COMP5046: Natural Language Processing

Assignment 1: Annotation Task

Tanzil Kazi
Student ID: 470239029
Semester 1, 2018

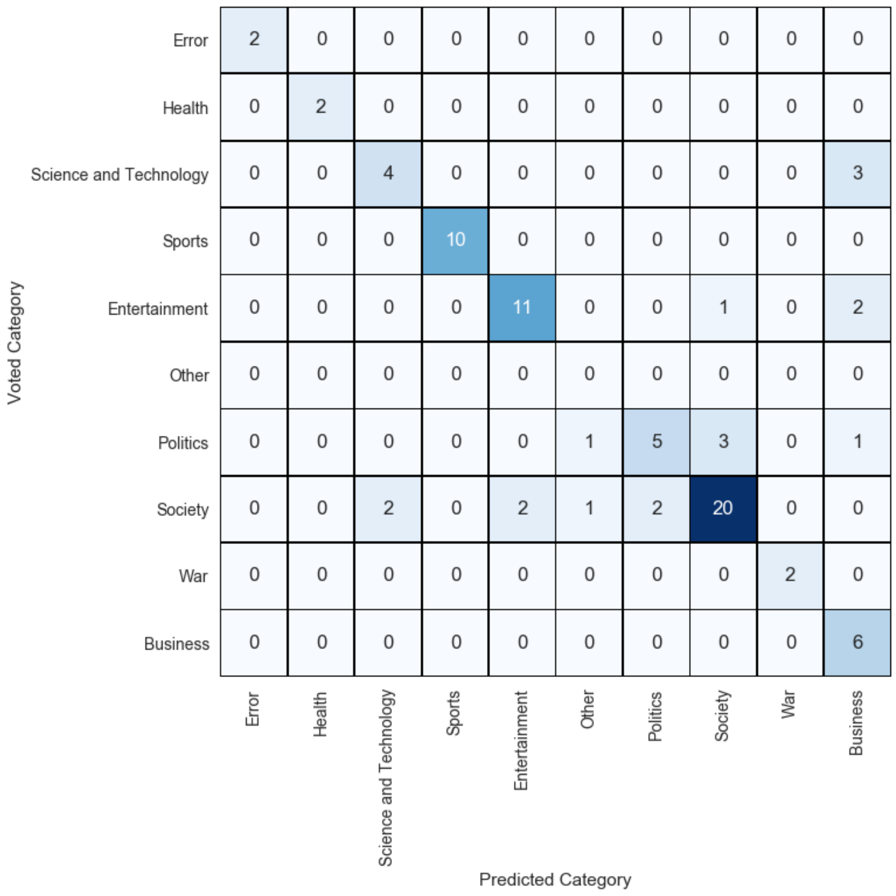
Contents

1. Results

80 articles have been annotated by 80 annotators (1 was removed as described in Section1.5) and the inter-annotator agreement statistics have been calculated in Python (see Appendix). The results are discussed below.

1.1. Confusion Matrix

The confusion matrix, shown below in *Figure 1*, compares the predicted category with the voted category for the 80 articles categorized for the annotation task. The voted category for each article was determined by selecting the category voted for the article by most annotators. The confusion matrix is an invaluable tool providing greater insights into the performance of a categorization task when compared to other performance metrics such as precision or accuracy. It can even expose relationships between categories.



Voted Category \ Predicted Category	Error	Health	Science and Technology	Sports	Entertainment	Other	Politics	Society	War	Business
Error	2	0	0	0	0	0	0	0	0	0
Health	0	2	0	0	0	0	0	0	0	0
Science and Technology	0	0	4	0	0	0	0	0	0	3
Sports	0	0	0	10	0	0	0	0	0	0
Entertainment	0	0	0	0	11	0	0	1	0	2
Other	0	0	0	0	0	0	0	0	0	0
Politics	0	0	0	0	0	1	5	3	0	1
Society	0	0	2	0	2	1	2	20	0	0
War	0	0	0	0	0	0	0	0	2	0
Business	0	0	0	0	0	0	0	0	0	6

