Concepts of Text Classification

CHIEN CHIN CHEN

Introduction (1/4)

Text classification is the task of assigning a textual object (e.g., a document) to a class.

Is not new … we are dealing with text classification every day.

- Every morning, you wake up …
  - Examine the GF/BF messages you missed last night <u>to know if she/he is happy</u>
  - Check emails and <u>move them to different categories</u>
  - Reading finance news <u>to predict up or down</u> of stock market

In these cases … **you are a classifier**.

- **Trained** by all the emails, messages, news you've read and experienced.

Introduction (2/4)

If we can do classification on our own **WHY SHOULD WE LEARN THIS TOPIC?**

You may not be a good classifier …
◦ You probably mis-classified a lot of your GF/BF's messages.
◦ You definitely cannot handle too many texts.
  ◦ Say that more that 500 emails per day.

Hence, we need an agent (classification model) to do classification

Introduction (3/4)

Classification is a **classic** data mining problem, and can be applied to various data and domains:

- ◦ Images – image classification
- ◦ Human Resource records – churn prediction
- ◦ Atmosphere data – weather forecast

Here, we focus on **text** classification which is applicable to different business tasks:

- ◦ Customer service message – knowing which service
- ◦ Product reviews – sentiment analysis (positive or negative)

Introduction (4/4)

We first introduce the basis of text classification.
- Definition, procedure of model construction, evaluation, and so on.

Then, we learn and **PRACTICE** well-known classification methods:
- Naïve Bayes method
- Rocchio method
- K nearest neighbor
- Support vector machine

Definition (1/2)

Text classification is a type of **supervised learning**.

- A **supervisor** (**oracle**) serves as a **teacher** directing the learning process.
  - Defining the classes and labeling training documents.

Consist of two phases:

Training Phase

Training Data

Validation Data

Testing Phase

test doc

test doc
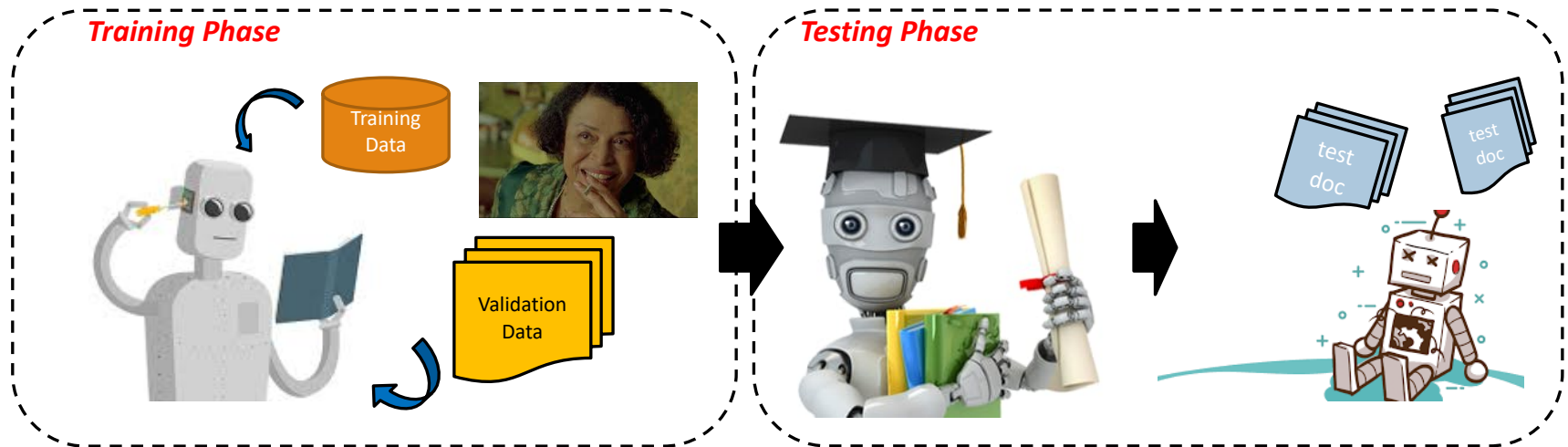
Definition (2/2)

Definition:

- **Document space *X***:
  - High-dimensional space.
  - Dimensions – terms.

