Statistical Modelling III

HES 505 Fall 2023: Session 24

Matt Williamson

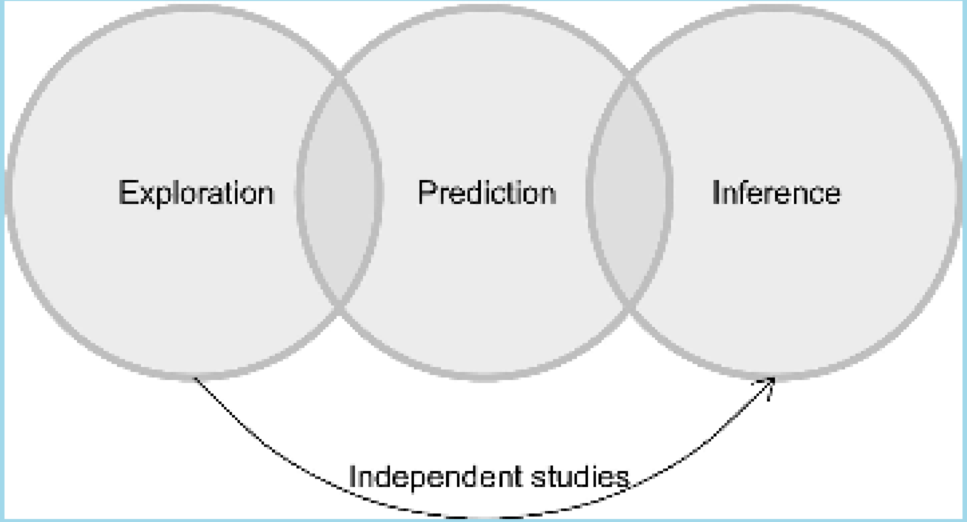
Objectives

By the end of today you should be able to:

- Articulate three different reasons for modeling and how they link to assessments of fit
- Describe and implement several test statistics for assessing model fit
- Describe and implement several assessments of classification
- Describe and implement resampling techniques to estimate predictive performance

The 3 Faces of Models

Best Model for What?



from Tradennick et al. 2021

- **Exploration:** describe patterns in the data and generate hypotheses
- **Inference:** evaluate the strength of evidence for some statement about the process
- **Prediction:** forecast outcomes at unsampled locations based on covariates

The Importance of Model Fit

- The general regression context:

$$\hat{y} = \mathbf{X}\hat{\beta}$$

- **Inference** is focused on robust estimates of given the data we have
- **Prediction** is focused on accurate forecasts of at locations where we have yet to collect the data

Inference and Presence/Absence Data

- is conditional on variables in the model **and** those not in the model

```r
nsamp <- 1000
df <- data.frame(x1 = rnorm(nsamp,0,1),
                 x2 = rnorm(nsamp,0,1),
                 x3 = rnorm(nsamp,0,1))

linpred <- 1 + 2*df$x1 -0.18*df$x2 -3.5*df$x3
y <- rbinom(nsamp, 1, plogis(linpred))
df <- cbind(df, y)

mod1 <- glm(y~x1 +x2, data=df, family="binomial")
mod2 <- glm(y~x1 +x2 + x3, data=df, family="binomial")
```

Inference & Presence/Absence Data

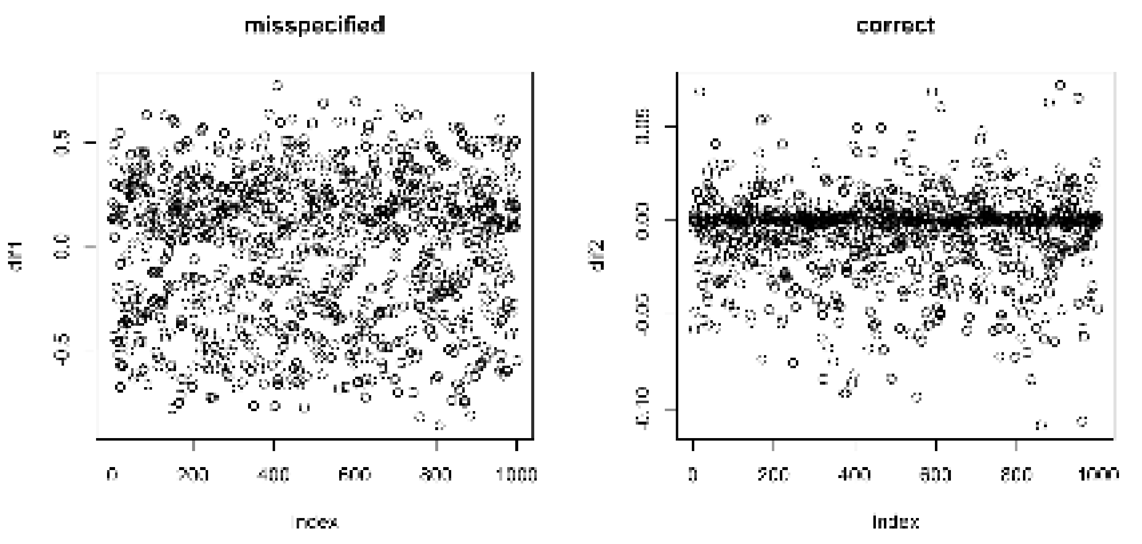
```r
coef(mod1)
```

```
(Intercept)          x1
x2
  0.5074517   0.8518245
-0.1433856
```

```r
coef(mod2)
```

