CS 5785: APPLIED MACHINE LEARNING

HOMEWORK 3

November 11, 2017

Sarah Le Cam - sdl83
Yunie Mao - ym224

Cornell Tech

Contents

SENTIMENT ANALYSIS FOR ONLINE REVIEWS

Question 1 (a)

We imported NumPy to parse each file and generate the data and labels. The labels are balanced across of the three files. In each file, there are 500 negative and 500 positive labels. In total, there are 1500 negative and 1500 positive labels.

Question 1 (b)

Since the dataset consist of online reviews that may contain noises and garbage, we performed the following transformations on the original dataset.

- **Transformed all words into lowercase:** This makes it easier to find the set of matching words in dataset.
- **Stripped all punctuations:** Punctuations are noises that disrupts the word matching and should be removed from the dataset.
- **Stripped all stopwords such as "and", "or" and "the":** Stopwords add minimal value and disrupt the word matching.
- **Lemmatized all words to convert plural nouns to singular, past tenses to present and comparative and superlative tenses to positive:** Imported WordNetLemmatizer from the nltk.stem module to lemmatize words in each sentence in the data. This helps to generalize the words which improves the matching accuracy

Question 1 (c)

For each file, we split the data into training and testing sets by taking the first 400 positive and negative samples as the training data and the remainder as the testing data. In total, our training set contained 1200 positive samples and 1200 negative samples. Our testing set contained 300 positive samples and 300 negative samples.

Question 1 (d)

In order to extract features, we used the training set to build a dictionary of all unique words in the reviews. We did not use the testing set because using it to select features could lead to overfitting of data. Using our dictionary, we built a feature vector for each review in the

training data, with its i^{th} index as the occurrences of the i^{th} dictionary word in the review. We then added each feature vector to a training feature matrix. We also created a testing feature matrix using the same method performed for the training feature.

In total, there are 4356 features represented by the feature vector. We randomly chose two reviews from the training set and reported their feature vectors.
"wow love place" → [1,1,1,0,0,...]
"crust not good" → [0,0,0,1,1,1,0,...]

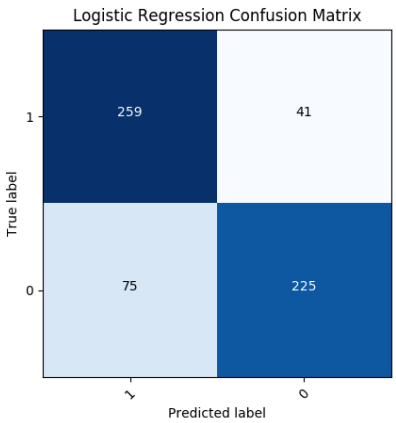
Question 1 (e)

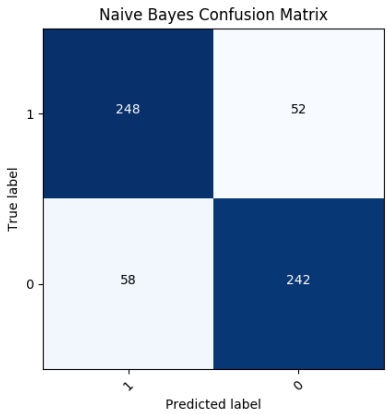
Because majority of all words across reviews do not appear in each individual review, most of the feature vector elements will be 0. In addition, the sentences are not of the same length; long sentences will yield higher frequencies of certain words. In order to handle the huge variance and to reduce the influence of the high frequency words in the feature vector, we applied l2 normalization as the postprocessing strategy. Since l2 minimizes sum of the square of the differences between the target value and the estimate values, applying l2 adjusts our model to handle the sparsity of our feature vectors and provides us a much more stable solution.

Question 1 (f)

To perform sentiment predictions, we trained a logistic regression model on the training data and fitted it on the testing data using Scikit-learn?s Logistic Regression package. We reported the classification accuracy score and plotted the confusion matrix. Based on the weight vector of our model, we displayed the top 10 words that play the most important roles in deciding the sentiment of the reviews. In addition, we trained a Multinomial Naive Bayes classifier using Scikit-learn?s built-in package and performed a similar analysis. Here, we compare the performance of the Logistic Regression to that of the Multinomial Naive Bayes in predicting sentiments.

Model	Accuracy Score	Confusion Matrix	Top 10 Most Significant Words
Logistic Regression	0.806666666667	$\begin{bmatrix} 259 & 41 \\ 75 & 225 \end{bmatrix}$	['great', 'bad', 'love', 'excellent', 'nice', 'delicious', 'poor', 'amaze', 'best', 'fantastic']
Multinomial Naive Bayes	0.816666666667	$\begin{bmatrix} 248 & 52 \\ 58 & 242 \end{bmatrix}$	['thereplacement', 'desperately', 'ironically', 'drawback', 'clearly', 'support', 'pander', 'sabotage', 'tech', 'certain']



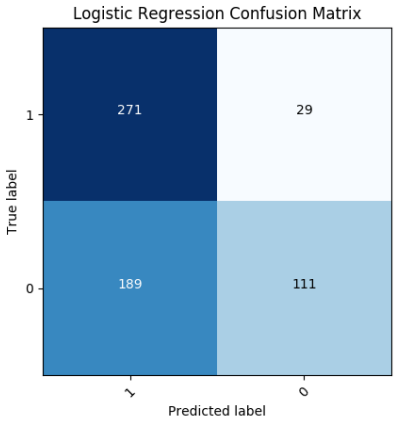


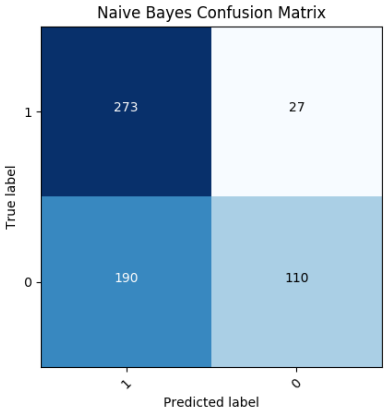
Question 1 (g)

Similar to the bag of words model, we constructed a dictionary of n-grams, contiguous sequences of words with n=2. We did not perform any postprocessing to normalize the training set as we had done with the bag of words. In this case, since we are looking at sequences of 2 words, any frequent words would not necessarily be frequent sequences, so there is not a strong need to reduce the effect of these frequent words. We then performed

the training and testing with Logistic Regression and the Multinomial Naive Bayes models and reported the accuracy, the confusion matrix, and the top 10 most frequent sequences of 2-grams.

Model	Accuracy Score	Confusion Matrix	Top 10 Most Significant Words
Logistic Regression	0.636666666667	$\begin{bmatrix} 271 & 29 \\ 189 & 111 \end{bmatrix}$	['work great', 'highly recommend', 'waste time', 'one best', 'great phone', 'great product', 'waste money', 'food good', 'really good', 'easy use']
Multinomial Naive Bayes	0.638333333333	$\begin{bmatrix} 273 & 27 \\ 190 & 110 \end{bmatrix}$	['get pair', 'awful muffle', 'first wear', 'wash machine', 'disbelief dish', 'really overprice', 'way fit', 'drop face', 'smell disgust', 'feel angry']





Question 1 (h)

Since the features in the bag of words model contain large redundancy, we implemented PCA to reduce the dimensions of the features to 10, 50 and 100 respectively. To perform PCA, we implemented the following:

- Computed the means of the feature data using np.mean
- Adjusted the feature data by subtracting its means
- Using Numpy's linalg svd function, computed the unitary matrix V
- Computed the dot product of the feature data and the nth rank of the the conjugate transpose of V

Using our PCA, we repeated the sentiment predictions using bag of words and n-grams to construct dictionaries and trained with Logistic Regression and Naive Bayes models to fit our test data.

