Machine Learning: Chenhao Tan

University of Colorado Boulder

LECTURE 11

Slides adapted from Jordan Boyd-Graber, Chris Ketelsen

Logistics

- HW1 grades & solutions out
- HW2 available on Github, due in 4 days

Outline

Train-val-test

K-fold cross validation

Evaluate classifiers

The story so far

We've seen several machine learning models now (decision tree, KNN, perceptron, Logistic Regression, etc)
You've done your own experiments where you've selected hyperparameters:

- K in K-nearest neighbors
- number of epochs in perceptron

We've talked about the importance of evaluating a learning model on unseen validation data
You've been introduced to the confusion matrix and what it can tell you about how your learning algorithm makes mistakes

The story so far

We've seen several machine learning models now (decision tree, KNN, perceptron, Logistic Regression, etc)

You've done your own experiments where you've selected hyperparameters:

- K in K-nearest neighbors
- number of epochs in perceptron

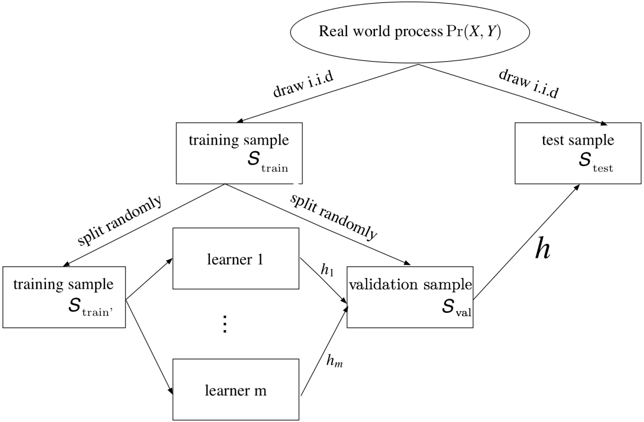
We've talked about the importance of evaluating a learning model on unseen validation data

You've been introduced to the confusion matrix and what it can tell you about how your learning algorithm makes mistakes

Next:

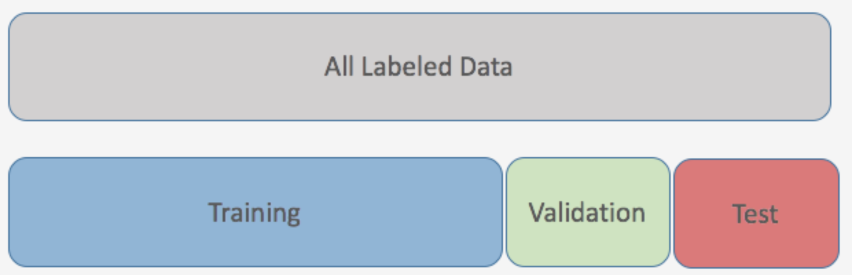
- Validation
- Evaluation metrics

Train-val-test



- training: run machine learning algorithm m times (e.g., parameter search).
- validation error: Errors $\mathrm{Err}_{S_{\text{val}}}(\hat{h}_i)$ is an estimate of $\mathrm{Err}_P(h_i)$.
- selection: Use h_i with $\min \mathrm{Err}_{S_{\text{val}}}(\hat{h}_i)$ for prediction on test examples.

Train-val-test



Typical ratio:

- 70%/10%/20%
- 80%/10%/10%

Outline

Train-val-test

K-fold cross validation

Evaluate classifiers

K-fold cross validation

When the number of training instances is small, it seems wasteful to have a separate validation set. What can we do?

K-fold cross validation

When the number of training instances is small, it seems wasteful to have a separate validation set. What can we do?