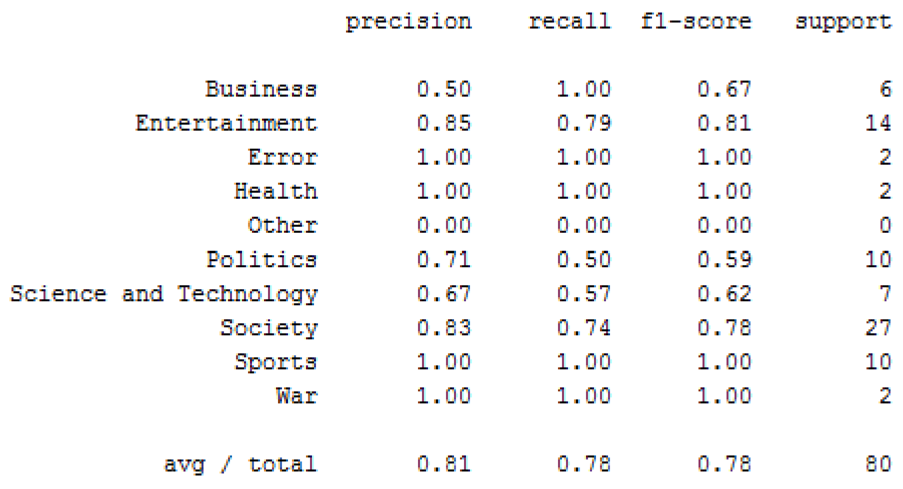
Figure 1: Confusion Matrix

The diagonal cells in the confusion matrix indicates the instances of agreement between the predicted category and the voted category – the larger the proportion of counts in the diagonal cells, the greater agreement exists for the particular category between annotators. From *Figure 1*, overall there seems to be some agreement between the predicted and voted annotations. Categories *Error*, *Health*, *Sports* and *War* only have counts in the diagonal cells indicating that all the predicted annotations are correct. This shows there are distinct patterns in the text (e.g. name of a sport or player, scoring details, events during a game etc.), which make it is relatively easy for humans to differentiate and successfully annotate articles for such categories.

The most common category selected is *Society*. The definition of society from the Oxford dictionary is "*The aggregate of people living together in a more or less ordered community*". Since society encompasses every aspect of human community, it is difficult for humans to differentiate (i.e. annotate) between society and topics such as politics and entertainment. This is evident from the confusion matrix. Even though there is strong agreement for the category *Society* (20 out of the 27 instances are in agreement), several articles on Society have been annotated as *Politics*, *Entertainment* and even *Science and Technology*. A similar story is witnessed with category *Business* – business as a topic can overlap with any other topic. Out of 12 articles, 6 have been predicted correctly but the other 6 articles are dispersed between *Science and Technology, Entertainment* and *Politics.*

1.2. Class-wise precision, recall and F-score

The class-wise precision, recall and F-score is shown in *Figure 2*.



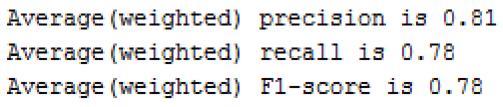
```
                        precision    recall  f1-score   support

              Business       0.50      1.00      0.67         6
         Entertainment       0.85      0.79      0.81        14
                 Error       1.00      1.00      1.00         2
                Health       1.00      1.00      1.00         2
                 Other       0.00      0.00      0.00         0
              Politics       0.71      0.50      0.59        10
Science and Technology       0.67      0.57      0.62         7
               Society       0.83      0.74      0.78        27
                Sports       1.00      1.00      1.00        10
                   War       1.00      1.00      1.00         2

           avg / total       0.81      0.78      0.78        80
```

Figure 2: Class-wise precision, recall and F-score

Precision, recall and F-score quantifies the information in a confusion matrix. Ideally, the aim in any annotation task is to maximize both precision and recall. In other use cases, it may be better to maximize either precision (e.g. in sentimental analyses of texts) or recall (e.g. fraud detection). Since F-score is the harmonic mean of precision and recall, annotation methods and practices should be tuned to maximizing F-score.

As mentioned in the previous section, categories *Error*, *Health*, *Sports* and *War* are all predicted accurately and this is reflected in a score of 1.0 for precision, recall and F-score for all the categories. Furthermore, as expected, category *Politics* has the lowest F-score, due to two factors – the difficulty for human annotators to definitely categorize articles in the category, and the low number of articles in the data set categorized as *Politics*. Category *Society* has a high F-score of 0.78 (even though several articles are predicted wrongly) because of the high support number. This highlights the importance of annotating large data sets; even though it is difficult to pinpoint the category of certain articles, having a large data set improves the quality of the annotation.

1.3. Averaged precision, recall and F-score



```
Average(weighted) precision is 0.81
Average(weighted) recall is 0.78
Average(weighted) F1-score is 0.78
```

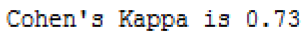
Figure 3: Overall precision, recall and F-score

Weighted-average is used to calculate the average statistics. The weighted average method is used since it takes into account the support number, which is the number of occurances of each category. Since the objective of the task is to perform inter-annotator agreement comparison, it is vital to take into account the support number since it has strong influence on the statistics of the category, as highlighted in the previous section. A single wrongly categorized article with low support number will have a much greater impact on the category's agreement statistics than for a category with a higher support number.

