Recognizing Spoken Digits

Luis Pereda Amaya

ECE 480: Applied Probability for Statistical Learning

Fall 2021

[Link to Jupyter Notebook]

Table of Contents

01
Background

Overview

The focus of this project is to use introductory statistical learning models, the mathematics of which we have studied in this course, to identify which of the Arabic digits 0 to 9 was spoken. Since our goal is to improve our classifiers based on modeling choices, the signal processing is done for us and we are given our data in the form of cepstral coefficients.[1]

Background on Cepstral Coefficients

The term cepstral is related to a new “spectral” representation domain that is neither the frequency nor the time domain, referred to instead by Bogert, Healy, and Tukey as the *quefrequency* domain, where the spectrum of the log of the spectrum of a time waveform is referred to as the cepstrum. Cepstrum coefficients were later found to be particularly useful in the area of speech or speaker recognition. [2]

Why this problem?

Speech identification using statistical learning approaches is an interesting problem for us to tackle because:

- This is a highly applicable problem, being solved by many popular technologies such as Siri or Alexa.
- Using models studied in the course we can generate accurate predictions, validating the usefulness of even relatively simple approaches.
- This problem is open to many modeling choices, allowing us to analyze how different features of a model can affect its predictiveness

Contextual Importance

As mentioned, this problem is interesting even outside of the academic/research realm. With many commercial applications of speech recognition, we can see the wide interest of such a problem even from our dataset. Finding that datasets of English spoken digits were too expensive, we relied on one of Arabic spoken digits; from this alone we can see how many care about such problems in speech recognition.

Additional Applications

Our modeling framework approach one effective at pattern recognition, so there are many other applications outside of speech recognition where this could be used. One potential example could be image classification, where animals photographed in a rainforest could be identified through a similar model to assist in cataloguing species in the rainforest. Rather than cepstral coefficients, the data used would be color values from pixels, and there would likely arise similar patterns for animals of the same species that could be detected.

Approach

The creation of the classifiers was made by modeling the distribution of cepstral coefficients with Gaussian Mixture Models (GMM), of which the parameters for the GMM we calculated via K-means clustering and the Expectation-Maximization algorithm (EM),

Gaussian Mixture Model

A GMM models a distribution of data as coming from a finite number of Gaussian distributions as shown below, where $0 < \boldsymbol{\omega} < 1$ represents the weight of each Gaussian [3]

$$p(x|\lambda) = \sum_{i=1}^{M} \omega_i \, g(x|\mu i, \Sigma i),$$

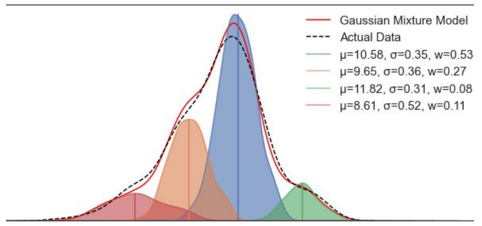
Where each Gaussian is represented by:

$$g(x|\mu i, \Sigma i) = \frac{1}{(2\pi)^{D/2}|\Sigma_i|^{1/2}} \exp\left(-\frac{1}{2}(x-\mu_i)^T \Sigma_i^{-1}(x-\mu_i)\right)$$

Gaussian Mixture Models continued

In order to model our data as a mixture of Gaussians, there are a number of modeling parameters to be considered. Among these are the number of Gaussians that will go into the mixture, the means, covariances, and weights of each mixture, as well as some choices made on how to use the data. To find the parameters of the Gaussians in this project, we use K-means clustering and the Expectation-Maximization algorithm.

Image taken from towardsdatascience.com



K-means Clustering

K-means is a clustering approach that defines a cluster as a representative point, the mean of all the points in that cluster. These clusters are created with an iterative process known as Lloyd’s algorithm. From Lloyd’s algorithm, K-means clustering is guaranteed to converge on at least a local minimum, providing data clustered from its unique patterns. [4]

K-means Clustering - Lloyd’s Algorithm

The clustering performed by K-means is done in the following steps:

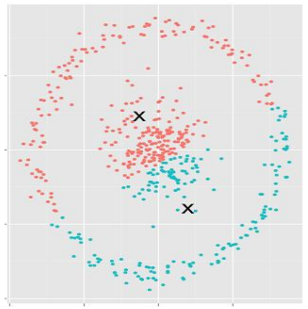
1. Data points are assigned to the closest cluster center
2. The cluster center is changed to the mean of the assigned points

This process is repeated until there is a convergence at a local minimum.[5]

K-means Clustering - Pitfalls

K-means is both an easy and effective way to cluster data, but it is not without its pitfalls. It is heavily dependent the initialization of its cluster centers, which defines whether the algorithm will converge on a global or local maximum. Additionally, K-means works well for data that is well -separated and spherical/hyper-spherical in nature, but is not excellent outside of this.

Image taken from Stats Stack Exchange



K-means Clustering: Uses

From our K-means model we can find useful data for this project:

- The cluster centers can be used as means for the GMM
- The number of clusters corresponds to the number of Gaussian mixture components
- The weight of each component is proportional to the data points in the cluster over the total number of data points
- Covariances can be calculated from the data in each cluster

K-means: Implementation

The Scikit-learn KMeans package was used to create the clusters in this lab.[6] From the computed clusters, we could set the parameters of the components of the Scikit-learn Gaussian mixture (means, covariances, weights, precisions, precisions cholesky).

Expectation-Maximization Algorithm

The aim of the EM algorithm is to find a model such that the likelihood of getting our data from that said model is maximized. The approach is an iterative search that optimizes the lower bound of the log likelihood on the data until the maximum for the lower bound is the maximum for the log likelihood overall [2].

Expectation Maximization: Jensen's Inequality

This is possible because of Jensen's inequality, where the log of the expected value of f(z) is always greater than the expected value of log f(z)[2].

$$log E_{p(z)}\{f(z)\} \geq E_{p(z)}\{log f(z)\}$$

Expectation Maximization: Pitfalls

As in most numerical approaches to these problems, there is the danger of converging to a local maximum rather than a global maximum, and this can be in large part fixed by sampling a variety of initializations.

Expectation Maximization: Uses and Implementation

Using the Scikit-learn [6] GaussianMixture package which leverages EM under the hood to compute the values for each of the components of the mixture, this API allowed for an efficient way to calculate Gaussian Mixture Model using Expectation Maximization

Maximum Likelihood Overview

With maximum likelihood, we are looking to find a model such that the likelihood of the data set coming from this model is maximized, where likelihood is defined by: [4]

$$L = p\left(\vec{t} \,\middle|\, \vec{X}, \vec{w}, \sigma^2\right) = \prod_{n=1}^{N} p(t_n \,|\, \vec{x_n} \,, \vec{w}, \sigma^2)$$

Maximum Likelihood Classifier

The goal of our classifier is to find a model based on the principle of maximum likelihood in a generative modeling sense to find the model for which the data set is most likely to have come from. This process is accomplished by differentiating the log likelihood and setting this value to 0, to analytically solve for **w** and $\boldsymbol{\sigma}$. In the program, this process was done numerically through the score() method of the sklearn GaussianMixture, which returns a value representing the likelihood of the dataset to be from that model. By finding the maximum score, we could classify the spoken digit by seeing what model the digit's data was most likely to come from.

More on Maximum Likelihood Classifiers

Classifying the spoken digits via this process was a favorable solution to our problem due to the nature of us having a model prepared for each digit, and the likelihood of the spoken digit coming from each model worked as an effective metric allowing us to find accurate predictions. Finding the score() method of the GaussianMixture package proved an efficient means of implementing this maximum likelihood classification, and allowed us to focus on concerns surrounding modeling choices rather than the implementation of this classifier.

02
Modeling

Modeling Choices

Number of clusters/components per digit

Gendered vs Non-Gendered Classifiers

Use of Spherical, Diagonal, Tied, and Full Covariances

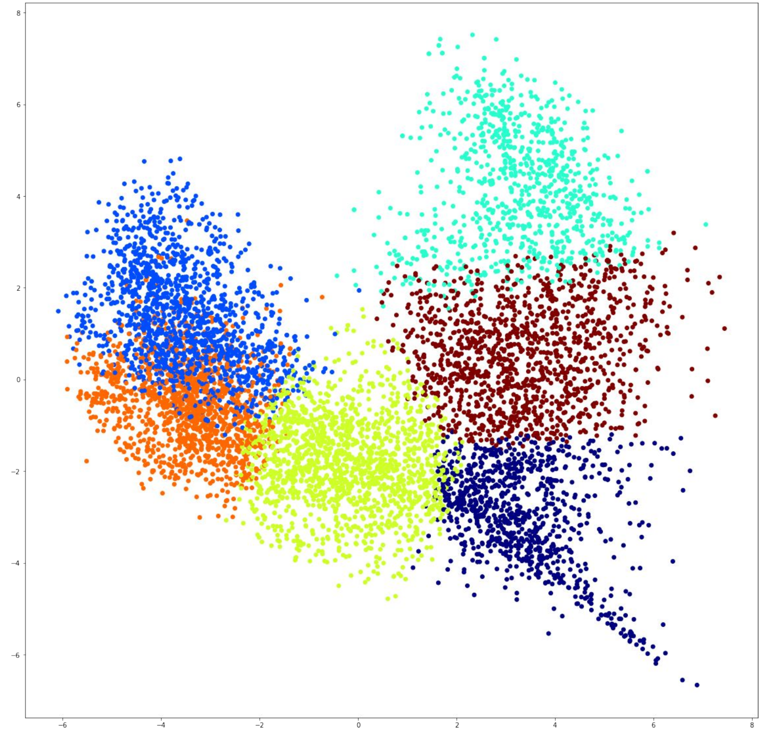
Subsets of Mel-Frequency Cepstral Coefficients

Number of clusters/components

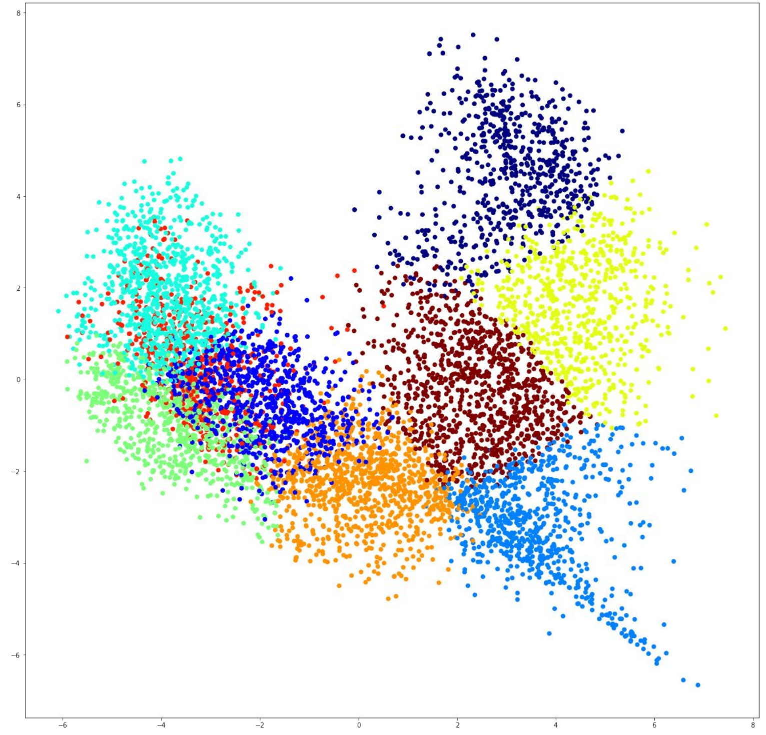
General approach and reasoning

As advised by Dr. Tantum, I initially set my number of clusters per digit to 2n - 1, where n is the number of phonemes. This in theory would account for each phoneme and each transition between phonemes to be accounted for as clusters. Due to the degree of uncertainty of the true number of phonemes in each word, I chose to use 2n - 1 as a starting point, then tampered with the cluster numbers until I observed the highest number of clusters plotted that were still clearly distinguishable. This approach also happened to maximize many of the accuracies of the models.

