```r
#R code: Discussion 7.                    Sta108, Fall 2007, Utts

### Using predict() function

#Example: Grocery Retailer: Problem 6.9
Data = read.table("CH06PR09.txt")
names(Data) = c("Hours","Cases","Costs","Holiday")
Fit = lm(Hours~Cases+Costs+Holiday, data=Data)

Xh=data.frame(Cases=24500, Costs=7.40, Holiday=0)

#90% the confidence interval for mean response at Xh
predict(Fit, Xh, se.fit=TRUE, interval="confidence", level=0.90)
#90% prediction interval for new response at Xh
predict(Fit, Xh, se.fit=TRUE, interval="prediction", level=0.90)


### Polynomial regression

#center variables by subtracting corresponding mean of each
#method 1:
x1 = Data$Cases - mean(Data$Cases)
x2 = Data$Costs - mean(Data$Costs)
#method 2:
x1 = scale(Data$Cases, center=TRUE, scale=FALSE)
x2 = scale(Data$Costs, center=TRUE, scale=FALSE)
  #can be applied to the whole dataset at once, by providing Data in place of Data$Cases
    #ask me if interested how to use this function
  #'center' option subtracts the columns by their means
  #'scale' option divides the (centered) columns of Data by their root-mean-square

#get the Square terms (second order terms)
x1sq = x1^2
x2sq = x2^2
#get Interactions
x1x2 = x1 * x2
#add these new variables to our data table
Data = cbind( Data, x1, x2, x1sq, x2sq, x1x2 )

#fit a quadratic model (8.1) in x2
Quad = lm( Hours ~ x2 + x2sq, data=Data )
summary(Quad)

#plot fitted quadratic model
plot( Data$x2, Data$Hours, main="Polynomial Model", xlab="Costs(centered)", ylab="Hours",
pch=19 )
#create (x,y) points for quadratic model
x = seq(-3, 3, by=.1)
y = 4333.44 + 25.97*x + 39.21*x^2
#points(x,y, col="purple")
#OR: though matrix product
y = cbind(1, x, x^2) %*% Quad$coeff
lines( x, y , col="blue")

#fit a second order regression model
#method 1: use prepared variables
Poly = lm( Hours ~ x1 + x2 + x1sq + x2sq + x1x2, data=Data )
summary(Poly)
#method 2: fitting interactions within lm() function
Poly = lm( Hours ~ x1 + x2 + x1sq + x2sq + x1*x2, data=Data)
summary(Poly)


### Interactions with lm() function
```

```
x3 = Data$Holiday - mean(Data$Holiday)
Data = cbind(Data, x3)

#the following produce the same results in R:
Inter = lm( Hours ~ x1 + x2 + x3 + x1*x2 + x1*x3 + x2*x3, data=Data)
summary(Inter)
Inter = lm( Hours ~ x1 + x2 + x3 + x1:x2 + x1:x3 + x2:x3, data=Data)
summary(Inter)
Inter = lm( Hours ~ x1*x2 + x1*x3 + x2*x3, data=Data)
summary(Inter)
  #notation ':' in lm() function will fit only specified interaction
  #notation '*' in lm() function will fit specified interaction plus all lower order terms
of this interaction

#a different result would be this
Inter2 = lm( Hours ~ x1:x2 + x1:x3 + x2:x3, data=Data)
summary(Inter2)
Inter3way = lm( Hours ~ x1*x2*x3, data=Data)
summary(Inter3way)


### Indicator variables

#new example
Data = data.frame(Y = c(.24, .21, .22, .32, .51, .56, .56, .67, .89, .92),
                    X1 = c(0, 0, 0, 0, 0, 0, 1, 1, 1, 1),
                    X2 = c(1, 1, 1, 2, 2, 2, 3, 3, 4, 4),
                    X3 = c("low","low","low","low","med","med","med",
                           "high","high","high"))
                    #X1 has 2 levels
                    #X2 has 4 levels, quantitative categorical variables,
                    #X3 has 3 levels, qualitative categorical variables
Data


#Indicators In Practice:
#THE FOLLOWING CORRESPONDS TO THE CODING CALLED “OPTION 1” IN CLASS:
#1. If variable is {0,1} only, you do NOT need to set any additional contrast options
#just use the variable name by itself or factor()
Fit = lm(Y ~ factor(X1), data=Data)
summary( Fit )



#2. If variable is NOT in the form {0,1}, and you want the last level to be the base level:
#set options(contrasts()) to set the base level to be the LAST level of the factor, by
typing:
options(contrasts = c("contr.SAS", "contr.SAS"))
   #now, anytime factor() function is used, the base level will be the LAST level of the
factor
   #(highest Number, or highest Letter in the alphabet)
Fit = lm(Y ~ factor(X2) + factor(X3), data=Data)
summary( Fit )
   #alternatively, you may create a 'factor'/indicator variable and store it in your
dataset:
Data$X2ind = factor(Data$X2)
Data$X3ind = factor(Data$X3)
Data
Fit = lm(Y ~ X2ind + X3ind, data=Data)
summary( Fit )

#3. If the variable is categorical, i.e. {text},
#use option 'contr.treatment' with base level set to desired level number, by typing:
```

```r
Data$X3factor = C( factor(Data$X3), contr.treatment(n=3, base=2) )
   #this creates column of [X3factor] inside your dataset Data,
   #which represents indicator variables with base level: 'low'
   #here, base level is chosen from [ 'high', 'low', 'med' ] factor levels in alphabetical
order
Fit = lm(Y ~ X3factor, data=Data)
summary( Fit )


#The following part of code is for LEARNING about contrast function C().
#I advise you to run the code in R and see the results for yourself.
#You will rarely need to use these.

  #create a categorical variable (with levels) from a numerical column
  #can be used when only TWO levels/categories are present
  factor(Data$X1)
     #here, base level is FIRST level of factor, SECOND level will be fitted by model
  summary( lm(Y ~ factor(X1), data=Data) )

  #create indicators with constrain: sum to zero (OPTION 2 IN CLASS NOTES), see (8.44)
alternative coding
  C( factor(Data$X1), contr.sum )
  C( factor(Data$X2), contr.sum )
  C( factor(Data$X3), contr.sum )

  #indicators that contrasts each level with base level (specified by 'base')
   #by default, base level is the FIRST level, or FIRST letter in alphabet, seen in dataset:
  C( factor(Data$X1), contr.treatment )
  C( factor(Data$X2), contr.treatment )
  C( factor(Data$X3), contr.treatment )
   #to set baseline: to SECOND level seen in the dataset
  C( factor(Data$X3), contr.treatment(n=3, base=2) )
      #'n' is the total number of levels present in X
      #'base' is the specified baseline level
   #to create baseline to be the LAST level, do {one} of the following, see (8.35):
   #1: change 'base' in 'contr.treatment'
   #2: use 'contr.SAS
  C( factor(Data$X2), contr.treatment(n=4, base=4) )
  C( factor(Data$X3), contr.treatment(n=3, base=3) )
  C( factor(Data$X1), contr.SAS )
  C( factor(Data$X2), contr.SAS )
  C( factor(Data$X3), contr.SAS )

  #note, with qualitative variables, the order is chosen based on dictionary order
  #so: level1 = "high", level2 = "low", level3 = "med", because of alphabetical ordering
```