Week 5 Cross-Validation and Bootstrap

Dr. Liwan Liyanage

School of Computing, Engineering and Mathematics

WESTERN SYDNEY
UNIVERSITY

Resampling and Error Estimation

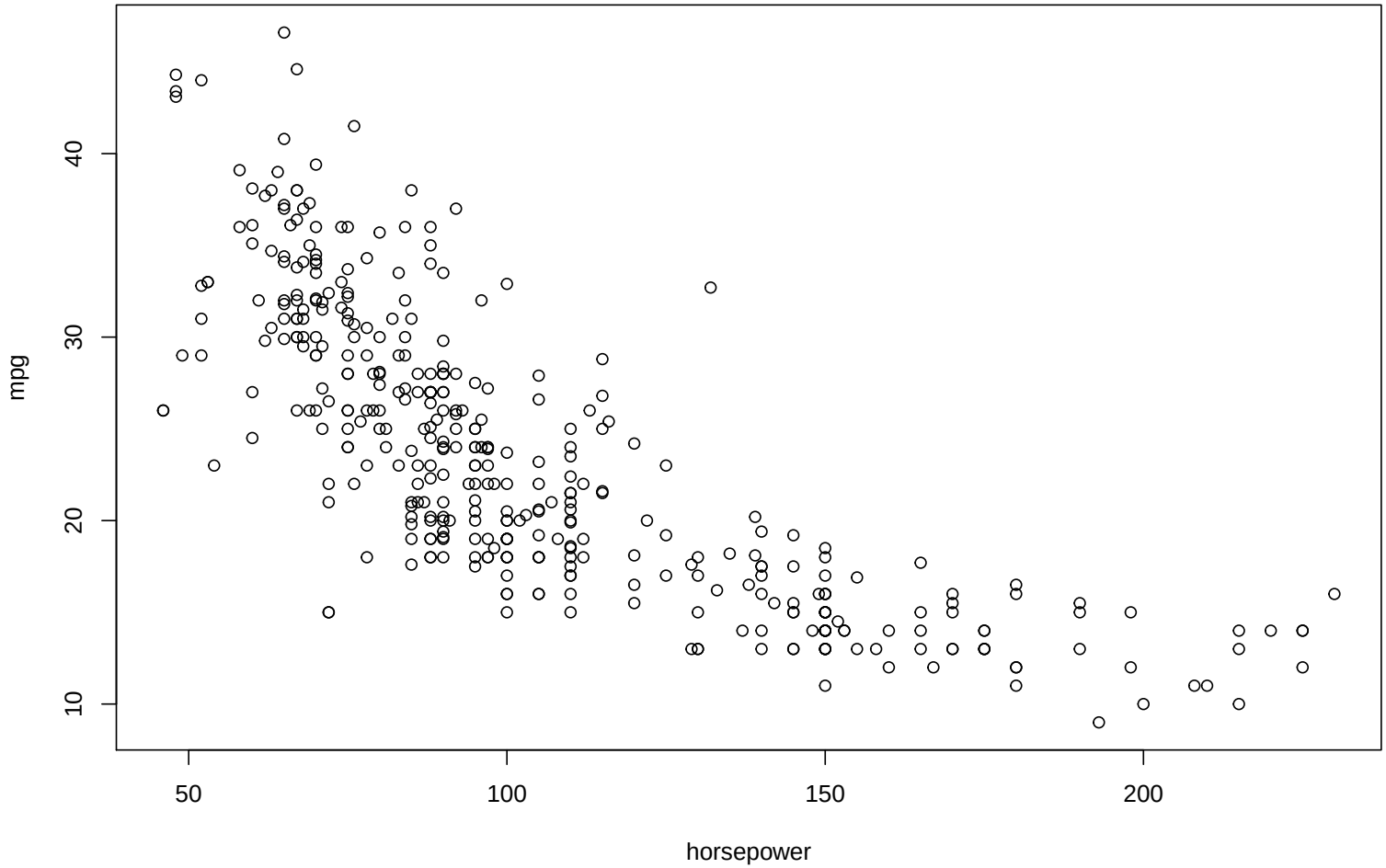
This week we will look at estimating the error in supervised models

- The validation Set Approach
- Leave-One-Out Cross-Validation
- k-Fold Cross-Validation
- The Bootstrap

WESTERN SYDNEY UNIVERSITY

Errors in Supervised models

Remember the *Auto* data from ***linear regression***



WESTERN SYDNEY UNIVERSITY

Auto data

Suppose we want to compare the fit of a **simple linear regression** $a + bx$ to a **quadratic fit** $a + bx + cx^2$.

We already know how to do this using the ***t-statistic*** and its ***p-value***.

–But this relies on **assumptions** - Normality etc.

```
model1 = lm(mpg~poly(horsepower, 2))
summary(model1)
```

```
##
## Call:
## lm(formula = mpg ~ poly(horsepower, 2))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.7135  -2.5943  -0.0859   2.2868  15.8961
##
```

WESTERN SYDNEY UNIVERSITY

Auto data Output

```
##
## Call:
## lm(formula = mpg ~ poly(horsepower, 2))
##
## Residuals:
##      Min       1Q   Median       3Q      Max
## -14.7135  -2.5943  -0.0859   2.2868  15.8961
##
## Coefficients:
##                             Estimate Std. Error t value Pr(>|t|)
## (Intercept)                  23.4459     0.2209  106.13   <2e-16
## poly(horsepower, 2)1 -120.1377     4.3739  -27.47   <2e-16
## poly(horsepower, 2)2   44.0895     4.3739   10.08   <2e-16
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 '
##
```

Auto data(Continues. . .)

Another way would be to compare the **error** of a simple linear and a quadratic fit.

What is Error?

- For a ***regression problem*** the most obvious answer is the ***Residual Sum of Squares (RSS)*** or ***Mean Square Error (MSE)***

$$RSS = \sum_{i=1}^{n} (y_i - \hat{y}_i)^2$$

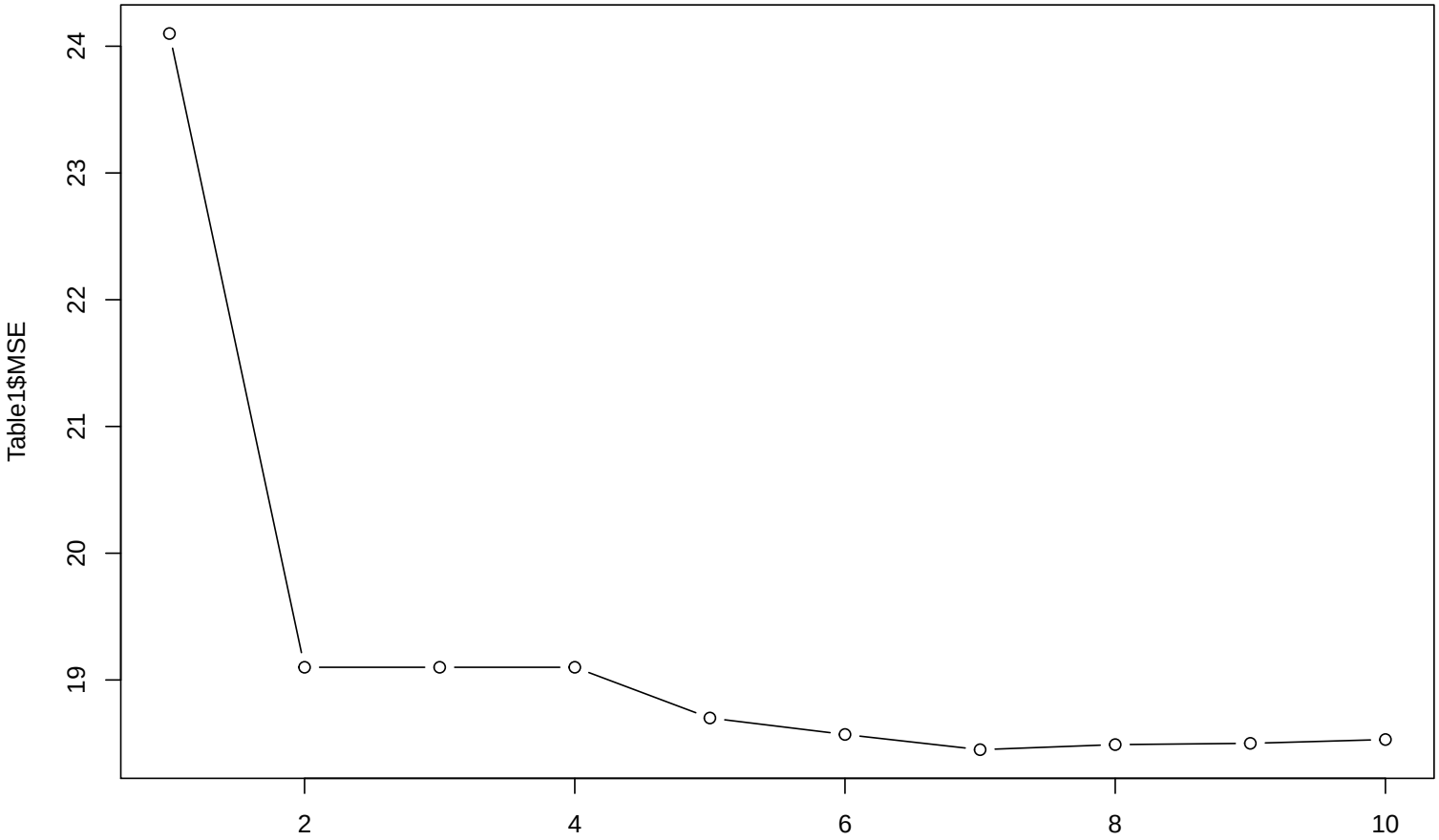
$$MSE = RSS/n$$

where $\hat{y}_i$ is the value from the model for each observed y_i.

WESTERN SYDNEY UNIVERSITY

Auto Data (Continued...)

There is a problem though. The MSE evaluated on the data that the model was fit to ***always decrease*** for more complex models.



WESTERN SYDNEY UNIVERSITY

Validation set Approach

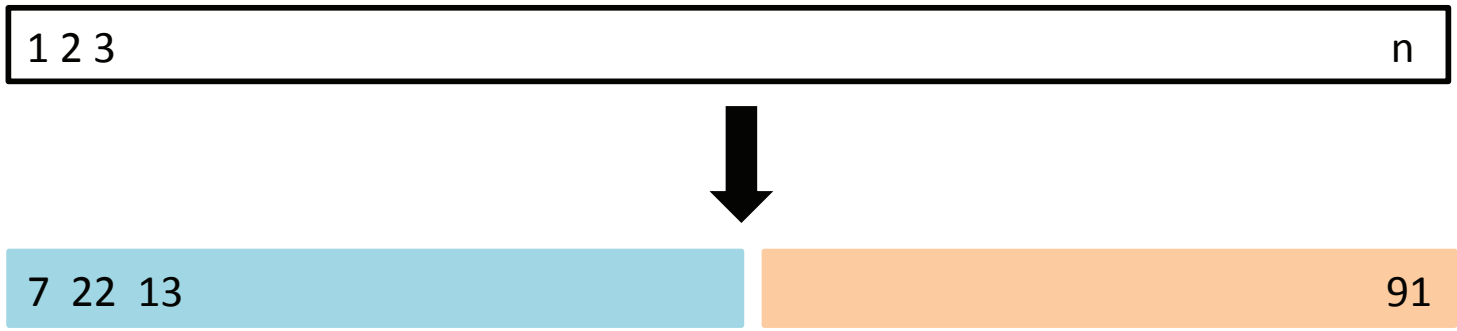
One approach is to ***hold out a Validation*** data set, i.e. split the data into two groups and build the model using one group (***training set***) and estimate the error (MSE) using the second group (***test or validation set***).

We can split the data into halves (at random), or two-thirds/one-third, or some other way.

We need enough training data that a good model can built, and enough test data that a good estimate of *error* can be found.

WESTERN SYDNEY UNIVERSITY

Validation Set Approach (Continued...)



