Project 1 Classification Report

EECS 219

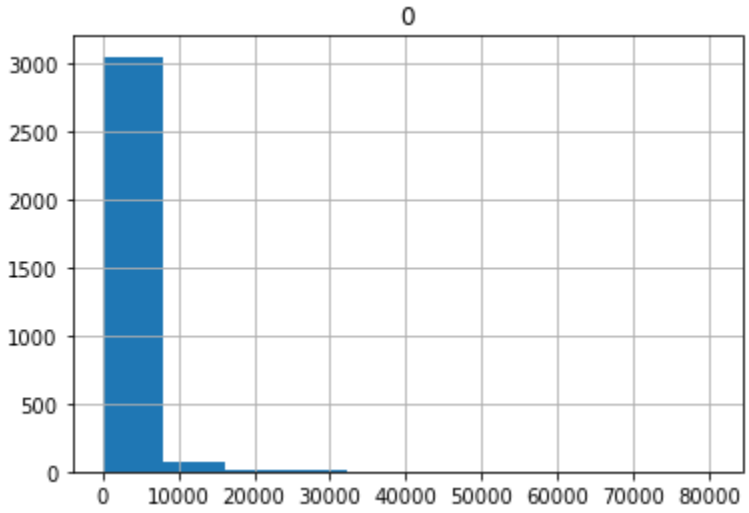
Group: Fengyuan Heying, Zheng Lu, Xinyu Wu, Dian Zhu

Question 1:

1.1 Overview:

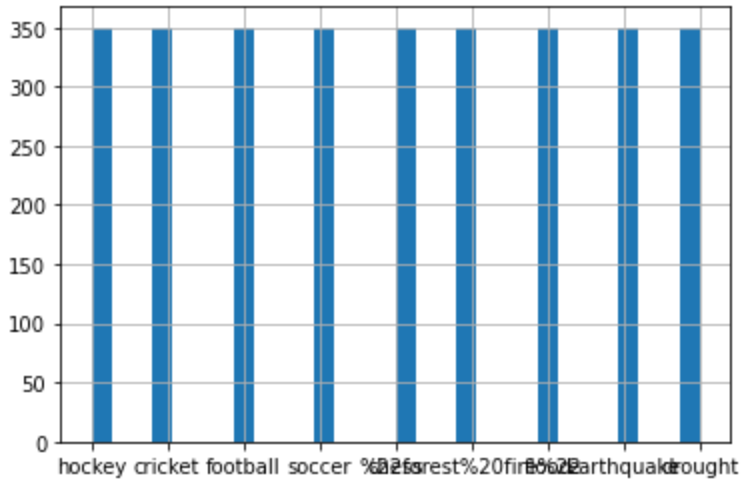
There are **3,150** samples and **8** features in the dataset.

1.2/1.3 Histograms/Interpretation:



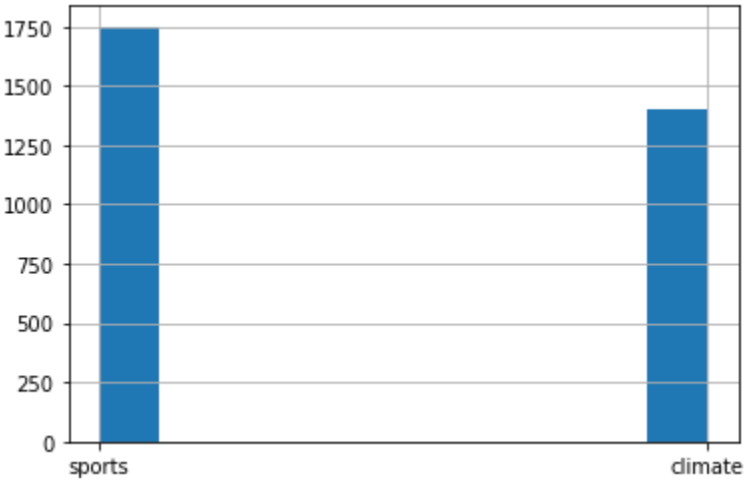
Histogram (a)

- We can see that approximately 3,000 samples have less than 8,000 alpha-numeric characters, 100 samples have between 8,000 and 18,000 characters, and 50 samples over 18,000 characters.



Histogram (b)

- The second histogram shows the frequency of each leaf label. There are 9 variables in the leaf label and each has the same amount, 350. It's a relatively balanced response value.



Histogram (c)

- The third histogram shows the relationship between root label and amount of each variable. As we can see from the root_topic histogram, text in the dataset is separated into two different root labels: sports and climate, and the dataset focuses more on sports. It shows that there are 1,750 “sports” variables and 1,400 “climate” variables.

