STATS 330

Handout 15
Using Regression Models for Prediction

Department of Statistics, University of Auckland

15.1 Outline I

15.1 Outline

15.2 Prediction

15.3 What's Coming Up?
15.3.1 Evaporation Data Example

15.4 Using GLMs for Prediction
15.4.1 Age of Abalone Example

15.5 Prediction for Individuals
15.5.1 BIRADS Example
15.5.2 Gardasil Example

15.2 Prediction

Prediction is one of the two most common uses of a regression model—the other is explanation.

There is an old Danish proverb:

> *Prediction is difficult, especially when dealing with the future.*

Or there is the following quote from Nostradamus:

> *For a long time, I have been making many predictions, far in advance, of events since come to pass, naming the particular locality. I acknowledge all to have been accomplished through divine power and inspiration.*

Those of us who do not possess "divine power and inspiration" should take the Danish proverb seriously.

Prediction

The purpose of a predictive regression model is (obviously) to make predictions:

- The value of the response Y is predicted for a *new* observation based on values of the explanatory variables and the fitted regression model.
- The performance of a predictive model is determined by how accurately and precisely it predicts the response for *new* observations.
- Notice the emphasis on *new* observations—the current data set is used to fit the predictive model but the purpose is to predict observations that are not included in the data.

Examples of Regression Analysis used for Prediction

Some examples that we will explore:

1. Predicting soil evaporation.
   - Ordinary linear regression is used to predict the rate of evaporation based on meteorological conditions.
2. Predicting the age of abalone.
   - Poisson regression is used to predict the age of individual abalone based on physical measurements.
3. Predicting whether or not an individual patient will complete the Gardasil vaccination programme.
   - Logistic regression is used to estimate the probability of completion based on available information about the patient.

Predictions for New Observations I

In each case, the regression model is to be used to predict the response for observations not included in the data set. These predictions may not be valid if the new observations are not consistent with the data used to fit the model.

- The data used to produce the soil evaporation data was collected at a site in Texas, USA, over 46 consecutive days (6 June through 21 July).
  - Risky to use this data to predict evaporation at a different site or at the same site at a different time of year.

Predictions for New Observations II

- The abalone (pāua) data represents measurements made on blacklip abalone collected from the North Coast and Islands of Bass Strait (Tasmania).
  - Using this data to predict the age of other species of abalone or the same species from a different region is problematic.
- The Gardasil pertains to female patients between the ages of 11 and 26 who visited one of four Johns Hopkins Medical Institutions in Maryland, USA.
  - We should only make predictions for future observations that share these characteristics.

The Utopian Case

The ideal situation would be:

1. There is a well defined population for which we want to be able to make predictions.

2. We take a random sample from this population (as described in STATS 340) and use this data to create the regression model.

3. This will ensure that the data used to create the predictive model is consistent with the new observations we wish to predict.

Unfortunately in practice, data is often collected on the basis of convenience without consideration of a target population or of obtaining a representative sample from this population. We can only deduce a reasonable target population based on the characteristics of the data.

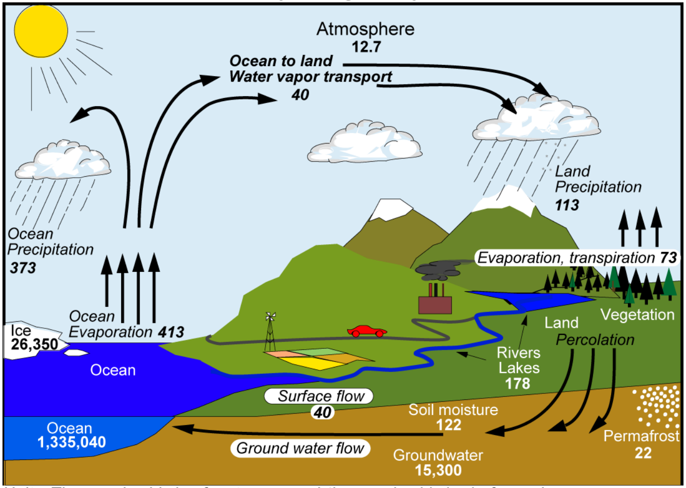
15.3 What's Coming Up?

First we will look at an example where the response is continuous using the Texas water evaporation data.

- First we will consider how the data was collected and take a preliminary look at the data.
- Then we will fit some preliminary models and discuss how we evaluate the predictive ability of a model.
- Finally, we will search for a good model and assess its predictive ability.

15.3.1 Evaporation Data Example

Hydrological Cycle



Units: Thousand cubic km for storage, and *thousand cubic km/yr* for exchanges

Evaporation Data

Data was recorded for factors affecting the evaporation water from a location in Texas, USA.

> *Source: Freund, R. J. (1979). "Multicollinearity etc., some 'new' examples", Proceedings of ASA Statistical Computing Section. Washington DC, USA: American Statistical Association, pp.111–2.*

- Ten variables were measured each day over a period of 46 days.
- The goal was to create a model that relates the daily amount of evaporation to the meteorological conditions that day.

Evaporation Example Variables

Evap:	daily total evaporation from soil (units not given).
MaxST:	maximum soil temperature ($^\circ$F).
MinST:	minimum soil temperature ($^\circ$F).
AvST:	average soil temperature (area under curve).
MaxAT:	maximum air temperature ($^\circ$F).
MinAT:	minimum air temperature ($^\circ$F).
AvAT:	average air temperature (area under curve).
MaxH:	maximum humidity (%).
MinH:	minimum humidity (%).
AvH:	average humidity (area under curve).
Wind:	total wind (miles per day).

The Evaporation Data

```r
library('TeachingDemos')
data(evap)
evap.df = evap[, -(1:3)]
head(evap.df, 5)
```

```
##   MaxST MinST AvST MaxAT MinAT AvAT MaxH MinH AvH Wind Evap
## 1    84    65  147    85    59  151   95   40 398  273   30
## 2    84    65  149    86    61  159   94   28 345  140   34
## 3    79    66  142    83    64  152   94   41 388  318   33
## 4    81    67  147    83    65  158   94   50 406  202   26
## 5    84    68  167    88    69  180   93   46 379  311   41
```

```r
str(evap.df)
```

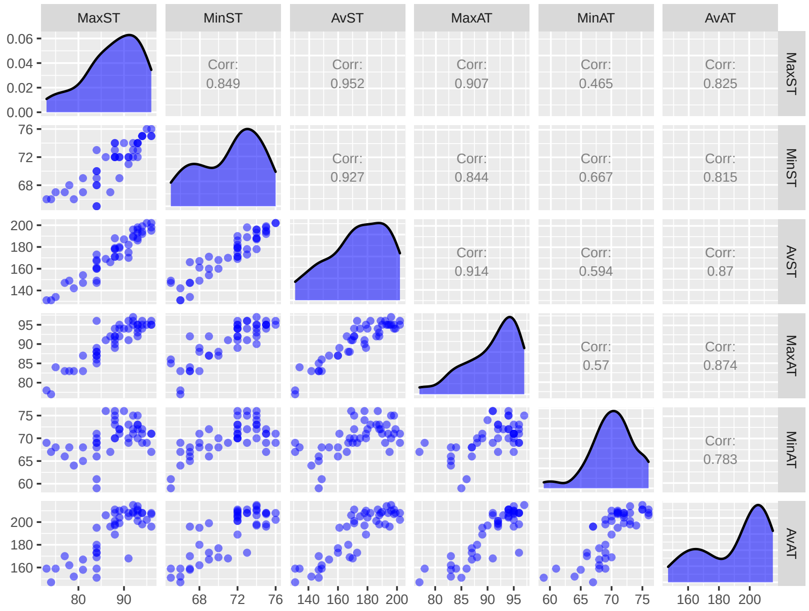
```
## 'data.frame': 46 obs. of  11 variables:
##  $ MaxST: num  84 84 79 81 84 74 73 75 84 86 ...
##  $ MinST: num  65 65 66 67 68 66 66 67 68 72 ...
##  $ AvST : num  147 149 142 147 167 131 131 134 161 169 ...
##  $ MaxAT: num  85 86 83 83 88 77 78 84 89 91 ...
##  $ MinAT: num  59 61 64 65 69 67 69 68 71 76 ...
##  $ AvAT : num  151 159 152 158 180 147 159 159 195 206 ...
##  $ MaxH : num  95 94 94 94 93 96 96 95 95 93 ...
##  $ MinH : num  40 28 41 50 46 73 72 70 63 56 ...
##  $ AvH  : num  398 345 388 406 379 478 462 461 430 406 ...
##  $ Wind : num  273 140 318 202 311 446 294 313 455 604 ...
##  $ Evap : num  30 34 33 26 41 4 5 20 31 38 ...
```

Correlation Matrix of the Regressors

```
round(cor(evap.df[, -11]), 2)

##        MaxST MinST  AvST MaxAT MinAT  AvAT  MaxH  MinH   AvH  Wind
## MaxST   1.00  0.85  0.95  0.91  0.47  0.83 -0.21 -0.67 -0.76 -0.08
## MinST   0.85  1.00  0.93  0.84  0.67  0.82 -0.16 -0.34 -0.47  0.02
## AvST    0.95  0.93  1.00  0.91  0.59  0.87 -0.19 -0.53 -0.68 -0.08
## MaxAT   0.91  0.84  0.91  1.00  0.57  0.87 -0.11 -0.53 -0.66 -0.07
## MinAT   0.47  0.67  0.59  0.57  1.00  0.78 -0.12  0.19 -0.06  0.43
## AvAT    0.83  0.82  0.87  0.87  0.78  1.00 -0.05 -0.31 -0.54  0.15
## MaxH   -0.21 -0.16 -0.19 -0.11 -0.12 -0.05  1.00  0.16  0.29 -0.13
## MinH   -0.67 -0.34 -0.53 -0.53  0.19 -0.31  0.16  1.00  0.91  0.34
## AvH    -0.76 -0.47 -0.68 -0.66 -0.06 -0.54  0.29  0.91  1.00  0.22
## Wind   -0.08  0.02 -0.08 -0.07  0.43  0.15 -0.13  0.34  0.22  1.00
```

- Note the high correlations between some pairs of regressors (especially those involving temperature).

Pairs Plot for Temperature Variables



The Fitted Evaporation Model

```
evap.lm = lm(Evap ~ ., data = evap.df)
print(coef(summary(evap.lm)))

##                Estimate Std. Error t value  Pr(>|t|)
## (Intercept) -79.244834  1.221e+02 -0.6492 0.5204303
## MaxST         2.355252  9.350e-01  2.5189 0.0164926
## MinST        -0.137321  1.007e+00 -0.1364 0.8923023
## AvST         -0.676199  3.060e-01 -2.2096 0.0337662
## MaxAT         0.473272  5.572e-01  0.8494 0.4014051
## MinAT         0.451802  7.661e-01  0.5898 0.5591279
## AvAT          0.024874  2.150e-01  0.1157 0.9085497
## MaxH          1.562590  1.124e+00  1.3900 0.1732994
## MinH          0.866240  4.676e-01  1.8526 0.0723914
## AvH          -0.597152  1.586e-01 -3.7640 0.0006143
## Wind          0.009267  8.716e-03  1.0633 0.2949265
```

- Quite of a few of the estimated coefficients are not significant.
- Some regressors are strongly correlated with each other.
- Strong indication that we have multicollinearity among our regressors.

Review of Multicollinearity

Multicollinearity is a near linear dependency between the explanatory variables (the values of the regressors are not very well spread out).

- It affects the stability of the fitted regression plane—applies to GLMs as well as ordinary regression.
- It inflates the standard errors in the estimated coefficients
  - May find that a number of coefficients are not significant but if you remove any one of these variables the others become significant

Detecting Multicollinearity

If we have several explanatory variables, then the variance of the estimated coefficient for the jth explanatory variable $(\widehat{\beta}_j)$ is

当我们有多个解释变量时，我们需要估计每个变量的系数，以了解它们对响应变量的影响。 -第 j 个解释变量（β j）的估计系数方差是衡量与估计值相关的不确定性的重要指标。 -估计系数的方差取决于误差项（σ 2）的方差和解释变量（Var（Xj））的方差。

$$\text{Var}(\widehat{\beta}_j) = \frac{\sigma^2/(n-1)}{\text{Var}(X_j)\,(1-R_j^2)}$$

R2 值用于衡量模型与数据的拟合程度。 -当我们有多个解释变量时，我们可以将每个变量视为响应变量，然后将其与剩余的解释变量进行回归，从而计算出每个变量的 R2 值。 -第 j 个解释变量（R2j）的 R2 值表示剩余的回归变量可以在多大程度上解释 Xj 中的变异性。 -R2j 的值必须介于 0 和 1 之间。

where R_j^2 is the R^2 value when X_j is treated as the response and regressed against the remaining explanatory variables.

- R_j^2 indicates how much of the variability in X_j can be explained by the remaining regressors and must be between 0 and 1.
- If X_j is orthogonal to all of the remaining explanatory variables then $R_j^2 = 0$ and

当解释变量与所有剩余解释变量正交时，这意味着它们之间没有相关性。 -在本例中，第 j 个解释变量（R2j）的 R2 值为 0。 -当 R2j 为 0 时，估计系数方差的公式简化为

$$\text{Var}(\widehat{\beta}_j) = \frac{\sigma^2/(n-1)}{\text{Var}(X_j)}.$$

Variance Inflation Factors (VIFs)

“膨胀” 一词用于方差膨胀因子 (VIF)，它量化了由于多重共线性而增加的方差

The jth variance inflation factor is defined as

$$\mathrm{VIF}_j = \frac{1}{1 - R_j^2}.$$

Thus we have

$$\mathrm{Var}(\widehat{\beta}_j) = \mathrm{VIF}_j \times \frac{\sigma^2/(n-1)}{\mathrm{Var}(X_j)}.$$

Therefore VIF_j represents the amount that $\mathrm{Var}(\widehat{\beta}_j)$ is inflated due to the correlation between X_j and the remaining regressors.

Calculating VIFs

The VIFs are equal to the diagonal elements of the inverse of the correlation matrix for the explanatory variables. For our evaporation data:

```
round(diag(solve(cor(evap.df[, 1:10]))), 2)

## MaxST MinST  AvST MaxAT MinAT  AvAT  MaxH  MinH   AvH  Wind
## 36.23 11.81 42.17  8.90  8.81 22.71  2.25 24.65 24.19  1.90
```

判断一个变量是否存在共线性的经验法则

- a values of 1 would indicate a regressor that is completely unaffected by multicollinearity.
- a value of 5 or more indicates appreciable multicollinearity.
- a value of 10 or more indicates serious multicollinearity.

We have evidence of serious multicollinearity between the regressors. This is a strong indication that the full model contains unnecessary regressors.

Some Important Questions

1. How do we evaluate how well this model predicts the evaporation rate for new observations?

2. Can we improve this model?

   - We have compelling evidence that not all regressors are needed in the model—should we drop some variables?

   - How do we search for a good predictive model?

Mean Square Prediction Error

First, consider how well this model predicts the evaporation rate for new observations. For continuous responses the most common measure of the precision of predictions is the *mean square prediction error* which is defined as

$$\mathrm{MSPE} = \mathrm{E}\left(Y - \widehat{Y}\right)^2. \tag{1}$$

均方预测误差是反映估计量与被估计量之间差异程度的一种度量。

- Also called the mean square error for predictions.
- The expectation applies to the set of possible predictions (i.e., the target population).
- Note: if $\widehat{Y}$ is replaced by $\mathrm{E}(Y)$ then we have the definition of the variance of Y.

