STATS 330

Handout 9
Model selection

Department of Statistics, University of Auckland

Model selection

The data sets we've looked at so far in this course have been very simple, with just one or two explanatory variabes. We have been able to fit all the models we have considered, and find one we are happy with.

However, many real-life data sets have a large number of explanatory variables, resulting in an overwhelming number of models you could possibly fit. You are faced with questions like

- Which variables should I include in the model?
- Should I include any nonlinear terms, like quadratics?
  - If so, which variables should I try using nonlinear terms for...
  - And how wiggly should I make these relationships?
- Should I include any interaction effects?
  - If so, between which variables?

If you have 10 possible explanatory variables there are $2^{10} = 1024$ possible models you could fit, **before** you even begin to consider nonlinear or interaction effects!

Model selection

We want to find an appropriate model from which to draw conclusions. We should ensure that we are happy with the assumptions of our final model, or that it is robust to any violations—but beyond that, what process do we take to find this model?

There are no hard and fast rules. If you give two different statisticians the same data set and the same questions to answer, they may very well end up doing different things. Statistical modelling is often considered an art as well as a science.

Model selection

In some cases decisions about whether or not you include particular variables, or exactly how you incorporate them into your model, will reflect what is already known about the process you are modelling.

For example, it is widely accepted within the scientific community that the concentration of greenhouse gases in the atmosphere is related to the warming of the planet. If we are fitting a model to predict long-term changes in temperature, we should consider using atmospheric greenhouse gas concentrations as an explanatory variable in our model, regardless of what our statistical model-selection tools tell us.

On the other hand, in many cases we must rely on the data themselves to guide us towards a sensible model, because it may not be known how explanatory variables are related to the response.

Overview

In this handout, we introduce a few tools that you can use to point you in a sensible direction during a model-fitting journey.

We will consider the following topics:

- Considering nonlinear terms
- Model comparison
  - Using the deviance
  - Using information-theoretic measures
    - AIC
    - AICc
    - BIC
- Model-search strategies
- Transformations of the response

Wine quality data

A panel of wine experts judged a total of 1599 Portugese red wines, and determined whether or not each one was of 'good quality'. Various physiochemical properties were also measured on the wines.

We wish to build a model, perhaps to understand what physiochemical properties result in a 'good quality' wine, or perhaps to predict whether or not a new wine will be 'good quality'.

Wine quality data

- good.quality: Whether or not the wine was rated 'good quality', either `TRUE` or `FALSE`.
- alcohol: Percentage alcohol by volume.
- fixed.acidity: Fixed concentration of tartaric acid (g per dm^3).
- residual.sugar: Concentration of residual sugars (g per dm^3).
- chlorides: Concentration of sodium chloride (g per dm^3).
- free.sulfur.dioxide: Concentration of free sulfur dioxide (mg per dm^3)
- density: Density of the wine (g per cm^3).
- pH: Acidity of the wine on the pH scale.
- sulphates: Concentration of potassium sulphate (g per dm^3).

Wine quality data

Here are the data from the first fifteen wines, for four of the variables:

```
##    fixed.acidity volatile.acidity citric.acid good.quality
## 1            7.4             0.700        0.00        FALSE
## 2            7.8             0.880        0.00        FALSE
## 3            7.8             0.760        0.04        FALSE
## 4           11.2             0.280        0.56        FALSE
## 5            7.4             0.700        0.00        FALSE
## 6            7.4             0.660        0.00        FALSE
## 7            7.9             0.600        0.06        FALSE
## 8            7.3             0.650        0.00         TRUE
## 9            7.8             0.580        0.02         TRUE
## 10           7.5             0.500        0.36        FALSE
## 11           6.7             0.580        0.08        FALSE
## 12           7.5             0.500        0.36        FALSE
## 13           5.6             0.615        0.00        FALSE
## 14           7.8             0.610        0.29        FALSE
## 15           8.9             0.620        0.18        FALSE
```

Wine quality data

It is sensible to consider a logistic regression model. We wish to determine how the physiochemical properties are related to the probability that a wine will be of 'good quality'.

The first model that comes to mind is a simple model that uses all the explanatory variables available:

```
wine.fit <- glm(good.quality ~ fixed.acidity +
                    residual.sugar + chlorides +
                    free.sulfur.dioxide +
                    density + pH + sulphates + alcohol,
                family = "binomial", data = wine.df)
```

Note that when we have ungrouped data, we do not need to specify the number of trials for a binomial response.

Detecting nonlinear relationships

It is sensible to consider adding nonlinear terms to our model.

For example, the relationship between percentage alcohol and the probability of a wine being 'good quality' is unlikely to be monotonic. People probably prefer wines between 10–15%.

When we have only a small number of numeric variables we can detect the need for nonlinearity from residuals, but it becomes more difficult when we have many variables.

How can we determine which variables might need nonlinear terms?

Detecting nonlinear relationships

Using GAM plots

One option is to use GAM plots. To create GAM plots, do the following:

1. Install the `mgcv` package if you don't have it already.
2. Load the `mgcv` package.
3. Replace your `glm()` function with `gam()`.
4. Wrap your numeric variables with `s()`.
5. Use the `plot()` function on the resulting model object.