Bag of Words

	PCA-10 Bow	PCA-50 Bow	PCA-100 Bow
LR	0.491666666667 [148 152] [153 147] ['love', 'good', 'late', 'nasty', 'tasty', 'place', 'crust', 'texture', 'wow', 'stop']	0.48 [108 192] [120 180] ['love', 'good', 'late', 'nasty', 'tasty', 'place', 'crust', 'texture', 'wow', 'stop']	0.496666666667 [99 201] [101 199] ['love', 'good', 'late', 'nasty', 'tasty', 'place', 'crust', 'texture', 'wow', 'stop']
NB	0.541666666667 [162 138] [137 163] ['good', 'nasty', 'place', 'late', 'texture', 'stop', 'crust', 'tasty', 'love', 'wow']	0.511666666667 [117 183] [110 190] ['late', 'good', 'nasty', 'stop', 'texture', 'place', 'crust', 'tasty', 'wow', 'love']	0.521666666667 [126 174] [113 187] ['late', 'nasty', 'tasty', 'stop', 'texture', 'good', 'place', 'crust', 'wow', 'love']

N-Gram

	PCA-10 2-Grams	PCA-50 2-Grams	PCA-100 2-Grams
LR	0.493333333333 [[6 294] [10 290]] ['exterior display', 'good wornout', 'char outside', '20the cover', 'deal include', 'say place', 'bother slow', 'unfold 18th', 'need see', '']	0.496666666667 [[287 13] [289 11]] ['exterior display', 'good wornout', 'char outside', '20the cover', 'deal include', 'say place', 'bother slow', 'unfold 18th', 'need see', '']	0.495 [[8 292] [11 289]] ['exterior display', 'good wornout', 'char outside', '20the cover', 'deal include', 'say place', 'bother slow', 'unfold 18th', 'need see', '']
NB	0.49 [[10 290] [16 284]] ['unfold 18th', '20the cover', 'say place', 'need see', '', 'char outside', 'bother slow', 'deal include', 'good wornout', 'exterior display']	0.501666666667 [[293 7] [292 8]] ['unfold 18th', '20the cover', 'char outside', 'need see', 'good wornout', 'bother slow', 'say place', '', 'deal include', 'exterior display']	0.485 [[275 25] [284 16]] ['unfold 18th', '20the cover', 'char outside', 'need see', 'good wornout', 'bother slow', 'say place', '', 'deal include', 'exterior display']

Question 1 (i)

We compared the performances of bag of words, 2-gram, and PCA for bag of words by examining the classification results, accuracy scores, and the weights learned from Logistic Regression and Naive Bayes training.

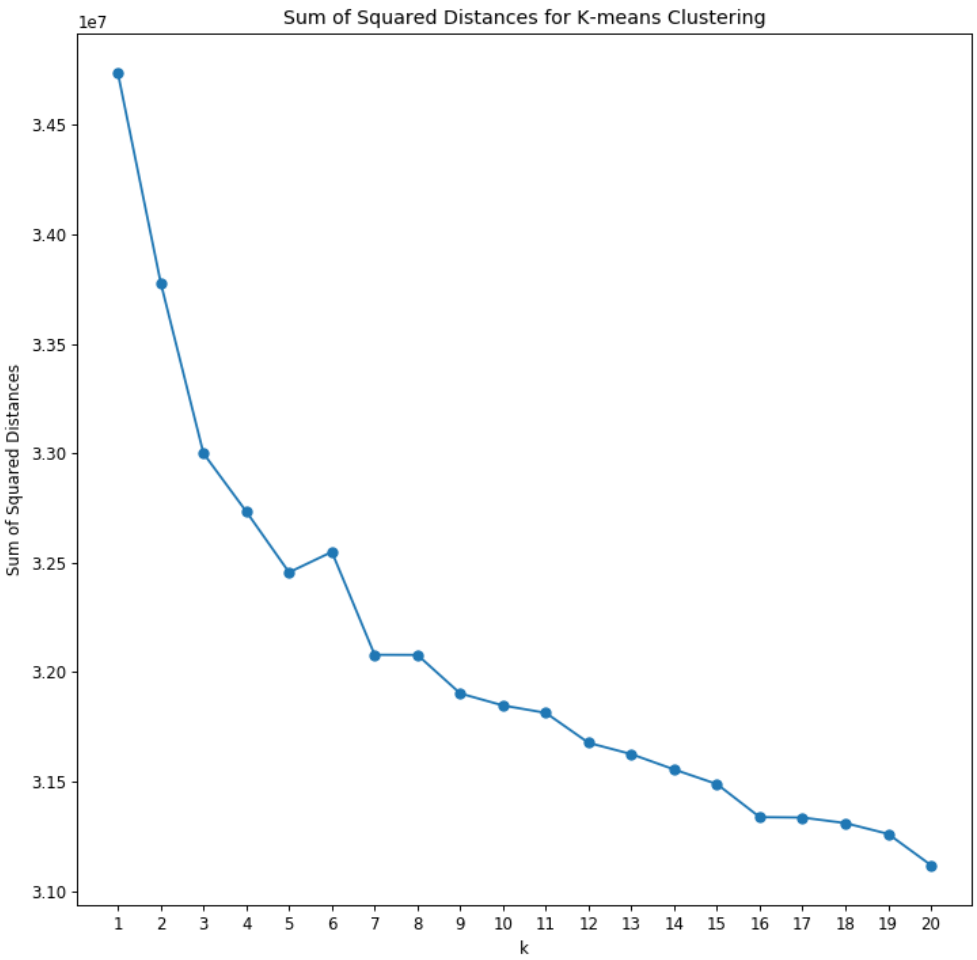
In general, people tend to use same set of words when providing reviews online, regardless of the services or product provided. By examining the set of most popular words used in user reviews, we can determine whether a given review expresses positive or negative sentiment.

Bag of Words	N-gram	PCA for Bag of Words	PCA for N-gram
Best performance in terms of accuracy. It had the least number of misclassifications. The Multinomial Naive Bayes model performed slightly better than the Logistic Regression model.	Performed worse than Bag of Words. 2-gram introduced unnecessary features that added more sparsity and created biases in the training data.	Performed worse than without using PCA. PCA reduces the dimensions which ignores the less significant words. This creates biases in the features and results in higher misclassification rates.	Worst performance in terms of accuracy. Using PCA to reduce to 10, 50 and 100 dimensions produced similar poor results. In order to attain better performance, the dimensions need to be increased.

CLUSTERING FOR TEXT ANALYSIS

Question 2 (a)

We first downloaded the science2k-doc-word.npy data using the NumPy library. We then ran K-Means for $k = 1, 2, ..., 20$ and found the sum of the squared distances for each k. We plotted the sum of the sum of the squared distances vs. the values of k. This allowed us to use the Elbow Method to identify a good k. The idea of the Elbow Method is to choose a value for k with a low sum of squared distances while managing complexity. The elbow represents where we start to have diminishing returns by increasing k. For this model, it seems that $k = 8$ is a good value.



