Receiver Operating Characteristic (ROC) Curves

Evaluating a classifier and predictive performance
Classification (supervised learning)

• In supervised learning, we are interested in building a model to predict the class (or outcome) of a new observation based on observable predictors using the train-set/test-set framework.
  – Which students are not likely to return for their second year of college?
  – Which animals admitted to a veterinary hospital are likely to survive (or not survive)?
  – Which bank customers are likely to default on a loan?
Some classification procedures:

• **Logistic regression** (0-1 response)
  – Choose variables, fit model, calculate $\hat{y}$ (i.e. $\hat{p}$)
  – $\hat{p} \geq 0.5$ predicted class 1, $\hat{p} < 0.5$ predicted class 0

• **Classification tree**
  – Use ‘greedy’ algorithm to choose variables and create decision tree (prediction at tree leaves)

• **Linear Discriminant Analysis (LDA)**
  – Built on multivariate normal (spheroids)
  – Classify new observation to nearest centroid
Assumptions:

• Logistic regression
  – Predictors are related to response with a sigmoidal mean structure
  – Conditional Bernoulli given the mean

• Classification tree (maybe?)
  – The data can be described by features
  – There exists some tree that isn’t too big that can predict reasonably well

• Linear discriminant analysis
  – Data follow multivariate Gaussian
  – Equal covariance matrices
## Pros/Cons of these classifiers:

<table>
<thead>
<tr>
<th>Pros</th>
<th>Cons</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Logistic Regression</strong></td>
<td>Interpretation of covariates. More robust to deviations from assumptions than LDA.</td>
</tr>
<tr>
<td></td>
<td>Computationally complex iterative algorithm and may not converge. Depends on sigmoidal mean structure and conditional binomial distributions.</td>
</tr>
<tr>
<td><strong>Trees</strong></td>
<td>Fewer assumptions of data, more flexible algorithm, easy interpretation of certain characteristics.</td>
</tr>
<tr>
<td></td>
<td>Instability (random forests can help), lack of smoothness, uses a greedy algorithm.</td>
</tr>
<tr>
<td><strong>LDA</strong></td>
<td>Dimensionality reduction, estimation under assumptions uses maximum likelihood</td>
</tr>
<tr>
<td></td>
<td>Not flexible when assumptions deviate from multivariate normal assumptions.</td>
</tr>
</tbody>
</table>

*NOTE: In simulation studies, we have found that each of these classifiers performs better than the others when the data were simulated under the given model assumptions.*
What makes a good classifier?

• Your predictions are correct.
  – True positives were correctly classified
  – True negatives were correctly classified

• Example: Disease diagnostic test
  – Patients are given the diagnostic test
  – Test gives those with the disease a ‘positive’
    • High sensitivity (correctly classifying true positives)
  – Test gives those without the disease a ‘negative’
    • High specificity (correctly classifying true negatives)
Misclassification rate

• Misclassification
  – The observation belongs to one class, but the model classifies it as a member of a different class.

• A classifier that makes no errors would be perfect
  – Not likely to find such a classifier in the real world
  – Noise
  – Other relevant variables not in the available data set
Misclassification rate

• We can present the accuracy of the results with a confusion matrix.

<table>
<thead>
<tr>
<th></th>
<th>Predict as 1</th>
<th>Predict as 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual 1</td>
<td>20</td>
<td>5</td>
</tr>
<tr>
<td>Actual 0</td>
<td>18</td>
<td>957</td>
</tr>
</tbody>
</table>

• Overall error rate = \( \frac{18+5}{1000} = 2.3\% \)

• NOTE: The prediction value is often calculated through leave-one-out cross-validation in logistic regression.
Beating the naïve classifier

• In this data set, the 0’s were much more prevalent than the 1’s.
• What if we just classified ALL observations as 0? How well do we do?
• Overall error rate = \(\frac{25}{1000} = 2.5\%\)

<table>
<thead>
<tr>
<th>Actual</th>
<th>Predict as 1</th>
<th>Predict as 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>Actual 1</td>
<td>20</td>
<td>5</td>
</tr>
<tr>
<td>Actual 0</td>
<td>18</td>
<td>957</td>
</tr>
</tbody>
</table>
Distinguishing between the two types of mistakes to be made

- **Sensitivity** $= \frac{\# \text{ of true positives declared ‘positive’}}{\# \text{ of true positives}}$
  
  Is your disease diagnostic tool getting the ones you want?