Question 2:

After splitting the training and testing data, we got a train data set with **2520** samples and **630** samples in test data.

Question 3:

3.1 Pros and Cons of lemmatization and affection:

- pros: lemmatization is more accurate and better quality. Lemmatization has more consideration and analysis about words, so it helps extract better features.
- cons: lemmatization takes longer and harder to implement. Computation complexity and cost is higher than stemming.
- Affection: As above states, lemmatization has a larger dictionary size than stemming.

3.2 min_df:

min_df removes the words that appear too seldom. So, as min_df increases, the TF-IDF matrix will decrease because it removes information.

3.3 Remove stop words/punctuation/number:

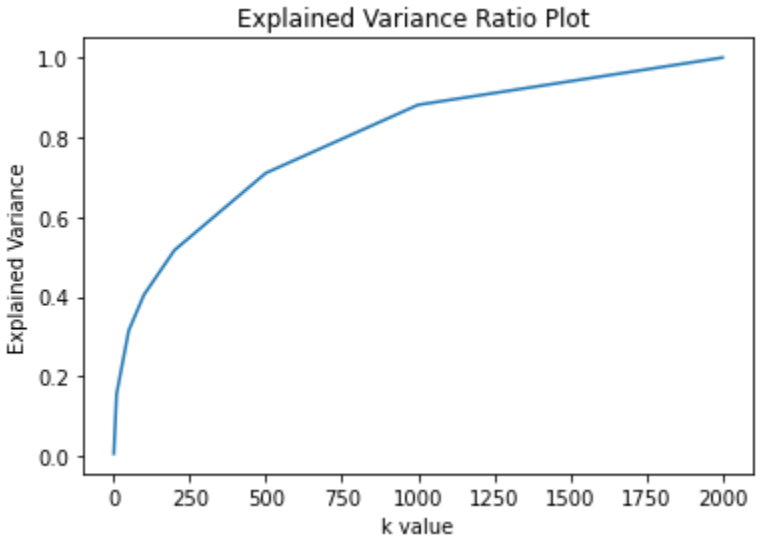
Lemmatization always goes first. Remove stopwords, punctuations, and numbers should be after lemmatization because stopwords, punctuations, and numbers are elements we are not interested in and they will affect lemmatization to normalize the words.

3.4 TF-IDF matrix:

The TF-IDF matrix of train data is (2520, 13715) and the TF-IDF matrix of test data is (630, 13715). Both matrices have the same row number of the original data set and expand to many columns. It shows that words in full_text are already normalized.

Question 4:

4.1 Plot and concavity:



Plot of explained variance ratio and k-value for LSI

- The plot shows the relationship between a sequence of k values (k=1,10,50,100,200,500,1000,2000) and explained variance ratio for LSI methods. The curve in the plot is relatively smooth and linear with an increasing tendency.
- It's obviously concave down meaning that as k increases, explained variance proportion increases as well. The concavity also suggests that k=50 should be the best k value which with simplest complexity and high variance explained.

4.2 Compare residual MSE and reason:

	Residual MSE (around 4 decimal points)
NMF	1697.8966
LSI	1669.3038

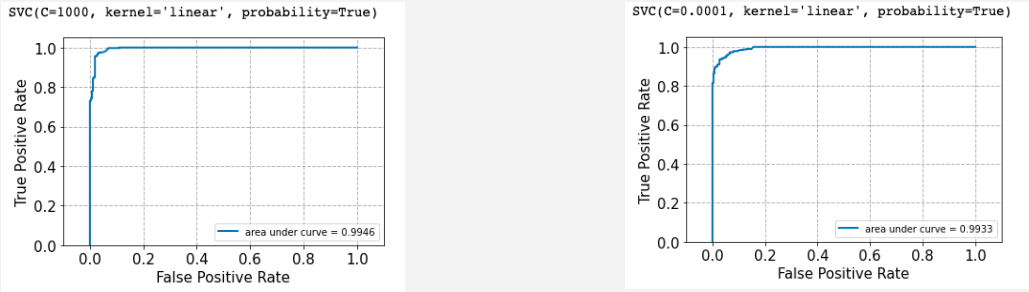
- With the same k=50 parameter, residual MSE error in NMF is larger.

- Reason: LSI filters out noise and tries to figure out the meaning and topic behind the words because it usually considers only essential components of term-by-document matrix. Bases and weights of NMF constraints to be positive, so only additive combinations are allowed. Simultaneously, bases and weights in NMF are sparse. It may cause losing a few useful information during dimension reduction compared with LSI. So, LSI will be more accurate than NMF and has a smaller MSE.

