MSIS 5223: Tutorial 9 – Regression in R

If you have Crawley's book *The R Book* you should read chapters 10 and 20.

Instructions

The purpose of this assignment is to help you become familiar with using R. In this assignment you will become familiar with validating the assumptions of regression, learn how to perform linear regression, multiple regression, polynomial regression, and logistic regression.

1. Assumption Validation

Prior to engaging in a regression analysis, you need to ensure your data meets five criteria. These criteria are assumptions and regression requires all of them to be valid:

- Linearity: linear relationship exists between x and y where $y = mx + b$
- Correlation: the independent variables (i.e. explanatory or predictor variables) are not related to each other
- Homoscedasticity: residuals exhibit a constant variance
- Independence: residuals are not related to each other
- Normality: residuals exhibit a normal distribution

The first assumption is fairly easy to check, but requires experience over time to make good judgment. Typically, a scatterplot or residual plot is used to assess linearity. A scatterplot is simply plotting the target variable on the y-axis and the predictor on the x-axis and is executed by typing in

plot(x, y)

where x is the predictor and y is the target. Using the ozone.data.txt dataset, for example, to check the linearity of *Radiation* against *Ozone Concentration* you would type in the following and receive the scatterplot shown below:

```
> plot(ozone_data$rad, ozone_data$ozone)#Check for linearity
```

Figure 1 Scatterplot Code for Linearity Assessment

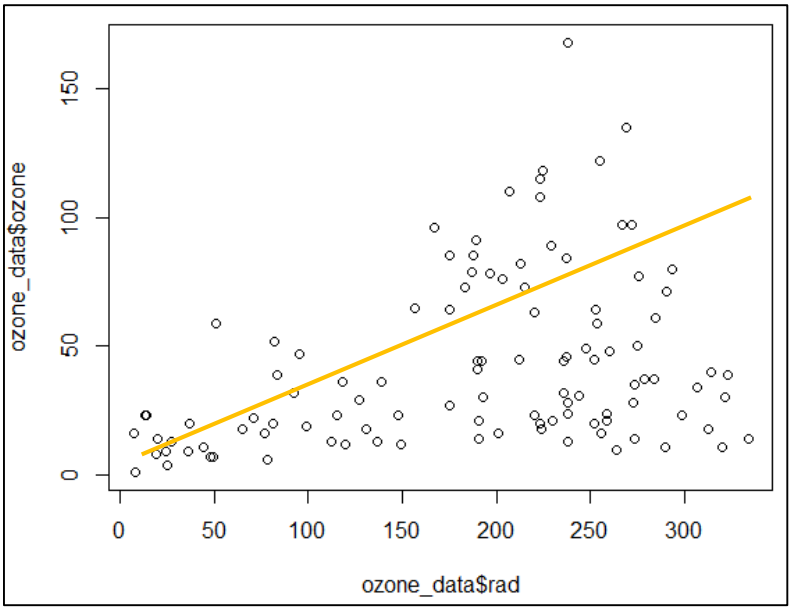

Figure 2 Scatterplot of Radiation and Ozone

Figure 2 is the scatterplot of *Radiation* and *Ozone Concentration*. An orange line is added to the plot to make it easier for you to assess the linearity. This is fairly linear and doesn't exhibit any curvature or other anomalies that would raise a red flag.

You may be thinking that this data doesn't look very linear to you. This concern is sometimes raised because the data on the right-side of the plot has a greater spread vertically than the data on the left-side. Linearity isn't concerned with the spread of the data. It merely is trying to fit a straight line to the data. This idea of spread is related to the third assumption, that of homoscedasticity. More on that later.

R provides an excellent function that produces multiple scatterplots simultaneously. If you have multiple variables and would like to create a single figure with all plots (instead of creating a single plot for all variables), use the following function:

```
pairs(data, panel=panel.smooth)
```

where data is your dataframe containing your variables. If your dataframe is hotel_data, then substitute data for hotel_data. The dataframe I created for the ozone dataset is called ozone_data. Using the pairs() function, I would type in the following:

```
> pairs(ozone_data, panel=panel.smooth)
```