You will get a plot for each variable in your model. If a variable's plot is not a straight line, consider using a polynomial term that approximates the shape of the function you see.

Detecting nonlinear relationships

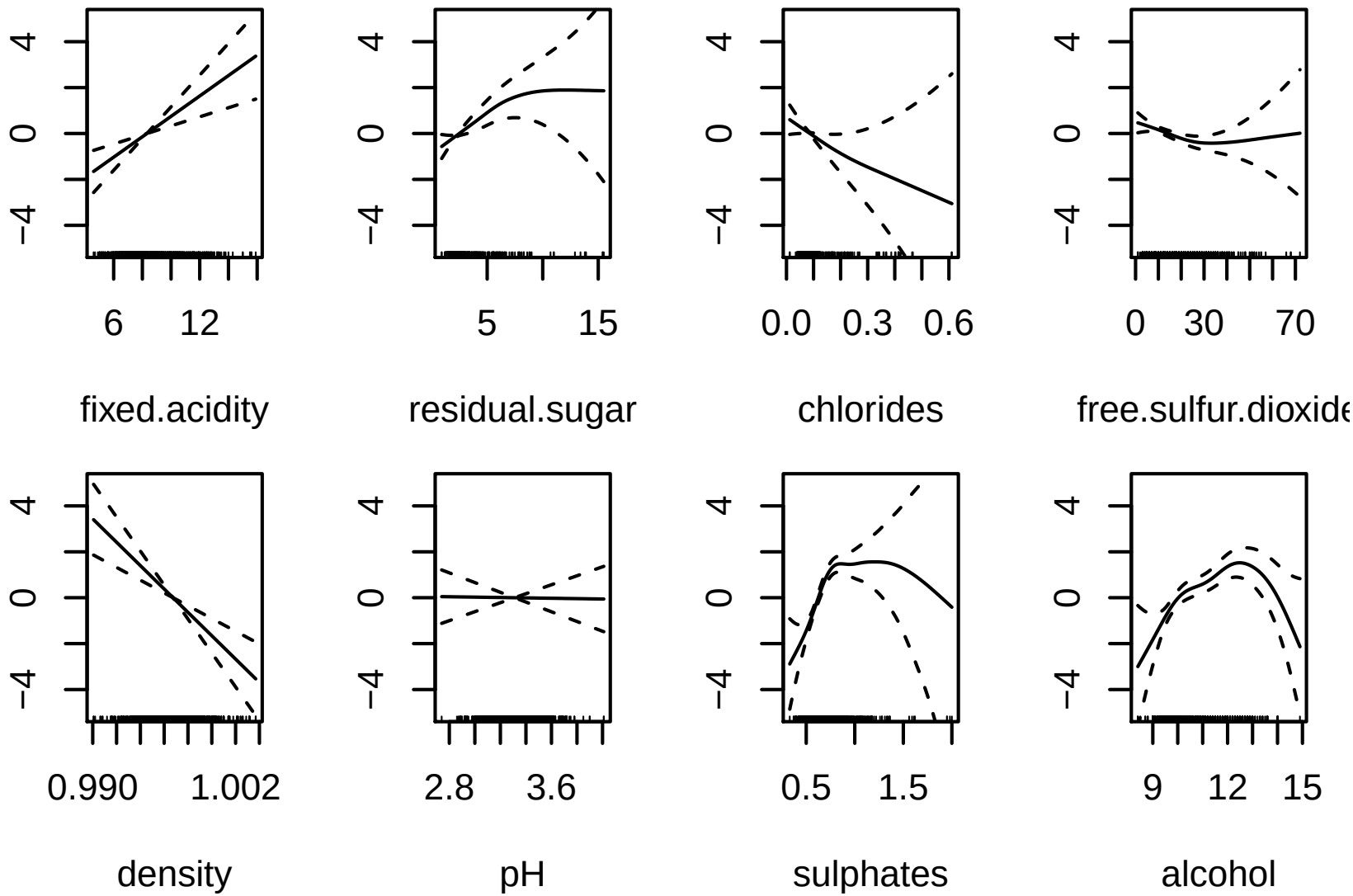
Using GAM plots

Here's how we do it:

```
## Step 1. Only if you don't have mgcv already.
install.packages("mgcv")
## Step 2.
library(mgcv)
## Steps 3 and 4.
gam.fit <- gam(good.quality ~ s(fixed.acidity) +
                    s(residual.sugar) + s(chlorides) +
                    s(free.sulfur.dioxide) +
                    s(density) + s(pH) + s(sulphates) +
                    s(alcohol), family = "binomial",
                data = wine.df)
## Step 5.
plot(gam.fit)
```

Detecting nonlinear relationships

Using GAM plots



Detecting nonlinear relationships

Using GAM plots

For each plot, consider whether or not you could draw a straight line that remained within the dotted lines. If not, consider fitting a polynomial for that variable. The order of your polynomial is determined by how wiggly your line would need to be to stay within the dotted lines.

- We probably need a quadratic term for alcohol by volume.
- We might need a quadratic term for concentration of potassium sulphate.
- We could consider quadratics for concentrations of sodium chloride and free sulfur dioxide.

