Statistical Modelling III

HES 505 Fall 2023: Session 24

Matt Williamson

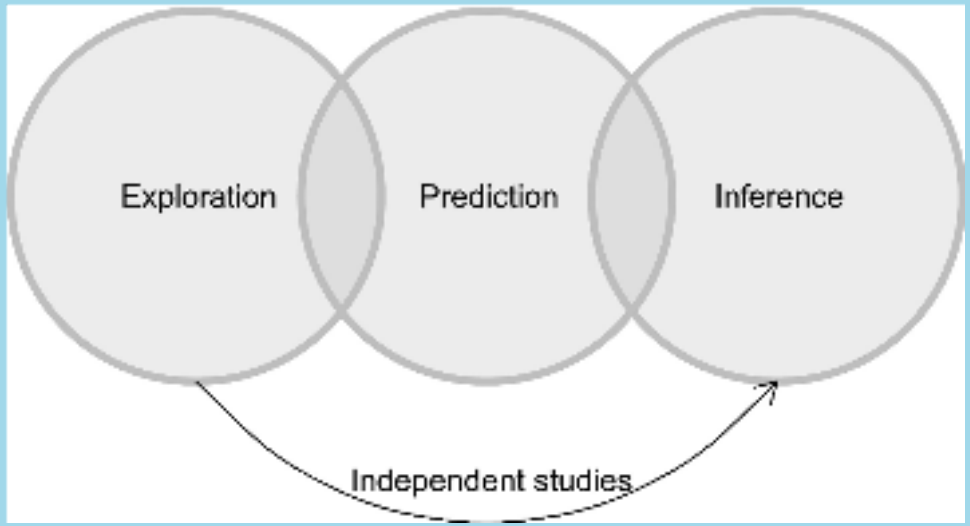
Objectives

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 $\hat{\beta}$ given the data we have
- **Prediction** is focused on accurate forecasts of $\hat{y}$ at locations where we have yet to collect the data

Inference and Presence/Absence Data

- $\hat{\beta}$ is conditional on variables in the model **and** those not in the model

```
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

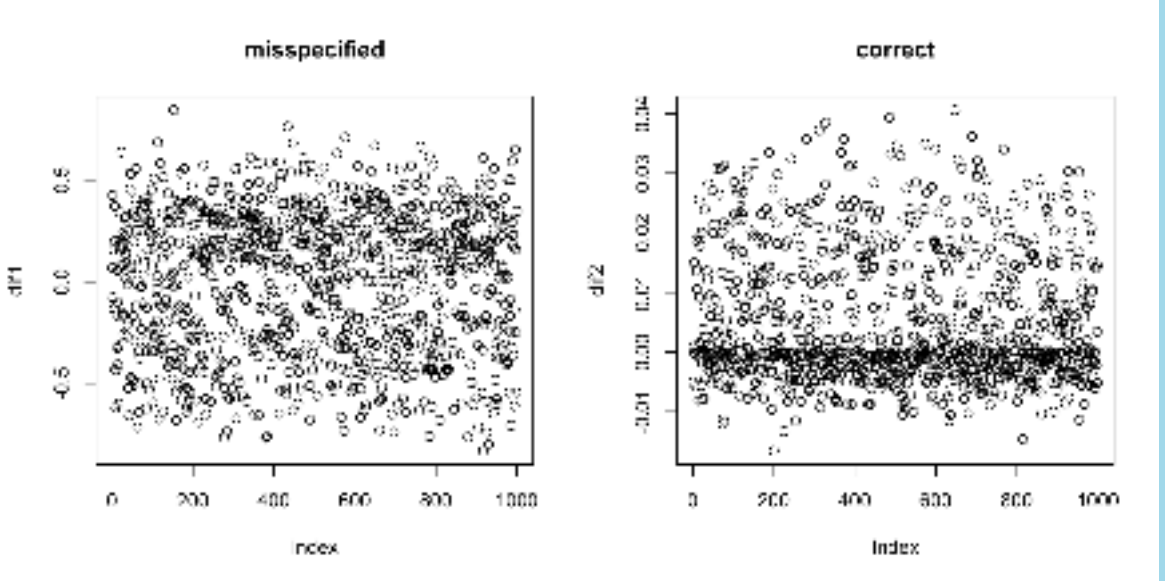
```
coef(mod1)
```

