EE2211 Pre-Tutorial 10

Dr Feng LIN
feng_lin@nus.edu.sg

Agenda

- Recap
- Self-learning
- Tutorial 10

Today's Attendance

Recap

- Dataset Partition:
  - Training/Validation/Testing
- Cross Validation
- Evaluation Metrics
  - Evaluating the quality of a trained machine learning system
  - Mean Square Error
  - Mean Absolute Error
  - Confusion Matrix
  - Cost Matrix

Training, Validation, Test

Training set	Validation set	Test set
For training the ML models	For validation: choosing the parameter or model	For testing the “real” performance and generalization

Example of hyper-parameters, which are set before the training process begins:

- Order of polynomial
- Regularization parameters (λ)
- Tree depth (decision trees, random forests)
- Number of trees (random forests)
- ..

K-fold Cross Validation

- In practice, we do the **k-fold cross validation**

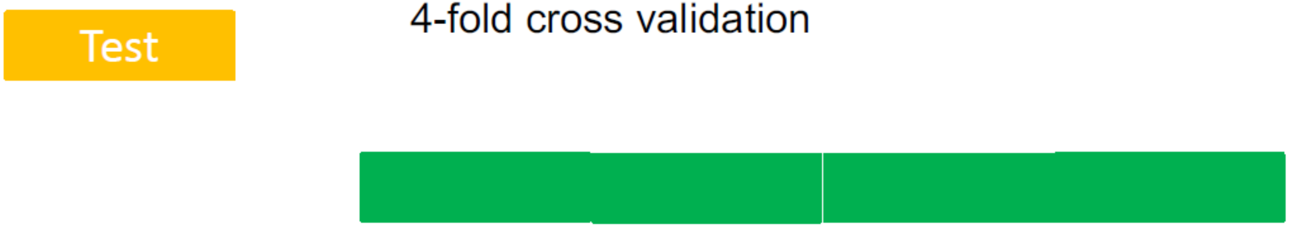
4-fold cross validation

Test

Step 1: take out *test set* from the dataset

K-fold Cross Validation

- In practice, we do the **k-fold cross validation**



Step 2: We partition the ***remaining part of the dataset*** (after taking out the test set), into ***k*** equal parts (equal in terms of number of samples).

K-fold Cross Validation

- In practice, we do the **k-fold cross validation**

Test

4-fold cross validation

Fold 1	Train	Train	Train	Validation
Fold 2	Train	Train	Validation	Train
Fold 3	Train	Validation	Train	Train
Fold 4	Validation	Train	Train	Train

Order of the samples are kept the same across all folds.

Step 3: We run ***k folds*** (i.e., k times) of experiments.
Within each fold, we use ***one part*** as ***validation set***, and the ***k-1 remaining parts*** as ***training set***. We use different validation sets for different folds.

K-fold Cross Validation

- In practice, we do the **k-fold cross validation**

Test

4-fold cross validation

					Classifiers Trained
Fold 1	Train	Train	Train	Validation	C_1^1 C_2^1
Fold 2	Train	Train	Validation	Train	C_1^2 C_2^2
Fold 3	Train	Validation	Train	Train	C_1^3 C_2^3
Fold 4	Validation	Train	Train	Train	C_1^4 C_2^4

1) C_1: Random Forest with 100 trees
2) C_2: Random Forest with 200 trees

Step 3.1: Within each fold, if we have n parameter/model candidates, we will train n models, and we check their validation performance.

K-fold Cross Validation

- In practice, we do the **k-fold cross validation**

Test

4-fold cross validation

					Classifiers Trained
Fold 1	Train	Train	Train	Validation	C_1^1 C_2^1
Fold 2	Train	Train	Validation	Train	C_1^2 C_2^2
Fold 3	Train	Validation	Train	Train	C_1^3 C_2^3
Fold 4	Validation	Train	Train	Train	C_1^4 C_2^4

Example: which one to select for test?

	Fold 1 Accuracy on Validation Set 1	Fold 2 Accuracy on Validation Set 2	Fold 3 Accuracy on Validation Set 3	Fold 4 Accuracy on Validation Set 4	Average Accuracy on All Validation Sets
Classifier with Param1 (e.g. 100 trees)	88% C_1^1	89% C_1^2	93% C_1^3	92% C_1^4	90.5%
Classifier with Param2 (e.g. 200 trees)	90% C_2^1	88% C_2^2	91% C_2^3	91% C_2^4	90%

Step 4: We select the parameter/model with best average validation performance over k folds.

