Goals for the lecture

you should understand the following concepts

• bias of an estimator
• test sets
• learning curves
• stratified sampling
• cross validation
• confusion matrices
• TP, FP, TN, FN
• ROC curves
• precision-recall curves
• recall/sensitivity/true positive rate (TPR)
• precision/positive predictive value (PPV)
• specificity and false positive rate (FPR or 1-specificity)
Bias of an estimator

\[ \theta \]  true value of parameter of interest (e.g. model accuracy)
\[ \hat{\theta} \]  estimator of parameter of interest (e.g. test set accuracy)

\[ \text{Bias}[\hat{\theta}] = E[\hat{\theta}] - \theta \]

e.g. polling methodologies often have an inherent bias

![Polling Methodologies Table]

Test sets revisited

How can we get an unbiased estimate of the accuracy of a learned model?

1. Labeled data set
2. Training set
3. Learning method
4. Test set
5. Learned model
6. Accuracy estimate
Test sets revisited

How can we get an unbiased estimate of the accuracy of a learned model?

- when learning a model, you should pretend that you don’t have the test data yet (it is “in the mail”)*

- if the test-set labels influence the learned model in any way, accuracy estimates will be biased

* In some applications it is reasonable to assume that you have access to the feature vector (i.e. $x$) but not the $y$ part of each test instance.

Learning curves

How does the accuracy of a learning method change as a function of the training-set size?

this can be assessed by plotting learning curves

Figure from Perlich et al. Journal of Machine Learning Research, 2003
Learning curves

given training/test set partition
- for each sample size \( s \) on learning curve
  - (optionally) repeat \( n \) times
    - randomly select \( s \) instances from training set
    - learn model
    - evaluate model on test set to determine accuracy \( a \)
  - plot \((s, a)\) or \((s, \text{avg. accuracy and error bars})\)

Limitations of using a single training/test partition

- we may not have enough data to make sufficiently large training and test sets
  - a larger test set gives us more reliable estimate of accuracy (i.e. a lower variance estimate)
  - but… a larger training set will be more representative of how much data we actually have for learning process

- a single training set doesn’t tell us how sensitive accuracy is to a particular training sample
Using multiple training/test partitions

- two general approaches for doing this
  - random resampling
  - cross validation

Random resampling

We can address the second issue by repeatedly randomly partitioning the available data into training and test sets.

labeled data set

random partitions

training sets
test sets
Stratified sampling

When randomly selecting training or validation sets, we may want to ensure that class proportions are maintained in each selected set.

- **labeled data set**
  - ++++++++ - - - -

  \[ \text{training set} \]
  - +++++ - -

  \[ \text{test set} \]
  - +++++ - -

  \[ \text{validation set} \]
  - +++ -

This can be done via stratified sampling: first stratify instances by class, then randomly select instances from each class proportionally.

Cross validation

- **labeled data set**
  -               

  \[ \text{partition data into } n \text{ subsamples} \]

  \[ \text{iteration} \]
  - train on | test on
  - ------- | -------
  - 1 s_2 s_3 s_4 s_5 s_1
  - 2 s_1 s_3 s_4 s_5 s_2
  - 3 s_1 s_2 s_4 s_5 s_3
  - 4 s_1 s_2 s_3 s_5 s_4
  - 5 s_1 s_2 s_3 s_4 s_5

- **iteratively leave one subsample out for the test set, train on the rest**
Cross validation example

Suppose we have 100 instances, and we want to estimate accuracy with cross validation

<table>
<thead>
<tr>
<th>iteration</th>
<th>train on</th>
<th>test on</th>
<th>correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>s₂ s₃ s₄ s₅</td>
<td>s₁</td>
<td>11 / 20</td>
</tr>
<tr>
<td>2</td>
<td>s₁ s₃ s₄ s₅</td>
<td>s₂</td>
<td>17 / 20</td>
</tr>
<tr>
<td>3</td>
<td>s₁ s₂ s₄ s₅</td>
<td>s₃</td>
<td>16 / 20</td>
</tr>
<tr>
<td>4</td>
<td>s₁ s₂ s₃ s₅</td>
<td>s₄</td>
<td>13 / 20</td>
</tr>
<tr>
<td>5</td>
<td>s₁ s₂ s₃ s₄</td>
<td>s₅</td>
<td>16 / 20</td>
</tr>
</tbody>
</table>

accuracy = 73/100 = 73%

Cross validation

- 10-fold cross validation is common, but smaller values of n are often used when learning takes a lot of time

- in leave-one-out cross validation, n = # instances

- in stratified cross validation, stratified sampling is used when partitioning the data

- CV makes efficient use of the available data for testing

- note that whenever we use multiple training sets, as in CV and random resampling, we are evaluating a learning method as opposed to an individual learned model
Confusion matrices

How can we understand what types of mistakes a learned model makes?

Confusion matrix for 2-class problems

\[
\text{accuracy} = \frac{TP + TN}{TP + FP + FN + TN}
\]

\[
\text{error} = 1 - \text{accuracy} = \frac{FP + FN}{TP + FP + FN + TN}
\]
Is accuracy an adequate measure of predictive performance?

Accuracy may not be a useful measure in cases where

- **there is a large class skew**
  - Is 98% accuracy good when 97% of the instances are negative?