Estimating the Mean Square Prediction Error

Ideally, to estimate the MSPE:

- Use the current data to fit the predictive model.
- Then take a random sample of new observations from the target population.
- Find the prediction error for each new observation (the difference between the predicted value and the observed value).
- The estimate of MSPE is the mean of the squared prediction errors.

A Poor Idea

It is tempting to consider the residuals for our fitted model as prediction errors and use these to get an estimate of MSPE (use the mean of the squared residuals as our estimate).

用训练数据计算MSPE不够客观

- This almost always gives an estimate of MSPE which is too small (an *optimistic* estimate). We want an *honest* estimate.
- Models usually fit the data that was used to create the model better than they fit new observations.

So we really should evaluate our model using data that wasn't used in the model building process.

A Better Idea

Often it is not feasible to collect new data to evaluate our fitted model. However if we have plenty of data, we can randomly divide the data into two parts.

- The *training set* which is used for model building and fitting.
- The *test set* which is used to estimate the prediction error.

The MSPE is estimated by

$$\widehat{\text{MSPE}} = \frac{1}{n_{\text{test}}} \sum_{y_i \,\in\, \text{test set}} (y_i - \widehat{y}_i)^2, \tag{2}$$

where $\widehat{y}_i$ is the predicted value based on the fitted model obtained from the training set and n_{test} is the number of observations in the test set.

A Clever Idea: Cross-Validation

10折交叉验证

If we don't have plenty of data (e.g., the evaporation data set only has 46 observations) then we can use a technique called *cross-validation*.

在交叉验证中，模型不是在全部数据上训练的，而是将数据拆分为 K 个子集（在这种情况下，K=10），每次训练模型时，都会使用 K-1 个子集进行训练，然后在剩下的一个子集上进行测试。这个过程重复 B 次（在这种情况下，B=1，表示不重复），以计算模型的性能指标。

- Randomly split the data into 10 parts, say. Ideally, the parts are of equal size. One part acts as a test set, the rest as the training set. We compute the prediction error from the test set as before.
- Repeat another 9 times, using a different part as the test set each time. Average the estimates to get a good MSPE estimate.
- Repeat for different "random splits" (argument `B` later).
- This is called 10-fold cross-validation. Can do 3-fold, 5-fold, or *n*-fold, but 10-fold seems to be very popular.

The **crossval** Package

Cross-validation can be done using the **crossval** package in **R**.

- First we need to create a function that specifies the model we want to fit using the training data and the statistic we want to evaluate using the test data.

```
predfun.lm <- function(train.x, train.y, test.x, test.y) {
  lm.fit <- lm(train.y ~ ., data = train.x)
  ynew <- predict(lm.fit, test.x)
  mean((ynew - test.y)^2)
}
```

- `train.x` and `test.x` only contain the regressors and `train.y` and `test.y` only contain the response.
- This function is using the training set data to fit the model and the test set data to evaluate the model.
- The inputs for our function must be in the order given above.

Using the `crossval` Function

Then we use the `crossval` function to perform the data splitting:

```
cv.out = crossval(predfun.lm, X = evap.df[, 1:10], Y = evap.df[, 11],
                  K = 10, B = 1, verbose = FALSE)
```

In addition to the function we wish to evaluate (`predfun.lm`), the inputs are:

`X`	is the matrix of the regressors.
`Y`	is the response vector.
`K`	is the number of folds used.
`B`	is the number of times the process is repeated.

Output from the crossval Function

We get the following output

```
cv.out


## $stat.cv
##           [,1]
## B1.F1    19.60
## B1.F2   141.28
## B1.F3    15.76
## B1.F4    52.08
## B1.F5    26.01
## B1.F6   117.40
## B1.F7    18.50
## B1.F8    73.18
## B1.F9    87.66
## B1.F10   82.69
##
## $stat
## [1] 63.42
##
## $stat.se
## [1] 14.05
```

Increasing the Number of Random Splits

To a get a more precise estimate of MSPE, we can increase the number of repetitions—let's try 100.

```
cv2.out = crossval(predfun.lm, X = evap.df[, 1:10], K = 10,
                   Y = evap.df[, 11], B = 100, verbose = FALSE)
c(cv2.out$stat, cv2.out$stat.se)

## [1] 65.096  1.567
```

So our estimate is 65.1 with a standard error of 1.6.

- Caution!!! The standard error refers to the *resampling* distribution, not the sampling distribution.
- Essentially we are generating an estimate of an estimate—the true cross-validation estimate of MSPE would be obtained by using all possible splits of the data. However that is too computationally expensive so we settle for using a random sample of the splits.
- Confused? Good, that means you're awake.

Interpreting MSPE

Our CV estimate of MSPE is 65.1—what does this mean?

- MSPE is about squared errors. To get back to the relevant scale take the square root of MSPE:

  $$\sqrt{\text{MSPE}} = \sqrt{65.1} = 8.1$$

  which gives an indication of the "typical" error.

- We used MSPE because it is the most common measure of prediction error but we could have used other measures.
  - E.g., Mean absolute prediction error or median absolute prediction error.

- Our CV procedure is apt to slightly over-estimate the MSPE.
  - For CV we only use 90% of data to fit the model each time.
  - To use this model to predict future observations, we will use all of the data to fit the model and thus get more precise estimates of the coefficients.

Model Selection

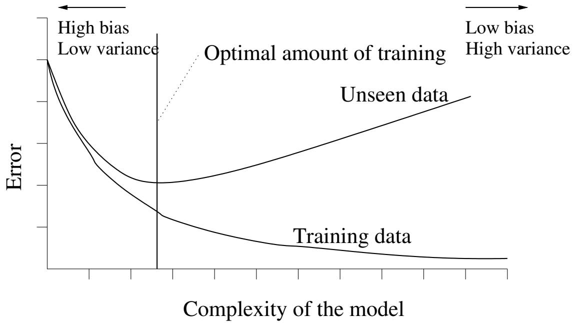
Now let's consider the question of whether we should use the full model or a subset model.

- Recall that for our fitted model many of the estimated coefficients were not significant and that many of the regressors were correlated with each other.
- It is tempting to think that the full model should be used because:
  - This ensures that all useful regressors are included.
  - Unnecessary regressors will have estimated coefficients near zero and thus have minimal impact on our predictions.

  This is WRONG!!!! **无关变量应去掉，引入噪声**
- Adding unnecessary regressors inflates the size of prediction errors—in effect they are modeling noise rather than signal.

Model Complexity versus Prediction Error

The following figure shows the general relationship between the complexity of a model and the prediction error.



- To minimize prediction error, model complexity must be balanced against how well the model fits the current data.

Over-Fitting

Over-fitting means that there are too many variables in the model—unneeded regressors have been included.

Consequences of over-fitting are:

- Fitted model is not good for prediction of new data—prediction error is underestimated.
- Model is too elaborate, models "noise" that will not be the same for new data.
- Variances of regression coefficients inflated (multicollinearity).

Under-Fitting

Under-fitting means that there are too few variables in the model—variables that could help explain the response have been left out.

Consequences of under-fitting are:

- Fitted model is not good for prediction—prediction is biased.
- Regression coefficients are biased (confounding).
- Estimate of error variance is inflated.

Back to the Evaporation Example

Data was recorded for factors affecting the evaporation water from a location in Texas, USA.

- Ten variables were measured each day over a period of 46 days.
- Our goal is to create a model that predicts the daily amount of evaporation based on the meteorological conditions that day.
- It is almost certainly not a good idea to use the model that contains all 10 variables since we have compelling evidence that this would result in over-fitting.
- The total number of subsets for our 10 regressors is $2^{10} = 1024$.
- So which of these is best?

Subset Selection I

Ideally, we want to choose the subset model that results in the smallest MSPE.

- It simply isn't feasible to use our CV approach to estimate the MSPE for each of the 1024 subset models. Instead we will first use AICc and then use BIC to screen the models since calculating AICc or BIC is computationally much less expensive than CV. Now

$$\begin{aligned} \text{AICc} &= -2\ell + \left(2k + \frac{2k^2 + 2k}{n - k - 1}\right), \\ \text{BIC} &= -2\ell + k \log n. \end{aligned}$$

  Recall that the -2ℓ term measures how well the model fits the data (smaller values indicate better fit) and the second term is a penalty for model complexity.

Subset Selection II

- Both AICc and BIC are balancing model fit with model complexity which is also what we need to do to find a good predictive model.
- However the penalties used for AICc or for BIC may not be precisely those needed to identify the model with the lowest MSPE.
- Therefore we will use AICc and BIC to produce a "short list" of models and then choose between these by estimating their MSPE using cross validation.

Subset Selection using AICc

```
library('MuMIn')
options(na.action = "na.fail", width=120)
evap.fits <- dredge(evap.lm)
print(round(evap.fits[1:10, ], 2))
```

```
##     (Intercept) AvAT   AvH  AvST MaxAT MaxH MaxST MinAT MinH MinST Wind df logLik  AICc delta weight
## 168       72.13 0.30 -0.47 -0.68    NA   NA  2.08    NA 0.59    NA   NA  7 -146.3 309.6  0.00   0.16
## 40        61.78 0.42 -0.31 -0.56    NA   NA  1.28    NA   NA    NA   NA  6 -147.8 309.8  0.22   0.15
## 247     -104.76   NA -0.64 -0.78    NA 1.77  2.87  0.90 0.96    NA   NA  8 -145.1 310.0  0.44   0.13
## 552       72.83 0.36 -0.33 -0.49    NA   NA  1.19    NA   NA    NA 0.01  7 -146.6 310.1  0.47   0.13
## 695      -77.90   NA -0.65 -0.60    NA 1.92  2.70    NA 1.13    NA 0.01  8 -145.4 310.7  1.11   0.09
## 183      -77.49   NA -0.70 -0.71    NA 1.81  3.13    NA 1.35    NA   NA  7 -147.0 310.9  1.28   0.09
## 680       78.25 0.28 -0.45 -0.60    NA   NA  1.85    NA 0.47    NA 0.01  8 -145.7 311.2  1.59   0.07
## 184      -10.97 0.21 -0.56 -0.71    NA 0.99  2.43    NA 0.85    NA   NA  8 -145.7 311.4  1.76   0.07
## 104       45.62 0.33 -0.34 -0.62    NA   NA  1.47  0.56   NA    NA   NA  7 -147.3 311.6  2.02   0.06
## 520      146.89 0.36 -0.38 -0.20    NA   NA    NA    NA   NA    NA 0.01  6 -148.8 311.8  2.19   0.05
```

- Note that in this case there are many models that have similar values of AICc (if the difference is less than ≈ 2 we really can't say one model is better than the other).

Subset Selection using BIC

```r
options(na.action = "na.fail", width=120)
evap2.fits <- dredge(evap.lm, rank="BIC")
print(round(evap2.fits[1:10, ], 2))
```

```
##     (Intercept) AvAT   AvH  AvST MaxAT MaxH MaxST MinAT MinH MinST Wind df logLik   BIC delta weight
## 40        61.78 0.42 -0.31 -0.56    NA   NA  1.28    NA   NA    NA   NA  6 -147.8 318.6  0.00   0.20
## 168       72.13 0.30 -0.47 -0.68    NA   NA  2.08    NA 0.59    NA   NA  7 -146.3 319.4  0.82   0.13
## 516      126.02 0.22 -0.35    NA    NA   NA    NA    NA   NA    NA 0.02  5 -150.3 319.8  1.13   0.11
## 8        141.07 0.44 -0.37 -0.26    NA   NA    NA    NA   NA    NA   NA  5 -150.3 319.8  1.20   0.11
## 552       72.83 0.36 -0.33 -0.49    NA   NA  1.19    NA   NA    NA 0.01  7 -146.6 319.9  1.29   0.11
## 520      146.89 0.36 -0.38 -0.20    NA   NA    NA    NA   NA    NA 0.01  6 -148.8 320.6  1.97   0.07
## 183      -77.49   NA -0.70 -0.71    NA 1.81  3.13    NA 1.35    NA   NA  7 -147.0 320.7  2.10   0.07
## 247     -104.76   NA -0.64 -0.78    NA 1.77  2.87   0.9 0.96    NA   NA  8 -145.1 320.8  2.14   0.07
## 772      165.79 0.33 -0.35    NA    NA   NA    NA    NA   NA -0.84 0.01  6 -148.9 320.9  2.23   0.07
## 4        108.91 0.26 -0.31    NA    NA   NA    NA    NA   NA    NA   NA  4 -152.9 321.1  2.44   0.06
```

- BIC tends to identify models with less terms than AICc—it penalises model complexity more heavily.
- There are many models that have similar values of BIC.

Diagnostics

We should apply the usual diagnostic procedures to make sure these models conform to the regression model assumptions and correct any flaws before evaluating their MSPEs.
For this case, we have the situation where the best models all have many regressors in common.

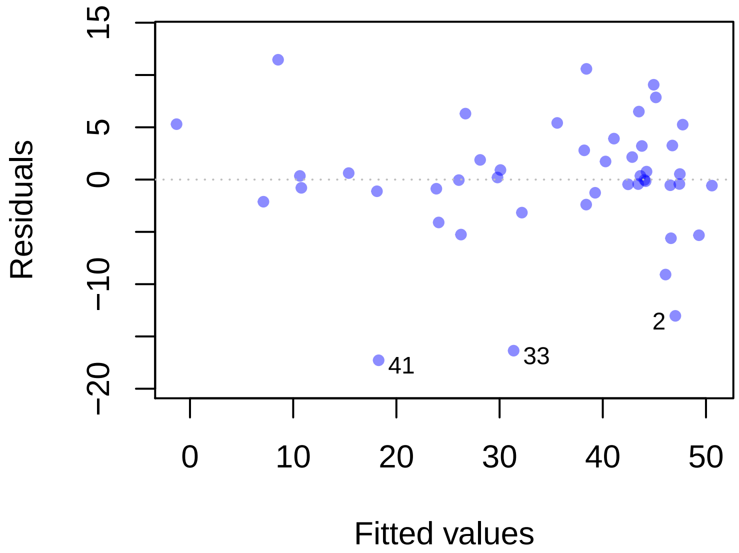
- The first model on the AICc list contains AvAT, AvH, AvST, MaxST and MinH as regressors.
- Doing diagnostics on this model should give us a good indication of whether there are any problems in general with these models.