As shown in *Figure 3*, the average predicted precision is 0.81 i.e. 81% of all the predictions made are in agreement with the voted categories. From the average recall, 78% of predictions are in agreement with the true categories. The relatively high values of precision and recall is reflected in the F1-score of 0.78.

1.4. Cohen's Kappa

Cohen's Kappa measures the degree of agreement between two annotators. Cohen's Kappa is always less than or equal to 1.0, with 1.0 implying complete agreement. The resulting Cohen's Kappa for the predictions is 0.73 as shown in *Figure 4*. This indicates relatively good agreement between the predicted and the voted annotation. It is interesting to note that this reflects the observation made from the diagonal cells in the Section 1.1 (Page 2).



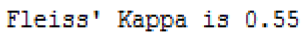
```
Cohen's Kappa is 0.73
```

Figure 4: Cohen's Kappa

1.5. Fleiss' Kappa

Fleiss' Kappa measures the degree of agreement between 80 annotators. In total, there are 81 annotators, but one of the annotators failed to categorize all the articles - in order to simplify the calculation, all votes of the annotator and the annotator was removed from all calculations.

Similar to Cohen's Kappa, Fleiss' Kappa values are less than or equal to 1.0, with 1.0 implying perfect agreement. As shown in *Figure 5*, a value of 0.55 is obtained indicating a fair amount of agreement between all the annotators. A fair amount of agreement is expected since most humans can broadly categorize articles – a more rigorous annotation schema and trained annotators will help to increase the agreement among the annotators.



```
Fleiss' Kappa is 0.55
```

Figure 5: Fleiss' Kappa

2. Discussion

2.1. Annotation task

Does it make sense to annotate a single category per article?

From the perspective of human intuition, assigning multiple categories to each article is not only easier but makes more practical sense. An article will always have a primary category, but categorizing an article into multiple sub-categories during annotation extracts more information from the articles which will lead to more sophisticated training models and allow for more granular classification. For example, an article discussing the performance of a new technology company can be categorized correctly as both *Business* and *Science and Technology*. In the annotation task, there were 3 instances (out of the 80) where articles were predicted as being in *Science and Technology* but were actually in *Business* (see *Figure 1*). Although having more categories per article will improve the performance of the classifier, implementation of such a multi-category classification method is more complex and time consuming. Extensive literature is available for single class classification techniques. Most multi-category classifier techniques combine such techniques which leads to greater complexity.

2.2. Annotation scheme

Does it make sense to increase the number granularity of categories?

The annotation schema supplied has 10 broad categories. A greater number of categories will allow human annotators to make more definite category assignments, leading to better training data and better classification models. This can be further improved by using a hierarchical schema relating the categories. For example, in the annotation task, the category *Society* can have sub-categories of *Entertainment* and *Politics* which will make it easier to differentiate between edge cases (several were noticed in *Figure 1*). One drawback of using more categories is more training data will be required to model the classifier. As seen in Section 1.2 (Page 3), more annotated data per category leads to greater classification performance for the category. Hence to build a strong classifier with more categories, more training data will be necessary.

2.3. Annotation tool

Did the tool make annotation efficient and accurate?

Google Spreadsheet is used as the annotation tool for the task. Google Spreadsheet is a web-based application which is available through a browser and is easy to use. The tool is sufficiently powerful for small to medium scale annotation tasks. A more dedicated annotation tool can be used as part of a classification pipeline where annotators can perform annotations and supporting functions will calculate inter-annotator agreement statistics leading to increased visibility and quality of annotations. Better annotations will always lead to more accurate training data.

2.4. Discussion and agreement

Did you find the discussions helpful?

The discussions were extremely valuable to build insight and to understand the nuances of the problem. Exploring the problem with the rest of the annotators expose different perspectives which would otherwise be missed.

3. Conclusion

In machine learning systems, accurate training data is critical to build high performing models. Correct training data can primarily be achieved by accurate annotation driven by a stringent annotation schema and trained annotators. Being a manual process, annotation is labourious and time consuming but the effort is justified. Due to the nature of the task, trained annotators are essential as they are able to develop the annotation process and schema depending on the specific problem at hand. Experienced annotators are able to improve and streamline the arduous process through discussion, collboration and prior experience.