- **there are differential misclassification costs** – say, getting a positive wrong costs more than getting a negative wrong
  - Consider a medical domain in which a false positive results in an extraneous test but a false negative results in a failure to treat a disease

- **we are most interested in a subset of high-confidence predictions**

Other accuracy metrics

<table>
<thead>
<tr>
<th>Actual Class</th>
<th>Predicted Class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Positive</td>
<td>True Positives (TP)</td>
</tr>
<tr>
<td>Negative</td>
<td>False Negatives (FN)</td>
</tr>
</tbody>
</table>

- **true positive rate (recall)** = \( \frac{TP}{\text{actual pos}} = \frac{TP}{TP + FN} \)
- **false positive rate** = \( \frac{FP}{\text{actual neg}} = \frac{FP}{TN + FP} \)
ROC curves

A Receiver Operating Characteristic (ROC) curve plots the TP-rate vs. the FP-rate as a threshold on the confidence of an instance being positive is varied.

Different methods can work better in different parts of ROC space.

Algorithm for creating an ROC curve

let \( \{(y^{(1)}, c^{(1)}) \ldots (y^{(m)}, c^{(m)})\} \) be the test-set instances sorted according to predicted confidence
\( c^{(i)} \) that each instance is positive

let num_neg, num_pos be the number of negative/positive instances in the test set

\( TP = 0, \ FP = 0 \)

\( last\_TP = 0 \)

for \( i = 1 \) to \( m \)

// find thresholds where there is a pos instance on high side, neg instance on low side
if \( (i > 1) \) and \( (c^{(i)} \neq c^{(i-1)}) \) and \( (y^{(i)} == \) neg \) and \( (TP > last\_TP) \)

\( FPR = FP / num\_neg, \ TPR = TP / num\_pos \)

output \((FPR, TPR)\) coordinate

\( last\_TP = TP \)

if \( y^{(i)} == \) pos

++TP

else

++FP

\( FPR = FP / num\_neg, \ TPR = TP / num\_pos \)

output \((FPR, TPR)\) coordinate
Plotting an ROC curve

<table>
<thead>
<tr>
<th>instance</th>
<th>confidence</th>
<th>correct class</th>
</tr>
</thead>
<tbody>
<tr>
<td>Ex 9</td>
<td>.99</td>
<td>+</td>
</tr>
<tr>
<td>Ex 7</td>
<td>.98 TPR= 2/5, FPR= 0/5</td>
<td>+</td>
</tr>
<tr>
<td>Ex 1</td>
<td>.72</td>
<td>-</td>
</tr>
<tr>
<td>Ex 2</td>
<td>.70</td>
<td>+</td>
</tr>
<tr>
<td>Ex 6</td>
<td>.65 TPR= 4/5, FPR= 1/5</td>
<td>+</td>
</tr>
<tr>
<td>Ex 10</td>
<td>.51</td>
<td>-</td>
</tr>
<tr>
<td>Ex 3</td>
<td>.39</td>
<td>-</td>
</tr>
<tr>
<td>Ex 5</td>
<td>.24 TPR= 5/5, FPR= 3/5</td>
<td>+</td>
</tr>
<tr>
<td>Ex 4</td>
<td>.11</td>
<td>-</td>
</tr>
<tr>
<td>Ex 8</td>
<td>.01 TPR= 5/5, FPR= 5/5</td>
<td>-</td>
</tr>
</tbody>
</table>

ROC curve example

Task: recognizing genomic units called operons

figure from Bockhorst et al., Bioinformatics 2003
ROC curves and misclassification costs

The best operating point depends on the relative costs of FN and FP misclassifications.

**ROC curves**

Does a low false-positive rate indicate that most positive predictions (i.e., predictions with confidence > some threshold) are correct?

suppose our TPR is 0.9, and FPR is 0.01

<table>
<thead>
<tr>
<th>fraction of instances that are positive</th>
<th>fraction of positive predictions that are correct</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5</td>
<td>0.989</td>
</tr>
<tr>
<td>0.1</td>
<td>0.909</td>
</tr>
<tr>
<td>0.01</td>
<td>0.476</td>
</tr>
<tr>
<td>0.001</td>
<td>0.083</td>
</tr>
</tbody>
</table>
Other accuracy metrics

<table>
<thead>
<tr>
<th>predicted class</th>
<th>actual class</th>
</tr>
</thead>
<tbody>
<tr>
<td>positive</td>
<td>true positives (TP)</td>
</tr>
<tr>
<td>negative</td>
<td>false negatives (FN)</td>
</tr>
</tbody>
</table>

recall (TP rate) = \frac{TP}{\text{actual pos}} = \frac{TP}{TP + FN}

precision (positive predictive value) = \frac{TP}{\text{predicted pos}} = \frac{TP}{TP + FP}

Precision/recall curves

A precision/recall curve plots the precision vs. recall (TP-rate) as a threshold on the confidence of an instance being positive is varied.
Precision/recall curve example

predicting patient risk for VTE

How do we get one ROC/PR curve when we do cross validation?

Approach 1
• make assumption that confidence values are comparable across folds
• pool predictions from all test sets
• plot the curve from the pooled predictions

Approach 2 (for ROC curves)
• plot individual curves for all test sets
• view each curve as a function
• plot the average curve for this set of functions
Comments on ROC and PR curves

both
• allow predictive performance to be assessed at various levels of confidence
• assume binary classification tasks
• sometimes summarized by calculating area under the curve

ROC curves
• insensitive to changes in class distribution (ROC curve does not change if the proportion of positive and negative instances in the test set are varied)
• can identify optimal classification thresholds for tasks with differential misclassification costs

precision/recall curves
• show the fraction of predictions that are false positives
• well suited for tasks with lots of negative instances