NYC Subway Ridership Analysis

May 3, 2015

1 Do more people ride with the NYC Subway if it is raining?

Author: Michael Lichtsinn
Date: 03.05.2015, Darmstadt, Germany

1.1 About this analysis

This is a data analysis performed during the Data Analyst Nanodegree by Udacity.com. The questions to be answered in this project is wheater or not more people in New York use the Subway when it is raining in New York.

1.2 Data

This analysis uses NYC Subway ridership data and the weather data for New York. The ridership data and the weather data were previously merged by Udacity. They also provided an improved data set with extra data points and variables. The latter is the data set that was used to perform the following data analysis. The dataset can be found here: https://www.dropbox.com/s/1lpoeh2w6px4diu/improved-dataset.zip?dl=0

A description of the variables of the data set can be found here: https://s3.amazonaws.com/uploads.hipchat.com/23756/665149/05bgLZqSsMycnkg/turnstile-weather-variables.pdf

In [2]:
```
#load the packages
import pandas
from ggplot import *
import scipy.stats
import statsmodels
import numpy
import datetime

%matplotlib inline

#ignore warnings to make it more pleasant for the reader
import warnings
warnings.filterwarnings('ignore')
```

In [3]:
```
#path to the csv file
path = 'D:/Udacity/P1 - NYC Subway data/improved-dataset/turnstile_weather_v2.csv'

#read csv and save as a data frame called "data"
data = pandas.io.parsers.read_csv(path, index_col=False)

#split into two data frames, one with rainy days and the other without
rain = data['ENTRIESn_hourly'][data['rain']==1]
norain = data['ENTRIESn_hourly'][data['rain']==0]
```

```python
#get a summary of the ENTRIESn_hourly by rain
data[['rain','ENTRIESn_hourly']].groupby('rain').describe()
```

Out[3]:

```
              ENTRIESn_hourly
rain
0     count      33064.000000
      mean        1845.539439
      std         2878.770848
      min            0.000000
      25%          269.000000
      50%          893.000000
      75%         2197.000000
      max        32814.000000
1     count       9585.000000
      mean        2028.196035
      std         3189.433373
      min            0.000000
      25%          295.000000
      50%          939.000000
      75%         2424.000000
      max        32289.000000
```

In [4]:

```python
#get a summary of the whole dataset
data.describe()
```

Out[4]:

```
          ENTRIESn        EXITSn  ENTRIESn_hourly  EXITSn_hourly  \
count  4.264900e+04  4.264900e+04     42649.000000   42649.000000
mean   2.812486e+07  1.986993e+07      1886.589955    1361.487866
std    3.043607e+07  2.028986e+07      2952.385585    2183.845409
min    0.000000e+00  0.000000e+00         0.000000       0.000000
25%    1.039762e+07  7.613712e+06       274.000000     237.000000
50%    1.818389e+07  1.331609e+07       905.000000     664.000000
75%    3.263049e+07  2.393771e+07      2255.000000    1537.000000
max    2.357746e+08  1.493782e+08     32814.000000   34828.000000

               hour      day_week       weekday      latitude     longitude  \
count  42649.000000  42649.000000  42649.000000  42649.000000  42649.000000
mean      10.046754      2.905719      0.714436     40.724647    -73.940364
std        6.938928      2.079231      0.451688      0.071650      0.059713
min        0.000000      0.000000      0.000000     40.576152    -74.073622
25%        4.000000      1.000000      0.000000     40.677107    -73.987342
50%       12.000000      3.000000      1.000000     40.717241    -73.953459
75%       16.000000      5.000000      1.000000     40.759123    -73.907733
max       20.000000      6.000000      1.000000     40.889185    -73.755383

                fog     ...          pressurei          rain         tempi  \
count  42649.000000     ...       42649.000000  42649.000000  42649.000000
mean       0.009824     ...          29.971096      0.224741     63.103780
std        0.098631     ...           0.137942      0.417417      8.455597
min        0.000000     ...          29.550000      0.000000     46.900000
25%        0.000000     ...          29.890000      0.000000     57.000000
50%        0.000000     ...          29.960000      0.000000     61.000000
75%        0.000000     ...          30.060000      0.000000     69.100000
max        1.000000     ...          30.320000      1.000000     86.000000
```

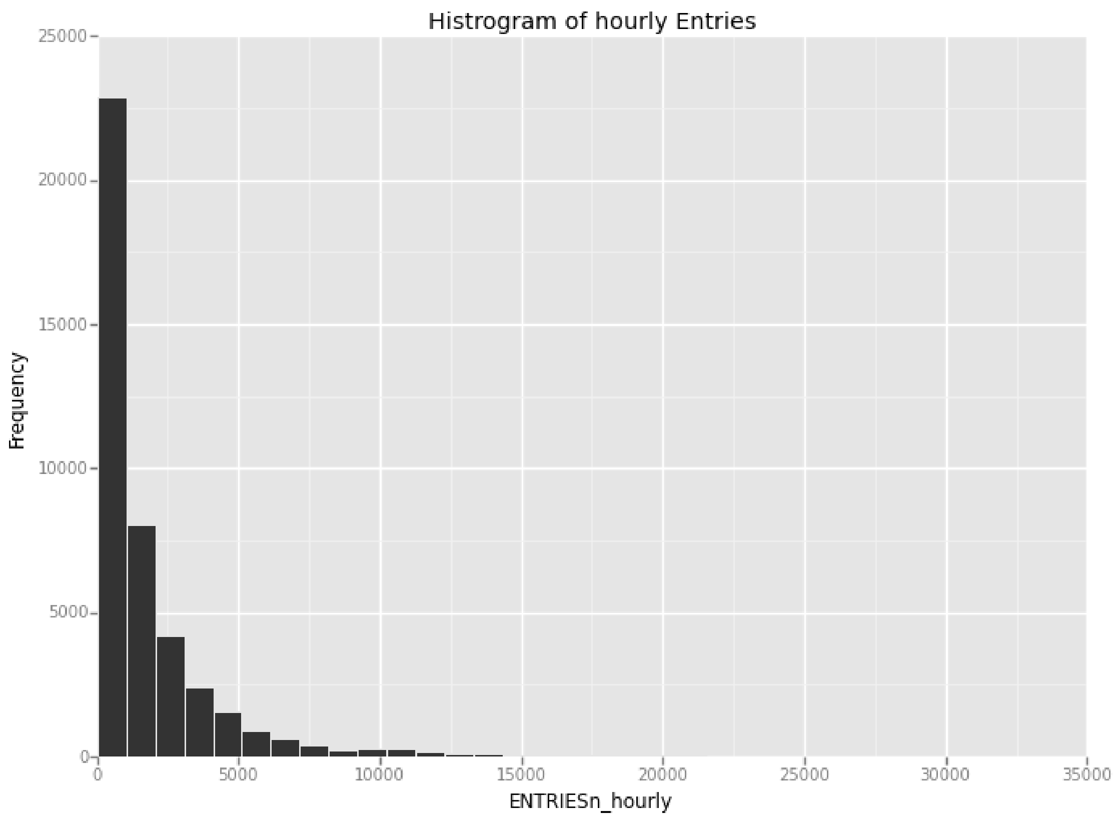
```
               wspdi   meanprecipi  meanpressurei     meantempi     meanwspdi  \
count   42649.000000  42649.000000   42649.000000  42649.000000  42649.000000
mean        6.927872      0.004618      29.971096     63.103780      6.927872
std         4.510178      0.016344       0.131158      6.939011      3.179832
min         0.000000      0.000000      29.590000     49.400000      0.000000
25%         4.600000      0.000000      29.913333     58.283333      4.816667
50%         6.900000      0.000000      29.958000     60.950000      6.166667
75%         9.200000      0.000000      30.060000     67.466667      8.850000
max        23.000000      0.157500      30.293333     79.800000     17.083333

         weather_lat   weather_lon
count   42649.000000  42649.000000
mean       40.728555    -73.938693
std         0.065420      0.059582
min        40.600204    -74.014870
25%        40.688591    -73.985130
50%        40.720570    -73.949150
75%        40.755226    -73.912033
max        40.862064    -73.694176

[8 rows x 21 columns]
```