Detecting nonlinear relationships

Using GAM plots

Adding quadratic terms for the four variables:

```
wine.poly.fit <- glm(good.quality ~ fixed.acidity +
                       residual.sugar + chlorides +
                       I(chlorides^2) +
                       free.sulfur.dioxide +
                       I(free.sulfur.dioxide^2) +
                       density + pH + sulphates +
                       + I(sulphates^2) + alcohol +
                       I(alcohol^2),
                     family = "binomial", data = wine.df)
```

As suspected, the model summary indicates we need the quadratics for alcohol by volume and concentration of potassium sulphate, but possibly not concentrations of sodium chloride and free sulfur dioxide.

Detecting nonlinear relationships

Using GAM plots

```
summary(wine.poly.fit)

## Call:
## glm(formula = good.quality ~ fixed.acidity + residual.sugar +
##
## Coefficients:
##                             Estimate Std. Error z value Pr(>|z|)
## (Intercept)                3.834e+02  1.071e+02   3.579 0.000345 ***
## fixed.acidity              4.063e-01  1.228e-01   3.310 0.000934 ***
## residual.sugar             2.558e-01  7.809e-02   3.276 0.001054 **
## chlorides                 -1.376e+01  6.767e+00  -2.034 0.041991 *
## I(chlorides^2)             2.194e+01  1.693e+01   1.296 0.195120
## free.sulfur.dioxide       -5.880e-02  2.598e-02  -2.263 0.023639 *
## I(free.sulfur.dioxide^2)   8.204e-04  5.588e-04   1.468 0.142093
## density                   -4.411e+02  1.098e+02  -4.017 5.89e-05 ***
## pH                        -3.791e-01  1.020e+00  -0.372 0.710174
## sulphates                  2.490e+01  4.941e+00   5.040 4.66e-07 ***
## I(sulphates^2)            -1.255e+01  3.189e+00  -3.934 8.36e-05 ***
## alcohol                    6.873e+00  1.498e+00   4.587 4.50e-06 ***
## I(alcohol^2)              -2.755e-01  6.563e-02  -4.198 2.69e-05 ***
## ---
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1269.92  on 1598  degrees of freedom
```

Model comparison

Deviance

In some cases, we wish to test the null hypothesis that a particular term in our model is not required. For example, on the previous slide we looked at p-values for the quadratic terms to determine whether or not we had evidence to suggest they exist.

When we look at each p-value we are essentially comparing two different models. For example, consider the quadratic term for alcohol by volume. When we look at this p-value we are asking ourselves the following:

> *"Is there evidence to suggest that we need this model with a quadratic term for alcohol by volume, rather than the model without the quadratic term?"*

Model comparison

Deviance

We are comparing this model:

```
wine.poly1.fit <- glm(good.quality ~ fixed.acidity +
                      residual.sugar + chlorides +
                      I(chlorides^2) +
                      free.sulfur.dioxide +
                      I(free.sulfur.dioxide^2) +
                      density + pH + sulphates +
                      + I(sulphates^2) + alcohol +
                      I(alcohol^2),
                    family = "binomial", data = wine.df)
```

To this model:

```
wine.poly2.fit <- glm(good.quality ~ fixed.acidity +
                      residual.sugar + chlorides +
                      I(chlorides^2) +
                      free.sulfur.dioxide +
                      I(free.sulfur.dioxide^2) +
                      density + pH + sulphates +
                      + I(sulphates^2) + alcohol,
                    family = "binomial", data = wine.df)
```

Model comparison

Submodels

In this scenario, `wine.poly2.fit` is a called a *submodel* of `wine.poly1.fit`.

If we can set parameter values in Model B at particular values so that we end up with Model A, then Model A is a submodel of Model B. Alternatively, we might say that Model A is nested within Model B. For example:

Model A: $g(\theta_i) = \beta_0 + \beta_1 x_i$
Model B: $g(\theta_i) = \beta_0 + \beta_1 x_i + \beta_2 z_i$

Model A is a submodel of Model B, because if we set $\beta_2 = 0$ then the two are equivalent.

Model comparison

Submodels

In this scenario, `wine.poly2.fit` is a called a *submodel* of `wine.poly1.fit`.

If we can set parameter values in Model B at particular values so that we end up with Model A, then Model A is a submodel of Model B. Alternatively, we might say that Model A is nested within Model B. For example:

Model C: $g(\theta_i) = \beta_0 + \beta_1 a_i$
Model D: $g(\theta_i) = \beta_0 + \beta_1 x_i + \beta_2 z_i$

Model C is **not** a submodel of Model D, because the explanatory variable a_i does not appear in Model D.

Model comparison

Deviance

How do we compare a full model to a submodel to determine which we prefer? One way is to compare the deviances.

From Handout 5: We have evidence for lack-of-fit if a model's deviance is too large. If adding an explanatory term decreases the deviance enough, then we might prefer this full model.

- **We like models with small deviance (high log-likelihood).**

From Handout 4: Adding too many explanatory terms will result in an overfitted, complex model.

- **We like simpler models with fewer parameters.**

Adding explanatory terms will almost always reduce the deviance, which is good, but it will add model complexity, which is bad. We have to somehow resolve this trade-off.