```
(Intercept)          x1
x2
 0.36077251  0.88013286
-0.06476286
```

```
coef(mod2)
```

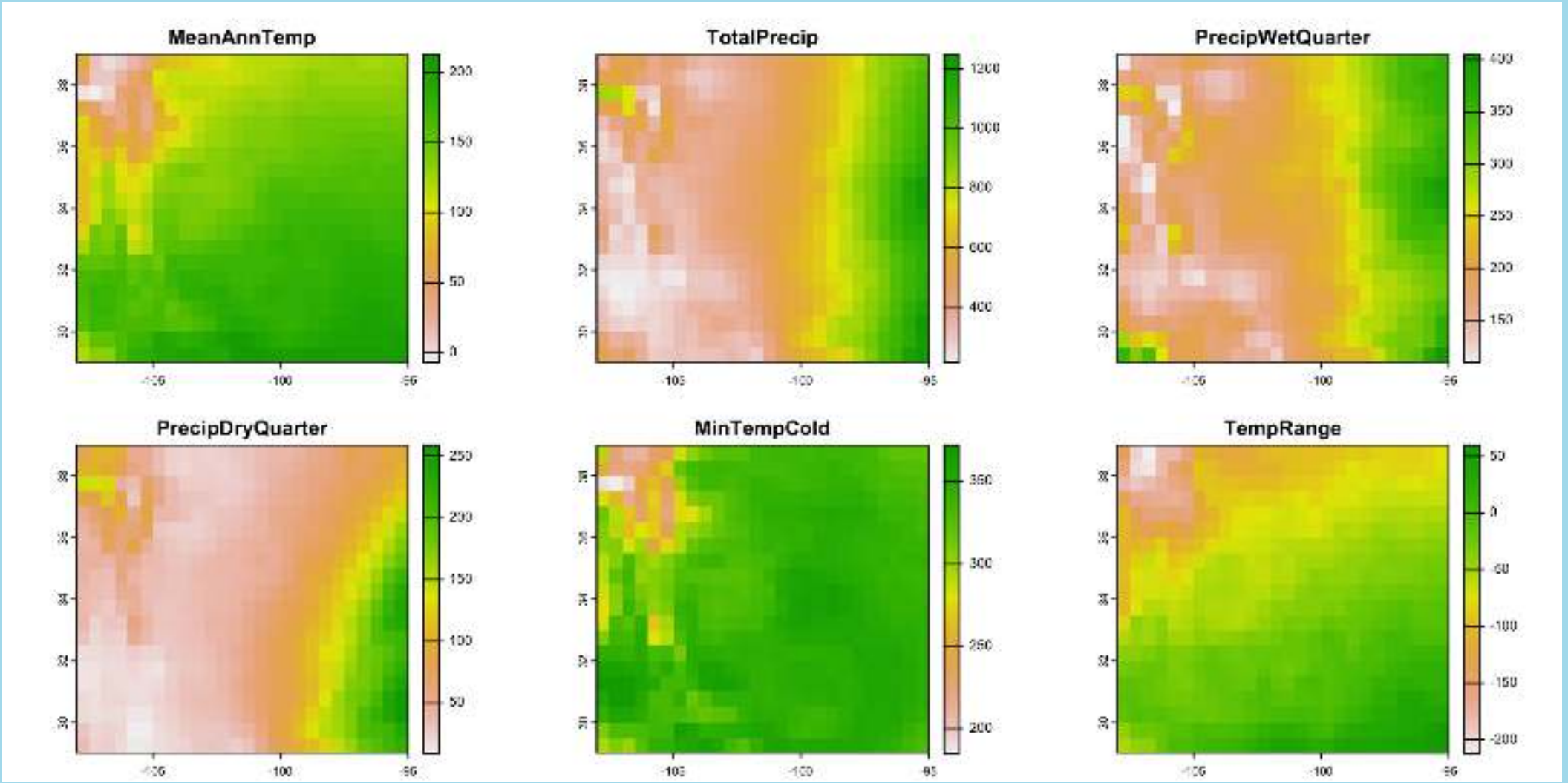
```
(Intercept)          x1
x2          x3
  0.9913699   2.1443776
-0.1659338  -3.7029481
```

```
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
TotalPrecip
PrecipWetQuarter
PrecipDryQuarter
MinTempCold
TempRange

Using Test Statistics

- R^2 for linear regression:

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

$$SS_{res} = \sum_i (y_i - f_i)^2$$

$$SS_{tot} = \sum_i (y_i - y)^2$$

- Perfect prediction ($f_i = y_i$); $SS_{res} = 0$; and $R^2 = 1$
- Null prediction (Intercept only) ($f_i = y$); $SS_{res} = SS_{tot}$; and $R^2 = 0$
- No direct way of implementing for logistic regression

Pseudo- R^2

$$R_L^2 = \frac{D_{null} - D_{fitted}}{D_{null}}$$

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

Cohen's Likelihood Ratio

```
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- R^2

$$R^2_{CS} = 1 - \left(\frac{L_0}{L_M}\right)^{(2/n)}$$
$$= 1 - \exp^{2(\ln(L_0) - \ln(L_M))/n}$$

- Cox and Snell R^2
- Likelihood (L), the probability of observing the sample given an assumed distribution
- Challenge: Maximum value is less than 1 and changes with n
- Correction by Nagelkerke so that maximum is 1

Cox and Snell R^2

```
1 logistic.null <- glm(y ~ 1,
2                       family=binomial(link="logit"),
3                       data=pts.df[,2:8])
4
5 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 R^2
- 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)

$$AIC = -2\ln(\hat{L}) + 2k$$

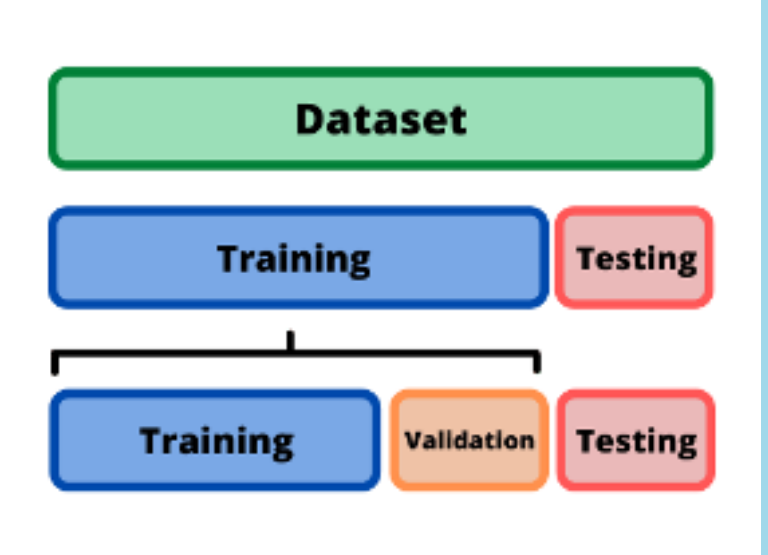
AIC Comparison

```
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)



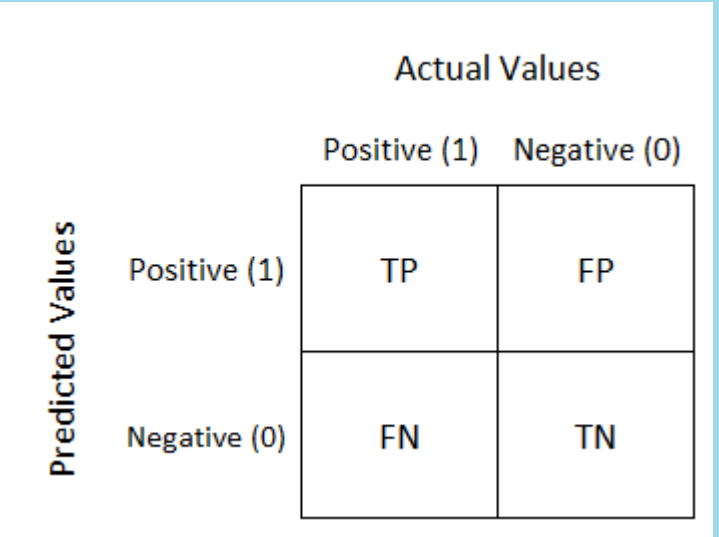
Subsampling your data with caret

```
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

- 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  26   1
       Yes  4   8

               Accuracy : 0.8718
                 95% CI : (0.7257, 0.957)
    No Information Rate : 0.7692
    P-Value [Acc > NIR] : 0.08607