Question 5:

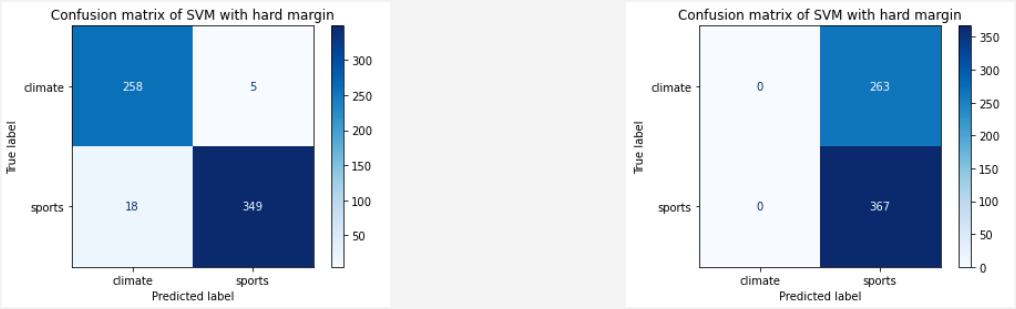
5.1 Train linear SVMs:

γ=1000 γ=0.001



ROC curve plots with 2 margins (SVM)

γ=1000 γ=0.001



Confusion Matrix plots with 2 margins (SVM)

γ=1000

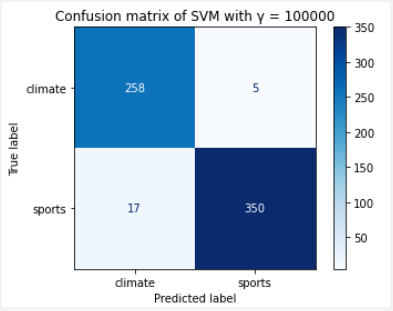
	Score type	Hard Margin
0	Accuracy	0.963492
1	Recall	0.950954
2	Precision	0.985876
3	F1	0.968100

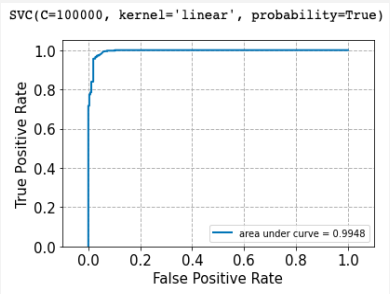
γ=0.001

	Score type	Soft Margin
0	Accuracy	0.582540
1	Recall	1.000000
2	Precision	0.582540
3	F1	0.736209

Score tables with 2 margins in the SVM

- Based on the ROC plots, confusion matrix, and score table above, γ = 1000 performs best. As γ increases, the accuracy of the SVM model increases.
- When γ = 100000, it has the highest accuracy, 96.5079%. But there's not a big difference between γ = 1000 and γ = 100000 in accuracy. However, it increases complexity of model and time spent on computing.





	Score type	γ = 100000
0	Accuracy	0.965079
1	Recall	0.953678
2	Precision	0.985915
3	F1	0.969529

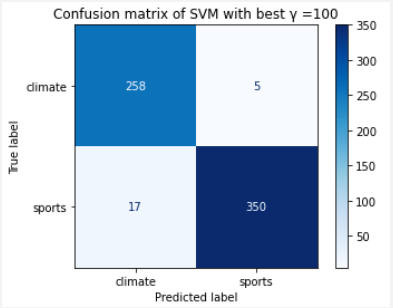
Summary with γ=100,000 (SVM)

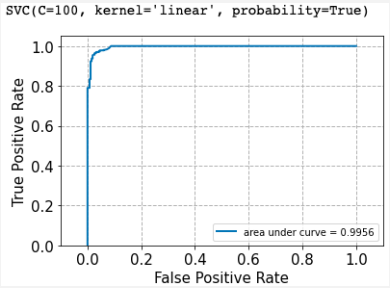
- Soft margin:
  - Area under the ROC curve measures the usefulness of the model. If there is more area, then the model is more efficient. But the area under the ROC curve with soft margin does not have a big difference compared to hard margin plot.

- ROC does not really reflect the soft margin because the ROC curve graphs the relationship between TPR and FPR. SVM with soft margin has the highest FPR and TPR.
- Soft margin SVM has a very low score and predicts all "Climates" values to "Sports". And here we only have two categories to classify which means SVM model with soft margin actually does not predict anything. It is caused by γ being too small and the margin too not big enough.