Using all training data:

- Input: a sample S and a learning algorithm A.
- Procedure: Randomly split S into K equally-sized folds

$$S_1, \ldots, S_K$$

  For each S_i, apply A to S_{-i}, get $\hat{h}_i$, and compute $\mathrm{Err}_{S_i}(\hat{h}_i)$
- Training performance estimates: $\frac{1}{K}\sum_{i=1}^{K} \mathrm{Err}_{S_i}(\hat{h}_i)$

K-fold Cross Validation

Example use:
5-fold CV: Randomly split $N = 25$ examples into five folds $F_i, i = 1, 2, 3, 4, 5$,

train on	test on	error rate
F_1, F_2, F_3, F_4	F_5	1/5
F_1, F_2, F_3, F_5	F_4	0/5
F_1, F_2, F_4, F_5	F_3	0/5
F_1, F_3, F_4, F_5	F_2	2/5
F_2, F_3, F_4, F_5	F_1	0/5

Average error rate: $\frac{1}{5}\sum_{i=1}^{5} \mathrm{Err}_{F_i} = 12\%$

K-fold Cross Validation

Example use:
5-fold CV: Randomly split $N = 25$ examples into five folds $F_i, i = 1, 2, 3, 4, 5$,

train on	test on	error rate
F_1, F_2, F_3, F_4	F_5	1/5
F_1, F_2, F_3, F_5	F_4	0/5
F_1, F_2, F_4, F_5	F_3	0/5
F_1, F_3, F_4, F_5	F_2	2/5
F_2, F_3, F_4, F_5	F_1	0/5

Average error rate: $\frac{1}{5}\sum_{i=1}^{5} \mathrm{Err}_{F_i} = 12\%$
Repeat this process for different hyperparameters and find the hyperparameter with the lowest error rate.

K-fold Cross Validation

Another example:

- Find good features F using S_{train}
- Split S_{train} into K folds
- For each fold, use the rest training data and features F to build a classifier and estimate *prediction error* using average error rates on each fold

K-fold Cross Validation

Another example (This is wrong!):

- Find good features F using S_{train}
- Split S_{train} into K folds
- For each fold, use the rest training data and features F to build a classifier and estimate *prediction error* using average error rates on each fold

Note: the feature selection step actually has information about the supposedly heldout set.
Never *ever* touch your test data in any way!

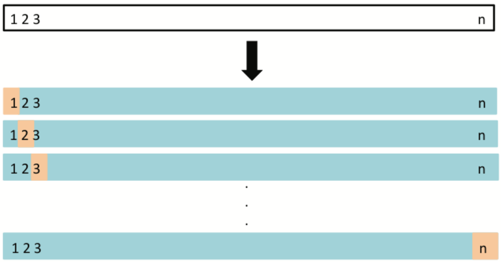
K-fold cross validation

K-fold cross validation can be used for

- selecting best models from training data
- nested cross-validation for performance estimation

Leave-one out cross validation

A special case where $k = N$



LOOCV error rate

$$\frac{1}{n}\sum_{i=1}^{n} I(y_i \neq \hat{y}_i^{h_{-i}}),$$

where h_{-i} represents the model trained using all the instances other than i.

Outline

Train-val-test

K-fold cross validation

Evaluate classifiers

Evaluating learned hypothesis

- Goal: Find h with small prediction error $\mathrm{Err}_P(h)$ over $P(X, Y)$
- Question: What is $\mathrm{Err}_P(\hat{h})$ of $\hat{h}$ obtained from training data S_{train}
- Training error and test error
    - Training error: $\mathrm{Err}_{S_{\mathrm{train}}}(\hat{h})$
    - Test error: $\mathrm{Err}_{S_{\mathrm{test}}}(\hat{h})$ is an estimate of $\mathrm{Err}_P(\hat{h})$

Key questions in practice

- Is model (hypothesis) $\hat{h}_1$ better than $\hat{h}_2$?
- Is algorithm A_1 better than A_2?

What is the true error of a hypothesis?

- Apply $\hat{h}$ to S_{test}, for each $(\boldsymbol{x}, y) \in S_{\text{test}}$ observer $\Delta(\hat{h}(\boldsymbol{x}), y)$.

What is the true error of a hypothesis?

- Apply $\hat{h}$ to S_{test}, for each $(\boldsymbol{x}, y) \in S_{\text{test}}$ observer $\Delta(\hat{h}(\boldsymbol{x}), y)$.
- Binomial distribution estimates: assume that each toss is independent and the probability of heads is p, then the probability of observing x heads in a sample of n independent coin tosses is

$$\Pr(X = x | p, n) = \frac{n!}{x!(n-x)!} p^x (1-p)^{n-x}$$

What is the true error of a hypothesis?

