Data Analyst Nanodegree

Project 1 - Analyzing the NYC Subway Dataset – Part 2

By
Tom Dzorevski
tomdzorevski@gmail.com
647 405 4752

Section 0. References

Mann–Whitney U test (26 May 2015) Retrieved (2015, May 28) from
http://en.wikipedia.org/wiki/Mann%E2%80%93Whitney_U_test

scipy.stats.mannwhitneyu (Jan 18, 2015) Retrieved (2015, May 28) from
http://docs.scipy.org/doc/scipy-0.15.1/reference/generated/scipy.stats.mannwhitneyu.html

GraphPad Software (n.d.) Analysis checklist: Mann-Whitney test Retrieved (2015, May 28) from
http://www.graphpad.com/guides/prism/6/statistics/index.htm?stat_checklist_mannwhitney.htm

Laerd Statistics (n.d.) Mann-Whitney U test in SPSS Retrieved (2015, May 28) from
https://statistics.laerd.com/premium-sample/mwut/mann-whitney-test-in-spss-2.php

(2015) Problem with Mann-Whitney U test in scipy [Web log post] Retrieved (2015, May 28) from
http://stats.stackexchange.com/questions/116315/problem-with-mann-whitney-u-test-in-scipy

DataRobot Inc.(2015) Ordinary Least Squares in Python [Web blog post] Retrieved (2015, May 28) from
http://www.datarobot.com/blog/ordinary-least-squares-in-python

Section 1. Statistical Test

1.1 Used a Mann-Whitney U-Test to analyze the NYC Subway Dataset to determine if the increase in ridership is statistically significant when it is raining. This is a one-tail test with a p-critical value of α= 0.05 used as the benchmark. The null hypothesis is that the mean of ridership when it is raining and not raining are the same.

Null Hypothesis H_0: $\mu_{rain} = \mu_{no\text{-}rain}$
There is no relationship between ridership when it is raining and not raining.
Alternative Hypothesis H_A: $\mu_{rain} > \mu_{no\text{-}rain}$

There an increase in ridership when it is raining.

$\alpha = 0.05$

1.2 The Mann-Whitney U- Test is applicable in the case because the test assumptions are met by the samples.

Mann-Whitney U-Test Assumption	Applicability
All the observation are independent of each other from both groups	For this test, the turnstile hourly entries are independent from each other.
One dependent variable that is measured as continuous or ordinal values so the values can be ranked	The dependent variable Entries Hourly is district which satisfies the assumption
There is one independent variable that consists of two categorical independent groups.	The independent variable is rain which is divided into two groups rain and no rain
If the distributions of the two groups have the same shape, then the Mann-Whitney U test determines whether there are differences in the medians.	The distributions of both the rain and no-rain samples have similar shapes with a positive skew as illustrated in figure 1.

1.3 The original data sample (turnstile_data_master_with_weather.csv) was analyzed using Python 2.2.9 Anaconda 2.2.0 (32-bit) Windows 7 version. The results are as follows:

No-Rain Sample

mean: $\mu_{no\text{-}rain} = 1090.3$

variance: $\sigma^2_{no\text{-}rain} = 5382422.9$

standard deviation: $\sigma_{no\text{-}rain} = 2320.0$

Rain Sample

mean: $\mu_{rain} = 1105.4$

variance: $\sigma_{rain} = 5382422.9$

standard deviation: $\sigma_{rain} = 2370.5$

Mann-Whitney U-Test

U = 1924409167.0

p = 0.0193[1]

1.4 The means from the samples support the alternative hypothesis H_A: $\mu_{rain} > \mu_{no\text{-}rain}$. Applying the Mann-Whitney U-Test a p= 0.0193 was determined for the one tail test (scipy.stats.mannwhitneyu). Since p <

[1] The p value is different than the value obtained in class Problem Set 3: Analyzing Subway Data > 3 –Mann-Whitney U-Test which had a p = 0.0250 with the same data. This value difference seem to be attributed to the different environments use to compute the p value. Since both p value calculations are less than the critical value, this discrepancy will not affect the analysis

α, the null hypothesis ($H_{0:}$ $\mu_{rain} = \mu_{no\text{-}rain}$) is rejected. There is a 1.93% chance to randomly get a sample with at least a ENTRIES_hourly mean than the rain sample. Thus, the alternative hypothesis (H_A: $\mu_{rain} > \mu_{no\text{-}rain}$) is accepted. Therefore, rain affects the ridership in the NY subway.