We then found the top 10 documents of each cluster with the lowest distance for the cluster mean and printed out the tittles of the documents:

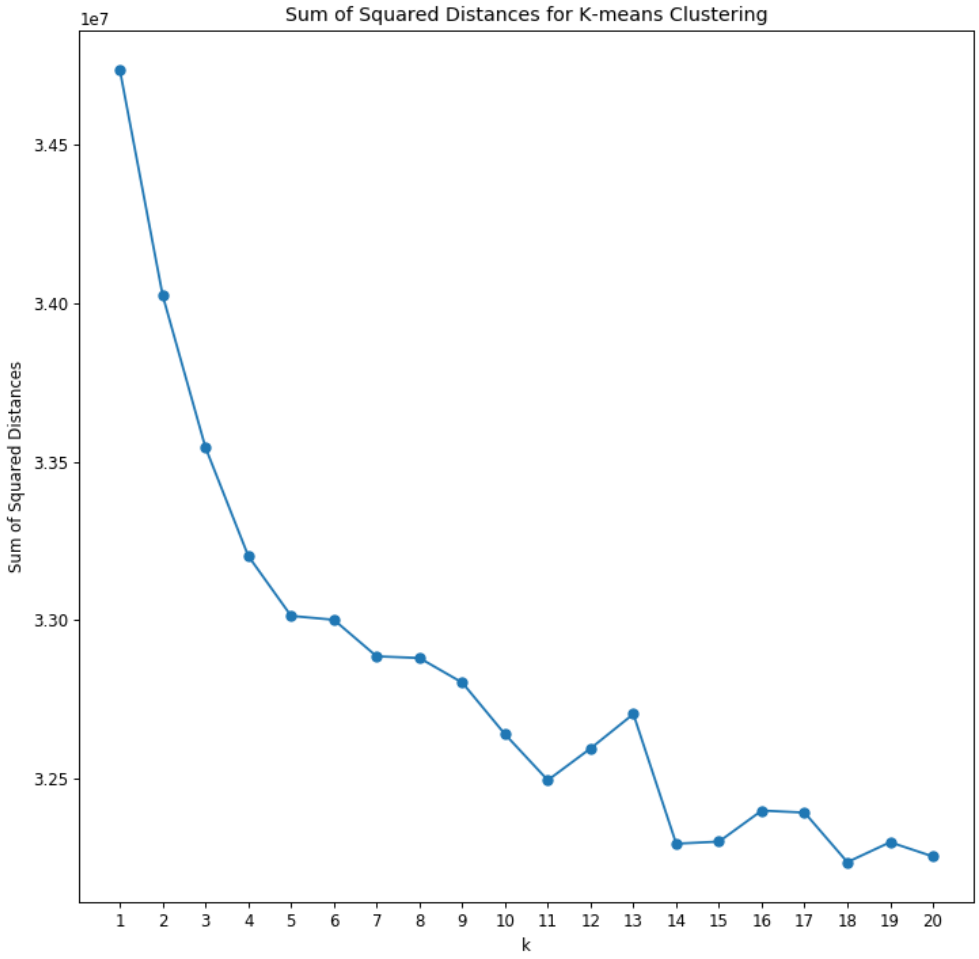
1: ['"Temperatures without Fevers?"', '"The Global Spread of Malaria in a Future, Warmer World"', '"Infectious History"'], 2: ['"Information Technology Takes a Different Tack"', '"Science Survives in Breakthrough States"', '"Vaccine Studies Stymied by Shortage of Animals"', '"The Violence of the Lambs"', '"Flushing out Nasty Viruses in the Balkans"', '"For Ґatheróf Abortion Drug, Vindication at Last"', '"New Brain Cells Prompt New Theory of Depression"', '"On a Slippery Slope to Mediocrity?"', '"Plants Join the Genome Sequencing Bandwagon"', '"In Europe, Hooligans Are Prime Subjects for Research"'], 3: ['"Suppression of Mutations in Mitochondrial DNA by tRNAs Imported from the Cytoplasm"', '"Distinct Classes of Yeast Promoters Revealed by Differential TAF Recruitment"', '"Efficient Initiation of HCV RNA Replication in Cell Culture"', '"T Cell-Independent Rescue of B Lymphocytes from Peripheral Immune Tolerance"', '"Reduced Food Intake and Body Weight in Mice Treated with Fatty Acid Synthase Inhibitors"', '"Patterning of the Zebrafish Retina by a Wave of Sonic Hedgehog Activity"', '"Coupling of Stress in the ER to Activation of JNK Protein Kinases by Transmembrane Protein Kinase IRE1"', '"An Anti-Apoptotic Role for the p53 Family Member, p73, during Developmental Neuron Death"', '"Disruption of Signaling by Yersinia Effector YopJ, a Ubiquitin-like Protein Protease"', '"Identification of Synergistic Signals Initiating Inner Ear Development"'], 4: ['"Thermal, Catalytic, Regiospecific Functionalization of Alkanes"', '"Influences of Dietary Uptake and Reactive Sulfides on Metal Bioavailability from Aquatic Sediments"', '"Clues from a Shocked Meteorite"', '"Homogenization of Fish Faunas across the United States"', '"Neutral, Single-Component Nickel (II) Polyolefin Catalysts That Tolerate Heteroatoms"', '"Is Bigger Better in Cricket?"', '"Into the Forbidden Zone"', '"How to Get along: Friendly Microbes in a Hostile World"', '"The Formation of Chondrules at High Gas Pressures in the Solar Nebula"', '"Information Storage and Retrieval through Quantum Phase"'], 5: ['"Reopening the Darkest Chapter in German Science"', '"Algorithmic Gladiators Vie for Digital Glory"', '"National Academy of Sciences Elects New Members"', '"Corrections and Clarifications: A Short Fe-Fe Distance in Peroxodiferric Ferritin: Control of Fe Substrate versus Cofactor Decay?"', '"Corrections and Clarifications: Charonś First Detailed Spectra Hold Many Surprises"', '"Corrections and Clarifications: Unearthing Monuments of the Yarmukians"', '"Corrections and Clarifications: Marking Time for a Kingdom"', '"Corrections and Clarifications: Ґaster, Cheaper, Betterón Trial"', '"Corrections and Clarifications: Close Encounters: Details Veto Depth from Shadows"', '"Corrections and Clarifications: A Nuclear Solution to Climatic Change?"'], 6: ['"Structure of Yeast Poly(A) Polymerase Alone