- Apply $\hat{h}$ to S_{test}, for each $(\boldsymbol{x}, y) \in S_{\text{test}}$ observer $\Delta(\hat{h}(\boldsymbol{x}), y)$.
- Binomial distribution estimates: assume that each toss is independent and the probability of heads is p, then the probability of observing x heads in a sample of n independent coin tosses is

$$\Pr(X = x | p, n) = \frac{n!}{x!(n-x)!} p^x (1-p)^{n-x}$$

  - Normal approximation
  - $\mathrm{Err}(\hat{h}) = \hat{p} = \frac{1}{n} \sum_{i=1}^{n} \Delta(\hat{h}(\boldsymbol{x}_i), y_i)$
  - Confidence interval: $\hat{p} \pm z_\alpha \sqrt{\frac{1}{n} \hat{p}(1-\hat{p})}$

Is hypothesis $\hat{h}_1$ better than $\hat{h}_2$?

Same test sample

- Apply $\hat{h}_1$ and $\hat{h}_2$ to S_{test}

Is hypothesis $\hat{h}_1$ better than $\hat{h}_2$?

Same test sample

- Apply $\hat{h}_1$ and $\hat{h}_2$ to S_{test}
- Decide if $\mathrm{Err}_P(\hat{h}_1) \neq \mathrm{Err}_P(\hat{h}_2)$
- Null hypothesis: $\mathrm{Err}_{S_{\text{test}}}(\hat{h}_1)$ and $\mathrm{Err}_{S_{\text{test}}}(\hat{h}_2)$ come from binomial distributions with same p
  Binomial Sign Test (McNemar's Test)

Is hypothesis $\hat{h}_1$ better than $\hat{h}_2$?

Different test samples

- Apply $\hat{h}_1$ to S_{test1} and $\hat{h}_2$ to S_{test2}

Is hypothesis $\hat{h}_1$ better than $\hat{h}_2$?

Different test samples

- Apply $\hat{h}_1$ to S_{test1} and $\hat{h}_2$ to S_{test2}
- Decide if $\mathrm{Err}_P(\hat{h}_1) \neq \mathrm{Err}_P(\hat{h}_2)$
- Null hypothesis: $\mathrm{Err}_{S_{\text{test1}}}(\hat{h}_1)$ and $\mathrm{Err}_{S_{\text{test2}}}(\hat{h}_2)$ come from binomial distributions with same p
  t-test

Is learning algorithm A_1 better than A_2?

- Given k samples of $S_1 \ldots S_k$ of labeled instances from $P(X, Y)$, each S_i randomly split into $S^i_{\text{test}}, S^i_{\text{train}}$.
- For each i, train A_1, A_2 on S^i_{train}, obtain $\hat{h}_i^{A_1}$ and $\hat{h}_i^{A_2}$, apply to S^i_{test} and compute $\mathrm{Err}_{S_{\text{test}}}(\hat{h}_i^{A_1}), \mathrm{Err}_{S_{\text{test}}}(\hat{h}_i^{A_2})$

Is learning algorithm A_1 better than A_2?

- Given k samples of $S_1 \dots S_k$ of labeled instances from $P(X, Y)$, each S_i randomly split into $S^i_{\text{test}}, S^i_{\text{train}}$.
- For each i, train A_1, A_2 on S^i_{train}, obtain $\hat{h}^{A_1}_i$ and $\hat{h}^{A_2}_i$, apply to S^i_{test} and compute $\mathrm{Err}_{S_{\text{test}}}(\hat{h}^{A_1}_i), \mathrm{Err}_{S_{\text{test}}}(\hat{h}^{A_2}_i)$
- Decide, if $\mathrm{E}_S(\mathrm{Err}_P(A_1(S_{\text{train}}))) \neq \mathrm{E}_S(\mathrm{Err}_P(A_2(S_{\text{train}})))$
- Null hypothesis: $\mathrm{Err}_{S_{\text{test}}}(A_1(S_{\text{train}}))$ and $\mathrm{Err}_{S_{\text{test}}}(A_2(S_{\text{train}}))$ come from same distribution over samples S
  t-test or Wilcoxon Signed-Rank Test