The Top Model

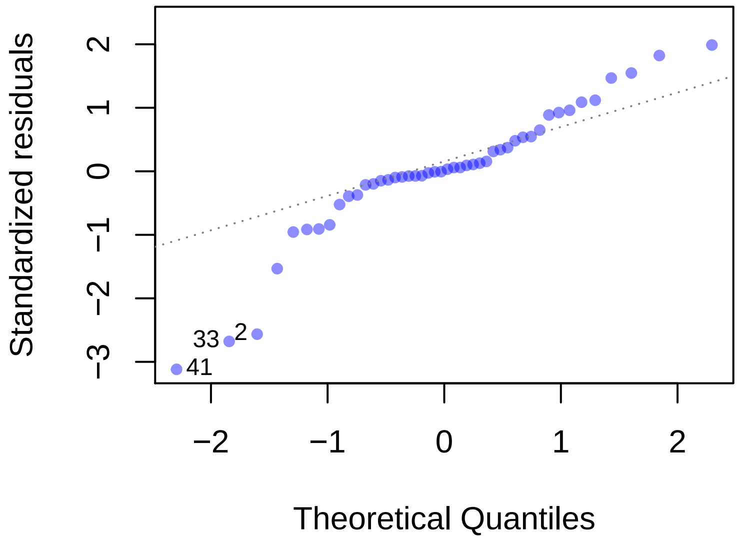
To extract the best model from our list:

```
model1.lm = get.models(evap.fits, 1)[[1]]
summary(model1.lm)


##
## Call:
## lm(formula = Evap ~ AvAT + AvH + AvST + MaxST + MinH + 1, data = evap.df)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -17.280  -1.226   0.081   3.109  11.459
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   72.131     44.557    1.62  0.11334
## AvAT           0.297      0.120    2.48  0.01759 *
## AvH           -0.471      0.110   -4.27  0.00012 ***
## AvST          -0.680      0.189   -3.59  0.00090 ***
## MaxST          2.083      0.755    2.76  0.00871 **
## MinH           0.587      0.357    1.65  0.10774
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.25 on 40 degrees of freedom
## Multiple R-squared:  0.838,Adjusted R-squared:  0.818
## F-statistic: 41.4 on 5 and 40 DF,  p-value: 8.72e-15
```

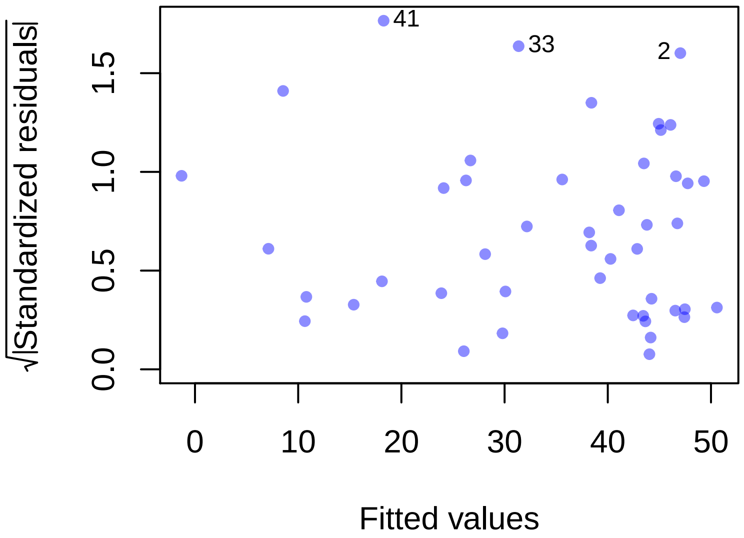
Residuals vs Fitted Values



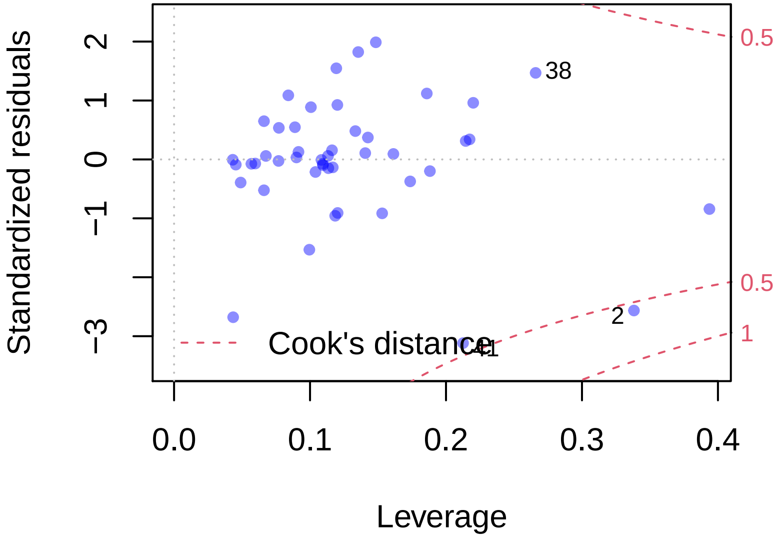
Normal Q-Q Plot



Scale Location Plot



Residuals vs Leverage



Which Model?

Our diagnostics didn't reveal any issues with our top model apart from an indication of some non-normality. This is a minor issue and we prefer not to transform the response unless it clearly improves our model (further investigation indicated this is not the case). Also observation 2 has a moderately high value of Cook's Distance and could be investigated.

We will take the top 5 models from each list to form our short list. This gives us a total of 7 models since 3 models occur in the top 5 for both lists.

We will use our CV procedure to estimate the MSPE for these models.

Some **R** Code I

First create a new **R** function that estimates MSPE for each of these 7 models.

```
predfun.lm <- function(train.x, train.y, test.x, test.y) {
lm1.fit <- lm(train.y ~ AvAT+AvH+AvST+MaxST+MinH, data=train.x)
 ynew <- predict(lm1.fit, test.x)
  out1 <- mean((ynew - test.y)^2)

lm2.fit <- lm(train.y ~ AvAT+AvH+AvST+MaxST, data=train.x)
 ynew <- predict(lm2.fit, test.x)
  out2 <- mean((ynew - test.y)^2)

lm3.fit <- lm(train.y ~ AvH+AvST+MaxH+MaxST+MinAT+MinH, data=train.x)
 ynew <- predict(lm3.fit, test.x)
  out3 <- mean((ynew - test.y)^2)

lm4.fit <- lm(train.y ~AvAT+AvH+AvST+MaxST +Wind, data=train.x)
 ynew <- predict(lm4.fit, test.x)
  out4 <- mean((ynew - test.y)^2)
```

Some **R** Code II

```
lm5.fit <- lm(train.y ~AvH+AvST+MaxH+MinH+MaxST+Wind, data=train.x)
 ynew <- predict(lm5.fit, test.x)
  out5 <- mean((ynew - test.y)^2)

lm6.fit <- lm(train.y ~ AvAT+AvH+Wind, data=train.x)
 ynew <- predict(lm6.fit, test.x)
  out6 <- mean((ynew - test.y)^2)

lm7.fit <- lm(train.y ~ AvAT+AvH+AvST, data=train.x)
 ynew <- predict(lm7.fit, test.x)
  out7 <- mean((ynew - test.y)^2)

c(out1, out2, out3, out4, out5, out6, out7)
}
```

Cross Validation

Now we use the `crossval` function to get estimates of the MSPEs.

```
cv.out = crossval(predfun.lm, X = evap.df[, 1:10], Y = evap.df[, 11],
                  K = 10, B = 500, verbose = FALSE)
round(cv.out$stat, 2)

## [1] 51.08 51.36 51.23 51.72 51.80 50.02 52.97

round(cv.out$stat.se, 2)

## [1] 0.65 0.67 0.64 0.69 0.63 0.66 0.59
```

- The estimated values of MSPE for all of these models are quite similar which is not surprising given the severe amount of collinearity among the regressors.
- These values are much lower than the estimated MSPE for the full model (65.1).

And the Winner is . . .

Model 6 from our list has the smallest estimated MSPE (50.02).

```
model6.lm = get.models(evap2.fits, 3)[[1]]
summary(model6.lm)


##
## Call:
## lm(formula = Evap ~ AvAT + AvH + Wind + 1, data = evap.df)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -20.82  -2.08   0.10   3.23  13.89
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 126.02471   24.64490    5.11  7.4e-06 ***
## AvAT          0.21952    0.05909    3.72  0.00059 ***
## AvH          -0.34667    0.04283   -8.09  4.1e-10 ***
## Wind          0.01597    0.00717    2.23  0.03141 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 6.65 on 42 degrees of freedom
## Multiple R-squared:  0.808,Adjusted R-squared:  0.794
## F-statistic: 58.7 on 3 and 42 DF,  p-value: 4.46e-15
```

- The model that contains AvAT, AvH and Wind has the smallest estimated MSPE but there are other models that have values almost as small.

A Word of Caution

There are a number of models that perform about as well as each other with respect to prediction (this happens quite often).

- Having chosen a model on the basis that it has the smallest estimated MSPE from a set of models with similar performance means that the MSPE has probably been underestimated.
- Suppose we have a set of models which all have the same true MSPE. By selecting the model with the smallest estimated MSPE, the chosen model is usually one for which the MSPE has been underestimated.
- Situation is similar for a set of models which all have true values of MSPE that are approximately the same.
- To get an unbiased estimate of MSPE, we need to use data that has not been involved in the model selection process.

Important Lessons from the Evaporation Example I

The evaporation data was used to highlight a number of important aspects of selecting a model for prediction.

1. We should be extremely cautious about using a model to predict for observations that are not consistent with those used to build the model.

   - In this example the data was collected at a single site in Texas, USA, over 46 consecutive days (6 June through 21 July). This restricts the scope of future observations for which this model is valid.

Important Lessons from the Evaporation Example II

2. Getting a realistic estimate of MSPE is tricky.

   - Ideally we would use different data (i) to select and fit the model and (ii) to estimate the MSPE but this requires lots of data.
   - Cross-validation can help us get a more realistic estimation of prediction error where data is limited but will slightly over estimate the true MSPE (due to only using part of the data for fitting the model).

Important Lessons from the Evaporation Example III

3. Model selection is important. This requires balancing model complexity with how well the model fits the data.

    - Having unnecessary regressors in the model (over-fitting) will inflate the MSPE.
    - Having too few regressors (under-fitting) also inflates the MSPE.
    - Information based criteria such as AICc or BIC can be used with search functions to create a short list. But it is better to base the final decision on estimates of MSPE.
    - Selecting a predictive model from a set of models that all are similar in terms of their true MSPE leads to an optimistic estimate of MSPE for the selected model.
    - Models based on sensible regressors are more apt to be reliable for future observations.

15.4 Using GLMs for Prediction

Using a GLM for prediction is similar in many ways to using a normal regression model.

- The model should only be used for future predictions of observations that are consistent with the data used to build the model.
- Model building and selection techniques are important in identifying a suitable model for prediction.
- Models based on sensible regressors are more apt to be reliable for future observations.
- To get a realistic assessment of prediction error, it is necessary to use observations not included in the model building process.

15.4.1 Age of Abalone Example

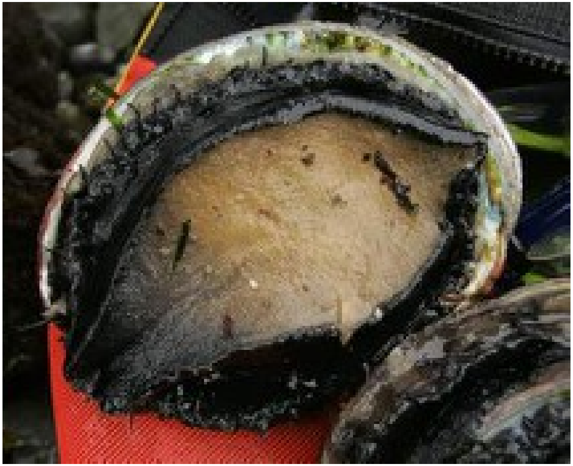
A data set consisting of physical measurements made on abalone is available from the UCI Machine Learning Repository.
The purpose of modelling is described as follows:

> *The age of abalone is determined by cutting the shell through the cone, staining it, and counting the number of rings through a microscope—a boring and time-consuming task. Other measurements, which are easier to obtain, are used to predict the age.*

- Data comes from Warwick J Nash, Tracy L Sellers, Simon R Talbot, Andrew J Cawthorn and Wes B Ford (1994). "The Population Biology of Abalone (Haliotis species) in Tasmania. I. Blacklip Abalone (H. rubra) from the North Coast and Islands of Bass Strait", Sea Fisheries Division, Technical Report No. 48 (ISSN 1034-3288)
- Data can be downloaded from the *UCI Machine Learning Repository*:

  https://archive.ics.uci.edu/ml/datasets/Abalone

Blacklip Abalone



Abalone Data Variables

Response

rings	:	number of rings (+1.5 gives the age in years).

Regressors

sex	:	factor with levels M, F, and I (infant)
length	:	longest shell measurement (mm).
diameter	:	perpendicular to length (mm).
height	:	measured with meat in shell (mm).
wwt	:	weight of whole abalone (g).
mwt	:	weight of meat (g).
gwt	:	weight of gut after bleeding (g).
swt	:	weight of dried shell (g).

Values of the continuous variables were scaled by dividing by 200.

Oodles of Data

The abalone data set contains 4177 observations.

- Measurements were made on blacklip abalone collected from the North Coast and Islands of Bass Strait (Tasmania, Aus.).
- We have lots of data which is good—we can divide the data into different sets for model building and fitting and for model testing.
- However having lots of data makes data exploration and cleaning more challenging.

What Lies Ahead

Our goal is to find a predictive model for the number of rings. Our course of action will consist of the following steps:

1. Data exploration and cleaning.
2. Model selection—use the `dredge()` function to identify a short list of promising models based on AICc and BIC.
3. Apply model diagnostics and model building procedures to see if these models can be improved.
4. Identify the best predictive model(s) by estimating the MSPE.
5. Summarise what we learn.

Data Inspection I

It's always a good idea to take a look at the data using the `head`, `str` and `summary` commands.

```
abalone.df <- read.table("../data/abalone.data", sep = ",")
attributes(abalone.df)$names <- c("sex", "length", "diameter", "height",
                                  "wwt", "mwt", "gwt", "swt", "rings")
abalone.df$sex = factor(abalone.df$sex, levels=c("F", "M", "I"))
head(abalone.df)


##   sex length diameter height    wwt    mwt    gwt   swt rings
## 1   M  0.455    0.365  0.095 0.5140 0.2245 0.1010 0.150    15
## 2   M  0.350    0.265  0.090 0.2255 0.0995 0.0485 0.070     7
## 3   F  0.530    0.420  0.135 0.6770 0.2565 0.1415 0.210     9
## 4   M  0.440    0.365  0.125 0.5160 0.2155 0.1140 0.155    10
## 5   I  0.330    0.255  0.080 0.2050 0.0895 0.0395 0.055     7
## 6   I  0.425    0.300  0.095 0.3515 0.1410 0.0775 0.120     8
```

Data Inspection II

```
str(abalone.df)
```

```
## 'data.frame': 4177 obs. of  9 variables:
##  $ sex     : Factor w/ 3 levels "F","M","I": 2 2 1 2 3 3 1 1 2 1 ...
##  $ length  : num  0.455 0.35 0.53 0.44 0.33 0.425 0.53 0.545 0.475 0.55 ...
##  $ diameter: num  0.365 0.265 0.42 0.365 0.255 0.3 0.415 0.425 0.37 0.44 ...
##  $ height  : num  0.095 0.09 0.135 0.125 0.08 0.095 0.15 0.125 0.125 0.15 ...
##  $ wwt     : num  0.514 0.226 0.677 0.516 0.205 ...
##  $ mwt     : num  0.2245 0.0995 0.2565 0.2155 0.0895 ...
##  $ gwt     : num  0.101 0.0485 0.1415 0.114 0.0395 ...
##  $ swt     : num  0.15 0.07 0.21 0.155 0.055 0.12 0.33 0.26 0.165 0.32 ...
##  $ rings   : int  15 7 9 10 7 8 20 16 9 19 ...
```