2 Statistical Test

To determine which statistical test is appropriate for the analysis, we have to have a look at the data first.

The most interesting variable are the hourly entries. A histrogramm will give us a high level look at the distribution of the number of people that are entering the subway.

In [5]:
```
#plot the data to check visually for normal distribution
ggplot(data,aes('ENTRIESn_hourly')) + \
    geom_histogram(binwidth = 1000) + \
    ylab('Frequency') + \
    ggtitle("Histrogram of hourly Entries")
```



Out[5]: <ggplot: (-9223372036842614682)>

This histogram doesn't look like a bell curve. The distribution does not seem normal. To be really sure, we could use the Shapiro-Wilk-Test to test if the sample is drawn from a normal distributed population. But the Shapiro-Wilk-Test may be inaccurate for $N > 5000$ and since we have a $N = 42649$ we do not know if the results are accurate.

Therefore we do not assume anything about the distribution. This also means, we cannot use the Welch-T-Test to test the difference, because this test assumes normal distribution. Since we do not assume anything about the distribution, we have to run a test that does not assume a particular distribution. This will be the Mann-Whitney-U-Test.

3 Mann-Whitney-U-Test

To test if more people ride the subay when it's raining, we will perform a Mann-Whitney-U-Test.

Under the null-hypothesis, the distribution of both groups - rainy and non-rainy- are equal. If we reject the null-hypothesis based on the results, that means that the distribution of both samples are different.

3.1 Interpreting the p-value

How to interpret the p-value of the Mann-Whitney-U-Test? Let's have a look:

> "If the groups are sampled from populations with identical distributions, what is the chance that random sampling would result in the mean ranks being as far apart (or more so) as observed in this experiment?"

> Source:http://graphpad.com/guides/prism/6/statistics/index.htm?how_the_mann-whitney_test_works.htm

If the p-value is small, we can conclude that the populations are distinct because the chance to get a difference in the observed means rank is small and highly unlikely.

In [6]:
```
#perform mann-whitney-u-test on the rainy and non-rainy group
[U, p] = scipy.stats.mannwhitneyu(rain,norain)

print(U,p)
```

```
153635120.5 nan
```

3.2 Possible Bug with scipy, compute the p-value with R

The NaN (Not a Number) in the results seems to be a bug, since this is also reported by other students in the forum. Oliva from Udacity postet an answer:

> Anyways, it seems from that post that the problem is likely with your OS or architecture. Are you using Windows, by any chance? It's a known issue that some students are receiving different values using scipy's Mann-Whitney U test on Windows, whether 32 bit or 64 bit. It's possible that receiving NaN is due to similar reasons.
>
> Olivia on the udacity forums
>
> http://discussions.udacity.com/t/mann-whitney-test-what-does-nan-mean/16373/4

Therefore I computed the p-value for the Mann-Whitney-U test using R with the following results:

> Wilcoxon rank sum test with continuity correction
>
> data: ENTRIESn_hourly by rain W = 153635121, p-value = 5.482e-06 alternative hypothesis: true location shift is not equal to 0

The means for rainy and non-rainy days are the following:
mean_norain = 1845.539439
mean_rain = 2028.196035
On average there are 183 more people riding the subway on rainy days in those samples.

3.3 Rejecting the null-hypothesis: distributions are statistically different

We have a significance level of alpha = 0.05 and a p-value of 0.000005482. Because we do a two-sided test we have to split the alpha in half (one for each side) to get our p-critical value. If the p-value is smaller than the p-critical-value, this means it is highly unlikely that the two samples are drawn from the same population, thus they differ in their distribution.

Since 0.000005482 is smaller than 0.025 we reject the null-hypothesis and can conclude that the samples from rainy days differ in their distribution compared to the sample from non-rainy days.

4 Linear Regression

We want to predict the number of hourly entries and use the a linear regression to achieve this. Linear regression means that there we assume a linear relationship between the independent variable of hourly entries and the dependent variables. For example, if rain has a positive linear relationship with the ridership as hinted in the statistical test above, then more rainy days will result in more people taking the subway. Based on the models learnt so far in the Intro to Data Science course I have choosen to implement Ordinary-Least-Squars (OLS) models using statsmodels. This model is easy to interpret though other models may explain more of the variance. For the learning purpose of this project, it will be sufficient.

4.1 Variables to chose

Rather than dumping all available variables into the model, I will choose those who have the highest probability of a linear relationship based on logical conclusions. To get an idea which variables are there to choose for our regression, let's think about the data first.

This is data about the New York City Subway Ridership. New York City is a city in the United States of America with 8.406 million people. The subway is a public transportation which is for use by the general public. Other forms of transportation include trams, taxis, carpooling, buses for the general public plus walking, cycling or driving with your own car as a private form of transportation.

