Statistical Machine Learning Coursework

Student ID: UP2089114

May 21, 2025

Question 1

(a) When $k = 2$:

$$f_X(x; \mu, 2) = \frac{1}{2c} \exp\left(-(x-\mu)^2\right) \Rightarrow \textbf{Normal distribution}.$$

The likelihood is:

$$L(\mu) \propto \prod_{i=1}^{n} \exp\left(-(x_i-\mu)^2\right) = \exp\left(-\sum_{i=1}^{n}(x_i-\mu)^2\right)$$

The log-likelihood is:

$$\ell(\mu) = ln(\exp\left(-\sum_{i=1}^{n}(x_i-\mu)^2\right) = \ell(\mu) = -\sum_{i=1}^{n}(x_i-\mu)^2$$

Taking the derivative:

$$\frac{d}{d\mu}\ell(\mu) = -2\sum_{i=1}^{n}(x_i-\mu) = -2\left(\sum x_i - n\mu\right)$$

Minimising derivative:

$$-2\left(\sum x_i - n\mu\right) = 0 \Rightarrow \sum x_i - n\mu = 0$$

$$\boxed{\hat{\mu}_{\text{MLE}} = \frac{1}{n}\sum x_i = mean(x)}$$

(b) When $k = 1$:

$$f_X(x; \mu, 1) = \frac{1}{c} \exp\left(-|x-\mu|\right) \Rightarrow \textbf{Laplace distribution}.$$

Likelihood:

$$L(\mu) = \prod_{i=1}^{n} \exp(-|x_i-\mu|) = \exp\left(-\sum_{i=1}^{n}|x_i-\mu|\right)$$

Log-likelihood:

$$\ell(\mu) = \ln(L(\mu)) = -\sum_{i=1}^{n} |x_i - \mu|$$

MLE:

$$\frac{d}{d\mu}\ell(\mu) = -\sum_{i=1}^{n} \frac{d}{d\mu}|x_i - \mu| = -\sum_{i=1}^{n} \text{sign}(x_i - \mu)$$

Where:

$$\text{sign}(x_i - \mu) = \begin{cases} 1 & \text{if } x_i > \mu \\ 0 & \text{if } x_i = \mu \\ -1 & \text{if } x_i < \mu \end{cases}$$

For the MLE to sum to zero, the number of x_i value that are bigger than μ need to balance the number less than μ. This only occurs when μ is the median value.

$$\boxed{\text{Therefore, when } \frac{d}{d\mu}\ell(\mu) = 0, \text{ then } \mu = \text{median}(x)}$$

Question 2

(a) The Bernoulli PMF is:

$$f_X(x;p) = \begin{cases} p & \text{if } x = 1 \\ 1-p & \text{if } x = 0 \end{cases} \quad \Rightarrow \quad f_X(x;p) = p^x(1-p)^{1-x}$$

(b) The likelihood is:

$$L(p) = \prod_{i=1}^{n} f_X(x_i;p) = \prod_{i=1}^{n} p^{x_i}(1-p)^{1-x_i} = \boxed{p^{s_n}(1-p)^{n-s_n}}, \quad \text{where } s_n = \sum x_i$$

(c) The log-likelihood is:

$$\ell(p) = \log L(p) = s_n \log p + (n - s_n)\log(1-p)$$

Taking the derivative:

$$\begin{aligned} \frac{d\ell}{dp} &= \frac{s_n}{p} - \frac{n-s_n}{1-p} \\ 0 &= \frac{s_n}{p} - \frac{n-s_n}{1-p} \\ \Rightarrow \frac{s_n}{p} &= \frac{n-s_n}{1-p} \\ \Rightarrow p &= \frac{s_n}{n} \end{aligned}$$

$$\boxed{t(x) = \frac{s_n}{n},\ \text{and with } s_n = 130,\ n = 200:\ \hat{p} = \frac{130}{200} = 0.65}$$

(d) For Bernoulli trials:

$$\text{Var}(p) = \frac{p(1-p)}{n} \quad \Rightarrow \quad \text{SE} = \sqrt{\frac{p(1-p)}{n}}$$

