City of Chicago, Analysis of the West Nile Virus

EDWIN, LING CHONG GOLD, LOW GUAT HWA, SAPNA

About West Nile Virus:

It is an infectious disease.

It is most commonly spread to people by the bite of an infected mosquito.

Cases of WNV occur during mosquito season , which starts in the summer and continues through fall.

WNV has the potential to cause prolonged disability or death in people who are infected.

There are no vaccines to prevent or medications to treat WNV in people. Since 1999, nearly 2,000 US residents have died from WNV complications

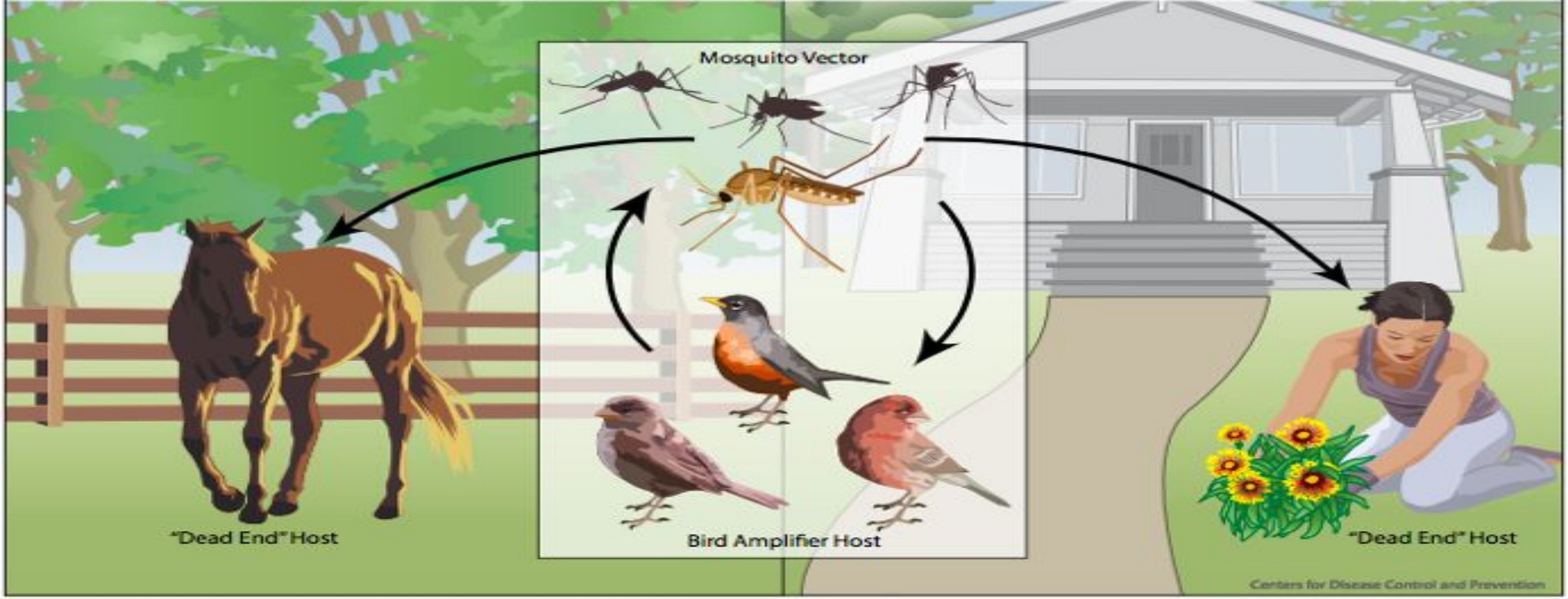
The most effective way to prevent infection from West Nile virus is to prevent mosquito bites.

The surveillance allows local government to take action (mainly in the form of mosquito spraying) to prevent the spread of WNV.

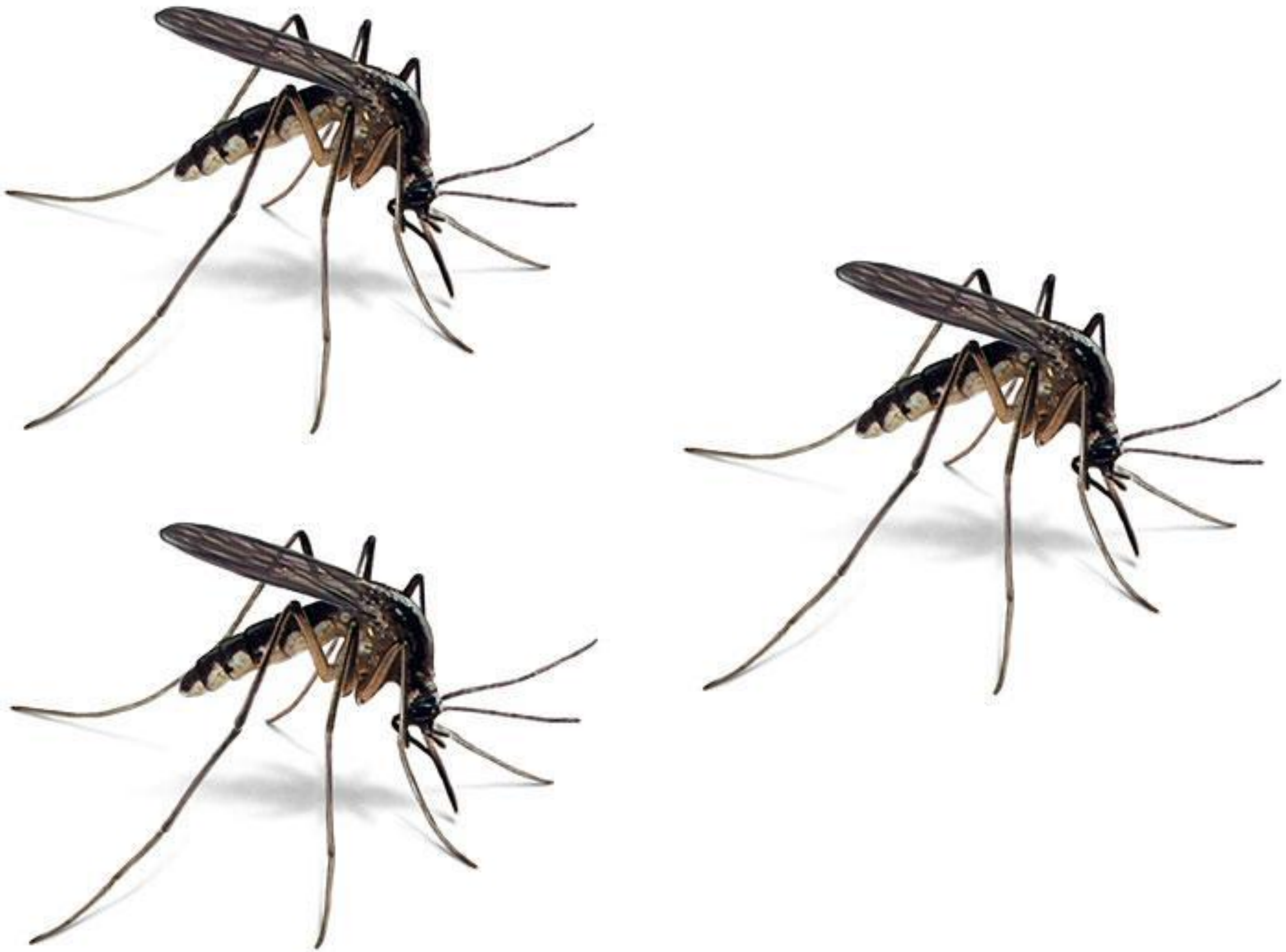
West Nile Virus Transmission Cycle

In nature, West Nile virus cycles between mosquitoes (especially *Culex* species) and birds. Some infected birds, can develop high levels of the virus in their bloodstream and mosquitoes can become infected by biting these infected birds. After about a week, infected mosquitoes can pass the virus to more birds when they bite.

Mosquitoes with West Nile virus also bite and infect people, horses and other mammals. However, humans, horses and other mammals are 'dead end' hosts. This means that they do not develop high levels of virus in their bloodstream, and cannot pass the virus on to other biting mosquitoes.



The main goal of our analytics is to create a model that can help to identify locations with high mosquito population so that pesticides can be sprayed in time to eliminate them.



Load Data

```
train = pd.read_csv('../datasets/original/train.csv')
test = pd.read_csv('../datasets/original/test.csv')
spray = pd.read_csv('../datasets/original/spray.csv')
weather = pd.read_csv('../datasets/original/weather.csv')
traps = pd.read_csv('../datasets/original/train.csv')[['Date', 'Trap','Longitude', 'Latitude', 'WnvPresent']]
```

Train Dataframe (2007/09/11/13)

```
train.head(2)
```

	Date	Address	Species	Block	Street	Trap	AddressNumberAndStreet	Latitude	Longitude	AddressAccuracy	NumMosquitos	WnvPresent
0	2007-05-29	4100 North Oak Park Avenue, Chicago, IL 60634,...	CULEX PIPIENS/RESTUANS	41	N OAK PARK AVE	T002	4100 N OAK PARK AVE, Chicago, IL	41.95469	-87.800991	9	1	0
1	2007-05-29	4100 North Oak Park Avenue, Chicago, IL 60634,...	CULEX RESTUANS	41	N OAK PARK AVE	T002	4100 N OAK PARK AVE, Chicago, IL	41.95469	-87.800991	9	1	0

Test Dataframe (2008/10/12/14)

```
test.head(2)
```

	Id	Date	Address	Species	Block	Street	Trap	AddressNumberAndStreet	Latitude	Longitude	AddressAccuracy
0	1	2008-06-11	4100 North Oak Park Avenue, Chicago, IL 60634,...	CULEX PIPIENS/RESTUANS	41	N OAK PARK AVE	T002	4100 N OAK PARK AVE, Chicago, IL	41.95469	-87.800991	9
1	2	2008-06-11	4100 North Oak Park Avenue, Chicago, IL 60634,...	CULEX RESTUANS	41	N OAK PARK AVE	T002	4100 N OAK PARK AVE, Chicago, IL	41.95469	-87.800991	9

Spray Dataframe (2011/13)

```
spray.head(2)
```

	Date	Time	Latitude	Longitude
0	2011-08-29	6:56:58 PM	42.391623	-88.089163
1	2011-08-29	6:57:08 PM	42.391348	-88.089163

Weather Dataframe (2007 - 2014)

```
weather.head(2)
```

	Station	Date	Tmax	Tmin	Tavg	Depart	DewPoint	WetBulb	Heat	Cool	Sunrise	Sunset	CodeSum	Depth
0	1	2007-05-01	83	50	67	14	51	56	0	2	0448	1849		0
1	2	2007-05-01	84	52	68	M	51	57	0	3	-	-		M

Water1	SnowFall	PrecipTotal	StnPressure	SeaLevel	ResultSpeed	ResultDir	AvgSpeed
M	0.0	0.00	29.10	29.82	1.7	27	9.2
M	M	0.00	29.18	29.82	2.7	25	9.6

Train Data

```
train.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 10506 entries, 0 to 10505
Data columns (total 12 columns):
Date                      10506 non-null object
Address                   10506 non-null object
Species                   10506 non-null object
Block                     10506 non-null int64
Street                    10506 non-null object
Trap                      10506 non-null object
AddressNumberAndStreet    10506 non-null object
Latitude                  10506 non-null float64
Longitude                 10506 non-null float64
AddressAccuracy           10506 non-null int64
NumMosquitos              10506 non-null int64
WnvPresent                10506 non-null int64
dtypes: float64(2), int64(4), object(6)
memory usage: 985.0+ KB
```

Test Data

```
test.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 116293 entries, 0 to 116292
Data columns (total 11 columns):
Id                        116293 non-null int64
Date                      116293 non-null object
Address                   116293 non-null object
Species                   116293 non-null object
Block                     116293 non-null int64
Street                    116293 non-null object
Trap                      116293 non-null object
AddressNumberAndStreet    116293 non-null object
Latitude                  116293 non-null float64
Longitude                 116293 non-null float64
AddressAccuracy           116293 non-null int64
dtypes: float64(2), int64(3), object(6)
memory usage: 9.8+ MB
```

Spray Data

```
spray.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 14835 entries, 0 to 14834
Data columns (total 4 columns):
Date         14835 non-null object
Time         14251 non-null object
Latitude     14835 non-null float64
Longitude    14835 non-null float64
dtypes: float64(2), object(2)
memory usage: 463.7+ KB
```

Weather Data

```
weather.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2944 entries, 0 to 2943
Data columns (total 22 columns):
Station        2944 non-null int64
Date           2944 non-null object
Tmax           2944 non-null int64
Tmin           2944 non-null int64
Tavg           2944 non-null object
Depart         2944 non-null object
DewPoint       2944 non-null int64
WetBulb        2944 non-null object
Heat           2944 non-null object
Cool           2944 non-null object
Sunrise        2944 non-null object
Sunset         2944 non-null object
CodeSum        2944 non-null object
Depth          2944 non-null object
Water1         2944 non-null object
SnowFall       2944 non-null object
PrecipTotal    2944 non-null object
StnPressure    2944 non-null object
SeaLevel       2944 non-null object
ResultSpeed    2944 non-null float64
ResultDir      2944 non-null int64
AvgSpeed       2944 non-null object
dtypes: float64(1), int64(5), object(16)
memory usage: 506.1+ KB
```




Cleaning

Weather Data

Replace missing value by Station 1's value
1. Depart Temp
2. Wet Bulb Temp
3. Sunrise & Sunset
4. Average Speed

Sunrise and Sunset: The time is same for the whole Chicago

```
#Use ffill-fill with next row (station 1's value)
#change value with 'M' with NaN
weather['Sunrise'] = weather['Sunrise'].replace('-',np.nan)
weather['Sunrise'].ffill(axis='rows',inplace=True)

weather['Sunset'] = weather['Sunset'].replace('-',np.nan)
weather['Sunset'].ffill(axis='rows',inplace=True)
```