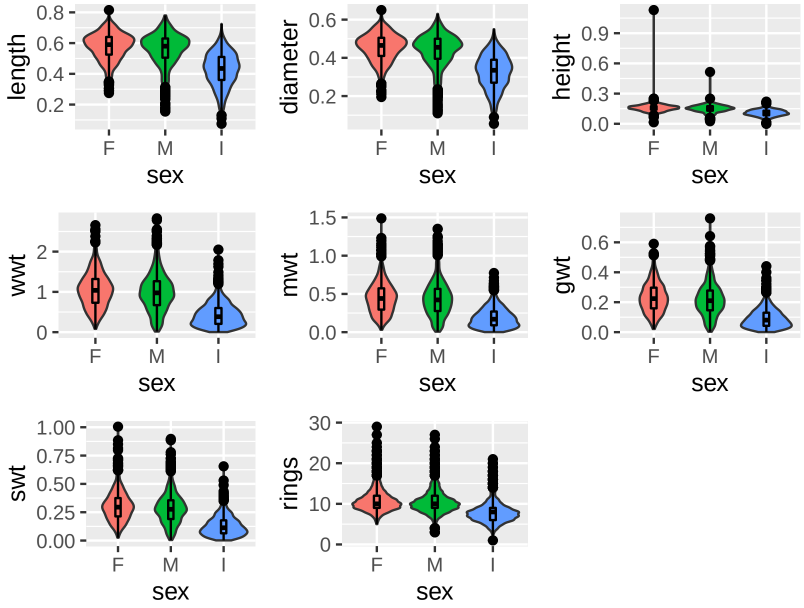
Data Inspection III

```
options(width=75)
summary(abalone.df)
```

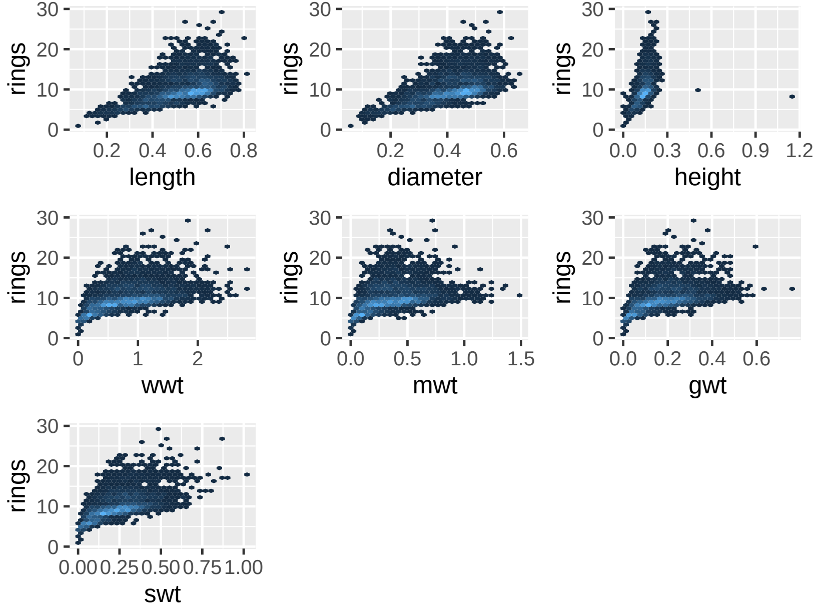
```
##  sex          length          diameter          height            wwt
##  F:1307   Min.   :0.075   Min.   :0.055   Min.   :0.000   Min.   :0.002
##  M:1528   1st Qu.:0.450   1st Qu.:0.350   1st Qu.:0.115   1st Qu.:0.442
##  I:1342   Median :0.545   Median :0.425   Median :0.140   Median :0.799
##           Mean   :0.524   Mean   :0.408   Mean   :0.140   Mean   :0.829
##           3rd Qu.:0.615   3rd Qu.:0.480   3rd Qu.:0.165   3rd Qu.:1.153
##           Max.   :0.815   Max.   :0.650   Max.   :1.130   Max.   :2.825
##       mwt              gwt               swt              rings
##  Min.   :0.001   Min.   :0.0005   Min.   :0.0015   Min.   : 1.00
##  1st Qu.:0.186   1st Qu.:0.0935   1st Qu.:0.1300   1st Qu.: 8.00
##  Median :0.336   Median :0.1710   Median :0.2340   Median : 9.00
##  Mean   :0.359   Mean   :0.1806   Mean   :0.2388   Mean   : 9.93
##  3rd Qu.:0.502   3rd Qu.:0.2530   3rd Qu.:0.3290   3rd Qu.:11.00
##  Max.   :1.488   Max.   :0.7600   Max.   :1.0050   Max.   :29.00
```

See any problems (look at `height`)?

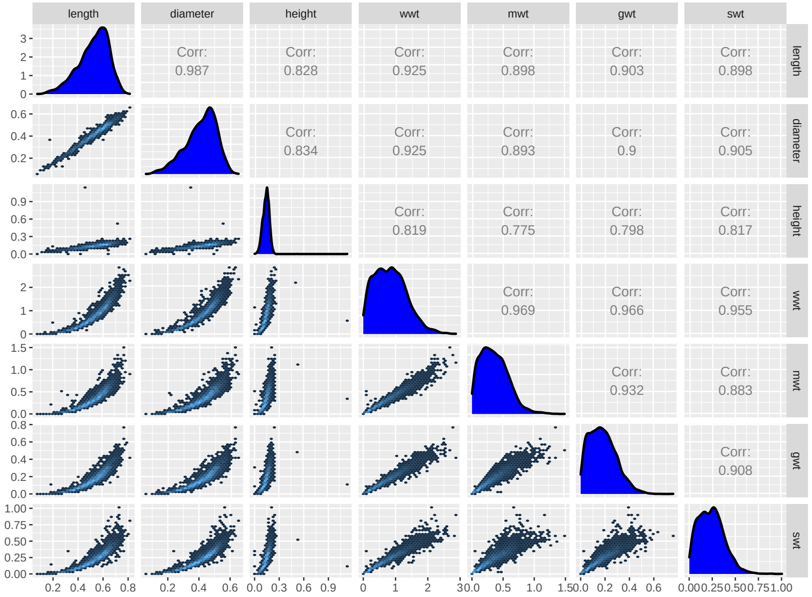
Violin Plots of the Numeric Variables Conditioned on `sex`



Hexbin Plots of the Numeric Regressors vs the Response



Pairs Plot of the Numeric Regressors

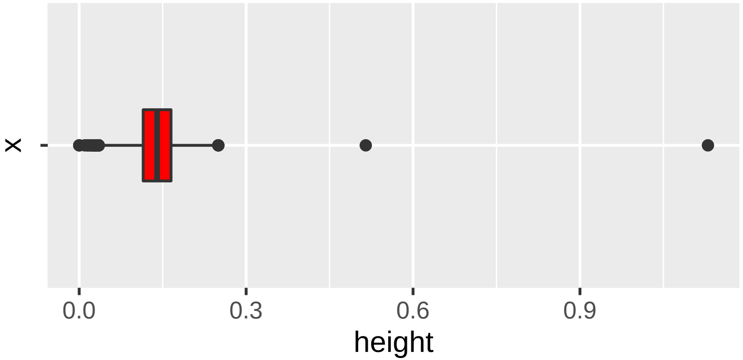


Initial Assessment of Data

Some areas of concern can be detected from the `summary` output and the various plot:

- Two unusually large values for `height`.
- Minimum value of `height` is 0.
- One unusual combination of `length` and `diameter`.

Box Plot of `height`



- Looks like the two large values are errors:

```
subset(abalone.df, height > 0.4)

##      sex length diameter height   wwt   mwt    gwt    swt rings
## 1418   M  0.705    0.565  0.515 2.210 1.107 0.4865 0.5120    10
## 2052   F  0.455    0.355  1.130 0.594 0.332 0.1160 0.1335     8
```

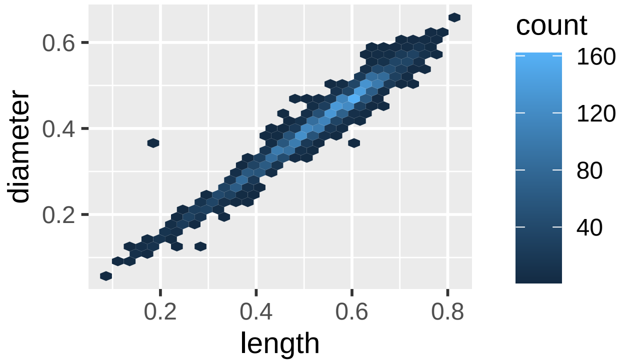
Small Values of height

There are several unusually small values of height.

```
subset(abalone.df, height < 0.02)

##      sex length diameter height    wwt    mwt    gwt    swt rings
## 237    I  0.075    0.055  0.010 0.0020 0.0010 0.0005 0.0015     1
## 1175   F  0.635    0.495  0.015 1.1565 0.5115 0.3080 0.2885     9
## 1258   I  0.430    0.340  0.000 0.4280 0.2065 0.0860 0.1150     8
## 2170   I  0.165    0.115  0.015 0.0145 0.0055 0.0030 0.0050     4
## 3997   I  0.315    0.230  0.000 0.1340 0.0575 0.0285 0.3505     6
```

- Observations 237 and 2170 seem plausible.
- Observations 1258 and 3997 are clearly mistakes (height = 0).
- Observation 1175 is very suspicious (height is very small whereas other measurements are large).

length vs. diameter Plot



- One point has `length` < `diameter` which should not happen given the way the measurements were defined.

```
subset(abalone.df, length < diameter)

##      sex length diameter height    wwt   mwt    gwt  swt rings
## 1211   I  0.185    0.375   0.12 0.4645 0.196 0.1045 0.15     6
```

Delete Unusual Observations

Given the large number of observations, we will just delete the suspect observations.

```
new.df = abalone.df[-c(1175, 1211, 1258, 1418, 2052, 3997), ]
row.names(new.df) = 1:nrow(new.df)
```

- It's a good idea to recreate the rownames—otherwise the deleted observation numbers are missing which can lead to confusion.

Divide Data into Two Parts

At this point, we have completed the data exploration and cleaning phase of our analysis. We are now ready to start looking for a predictive model. Given we are "data rich" we will split our data into two parts.

- These will be used for model building and model evaluation.
- Note that more observations were put into the model building (selection and estimation) part—reflects the relative importance of finding a good model versus getting an accurate estimate of prediction error.

The data can be randomly divided into two parts as follows:

```
set.seed(145)
sam = sample(1:nrow(new.df))  # i.e., sample(1:4171)
abalone1.df = new.df[sam[1:3500], ]
row.names(abalone1.df) = 1:nrow(abalone1.df)
abalone2.df = new.df[sam[3501:nrow(new.df)], ]
row.names(abalone2.df) = 1:nrow(abalone2.df)
```

The Full Model

```
##
## Call:
## glm(formula = rings ~ sex + length + diameter + height + wwt +
##     mwt + gwt + swt, family = poisson, data = abalone1.df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.008  -0.445  -0.116   0.284   3.244
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  1.43330    0.05226   27.42  < 2e-16
## sexM         0.00598    0.01251    0.48   0.6326
## sexI        -0.09799    0.01667   -5.88  4.2e-09
## length       0.16643    0.28757    0.58   0.5628
## diameter     1.27532    0.35298    3.61   0.0003
## height       2.65175    0.35720    7.42  1.1e-13
## wwt          0.73750    0.10171    7.25  4.1e-13
## mwt         -1.81037    0.11900  -15.21  < 2e-16
## gwt         -0.93426    0.18657   -5.01  5.5e-07
## swt          0.39870    0.15764    2.53   0.0114
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 3579.3  on 3499  degrees of freedom
## Residual deviance: 1519.6  on 3490  degrees of freedom
## AIC: 15903
##
## Number of Fisher Scoring iterations: 4
```

Suspect underdispersion since residual deviance $\ll$ df's

The Quasi-Poisson Model

```
##
## Call:
## glm(formula = rings ~ sex + length + diameter + height + wwt +
##     mwt + gwt + swt, family = quasipoisson, data = abalone1.df)
##
## Deviance Residuals:
##    Min       1Q   Median       3Q      Max
## -2.008   -0.445   -0.116    0.284    3.244
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)  1.43330    0.03541   40.48  < 2e-16
## sexM         0.00598    0.00848    0.71  0.48041
## sexI        -0.09799    0.01130   -8.67  < 2e-16
## length       0.16643    0.19482    0.85  0.39301
## diameter     1.27532    0.23913    5.33  1.0e-07
## height       2.65175    0.24200   10.96  < 2e-16
## wwt          0.73750    0.06890   10.70  < 2e-16
## mwt         -1.81037    0.08062  -22.45  < 2e-16
## gwt         -0.93426    0.12639   -7.39  1.8e-13
## swt          0.39870    0.10680    3.73  0.00019
##
## (Dispersion parameter for quasipoisson family taken to be 0.459)
##
##     Null deviance: 3579.3  on 3499  degrees of freedom
## Residual deviance: 1519.6  on 3490  degrees of freedom
## AIC: NA
##
## Number of Fisher Scoring iterations: 4
```

Underdispersion

We have clear indication of underdispersion which is not surprising since we don't expect the occurrence of rings to be independent (as a Poisson model would suggest).

- The only model that we have considered in this course that can compensate for underdispersion in count data is the quasi-Poisson model. Unfortunately, `dredge()` won't work on a quasi-Poisson model (we can't use the residual deviance to both estimate the scale parameter and to evaluate how well the model fits the data).

- Using the Poisson model is not ideal but it's not all that bad.
  - Get the same fitted model as for the quasi-Poisson model.
  - We evaluate a predictive model based on how well it predicts new observation. Underdispersion would be much more serious if we were using the model for explanation as it affects inference.
  - Underdispersion means we should be able to get better predictions than if the Poisson model was valid—so in that sense it is good.

Model Search using AICc

Again, we will use `dredge()` to search for a suitable predictive model.

```
abalone1.glm = glm(rings ~ sex +length + diameter + height + wwt +
                         mwt + gwt + swt, poisson, data = abalone1.df)
abalone.fits = dredge(abalone1.glm)
head(abalone.fits)


## Global model call: glm(formula = rings ~ sex + length + diameter + height + wwt +
##     mwt + gwt + swt, family = poisson, data = abalone1.df)
## ---
## Model selection table
##     (Intrc) dimtr     gwt heght  lngth    mwt sex    swt    wwt df logLik  AICc delta weight
## 248   1.444 1.453 -0.9238 2.657        -1.804   + 0.3963 0.7370  9  -7942 15902  0.00  0.626
## 256   1.433 1.275 -0.9343 2.652 0.1664 -1.810   + 0.3987 0.7375 10  -7942 15903  1.68  0.271
## 184   1.421 1.507 -1.0890 2.812        -1.986   +        0.9410  8  -7945 15906  4.31  0.072
## 192   1.412 1.351 -1.0990 2.808 0.1464 -1.993   +        0.9425  9  -7945 15908  6.06  0.030
## 255   1.439       -0.9638 2.874 1.0710 -1.807   + 0.4458 0.7435  9  -7948 15914 12.71  0.001
## 191   1.415       -1.1510 3.065 1.1090 -2.012   +        0.9750  8  -7952 15920 18.80  0.000
## Models ranked by AICc(x)
```

- The second model on this list is the full model.