Since p is unknown, we use the estimate $\hat{p}_{\text{MLE}} = \frac{s_n}{n}$.

$$\boxed{\widehat{\text{SE}} = \sqrt{\frac{\hat{p}(1-\hat{p})}{n}}, \quad \text{where } \hat{p} = \frac{s_n}{n} = \frac{\sum x_i}{n}}$$

(e)

$$\hat{p} = \frac{130}{200} = 0.65 \qquad \widehat{\text{SE}} = \sqrt{\frac{0.65(1-0.65)}{200}} \approx 0.0337$$

$$95\%\ \text{CI: } \hat{p} \pm 1.96 \cdot \widehat{\text{SE}} = 0.65 \pm 1.96 \cdot 0.0337 = \boxed{(0.584,\ 0.716)}$$

Interpretation: If we repeated this experiment many times, we are 95% confident that this interval contains the true value of p.

Fisher's Principle: Fisher's principle states the expected sex ratio should be 50:50 (even split between number of male and female badgers) so p=0.5. 0.5 is outside the confidence interval, so there is strong evidence this violates Fisher's principle. It is likely this badger colony is female-biased as the interval is closer 1 (female).

(f) Taking a Bayesian approach, treating P as a random variable.

1. **Prior:** Uniform prior on $p \in [0, 1]$

$$f_P(p) = \begin{cases} 1 & \text{if } 0 \le p \le 1 \\ 0 & \text{otherwise} \end{cases} \quad \Rightarrow \quad f_P(p) \propto 1$$

2. **Likelihood:**

$$f_X(x \mid p) = p^{s_n}(1-p)^{n-s_n}$$

3. **Posterior:** Proportional to likelihood $\times$ prior

$$f_{P|X}(p \mid x) \propto p^{s_n}(1-p)^{n-s_n}$$

4. **Shape of posterior:** Matches the form of the Beta distribution:

$$f_{P|X}(p \mid x) = \text{Beta}(p \mid \alpha, \beta) \quad \text{where } \alpha = s_n + 1,\ \beta = n - s_n + 1$$

5. **Posterior (normalised form):**

$$\boxed{f_{P|X}(p \mid x) = \frac{1}{B(s_n + 1,\ n - s_n + 1)} \cdot p^{s_n}(1-p)^{n-s_n}}$$

6. **Distribution:**

$$f_{P|X}(p \mid x) \sim \text{Beta}(s_n + 1, n - s_n + 1)$$

(g)

Parameters: Given $s_n = 130$, $n = 200$, we have:

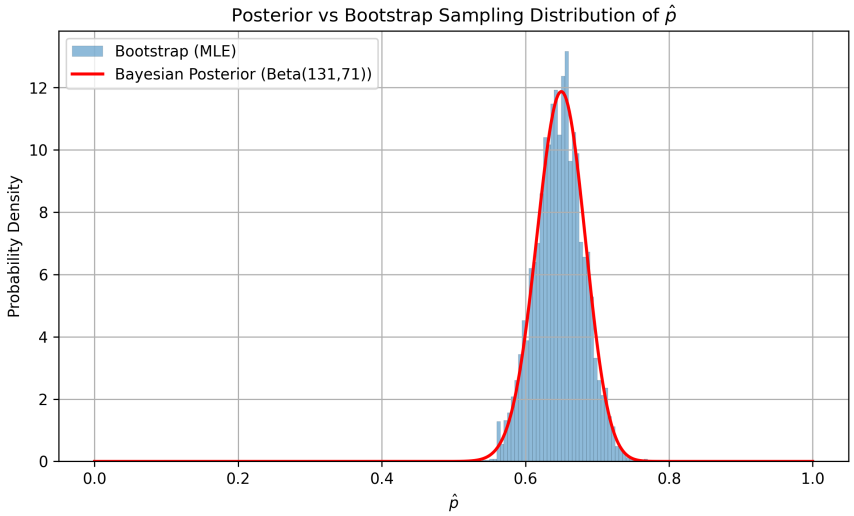
$$f_{P|X}(p \mid x) \sim \text{Beta}(131, 71)$$

Both Bayesian and Frequentist distribution is centered near 0.65, where question 4e showed:

$$\hat{p} = 0.65$$

This indicates strong agreement between both methods. The posterior is slightly smoother, while bootstrap shows some sampling variability.