and in Complex with 3ᐟdATP"', '"Structure of Murine CTLA-4 and Its Role in Modulating T Cell Responsiveness"', '"Structure of the S15,S6,S18-rRNA Complex: Assembly of the 30S Ribosome Central Domain"', '"Atomic Structure of PDE4: Insights into Phosphodiesterase Mechanism and Specificity"', '"The Productive Conformation of Arachidonic Acid Bound to Prostaglandin Synthase"', '"Twists in Catalysis: Alternating Conformations of Escherichia coli Thioredoxin Reductase"', '"Redox Signaling in Chloroplasts: Cleavage of Disulfides by an Iron-Sulfur Cluster"', '"Convergent Solutions to Binding at a Protein-Protein Interface"', '"Structure of the Protease Domain of Memapsin 2 (b-Secretase) Complexed with Inhibitor"', '"Structure and Function of a Human <latex>TAF_{II}250</latex> Double Bromodomain Module"'], 7: ['"High-Gain Harmonic-Generation Free-Electron Laser"', '"Anomalous Polarization Profiles in Sunspots: Possible Origin of Umbral Flashes"', '"A Light-Emitting Field-Effect Transistor"', '"Discovery of a High-Energy Gamma-Ray-Emitting Persistent Microquasar"', '"Spontaneous Ordering of Oxide Nanostructures"', '"Dispersive Multiplexing in Multimode Optical Fiber"', '"Triple Vortex Ring Structure in Superfluid Helium II"', '"Tunable Resistance of a Carbon Nanotube-Graphite Interface"', '"Three-Layered Atmospheric Structure in Accretion Disks around Stellar-Mass Black Holes"', '"Direct Observation of Dynamical Heterogeneities in Colloidal Hard-Sphere Suspensions"'], 8: ['"Reconstruction of the Amazon Basin Effective Moisture Availability over the past 14,000 Years"', '"Greenland Ice Sheet: High-Elevation Balance and Peripheral Thinning"', '"Isotopic Evidence for Variations in the Marine Calcium Cycle over the Cenozoic"', '"Mass Balance of the Greenland Ice Sheet at High Elevations"', '"Rapid Kimberlite Ascent and the Significance of Ar-Ar Ages in Xenolith Phlogopites"', '"Glacial Climate Instability"', '"Variable Carbon Sinks"', '"The Role of the Southern Ocean in Uptake and Storage of Anthropogenic Carbon Dioxide"', '"Remobilization in the Cratonic Lithosphere Recorded in Polycrystalline Diamond"', '"Temporal Trends in Deep Ocean Redfield Ratios"']

The most prominent documents in each cluster (those with the smallest distance to the mean) seem to have similar topics. For example, in cluster 8 (featured above) the top 10 documents are all related to oceans and water. Then, an algorithm such as this could be used to find articles related to a certain topic or with a similar area of interest as another document. A scientific search engine could use this to display potential further research and interesting document links.

Question 2 (b)

We downloaded the science2k-word-doc.npy data using the NumPy library. Again, we ran K-Means for $k = 1, 2, ..., 20$ and found the sum of the squared distances for each k, which we plotted. We then used the Elbow Method to identify a good k and chose $k = 8$.



We then found the top 10 terms of each cluster with the lowest distance for the cluster mean and printed out the tittles of the documents:

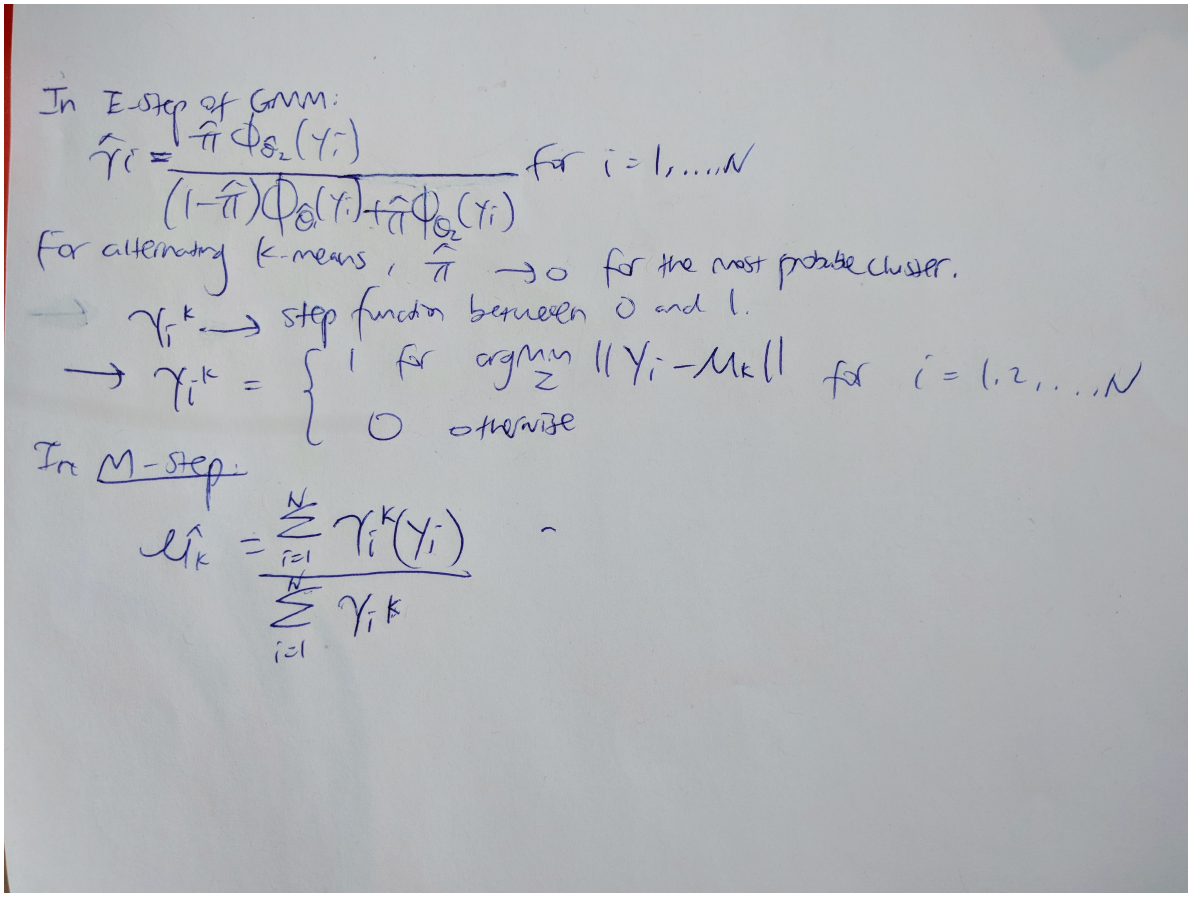
1: [’org’, ’sciencemag’, ’vol’, ’lacz’, ’bcl’, ’introns’, ’myc’, ’elisa’, ’p21’, ’cdnas’], 2: [’biochem’, ’terminus’, ’cooh’, ’nh2’, ’cdna’, ’inhibitor’, ’incubated’, ’affinity’, ’blot’, ’specificity’], 3: [’recalls’,

'clinton', 'security', 'fight', 'prize', 'spending', 'campaign', 'hes', 'rights', 'pay'], 4: ['excitations', 'coherence', 'resonant', 'anisotropic', 'electrostatic', 'disordered', 'fermi', 'anisotropy', 'doped', 'orientations'], 5: ['parameters', 'start', 'estimate', 'volume', 'gray', 'decrease', 'estimates', 'eds', 'extent', 'error'], 6: ['lcts', 'aptamers', 'trxr', 'dnag', 'neas', 'proteorhodopsin', 'doxy', 'rory', 'lg268', 'nompc'], 7: ['case']

The most prominent terms in each cluster (those with the smallest distance to the mean) seem to have similar themes. For example, in cluster 5 (featured above) the top 10 terms are nearly all related to measurements. Then, an algorithm such as this could be used to find related terms within a certain topic or terms related to another given term. A scientific search engine could use this to display potential further research in terms of other topics or to help searchers better direct their searches.

