**Categorical Regression I**

So in the Hypothesis test file, we saw how to calculate a p-value for comparing the means of two different groups. And in the Regression file, we saw how to calculate a p-value for a regression curve. And how to compare two different regression curves. Now we’re going to kind of combine the two concepts together. For instance, we’d like to be able take a regression curve for one set of data, say group A, and a regression curve for another set of data, say group B, and calculate a p-value for whether the regressions are meaningfully different. Or another way to say it, is there any predictive power in classifying data into one of those two groups, or should we just treat them as all the same group? So this would be like doing a regression fit over independent variables, some of which are continuous, and some of which are discreet (the group index).

**Categorical Regression**

First we’ll revisit the case of simply comparing the means of different groups, and seeing if the means are different in a statistical sense. This is called a T-test. But we will extend the number of groups to be some general G. Now let’s say we have n datapoints, yi. These points will be collected from either group A, B, C, …, or G. There are three groups illustrated below: A, B, C. To kind of make a connection with the regression stuff, I’m going to label them on a set of ‘group’ axes, as group coordinate is an independent variable of sorts, and will play a similar role as the independent variables in the regression case.

Diagram

Description automatically generated

And the data points are, say:

Chart, scatter chart, box and whisker chart

Description automatically generated

Insofar as we are interested in modeling the means of our groups, we can fit the data with an equation such as follows.



where εi is a normally distributed random variable with mean 0 and std. σ. Now the coordinate (Ai, Bi, Ci) for each data point will be (1,0,0) if it is in group A, (0,1,0) if it is in group B, and (0,0,1) if it is in group C. Thus the model equation for a data point in group A will be Yi = vA + εi; and for a datapoint in group B will be Yi = vB + εi, and for a data point in group C will be Yi = vC + εi. If we think about it, we can tell what the fit parameters vA,B,C will be in this model. They will just be the averages of the data points in their respective groups: A,B,C. But supposing we didn’t know that, we could work it out using the framework expounded in the mult-variable linear regression file. We’d say,



We can put this in the following matrix form,



where,



In this context, X is called the **design matrix**. We’ll recall from the Regression file’s appendix that the best fit parameters are:



In this case, that works out to:



It makes sense that the best single-parameter fit we can make to each set of data is it’s average. So our discrete regression equation is:



This ‘best-fit’ equation minimizes the SSEf, given by:



which, for what’s it’s worth, reduces to,



where SSEWG designates the sum of the squared errors within groups. We may view the fit parameters with the data points below:

Chart, box and whisker chart

Description automatically generated

where is the overall mean. Most of the results which we discussed in the previous files carry over here. So moving on,

**Point Estimators for vi**

As before, we’ll recognize that **v** is really a random variable **V**, given by:



where **Y** is the random variable at the top of the page. And we found in the multiple regression file that:



Of course <Vi> is just the means of the proposed groups, as we saw in our example above. And no doubt <Vi>2 is just the variance of the means within those groups. Well, let’s just examine our illustrative example case,



And so the inverse is:



in which case,



So indeed these are the variances of the sub-group averages, generically, σ2/ngroup

And finally, we found a good point estimator for σ2, the variance of the ε random variable, was:



where n is the number of data points, and f is the number of d.o.f. of our model. And so we have:



From our little example above, we know these will come to:



This makes sense. And as for σ itself, in our example,



So if we know know σ2 for sure, then we can say the ZVi guy,



follows a unit normal distribution. And if we don’t know it for sure (not sure how we would), then we can say,



follows a Student’s T distribution with ν = n-f d.o.f. If ν > 30 or so, then this is just a normal distribution for all intents.

**Hypothesis Testing of Parameters**

If we are running an experiment, and get some particular values, <**V**i> = vi, for our linear regression, we may wish to see whether our results invalidate someone else’s linear regression values, or the standard/accepted linear regression values, vi\*. We can do a hypothesis test for this. We know how Vi are distributed, according to our experiments, and so I guess we could form the Z-statistics,



and then calculate the p-values, the probabilities that values at least as extreme as vi\* occur. So we’d calculate,



p(z) would be the aforementioned Student’s T distribution of course.