                  Kappa : 0.6766

 Mcnemar's Test P-Value : 0.37109

            Sensitivity : 0.8667
            Specificity : 0.8889
         Pos Pred Value : 0.9630
         Neg Pred Value : 0.6667
             Prevalence : 0.7692
         Detection Rate : 0.6667
```

Confusion Matrices

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

$$\text{Sensitivity} = \frac{TP}{TP + FN}$$

$$\text{Specificity} = \frac{TN}{FP + TN}$$

$$\text{Precision} = \frac{TP}{TP + FP}$$

$$\text{Recall} = \frac{TP}{TP + FN}$$

Depends upon threshold!!

Confusion Matrices in R

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

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

```
Confusion Matrix and Statistics

          Reference
Prediction No Yes
       No  26   1
       Yes  5   7

               Accuracy : 0.8462
                 95% CI : (0.6947, 0.9414)
    No Information Rate : 0.7949
    P-Value [Acc > NIR] : 0.2851

                  Kappa : 0.602

 Mcnemar's Test P-Value : 0.2207

            Sensitivity : 0.8387
            Specificity : 0.8750
         Pos Pred Value : 0.9630
         Neg Pred Value : 0.5833
             Prevalence : 0.7949
         Detection Rate : 0.6667
```

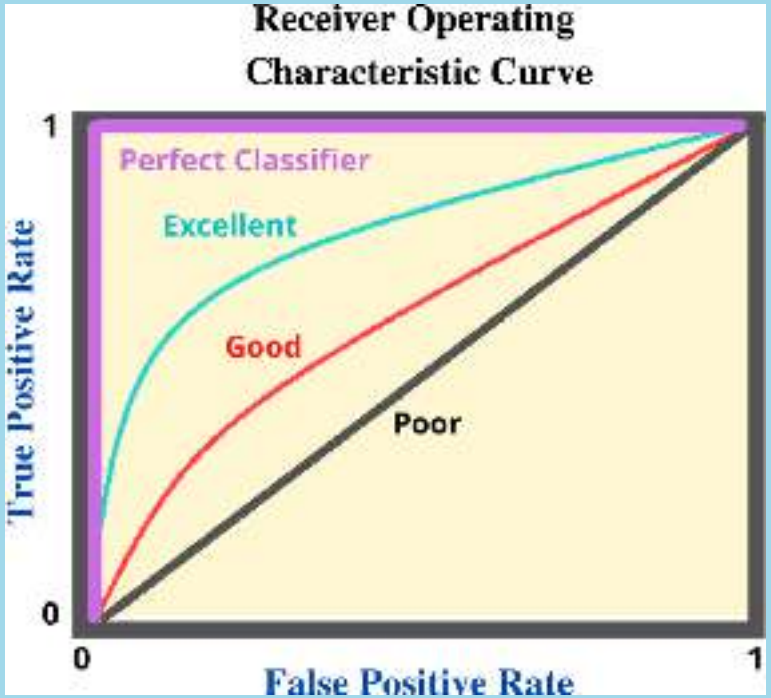
Confusion Matrices in R

```
1 library(randomForest)
2 class.model <- y ~ .
3 rf <- randomForest(class.model, da
4 predict.rf <- predict(rf, newdata=
```

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

```
Confusion Matrix and Statistics

          Reference
Prediction No Yes
       No  27   0
       Yes  5   7

               Accuracy : 0.8718
                 95% CI : (0.7257, 0.957)
    No Information Rate : 0.8205
    P-Value [Acc > NIR] : 0.27535

