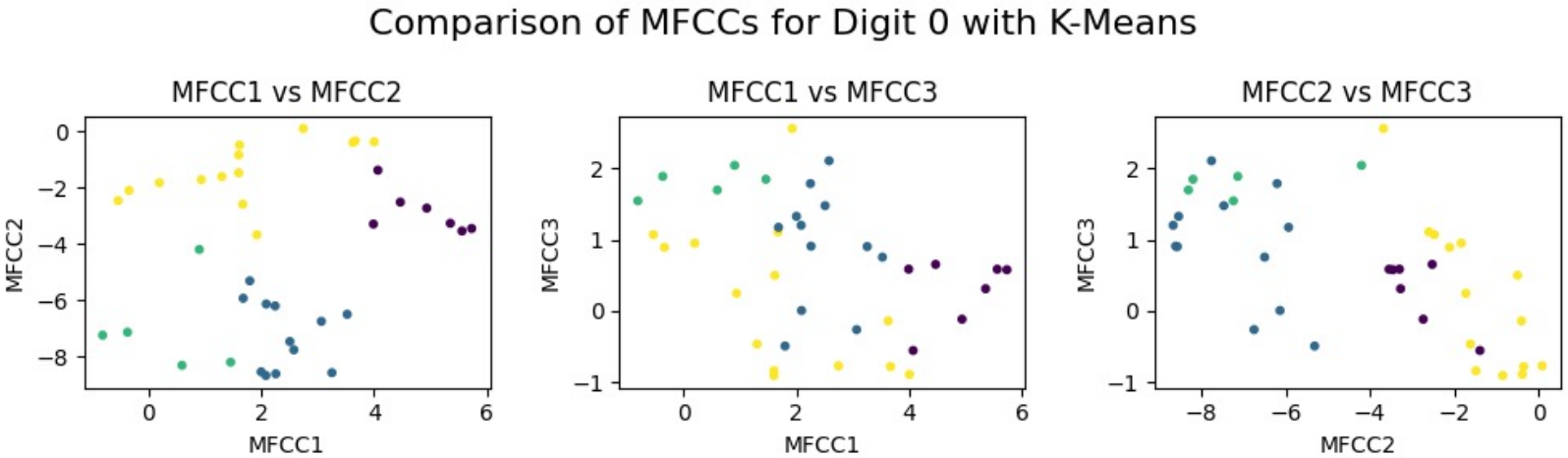
Data Visualization: First 3 MFCCs as a Function of Analysis Window

This visualization shows the MFCC variations that exist between speakers (frames/analysis windows) within an utterance of the same digit. An ideal model should be able to capture these variations.



Visualization: MFCC Comparisons

These helps visualize whether the data is naturally separable or whether clusters overlap significantly. Clear separations in the scatter plot suggest the presence of distinct patterns such as different phonemes. Note that with the use of the K-Means algorithm, there are hard boundaries between clusters. There are at about 4 phonemes in the digit 0 and this is captured particularly well in the MFCC1 vs MFCC2 plot.



Methodology: Parameter Estimation via K-Means

As mentioned earlier, GMMs are to be used for this project. GMM parameters can be estimated via K-Means or Expectation Maximization. This project explores both to draw conclusions about their impact on overall model performance.

K-Means is an unsupervised classification algorithm that can be used to estimate the parameters (π_k, μ_k , Σ_k) that are used in GMMs. The parameters are calculated as follows:

$\pi_k = \frac{n_k}{N}$, where n_k is the number of observations in a cluster and N is the total number of observations

μ_k is the mean of the observations in the kth cluster and under K-means, these centroids are initially randomized but move closer to their true location with each iteration of the algorithm.

Σ_k is the covariance of the observations in the kth cluster. This is calculated based on the cluster assignments performed by the algorithm.

The goal of the K-Means algorithm is to minimize the distance between the centroid of a cluster and its assigned observations. Thus, the cycle of cluster centroids moving closer to their correct position and observations getting re-assigned to those clusters continues until that distance is minimized or a maximum number of iterations is reached.

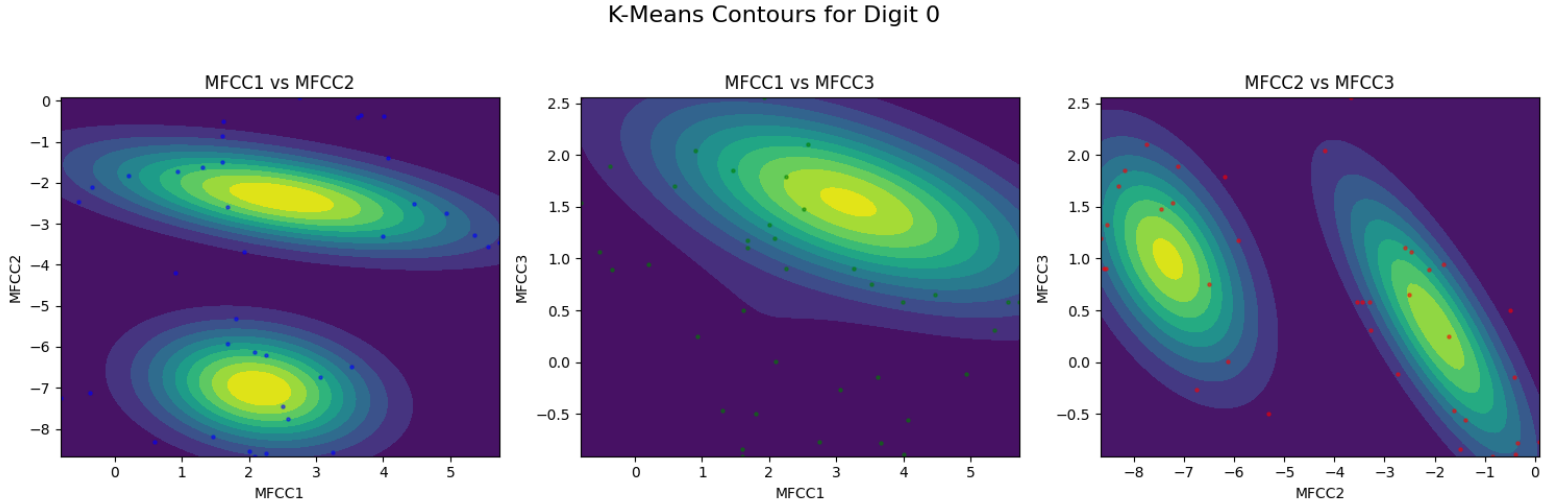
K-Means Analysis

Here are the advantages and disadvantages of parameter estimation via K-Means:

