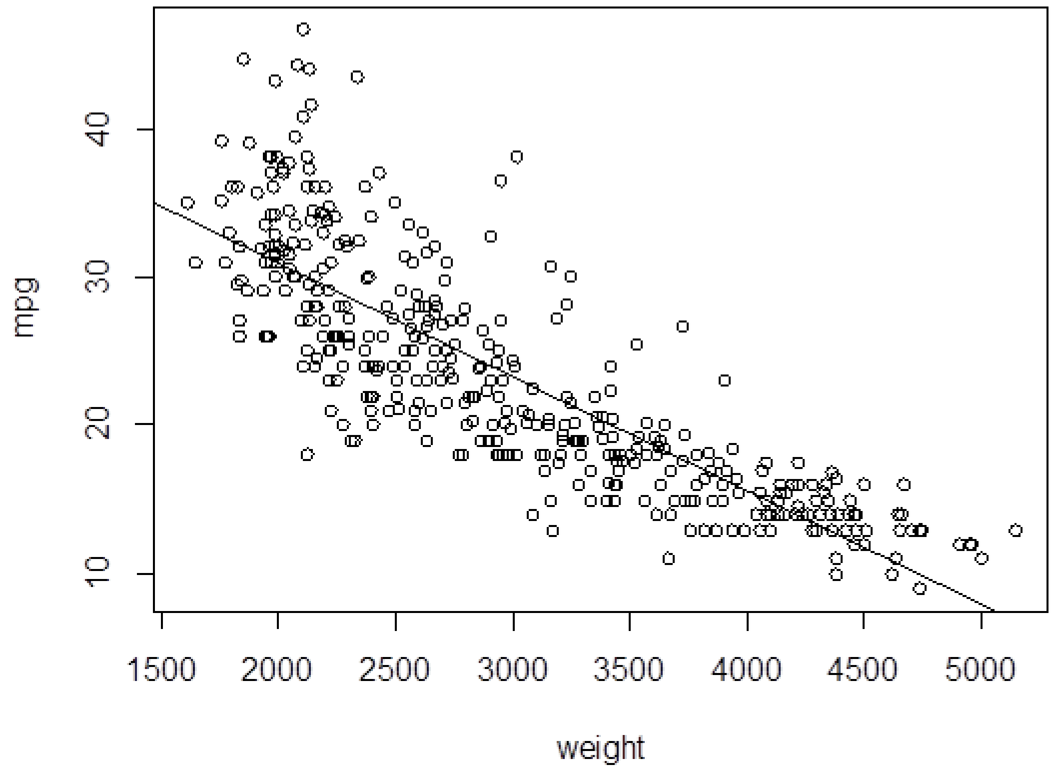
POLYNOMIALS AND SPLINES

-

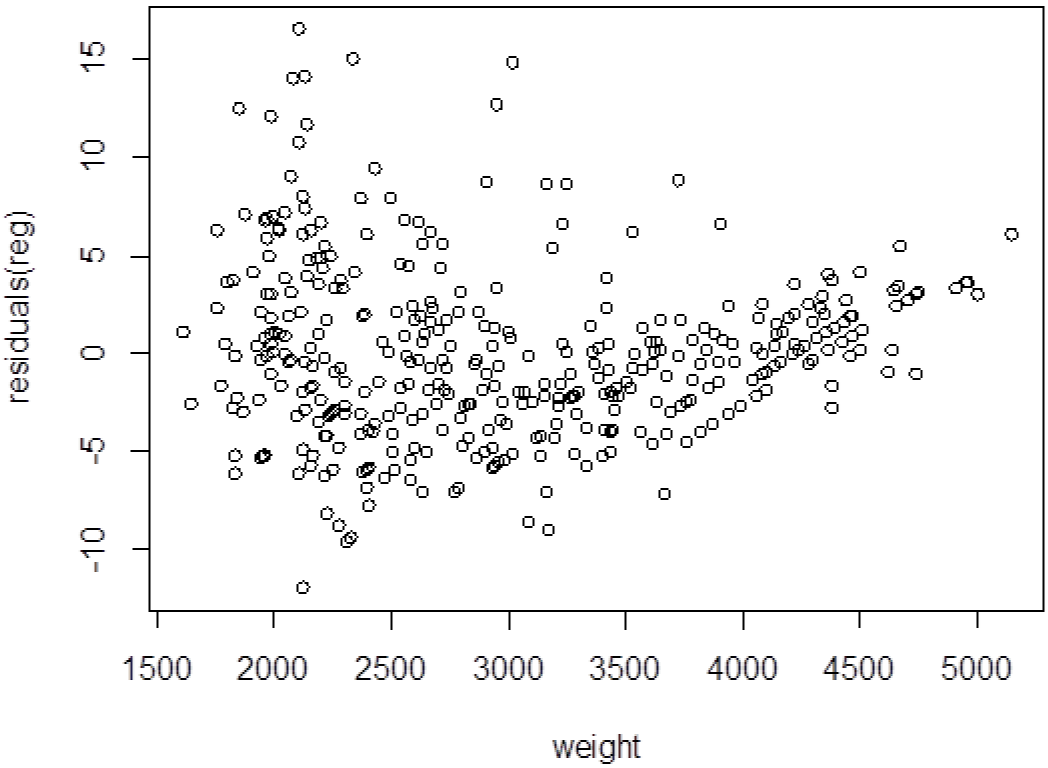
1. Fitting a polynomial regression

```
> attach(Auto);
> plot( weight, mpg )
> reg = lm( mpg ~ weight )
> abline(reg)
```