4.2 Reasons to take the subway

Reasons to take the subway are basically the same as using any form of public transportation:

Need for transportation * e.g. commuting to work

Convenience * Faster * Cheaper * Weatherconditions

4.3 Chosen variables

The variables I chose to include are those who will represent convienience and need for transportation. For example, if it is raining, it is more convienent to take the subway then to walk or go by bike.

Need for transportation * weekday - on weekdays most people need to go to work, thus have a need for transportation * hour - normally people commute to work in the morning and back in the evening

Weatherconditions * fog - indicates if it was foggy that day * rain - indictaes if it was raining that day * tempi - temperature at the time and location * conds - weather condition for time and location

I have also included UNIT, the turnstile units installed on subway station that collected the information, as it greatly increaed the R^2 of my model.

'UNIT', 'weekday' and 'conds' as modeled as dummy variables, because they are categorical variables and represent qualitative data, not quantitative data. Once modeled as dummy variable we can use them in a quantitative model.

In [7]:
```
#Turn UNIT into a dummy variable
dummy_weekday = pandas.get_dummies(data['weekday'],prefix='weekday')
dummy_conds = pandas.get_dummies(data['conds'],prefix='conds')
dummy_unit = pandas.get_dummies(data['UNIT'],prefix='UNIT')

#defining the dependent and independent variables
y = data['ENTRIESn_hourly']
x = data[['hour']]
x = x.join(dummy_weekday)
x = x.join(dummy_conds)
x = x.join(dummy_unit)

#tempi, fog, precipi didn't improve the R^2 at all

#fitting the model
model = statsmodels.regression.linear_model.OLS(y,x)
results = model.fit()
parameter = results.params
results.summary()
```

Out[7]:
```
<class 'statsmodels.iolib.summary.Summary'>
"""
                            OLS Regression Results
==============================================================================
Dep. Variable:        ENTRIESn_hourly   R-squared:                       0.484
```

```
Model:                            OLS   Adj. R-squared:                  0.481
Method:                 Least Squares   F-statistic:                     158.1
Date:                Sun, 03 May 2015   Prob (F-statistic):               0.00
Time:                        21:30:25   Log-Likelihood:            -3.8717e+05
No. Observations:               42649   AIC:                         7.748e+05
Df Residuals:                   42396   BIC:                         7.770e+05
Df Model:                         252
Covariance Type:            nonrobust
==========================================================================================
                             coef    std err          t      P>|t|      [95.0% Conf. Int.]
------------------------------------------------------------------------------------------
hour                     123.6972      1.501     82.424      0.000       120.756   126.639
weekday_0               -162.8296     45.353     -3.590      0.000      -251.722   -73.938
weekday_1                785.2381     43.575     18.020      0.000       699.831   870.646
conds_Clear              137.2742     46.110      2.977      0.003        46.898   227.650
conds_Fog                227.2959    291.423      0.780      0.435      -343.899   798.491
conds_Haze                45.1580     68.465      0.660      0.510       -89.034   179.350
conds_Heavy Rain        -514.1154    123.715     -4.156      0.000      -756.599  -271.632
conds_Light Drizzle     -346.7095    116.200     -2.984      0.003      -574.465  -118.954
conds_Light Rain         510.8856     61.765      8.271      0.000       389.825   631.946
conds_Mist               678.0639    405.120      1.674      0.094      -115.979  1472.107
conds_Mostly Cloudy      -70.3318     48.133     -1.461      0.144      -164.674    24.011
conds_Overcast            45.6196     45.398      1.005      0.315       -43.362   134.601
conds_Partly Cloudy       82.8562     58.870      1.407      0.159       -32.530   198.242
conds_Rain              -455.2409     76.134     -5.980      0.000      -604.464  -306.018
conds_Scattered Clouds   281.6526     53.620      5.253      0.000       176.556   386.749
UNIT_R003              -1637.9655    163.884     -9.995      0.000     -1959.182 -1316.749
UNIT_R004              -1286.8453    160.704     -8.008      0.000     -1601.829  -971.862
UNIT_R005              -1278.2940    162.095     -7.886      0.000     -1596.004  -960.584
UNIT_R006              -1162.4226    158.482     -7.335      0.000     -1473.051  -851.794
UNIT_R007              -1452.6219    163.028     -8.910      0.000     -1772.160 -1133.084
UNIT_R008              -1446.5845    163.513     -8.847      0.000     -1767.073 -1126.096
UNIT_R009              -1492.7518    160.684     -9.290      0.000     -1807.697 -1177.807
UNIT_R011               5528.5381    156.508     35.324      0.000      5221.778  5835.298
UNIT_R012               6871.9603    155.670     44.144      0.000      6566.844  7177.077
UNIT_R013                770.4012    155.670      4.949      0.000       465.285  1075.518
UNIT_R016              -1040.6920    156.506     -6.650      0.000     -1347.448  -733.936
UNIT_R017               2385.3958    155.670     15.323      0.000      2080.279  2690.512
UNIT_R018               5973.8049    156.245     38.234      0.000      5667.561  6280.049
UNIT_R019               1459.0273    155.915      9.358      0.000      1153.431  1764.624
UNIT_R020               4561.4926    155.670     29.302      0.000      4256.376  4866.609
UNIT_R021               2882.1545    156.511     18.415      0.000      2575.390  3188.919
UNIT_R022               7705.8851    155.670     49.501      0.000      7400.768  8011.002
UNIT_R023               4341.0625    155.670     27.886      0.000      4035.946  4646.179
UNIT_R024               1424.7166    156.330      9.114      0.000      1118.307  1731.126
UNIT_R025               3555.9198    155.915     22.807      0.000      3250.323  3861.516
UNIT_R027               1155.4980    155.670      7.423      0.000       850.381  1460.615
UNIT_R029               5417.4711    155.670     34.801      0.000      5112.354  5722.588
UNIT_R030               1287.6646    155.670      8.272      0.000       982.548  1592.781
UNIT_R031               2539.4334    155.670     16.313      0.000      2234.317  2844.550
UNIT_R032               2638.7228    156.087     16.905      0.000      2332.790  2944.656
UNIT_R033               6422.4442    155.670     41.257      0.000      6117.328  6727.561
UNIT_R034               -636.2188    162.799     -3.908      0.000      -955.308  -317.130
UNIT_R035                994.0471    156.506      6.351      0.000       687.291  1300.803
```