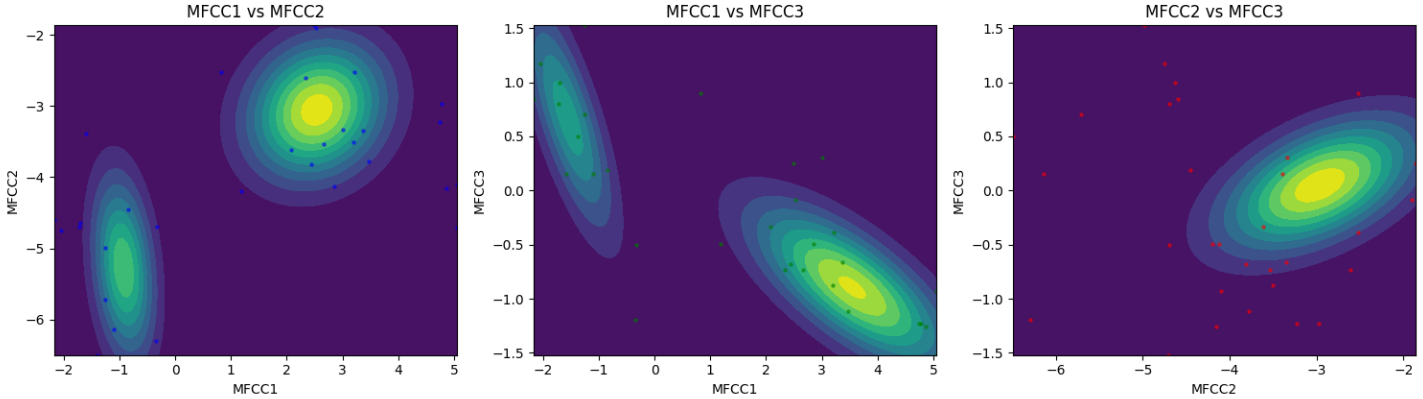
Advantages	Disadvantages
K-Means is faster because it is less computationally intensive	K-Means assumes hard boundaries, reducing model accuracy when there is overlap between observations
K-Means can handle larger datasets more efficiently than EM, which can struggle with large covariance matrix computations.	K-Means assumes spherical clusters, reducing model flexibility

Visualization: Digit GMMs via K-Means (0-2)

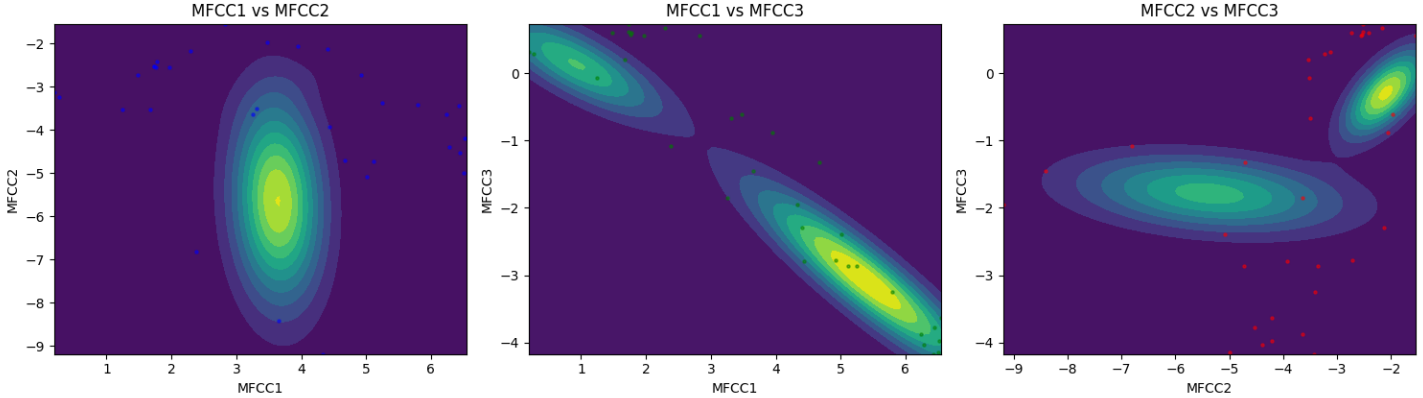
Below are clusters created via K-Means. They are clearly delineated, indicative of their hard boundaries. This leaves some observations poorly classified. The next slide contains similar plots.



K-Means Contours for Digit 1



K-Means Contours for Digit 2



Methodology: Parameter Estimation via EM

Expectation-Maximization (EM) is another unsupervised classification algorithm that can be used to estimate the parameters (π_k, μ_k , Σ_k) that are used in GMMs. It makes use of component responsibilities to calculate the parameters. Component responsibilities are the probabilities that a given observation belongs to a specific Gaussian component within the mixture model.

EM is broken up into the **E-Step** and the **M-Step**. During the E-Step, component responsibilities are calculated using initialized parameters. This is given as:

$$r_{n,k} = \frac{\pi_k N(x_n|\mu_k, \Sigma_k)}{\sum_{j=1}^{K} \pi_j N(x_n|\mu_k, \Sigma_k)}$$

EM Cont.

The M-step involves the estimation of model parameters given component responsibilities. These are given as follows:

$\pi_k = \frac{N_k}{N}$, where π_k is effective probability of the kth mixture component given as,
$N_k = \sum_{n=1}^{N} r_{n,k}$ and N is the total number of observations

$\mu_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{n,k}\, x_n$ where μ_k is the effective mean of the kth mixture component

$\Sigma_k = \frac{1}{N_k} \sum_{n=1}^{N} r_{n,k}\, (x_n - \mu_k)(x_n - \mu_k)^T$ where Σ_k is the effective covariance of the Kth mixture component.

The E and M steps alternate until there is convergence of the parameters or a maximum number of iterations is reached.