Depart temp: It is difference in temp for the day against for the past 30 years

```
#Use ffill-fill with previous row (station 1's value)
#change value with 'M' with NaN
weather['Depart'] = weather['Depart'].replace('M',np.nan)
weather['Depart'].ffill(axis='rows',inplace=True)
```

Average Speed: We have 3 missing data in AvgSpeed. Going through the data, the difference between the 2 station's AvgSpeed minimal, at a range of 0 to 2. As such we will fill the missing Average speed with the average speed of the corresponding station for the same day.

```
weather['AvgSpeed'] = weather['AvgSpeed'].replace('M', np.nan)
weather['AvgSpeed'].ffill(axis='rows', inplace=True)
```

Wet Bulb temp: Lowest temp that can be reached by evaporating water into air.

```
#fill with the other station's value
for i,index in enumerate(weather[weather['WetBulb']=='M'].index):
    print(i,index)
    if weather.loc[index,'Station'] == 1:
        weather.loc[index,'WetBulb'] = weather.loc[index+1,'WetBulb']
    else:
        weather.loc[index,'WetBulb'] = weather.loc[index-1,'WetBulb']
```

Cleaning

Weather Data

As our data is collected from the period of time when Chicago do not experience any snowfall, we will fill up the missing data with 0 and convert them to numeric

No Snowfall
1. Depth
2. Water1
3. Snowfall

```
weather['Depth'] = weather['Depth'].replace('M', 0)
weather['Depth']=pd.to_numeric(weather['Depth'])
```

```
weather['Water1'] = weather['Water1'].replace('M', 0)
weather['Water1']=pd.to_numeric(weather['Water1'])
```

```
weather['SnowFall'] = weather['SnowFall'].replace('M', 0)
weather['SnowFall'] = weather['SnowFall'].replace('   T', 0)
weather['SnowFall'] = pd.to_numeric(weather['SnowFall'])
```

No Outliers and so replaced with its mean value

Mean
1. Station Pressure
2. Sea Level

```
# Replace missing values with null values
weather['StnPressure'] = weather['StnPressure'].replace(to_replace='M', value=np.nan)
```

```
#fill it up with mean value
weather['StnPressure'] = weather['StnPressure'].fillna(round(weather['StnPressure'].astype(float).mean(),2))
weather['StnPressure']=pd.to_numeric(weather['StnPressure'])
```

Cleaning : 

Code Sum

Code Sum indicates the weather phenomena for the day. Indicated in Kaggle's data description, if CodeSum is blank, it means there are no signs of any special weather phenomena. We will replace the blanks with 'No Sign'

```
#Replace empty values as 'No Sign'
weather['CodeSum'] = weather['CodeSum'].replace(to_replace = ' ', value = 'No Sign')
```

Spray Data

We have 584 null values in time.

	Date	Time	Latitude	Longitude
1029	2011-09-07	7:44:32 PM	41.986460	-87.794225
1614	2011-09-07	7:46:30 PM	41.973465	-87.827643

The entry before and after our null values are also of the same date. The time before the start of our null values is 7:44:32 PM and the time after our null values is 7:46:30 PM. Looking at other rows of data, we can see that time for the same date are in running order when going down the rows, as such we will fill the null values with 7:45:00 PM. Then convert the time to 24 hour format.

Cleaning

Train Data

Train DataSet has no null values so nothing needs to be done for this step.
There is a total of 10506 observations and 12 columns.

Drop
1. Street
2. Block
3. AddressNumber&Street

Dropping Street, Block, Address Number and Street Columns

```
# Drop Columns for Train Dataset
train.drop(columns = ['Street','Block','AddressNumberAndStreet'], inplace=True)
train.head()
```

Test Data

Test DataSet has no null values There is a total of 116293 observations and 11 columns. Test Dataset does not contain NumMosquitos and WnvPresent columns.
But has an additional Id column.

Drop
1. Street
2. Block
3. AddressNumber&Street

Dropping Street, Block, Address Number and Street Columns

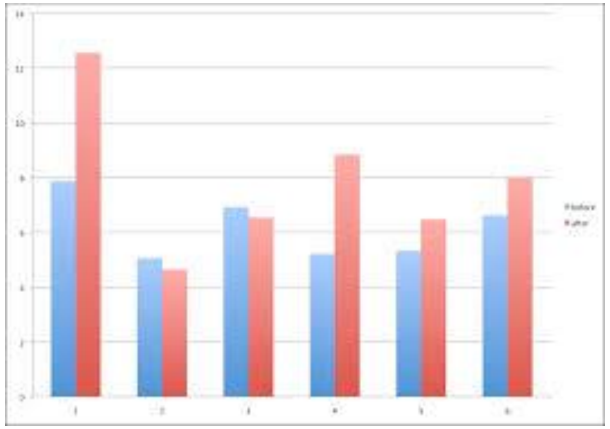
```
# Drop Columns for Test Dataset
test.drop(columns = ['Street','Block','AddressNumberAndStreet'], inplace=True)
test.head()
```