Section 2. Linear Regression

2.1 Used the Ordinary Least-Squares (OLS) linear regression from the statsmodels Python package. The input data was the improved data set (turnstile_weather_v2.csv). Note the original data set could not be used because when processing the data, the Anaconda python window 32-bit version, encountered a memory error.

2.2 The features in the used by the model are 'rain', 'precipi', 'hour', 'meantempi', 'holiday', and 'fog' with the additional dummy variables on the variables 'UNIT' and 'day_week'.

2.3 The features selected were select for the following reasons:

> 'rain': The rain vs no-rain histograms as in figure 1 indicated an increase in ridership when it was raining.
>
> 'precipi': Ridership might be impacted by the amount of precipitation. Suspect a drizzle would have less impact on ridership than a down pour.
>
> 'hour': The hour of the day would logically impact ridership. During work days ridership would be higher when people are traveling to and from work. Also, ridership would be less when most people are sleeping. Also, adding it greatly increased the R^2 value.
>
> 'meatempi': For extreme temperatures, it is likely that more people would take the subway to avoid the extremes. For example, in extremely cold temperatures, individuals maybe more inclined to take the subway to avoid the cold.
>
> 'holiday': In the data sample, the ridership on Memorial Day May 30, 2011 was significantly lower than other Mondays as illustrated in figure 2. Thus, the 'holiday' variable was added to take into account the effects of holidays.
>
> 'UNIT': This variable was converted to a dummy variable that greatly increased the R^2 value.
>
> 'day_week': This variable was converted to a dummy variable which resulted in a significantly increased the R^2 value.

2.4 These are the coefficients of the non-dummy variables.

Variable	Coefficient	[95.0% Conf. Int.]	
rain	55.558	0.211,	110.906
precipi	-3289.0988	-4176.005,	-2402.193
hour	122.5949	119.682,	125.508
meantempi	-5.8304	-9.381,	-2.280
holiday	-908.3306	-1048.762,	-767.899
fog	-908.3306	-598.133,	-165.903

2.5 The R^2 (coefficient of determination) is 0.489.

2.6 The R^2 indicates that 48.9% of variance in the response variable 'Entries_hourly' can be explained by the explanatory variables. The remaining 51.1% can be attributed to lurking variables or inherent variability. I feel that this linear model is not appropriate to predict ridership for this dataset, because the R^2 value is only around 50%.

