Lab 8: Classification

EECS 16B Fall 2023

https://links.eecs16b.org/lab8-slides

Classification Lab!

Note: many SVD/PCA concepts are new and won’t be covered in lecture until next week, so please pay close attention to the presentation & ask any questions as needed!

Today’s Lab

- Read lab note before starting lab!
- Long lab!
  a. Front end verification
  b. Voice data collection - record 6 different words!
  c. Data preprocessing
  d. Use SVD and PCA to find PCA basis to project data onto
  e. Find mean clusters to distinguish between words
  f. Implement and test classifier
  g. Tuning Hyperparameters for Best Performance - achieve 80% accuracy on 4 chosen words
  h. Arduino implementation of PCA Classify to classify words live

Part 1: Mic board Circuit Verification

- Check the voltage at each node of mic board:
  - VDD: 5 V
  - VSS: Ground
  - OUT: centered ~2.5 V
  - OS2: ~2.5 V
  - OS1: ~ <2.5 V
- Check signal at front end audio circuit output:
  - Output of non-inverting amplifier for low-pass filter
  - Centered ~2.5 V
  - > 2.5 V Vpp when making noise at mic

Part 2: Data Collection and Speech Patterns

- SIXT33N will know four voice commands
  - Correspond to go straight far, straight close, turn left, turn right
- Make recordings of each of our word commands
  - record 6 words to be safe, choose the best 4 later
  - You may choose any words you want!

Word Choice Guidelines

- Speech pattern recognition, not word recognition
  - Actual word doesn’t matter, but the speech pattern for the word does!
  - Enunciate syllables well to make clear distinctions
- Generally, try to use words with:
  - Different syllables (e.g. pear, apple, banana, watermelon)
  - different endings (hard vs. soft, , e.g back vs. shoe)
- Remember the way you say each word, as you’ll have to replicate it later!
  - Can record yourself with your phone so you have a record of how you said each word

Data Collection Guidelines

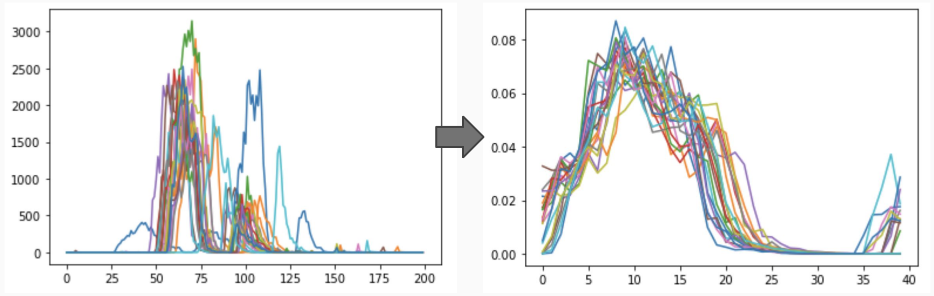
- The Arduino will record ~2 seconds of micboard output every few seconds
  - When (any) Arduino LEDs are ON, then it is RECORDING
  - The three Arduino LEDs will count up over the ~2 second duration
  - Data is then passed to your computer, saved to a csv file
- Try to say each word the *same way* at the *same distance* from the mic
- Make sure your entire word gets captured by the recording window
- You will only be powering your Arduino **through the USB** this week
  - Ensure that you are using the PSU for front-end power, and not your batteries

Data Collection Guidelines (continued)

- Collect ~40-45 recordings for each of 6 words
  - recommend checking your words by a TA/ASE to make sure they are distinct enough
  - don’t spend too much time choosing words! This is a long lab
- Manually delete outliers to have exactly 40 recordings per word
  - Tip: check for outliers by making line plots of .csv file in Excel
  - If you don’t have any major outliers, delete any rows from the start/end until you reach 40

Part 3: Data Preprocessing/ Word Alignment

- Trim and align each recording to locate and isolate the spoken word
  - ***Threshold*** - percentage of the max value for the sample counts as a spoken command
  - ***Pre-length -*** how many timesteps before we hit this threshold did we start speaking the word
  - ***Length -*** how long the sample is



Word example: "meat"

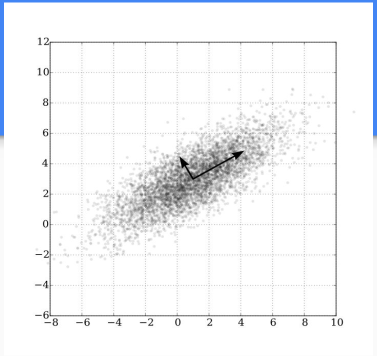
Part 4: Computing SVD on our Data

- split our data into (70%) training and (30%) testing data
  - Use training data for the steps below, save testing data for Part 6
- stack the aligned words in a data matrix
- Zero-mean ("demean") the data using the mean of each timestep (each "feature") in preparation for SVD

