Regression analysis is a staple of machine learning (you mean it's not just for statistics?). It is often the case though that individuals without a strong background in statistics will take the results they find in their regression (OLS) outputs and run with them without examining the assumptions behind their models. Note, if you have a good background in statistics, the rest of this will probably bore you.

For linear models the first of these assumptions is obviously that the coefficients are in fact linear. One of the first assumptions is a lack of multicollinearity (modelling something that is roughly linear with something else in the model – education and income for example - this is the one that most people get in an intro course on how to prepare their data but there are several additional issues to consider).Other key assumptions are no autocorrelation, a lack of correlation with the error terms (no endogeneity), and a fairly homoscedastic distribution.

A side note: For ordered probit and ordered logit models assumptions include the independence of irrelevant alternatives and that the distribution of observations is roughly parallel (the distribution of 1 is roughly parallel to the distribution of 2 which roughly parallel to the distribution of 3 and so on). I will deal with these at a later date though – for now let's just look at the linear models.

A multiple regression or ordered model run in R, STATA, Python, SAS, etc. will certainly give you an output with coefficients, standard errors and therefore p-values (a topic for another day); if the assumptions of your model are violated then your model will not be correct. Below I have described some of the techniques for addressing the assumptions of both linear and ordered models, what they mean (in plane language) and the relevant commands for calling them up in R.

To do this you will need the following R packages

```
>library(MASS)

>library(lmtest)

>library(prais)

>library(systemfit)

>library(stats)
```

Note: A good first step before performing any further analysis is to examine the correlation matrix of your data.

Homoscedasticity

The assumption of homoscedasticity is simply that at each value of X, the Y value has roughly the same variance. Imagine that you have a piece of bread (it is square so it makes for a good example) and you want to spread peanut butter across the bread. If you spread it across in one even stroke and the bread is covered evenly then this would essentially be homoscedasticity. If, however, you spread it and a lot more wound up on the left hand side than the right hand side this would be heteroscedastic. In terms of application this means that even though we have a value for b of X the value would not truly reflect X for all of Y since the variation is not approximately the same for all X and Y variables. This is a serious problem, as you can probably surmise, since it means that the standard errors in your model are biased and therefore no accurate.

Testing

There are a number of tests you can run to check for Homoscedasticity (Whites Test, Breusch-Pagan, and Goldfeld-Quandt – best for group data). My preferred method is the Breusch-Pagan. This test takes the reisiduals for each variable in the model and squares them (thus normalizing) and then models them as explanatory variables of Y. It then performs a Chi-Square test with the sum of squares for this new model and the sum of squares of the initial (your original) model. If the Chi-Square value is significant then it indicates that the model is probably not homoscedastic, e.g. the variance is unequal and the standard errors are biased..

```
>data<-read.csv(file = "~/some.data.csv")

>fit1 <- lm(Y~X1+X2+X3, data = data)

>bptest(fit1)
```

Graphically, you would want to plot the residuals over the fitted coefficients.

```
>plot(fitted(fit1), residuals(fit1))
```

Addressing

One means of addressing heteroscedasticity in a model is to weight the regression – typically by the variance in the variable or variables causing the heteroscedasticity. In practical terms, these variables have more leverage than they should in the model, the weighting holds them down so that they are approximately normal (so divide by the variance).

There are multiple means of weighting regressions to deal with heteroscedasticity. Below I will only address two: calculating fitted values from absolute residuals and standard variance weighting. Note that for Poisson or Binomial data the Box-Cox transformation would be preferred (a different topic though).

Approach 1:

>fit1<-lm(Y~X1+X2+X2, data=data) # assume that we see from the BP test that we have heteroscedasticity

> wts<-1/fitted(lm(abs(residuals(fit1))~fitted(fit1)))^2 #we are squaring the difference between each point X at its corresponding Y value thus forcing the data into a more normal distribution (for our model at least)

>fit2<-lm(Y~X1+X2+X3, data=data, weights=wts)

Approach 2:

>fit1<-lm(Y~X1+X2+X2, data=data) # assume that we see from the BP test that we have heteroscedasticity

>fit2<-lm(Y~X1+X2+X3, data=data, weights = 1/SD^2)

Autocorrelation

Things are correlated over space and time. So things that are in close proximity to each other tend to be more alike and what happened previously is correlated with what is happening now. For example, we see people of similar incomes living proximity to each other (spatial correlation) and if you are a waiter and you work 40 hours per week then your salary next week will be very similar to your salary this week (serial correlation). The presence of autocorrelation affects both your coefficients and your error terms since we are seeing things that are measured from the same source but had not prior knowledge of it. In general autocorrelation can be attributed to model design or sample selection issues. For example if I wanted to model the effects of political views on income over a three year period and I drop all of the three years worth of data into my model then the income from year 1 will be correlated with year 2 which will in turn be correlated with year 3.

