**Linear Regression Models**

* regmodel=lm(y~x) #fit a regression model
* summary(regmodel) #get results from fitting the regression model
* anova(regmodel) #get the ANOVA table for the regression fit
* plot(regmodel) #get four plots, including normal probability plot, of residuals
* fits=regmodel$fitted #store the fitted values in variable named "fits"
* resids=regmodel$residuals #store the residual values in a variable named "resids"
* beta1hat=regmodel$coeff[2] #assign the slope coefficient to the name "beta1hat"
* confint(regmodel) #CIs for all parameters

Prediction:

* predict.lm(regmodel, interval="confidence") #make prediction and give confidence interval for the mean response.
* predict.lm(regmodel, interval="prediction") #make prediction and give prediction interval for the individual response.
* newx=data.frame(X=4) #create a new data frame with one new x\* value of 4
* predict.lm(regmodel, newx, interval="confidence") #get a CI for the mean response at the value x\*
* predict.lm(regmodel, newx, interval="prediction") #get a CI for individual at the value x\*
* Prediction Band: We need the Working-Hotelling multiplier W.

CI<-predict(copier.model,newx,se.fit=TRUE); W<-sqrt(2\*qf(0.90,2,n-2)); Band<-cbind(CI$fit - W \* CI$se.fit, CI$fit + W \* CI$se.fit )

Tests for homogeneity of variance

* library(lmtest); bptest(regmodel) #get the Breusch-Pagan test (lmtest package must be installed)
* levene.test(Y, groupvariable) #get the Levene test (lawstat package must be installed)

Tests for normality

* + library(stats); shapiro.test(residuals)

Lack-of-fit test

* + Reduced=lm(y~x) #fit reduced model
  + Full=lm(y~0+as.factor(x)) #fit full model
  + anova(Reduced, Full) #get lack-of-fit test

BoxCox transformations

* boxcox(regmodel) #evaluate possible Box-Cox transformations (MASS package must be installed)

Lowess: Apply loess smoothing using the default span value of 0.8. You can change the curve by changing the span value.

* y.loess <- loess(y ~ x, span=0.8, data.frame(x=X, y=Y),data=mydata)# Compute loess smoothed values for all points along the curve
* y.predict <- predict(y.loess, data.frame(x=X))
* plot(Y~X); lines(X,y.predict) # Plots the curve.

Model Selection

* library(leaps) #load the leaps package
* allmods = regsubsets(y~x1+x2+x3+x4, nbest=2, data=mydata) #(leaps package must be loaded), identify best two models for 1, 2, 3 predictors
* summary(allmods) # get summary of best subsets
* summary(allmods)$adjr2 #adjusted R^2 for some models
* summary(allmods)$cp #Cp for some models
* plot(allmods, scale="adjr2") # plot that identifies models
* plot(allmods, scale="Cp") # plot that identifies models
* fullmodel=lm(y~., data=mydata) # regress y on everything in mydata
* MSE=(summary(fullmodel)$sigma)^2 # store MSE for the full model
* extractAIC(lm(y~x1+x2+x3), scale=MSE) #get Cp (equivalent to AIC)
* step(fullmodel, scale=MSE, direction="backward") #backward elimination
* none(lm(y~1) #regress y on the constant only
* step(none, scope=list(upper=fullmodel), scale=MSE) #use Cp in stepwise regression

Diagnostics

* sresids=rstandard(regmodel) #store the standardized residuals in a variable named "sresids"
* standresid=stdres(regmodel) #store the standardized residuals in a variable named "standresids"
* stud.del.resids=rstudent(regmodel) #store the studentized deleted residuals in a variable named "stud.del.resids"
* hatvalues(regmodel) #get the leverage values (hi)
* cooks.distance(regmodel) #get Cook's distance
* dfbetas(regmodel) #print all dfbetas
* dfbetas(regmodel)[4,1] #dfbeta for case 4, first coefficient (i.e., b\_0)
* dffits(regmodel) [4] #dffits for case 4
* influence(regmodel) #various influence statistics, including hat values and dfbeta (not dfbetas) values
* library(car) #load the package car
  + vif(regmodel) #variance inflation factors
  + avPlots(regmodel) #added variable plots