CS 464 Introduction to Machine Learning

Fall 2022

Homework 1

Ayşın Tümay

21902058

Section-2

Question 1.1:

$$P(H) = \frac{64}{100}, \quad P(L) = \frac{24}{100}, \quad P(F) = \frac{12}{100}$$

$$P(S_M|H) = \frac{87}{100}, \quad P(S_M|L) = \frac{21}{100}, \quad P(FS_M|F) = \frac{4}{100}$$

By the Total Probability law,

$$P(S_M) = \sum_{\text{İ}} P(S_M|i)P(i), \qquad i = H, L, F$$

$$= P(S_M|H)P(H) + P(S_M|L)P(L) + P(S_M|F)P(F)$$

$$= \frac{87}{100} \cdot \frac{64}{100} + \frac{21}{100} \cdot \frac{24}{100} + \frac{4}{100} \cdot \frac{12}{100}$$

$$P(S_M) = 0.6120$$

Question 1.2: By the Bayes Rule,

$$P(H|S_M) = \frac{P(H, S_M)}{P(S_M)} = \frac{P(S_M|H) \cdot P(H)}{\sum_{\text{İ}} P(S_M|i)P(i)}$$

$$P(H|S_M) = \frac{\frac{87}{100} \cdot \frac{64}{100}}{0.612} = 0.9098$$

Question 1.3: By the Total Probability Law and Bayes Rule,

$$P(S_U) = \sum_{\text{İ}} P(S_U|i)P(i), \qquad i = H, L, F$$

$$= P(S_U|H)P(H) + P(S_U|L)P(L) + P(S_U|F)P(F)$$

$$= (1 - P(S_M|H)) \cdot P(H) + P(1 - (S_M|L)) \cdot P(L) + (1 - P(S_M|F)) \cdot P(F)$$

$$= \frac{13}{100} \cdot \frac{64}{100} + \frac{79}{100} \cdot \frac{24}{100} + \frac{96}{100} \cdot \frac{12}{100}$$

$$= 0.388$$

$$P(H|S_U) = \frac{P(H, S_U)}{P(S_U)} = \frac{P(S_U|H) \cdot P(H)}{\sum_{\text{İ}} P(S_U|i)P(i)}$$

$$P(H|S_U) = \frac{\frac{13}{100} \cdot \frac{64}{100}}{0.388} = 0.2144$$

Question 2:

2.1.

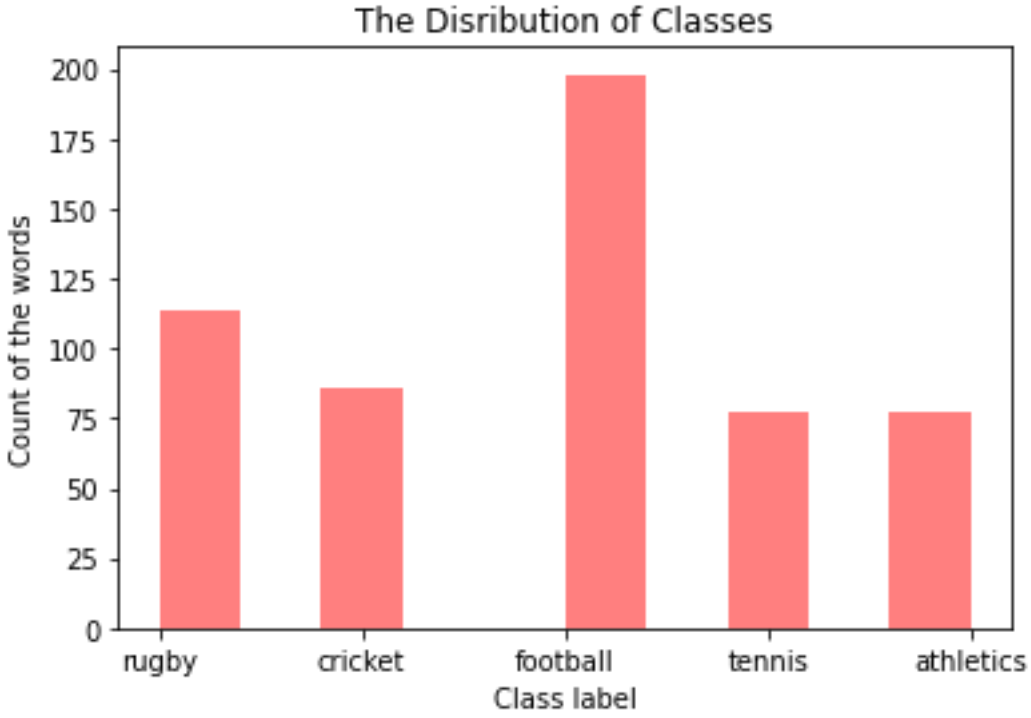

Fig. 1: The distribution of the training set classes.

2.2.

```
In [29]: train.class_label.skew(axis = 0)

Out[29]: -0.097310105999175
```

Fig.2: The skewness of the training set.

The class is skewed towards the left tail of the distribution, rugby. Skewness badly affects Naïve Bayes Classifier. The probabilities are assumed conditionally independent. However, when there is a distribution bias towards a class, this underestimates the dependencies between the classes. This may be tackled by calculating the divergence between the ideal and skewed distribution. Then, the divergence can be added to every class to eliminate the bias. This is exactly what we will perform by smoothing.

2.3.