Model comparison

Deviance

Deviance of the model with the quadratic alcohol term:

```
deviance(wine.poly1.fit)

## [1] 847.4557
```

Deviance of the model without the quadratic alcohol term:

```
deviance(wine.poly2.fit)

## [1] 865.6304
```

Difference between the deviances:

```
deviance(wine.poly2.fit) - deviance(wine.poly1.fit)

## [1] 18.17466
```

Model comparison

Deviance

Is a 18.17 improvement in the deviance worth the extra parameter?

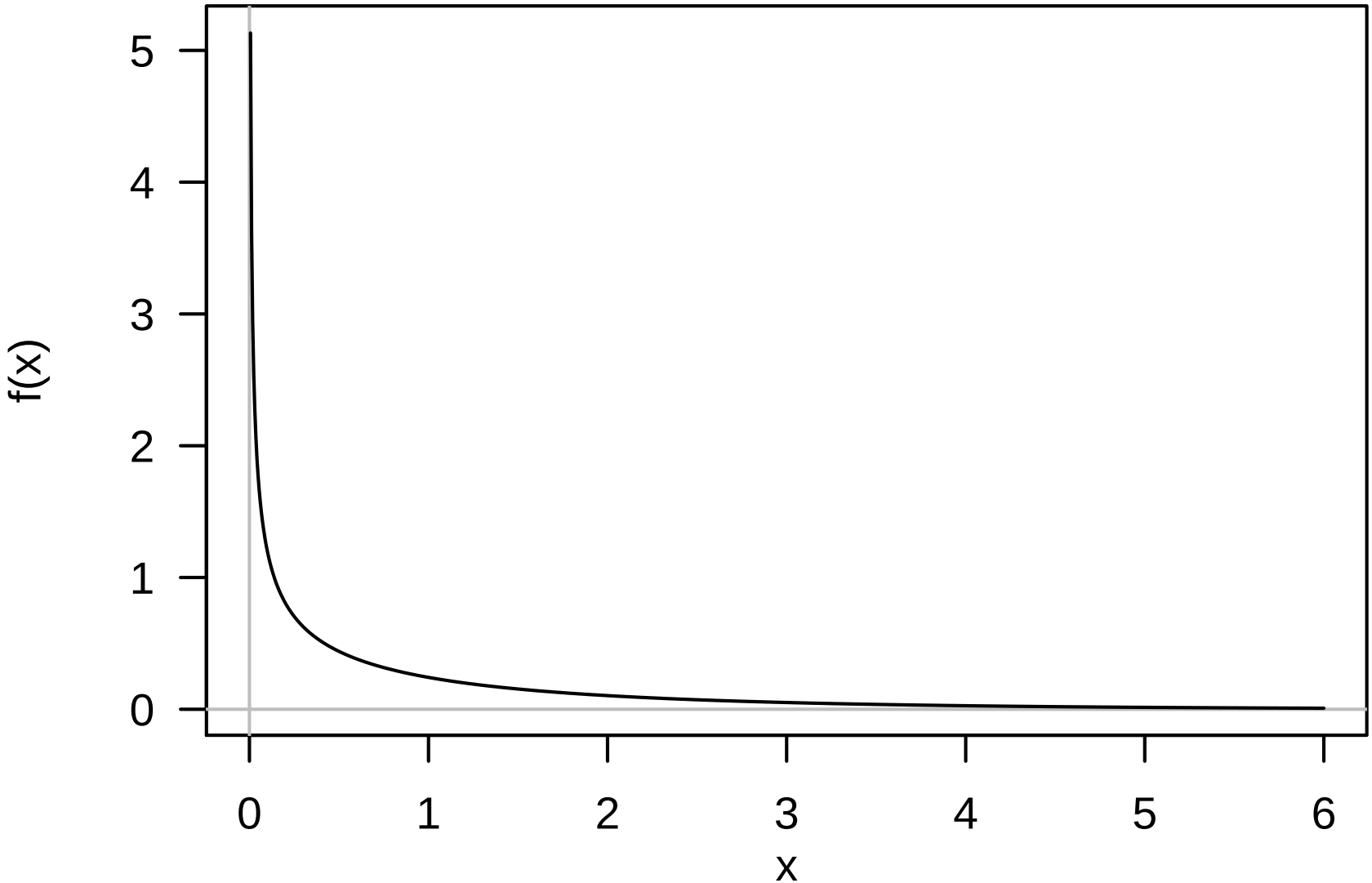
If the submodel is correct and the extra parameters in the full model are not required, then the difference between the deviances comes from a distribution that can be approximated by a chi-squared distribution with $k_2 - k_1$ parameters, where k_1 is the number of parameters in the submodel and k_2 is the number of parameters in the full model. Therefore $k_2 - k_1$ is the number of extra parameters in the full model.

In this case, if the submodel `wine.poly2.fit` is correct and we do not need the extra quadratic term in `wine.poly1.fit`, then the difference in the deviances can be approximated by a chi-squared distribution with 1 degree of freedom.