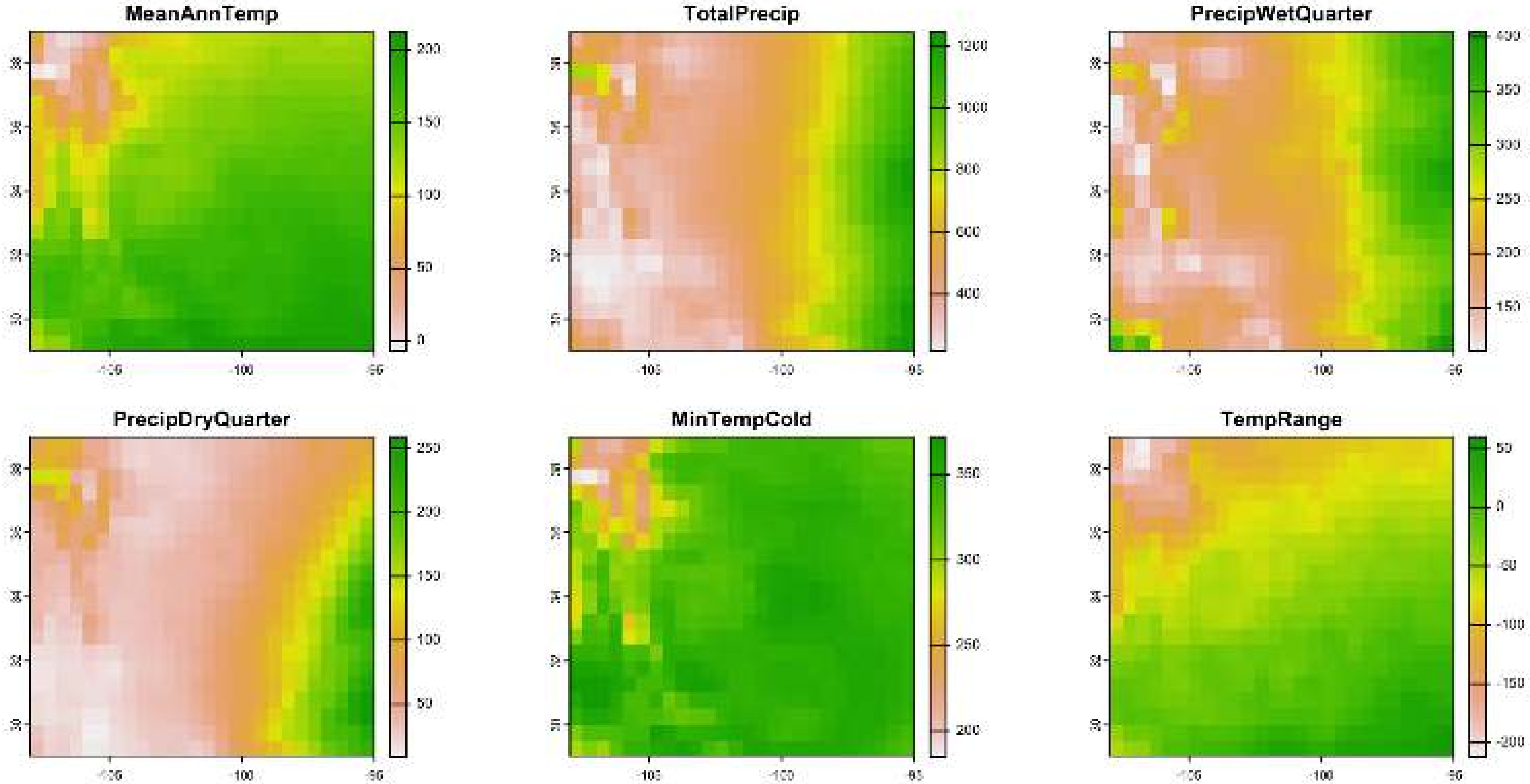
```
(Intercept)          x1
x2         x3
  1.0232517   1.9500362
-0.2995488  -3.2577412
```

```r
prd1 <- predict(mod1, df, "response")
dif1 <- plogis(linpred) - prd1
prd2 <- predict(mod2, df, "response")
dif2 <- plogis(linpred) - prd2
```