Figure 3 Creating Mulitple Scatterplots

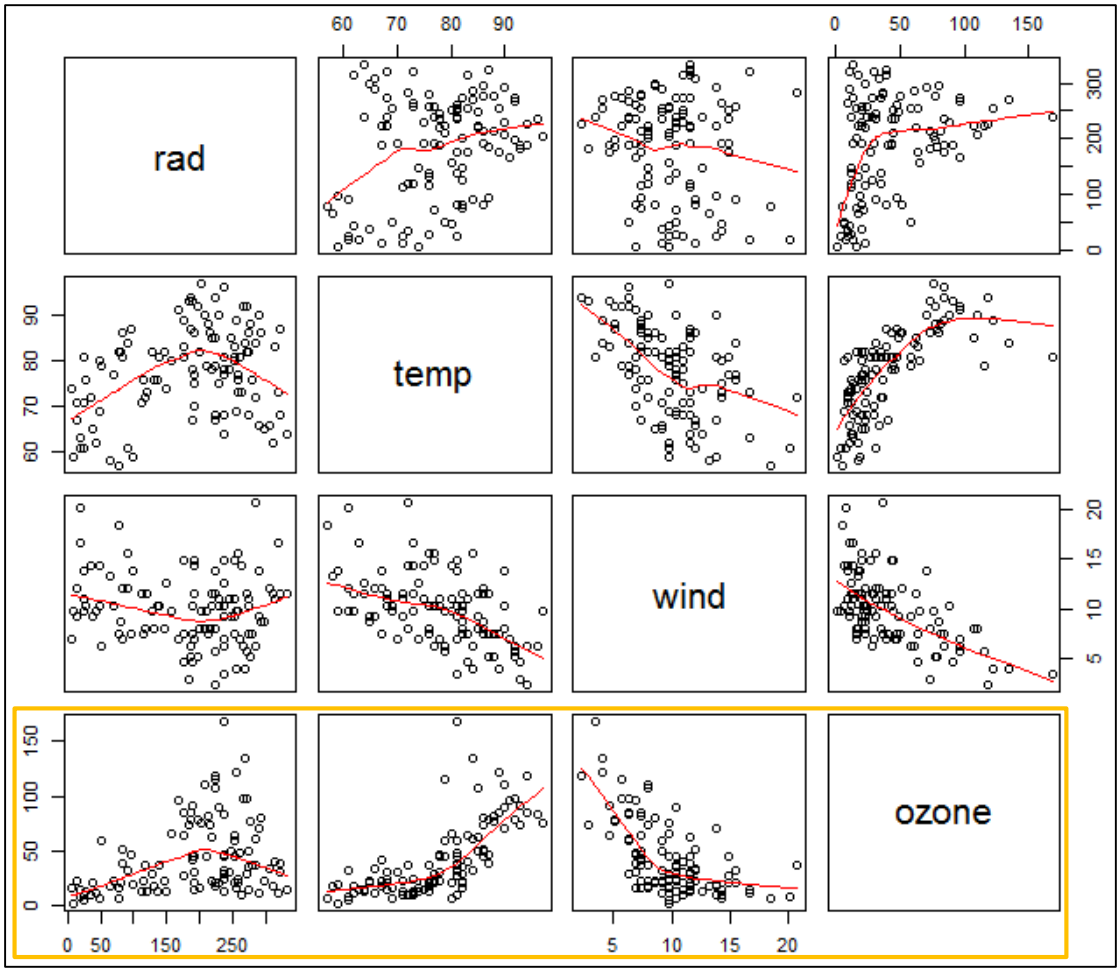

Figure 4 Scatterplots Created from Pairs()

The result produces every possible combination of scatterplot possible in your dataset. The ozone dataset contains four variables and results in producing twelve plots. In essence, however, only six are created because those in the top-right are merely the inverse of those in the bottom-left.

To read the plot, find the target variable; this is your y-coordinate. Next, look at the row containing the target variable. It is highlighted in orange in Figure 4. Looking from left to right, the order of the plots is *radiation-ozone*, *temperature-ozone*, and *wind-ozone*. Compare the first one, *radiation-ozone*, with the results in Figure 2. The plots are identical. *Temperature* and *wind* appear to have slight curves in the data and may not be linear enough. It is possible to use a transformation on these variables to straighten them out.

The next assumption to check in regression is that of multicollinearity. That is, your independent variables (i.e. predictor variables) are not related to each other. Your predictor variables should not influence each other. This type of relationship is alright between a predictor variable and your target variable (indeed, it should be highly related!). Two methods are typically used to assess collinearity (two related variables) or multicollinearity (three or more related variables).

The first method is correlation analysis. This is the more subjective assessment of the two methods. A correlation matrix typically provides the Pearson correlations among your variables. The code, along with the output, is presented below.

```
> ozone_corr = cor(ozone_data)#pearson correlation
> ozone_corr
              rad       temp       wind      ozone
rad     1.0000000  0.2940876 -0.1273656  0.3483417
temp    0.2940876  1.0000000 -0.4971459  0.6985414
wind   -0.1273656 -0.4971459  1.0000000 -0.6129508
ozone   0.3483417  0.6985414 -0.6129508  1.0000000
```

Figure 5 Correlation Matrix for Ozone Data

The values range from -1.0 to 1.0 with an absolute value of 1.0 representing an exact relationship. Think of it as the extent to which the data points move together on a scatterplot. Looking at the names of the variables in the far left column, find ozone. Moving from left to right, the correlation values of each variable are listed. *Radiation* has a correlation of 34.8% with *Ozone Concentration*. This is a decent level of correlation and is good.