                  Kappa : 0.6597

 Mcnemar's Test P-Value : 0.07364

            Sensitivity : 0.8438
            Specificity : 1.0000
         Pos Pred Value : 1.0000
         Neg Pred Value : 0.5833
             Prevalence : 0.8205
         Detection Rate : 0.6923
```

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

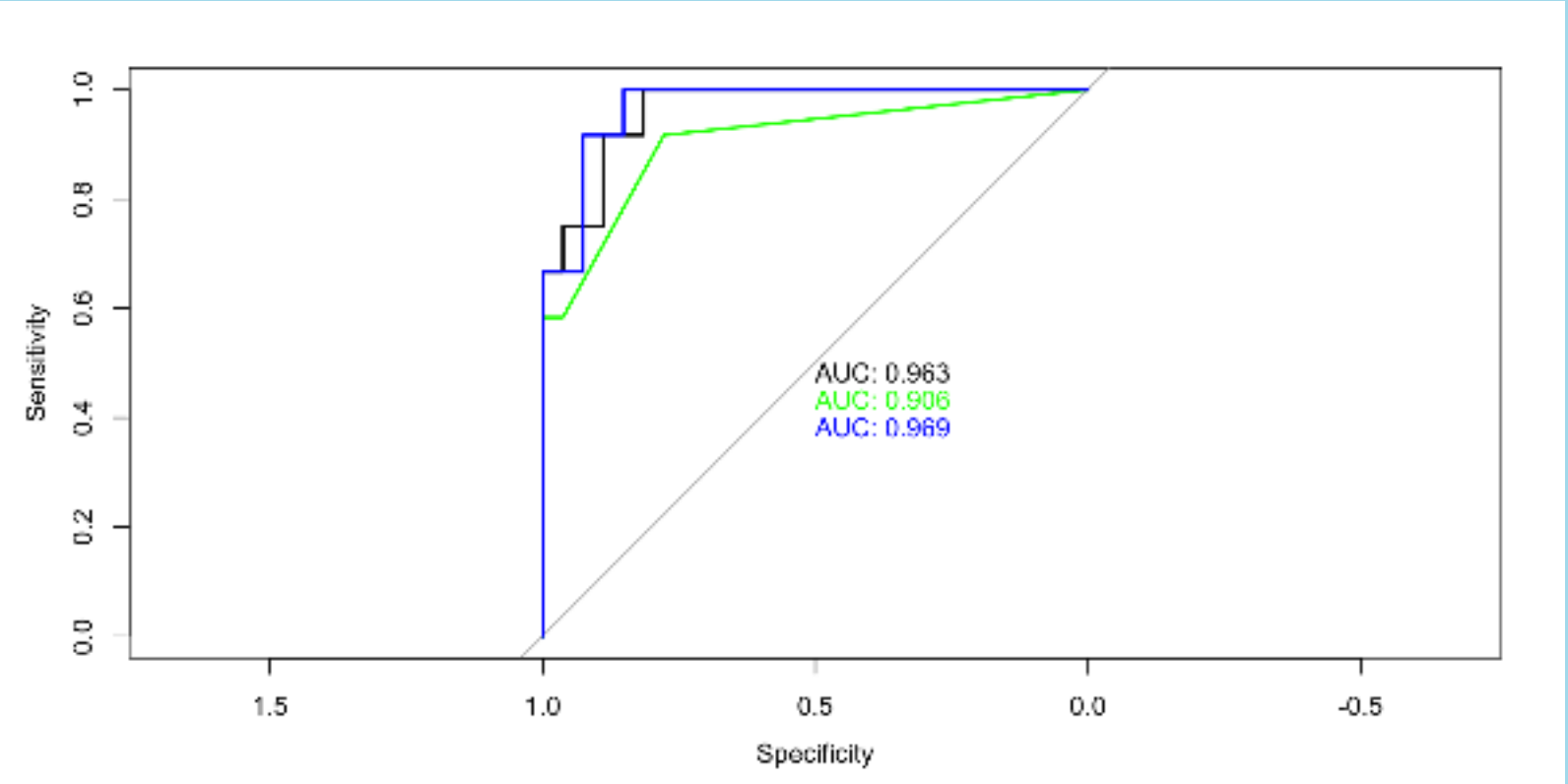
ROC in R (using pROC)

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

```
1  library(pROC)
2  train.log <- glm(y ~ .,
3                   family="binomial",
4                   data=training[,2:8])
5
6  predicted.log <- predict(train.log,
7                           newdata=testing[,2:8],
8                           type="response")
9
10 predict.tree <- predict(tree.model, newdata=testing[,2:8], type="vector")[,
11
12 predict.rf <- predict(rf, newdata=testing[,2:8], type="prob")[,2]
```

ROC in R (using pROC)