EDA

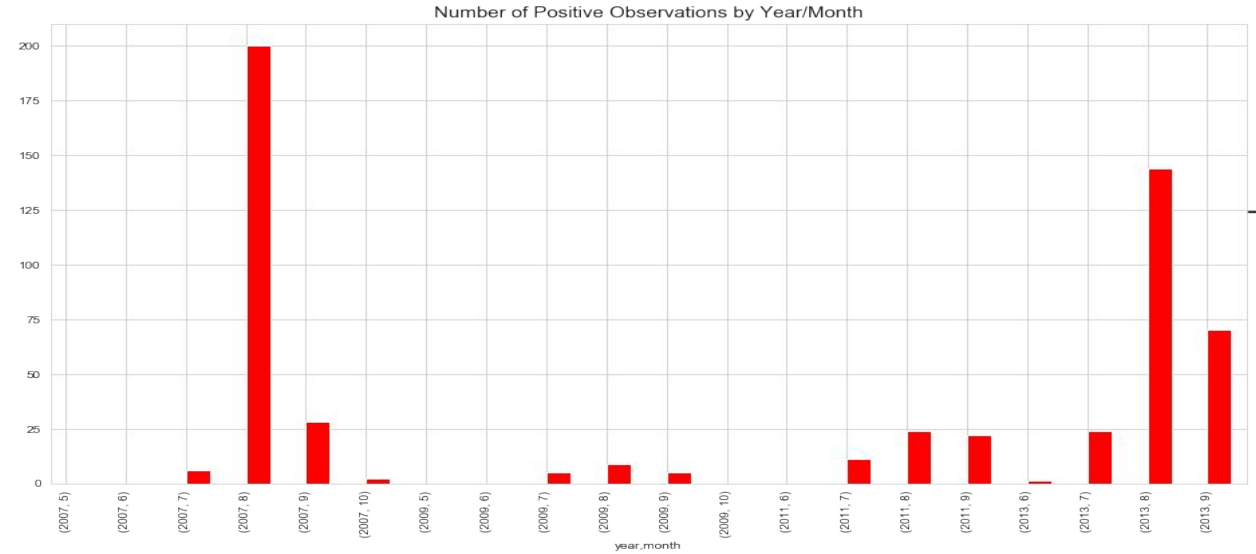
EDA: WnvPresent Against Features

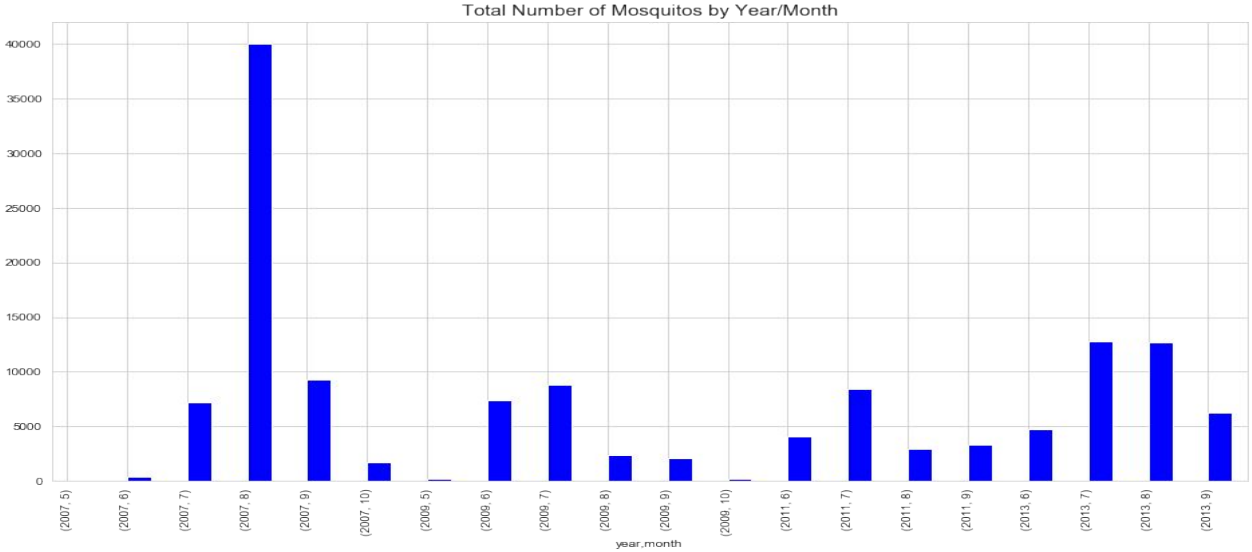
- Trends

- Relationships



EDA: Trends



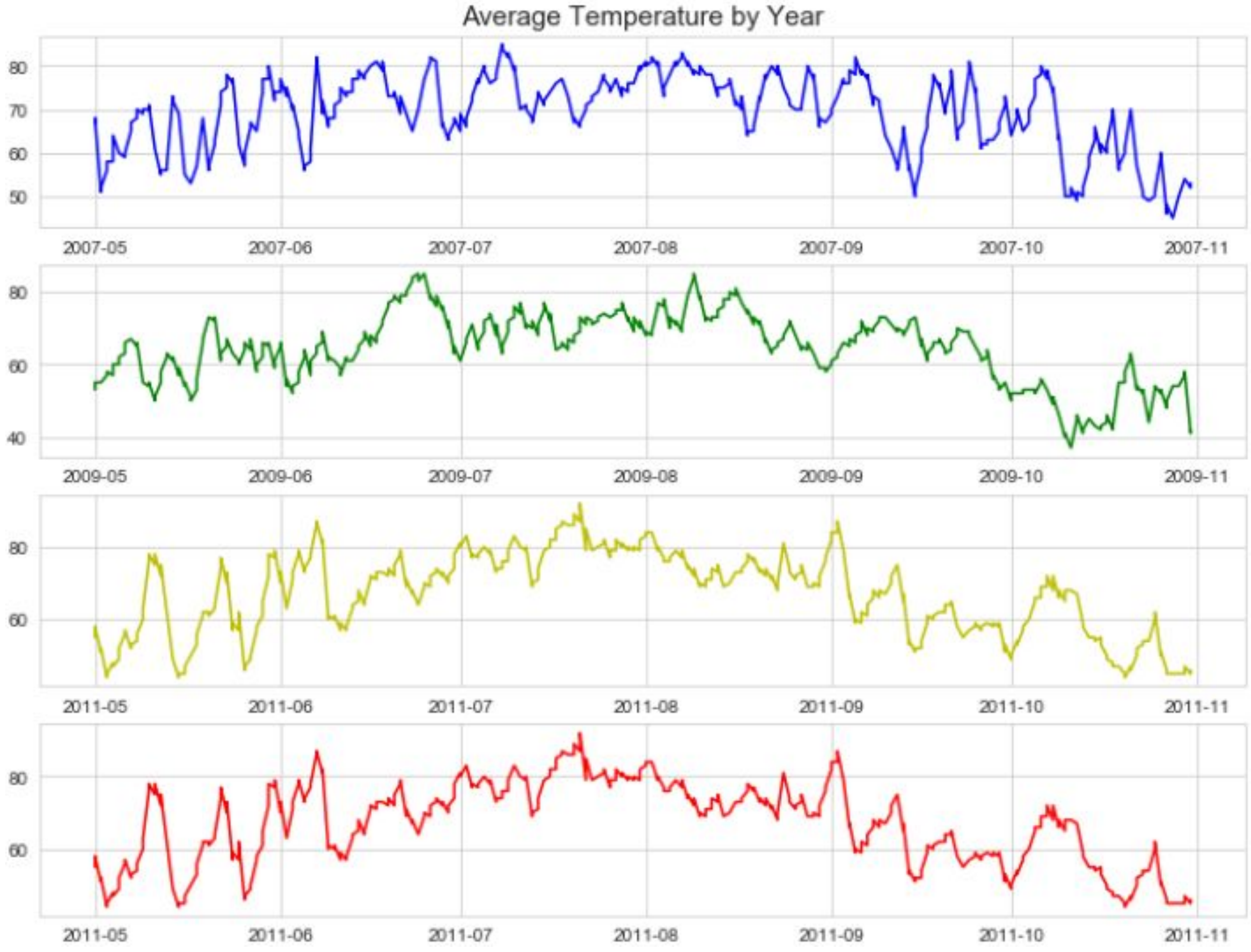


Red = WnvPresent

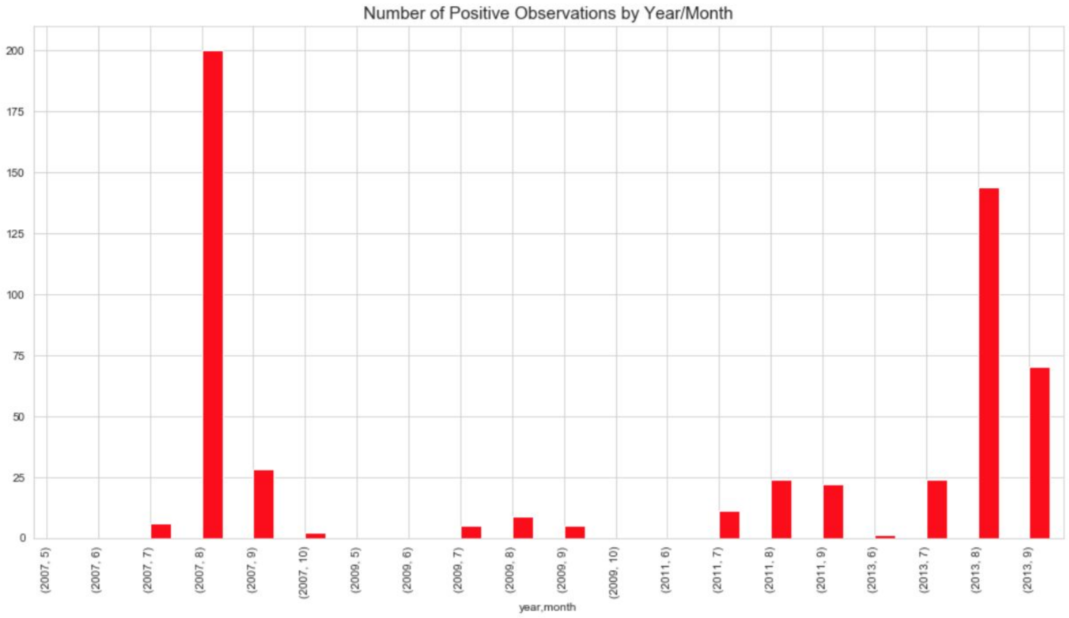
Blue = Number of Mosquitoes

We will be using the observations plot throughout this section

EDA: Trends

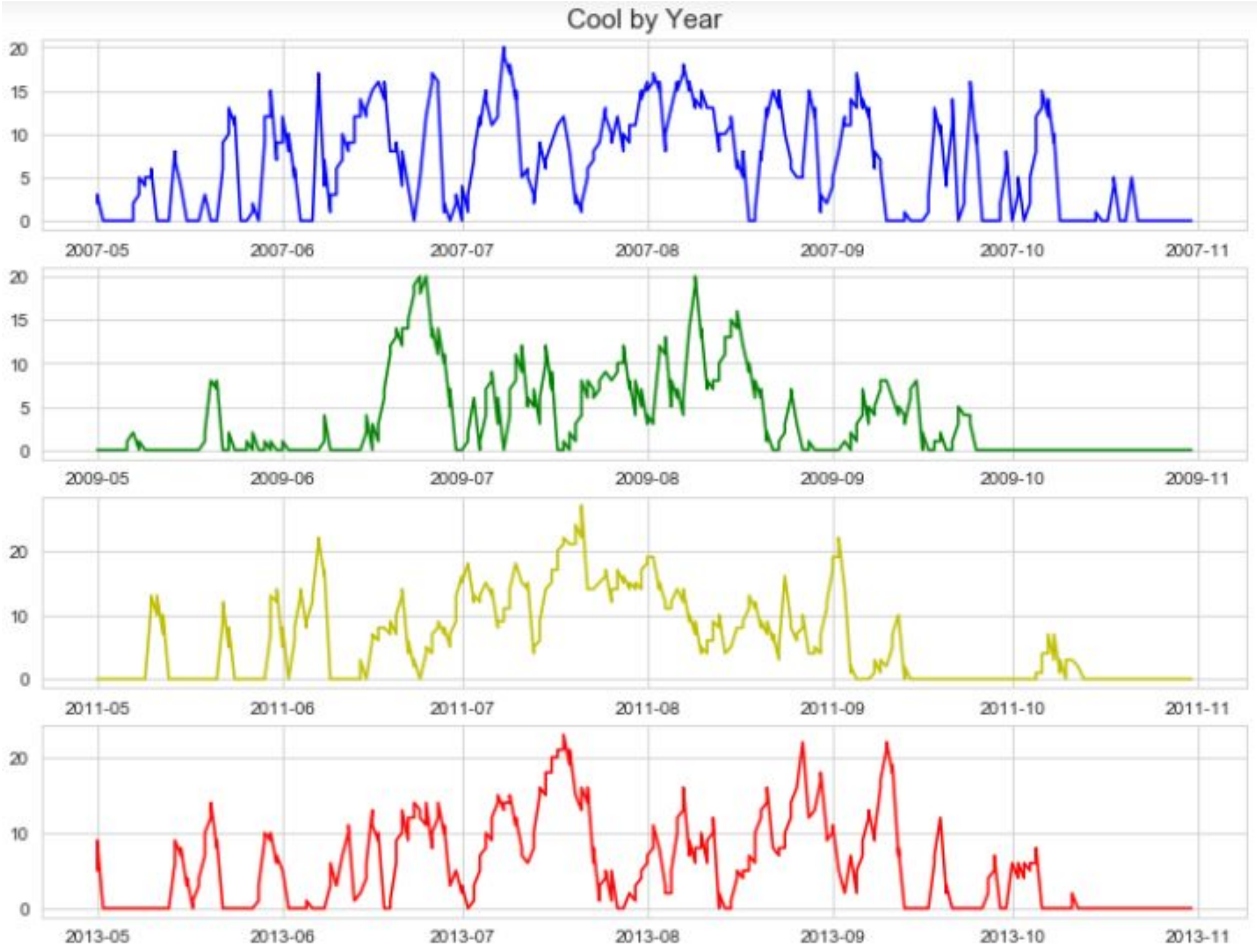


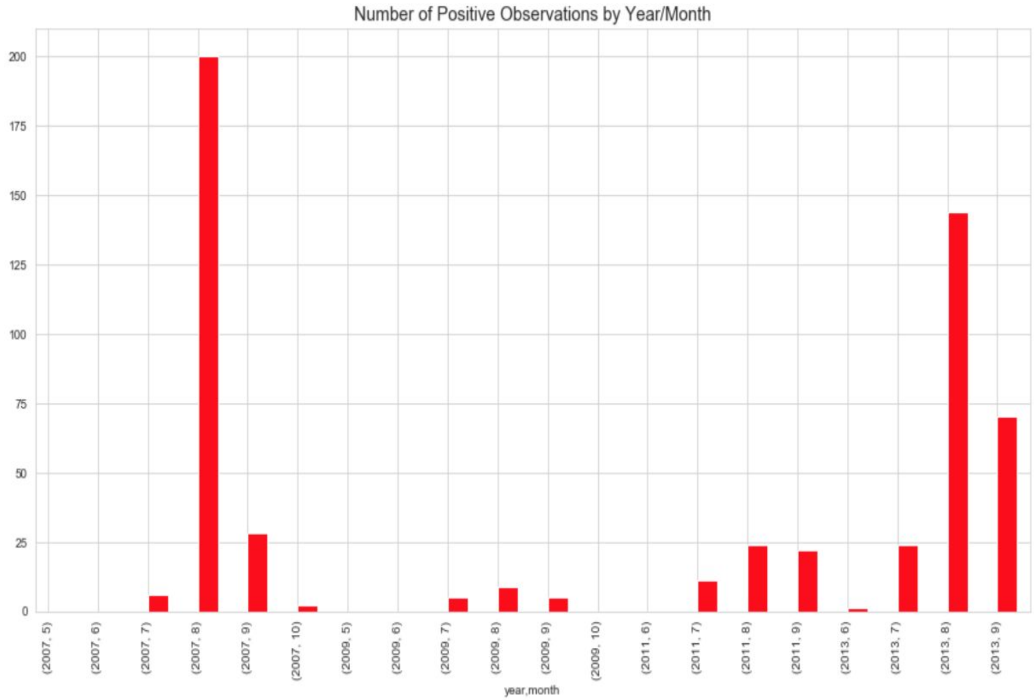
Average Temperature by Year
80
70
60
50
2007-05
2007-06
2007-07
2007-08
2007-09
2007-10
2007-11
80
60
40
2009-05
2009-06
2009-07
2009-08
2009-09
2009-10
2009-11
80
60
2011-05
2011-06
2011-07
2011-08
2011-09
2011-10
2011-11
80
60
2011-05
2011-06
2011-07
2011-08
2011-09
2011-10
2011-11



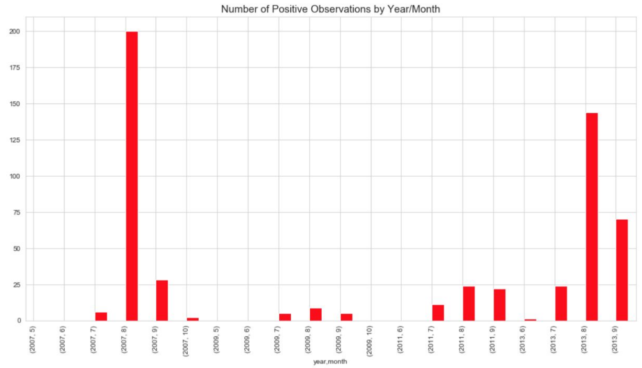
Number of Positive Observations by Year/Month
200
175
150
125
100
75
50
25
0
(2007, 5)
(2007, 6)
(2007, 7)
(2007, 8)
(2007, 9)
(2007, 10)
(2009, 5)
(2009, 6)
(2009, 7)
(2009, 8)
(2009, 9)
(2009, 10)
(2011, 6)
(2011, 7)
(2011, 8)
(2011, 9)
(2013, 6)
(2013, 7)
(2013, 8)
(2013, 9)
year,month