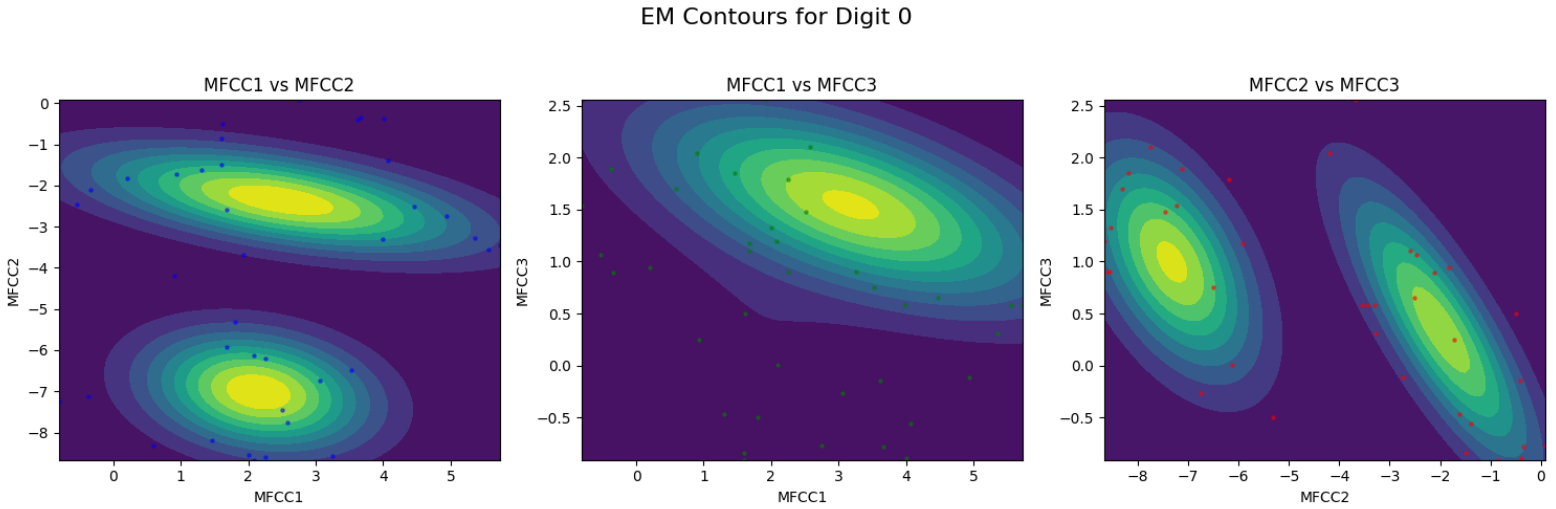
EM Analysis

Here are the advantages and disadvantages of parameter estimation via EM:

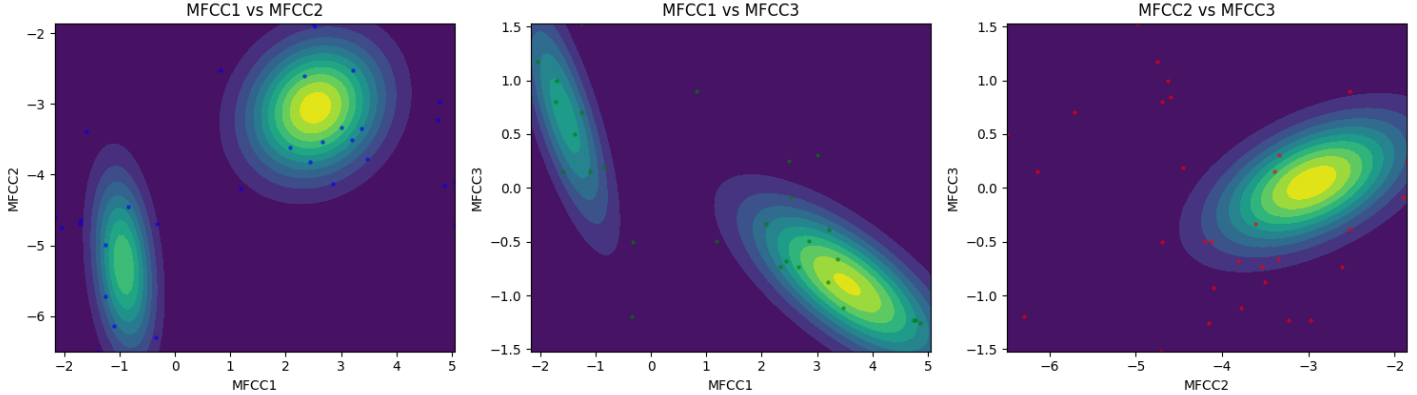
Advantages	Disadvantages
EM works well with observations that have soft boundaries or overlap	EM can be quite slow when the data is more spread out
EM is guaranteed to converge even when given random initial parameters	EM may converge to a local maxima because it maximizes the likelihood function iteratively

Visualization: Digit GMMs via EM (0-2)

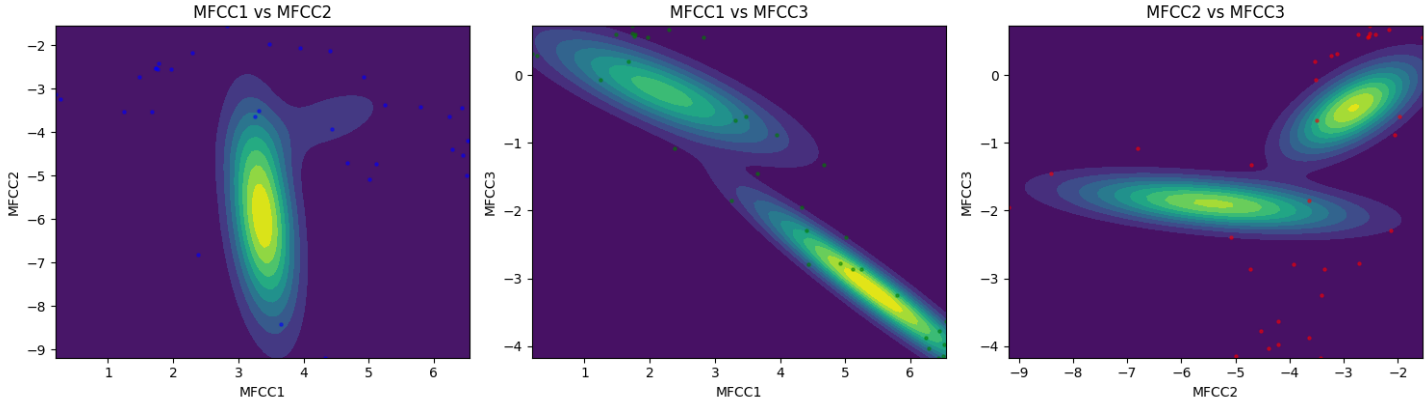
Below are clusters created via Expectation-Maximization The clusters show the overlap between observations. This is on display on the next slide.



EM Contours for Digit 1



EM Contours for Digit 2



K-Means vs. EM

When considering which approach to apply for classification, I've realized that it's very domain-specific. With this project's exploration of digit classification, the relationships between MFCCs plays a big part in the effectiveness of either approach. As highlighted in the MFCC Comparisons (Slide 10), the degree of separation indicated by the relationships between the MFCCs is the most important factor. Due to human aspect of this dataset introducing variability in the pronunciations of digits and thus, overlap in the MFCCs, I would posit that EM is going to produce more accurate results at the cost of increased computation and higher variance. As I work through the project, this theory will be evaluated.

Methodology: Covariance Constraints

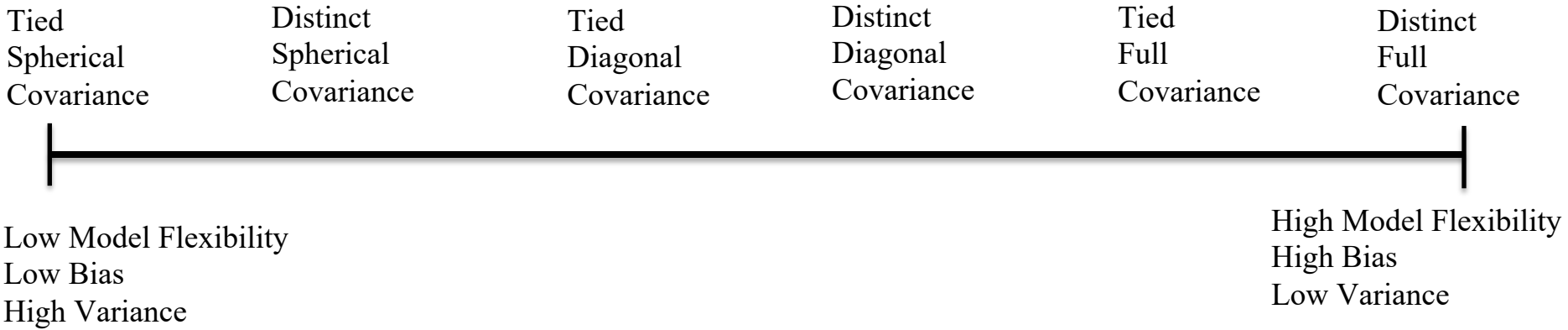
As highlighted in the discussion of K-Means and EM, the covariance matrices of each cluster, Σ_k, are an important variable to be considered in the efficacy of a given GMM. This project will be exploring the impact decisions made about Σ_k have on model functionality.