Source: An introduction to Statistical Learning with applications in R

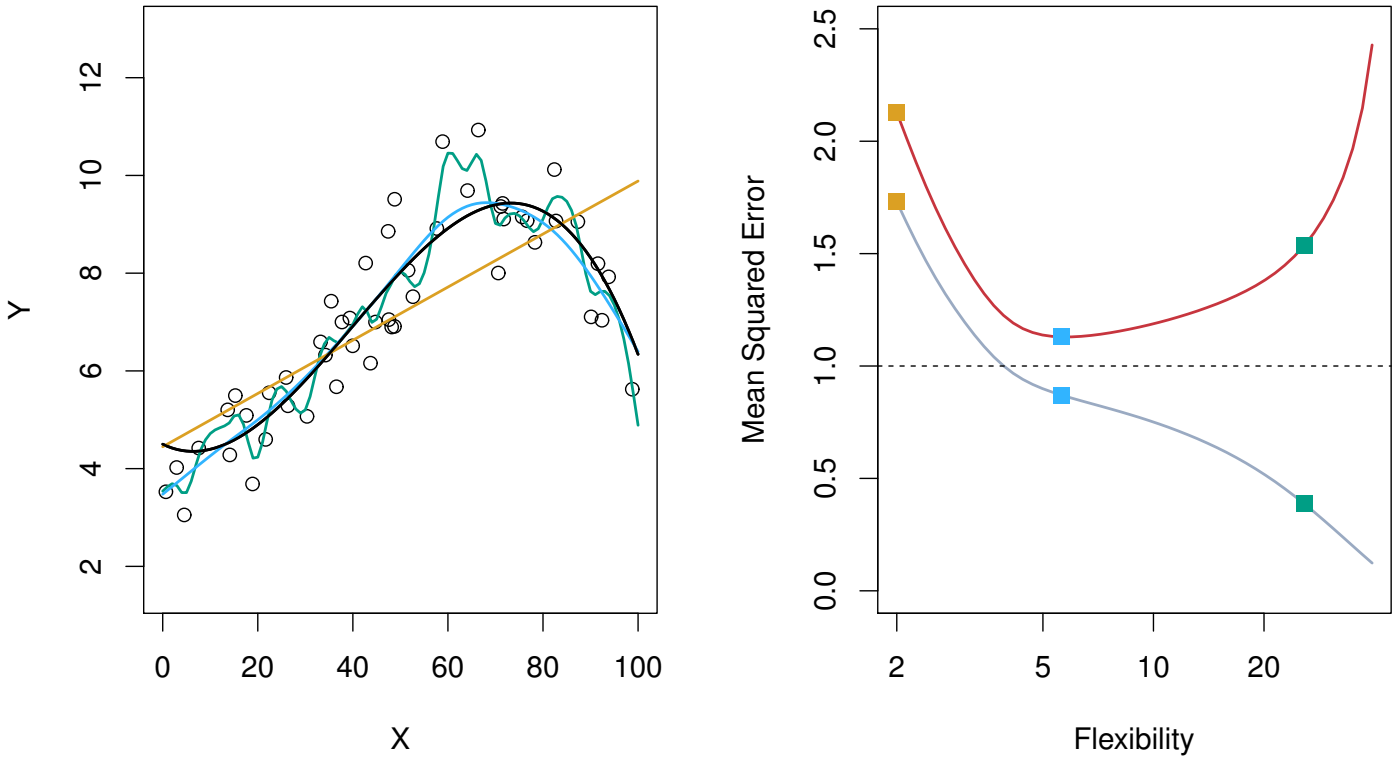
WESTERN SYDNEY UNIVERSITY

Training and Validation Error

- The MSE evaluated using the same data that the model was fit to is called the ***Training Error***
- The more complex the method, the smaller the **Training Error**.
- We need an error that is **more representative** of the error that might be seem on future data.
- We call this ***Validation Error***.

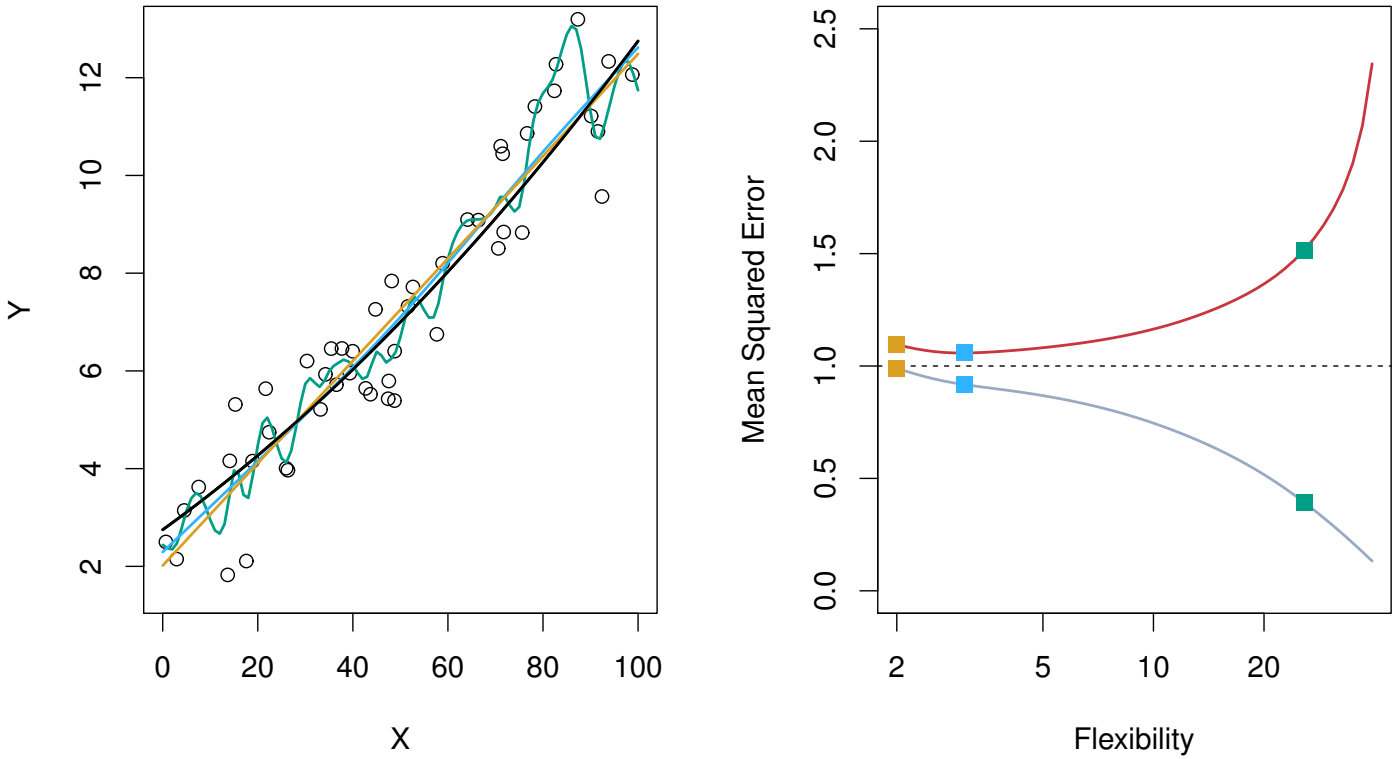
WESTERN SYDNEY UNIVERSITY

Training vs. Testing Error - Example 1



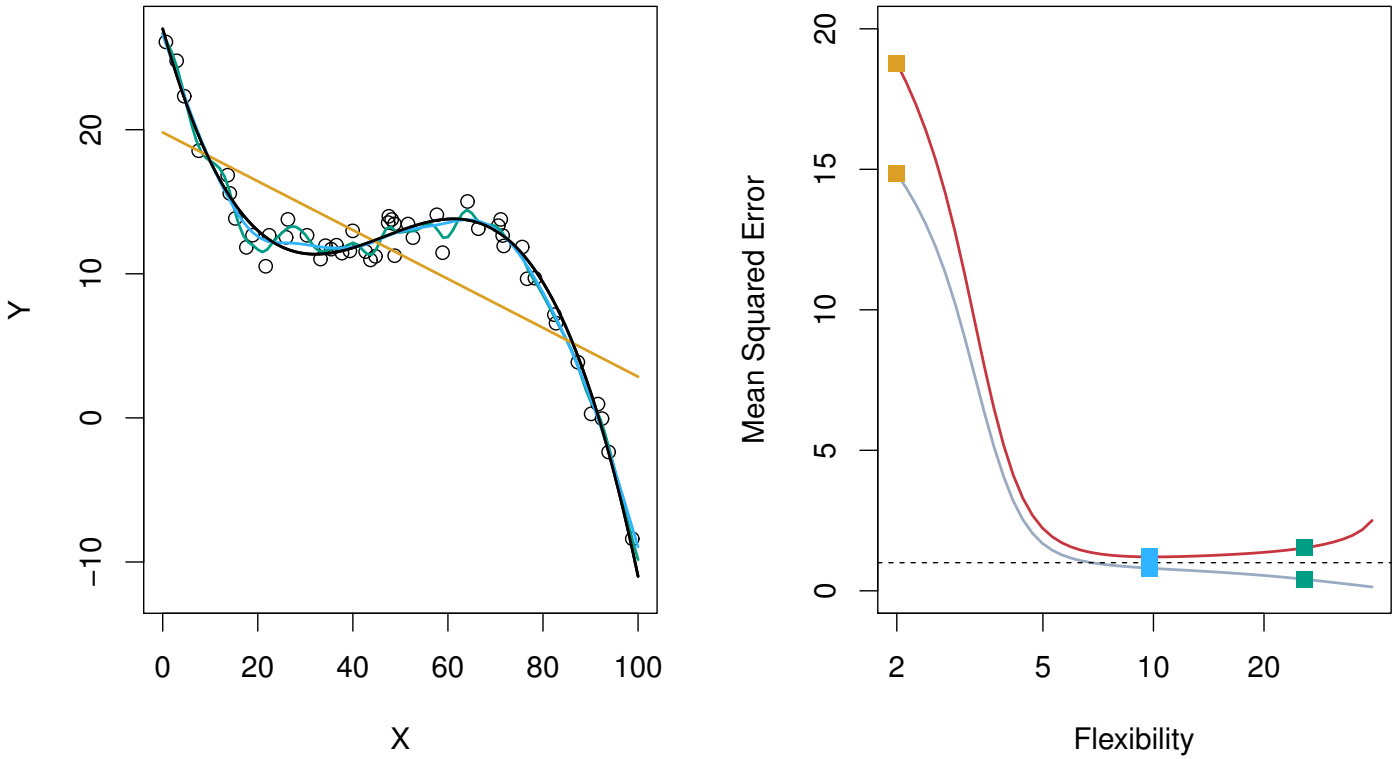
Source: An introduction to Statistical Learning with applications in R

Training vs. Testing Error - Example 2



Source: An introduction to Statistical Learning with applications in R

Training vs. Testing Error - Example 3



Source: An introduction to Statistical Learning with applications in R

Validation data set (Continued...)

We randomly split the data into halves;

```
set.seed(1)
tr = sample(1:392,196)
train=Auto[tr , ]
dim(train)
```

```
## [1] 196   9
```

WESTERN SYDNEY UNIVERSITY

Validation data set (Continued...)

