Question 1:

- **Overview:**

The dataset has 3476 rows and 8 columns.

- **Histograms:**
  - (a)

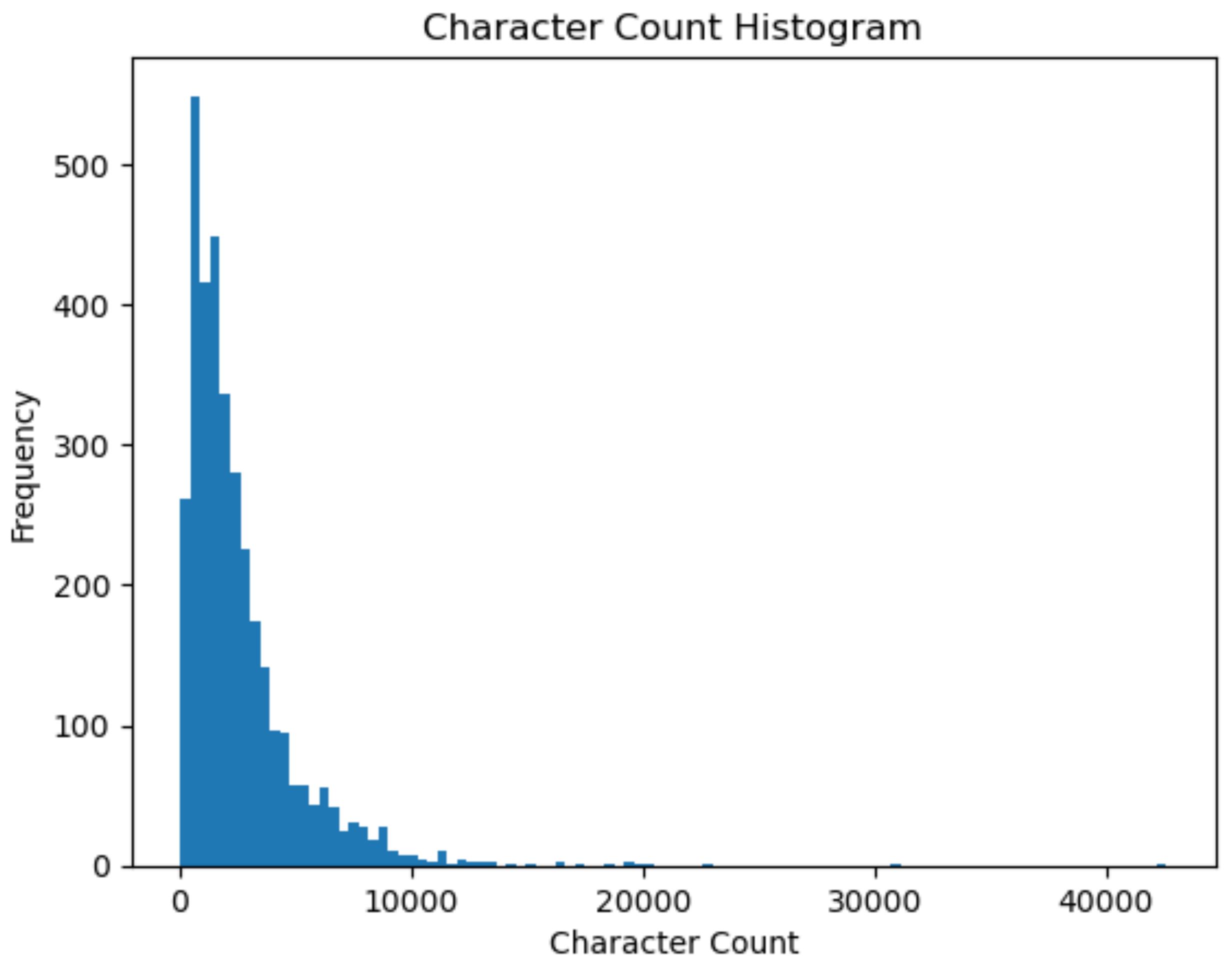


Figure 1: Frequency of Count of Alpha-numeric Characters per Row in Feature full_text

- (b)

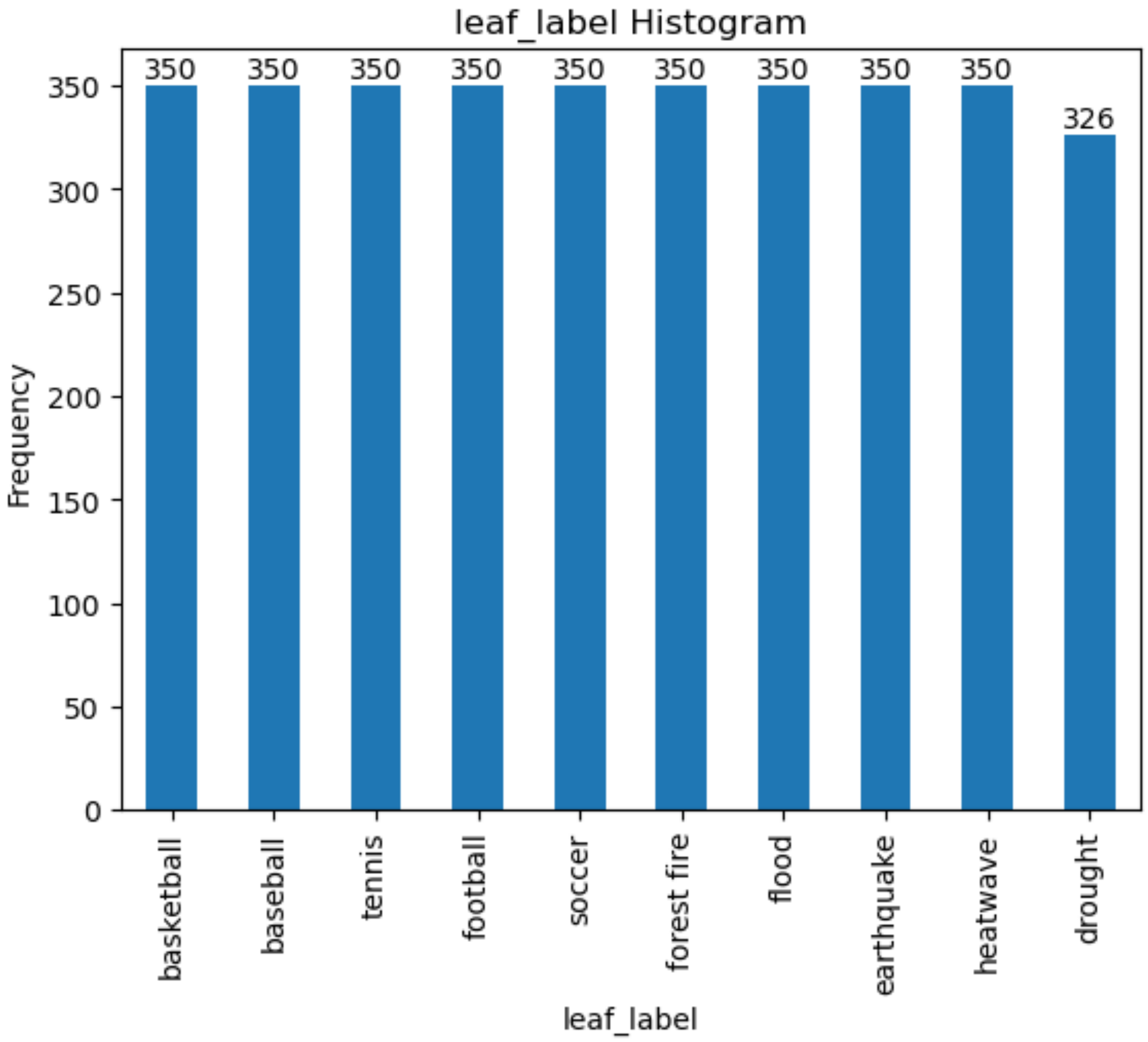


Figure 2: Frequency of leaf_label Class

- (c)

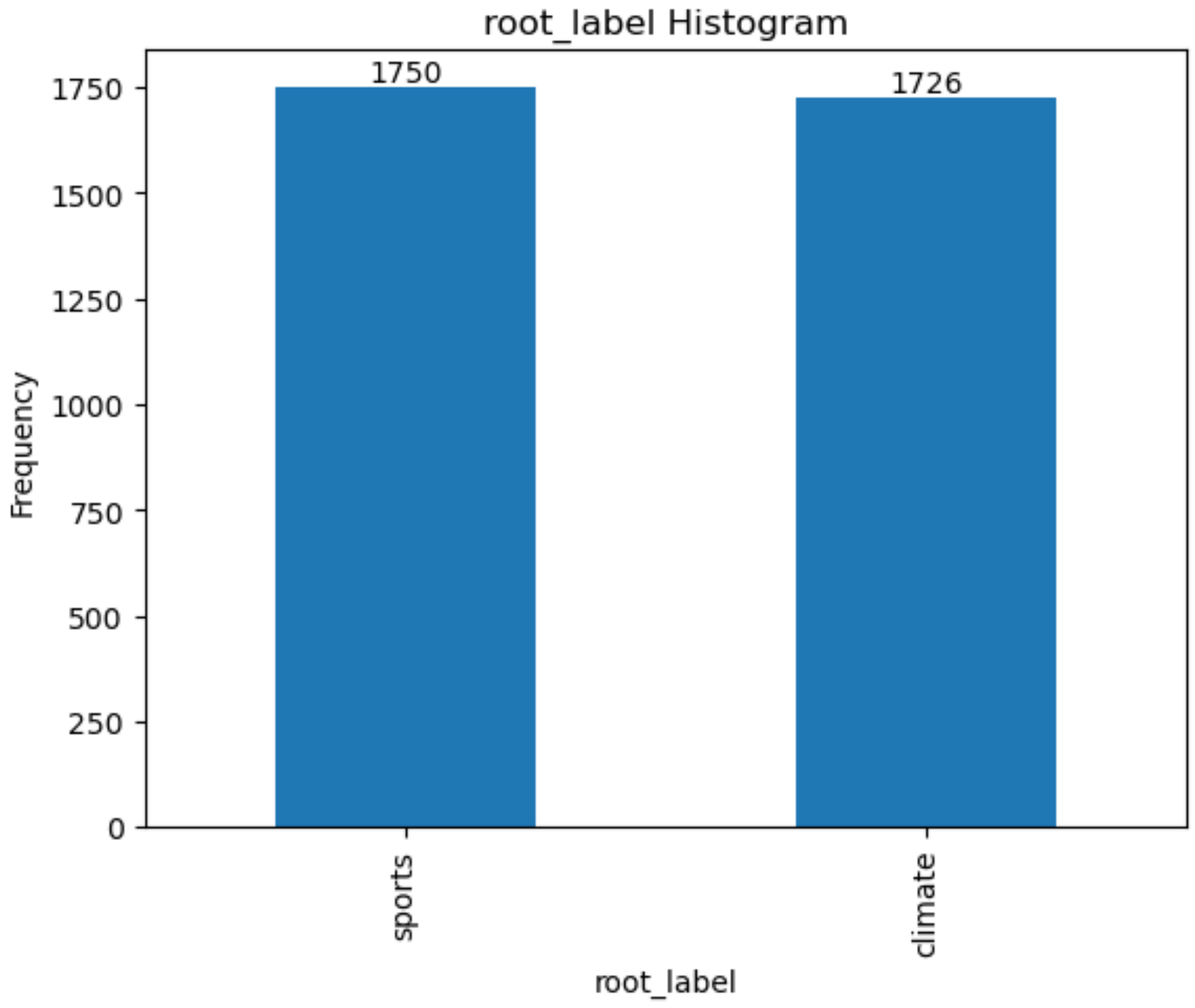


Figure 3: Frequency of root_label Class

- **Interpret plots:**

As we can see from figure 1, it is right-skewed. The majority of alpha-numeric character count per line in feature full_text lies between 0 and 5000. This implies most rows have short sentences, like summary. However, some data points have much longer sentences, which may require special techniques when preprocessing to avoid issues. The distribution helps us decide preprocessing methods and know where we need to pay special attentions to.