Model Search using BIC

```
abalone2.fits = dredge(abalone1.glm, rank="BIC")
head(abalone2.fits)


## Global model call: glm(formula = rings ~ sex + length + diameter + height + wwt +
##     mwt + gwt + swt, family = poisson, data = abalone1.df)
## ---
## Model selection table
##     (Intrc) dimtr     gwt heght  lngth    mwt sex    swt    wwt df logLik   BIC delta weight
## 184   1.421 1.507 -1.0890 2.812        -1.986   +        0.9410  8  -7945 15955  0.00  0.701
## 248   1.444 1.453 -0.9238 2.657        -1.804   + 0.3963 0.7370  9  -7942 15957  1.84  0.279
## 192   1.412 1.351 -1.0990 2.808 0.1464 -1.993   +        0.9425  9  -7945 15963  7.90  0.013
## 256   1.433 1.275 -0.9343 2.652 0.1664 -1.810   + 0.3987 0.7375 10  -7942 15965  9.66  0.006
## 191   1.415       -1.1510 3.065 1.1090 -2.012   +        0.9750  8  -7952 15970 14.49  0.001
## 255   1.439       -0.9638 2.874 1.0710 -1.807   + 0.4458 0.7435  9  -7948 15970 14.55  0.000
## Models ranked by BIC(x)
```

- The same six models show up for BIC as for AICc but the order is different.

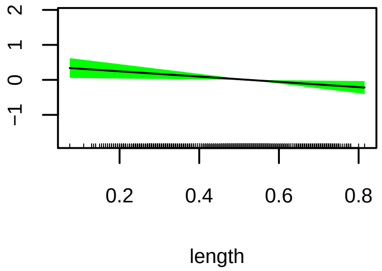
Model Checking

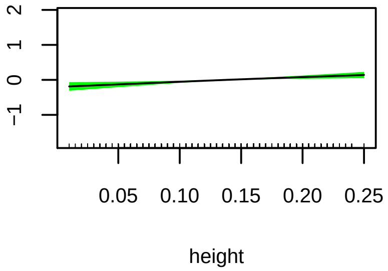
We can fit a GAM using all of the regressors to explore ways of improving our model. Using **mgcv**,

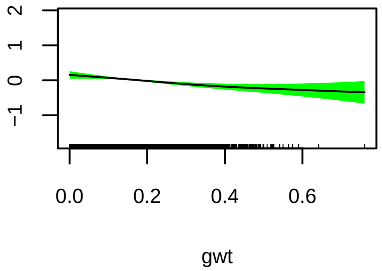
```
abaloneA.gam <- gam(rings~sex+s(diameter)+s(length)+s(height)+s(wwt)+
                    s(mwt)+s(gwt)+s(swt), poisson, data = abalone1.df)

print(summary(abaloneA.gam)$p.table, digits = 3)


##              Estimate Std. Error z value Pr(>|z|)
## (Intercept)   2.28512     0.0101 226.146 0.000000
## sexM          0.00381     0.0125   0.304 0.761062
## sexI         -0.06535     0.0173  -3.767 0.000165


print(summary(abaloneA.gam)$s.table, digits = 3)


##               edf Ref.df Chi.sq  p-value
## s(diameter) 4.64   5.73  36.52 6.09e-06
## s(length)   1.00   1.00   5.71 1.68e-02
## s(height)   1.19   1.35  12.24 1.13e-03
## s(wwt)      4.41   5.53  87.28 1.44e-16
## s(mwt)      4.35   5.49 239.91 5.43e-48
## s(gwt)      2.04   2.68  19.56 3.13e-04
## s(swt)      2.95   3.82  24.85 5.04e-05
```

GAM plots I

Plots for `length`, `height` and `gwt` are essentially linear.

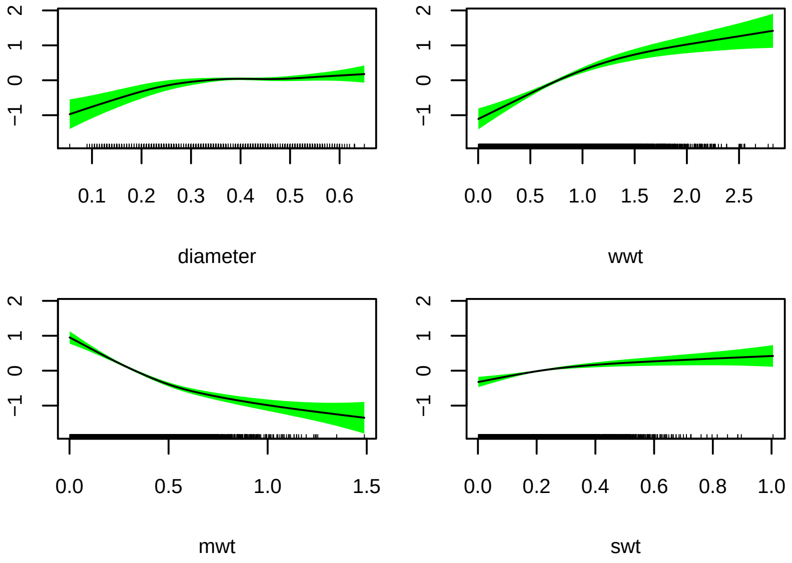






GAM plots II

Plots for `diameter`, `wwt`, `mwt` and `swt` indicate curvature.



Models with Quadratic Terms

The GAM plots suggest that `length`, `height` and `gwt` are essentially linear but it may be useful to add squared terms for `diameter`, `wwt`, `mwt` and `swt`.

```
abalone2.glm = glm(rings ~ sex + length + diameter + I(diameter^2)+ height + wwt + I(wwt^2)+
    mwt + I(mwt^2) +  gwt + swt+I(swt^2), poisson, data = abalone1.df)
abalone2.fits = dredge(abalone2.glm, rank="BIC")
head(abalone2.fits, 7)


## Global model call: glm(formula = rings ~ sex + length + diameter + I(diameter^2) +
##     height + wwt + I(wwt^2) + mwt + I(mwt^2) + gwt + swt + I(swt^2),
##     family = poisson, data = abalone1.df)
## ---
## Model selection table
##      (Intrc) dimtr dimtr^2     gwt heght   lngth    mwt  mwt^2 sex    swt   swt^2    wwt   wwt^2 df
## 3568  0.8907 5.051  -5.502 -0.8369 1.474         -3.917 1.8270   + 0.6324         1.8070 -0.3405 12
## 3584  0.8973 6.010  -5.972 -0.7798 1.404 -0.6759 -3.855 1.8030   + 0.6453         1.8330 -0.3454 13
## 4080  0.9040 5.004  -5.582 -0.8389 1.395         -3.613 1.5800   + 1.4010 -0.8385 1.5010 -0.2318 13
## 2032  0.8403 5.456  -6.147 -0.8085 1.429         -2.842 0.9443   + 2.2730 -1.7970 0.8476         12
## 3440  0.8399 4.938  -5.449 -0.8059 1.496         -4.094 1.9510     0.6293         1.9890 -0.3960 10
## 3312  0.8903 4.906  -5.111 -1.1040 1.777         -4.197 1.8210   +                2.1050 -0.3370 11
## 4096  0.9113 5.988  -6.067 -0.7803 1.320 -0.6954 -3.540 1.5470   + 1.4390 -0.8656 1.5180 -0.2333 14
##      logLik   BIC delta weight
## 3568  -7865 15827  0.00  0.693
## 3584  -7862 15831  3.54  0.118
## 4080  -7862 15831  3.96  0.096
## 2032  -7868 15833  5.94  0.036
## 3440  -7876 15834  7.06  0.020
## 3312  -7872 15834  7.14  0.019
## 4096  -7860 15834  7.24  0.019
## Models ranked by BIC(x)
```

Estimates of MSPE

So far we have used our “training data” to create a list of candidate predictive models. Now we use the “test data” to evaluate the MSPE of these models.

- The following loop in **R** evaluates the MSPE for the first 12 models on our list.

```
first12 <- 12; out = rep(0, first12)
for (i in 1:first12) {
  preds = predict(get.models(abalone2.fits, i)[[1]],
                  newdata = abalone2.df, type = "response")
  out[i] = mean((preds - abalone2.df$rings)^2)
}
round(out, 2)

##  [1] 4.14 4.13 4.12 4.13 4.20 4.15 4.11 4.17 4.16 4.18 4.15 4.12
```

- Model 7 on this list (the full model) produces the smallest estimated MSPE but is not significantly better than some of the other models.

A More Comprehensive Search I

We should be a bit concerned that our short list of the best models was produced by calculating the AICc (or BIC) assuming a Poisson distribution. Given the presence of underdispersion, the penalties used for model complexity may not be severe enough. One way to make our search more comprehensive would be to estimate the MSPE for the best models of each size over a range of sizes.

The following code will evaluate the best 7 models of each size for model df's from 5 through 14 (full model).

```
nbest = 7; mindf = 5; maxdf = 14; dfs = mindf:maxdf
out = matrix(NA, length(dfs), nbest)  # Storage
dimnames(out) = list(as.character(dfs), as.character(1:nbest))
for (i in 1:length(dfs)) {
  List = which(abalone2.fits$df == dfs[i])[1:nbest]
  for (j in 1:nbest) {
    if(!is.na(List[j])) {
      preds = predict(get.models(abalone2.fits, List[j])[[1]],
                      newdata = abalone2.df, type = "response")
      out[i, j] = mean((preds - abalone2.df$rings)^2)
}}}
```

A More Comprehensive Search II

The output indicates that the full model is giving the lowest estimated MSPE.

```
round(out, 3)
```

```
##        1     2     3     4     5     6     7
## 5  4.774 4.651 4.549 4.635 4.602 4.659 4.674
## 6  4.637 4.438 4.446 4.640 4.425 4.632 4.465
## 7  4.362 4.433 4.362 4.558 4.391 4.334 4.492
## 8  4.269 4.253 4.298 4.304 4.282 4.321 4.272
## 9  4.200 4.229 4.231 4.235 4.254 4.250 4.242
## 10 4.199 4.206 4.212 4.201 4.212 4.183 4.207
## 11 4.146 4.166 4.178 4.179 4.184 4.159 4.187
## 12 4.143 4.132 4.156 4.153 4.143 4.138 4.159
## 13 4.132 4.122 4.120 4.131 4.146 4.148 4.241
## 14 4.112    NA    NA    NA    NA    NA    NA
```

The Full Model

Let's take a look at the full model:

```
##
## Call:
## glm(formula = rings ~ sex + length + diameter + I(diameter^2) +
##     height + wwt + I(wwt^2) + mwt + I(mwt^2) + gwt + swt + I(swt^2),
##     family = poisson, data = abalone1.df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.242  -0.419  -0.097   0.271   3.722
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept)    0.91130    0.10023    9.09  < 2e-16 ***
## sexM           0.00272    0.01251    0.22  0.82813
## sexI          -0.06923    0.01717   -4.03  5.5e-05 ***
## length        -0.69542    0.31460   -2.21  0.02707 *
## diameter       5.98847    0.73460    8.15  3.6e-16 ***
## I(diameter^2) -6.06668    0.77213   -7.86  3.9e-15 ***
## height         1.31993    0.37879    3.48  0.00049 ***
## wwt            1.51762    0.22993    6.60  4.1e-11 ***
## I(wwt^2)      -0.23332    0.07303   -3.19  0.00140 **
## mwt           -3.54029    0.30672  -11.54  < 2e-16 ***
## I(mwt^2)       1.54743    0.23234    6.66  2.7e-11 ***
## gwt           -0.78031    0.18955   -4.12  3.8e-05 ***
## swt            1.43916    0.41080    3.50  0.00046 ***
## I(swt^2)      -0.86556    0.41318   -2.09  0.03618 *
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for poisson family taken to be 1)
##
##     Null deviance: 3579.3  on 3499  degrees of freedom
## Residual deviance: 1356.6  on 3486  degrees of freedom
```

Integer Estimates I

Since the number of rings is a count, it may be desirable to produce an integer estimate which is achieved by rounding the estimate to the nearest integer.

```
preds.int = round(predict(abalone2.glm,
                  newdata = abalone2.df, type = "response"), 0)
head(preds.int, 8)

##  1  2  3  4  5  6  7  8
##  8 11  6  9 10 12 11 10
```

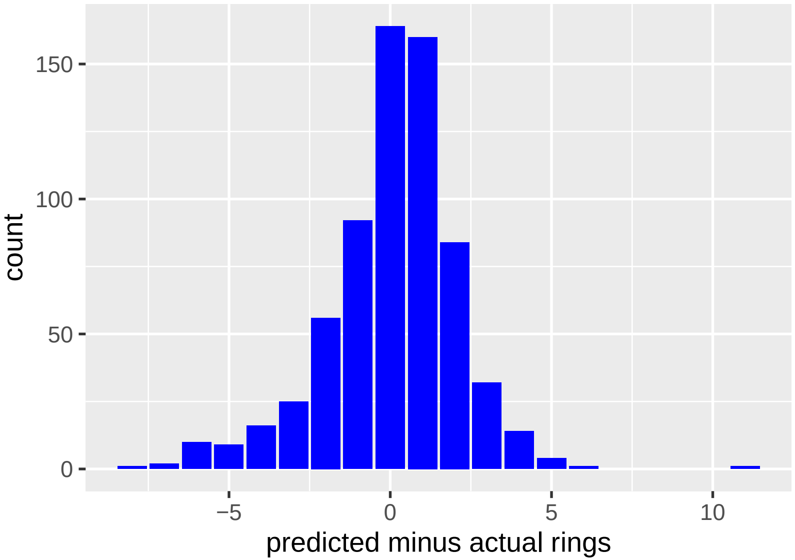
Integer Estimates II

We can tabulate the difference between the integer estimates and the true values as follows.

```
counts <- table(preds.int - abalone2.df$rings)
counts

##
##  -8  -7  -6  -5  -4  -3  -2  -1   0   1   2   3   4   5   6  11
##   1   2  10   9  16  25  56  92 164 160  84  32  14   4   1   1
```

- Most estimates are within 2 of the actual number of rings.

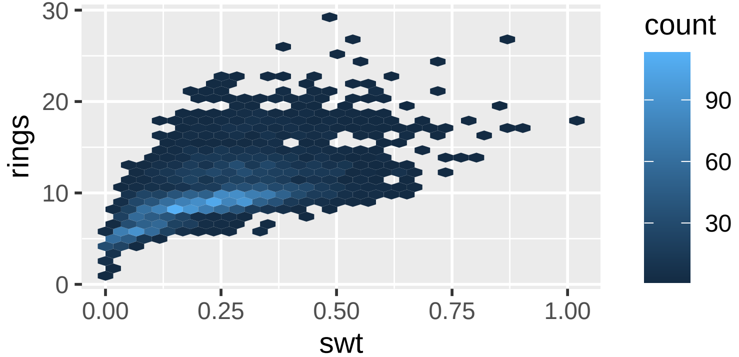
Barplot of Errors



More about Errors

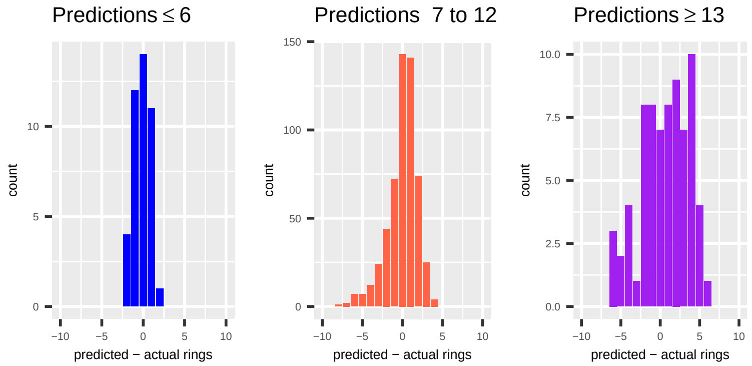
We would expect that it becomes more difficult to make precise predictions as the actual number of rings increases.

