Multivariate Analysis III

## HES 505 Fall 2022: Session 21

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?
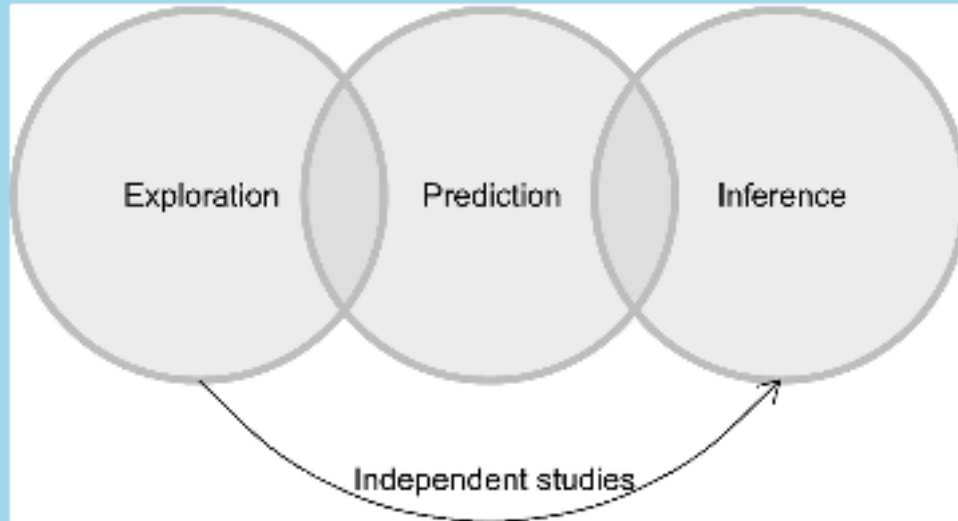
from Tradennick et al. 2021

- **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

```r
 1  nsamp <- 1000
 2  df <- data.frame(x1 = rnorm(nsamp,0,1),
 3                   x2 = rnorm(nsamp,0,1),
 4                   x3 = rnorm(nsamp,0,1))
 5
 6  linpred <- 1 + 2*df$x1 -0.18*df$x2 -3.5*df$x3
 7  y <- rbinom(nsamp, 1, plogis(linpred))
 8  df <- cbind(df, y)
 9
10  mod1 <- glm(y~x1 +x2, data=df, family="binomial")
11  mod2 <- glm(y~x1 +x2 + x3, data=df, family="binomial")
```

Inference & Presence/Absence Data

```
1  coef(mod1)
2  coef(mod2)
```

```
1  prd1 <- predict(mod1, df, "response")
2  dif1 <- plogis(linpred) - prd1
3  prd2 <- predict(mod2, df, "response")
4  dif2 <- plogis(linpred) - prd2
```

Inferring coefficient effects requires that your model fit the data well

Assessing Model Fit

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

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

- 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

```r
1  logistic.rich <- glm(y ~ MeanAnnTemp + PrecipWetQuarter + PrecipDryQuarter,
2                       family=binomial(link="logit"),
3                       data=pts.df[,2:8])
4
5  with(logistic.rich,
6      null.deviance - deviance)/with(logistic.rich,
7                                     null.deviance)
```

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

```r
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
```

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 $\neq$ good)

- Theoretical and subsampling methods

Theoretical Assessment of Predictive Performance

- 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)

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

Hirotugu Akaike of AIC

AIC Comparison

```r
1  logistic.null <- glm(y ~ 1,
2                       family=binomial(link="logit"),
3                       data=pts.df[,2:8])
4
5  logistic.rich <- glm(y ~ MeanAnnTemp + PrecipWetQuarter + PrecipDryQuarter,
6                       family=binomial(link="logit"),
7                       data=pts.df[,2:8])
8
9  AIC(logistic.null, logistic.rich)
```