From figure 2, we can see that the dataset is mostly balances, with class “drought” slightly fewer than others. This difference can be ignored as it is unlikely to affect the training process. This can ensure the model will not be trained to favor certain labels, which is likely to happen when using unbalanced datasets.

From figure 3, we can conclude that the dataset is nearly perfect balanced on a higher-level class. This will benefit the hierarchical classification tasks. We need to maintain this balanced structure when splitting the dataset into training set and testing set for a better performance.

Question 2:

There are 2780 rows of training samples and 696 rows of testing samples.

Question 3:

- **Pros and cons:**

Lemmatization is more accurate since the result is a real word. It preserves contextual meaning and will lead to a smaller dictionary size. Stemming is faster than lemmatization while it is less accurate. It can result in a non-existent word. The dictionary size is a little bit larger.

- **Affect of min_df:**

Decreasing the value of min_df will include more feature words in the dictionary. This may lead to a higher risk of overfitting due to the rare words. Increasing the value of min_df will exclude more words, focussing on more common ones. This may lose some important information, but improve the model generalization.

- **Removing stopwords before or after lemmatizing:**

We should remove the stop words after the lemmatization because it reduces the words into their base form, which will benefit the stopwords removal. However, the numbers should be removed before lemmatization because numbers usually do not have meaningful lemmas.

- **Shape of TF-IDF-processed train and test matrices:**

The training matrix has 2780 rows and 13874 columns.
The test matrix has 696 rows and 13874 columns.

Question 4:

- **What does the plot look like? What does the plot's concavity suggest?**

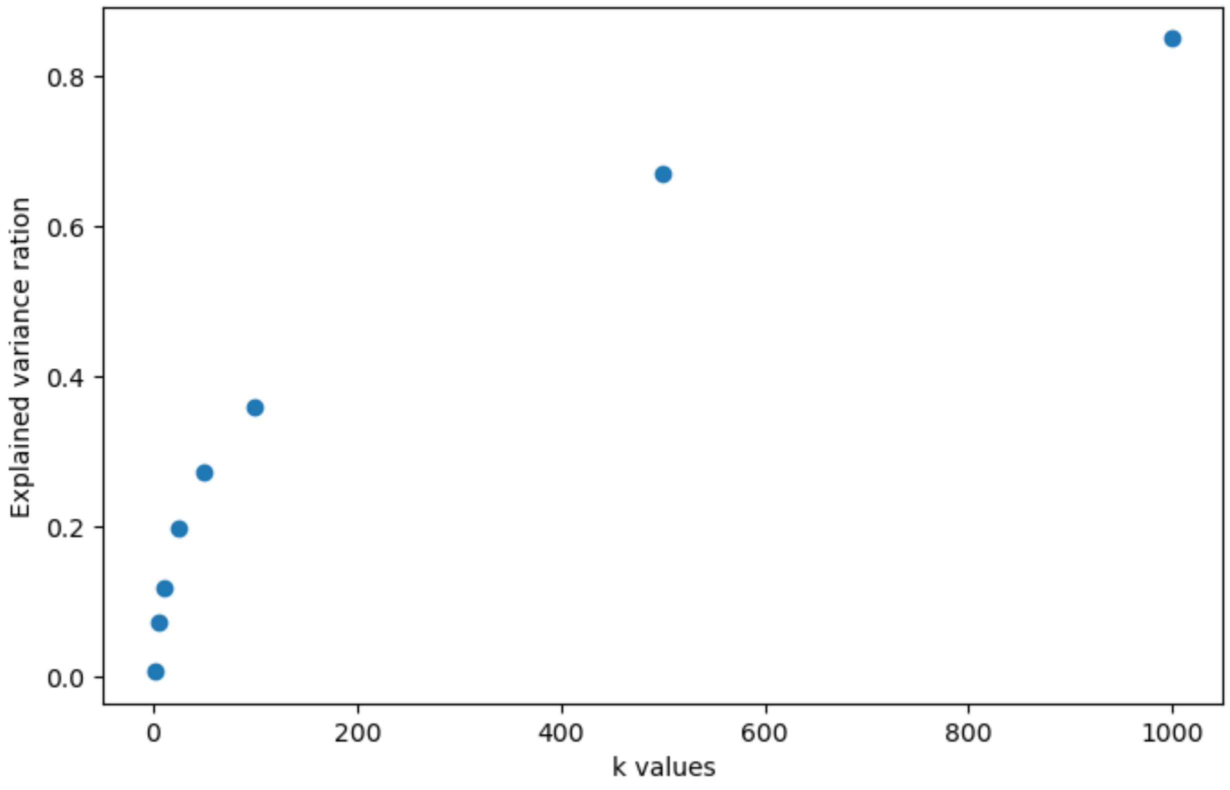


Figure 4: Explained Variance Ratio across Multiple Different k

The plot looks like an asymptotic line, where the variance increase rapidly at the beginning and slow down with increasing k value. This suggest that most of the significant patterns, most variance, in the data are captured by the first few components, and later components capture the less significant patterns.

- **Which reconstruction residual MSE error is larger and why?**

The MSE error of NMF is larger, which is 2172.5907 with k =25, while the error of LSI is 2158.6332. This is because LSI allows non-negative values, which provides more flexibility for TF_IDF matrix, where there are a lot of zero values.

Question 5:

- **Evaluation scores of Gemma = 2000, 0.0005, and 100000. Performance analysis.**

Gemma = 2000:

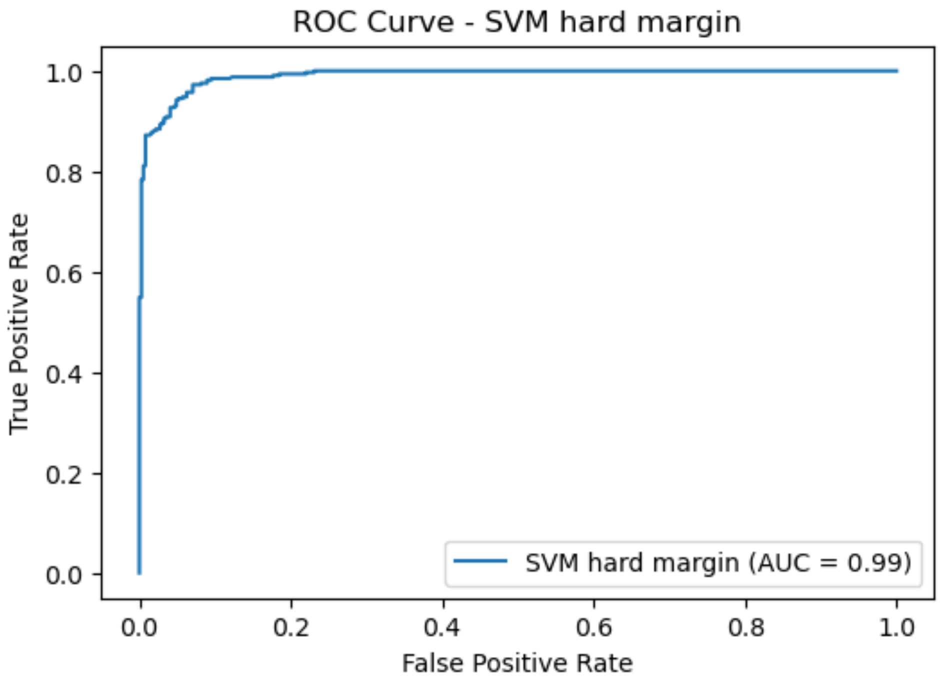


Figure 5: ROC Curve of Hard Margin SVM

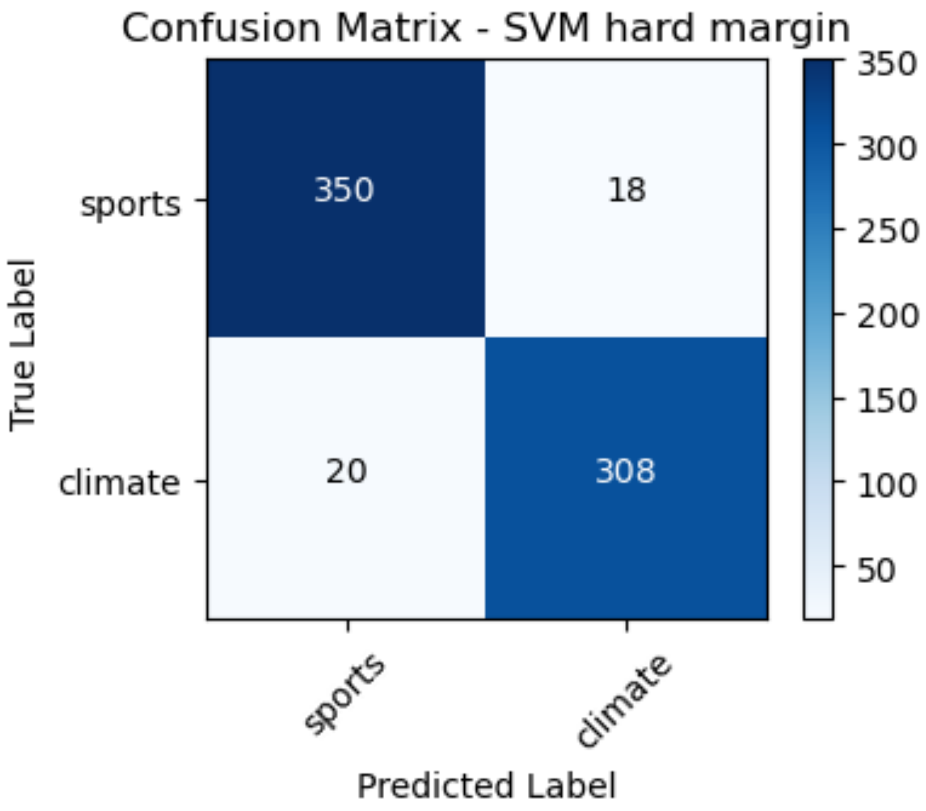
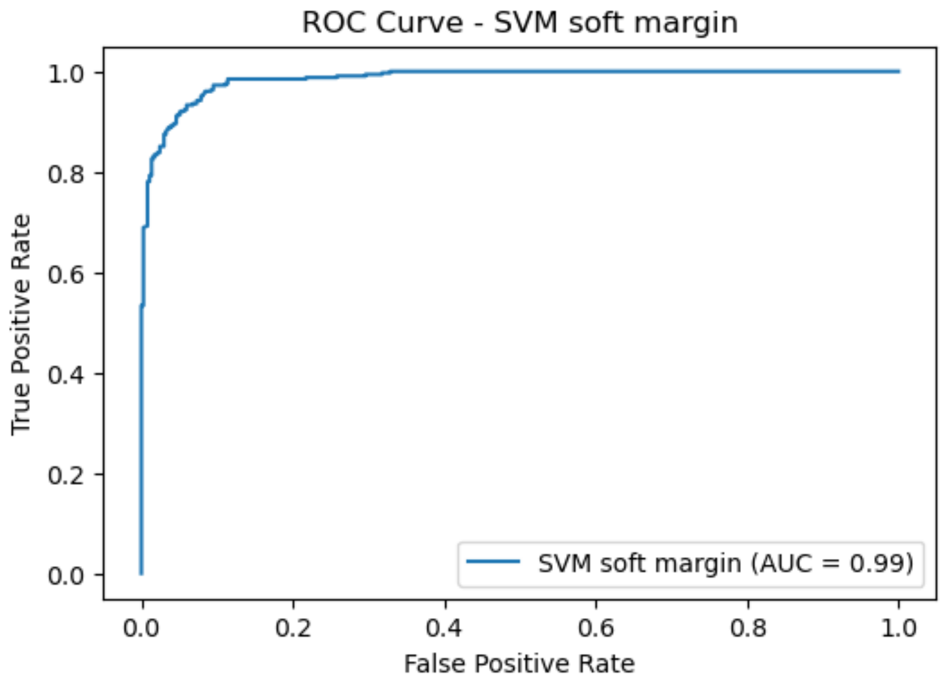


Figure 6: Confusion Matrix of Hard Margin SVM

Accuracy: 0.9454
Recall: 0.9390
Precision: 0.9448
F1 Score: 0.9419



Gemma = 0.0005

Figure 7: ROC Curve of Soft Margin SVM

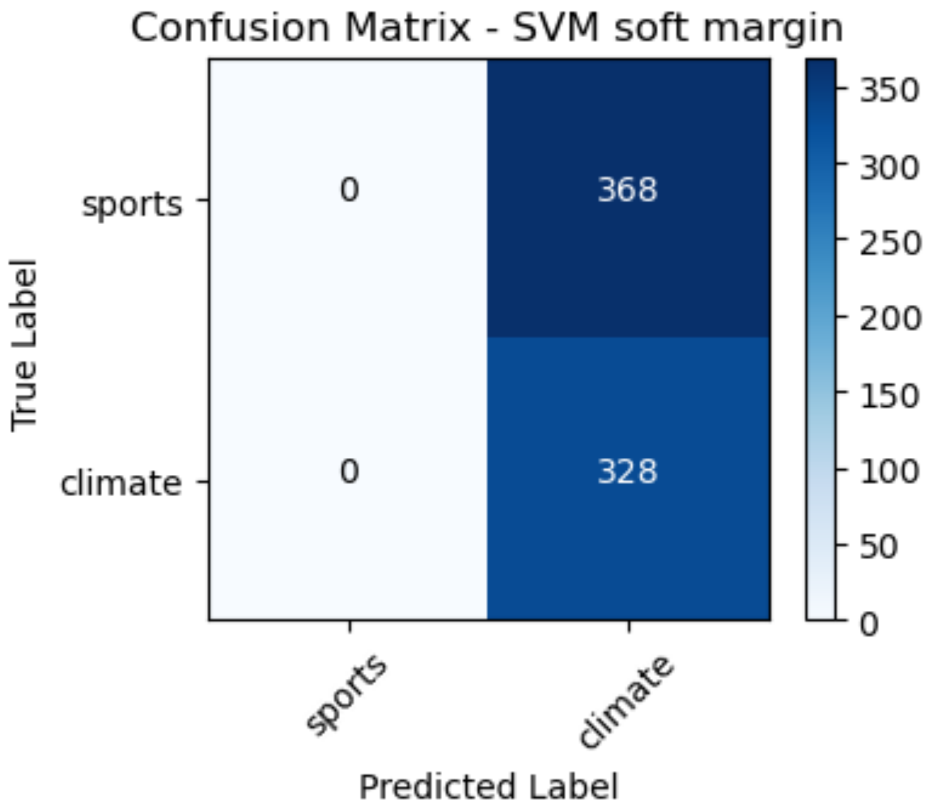
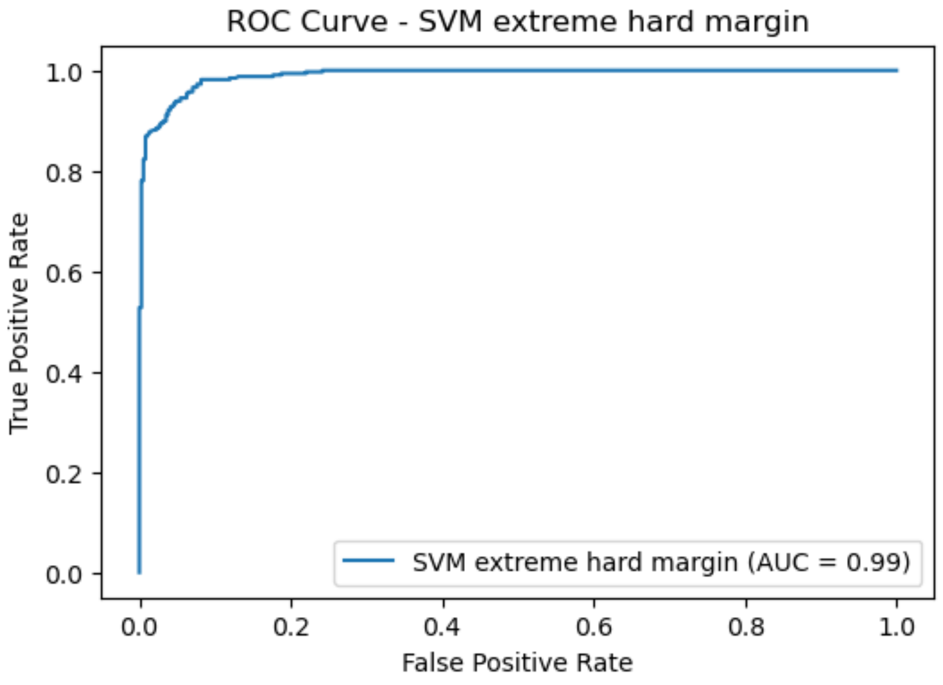


Figure 8: Confusion Matrix of Soft Margin SVM

Accuracy: 0.4713
Recall: 1.0000
Precision: 0.4713
F1 Score: 0.6406

Gemma = 100000



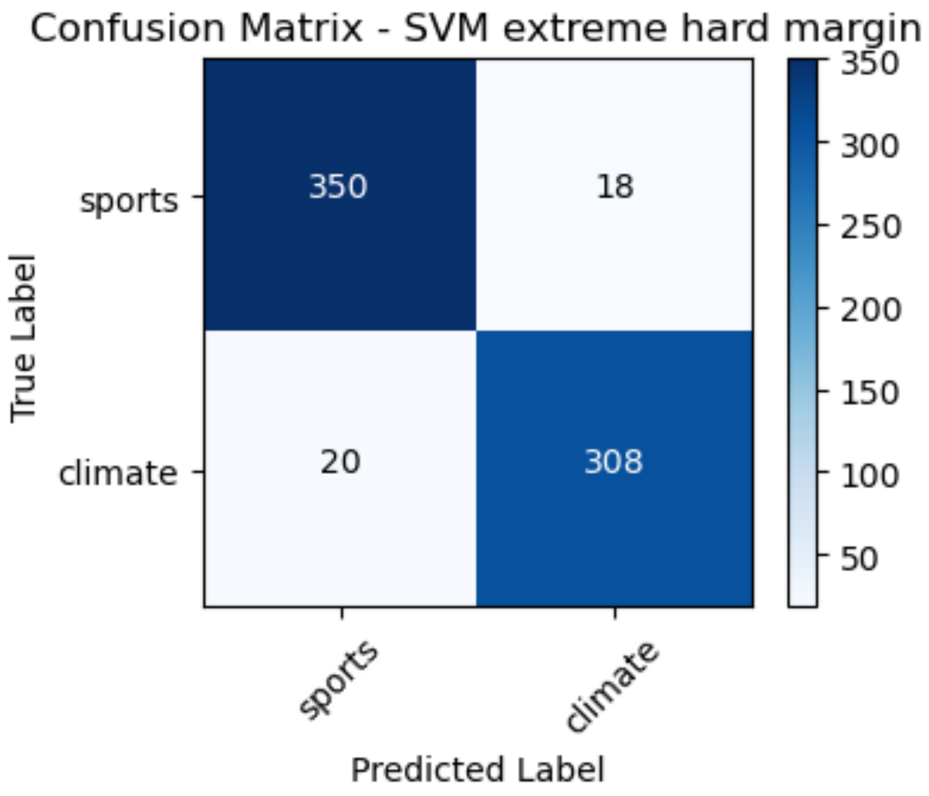


Figure 9: ROC Curve of Extreme Hard Margin SVM

Figure 10: Confusion Matrix of Extreme Hard Margin SVM

Accuracy: 0.9454
Recall: 0.9390
Precision: 0.9448
F1 Score: 0.9419

Based on the output scores, Gemma = 2000, hard margin SVM performs better. When Gemma = 100000, the model performs the same as the hard margin SVM.