Limitations can be applied to the covariances and these have different effects on bias, variance and model speed. A GMM's covariances can either be **Tied** or **Distinct** and in either category, the covariances could also be **Full**, **Diagonal** or **Spherical**.

Tied covariances imply that all mixture components in the GMM share a covariance matrix while the distinct case has each mixture component having its own covariance matrix. Consequently, the tied framework makes the model less flexible and introduces higher bias but lower variance. The distinct covariances make the model more flexible but introduce higher variance.

Covariance Constraints Cont.

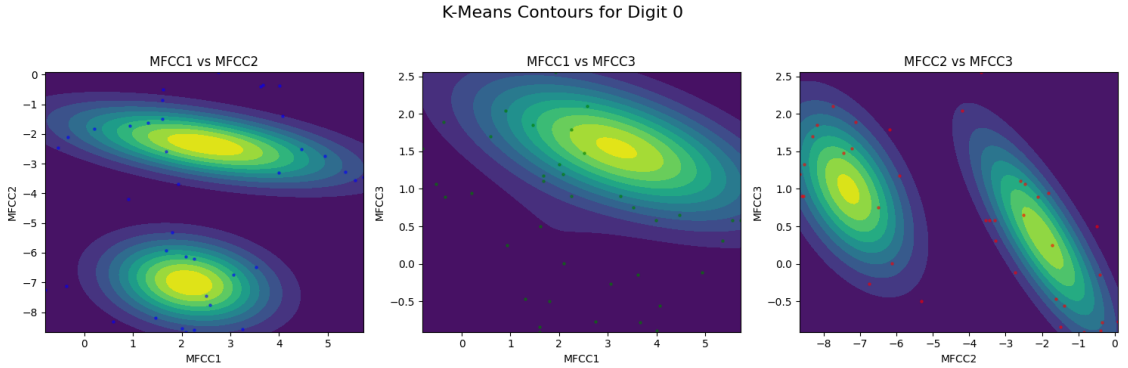
Full Covariances are fully populated. Diagonal Covariances only have their diagonal elements, but they are potentially distinct. Spherical Covariances also only have their diagonal elements, but they are all the same. The influence of these constraints on variance and bias are summarized below:



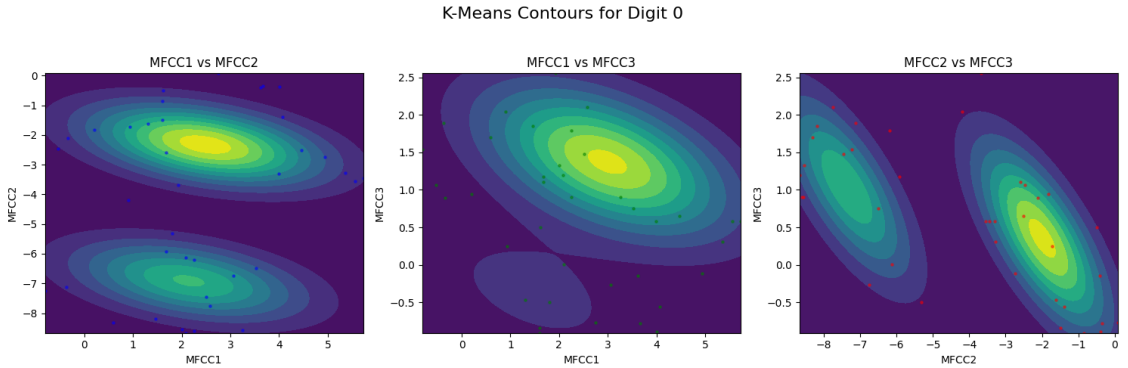
Visualization: Tied vs Distinct Full Covariance (K-Means)

These plots show that full covariances capture the distribution of the data well.

This can be seen clearly in the MFCC1 vs MFCC2 plot.



Distinct

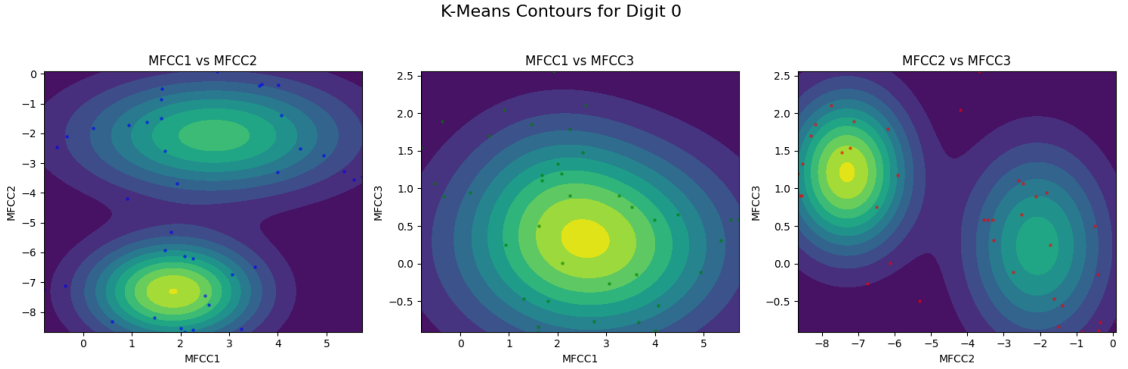


Tied

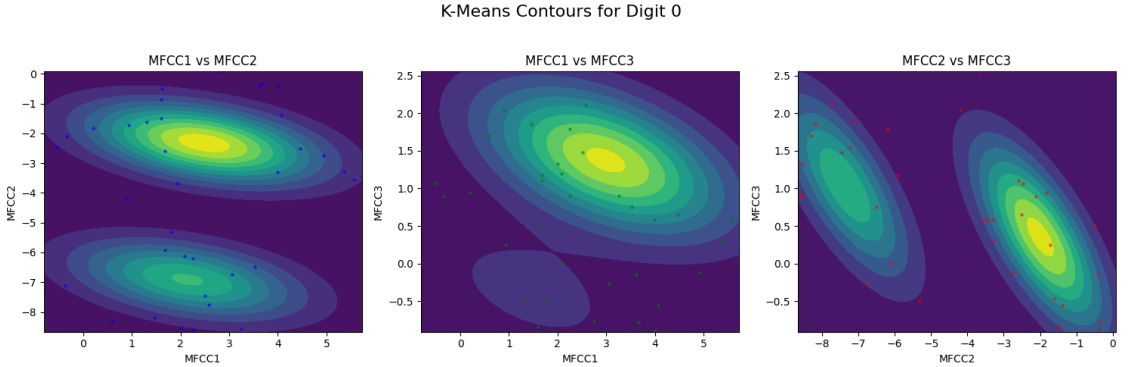
Visualization: Tied vs Distinct Diagonal Covariance (K-Means)

These plots show that diagonal covariances do not capture the distribution of the data particularly well.