```
1  plot(roc(testing$y, predicted.log), print.auc=TRUE)
2
3  plot(roc(testing$y, predict.tree), print.auc=TRUE, print.auc.y = 0.45, col=
4
5  plot(roc(testing$y, predict.rf), print.auc=TRUE, print.auc.y = 0.4, col="bl
```



Cross-validation

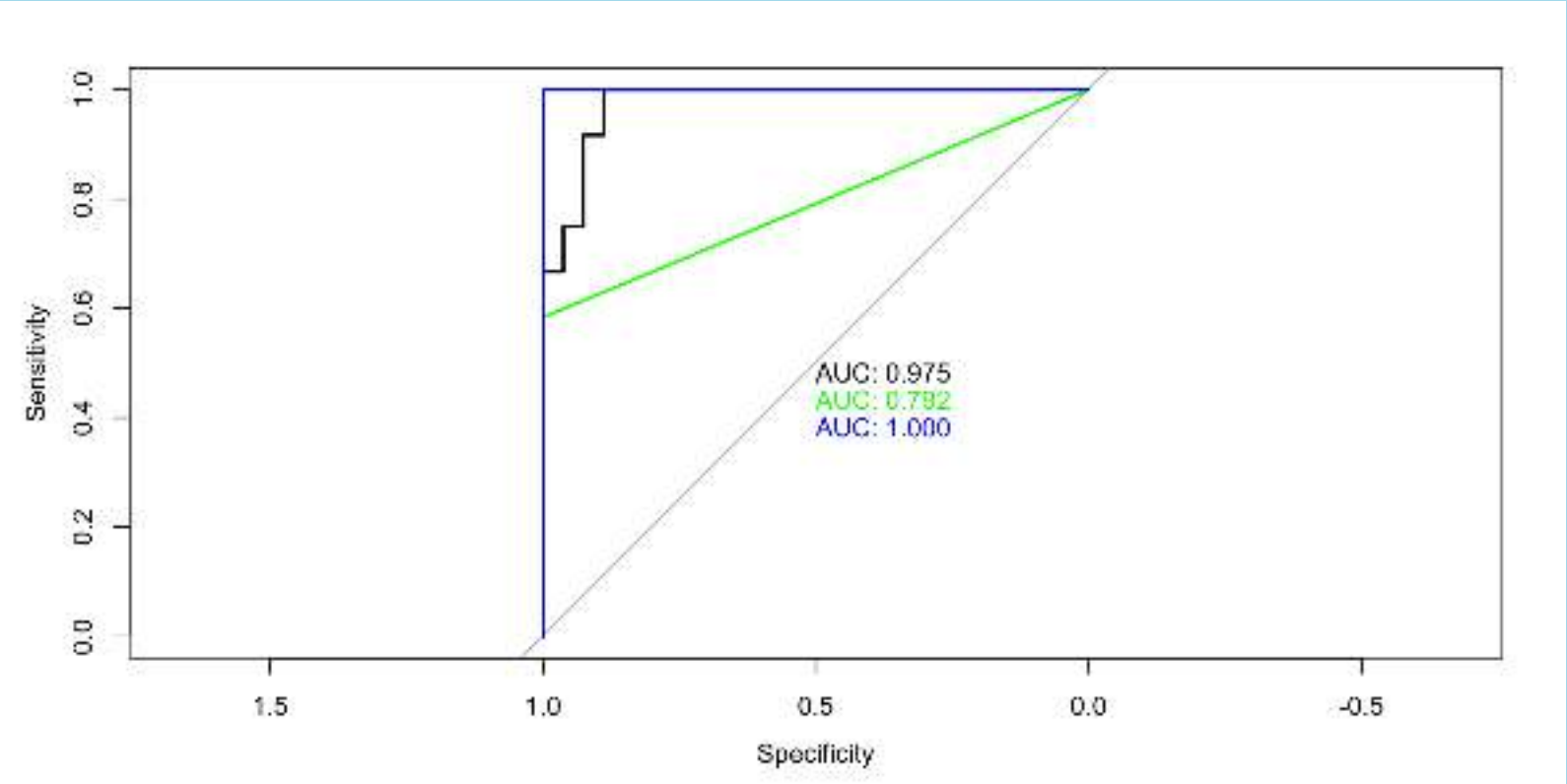
- Often want to make sure that fit/accuracy not a function of partition choice
- Cross-validation allows resampling of data (multiple times)
- K-fold - Data are split into K datasets of ~ equal size, model fit to $(K-1)(\frac{n}{K})$ observations to predict heldout set
- Leave One Out (LOO) - Model fit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

Crossvalidation in R using caret

