MSIS 5223: Tutorial 12 – Regression with Categorical Data in Python

1. Binning a Variable

The example presented here will use the data file reduction_data_new.txt. Only a subset of this data will be used including the following columns: *peruse01*, *intent01*, *gender*, *educ_level*, and *age*. The columns are renamed to improve readability. Additionally, missing values are removed from the dataframe.

```
In [128]: red_data = reduction_data[['peruse01', 'intent01', 'gender', 'educ_level', 'age']]

In [129]: red_data.columns = ['usefulness', 'intent', 'gender', 'education', 'age']

In [130]: red_data = red_data.dropna()

In [131]: red_data.reset_index(inplace=True)
```

Figure 1 Preparing the Data for Analysis

The basic structure of the dataframe reveals three numeric columns (*usefulness*, *intent*, and *age*) and two categorical columns (*gender* and *education*) (see Figure 2 below). While the temptation to use *age* as a continuous variable is strong, it is not usable in its current format. Look at Figure 2 below. The two lines of code at the bottom reveal the ranges for *intent* and *age*. *Intent* has a range of 1 to 7 while *age* spans 18 to 48. As these are not on the same scale, they are not directly comparable. One solution would be to standardize both variables; the other is to convert *age* into a categorical variable.

The purpose of binning is to impose “levels” or “categories” onto a numerical variable. Think back to the example of income. It is possible to have discrete values of income, but often it is easier to work with income brackets. The same principal applies to age.

```
In [132]: red_data.dtypes
Out[132]:
index            int64
usefulness       int64
intent           int64
gender        category
education     category
age            float64
dtype: object

In [133]: red_data.head()
Out[133]:
   index  usefulness  intent gender education   age
0      0           7       7    1.0       4.0  22.0
1      1           6       7    2.0       4.0  32.0
2      2           5       5    1.0       2.0  19.0
3      3           5       5    1.0       3.0  21.0
4      4           7       6    2.0       3.0  21.0

In [134]: red_data['age'].max()
Out[134]: 48.0

In [135]: red_data['age'].min()
Out[135]: 18.0

In [136]: red_data['intent'].max()
Out[136]: 7

In [137]: red_data['intent'].min()
Out[137]: 1
```

Figure 2 Basic Structure of the Dataframe

Binning is quite simple using the module scipy.stats.binned_statistic. The process of binning requires you to select a size for each bin. That is, how many units will each bin hold? In this case, one unit of age is one year. This data contains a range of 31 years because the maximum value is 48 and the minimum is 18, inclusive. It is always good to start off with a conservative number of bins; anywhere from 5 to 8. Let's start off with 6 bins. Divide 31 by 6 and the result is approximately 5; that is, each bin will contain 5 years.

```
In [163]: bin_counts,bin_edges,binnum = binned_statistic(red_data['age'],
     .....:                                              red_data['age'],
     .....:                                              statistic='count',
     .....:                                              bins=6)
     .....:

In [164]: bin_counts
Out[164]: array([ 116.,   32.,   10.,    2.,    3.,    2.])

In [165]: bin_edges
Out[165]: array([ 18.,  23.,  28.,  33.,  38.,  43.,  48.])
```

Figure 3 Binning by 5-Year Increments

Figure 3 presents the result of creating 5-year incremental bins. The first line is the function used to process the bins. It returns three objects: The counts within each bin, the number that represents the start of each bin, and actual data values contained within each bin. For example, the first bin contains 116 records with18 as the start of the bin; 23 is the start of the next bin, and it contains 32 records.

Notice the three bins on the left contain the most data: 158 records out of 165. The figure below presents a histogram of *age*. Not surprisingly, the data is skewed to the right with the majority on the left side. Perhaps the bin sizes are too great. Let's use a bin size of 2 instead of 5.

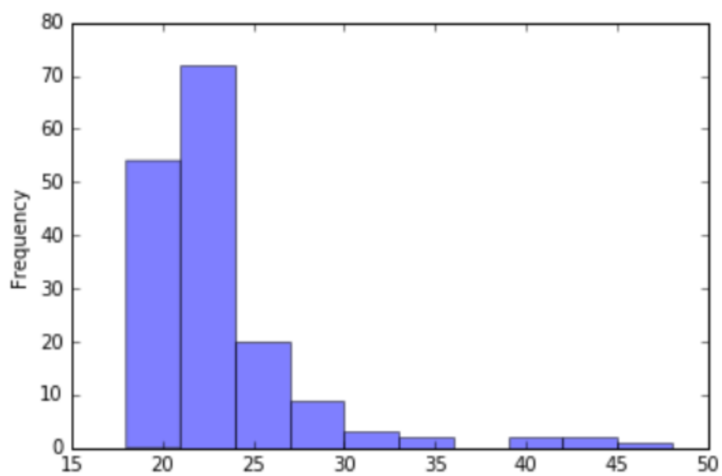

Figure 4 Histogram of Age

```
In [166]: bin_counts,bin_edges,binnum = binned_statistic(red_data['age'],
     .....:                                               red_data['age'],
     .....:                                               statistic='count',
     .....:                                               bins=15)
     .....:

In [167]: bin_counts
Out[167]:
array([ 21.,  69.,  36.,  15.,   7.,   7.,   2.,   2.,   1.,   0.,   1.,
         1.,   2.,   0.,   1.])

In [168]: bin_edges
Out[168]:
array([ 18.,  20.,  22.,  24.,  26.,  28.,  30.,  32.,  34.,  36.,  38.,
        40.,  42.,  44.,  46.,  48.])
```