- **What happens to the soft margin SVM?**

Based on the confusion matrix, all samples with sports label are classified as climate while all samples with climate label are correctly classified. This is because small Gemma value penalizes misclassification less, which probably leads to underfitting. So the classifier's decision boundary becomes less strict, which leads to the misclassification of all sports samples.

- **Does the ROC curve reflect the performance of the soft margin SVM? Why?**

The ROC curve partially reflects the performance of the soft margin SVM. It shows how well the model classify between two classes across different thresholds. It plots the true positive rate against the false positive rate. The top-left corner of the plot suggests that the model can effectively classify two classes, while it will not reflect the issue of precision.

- **Best Gemma from cross-validation.**

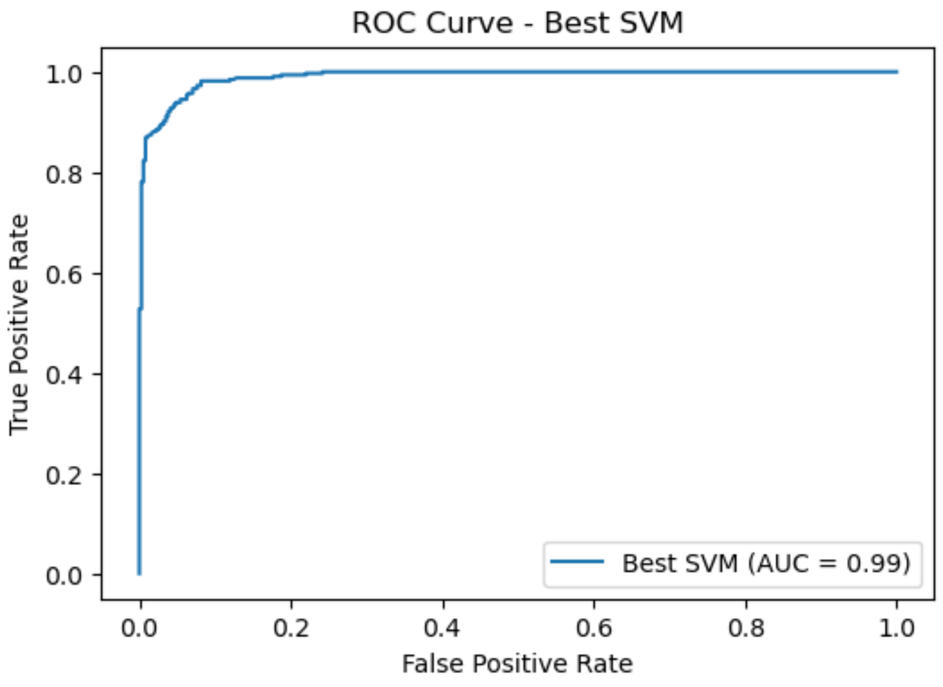


Figure 11: ROC Curve of Best SVM

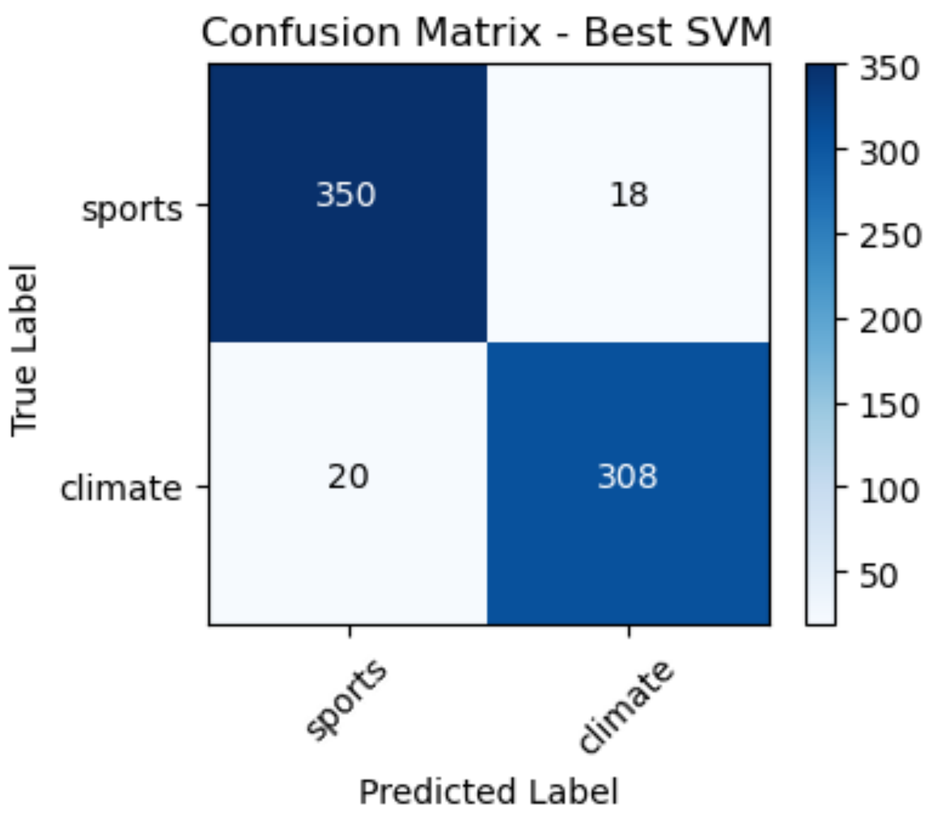


Figure 12: Confusion Matrix of Best SVM

```
Gemma = 0.001, Mean Accuracy = 0.5029
Gemma = 0.01, Mean Accuracy = 0.7838
Gemma = 0.1, Mean Accuracy = 0.9284
Gemma = 1, Mean Accuracy = 0.9406
Gemma = 10, Mean Accuracy = 0.9500
Gemma = 100, Mean Accuracy = 0.9522
Gemma = 1000, Mean Accuracy = 0.9511
Gemma = 10000, Mean Accuracy = 0.9529
Gemma = 100000, Mean Accuracy = 0.9525
Gemma = 1000000, Mean Accuracy = 0.9525

Best Gemma from Cross-Validation: 10000
Accuracy: 0.9454
Recall: 0.9390
Precision: 0.9448
F1 Score: 0.9419
```

Question 6:

- **Logistic classifier without regularization:**

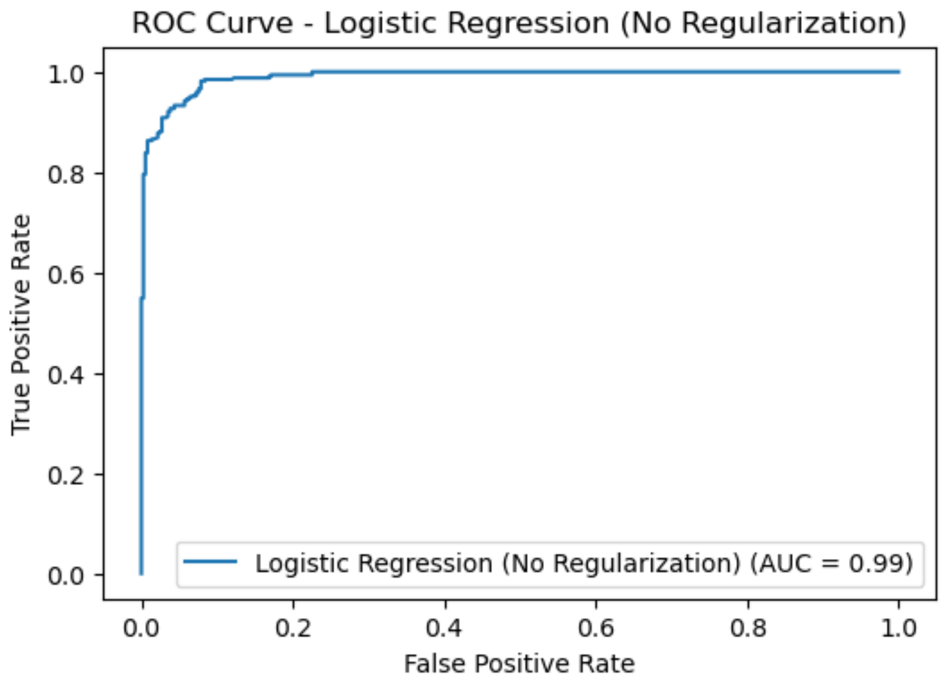


Figure 13: ROC Curve of Logistic Classifier without Regularization

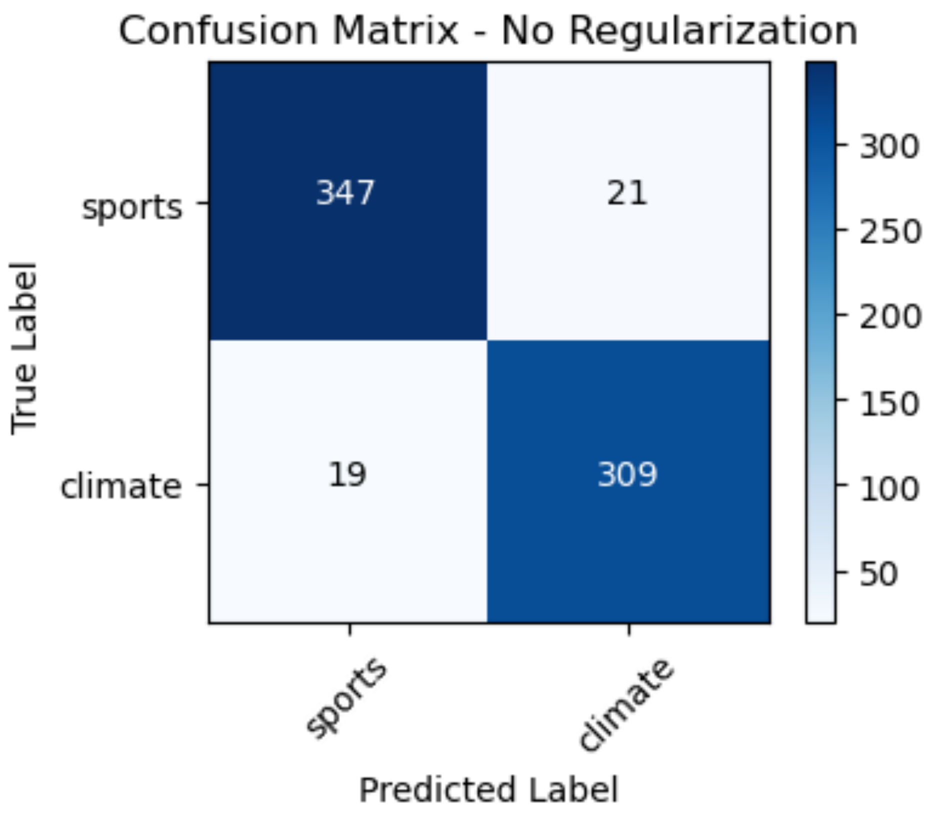


Figure 14: Confusion Matrix of Logistic Classifier without Regularization

Accuracy: 0.9425
Recall: 0.9421
Precision: 0.9364
F1 Score: 0.9392

- ***Optimal regularization strength for* logistic classifiers with L1 and L2 regularization**

The optimal regularization strength for L1 regularization is 10. The optimal regularization strength for L2 regularization is 100. The best score from GridSearchCV are both 0.9504.