Returning to the assumption of multicollinearity, the correlation matrix should be used to look at the relationships among the independent variables. Looking at Figure 5, this would include the intersection of rad-temp, rad-wind, and temp-wind. Wind and temperature have a fairly high value of -49.7%. This isn't a cause for alarm. Typically, any correlation greater than the absolute value of 70.0% should be a cause for concern.

An important feature of the correlation matrix is the diagonal, where all the values are 100%. All of these values are at the intersection of a variable with itself. That is, wind-wind, rad-rad-, temp-temp, and ozone-ozone. It shouldn't be surprising that a variable is highly correlated with itself.

This only becomes worrisome if two different variables have a correlation of 100%. This would indicate they have exactly the same values associated at each row. To illustrate this point, the code in R was used to create a new dataframe containing three variables *x*, *y*, and *z*.

```
> x = c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20)
> y = c(1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17,18,19,20)
> z = c(65,1,5,3,18,61,3,61,6,1,6,61,61,6,6,87,651,31,84,1)
> test_data = data.frame(x,y,z)
```

Figure 6 Creation of Example Dataframe

The data is shown to the right in Figure 7. The first two columns of data, *x* and *y*, have the exact same values. The third column, *z*, has a random set of numbers I pressed into my keypad. A correlation matrix is computed and the results are shown below in Figure 8. Not surprisingly, the variables *x* and *y* have a correlation of 100%. Variable *z*, on the other hand, has a correlation of 29.6% (pretty high, considering I randomly types in numbers). What this means is, the variables *x* and *y* are truly the same variable. Look at Figure 7 again. Can you honestly say there is a difference between them other than the fact that one is named *x* and one is named *y*? The data is exactly the same. Basically, it is measuring the same thing. Go ahead and plot this dataset using the pairs() function and confirm this.

```
> test_data
   x  y  z
1  1  1 65
2  2  2  1
3  3  3  5
4  4  4  3
5  5  5 18
6  6  6 61
7  7  7  3
8  8  8 61
9  9  9  6
10 10 10  1
```

Figure 7 First 10 Rows of Data

```
> cor(test_data)
          x         y         z
x 1.0000000 1.0000000 0.2957994
y 1.0000000 1.0000000 0.2957994
z 0.2957994 0.2957994 1.0000000
```

Figure 8 Correlation Matrix

This is important for the assumption. The closer a correlation score between two variables comes to 100%, the more likely they are the same variable. If you had done a principal components analysis with factor analysis, this problem with variables *x* and *y* wouldn't have occurred. Factor analysis would have shown that *x* and *y* are really the same thing.

The cutoff criteria of a correlation value of 70% is a rule-of-thumb. Some statisticians recommend a cutoff of 90%. Still, others say a lower value is more

conservative. The default correlation function in R does not provide p-values. To obtain p-values, use the rcorr() function from the library Hmisc. It should be noted, the library Hmisc overrides some of the functions used in the library pysch; only use Hmisc after you finish using pysch. Another important note is the rcorr() function accepts matrices, not dataframes. You will need to convert your dataframe into a matrix. See the code below for the process and output.

```
> rcorr(as.matrix(ozone_data))
        rad  temp  wind ozone
rad    1.00  0.29 -0.13  0.35
temp   0.29  1.00 -0.50  0.70
wind  -0.13 -0.50  1.00 -0.61
ozone  0.35  0.70 -0.61  1.00

n= 111


P
      rad    temp   wind   ozone
rad          0.0017 0.1828 0.0002
temp  0.0017        0.0000 0.0000
wind  0.1828 0.0000        0.0000
ozone 0.0002 0.0000 0.0000
```

Figure 9 Correlations with P-Values

An alternative method to assessing multicollinearity is using a variance inflation factor, or VIF for short. Without getting too technical, a VIF is calculated for each variable after a regression is run and provides an index of the extent to which multicollinearity exists. To calculate the VIF scores, use the library car. See the example below. The regression object ozone_reg2 was created prior to running vif() (shown later in section 3, Multiple Regression) using the lm() function. Again, the vif() function is run only after you have run a regression model.

```
> vif(ozone_reg2)
 ozone_data$rad ozone_data$wind ozone_data$temp
       1.095241        1.328979        1.431201
```

Figure 10 VIF Scores for Ozone Regression

A lower VIF score is desirable. By conservative standards, values that are greater than 5.0 are considered multicollinearity; a less conservative approach indicates values greater than 10.0 have multicollinearity. I go with the lower, restrictive number of 5.0 because it is less debatable. In the results above, all three independent variables have low

enough values to rule out multicollinearity. The assumption of no multicollinearity is valid for this model and data.