This can be seen clearly in the distinct MFCC1 vs MFCC3 plot.



Distinct

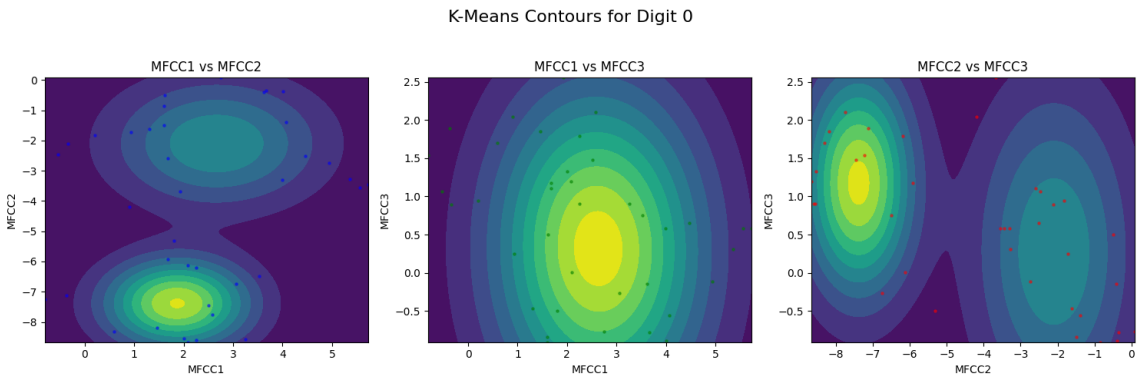


Tied

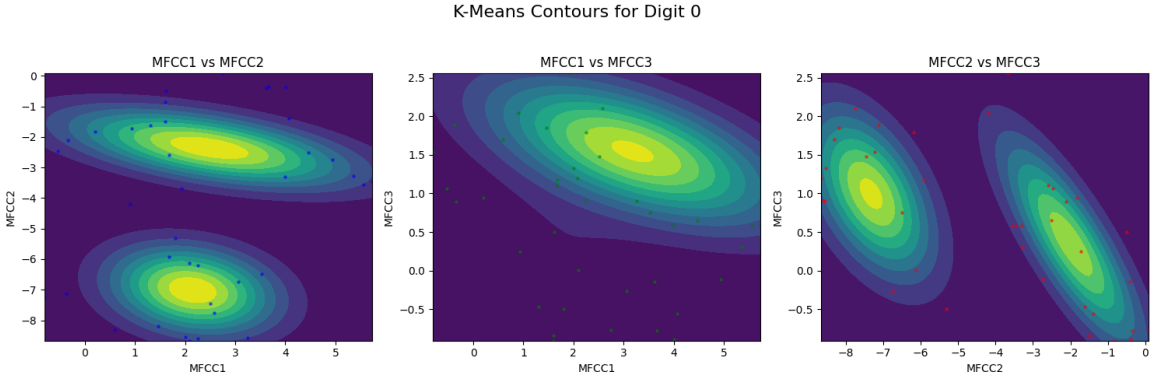
Visualization: Tied vs Distinct Spherical Covariance (K-Means)

These plots show that spherical covariances do not capture the distribution of the data particularly well.

This can be seen clearly in the distinct MFCC1 vs MFCC3 plot.



Distinct



Tied

Maximum Likelihood Classification

With the framework of the models now established, a vital aspect of this project is evaluating its ability to correctly identify the spoken digits. To do that, I've applied maximum likelihood classification to populate a confusion matrix. Maximum Likelihood Classification (MLC) is used to place data points into categories based on the likelihood of the data belonging to each class.

That likelihood is derived from Maximum Likelihood Estimations (MLE). MLE are estimates of the parameters of a probability distribution that maximize the likelihood function, which measures how well the model explains the observed data. Put succinctly, MLE identifies the parameter values that make the observed data most probable under the assumed statistical model.

For $\mathrm{X} = \{x_1, x_2 \ldots x_n\}$ where X represents the observations (data) and θ represents the parameters for the model, the likelihood function is given as:

$$\mathcal{L}(\theta, \mathrm{X}) = \mathrm{P}(\mathrm{X}|\theta)$$

where $\mathrm{P}(\mathrm{X}|\theta)$ is the function that represents the model. For independent and identically distributed data,

$$\mathcal{L}(\theta, \mathrm{X}) = \mathrm{P}(\mathrm{X}|\theta) = \prod_{i=0}^{n} \mathrm{P}(X_i|\theta)$$

$\mathcal{L}(\theta, \mathrm{X})$ is generally maximized to find MLE

MLE – Log-Likelihood Maximization

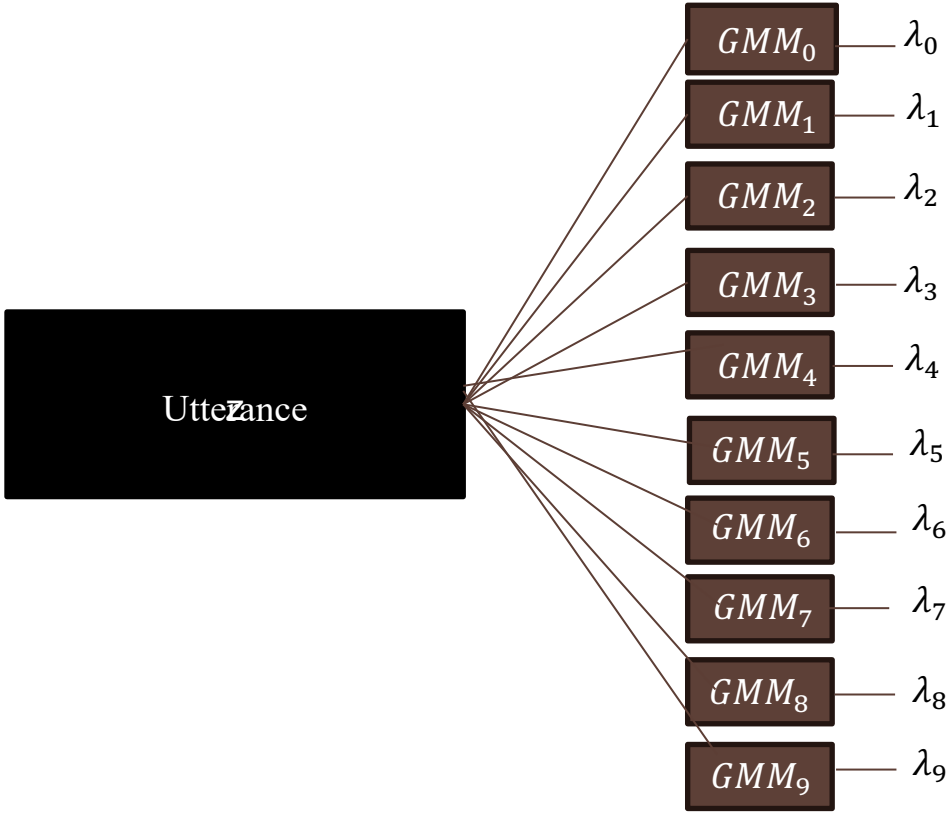
The dataset for this problem is particularly large. As the likelihood function is often a product of probabilities, bound between 0 and 1, I wanted to avoid values tending to zero, an error known as numerical underflow. Most software tends to round extremely small values down to zero, and this makes it difficult to make any useful conclusions. Thus, I will be taking the logarithm of the likelihood function and maximizing that to find the MLE.