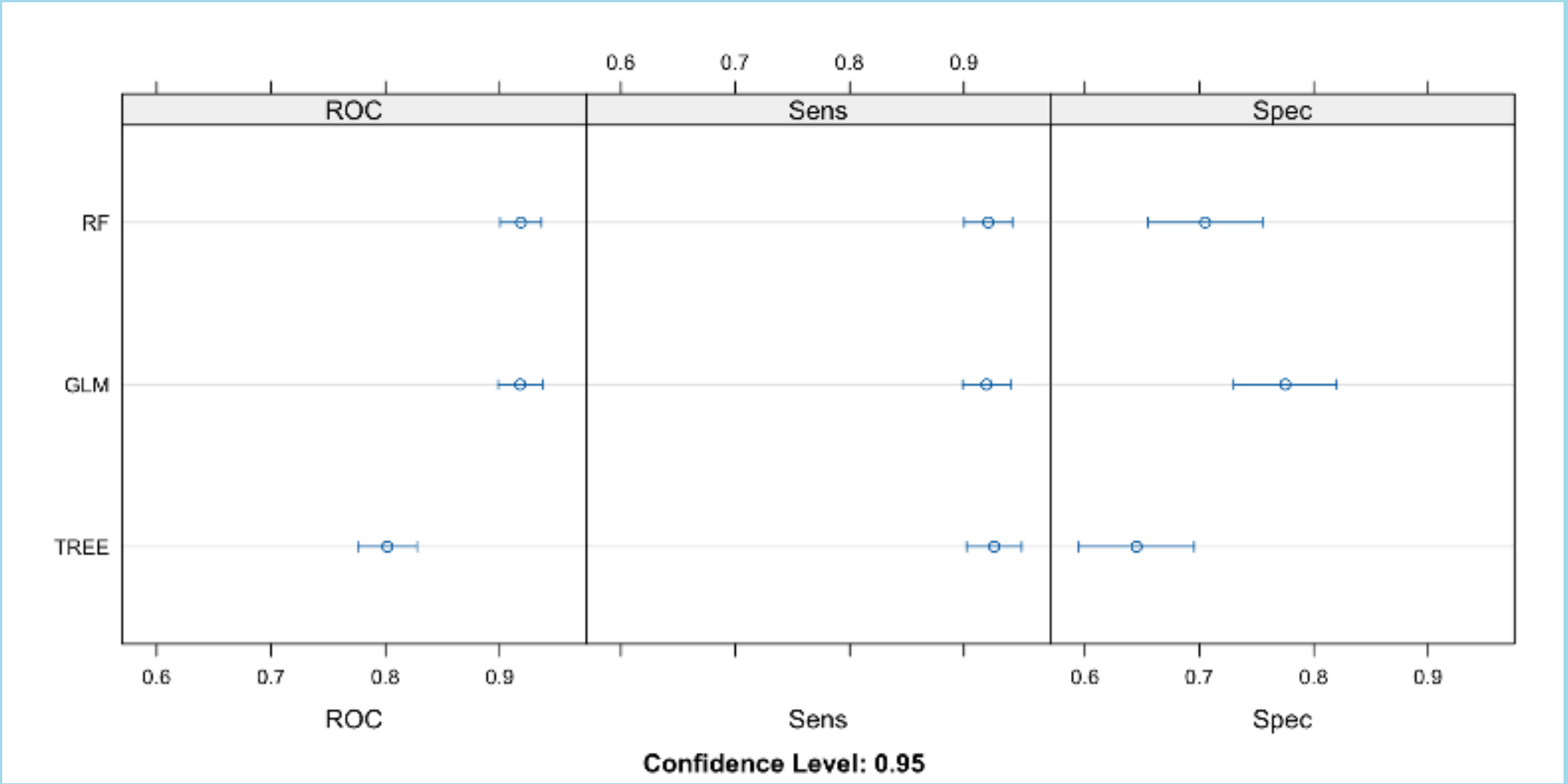
```
fitControl <- trainControl(method = "repeatedcv",
                           number = 10,
                           repeats = 10,
                           classProbs = TRUE,
                           summaryFunction = twoClassSummary)

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

```
1 plot(roc(testing$y, pred.log), print.auc=TRUE)
2
3 plot(roc(testing$y, pred.tree), print.auc=TRUE, print.auc.y = 0.45, col="gr
4
5 plot(roc(testing$y, pred.rf), print.auc=TRUE, print.auc.y = 0.4, col="blue"
```



Crossvalidation in R using caret



Spatial predictions

```
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