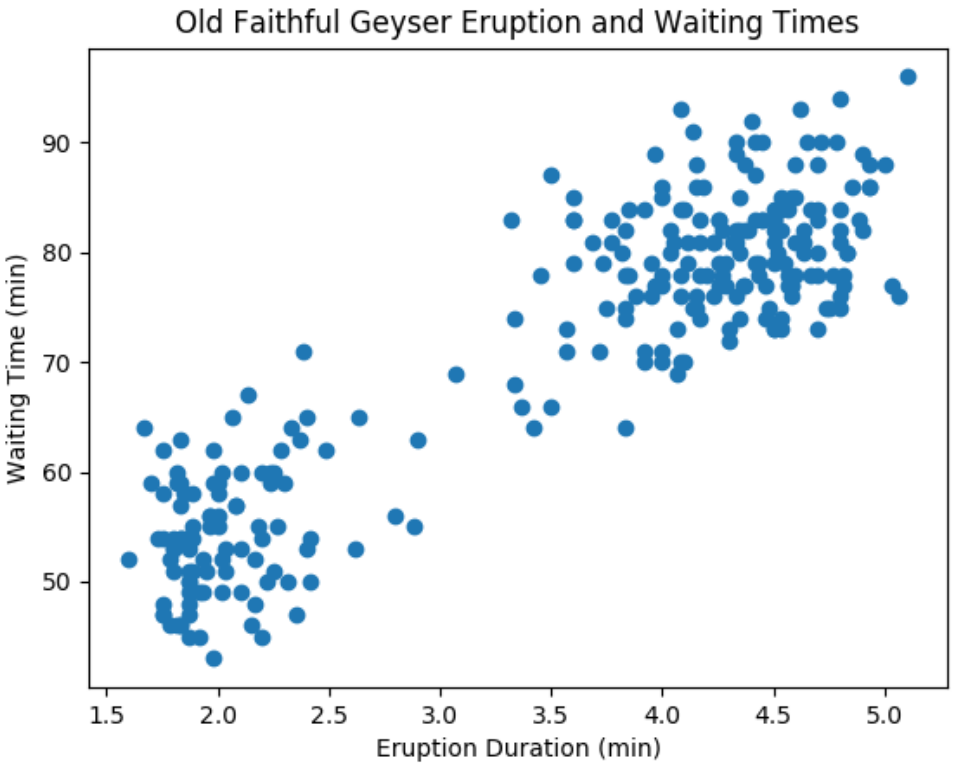
EM ALGORITHM AND IMPLEMENTATION

Question 3 (a)



Question 3 (b)

We downloaded the Old Faithful Geyser dataset that contained 272 samples of the geyser eruption duration and the waiting time between the eruptions. Using Numpy?s loadtxt function, we parsed the data and plotted the points on a 2-D plane.



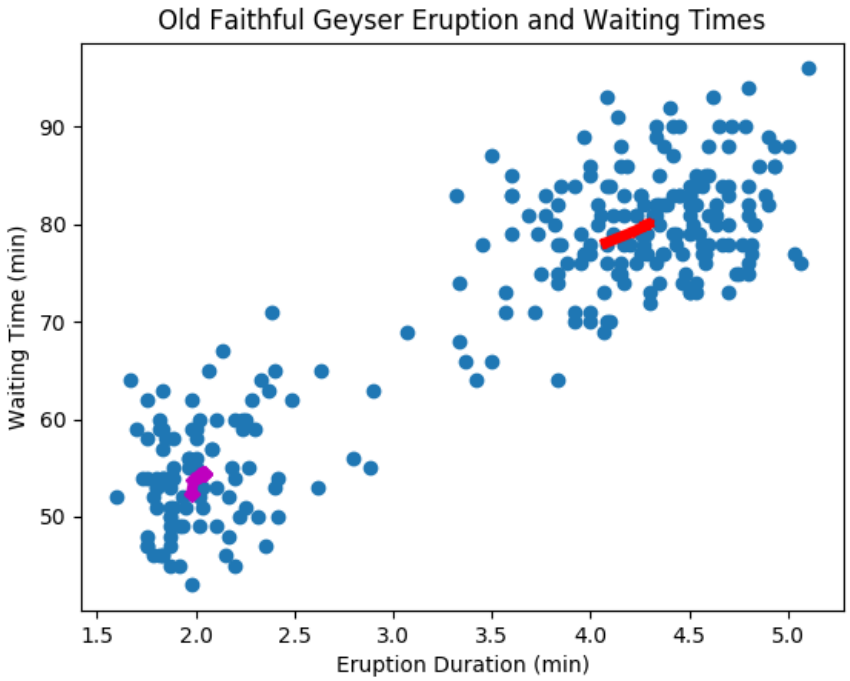
Question 3 (c)

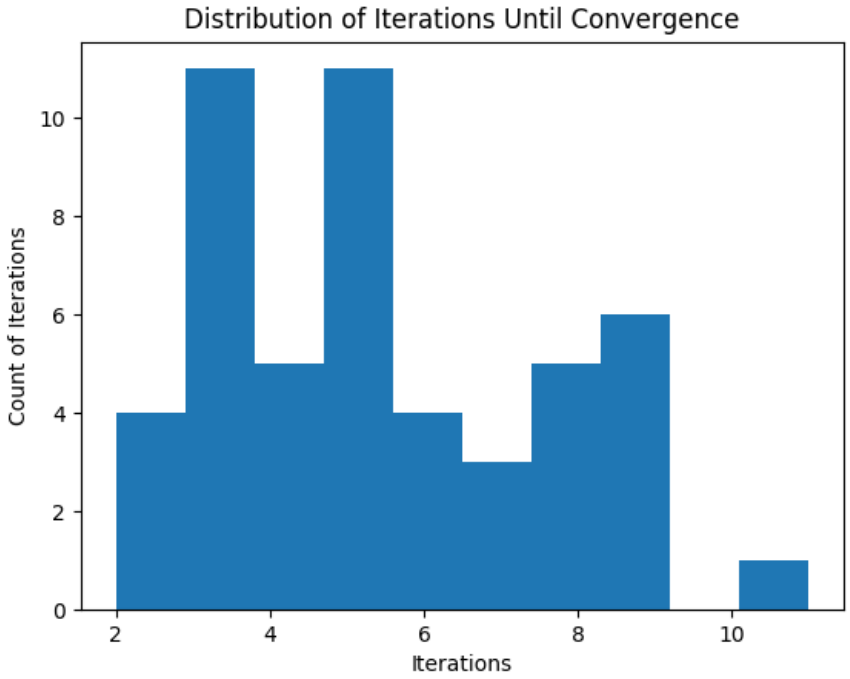
In order to fit all the data points using the EM algorithm, we implemented a Gaussian-Mixture Model by performing the following steps:

- Initialized the sample mean matrix by randomly selecting 2 sets of points from the sample data. Initialized the covariance matrix to a spherical identity matrix. Initialized the priors with equal weights assigned to each cluster.
- Performed the expectation step by iterating over n data sets and computing the responsibilities of all the samples.
- Using the responsibilities from the expectation step, performed the maximization step by iterating over n data sets and updating the mean, covariance and prior matrices.
- At each iteration, computed the log likelihood as a function of the responsibilities and kept track of the deltas between consecutive log likelihood values.
- Set a threshold (0.001) for convergence and iterate until the delta of the log likelihood

is smaller than that threshold value. Recorded the iteration count at the convergence step.

