MCurcio-AB-Testing

July 27, 2023

1 'catsVSdogs.com' A/B Testing Analysis

To: Magnimind

From: Matt Curcio, matt.curcio.us@gmail.com

Date: 2023-07-01

Re: 'catsVSdogs.com' AB Testing

1.1 Executive Summary

Note: This report describes work for the mythical company, catsVSdogs.com.

Market research showed that using animal icons on the purchase page of www.catsVSdogs.com might promote a higher conversion rate, i.e. better sales. These cat and the dog icons were chosen for AB testing.

Order Icon: Cat vs. Dog

Control	Exposed

1.1.1 Conclusions

1. Fisher's Exact Test (P-value = 0.531): There is no association between the two icons and any change in conversion rates.
2. Phi Coefficient (Φ = -0.0183): There is no relationship between the two icons and a change in conversion rates.
3. Z-Test (P-value = 0.518): There is no relationship between the two icons and a change in conversion rates.
4. **Do not change the icons for www.catsVSdogs.com at this time.**

1.2 Introduction

This A/B Test was carried out between July 3-10, 2020, Friday to Friday with 1243 participants.

Experimental Data	Proportion
Control: Conversion-Ratio using Cat icon	322 / 586 = 54.9%
Exposed: Conversion-Ratio using Dog icon	349 / 657 = 53.1%

1.2.1 Three statistical tests:

1. Z-test for Proportions,
2. Fisher's Exact Test,
3. Phi-Coefficient test for comparison.

1.2.2 Graphics

1. Barplot of dates
2. Histogram of Hours Vs Counts
3. Browser Word Cloud
4. Device Make Word Cloud

1.3 Initial Data Analysis

Data can be found at: **ad-ab-testing**, https://www.kaggle.com/datasets/osuolaleemmanuel/ad-ab-testing.

[2]:
```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
%matplotlib inline

# conda install -c conda-forge wordcloud
from wordcloud import WordCloud
from scipy.stats import norm
import scipy.stats as stats
```

[3]:
```
path = 'data/'
filename = 'AdSmartABdata.csv'
df = pd.read_csv(path+filename)

print('\nDataframe dimensions: ',
      df.shape[0], 'Observations &', df.shape[1], 'features')

df.head(5)
```

```
Dataframe dimensions:  8077 Observations & 9 features
```

[3]:

	auction_id	experiment	date	hour	\
0	0008ef63-77a7-448b-bd1e-075f42c55e39	exposed	2020-07-10	8	
1	000eabc5-17ce-4137-8efe-44734d914446	exposed	2020-07-07	10	
2	0016d14a-ae18-4a02-a204-6ba53b52f2ed	exposed	2020-07-05	2	
3	00187412-2932-4542-a8ef-3633901c98d9	control	2020-07-03	15	
4	001a7785-d3fe-4e11-a344-c8735acacc2c	control	2020-07-03	15	

	device_make	platform_os	browser	yes	no
0	Generic Smartphone	6	Chrome Mobile	0	0
1	Generic Smartphone	6	Chrome Mobile	0	0
2	E5823	6	Chrome Mobile WebView	0	1
3	Samsung SM-A705FN	6	Facebook	0	0
4	Generic Smartphone	6	Chrome Mobile	0	0

1.3.1 Note 1

1. The `zip` file has NO descriptive information on the columns `auction_id` and `platform_os` columns. These variables will not be used in this analysis.

[4]:
```
# Check for NULLS: NO NULLS FOUND

df.isnull().sum()
```

[4]:
```
auction_id     0
experiment     0
date           0
hour           0
device_make    0
platform_os    0
browser        0
yes            0
no             0
dtype: int64
```

[5]:
```
# Reduce all letters to lower case

df['device_make'] = df['device_make'].str.lower()
df['browser'] = df['browser'].str.lower()
df['experiment'] = df['experiment'].str.lower()

# Delete columns 'auction_id'(0) & 'platform_os'(5)

df_mod = df.drop(df.columns[[0, 5]], axis=1, inplace=False)
df_mod.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 8077 entries, 0 to 8076
Data columns (total 7 columns):
 #   Column       Non-Null Count  Dtype 
---  ------       --------------  ----- 
 0   experiment   8077 non-null   object
 1   date         8077 non-null   object
 2   hour         8077 non-null   int64 
 3   device_make  8077 non-null   object
 4   browser      8077 non-null   object
 5   yes          8077 non-null   int64 
 6   no           8077 non-null   int64 
dtypes: int64(3), object(4)
memory usage: 441.8+ KB
```