The third assumption assessed for regression is homoscedasticity, or constant variance. That is, the spread of the data over *x* and *y* is consistent and doesn't change. When the data is not evenly distributed across the range of data, the data is said to have heteroscedasticity. Only a single plot is produced, plotting the residual values on the *y*-axis and the predicted values (sometimes called the fitted values) on the *x*-axis. Just like the VIF scores, this can only be produced after a regression model is built. The code is simple as shown below

```
plot(regression_object)
```

where `regression_object` is the regression model under question. For the ozone data, if a regression model is created called `ozone_reg`, then the code would be `plot(ozone_reg)`. This actually creates a series of plots and the "Enter" key must be pressed to cycle through each of them. The very first plot is the residual-vs-predicted plot that is desired (see figure below).

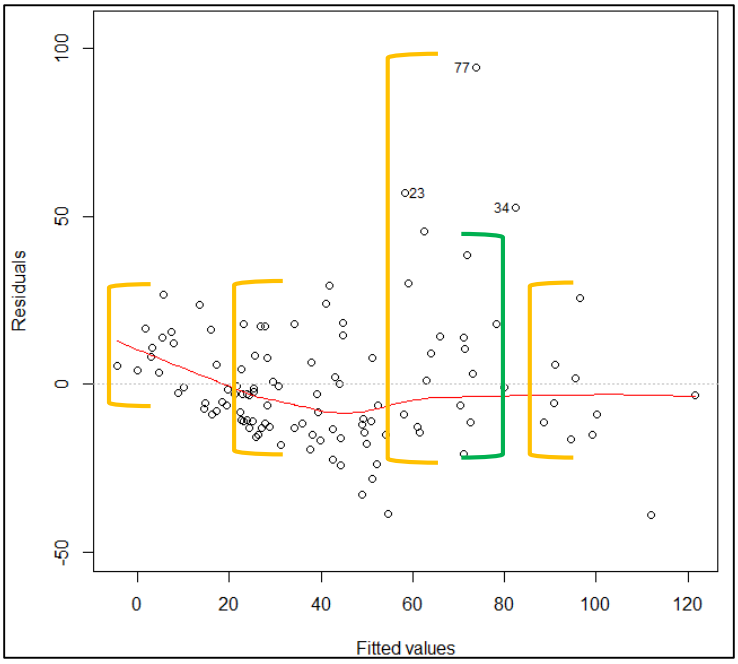

Figure 11 Checking for Homoscedasticity

The data points in Figure 11 are not evenly distributed across the *x*-axis. I have placed four orange brackets at key areas on the plot showing the spread of the data. The tail ends appear to have a fairly even spread, while a section in the middle has a large

spread very different from the rest of the body. Notice that data points 23, 34, and 77 appear to be causing this large spread. The green bracket shows where the spread would be if those three data points were removed. It is possible that they are outliers. Their presence is causing heteroscedasticity, or non-constant variance. With the outliers present, this wouldn't pass the test for this assumption, and regression would be ruled out. It is still too early to decide what possible remedies might be taken.

The fourth assumption to check is that of independence of residuals. Time series data usually violates this assumption and network analysis data as well. To check for this, simply plot the residual values on the *y*-axis and the time-ordered data on the *x*-axis. You can do this using the plot() function in R (see In-Class Exercise 2 for more on using plots). This is a somewhat subjective assessment, however, and usually needs to be checked using a more objective test because of a lack of time-ordered data.

The Durbin-Watson test is part of the library car. Ensure the library is installed prior to running it. This is a straight-forward function; simply type in durbinWatsonTest(reg_model) where reg_model is your regression model. Below is the output of running the test on one of the ozone regression models. The result is not significant; therefore, there is no evidence to suggest this assumption is violated.

```
> durbinWatsonTest(ozone_reg)
 lag Autocorrelation D-W Statistic p-value
   1       0.03173524      1.935028   0.628
 Alternative hypothesis: rho != 0
```

Figure 12 Durbin-Watson Test for Independence

The last assumption is that of normally distributed residuals. A majority of people confuse this with having a normal distribution for each variable. These are two different concepts. In a previous In-Class Exercise, you learned to assess normality using a QQ Plot and the Shapiro-Wilk test. Those assessments were for checking normality of variables, not an entire model. While it is good to have normally distributed variables, it is not a required condition to conduct regression. The residuals, on the other hand, must be normally distributed. This can be assessed by creating a QQ Plot. Luckily, R's regression object, when plotted, provides such a plot.