$$\theta_{MLE} = \max\left[\log\left(\mathcal{L}(\theta, \mathrm{X})\right)\right] = \max\left[\sum_{i=1}^{n} \log\left(P(X_i|\theta)\right)\right]$$

For the GMMs used in this project, the θ_{MLE} is given as follows:

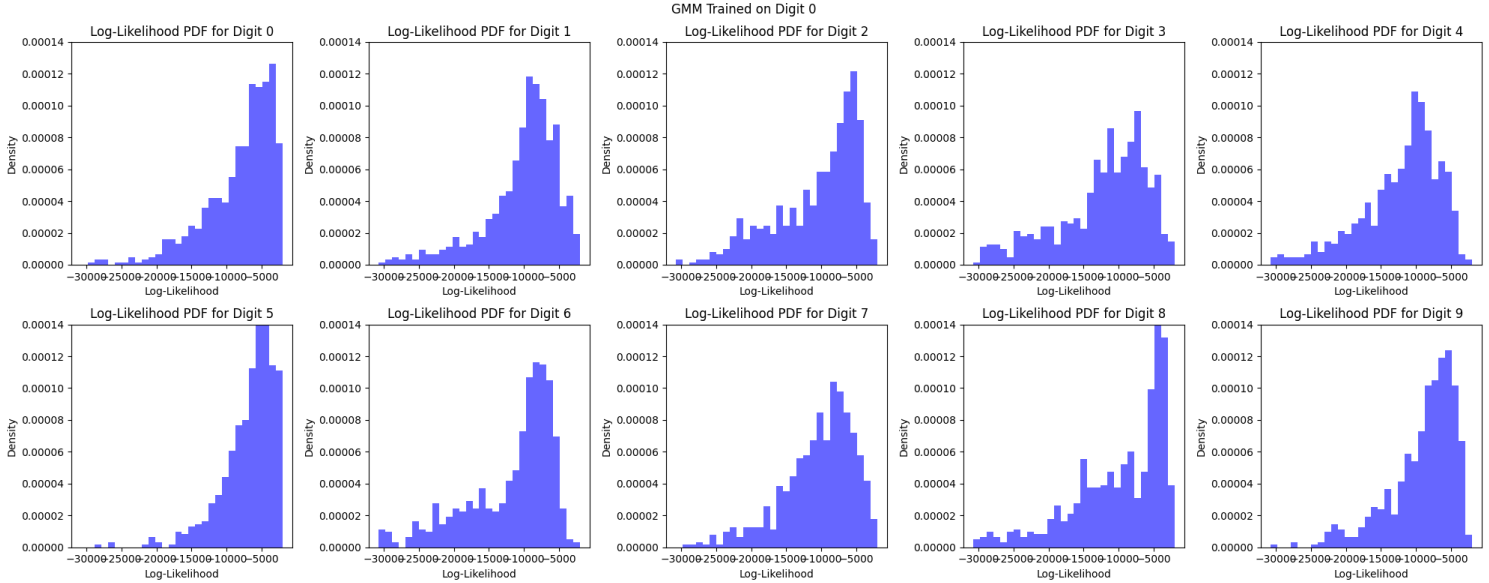
$$\theta_{MLE} = \max\left[\sum_{i=1}^{n} \log\left(\sum_{k=1}^{K} \pi_k \, N(x_i|\mu_k, \Sigma_k)\right)\right]$$

Consequently, I will be creating a GMM based on each digit. Maximum Likelihood Classification is useful here because given a set of data (a single utterance), the highest MLE should be produced by the GMM generated using the actual digit encoded by the MFFCs and the model will return the digit it believes the utterance represents.



Max($\lambda_0 \ldots \lambda_9$) informs us that Utterance = i

Visualization: MLC



We can see that the the log-likelihood is somewhat tightest and rightmost for the plot of “0”. This is reasonable because that was what the GMM was trained on. However, there are some similarities to the digit 5 and 9 plots and this reflects that the model is not 100% accurate

Confusion Matrices

Confusion matrices allow us to summarize the findings from maximum likelihood classification and visually analyze model performance. The terms used are defined below

- **True Positives (TP)** are correct classifications for a specific class.
- **False Positives (FP)** are incorrectly classified as a specific class.
- **True Negatives (TN)** are correctly not classified as a specific class.
- **False Negatives (FN)** are incorrectly not classified as a specific class.

		Actual Class	
		Positive (**P**)	Negative (**N**)
Predicted Class	Positive (**P**)	True Positive (**TP**)	False Positive (**FP**)
	Negative (**N**)	False Negative (**FN**)	True Negative (**TN**)

Confusion Matrices Cont.

As hinted at previously, confusion matrices reflects how well MLC aligns with the true values of the data. Summarily, high TP rates indicate that MLC is accurate and high FP rates could highlight assumptions the model has made that do not reflect the data correctly.

For ease of interpretation, for a given row i and column j, the value at (i, j = i) is the number of times the model correctly identified the digit. All other column entries are the number of times the model identified the digit i as the digit j.