- **Classes *C***:
  - Also called *categories* or *labels*.
  - $C = \{c_1, c_2, ..., c_J\}$.
  - **A fixed set**, <u>defined by human experts</u> for the needs of an application.
  - E.g., $C = \{\text{positive, negative,}\}$.

- **Training set *D***:
  - $D = \{<d, c>\}$, where $<d, c> \in X \times C$.
  - E.g., $<d, c> = <$'I don't like the room service …', negative$>$

Training Phase

A **_learning algorithm_** **Γ** is adopted to learn a **_classifier_**

(classification model) γ from the **training data** D.

- ◦ The learning method **Γ** takes the training set D as input and return the learned classifier γ.

- ◦ Mathematically, $\Gamma(D) = \gamma$

Γ can be _Naïve Bayes method_, _SVM algorithm, KNN …_

**different learning algorithms
different classification behavior**

Testing Phase (1/2)

Once we obtain an **appropriate** classifier γ, we apply it to new document and predict the document's class

- Mathematically, the classifier γ acts a function that $\gamma : X \rightarrow C$

In many cases, we also need to report the performance of the classifier.

- In addition to training data, we need **test data** to evaluate model performance.

Testing Phase (2/2)

Test set (or test data)
- Evaluate a model's performance.
- Chosen **independently** of the training data!!
  - To know how good (generalized) the classifier is, when dealing with **unseen** data.
- Otherwise, classification performance is **overestimated**.

考 試 前 ：

背之呼吸 記憶吐司

Remember … when starting to train/test with labeled data
- Always separate data into <u>a training portion</u> and <u>a testing portion</u>.
- The testing data is normally a small percentage (5~10%) of the total data.

Validation Phase (1/2)

Sometimes, **parameter tuning** is required if there are model parameters.
- ◦ Classification performance depends on parameter settings.
- ◦ E.g., assign (classify) a document to a category against a **similarity threshold**.

IT IS WRONG to tune model parameters to maximize performance on the test data!!
- ◦ This way, again, performance of the model is overestimated.

We require one data set called *validation set* (or development test collection).
- ◦ Tune model parameters against the validation dataset.
- ◦ And report the performance results on the test dataset in an unbiased manner.

Validation Phase (2/2)

The validation data <u>needs to be **independent** of both the training data and the testing data</u>.

◦ Normally, we collect 10% of the training data to form the validation data.

◦ Again, to keep training data as big as possible!!

In practice ... once the evaluation process is done, the validation and testing data can be bundled back into the training data to produce a new model.

◦ To maximize the amount of data used to generate models.

K-Fold Cross Validation (1/3)

The amount of data that can be used for training and testing is limited.

◦ So, it is possible that the sample used for training (or testing) may not be representative!!

K-fold cross-validation:

◦ A frequently-used approach to obtain reliable performance results.

◦ Decide on a fixed number (i.e., *k*) of ***folds*** and **randomly** split the data into **approximately equal partitions**.

◦ Each partition **in turn** is used for testing while the remainder is used for training.

   ◦ Every labeled instance has been used exactly once for testing.

◦ *K* is 10, usually.

K-Fold Cross Validation (2/3)

Note that sometimes people use *K*-fold cross validation for **parameter tuning**.

First, split data into training and testing portions.

Training:
- Divide **training data** into *K* folds.
- For each parameter setting, average the K-fold validation results.
- Compare the performances of different settings to determine the final parameter values.
- Re-train the model using the whole training data and the parameter values.

Testing:
- Apply the new model to the testing data and report the final performance result.