Section 3. Visualization

3.1 Histogram of Entries Hourly

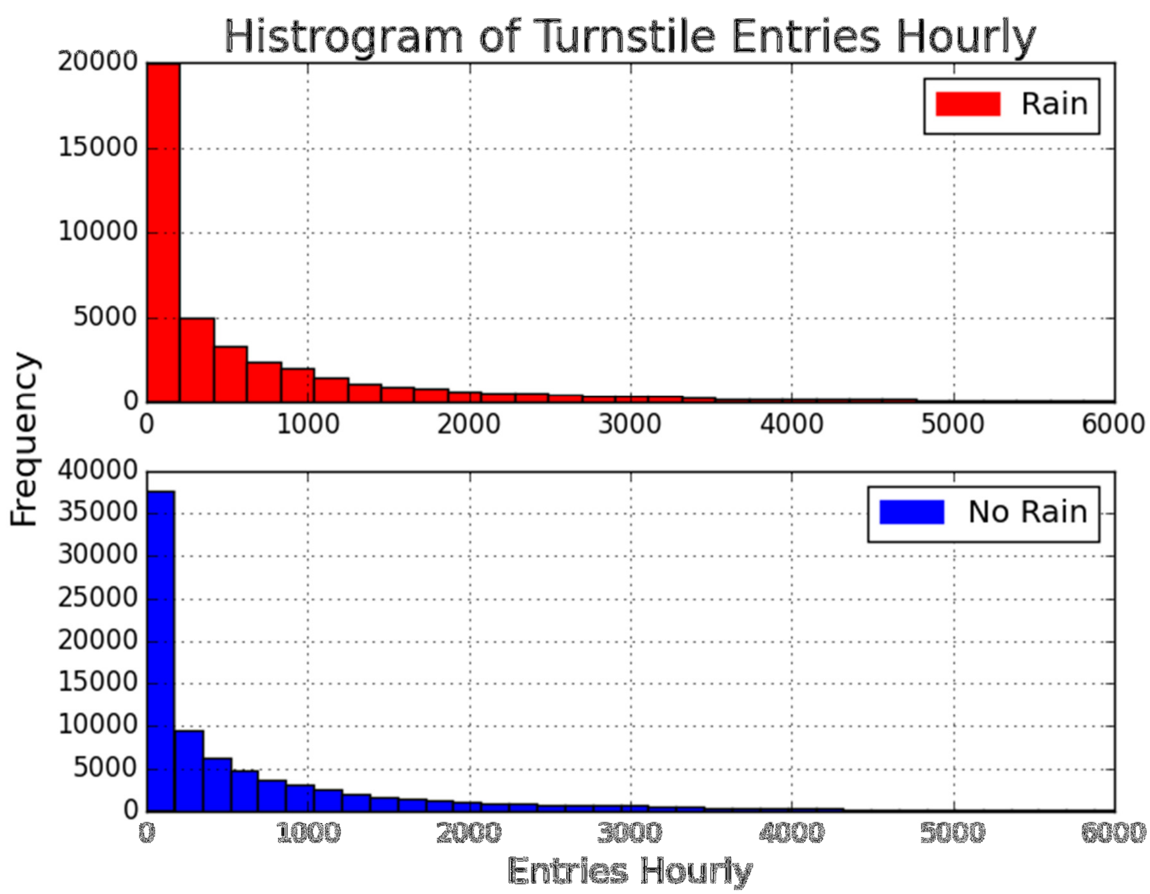

Figure 1 - Histogram of Entries Hourly using the original data set (turnstile_data_master_with_weather.csv)

In figure 1, the Rain sample size is 44,104 which much smaller than the No Rain sample size of 87,847. Also, the x-axis has been truncated at 6,000, which cuts off the outliers in the long tails that extend past 50,000. Figure 1 illustrates both distributions are positively skewed, and they have similar shapes.