- For a Poisson distribution the variance increases as the mean increases.
- For this particular data set, relationships between the number of rings and the regressors tend to "level off" as the values of the regressors increase.



Conditional Barplots of Errors

In this case, it is useful to look at barplots of errors conditioned on the number of rings.



- Clearly the predictions become less precise as the number of rings increases which is what we expect for count data.

Important Lessons from the Abalone Data I

1. Using a Poisson regression to predict a count response variable is very similar to using ordinary regression to predict a continuous response.

2. The predictions are only valid for future observations that are consistent with the data used to create the model.
   - This data consists of observations collected on blacklip abalone at a particular location in Tasmania in the early 1990s. We can only be certain that prediction is valid under those conditions.

Important Lessons from the Abalone Data II

3. Using the same data to identify the model and estimate how well it predicts will usually result in an optimistic assessment.
    - In situations where there is a large number of observations the data can be divided into a training set and a test set.

4. The Poisson regression model estimates the mean count. If we need to predict a specific count for an individual then the obvious solution is to round to the nearest integer.

5. For count data we expect the size of prediction errors to increase as the expected number of counts increases.

Prediction using Logistic Regression

Prediction for binary response variables can be done using logistic regression. There are two main types of applications.

1. The goal is to estimate the probability of a positive response for a group of individuals who have the same values of the explanatory variables.
2. The goal is to predict whether an individual will have a positive response.

Applications of type 2 are very common and often seen in medical applications. As a result specific techniques (e.g., ROC curves) and terminology (e.g., specificity and sensitivity) have evolved.

More **R** Code I

For the abalone example we used the **ggplot2** package to produce some of the graphics. This package can produce elegant and complex plots but it can be a challenge to learn. Of course, there are introductory tutorials available on-line:

http://r-statistics.co/Complete-Ggplot2-Tutorial-Part1-With-R-Code.html

The hexbin plot on slide 90 was produced using the following code:

```r
ggplot(abalone.df, aes(swt, rings)) + geom_hex()
```

More R Code II

The array of plots on slide 67 was produced by:

```
pmat = ggpairs(abalone.df, columns = 2:8, upper = list(continuous = wrap("cor", size = 2)),
  lower = list(continuous = "blank"), diag = list(continuous = wrap("densityDiag", fill = "blue")))
        + theme_grey(base_size = 6)

pmat[2, 1] = ggplot(abalone.df, aes(length, diameter)) + geom_hex()
pmat[3, 1] = ggplot(abalone.df, aes(length, height)) + geom_hex()
pmat[4, 1] = ggplot(abalone.df, aes(length, wwt)) + geom_hex()
pmat[5, 1] = ggplot(abalone.df, aes(length, mwt)) + geom_hex()
pmat[6, 1] = ggplot(abalone.df, aes(length, gwt)) + geom_hex()
pmat[7, 1] = ggplot(abalone.df, aes(length, swt)) + geom_hex()
pmat[3, 2] = ggplot(abalone.df, aes(diameter, height)) + geom_hex()
pmat[4, 2] = ggplot(abalone.df, aes(diameter, wwt)) + geom_hex()
pmat[5, 2] = ggplot(abalone.df, aes(diameter, mwt)) + geom_hex()
pmat[6, 2] = ggplot(abalone.df, aes(diameter, gwt)) + geom_hex()
pmat[7, 2] = ggplot(abalone.df, aes(diameter, swt)) + geom_hex()
pmat[4, 3] = ggplot(abalone.df, aes(height, wwt)) + geom_hex()
pmat[5, 3] = ggplot(abalone.df, aes(height, mwt)) + geom_hex()
pmat[6, 3] = ggplot(abalone.df, aes(height, gwt)) + geom_hex()
pmat[7, 3] = ggplot(abalone.df, aes(height, swt)) + geom_hex()
pmat[5, 4] = ggplot(abalone.df, aes(wwt, mwt)) + geom_hex()
pmat[6, 4] = ggplot(abalone.df, aes(wwt, gwt)) + geom_hex()
pmat[7, 4] = ggplot(abalone.df, aes(wwt, swt)) + geom_hex()
pmat[6, 5] = ggplot(abalone.df, aes(mwt, gwt)) + geom_hex()
pmat[7, 5] = ggplot(abalone.df, aes(mwt, swt)) + geom_hex()
pmat[7, 6] = ggplot(abalone.df, aes(gwt, swt)) + geom_hex()

print(pmat)
```

More **R** Code III

The boxplot on slide 69 was produced by:

```
ggplot(abalone.df, aes(x = "", y = height)) +
       geom_boxplot(width = .3, fill = "red") + coord_flip()
```

The barplot on slide 89 was produced by:

```
mydata=data.frame(preds.int = preds.int, rings = abalone2.df$rings,
                  diffs = (preds.int - abalone2.df$rings))
ggplot(mydata, aes(diffs)) + geom_bar(fill = "blue") +
               labs(x = "predicted minus actual rings")
```

More R Code IV

The barplots on slide 91 were produced by:

```r
library('gridExtra')

My_Theme = theme(plot.title = element_text(size = 8),
  axis.title.x = element_text(size = 5), axis.text.x = element_text(size = 4),
  axis.title.y = element_text(size = 5), axis.text.y = element_text(size = 4))

p1 = ggplot(mydata[mydata$preds.int < 6.5, ], aes(diffs)) + geom_bar(fill = "blue")
   + labs(x = "predicted - actual rings") + ggtitle(expression("Predictions" <= 6))
   + My_Theme + xlim(-10, 10)

p2 = ggplot(mydata[mydata$preds.int < 12.5 & mydata$preds.int > 6.5, ], aes(diffs))
   + geom_bar(fill = "tomato") + labs(x = "predicted - actual rings")
   + ggtitle("Predictions  7 to 12") + My_Theme + xlim(-10, 10)

p3 = ggplot(mydata[mydata$preds.int > 12.5, ], aes(diffs)) + geom_bar(fill = "purple")
   + labs(x = "predicted - actual rings") + ggtitle(expression("Predictions" >= 13))
   + My_Theme + xlim(-10, 10)

grid.arrange(p1, p2, p3, nrow = 1)
```

The Road Ahead

We will concentrate on predictions for individuals as this is the more common type of application.

- We will begin by describing how predictions for individuals can be made using a logistic regression model.
- Then we will introduce the key characteristics (prediction error, sensitivity and specificity) that are used to evaluate predictive logistic regression models.
- Two data sets will be used to illustrate the methodology used to identify and assess a predictive logistic regression model.

15.5 Prediction for Individuals

Suppose we have fitted a logistic model and we want to use the model to make a prediction for a new observation. Given values of the explanatory variables x for the new observation, how do we predict the y-value, 0 or 1?

- Work out the estimated log odds for the new observation:

$$\text{Estimated log odds} = \widehat{\beta}_0 + \widehat{\beta}_1 x_1 + \cdots + \widehat{\beta}_k x_k.$$

- If log odds < 0 then predict a 0; if log odds ≥ 0 then predict a 1.
- Equivalent to predicting a 0 if the estimated probability is less than 0.5, since

$$\text{Prob} = \frac{\exp(\text{log odds})}{1 + \exp(\text{log odds})}.$$

Prediction Error, Sensitivity and Specificity

Prediction error is the probability of a wrong classification (0s predicted as 1s, 1s predicted as 0s).

It is often useful to evaluate how well the model works when the true value is 1 and the true value is 0 separately.

- **Sensitivity**: probability of predicting a 1 when the case is truly a 1: the true positive rate.
- **Specificity**: probability of predicting a 0 when the case is truly a 0: true negative rate ($1-$specificity is called the "false positive rate").

Calculating Sensitivity and Specificity

Consider the following table (also known as a confusion matrix):

		Model Predicts	
		Failure (0)	Success (1)
Actual	Failure (0)	300	200
Actual	Success (1)	250	600

$$\text{Estimated Specificity} = 300/(300+200) = 0.60,$$

$$\text{Estimated Sensitivity} = 600/(600+250) \approx 0.70,$$

$$\text{Estimated Prediction Error} = (200+250)/1350 \approx 0.33.$$

15.5.1 BIRADS Example

BIRADS (Breast Imaging Reporting and Data System) is a system developed by the American College of Radiology to translate information from mammograms into a score that summarises the outcome. The possible scores are:

0: Incomplete

1: Negative

2: Benign

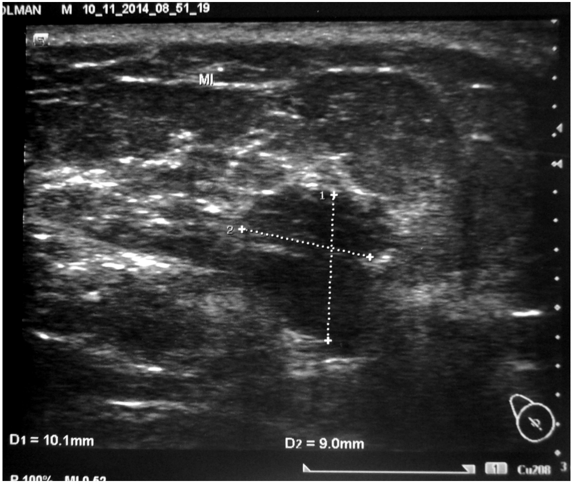
3: Probably benign

4: Suspicious

5: Highly suggestive of malignancy

6: Known biopsy—proven malignancy

Mammogram Image



BIRADS Data I

The data we are going to look at comes from the UCI Machine Learning Repository:

https://archive.ics.uci.edu/ml/datasets/Mammographic+Mass

The original source for this data is:

M. Elter, R. Schulz-Wendtland and T. Wittenberg (2007). "The prediction of breast cancer biopsy outcomes using two CAD approaches that both emphasize an intelligible decision process." Medical Physics 34(11): 4164–4172.

BIRADS Data II

The data was collected at the Institute of Radiology of the University Erlangen-Nuremberg between 2003 and 2006. It consists of the BIRADS score, the patient's age and three BIRADS attributes as well as whether the mass lesion turned out to be benign or malignant for 516 benign and 445 malignant masses.

- Strictly speaking predictions should only be made for new observations made by technicians at this institution during this time period. However, a practitioner may be willing to expand the scope to new observations made by technicians who have had similar training, are using similar protocols and are using similar equipment. Sometimes, the scope for predictive models are expanded beyond the strict bounds determined by data used to create the model. In such cases, it should be reasonable to expect that the relevant characteristics of the new observation are similar to those for the original data and to recognize that there is some risk in doing this.

BIRADS Data Background

The following background information comes from the UCI Machine Learning Repository.

> *Mammography is the most effective method for breast cancer screening available today. However, the low positive predictive value of breast biopsy resulting from mammogram interpretation leads to approximately 70% unnecessary biopsies with benign outcomes. To reduce the high number of unnecessary breast biopsies, several computer-aided diagnosis (CAD) systems have been proposed. These systems help physicians in their decision to perform a breast biopsy on a suspicious lesion seen in a mammogram or to perform a short term follow-up examination instead. This data set can be used to predict the severity (benign or malignant) of a mammographic mass lesion from BIRADS attributes and the patient's age.*

BIRADS Data Variables

We are going to fit a logistic regression model that uses the BIRADS score and age of the patient to predict the probability that an observed mammographic mass lesion is malignant (rather than benign).

The variables we are interested in are:

birads: the BIRADS score.

age: the age of the patient in years.

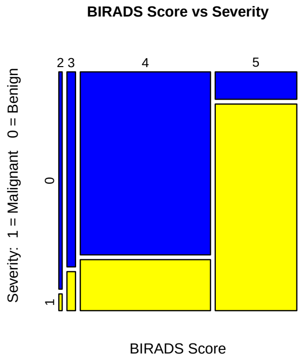
severity: $0 =$ benign and $1 =$ malignant.

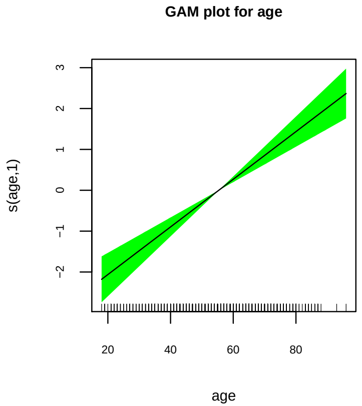
BIRADS Data Summary

An initial look at our data.

```
str(birads.df)

## 'data.frame': 938 obs. of  3 variables:
##  $ birads  : Factor w/ 4 levels "2","3","4","5": 4 3 4 3 4 3 3 4 4 4 ...
##  $ age     : num  67 43 58 28 74 65 70 42 57 60 ...
##  $ severity: Factor w/ 2 levels "0","1": 2 2 2 1 2 1 1 1 2 2 ...
##  - attr(*, "na.action")= 'omit' Named int [1:5] 442 452 678 870 907
##   ..- attr(*, "names")= chr [1:5] "444" "454" "684" "885" ...

summary(birads.df)

##  birads        age        severity
##  2: 14    Min.   :18.0   0:510
##  3: 36    1st Qu.:45.0   1:428
##  4:546    Median :57.0
##  5:342    Mean   :55.4
##           3rd Qu.:66.0
##           Max.   :96.0
```

BIRADS Data Plots





- Probability that mass is malignant increases with BIRADS score.
- A linear term seems appropriate for age.

BIRADS Logistic Regression Model

```
birads.glm = glm(severity~birads+age, binomial, data=birads.df)
summary(birads.glm)


##
## Call:
## glm(formula = severity ~ birads + age, family = binomial, data = birads.df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -2.689  -0.675  -0.331   0.500   2.400
##
## Coefficients:
##             Estimate Std. Error z value Pr(>|z|)
## (Intercept)  -5.8109     1.1462   -5.07  4.0e-07 ***
## birads3       1.3103     1.1561    1.13     0.26
## birads4       1.3573     1.0622    1.28     0.20
## birads5       4.3877     1.0709    4.10  4.2e-05 ***
## age           0.0583     0.0075    7.77  7.8e-15 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1293.2  on 937  degrees of freedom
## Residual deviance:  789.0  on 933  degrees of freedom
## AIC: 799
##
## Number of Fisher Scoring iterations: 5
```

Predictions

Suppose we adopt the convention of predicting a 0 (benign) if the estimated probability is ≤ 0.5 and predicting a 1 (malignant) otherwise. Then our confusion matrix is given by:

```r
table(actual = birads.df$severity, pred = round(fitted(birads.glm)))
```

```
##       pred
## actual   0   1
##      0 465  45
##      1 118 310
```

Estimates of sensitivity, specificity and prediction error are:

$$\text{Estimated Specificity} = 465/(465+45) \approx 0.912,$$

$$\text{Estimated Sensitivity} = 310/(310+118) \approx 0.724,$$

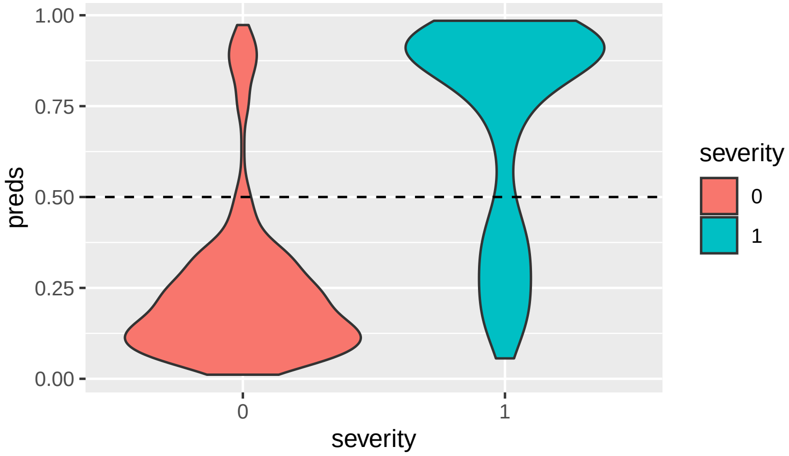
$$\text{Estimated Prediction Error} = (45{+}118)/(465{+}45{+}310{+}118) \approx 0.174.$$

Adjusting the Predictions

We have predicted `severity` $= 1$ (malignant) if the estimated probability is ≥ 0.5 and `severity` $= 0$ (benign) otherwise.