K-means: 6 vs 9 clusters of digit 0

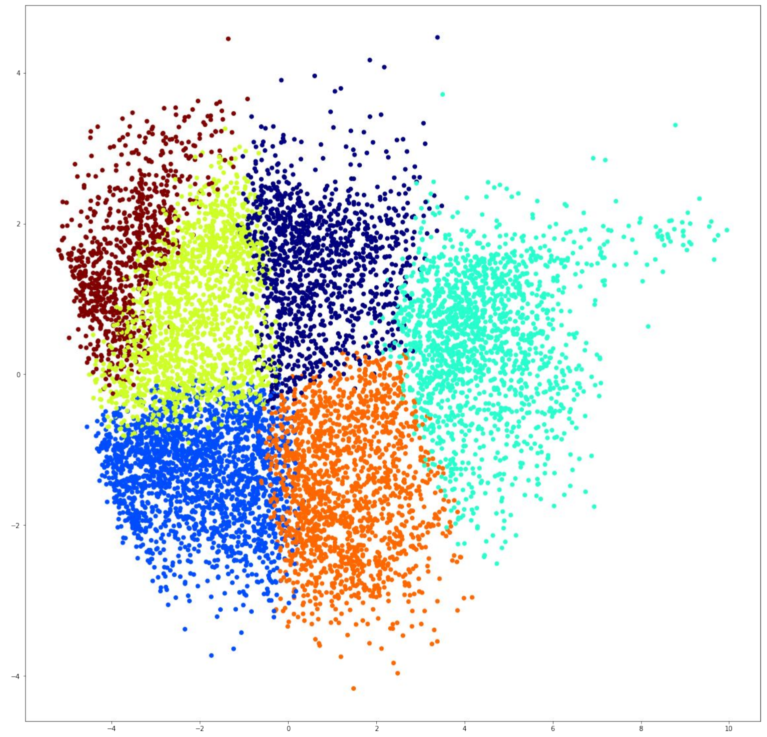


6 clusters

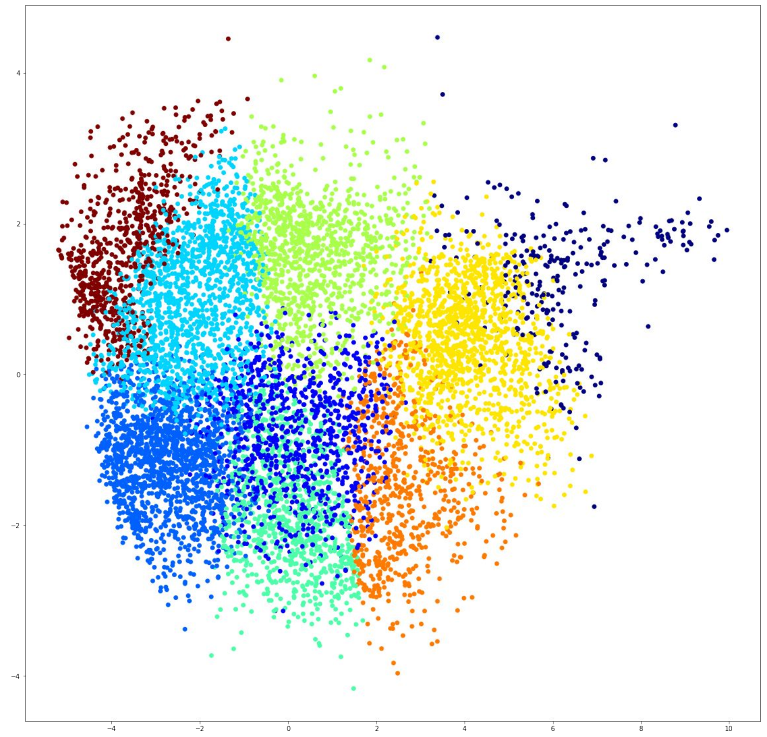


9 clusters

K-means: 7 vs 10 clusters of digit 4

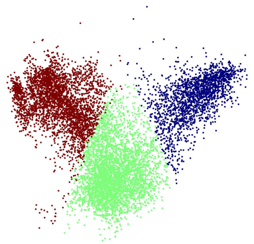


7 clusters



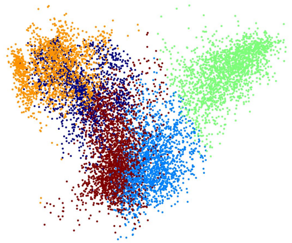
10 clusters

Accuracies for digit 2
(EM model, Diagonal covariance, nongendered)



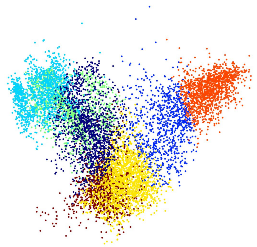
83%

3 clusters



90%

5 clusters



85%

7 clusters

For cases such as the digit 2, **ithnayn,** where clustering was unclear, I attempted to empirically maximize the accuracies

Clusters per Digit

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

Digit	# of Clusters
5	6
6	5
7	5
8	9
9	4

Gendered vs Non-gendered

Reasoning

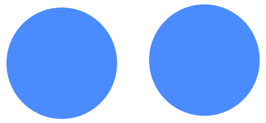
From our data set we had the knowledge of which speakers were male and which were female, so I figured to investigate this added degree of supervision. I created models that were trained on all of the speakers without regard to gender as well as models trained on men and models trained on women. I tested the gendered models against their respective gender to compare their performance against the model trained irrespective of gender

Type of covariance

Reasoning

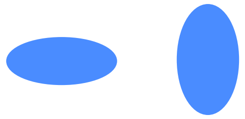
Using different types of covariances allowed me to try out models on different points of the bias-variance curve. This choice in the flexibility of the models allowed for fundamentally different kinds of models that showed strengths for different combinations of modeling choices.

Types of covariances



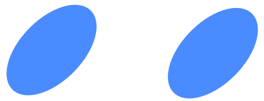
Spherical

All variances same for all clusters along covariance diagonal



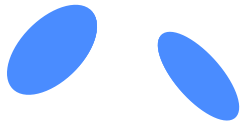
Diagonal

Unique diagonal covariance matrix for each cluster



Tied

Same covariance matrix across all clusters

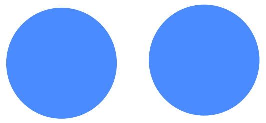


Full

Unique full covariance matrix for each cluster

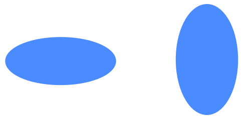
Methodology

Only found myself for K-means, for EM the sklearn package provides the covariances