3.2 Visualization of Ridership by day-of-week

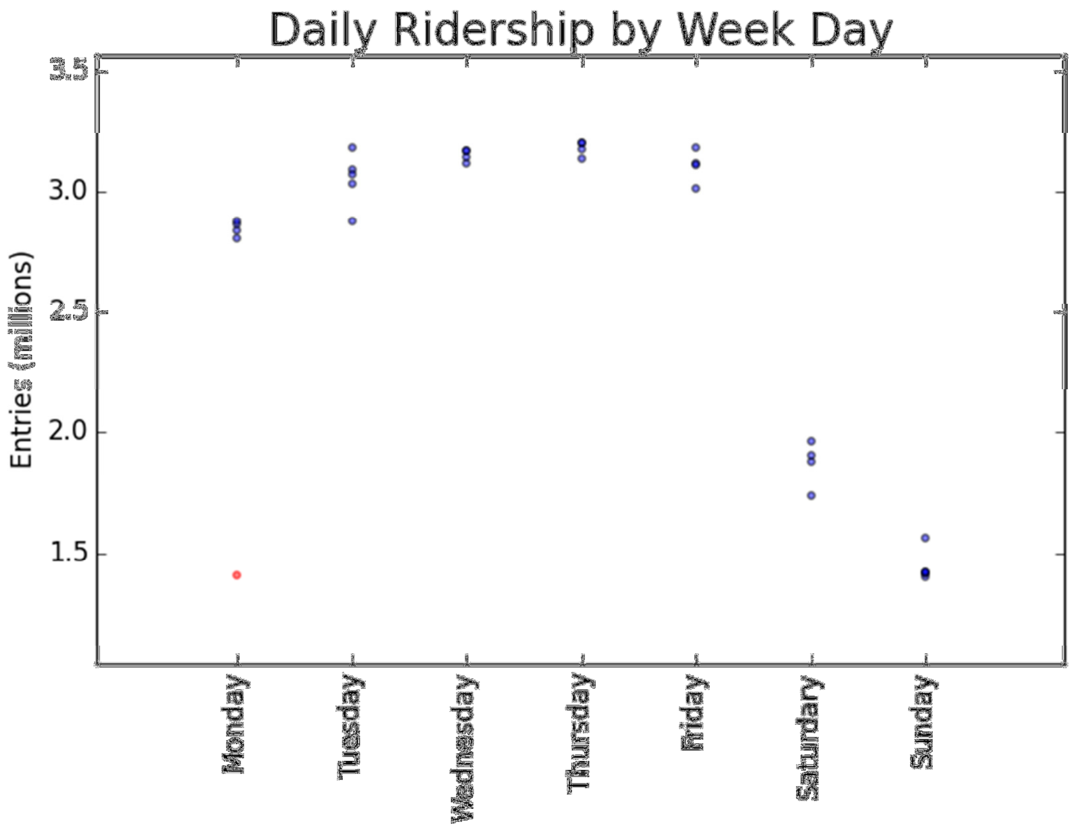

Figure 2 - Ridership by Day

Figure 2 illustrates the ridership by day. The red point indicates a holiday (Memorial Day). The holiday has significantly smaller ridership than a normal Monday workday. The ridership is significantly higher on workdays than weekends or holidays.

Section 4. Conclusion

The ridership of the NY Subway increases when it is raining. The analysis of the data showed that the mean of turnstile hourly entries increased to $\mu_{rain} = 1105.4$ from $\mu_{no\text{-}rain} = 1090.3$ when it is raining compared to not raining. A one-tail statistical test with a null hypothesis of H_0: $\mu_{rain} = \mu_{no\text{-}rain}$, an alternative hypothesis of H_A: $\mu_{rain} > \mu_{no\text{-}rain}$ and alpha level of $\alpha = 0.05$ was used to compare the samples. A Mann-Whitney U-Test was applied on the samples, and it produced a p value of $p = 0.0193$. Since $p < \alpha$, the null hypothesis was reject and the alternative hypothesis was accepted. Conclude the difference in the means is statistically significant. A OLS linear regression was applied to the data with the rain attribute as feature. The rain feature had coefficient of 55.558 and 95% confidence interval of 0.211 to 110.906. Thus, the positive coefficient and range further supports the alternative hypothesis that rain would increase NY Subway ridership.

Section 5. Reflection

The OLS linear regression only had a R^2 of 0.489 which means the 48.9% of the variance can be attributed to the explanatory variables, with 51.1% attributed to lurking variables or inherent variance. Most likely there are lurking variables that are not part of the data set. Some potential lurking variables are sporting events, concerts, parades, celebrations, tourism, and season. As outlined in figure 2, holidays like Memorial Day significantly impact ridership. Potentially a non-linear model may provide a better prediction function. The statistical test used just analyzed the rain variable. Additional tests can be performed on other variables to determine if there is a statistical difference by other variables. Also a different statistical test could be used to further investigate the Null Hypothesis. The sample data was for only for a single month. Data for the additional months may provide additional incites regarding the data and other variables that will affect ridership.

Appendix 1 – Code Module used in the Analysis

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import matplotlib.patches as mpatches
import statsmodels.api as sm

path = r'D:\Toms\BigData\UdaCity\IntroToDataScience' + '\\'

def nyc_trunstile_data_with_weather():
          '''
          Return the original dataset as a DataFrame
          '''
          return pd.read_csv(path + 'turnstile_data_master_with_weather.csv')

def nyc_trunstile_V2():
          '''
          Return the enhanced dataset as a DataFrame
          '''

          return pd.read_csv(path + 'turnstile_weather_v2.csv')


def scatter_plot_ridership_by_day(data):
          '''
          Return a scatter plot of Ridership by Day
          It will highlight the memorial holiday
          The legend could not be added because was obscuring the
          Friday data points.
          '''