We then fit a polynomial regression using only the observations corresponding to the training data set.

```
model2 = lm(mpg~poly(horsepower, 2), data=Auto, subset = tr)
#we select only the training set using "subset"
coef(model2)
```

```
##          (Intercept) poly(horsepower, 2)1 poly(horsepower,
##              23.54955            -123.58813                47.71
```

```
# calculating MSE
EMS2=mean((mpg -predict (model2 ,Auto))[-tr ]^2)
EMS2
```

```
## [1] 18.71646
```

WESTERN SYDNEY UNIVERSITY

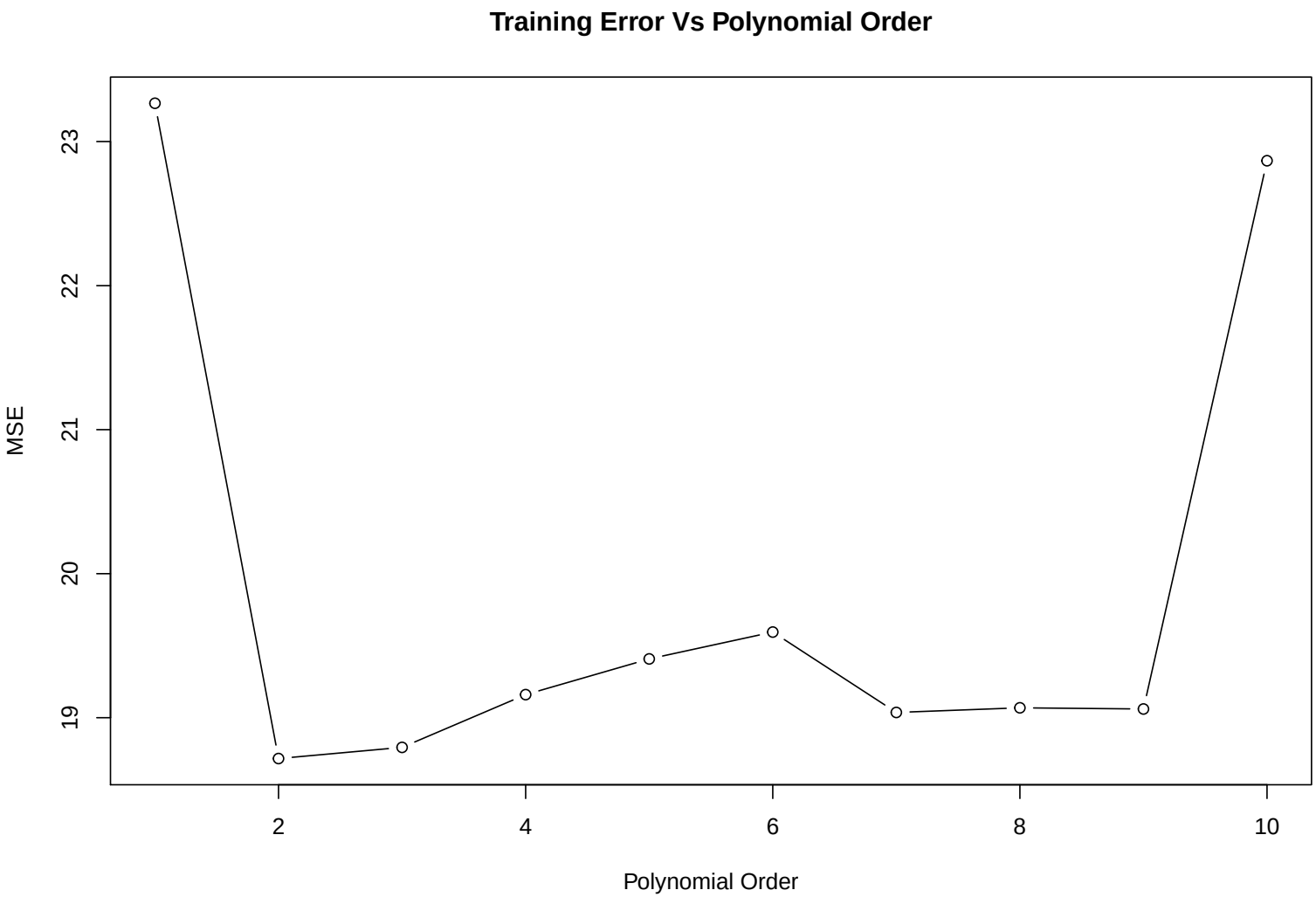
Calculating EMSs for Polynomial Regression Models of Orders p = 1,...10 using Validation Data Set

Random 50/50 split Train/Validation

```
poly_order = c(1:10)
MSE2 = rep(0, 10)
Table2 = data.frame(poly_order, MSE2)

m1 = lm(mpg~poly(horsepower, 1), data=Auto, subset = tr)
Table2[1,2] = mean((mpg -predict (m1 ,Auto))[-tr ]^2)

m2 = lm(mpg~poly(horsepower, 2), data=Auto, subset = tr)
Table2[2,2] = mean((mpg -predict (m2 ,Auto))[-tr ]^2)

m3 = lm(mpg~poly(horsepower, 3), data=Auto, subset = tr)
Table2[3,2] = mean((mpg -predict (m3 ,Auto))[-tr ]^2)
```

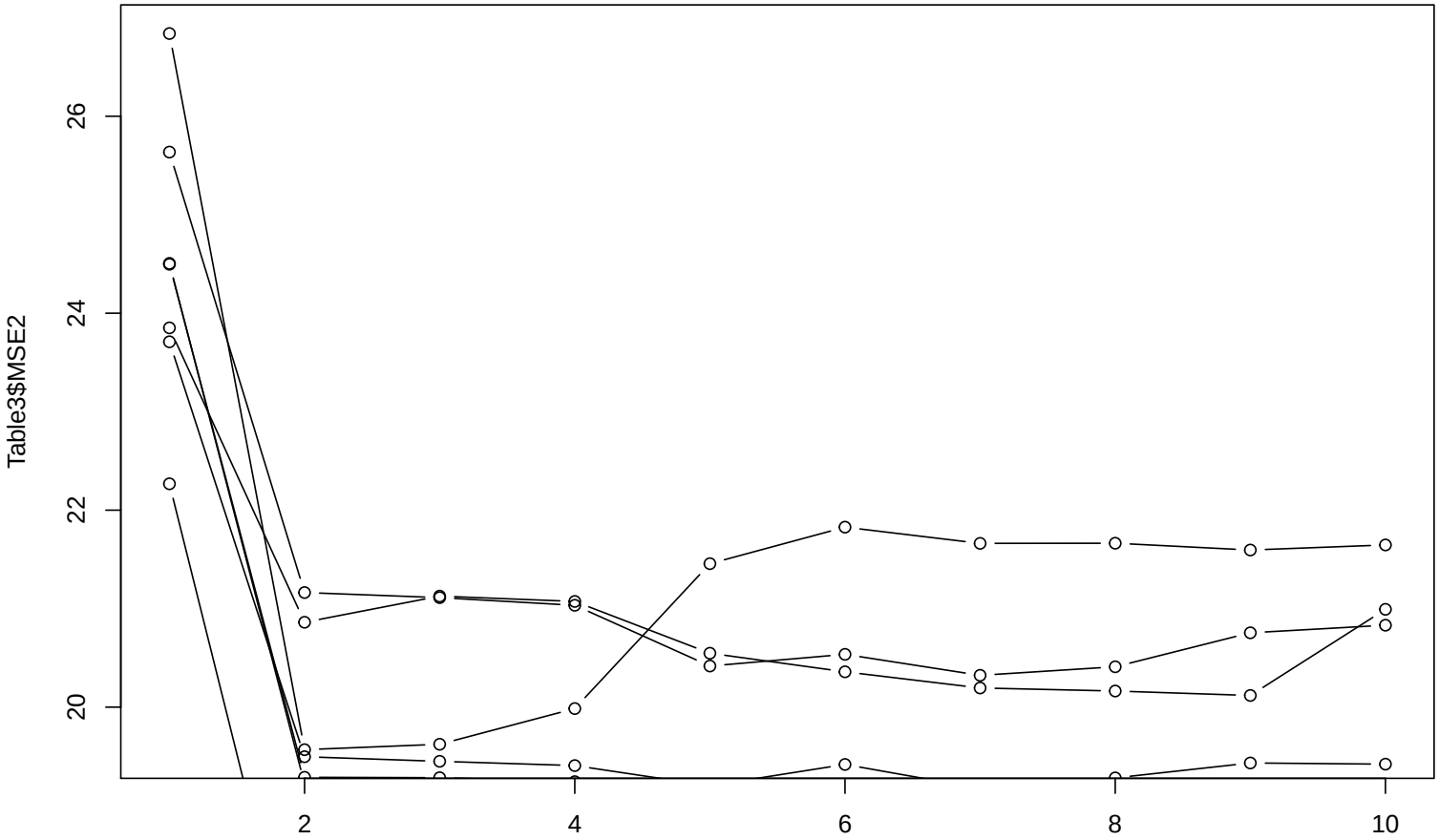
Plot of EMS Versers Order p

```
plot(poly_order, Table2$MSE2, type='b',xlab = "Polynomial Order", ylab = "M
```



Function of Validation Data Set (Continued...)

The problem is it depends on the split... Following plot demonstrate how different splits vary the **Model Error**



Leave One Out Cross Validation

A related approach is to not split the data 50/50 or two-thirds/one-thirds but to simply remove ***one observation***...

Then the ***Training*** data will be nearly all the original data (except one point).

However, the ***Validation*** data will be only one point, and so the MSE estimate will be poor

For Leave-One-Out-Cross-Validation we repeat the above, leaving out all the observations in turn. The MSE is then estimated from all the errors aggregated.

WESTERN SYDNEY UNIVERSITY

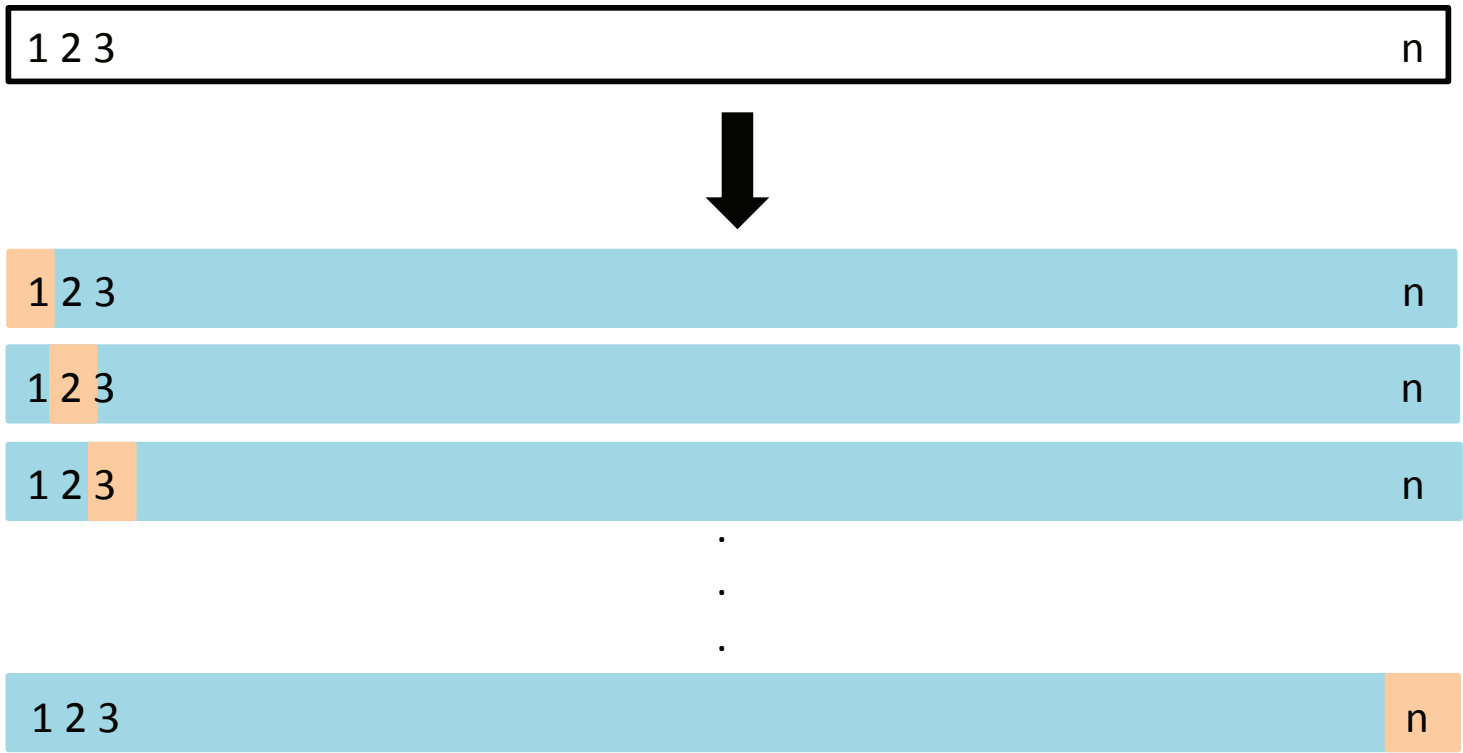
Leave One Out Cross Validation (LOOCV)

Procedure:

1. Leave-out one observation
2. Fit model using the remainder
3. Predict the left out observation and measure *error*
4. Repeat 1-3 leaving out each observation in turn

WESTERN SYDNEY UNIVERSITY

Leave One Out Cross Validation (LOOCV) (Continued...)