5.2 Cross-Validation:

After cross-validation, we find the best **γ = 100**.



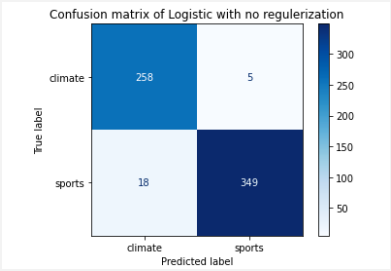


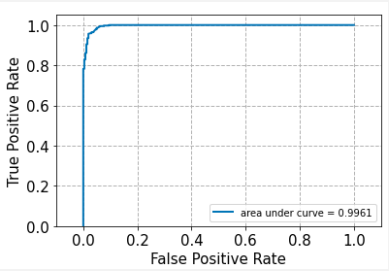
	Score type	SVM with best γ =100
0	Accuracy	0.965079
1	Recall	0.953678
2	Precision	0.985915
3	F1	0.969529

Summary with γ=100 (SVM)

Question 6:

6.1 Train Logistic Model:





	Score type	Logistic No Regularization
0	Accuracy	0.963492
1	Recall	0.950954
2	Precision	0.985876
3	F1	0.968100

Logistic regression (No Regularization) Summary

6.2 Optimal regularization coefficient:

- After using 5-fold cross-validation, we found that the optimal regularization strength for logistic regression with L1 regularization is **100** and the optimal regularization strength for logistic regression with L2 regularization is **1000**.
- Comparison:

	Score type	No Regularization	L1 Regularization	L2 Regularization
0	Accuracy	0.963492	0.963492	0.965079
1	Recall	0.950954	0.950954	0.953678
2	Precision	0.985876	0.985876	0.985915
3	F1	0.968100	0.968100	0.969529

Comparison Performance

- Regularization:
    - Regularization does not improve the performance on the seeing data set. L1 does not influence performance. But adding Regularization L2 will have better

generalization performance on unseeing data, test data set. It also helps us better access variables that play a big role and reduce multicollinearity problems.

- If adding regularization to the model, variable coefficient shrinks and towards 0, estimators with greater variability generally shrink more.
- L1 is lasso regression, which adds the "absolute value of magnitude" of the coefficient as a penalty term to the loss function. L2 Regularization, also called a ridge regression, adds the "squared magnitude" of the coefficient as the penalty term to the loss function. Both of them have less flexibility because highly-variable parameters are shrunk. Lasso will perform better when the independent variable does not really influence the response variable. On the contrary, ridge regression will have better performance with large predictors impacting the response variable.

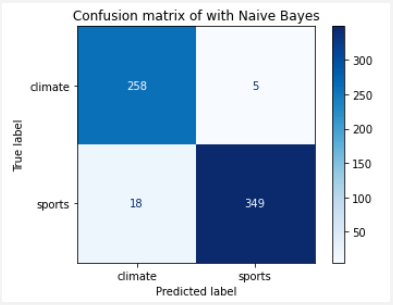
- Difference between SVM and Logistic:
  - SVM tries to make a decision boundary in such a way that the margin between the two classes is as wide as possible. It works better with text and image.
  - Logistic Regression yields better on identified independent variables. It determines the decision boundary based on the conditional probability.

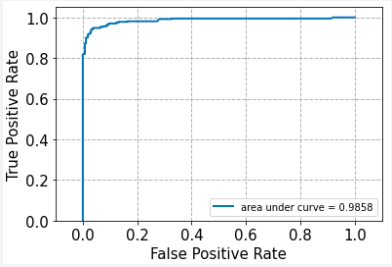
$$P(Y = 1) = \frac{e^{\beta_0+\beta_1 X}}{1+e^{\beta_0+\beta_1 X}}$$

  - They perform differently because the way they find decision boundaries is different. SVM is based on geometrical properties of the data while logistic regression is based on statistical approaches. Additionally, the number of features and examples also influence the model performance. SVM performs better with large scale data.

- The difference is not statistically significant because SVM with best γ = 100 and logistic regression output similar accuracy. The difference accuracy between two models is 0.001587 which is not statistically significant.

Question 7:





	Score type	Naive Bayes
0	Accuracy	0.934921
1	Recall	0.899183
2	Precision	0.988024
3	F1	0.941512

Naive Bayes GaussianNB classifier Summary

- Evaluation:

  Based on the summary of Naive Bayes model, we can see that its score statistics are relatively lower than Logistic Regression and SVM. Naive Bayes Model in this case is more easily to predict “sports” as “climate”. Its False Negative rate is higher than False Positive rate.

