MODULE 3.1: CLASSIFICATION AND HIPOTHESIS TESTING

CASE STUDY ACTIVITY TUTORIAL

CASE STUDY 1 - CHALLENGER

CASE STUDY ACTIVITY TUTORIAL

CASE STUDY 1 - CHALLENGER

Faculty: David Gamarnik

In this document, we walk through some helpful tips to get you started with performing the analysis for the Challenger Case Study on your own. In this tutorial, we provide examples and some pseudo-code for the following programming environments: **R**, **Python**. We cover the following:

Topics

The Data

The entire data of observations is shown below. The data consists of three columns:

Observation: this is simply the count of the observation.
Y: this is the failure/success label for each observation. If there was a failure, Y=1. Else, Y=0.
X: this is the temperature (degrees Fahrenheit) at launch.

Copy the table below in to a csv (comma separate variable) file format. A sample is also separately made available with this document. It is titled "challenger-data.csv".

Observation	Y	X
1	1	53
2	1	53
3	1	53
4	0	53
5	0	53
6	1	57
7	0	57
8	0	57
9	0	57
10	0	57
11	1	58
12	0	58
13	0	58

Observation	Y	X
61	0	70
62	0	70
63	0	70
64	0	70
65	0	70
66	0	70
67	0	70
68	0	70
69	0	70
70	0	70
71	0	72
72	0	72
73	0	72

14	0	58
15	0	58
16	1	63
17	0	63
18	0	63
19	0	63
20	0	63
21	0	66
22	0	66
23	0	66
24	0	66
25	0	66
26	0	67
27	0	67
28	0	67
29	0	67
30	0	67
31	0	67
32	0	67
33	0	67
34	0	67
35	0	67
36	0	67
37	0	67
38	0	67
39	0	67
40	0	67
41	0	68
42	0	68
43	0	68
44	0	68
45	0	68
46	0	69
47	0	69
48	0	69
49	0	69
50	0	69
51	1	70
52	0	70
53	0	70
54	0	70
55	0	70
56	1	70
57	0	70
58	0	70
59	0	70
60	0	70

74	0	72
75	0	72
76	0	73
77	0	73
78	0	73
79	0	73
80	0	73
81	1	75
82	1	75
83	0	75
84	0	75
85	0	75
86	0	75
87	0	75
88	0	75
89	0	75
90	0	75
91	0	76
92	0	76
93	0	76
94	0	76
95	0	76
96	0	76
97	0	76
98	0	76
99	0	76
100	0	76
101	0	78
102	0	78
103	0	78
104	0	78
105	0	78
106	0	79
107	0	79
108	0	79
109	0	79
110	0	79
111	0	80
112	0	80
113	0	80
114	0	80
115	0	80
116	0	81
117	0	81
118	0	81
119	0	81
120	0	81

Accessing the Data

The data can be accessed from the csv file "challenger-data.csv" via the following function calls:

In R:

```
data = read.csv("challenger-data.csv")
```

In Python:

We will use the Pandas (and Numpy) libraries to help process the data in Python. Please ensure you have them installed before using the code below.

```
import pandas as pd
import numpy as np

data = pd.read_csv("challenger-data.csv")
```

We now have the data readily accessible for further analysis.

Visualizing the Data

Using our data structure, it is possible to plot the data on a graph. For instance, if we want to plot the frequency of failures at each temperature on a plot, we can do it in the following manner.

In R:

```
install.packages('plyr')                    # we need this package for computing frequency counts
Library(plyr)

failures = subset(data, data$Y == 1)        # subset of the data for just failures
no_failures = subset(data, data$Y == 0)     # subset of the data for no failures

failures_freq = count(failures, 'X')        # count of failures, for each temperature
no_failures_freq = count(no_failures, 'X')  # count of no failures, for each temperature


plot(no_failures$X, integer(length(no_failures$X)), ylim=c(-0.5, 3.5), col='blue', xlab='X:
Temperature', ylab='Number of Failures', pch=19)    # plot the no failures first

points(failures_freq$X, failures_freq$freq, col='red', pch=19)    #add the failures
```

In Python:

We will be using the Matplotlib library for plotting. Pleas ensure it is installed before proceeding.

```
# subsetting data
failures = data.loc[(data.Y == 1)]
no_failures = data.loc[(data.Y == 0)]

# frequencies
```

```
failures_freq = failures.X.value_counts()#failures.groupby('X')
no_failures_freq = no_failures.X.value_counts()

# plotting
import matplotlib as mpl
from matplotlib import pyplot as plt
plt.scatter(failures_freq.index, failures_freq, c='red', s=40)
plt.scatter(no_failures_freq.index, np.zeros(len(no_failures_freq)), c='blue', s=40)
plt.xlabel('X: Temperature')
plt.ylabel('Number of Failures')

plt.show()
```

Logistic Regression

We are now ready to use Logistic Regression.

In R,

```
model = glm(data$Y ~ data$X,family=binomial(link='logit'),data=data)      # build the model
summary(model)                                                            # summarize the model
```

In Python,

You will need to have the following libraries installed before proceeding:

```
patsy
statsmodels
from patsy import dmatrices
import statsmodels.discrete.discrete_model as sm

#get the data in correct format
y, X = dmatrices('Y ~ X', data, return_type = 'dataframe')

#build the model
logit = sm.Logit(y, X)
result = logit.fit()

# summarize the model
print result.summary()
```

We now have the model and the summaries should provide the coefficient, intercept, standard errors and p-values.