The computation cost of this procedure is very very very high …

K-Fold Cross Validation (3/3)
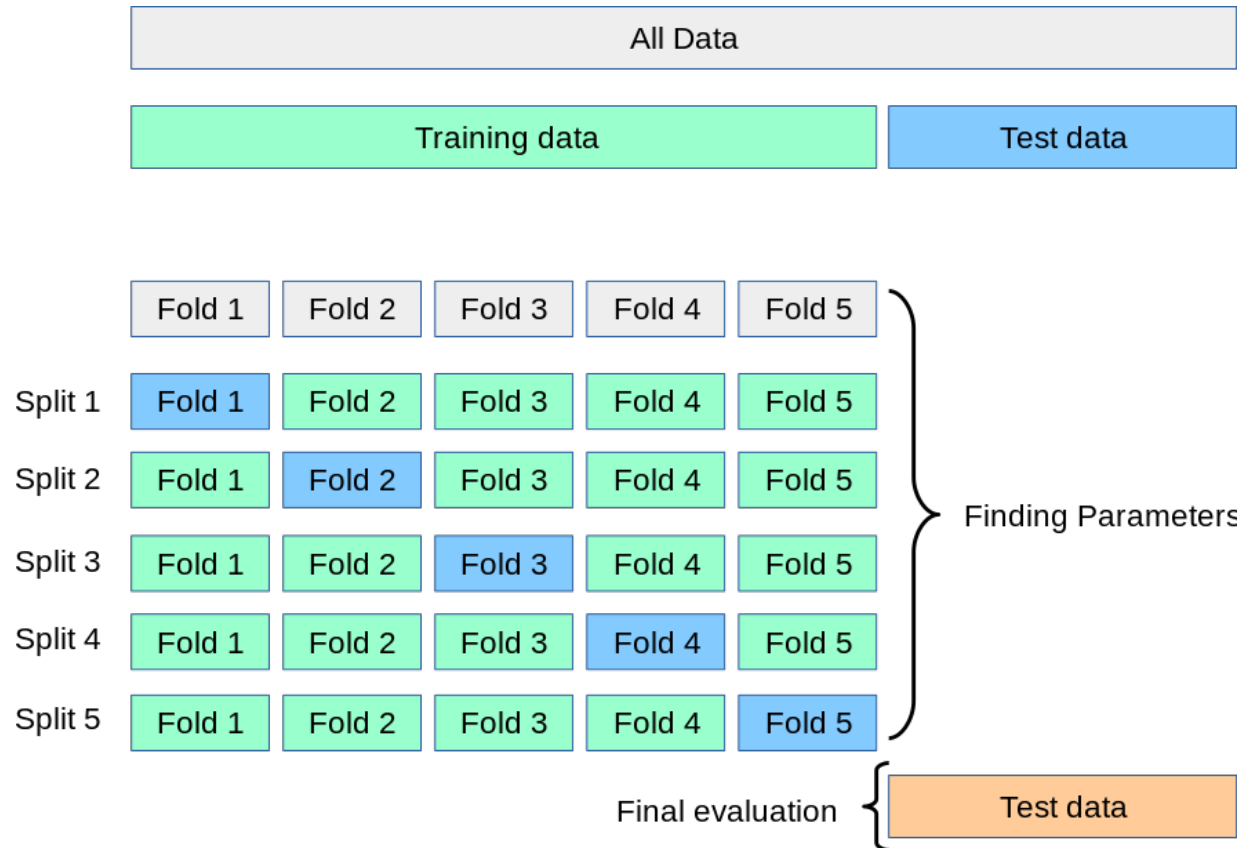
https://scikit-learn.org/stable/modules/cross_validation.html

Performance Metrics (1/12)

Three important metrics: **precision, recall, and F1**

Example:

	class 1	
	truth yes	truth no
model yes	# of true positives (tp)	# of false positives (fp)
model no	# of false negatives (fn)	# of true negatives (tn)

- Precision = tp / (tp+fp)
- Recall = tp / (tp+fn)

	class 1	
	truth yes	truth no
model yes	15	5
model no	10	970

- Precision: 15/20 = 0.75
- Recall : 15/25 = 0.6

Performance Metrics (2/12)

Different circumstances prefer different metrics:
- High precision: classify spam mails.
- High recall: Identify negative reviews about a certain product.