Using the same steps to check for homoscedasticity, simply type in plot(regression_object) and cycle to the second plot. To the right is the QQ Plot for the ozone dataset. This plot is read just like you were checking the normality for a variable. Notice the right-tail of the data extends upward, away from the normal line (i.e. the dashed line). Also notice that the data points 23, 34, and 77 are at the tail end. Again, it is quite possible that these data points are outliers. With the data points in the model, the assumption of normally distributed residuals fails for the ozone data. One solution is to remove these three data points; another solution is to transform the data, forcing it to be more normal.

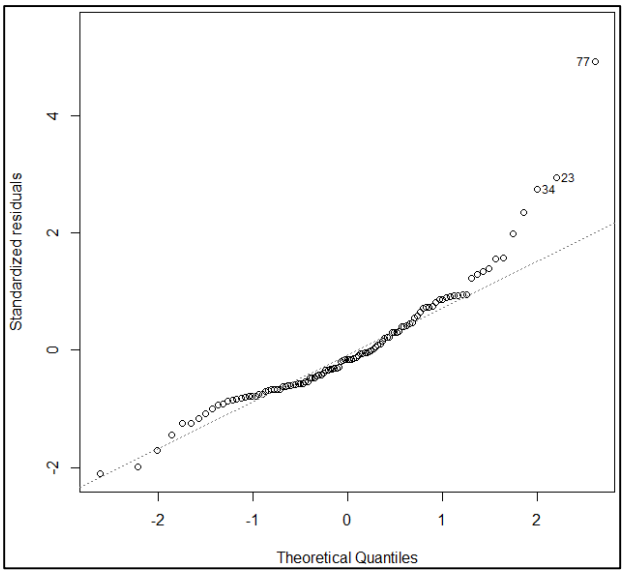

Figure 13 QQ Plot for Ozone Data

2. Linear & Multiple Regression

The two basic regression models are simple linear regression and multiple regression. These form the basis for understanding other types of regression such as polynomial regression and logistic regression. Recall that the equation for a line is

$$y = mx + b$$

where *y* is the dependent variable, *x* is the independent variable, and *m* and *b* are parameters, or constants. The parameter *b* is the *y*-intercept, or the value of *y* when *x* equals 0. The parameter *m* is the slope of the line and indicates to extent to which *y* changes with *x*.

Simple linear regression is merely the equation for a line. It contains only a single variable. The notation for the line equation changes when talking about regression. The equation is given as follows:

$$y = \beta_0 + \beta_1 x_1$$

Multiple regression takes this equation and extends it by introducing multiple slopes with multiple variables.

The equation is given as

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \cdots + \beta_n x_n$$

where *n* is the number of variables if the model has more than two. The temptation exists to throw everything into a regression equation and see which variables work with the dependent variable. This is known as the kitchen sink model. Like with any model, the approach here should be about parsimony. That is, make it simple. The more complex the model becomes (i.e. the more variables in the model) the more difficult it becomes to implement it in the real world. Too many moving parts!

Return to the example of the ozone data. This dataset contains *ozone concentration*, *wind speed*, *air temperature*, and *intensity of solar radiation*. Using these variables, the regression equation would look like the following:

$$ozone = \beta_0 + \beta_1 rad + \beta_1 wind + \beta_2 temp$$

In R, the function used to create a regression model is lm(), which is short for linear model. You must specify the regression model using notation similar to that of the regression equation. See the screenshot below for the regression model and its output.

```
> ozone_reg = lm(ozone_data$ozone~ozone_data$rad+ozone_data$wind+ozone_data$temp)
> summary(ozone_reg)

Call:
lm(formula = ozone_data$ozone ~ ozone_data$rad + ozone_data$wind +
    ozone_data$temp)

Residuals:
    Min      1Q  Median      3Q     Max
-40.485 -14.210  -3.556  10.124  95.600

Coefficients:
                 Estimate Std. Error t value Pr(>|t|)
(Intercept)     -64.23208   23.04204  -2.788  0.00628 **
ozone_data$rad    0.05980    0.02318   2.580  0.01124 *
ozone_data$wind  -3.33760    0.65384  -5.105 1.45e-06 ***
ozone_data$temp   1.65121    0.25341   6.516 2.43e-09 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 21.17 on 107 degrees of freedom
Multiple R-squared:  0.6062,    Adjusted R-squared:  0.5952
F-statistic: 54.91 on 3 and 107 DF,  p-value: < 2.2e-16
```

Figure 14 Regression Model for Ozone Data

The output provides a lot of information about the regression model and can be overwhelming. The first place to start is at the bottom of the output. The first thing to check is the resulting F-statistic and p-value for the F-test. This is an ANOVA. Every p-value is associated with a hypothesis test. Each hypothesis test has a null hypothesis,

designated as H_0, and at least one alternative hypothesis, designated as H_A. The F-test for a regression has the following hypotheses:

H_0: The null hypothesis states that all slopes, or βs, are equal to 0; that is, they have no effect on the dependent variable.