> Inferring coefficient effects requires that your model fit the data well

Assessing Model Fit


MeanAnnTemp
200
150
100
50
0
-105
-100
-95
TotalPrecip
1200
1000
800
600
400
-105
-100
-95
PrecipWetQuarter
400
350
300
250
200
150
-105
-100
-95
PrecipDryQuarter
250
200
150
100
50
-105
-100
-95
MinTempCold
350
300
250
200
-105
-100
-95
TempRange
50
0
-50
-100
-150
-200
-105
-100
-95

Using Test Statistics

- for linear regression:
  - Perfect prediction (); ; and
  - Null prediction (Intercept only) (); ; and
  - No direct way of implementing for logistic regression

Pseudo-

- Cohen's Likelihood Ratio
- Deviance (), the difference between the model and some hypothetical perfect model (lower is better)
- Challenge: Not monotonically related to
- Challenge: How high is too high?

Cohen’s Likelihood Ratio

```r
logistic.rich <- glm(y ~ MeanAnnTemp + PrecipWetQuarter + PrecipDryQuarter,
                     family=binomial(link="logit"),
                     data=pts.df[,2:8])

with(logistic.rich,
     null.deviance - deviance)/with(logistic.rich,
                                    null.deviance)
```

```
[1] 0.4495966
```

Pseudo-

- Cox and Snell
- Likelihood (), the probability of observing the sample given an assumed distribution
- Challenge: Maximum value is less than 1 and changes with
- Correction by Nagelkerke so that maximum is 1

Cox and Snell

```r
logistic.null <- glm(y ~ 1,
                     family=binomial(link="logit"),
                     data=pts.df[,2:8])

1 - exp(2*(logLik(logistic.null)[1] - logLik(logistic.rich)[1])/nobs(logist
```

```
[1] 0.4308873
```

Using Test Statistics

- Based on the data used in the model (i.e., not prediction)
- Likelihood Ratio behaves most similarly to
- Cox and Snell (and Nagelkerke) increases with more presences
- Ongoing debate over which is “best”
- **Don’t defer to a single statistic**

Assessing Predictive Ability

Predictive Performance and Fit

- Predictive performance can be an estimate of fit
- Comparisons are often relative (better good)
- Theoretical and subsampling methods

Theoretical Assessment of Predictive Performance

Hirotugu Akaike of AIC

- Information Criterion Methods
- Minimize the amount of information lost by using model to approximate true process
- Trade-off between fit and overfitting
- Can’t know the true process (so comparisons are relative)