Generally, precision and recall trade off against one another.
- That makes model comparisons harder!!
- Is that fair to say "my precision is better than your recall"??

To compare models in a fair manner, we need to average them into a single value.
- F_1 – the **harmonic mean** of precision and recall
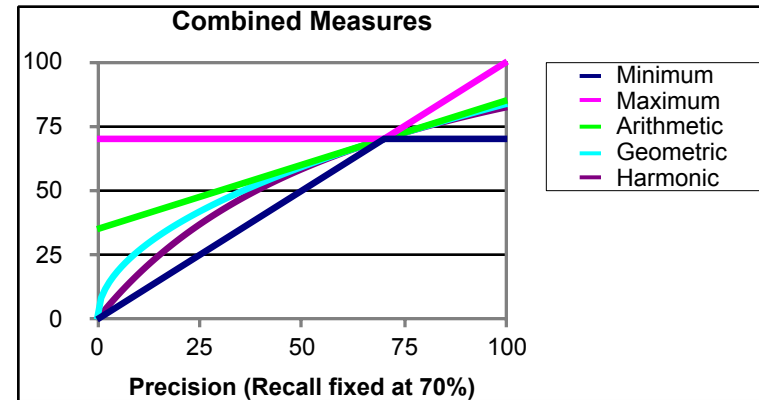
Performance Metrics (3/12)

$$F_1 = \cfrac{1}{(0.5\frac{1}{P} + 0.5\frac{1}{R})}$$

$$= \cfrac{2PR}{(P + R)}$$

Why F_1 measure?

◦ F1 is a **conservative** approach, and consider the two measures.

Note that precision, recall, and F1 are all between 0 and 1.

◦ The higher the better.

Combined Measures

Precision (Recall fixed at 70%)

Legend: Minimum, Maximum, Arithmetic, Geometric, Harmonic

The F_1 measure is often quite **close to the minimum** of the two numbers.

Performance Metrics (4/12)

In data mining, one popular metrics is **accuracy**

- accuracy = (tp + tn) / (tp+fp+fn+tn)

	class 1	
	truth yes	truth no
model yes	# of true positives (tp)	# of false positives (fp)
model no	# of false negatives (fn)	# of true negatives (tn)

Accuracy sometimes is **not** a good metrics if data is so skew...

- E.g. classifying spam mail.

	class 1	
	truth yes	truth no
model yes	0	0
model no	25	975

- In this case, the accuracy of the mode is:

$$(975) / (0+0+25+975) = \textbf{0.975}$$

- A model that always says no can achieve 0.975 accuracy rate!!

Performance Metrics (5/12)
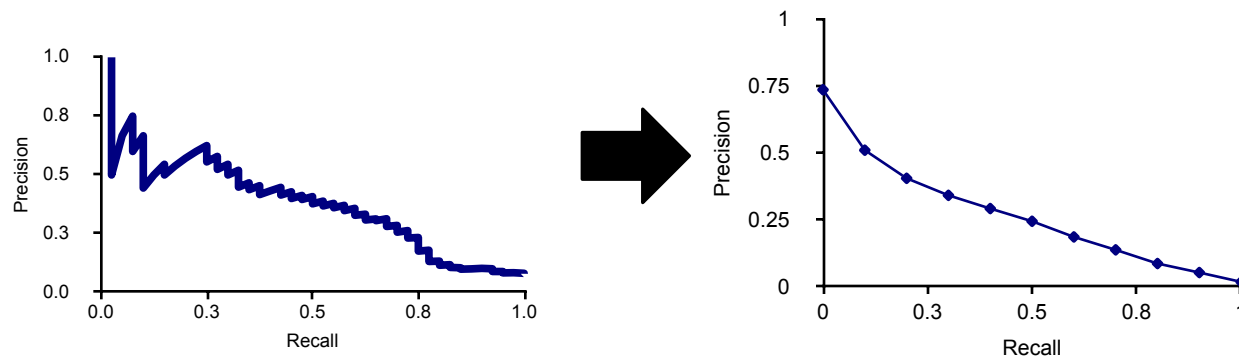
Classification decisions are usually with a score.