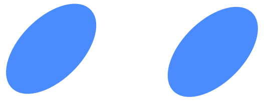
Spherical

Demeaned data of all clusters then found variance of demeaned data



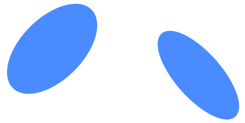
Diagonal

Found covariance matrix for each cluster then took diagonal matrix



Tied

Demeaned data of all clusters then found covariance of demeaned data

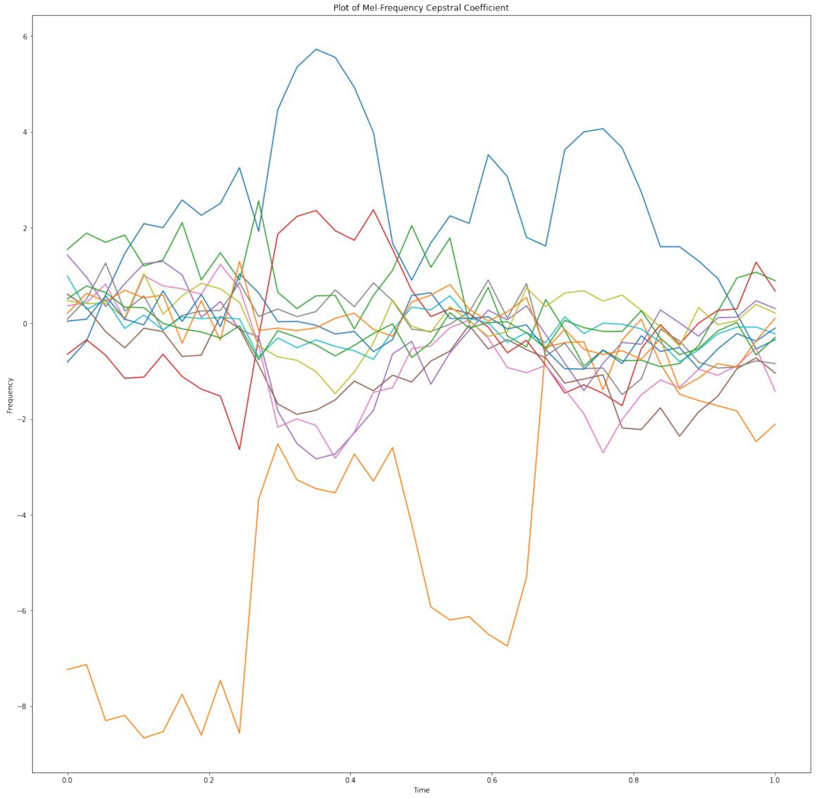


Full

Found covariance matrix for each cluster

Subsets of MFCC’s

MFCC’s for first speaker in training set



Reasoning

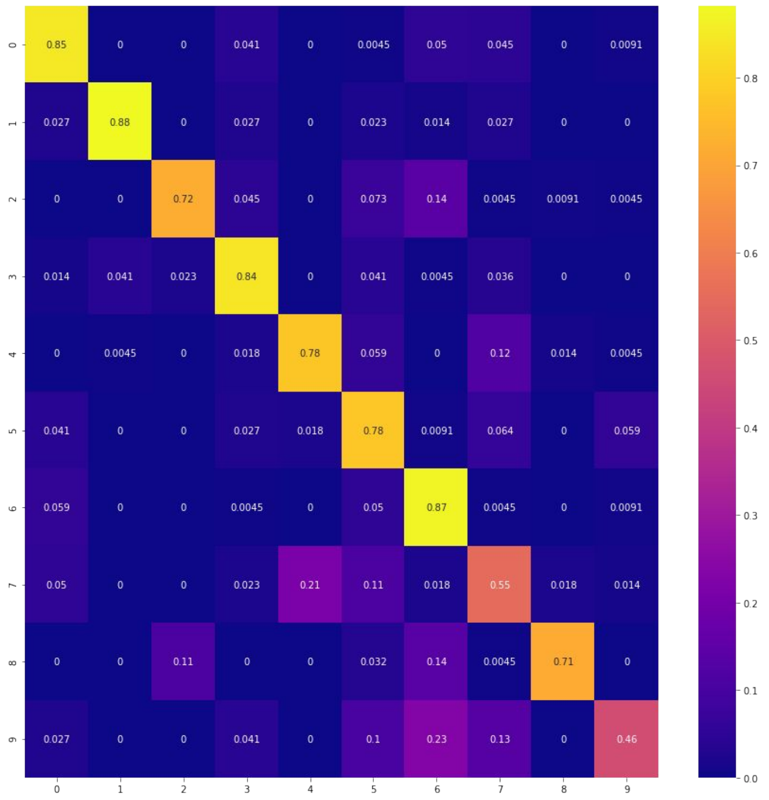
I felt compelled to explore taking subsets of the MFCC's rather than all of them because from the plot on the slide prior, it appears that some of the coefficients vary in amplitude far less than others, so I wondered if removing some from the data would perhaps help the models in recognizing more obvious patterns.

Results and Interpretations

K-means model, non-gendered, spherical

Accuracy:

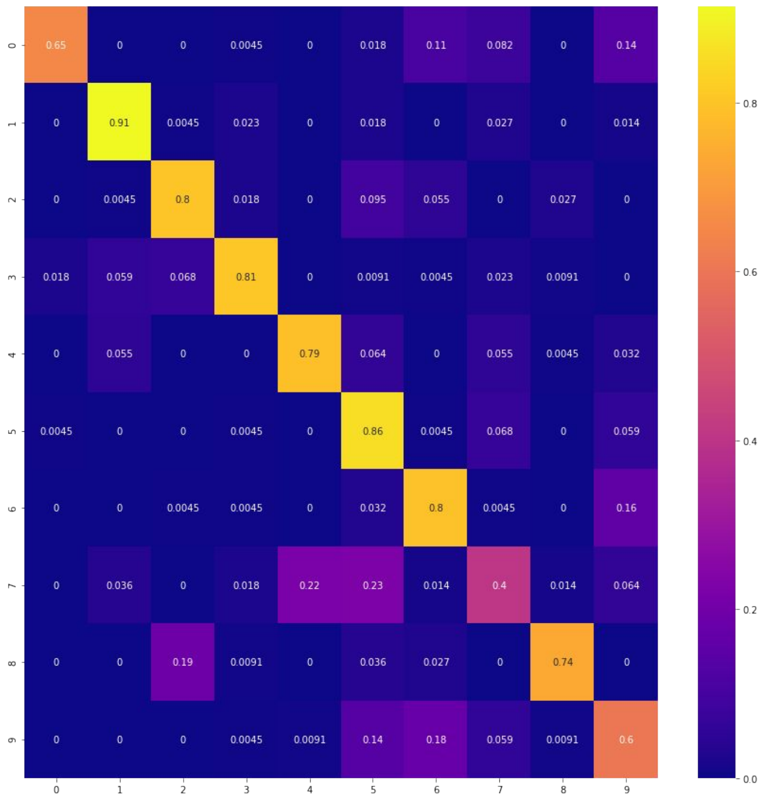
74.5%



	0	1	2	3	4	5	6	7	8	9
0	0.85	0	0	0.041	0	0.0045	0.05	0.045	0	0.0091
1	0.027	0.88	0	0.027	0	0.023	0.014	0.027	0	0
2	0	0	0.72	0.045	0	0.073	0.14	0.0045	0.0091	0.0045
3	0.014	0.041	0.023	0.84	0	0.041	0.0045	0.036	0	0
4	0	0.0045	0	0.018	0.78	0.059	0	0.12	0.014	0.0045
5	0.041	0	0	0.027	0.018	0.78	0.0091	0.064	0	0.059
6	0.059	0	0	0.0045	0	0.05	0.87	0.0045	0	0.0091
7	0.05	0	0	0.023	0.21	0.11	0.018	0.55	0.018	0.014
8	0	0	0.11	0	0	0.032	0.14	0.0045	0.71	0
9	0.027	0	0	0.041	0	0.1	0.23	0.13	0	0.46

K-means model, non-gendered, diagonal

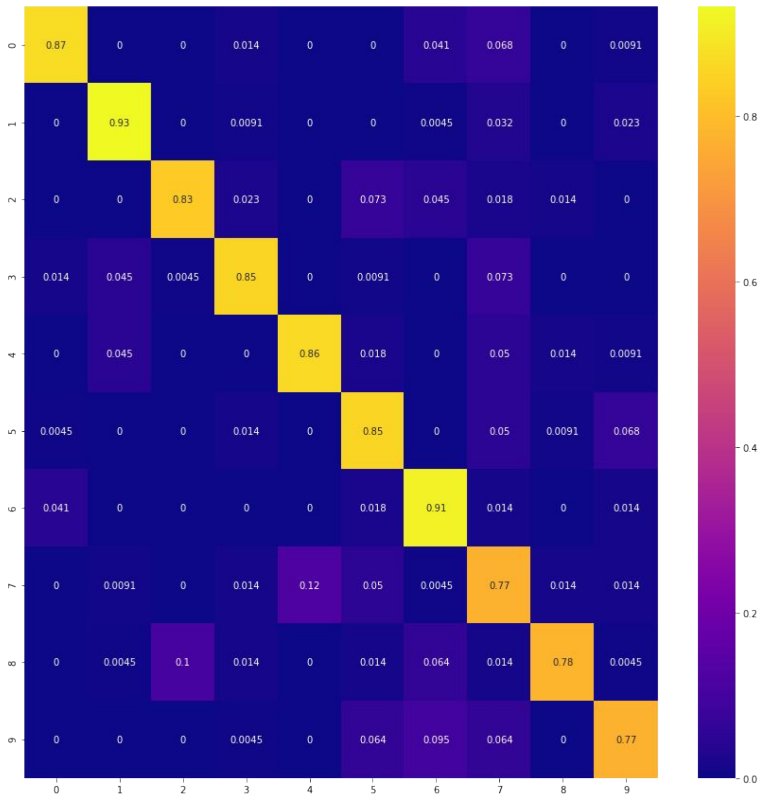
Accuracy: 73.6%



K-means model, non-gendered, tied

Accuracy:

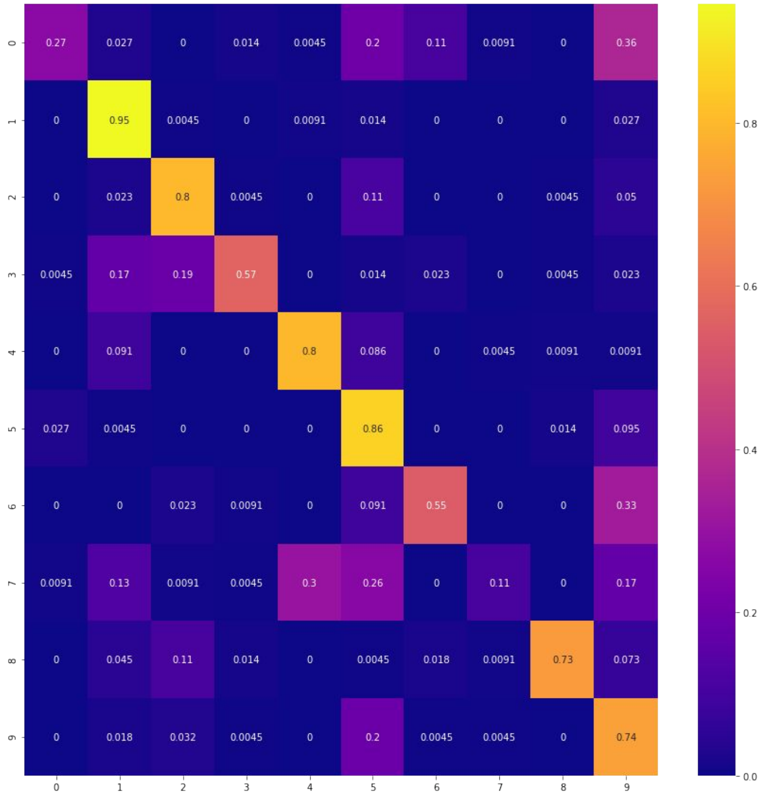
84.4%



	0	1	2	3	4	5	6	7	8	9
0	0.87	0	0	0.014	0	0	0.041	0.068	0	0.0091
1	0	0.93	0	0.0091	0	0	0.0045	0.032	0	0.023
2	0	0	0.83	0.023	0	0.073	0.045	0.018	0.014	0
3	0.014	0.045	0.0045	0.85	0	0.0091	0	0.073	0	0
4	0	0.045	0	0	0.86	0.018	0	0.05	0.014	0.0091
5	0.0045	0	0	0.014	0	0.85	0	0.05	0.0091	0.068
6	0.041	0	0	0	0	0.018	0.91	0.014	0	0.014
7	0	0.0091	0	0.014	0.12	0.05	0.0045	0.77	0.014	0.014
8	0	0.0045	0.1	0.014	0	0.014	0.064	0.014	0.78	0.0045
9	0	0	0	0.0045	0	0.064	0.095	0.064	0	0.77