Figure 5 Binning with 2-Year Increments

Binning with 2-year increments has created a more even distribution of the sample within each bin, as shown in Figure 5. While the majority of the data is retained in the first few bins, it isn't an overwhelming amount. This is much better, but does not have equally distributed bins. The first four bins still contain the majority of the data. In fact, all the other combined contain 24 data points. Perhaps forcing the other bins into one single bin would make sense in order to keep a fairly consistent distribution.

```
In [169]: bin_interval = [18, 20, 22, 24, 26, 50]

In [170]: bin_counts, bin_edges, binnum = binned_statistic(red_data['age'],
     .....:                                                 red_data['age'],
     .....:                                                 statistic='count',
     .....:                                                 bins=bin_interval)
     .....:

In [171]: bin_counts
Out[171]: array([ 21.,  69.,  36.,  15.,  24.])

In [172]: bin_edges
Out[172]: array([ 18.,  20.,  22.,  24.,  26.,  50.])
```

Figure 6 Consolidating Bins

Figure 6 presents the adjustment to the bins. Notice the last bin contains data with ages between 26 and 50. This has a total of 24 records, which is a good amount. The first bin contains 21, so this is close enough to make the distribution of all these bins fairly normal. Think of this process as similar to applying a logarithmic transformation.

Now that the bin values have been selected, the *age* column must be converted into these bins and added back into the dataframe. Figure 7 below presents the code that follows these steps. The function `cut` comes from the library `Pandas`. Essentially, it finds values within a target—the variable *age* in this case—and replaces them based on criteria provided. For example, if the age is 18 through 19, then the new value is a character "18-19"; if the age is 26 through 48, then the new value is "26-48". Notice the labels for the columns are manually created. The labels can be made automatically, but will only be given incremental, numerical values.

```
In [173]: binlabels = ['age_18_19', 'age_20_21', 'age_22_23', 'age_24_25', 'age_26_48']

In [174]: age_categ = pd.cut(red_data['age'], bin_interval, right=False, retbins=False, labels=binlabels)

In [175]: age_categ.name = 'age_categ'

In [176]: red_data = red_data.join(pd.DataFrame(age_categ))
```

Figure 7 Recoding Age Into New Values

The third line of code renames the newly created data called *age_categ*. The fourth line of code adds this newly created data back into the dataframe as a new column.

At this point, no indicator variables have been created. This process has only led to the creation of a new categorical variable based on *age*. To create a dummy variable, use the library pandas. The process is straight forward because the library does all the heavy lifting. The figure below presents the results of using the function get_dummies(). Each new column contains the name of the original column. As Python creates each new column, the category level is used to denote the name of each new dummy variable.

```
In [177]: red_dummy1 = pd.get_dummies(red_data['age_categ'])

In [178]: red_dummy1.head()
Out[178]:
   age_18_19  age_20_21  age_22_23  age_24_25  age_26_48
0        0.0        0.0        1.0        0.0        0.0
1        0.0        0.0        0.0        0.0        1.0
2        1.0        0.0        0.0        0.0        0.0
3        0.0        1.0        0.0        0.0        0.0
4        0.0        1.0        0.0        0.0        0.0

In [179]: red_data = red_data.join(red_dummy1)
```

Figure 8 Creating Indicator Variables from Age_Categ

The newly created dummies are converted into a dataframe object and then placed back into the dataframe. Within the tutorial script the variable *education* is also converted into indicator variables; however, because it is repetitive, the process will not be covered in this tutorial document.

One final note should be mentioned here. Recall, when creating indicator variables, the process is to create *n*-1 new columns. The function dummy() within Python does not provide such a fine distinction. It creates *n* new columns; so, one more column than needed is created. When creating a regression model, merely leave off one of the dummy variables from the regression equation. Personally, I just leave off the last of the columns. Unlike R, Python will not indicate an error by placing the value "N/A" in the regression output. This requires more awareness on the part of the data miner.

2. Regression with Categorical Data

Now that the conversion process is complete, a regression model may be created. This process is very similar to that of multiple regression, except the assumptions of linear regression are not assessed. Since these columns are binary, a linear relationship is not possible. Keep in mind when using binary data, the result is just like ANOVA; therefore, the assumptions of linear regression are not required.

The first regression equation will use gender and the age dummy variables. When the indicator variables for age were created, five new columns were added to the dataframe. Recall, the function `get_dummies()` creates one too many indicator variables. Leave the last indicator variable out of the equation. Figure 9 provides the results of this assessment.