◦ For instance, Naïve Bayes classification model predicts the probability that a document belongs to a class.

Performance curves can be plotted by ranking the classification decisions according to their scores.

◦ *Precision-recall curves*
◦ *ROC curves*

Performance Metrics (6/12)

Precision-recall curve

◦ Often have a saw-tooth shape, and hard to read (compare).

11-point (interpolated average) precision-recall curve:

◦ The interpolated precision is measured at the 11 recall levels of 0.0, 0.1, 0.2, …, 1.0.

◦ For each recall level, **arithmetic mean** of the interpolated precision is calculated.
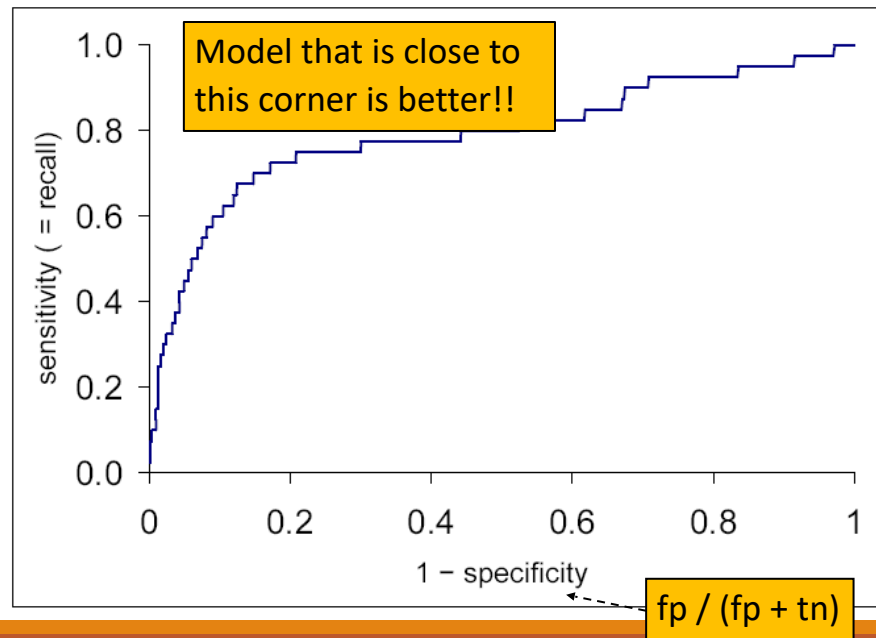
Performance Metrics (7/12)

11-point precision-recall curve is useful to model comparisons

Model that is close to this corner is better!!
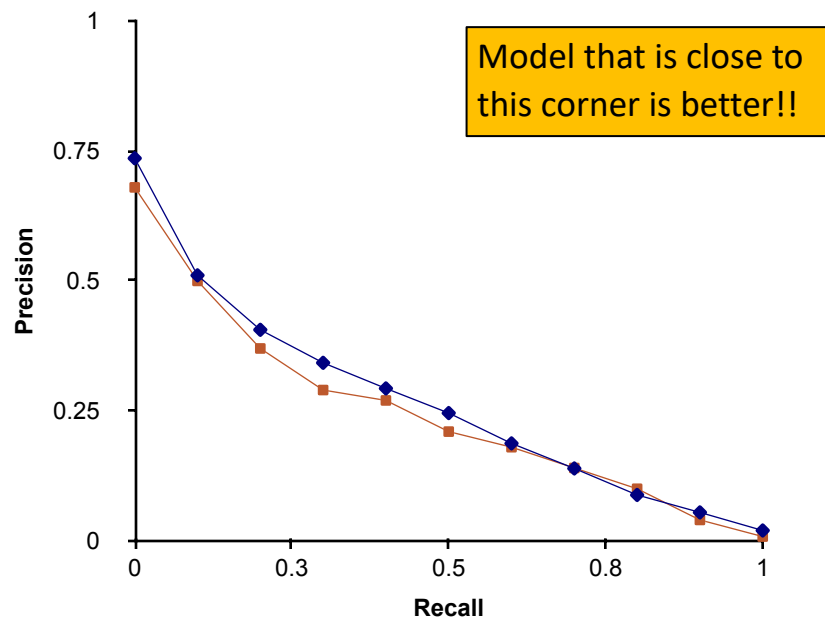
ROC curve:

◦ Plot the <u>true positive rate (recall)</u> against the <u>false positive rate</u>.

◦ **AUC** (Area Under the Curve) is also a measure for model comparisons

  ◦ The range is [0, 1]

Model that is close to this corner is better!!

fp / (fp + tn)

Performance Metrics (9/12)

Multi-class classification:
◦ Many text classification tasks are multi-class.
  ◦ E.g., news documents classification – sports, entertainment, stock, …

Confusion Matrix: a tool for analyzing a classification model under multi-class classification problems.

true class \ assigned class	money-fx	trade	interest	wheat	corn	grain
money-fx	95	0	10	0	0	0
trade	1	1	90	0	1	0
interest	13	0	0	0	0	0
wheat	0	0	1	34	3	7
corn	1	0	2	13	26	5
grain	0	0	2	14	5	10

Indicate how many documents from c_x were **incorrectly** assigned to c_y.

Help know the strength/weakness of a model

Performance Metrics (10/12)

For model comparisons, we often want to compute a single score the combines the results of different classes.

Two kinds of average for doing this:

- ***Macro-averaging***:
  - Compute a simple average over classes.
  - E.g., *precision$_{avg}$* = 1/*m* (*precision$_1$*+*precision$_2$*+…+*precision$_m$*).

- ***Micro-averaging***:
  - First aggregation the decisions across classes.
  - Then compute a measure on the pooled contingency table.

Performance Metrics (11/12)

Example:

	class 1	
	truth yes	truth no
model yes	10	10
model no	10	970

Precision: 0.5
Recall: 0.5

	class 2	
	truth yes	truth no
call yes	90	10
call no	10	890

Precision: 0.9
Recall: 0.9

Macro-averaging precision: (0.5+0.9)/2=0.7
Macro-averaging recall: (0.5+0.9)/2=0.7

Micro-averaging precision: 100/(100+20)=0.833
Micro-averaging recall: 100/(100+20)=0.833

	pooled table	
	truth yes	truth no
call yes	100	20
call no	20	1860

Performance Metrics (12/12)

The differences between micro and macro can be large!!

In micro-averaging, <u>large classes usually dominate small classes</u>.

◦ For example, the micro-averaging precision of the last example is 0.833, much closer to the precision of class 2.

To get a sense of effectiveness on small classes, you should compute macro-averaged results.

One/Any-of Classification (1/2)

Classification involved more than two classes can be **one-of** or **any-of**!!

Any-of classification – a document can belong to several classes, a single class, or none!!

◦ Training:
  - ◦ Build a two-class classifier **γ_j for each class** c_j, that returns either c_j or $not\text{-}c_j$.
  - ◦ Training set consists of the set of documents in C (positive) and its complement (negative).
◦ Testing:
  - ◦ Given the test document, apply each classifier separately.

One/Any-of Classification (2/2)

One-of classification – a document must belong to **exactly one** of the classes.

Many classification models can handle multi-class problem directly.

Some classification methods can only produce binary classifiers.
- Training:
  - Build a classifier for each class.
  - The training set consists of the set of documents in the class (positive) and its complement (negative).
- Testing:
  - Apply the test document to each classifier separately.
  - Assign the document to the class with the maximum score.