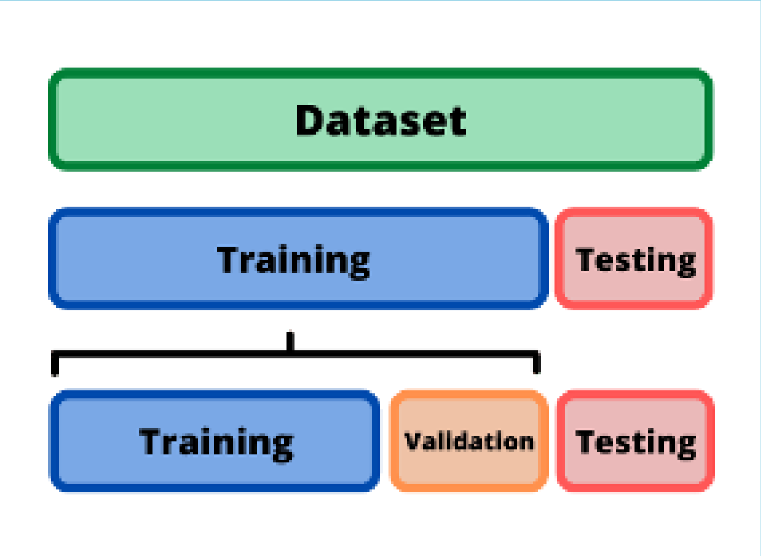
AIC Comparison

```r
logistic.null <- glm(y ~ 1,
                     family=binomial(link="logit"),
                     data=pts.df[,2:8])

logistic.rich <- glm(y ~ MeanAnnTemp + PrecipWetQuarter + PrecipDryQuarter,
                     family=binomial(link="logit"),
                     data=pts.df[,2:8])

AIC(logistic.null, logistic.rich)
```

```
              df       AIC
logistic.null  1 127.37389
logistic.rich  4  77.00622
```

Sub-sampling Methods

- Split data into *training* and *testing*
- Testing set needs to be large enough for results to be statistically meaningful
- Test set should be representative of the data as a whole
- Validation data used to tune parameters (not always)

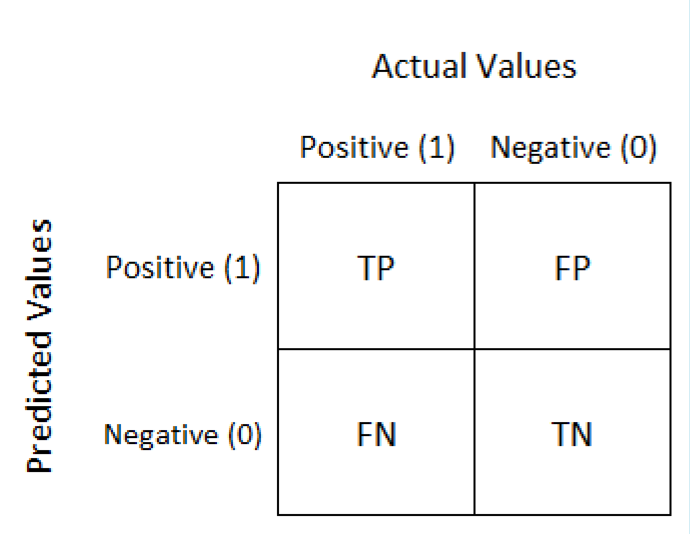


Subsampling your data with caret

```r
pts.df$y <- as.factor(ifelse(pts.df$y == 1, "Yes", "No"))
library(caret)
Train <- createDataPartition(pts.df$y, p=0.6, list=FALSE)

training <- pts.df[ Train, ]
testing <- pts.df[ -Train, ]
```

Misclassification

- Confusion matrices compare actual values to predictions



		Actual Values	
		Positive (1)	Negative (0)
Predicted Values	Positive (1)	TP	FP
	Negative (0)	FN	TN

- True Positive (TN) - This is correctly classified as the class if interest / target.
- True Negative (TN) - This is correctly classified as not a class of interest / target.
- False Positive (FP) - This is wrongly classified as the class of interest / target.
- False Negative (FN) - This is wrongly classified as not a class of interest / target.

Confusion Matrices in R

```
train.log <- glm(y ~ .,
                 family="binomial"
                 data=training[,2:

predicted.log <- predict(train.log
                         newdata=t
                         type="res

pred <- as.factor(
  ifelse(predicted.log > 0.5,
                         "Yes",
                         "No"))
```

```
confusionMatrix(testing$y, pred)
```

```
Confusion Matrix and Statistics

          Reference
Prediction No Yes
       No  27   0
       Yes  2  10

               Accuracy : 0.9487
                 95% CI : (0.8268, 0.9937)
    No Information Rate : 0.7436
    P-Value [Acc > NIR] : 0.0009839

                  Kappa : 0.8738

 Mcnemar's Test P-Value : 0.4795001

            Sensitivity : 0.9310
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.8333
             Prevalence : 0.7436
         Detection Rate : 0.6923
```

Confusion Matrices

Depends upon threshold!!