4. Appendix

The following Python code was used to calculate the inter-annotator agreement statistics.

```
# COMP5046: Natural Language Processing
# Assignment 1: Annotation Task
# by Tanzil Kazi (ID: 470239029)

import pandas as pd
from collections import Counter
from sklearn import metrics as mets
import seaborn as sb
import matplotlib.pyplot as plot
import numpy as np
import warnings
warnings.filterwarnings("ignore")

# Import file from CSV to dataframe
anno_file = pd.read_csv("annotations.csv")
my_userID = "tkaz8954"

# Extract unique IDs (IDs are titles of articles), categories and userIDs
IDs = list(set(anno_file['ID']))
labels = list(set(anno_file['Annotation']))
userIDs = list(set(anno_file['Annotator']))

# voted_results consist of annotations' IDs and their true categories
voted_results = {}
predictions = {}

# Apply groupby to group Annotators and count the number of annotations, select annotation not
equal to 80
valid_annotator = anno_file.groupby(['Annotator']).count()['Annotation']
drop_annotators = valid_annotator[valid_annotator != 80].index.values

# Drop the annotators and create a new copy of the dataframe
for x in drop_annotators:
    drop_indexes = anno_file.loc[anno_file['Annotator'] == x].index.values
    anno_file = anno_file.drop(drop_indexes)

for ID in IDs:
    # Slices the dataframe for each title and counts the Annotation column
    label_counts = Counter(anno_file.loc[anno_file['ID'] == ID]['Annotation'])
    # Extract the most common label (which appears in the first element) and also extract it's
name
    true_label = label_counts.most_common()[0][0]
    voted_results[ID] = true_label

for index,row in anno_file.loc[anno_file['Annotator'] == my_userID].iterrows():
    predictions[row['ID']] = row['Annotation']

# Extract voted and predicted categories into the arrays - the arrays are aligned
voted_category = []
predicted_category = []
for key, value in voted_results.items():
    voted_category.append(value)
    predicted_category.append(predictions[key])

# Confusion Matrix
conf_matrix = mets.confusion_matrix(voted_category, predicted_category, labels=labels)
df_conf_matrix = pd.DataFrame(conf_matrix,index=labels,columns=labels)

fig = plot.figure(figsize=(6,6))
conf_map = sb.heatmap(df_conf_matrix,
annot=True,cmap="Blues",linewidths=0.5,linecolor="black",cbar=False,square=True)
plot.xticks(rotation=90)
plot.yticks(rotation=0)
plot.xlabel("Predicted Category")
plot.ylabel("Voted Category")

#Precision, recall, f1_score
class_rpt = mets.classification_report(voted_category, predicted_category)
print(class_rpt)

# Precision
precision = mets.precision_score(voted_category, predicted_category, average='weighted',
labels=labels)
print("Average(weighted) precision is {0:.2f}".format(precision))

# Recall
recall = mets.recall_score(voted_category, predicted_category, average='weighted',
labels=labels)
print("Average(weighted) recall is {0:.2f}".format(recall))

# F1-score
f_score = mets.recall_score(voted_category, predicted_category, average='weighted',
labels=labels)
print("Average(weighted) F1-score is {0:.2f}".format(f_score))

# Cohen's Kappa
ckappa = mets.cohen_kappa_score(voted_category, predicted_category, labels=labels)
print("\nCohen's Kappa is {0:.2f}".format(ckappa))

# Fleiss Kappa
# Create an index for the IDs - IDs are strings and can't be used as index in dataframe. The
indexes are 0-79.
ID_index = {}
ID_index_counter = 0
for ID in IDs:
    ID_index[ID] = ID_index_counter
    ID_index_counter = ID_index_counter + 1

# Create empty dataframe to store
df_fkappa = pd.DataFrame(columns=labels,index=list(range(len(IDs)))).fillna(0)

# For each subject (ID), extract all annotations and count them. Then insert the counts in
dataframe
for ID in IDs:
    annotation = anno_file.loc[anno_file['ID'] == ID]['Annotation']
    annotation_counts = annotation.value_counts()
    for label, count in annotation_counts.items():
        df_fkappa[label][ID_index.get(ID)] = count

# Calculating Fleiss' Kappa as per
https://en.wikipedia.org/wiki/Fleiss%27_kappa#Worked_example
sum_all_cells = df_fkappa.sum(axis=1).sum()
p = df_fkappa.sum(axis=0)/sum_all_cells

df_fkappa_sq = np.square(df_fkappa)
n_annotators = len(userIDs) - len(drop_annotators)
P = (df_fkappa_sq.sum(axis=1) - n_annotators)/(n_annotators*(n_annotators-1))

Pbar = sum(P)/len(IDs)
Pbar_e = np.sum(np.square(p))

fkappa = (Pbar-Pbar_e)/(1-Pbar_e)
print("\nFleiss' Kappa is {0:.2f}".format(fkappa))

# Display confusion matrix
plot.show()
```