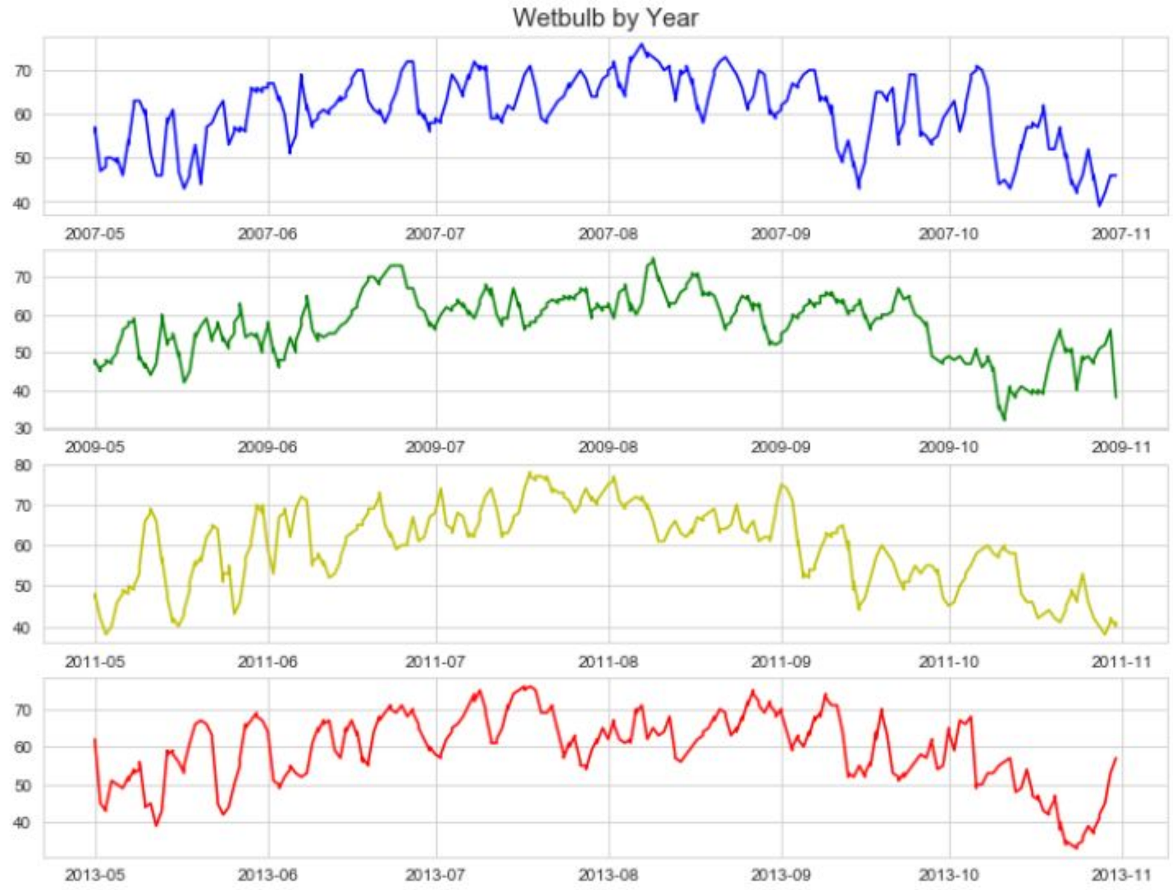
EDA: Trends

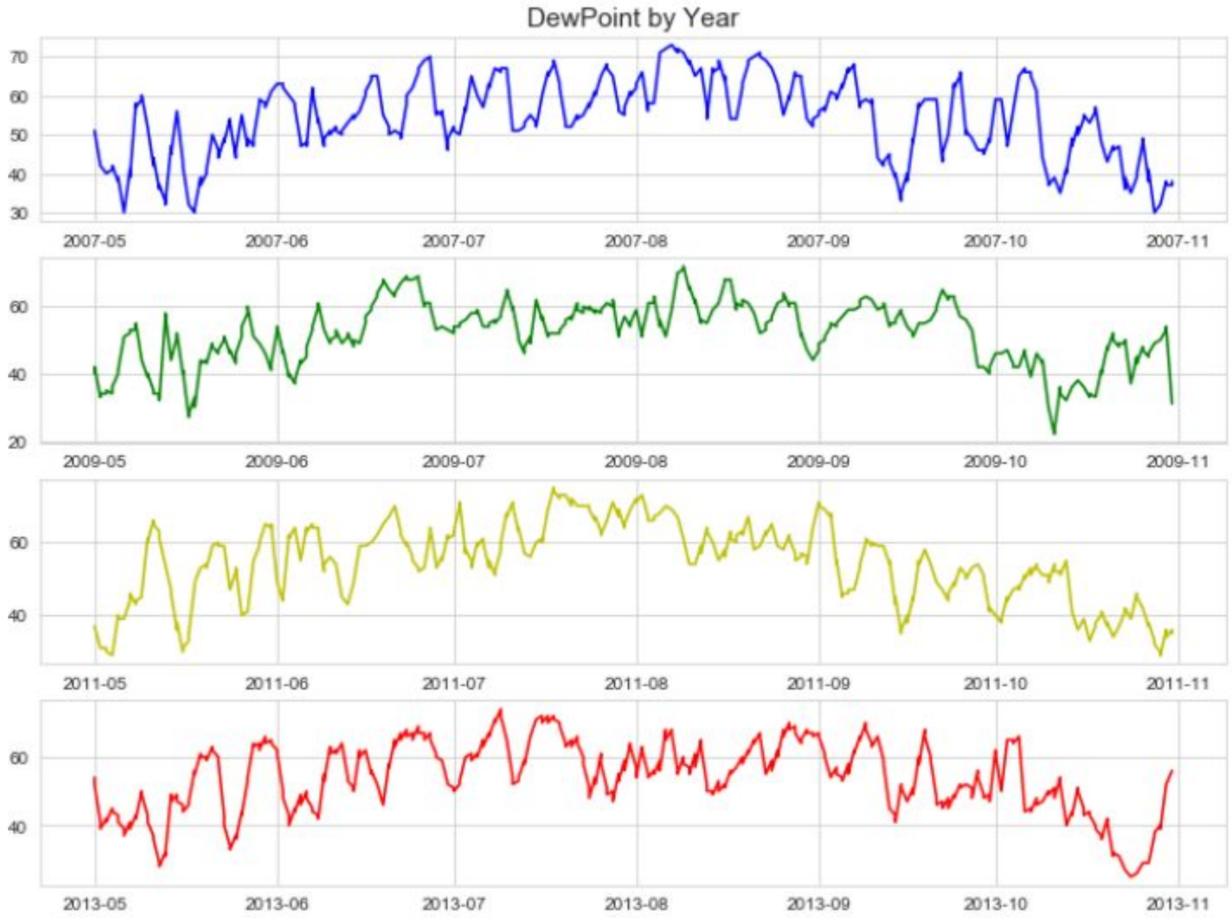

Cool by Year
20
15
10
5
0
2007-05
2007-06
2007-07
2007-08
2007-09
2007-10
2007-11
2009-05
2009-06
2009-07
2009-08
2009-09
2009-10
2009-11
2011-05
2011-06
2011-07
2011-08
2011-09
2011-10
2011-11
2013-05
2013-06
2013-07
2013-08
2013-09
2013-10
2013-11


Number of Positive Observations by Year/Month
200
175
150
125
100
75
50
25
0
(2007, 5)
(2007, 6)
(2007, 7)
(2007, 8)
(2007, 9)
(2007, 10)
(2009, 5)
(2009, 6)
(2009, 7)
(2009, 8)
(2009, 9)
(2009, 10)
(2011, 6)
(2011, 7)
(2011, 8)
(2011, 9)
(2013, 6)
(2013, 7)
(2013, 8)
(2013, 9)
year,month

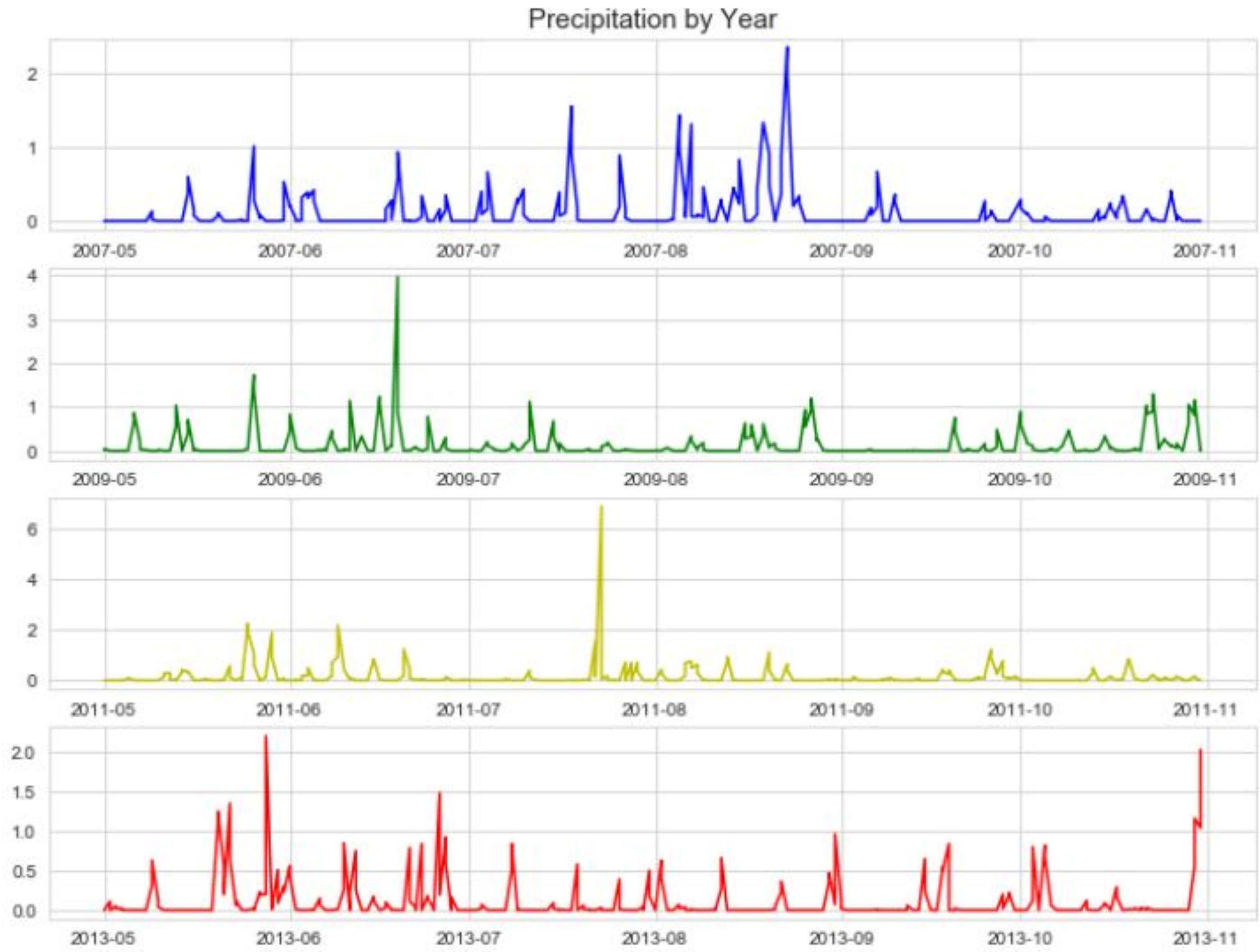
EDA: Trends


Number of Positive Observations by Year/Month
year,month

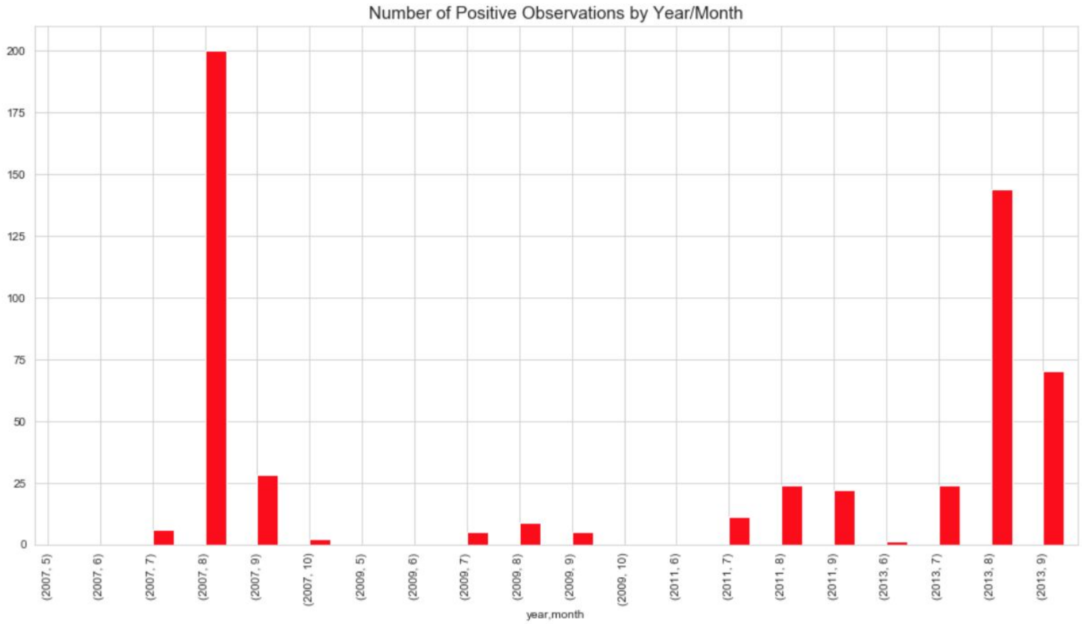

Wetbulb by Year
2007-05
2007-06
2007-07
2007-08
2007-09
2007-10
2007-11
2009-05
2009-06
2009-07
2009-08
2009-09
2009-10
2009-11
2011-05
2011-06
2011-07
2011-08
2011-09
2011-10
2011-11
2013-05
2013-06
2013-07
2013-08
2013-09
2013-10
2013-11


DewPoint by Year
2007-05
2007-06
2007-07
2007-08
2007-09
2007-10
2007-11
2009-05
2009-06
2009-07
2009-08
2009-09
2009-10
2009-11
2011-05
2011-06
2011-07
2011-08
2011-09
2011-10
2011-11
2013-05
2013-06
2013-07
2013-08
2013-09
2013-10
2013-11

EDA: Trends

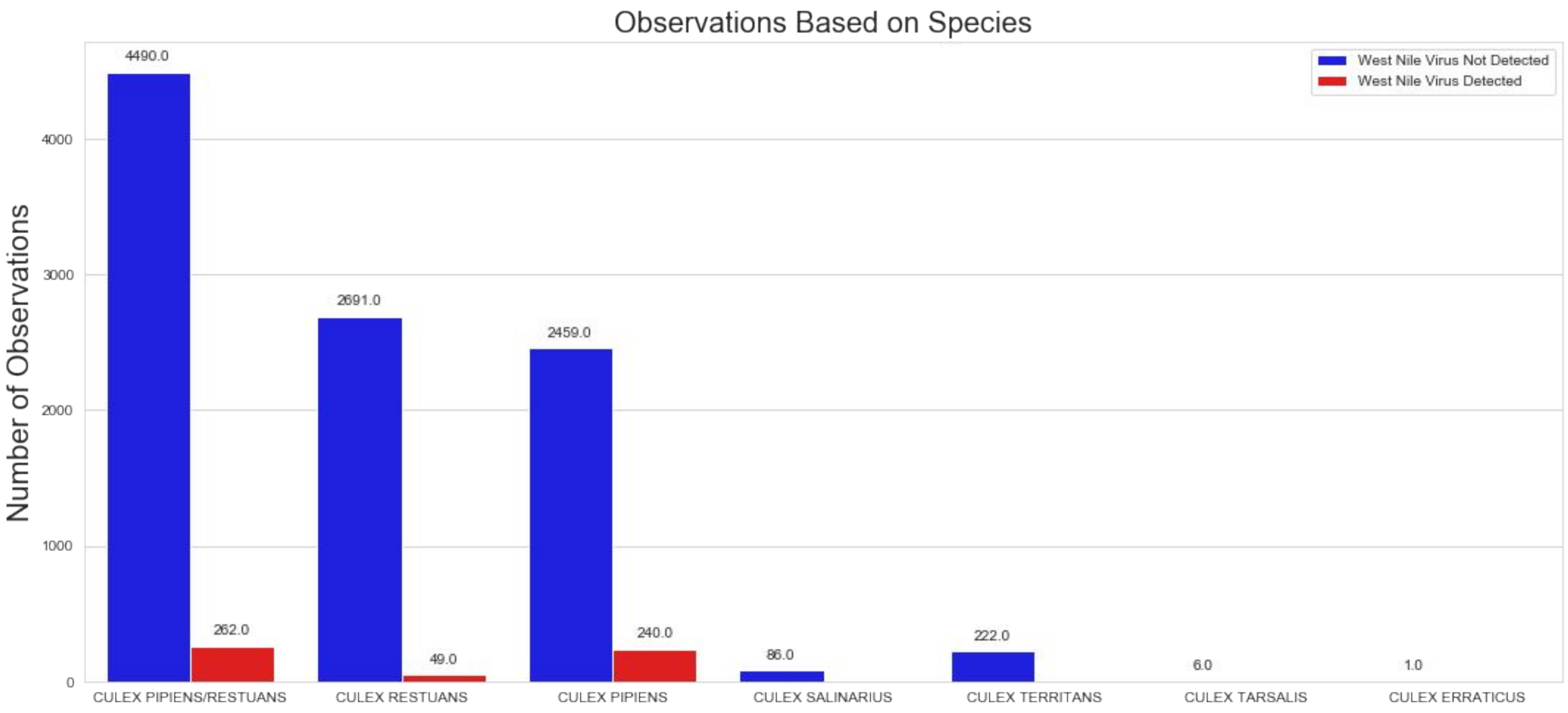


Precipitation by Year
2
1
0
2007-05
2007-06
2007-07
2007-08
2007-09
2007-10
2007-11
4
3
2
1
0
2009-05
2009-06
2009-07
2009-08
2009-09
2009-10
2009-11
6
4
2
0
2011-05
2011-06
2011-07
2011-08
2011-09
2011-10
2011-11
2.0
1.5
1.0
0.5
0.0
2013-05
2013-06
2013-07
2013-08
2013-09
2013-10
2013-11