Confusion Matrices in R

```
library(tree)
tree.model <- tree(y ~ . , trainin
predict.tree <- predict(tree.model
```

```
confusionMatrix(testing$y, predict.tree)
```

```
Confusion Matrix and Statistics

          Reference
Prediction No Yes
       No  21   6
       Yes  1  11

               Accuracy : 0.8205
                 95% CI : (0.6647, 0.9246)
    No Information Rate : 0.5641
    P-Value [Acc > NIR] : 0.0006866

                  Kappa : 0.6224

 Mcnemar's Test P-Value : 0.1305700

            Sensitivity : 0.9545
            Specificity : 0.6471
         Pos Pred Value : 0.7778
         Neg Pred Value : 0.9167
             Prevalence : 0.5641
         Detection Rate : 0.5385
```

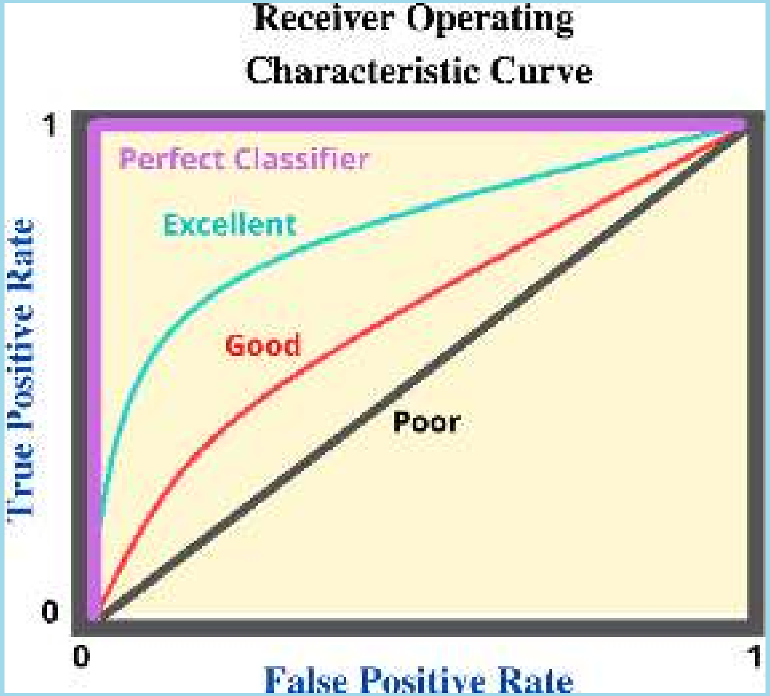
Confusion Matrices in R

```r
library(randomForest)
class.model <- y ~ .
rf <- randomForest(class.model, da
predict.rf <- predict(rf, newdata=
```

```r
confusionMatrix(testing$y, predict.rf)
```

```
Confusion Matrix and Statistics

          Reference
Prediction No Yes
       No  22   5
       Yes  1  11

               Accuracy : 0.8462
                 95% CI : (0.6947, 0.9414)
    No Information Rate : 0.5897
    P-Value [Acc > NIR] : 0.000553

                  Kappa : 0.6695

 Mcnemar's Test P-Value : 0.220671

            Sensitivity : 0.9565
            Specificity : 0.6875
         Pos Pred Value : 0.8148
         Neg Pred Value : 0.9167
             Prevalence : 0.5897
         Detection Rate : 0.5641
```

Threshold-Free Methods



- Receiver Operating Characteristic Curves
- Illustrates discrimination of binary classifier as the threshold is varied
- Area Under the Curve (AUC) provides an estimate of classification ability

Criticisms of ROC/AUC

- Treats false positives and false negatives equally
- Undervalues models that predict across smaller geographies
- Focus on *discrimination* and not *calibration*
- New methods for presence-only data