[6]:
```
df_mod.head()
```

[6]:

	experiment	date	hour	device_make	browser	yes	no
0	exposed	2020-07-10	8	generic smartphone	chrome mobile	0	0
1	exposed	2020-07-07	10	generic smartphone	chrome mobile	0	0
2	exposed	2020-07-05	2	e5823	chrome mobile webview	0	1
3	control	2020-07-03	15	samsung sm-a705fn	facebook	0	0
4	control	2020-07-03	15	generic smartphone	chrome mobile	0	0

1.3.2 Note 2

1. Data does not contain any missing values, therefore all 8077 observations can be used.

TOP

1.4 Exploratory Data Analysis

1.4.1 Date Plot

- The experiment was run between `July 3-10, 2020, (Friday to Friday)`

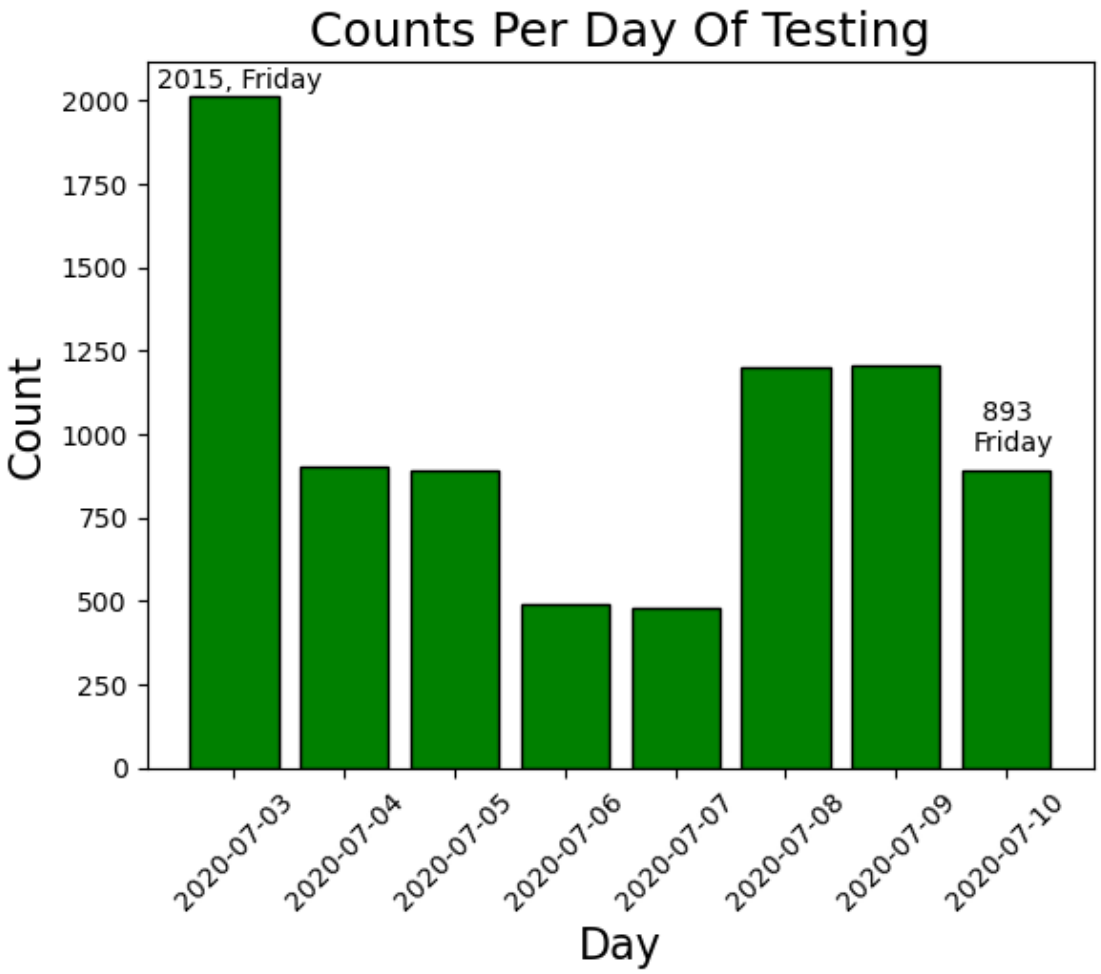
[7]:
```
# Experiment Dates
# Group by date
df_dates = df_mod.groupby('date')['date'].count()
print(df_dates)
type(df_dates)
```

```
date
2020-07-03    2015
2020-07-04     903
2020-07-05     890
2020-07-06     490
2020-07-07     480
2020-07-08    1198
2020-07-09    1208
2020-07-10     893
Name: date, dtype: int64
```