Question 8:

	Feature Extraction		Dimension Reduction		Classifier	Performances
	min_df	Lemmatization vs Stemming	Method	n_components (k columns)		
1	5	Lemmatization	LSI	k=80	SVM	0.965079
2	5	Lemmatization	LSI	k=80	Logistic with L1	0.969841

3	5	Lemmatization	LSI	k=80	Logistic with L2	0.965079
4	5	Lemmatization	NMF	k=80	SVM	0.960317
5	3	Lemmatization	LSI	k=80	Logistic with L1	0.968254

Question 9:

Naive Bayes

	Score type	Naive Bayes Multi
0	Accuracy	0.742857
1	Recall	0.742857
2	Precision	0.740440
3	F1	0.735078

SVM (One VS One)

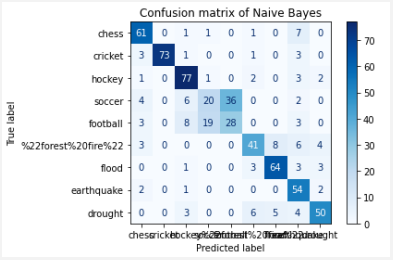
	Score type	SVM with One VS One
0	Accuracy	0.798413
1	Recall	0.798413
2	Precision	0.826654
3	F1	0.805437

SVM (One VS Rest)

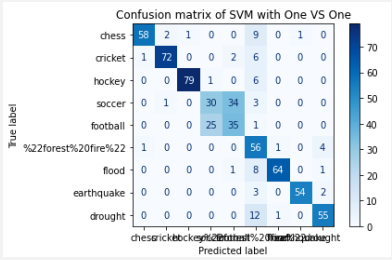
	Score type	SVM with One VS Rest
0	Accuracy	0.825397
1	Recall	0.825397
2	Precision	0.826378
3	F1	0.824860

Score summary on Multi Classification

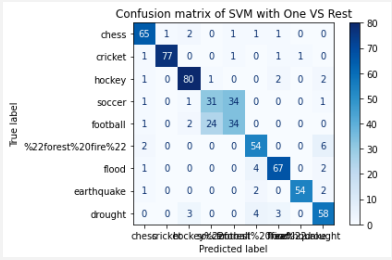
Naive Bayes



SVM (One VS One)



SVM (One VS Rest)



Confusion Matrix Comparison on Multi Classification

9.1 Confusion Matrix:

- Three confusion matrix plots are all 9*9 matrix. Each plot has a certain square that is dark blue which means there are higher misclassification rates inside the square. It indicates that three models are easily misclassified as “soccer” and “football”.

- There are distinct blocks on each major diagonal plot.
- It shows that all three models easily misclassify "soccer" and "football".

9.2 Merge:

	SVM (One VS One)	SVM (One VS Rest)
Before merge		

	Score type	SVM with One VS One
0	Accuracy	0.798413
1	Recall	0.798413
2	Precision	0.826654
3	F1	0.805437

	Score type	SVM with One VS Rest
0	Accuracy	0.825397
1	Recall	0.825397
2	Precision	0.826378
3	F1	0.824860

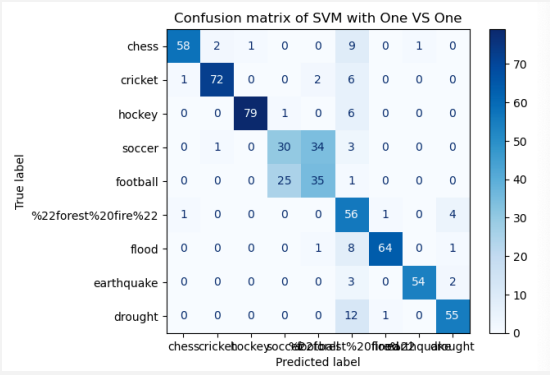
After merge

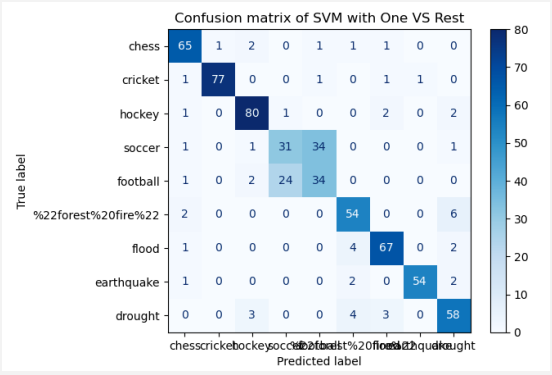
	Score type	SVM with One VS One (after merge)
0	Accuracy	0.892063
1	Recall	0.892063
2	Precision	0.919588
3	F1	0.899371