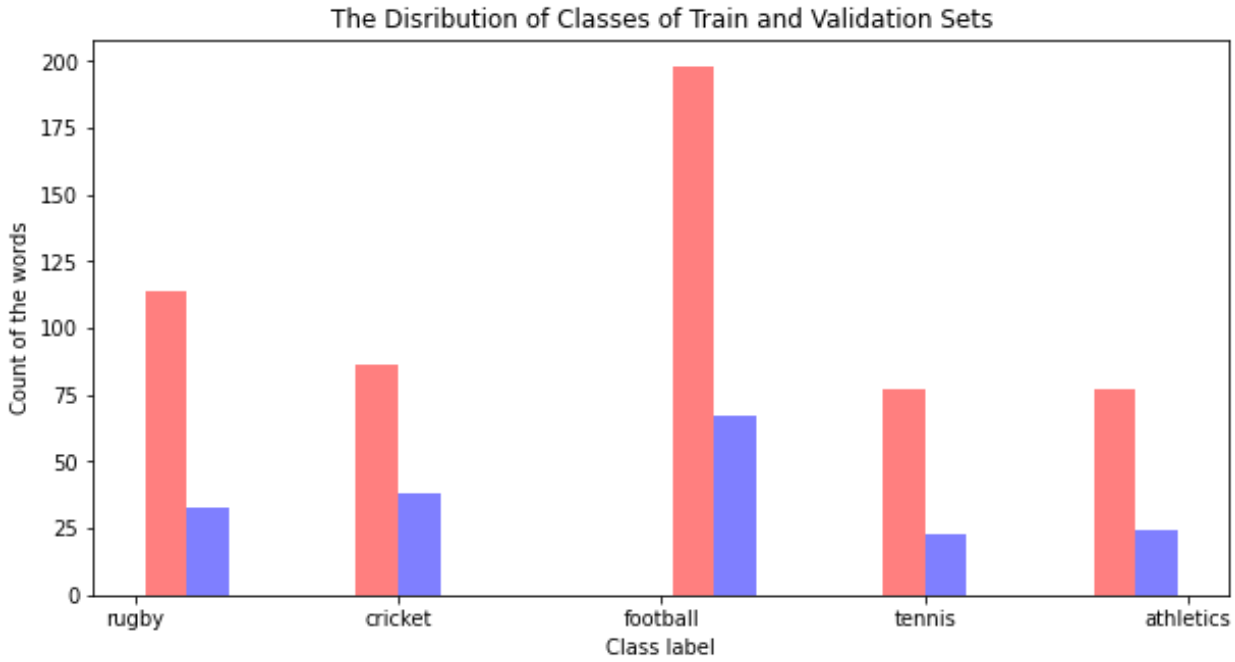

Fig.3: The distribution of the training and validation set classes.

```
In [47]: test.class_label.skew(axis = 0)
Out[47]: 0.053970839502950604
```

Fig.4: The skewness of the validation set.

Train and validation sets have different skewness values. However, the abundant class is still football while the sparce class is athletics. They do not have the same distribution, but they clearly have a similar distribution. If the training set distribution is totally different than the validation distribution, the validation score would be less than the training score.

	ground truth	predictions
0	4	0
1	2	0
2	1	0
3	2	0
4	4	0
...	...	...
180	2	0
181	0	0
182	4	0
183	2	0
184	2	0

185 rows × 2 columns

Fig.4: Confusion matrix of the MLE estimator.

2.4.

The Dirichlet prior improves the validation score. Since there are many -inf predictions due to the 0 occurrence of words, these values are evaluated as a very small numbers which is not very helpful for $\theta_{j|y=y_k}$. The accuracy of the MLE estimator without any smoothing resulted in 24.32% accuracy when -inf value is equated to np.nan_to_num(-np.inf).

In the MAP estimator with Dirichlet prior equal to 1, the accuracy is 97.30%. This great improvement is inherent in the bag-of-words implementation. The occurrences are placed as numbers in the dataset which means there are a lot of zeros, as seen in Fig.6. The words that occur rarely create noise. In this dataset, there were 16 words with 0 occurrences among all documents as seen in Fig.7. If the noise is included to the classification, it causes overfitting. Also, Fig.8 displays the mean of each word occurrence sorted in the descending order. As observed, the mean is at most 1.2 compared to the max values in Fig.9. Considering the statistics, it is a beneficial method to use smoothing so that the skewed distribution can be eliminated a bit.

	ground truth	predictions
0	4	4
1	2	2
2	1	1
3	2	2
4	4	4
...	...	...
180	2	2
181	0	0
182	4	4
183	2	2
184	2	2

Fig.5: Confusion matrix of the MAP estimator. column.

```
In [8]: (train==0).sum()

Out[8]: claxton        548
        hunt           546
        first          273
        major          495
        medal          523
                       ...
        fiveset        552
        mario          549
        ancic          549
        lundgren       549
        class_label     77
        Length: 4614, dtype: int64
```

Fig.6: The number of 0s in each word

```
In [24]: train.sum()[train.sum()==0]

Out[24]: 3000m         0
         raw           0
         sadli         0
         glamorgan     0
         bu            0
         condemn       0
         alec          0
         24th          0
         section       0
         centrehalf    0
         bullock       0
         sfa           0
         tag           0
         wigan         0
         bbc1          0
         fiveset       0
         dtype: int64
```

Fig.7: The words that have never occurred.

```
In [46]: train.mean().sort_values(ascending=False).head()

Out[46]: class_label    2.050725
         plai           1.260870
         game           1.181159
         win            1.023551
         player         0.987319
         england        0.961957
         first          0.952899
         against        0.869565
         year           0.806159
         time           0.742754
         dtype: float64
```

Fig.8: The mean of the occurrences of each word.

```
In [51]: train.max().sort_values(ascending=False)

Out[51]: roddick       53
         nadal         46
         zealand       32
         dallaglio     25
         point         25
                       ..
         sadli          0
         bu             0
         glamorgan      0
         wigan          0
         centrehalf     0
         Length: 4614, dtype: int64
```

Fig.9: The maximum number of occurrences of each word.