- **Performance comparison:**

No regularization:
Accuracy: 0.9425
Recall: 0.9421
Precision: 0.9364
F1 Score: 0.9392

L1 regularization:
Accuracy: 0.9397

Recall: 0.9329
Precision: 0.9387
F1 Score: 0.9358

L2 regularization:
Accuracy: 0.9411
Recall: 0.9329
Precision: 0.9415
F1 Score: 0.9372

Based on the score, the logistic classifier without regularization has the best performance, with only precision slightly lower than other two.

- **How does the regularization parameter affect the test error? How are the learnt coefficients affected? Why might one be interested in each type of regularizations?**

In general, a small regularization strength allows the model to fit the data closely, may lead to overfitting. A large regularization strength penalizes large coefficients, may simplify the model but lead to underfitting.Without regularization, the model tends to be unstable, produce larger test error. The L1 regularization tends to produce a sparse model by the zero coefficients leaned. The L2 regularization will evenly distributes the coefficients across all features, shrinking them towards zero without actually making any exactly zero. This prevents one feature from dominating. So we may be interested in L1 regularization when we would like to remove less relevant features and making the model simpler. But we may be interested in L2 regularization when features are correlated.

- **Difference between the ways SVM and Logistic Regression find their decision boundary. Why their performances differ? Is this difference statistically significant?**

Logistic regression focuses on minimizing the log-loss, predicting the label based on the class probabilities. The SVM focuses on the hyperplane that maximizes the margin between two classes, which is determined by the closest data. The performances differ because logistic regression is sensitive to the outliers while SVM will basically ignore the outliers. The difference should be considered under the use case. We cannot say one model is statistically greater than another unless we have a specific use case.

Question 7:

- **Naive Bayes classifier performance:**

Accuracy: 0.9080
Recall: 0.8384
Precision: 0.9615
F1 Score: 0.8958

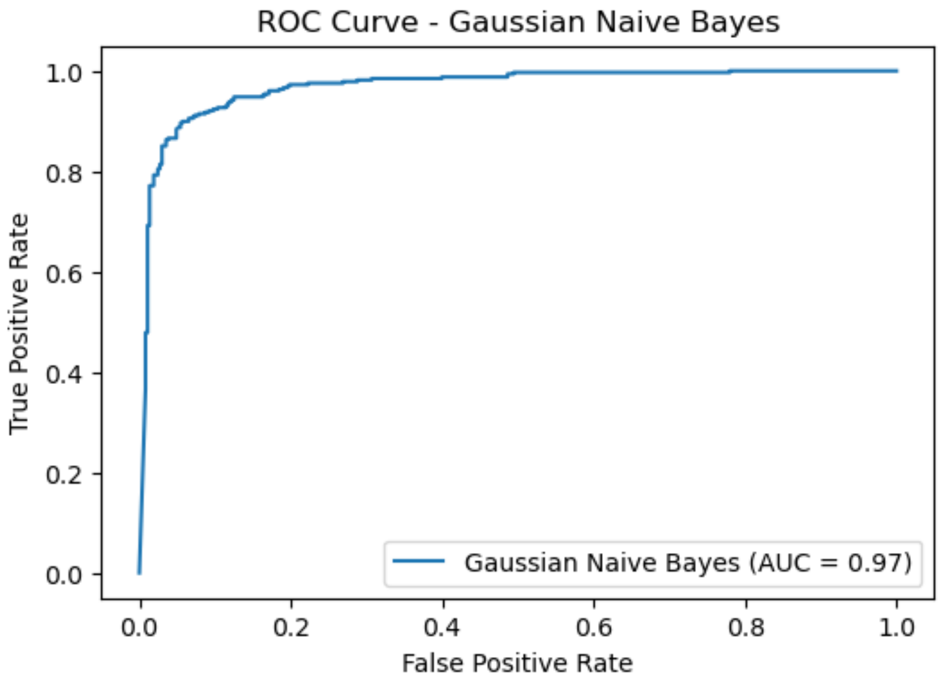


Figure 15: ROC Curve of Naive Bayes Classifier

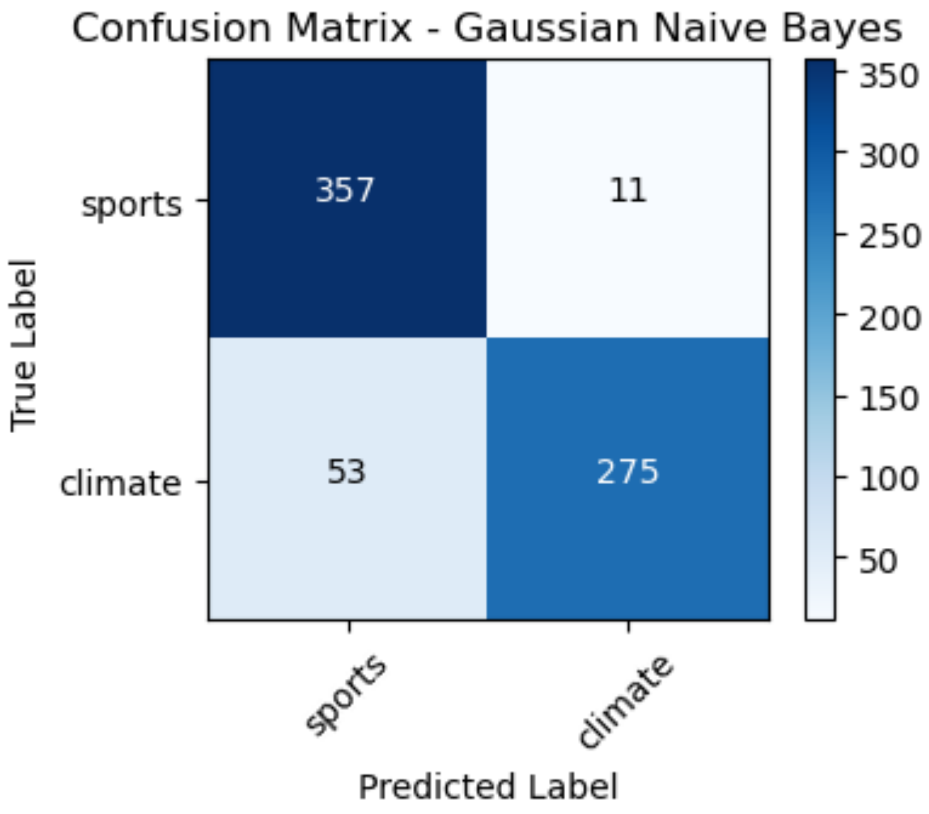


Figure 16: Confusion Matrix of Naive Bayes

Question 8:

- **5 best combinations:**

```
--- Model Configuration 58: {'classifier': LogisticRegression(C=100, random_state=42,
solver='liblinear'), 'dim_reduction': TruncatedSVD(n_components=100,
random_state=42), 'dim_reduction__n_components': 100, 'text_processing':
'Lemmatization', 'vectorizer__min_df': 5} ---
Confusion Matrix:
[[350  18]
 [  7 321]]
Accuracy: 0.9641
Recall: 0.9787
Precision: 0.9469
F1 Score: 0.9625

--- Model Configuration 34: {'classifier': LogisticRegression(C=10, penalty='l1',
random_state=42, solver='liblinear'), 'dim_reduction':
TruncatedSVD(n_components=100, random_state=42),
'dim_reduction__n_components': 100, 'text_processing': 'Lemmatization',
'vectorizer__min_df': 5} ---
Confusion Matrix:
[[351  17]
 [  9 319]]
Accuracy: 0.9626
Recall: 0.9726
Precision: 0.9494
F1 Score: 0.9608

--- Model Configuration 57: {'classifier': LogisticRegression(C=100, random_state=42,
solver='liblinear'), 'dim_reduction': TruncatedSVD(n_components=100,
random_state=42), 'dim_reduction__n_components': 100, 'text_processing':
'Lemmatization', 'vectorizer__min_df': 2} ---
Confusion Matrix:
[[351  17]
 [  7 321]]
Accuracy: 0.9655
Recall: 0.9787
Precision: 0.9497
F1 Score: 0.9640

--- Model Configuration 9: {'classifier': SVC(C=10000, kernel='linear',
random_state=42), 'dim_reduction': TruncatedSVD(n_components=100,
random_state=42), 'dim_reduction__n_components': 100, 'text_processing':
'Lemmatization', 'vectorizer__min_df': 2} ---
Confusion Matrix:
```

```
[[351  17]
 [  6 322]]
Accuracy: 0.9670
Recall: 0.9817
Precision: 0.9499
F1 Score: 0.9655

--- Model Configuration 10: {'classifier': SVC(C=10000, kernel='linear',
random_state=42), 'dim_reduction': TruncatedSVD(n_components=100,
random_state=42), 'dim_reduction__n_components': 100, 'text_processing':
'Lemmatization', 'vectorizer__min_df': 5} ---
Top Model 10
Confusion Matrix:
[[350  18]
 [  6 322]]
Accuracy: 0.9655
Recall: 0.9817
Precision: 0.9471
F1 Score: 0.9641
```

Question 9:

- **Performances of Naive Bayes and multiclass SVM (both One VS One and One VS the rest methods). How did you resolve the class imbalance issue in the One VS the rest model?**

Scores for Naive bayes without merging:
Accuracy: 0.6925
Recall: 0.6911
Precision: 0.6820
F1 Score: 0.6742