Model comparison

Deviance

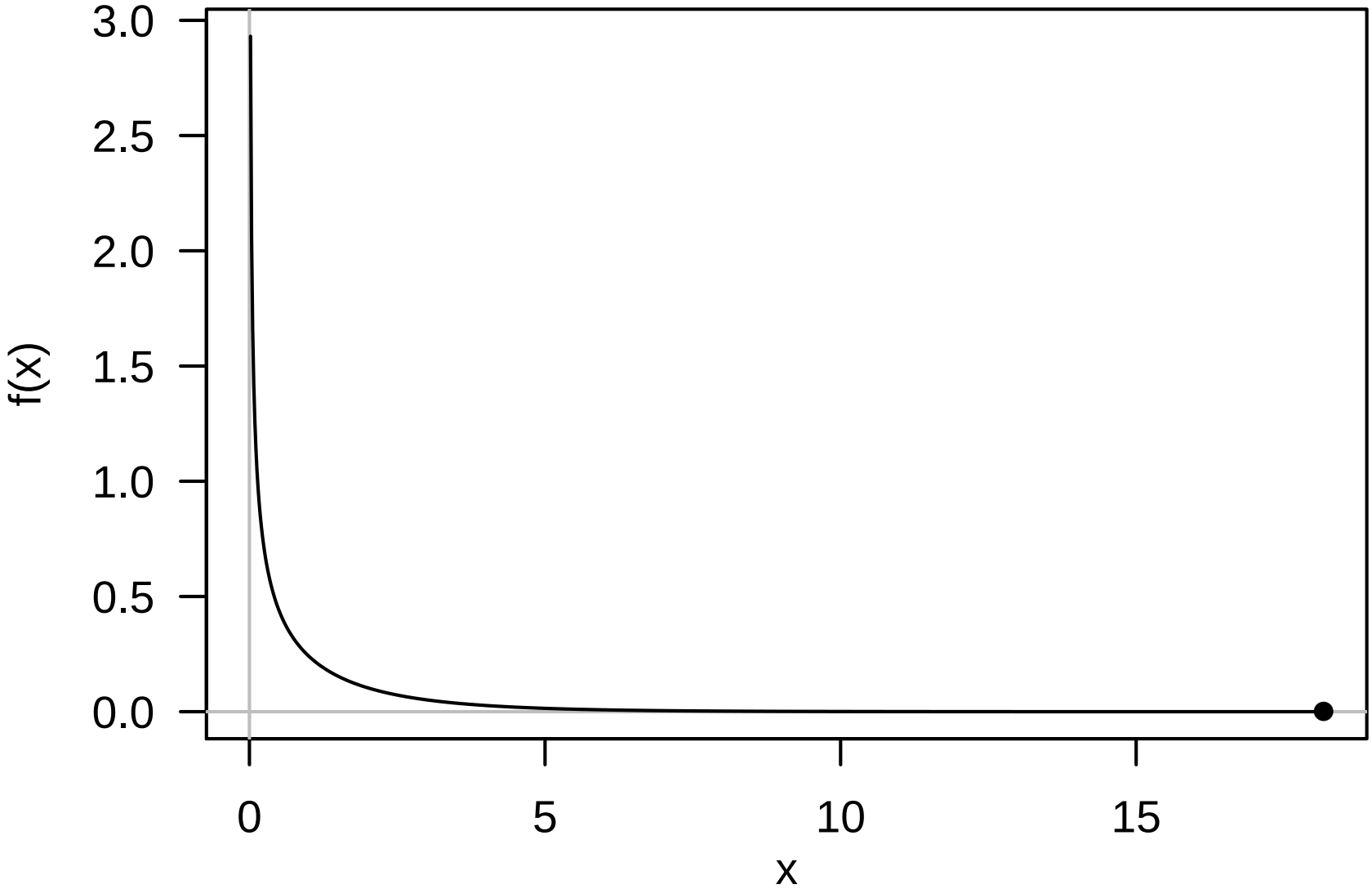
Chi-squared distribution with 1 degree of freedom:



Model comparison

Deviance

Chi-squared distribution with 1 degree of freedom:



Model comparison

Deviance

In this case, it is not plausible that the difference in deviances comes from a chi-squared distribution with 1 degree of freedom, so we have evidence to suggest we need the quadratic term for alcohol by volume.

Formally, we can compute a p-value to test the following null and alternative hypotheses:

- H_0: The submodel is appropriate. We don't need the extra parameter(s) in the full model.
- H_1: The submodel is not appropriate. We need at least one of the additional parameters in the full model.

```
1 - pchisq(18.17, 1)
```

```
## [1] 2.01543e-05
```

We have strong evidence against the null hypothesis.

Model comparison

Deviance

In this case, it is not plausible that the difference in deviances comes from a chi-squared distribution with 1 degree of freedom, so we have evidence to suggest we need the quadratic term for alcohol by volume.

Formally, we can compute a p-value to test the following null and alternative hypotheses:

- H_0: The submodel is appropriate. There is no quadratic effect of alcohol by volume.
- H_1: The submodel is not appropriate. There is a quadratic effect of alcohol by volume.

```
1 - pchisq(18.17, 1)
```

```
## [1] 2.01543e-05
```

We have strong evidence against the null hypothesis.