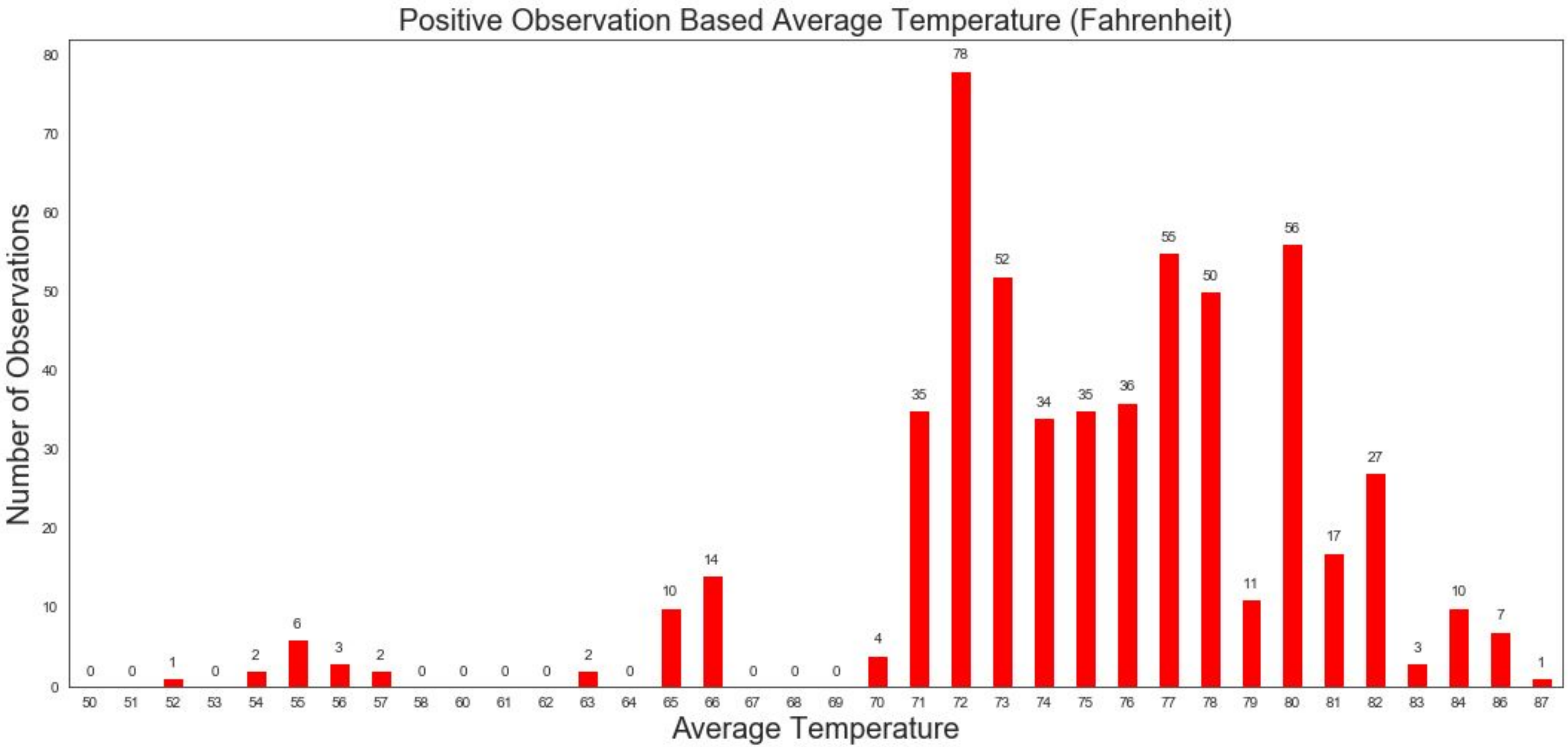


Number of Positive Observations by Year/Month
200
175
150
125
100
75
50
25
0
(2007, 5)
(2007, 6)
(2007, 7)
(2007, 8)
(2007, 9)
(2007, 10)
(2009, 5)
(2009, 6)
(2009, 7)
(2009, 8)
(2009, 9)
(2009, 10)
(2011, 6)
(2011, 7)
(2011, 8)
(2011, 9)
(2013, 6)
(2013, 7)
(2013, 8)
(2013, 9)
year,month