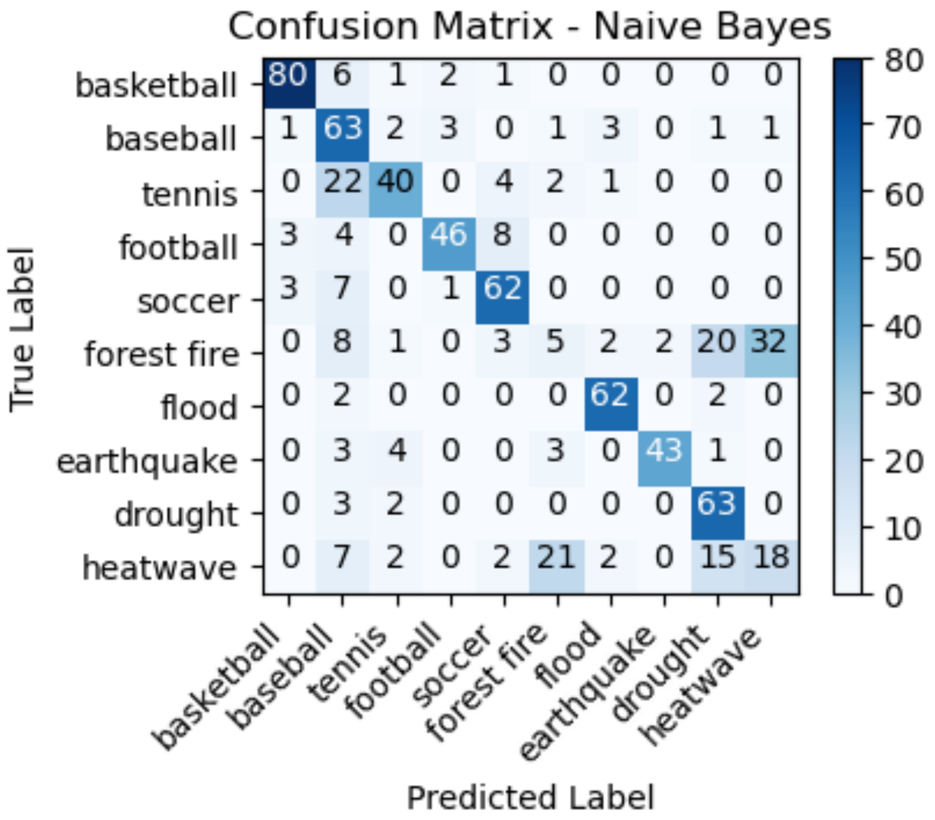


Figure 17: Confusion Matrix of Naive Bayes without Merging

Scores for SVM One VS One method without merging:
Accuracy: 0.7672
Recall: 0.7678
Precision: 0.7867
F1 Score: 0.7753

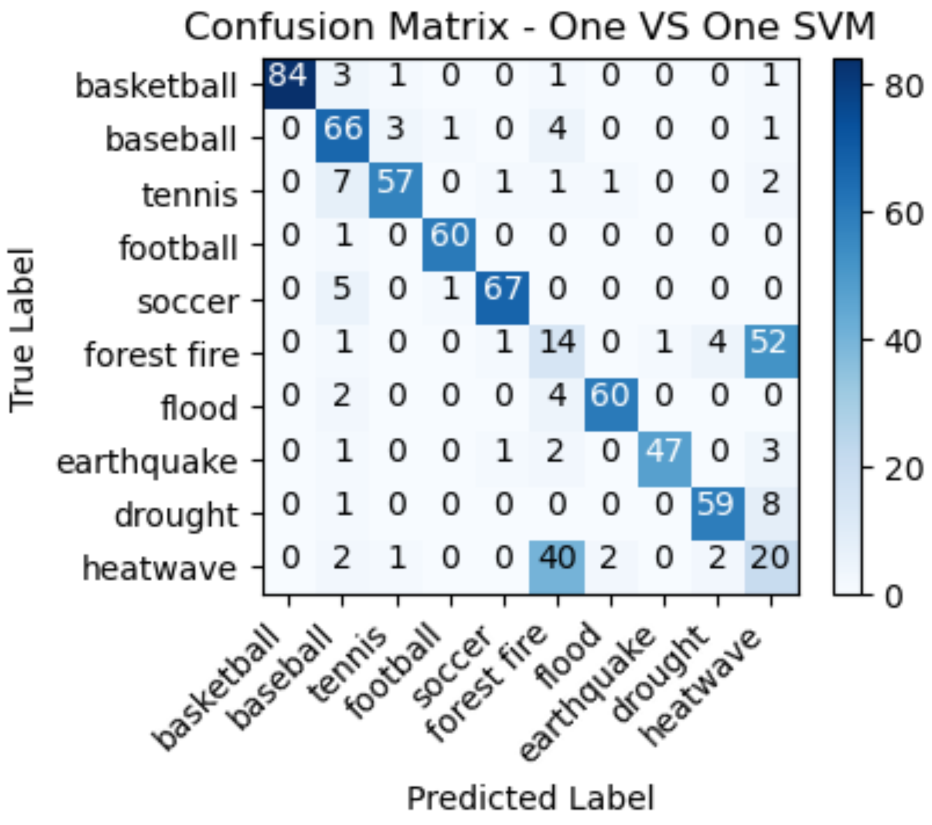


Figure 18: Confusion Matrix of SVM One VS One Method without Merging

Scores for SVM One VS the rest method without merging:
Accuracy: 0.7787
Recall: 0.7802
Precision: 0.7869
F1 Score: 0.7829

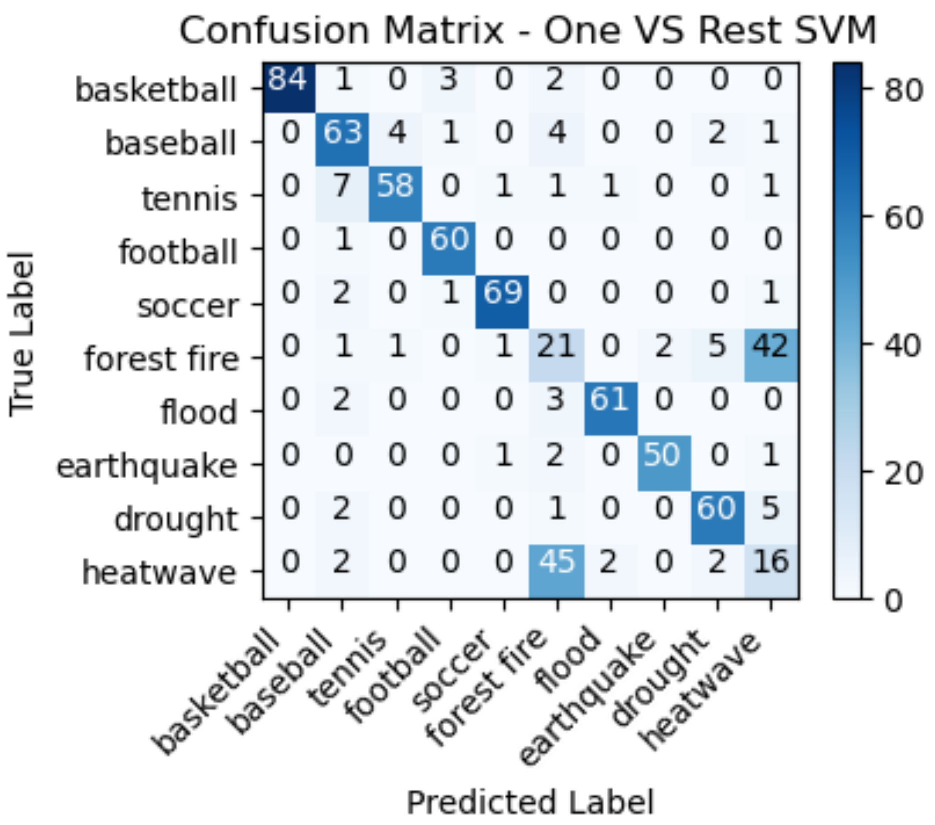


Figure 20: Confusion Matrix of SVM One VS the rest Method without Merging

To resolve the class imbalance issue in the One VS the rest method, we used class_weight=‘balanced’ in the SVM classifier. According to the documentation, this option automatically adjusts the weight of each class inversely proportional to its frequency in the training data. As a result, the minority classes receive higher penalties for misclassification, ensuring that the model learns to identify them effectively.

- **Do you observe any structure in the confusion matrix? Are there distinct visible blocks on the major diagonal? What does that mean?**

The confusion matrix shows a structure with diagonal blocks indicating correct classifications for most labels. However, misclassification occurs for labels that are similar to each other, such as "forest Fire" and “heatwave." There are distinct visible blocks on the major diagonal. These blocks indicate the correct classification and the model's ability to distinguish between the given classes.

- **Based on your observation from the previous part, suggest a subset of labels that should be merged into a new larger label and recompute the accuracy and plot the confusion matrix. How did the accuracy change in One VS One and One VS the rest?**

I suggest to merge label ‘heatwave’ with ‘forest fire’ since they are often misclassified as the other one.

Scores for Naive Bayes after merging:
Accuracy: 0.7658
Recall: 0.7892
Precision: 0.8005
F1 Score: 0.7779

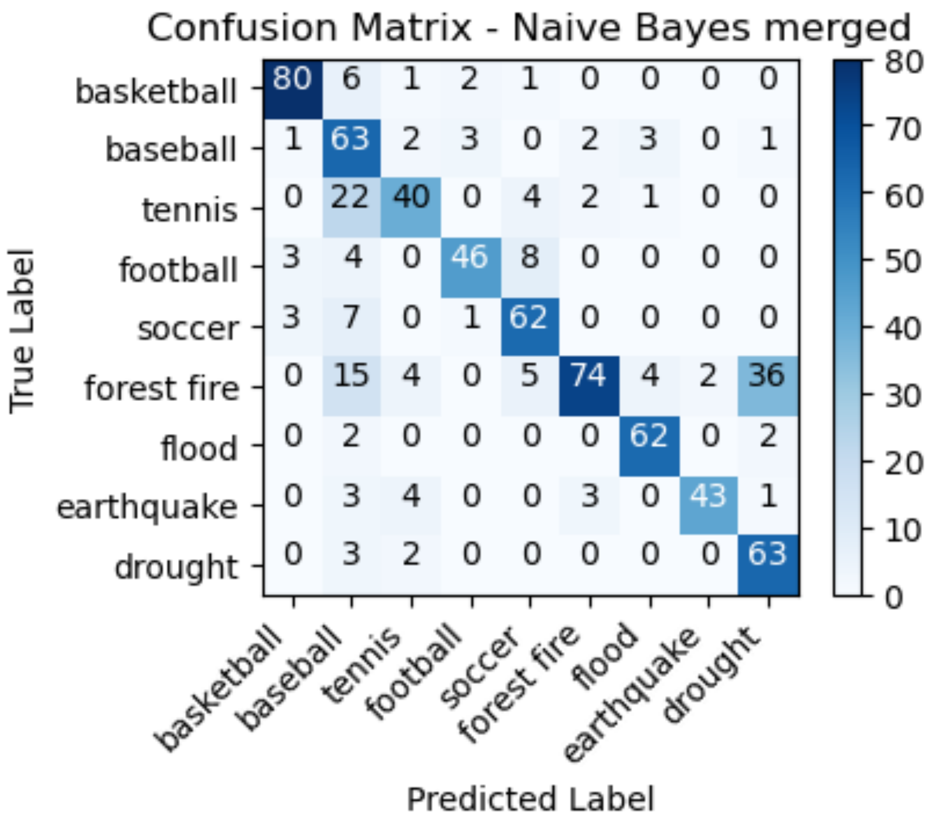


Figure 21: Confusion Matrix of Naive Bayes after Merging

Scores for SVM One VS One method after merging imbalanced:
Accuracy: 0.8836
Recall: 0.8785
Precision: 0.9144
F1 Score: 0.8932