Model comparison

Deviance: Doing it in R

We can compare a submodel to a full model using the `anova()` function:

```
anova(wine.poly2.fit, wine.poly1.fit, test = "Chisq")

## Analysis of Deviance Table
##
## Model 1: good.quality ~ fixed.acidity + residual.sugar + chlorides + I(chlor
##     free.sulfur.dioxide + I(free.sulfur.dioxide^2) + density +
##     pH + sulphates + +I(sulphates^2) + alcohol
## Model 2: good.quality ~ fixed.acidity + residual.sugar + chlorides + I(chlor
##     free.sulfur.dioxide + I(free.sulfur.dioxide^2) + density +
##     pH + sulphates + +I(sulphates^2) + alcohol + I(alcohol^2)
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1      1587     865.63
## 2      1586     847.46  1   18.175 2.015e-05 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

This is known as a likelihood-ratio test.

Model comparison

Deviance

In this case, the null hypothesis we have tested is the same as that tested by the p-value for the quadratic effect in the `summary()` table. You will notice that the two p-values are different, but very similar. This is because they use slightly different methods.

So why use the `anova()` method? One reason is because it allows us to test *composite* null hypotheses, where a submodel may differ from the full model by more than one parameter.

For example, we can test the following hypotheses by comparing our initial submodel without polynomials with our full model with all polynomials:

- H_0: The submodel is appropriate. We don't need any quadratics in our model.
- H_1: The submodel is not appropriate. We need at least one of the quadratic terms in our model.

Model comparison

Deviance

```
anova(wine.fit, wine.poly1.fit, test = "Chisq")

## Analysis of Deviance Table
##
## Model 1: good.quality ~ fixed.acidity + residual.sugar + chlorides + free.su
##     density + pH + sulphates + alcohol
## Model 2: good.quality ~ fixed.acidity + residual.sugar + chlorides + I(chlor
##     free.sulfur.dioxide + I(free.sulfur.dioxide^2) + density +
##     pH + sulphates + +I(sulphates^2) + alcohol + I(alcohol^2)
##   Resid. Df Resid. Dev Df Deviance  Pr(>Chi)
## 1      1590     909.11
## 2      1586     847.46  4   61.652 1.304e-12 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

We have very strong evidence to suggest that we need at least one of the quadratic terms in the full model.

Model comparison

Deviance: Some problems

The approach we have used here only allows us to compare between two models at once. Also, we can only compare models if one is the submodel of another. For example, we cannot use this method to decide whether we'd rather include variable x_i or variable z_i by comparing the following models:

Model A: $g(\theta_i) = \beta_0 + \beta_1 x_i$
Model B: $g(\theta_i) = \beta_0 + \beta_1 z_i$

This is because neither model is a submodel of the other.

Also, strictly speading, p-values are designed to test hypotheses of particular interest that we formulated prior to collecting data and fitting models. Although they can give us an indication, they are not designed to help us decide which model to fit.

Model comparison

Information-theoretic approaches

A popular way of comparing models is to use a so-called 'information-theoretic' approach. For a given model we calculate a value that trades off model fit with complexity.

Unlike comparing deviances using the likelihood-ratio tests carried out via `anova()`, **we can use information-theoretic approaches to compare models, even when one is not a submodel of the other.**

Also, we can compare many models all in one go. We don't have to consider a single pairwise comparison between a model and a submodel.

Model comparison

Information-theoretic approaches: AIC

We want to find a model with low deviance—equivalently, a high log-likelihood—but also a small number of parameters. A common way to do resolve this trade-off is to calculate the 'Akaike Information Criterion':

$$\text{AIC} = -2\ell + 2k$$

Where ℓ is the model's log-likelihood, and k is the number of parameters.

The smaller the better. If a model has a high log-likelihood (a good thing), then the first term will be small. If a model is simple (another good thing), the second term will also be small.

Model comparison

Information-theoretic approaches: AIC

$$\text{AIC} = -2\ell + 2k$$

You can think of the first term as measuring model fit: how close are the model's fitted values to the observations?

The second term penalises model complexity, by adding a penalty of 2 for every parameter. Adding an extra parameter is only worthwhile if it decreases the first term by more than this penalty.

The penalty of 2 may sound like an arbitrarily chosen number, but actually has justification from statistical theory.

Model comparison

Information-theoretic approaches: AICc

Calculating the AIC is easily done in R:

```
AIC(wine.fit, wine.poly1.fit, wine.poly2.fit)

##                  df      AIC
## wine.fit          9 927.1078
## wine.poly1.fit   13 873.4557
## wine.poly2.fit   12 889.6304
```

The model with all the polynomial terms has the smallest value, and is therefore best-supported by AIC out of those we have fitted.

Unlike model selection using p-values, we can compare two models even if one is not nested within the other.

Model comparison

Information-theoretic approaches: AICc

The theoretical justification for a penalty of 2 is based on an asymptotic approximation that only holds if the number of observations, n is large.

AICc ('corrected AIC') is a version of AIC that includes an additional term to 'correct' the penalty when we do not have large samples:

$$\text{AICc} = -2\ell + \left(2k + \frac{2k^2 + 2k}{n - k - 1}\right)$$

When sample sizes are small, we should probably use AICc rather than AIC. When sample sizes are large, then the additional term is almost zero, so AIC and AICc are virtually equivalent.

Model comparison

Information-theoretic approaches: AICc

Calculating the AICc is also easily done in R, although you need to load the `MuMIn` package first:

```
library(MuMIn)
AICc(wine.fit, wine.poly1.fit, wine.poly2.fit)

##                  df     AICc
## wine.fit          9 927.2211
## wine.poly1.fit   13 873.6854
## wine.poly2.fit   12 889.8271
```

The model with all the polynomial terms has the smallest value, and is therefore best-supported by AICc out of those we have fitted. We have a large sample here, so it doesn't matter whether we use AIC or AICc.