Figure 1: Posterior vs Bootstrap



(h)

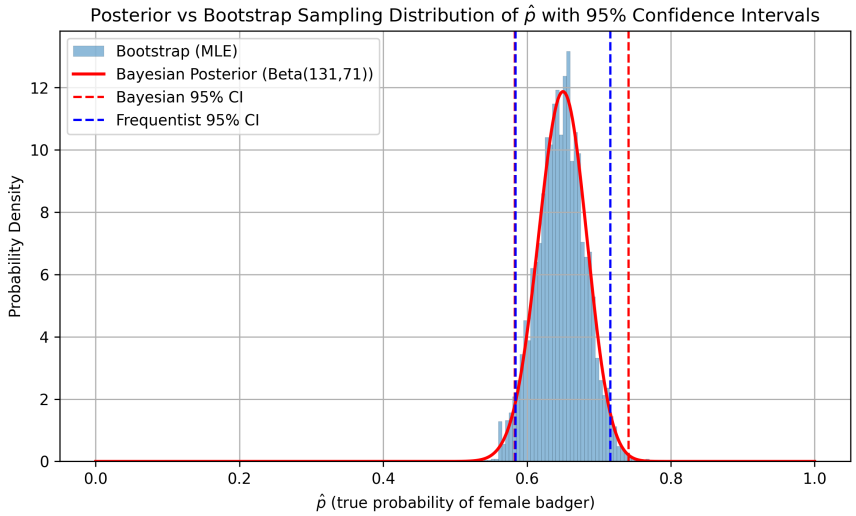
Confidence interval $= [0.584,\ 0.716]$,
Credible interval: Posterior $\text{Beta}(\alpha = 131, \beta = 71)$

Mean $= \frac{131}{202} = 0.6485$, **Variance** $= \frac{131 \times 71}{202^2 \times 203} \approx 0.001128$

Standard Error $= \sqrt{0.001128} = 0.0335$

95% CI $= 0.6485 \pm 1.96 \cdot 0.0335 = \boxed{[0.5864, 0.7161]}$

Figure 2: Posterior vs Bootstrap



Both intervals are similar, indicating consistency between frequentist and Bayesian methods.

Question 3

(a)

Let $y =$ range (m), $\quad x =$ firing angle (radians)

$$\text{RSS}(a) = \sum_{i=1}^{n} (y_i - a\sin(2x_i))^2$$

Outer function:

$$\frac{d}{da}\text{RSS} = 2(y_i - a\sin(2x_i))$$

Inner function:

$$\frac{d}{da}(y_i - a\sin(2x_i)) = -\sin(2x_i)$$

So,

$$\frac{d}{da}\text{RSS}(a) = \sum -2\sin(2x_i)\,(y_i - a\sin(2x_i))$$

Set derivative to 0:

$$\sum \sin(2x_i)\,(y_i - a\sin(2x_i)) = 0$$

$$\sum y_i \sin(2x_i) - a\sum \sin^2(2x_i) = 0$$

Solving for a:

$$\hat{a} = \frac{\sum_{i=1}^{n} y_i \sin(2x_i)}{\sum_{i=1}^{n} \sin^2(2x_i)}$$

(b)

k	x	y	$y \cdot \sin(2x)$	$\sin^2(2x)$
1	0.1	1.9	0.37747	0.03946
2	0.2	4.0	1.5576	0.15646
3	0.3	5.5	3.10553	0.31821

$$\sum y \sin(2x) = 5.0406, \quad \sum \sin^2(2x) = 0.50993$$

$$\boxed{\hat{a} = \frac{5.0406}{0.50993} = 9.89}$$

$$\boxed{\textbf{Maximum Range} = 9.89 \text{ meters}}$$

(c)