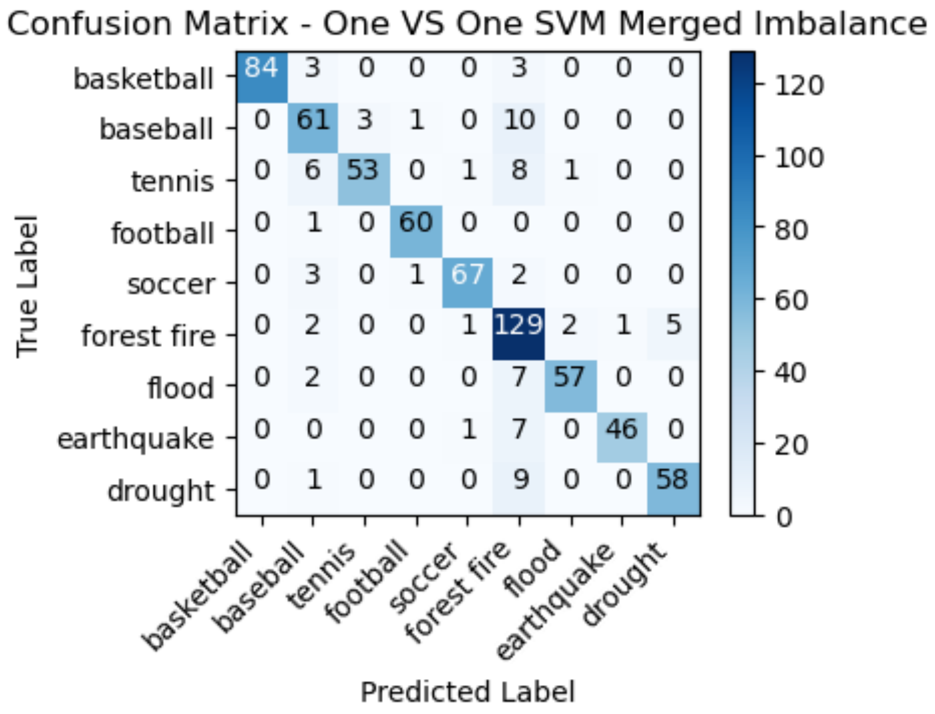
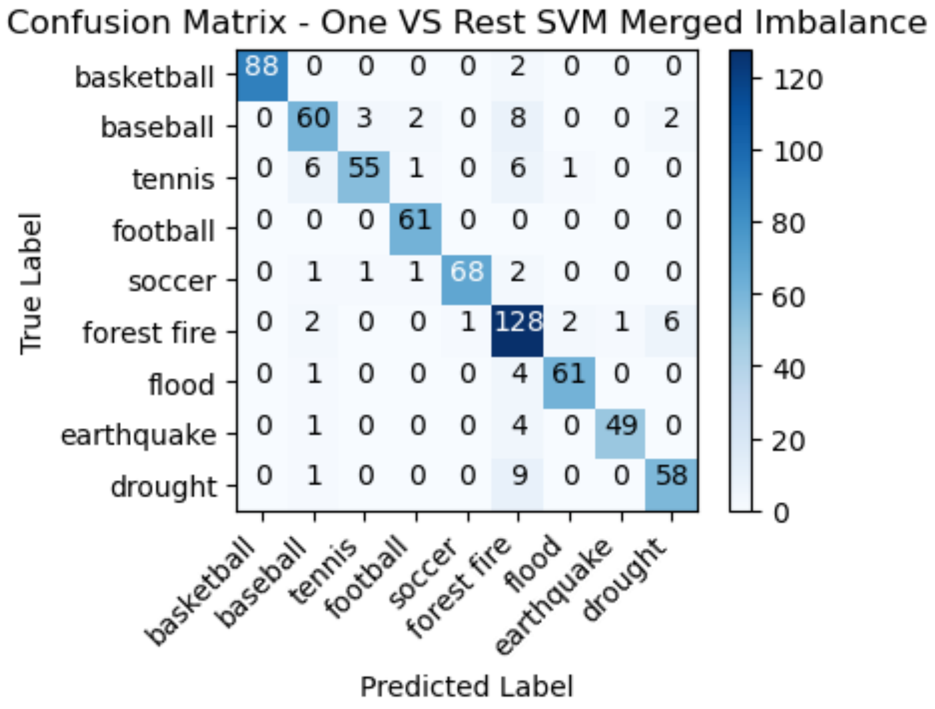


Figure 22: Confusion Matrix of SVM One VS One Method after Merging Imbalanced

Scores for SVM One VS the rest method after merging imbalanced:
Accuracy: 0.9023
Recall: 0.9006
Precision: 0.9207
F1 Score: 0.9091



	basketball	baseball	tennis	football	soccer	forest fire	flood	earthquake	drought
basketball	88	0	0	0	0	2	0	0	0
baseball	0	60	3	2	0	8	0	0	2
tennis	0	6	55	1	0	6	1	0	0
football	0	0	0	61	0	0	0	0	0
soccer	0	1	1	1	68	2	0	0	0
forest fire	0	2	0	0	1	128	2	1	6
flood	0	1	0	0	0	4	61	0	0
earthquake	0	1	0	0	0	4	0	49	0
drought	0	1	0	0	0	9	0	0	58

Figure 23: Confusion Matrix of SVM One VS the rest method after Merging Imbalanced

The accuracy in One VS One method increased from 0.7672 to 0.8836.
The accuracy in One VS the rest method increased from 0.7787 to 0.9023.

- **Does class imbalance impact the performance of the classification once some classes are merged? Provide a resolution for the class imbalance and recompute the accuracy and plot the confusion matrix in One VS One and One VS the rest?**

Class imbalance will impact the performance of the classification once some classes are merged. Again, to resolve the imbalance issue after merging, we use class_weight='balanced' in the SVM classifier. After balancing the class, the performance is improved.

Scores for SVM One VS One method after merging balanced:
Accuracy: 0.9023
Recall: 0.9032
Precision: 0.9161
F1 Score: 0.9079

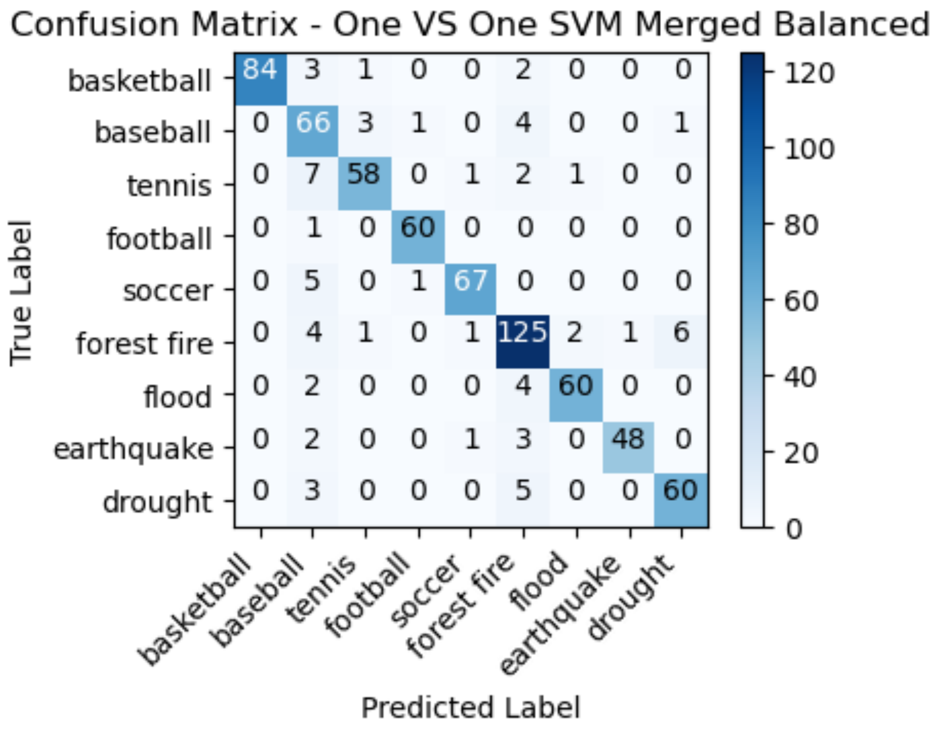
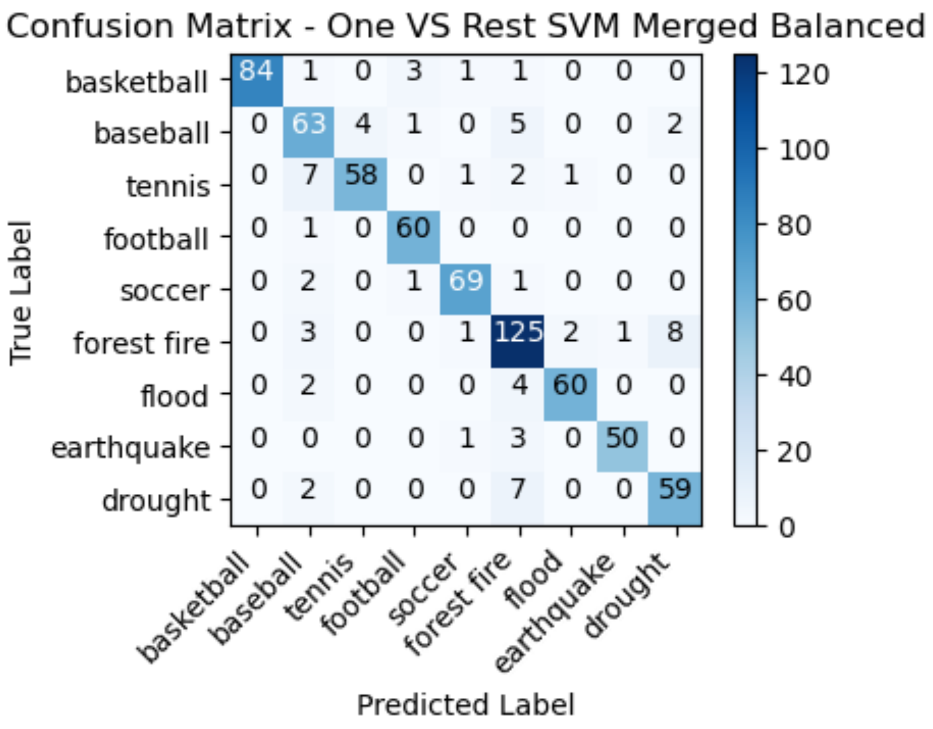


Figure 24: Confusion Matrix of SVM One VS One Method after Merging Balanced

Scores for SVM One VS the rest method after merging balanced:
Accuracy: 0.9023
Recall: 0.9042
Precision: 0.9127
F1 Score: 0.9076



True Label \ Predicted Label	basketball	baseball	tennis	football	soccer	forest fire	flood	earthquake	drought
basketball	84	1	0	3	1	1	0	0	0
baseball	0	63	4	1	0	5	0	0	2
tennis	0	7	58	0	1	2	1	0	0
football	0	1	0	60	0	0	0	0	0
soccer	0	2	0	1	69	1	0	0	0
forest fire	0	3	0	0	1	125	2	1	8
flood	0	2	0	0	0	4	60	0	0
earthquake	0	0	0	0	1	3	0	50	0
drought	0	2	0	0	0	7	0	0	59

Figure 25: Confusion Matrix of SVM One VS the rest method after Merging Balanced

Question 10:

- **Why are GLoVE embeddings trained on the ratio of co-occurrence probabilities rather than the probabilities themselves?**

This is because the ratio would capture the semantic relations between words more effectively. While raw probabilities only reflect the likelihood of two words happening together, the ratio would compare how often they occur together relative to how often they occur together with other words. So using this ration will make the embeddings more robust in capturing the global and local structure.