Model comparison

Information-theoretic approaches: AIC

The difference in AIC/AICc between two models tells you how much support one model has over the other. Below is a rule of thumb for comparing AIC/AICc models:

- If the difference is less than 2, then both models are similarly supported by the data.
- If the difference is between 2 and 4, the one with the smaller AIC is slightly better supported.
- If the difference is between 4 and 10, the one with the smaller AIC is considerably better supported.
- If the difference is over 10, the one with the larger AIC has essentially no support.

Model comparison

Information-theoretic approaches: BIC

Another penalty term was derived using a different theoretical argument, resulting in BIC, 'Bayesian Information Criterion':

$$\text{BIC} = -2\ell + \log(n)k$$

So the penalty for each parameter is $\log(n)$, rather than 2, where n is the number of observations. As long as $n > 8$, BIC penalises additional parameters more harshly than AIC or AICc, so BIC tends to favour simpler models.

Model comparison

Information-theoretic approaches: BIC

Calculating the BIC is also easily done in R:

```
BIC(wine.fit, wine.poly1.fit, wine.poly2.fit)

##                   df      BIC
## wine.fit           9 975.5020
## wine.poly1.fit    13 943.3585
## wine.poly2.fit    12 954.1560
```

Here we have $n = 1599$, so each parameter comes with a penalty of $\log(1599) = 7.38$. Nevertheless, BIC still favours the most complicated model with all of the polynomial terms.

Model comparison

Information-theoretic approaches: AIC(c) vs BIC

AIC(c) and BIC are both justified by statistical theory, yet they may select different models. Why the difference? The answer is that they are both trying to answer slightly different questions:

AIC:

- Out of our candidate models, which is most similar to the 'true model'?

BIC:

- Out of our candidate models, which is most likely to be the 'true model'?

In other words, AIC(c) acknowledges that we probably haven't found the 'true model', and we're just trying to find the closest approximation. On the other hand, BIC assumes that one of the candidates is the 'true' one, and attempts to determine which it is.

Model comparison

Information-theoretic approaches: A warning

AIC, AICc, and BIC are *relative* measures. They allow us to compare models, but they do not offer a judgement about whether or not the models are any good. The model with the lowest value is not necessarily a good model, it might just be the least-worst model out of the candidates.

We can only compare models that are fitted to the same response variable. We can't use these criteria to compare models fitted to different data sets. We can't even use them to compare models before and after a transformation of the response variable—because this is effectively a different response.

Model search strategies

All possible regressions

Now we know how to compare models. But how do we choose our candidate set of models?

One method is to fit *every single model you can think of*. This is the 'all possible regressions' approach to model selection, and is implemented in the `dredge` function in the `MuMIn` package. You need to change a global option in R for this to work, see below.

We input a full model with all the terms we wish to consider, and the function will fit all possible submodels. The code below considers all possible combinations of linear terms, but not quadratics, because the full model we have provided does not have any quadratic terms:

```
library(MuMIn)
options(na.action = "na.fail")
all.fits <- dredge(wine.fit)
```

Model search strategies

All possible regressions: `dredge()`

Running `head(all.fits)` shows the best six models, with ranking using AICc by default:

```
> head(all.fits)
Global model call: glm(formula = good.quality ~ fixed.acidity + residual.sugar +
    chlorides + free.sulfur.dioxide + density + pH + sulphates +
    alcohol, family = "binomial", data = wine.df)
---
Model selection table
    (Int)    alc    chl    dns fxd.acd fre.slf.dxd     pH rsd.sgr   slp df   logLik  AICc delta weight
224 337.3 0.7789 -9.665 -354.9  0.4531    -0.01895         0.2161 3.873  8 -454.966 926.0  0.00  0.492
256 386.6 0.7235 -9.095 -407.4  0.5345    -0.01897 0.8498  0.2410 3.979  9 -454.554 927.2  1.20  0.270
208 328.3 0.8020 -9.315 -346.4  0.4706                     0.2040 3.711  7 -457.262 928.6  2.57  0.136
240 374.8 0.7499 -8.777 -396.1  0.5479             0.8120  0.2273 3.812  8 -456.881 929.9  3.83  0.073
160 223.2 0.8947 -9.596 -240.5  0.3963    -0.01722                 3.674  7 -459.147 932.4  6.34  0.021
144 221.4 0.9078 -9.217 -239.3  0.4149                          3.533  6 -461.047 934.1  8.12  0.008
Models ranked by AICc(x)
```

From the models we've tried, the one with all explanatory variables except pH has the smallest AICc.