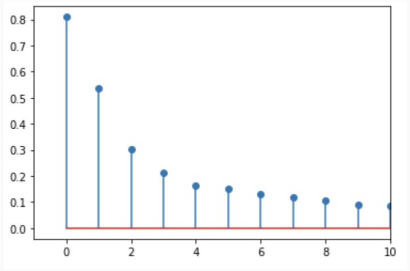
$$\text{demeaned\_A} = \underbrace{\begin{bmatrix} \text{word1\_processed\_train} \\ \text{word2\_processed\_train} \\ \text{word3\_processed\_train} \\ \text{word4\_processed\_train} \end{bmatrix}}_{\text{processed\_A}} - \text{mean(processed\_A)}$$

SVD and PCA

- **PCA** = Principal Component Analysis
  - **Principal components:** basis vectors that maximize variation
  - Oftentimes, we can capture most of the data's behavior with just a few principal components!
    - Fewer dimensions is easier to work with, especially on Arduino
- **SVD** = Singular Value Decomposition
  - gives us a way to find the principal components of a data set
  - Most significant principal components correspond to largest sigma/singular value
  - Numpy has useful functions for us: **numpy.linalg.svd()**



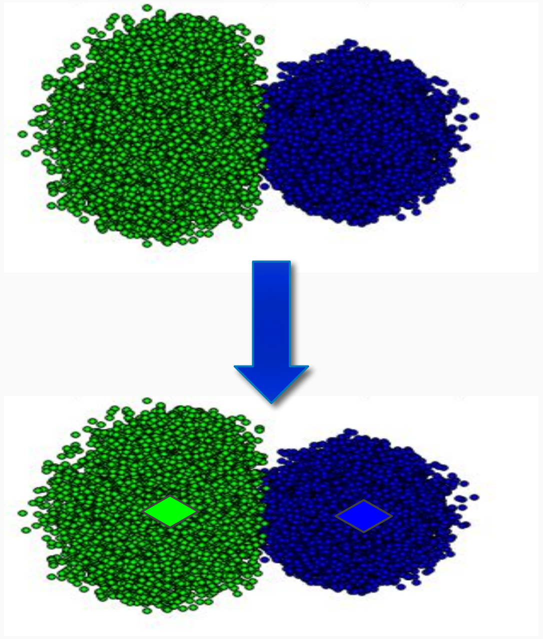
Principal Components of Data Example



Sigma Values Example

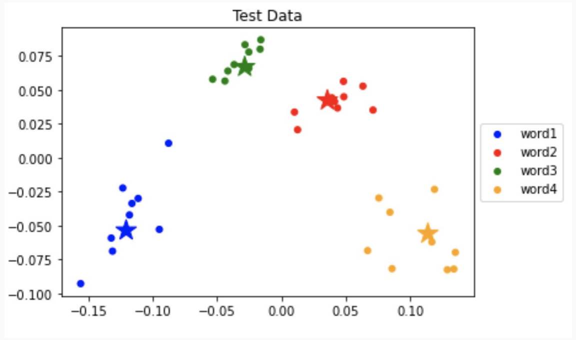
Part 5: Mean Centroid Classification

- From Part 4, we now have:
  - Labelled training data
  - Axes of most variation (PCA basis vectors)
- Project data onto basis vectors
- Find projected centroid (mean) for each word
  - Classify new, unlabelled data by projecting it onto these basis vectors and finding which centroid it is closest to



Part 6: Testing your classifier + Tuning

- From Part 4: we saved 30% of our data as test data
- Verify the accuracy of our classifier
  - project test data onto our PCA vectors
  - subtract projected mean vector
  - assign to closest centroid in 3D space
  - check if classified centroid is the same as the data label
- Aiming for 80% accuracy



Part 7: Arduino Implementation

- Two additional parameters in classify.ino:
  - EUCLIDEAN_THRESHOLD
    - The classified word must be within *EUCLIDIAN_THRESHOLD* distance of the centroid in order to be successfully classified
    - Otherwise, it is considered noise
    - Refer to centroid plots from SVD/PCA for reasonable values
  - LOUDNESS_THRESHOLD (typically >100)
    - Minimum volume needed for the Arduino to attempt to classify
    - Ensure that the Arduino doesn’t attempt to classify background noise

Tips, Tricks, and Warnings

- Make sure the pins used in your code are the same as on your Arduino!
- Do NOT plug in your batteries in this lab
- You have free reign over choosing your words, but choose them in an educated manner according to the guidelines!
    - You don’t want to have to keep recording words because they sound too similar
    - Note that you will have to use your words in front of course staff!
- If the word isn’t classifying properly, you can add print statements to help debug what’s happening in classify.ino!

Forms & Information

- Help request form: https://eecs16b.org/lab-help
- Checkoff request form: https://eecs16b.org/lab-checkoff
- Extension Requests: https://eecs16b.org/extensions
- Makeup Lab: https://makeup.eecs16b.org
- Slides: https://links.eecs16b.org/lab8-slides
- Anon Feedback: https://eecs16b.org/lab-anon-feedback
- Lab Grades error: https://links.eecs16b.org/lab-checkoff-error