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)
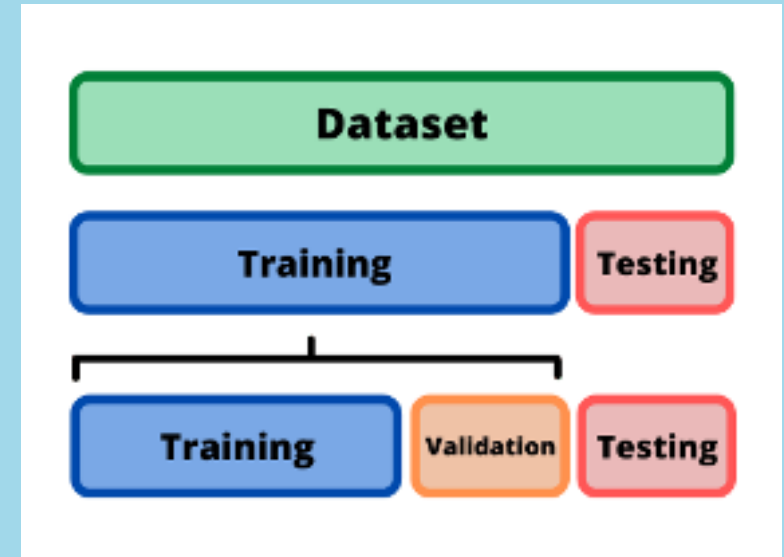
Subsampling your data with **caret**

```r
1  pts.df$y <- as.factor(ifelse(pts.df$y == 1, "Yes", "No"))
2  library(caret)
3  Train <- createDataPartition(pts.df$y, p=0.6, list=FALSE)
4
5  training <- pts.df[ Train, ]
6  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.

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

Predicted Values

Confusion Matrices in R

```r
1  train.log <- glm(y ~ .,
2                   family="binomial"
3                   data=training[,2:
4
5  predicted.log <- predict(train.log
6                           newdata=t
7                           type="res
8
9  pred <- as.factor(
10    ifelse(predicted.log > 0.5,
11                          "Yes",
12                          "No"))
```

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

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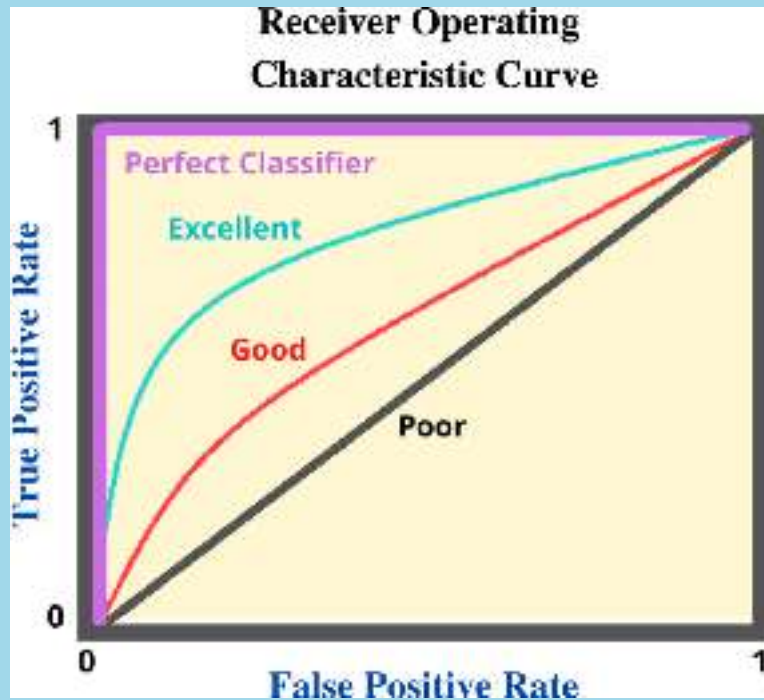
```
1  confusionMatrix(testing$y, predict.tree)
```

Confusion Matrices in R

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

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

29

Threshold-Free Methods

Receiver Operating Characteristic Curve

- 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

Cross-validation

Spatial predictions…