Testing

The standard approach for detecting autocorrelation is the Durbin-Watson test. The DW Test as it is commonly called is essentially just the sum of squared errors divided by the ordered sum or squared differences (difference in error of observation 2 minus observation1, difference in observation 3 minus observation 2, etc. and then squared). Therefore it tests for correlation of lag 1 predictors (same predictor from the prior time period which can also be used as spatial properties), there are also implementations for panel data.

For reasons I won't get into here, the DW test statistic is always between 0 and 4. A lack of autocorrelation is indicated by values at or around 2 (generally between 1.5 and 2.5 is acceptable depending on sample size). Values less than 1.5 typically indicate negative autocorrelation and greater than 2.5 typically indicate positive autocorrelation.

Note that some consider the DW test to be a bit archaic, but it can still provide useful information about possible errors in the model.

```
>fit1<-lm(Y~X1+X2+X3, data=data)

>dwtest(fit1)
```

Addressing

Since the DW test looks for lag 1 correlations one of the simplest means of addressing this is by differencing the data (essentially subtracting period 1 observations from period 2 and so on). Rather than going through and doing this manually, we can use what is known as a Prais-Winsten transformation. Note that because we are differencing, the Prais-Winsten replaces the values of X and Y at time with Y*SQRT(1-rho^2) and X*SQRT(1-rho^2) where rho is the autocorrelation coefficient.

```
>fit1<-lm(Y~X1+X2+X3, data=data)
```

```
>dwtest(fit1) #assume the value comes back significant. # Calculate Autocorrelation and Partial Autocorrelation Coefficients
```

```
>acf(resid(ols),lag.max = 5,plot = FALSE)
```

```
>pacf(resid(ols),lag.max = 5,plot = FALSE)
```

The Prais-Winsten regression will give you coefficients that are free (generally although specific cases get wonky).

Endogeneity

A killer of multiple regression models is endogeneity, meaning that an explanatory variable is correlated with the error term. This can be caused by omitted variables (we are not measuring something we should), simultaneity (something is occurring along with an explanatory variable that is not being measured), or autocorrelative loops (something that X does to Y in turn causes Y to do something to X). For example, if we wanted to predict the price of a widget using quantity produced then we would have endogeneity since producers change their prices in response to demand and consumers alter their purchasing behavior in response to price.

Testing

A quick way of testing for endogeneity is to do a regression with the residuals. If the model using the residuals is significantly different than 0 then there is likely autocorrelation. Note that this does not get into where the source of that autocorrelation may be.

Performing this analysis is fairly straight forward since all we are doing is regression with residuals. Of course this also means that you would want to have an idea of what may be causing the endogeneity in the first place. This means you need to have a sound understanding of the thing you are trying to predict. To determine if the difference is truly different than 0 we use the Likelihood Ratio Test - it approximates the Wald test and essentially tells you if one model explains a significant amount of variation versus another model. This simply examines if the log Likelihood of the new model (with the residuals) is significantly different (explaining more variation) than the initial model. Still, if you find significance at all in the regression with residuals it is probably a good idea to go back and explore your model more carefully.

```
>fit1<-lm(Y~X1+X2+X3, data)

>e1<-residuals(fit1)

>fit.r1<-lm(Y~e1, data=data)

>lrtest(fit.r1, fit1)
```

Addressing

One way of addressing endogeneity is through instrumental variable regression or two-stage least squares. This assumes that we have identified the variable causing the endogeneity, to do this we would need to look at the correlation matrix of the data (which you should always do at the beginning anyway)

```
>cor(data) #assume we see that cor(X1, X2) = 0.3 and cor(X2,X3)=0.7 then X3 is a strong instrumental variable for X2; and cor(X1, X3) = 0.001 then X1 and X3 have little relationship whatsoever
```

This means that either X2 and X3 are measuring something similar, or that something in X2 is occurring that we are not seeing that has an effect on X3 (or vice versa, this is where understanding your model comes into play).

```
>fit1<-lm(Y~X1+X2+X3, data=data)

>fit2<-lm(X2~X1+X3, data=data)

>Xhat<-fitted.values(fit2)

>fit3<-lm(Y~X1+X2+Xhat, data=data)
```