Model search strategies

All possible regressions: `dredge()`

To get the model object for the best model we can use the `get.models()` function:

```
first.model <- get.models(all.fits, 1)[[1]]
summary(first.model)


## Call:
## glm(formula = good.quality ~ alcohol + chlorides + density +
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          3.373e+02  8.436e+01   3.998 6.38e-05 ***
## alcohol              7.789e-01  1.077e-01   7.232 4.75e-13 ***
## chlorides           -9.665e+00  3.316e+00  -2.915  0.00356 **
## density             -3.549e+02  8.461e+01  -4.195 2.73e-05 ***
## fixed.acidity        4.531e-01  7.225e-02   6.272 3.57e-10 ***
## free.sulfur.dioxide -1.895e-02  9.022e-03  -2.100  0.03571 *
## residual.sugar       2.161e-01  6.968e-02   3.101  0.00193 **
## sulphates            3.873e+00  5.071e-01   7.638 2.20e-14 ***
## ---
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1269.92  on 1598  degrees of freedom
## Residual deviance:  909.93  on 1591  degrees of freedom
```

Model search strategies

All possible regressions: `dredge()`

For the second-best model:

```
second.model <- get.models(all.fits, 2)[[1]]
summary(second.model)


## Call:
## glm(formula = good.quality ~ alcohol + chlorides + density +
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          3.866e+02  1.004e+02   3.851 0.000118 ***
## alcohol              7.235e-01  1.235e-01   5.857 4.72e-09 ***
## chlorides           -9.095e+00  3.339e+00  -2.724 0.006454 **
## density             -4.074e+02  1.026e+02  -3.972 7.12e-05 ***
## fixed.acidity        5.345e-01  1.150e-01   4.649 3.33e-06 ***
## free.sulfur.dioxide -1.897e-02  8.976e-03  -2.114 0.034529 *
## pH                   8.498e-01  9.330e-01   0.911 0.362426
## residual.sugar       2.410e-01  7.444e-02   3.238 0.001205 **
## sulphates            3.979e+00  5.189e-01   7.669 1.73e-14 ***
## ---
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1269.92  on 1598  degrees of freedom
## Residual deviance:  909.11  on 1590  degrees of freedom
```

Model search strategies

Stepwise: Backwards selection

An alternative strategy is as follows:

1. Choose your model selection criterion (e.g., AICc).
2. Fit a full model and calculate the criterion
3. Consider all possible models with one fewer parameter, calculating the criterion for each.
4. If the full model has the best criterion value, select it as your final model. Otherwise, move to the submodel with the best criterion value, and repeat steps 3 and 4.

This is known as 'backward selection', and is essentially the strategy you used in STATS 20X: you fitted the most complicated model, and then dropped one variable at a time until you were happy. The only difference is that you decided to drop variables using p-values, not an information criterion.

Model search strategies

Stepwise: Forwards selection

We can also move in the opposite direction:

1. Choose your model selection criterion (e.g., AICc).
2. Fit the null model.
3. Consider all possible models with one additional parameter, calculating the criterion for each.
4. If the null model has the best criterion value, select it as your final model. Otherwise, move to the model with the best criterion value, and repeat steps 3–5.

This is known as 'forwards selection'. We can also use a strategy that uses ideas from both; at each step, it considers both adding a single term and removing a single term.

Model search strategies

Stepwise: Doing it in R

The `step()` function in R implements stepwise selction. You use it similarly to `dredge()`: fit a full model, specify it as the main argument, and choose your stepping direction:

```
step(wine.fit, direction = "backward")
```

```
step(wine.fit, direction = "forward")
```

```
step(wine.fit, direction = "both")
```

By default, AIC is used as the criterion.

Model search strategies

Stepwise: Doing it in R

The resulting object from the function is the model object for the best model the stepwise selection approach found:

```
best.backwards.model <- step(wine.fit, direction = "backward")
summary(best.backwards.model)
```

```
## Call:
## glm(formula = good.quality ~ fixed.acidity + residual.sugar +
##
## Coefficients:
##                       Estimate Std. Error z value Pr(>|z|)
## (Intercept)          3.373e+02  8.436e+01   3.998 6.38e-05 ***
## fixed.acidity        4.531e-01  7.225e-02   6.272 3.57e-10 ***
## residual.sugar       2.161e-01  6.968e-02   3.101  0.00193 **
## chlorides           -9.665e+00  3.316e+00  -2.915  0.00356 **
## free.sulfur.dioxide -1.895e-02  9.022e-03  -2.100  0.03571 *
## density             -3.549e+02  8.461e+01  -4.195 2.73e-05 ***
## sulphates            3.873e+00  5.071e-01   7.638 2.20e-14 ***
## alcohol              7.789e-01  1.077e-01   7.232 4.75e-13 ***
## ---
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 1269.92  on 1598  degrees of freedom
## Residual deviance:  909.93  on 1591  degrees of freedom
```

Model search strategies

All possible regressions or stepwise?

Should you use an all possible regressions approach or a stepwise approach?

Sometimes there are literally millions of possible models to fit, and it may take a very long time for your computer to fit all of them. You can fit a single GLM in the blink of an eye, but it would take you a long time to blink one million times!

The advantage of a stepwise approach is that you don't have to fit every single model.

The disadvantage of a stepwise approach is that you haven't fitted every single model: you might have missed the best one!

Summary

To summarise, in this handout we have discussed tools to select possible quadratic terms and to compare different models.

The model search strategies are useful for guiding you towards a useful model, but never forget that the information criterion are not a total replacement for commonsense. Some models may be well supported by AICc, for example, yet include effects that we know don't exist, or exclude effects that we know are important.

It can be dangerous to allow a computer to do **all** the thinking for you...