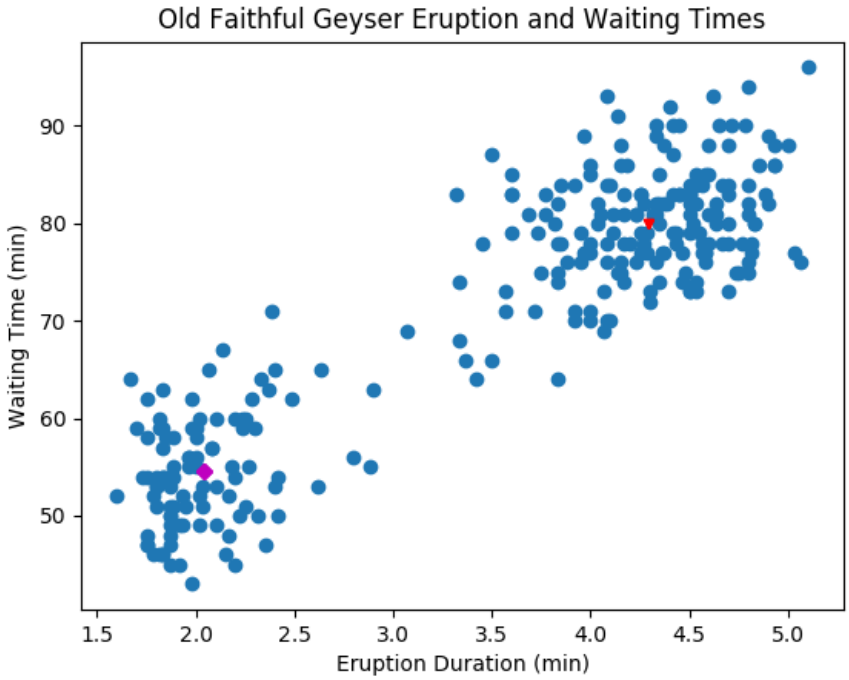
We plotted the 2 dimensional trajectories of the mean vectors in each of the clusters. In addition, we ran our GMM program 50 times with different randomly initialized parameters. The below plot shows the distribution of the total number of iterations needed for our algorithm to reach convergence.

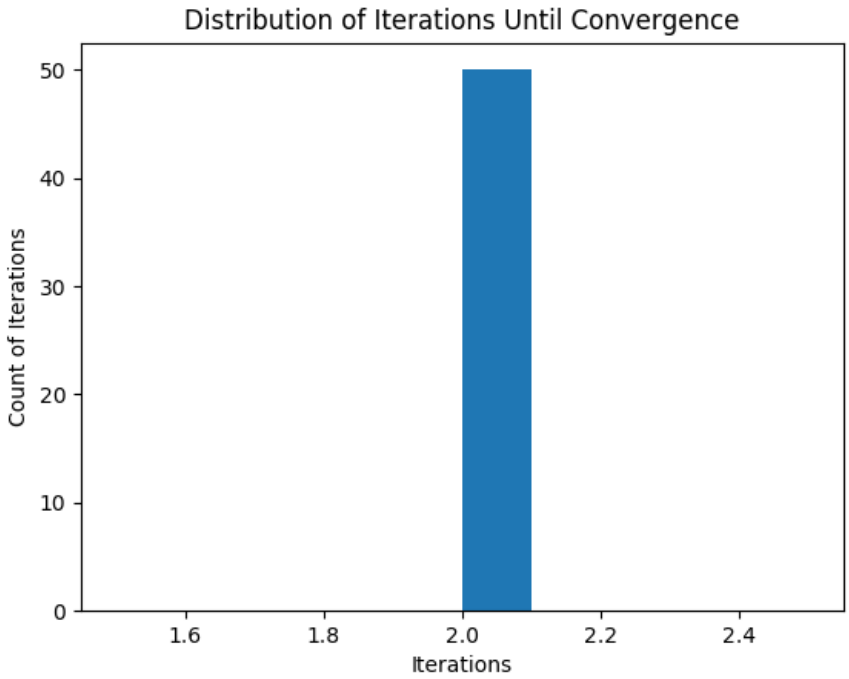




Question 3 (d)

Using Scikit-learn's KMeans clustering algorithm with K = 2, we labeled each data point to one of the two clusters. Using the maximum likelihood over the labeled data points, we initialized the mean and the covariance matrices. Similar to the procedure in 3c, we performed the expectation and maximization steps, plotted the 2-dimensional mean vectors and the distribution of the iterations until convergence.





Comparing the performance of GMM with random initialization to the one with KMeans initialization, we find that convergence is accelerated by parameter initialization using KMeans.

The histograms showed that convergence happened after only 2 iterations using KMeans initialization and after 5-7 iterations using the random initialization method. In addition, the plots showing the trajectories of the mean vectors also support the observation that KMeans initialization reduces the number of iterations for convergence.

WRITTEN EXERCISES

Question 1

Ex. 14.2 Consider a mixture model density in p-dimensional feature space,

$$g(x) = \sum_{k=1}^{K} \pi_k g_k(x), \tag{14.114}$$

where $g_k = N(\mu_k, \mathbf{L} \cdot \sigma^2)$ and $\pi_k \geq 0 \, \forall k$ with $\sum_k \pi_k = 1$. Here $\{\mu_k, \pi_k\}, k = 1, \ldots, K$ and σ^2 are unknown parameters.

Suppose we have data $x_1, x_2, \ldots, x_N \sim g(x)$ and we wish to fit the mixture model.

1. Write down the log-likelihood of the data
2. Derive an EM algorithm for computing the maximum likelihood estimates (see Section 8.1).
3. Show that if σ has a known value in the mixture model and we take $\sigma \to 0$, then in a sense this EM algorithm coincides with K-means clustering.

Question 1 (a)

We are given:

$$g(x) = \sum_{k=1}^{K} \pi_k \cdot g_k(x)$$

Therefore, for each data point x_i:

$$g(x_i) = \sum_{k=1}^{K} \pi_k \cdot g_k(x_i)$$

Then, the likelihood of the whole dataset is:

$$\text{likelihood} = \prod_{i=1}^{N} g(x_i)$$

$$\begin{aligned} \text{Log-likelihood} &= log(\prod_{n=1}^{N} g(x_i)) \\ &= \sum_{i=1}^{N} log(g(x_i)) \\ &= \sum_{i=1}^{N} log(\sum_{k=1}^{K} \pi_k \cdot g_k(x_i)) \end{aligned} \tag{1}$$

Question 1 (b)

[INITIALISATION] First, for each cluster, take initial guesses for the values of the mean $\hat{\mu}_k$, the variance $\hat{\sigma}_k^2$, and the prior $\hat{\pi}_k$, $k \in \{1, 2, ..., K\}$.

[E-STEP] Compute the responsibility of the model, $\hat{\gamma}_i$, for each datapoint, x_i, $c \in \{1, 2, ..., K\}$:

$$\gamma_i(c) = \frac{\pi_c \cdot g_c(x_i)}{\sum_{k=1}^{K} \pi_k \cdot g_k(x_i)}$$

[M-STEP] Compute the new weighted means and variances:

$$\mu_c = \frac{\sum_{i=1}^{N} \gamma_i(c) \cdot x_i}{\sum_{i=1}^{N} \gamma_i(c)}$$

$$\sigma_c^2 = \frac{\sum_{i=1}^{N} \gamma_i(c) \cdot (x_i - \mu_c)^2}{\sum_{i=1}^{N} \gamma_i(c)}$$

$$\pi_c = \frac{\sum_{i=1}^{N} \gamma_i(c)}{N}$$

[REPEAT] Repeat the E-Step and M-Step until convergence.