Source: An introduction to Statistical Learning with applications in R

Auto data - LOOCV

```
library (boot)
model3=glm(mpg~poly(horsepower, 2) ,data=Auto)
coef(model3)
```

```
##           (Intercept) poly(horsepower, 2)1 poly(horsepower, 2)2
##              23.44592            -120.13774              44.08953
```

```
lm_fit=lm(mpg~poly(horsepower, 2) ,data=Auto)
coef(lm_fit)
```

```
##           (Intercept) poly(horsepower, 2)1 poly(horsepower, 2)2
##              23.44592            -120.13774              44.08953
```

We will perform linear regression using *glm()* function rather than the *lm()* function because the latter can be used together with *cv.glm()*. **cv** stands for **Cross Validation**

Auto data - LOOCV (Continued...)

```
cv_err =cv.glm(Auto ,model3)
cv_err$delta
```

```
## [1] 19.24821 19.24787
```

The **cv.glm()** function produces a list with several components. The two numbers in the delta vector contain the ***cross-validation results***. In this case the numbers are identical (up to two decimal places) and correspond to the ***LOOCV statistic***.

Note that the formula to calculate the above 2 values are given in page 21 and 22.

WESTERN SYDNEY UNIVERSITY

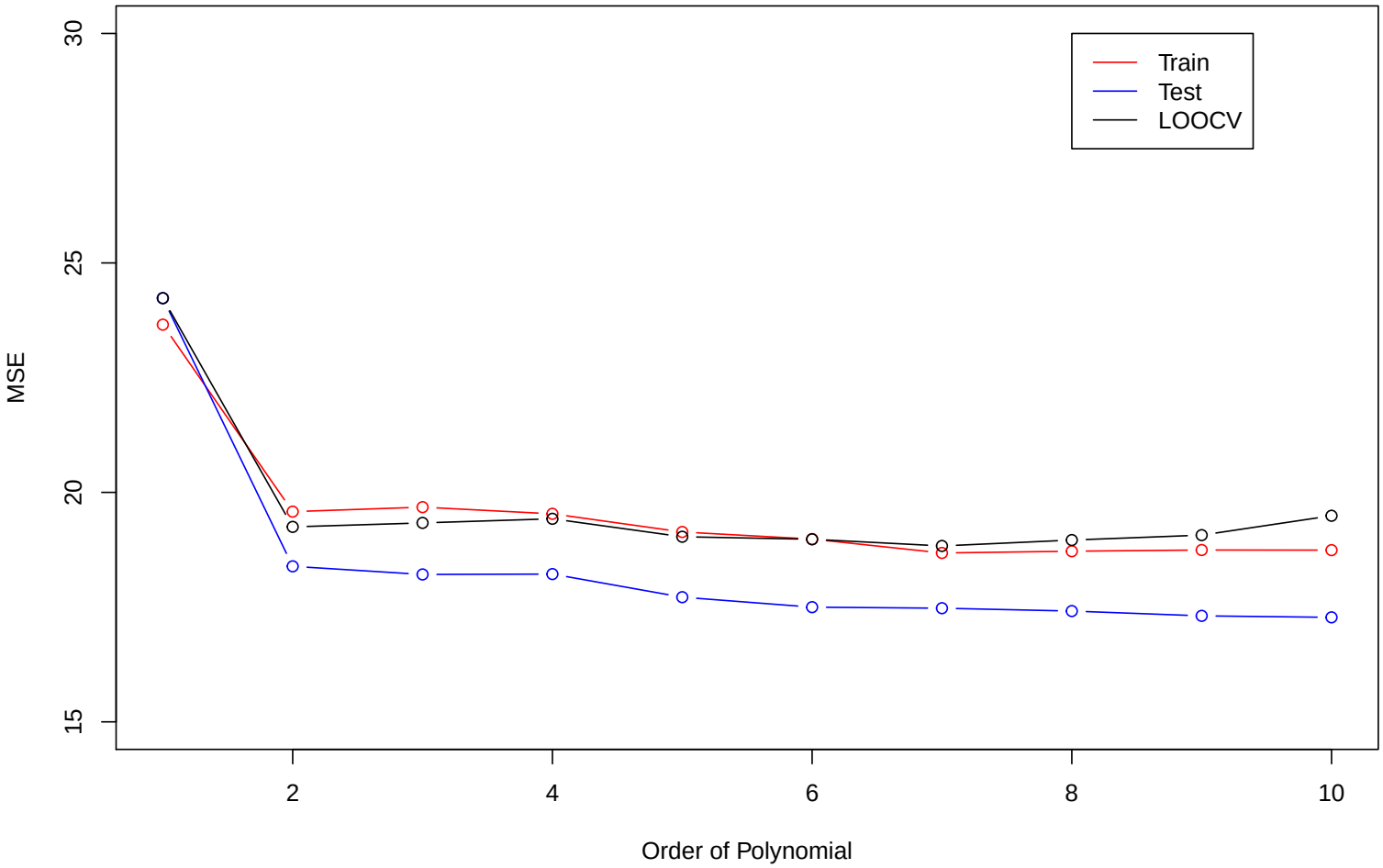
Auto data - LOOCV (Continued...)

```
cv_error=rep(0,10)
for (i in 1:10){
glm.fit=glm(mpg~poly(horsepower,i),data=Auto)
cv_error[i]=cv.glm(Auto,glm.fit)$delta[1]
}
cv_error
```

```
##  [1] 24.23151 19.24821 19.33498 19.42443 19.03321 18.97864 18.83305
##  [8] 18.96115 19.06863 19.49093
```

WESTERN SYDNEY UNIVERSITY

Auto data - LOOCV (Continued...)



WESTERN SYDNEY UNIVERSITY

LOOCV

So, in mathematical terms LOOCV estimate of MSE can be written as

$$1/n \sum_{i=1}^{n} (y_i - \hat{y}_i^{(j)})^2$$

where $\hat{y}_i^{(j)}$ indicates the prediction of y_i using a model that has left observation j out - indicated by the $^{(j)}$.

WESTERN SYDNEY UNIVERSITY

LOOCV - Linear Models

Leave-one-out-cross-validation is a deterministic measure - there is no sampling involved. However, it can be computational expensive as n copies of the model must be built, each using $n-1$ observations.

For ***linear*** models, i.e. models that are linear in their parameters; e.g. all polynomial regressions, there is a short-cut. It turns out;

$$MSE_{LOOCV} = 1/n \sum_{i=1}^{n} (y_i - \hat{y}_i^{(j)})^2 = 1/n \sum_{i=1}^{n} (y_i - \hat{y}_i / 1 - h_{ij})^2$$

where $\hat{y}_i$ is the prediction from the model using all the data, and h_{ij} is called the ***leverage*** and does not depend on the $y's$.

WESTERN SYDNEY UNIVERSITY

k-fold cross-validation

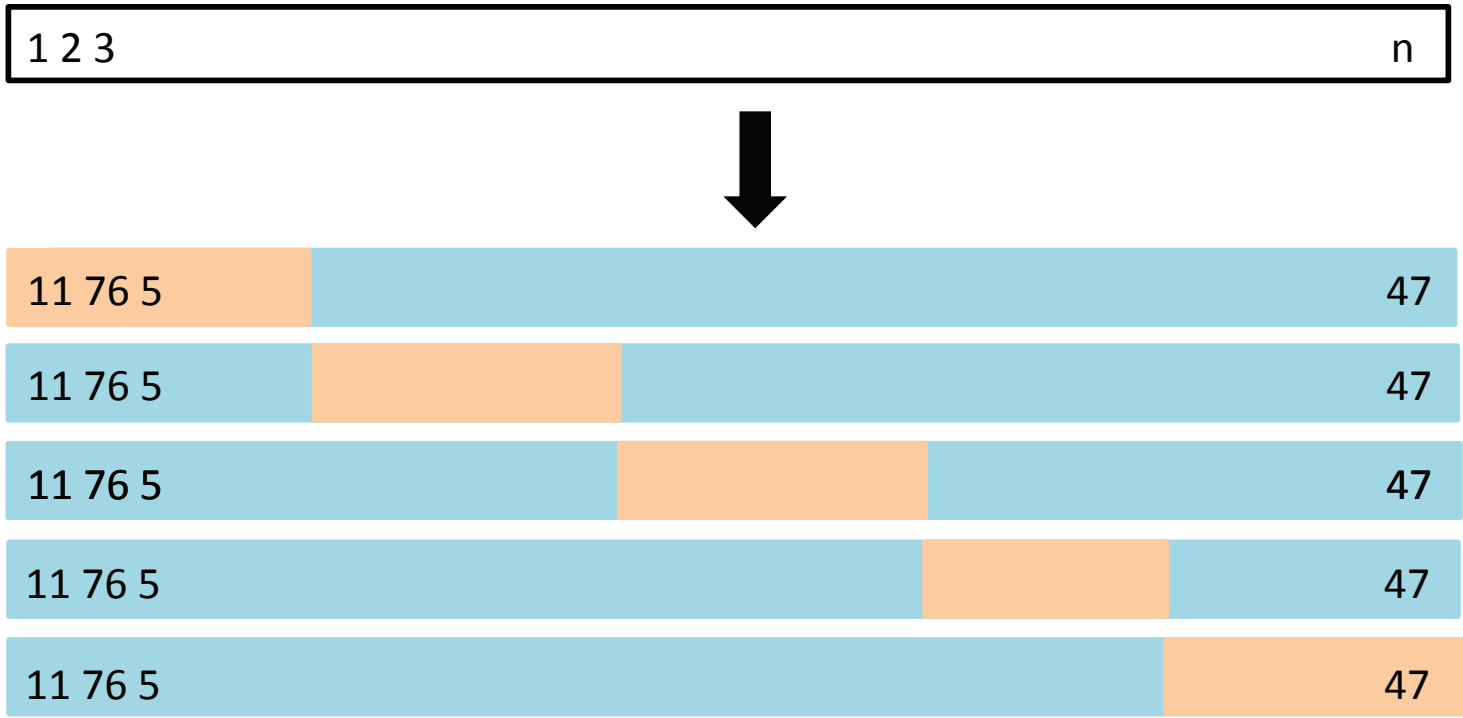
When LOOCV has to be done in the *brute force* way, it can be quite computationally expensive. So, a fast approach is to use k-fold cross-validation.

For some number k,

1. Divide the data into k ***equal sized*** groups randomly.
2. Build model on $k-1$ of the groups
3. Predict the 1 group left out
4. Repeat until all k groups have been left out.

WESTERN SYDNEY UNIVERSITY

5-fold cross-validation



Source: An introduction to Statistical Learning with applications in R

WESTERN SYDNEY UNIVERSITY

k-fold cross-validation (Continued...)