[7]:
```
pandas.core.series.Series
```

1.4.2 Barplot of dates

[8]:
```
df_date = {'Date': ['2020-07-03', '2020-07-04', '2020-07-05', '2020-07-06',␣
 ↪'2020-07-07', '2020-07-08', '2020-07-09', '2020-07-10'],
           'Counts': [2015, 903, 890, 490, 480, 1198, 1208, 893]}

plt.bar(df_date['Date'], df_date['Counts'], color='green', edgecolor='black')
plt.xticks(rotation=45)
plt.title('Counts Per Day Of Testing', fontsize=18)
plt.ylabel('Count', fontsize=16)
plt.xlabel('Day', fontsize=16)
plt.text(-0.7, 2040, "2015, Friday")
plt.text(6.7, 950, " 893\nFriday")
plt.show()
```



1.4.3 Note 3

1. Friday, July 3rd, 2020 traffic = 2015 impressions
2. Friday, July 10th, 2020 traffic = 815 impressions
3. The 247% greater traffic on 7/3 versus 7/10 may be a *novelty* effect shown on the first day.

TOP

1.4.4 Histogram for Hours

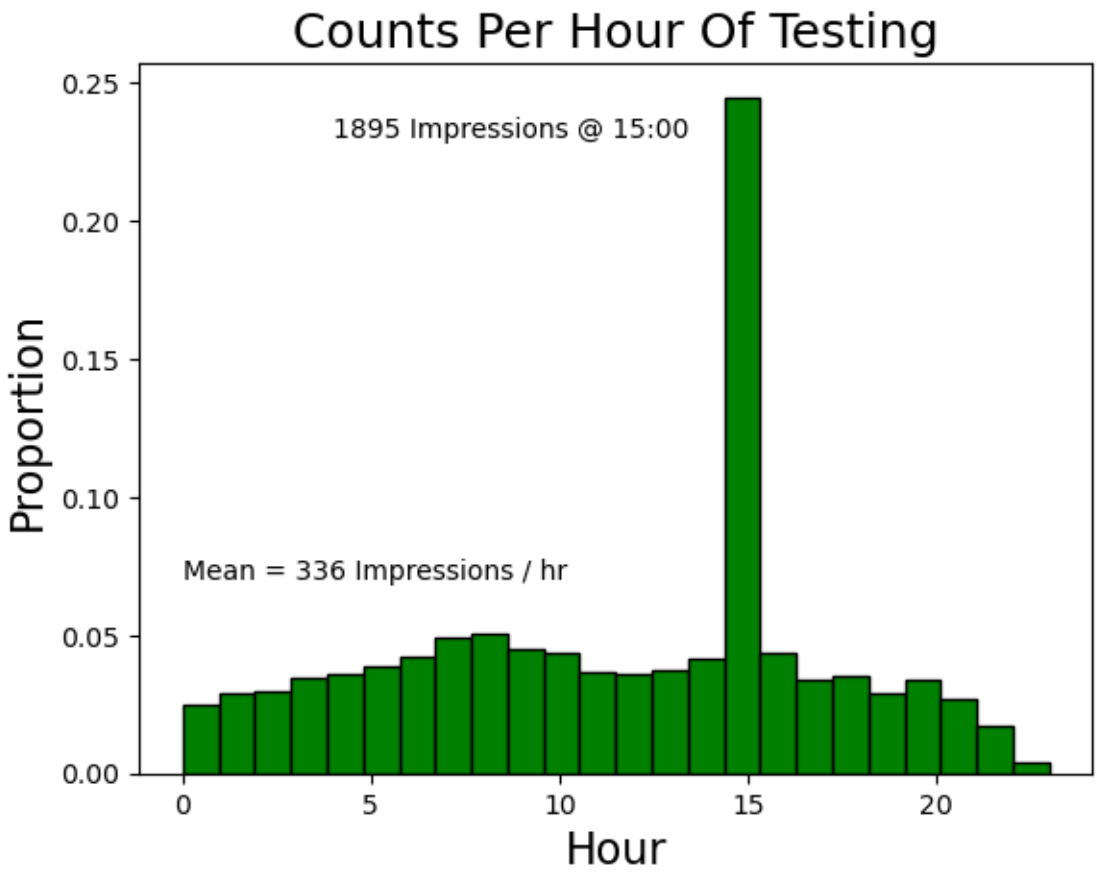
[9]:
```
# Experiment Hour
# Group by Hour
df_hour = df_mod.groupby('hour')['hour'].count()
print(df_hour)
type(df_hour)
print('\nMean over 24 hours =', df_hour.mean())
```

```
hour
0      194
1      222
2      230
3      266
4      281
5      302
6      327
7      381
8      394
9      346
10     336
11     282
12     278
13     290
14     319
15    1895
16     335
17     263
18     273
19     227
20     264
21     206
22     135
23      31
Name: hour, dtype: int64

Mean over 24 hours = 336.5416666666667
```