- We can generalize this to predict a case (malignant) if $\widehat{p} \geq c$ for some constant c where $0 \leq c \leq 1$.
  - If c is 0 every observation will be predicted as a case (1) and we will have sensitivity $= 1$ and specificity $= 0$.
  - If c is 1 every observation will be predicted as a non-case (0) and we will have sensitivity $= 0$ and specificity $= 1$.
  - As c is varied from 0 to 1, sensitivity goes from 1 to 0 and specificity goes from 0 to 1.
- Thus c can be set to balance the relative importance of sensitivity and specificity.

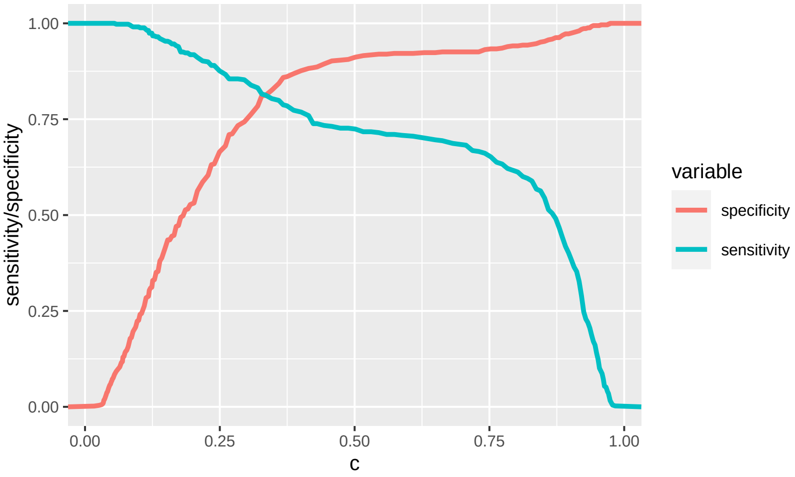
Violinplots for `severity` = 0 and `severity` = 1



- Sensitivity is the proportion above the dashed line for `severity` = 1 and specificity is the proportion below the dashed line for `severity` = 0.
- Shift the line up: sensitivity decreases and specificity increases.
- Shift the line down: sensitivity increases and specificity decreases.

Sensitivity/Specificity Trade-Off

Another way to visualize this trade-off is to plot both sensitivity and specificity versus *c* on the same graph.



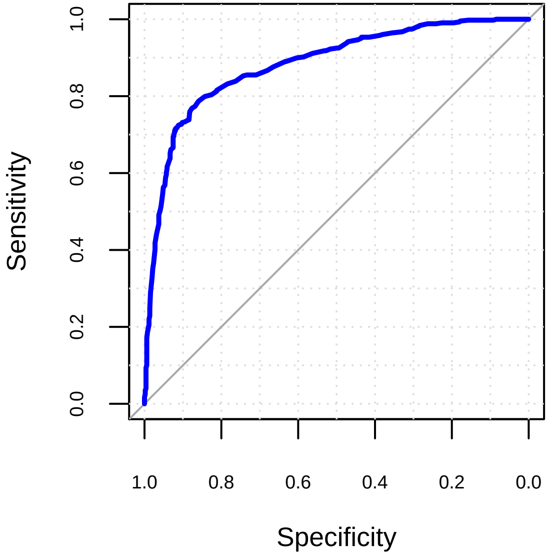
ROC Curves

A ROC (Receiver Operating Characteristic) curve is perhaps the most common method used to visualize the trade-off between sensitivity and specificity.

- The ROC curve is a plot of the true positive rate (sensitivity) versus the false positive rate (1−specificity) as c changes.
- A number of **R** packages include functions that produce ROC curves—we will use the **pROC** package.
  - This package plots sensitivity versus specificity but reverses the scale for sensitivity (1 to 0 rather than 0 to 1). This results in the same graph as plotting sensitivity versus 1−specificity.

```r
birads.roc <- roc(response = birads.df$severity,
                  predictor = fitted.values(birads.glm))
plot(birads.roc, col = "blue", grid = TRUE, lwd=2.5)
```

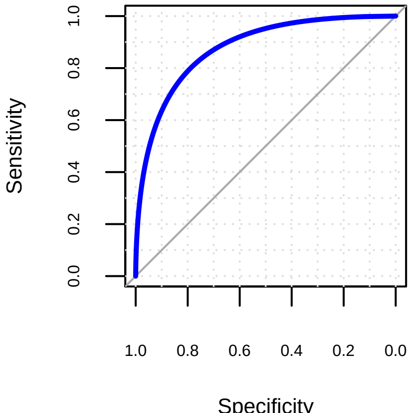
ROC Curve for BIRADS Model



Smoothed ROC Curve

There is also an option to create a smoothed ROC curve.

```
smooth.roc = roc(response = birads.df$severity,
                 predictor = fitted.values(birads.glm), smooth = TRUE)
plot(smooth.roc, col = "blue", grid = TRUE, lwd=2.5,
     cex.lab=0.7, cex.axis=0.5)
```

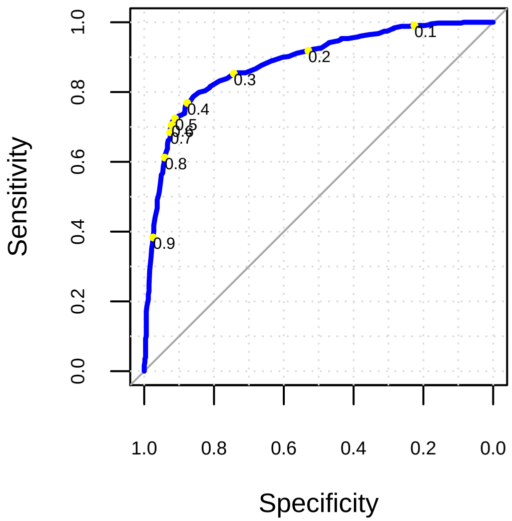


Choosing c for a Predictor I

Selecting a value of c depends on the relative importance of sensitivity and specificity.

- For the BIRADS example the consequences of a false positive (benign mass classified as malignant) and a false negative (malignant mass classified as benign) are clearly different—we would expect that the consequences of a false negative are more serious.
- By adjusting c we can obtain any combination of sensitivity and specificity that lies on the ROC curve.

Choosing c for a Predictor II

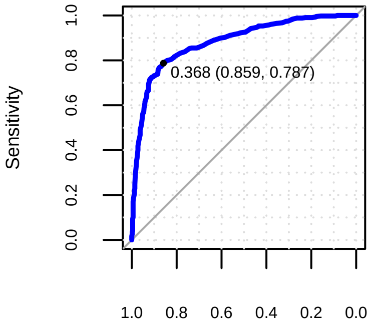


Choosing c for a Predictor III

For situations where sensitivity and specificity are equally important a couple of sensible options are:

1. Choose c to maximize sensitivity + specificity.
   - Use `print.thres = "best"` to print this value on the ROC curve.

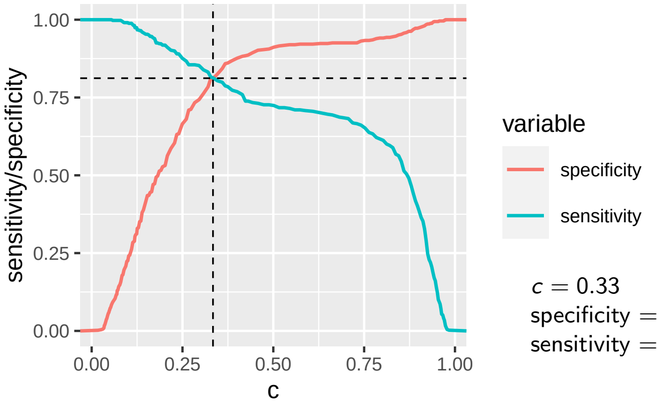
```r
plot(birads.roc, print.thres = "best", col = "blue", grid = TRUE,
     lwd=2.5, cex.lab=0.6, cex.axis=0.5, print.thres.cex=0.5)
```



$c = 0.368$,
specificity $= 0.859$,
sensitivity $= 0.787$.

Choosing c for a Predictor IV

2. Choose c to maximize the minimum of sensitivity and specificity.
   - This is the point where the lines cross in the following plot.



$c = 0.33$
specificity $= 0.81$
sensitivity $= 0.81$

Still More **R** Code I

When you use the `roc()` function, it produces a vector of threshold values and vectors of the corresponding estimated values of the sensitivity and specificity.

For the `birads.roc` object we created, these are contained in `birads.roc$thresholds`, `birads.roc$sensitivities` and `birads.roc$specificities` respectively.

```
str(birads.roc$thresholds)

##  num [1:158] -Inf 0.0169 0.0262 0.031 0.0328 ...

str(birads.roc$sensitivities)

##  num [1:158] 1 1 1 1 1 1 1 1 1 1 ...

str(birads.roc$specificities)

##  num [1:158] 0 0.00196 0.00392 0.00588 0.00784 ...
```

Still More **R** Code II

To find c to maximize sensitivity + specificity.

```
(ind1 <- which.max(birads.roc$sensitivities + birads.roc$specificities))

## [1] 84

birads.roc$thresholds[ind1]

## [1] 0.3676

birads.roc$sensitivities[ind1]

## [1] 0.7874

birads.roc$specificities[ind1]

## [1] 0.8588
```

Still More **R** Code III

To find c to maximize the minimum of sensitivity and specificity.

```
(ind2 <- with(birads.roc, which.min(abs(sensitivities - specificities))))

## [1] 81

birads.roc$thresholds[ind2]

## [1] 0.3344

birads.roc$sensitivities[ind2]

## [1] 0.8131

birads.roc$specificities[ind2]

## [1] 0.8118
```

Still More **R** Code IV

The following code produced the plot on slide 122:

```
mynew.df = data.frame(cc          = birads.roc$thresholds,
                      specificity = birads.roc$specificities,
                      sensitivity = birads.roc$sensitivities)

mynew2.df = melt(mynew.df, id="cc")

ggplot()
  + geom_line(data = mynew2.df, aes(x=cc, y=value, color=variable), size=1.)
  + labs(x = "c", y = "sensitivity/specificity")
  + geom_vline(xintercept = .334, lty = 2, size = .3)
  + geom_hline(yintercept = .812, lty = 2, size = .3)
  + theme_grey(base_size = 9)
```

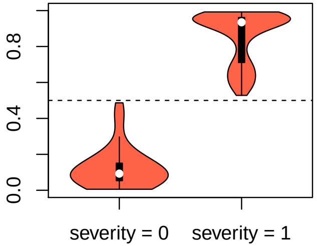
Evaluating a Logistic Predictive Model

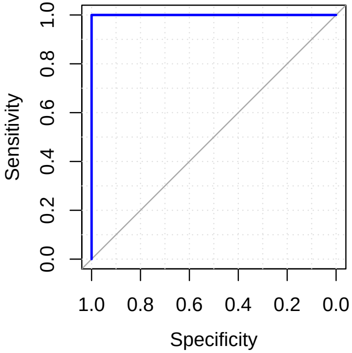
The ROC curve summarises a predictive model's ability to distinguish between cases and non-cases over all possible choices for the cutoff (c).

- Often the area under the ROC curve (AUC) is used as a summary measure of a model's predictive ability—this is fine but keep in mind that a single number cannot possibly retain all the information in a curve.
- To help understand how AUC works, we will consider two hypothetical predictors:
  - The perfect predictor is always able to correctly identify both cases and non-cases.
  - The pointless predictor has no ability to distinguish between cases and non-cases—it does no better than using a random process.

The Perfect Predictor ROC Curve

A perfect predictor occurs when the set of $\widehat{p}$s for $Y = 1$ does not overlap with that for $Y = 0$.

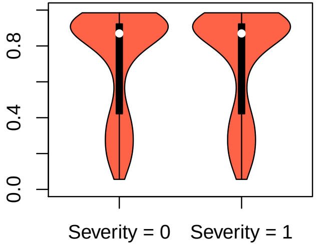


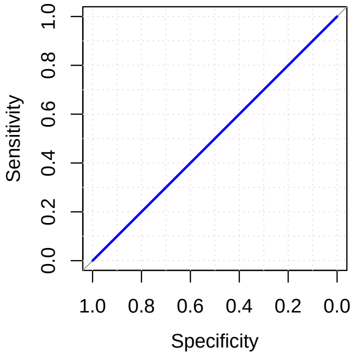


- Can choose c such that our estimated sensitivity and specificity are both 1.

The Pointless Predictor ROC Curve

If the set of $\widehat{p}$s for $Y = 1$ is identical to that for $Y = 0$ then our predictor is not useful for separating cases from non-cases.



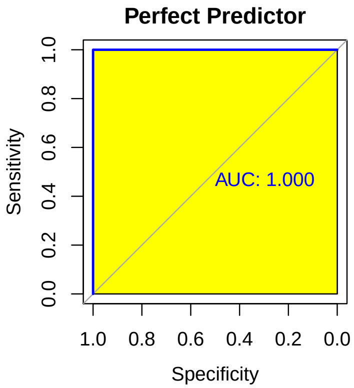


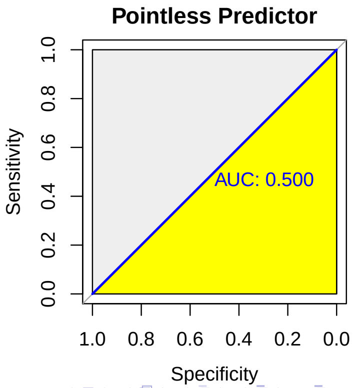
- Estimated sensitivity is always equal to 1 minus the estimated specificity.
- For a predictor to be useful, its ROC curve must be above this diagonal line—the further above the line the better.

Evaluating a Predictor

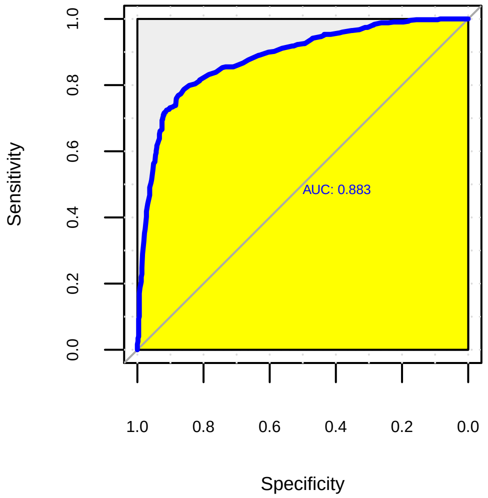
The area under the ROC curve (AUC) is often used as a measure of goodness for predictors.