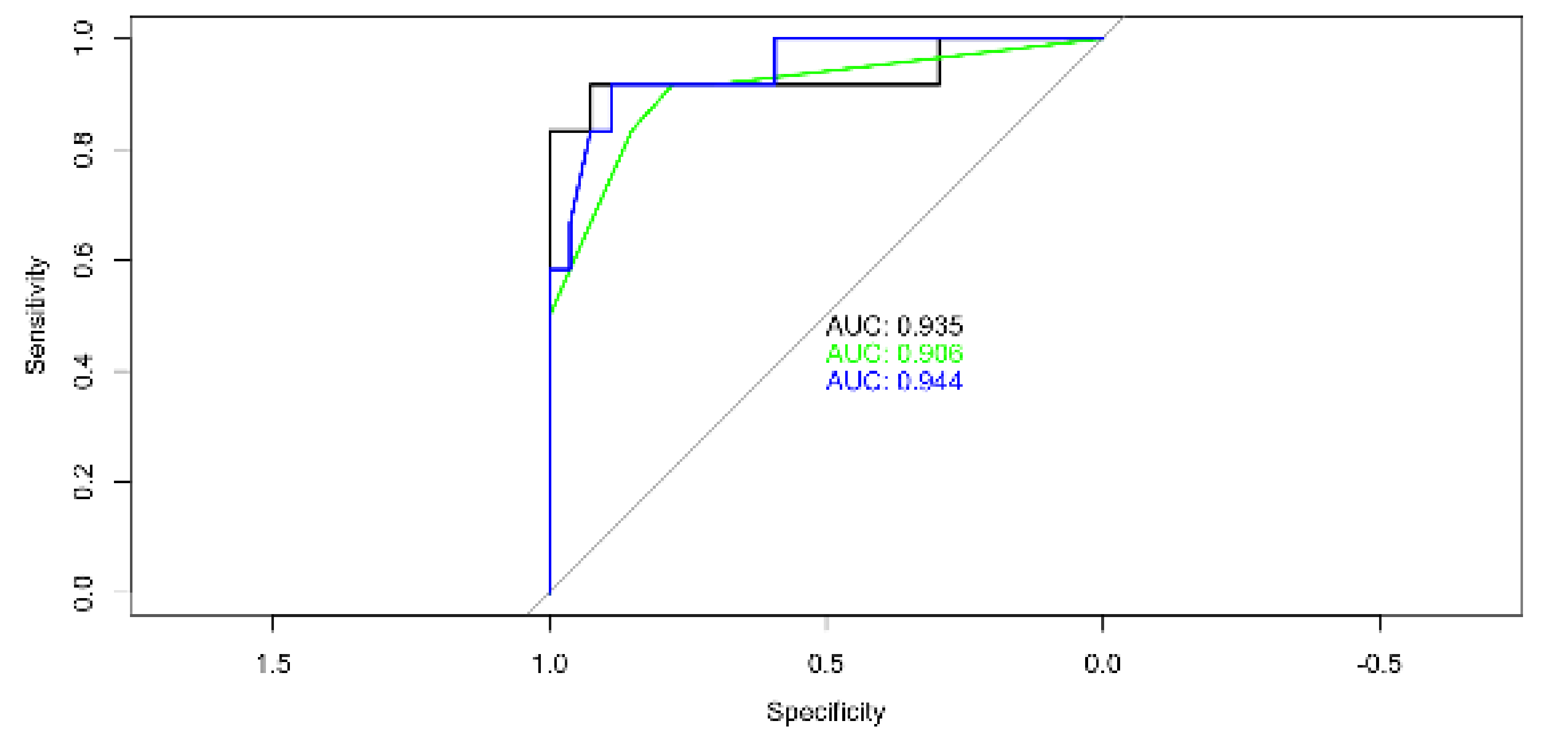
ROC in R (using pROC)

- Generate predictions (note the difference for tree and rf)

```r
library(pROC)
train.log <- glm(y ~ .,
                 family="binomial",
                 data=training[,2:8])

predicted.log <- predict(train.log,
                         newdata=testing[,2:8],
                         type="response")

predict.tree <- predict(tree.model, newdata=testing[,2:8], type="vector")[,

predict.rf <- predict(rf, newdata=testing[,2:8], type="prob")[,2]
```

ROC in R (using pROC)

```
plot(roc(testing$y, predicted.log), print.auc=TRUE)

plot(roc(testing$y, predict.tree), print.auc=TRUE, print.auc.y = 0.45, col=

plot(roc(testing$y, predict.rf), print.auc=TRUE, print.auc.y = 0.4, col="bl
```