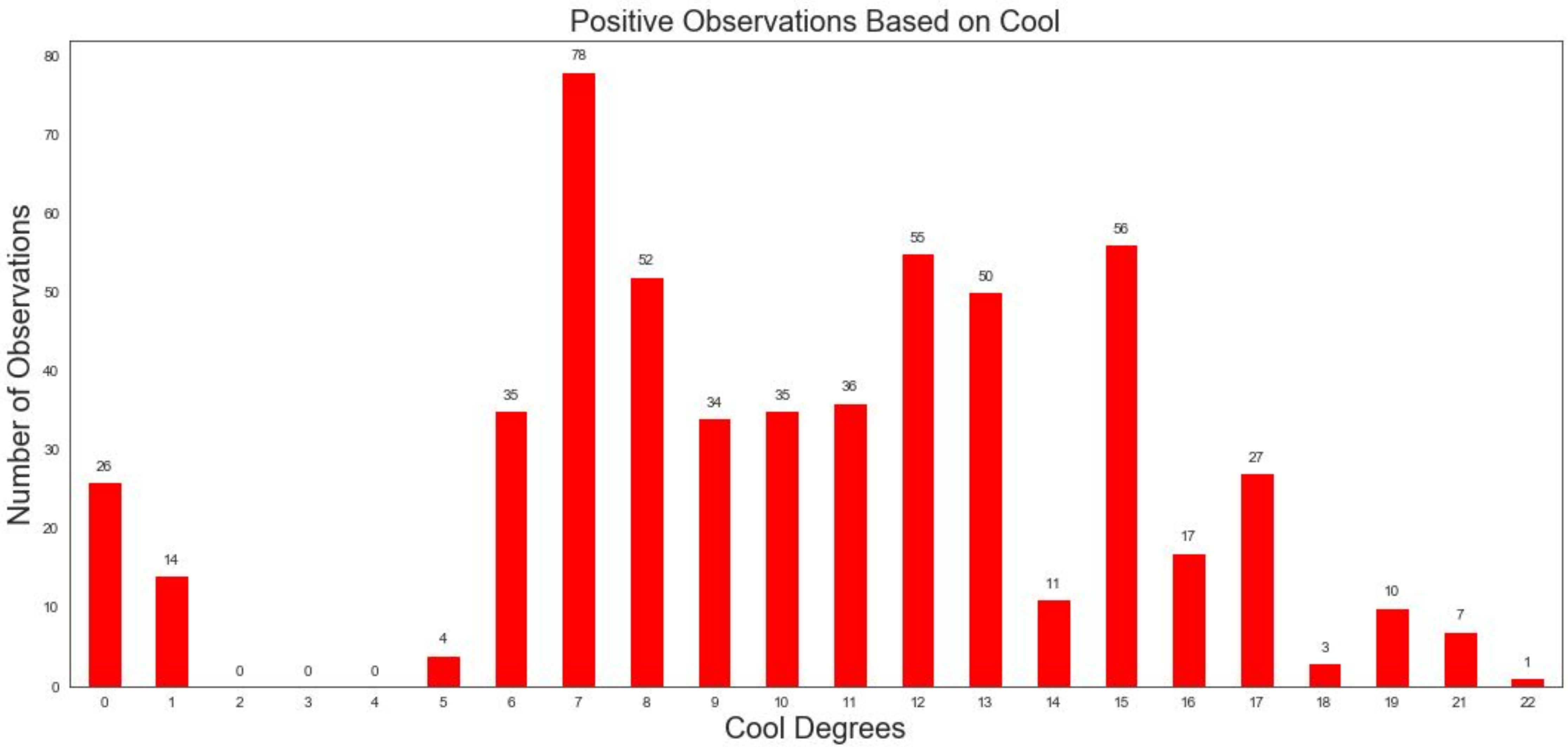
EDA: Relationship



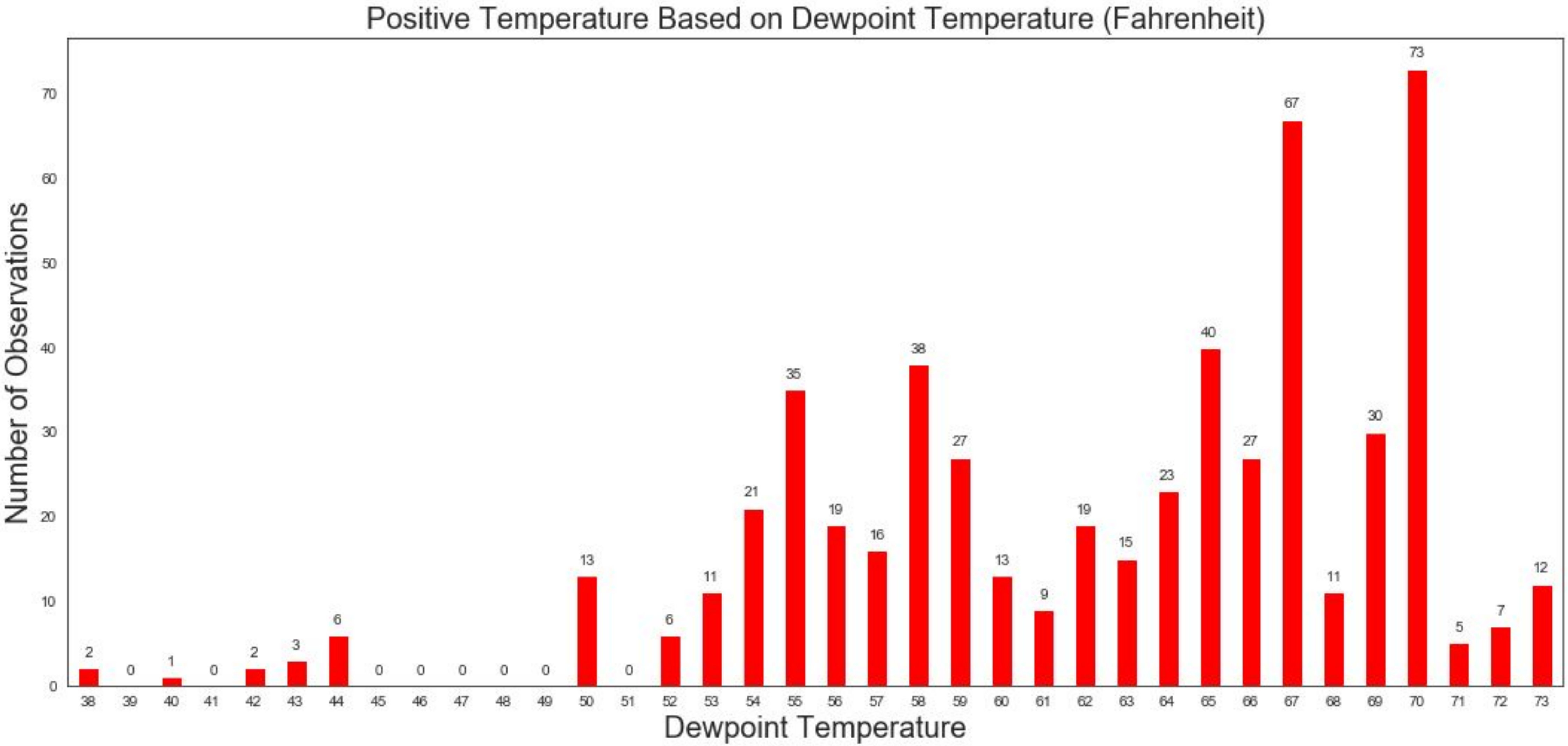
EDA: Relationship



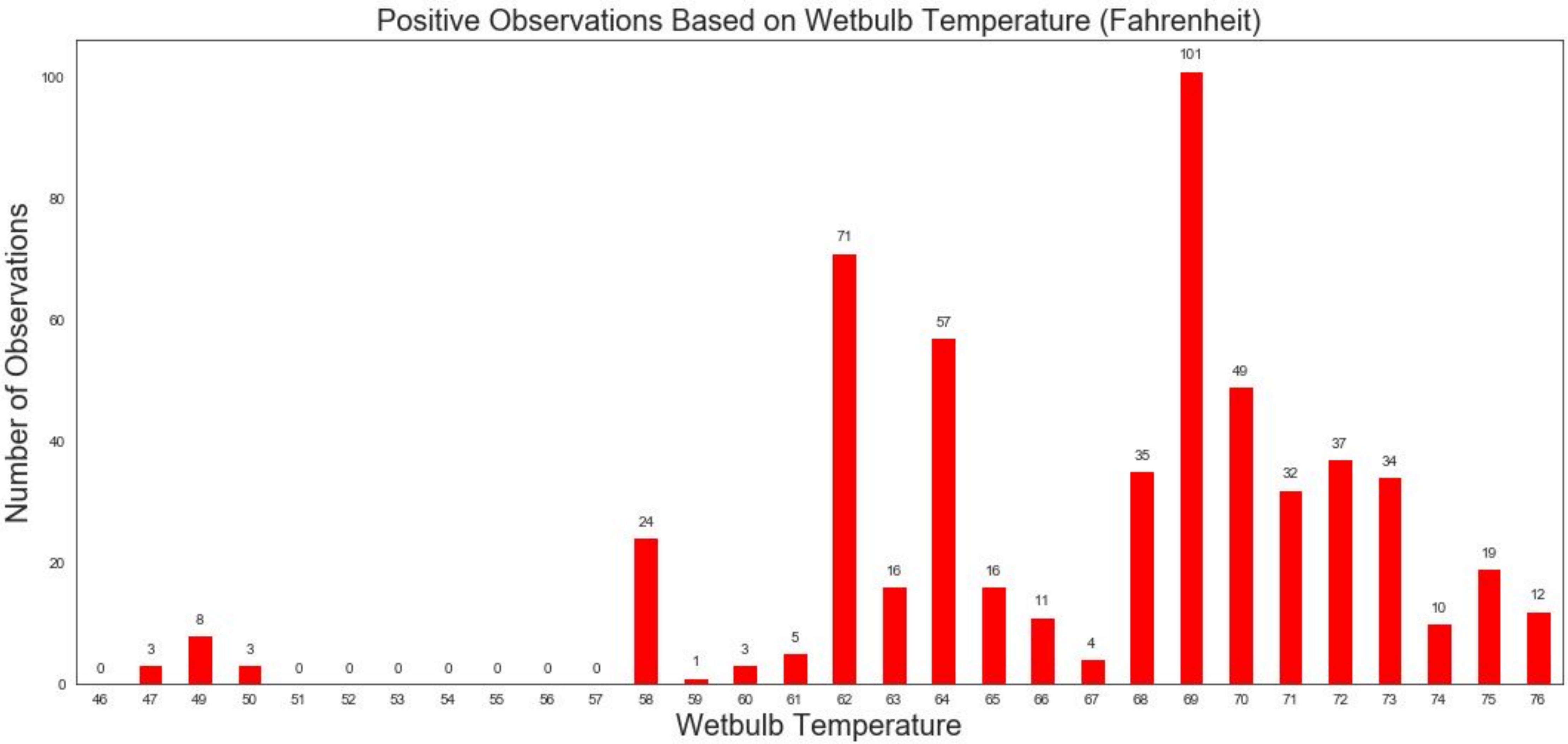
EDA: Relationship



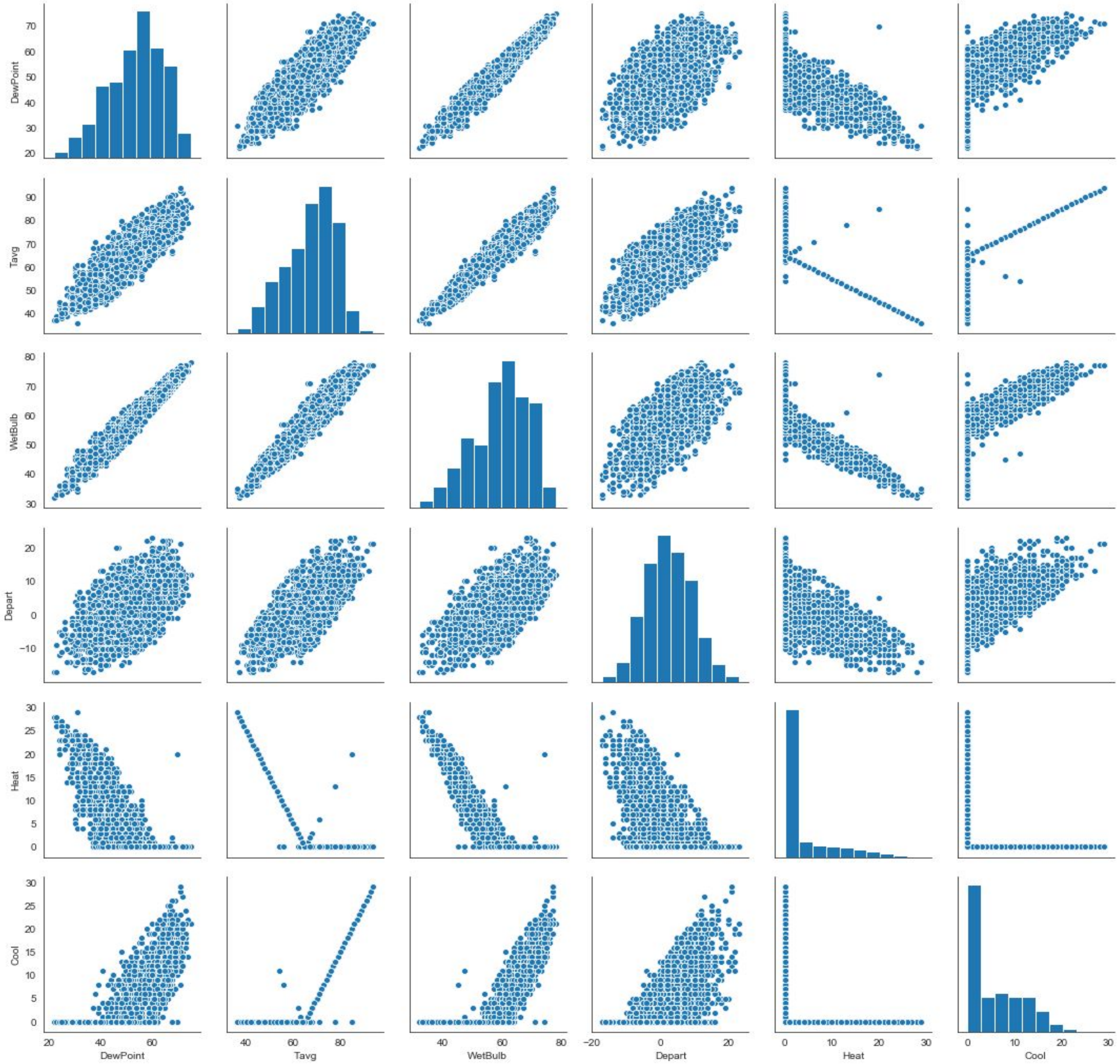
EDA: Relationship



EDA: Relationship



EDA: Relationship

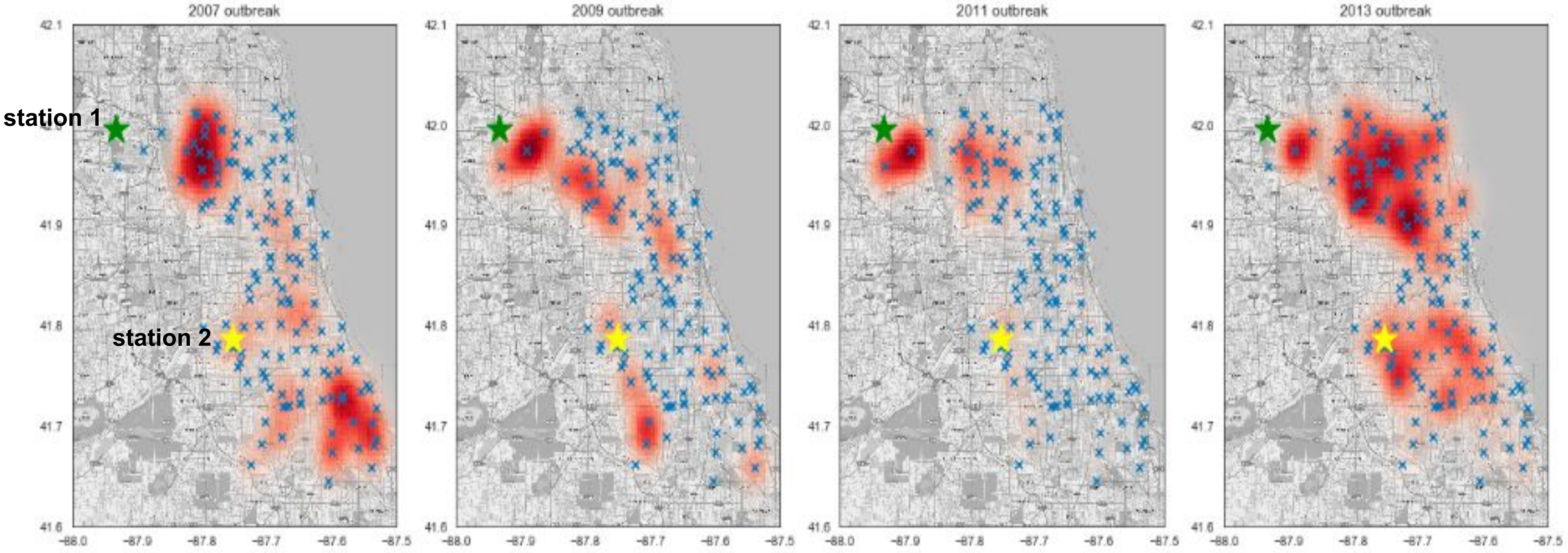


- WetBulb & DewPoint
  - Both are measurements about humidity

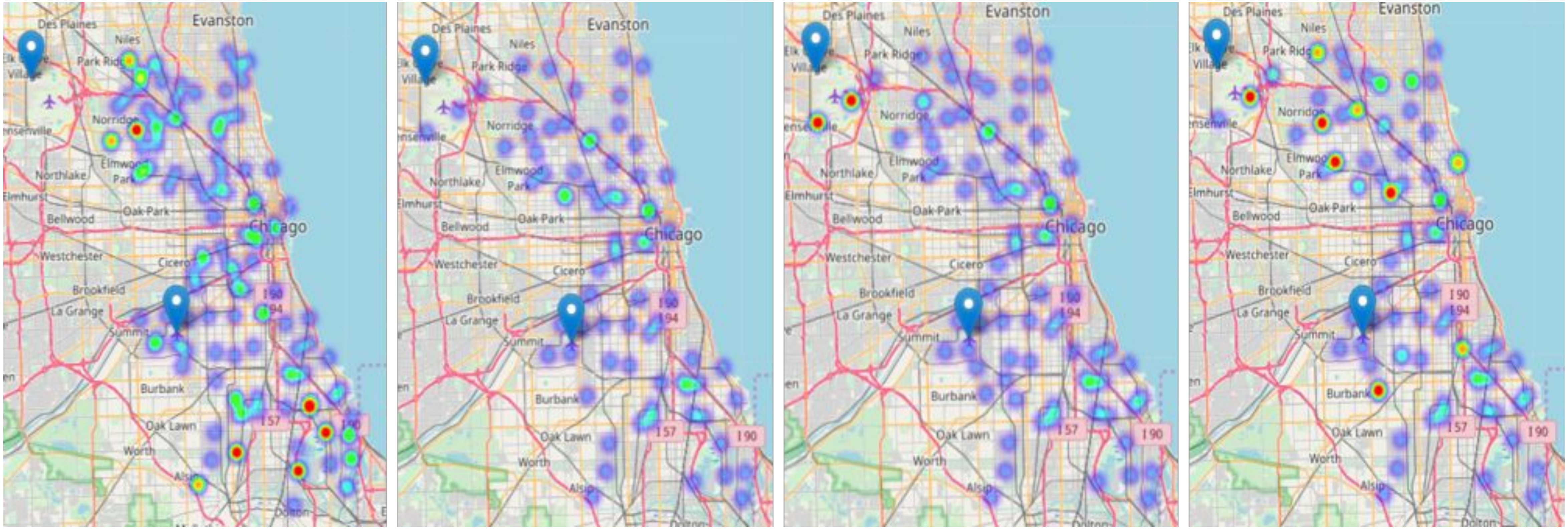
- Tavg & WetBulb
  - Increase in temperature, increase in humidity

Trap Analysis / Feature Engineering

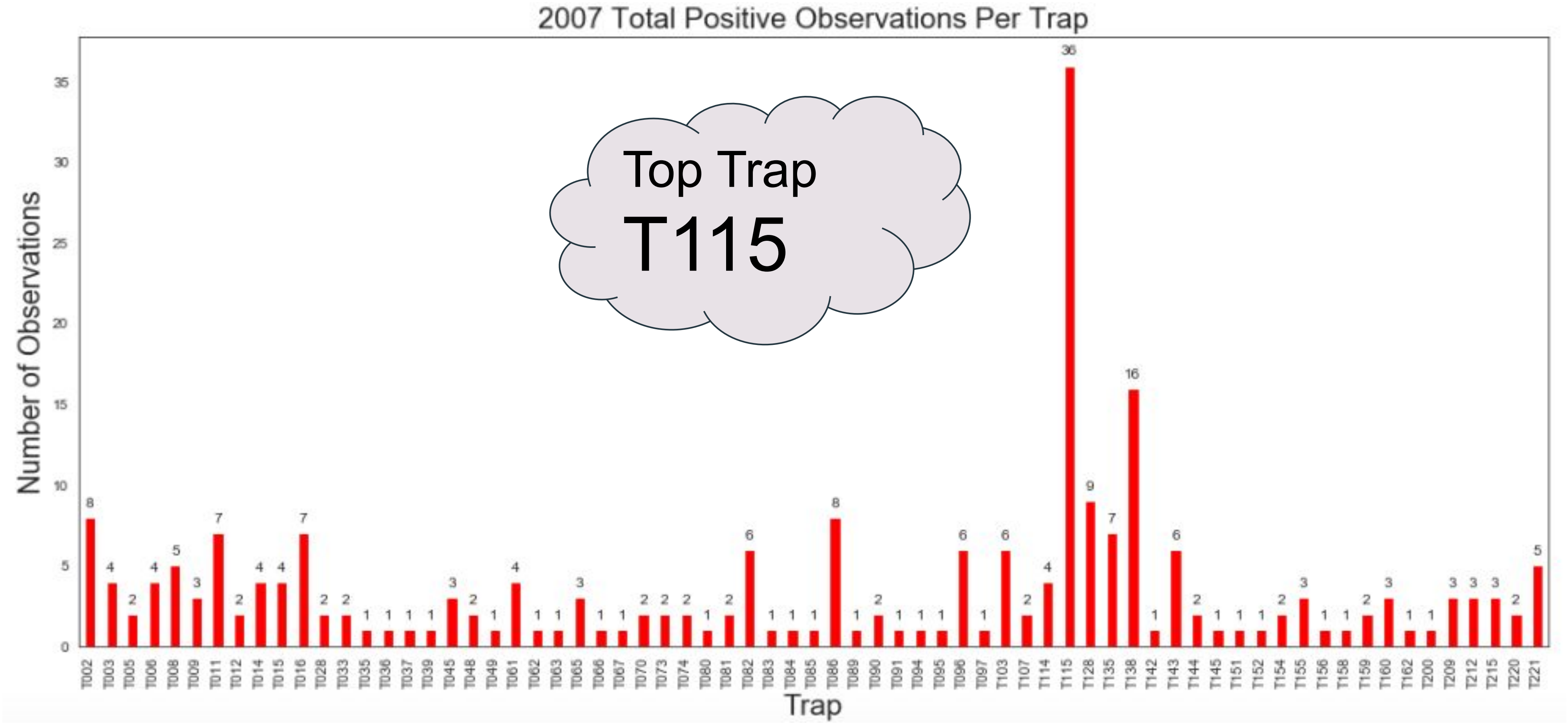
Heatmap of outbreak(2007,2009,2011,2013)