UNIT_R036	-1038.7339	159.706	-6.504	0.000	-1351.761	-725.707
UNIT_R037	-955.5769	157.095	-6.083	0.000	-1263.486	-647.668
UNIT_R038	-1589.5399	160.616	-9.897	0.000	-1904.350	-1274.730
UNIT_R039	-1027.1959	164.870	-6.230	0.000	-1350.344	-704.048
UNIT_R040	-551.1925	156.665	-3.518	0.000	-858.260	-244.125
UNIT_R041	1287.4764	155.670	8.271	0.000	982.360	1592.593
UNIT_R042	-1197.9319	158.225	-7.571	0.000	-1508.056	-887.808
UNIT_R043	1074.6109	155.670	6.903	0.000	769.494	1379.727
UNIT_R044	2866.0195	155.670	18.411	0.000	2560.903	3171.136
UNIT_R046	6532.4711	155.670	41.964	0.000	6227.354	6837.588
UNIT_R049	960.2829	155.670	6.169	0.000	655.166	1265.399
UNIT_R050	2212.0838	156.510	14.134	0.000	1905.320	2518.847
UNIT_R051	3322.0248	155.670	21.340	0.000	3016.908	3627.141
UNIT_R052	-551.9929	162.017	-3.407	0.001	-869.550	-234.436
UNIT_R053	1373.3071	156.666	8.766	0.000	1066.240	1680.375
UNIT_R054	-343.7477	156.511	-2.196	0.028	-650.512	-36.984
UNIT_R055	6528.4006	155.832	41.894	0.000	6222.968	6833.833
UNIT_R056	-358.3967	156.508	-2.290	0.022	-665.156	-51.637
UNIT_R057	3070.4119	155.670	19.724	0.000	2765.295	3375.529
UNIT_R058	-1168.9341	156.088	-7.489	0.000	-1474.871	-862.998
UNIT_R059	-602.6159	159.565	-3.777	0.000	-915.367	-289.865
UNIT_R060	-995.3806	158.228	-6.291	0.000	-1305.511	-685.251
UNIT_R061	-1167.0231	164.243	-7.105	0.000	-1488.943	-845.103
UNIT_R062	924.2721	155.670	5.937	0.000	619.156	1229.389
UNIT_R063	-599.1354	163.757	-3.659	0.000	-920.102	-278.169
UNIT_R064	-948.2477	160.005	-5.926	0.000	-1261.860	-634.635
UNIT_R065	-925.8623	161.846	-5.721	0.000	-1243.083	-608.641
UNIT_R066	-1506.0832	162.787	-9.252	0.000	-1825.149	-1187.018
UNIT_R067	-891.2027	164.396	-5.421	0.000	-1213.423	-568.982
UNIT_R068	-1294.5733	164.403	-7.874	0.000	-1616.806	-972.341
UNIT_R069	-834.2241	161.536	-5.164	0.000	-1150.837	-517.611
UNIT_R070	-24.7494	155.670	-0.159	0.874	-329.866	280.367
UNIT_R080	1799.5786	155.670	11.560	0.000	1494.462	2104.695
UNIT_R081	1752.1834	156.507	11.196	0.000	1445.427	2058.939
UNIT_R082	-299.7230	156.510	-1.915	0.055	-606.486	7.040
UNIT_R083	1313.9872	155.670	8.441	0.000	1008.871	1619.104
UNIT_R084	8218.2614	155.670	52.793	0.000	7913.145	8523.378
UNIT_R085	806.5853	156.511	5.154	0.000	499.821	1113.350
UNIT_R086	787.5786	155.670	5.059	0.000	482.462	1092.695
UNIT_R087	-567.1675	157.363	-3.604	0.000	-875.603	-258.732
UNIT_R089	-1286.9040	156.506	-8.223	0.000	-1593.660	-980.148
UNIT_R090	-1341.8939	164.408	-8.162	0.000	-1664.136	-1019.652
UNIT_R091	-644.1590	162.951	-3.953	0.000	-963.547	-324.771
UNIT_R092	217.7250	160.183	1.359	0.174	-96.236	531.686
UNIT_R093	253.1256	161.085	1.571	0.116	-62.605	568.856
UNIT_R094	-27.2503	156.669	-0.174	0.862	-334.325	279.825
UNIT_R095	388.8047	157.952	2.462	0.014	79.215	698.394
UNIT_R096	559.6432	156.248	3.582	0.000	253.394	865.892
UNIT_R097	1182.8428	156.245	7.570	0.000	876.599	1489.087
UNIT_R098	22.1377	155.670	0.142	0.887	-282.979	327.254
UNIT_R099	579.1646	155.670	3.720	0.000	274.048	884.281
UNIT_R100	-1229.8273	157.531	-7.807	0.000	-1538.590	-921.064
UNIT_R101	1013.5571	155.670	6.511	0.000	708.441	1318.674
UNIT_R102	1902.3958	155.670	12.221	0.000	1597.279	2207.512