K-means model, non-gendered, full

Accuracy:
63.7%



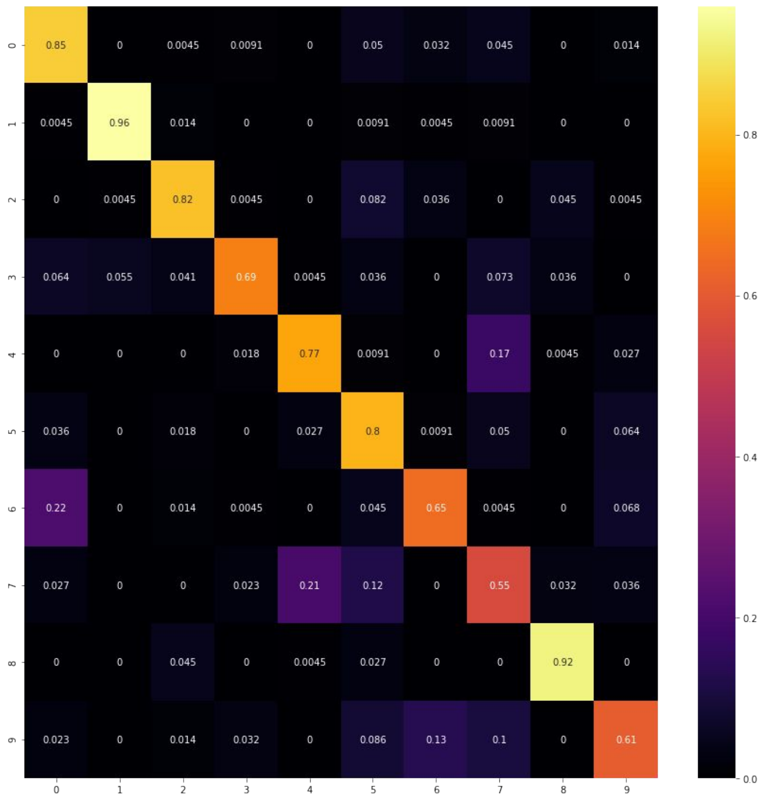
	0	1	2	3	4	5	6	7	8	9
0	0.27	0.027	0	0.014	0.0045	0.2	0.11	0.0091	0	0.36
1	0	0.95	0.0045	0	0.0091	0.014	0	0	0	0.027
2	0	0.023	0.8	0.0045	0	0.11	0	0	0.0045	0.05
3	0.0045	0.17	0.19	0.57	0	0.014	0.023	0	0.0045	0.023
4	0	0.091	0	0	0.8	0.086	0	0.0045	0.0091	0.0091
5	0.027	0.0045	0	0	0	0.86	0	0	0.014	0.095
6	0	0	0.023	0.0091	0	0.091	0.55	0	0	0.33
7	0.0091	0.13	0.0091	0.0045	0.3	0.26	0	0.11	0	0.17
8	0	0.045	0.11	0.014	0	0.0045	0.018	0.0091	0.73	0.073
9	0	0.018	0.032	0.0045	0	0.2	0.0045	0.0045	0	0.74

K-means Covariance Interpretations

The idea of the bias-variance trade-off provides an explanation for the low predictiveness of the full covariance model. Such a model tends to overfit with covariances likely too unique to each cluster, while the other more flexible models allow for more predictability of test data. The high results for tied show that this covariance type lies at an ideal point in the bias-variance curve allowing for sufficient complexity and flexibility.

EM model, non-gendered, spherical

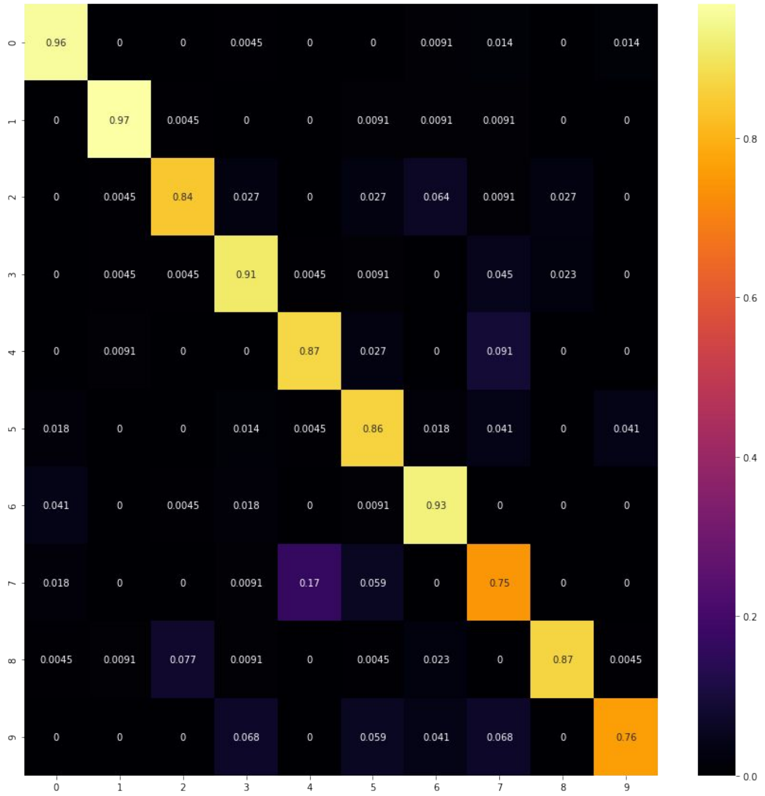
Accuracy:
76.1%



	0	1	2	3	4	5	6	7	8	9
0	0.85	0	0.0045	0.0091	0	0.05	0.032	0.045	0	0.014
1	0.0045	0.96	0.014	0	0	0.0091	0.0045	0.0091	0	0
2	0	0.0045	0.82	0.0045	0	0.082	0.036	0	0.045	0.0045
3	0.064	0.055	0.041	0.69	0.0045	0.036	0	0.073	0.036	0
4	0	0	0	0.018	0.77	0.0091	0	0.17	0.0045	0.027
5	0.036	0	0.018	0	0.027	0.8	0.0091	0.05	0	0.064
6	0.22	0	0.014	0.0045	0	0.045	0.65	0.0045	0	0.068
7	0.027	0	0	0.023	0.21	0.12	0	0.55	0.032	0.036
8	0	0	0.045	0	0.0045	0.027	0	0	0.92	0
9	0.023	0	0.014	0.032	0	0.086	0.13	0.1	0	0.61

EM model, non-gendered, diagonal

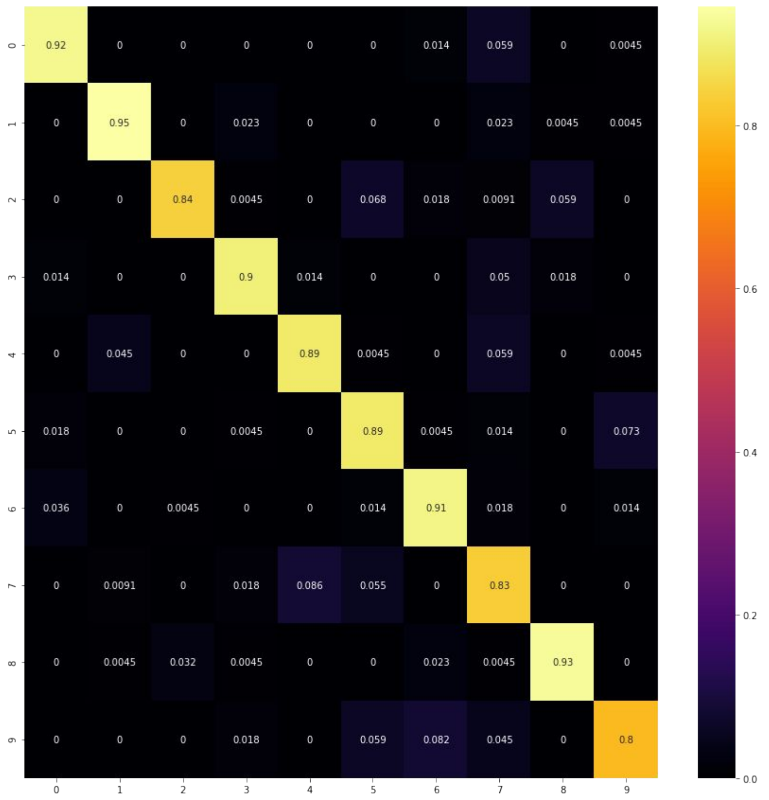
Accuracy:
87.2%



	0	1	2	3	4	5	6	7	8	9
0	0.96	0	0	0.0045	0	0	0.0091	0.014	0	0.014
1	0	0.97	0.0045	0	0	0.0091	0.0091	0.0091	0	0
2	0	0.0045	0.84	0.027	0	0.027	0.064	0.0091	0.027	0
3	0	0.0045	0.0045	0.91	0.0045	0.0091	0	0.045	0.023	0
4	0	0.0091	0	0	0.87	0.027	0	0.091	0	0
5	0.018	0	0	0.014	0.0045	0.86	0.018	0.041	0	0.041
6	0.041	0	0.0045	0.018	0	0.0091	0.93	0	0	0
7	0.018	0	0	0.0091	0.17	0.059	0	0.75	0	0
8	0.0045	0.0091	0.077	0.0091	0	0.0045	0.023	0	0.87	0.0045
9	0	0	0	0.068	0	0.059	0.041	0.068	0	0.76

EM model, non-gendered, tied

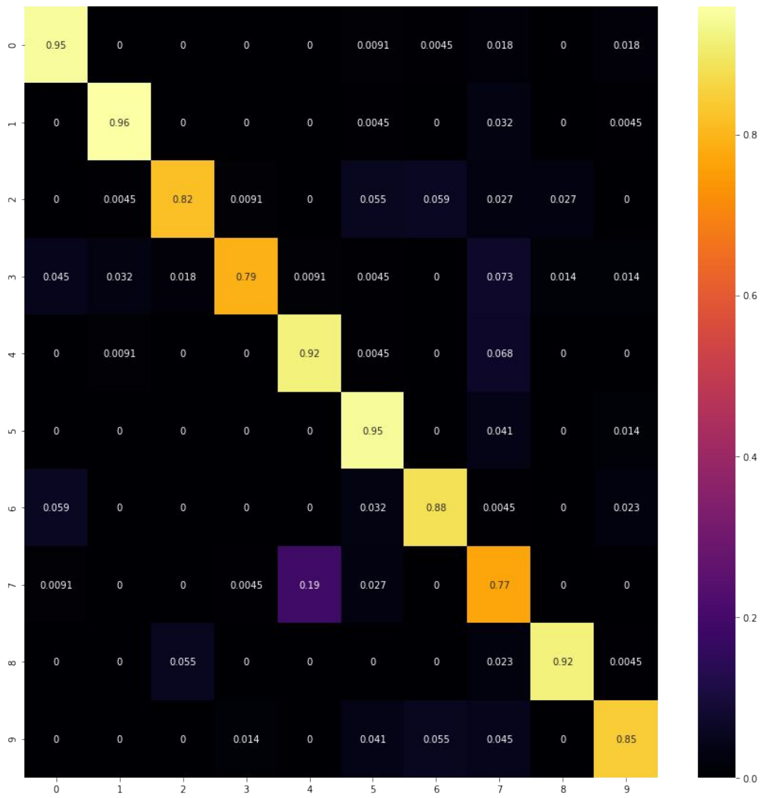
Accuracy:
88.6%



	0	1	2	3	4	5	6	7	8	9
0	0.92	0	0	0	0	0	0.014	0.059	0	0.0045
1	0	0.95	0	0.023	0	0	0	0.023	0.0045	0.0045
2	0	0	0.84	0.0045	0	0.068	0.018	0.0091	0.059	0
3	0.014	0	0	0.9	0.014	0	0	0.05	0.018	0
4	0	0.045	0	0	0.89	0.0045	0	0.059	0	0.0045
5	0.018	0	0	0.0045	0	0.89	0.0045	0.014	0	0.073
6	0.036	0	0.0045	0	0	0.014	0.91	0.018	0	0.014
7	0	0.0091	0	0.018	0.086	0.055	0	0.83	0	0
8	0	0.0045	0.032	0.0045	0	0	0.023	0.0045	0.93	0
9	0	0	0	0.018	0	0.059	0.082	0.045	0	0.8