1.4.5 Histogram of Hours Vs Counts

[10]:
```
# Histogram of Hours Vs Counts
df_hour = df_mod['hour']
df_hour.plot(kind='hist', color='green',
             density=True, edgecolor='black', bins=24)
plt.title('Counts Per Hour Of Testing', fontsize=18)
plt.xlabel('Hour', fontsize=16)
plt.ylabel('Proportion', fontsize=16)
plt.text(4, 0.23, "1895 Impressions @ 15:00")
plt.text(0, 0.07, "Mean = 336 Impressions / hr")
```

[10]:
```
Text(0, 0.07, 'Mean = 336 Impressions / hr')
```



1.4.6 Note 4

1. 1895 impressions were collected at hour 15.
2. The mean of 24 hours is 336 impressions.
3. The **564%** greater traffic on **hr=1500** over the **mean=336** may need to be further investigated.

TOP

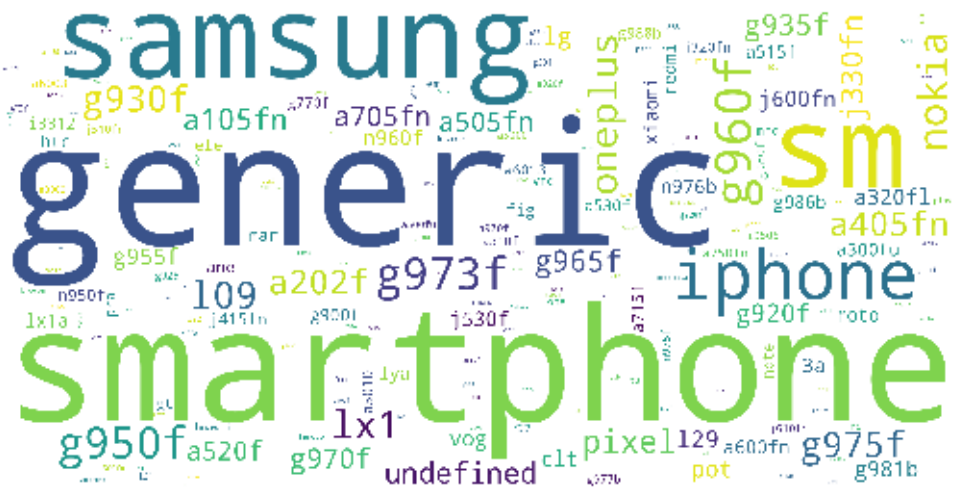
1.4.7 Browser Word Cloud

[11]:
```
# Create text variable
text1 = " ".join(browser for browser in df_mod.browser)

# Creatd word_cloud
word_cloud1 = WordCloud(collocations=False,
                        background_color='white',
                        width=3000,
                        height=1500).generate(text1)
```

```
# Save image
word_cloud1.to_file('figures/browser_wordcloud.png')

# Display Word Cloud
plt.imshow(word_cloud1, interpolation='none')
plt.axis("off")
plt.show()
```

