Prediction intervals with R

```
> kars = read.table("http://www.utstat.utoronto.ca/~brunner/data/legal/mcars4.data")
> head(kars); attach(kars)
  Cntry lper100k weight length
1    US     19.8   2178   5.92
2 Japan      9.9   1026   4.32
3    US     10.8   1188   4.27
4    US     12.5   1444   5.11
5    US     12.5   1485   5.03
6    US     12.5   1485   5.03
> contrasts(Cntry)
      Japan US
Europ     0  0
Japan     1  0
US        0  1
> fullmodel = lm(lper100k ~ weight + length + Cntry)
> summary(fullmodel)

Call:
lm(formula = lper100k ~ weight + length + Cntry)

Residuals:
    Min      1Q  Median      3Q     Max
-4.5063 -0.8813  0.0147  1.3043  2.9432

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -5.789215   2.855736  -2.027 0.045441 *
weight       0.005457   0.001472   3.707 0.000352 ***
length       2.345968   0.980329   2.393 0.018676 *
CntryJapan   0.506517   0.660158   0.767 0.444826
CntryUS     -1.487722   0.575633  -2.584 0.011274 *
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 1.703 on 95 degrees of freedom
Multiple R-squared:  0.7431,  Adjusted R-squared:  0.7323
F-statistic: 68.71 on 4 and 95 DF,  p-value: < 2.2e-16

> fullmodel$df.residual # n-p
[1] 95
```

```
>
> ###### Predictions, confidence intervals and prediction intervals ######
>
> # Predict litres per 100 km for a Japanese car weighing
> # 1295kg, 4.52m long (1990 Toyota Camry)
>
> b = fullmodel$coefficients; b
 (Intercept)       weight       length   CntryJapan      CntryUS
-5.789214693  0.005456609  2.345968436  0.506517030 -1.487721833
> ell = c(1,1295,4.52,1,0)
> yhat = sum(ell*b); # ell-prime betahat
> yhat
[1] 12.38739
>
> # Confidence interval for E(ell-prime beta)
> # First the hard way
```

$$\boldsymbol{\ell}^\top \widehat{\boldsymbol{\beta}} \pm t_{\alpha/2} \sqrt{MSE\, \boldsymbol{\ell}^\top (X^\top X)^{-1} \boldsymbol{\ell}}$$

```
>
> tcrit = qt(0.975,df=fullmodel$df.residual) # t_alpha/2
> MSE.XpXinv = vcov(fullmodel)
> ell = as.matrix(ell) # Now it's a column vector
> me95 = tcrit * sqrt( as.numeric(t(ell) %*% MSE.XpXinv %*% ell) )
> lower95 = yhat - me95; upper95 = yhat + me95
> c(lower95, upper95) # 95% Confidence interval for ell-prime beta
[1] 11.37128 13.40349
>
> # Use the predict function
> # help(predict.lm)
>
> camry1990 = data.frame(weight=1295,length=4.52,Cntry='Japan')
> camry1990
  weight length Cntry
1   1295   4.52 Japan
> predict(fullmodel,newdata=camry1990) # Compare yhat = 12.38739
       1
12.38739
> predict(fullmodel,newdata=camry1990, interval='confidence')
       fit      lwr      upr
1 12.38739 11.37128 13.40349




>
```

```
> # With 95 percent prediction interval (95 is default)
```

$$\boldsymbol{\ell}^\top \widehat{\boldsymbol{\beta}} \pm t_{\alpha/2} \sqrt{MSE \left(1 + \boldsymbol{\ell}'(X'X)^{-1}\boldsymbol{\ell}\right)}$$

```
> predict(fullmodel,newdata=camry1990, interval='prediction')
       fit      lwr      upr
1 12.38739 8.856608 15.91817
>

>  # Multiple predictions
> cadillac1990 = data.frame(weight=1800,length=5.22,Cntry='US')
> volvo1990 = data.frame(weight=1371,length=4.823,Cntry='Europ')
> newcars = rbind(camry1990,cadillac1990,volvo1990); newcars
  weight length Cntry
1   1295  4.520 Japan
2   1800  5.220    US
3   1371  4.823 Europ
>
> is.data.frame(newcars)
[1] TRUE
>
> predict(fullmodel,newdata=newcars, interval='prediction')
       fit       lwr      upr
1 12.38739  8.856608 15.91817
2 14.79091 11.354379 18.22745
3 13.00640  9.481598 16.53121
>
```

```
> # "Predict" fuel consumption in the data set
> cbind(lper100k, predict(fullmodel,interval='prediction')  )
    lper100k       fit       lwr      upr
1       19.8 18.495691 15.012790 21.97859
2        9.9 10.450367  6.940724 13.96001
3       10.8  9.222800  5.747978 12.69762
4       12.5 12.590305  9.162660 16.01795
5       12.5 12.626349  9.219284 16.03341
6       12.5 12.626349  9.219284 16.03341
7       10.4  9.766491  6.236956 13.29603
8       13.2 14.570386 11.135730 18.00504
9       17.0 14.056832 10.515850 17.59782
10       9.2  8.330626  4.870298 11.79095
11       7.9  8.751877  5.216459 12.28730
12      11.9 11.281870  7.822692 14.74105
13      11.3 15.406435 11.883434 18.92944
14       9.2  7.170520  3.599857 10.74118
15      13.2 11.001447  7.491136 14.51176
16       8.2  7.613695  4.137356 11.09003
17      19.8 18.495691 15.012790 21.97859
18      17.0 14.451529 11.023363 17.87969
19      14.0 13.487155  9.955616 17.01869
20       7.7  9.948953  6.437389 13.46052
21      17.0 15.853171 12.417118 19.28922


91      17.0 15.853171 12.417118 19.28922
92      17.0 14.451529 11.023363 17.87969
93      14.0 15.020500 11.545955 18.49504
94      19.8 18.495691 15.012790 21.97859
95       5.8  8.463068  4.948793 11.97734
96       5.8  8.463068  4.948793 11.97734
97      14.0 13.487155  9.955616 17.01869
98       9.5  7.954714  4.486636 11.42279
99      14.9 14.402344 10.979352 17.82534
100     13.2 12.884962  9.471886 16.29804
Warning message:
In predict.lm(fullmodel, interval = "prediction") :
  predictions on current data refer to _future_ responses
```