- **Specificity** $= \frac{\# \text{ of true negatives not declared ‘positive’}}{\# \text{ of true negatives}}$
  
  Is your disease diagnostic tool NOT getting the ones you DON’T want?

- For a given classifier, we would like both to be high.
Receiver Operating Characteristic (ROC) curve

• To create an ROC curve, we first order the predicted probabilities from highest to lowest.
• Highest probabilities are predicted to have the disease (we’ll want to classify those as ‘disease’).
• Lowest probabilities are predicted to not have the disease (we’ll want to classify those as ‘not disease’).
• Probability = 0.5? Flip of the coin.
ROC curve

- To create an ROC curve, we start at the highest predicted probability and work our way down the list (i.e. highest prob to lowest prob), we stop at each position $C$ (i.e. potential cutoff) and use it as the classifier (i.e. $C$ or above = 1, below $C$ = 0), and ask “How well does this classifier do? Sensitivity? Specificity?”

<table>
<thead>
<tr>
<th>preds</th>
<th>Classified as ‘1’</th>
<th>Classified as ‘0’</th>
</tr>
</thead>
<tbody>
<tr>
<td>[1,] 0.9299</td>
<td>[2,] 0.9033</td>
<td>[3,] 0.8918</td>
</tr>
<tr>
<td>[4,] 0.8851</td>
<td>[5,] 0.8687</td>
<td>.</td>
</tr>
</tbody>
</table>

← Most likely to have disease

For example, if we classify cases with $C=0.8918$ and higher as ‘1’ and probabilities less than $C=0.8918$ as ‘0’, how well do we do?
NOTE: If we start at the top of the list and we don’t go very far down the list for C, we’ll probably get mostly true positives (i.e. low false positive rate, good), but there are still lots of true positives that we missed (low true positive rate, bad).

Classified as ‘1’

[1,] 0.9299 ← Most likely to have disease
[2,] 0.9033
[3,] 0.8918
[4,] 0.8851
[5,] 0.8687

Classified as ‘0’

...
The ROC curve shows us what happens to sensitivity and specificity as we move our C (threshold for classifier) down the list.

C at high predicted probability (very few cases declared ‘disease’).
Low false positive rate (good).
Low true positive rate (bad).

C at low predicted probability (almost all cases declared ‘disease’).
High false positive rate (bad).
High true positive rate (good).

Sensitivity and specificity info from the “first” C threshold.
Sensitivity and specificity info from the “last” C threshold.
We essentially start at the point (0,0). If the first case (highest probability) is correctly classified when declared ‘disease’, we move vertically up (true positive). If the first case (highest probability) is incorrectly classified as ‘disease, we move to the right (false positive).

Each move down the list of probabilities coincides with either a move up on the ROC curve (correct choice as ‘disease’) or to the right (incorrect choice when declared ‘disease’). Thus, we see a ‘stair step’ phenomenon in the ROC curve. For small data sets, this is very apparent.

As we start moving down the list, we want to move up on the ROC curve, not to the right.
See animation of ROC curve creation

http://homepage.stat.uiowa.edu/~rdecook/stat6220/ROC_animated1.html
What ROC shape says it’s a good classifier?

• We want it to go up vertically very quickly
  – i.e. As we move down the list of predicted probabilities, we’re getting all ‘diseased’ cases and no ‘non-diseased’.