**Confidence Intervals for Parameters**

We can also calculate confidence intervals for the statistics Vi. Just like we’ve done with other statistics, we’d say, at the 1-α confidence level,



And the zα value would be for a Student’s T distribution, I imagine with ν = n-f d.o.f.

**Goodness of Fit: R2 value**

We can define a goodness of fit just as before.



where fi are the regression curve data points, which one could say is a random variable since it depends in known way (if have formula) on the random variable Yi, the data points we’re trying to fit. SSEf = sum of squares about f, and SSEm = sum of squares about mean. This is actually a general formula that applies to all curve fits. Apropos the fraction, the numerator is the ‘variation around the fit, or f’, and the denominator is the ‘variation around the mean’). Can see R2 = 1 if there is no variation around the fit, i.e., if the curve/fit fits the data exactly. On the other hand, if the curve fits the data no better than the mean, then we get R2 = 0. If you have an R2 = 0.75, then you can say that 75% of the variation of the data is explained/predicted by the regression curve f. As usual, we can interpret R2 as:



And last,

**Hypothesis Testing Different Regression Models**

In the context of a T-test, we’d be interested in knowing whether splitting the data into the groups A, B, C, …, G and computing separate means is a meaningful procedure. How would we know if instead, all the groups really came from the same distribution, with mean μ, and we’re just seeing noise? So our H0 would be:

H0 = assumption that the data is described by model **Y** = μ + ε, which has f = 1 degrees of freedom.

Note SSEm would be the sum of the square errors for this model.Then let’s compare to another model with f = G (the number of groups) degrees of freedom, fitting **Y** = X**v**(f) + ε, where v(f) is basically a column vector of all the different group μ’s. And let SSEf be its sum of squared errors. We would anticipate this to be smaller of course, i.e., SSEm > SSEf. The alternative hypothesis would be:

HA = assumption that at least one of the extra f - 1 parameters (μ’s) in the new model Y = Xv(f) + ε is different from the overall μ.

Turns out SSEf follows a known probability distribution, given the Null Hypothesis H0. Well we can form a test statistic,



and n = number of data points, and f = G = # of groups, again. We might recall SSEf/(n-f) is just the point estimator for σ2 in the f-model. Might interpret Z as:



Turns out this follows an F-distribution (B is the β function).



which is the probability density of getting an Z-value of x, given the null hypothesis is true. So we can calculate a p-value,

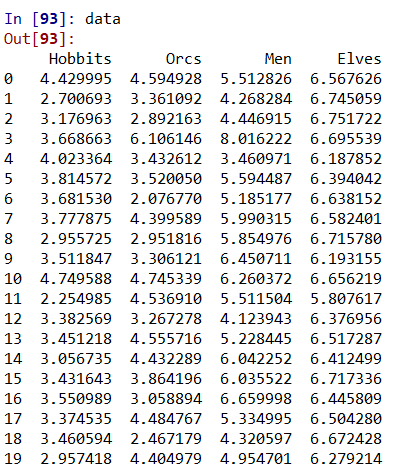
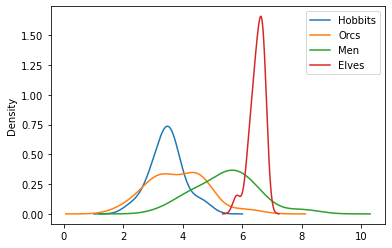


which would be the probability that we’d get an Z-statistic Z\* or higher, out of the new model, if the Null hypothesis were true. So if the f model has true explanatory power, then we should find Z\* >> 0 and the p-value should be small (less than 0.05 at the 95% significance level). But note for instance that if f = n, i.e., if the number of fit parameters (groups in this case) equals the number of data points (allowing an exact fit of f to our data points), then Z\* = 0, and so our p-value would be 1 I think. And this would mean that our extra parameters are meaningless.