UNIT_R103	-356.1450	161.997	-2.198	0.028	-673.662	-38.628
UNIT_R104	-439.9872	156.672	-2.808	0.005	-747.067	-132.908
UNIT_R105	1552.0947	155.670	9.970	0.000	1246.978	1857.211
UNIT_R106	-637.2247	164.392	-3.876	0.000	-959.436	-315.013
UNIT_R107	-1230.2332	164.885	-7.461	0.000	-1553.410	-907.056
UNIT_R108	3441.2614	155.670	22.106	0.000	3136.145	3746.378
UNIT_R111	1438.3851	155.670	9.240	0.000	1133.268	1743.502
UNIT_R112	-98.6703	156.248	-0.631	0.528	-404.919	207.578
UNIT_R114	-871.3224	156.330	-5.574	0.000	-1177.732	-564.913
UNIT_R115	-493.4834	155.915	-3.165	0.002	-799.080	-187.887
UNIT_R116	1417.5302	155.670	9.106	0.000	1112.414	1722.647
UNIT_R117	-846.8785	163.902	-5.167	0.000	-1168.130	-525.627
UNIT_R119	101.8474	158.812	0.641	0.521	-209.427	413.122
UNIT_R120	-240.9131	161.068	-1.496	0.135	-556.610	74.784
UNIT_R121	-278.9752	160.007	-1.744	0.081	-592.591	34.641
UNIT_R122	806.9208	157.955	5.109	0.000	497.326	1116.516
UNIT_R123	-132.9485	158.228	-0.840	0.401	-443.078	177.181
UNIT_R124	-1082.3844	162.318	-6.668	0.000	-1400.531	-764.238
UNIT_R126	79.4388	155.670	0.510	0.610	-225.678	384.555
UNIT_R127	3020.1485	155.670	19.401	0.000	2715.032	3325.265
UNIT_R137	674.3092	155.832	4.327	0.000	368.876	979.742
UNIT_R139	760.2467	156.087	4.871	0.000	454.314	1066.179
UNIT_R163	1555.8851	155.670	9.995	0.000	1250.768	1861.002
UNIT_R172	121.9227	155.670	0.783	0.434	-183.194	427.039
UNIT_R179	5002.6969	155.670	32.137	0.000	4697.580	5307.813
UNIT_R181	13.9020	158.671	0.088	0.930	-297.096	324.900
UNIT_R183	-939.5923	164.894	-5.698	0.000	-1262.787	-616.397
UNIT_R184	-716.2871	162.793	-4.400	0.000	-1035.365	-397.210
UNIT_R186	-674.7405	157.798	-4.276	0.000	-984.027	-365.454
UNIT_R188	560.4255	156.087	3.590	0.000	254.492	866.358
UNIT_R189	-345.8153	159.559	-2.167	0.030	-658.555	-33.076
UNIT_R194	247.8730	158.665	1.562	0.118	-63.114	558.859
UNIT_R196	-432.5871	156.937	-2.756	0.006	-740.186	-124.988
UNIT_R198	348.6257	156.511	2.227	0.026	41.861	655.391
UNIT_R199	-1030.5757	158.665	-6.495	0.000	-1341.563	-719.588
UNIT_R200	-679.1048	157.604	-4.309	0.000	-988.011	-370.198
UNIT_R202	461.7249	156.663	2.947	0.003	154.663	768.787
UNIT_R203	21.0812	160.463	0.131	0.895	-293.429	335.592
UNIT_R204	-316.5773	155.670	-2.034	0.042	-621.694	-11.461
UNIT_R205	-249.2621	157.515	-1.582	0.114	-557.995	59.470
UNIT_R207	236.6168	156.091	1.516	0.130	-69.326	542.559
UNIT_R208	763.2296	157.519	4.845	0.000	454.489	1071.970
UNIT_R209	-915.3482	163.754	-5.590	0.000	-1236.310	-594.386
UNIT_R210	-1199.3259	159.560	-7.516	0.000	-1512.068	-886.584
UNIT_R211	640.8528	155.670	4.117	0.000	335.736	945.969
UNIT_R212	-72.7965	156.087	-0.466	0.641	-378.729	233.136
UNIT_R213	-578.5329	159.115	-3.636	0.000	-890.401	-266.665
UNIT_R214	-1039.0609	164.246	-6.326	0.000	-1360.986	-717.136
UNIT_R215	-161.4205	156.511	-1.031	0.302	-468.185	145.344
UNIT_R216	-990.7873	156.509	-6.331	0.000	-1297.547	-684.027
UNIT_R217	-709.9897	163.745	-4.336	0.000	-1030.934	-389.045
UNIT_R218	224.7727	156.252	1.439	0.150	-81.484	531.030
UNIT_R219	-489.6692	156.665	-3.126	0.002	-796.736	-182.602
UNIT_R220	-284.4214	155.670	-1.827	0.068	-589.538	20.695

UNIT_R221	−306.8964	164.244	−1.869	0.062	−628.817	15.024
UNIT_R223	363.5892	156.248	2.327	0.020	57.340	669.838
UNIT_R224	−1001.7153	160.008	−6.260	0.000	−1315.335	−688.095
UNIT_R225	−1176.8079	157.798	−7.458	0.000	−1486.095	−867.521
UNIT_R226	−977.1947	162.950	−5.997	0.000	−1296.580	−657.809
UNIT_R227	−665.1580	155.670	−4.273	0.000	−970.275	−360.041
UNIT_R228	−620.4966	163.271	−3.800	0.000	−940.511	−300.483
UNIT_R229	−1145.6076	162.317	−7.058	0.000	−1463.751	−827.464
UNIT_R230	−1204.2141	160.466	−7.504	0.000	−1518.730	−889.698
UNIT_R231	−777.9833	157.796	−4.930	0.000	−1087.267	−468.700
UNIT_R232	−694.7519	161.849	−4.293	0.000	−1011.979	−377.525
UNIT_R233	−597.6667	165.229	−3.617	0.000	−921.518	−273.815
UNIT_R234	−1388.2906	163.755	−8.478	0.000	−1709.253	−1067.328
UNIT_R235	809.1686	156.088	5.184	0.000	503.232	1115.105
UNIT_R236	−219.9724	157.509	−1.397	0.163	−528.693	88.749
UNIT_R237	−1056.0775	163.423	−6.462	0.000	−1376.390	−735.765
UNIT_R238	347.1199	156.248	2.222	0.026	40.870	653.370
UNIT_R239	−867.9967	155.670	−5.576	0.000	−1173.113	−562.880
UNIT_R240	906.2410	156.936	5.775	0.000	598.643	1213.839
UNIT_R242	−1185.8108	159.107	−7.453	0.000	−1497.664	−873.958
UNIT_R243	−349.5289	161.532	−2.164	0.030	−666.135	−32.923
UNIT_R244	−152.4518	161.529	−0.944	0.345	−469.051	164.147
UNIT_R246	−1038.2968	163.286	−6.359	0.000	−1358.340	−718.253
UNIT_R247	−1554.6611	166.234	−9.352	0.000	−1880.482	−1228.840
UNIT_R248	1342.5764	156.087	8.601	0.000	1036.644	1648.509
UNIT_R249	−382.1344	158.227	−2.415	0.016	−692.262	−72.007
UNIT_R250	−766.8243	159.114	−4.819	0.000	−1078.691	−454.957
UNIT_R251	−478.8563	156.931	−3.051	0.002	−786.444	−171.269
UNIT_R252	−798.2883	156.510	−5.101	0.000	−1105.051	−491.525
UNIT_R253	−1040.5868	161.995	−6.424	0.000	−1358.101	−723.073
UNIT_R254	824.9967	156.256	5.280	0.000	518.732	1131.261
UNIT_R255	−958.2259	157.601	−6.080	0.000	−1267.128	−649.324
UNIT_R256	−704.8519	156.513	−4.503	0.000	−1011.621	−398.083
UNIT_R257	119.1754	155.670	0.766	0.444	−185.941	424.292
UNIT_R258	−237.4648	156.935	−1.513	0.130	−545.061	70.131
UNIT_R259	−817.5787	158.235	−5.167	0.000	−1127.722	−507.436
UNIT_R260	−646.2816	169.899	−3.804	0.000	−979.287	−313.276
UNIT_R261	−169.8427	159.706	−1.063	0.288	−482.869	143.184
UNIT_R262	−1231.9400	166.237	−7.411	0.000	−1557.769	−906.111
UNIT_R263	−1621.7817	159.121	−10.192	0.000	−1933.662	−1309.901
UNIT_R264	−1297.7682	156.093	−8.314	0.000	−1603.713	−991.824
UNIT_R265	−841.4118	162.315	−5.184	0.000	−1159.553	−523.270
UNIT_R266	−854.1027	156.248	−5.466	0.000	−1160.351	−547.854
UNIT_R269	−876.1129	156.508	−5.598	0.000	−1182.873	−569.353
UNIT_R270	−1247.1577	163.280	−7.638	0.000	−1567.189	−927.126
UNIT_R271	−1361.2598	162.799	−8.362	0.000	−1680.350	−1042.170
UNIT_R273	−424.3398	163.772	−2.591	0.010	−745.337	−103.343
UNIT_R274	−784.8426	161.852	−4.849	0.000	−1102.076	−467.610
UNIT_R275	−849.7514	159.724	−5.320	0.000	−1162.815	−536.688
UNIT_R276	−358.2870	155.670	−2.302	0.021	−663.404	−53.170
UNIT_R277	−1246.0191	167.774	−7.427	0.000	−1574.860	−917.178
UNIT_R278	−1361.5246	162.318	−8.388	0.000	−1679.672	−1043.378
UNIT_R279	−978.1753	158.821	−6.159	0.000	−1289.468	−666.883
UNIT_R280	−1085.7043	166.752	−6.511	0.000	−1412.542	−758.867