Model: $Y_i = a \sin(2x_i) + \varepsilon$

Assume:

$$f_{Y|X}(y|x) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2}(y - a\sin(2x_i))^2\right)$$

Log-likelihood:

$$\mathcal{L}(a, \sigma) = \prod_{i=1}^{n} \left(\frac{1}{\sqrt{2\pi}\sigma} \exp\left(-\frac{1}{2\sigma^2}(y_i - a\sin(2x_i))^2\right)\right)$$

$$= \left(\frac{1}{\sqrt{2\pi}\sigma}\right)^n \exp\left(-\frac{1}{2\sigma^2}\sum_{i=1}^{n}(y_i - a\sin(2x_i))^2\right)$$

$$= \frac{1}{\sigma^n(2\pi)^{n/2}} \exp\left(-\frac{\text{RSS}(a)}{2\sigma^2}\right)$$

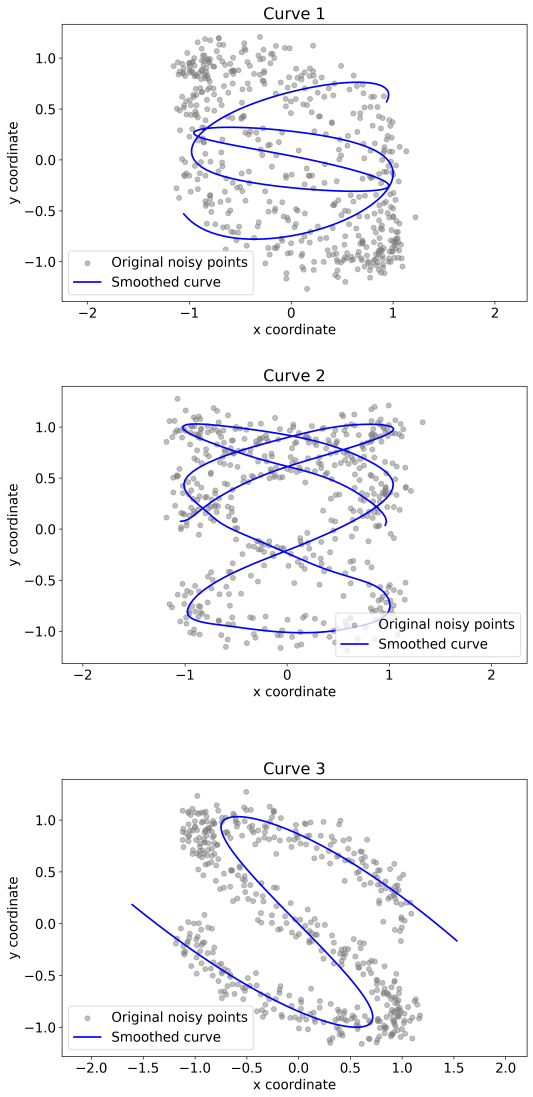
So,

$$\mathcal{L}(a, \sigma) = \frac{1}{\sigma^n(2\pi)^{n/2}} \exp\left(-\frac{\text{RSS}(a)}{2\sigma^2}\right)$$

Conclusion: Why MLE and Least Squares give the same $\hat{a}$:
When maximising $\mathcal{L}(a, \sigma)$ with respect to a, the constant and σ don't depend on a. The exponential term depends only on RSS(a). Therefore, maximising the likelihood is equivalent to minimising RSS and the MLE of a is the same as the least squares estimate of a.