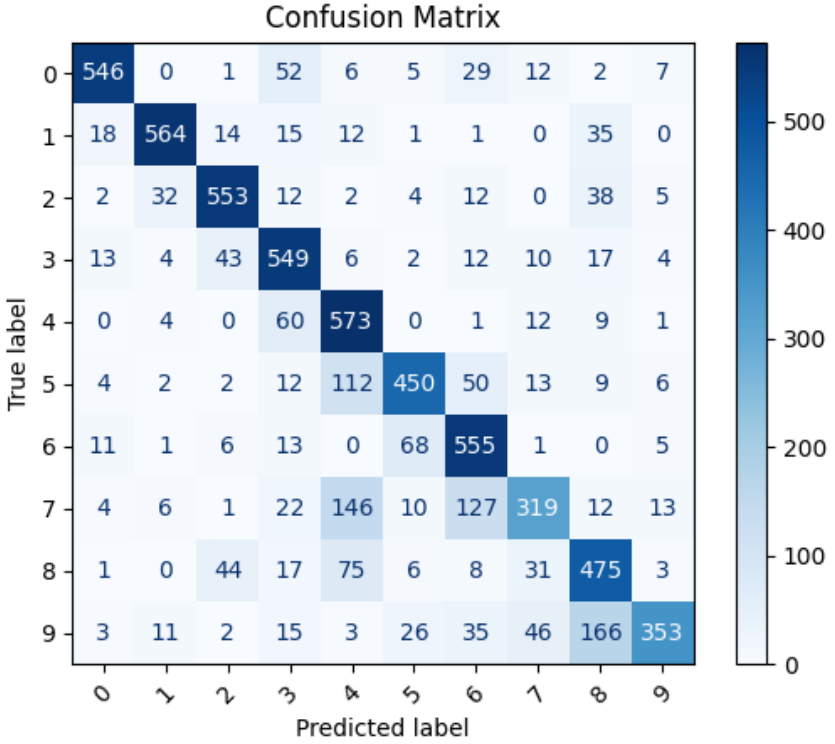
Visualization: Confusion Matrix

The accuracy of the model is calculated using this confusion matrix as follows:

Accuracy:

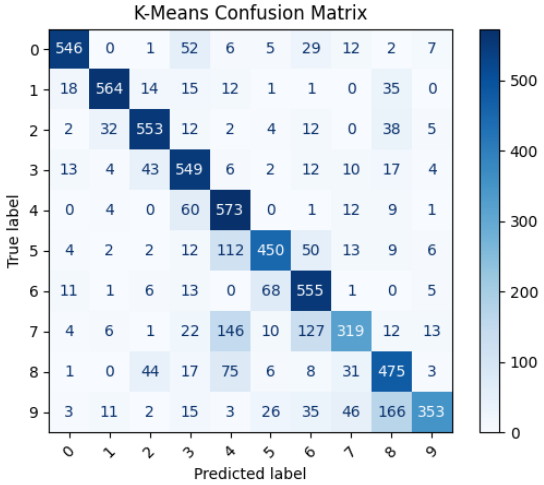
$$\sum_{i=0}^{9} p(predict\ digit\ i\ spoken\ |digit\ i\ spoken) * p(digit\ i\ spoken)$$

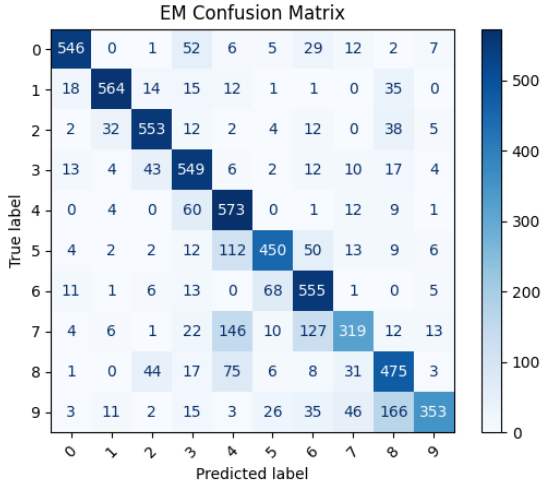
The confusion matrix here was generated with a covariance type of Distinct Full with GMM parameters generated via K-Means and the accuracy was found to be **74.80%**. This makes sense because a distinct full covariance is providing as much flexibility to the model as possible. This segues nicely into exploring how different modeling choices affect model accuracy.



K-Means vs. EM and Model Accuracy

Below are the confusion matrices produced by estimating GMM parameters via K-Means and Expectation Maximization. Both were generated using distinct full covariances.



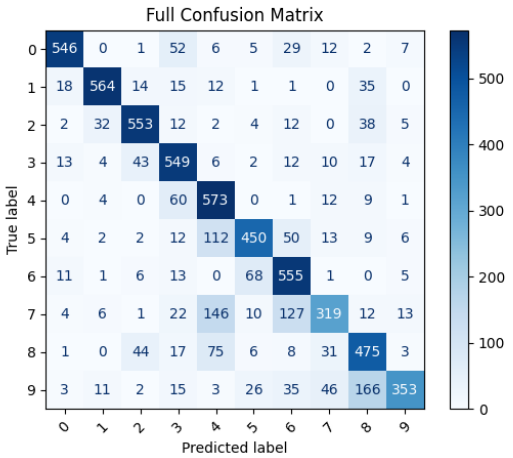


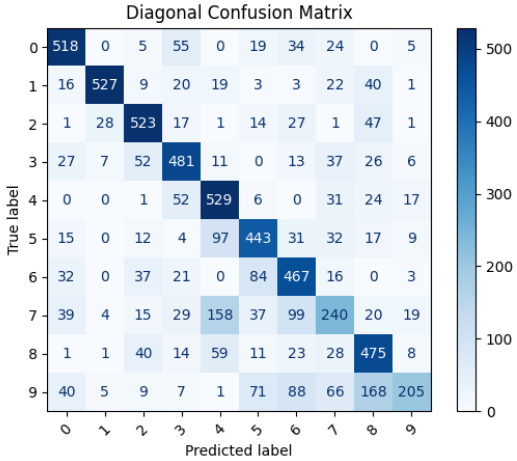
There appears to be no difference in the model's predictive abilities based on what was used to generate the parameters as the confusion matrices are the same. This is because both K-Means and EM are used as tools to initialize GMMs, but they do not alter the fundamental structure of the GMM itself. K-Means is often used to provide initial cluster assignments for EM to refine, but the resulting GMM (after convergence) represents the same clusters regardless of the initialization method. Furthermore, the confusion matrix is calculated based on the final clustering assignments which in most cases should originate due to convergence reached by initialization by K-Means and initialization by EM

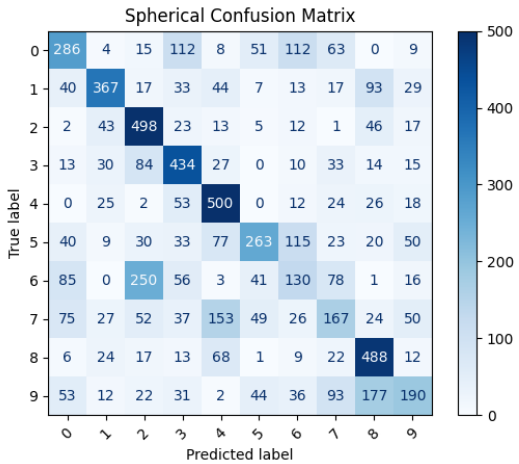
However, if either approach leads to incorrect convergence, possibly getting stuck in a local maxima, the confusion matrices would differ significantly.

Covariance Choices and Model Accuracy

Below are the confusion matrices generated with different covariance constraints. The GMM parameters were all estimated via K-Means and the covariances for each component are distinct.







The confusion matrices show that full covariances provide the highest accuracy (74.80%) for the model. Diagonal covariances are less accurate (66.79%) but appear to be considerably more accurate than spherical covariances (50.35%).