UNIT_R281	-476.8797	158.229	-3.014	0.003	-787.011	-166.749
UNIT_R282	-179.9558	156.511	-1.150	0.250	-486.721	126.809
UNIT_R284	-989.8915	156.510	-6.325	0.000	-1296.655	-683.128
UNIT_R285	-1130.5629	164.523	-6.872	0.000	-1453.031	-808.094
UNIT_R287	-1152.6667	164.737	-6.997	0.000	-1475.554	-829.780
UNIT_R291	34.9442	155.670	0.224	0.822	-270.172	340.061
UNIT_R294	-803.2795	159.561	-5.034	0.000	-1116.021	-490.537
UNIT_R295	-1163.4263	180.775	-6.436	0.000	-1517.749	-809.103
UNIT_R300	444.0087	155.670	2.852	0.004	138.892	749.125
UNIT_R303	-521.3082	158.227	-3.295	0.001	-831.436	-211.180
UNIT_R304	-691.1363	156.511	-4.416	0.000	-997.901	-384.371
UNIT_R307	-1381.9928	164.900	-8.381	0.000	-1705.199	-1058.786
UNIT_R308	-951.8084	161.546	-5.892	0.000	-1268.442	-635.175
UNIT_R309	-947.1338	161.063	-5.881	0.000	-1262.820	-631.447
UNIT_R310	-436.8201	162.958	-2.681	0.007	-756.220	-117.420
UNIT_R311	-1360.4581	160.463	-8.478	0.000	-1674.969	-1045.948
UNIT_R312	-1433.7632	156.934	-9.136	0.000	-1741.356	-1126.170
UNIT_R313	-1671.4447	165.375	-10.107	0.000	-1995.583	-1347.306
UNIT_R318	-1224.7900	157.360	-7.783	0.000	-1533.219	-916.361
UNIT_R319	-409.7415	158.664	-2.582	0.010	-720.727	-98.756
UNIT_R321	-693.3945	155.670	-4.454	0.000	-998.511	-388.278
UNIT_R322	5.5385	158.667	0.035	0.972	-305.452	316.529
UNIT_R323	-478.4487	161.843	-2.956	0.003	-795.665	-161.233
UNIT_R325	-1412.6329	160.622	-8.795	0.000	-1727.455	-1097.811
UNIT_R330	-776.4219	159.557	-4.866	0.000	-1089.157	-463.687
UNIT_R335	-1328.5186	165.976	-8.004	0.000	-1653.835	-1003.202
UNIT_R336	-1703.8013	165.944	-10.267	0.000	-2029.054	-1378.549
UNIT_R337	-1659.6367	164.019	-10.119	0.000	-1981.118	-1338.156
UNIT_R338	-1806.7732	161.171	-11.210	0.000	-2122.672	-1490.874
UNIT_R341	-1249.9935	157.182	-7.953	0.000	-1558.072	-941.915
UNIT_R344	-1287.5925	165.744	-7.769	0.000	-1612.454	-962.731
UNIT_R345	-1290.9794	160.467	-8.045	0.000	-1605.498	-976.461
UNIT_R346	-514.6591	160.925	-3.198	0.001	-830.076	-199.243
UNIT_R348	-1622.1590	161.062	-10.072	0.000	-1937.844	-1306.474
UNIT_R354	-1537.1902	164.498	-9.345	0.000	-1859.609	-1214.771
UNIT_R356	-675.0145	160.272	-4.212	0.000	-989.151	-360.878
UNIT_R358	-1503.8433	164.496	-9.142	0.000	-1826.258	-1181.428
UNIT_R370	-1285.0821	160.002	-8.032	0.000	-1598.690	-971.474
UNIT_R371	-1069.7195	161.847	-6.609	0.000	-1386.943	-752.496
UNIT_R372	-1055.2353	163.270	-6.463	0.000	-1375.248	-735.222
UNIT_R373	-1133.0518	163.770	-6.919	0.000	-1454.045	-812.059
UNIT_R382	-841.8163	159.797	-5.268	0.000	-1155.022	-528.610
UNIT_R424	-1396.2345	165.231	-8.450	0.000	-1720.091	-1072.378
UNIT_R429	-782.9658	157.099	-4.984	0.000	-1090.883	-475.048
UNIT_R453	3.0551	166.753	0.018	0.985	-323.785	329.895
UNIT_R454	-1652.7769	164.237	-10.063	0.000	-1974.684	-1330.870
UNIT_R455	-1697.5658	163.775	-10.365	0.000	-2018.568	-1376.564
UNIT_R456	-1558.2146	159.564	-9.765	0.000	-1870.962	-1245.467
UNIT_R459	-1798.7634	226.109	-7.955	0.000	-2241.941	-1355.586
UNIT_R464	-1853.4070	163.501	-11.336	0.000	-2173.871	-1532.943

==

Omnibus:	27485.974	Durbin-Watson:	1.609
Prob(Omnibus):	0.000	Jarque-Bera (JB):	821188.592
Skew:	2.642	Prob(JB):	0.00

```
Kurtosis:                      23.837   Cond. No.                     1.44e+16
==============================================================================

Warnings:
[1] Standard Errors assume that the covariance matrix of the errors is correctly specified.
[2] The smallest eigenvalue is 3.09e-26. This might indicate that there are
strong multicollinearity problems or that the design matrix is singular.
"""
```

4.4 Interpretation

Suprisingly 'tempi', 'fog', 'precipi' didn't improve the R^2 at all, so I did not include them in my final model. The specific weather condtion for the time and location improves the R^2 more than the variable rain. Since both measure the same and 'conds' is more specific and a much better predictor, I dropped 'rain' from the model.