H_A: The alternative hypothesis states that at least one or more slopes is not equal to 0; that is, at least one slope has an effect on the dependent variable.

The null hypothesis is stating that all of the slopes of the line have no effect because 0 times any number is 0. Mathematically, it is expressed in the following way:

$$H_0: \beta_1 = \beta_2 = \cdots \beta_n = 0$$

If the p-value for this test is significant, then the null is rejected; at least one of the slopes influences the target variable. Note, however, that the F-test is unable to specify which variable is significant. It can only tell you that at least one of them is significant. *This is not a measure of how good the model is!* The coefficient of determination, or R-Squared, provides a measure of how good a regression model is. More on that later.

In the output above, the p-value for the F-statistic is less than $2.2e^{-16}$. Highly significant. If it had not been significant, then you would stop right away and not continue with the analysis. This means that either *radiation, wind,* or *temperature* is influential; or, possibly all three are influential.

With a significant F, how would someone know if a specific variable has an effect on the target variable? This is done by assessing the t-statistic associated with each independent variable. The hypotheses for the t-test in a regression are as follows:

H_0: The null hypothesis states that the slope, or β, is equal to 0; that is, this independent variable has no effect on the dependent variable.

H_A: The alternative hypothesis states that this slope is not equal to 0; that is, this independent variable has an effect on the dependent variable.

Mathematically, it is expressed as follows:

$$H_0: \beta_i = 0$$

Look at the p-value for each for each of the variables (the column on the far right of the middle highlighted area). *Radiation* has a p-value of 0.011, which is significant at the 0.05 level. This means it has a strong effect on *ozone*. The other two variables are

even more significant, indicating they have a stronger effect on *ozone*. How strong of an effect do they have?

This information is easy to find. Looking at the regression output, the first column provides the coefficients, or estimates, of each variable. *Radiation* has a slope of 0.059, *wind* has a slope of -3.338, and *temperature* has a value of 1.651. Using this information, the regression equation becomes

$$ozone = -64.232 + 0.060rad - 3.338wind + 1.651temp$$

At this point in building the model, you can now check for homoscedasticity, variance inflation factor, and normally distributed residuals. As stated earlier in this exercise, this can only be done after the regression model is created. To check for homoscedasticity and normally distributed residuals, simply look at a plot of the regression model. To check for the variance inflation factors, use the function provided by the library car.

Using the ozone data as an example, the code for performing these assumption checks would look like the following:

```
> ozone_reg2 = lm(log(ozone)~rad+wind+temp,data=ozone_data)
> vif(ozone_reg2)
     rad     wind     temp
1.095241 1.328979 1.431201
> durbinWatsonTest(ozone_reg2)
 lag Autocorrelation D-W Statistic p-value
   1      0.08887441      1.806428   0.292
 Alternative hypothesis: rho != 0
> plot(ozone_reg2)
```

Figure 15 Assessing Assumptions in Regression

Assume that no problems exist with this data. Two more steps are required for building the regression model. The model as a whole needs to be checked for how good it represents the data. Once done, the most important part of building a regression model is interpreting the results.

Looking at Figure 14, the R-Squared and the Adjusted R-Squared are given at the bottom of the output. A value of 60% is provided. This is a very good number for this initial model knowing the problems it has with two of the assumptions. Remember, use the Adjusted R-Squared value, not the R-Squared.

To interpret the model, simply use the coefficients, or slopes, to provide the change in the dependent variable wrought by the independent variable. For example, *wind* has a slope of -3.338. This means that for every unit increase in *wind*, *ozone* will

decrease by -3.338. This indicates that in areas with higher wind speeds the level of ozone is depleted more. For *temperature*, every increase in 1 unit, *ozone* will increase by 1.651.

3. Polynomial Regression

Unless you have a lot of experience with regression, you may not understand how often data will not conform to a linear relationship. Data often follow different curvilinear paths. One such method that uses the principals of regression is known as polynomial regression. Below, in the figure, are some examples of four different polynomial regression equations.