Question 1 (c)

We would like to show that if $\sigma \to 0$, this EM algorithm coincides with K-Means clustering. If $\sigma \to 0$, then, in the E-Step, the responsibility of the model will be:

$$\gamma_i(c) = \begin{cases} 1, & \text{if } x_i \text{ is classified as } c \\ 0, & \text{otherwise} \end{cases}$$

Then, in the M-Step, the means will only be updated for the data points in a given class:

$$\mu_c = \frac{\sum_{i=1}^{N} \gamma_i(c) \cdot x_i}{\sum_{i=1}^{N} \gamma_i(c)} \text{ if } x_i \in \text{ c}$$

r This is equivalent to the K-Means algorithm.

Question 2

Ex. 14.11 *Classical multidimensional scaling.* Let $\mathbf{S}$ be the centered inner product matrix with elements $\langle x_i - \bar{x}, x_j - \bar{x} \rangle$. Let $\lambda_1 > \lambda_2 > \cdots > \lambda_k$ be the k largest eigenvalues of $\mathbf{S}$, with associated eigenvectors $\mathbf{E}_k = (\mathbf{e}_1, \mathbf{e}_2, \ldots, \mathbf{e}_k)$. Let $\mathbf{D}_k$ be a diagonal matrix with diagonal entries $\sqrt{\lambda_1}$, $\sqrt{\lambda_2}$, $\ldots$, $\sqrt{\lambda_k}$. Show that the solutions z_i to the classical scaling problem (14.100) are the *rows* of $\mathbf{E}_k \mathbf{D}_k$.

$$S_C(z_1, z_2, \ldots, z_N) = \sum_{i,i'} (s_{ii'} - \langle z_i - \bar{z}, z_{i'} - \bar{z} \rangle)^2 \qquad (14.100)$$

We have S, the centered inner product matrix, such that:

$$s_{ij} = < x_i - \bar{x}, x_j - \bar{x} >$$

Let x_i $(i = 1, \ldots, n)$ be the i$^{\text{th}}$ of n data point in p dimensional Euclidean space such that $x_i = (x_{i1}, \ldots, x_{ip})$. Let $B = (x_i - \bar{x}, \ldots, x_n - \bar{x})$, the matrix of centered points. We can then denote $S = BB^T$. In order to minimize 14.100, we want to find Z such that $S = ZZ^T \iff BB^T = ZZ^T \iff B = Z$.

$$\text{rank}(S) = \text{rank}(BB^T) = \text{rank}(B) = p$$

Since S is symmetric and positive semidefinite and of rank p, it must have p non-negative eigenvalues and $n - p$ zero eigenvalues. Then S can be written as it's Singular Value Decomposition:

$$S = V \Lambda V^T$$

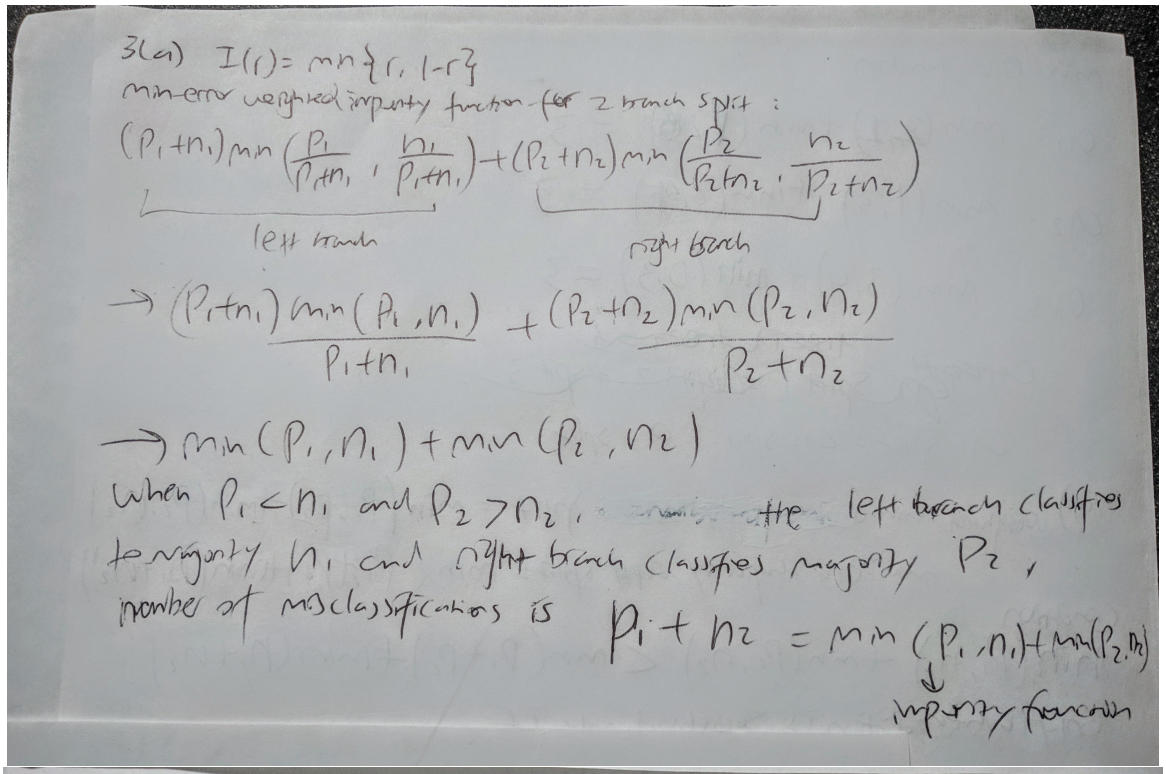
$$\Lambda = \begin{bmatrix} \lambda_1 & 0 & \cdots & 0 \\ 0 & \lambda_2 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & \cdots & \cdots & \lambda_k \end{bmatrix} = D_k^2$$