Evaluation Metrics

Regression

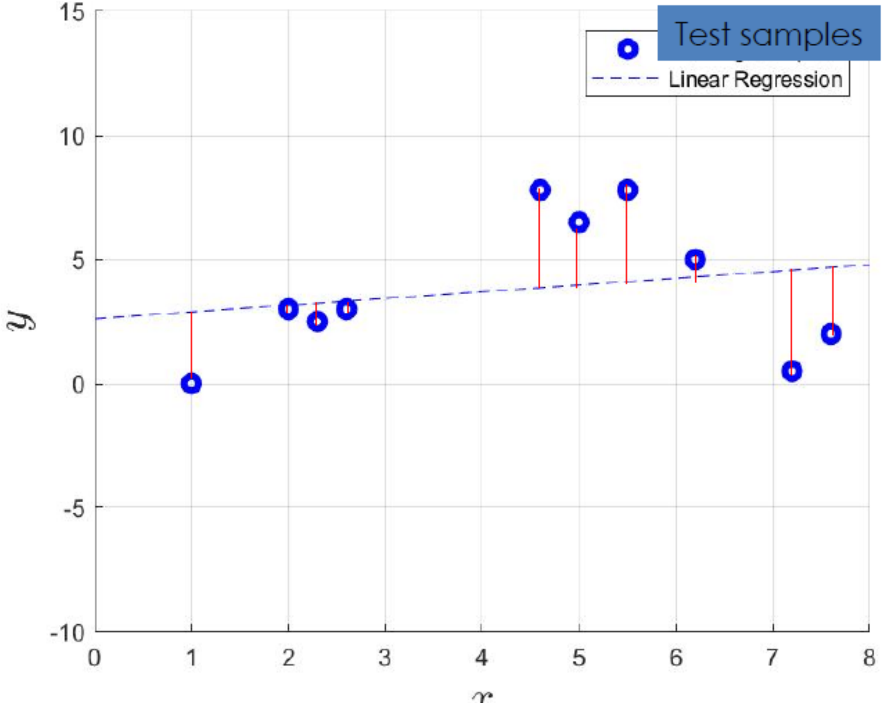
Mean Square Error

$$(\text{MSE} = \frac{\Sigma_{i=1}^{n}(y_i - \hat{y}_i)^2}{n})$$

Mean Absolute Error

$$(\text{MAE} = \frac{\Sigma_{i=1}^{n}|y_i - \hat{y}_i|}{n})$$

where y_i denotes the target output and $\hat{y}_i$ denotes the predicted output for sample i.



Evaluation Metrics

Classification

(True Positive Rate) TPR = TP/(TP+FN) Recall
(False Negative Rate) FNR = FN/(TP+FN)

(True Negative Rate) TNR = TN/(FP+TN)
(False Positive Rate) FPR = FP/(FP+TN)

TPR + FNR = 1 (100% of positive-class data)
TNR + FPR = 1 (100% of negative-class data)

Confusion Matrix for Binary Classification

	$\hat{P}$ (predicted)	$\hat{N}$ (predicted)	
P (actual)	TP	FN	Recall TP/(TP+FN)
N (actual)	FP	TN	
	Precision TP/(TP+FP)		Accuracy (TP+TN)/(TP+TN+FP+FN)

Evaluation Metrics

Classification

Cost Matrix for Binary Classification

	$\hat{\mathrm{P}}$ (predicted)	$\hat{\mathrm{N}}$ (predicted)
P (actual)	$C_{p,p}$ * TP	$C_{p,n}$ * FN
N (actual)	$C_{n,p}$ * FP	$C_{n,n}$ * TN

Total cost:
$C_{p,p}$ * TP +
$C_{p,n}$ * FN +
$C_{n,p}$ * FP +
$C_{n,n}$ * TN

Main Idea: To assign different ***penalties*** for different entries. Higher penalties for more severe results. Smaller costs are preferred.

Usually, $C_{p,p}$ and $C_{n,n}$ are set to 0; $C_{n,p}$ and $C_{p,n}$ may and may not equal

Evaluation Metrics

- Example of cost matrix
  - Assume we would like to develop a self-driving car system
  - We have an ML system that detects the pedestrians using camera, by conducing a binary classification
    - When it detects a person (positive class), the car should stop
    - When no person is detected (negative class), the car keeps going

Credit: automotiveworld.com

True Positive (cost $C_{p,p}$)
There is person, ML detects person and car stops

True Negative (cost $C_{n,n}$)
There is no person, car keeps going

False Positive (cost $C_{n,p}$)
There is no person, ML detects person and car stops

False Negative (cost $C_{p,n}$)
There is person, ML fails to detect person and car keeps going

$C_{n,p}$? $C_{p,n}$ (>, <, or =)

Evaluation Metrics

- For unbalanced data…
  - Assume we have 1000 samples, of which 10 are *positive* and 990 are *negative*
  - Accuracy = 990/1000=0.99!
    Very high number!
  - Yet, half of the Class-1 are Classified to Class-2!

	Class-1 (predicted)	Class-2 (predicted)
Class-1 (actual)	5 (TP)	5 (FN)
Class-2 (actual)	5 (FP)	985 (TN)

The goal is to highlight the problems of the results!

In this case, we shall

1) Use cost matrix, assign different costs for each entry
2) Use Precision and Recall! Precision = 0.5 and Recall = 0.5

Evaluation Metrics

Classification

Confusion Matrix for Multicategory Classification

	$P_{\hat{1}}$ (predicted)	$P_{\hat{2}}$ (predicted)		$P_{\hat{C}}$ (predicted)
P_1 (actual)	$P_{1,\hat{1}}$	$P_{1,\hat{2}}$	$\cdots$	$P_{1,\hat{C}}$
P_2 (actual)	$P_{2,\hat{1}}$	$P_{2,\hat{2}}$	$\cdots$	$P_{2,\hat{C}}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
P_C (actual)	$P_{C,\hat{1}}$	$P_{C,\hat{2}}$		$P_{C,\hat{C}}$

THANK YOU