Question 4a

Figure 3: Smoothing Curves



Model Building Explanation

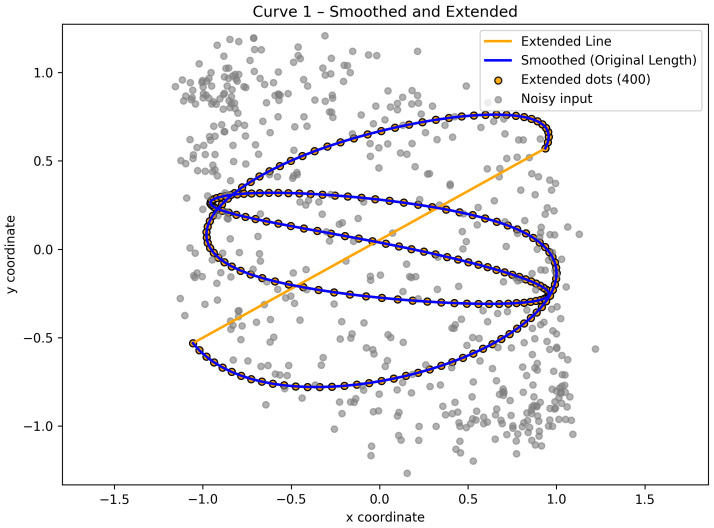
- From the SciPy library, I used spline smoothing to remove noise from Elizabeth's dots and produce smooth curves she was aiming for.
- Firstly, I measured how far along the curve each dot is by parameterising the points by 'arc length.'
- Then, separate smoothing splines were fitted to the x and y coordinates of each point using `UnivariateSpline`.
- I controlled the amount of smoothing using a parameter `s` by using 5-fold cross-validation (from `scikit-learn`) to automatically find the best value for `s`, based on minimising error rather than adjusting parameters manually by eye.
- Since Elizabeth's drawing was done by hand, and mine was produced automatically from noisy data, the smooth curve produced did not precisely match her target curve. The algorithm used optimised for smoothness rather than similarity to Elizabeth's drawing.

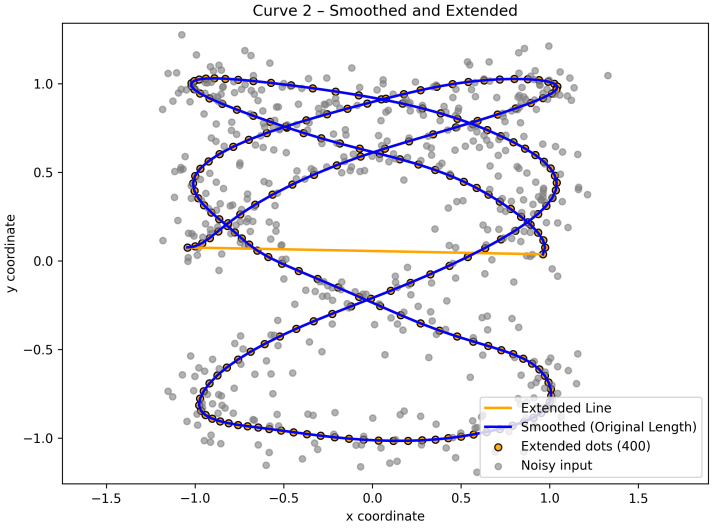
Pseudocode:

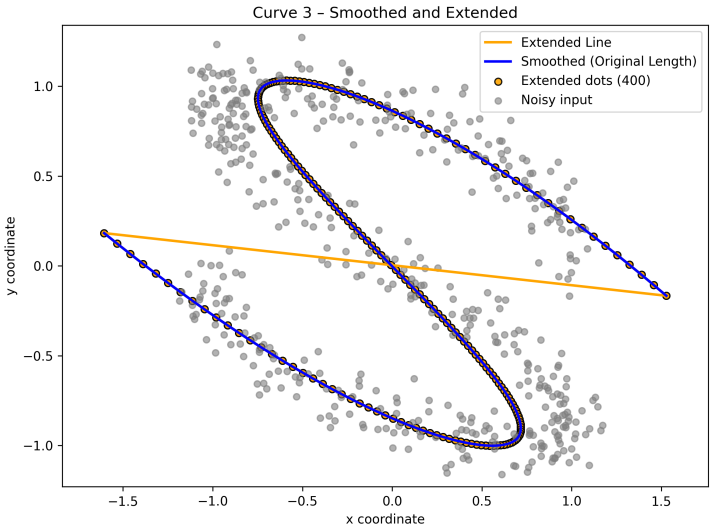
```
FUNCTION SmoothAndExtendDoubled(x, y, resolution = 200):
    // Step 1: Parameterise curve by the arc length
    t = cumulative sum of distances between (x, y) points
    Normalise t to range [0, 1]

    // Step 2: Smoothing parameters for x and y
    s_x = ComputeOptimalSmoothingParameter(t, x)
    s_y = ComputeOptimalSmoothingParameter(t, y)

    // Step 3: Fitting smoothing splines to x and y using t
    spline_x = FitSpline(t, x, smoothing = s_x)
    spline_y = FitSpline(t, y, smoothing = s_y)
```