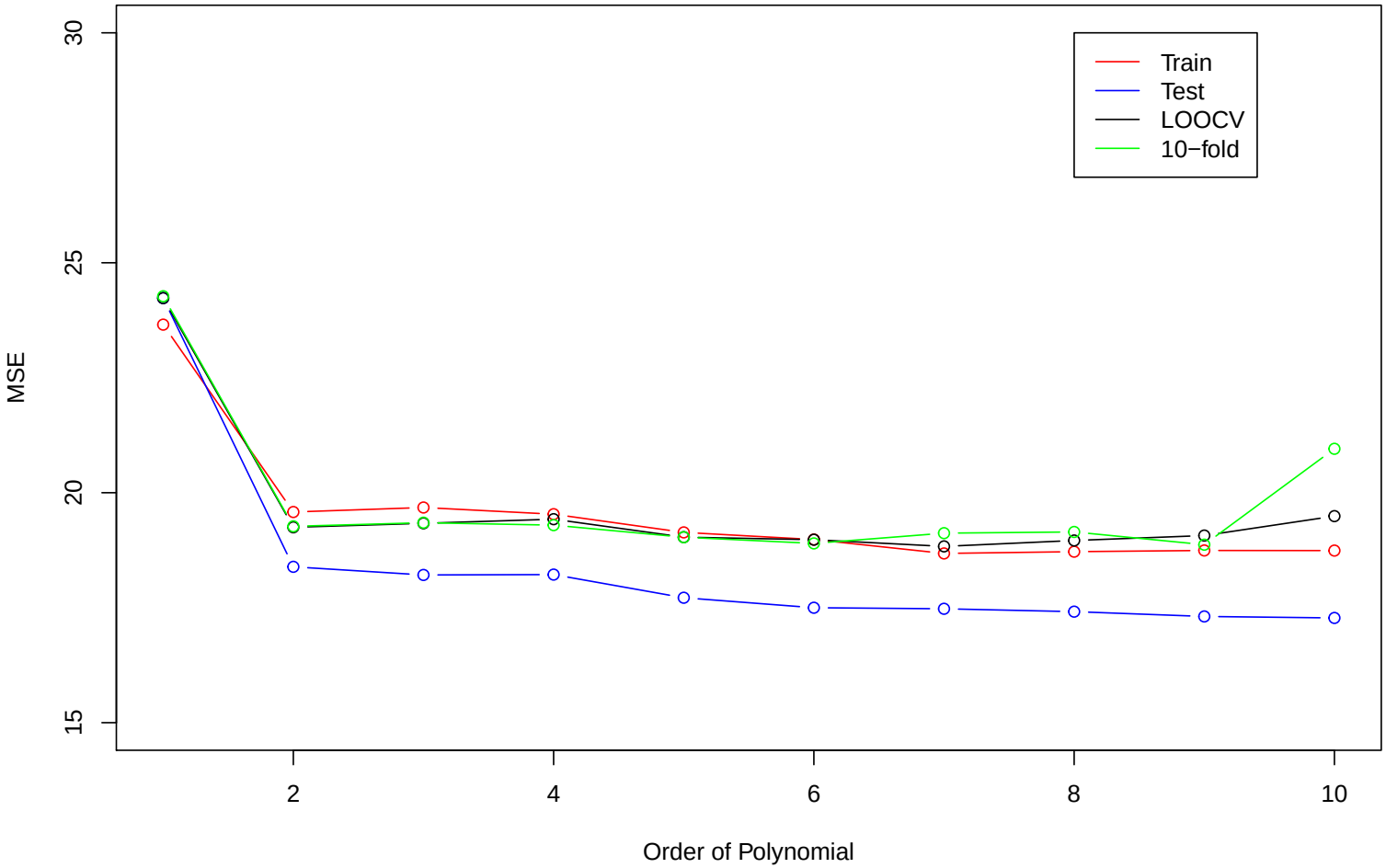
For k-fold cross-validation only k models are built each on approximately $(k - 1/k)\ n$ data points.

- Computationally faster
- But each model has less data
- and there is randomness in the group selection

WESTERN SYDNEY UNIVERSITY

k-fold cross-validation (Continued...)

Lets consider K=10



WESTERN SYDNEY UNIVERSITY

Example 2: Multiple Linear Regression Cross Validation

Let's consider several candidate models with diferent vaiables/combinations like this...

```
m1 <- glm(mpg~weight, data = Auto)
m2 <- glm(mpg~weight+year, data = Auto)
m3 <- glm(mpg~weight+year+displacement, data = Auto)
m4 <- glm(mpg~weight+year+displacement+cylinders, data = Auto)
m5 <- glm(mpg~weight+year+displacement+cylinders+acceleration, data = Auto)
m6 <- glm(mpg~weight+year+displacement+cylinders+acceleration+horsepower, data = Au
```

Suppose we want to select the best model out of these. (Note that here we have not considered all possible combinations of variables)

WESTERN SYDNEY UNIVERSITY

Validation-set approach

Let's use validation-set approach for each of the models and select the model with minimum validation error.

```
#Validation-set approach 60/40
set.seed(2)
tr.id <-  sample(1:nrow(Auto),nrow(Auto)*.6)
#train dataset
train <- Auto[tr.id , ]
#validation dataset
validation <- Auto[-tr.id , ]
dim(train)
```

```
## [1] 235   9
```

```
dim(validation)
```

```
## [1] 157   9
```

Error Estimates

```r
#create initial error vector with values 0's
train_error <- rep(0,6)
#create initial error vector with values 0's
validation_error <- rep(0,6)
```

WESTERN SYDNEY UNIVERSITY

Model 1: mpg Vs weight

```
#fit model1 for train data
m1t = glm(m1, subset = tr.id)
#prdict mpg using above model
mpg_hat1 <- predict (m1t ,Auto)
#calculating train error
train_error[1] <- mean((mpg-mpg_hat1)[tr.id]^2)
#calculating validation error
validation_error[1] <- mean((mpg-mpg_hat1)[-tr.id]^2)
train_error
```

```
## [1] 18.82841  0.00000  0.00000  0.00000  0.00000  0.00000
```

```
validation_error
```

```
## [1] 18.9272  0.0000  0.0000  0.0000  0.0000  0.0000
```

WESTERN SYDNEY UNIVERSITY

Model 2: mpg Vs weight + year

```
m2t = glm(m2, subset = tr.id)
mpg_hat2 <- predict (m2t ,Auto)
train_error[2] <- mean((mpg-mpg_hat2)[tr.id]^2)
validation_error[2] <- mean((mpg-mpg_hat2)[-tr.id]^2)
train_error
```

```
## [1] 18.82841 11.48855  0.00000  0.00000  0.00000  0.00000
```

```
validation_error
```

```
## [1] 18.92720 12.25177  0.00000  0.00000  0.00000  0.00000
```

WESTERN SYDNEY UNIVERSITY

Model 3: mpg Vs weight + year + displacement

```
m3t = glm(m3, subset = tr.id)
mpg_hat3 <- predict (m3t ,Auto)
train_error[3] <- mean((mpg-mpg_hat3)[tr.id]^2)
validation_error[3] <- mean((mpg-mpg_hat3)[-tr.id]^2)
train_error
```

```
## [1] 18.82841 11.48855 11.47487  0.00000  0.00000  0.00000
```

```
validation_error
```

```
## [1] 18.92720 12.25177 12.29397  0.00000  0.00000  0.00000
```

WESTERN SYDNEY UNIVERSITY

Model 4: mpg Vs weight + year + displacement + cylinders

```
m4t = glm(m4, subset = tr.id)
mpg_hat4 <- predict (m4t ,Auto)
train_error[4] <- mean((mpg-mpg_hat4)[tr.id]^2)
validation_error[4] <- mean((mpg-mpg_hat4)[-tr.id]^2)
train_error
```

```
## [1] 18.82841 11.48855 11.47487 11.46708  0.00000  0.00000
```

```
validation_error
```

```
## [1] 18.92720 12.25177 12.29397 12.24129  0.00000  0.00000
```

WESTERN SYDNEY UNIVERSITY

Model 5: mpg Vs weight + year + displacement + cylinders + acceleration

```
m5t = glm(m5, subset = tr.id)
mpg_hat5 <- predict (m5t ,Auto)
train_error[5] <- mean((mpg-mpg_hat5)[tr.id]^2)
validation_error[5] <- mean((mpg-mpg_hat5)[-tr.id]^2)
train_error
```

```
## [1] 18.82841 11.48855 11.47487 11.46708 11.34923  0.00000
```

```
validation_error
```

```
## [1] 18.92720 12.25177 12.29397 12.24129 12.39363  0.00000
```

WESTERN SYDNEY UNIVERSITY

Model 6: mpg Vs weight + year + displacement + cylinders + acceleration + horsepower

```
m6t = glm(m6, subset = tr.id)
mpg_hat6 <- predict (m6t ,Auto)
train_error[6] <- mean((mpg-mpg_hat6)[tr.id]^2)
validation_error[6] <- mean((mpg-mpg_hat6)[-tr.id]^2)
train_error
```

```
## [1] 18.82841 11.48855 11.47487 11.46708 11.34923 11.27159
```

```
validation_error
```

```
## [1] 18.92720 12.25177 12.29397 12.24129 12.39363 12.67005
```

WESTERN SYDNEY UNIVERSITY

Plot

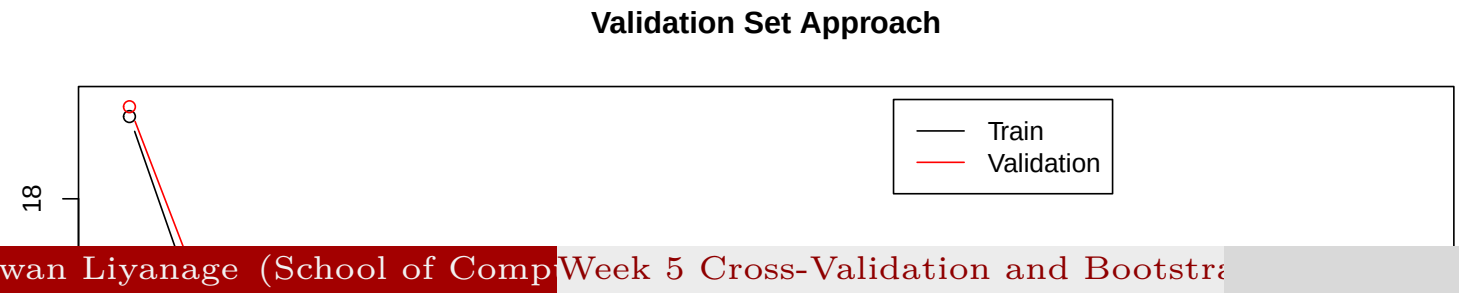
```
train_error
```

```
## [1] 18.82841 11.48855 11.47487 11.46708 11.34923 11.27159
```

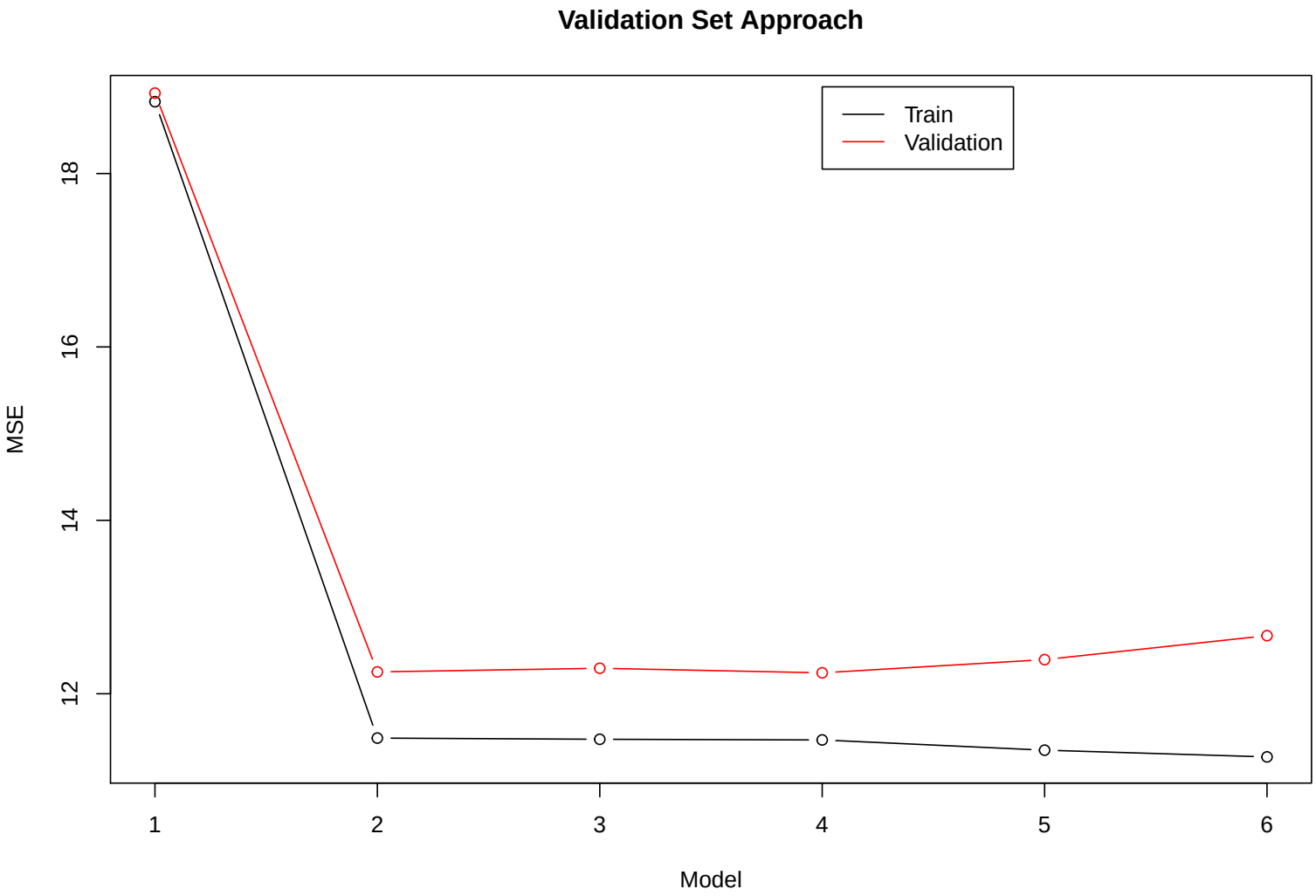
```
validation_error
```

```
## [1] 18.92720 12.25177 12.29397 12.24129 12.39363 12.67005
```

```
model <- c(1:6)
plot(model, train_error, type = "b", xlab = "Model", ylab = "M
{lines(validation_error, type = "b", col = "red")}
legend(x = 4, y = 19, c("Train", "Validation"), col = c("black
```