• We know in the end (when everyone is classified as a ‘disease’ case) all the negatives will be misclassified and all the positives will be correctly classified. So, we’ll always end at (1,1).
Comparing classifiers with ROC analysis

AUC or area under the curve compares the classifiers too.

Best of these
Worst of these
Choosing at random

Unmatched (AUC = 0.97)
TSS (AUC = 0.88)
Region (AUC = 0.71)
Chance
ROC curve: Provost’s Office Client

• Example: Predict which students will not return for their second year of college.
  
  – We use the predicted probabilities from the **logistic regression** for classification.
  
  – If the probability of a case being in class 1 (not retained) is equal to or greater than 0.5, that case is classified as a 1.
  
  – Any case with an estimated probability of less than 0.5 would be classified as a 0 (retained).
Provost’s Office Example

• Logistic regression variables
  – Stafford loan
  – Live on campus
  – First generation college student
  – RAI
  – Selective program of study
  – High School GPA
Provost’s Office Example

- The predicted probability of not being retained is used to classify each case.
- Students with a very high probability are expected to not return.
- Students with a very low probability are expected to return in their 2nd year.
The ROC curve from the given classifier: logistic regression predicted probabilities... meh

*Plot generated from ROCR package in R.*
Provost’s Office Example

The blue point represents the classifier based on probability=0.5 cutoff.

Sensitivity:
\[ \frac{41}{859} = 4.77\% \]

Specificity:
\[ \frac{4762}{4799} = 99.23\% \]

Overall Misclassification:
\[ \frac{855}{5658} = 0.1511 \]

*Point calculated by hand and added.*
Provost’s Office Example

We can also look at **accuracy** (1-misclassification rate) vs. the cutoff. The best cutoff for highest accuracy is 0.4623.

**Sensitivity:**
66/859 = 7.68%

**Specificity:**
4750/4799 = 98.97%

**Overall Misclassification:**
842/5658 = 0.1488

(Pretty close to our 0.5 threshold)

*Plot generated from ROCR package in R.*
Provost’s Office Example

The green point represents the optimal classifier based on equal importance of sensitivity and specificity (as max of “sensitivity + specificity”) which is prob=0.191 cutoff.

Sensitivity: 55.2%  
Specificity: 77.8%

Overall Misclassification:  
1456/5658 = 0.2573

*Point determined from Epi package.
Provost’s Office Example

Optimal cutoffs can be found from software or calculated.

Min euclid distance cutoff (0.146)
Max sens + spec cutoff (0.191)  Our 0.5 cutoff
Cutoff where everyone declared positive.
Cutoff where no one declared positive.
Comparing classifiers with ROCs

The curves may visually “look” different, but are they really? If we collected data again, would they look this different?
Bootstrapping the ROC

Can use bootstrapping to create a confidence band for the sensitivity (given specificity) for the ROC curve.

*Plot generated from \textit{pROC} package in R.*
Bootstrapping the ROC

But the bootstrap results don’t always look so nice.

ROC curve with 95% CIs

AUC: 83.0% (60.4%–100.0%)

*Plot generated from pROC package in R.*
Bootstrapping ROC curves for comparison

Semi-transparent coloring in R can come-in handy.

# semi-transparent red:
col="#ff000030"

# semi-transparent blue:
col="#0000ff20"

*Plot generated from pROC package in R.*
‘NotRetained’ is the 0-1 response variable.
‘preds’ are the leave-one-out predicted probabilities.