	Score type	SVM with One VS Rest (after merge)
0	Accuracy	0.917460
1	Recall	0.917460
2	Precision	0.918946
3	F1	0.917767

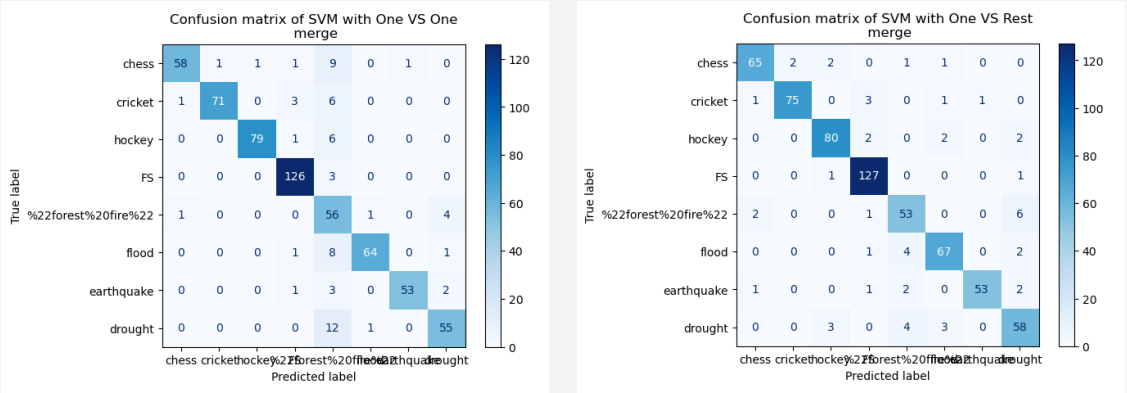
Score comparison on Multi Classification after merge

	SVM (One VS One)	SVM (One VS Rest)
Before merge		





After merge



Confusion Matrix Comparison on Multi Classification after merge

- Accuracy improved almost 10% compared with before merge labels.

9.3 Imbalance class:

- Class imbalance impacts the performance of classification after merging “soccer” and “football” labels. Before merge, SVM predictions on “soccer” and “football” are relatively accurate, which only has total 4 wrong in One VS One and total 3 wrong in One VS Rest. And it’s also easy to classify “drought” wrong. After merge, SVM is more accurate on classifying all other labels except the merge label, "FS". SVM classifies 7 and 8 wrong for One VS One and One VS Rest. Classification on other labels remains similar.
- To solve imbalance class, model One VS Rest model should add the following parameter:

class_weight = "balanced"

```
clf_multi_balance = OneVsOneClassifier(svm.SVC(kernel='linear',
random_state=42, class_weight = "balanced"),n_jobs=-1)

clf_balance_rest = OneVsRestClassifier(svm.SVC(kernel='linear',
random_state=42,class_weight = "balanced"),n_jobs=-1)
```

- Recompute accuracy and confusion matrix:

SVM (One VS One)

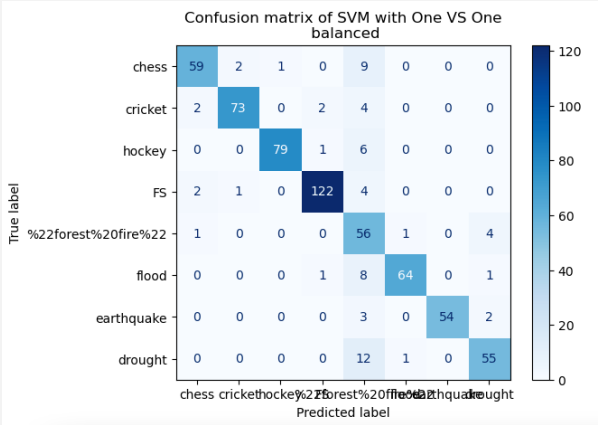
	Score type	SVM with One VS One (after balance)
0	Accuracy	0.892063
1	Recall	0.892063
2	Precision	0.917786
3	F1	0.899367

SVM (One VS Rest)