MSE Plot



WESTERN SYDNEY UNIVERSITY

LOOCV approach

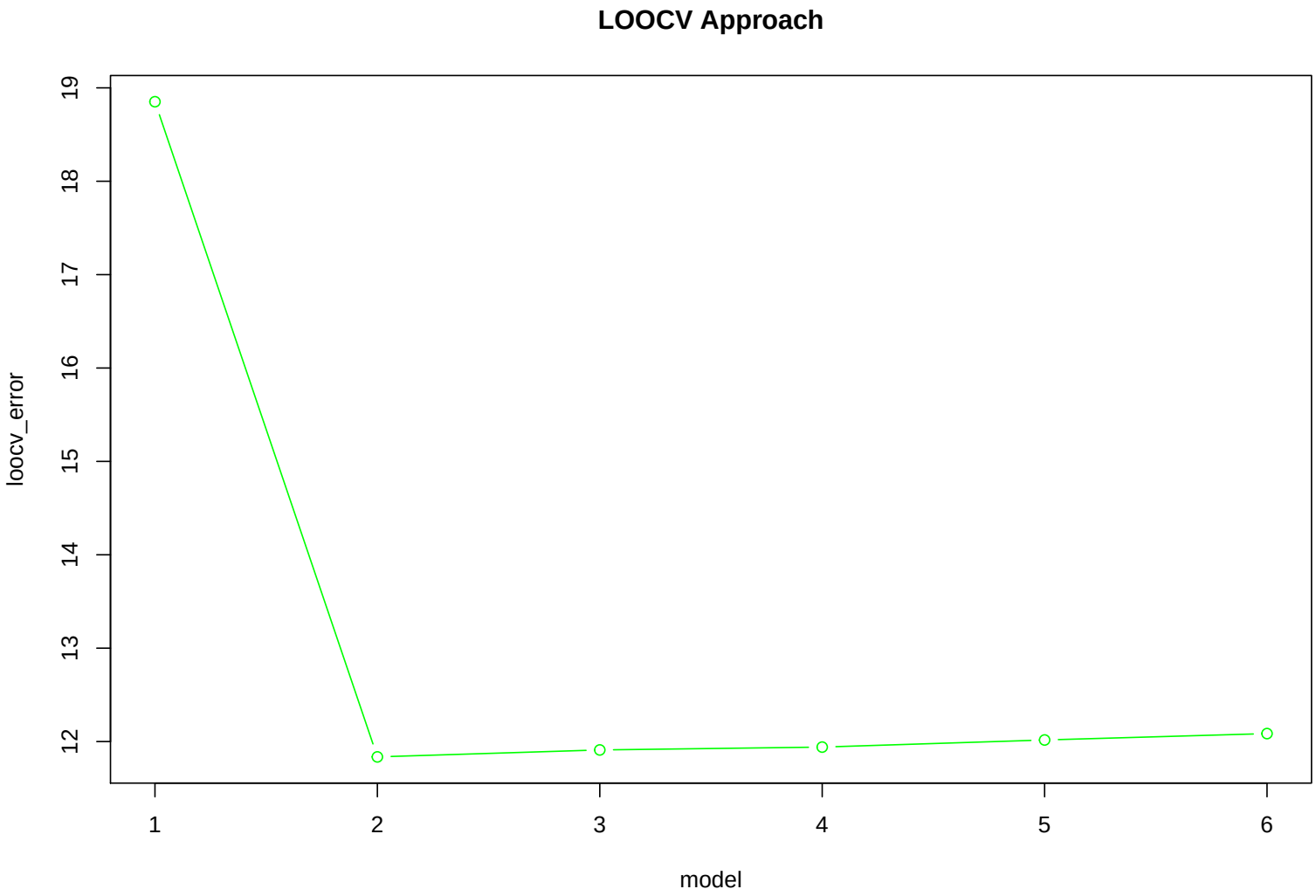
Now, let's use LOOCV approach for each of the models and select the model with minimum loocv error.

```
loocv_error=rep(0,6) #create initial error vector with values
loocv_error[1]=cv.glm(Auto,m1)$delta[1]
loocv_error[2]=cv.glm(Auto,m2)$delta[1]
loocv_error[3]=cv.glm(Auto,m3)$delta[1]
loocv_error[4]=cv.glm(Auto,m4)$delta[1]
loocv_error[5]=cv.glm(Auto,m5)$delta[1]
loocv_error[6]=cv.glm(Auto,m6)$delta[1]
loocv_error
```

```
## [1] 18.85161 11.83564 11.91013 11.94075 12.01676 12.08526
```

WESTERN SYDNEY UNIVERSITY

Plot LOOCV Error



WESTERN SYDNEY UNIVERSITY

K-fold cross-validation

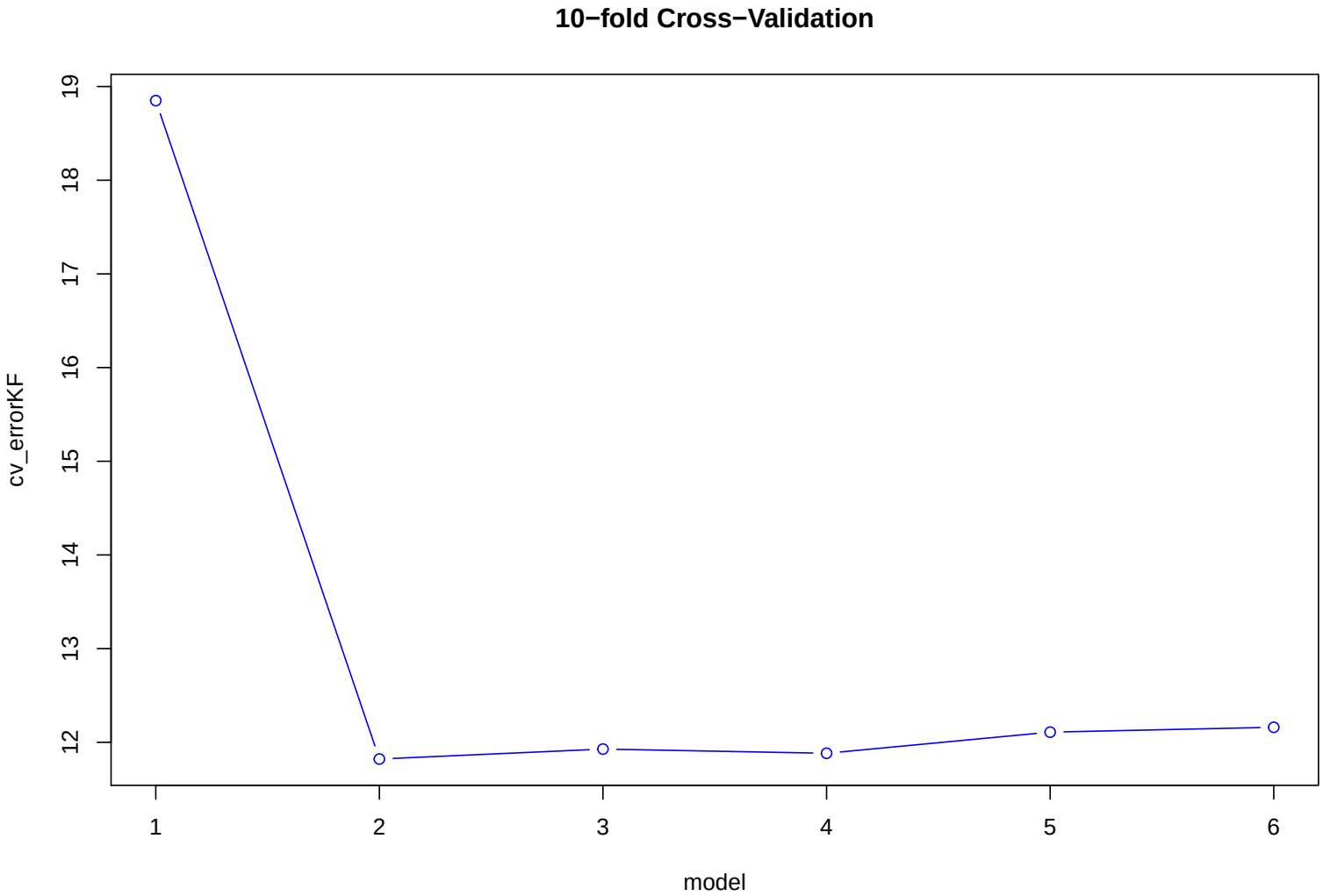
Now, let's use 10-fold cross-validation for each of the models and select the model with minimum cross-validation error.

```
cv_errorKF= rep (0,6)
set.seed(2) #seed can be any value
cv_errorKF[1] <- cv.glm(Auto,m1, K=10)$delta[1]
cv_errorKF[2] <- cv.glm(Auto,m2, K=10)$delta[1]
cv_errorKF[3] <- cv.glm(Auto,m3, K=10)$delta[1]
cv_errorKF[4] <- cv.glm(Auto,m4, K=10)$delta[1]
cv_errorKF[5] <- cv.glm(Auto,m5, K=10)$delta[1]
cv_errorKF[6] <- cv.glm(Auto,m6, K=10)$delta[1]

cv_errorKF
```

WESTERN SYDNEY UNIVERSITY

Plot K-fold cross-validation Error

```
## [1] 18.84942 11.82160 11.92776 11.88292 12.10859 12.16042
```