1.4.8 Device Make Word Cloud

[12]:
```
# Create the text variable
text1 = " ".join(device_make for device_make in df_mod.device_make)

# Create word_cloud
word_cloud1 = WordCloud(collocations=False,
                        background_color='white',
                        width=3000,
                        height=1500).generate(text1)
# Save image
word_cloud1.to_file('figures/device_wordcloud.png')

# Display Word Cloud
plt.imshow(word_cloud1, interpolation='none')
plt.axis("off")
plt.show()
```



TOP

1.5 Statistical Tests & Analysis

1.5.1 Fisher's Exact Test

Fisher's Exact Test is a Non-Parametric test. In plain English, Fisher's test does not make the same assumptions as the z-test. In fact, Fisher's test calculates the exact probabality of an occurence.

Fisher's test assumes:

1. Random sampling
2. Independence
3. Mutually exclusive groups

It is appropriate to use Fisher's Exact Test when:

1. Variables are categorical and the result is classified in two different groups.
2. One is looking for the relationship between two variables.
3. The response variables are recorded using counts.

Fisher's Exact Test uses the hypotheses test:

- H0: The two variables are independent.
- H1: The two variables are not independent,
  - i.e. the difference in conversion rates is related to using the different icons.

[13]:
```
# KEEP 'PARTICIPATING' MEMBERS ONLY
# Where 'yes' or 'no' columns are equal to 1.

# REMOVE columns [[0,2,3,4,5,6]] from ORIGINAL dataframe, **df**
df_mod = df.drop(df.columns[[0,2,3,4,5,6]], axis=1, inplace=False)

# KEEP PARTICIPATING MEMBERS ONLY
```

```
df_participants = df_mod[(df_mod['yes'] == 1) | (df_mod['no'] == 1)]
print('\nDimensions of Participants dataframe =', df_participants.shape)

df_participants.head(5)
```

```
Dimensions of Participants dataframe = (1243, 3)
```

[13]:

	experiment	yes	no
2	exposed	0	1
16	exposed	1	0
20	exposed	0	1
23	control	1	0
27	control	0	1

[14]:
```
fishers_a = df_participants[(df_participants['experiment'] == 'control') &
                            (df_participants['no'] == 0)].count()

fishers_b = df_participants[(df_participants['experiment'] == 'control') &
                            (df_participants['no'] == 1)].count()

fishers_c = df_participants[(df_participants['experiment'] == 'exposed') &
                            (df_participants['no'] == 0)].count()

fishers_d = df_participants[(df_participants['experiment'] == 'exposed') &
                            (df_participants['no'] == 1)].count()
```

[15]:
```
# Fisher 2x2 Contingency Table

print('fishers_a =', fishers_a[0])
print('fishers_b =', fishers_b[0])
print('fishers_c =', fishers_c[0])
print('fishers_d =', fishers_d[0])
```

```
fishers_a = 264
fishers_b = 322
fishers_c = 308
fishers_d = 349
```

[16]:
```
df_participants['experiment'].value_counts()
```

[16]:
```
exposed    657
control    586
Name: experiment, dtype: int64
```

[17]:
```
# Counts
```

```
df = pd.DataFrame({'No': [264, 308], 'Yes': [322, 349]},
                  index=pd.Index(['Control', 'Exposed']))
df
```

[17]:

	No	Yes
Control	264	322
Exposed	308	349

[18]:
```
# Proportions Table

df_prop = pd.DataFrame({'No': [0.212, 0.248], 'Yes': [0.259, 0.281]},
                       index=pd.Index(['Control', 'Exposed']))
df_prop
```

[18]:

	No	Yes
Control	0.212	0.259
Exposed	0.248	0.281

[19]:
```
# Fishers exact test on the data

odds_ratio, p_value = stats.fisher_exact(df, alternative="two-sided")
print("\nOdds ratio is: " + str(odds_ratio))
print("\nP-value is: " + str(p_value))
```

```
Odds ratio is: 0.9290150842945873

P-value is: 0.5309716576381456
```

1.5.2 Results of Fisher's Exact Test

1. P-value = 0.531, therefore this result will occur on average 53% of the time.
2. Fisher's test also produces an odds ratio for betting people. Using these icons, the odds of increasing the conversion rates are 0.93 : 1.0.
    - In other words, the odds of increasing sales is ~1:1, no change.