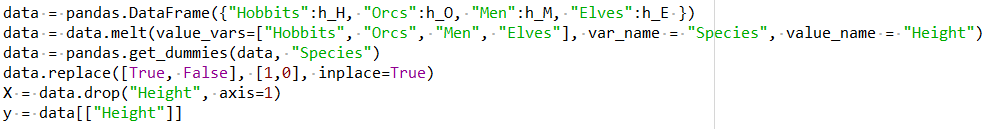
Might recall that in the Hypothesis Testing file, we used we used T-tests to analyze the probability that two samples, hypothetically drawn from the same population would differ in their means by as much as they do. Our F-test above seems to accomplish much the same thing, but can be applied to more than just two samples. And instead of comparing the means of the samples, it compares the variances (SSEf) of the samples, estimating the probability that the variance of the fit should be as small as it is if the samples all come from the same population. Perhaps this is why it’s called **ANOVA** (analysis of variance). Though the F-test is more general, it might not be as rigorous? It kind of presumes the same σ2 for each sample (they can have different SY’s only through different n’s), whereas the T-test allows for different σ2’s.

**Example**

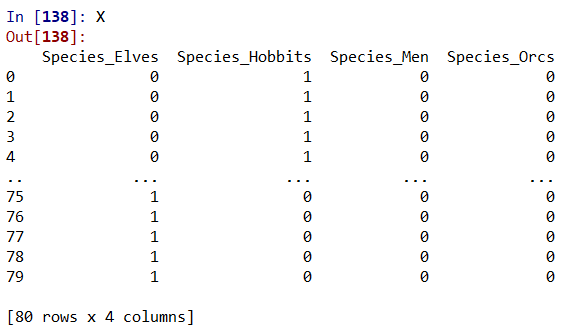
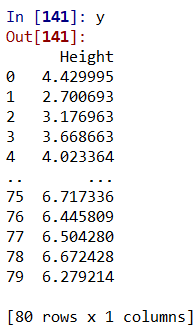
I made up some heights of four different species of middle earth denizens.

To apply our formalism, though, we need rewrite the dataframe as just a list of heights and species. So I used the pandas melt function to change this to a column of heights and a column of species. The one-hot-encoded the species. Then separated the species columns and height column.



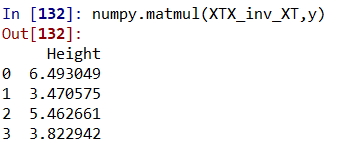
We get:

And we’d like to see if we can indeed distinguish four categories on the basis of height. So first, the fit parameters would be:



Working this out, using numpy, we find:



So,



I plotted these predictions against the data points, below,



Now these numbers are indeed just the averages of the respective, prospective, categories. Next we’d like to have a confidence interval for these guys. To work that out, first we need an estimate for σ2, the variance of the data points about the fit. This is σ2 ≈ SY2. We can get SSEf from subtracting the fit height values from the actual height values, squaring each, and summing. We find:



Okay, now using this, we can get the variances in the parameters via:



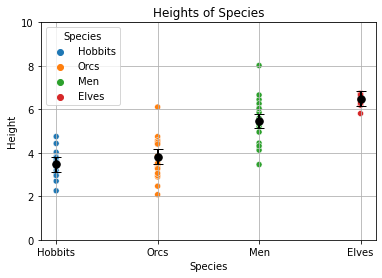
We find for the variances of the averages,



So 95% confidence intervals for our parameters are (using Student’s T distribution zα/2 with n – f = 80 – 4 = 76 d.o.f., so it’s essentially a normal distribution):



We get the same errorbars for every species, because the statistical model of categorical regression (all that stuff above) assumes this from the outset – that σ is same for everybody. I plotted the error bars below,



