Text Classification (Part II)

[DAT640] Information Retrieval and Text Mining

Krisztian Balog
University of Stavanger

August 26, 2019

Recap

- Problem of text classification
- Evaluation measures
- Preparing hold-out data for model development

Multiclass classification

- Imagine that you need to automatically sort news stories according to their topical categories

comp.graphics comp.os.ms-windows.misc comp.sys.ibm.pc.hardware comp.sys.mac.hardware comp.windows.x	rec.autos rec.motorcycles rec.sport.baseball rec.sport.hockey	sci.crypt sci.electronics sci.med sci.space
misc.forsale	talk.politics.misc talk.politics.guns talk.politics.mideast	talk.religion.misc alt.atheism soc.religion.christian

Table: Categories in the 20-Newsgroups dataset

Multiclass classification

- Many classification algorithms are originally designed for binary classification
- Two main strategies for applying binary classification approaches to the multiclass case
  - One-against-rest
  - One-against-one
- Both apply a voting scheme to combine predictions
  - A tie-breaking procedure is needed (not detailed here)

One-against-rest

- Assume there are k possible target classes $(y_1, \ldots, y_k)$
- For each target class
  - Instances that belong to y_i are positive examples
  - All other instances y_j, $j \neq i$ are negative examples
- Combining predictions
  - If an instance is classified positive, the positive class gets a vote
  - If an instance is classified negative, all classes except for the positive class receive a vote

Example

- 4 classes (y_1, y_2, y_3, y_4)
- Classifying a given test instance (dots indicate the votes cast):

y_1	$+$	●
y_2	-	
y_3	-	●
y_4	-	
Pred.	$+$	

y_1	-	●
y_2	$+$	
y_3	-	●
y_4	-	●
Pred.	-	

y_1	-	●
y_2	-	●
y_3	$+$	
y_4	-	●
Pred.	-	

y_1	-	●
y_2	-	●
y_3	-	●
y_4	$+$	
Pred.	-	

- Sum votes received: $(y_1$,●●●●), $(y_2$,●●), $(y_3$,●●), $(y_4$,●●)

One-against-one

- Assume there are k possible target classes $(y_1, \ldots, y_k)$
- Construct a binary classifier for each pair of classes (y_i, y_j)
  - $\frac{k \cdot (k-1)}{2}$ binary classifiers in total
- Combining predictions
  - The positive class receives a vote in each pairwise comparison

Example

- 4 classes (y_1, y_2, y_3, y_4)
- Classifying a given test instance (dots indicate the votes cast):

y_1	+	•
y_2	-	
Pred.	+	

y_1	+	•
y_3	-	
Pred.	+	

y_1	+	
y_4	-	•
Pred.	-	

y_2	+	•
y_3	-	
Pred.	+	

y_2	+	
y_4	-	•
Pred.	-	

y_3	+	•
y_4	-	
Pred.	+	

- Sum votes received: $(y_1$,••), $(y_2$,•), $(y_3$,•), $(y_4$,••)

Discussion

Question

How to evaluate multiclass classification?
Which of the evaluation measures from binary classification can be applied?

Evaluating multiclass classification

- Accuracy can still be computed as

$$ACC = \frac{\text{\#correctly classified instances}}{\text{\#total number of instances}}$$

- For other metrics
  - View it as a set of k binary classification problems (k is the number of classes)
  - Create confusion matrix for each class by evaluating "one against the rest"
  - Average over all classes

Confusion matrix

		Predicted				
		1	2	3	$\dots$	k
Actual	1	**24**	0	2		0
	2	0	**10**	1		1
	3	1	0	**9**		0
	$\dots$					
	k	2	0	1		**30**

Binary confusion matrices, one-against-rest

		Predicted				
		1	2	3	...	k
Actual	1	**24**	0	2		0
	2	0	**10**	1		1
	3	1	0	**9**		0
	...					
	k	2	0	1		**30**

$\Rightarrow$

		Predicted	
		1	$\neg 1$
Act.	1	TP=24	FN=3
	$\neg 1$	FP=2	TN=52

		Predicted	
		2	$\neg 2$
Act.	2	TP=10	FN=2
	$\neg 2$	FP=0	TN=69

...

For the sake of this illustration, we assume that the cells which are not shown are all zeros.

Averaging over classes

- Averaging can be performed on the instance level or on the class level
- **Micro-averaging** aggregates the results of individual instances across all classes
  - All instances are treated equal
- **Macro-averaging** computes the measure independently for each class and then take the average
  - All classes are treated equal

Micro-averaging

- **Precision**

$$P_\mu = \frac{\sum_{i=1}^{k} TP_i}{\sum_{i=1}^{k}(TP_i + FP_i)}$$

- **Recall**

$$R_\mu = \frac{\sum_{i=1}^{k} TP_i}{\sum_{i=1}^{k}(TP_i + FN_i)}$$

- **F1-score**

$$F1_\mu = \frac{2 \cdot P_\mu \cdot R_\mu}{P_\mu + R_\mu}$$

		predicted	
		i	$\neg i$
actual	i	TP_i	FN_i
	$\neg i$	FP_i	TN_i

Macro-averaging

- **Precision**

$$P_M = \frac{\sum_{i=1}^{k} \frac{TP_i}{TP_i + FP_i}}{k}$$

- **Recall**

$$R_M = \frac{\sum_{i=1}^{k} \frac{TP_i}{TP_i + FN_i}}{k}$$

- **F1-score**

$$F1_M = \frac{\sum_{i=1}^{k} \frac{2 \cdot P_i \cdot R_i}{P_i + R_i}}{k}$$

  - where P_i and R_i are Precision and Recall, respectively, for class i

		predicted	
		i	$\neg i$
actual	i	TP_i	FN_i
	$\neg i$	FP_i	TN_i

Discussion

Question

In which cases should micro- or macro-averaging be preferred over the other?
What are the caveats?

Exercise #1 (paper-based)

Compute both micro- and macro-averaged Accuracy, Precision, Recall, and F1-score for a multiclass classifier that made the following predictions

Id	Actual	Predicted
1	1	1
2	1	1
3	2	1
4	2	2
5	2	3
6	3	2
7	3	3
8	3	1
9	3	3
10	4	4
11	4	2
12	4	3

Exercise #2 (coding)

Implement the computation of Accuracy, and both micro- and macro-averaged Precision, Recall, and F1-score in Python