 # Look at the residuals. Is linear model adequate?

```
> plot( weight, residuals(reg) )
```



 # If a polynomial regression is appropriate, what is the best degree?

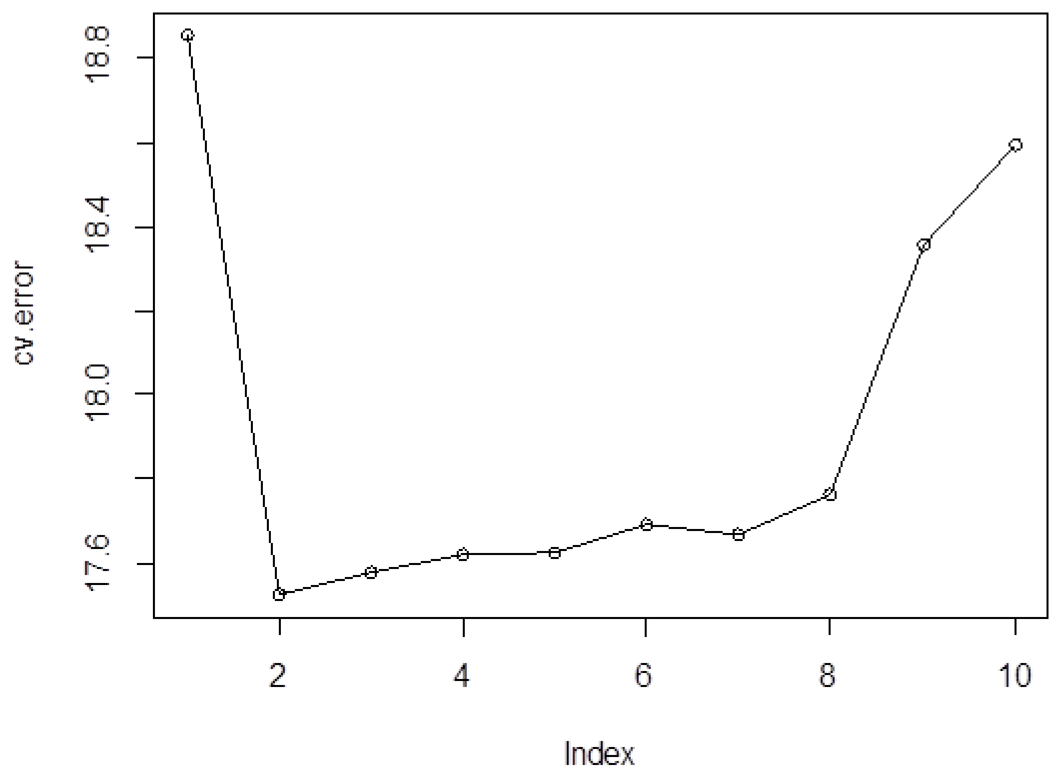
```
> install.packages("leaps")
> library(leaps)
> polynomial.fit = regsubsets( mpg ~ poly(weight,10), data=Auto )
> summary(polynomial.fit)
poly(weight, 10)1 poly(weight, 10)2 poly(weight, 10)3 poly(weight, 10)4 poly
(weight, 10)5 poly(weight, 10)6
1  ( 1 ) "*"               " "               " "               " "
" "               " "
2  ( 1 ) "*"               "*"               " "               " "
" "               " "
3  ( 1 ) "*"               "*"               " "               " "
" "               " "
4  ( 1 ) "*"               "*"               " "               " "
" "               " "
5  ( 1 ) "*"               "*"               " "               " "
"*"               " "
6  ( 1 ) "*"               "*"               " "               " "
"*"               "*"
7  ( 1 ) "*"               "*"               " "               " "
"*"               "*"
8  ( 1 ) "*"               "*"               " "               "*"
"*"               "*"
         poly(weight, 10)7 poly(weight, 10)8 poly(weight, 10)9 poly(weight, 10)10
1  ( 1 ) " "               " "               " "               " "
2  ( 1 ) " "               " "               " "               " "
3  ( 1 ) " "               " "               " "               "*"
4  ( 1 ) "*"               " "               " "               "*"
5  ( 1 ) "*"               " "               " "               "*"
6  ( 1 ) "*"               " "               " "               "*"
7  ( 1 ) "*"               " "               "*"               "*"
8  ( 1 ) "*"               " "               "*"               "*"

> which.max(summary(polynomial.fit)$adjr2)
[1] 4
```

> which.min(summary(polynomial.fit)$cp)
[1] 3
> which.min(summary(polynomial.fit)$bic)
[1] 2

 # BIC chooses a quadratic model "*mpg =* $\beta_0 + \beta_1$ *(weight) +* β_2*(weight)*^{2} *+* ε". Mallows C_p and adjusted R^2 add higher order terms.

 # Cross-validation agrees with the quadratic model…
> library(boot)
> cv.error = rep(0,10)
> for (p in 1:10){
+ polynomial = glm(mpg ~ poly(weight,p))
+ cv.error[p] = cv.glm(Auto, polynomial)$delta[1] }
> cv.error
 [1] 18.85161 17.52474 17.57811 17.62324 17.62822 17.69418 17.66695 17.76456 18.35543 18.59401
> plot(cv.error); lines(cv.error)



 # So, we choose the quadratic regression – degree 2 polynomial. Its prediction MSE is 17.52.
> poly2 = lm(mpg ~ poly(weight,2))
> summary(poly2)

```
Coefficients:
                   Estimate Std. Error t value Pr(>|t|)
(Intercept)         23.4459     0.2109 111.151  < 2e-16 ***
poly(weight, 2)1 -128.4436     4.1763 -30.755  < 2e-16 ***
poly(weight, 2)2   23.1589     4.1763   5.545 5.43e-08 ***
```

```
> plot( weight,mpg )
> Yhat_poly2 = fitted.values( poly2 )
> points( weight, Yhat_poly2, col="red", lwd=3 )
```