          grp = data.groupby(['day_week', 'DATEn'])['ENTRIESn_hourly']
          entriesByDayWithOutHoliday = {}
          entriesByDay = {}
          holidaysByDay = {}
          holidays = []
          holidaysDays = []
          entries = []
          entryDays = []
          for (k1, k2), values in grp:
                    if k2 != '05-30-11':
                              entriesByDayWithOutHoliday.setdefault(k1, []).append(np.sum(values))
                    entriesByDay.setdefault(k1, []).append(np.sum(values))
                    if k2 == '05-30-11':
                              holidaysByDay.setdefault(k1, []).append(np.sum(values))
                    holidaysByDay.setdefault(k1, [])
                    #print(k1, k2, k2 == '05-30-11', '  ', len(values), '  ', np.sum(values), '  ', np.mean(values))
          #print(.setdefault(k1, []).append[np.sum(values)])
          meanRidershipByDay = []
          days = [0 ,1, 2, 3, 4, 5, 6]
          labels = [ 'Monday', 'Tuesday', 'Wednesday', 'Thursday', 'Friday', 'Saturday', 'Sunday']
          labels2 = []
          holidayLabels = []
          for day in days:
                    meanRidershipByDay.append(np.mean(entriesByDayWithOutHoliday[day]))
                    dayEntries = entriesByDayWithOutHoliday[day]
                    for dayEntry in dayEntries:
                              entries.append(dayEntry)
                              entryDays.append(day)
                              labels2.append(labels[day])

                    holidayDayEntries = holidaysByDay[day]
                    if len(holidayDayEntries) == 0:
                              holidays.append(None)
                              holidaysDays.append(day)
                              holidayLabels.append(labels[day])
                    for dayEntry in holidayDayEntries:
                              holidays.append(dayEntry)
                              holidaysDays.append(day)
                              holidayLabels.append(labels[day])

          #print(entryDays)
          #print(entries)

          plot = plt.figure()
          plt.scatter(entryDays, entries, alpha=0.5, s=10)
          plt.title("Daily Ridership by Week Day", fontsize=20)
          plt.xticks(entryDays, labels2, rotation='vertical')
          plt.scatter(holidaysDays, holidays, alpha=0.5, color='red', s=10)
          # ytikcs
          locs,labels = plt.yticks()
          plt.yticks(locs, map(lambda x: "%.1f" % x, locs/1e6))
          plt.ylabel('Entries (millions)')
          plt.margins(0.2)
          plt.subplots_adjust(bottom=0.25)
          return plt

def histograms_entries_hourly_rain_norain_2_subplots(data):
          '''
          Return a plot containing 2 histogram subplots of entries
```

```
        hourly for sample rain and no-rain.
        The x-axis is clipped at 6000.
        '''

        rain_df = data[data['rain'] == 1]
        noRain_df = data[data['rain'] == 0]
        fig = plt.figure()
        ax1 = plt.subplot(2,1,1)
        rain_df['ENTRIESn_hourly'].hist(bins=250, color='red')
        plt.xlim(0,6000)
        red_patch = mpatches.Patch(color='red', label='Rain')
        plt.legend(handles=[red_patch])
        plt.title('Histrogram of Turnstile Entries Hourly', fontsize=20)

        ax2= plt.subplot(2,1,2)
        noRain_df['ENTRIESn_hourly'].hist(bins=250, color='blue')
        #plt.ylim(0,15000)
        plt.xlim(0,6000)
        #ax.ylabel('Frequency')
        plt.xlabel('Entries Hourly', fontsize= 16)
        fig.text(0.03, 0.5, 'Frequency', ha='center', va='center', rotation='vertical', fontsize=16)
        #ax2.set_title("No Rain")
        blue_patch = mpatches.Patch(color='blue', label='No Rain')
        plt.legend(handles=[blue_patch])

        return plt

def ols_estimate_prediction(weather_turnstile):
        '''
        Return a tuple of the OLS estimate and its prediction
        '''

        weather_turnstile['holiday'] = weather_turnstile['DATEn'].map(lambda x: int(x == '05-30-11'))
        variables = ['rain', 'precipi', 'hour', 'meantempi', 'holiday', 'fog']

        features = weather_turnstile[variables]
        #print(features)
    # Add UNIT to features using dummy variables
        dummy_units = pd.get_dummies(weather_turnstile['UNIT'], prefix='unit')
        dummy_days = pd.get_dummies(weather_turnstile['day_week'], prefix='day')
        #print(dummy_units)
        features = features.join(dummy_units)
        features = features.join(dummy_days)
        #print(features)
        y = weather_turnstile['ENTRIESn_hourly']
        X = sm.add_constant(features)
        est = sm.OLS(y, X)
        est = est.fit()
        #print(variables)
        #print(est.summary())
        prediction = est.predict(X)

        return est, prediction
```