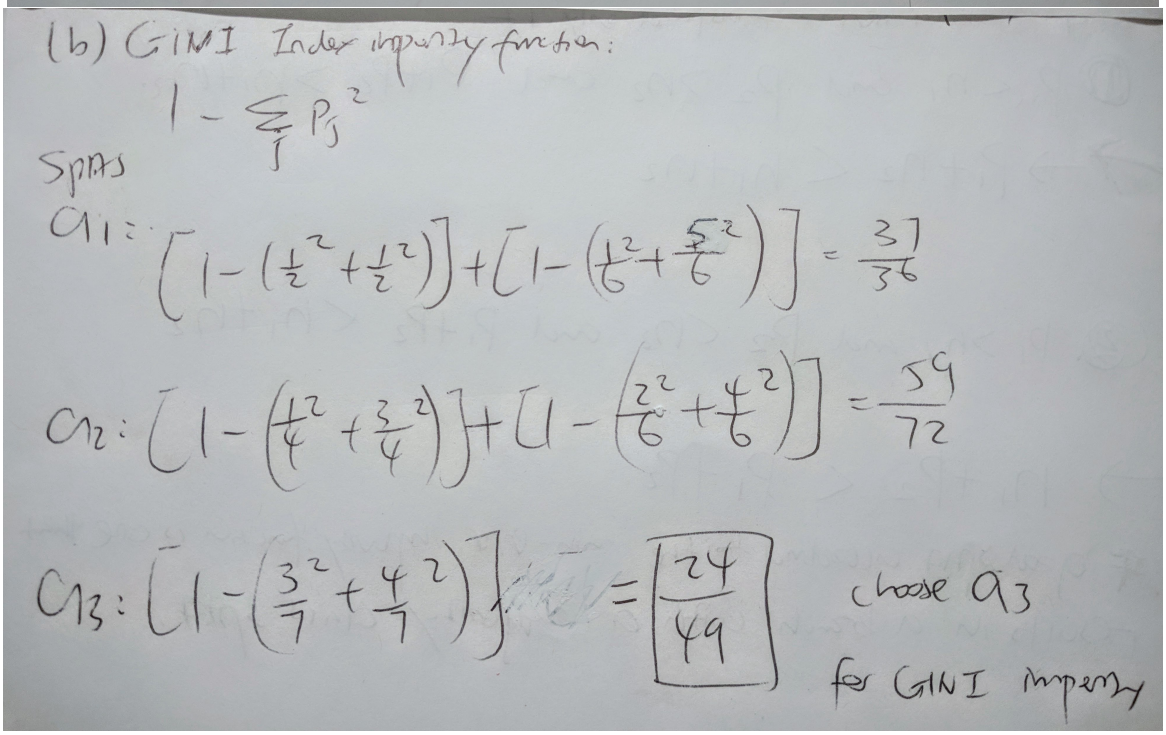
$$V = E_k$$

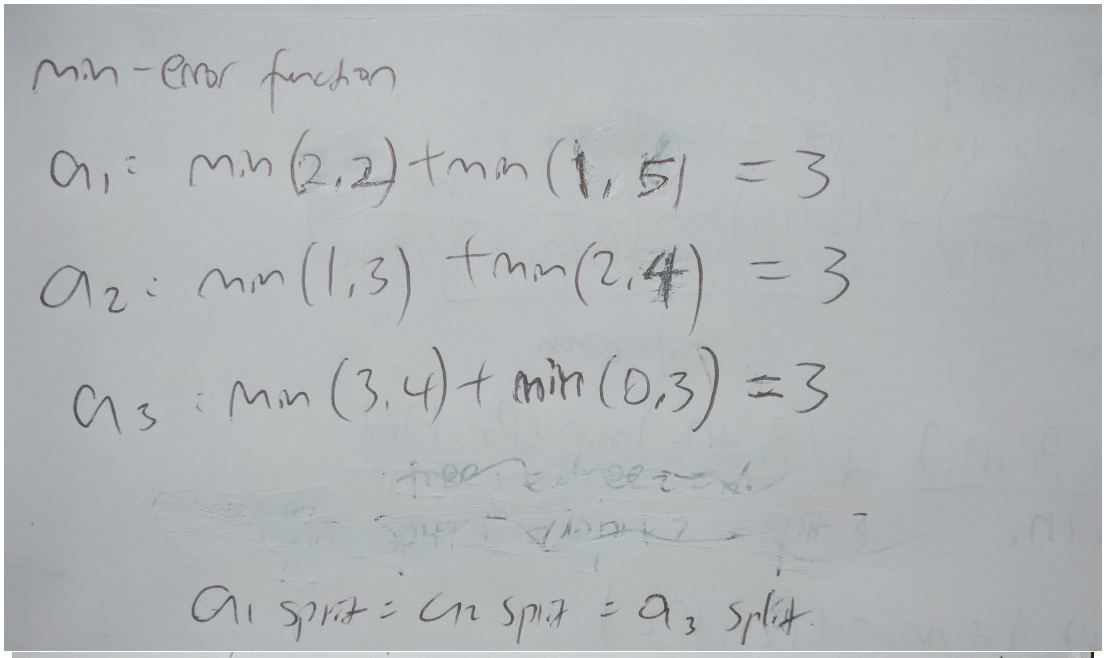
$$
\begin{aligned}
S &= V \Lambda V^T \\
&= E_k D_k^2 E_k^T \\
&= (E_k \cdot D_k) \cdot (D_k \cdot E_k^T) \\
&= (E_k \cdot D_k) \cdot (D_k^T \cdot E_k^T) \\
&= (E_k \cdot D_k) \cdot (E_k \cdot D_k)^T
\end{aligned} \tag{2}
$$

Therefore, $B = E_k \cdot D_k$ and the optimal solution $Z = E_k \cdot D_k$. Hence, the solutions z_i to the classical scaling problem are the rows of $E_k \cdot D_k$.

Question 3


3(a) I(r) = min{r, 1-r}
min-error weighted impurity function for 2 branch split :
left branch
right branch
when P1 < n1 and P2 > n2, the left branch classifies to majority n1 and right branch classifies majority P2,
number of misclassifications is
impurity function


(b) GINI Index impurity function:
Splits
choose a3
for GINI impurity



(c) weighted min-error impurity before split = $\min(P_1+P_2, n_1+n_2)$

min-error impurity after split = $\min(P_1, n_1) + \min(P_2, n_2)$

$\min(P_1, n_1) + \min(P_2, n_2) < \min(P_1+P_2, n_1+n_2)$

This condition is satisfied only if

① $P_1 < n_1$ and $P_2 > n_2$ and $P_1+P_2 > n_1+n_2$

$\rightarrow P_1+n_2 < n_1+n_2$

② $P_1 > n_1$ and $P_2 < n_2$ and $P_1+P_2 < n_1+n_2$

$\rightarrow n_1+P_2 < P_1+P_2$

A good split according to the min-error impurity function is one that results in a branch with a majority class split.

Question 3 (d)

While the min-error impurity function performs well for class separation, it is unsuitable for growing a decision tree. Because it assumes that a branch split with a differing majority class is a good split, it tends to ignore cases where the misclassified samples make up a small portion of the branch size. We see this case in part (b) with feature a3, where one branch had no misclassification. In this case, the min-error function treats this as a good split and does not continue to grow the tree.

SOURCES & EXTERNAL LIBRARIES

Stéfan van der Walt, S. Chris Colbert and Gaël Varoquaux. *The NumPy Array: A Structure for Efficient Numerical Computation*, Computing in Science & Engineering, 13, 22-30 (2011), DOI:10.1109/MCSE.2011.37

John D. Hunter. *Matplotlib: A 2D Graphics Environment*, Computing in Science & Engineering, 9, 90-95 (2007), DOI:10.1109/MCSE.2007.55

Jones E, Oliphant E, Peterson P, et al. *SciPy: Open Source Scientific Tools for Python*, 2001-, `http://www.scipy.org/`

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, Édouard Duchesnay. *Scikit-learn: Machine Learning in Python*, Journal of Machine Learning Research, 12, 2825-2830 (2011)

Wes McKinney. *Data Structures for Statistical Computing in Python*, Proceedings of the 9th Python in Science Conference, 51-56 (2010)

Kotzias et. al,. *From Group to Individual Labels using Deep Features*, KDD 2015

Hardle, W. (1991) *Smoothing Techniques with Implementation in S.*, New York: Springer.

Azzalini, A. and Bowman, A. W. (1990). *A look at some data on the Old Faithful geyser*. Applied Statistics 39, 357-365.

"Metric Multidimensional Scaling." *15.2 Metric Multidimensional Scaling*, sfb649.wiwi.hu-berlin.de/fedc_homepage/xplore/tutorials/mvahtmlnode99.html.