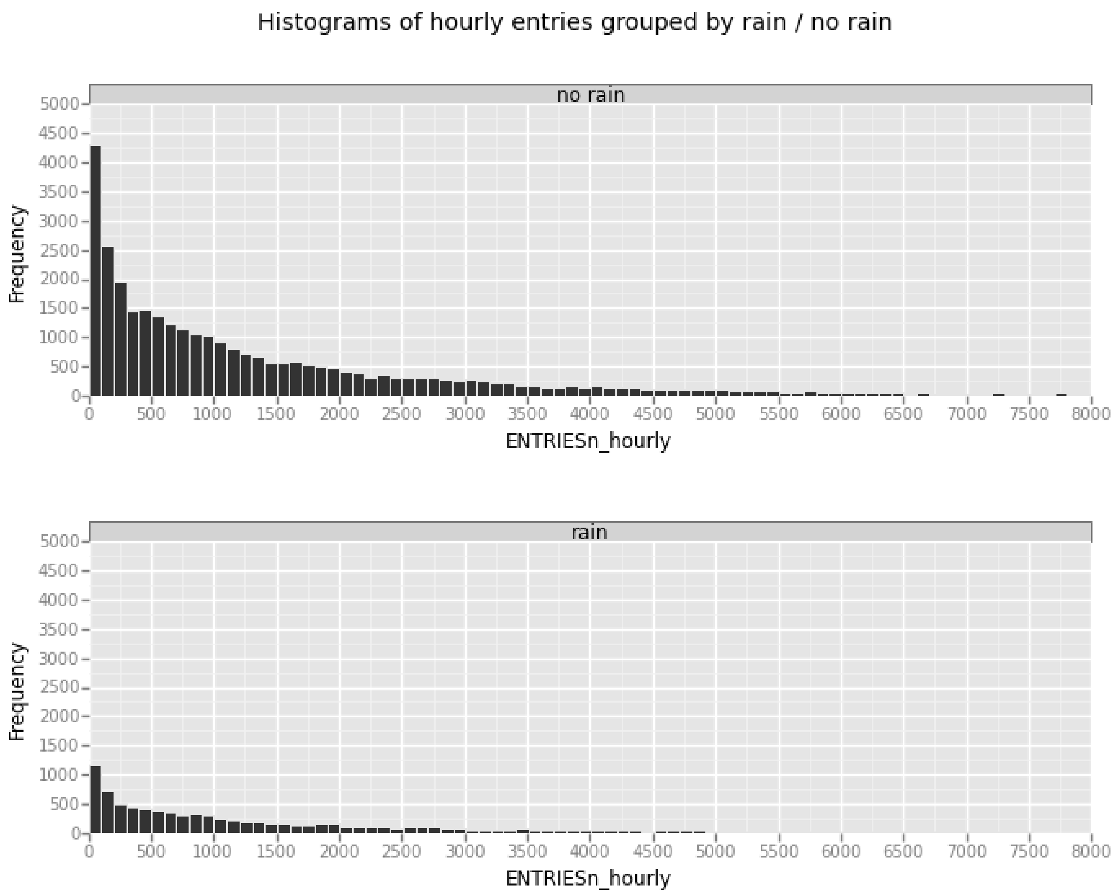
The weekday variable have expected outcomes since the coefficent is negative on weekend and positive during weekdays when people usually go to work. Some weather conditions in the 'conds' variable are insignificant, like fog, haze, mist, mostly cloudy, overcast and partly cloudy. Suprisingly, the coeficient for the different rain types differ in their direction. While light rain has a positive coefienct, light drizzle, rain and heavy rain have a negative coeficient.

R^2 is 0.484 and adjusted R^2 is 0.481. The adjusted R^2 of 0.481 means, that the independent variables in OLS model can predict 48,1% of the variation of the dependent variable of hourly entries. That's nearly half of the variance. Other models than linear may yield a better R^2, but the advantage of a linear model is the interpretability. A linear relationship keeps the model simple and understandable whereas more complex models need more interpretation.

Further examinations with more complex models should be conducted to improve the R^2, if needed.