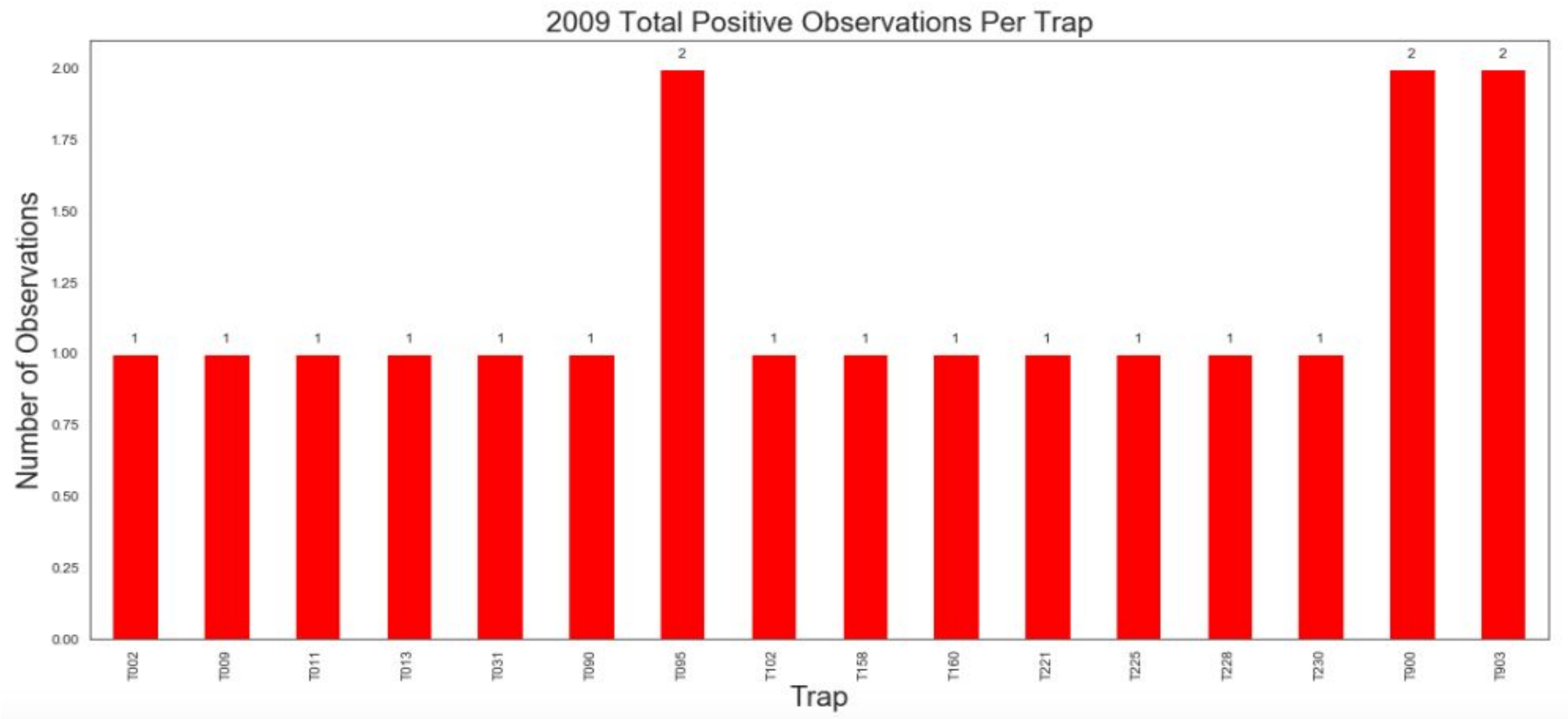
Heatmap of outbreak(2007,2009,2011,2013)



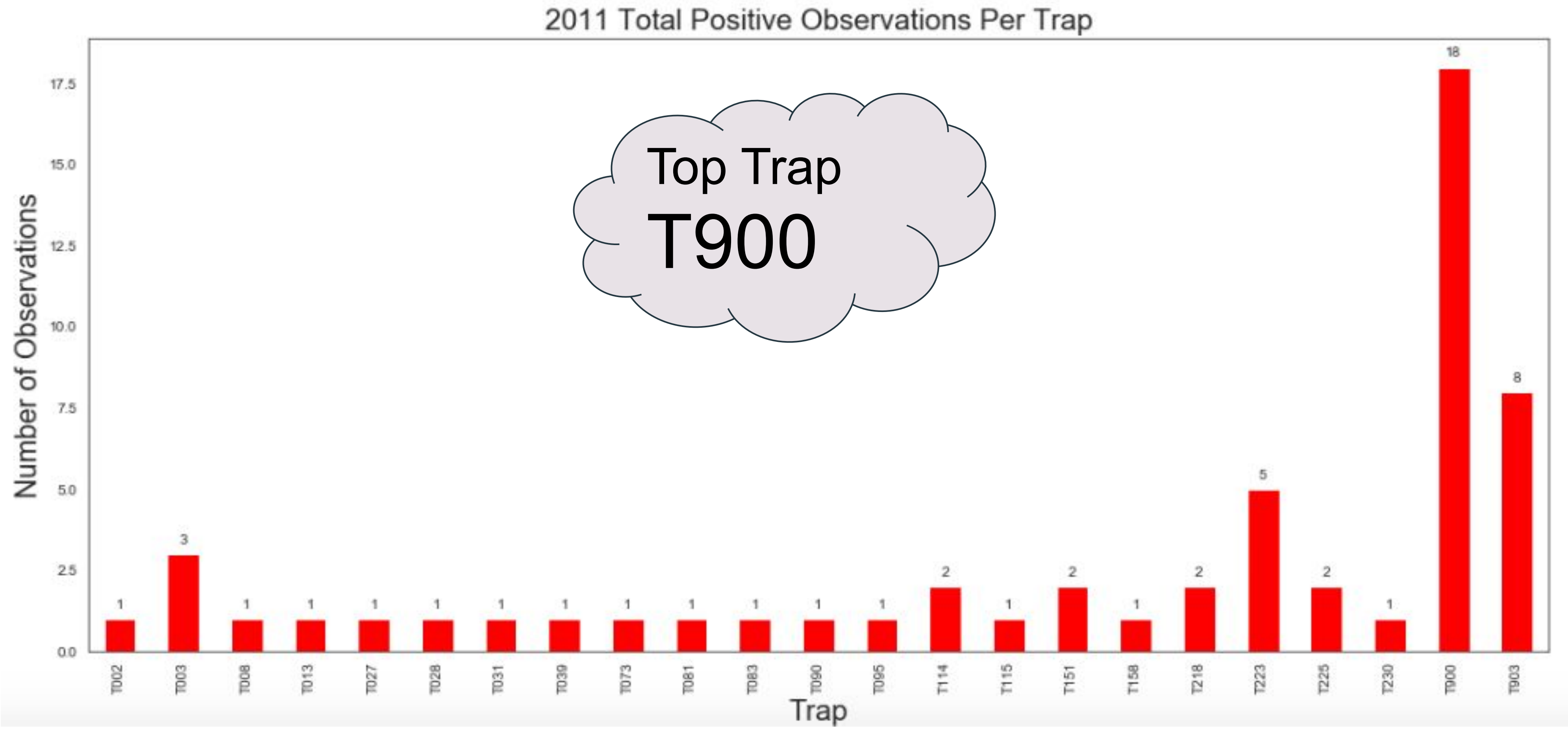
Trap Infection Count (2007)



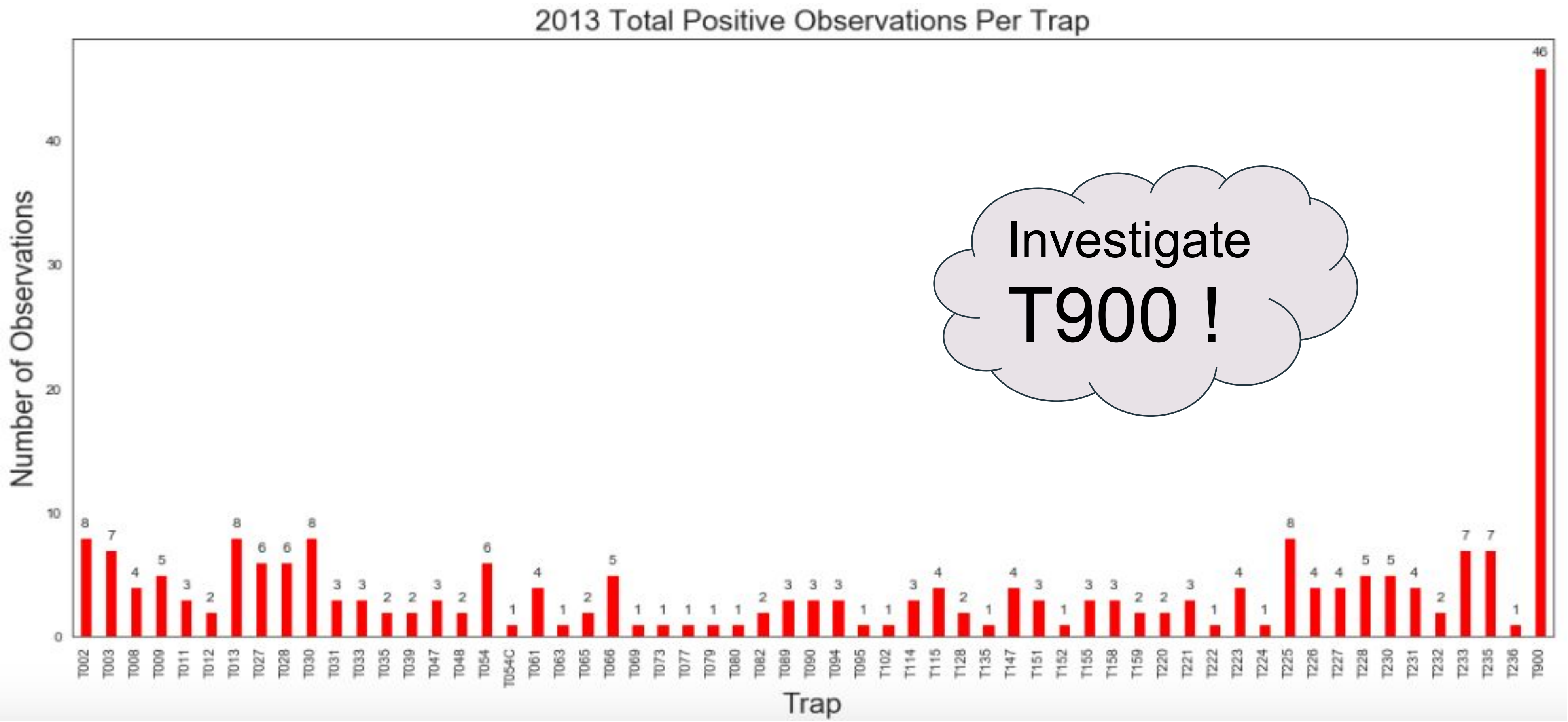
Trap Infection Count (2009)



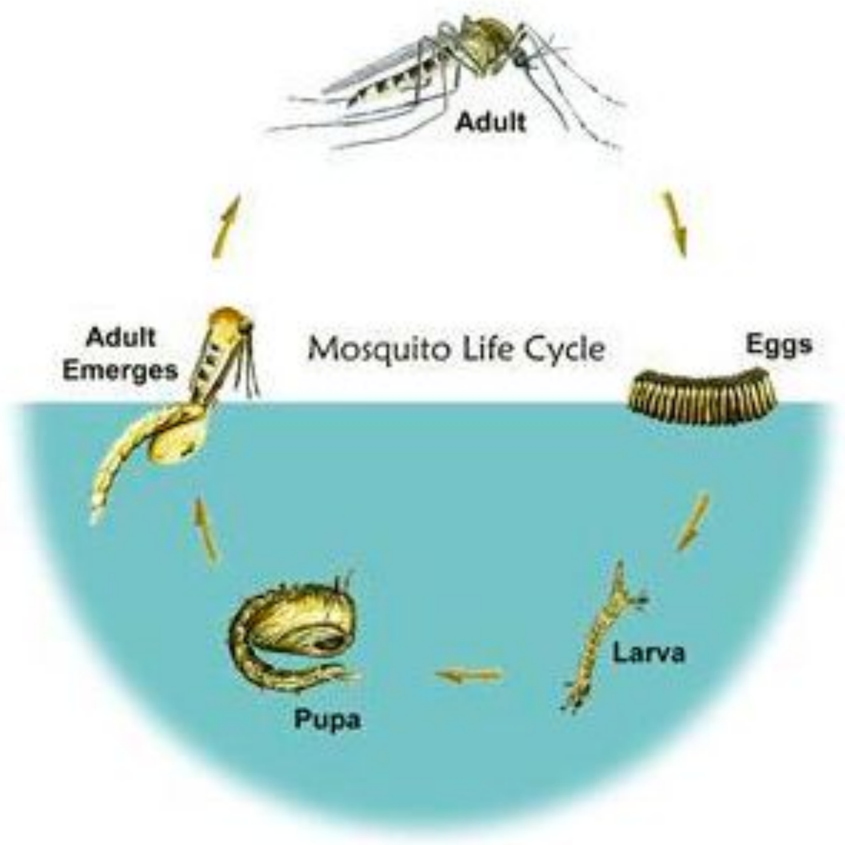
Trap Infection Count (2011)



Trap Infection Count (2013)



Investigate Trap T900



Where does the mosquitos come from ?

Life Cycle of Mosquitos

- Lay eggs near **water**
- eggs -> Larva -> Pupa, in **water** (2-3 days)
- full cycle to adult ~ 5-8 days

- Investigate weather columns link to humidity!
- Investigate temperature!