These error bars may seem small, but remember they are error bars on the *average* value, not on the actual sample values. So they give us an interval within which repeated samples’ *averages* are expected to lie. Let’s get the R2 coefficient. This is:



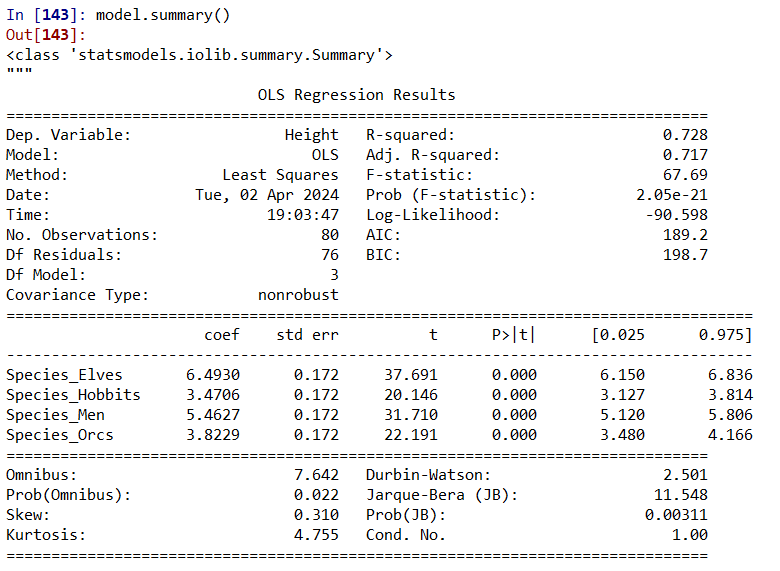
And finally, let’s do the F-statistic thing. We’ll take the Null Hypothesis to say that we can just characterize the data by a single mean, f0. And then we’ll take the Alternative Hypothesis to be that at least one of our f = 4 model’s extra parameters, i.e. hE, hH, hM, hO, is required. So then we’d form,



Now we can use this to calculate the probability that our Null hypothesis is true.



But this is not what statsmodels gets!? I think something’s wrong with scipy b/c no matter what Z\* past 67.6 I fill in, I get the same number, though it should get smaller. Well, so we can see the likelihood that our heights are just randomly distributed about the mean is astronomically low. We can get a lot of these parameters from the *statsmodels* library in Python. Here’s its readout below. We can see the R2, the regression coefficients, the 95% confidence intervals, and the F-statistic thing. Looks it just does the F-statistic where we’re comparing to the mean model, where f0 = 1. I guess that makes sense.



**Appendix 1**

Gonna take a look at another way to write the regression equation. It’s more common to do it this way. Let our fit equation be:



where vA,B,C are some fitting parameters that have to do with the groups. Written this way, we eliminate an extraneous d.o.f., A, I guess. Continuing with my example above, we’d want to fit the following,



We can put this in the following matrix form,



where,



And the best fit parameters will be given by:



So our regression equation is:



And this makes sense too. When A = 1 (so B = C = 0) we get A, and we get the other averages when A = 0 and B = 1 or C = 1. And the SSEf would be:



So same as last time.

**Appendix 2**

So let’s define some things. The total variability of the data is SSEm (sum squared error about mean), defined below.



where we clearly define the squared error within groups, and squared error between groups as:



and have that:



For visualization’s sake, consider picture with the means displayed,

Chart, box and whisker chart

Description automatically generated

and we may roughly visualize the three SSE’s as (taking top red datapoint for example):