Question 4b

Figure 4: Extended curves after smoothing







Extension Explanation

- I used the smoothed curve from Part (a), which had 200 points.
- Then, created another 200 points that continue the curve naturally, which then repeated the smoothed shape.
- This totalled to 400 smooth points, with the second set of 200 points being a continuation that follows the same pattern.
- The process is also fully automated, see pseudocode below.

```
FUNCTION SmoothAndExtendDoubled(x, y, resolution = 200):
  Step 4: Extend the parameter range to double the original
  t_extended = generate 2 * resolution values from 0 to 2

  Step 5: Wrap modulo around spline to simulate extension
  x_extended = EvaluateSpline(spline_x, t_extended mod 1)
  y_extended = EvaluateSpline(spline_y, t_extended mod 1)

  RETURN x_extended, y_extended, resolution
```

Question 5a

A multinomial logistic regression classifier was trained to predict Dave's emotional state from the measurements of his mouth shape. New, unseen data was produced by splitting the dataset into training (70%) and an unseen test set (30%). The model was evaluated on the unseen test set and had an overall accuracy of 67.30%.

Figure 5: Logistic Regression Results on Emotion Classification (Mouth Shape Only)

Class	Precision	Recall	F1-score	% Correct	% Incorrect
0 (happy)	0.73	0.71	0.72	71.3%	28.7%
1 (sad)	0.73	0.89	0.81	89.2%	10.8%
2 (stimulated)	0.51	0.40	0.45	40.2%	59.8%

Table 1: Confusion Matrix (rows = true class, columns = predicted class)

True \ Predicted	0	1	2
0 (happy)	72	1	28
1 (sad)	1	91	10
2 (stimulated)	26	32	39

With high precision and recall, the logistic regression model performed well on 'happy' and 'sad' classes, with 'sad' classifications slightly outperforming 'happy' classes. The 'stimulated' class was often mistaken for the other two classes, showing significantly worse accuracy; reflected in the confusion matrix with many 'stimulated' classes incorrectly classified. Overall, mouth shape features help distinguish "happy" and "sad", but are less effective for "stimulated".