Use 0.5 as the cutoff for classifying:
pred.outcome=round(preds)

Gather the data for the ROC curve:
ROC.info.LogReg=data.frame(NotRetained,preds,pred.outcome)

Put in order by predicted probability, largest first:
ROC.info.LogReg=ROC.info.LogReg[rev(order(ROC.info.LogReg$preds)),]

A 'positive' will be considered as someone who was not retained.
(num.true.pos=sum(NotRetained==1))
(num.true.neg=sum(NotRetained==0))

plot(c(0,1),c(0,1),type="n",xlab="1-Specificity (i.e. % of true neg's incorrectly declared positive)",ylab="Sensitivity (% of true pos's declared positive)",main="ROC curve (logistic regression)",sub="False Positive Rate")
x=cumsum(1-ROC.info.LogReg$NotRetained)/num.true.neg
y=cumsum(ROC.info.LogReg$NotRetained)/num.true.pos
lines(x,y)
abline(0,1)
## 1b) Manually calculating AUC:

### Calculate AUC thinking as geometric trapezoid shapes:


### Same x,y labeling as previous slide:

```r
x = cumsum(1-ROC.info.LogReg$NotRetained)/num.true.neg
y = cumsum(ROC.info.LogReg$NotRetained)/num.true.pos
```

height = (y[-1]+y[-n])/2

width = diff(x)

sum(height*width)
## 1c) Manually calculating sensitivity, specificity, misclassification rate for 0.5 threshold:

All the needed info is in the following table:

```r
table(NotRetained, pred.outcome)
```

```
#       0    1
#  0 4762   37
#  1  818   41
```
```r
# 2) ROCR package for creating ROC curve: 
library(ROCR)

pred = prediction(preds, NotRetained)

perf = performance(pred, "tpr", "fpr")

plot(perf, main = "ROC from logistic regression classifier")

(AUC.ROCR = performance(pred, "auc")@y.values[[1]])

## Plot accuracy (1-misclassification rate) vs. cutoff:
acc = performance(pred, "acc")
(ac.val = max(unlist(acc@y.values)))
#[1] 0.8511842

th = unlist(acc@x.values)[unlist(acc@y.values) == ac.val]

plot(acc)
abline(v = th, col = 'grey', lty = 2)
th
#[1] 0.462324
```
## 3) pROC package for bootstrapped ROC:

```r
library(pROC)

# 'NotRetained' is the 0-1 response variable.
# 'preds' are the leave-one-out predicted probabilities.

# Use 0.5 as the cutoff for classifying:
pred.outcome = round(preds)

rocobj = plot.roc(NotRetained, preds, main="ROC curve with 95% CIs",
percent=TRUE, ci=TRUE, print.auc=TRUE)

# Calculate CI of sensitivity at select set of specificities and form a 'band' (might take a bit):
ciobj = ci.se(rocobj, specificities=seq(0, 100, 5))
plot(ciobj, type="shape", col="#1c61b6AA") # blue band

# Use bootstrap to add CI in both directions at "best" cutoff:
plot(ci(rocobj, of="thresholds", thresholds="best"), col="yellow", lwd=2)
```
## Overlay Bootstrapped ROC's:
rocobj = plot.roc(NotRetained, preds, main="ROC curve with 95% Bootstrapped CIs", percent=TRUE, ci=TRUE, print.auc=FALSE)

ciobj = ci.se(rocobj, specificities=seq(0, 100, 5))
plot(ciobj, type="shape", col="#ff000030")  # semi-transparent red color

## Gather info on second classifier:
ROC.info.LogReg.2 = data.frame(NotRetained, preds.2, pred.outcome.2)

## Overlay the second ROC curve onto the first:
rocobj2 = plot.roc(ROC.info.LogReg.2$NotRetained, ROC.info.LogReg.2$preds.2, percent=TRUE, ci=TRUE, print.auc=FALSE, add=TRUE)

ciobj2 = ci.se(rocobj2, specificities=seq(0, 100, 5))
plot(ciobj2, type="shape", col="#0000ff20")  # semi-transparent blue color
Some references

  – https://research-information.bristol.ac.uk/files/94977288/Peter_Flach_ROC_Analysis.pdf

• James, G., Witten, D., Hastie, T. and Tibshirani, R. (2015). An Introduction to Statistical Learning with Applications in R.
  – http://www-bcf.usc.edu/~gareth/ISL/
  – Click ‘Download the book PDF’