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 intent   R-squared:                       0.069
Model:                            OLS   Adj. R-squared:                  0.040
Method:                 Least Squares   F-statistic:                     2.362
Date:                Fri, 14 Oct 2016   Prob (F-statistic):             0.0424
Time:                        11:19:57   Log-Likelihood:                -315.24
No. Observations:                 165   AIC:                             642.5
Df Residuals:                     159   BIC:                             661.1
Df Model:                           5
Covariance Type:            nonrobust
=================================================================================
                    coef    std err          t      P>|t|      [95.0% Conf. Int.]
---------------------------------------------------------------------------------
Intercept         6.4296      0.354     18.156      0.000         5.730     7.129
gender[T.2.0]    -0.1455      0.264     -0.550      0.583        -0.668     0.377
age_18_19        -0.7482      0.498     -1.503      0.135        -1.731     0.235
age_20_21        -1.2108      0.396     -3.056      0.003        -1.993    -0.428
age_22_23        -0.6270      0.439     -1.429      0.155        -1.494     0.240
age_24_25        -0.3811      0.548     -0.695      0.488        -1.464     0.702
==============================================================================
Omnibus:                       44.764   Durbin-Watson:                   1.671
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               71.314
Skew:                          -1.466   Prob(JB):                     3.27e-16
Kurtosis:                       4.332   Cond. No.                         7.74
==============================================================================
```

Figure 9 Gender and Age on Intent

The age range of 20 to 21 is found to be significant when predicting the usage of mobile technology. The interpretation of this is no different from multiple regression. Using the significant variables, create a regression equation using the coefficients and the intercept. Insert the actual values from the dataset and find the value for the dependent variable.

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 intent   R-squared:                       0.104
Model:                            OLS   Adj. R-squared:                  0.052
Method:                 Least Squares   F-statistic:                     2.000
Date:                Fri, 14 Oct 2016   Prob (F-statistic):             0.0427
Time:                        11:22:57   Log-Likelihood:                -312.08
No. Observations:                 165   AIC:                             644.2
Df Residuals:                     155   BIC:                             675.2
Df Model:                           9
Covariance Type:            nonrobust
=================================================================================
                 coef    std err          t      P>|t|      [95.0% Conf. Int.]
---------------------------------------------------------------------------------
Intercept      6.4003      0.617     10.367      0.000         5.181     7.620
gender[T.2.0] -0.0983      0.266     -0.370      0.712        -0.623     0.426
age_18_19     -0.9993      0.546     -1.829      0.069        -2.079     0.080
age_20_21     -1.5252      0.441     -3.460      0.001        -2.396    -0.654
age_22_23     -0.8254      0.460     -1.794      0.075        -1.734     0.084
age_24_25     -0.4587      0.553     -0.829      0.408        -1.552     0.634
educ_2        -0.2367      0.787     -0.301      0.764        -1.791     1.318
educ_3         0.4664      0.655      0.712      0.478        -0.828     1.761
educ_4        -0.0786      0.755     -0.104      0.917        -1.569     1.412
educ_5        -0.3481      0.689     -0.505      0.614        -1.710     1.014
==============================================================================
Omnibus:                       42.011   Durbin-Watson:                   1.655
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               64.732
Skew:                          -1.401   Prob(JB):                     8.78e-15
Kurtosis:                       4.252   Cond. No.                         15.3
==============================================================================
```

Figure 10 Gender, Age, and Education

Figure 10 presents another regression equation using the education dummy variables. The addition of these variables does not improve the regression, resulting in no significant relationships.

```
                            OLS Regression Results
==============================================================================
Dep. Variable:                 intent   R-squared:                       0.144
Model:                            OLS   Adj. R-squared:                  0.112
Method:                 Least Squares   F-statistic:                     4.435
Date:                Fri, 14 Oct 2016   Prob (F-statistic):           0.000357
Time:                        11:23:45   Log-Likelihood:                -308.31
No. Observations:                 165   AIC:                             630.6
Df Residuals:                     158   BIC:                             652.4
Df Model:                           6
Covariance Type:            nonrobust
=================================================================================
                 coef    std err          t      P>|t|      [95.0% Conf. Int.]
---------------------------------------------------------------------------------
Intercept      3.9348      0.752      5.232      0.000         2.449     5.420
gender[T.2.0] -0.1787      0.255     -0.702      0.484        -0.681     0.324
age_18_19     -0.7765      0.479     -1.621      0.107        -1.722     0.170
age_20_21     -1.3687      0.383     -3.569      0.000        -2.126    -0.611
age_22_23     -0.8253      0.426     -1.939      0.054        -1.666     0.015
age_24_25     -0.4781      0.528     -0.906      0.367        -1.521     0.565
usefulness     0.4593      0.123      3.721      0.000         0.216     0.703
==============================================================================
Omnibus:                       37.690   Durbin-Watson:                   1.764
Prob(Omnibus):                  0.000   Jarque-Bera (JB):               55.231
Skew:                          -1.288   Prob(JB):                     1.02e-12
Kurtosis:                       4.184   Cond. No.                         40.6
==============================================================================
```

Figure 11 Gender, Age, and Usefulness

The last regression model represents an ANCOVA model. Education has been removed from the model and replaced with *usefulness*. Recall, this variable is continuous and not a categorical variable. Not surprisingly, this is a significant variable in the model.