Question 5b

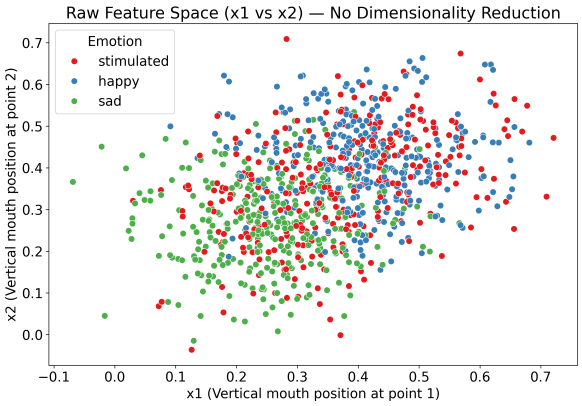


Figure 6: Before PCA

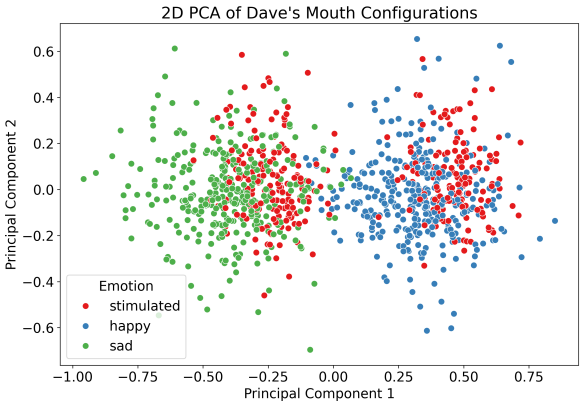


Figure 7: After PCA

The first scatter plot represents high dimensional data (100D) using the first 2 raw features (x1 and x2), reflecting how noisy the data is. Principal Component Analysis (PCA) was used to reduce this high dimensional data (100D) to 2 dimensions (2D). Before PCA there was very little separation between emotions. After PCA, 'happy' and 'sad' are more distinguishable, while 'stimulated'

overlaps both 'happy' and 'sad,' supporting earlier findings that the 'stimulated' emotion is harder to classify.

Question 5c

K-Means clustering was applied to the PCA reduced data to group mouth configurations into 3 clusters by minimising variance within clusters. This iteratively assigned points to nearest centroids, updating them until they converged. Although the Elbow Method (Figure 9) shows sharpest drop in inertia when k=2, after k=3, improvements flatten, suggesting 3 clusters is a reasonable cutoff point. Since our dataset consists of 3 emotion labels aswell, a k-value of 3 was selected.

Figure 8: K-Means Clustering

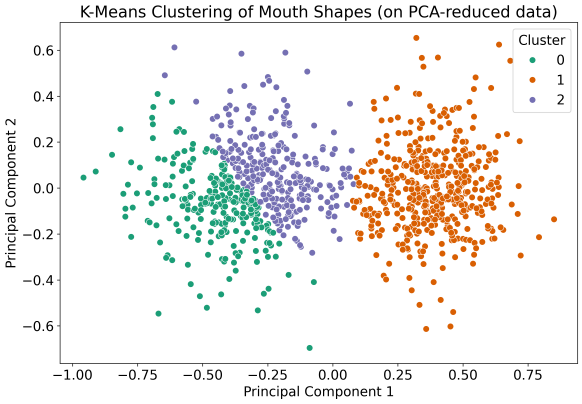


Figure 9: Elbow Method

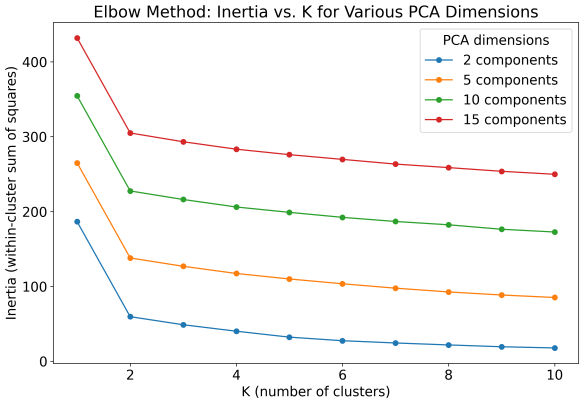


Figure 8 shows clusters are visually distinct in 2D PCA space. Some clusters (0 and 1) align with specific emotions. Cluster 2 shows some overlap. This K-Means model suggests some emotions are separable, but others are not well-isolated.

Table 2: Crosstab of Predicted Clusters vs. True Emotions

Cluster	Happy	Sad	Stimulated
0	0	220	37
1	313	0	148
2	25	120	137

Crosstab was applied to identify each cluster's corresponding emotion label, see Table 2. Cluster 0 largely contains the 'sad' class, representing the 'sad' emotion, Cluster 1 corresponds to the 'happy' class consisting most 'happy' instances, while Cluster 2 is more mixed, with significant numbers from both 'stimulated' and 'sad,' suggesting the stimulated emotions overlaps largely with the 'sad' emotion. These results, indicate Dave's 'happy' and 'sad' emotional state formed distinct clusters, while 'stimulated' is less well-separated, matching previous findings in both PCA and classification performance.