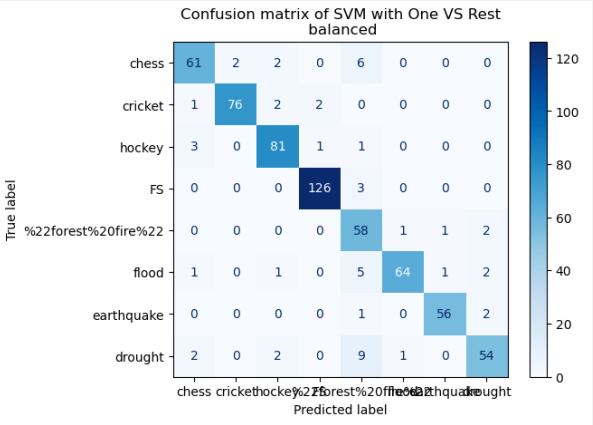
	Score type	SVM with One VS Rest (after balance)
0	Accuracy	0.914286
1	Recall	0.914286
2	Precision	0.922257
3	F1	0.915762

Score summary on Multi Classification after balanced

SVM (One VS One)



SVM (One VS Rest)



Confusion Matrix on Multi Classification after balanced

Question 10:

(a) Co-occurrence probabilities perform better than individual probabilities. The ratio is easier to distinguish relevant words from irrelevant words. It's also better to discriminate between two relevant words. Because the ratio of co-occurrence probabilities can help better distinguish relevant words from irrelevant words and can help distinguish one word from another given two relevant words

(b) Yes, because the co-occurence of other words in the context of running is same for both sentences. Running here has two different meanings. For “James is running in the park” , running means moving at a fast speed. But for the second sentence, “James is running for presidency”, running means persuading or competing in an election. But for GLoVE embedding, it cares about co-occurrence between word and word. One vector can only represent one meaning. If it wants to show more meaning, then it will need more vectors. The relationship between running and other words in two sentences are the same. One of the example could be as following:

$$\frac{P(k_1=park|running)}{P(k_1=park|James)} = \frac{P(k_2=presidency|running)}{P(k_2=presidency/James)}$$

(c) Comparison:

The value ||GLoVE["queen"] - GLoVE["king"] - GLoVE["wife"] +GLoVE["husband"]||2 should be highest since the analogy of queen to king to wife to husband is not as similar compared to the other two, and thus has the highest value. The value of ||GLoVE["queen"] - GLoVE["king"]||2 is the second highest because queen is similar to king but not as similar as the analogy of wife and husband The value of ||GLoVE["wife"] - GLoVE["husband"]||2 should be lowest because wife and husband are the most similar to each other.

(d) I would rather lemmatize the word before mapping it to GLoVE embedding because it is possible that stemming will cause same words of different form (like see and saw) to be considered as different words and this will affect the ratio of co-occurrence probabilities of a word in the context of another word. Lemmatization keeps more information on words. Then, GLoVE will process more informatively and output the train data.

Question 11:

(a) Description:

For each word in document's full text:

compute the similarity of the word and sport by: glove[word]-glove["sports"];

compute the similarity of the word and climate by: glove[word]-glove["climate"];

accumulate the similarity to represent the document's similarity to sports and the document's similarity to climate;

compare the document's similarity to sports and climate;

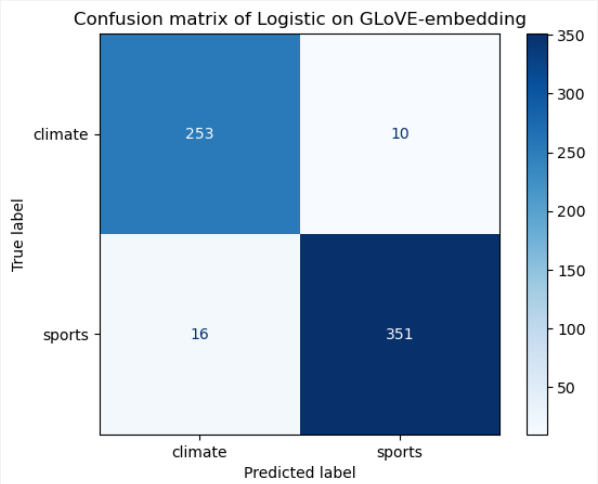
choose the more similar one and use the normalized vector that represents the similarity to represent the document.

(b) After the feature engineering process, we finally output a Glove_train data with 2520 samples which is the same as the original train data set.

(c) We select a Logistic Regression model to evaluate the GLoVE-based feature. We also apply cross validation to find the best parameters.