Cross-validation

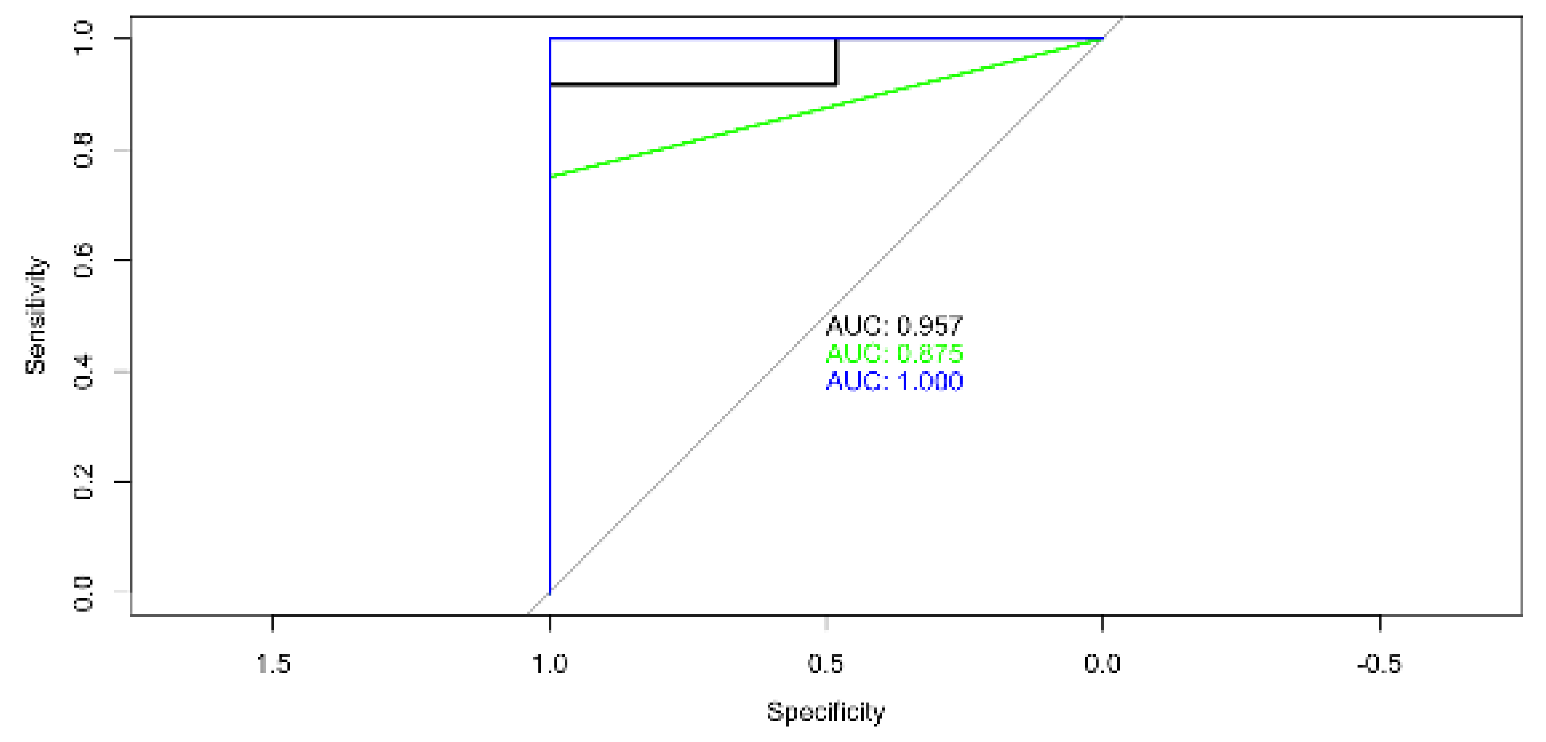
- Often want to make sure that fit/accuracy not a function of partition choice
- Cross-validation allows resampling of data (multiple times)
- K-fold - Data are split into K datasets of ~ equal size, model fit to observations to predict heldout set
- Leave One Out (LOO) - Model fit to n-1 observations to predict the held out observation