- For a perfect predictor, the AUC is 1.
- For a pointless predictor, the AUC 0.5.
- The closer the AUC is to 1, the better the predictor.





AUC for the BIRADS Model



- The estimated AUC is 0.883 with a 95% CI of 0.861 to 0.905

Interpreting AUC

Some points to keep in mind when interpreting AUC:

- AUC is an indication of a model's predictive ability averaged over all possible thresholds.
- What constitutes a "good" value of AUC is situation dependent—some applications are more noisy than others making prediction less precise.
- AUC can be used to compare competing models for a particular application. Keep in mind that we are dealing with estimates of the AUC.

BIRADS Example Recap

The BIRADS example illustrates how a logistic regression model can be use to predict the occurrence of a specified event (case).

- The logistic model relates the probability of a case to the levels of the regressors.
- To translate the logistic model into a predictive model: specify a threshold c and predict a case if $\widehat{p} \geq c$ and a non-case otherwise.
- Three important characteristics of the predictive model are sensitivity (probability of correct prediction for an actual case), specificity (probability of correct prediction for a non-case) and the prediction error (probability of an incorrect prediction).
- The ROC curve summarizes the trade-off between sensitivity and specificity as the threshold is varied.
- AUC is a summary measure of a model's predictive ability.

15.5.2 Gardasil Example

Now we will consider a different data set to illustrate model selection for a predictive logistic model. The data pertains to female patients between the ages of 11 and 26 who visited one of four Johns Hopkins Medical Institutions in Maryland, USA, for the purpose of being vaccinated with the anti-human papillomavirus (HPV) medication Gardasil (note that we should only make predictions for future observations that share these characteristics). The vaccination protocol requires three shots to be completed within one year of the first shot.

From Wikipedia:

> *Human papillomavirus infection is an infection by human papillomavirus (HPV). Most HPV infections cause no symptoms and resolve spontaneously. In some people, an HPV infection persists and results in warts or precancerous lesions. The precancerous lesions increase the risk of cancer of the cervix, vulva, vagina, penis, anus, mouth, or throat. Nearly all cervical cancer is due to HPV with two types, HPV16 and HPV18, accounting for 70% of cases.*

Gardasil

Gardasil Data

Response

Completed: did the patient complete the three-shot regimen within 12 months (0 = no, 1 = yes)?

Regressors

Age: age of the patient in years.

AgeGroup: patient's age group (0 = 11–17 years, 1 = 18–26).

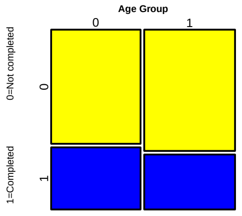
Race: patient's race (0 = white, 1 = black, 2 = Hispanic, 3 = other/unknown).

InsuranceType: type of insurance that the patient had (0 = medical assistance, 1 = private insurance, 2 = hospital based [EHF], 3 = military).

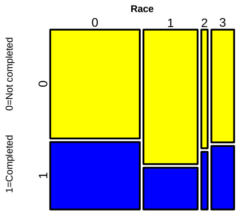
Location: clinic visited by patient (1 = Odenton, 2 = White Marsh, 3 = Johns Hopkins Outpatient Center, 4 = Bayview).

PracticeType: type of practice that the patient visited (0 = pediatric, 1 = family practice, 2 = OB-GYN).

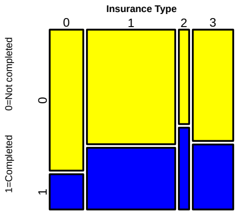
Mosaic plots I



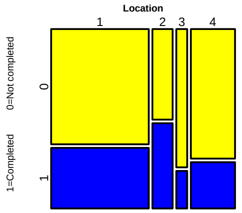
0 = 11 to 17 years 1 = 18 to 26 years



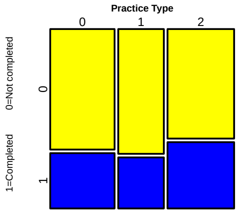
0=white 1=black 2=hispanic 3=other



0 = medical assist. 1 = private 2 = hospital 3 = military



1 = Odenton 2 = White Marsh 3 = Johns Hopkins 4 = Bay



0 = pediatric 1 = family practice 2 = OB−GYN

Divide Data into Two Parts

The mosaic plots indicate some differences in the proportion of patients that complete the vaccination programme for different levels of the regressors. We will now consider building a predictive model. Given we have a reasonably large data set (1413 observations), we will divide the data into training and test sets.

```
N <- 1000;  # set.seed(something) should be used here
sam = sample(1:nrow(gardasil.df))
gard1.df = gardasil.df[sam[1:N], ]
row.names(gard1.df) = 1:nrow(gard1.df)
gard2.df = gardasil.df[sam[(N+1):nrow(gardasil.df)], ]
row.names(gard2.df) = 1:nrow(gard2.df)
```

- These will be used for model selection and model fitting (`gard1.df`) and model evaluation (`gard2.df`).

Data Summaries

We should take a look at the output from `summary` for the training set and test set to make sure that for each variable the splits between levels are reasonably consistent.

```
summary(gard1.df)


##       Age        AgeGroup Race    Completed InsuranceType Location PracticeType
##  Min.   :11.0   0:486    0:518   0:675     0:192         1:568    0:359
##  1st Qu.:15.0   1:514    1:322   1:325     1:516         2:113    1:260
##  Median :18.0            2: 37             2: 61         3: 66    2:381
##  Mean   :18.7            3:123             3:231         4:253
##  3rd Qu.:22.0
##  Max.   :26.0


summary(gard2.df)


##       Age        AgeGroup Race    Completed InsuranceType Location PracticeType
##  Min.   :11.0   0:215    0:214   0:269     0: 83         1:230    0:156
##  1st Qu.:15.0   1:198    1:121   1:144     1:207         2: 52    1:105
##  Median :18.0            2: 15             2: 23         3: 23    2:152
##  Mean   :18.3            3: 63             3:100         4:108
##  3rd Qu.:22.0
##  Max.   :26.0
```

Model Selection

Our approach to searching for a predictive model will be similar to what we have done before:

- Use `dredge()` to search for promising models using the AICc and BIC criteria using the training data.
- Evaluate the short-listed models using a criteria directly related to their predictive ability (in this case we will use AUC) using the test data.

Model Searches I

Search using AICc as the selection criteria.

```
gard.glm <- glm(Completed ~ Age+AgeGroup+ Race+InsuranceType+
                Location+PracticeType, binomial, data = gard1.df)

gard1.fits = dredge(gard.glm)
head(gard1.fits)


## Global model call: glm(formula = Completed ~ Age + AgeGroup + Race + InsuranceType +
##     Location + PracticeType, family = binomial, data = gard1.df)
## ---
## Model selection table
##     (Intrc)      Age AgGrp InsrT Loctn PrctT Race df logLik AICc delta weight
## 55 -1.17500              +     +           +    + 10 -603.1 1226  0.00  0.325
## 56 -0.62660 -0.03773     +     +           +    + 11 -602.4 1227  0.69  0.231
## 54 -0.09464 -0.07440           +           +    + 10 -603.6 1227  1.08  0.190
## 63 -0.95770              +     +     +     +    + 13 -601.1 1229  2.22  0.107
## 64 -0.39930 -0.03758     +     +     +     +    + 14 -600.4 1229  2.94  0.075
## 23 -1.41600              +     +           +       7 -607.6 1229  3.00  0.073
## Models ranked by AICc(x)
```

Model Searches II

Search using BIC as the selection criteria.

```
gard2.fits = dredge(gard.glm, rank="BIC")
head(gard2.fits)


## Global model call: glm(formula = Completed ~ Age + AgeGroup + Race + InsuranceType +
##     Location + PracticeType, family = binomial, data = gard1.df)
## ---
## Model selection table
##    (Intrc)      Age AgGrp InsrT PrctT df logLik  BIC delta weight
## 7  -1.3340             +     +         5 -614.4 1263  0.00  0.349
## 23 -1.4160             +     +     +   7 -607.6 1264  0.17  0.321
## 5  -1.4330                   +         4 -619.1 1266  2.38  0.106
## 6  -0.7041 -0.04501          +         5 -615.7 1266  2.42  0.104
## 22 -0.3867 -0.07186          +     +   7 -609.0 1266  2.83  0.085
## 1  -0.7309                             1 -630.6 1268  4.63  0.034
## Models ranked by BIC(x)
```

- BIC is selecting smaller models than AICc—in fact the sixth model on the BIC list is the null (intercept only) model.

Estimates of the AUCs I

The following programmes estimate the value of AUC using the test data for the best 30 models from each of our lists.

The AICc list:

```
options(width=66)
out = rep(0, 30)
for (i in 1:30) {
  newpreds = predict(get.models(gard1.fits, i)[[1]],
                     newdata = gard2.df, type = "response")
  my.roc = roc(response = gard2.df$Completed,
                          predictor = newpreds, ci = TRUE)
  out[i] = my.roc$auc
}
round(out, 3)

##  [1] 0.612 0.623 0.626 0.635 0.639 0.589 0.644 0.627 0.602 0.632
## [11] 0.634 0.610 0.635 0.637 0.636 0.631 0.621 0.614 0.636 0.637
## [21] 0.617 0.625 0.616 0.611 0.630 0.629 0.628 0.641 0.634 0.573
```

Estimates of the AUCs II

The BIC list:

```
options(width=66)
out = rep(0, 30)
for (i in 1:30) {
  newpreds = predict(get.models(gard2.fits, i)[[1]],
                     newdata = gard2.df, type = "response")
  my.roc = roc(response = gard2.df$Completed,
               predictor = newpreds, ci = TRUE)
  out[i] = my.roc$auc
}
round(out, 3)

##  [1] 0.573 0.589 0.571 0.595 0.610 0.500 0.612 0.551 0.602 0.599
## [11] 0.582 0.581 0.625 0.500 0.600 0.621 0.518 0.574 0.614 0.594
## [21] 0.588 0.618 0.617 0.593 0.626 0.612 0.611 0.537 0.597 0.630
```

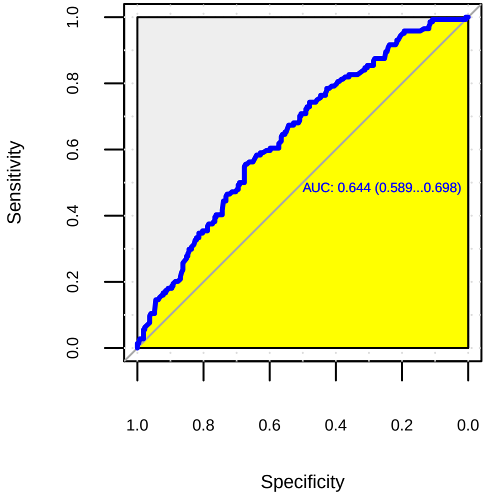
- The highest estimate of AUC (0.644) corresponds to model 7 from the AICc list. but there are a number of other models with estimates that are nearly as high.

Model 7 from the AICc list

```
model7 = get.models(gard1.fits, 7)[[1]]
summary(model7)


##
## Call:
## glm(formula = Completed ~ Age + InsuranceType + Location + PracticeType +
##     Race + 1, family = binomial, data = gard1.df)
##
## Deviance Residuals:
##    Min      1Q  Median      3Q     Max
## -1.473  -0.906  -0.710   1.272   2.112
##
## Coefficients:
##                Estimate Std. Error z value Pr(>|z|)
## (Intercept)      0.1226     0.4435    0.28   0.7822
## Age             -0.0723     0.0226   -3.20   0.0014 **
## InsuranceType1   0.6553     0.2488    2.63   0.0084 **
## InsuranceType2   1.0636     0.3380    3.15   0.0017 **
## InsuranceType3   0.7266     0.2952    2.46   0.0138 *
## Location2        0.2926     0.2582    1.13   0.2571
## Location3       -0.2313     0.3540   -0.65   0.5135
## Location4       -0.2715     0.2376   -1.14   0.2532
## PracticeType1   -0.2453     0.2311   -1.06   0.2884
## PracticeType2    0.3520     0.2488    1.41   0.1571
## Race1           -0.4787     0.1709   -2.80   0.0051 **
## Race2            0.0440     0.3756    0.12   0.9068
## Race3            0.0543     0.2201    0.25   0.8050
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1261.2  on 999  degrees of freedom
## Residual deviance: 1203.1  on 987  degrees of freedom
```

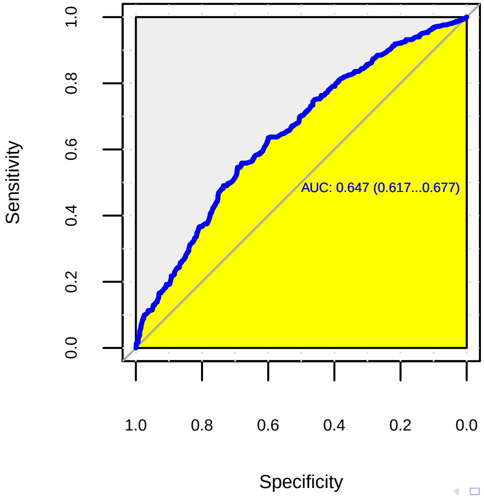
The ROC for Model 7

If we estimate the ROC curve using the test data we get



The ROC for Full Data

Given that “model 7” has been selected, we should refit this model using the entire data set. If we do this and estimate the ROC curve using the full data set we get:



Gardasil Example Summary

The Gardasil data was used to illustrate how we can approach model selection when we are looking for a predictive model for a binary response.

- AICc and BIC were used as a convenient way of identifying a set of promising subset models.
- AUC was then used to select the "best" model from this set.
- There were a number of models that had very similar estimated values of AUC as the selected model—any of these could be used as a sensible predictive model.
- None of the models had a very high value of AUC and thus none perform very well in predicting completion.
  - Conclude that this set of regressors cannot produce precise predictions about completion—to get a better predictive model it would be necessary to expand the set of possible regressors.

Prediction for Binary Responses Takeaways

Several important takeaways with respect to using a logistic regression model to predict a binary response variable are similar to those for predictive models based on ordinary regression and Poison regression.

- The predictions are only valid for future observations that are consistent with the data used to create the model.
- Using the same data to identify the model and estimate how well it predicts will usually result in an optimistic assessment.
- The logistic regression model estimates the probability of an event. If we need to predict a specific event outcome (occurs or doesn't occur) then a threshold c is set such that the event is predicted to occur if the estimated probability is above c and to not occur otherwise.
    - A ROC curve is often used to assess the trade-off between sensitivity (true positive rate) and specificity (true negative rate) as the value of c is varied.

Prediction Models Conclusions

This brings us to the end of our section on predictive models. Some important "big picture" take-aways:

1. The purpose of a predictive model is to predict future observations. The future observations must be consistent with the data used to create the model.
2. Predictive models should be evaluated in terms of characteristics of the distribution of their prediction errors.
3. Using the same data for model selection, model fitting and evaluation will almost always result in an optimistic estimate of predictive ability. To counteract this problem we can randomly split the data into a training set and test set or use cross-validation.
4. Often a number of models will be identified that are quite similar in terms of their predictive ability.