Chart

Description automatically generated

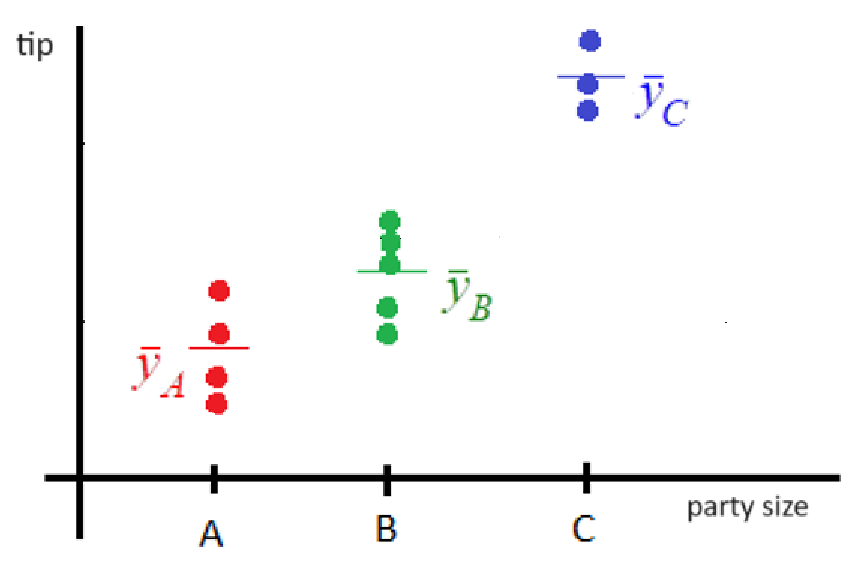
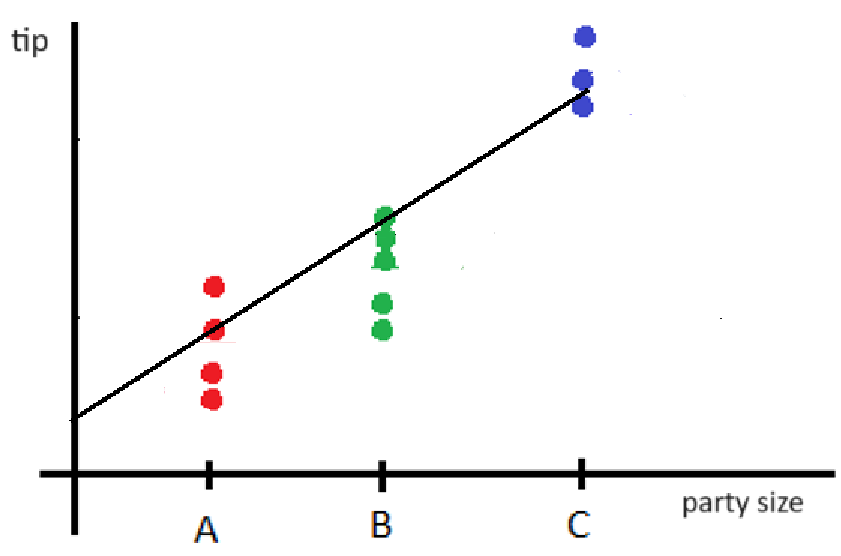
So our equation says the variability (SSEm) of the data points about = the sum of the variabilities (SSEWG) of each group’s data about its own mean + the variabilities (SSEBG) of the group’s means about the overall mean. We would say that SSEm is the variability unexplained by , or, just, the variability. And SSEWG is the variability unexplained by group means, or one might say, by the regression curve/fit (even though it’s just three data points). And I guess could say SSEBG is just the variability of the regression fit, whatever. Anyway, in terms of these definitions, note we can write R2 and our Z statistic as:



and again f = G (the number of groups) in this context.

**Appendix 3**

General question. We could make our categorical variables numbers, in some instances. Say we’re looking at the tips dataset. We can treat the dinner party size (A = 1, B = 2, C = 3, D = 4, E = 5, etc.) as a categorical variable or a numerical variable. If the former, then we’d model the points with the separate means of each category. If we treat as numerical, then we’d have to run a regression line through the data.

The categorical approach introduces greater d.o.f. (3 in this simple example) than the regression approach (2 in this example). I guess in general,it’d be better to treat as categorical. Practically, we’d do this with one-hot-encoding.