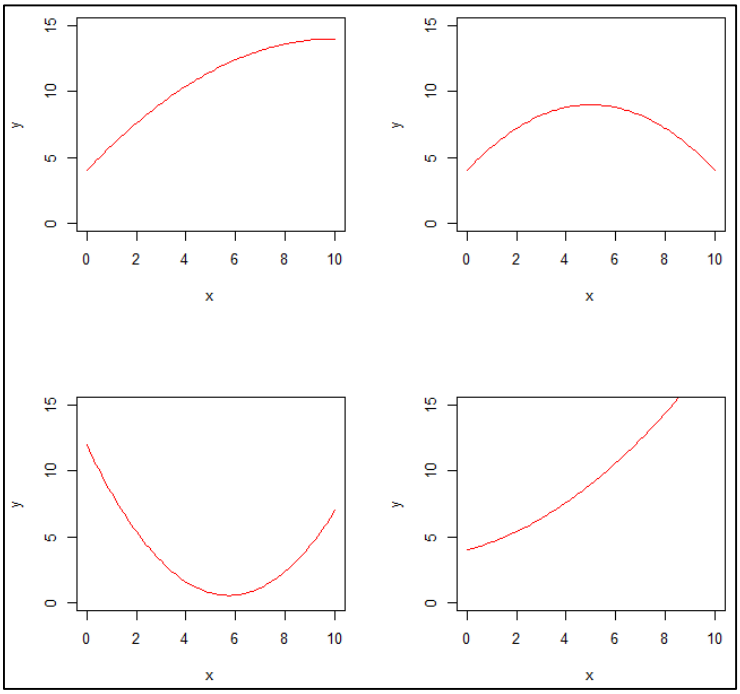

Figure 16 Examples of Polynomial Regression Equations

The following are the equations represented:

- Top-Left: $y = 4 + 2x - 0.1x^2$
- Top-Right: $y = 4 + 2x - 0.2x^2$
- Bottom-Left: $y = 12 - 4x + 0.35x^2$
- Bottom-Right: $y = 4 + 0.5x + 0.1x^2$

The quadratic examples above merely illustrate the possibilities that exist for polynomial regression. A good way to illustrate the power and potential a polynomial

regression model has over a linear regression model is to compare the fit between the two models. Figure 17 presents two regression functions for the same set of data. The one on the left-side is a linear model while the one on the right-side is a polynomial model with a quadratic element.

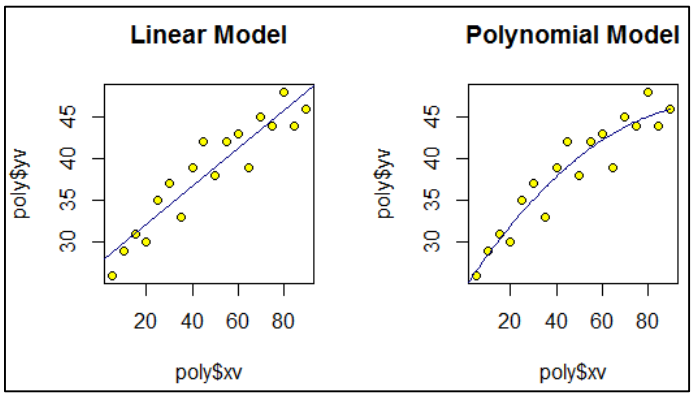

Figure 17 Linear Regression vs Polynomial Regression

Just by glancing at the two different models, you can see that the polynomial model appears to fit the data better. A linear regression actually yields decent results, with a significant value less than 0.0001 for the slope and an adjusted R-Squared of 86.5%. Checking the assumptions of homoscedasticity and normally distributed residuals does reveal some problems, as shown in the figure below.

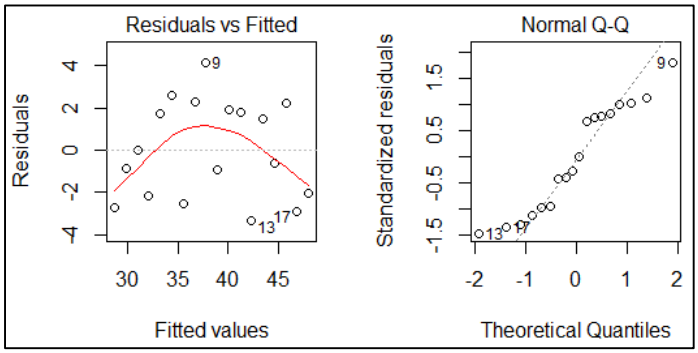

Figure 18 Linear Model: Check for Homoscedasticity and Normality

The results of the polynomial model are also encouraging. The slope has a significant t-statistic as well as a significant quadratic term (see output below). More importantly, the adjusted R-Squared is higher, at 89.2%.

```
Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) 24.6323529  1.6916139  14.561 2.95e-10 ***
xv           0.4057534  0.0819860   4.949 0.000175 ***
I(xv^2)     -0.0018834  0.0008386  -2.246 0.040203 *
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 2.131 on 15 degrees of freedom
Multiple R-squared:  0.9046,    Adjusted R-squared:  0.8919
F-statistic: 71.12 on 2 and 15 DF,  p-value: 2.222e-08
```