	Score type	GLoVE
0	Accuracy	0.958730
1	Recall	0.956403
2	Precision	0.972299
3	F1	0.964286

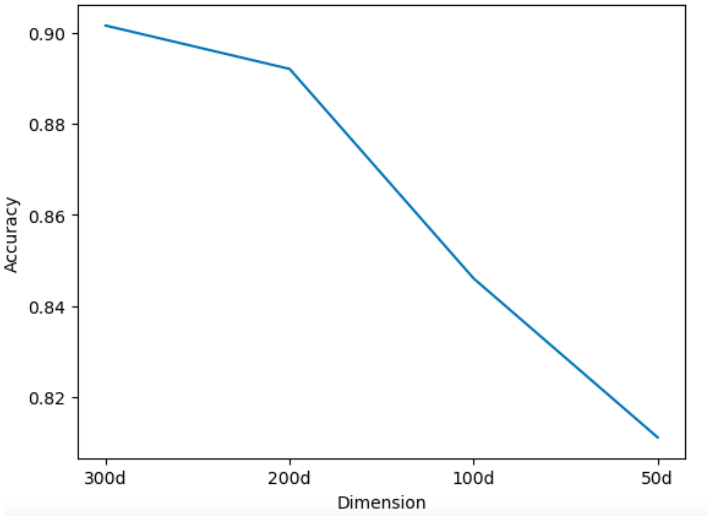
Score Summary under Logistic Regression with GLoVE methods



Confusion Matrix under Logistic Regression with GLoVE methods

Question 12:

- Plot of relationship between dimension and accuracy:



Plot of dimension and accuracy

- Observed trend:

  As dimensions decrease, accuracy goes down. So, its trend is decreasing.

- Expectation:

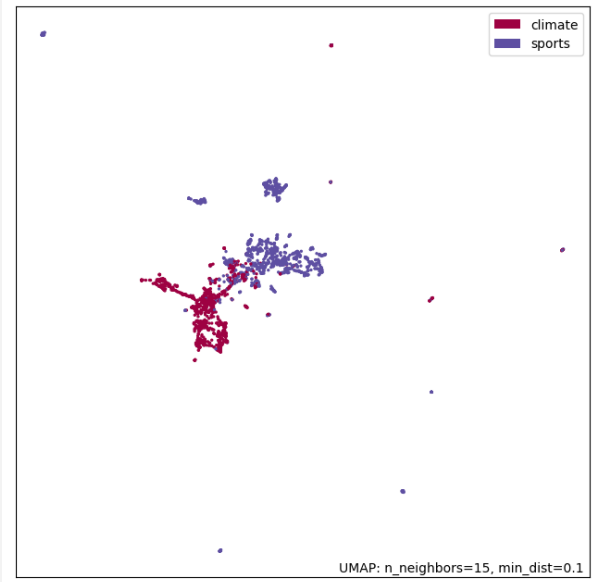
  Theoretically, as dimension decreases, accuracy of model also should decrease. So, the trend we plot is what we should expect.

- Reason/Comparison:

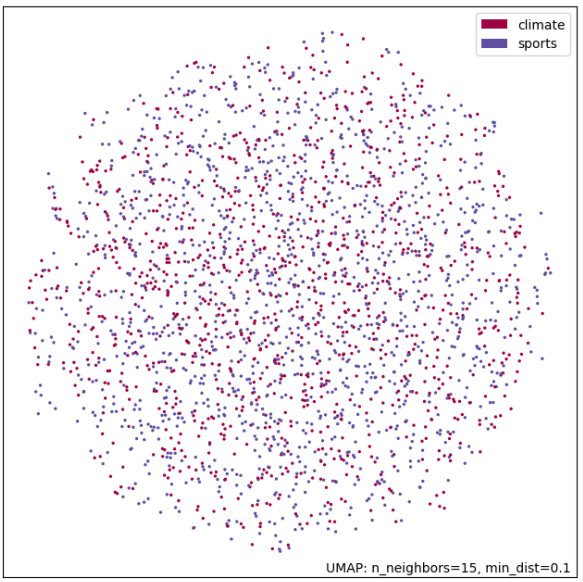
  GLoVE captures meaning in vector space like overall statistics of how often it appears by creating word vectors. When dimension is reduced, we lose more information. GloVe is training data that aggregates global word-word co-occurrence statistics for words and represents the word vector space.

Question 13:

GLoVE-based



Random vectors



Comparison of Two Vectors

- Cluster only formed on GLoVE-based vectors. It's two obvious categories in the left plot. Based on the two plots, we can see there is a huge difference. For GLoVE-based train data, they gather together more. It's easier for models to find a decision boundary and classify. For random vectors, it's really hard to find a certain pattern to cluster two classes.
- Meanwhile, we can see a few points on GLoVE-based are really far away from others which shows that when there is a large data set, it may have some disproportionate importance.