As discussed earlier, this is because full covariances capture more of the data's actual shape. It places no assumptions on the geometry of the clusters. Diagonal covariances assume that the clusters are aligned with a horizontal axis and elliptical in shape which might lead to poorly capturing relationships between observations. Spherical covariances assume that clusters are perfectly circular or spherical and this is particularly prone to incorrectly capturing relationships.

I concluded that exploration with distinct covariances was sufficient because the use of tied covariances would likely reveal the same information discussed above with a considerable drop in accuracy as a result of the added restriction.

Variation in Digit Prediction

The confusion matrices allow me to infer the model's general ability to predict that specific digits. From the matrices, for the digits 7 and 9, the number of correct inferences is fairly lower than the other digits. The digit 7, for example, is often misclassified as the digit 4. The low TP rate is captured in the lighter shade of blue used for the squares in the matrix. There are several reasons this is the case chief of which I believe is the presence of acoustic similarity (MFCC overlap between digits).

The case for acoustic similarities between 7 and 4 is particularly strong because there is only one different phoneme between the two with 7 being *s-a-b-ah* and 8 being *a-r-b-ah*. 9 falls into this category as well with its phonemes being is t-i-s-ah.

The model is better at predicting the other digits likely due to lower MFCC overlap.

Challenges in MLC Application

When applying MLC to draw conclusions, I initially struggled with visually interpreting the density and compactness of the log-likelihood graphs to infer what digit the current GMM was based on. For a given GMM, I imagined when the log-likelihood of the correct digit was plotted, there would be a stark contrast between its magnitude and the others. However, this was not the case, and I had to pay more attention to the spread of the graph as well as its height, with incorrect digits generally producing a log-likelihood graph with a wider spread. In the model, this issue was simply abstracted to a maximum function to find the maximum amongst log-likelihoods.

Additional Exploration: Speaker Specific Models

Currently, a single model is broadly used for classification for all speakers. There are certain advantages that the “one-size-fits-all” approach provides such as improved computational efficiency and a reduced tendency for overfitting. However, it may struggle in cases where speaker pronunciations or accents are inconsistent. Thus, this brings into question of the possibility of speaker specific models. These speaker specific models would be more attuned to speaker nuances.

However, the usefulness of this narrowing of scope is highly context-specific and given this problem, I doubt this would be particularly useful. The dataset we are working with is set up so that the speakers from the training and testing data are different. Hence, any model trained on specific speakers from the training dataset would likely not have have improved accuracy over the more general case and would not be worth the increased computational effort. Nevertheless, it was beneficial to consider the tradeoffs that a slightly different approach would have introduced.

Conclusions

From this project, I've concluded that the type of covariance is the most particularly important modeling choice for this task. As stated previously, this choice strongly influences how much the model truly captures the distribution of the data and thus, the model's accuracy is very strongly correlated to any covariance constraints applied. The method of initialization (K-Means vs. EM) is less relevant because the model reaches a global maxima in both cases.

In the event I had to make a model recommendation, I would propose GMMs initialized via K-Means with distinct full covariances for each component. A Maximum Likelihood Classifier based on maximum log-likelihoods should be used to make digit predictions. This system balances relatively low computational effort while providing decent accuracy. However, there is much room for improvement. Specifically, experimentation with the number of components and the number of MFCCs used for classification could be performed to further refine the model.

Currently, this model has a constant number of components of 2. It may be useful to increase the number of components to properly represent the number of phonemes present. For broad classification purposes, I imagine this new number could be the average number of phonemes in all 10 digits. Furthermore, the model uses all 13 MFCCs. It's possible that some of them are superfluous or introducing unnecessary noise. With more time and resources, I would explore the optimum number of MFCCs to use.

If I were to redo this project, I would start of by exploring different modeling choices particularly the improvements highlighted above. I would also engage more with other students to gain different perspectives on tackling this task. However, I would retain my decision not to make models for each individual speaker because as discussed earlier, the benefits are negligible.

Overall, this project was very enlightening, and I've gained a deeper understanding of the numerous decisions that go into making a successful classification model and an appreciation for the balance of aiming for the highest level of accuracy while keeping contextual relevance in mind.

Collaborations

I mainly worked with the ECE 480 TAs. The teaching assistants were an amazing resource I used to ensure I had a proper understanding of the task at hand and to sanity-check the results my model was producing.

References

- **Phoneme Definition:** Mousumi Malakar, Ravindra B. Keskar, Progress of machine learning based automatic phoneme recognition and its prospect, Speech Communication, Volume 135, 2021, Pages 37-53, ISSN 0167-6393, https://doi.org/10.1016/j.specom.2021.09.006.
- **Cepstral Coefficient Definition:** Introduction to Speech Processing | Ricardo Gutierrez-Osuna | CSE@TAMU, https://people.engr.tamu.edu/rgutier/lectures/sp/l9.pdf
- **Gaussian Mixture Models**: Gomes, C. G., & Boisvert, J. & Deutsch, C.V. (2022). Gaussian Mixture Models. In J. L. Deutsch (Ed.), *Geostatistics Lessons*. Retrieved from http://www.geostatisticslessons.com/lessons/gmm
- **GMM Image Source**: https://medium.com/@sudhanvaiitr/mixture-of-gaussians-688917016e78
- **Dataset**: Bedda, Mouldi and Nacereddine Hammami. 2008. Spoken Arabic Digit. UCI Machine Learning Repository. https://doi.org/10.24432/C52C9Q.
- **K-Means Algorithm**: Abiodun M. Ikotun, Absalom E. Ezugwu, Laith Abualigah, Belal Abuhaija, Jia Heming,K-means clustering algorithms: A comprehensive review, variants analysis, and advances in the era of big data, Information Sciences, Volume 622, 2023, Pages 178-210, ISSN 0020-0255, https://doi.org/10.1016/j.ins.2022.11.139.

References Cont.

- **EM Algorithm:** Wang, Su. (2017). EM-algorithm and Clustering: a Tutorial.

- **Maximum Likelihood Classification**: Miura, Keiji. (2011). An Introduction to Maximum Likelihood Estimation and Information Geometry. Interdisciplinary Information Sciences (IIS). 17. 10.4036/iis.2011.155.

- **Log-Likelihoods:** Taboga, Marco (2021). "Log-likelihood", Lectures on probability theory and mathematical statistics. Kindle Direct Publishing. Online appendix. https://www.statlect.com/glossary/log-likelihood.

- **Python Packages**

  - **Numpy** - Harris, Charles R., K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, ... and Travis E. Oliphant. "Array programming with NumPy." *Nature* 585, no. 7825 (2020): 357-362. https://doi.org/10.1038/s41586-020-2649-2.

  - **Matplotlib** - Hunter, John D. "Matplotlib: A 2D Graphics Environment." *Computing in Science & Engineering* 9, no. 3 (2007): 90-95. https://doi.org/10.1109/MCSE.2007.55.

  - **Sklearn** - Pedregosa, Fabian, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, et al. "Scikit-learn: Machine Learning in Python." *Journal of Machine Learning Research* 12 (2011): 2825-2830.