Model Comparison

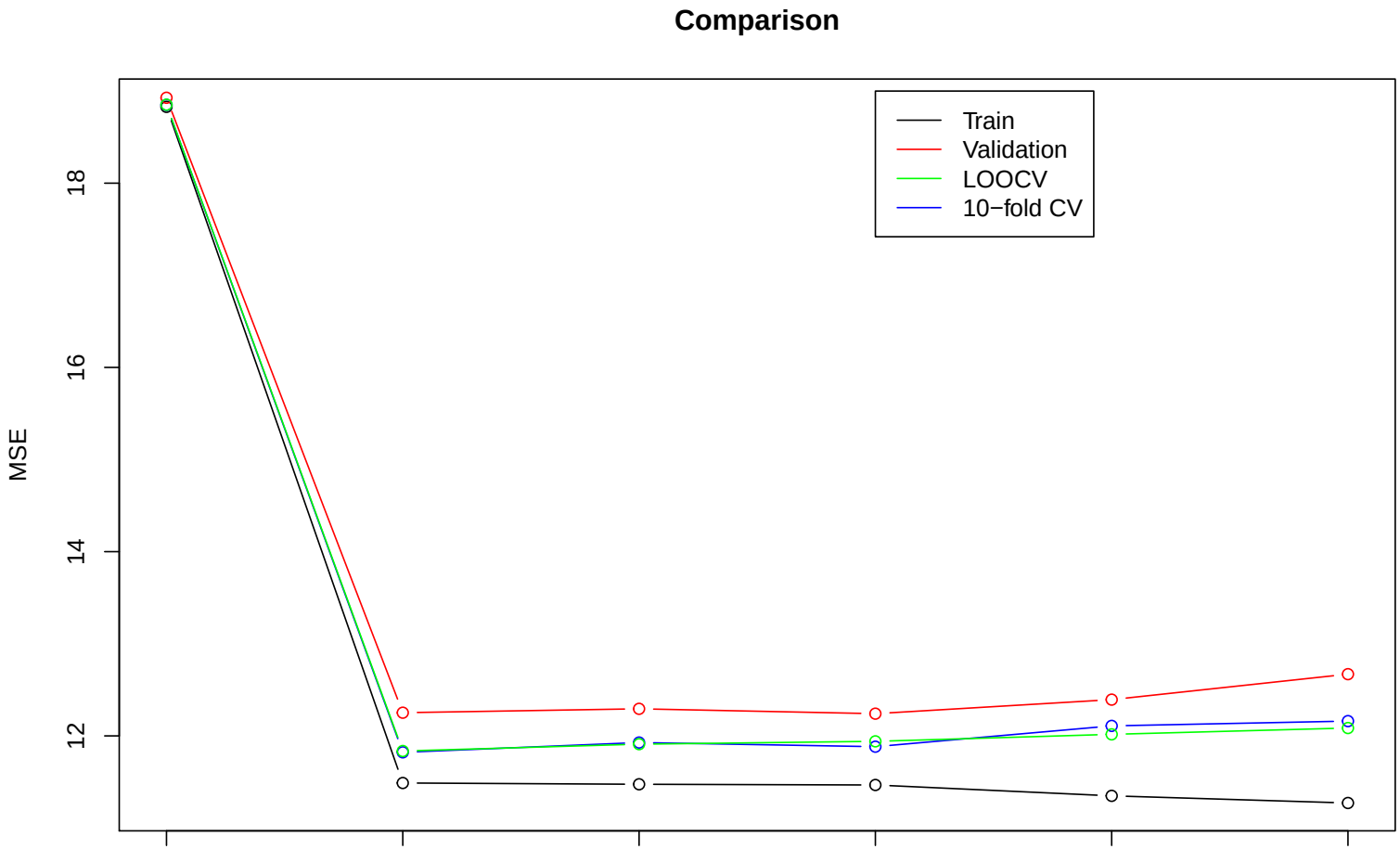
Now let's compare all the approaches.

```
plot(model, train_error, type = "b", xlab = "Model", ylab = "M
{    lines(validation_error, type = "b", col = "red")
    lines(cv_errorKF, type = "b", col = "blue")
    lines(loocv_error, type = "b", col = "green")
}

legend(x = 4, y = 19, c("Train", "Validation", "LOOCV", "10-fo
```

WESTERN SYDNEY UNIVERSITY

Plot for Model Comparison

Now let's compare all the approaches.Considering all the approaches, we can conclude that model 2 is the best model among the models considered.



WESTERN SYDNEY UNIVERSITY

Classification

In classification models we are not modelling a continuous respose, but a class. The simplest case is two classes - for example, *"has disease"* versus *"does not have disease"*

Here MSE and RSS is **NOT** valid measures of error.

Instead the proportion of error is used.

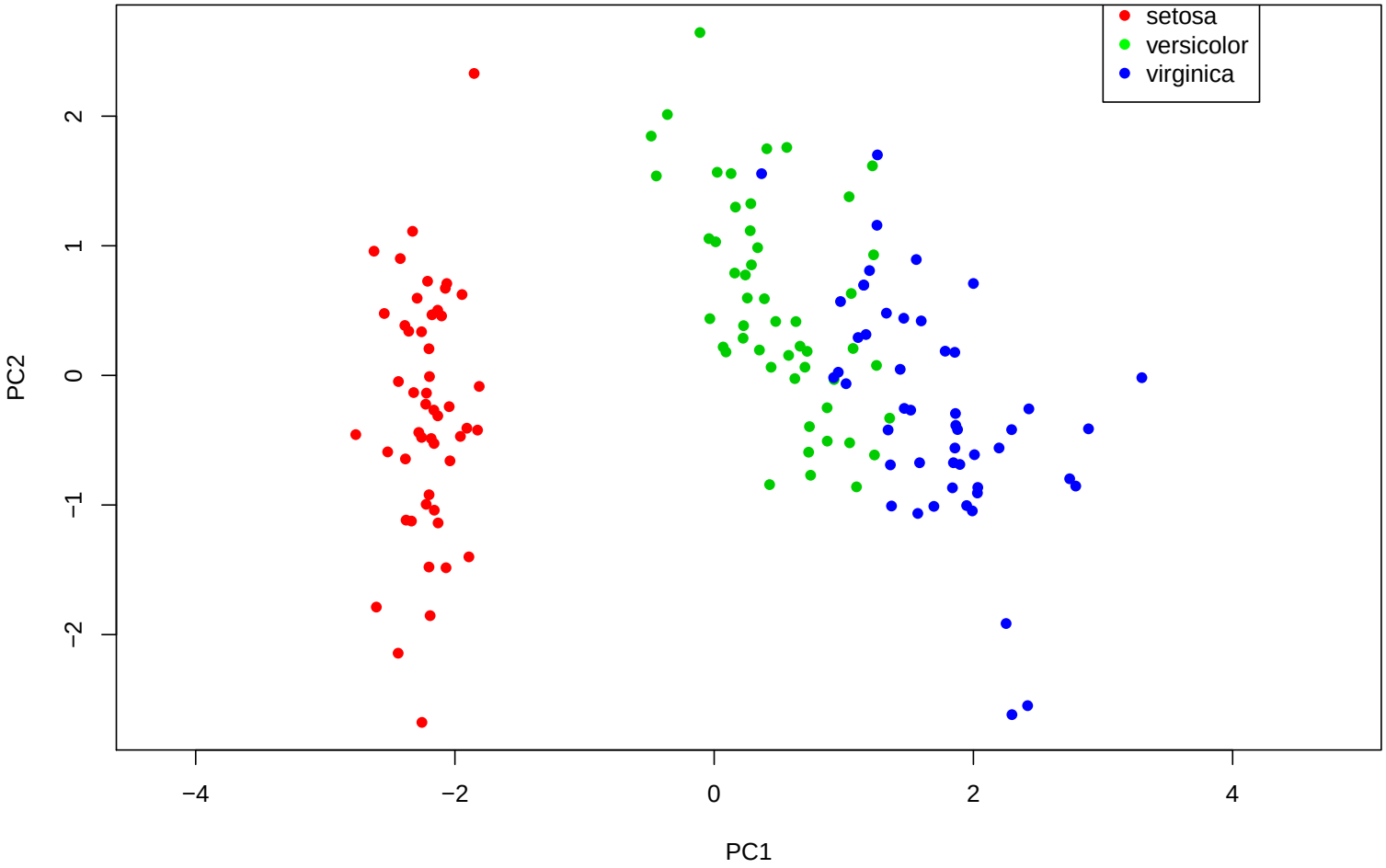
$$Err = 1/n \sum_{i=1}^{n} I(y_i \neq \hat{y}_i)$$

where $I(.)$ is an ***indicator function***, taking value "1" if condition is true and "0" if false.

WESTERN SYDNEY UNIVERSITY

Classification - Example

Remember the ***Iris data***. Lets try and classify species.



WESTERN SYDNEY UNIVERSITY

Classification - Example

Following table gives the misclassification rate in three approaches: **Training, LOOCV, K-foldCV**

Logistic Regression: Setosa vs. Versicolor

Training	LOOCV	k-fold
0	0	0

Logistic Regression: Virginica vs. Versicolor

Training	LOOCV	k-fold
0.02	0.0346	0.0496

WESTERN SYDNEY UNIVERSITY

Bootstrap

- Suppose you have a ***statistic*** based on some data. (A statistic is just a function of the data.)
- You are probably interested in the properties of that statistic
- What's its mean? variance? etc.
- In many cases, the ***distribution*** of the statistic can be calculated and its properties derived from that
- For example, the sample mean of Normal data, is Normally distributed
- In general, the sample mean of a large data set, is also Normally distributed.
- Sometimes this is difficult to do (mathematically) +E.g. median

WESTERN SYDNEY UNIVERSITY

The median

The median of a sample is ***any*** number that has half (50%) the sample larger and half (50%) smaller than it.

If there are an odd number of points in the sample, the median is the middle point.

If there are an even number, the median is (by convention) the midpoint of the two middle points.

WESTERN SYDNEY UNIVERSITY

Bootstrap (Continued...)

What is the distribution of the median?

Suppose we have a sample of size n. If we could get a lot of samples of the same size, we could draw a histogram of the medians.

```
x = 10*rexp(11)
n = length(x)
B = 100
M = rep(NA, B)
M[1] = median(x)
for(i in 2:B)M[i] = median(sample(x, replace=TRUE))
mean(M)
```

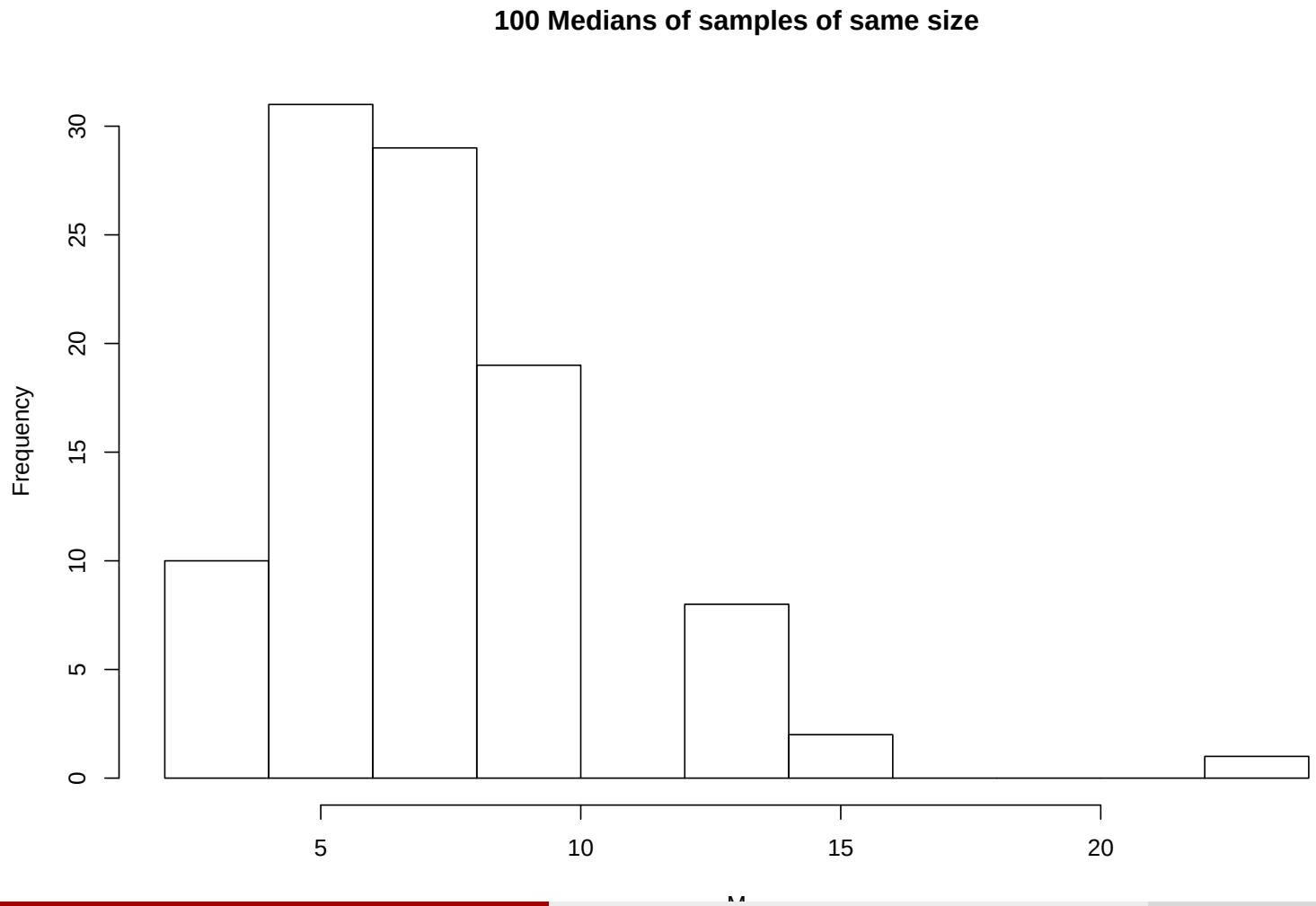
```
## [1] 6.769362
```

```
sd(M)
```

```
## [1] 3.374339
```

Bootstrap (Continued...)

```
hist(M, main ="100 Medians of samples of same size")
```



WESTERN SYDNEY UNIVERSITY

Bootstrap (Continued...)