- **In the two sentences: “James is running in the park.” and “James is running for the presidency.”, would GLoVE embeddings return the same vector for the word running in both cases? Why or why not?**

It would return the same vector for the word running, because GloVe does not consider the specific context of individual sentences.

- **What do you expect for the values of, ||GLoVE["left"] - GLoVE["right"]||2, ||GLoVE["wife"] - GLoVE["husband"]||2 and ||GLoVE["wife"] - GLoVE["orange"]||2 ? Compare these values.**

For left and right, the value would be relatively small, because they have similar meaning and often appear in similar contexts. The shared contexts will make their embeddings close to each other. For wife and husband, the value would also be small due to the same reason as above. However, for wife and orange, the value would be relatively larger as wife and orange are less relevant and they are less likely to occur together.

- **Given a word, would you rather stem or lemmatize the word before mapping it to its GLoVE embedding?**

I would lemmatize the word before mapping. Lemmatization would preserve the meaning of a word while stemming sometimes would not. And a stemmed word may not exist in the GloVe vocabulary, leading to mapping issues.

Question 11:

The representation of text segments has a dimension of [2780, 300], which does not exceed the dimension of the GloVe embedding.
The classifier chosen is Logistic Regression with L1 regularization. The scores are the following:
Confusion Matrix:

```
[[348  20]
 [ 11 317]]
Accuracy: 0.9555
Recall: 0.9561
Precision: 0.9550
F1 Score: 0.9554
```

Question 12:

- **Plot the relationship between the dimension of the pre-trained GLoVE embedding and the resulting accuracy of the model in the classification task. Describe the observed trend. Is this trend expected? Why or why not? In this part use the different sets of GLoVE vectors from the link.**

Based on the trend shown in the plot, generally, the accuracy increase with the increasing dimensions. This trend is expected because lower dimensions will not encode sufficient semantic information. Higher dimensions will have the ability to capture most relevant information.

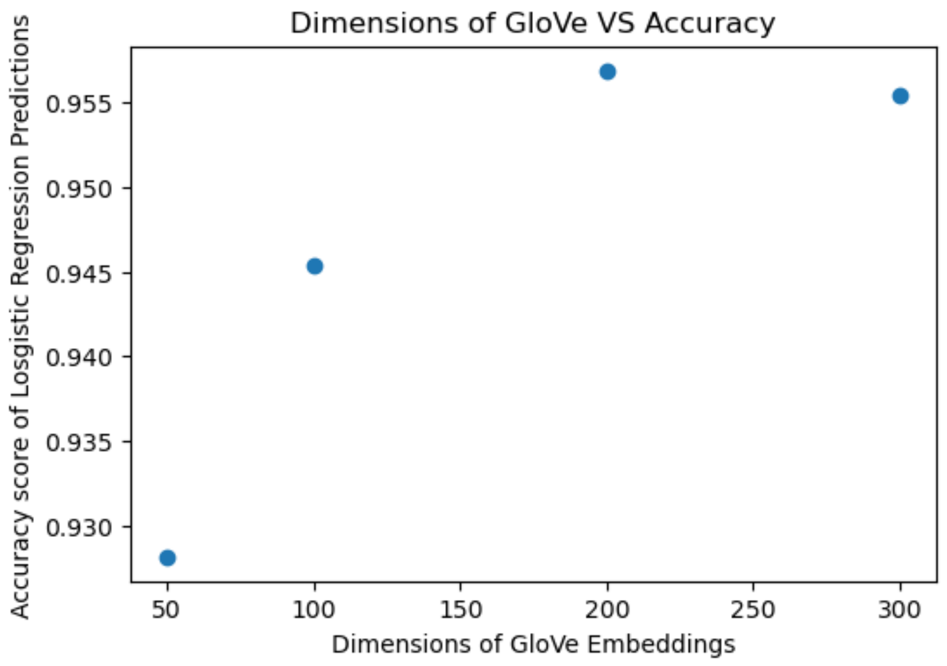


Figure 26: Dimensions of GloVe Embeddings VS Accuracy

Question 13:

- **Compare and contrast the two visualizations. Are there clusters formed in either or both of the plots? We will pursue the clustering aspect further in the next project.**

The "Random Cluster" plot shows a uniform distribution of random vectors with no clusters, due to the lack of semantic information. In contrast, the "GLoVE Cluster" plot shows the class-based clusters for "sports" and “climate,”. The contextual relations are captured. This comparison indicates the effectiveness of GLoVE embeddings in encoding information that aligns with document classes.

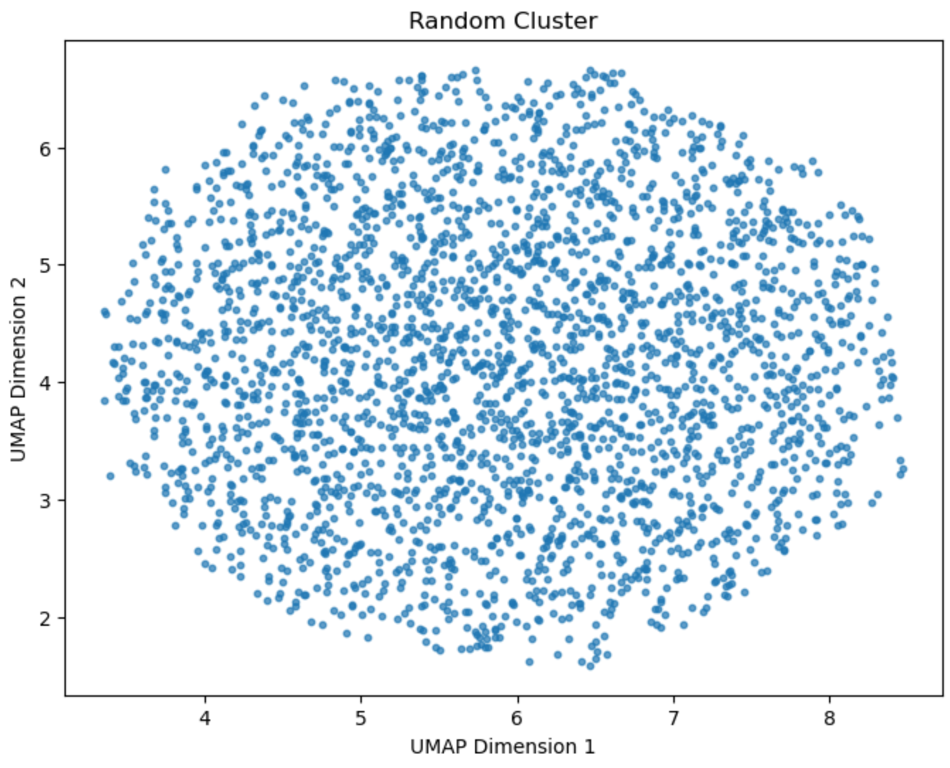


Figure 27: Random Cluster

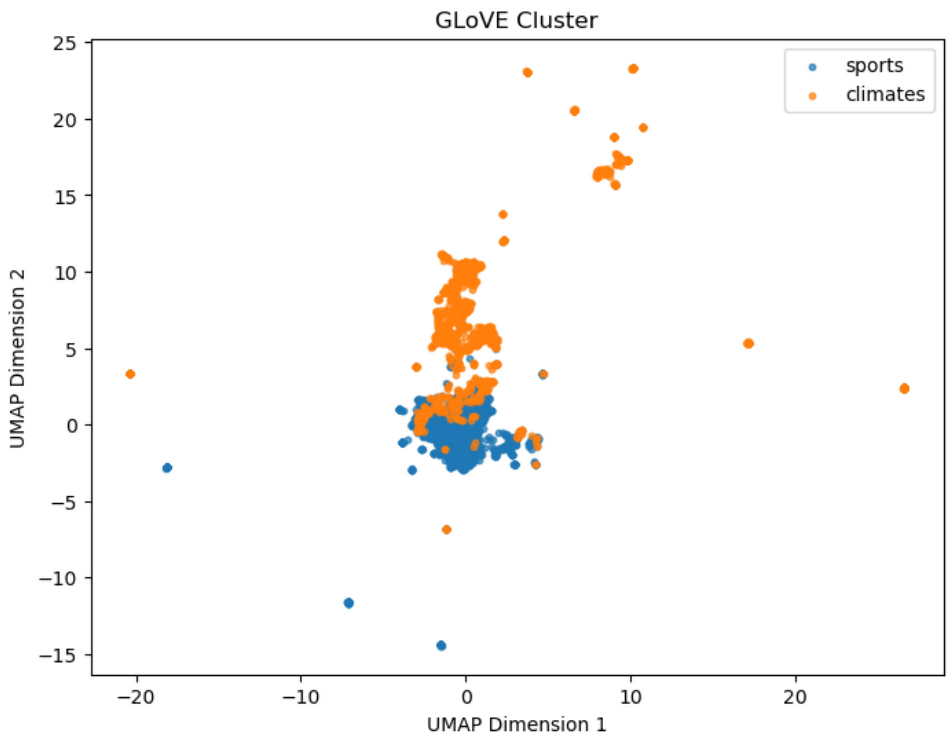


Figure 28: GloVe Cluster