Crossvalidation in R using caret

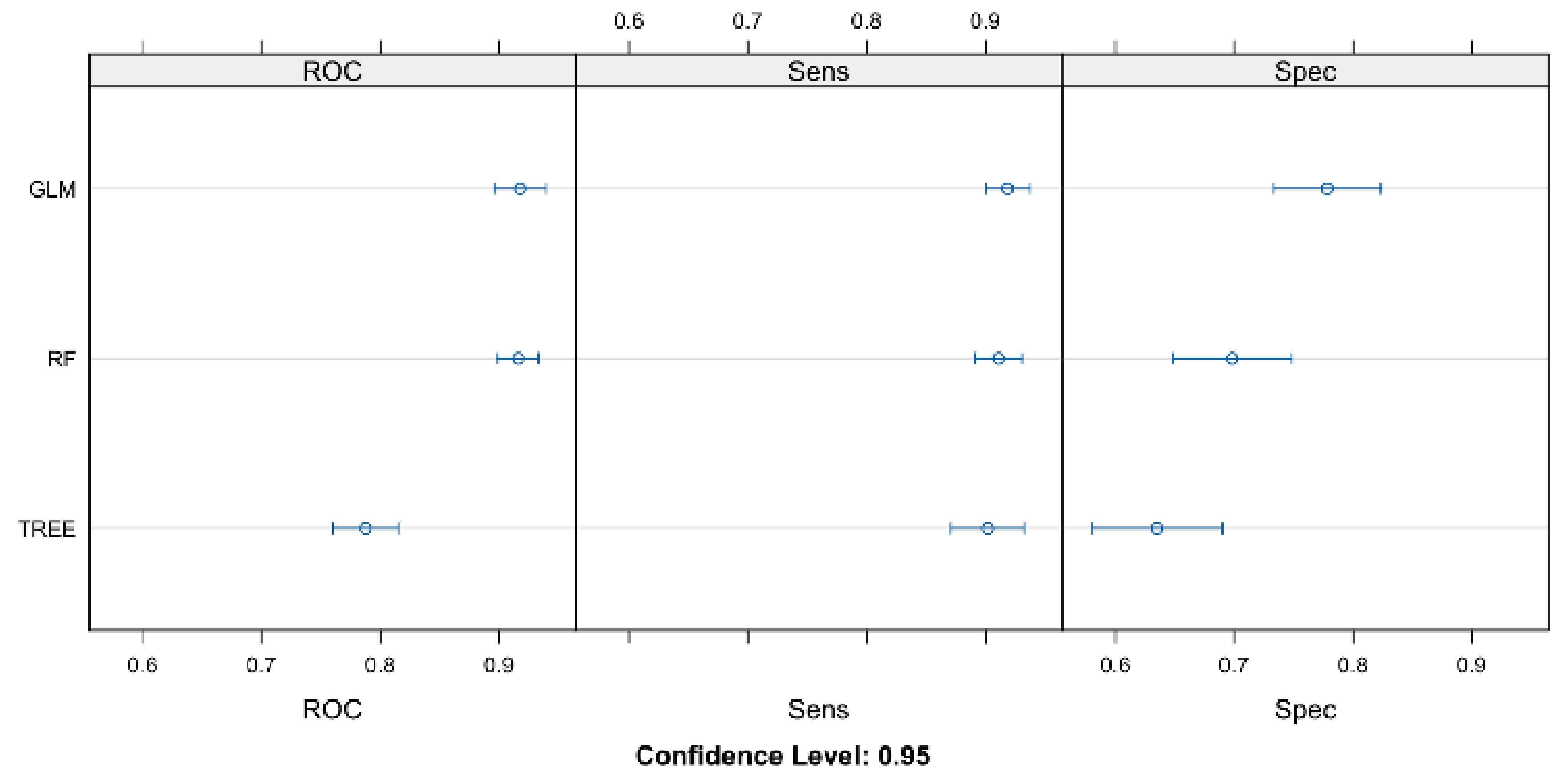
```r
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 10,
                           classProbs = TRUE,
                           summaryFunction = twoClassSummary)

log.model <- train(y ~., data = pts.df[,2:8],
               method = "glm",
               trControl = fitControl)
pred.log <- predict(log.model, newdata = testing[,2:8], type="prob")[,2]

tree.model <- train(y ~., data = pts.df[,2:8],
               method = "rpart",
               trControl = fitControl)

pred.tree <- predict(tree.model, newdata=testing[,2:8], type="prob")[,2]

rf.model <- train(y ~., data = pts.df[,2:8],
```

Crossvalidation in R using caret

```r
plot(roc(testing$y, pred.log), print.auc=TRUE)

plot(roc(testing$y, pred.tree), print.auc=TRUE, print.auc.y = 0.45, col="gr

plot(roc(testing$y, pred.rf), print.auc=TRUE, print.auc.y = 0.4, col="blue"
```



Crossvalidation in R using caret



Spatial predictions

```r
best.rf <- rf.model$finalModel
best.glm <- log.model$finalModel

rf.spatial <- terra::predict(pred.stack.scl, best.rf, type="prob")


glm.spatial <- terra::predict(pred.stack.scl, best.glm,type="response" )
```

Spatial predictions