TOP

1.5.3 Phi Coefficient of Association

The Phi Coefficient of Association (Φ) is a measure of the degree of association (cooperative effect) between two binary variables.

The Phi Coefficient test can be interpreted similarly to Pearson's correlation coefficient. Chiefly, does any change in conversion rate positively (or negatively) correlate with changing the two icons?

- H0: There IS NO statistically significant relationship between the change in conversion rate and changing the cat and dog icons.

- H1: There IS a statistically significant relationship between the two variables.

It is appropriate to use Phi in the following scenario:

1. To determine the relationship between two variables
2. The variables of interest are binary
3. There are only two variables

$$\Phi = \frac{a \cdot d - b \cdot c}{\sqrt{efgh}}$$

Where:

Experiment	NO	YES	Sums
Control	a = 264	b = 322	e = 586
Exposed	c = 308	d = 349	f = 657
Sums	g = 572	h = 671	1243

[20]:
```
import math

numerator = (264 * 349)-(308 * 322)

denominator = ((586)*(657)*(572)*(671))**(0.5)

print('\nThe numerator of Phi =', numerator)
print('\nThe denominator of Phi =', math.floor(denominator))
print('\nPhi Coefficient =', numerator/denominator)
```

```
The numerator of Phi = -7040

The denominator of Phi = 384406

Phi Coefficient = -0.018313944421528762
```

$$\Phi = \frac{-7,040}{384,406} = -0.0183$$

The Phi Coefficient test is interpreted similarly to Pearson's correlation coefficient

The Phi Coefficient takes on values between -1 and 1 where: - -1 indicates a perfectly negative relationship between the two variables. - 0 indicates no association between the two variables. - 1 indicates a perfectly positive relationship between the two variables.

1.5.4 Results of Phi Coefficient of Association

1. Since Phi = -0.0183, there is no relationship between the the icons and increased sales.
    - The Phi Coefficient is very close to zero, there is no link between the icons and increased sales.

TOP

1.5.5 Two Sample Z-test

This test uses a simple normal test for proportions. It should be the same as running the mean z-test on the data encoded 1 for event and 0 for no event so that the sum corresponds to the count.

In the one and two sample cases with two-sided alternative, this test produces the same p-value as proportions_chisquare, since the chisquare is the distribution of the square of a standard normal distribution.

https://www.statsmodels.org/stable/generated/statsmodels.stats.proportion.proportions_ztest.html

- H0: $\mu = \mu_0$
- HA: $\mu \neq \mu_0$

[21]:
```
import numpy as np
from statsmodels.stats.proportion import proportions_ztest

count = np.array([322, 349])

nobs = np.array([586, 657])

stat, pval = proportions_ztest(count, nobs)

print('Z-score =', '{0:0.3f}'.format(stat))
print('P-value =', '{0:0.3f}'.format(pval))
```

```
Z-score = 0.646
P-value = 0.518
```

1.5.6 Results for the Z-test

1. Since the P-value = 0.518, the proportions are equal. There is no difference between the two conversion rates.

TOP

1.5.7 Ratios of Respondents to total & Exposed to Control

Conversion Rate: Conversion rate, defined as the proportion of sessions ending up with a transaction.

Conversion Rate = # of converted / total number · 100%

Experiment	NO	YES	Sums
Control	a = 264	b = 322	e = 586
Exposed	c = 308	d = 349	f = 657
Sums	g = 572	h = 671	1243

[22]:
```
1243/8077, 322/586, 349/657
```

[22]:
```
(0.15389377244026248, 0.5494880546075085, 0.5312024353120244)
```

Class	Proportion
Participants vs Total Impressions ratio	15.4%
Control: Conversion-Ratio of Cat-people	54.9%
Exposed: Conversion-Ratio of Dog-people	53.1%

1.6 Conclusions

1. No changes to www.catsVSdogs.com should be made at this time.

2. Fisher's Exact Test (P-value = 0.531) suggests there is no association between the two icons and any change of conversion rates. These results will occur 53% of the time in a random trial.

3. Phi Coefficient of Association (Φ = -0.0183) suggests there is no relationship between the two icons and any change of conversion rates.

4. Z-test (P-value = 0.518) the proportions are equal. There is no difference between the two conversion rates.

5. The **564% greater traffic on hr=1500 versus the mean=336**, as seen on the Hour histogram, could be a serious problem and may need to be investigated.

TOP