EM model, non-gendered, full

Accuracy: 88.0%



	0	1	2	3	4	5	6	7	8	9
0	0.95	0	0	0	0	0.0091	0.0045	0.018	0	0.018
1	0	0.96	0	0	0	0.0045	0	0.032	0	0.0045
2	0	0.0045	0.82	0.0091	0	0.055	0.059	0.027	0.027	0
3	0.045	0.032	0.018	0.79	0.0091	0.0045	0	0.073	0.014	0.014
4	0	0.0091	0	0	0.92	0.0045	0	0.068	0	0
5	0	0	0	0	0	0.95	0	0.041	0	0.014
6	0.059	0	0	0	0	0.032	0.88	0.0045	0	0.023
7	0.0091	0	0	0.0045	0.19	0.027	0	0.77	0	0
8	0	0	0.055	0	0	0	0	0.023	0.92	0.0045
9	0	0	0	0.014	0	0.041	0.055	0.045	0	0.85

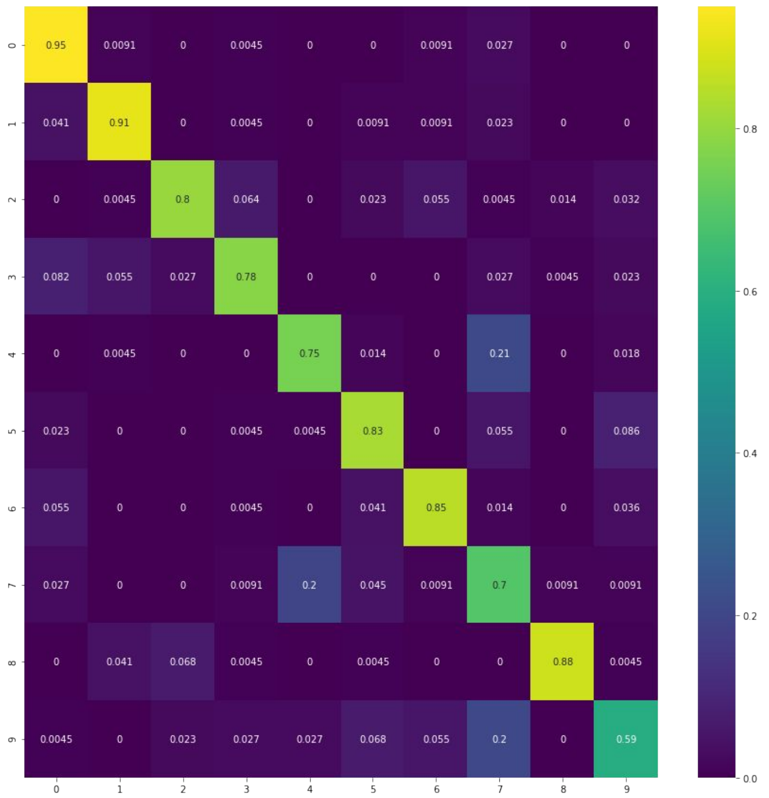
EM Covariance Interpretations

These results were similar to those of K-means, with the full and diagonal covariances models being much more accurate as the notable exception. While I cannot say with full certainty, my guess is for this data the EM algorithm itself generated models with overall less bias than K-means, and the heightened accuracy of these models allowed them to generalize well even with more complexity.

K-means model, gendered, spherical

Accuracy:

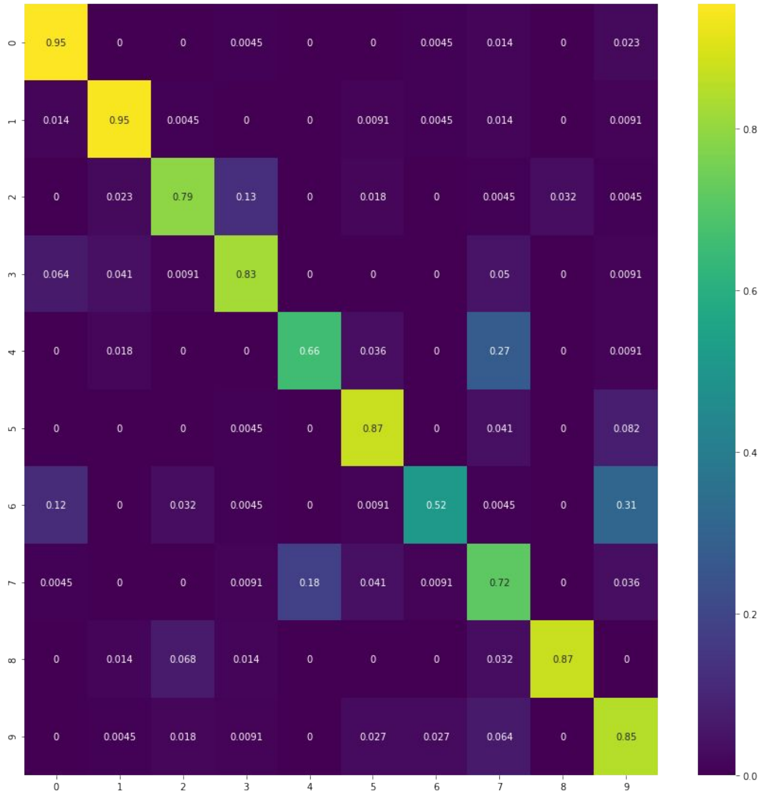
80.6%



	0	1	2	3	4	5	6	7	8	9
0	0.95	0.0091	0	0.0045	0	0	0.0091	0.027	0	0
1	0.041	0.91	0	0.0045	0	0.0091	0.0091	0.023	0	0
2	0	0.0045	0.8	0.064	0	0.023	0.055	0.0045	0.014	0.032
3	0.082	0.055	0.027	0.78	0	0	0	0.027	0.0045	0.023
4	0	0.0045	0	0	0.75	0.014	0	0.21	0	0.018
5	0.023	0	0	0.0045	0.0045	0.83	0	0.055	0	0.086
6	0.055	0	0	0.0045	0	0.041	0.85	0.014	0	0.036
7	0.027	0	0	0.0091	0.2	0.045	0.0091	0.7	0.0091	0.0091
8	0	0.041	0.068	0.0045	0	0.0045	0	0	0.88	0.0045
9	0.0045	0	0.023	0.027	0.027	0.068	0.055	0.2	0	0.59

K-means model, gendered, diagonal

Accuracy:
80.1%

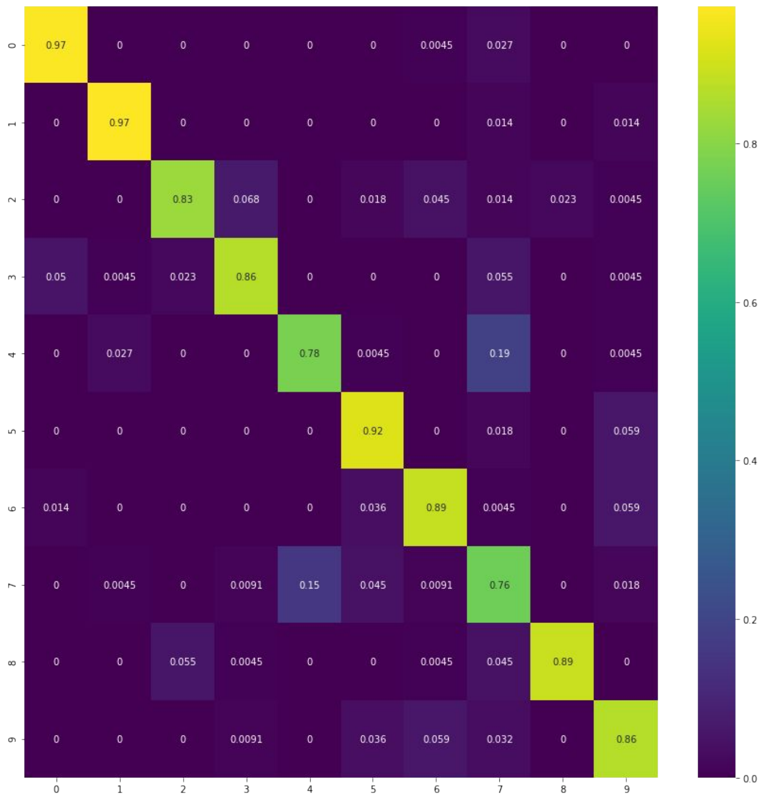


	0	1	2	3	4	5	6	7	8	9
0	0.95	0	0	0.0045	0	0	0.0045	0.014	0	0.023
1	0.014	0.95	0.0045	0	0	0.0091	0.0045	0.014	0	0.0091
2	0	0.023	0.79	0.13	0	0.018	0	0.0045	0.032	0.0045
3	0.064	0.041	0.0091	0.83	0	0	0	0.05	0	0.0091
4	0	0.018	0	0	0.66	0.036	0	0.27	0	0.0091
5	0	0	0	0.0045	0	0.87	0	0.041	0	0.082
6	0.12	0	0.032	0.0045	0	0.0091	0.52	0.0045	0	0.31
7	0.0045	0	0	0.0091	0.18	0.041	0.0091	0.72	0	0.036
8	0	0.014	0.068	0.014	0	0	0	0.032	0.87	0
9	0	0.0045	0.018	0.0091	0	0.027	0.027	0.064	0	0.85

K-means model, gendered, tied

Accuracy:

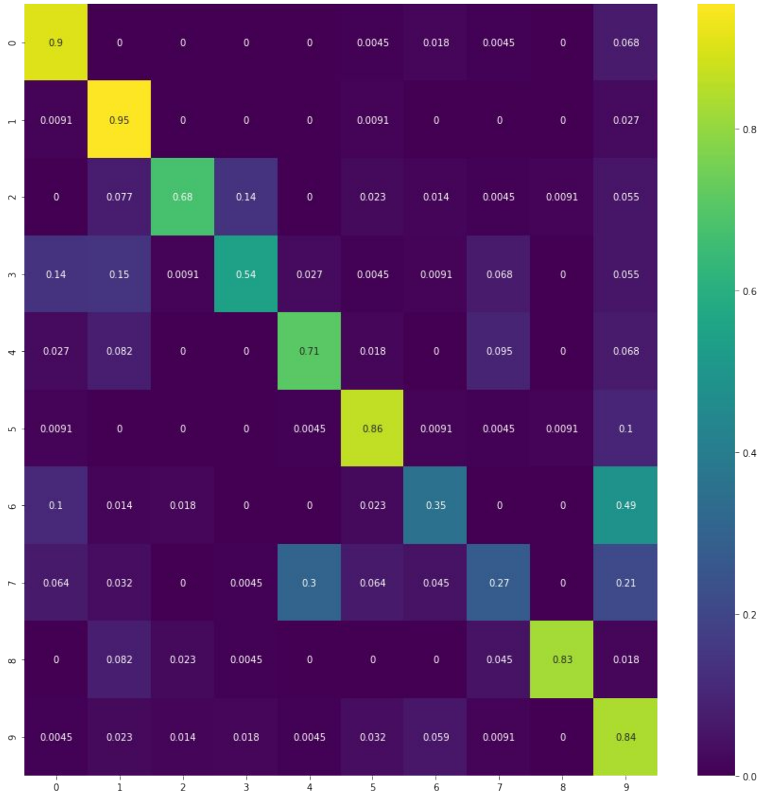
87.3%



	0	1	2	3	4	5	6	7	8	9
0	0.97	0	0	0	0	0	0.0045	0.027	0	0
1	0	0.97	0	0	0	0	0	0.014	0	0.014
2	0	0	0.83	0.068	0	0.018	0.045	0.014	0.023	0.0045
3	0.05	0.0045	0.023	0.86	0	0	0	0.055	0	0.0045
4	0	0.027	0	0	0.78	0.0045	0	0.19	0	0.0045
5	0	0	0	0	0	0.92	0	0.018	0	0.059
6	0.014	0	0	0	0	0.036	0.89	0.0045	0	0.059
7	0	0.0045	0	0.0091	0.15	0.045	0.0091	0.76	0	0.018
8	0	0	0.055	0.0045	0	0	0.0045	0.045	0.89	0
9	0	0	0	0.0091	0	0.036	0.059	0.032	0	0.86

K-means model, gendered, full

Accuracy:
69.4%



	0	1	2	3	4	5	6	7	8	9
0	0.9	0	0	0	0	0.0045	0.018	0.0045	0	0.068
1	0.0091	0.95	0	0	0	0.0091	0	0	0	0.027
2	0	0.077	0.68	0.14	0	0.023	0.014	0.0045	0.0091	0.055
3	0.14	0.15	0.0091	0.54	0.027	0.0045	0.0091	0.068	0	0.055
4	0.027	0.082	0	0	0.71	0.018	0	0.095	0	0.068
5	0.0091	0	0	0	0.0045	0.86	0.0091	0.0045	0.0091	0.1
6	0.1	0.014	0.018	0	0	0.023	0.35	0	0	0.49
7	0.064	0.032	0	0.0045	0.3	0.064	0.045	0.27	0	0.21
8	0	0.082	0.023	0.0045	0	0	0	0.045	0.83	0.018
9	0.0045	0.023	0.014	0.018	0.0045	0.032	0.059	0.0091	0	0.84

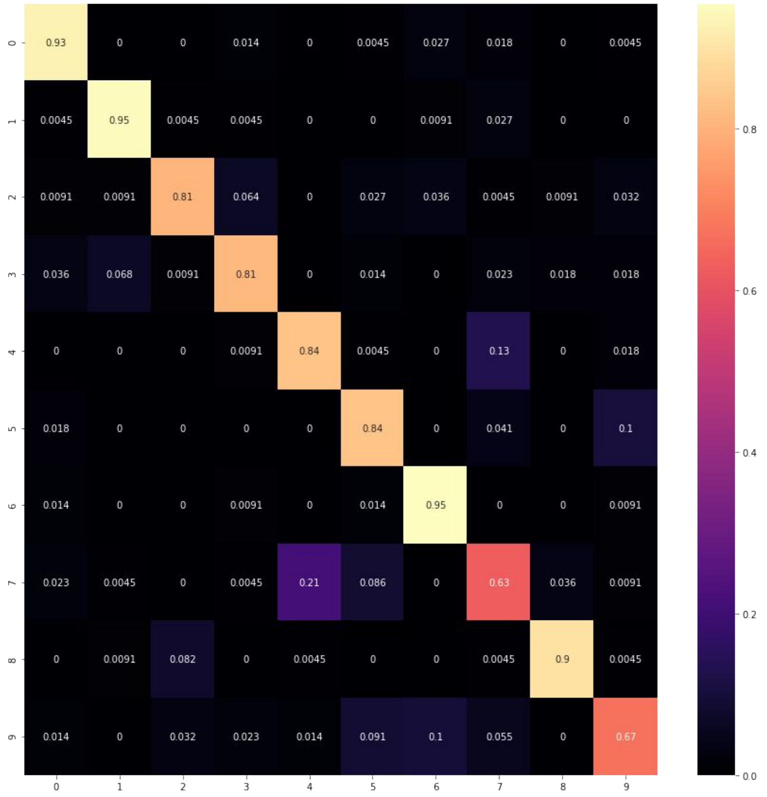
K-means Gender Interpretations

The results from splitting up the models by gender improved the model as I expected it would. This additional supervision of the data in the models allowed for the reduction of a large variable in gender, and allowed for the pattern recognition of finding the digit to be done more easily. The only real disadvantage of this modeling type was increased development time in parsing the data and in the creation and testing of the models, but this was effectively negligible.

EM model, gendered, spherical

Accuracy:

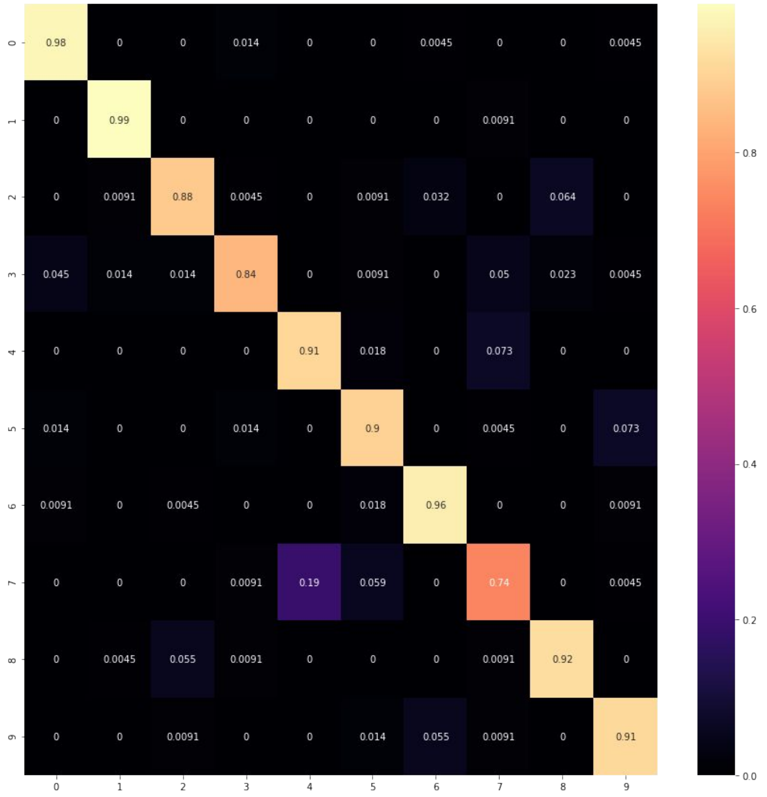
83.3%



	0	1	2	3	4	5	6	7	8	9
0	0.93	0	0	0.014	0	0.0045	0.027	0.018	0	0.0045
1	0.0045	0.95	0.0045	0.0045	0	0	0.0091	0.027	0	0
2	0.0091	0.0091	0.81	0.064	0	0.027	0.036	0.0045	0.0091	0.032
3	0.036	0.068	0.0091	0.81	0	0.014	0	0.023	0.018	0.018
4	0	0	0	0.0091	0.84	0.0045	0	0.13	0	0.018
5	0.018	0	0	0	0	0.84	0	0.041	0	0.1
6	0.014	0	0	0.0091	0	0.014	0.95	0	0	0.0091
7	0.023	0.0045	0	0.0045	0.21	0.086	0	0.63	0.036	0.0091
8	0	0.0091	0.082	0	0.0045	0	0	0.0045	0.9	0.0045
9	0.014	0	0.032	0.023	0.014	0.091	0.1	0.055	0	0.67

EM model, gendered, diagonal

<u>Accuracy:</u>
90.3%

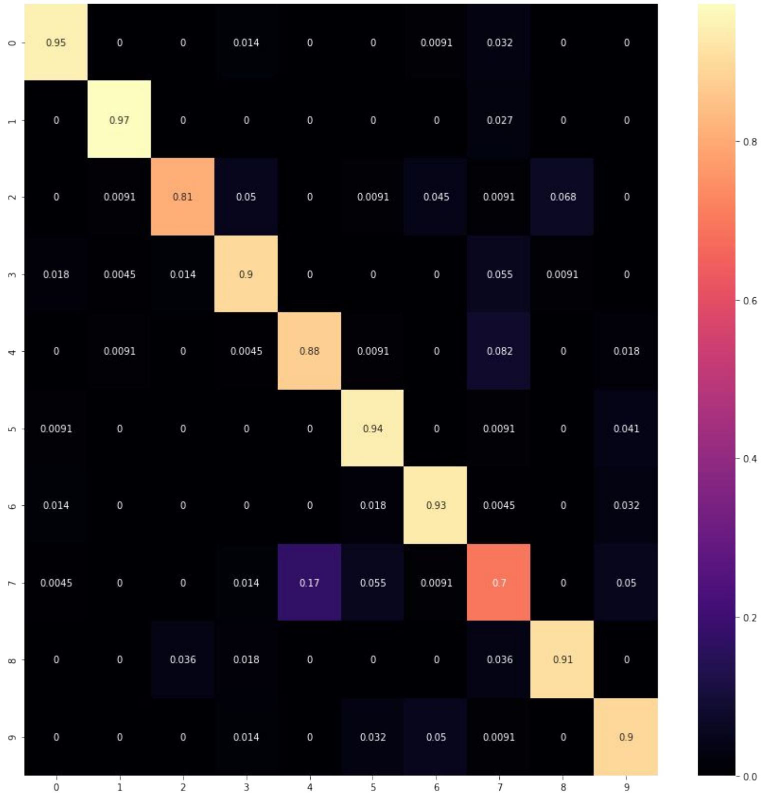


	0	1	2	3	4	5	6	7	8	9
0	0.98	0	0	0.014	0	0	0.0045	0	0	0.0045
1	0	0.99	0	0	0	0	0	0.0091	0	0
2	0	0.0091	0.88	0.0045	0	0.0091	0.032	0	0.064	0
3	0.045	0.014	0.014	0.84	0	0.0091	0	0.05	0.023	0.0045
4	0	0	0	0	0.91	0.018	0	0.073	0	0
5	0.014	0	0	0.014	0	0.9	0	0.0045	0	0.073
6	0.0091	0	0.0045	0	0	0.018	0.96	0	0	0.0091
7	0	0	0	0.0091	0.19	0.059	0	0.74	0	0.0045
8	0	0.0045	0.055	0.0091	0	0	0	0.0091	0.92	0
9	0	0	0.0091	0	0	0.014	0.055	0.0091	0	0.91

EM model, gendered, tied

Accuracy:

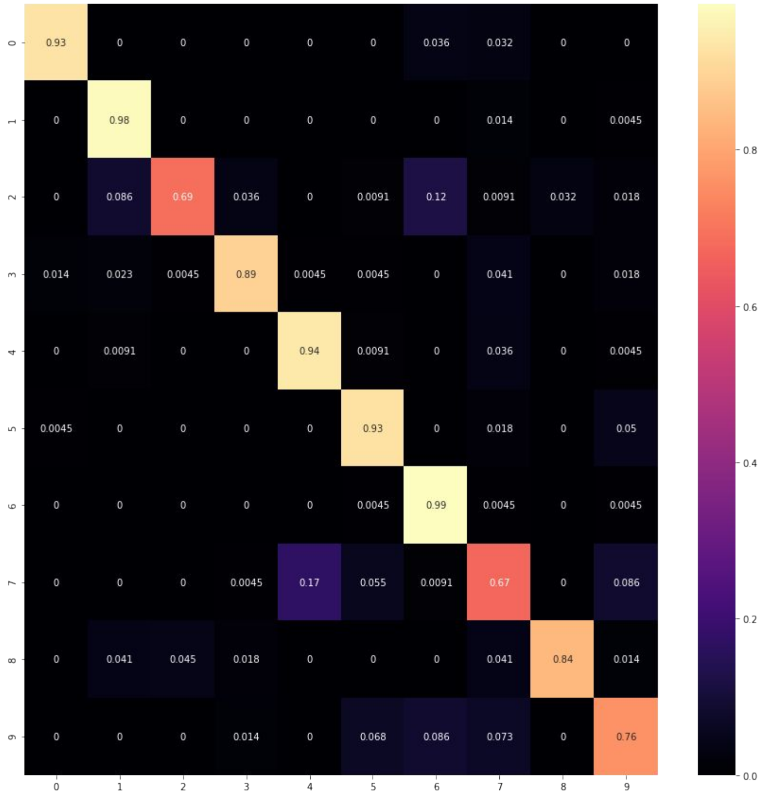
88.8%



	0	1	2	3	4	5	6	7	8	9
0	0.95	0	0	0.014	0	0	0.0091	0.032	0	0
1	0	0.97	0	0	0	0	0	0.027	0	0
2	0	0.0091	0.81	0.05	0	0.0091	0.045	0.0091	0.068	0
3	0.018	0.0045	0.014	0.9	0	0	0	0.055	0.0091	0
4	0	0.0091	0	0.0045	0.88	0.0091	0	0.082	0	0.018
5	0.0091	0	0	0	0	0.94	0	0.0091	0	0.041
6	0.014	0	0	0	0	0.018	0.93	0.0045	0	0.032
7	0.0045	0	0	0.014	0.17	0.055	0.0091	0.7	0	0.05
8	0	0	0.036	0.018	0	0	0	0.036	0.91	0
9	0	0	0	0.014	0	0.032	0.05	0.0091	0	0.9

EM model, non-gendered, full

Accuracy:
86.2%



	0	1	2	3	4	5	6	7	8	9
0	0.93	0	0	0	0	0	0.036	0.032	0	0
1	0	0.98	0	0	0	0	0	0.014	0	0.0045
2	0	0.086	0.69	0.036	0	0.0091	0.12	0.0091	0.032	0.018
3	0.014	0.023	0.0045	0.89	0.0045	0.0045	0	0.041	0	0.018
4	0	0.0091	0	0	0.94	0.0091	0	0.036	0	0.0045
5	0.0045	0	0	0	0	0.93	0	0.018	0	0.05
6	0	0	0	0	0	0.0045	0.99	0.0045	0	0.0045
7	0	0	0	0.0045	0.17	0.055	0.0091	0.67	0	0.086
8	0	0.041	0.045	0.018	0	0	0	0.041	0.84	0.014
9	0	0	0	0.014	0	0.068	0.086	0.073	0	0.76

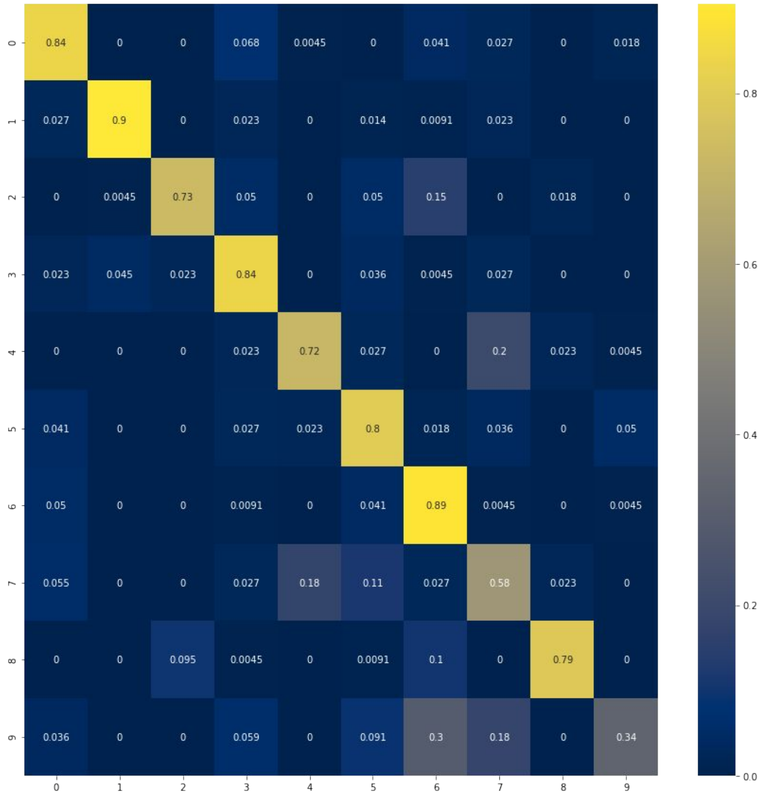
EM Gender Interpretations

These results for the gendered EM models essentially lead us to the same conclusions as the K-means gendered models. The additional labels on the data as either male or female allowed for better predictions, as we expected.

K-means model, spherical, split MFCC’s : 1-12

Accuracy:

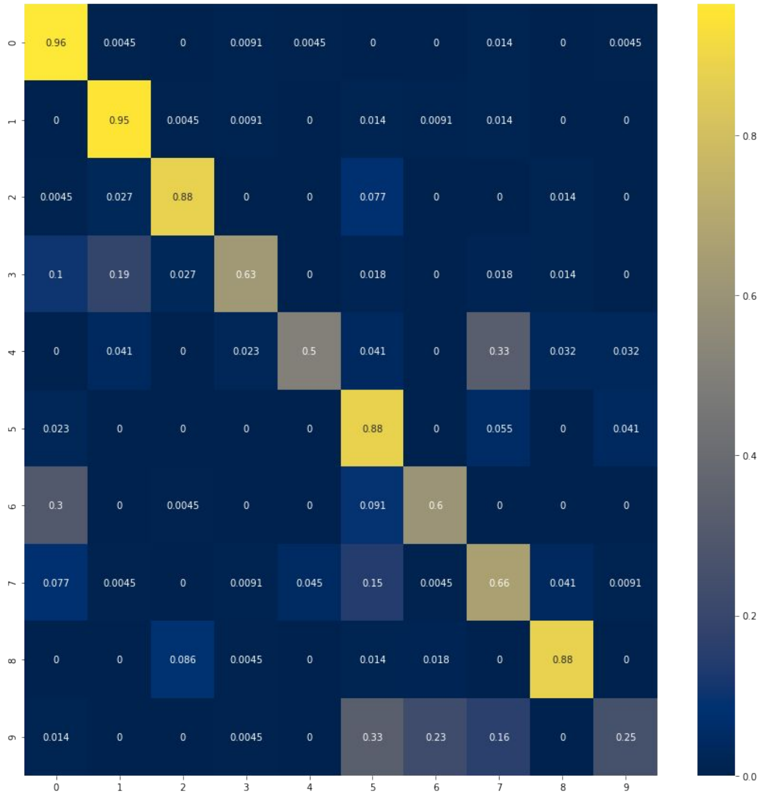
74.4%



	0	1	2	3	4	5	6	7	8	9
0	0.84	0	0	0.068	0.0045	0	0.041	0.027	0	0.018
1	0.027	0.9	0	0.023	0	0.014	0.0091	0.023	0	0
2	0	0.0045	0.73	0.05	0	0.05	0.15	0	0.018	0
3	0.023	0.045	0.023	0.84	0	0.036	0.0045	0.027	0	0
4	0	0	0	0.023	0.72	0.027	0	0.2	0.023	0.0045
5	0.041	0	0	0.027	0.023	0.8	0.018	0.036	0	0.05
6	0.05	0	0	0.0091	0	0.041	0.89	0.0045	0	0.0045
7	0.055	0	0	0.027	0.18	0.11	0.027	0.58	0.023	0
8	0	0	0.095	0.0045	0	0.0091	0.1	0	0.79	0
9	0.036	0	0	0.059	0	0.091	0.3	0.18	0	0.34

K-means model, diagonal, split MFCC's : 1-12

Accuracy:
72.0%

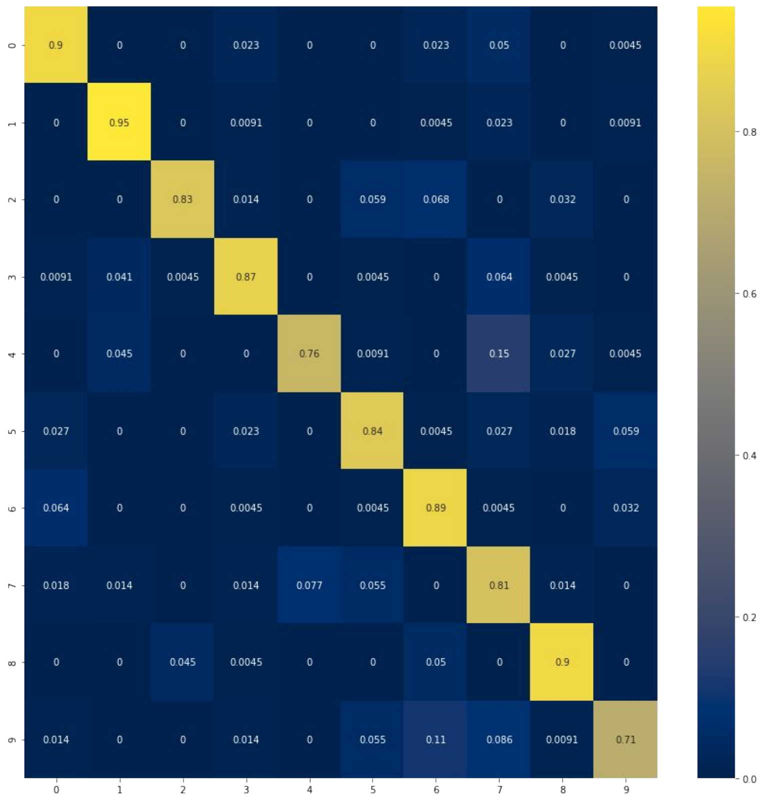


	0	1	2	3	4	5	6	7	8	9
0	0.96	0.0045	0	0.0091	0.0045	0	0	0.014	0	0.0045
1	0	0.95	0.0045	0.0091	0	0.014	0.0091	0.014	0	0
2	0.0045	0.027	0.88	0	0	0.077	0	0	0.014	0
3	0.1	0.19	0.027	0.63	0	0.018	0	0.018	0.014	0
4	0	0.041	0	0.023	0.5	0.041	0	0.33	0.032	0.032
5	0.023	0	0	0	0	0.88	0	0.055	0	0.041
6	0.3	0	0.0045	0	0	0.091	0.6	0	0	0
7	0.077	0.0045	0	0.0091	0.045	0.15	0.0045	0.66	0.041	0.0091
8	0	0	0.086	0.0045	0	0.014	0.018	0	0.88	0
9	0.014	0	0	0.0045	0	0.33	0.23	0.16	0	0.25