5 Visualization

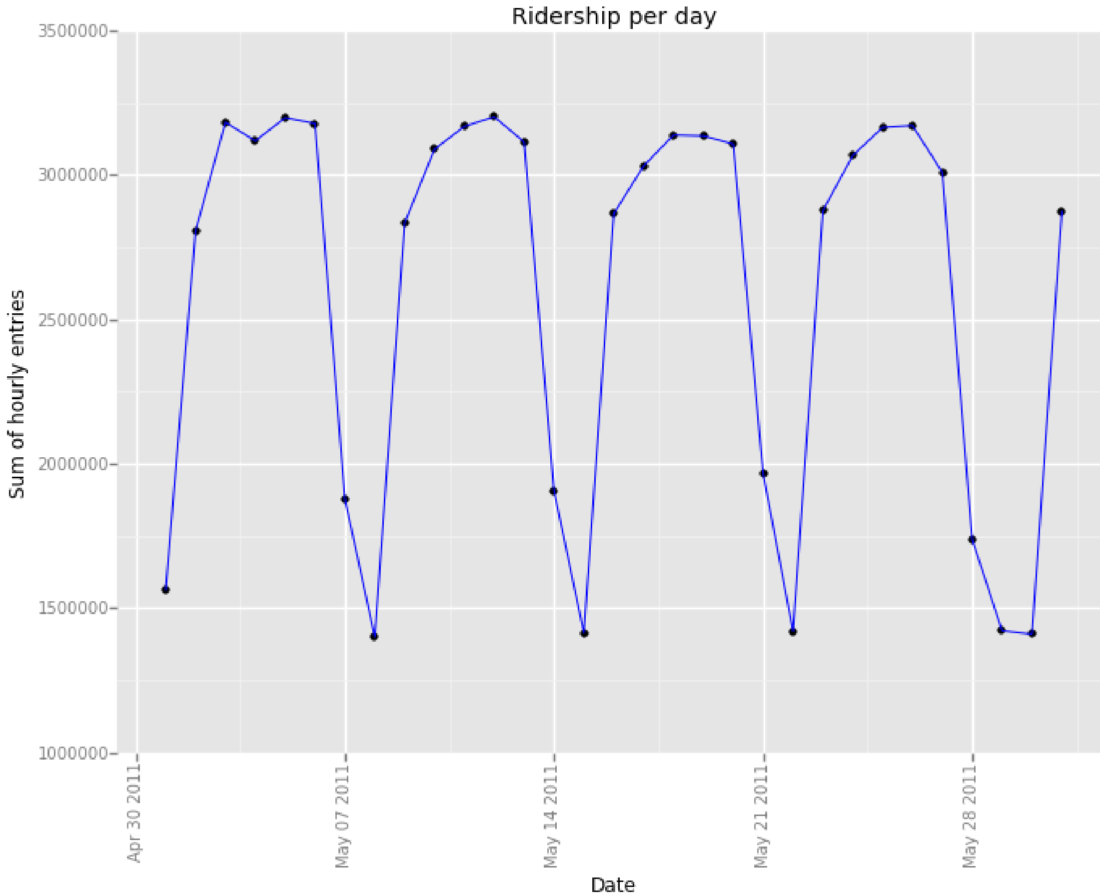
```
In [15]: #add a new coloum with labeled rain for use in the histogram
         data['labeled_rain'] = None
         data['labeled_rain'][data['rain'] == 1] = 'rain'
         data['labeled_rain'][data['rain'] == 0] = 'no rain'

         #plot the histograms by the coloum labeled_rain with the same scale on the y-axis to make it e
         ggplot(data,aes('ENTRIESn_hourly')) + \
                 geom_histogram(binwidth = 100) + \
                 facet_wrap('labeled_rain') + \
                 xlim(0,8000) + \
                 ylim(0,5000) + \
                 ylab('Frequency') + \
                 ggtitle("Histograms of hourly entries grouped by rain / no rain")
```



Out[15]: <ggplot: (-9223372036841969092)>

As we can see in the both histograms above, there is more data on non-rainy days then rainy days. From the description of the data in the introduction we can see that there are 33064 entries on non-rainy days and 9585 on rainy days. Additionally, both samples seem to be non-normal, because both don't have a shape like a bell curve.

In [29]:
```
#let's plot ridership by time-of-day
data_aggregated = data[['ENTRIESn_hourly', 'DATEn']].groupby('DATEn', as_index=False).sum()

#convert the DATEn to a real date in order to avoid errors while plotting
#define a function to extract the date
f = lambda x: datetime.datetime.strptime(x, '%m-%d-%y')
#apply the function on each row and store the result in a new variable called 'extracted_date'
data_aggregated['extracted_date']=data_aggregated['DATEn'].apply(f)

#data_aggregated
ggplot(data_aggregated,aes(x='extracted_date',y='ENTRIESn_hourly')) + \
        geom_line(color='blue') + \
        geom_point(color='black') + \
        xlab('Date') + \
        theme(axis_text_x=element_text(angle=90)) + \
        ylab('Sum of hourly entries') +\
        ggtitle('Ridership per day')
```



Out[29]: <ggplot: (51547333)>

In the ridership per day, we can clearly see that there is a cyclic pattern in the amount of hourly entries. We can clearly see the weekends, for example, May 7th 2011 was a saturday where we can see a drop of the entries and a even further drop on the following sunday. This patterns repeats every seven days, except for May 30th, which could be a public holiday or something else. We need more data for this particular date to investigate this behaviour.

6 Conclusion

There is a statistically difference between the amount of riderships on rainy days and those on non-rainy days. The Mann-Whitney-U-Test showed that both samples are not from the same population and therfore have different distributions. The compared means show that on average there are 183 more people riding the subway on rainy days in those samples. Based on those results, people do ride the NYC subway more when it is raining.

The linear regression though showed a more complex relationship between specific weather conditions and ridership. Suprisingly, the coefficient for the different rain types differ in their direction. While light rain has a positive coefficient, light drizzle, rain and heavy rain have a negative coeficient, meaning in this model less people are riding the subway during those weather conditions. With a adjusted R^2 of 0.481 the Oridnary-Least-Squares model used here can predict 48,1% of the variation of the dependent variable of hourly entries, leaving 51,9% of the variance unexplained due to other factors like unkown variables. Further examinations with more complex models than the OLS should be conducted to improve the R^2.

7 Reflection

The data consists only of one month in may 2011. Predictions may be not accurate for colder months. For example, those rainy conditions that had a negative coefficient could turn positive in winter months. The weekday variable had a significant impact. I strongly suggest to include additional information on public holidays in New York City, as they may have a similar effect since people don't need to go to work.

Given the adjusted R^2 of 48,1%, there is room for improvement because more than half of the variance is not explained by the variables. A possible solution could be the use of non-linear models. They might not be as eays to interpret, but may explain more of the variance left.

8 Supplementary Materials

8.1 References

This is the complete reference list used for writing and perfoming the data analyis above.

Importing csv files:

http://wesmckinney.com/blog/update-on-upcoming-pandas-v0-10-new-file-parser-other-performance-wins/

Installing iPython Notebook and Pandas:

http://twiecki.github.io/blog/2014/11/18/python-for-data-science/

Check for normal distribution with the Shapiro-Wilk-Test:

http://docs.scipy.org/doc/scipy-0.15.1/reference/generated/scipy.stats.shapiro.html

Mann-Whitney-U-Test:

http://docs.scipy.org/doc/scipy-0.14.0/reference/generated/scipy.stats.mannwhitneyu.html

More Explanations on the Mann-Whitney-U-Test:

http://forums.udacity.com/questions/100153716/if-the-mann-whitney-u-test-returns-a-one-sided-p-value-what-is-the-null-hypothesis

Mann-Whitney-U-test with R:

http://www.statmethods.net/stats/nonparametric.html

Performing t-tests:

http://statistics.berkeley.edu/computing/r-t-tests

Performing linear regression OLS with statsmodel:

http://statsmodels.sourceforge.net/0.5.0/generated/statsmodels.regression.linear_model.OLS.html

Histogram with ggplot and how to do it inline:

http://stackoverflow.com/questions/19377371/how-to-make-a-histogram-in-ipython-notebook-using-ggplot2-for-python