Figure 19 Results of Polynomial Regression

Is this change in R-Squared a significant improvement over the other model? To determine that, an ANOVA can be run to compare the linear model against the polynomial model. The results, as shown in Figure 20, indicate the answer to be yes. The result is a significant difference according to the F-test.

```
> anova(model1,model2)
Analysis of Variance Table

Model 1: yv ~ xv
Model 2: yv ~ xv + I(xv^2)
  Res.Df    RSS Df Sum of Sq      F Pr(>F)
1     16 91.057
2     15 68.143  1    22.915 5.0441 0.0402 *
```

Figure 20 ANOVA Comparing Linear Model to Poly. Model

4. Logistic Regression

A logistic regression differs in many ways from linear regression and polynomial regression. First of all, it is part of the generalized linear models, or GLM, as opposed to linear models (LM) like linear regression and polynomial regression. This means that the assumptions of regression do not apply to logistic regression. Secondly, the target variable is coded with a categorical value. If the categorical value is a binary value—typically 0/1, true/false, win/lose, male/female, -1/1—then it is binomial logistic regression. If the target variable has three or more categories, then it is referred to as multinomial logistic regression. Because binomial logistic regression is simpler to understand and more common, it is typically referred to as logistic regression. This tutorial will focus on binomial logistic regression.

The example data for this part of the tutorial is grading from a previous semester of a course I taught. The data includes only A and B grades. The dataset is class_performance.txt. The dataset contains seven columns of data. The dependent variable is *Grade*, which is a factor (i.e. categorical) containing A or B. In the logit

model, the predictor variables will be *Project* and *Exam*, as well as an interaction term including the two predictors.

```
> stdt_reg1 = glm(Grade~Project*Exam, binomial, data=stdt_data)
> summary(stdt_reg1)

Call:
glm(formula = Grade ~ Project * Exam, family = binomial, data = stdt_data)

Deviance Residuals:
     Min        1Q    Median        3Q       Max
-1.45985  -0.45338  -0.08094   0.07299   2.97246

Coefficients:
             Estimate Std. Error z value Pr(>|z|)
(Intercept)    -389.5      229.2  -1.700   0.0892 .
Project         421.3      242.7   1.736   0.0826 .
Exam            794.9      415.9   1.911   0.0560 .
Project:Exam   -856.4      440.3  -1.945   0.0517 .
```

Figure 21 Logit Model of Grade Data

At first glance it would appear that the model is not very good. None of the predictors are very significant. The first inclination is to toss out the entire model and start from scratch. The other consideration may be to remove the variable *Project* because it has the worst p-value. Before jumping to such a hasty conclusion, it may be beneficial to look at the deviance scores attributed to each of the predictors, including the interaction.

```
> anova(stdt_reg1, test="Chisq")
Analysis of Deviance Table

Model: binomial, link: logit

Response: Grade

Terms added sequentially (first to last)


             Df Deviance Resid. Df Resid. Dev  Pr(>Chi)
NULL                         68     92.367
Project       1  29.4787        67     62.889 5.653e-08 ***
Exam          1  19.0153        66     43.873 1.297e-05 ***
Project:Exam  1   4.8361        65     39.037   0.02787 *
```

Figure 22 Deviance Scores from Logit

Figure 22 reveals the deviance scores and the associated p-value for each variable. Notice that the two main effects are highly significant; the interaction term appears to contribute the least deviance. This would indicate the interaction term may be pulling away needed variance from the main effects, and should be removed from the model.

```
> stdt_reg2 = glm(Grade~Project+Exam, binomial, data=stdt_data)
> summary(stdt_reg2)

Call:
glm(formula = Grade ~ Project + Exam, family = binomial, data = stdt_data)

Deviance Residuals:
    Min       1Q   Median       3Q      Max
-1.5173  -0.5982  -0.2045   0.2731   2.4784

Coefficients:
            Estimate Std. Error z value Pr(>|z|)
(Intercept)   89.334     23.778   3.757 0.000172 ***
Project      -83.323     23.200  -3.592 0.000329 ***
Exam         -17.421      5.683  -3.066 0.002172 **
```

Figure 23 Revised Logit Function

The results from the newly revised logistic function in Figure 23 agree with the previous conclusion. The interaction term was degrading the model and its removal results in better performance. The results indicate that both *Project* and *Exam* are influential in students obtaining a grade within the classroom.