K-means model, tied, split MFCC’s : 1-12

Accuracy:

84.7%

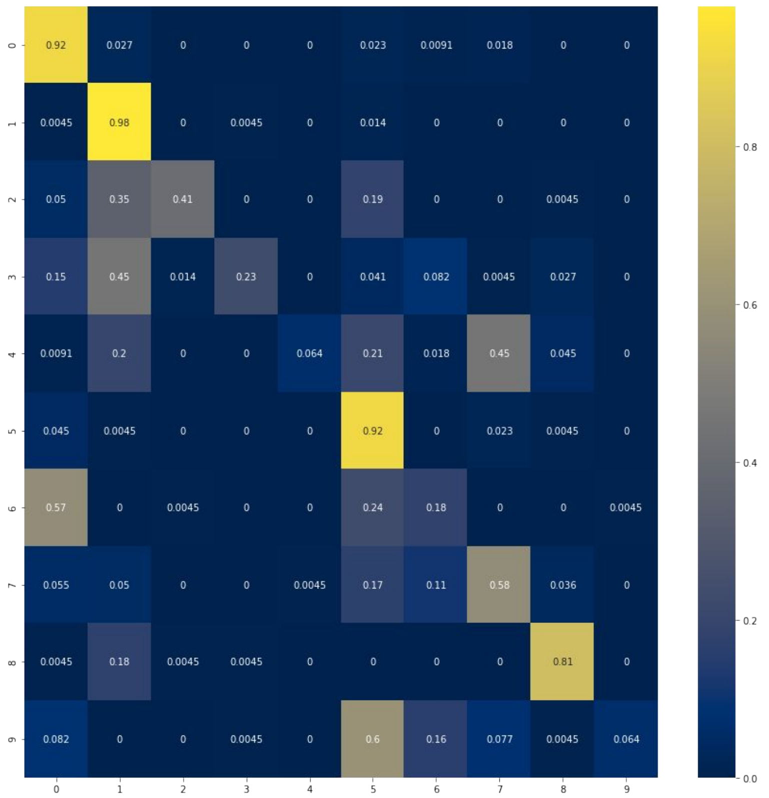


	0	1	2	3	4	5	6	7	8	9
0	0.9	0	0	0.023	0	0	0.023	0.05	0	0.0045
1	0	0.95	0	0.0091	0	0	0.0045	0.023	0	0.0091
2	0	0	0.83	0.014	0	0.059	0.068	0	0.032	0
3	0.0091	0.041	0.0045	0.87	0	0.0045	0	0.064	0.0045	0
4	0	0.045	0	0	0.76	0.0091	0	0.15	0.027	0.0045
5	0.027	0	0	0.023	0	0.84	0.0045	0.027	0.018	0.059
6	0.064	0	0	0.0045	0	0.0045	0.89	0.0045	0	0.032
7	0.018	0.014	0	0.014	0.077	0.055	0	0.81	0.014	0
8	0	0	0.045	0.0045	0	0	0.05	0	0.9	0
9	0.014	0	0	0.014	0	0.055	0.11	0.086	0.0091	0.71

K-means model, full, split MFCC’s : 1-12

Accuracy:

51.6%



	0	1	2	3	4	5	6	7	8	9
0	0.92	0.027	0	0	0	0.023	0.0091	0.018	0	0
1	0.0045	0.98	0	0.0045	0	0.014	0	0	0	0
2	0.05	0.35	0.41	0	0	0.19	0	0	0.0045	0
3	0.15	0.45	0.014	0.23	0	0.041	0.082	0.0045	0.027	0
4	0.0091	0.2	0	0	0.064	0.21	0.018	0.45	0.045	0
5	0.045	0.0045	0	0	0	0.92	0	0.023	0.0045	0
6	0.57	0	0.0045	0	0	0.24	0.18	0	0	0.0045
7	0.055	0.05	0	0	0.0045	0.17	0.11	0.58	0.036	0
8	0.0045	0.18	0.0045	0.0045	0	0	0	0	0.81	0
9	0.082	0	0	0.0045	0	0.6	0.16	0.077	0.0045	0.064

Subsets of MFCC’s Interpretations

The results for taking only the first 12 MFCC’s reaffirmed my expectations that training on 12/13 of the data would give us worse results than training on the whole data. Even though some of the higher coefficients appeared less informative on some of the plots I studied, having more data in these sorts of numerical approaches tends to lead to better results. This sort of approach would potentially have worked well had we known the measurements of a particular coefficient were very noisy.

03

Takeaways

Accuracy by digit:

Across all the models, digits **0, 1** and **8** appeared to be most accurate, while the results for digits **2** and **7** appeared to be the lowest across the digits. **Sifir**, **wahad**, and **thamanieh** do not appear to share an obvious pattern, and neither do **ithnayn** or **seb’a**, so I believe the differences in these accuracies may come down to the choice in number of clusters. The digits classified accurately may also have more standard pronunciations across Arabic dialects while there may be more variation for the digits classified with lower accuracy.

Conclusions: Covariance Types

Covariance type was one modeling choice that showed a large impact on model performance. This makes sense considering the effect of the covariance type on determining the complexity and flexibility of the model. Full covariances provide a much more complex and less flexible model than spherical covariances, and these differences were exemplified in the results. All in all, models using tied covariance appeared to be most predictive for this data, with the highest performance for all but one combinations of modeling parameters. Full covariance models were least accurate for K-means while spherical covariance models were least accurate for EM.

Conclusions: Gendered models

Separating models based on gender had a significant and consistent improvement across accuracies for all of the models. This is what we would expect with the additional supervision of our data, knowing an additional important variable about each data point.

Conclusions: Number of clusters/components

Number of clusters/components chosen was certainly an important modeling choice, but was generally similarly successful within a range, allowing room for error. If these numbers were thrown off drastically then this largely impacted the predictiveness of the models, but I found these parameters to be less useful and important once clusters near the 2n - 1 phonemes rule was used.

Conclusions: Subsets of MFCC’s

I found taking subsets of the MFCC’s to be generally not useful as a modeling choice. Without any knowledge of some of the MFCCs being noisier than others, I found results consistent with simply being provided less data. This resulted in lower accuracies any time I took subsets of the coefficients. For these reasons I found this to be a less important modeling choice than some of the aforementioned ones.

Ideal System

Based on the results, I would choose an EM gendered model (separate for men and women), using diagonal covariance and **6, 5, 5, 6, 7, 6, 5, 5, 9, and 4** components for the Gaussian mixture model of each component respectively.

The reasoning behind this is that across all of the models I tested, this model had the highest accuracy of all with **90.3% accuracy.**

Reflections on modeling system

This modeling system worked exceptionally well in connecting theory from the course into actual implementation, allowing us to achieve anywhere from **80-90% accuracy** in our best models. These were sufficiently simple to implement, and allowed us to explore the unique effects of different modeling choices on the accuracies of our model. Some things I would improve would be to potentially find a model that could breach accuracies of over 95% accuracy, as it would feel very impressive to reach such levels of prediction in an introductory statistical learning class.

More reflections

While the maximum likelihood classification that was used in the system worked well, and was an appropriate extension of the mathematics which we learned in class, it would have also been interesting to work with a different kind of classifier as well.

Lessons Learned

Next time in doing this project I would have worked from the get-go to get started on this project, and not have been prohibited by the initial inertia that comes from taking on a large project. Once the initial energy is put in to understanding this project, I found it incredibly interesting and a fun way to spend my time in trying to improve the models' accuracies. I would love to have more time to continue playing with the modeling choices and even new kinds of models, as this is one of the most enjoyable project experiences I have had while at Duke.

More Lessons Learned

I found that working with people worked really well for this project, both in the sense that I would learn of new approaches and get excited about them, as well as I could deepen my understanding of the topics during conversations with friends about them. I enjoyed the mix of collaboration with individual responsibility, and all in all that made this project a great pleasure to work on.

04

Credits

Collaborators

Wyatt Focht

Discussed conceptual questions and reasoning for modeling choices

Paul Truitt

Debated reasonings for certain models performing better than others

Collaborators

Remy Cross

Worked together on parsing the data and discussing best approaches for modeling

Matthew Giglio

Discussed packages to use for the different models. Helped him with finding covariances, helped me with finding maximum likelihood from package

Collaborators

Arman Shekarriz

Discussed general approaches and results

Owen Mackenzie

Discussed initial approaches to modeling and data organization

James Lim

Discussed visualizations and helped him with making GMM from K-means

Cited sources

- [1] Project description provided by Dr. Tantum
- [2] 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.
- [3] Reynolds, Douglas A. "Gaussian mixture models." Encyclopedia of biometrics 741 (2009): 659-663.

- [4] Simon Rogers and Mark Girolami. 2016. A First Course in Machine Learning, Second Edition (2nd. ed.). Chapman & Hall/CRC.

- [5] T. Kanungo, D. M. Mount, N. S. Netanyahu, C. D. Piatko, R. Silverman and A. Y. Wu, "An efficient k-means clustering algorithm: analysis and implementation," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 24, no. 7, pp. 881-892, July 2002, doi: 10.1109/TPAMI.2002.1017616.

Cited sources

- Scikit-learn: Machine Learning in Python, Pedregosa *et al.*, JMLR 12, pp. 2825-2830, 2011.
    - sklearn packages used:
        - PCA from sklearn.decomposition
        - KMeans from sklearn.cluster
        - GaussianMixture from sklearn.mixture
        - confusion_matrix from sklearn.metrics

Citations of other packages used

- J. D. Hunter, "Matplotlib: A 2D Graphics Environment", Computing in Science & Engineering, vol. 9, no. 3, pp. 90-95, 2007.
- 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. (Publisher link).
- Waskom, M. et al., 2017. *mwaskom/seaborn: v0.8.1 (September 2017)*, Zenodo. Available at: https://doi.org/10.5281/zenodo.883859.
- Data structures for statistical computing in python, McKinney, Proceedings of the 9th Python in Science Conference, Volume 445, 2010.

Thank you for reading through!