PROBLEM: We only have ***ONE*** sample...

The bootstrap idea is to use the original sample many times. We sample from the original sample with ***replacement***.

Sampling with replacement means each original data point can occur multiple times, and some may not occur in the bootstrap sample

- Original sample : 1, 2, 3, 4, 5, 6, 7, 8, 9, 10
- Bootstrap sample : 1, 3, 4, 4, 4, 4, 7, 8, 10, 10

We can repeat this as many times as we like.

WESTERN SYDNEY UNIVERSITY

Bootstrap Median

What is the distribution of the median?

```
x = 1:10
n = length(x)
B = 100
M = rep(NA, B)
M[1] = median(x)
for(i in 2:B)M[i] = median(sample(x, replace=TRUE))
mean(M)
```

```
## [1] 5.18
```

```
sd(M)
```

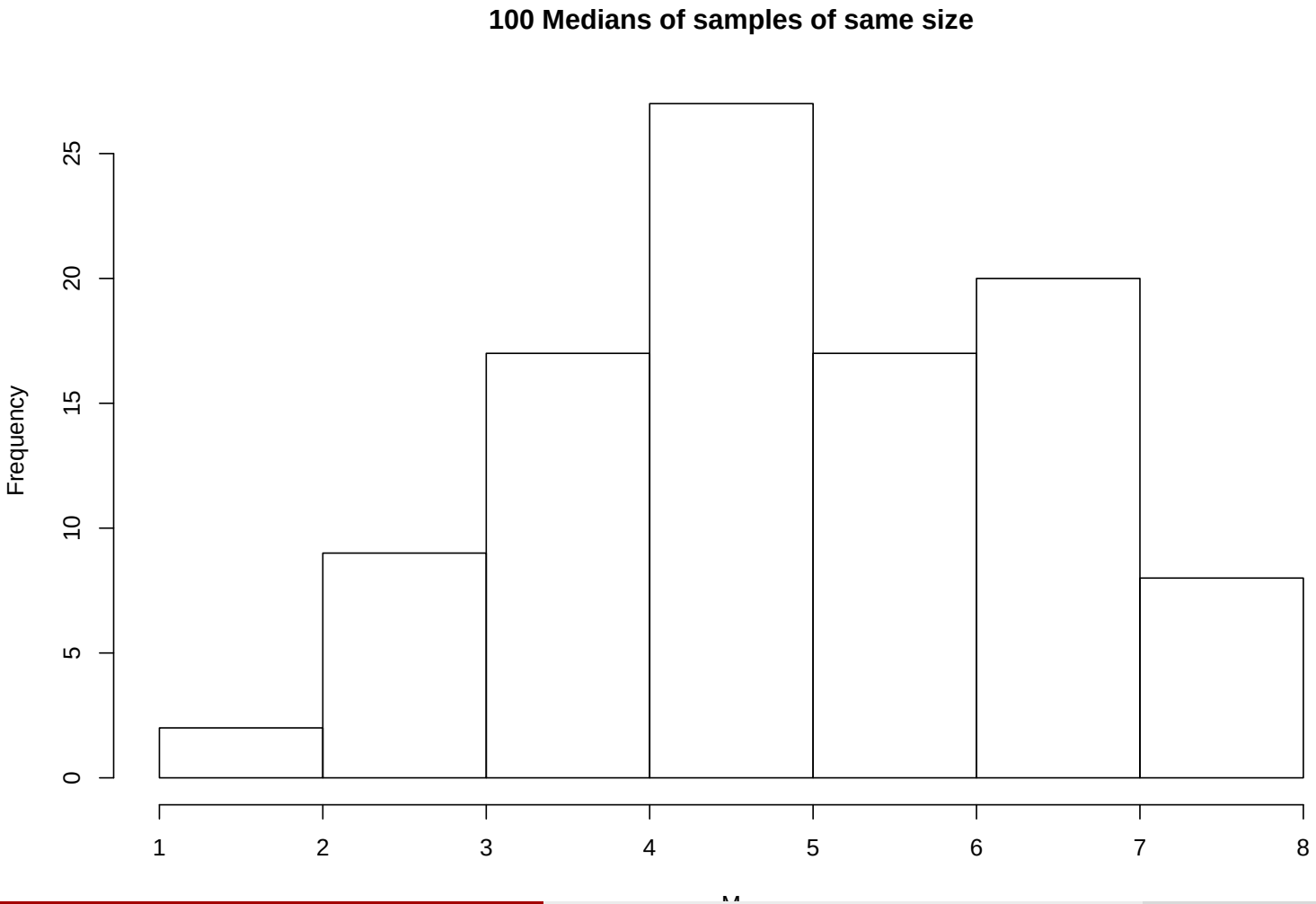
```
## [1] 1.488203
```

WESTERN SYDNEY UNIVERSITY

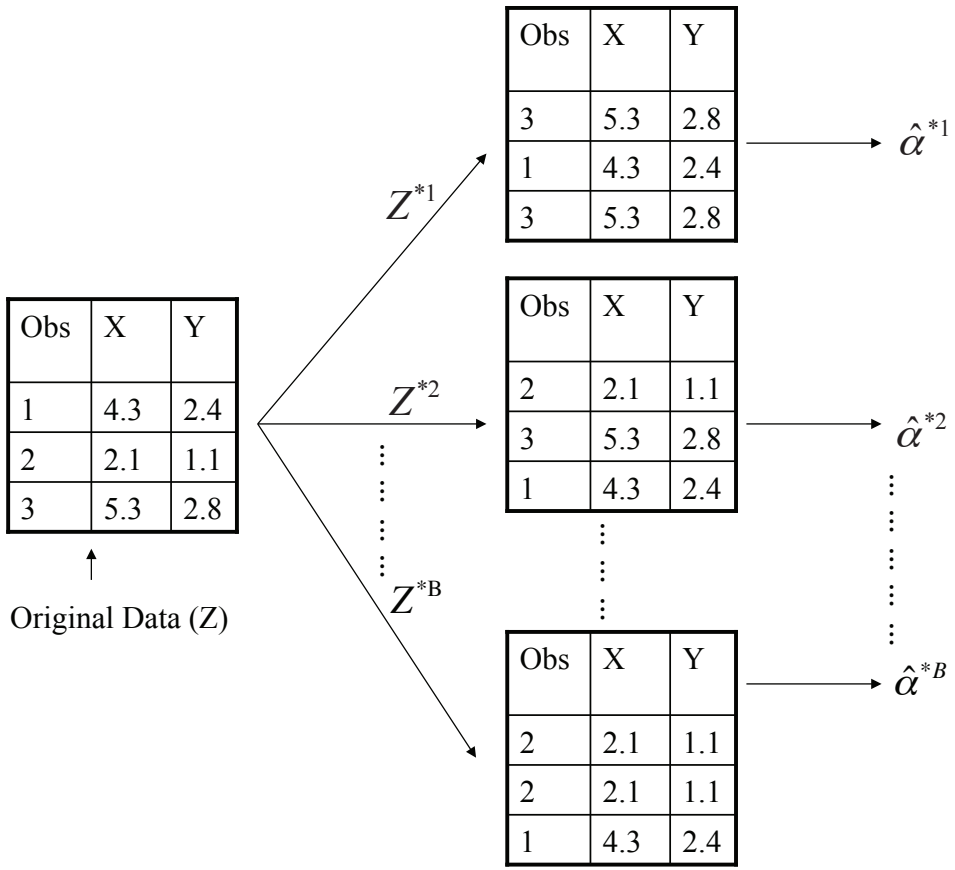
The ***mean median*** is 5.18 and standard deviation is 1.4882028.

Bootstrap Median (Continued...)

```
hist(M, main ="100 Medians of samples of same size")
```

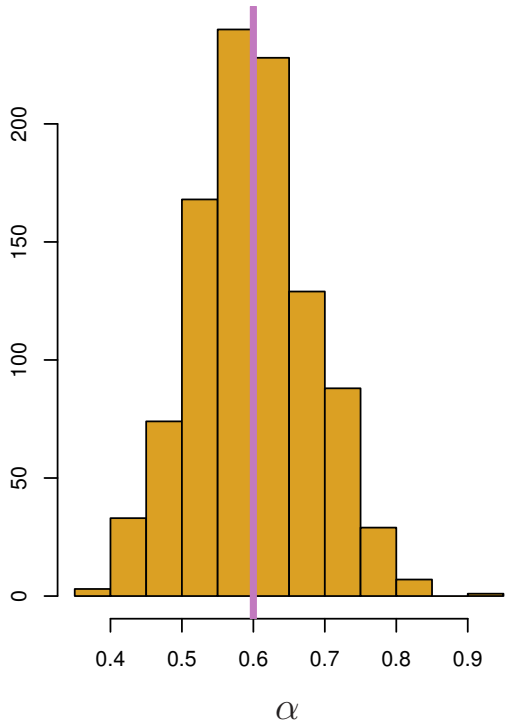


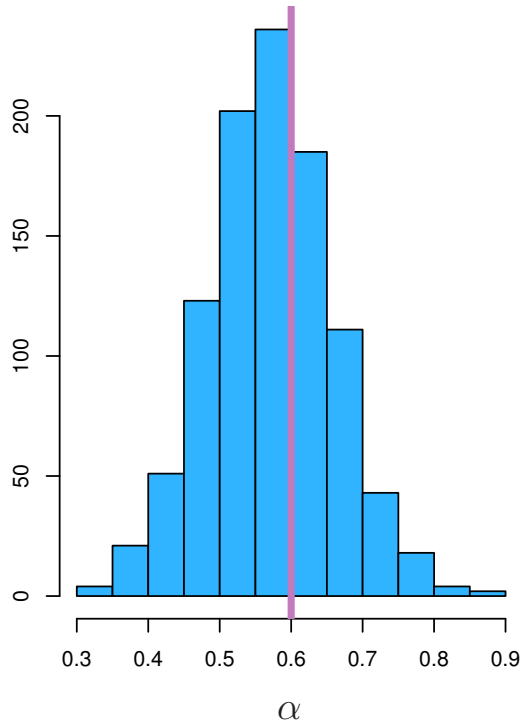
Bootstrap - Applications in assessing the accurcy of parameter estimates

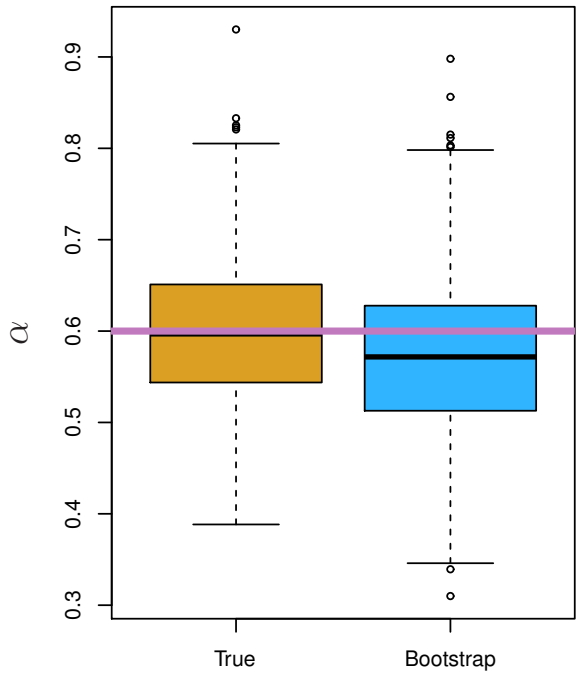


Source: An introduction to Statistical Learning with applications in R

Bootstrap - Applications in assessing the accurcy of parameter estimates (Continued...)







Source: An introduction to Statistical Learning with applications in R

TEXT BOOK

Lecture notes are based on the textbook.

For further reference refer;

Prescribed Textbook - Chapter 5

James, G., Witten, D., Hastie, T., & Tibshirani, R. (2013). An Introduction to Statistical Learning: with Applications in R Springer.

WESTERN SYDNEY UNIVERSITY