Trap T900 (prior 7 days weather readings)

		2011-06-09	2011-06-08	2011-06-07	2011-06-06	2011-06-05	2011-06-04	2011-06-03
		PrecipTotal_1dayb	**PrecipTotal_2dayb**	**PrecipTotal_3dayb**	**PrecipTotal_4dayb**	**PrecipTotal_5dayb**	**PrecipTotal_6dayb**	**PrecipTotal_7dayb**
Date	**Station**							
2011-06-10	1	0.93	0.17	0.0	0.0	0.00	0.18	0.01
	2	2.17	0.69	0.0	0.0	0.01	0.50	0.18

DewPoint_1dayb	DewPoint_2dayb	DewPoint_3dayb	DewPoint_4dayb	DewPoint_5dayb	DewPoint_6dayb	DewPoint_7dayb
52.0	64.0	65.0	64.0	55.0	64.0	62.0
54.0	63.0	64.0	63.0	55.0	64.0	61.0

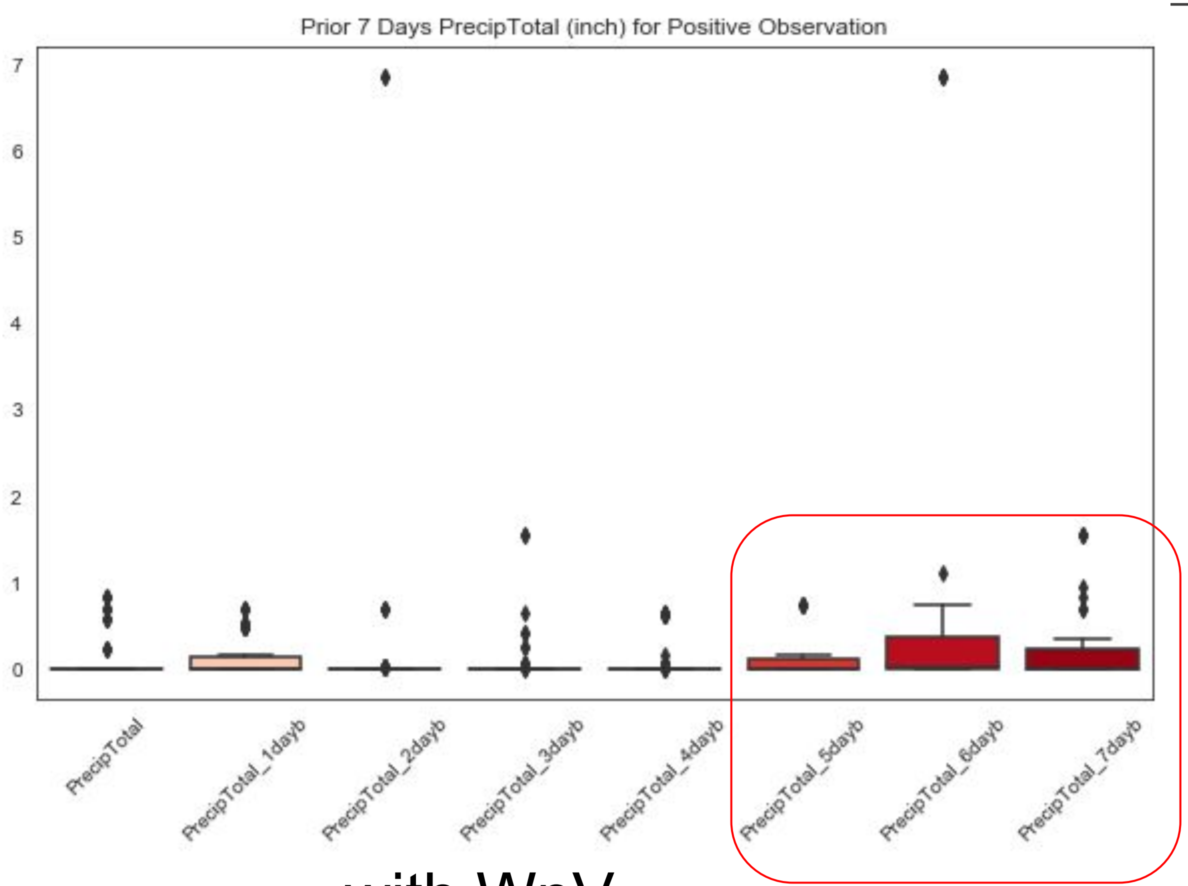
WetBulb_1dayb	WetBulb_2dayb	WetBulb_3dayb	WetBulb_4dayb	WetBulb_5dayb	WetBulb_6dayb	WetBulb_7dayb
55.0	71.0	72.0	69.0	62.0	68.0	67.0
56.0	71.0	72.0	69.0	62.0	69.0	67.0

Tavg_1dayb	Tavg_2dayb	Tavg_3dayb	Tavg_4dayb	Tavg_5dayb	Tavg_6dayb	Tavg_7dayb
60.0	81.0	86.0	77.0	73.0	78.0	71.0
61.0	82.0	87.0	78.0	74.0	79.0	73.0

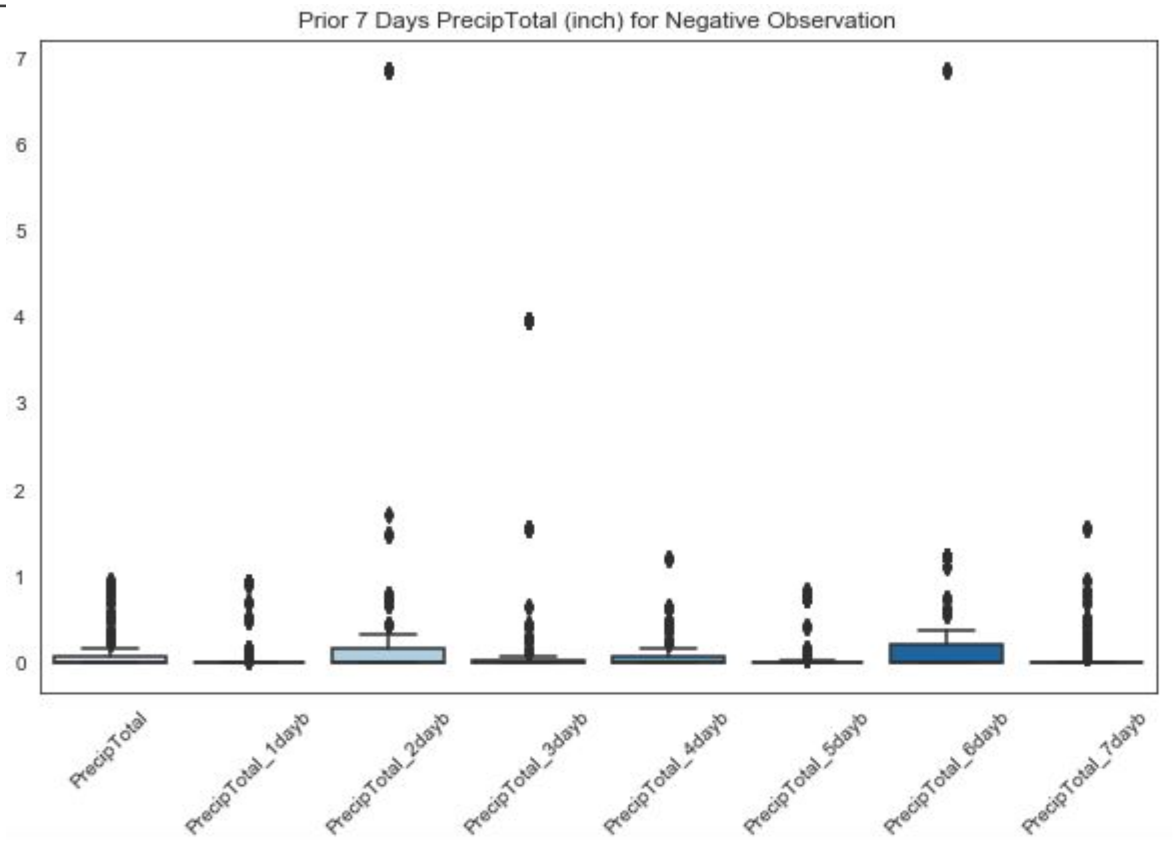
<u>Columns</u>
PrecipTotal
WetBulb
DewPoint
Tavg

4 x 7 = 28 new features created

Trap T900 (PrecipTotal Column)

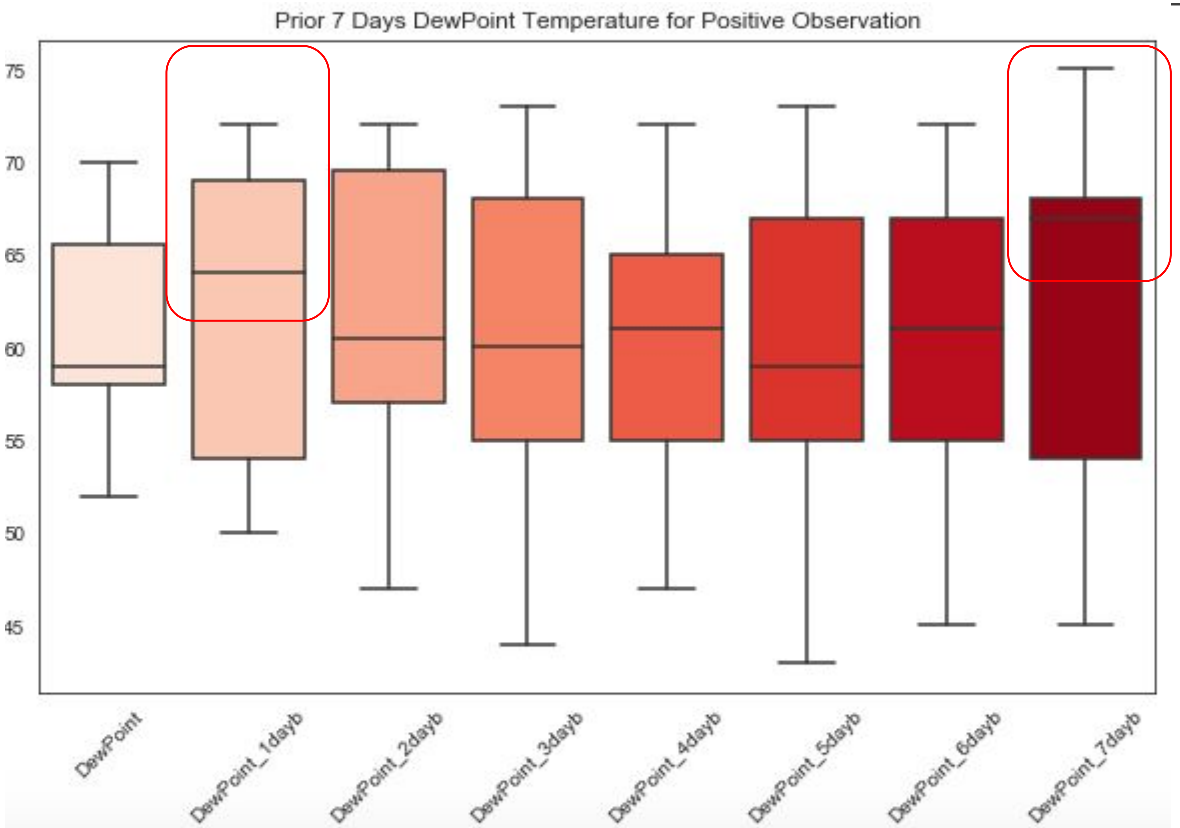


with WnV

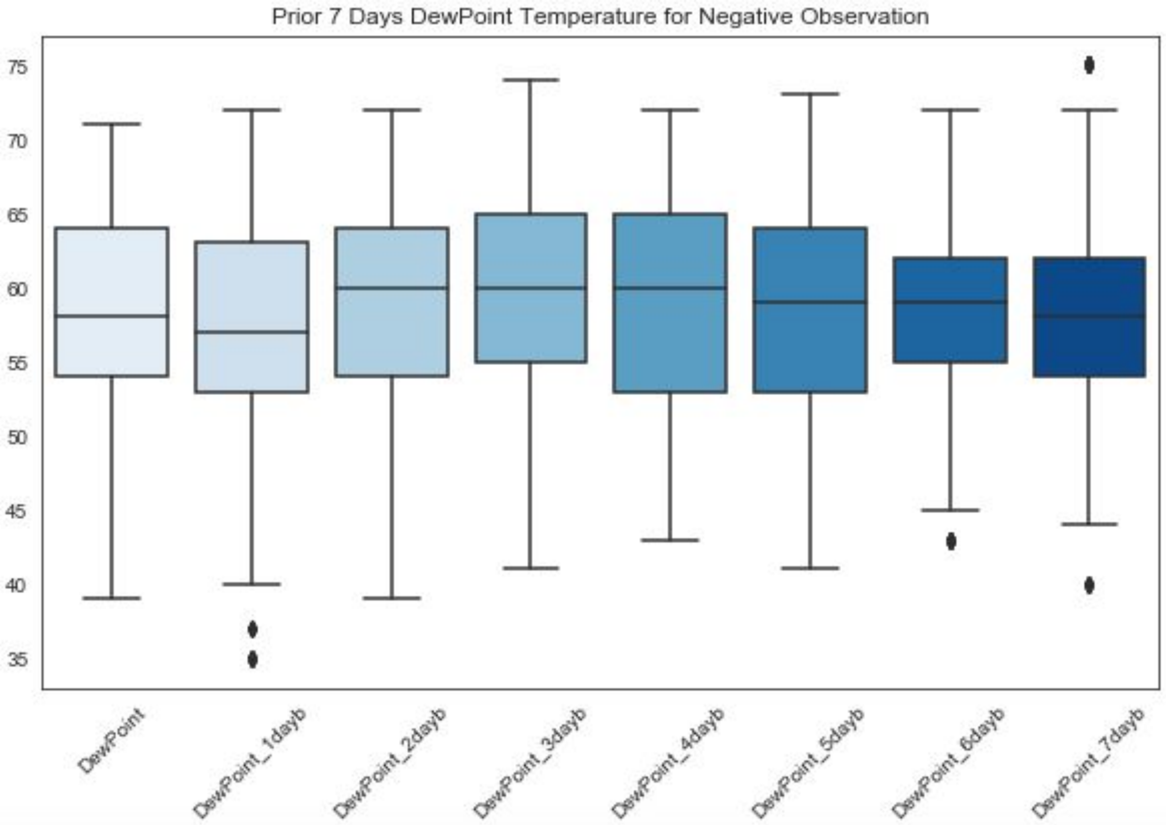


without WnV

Trap T900 (DewPoint Column)

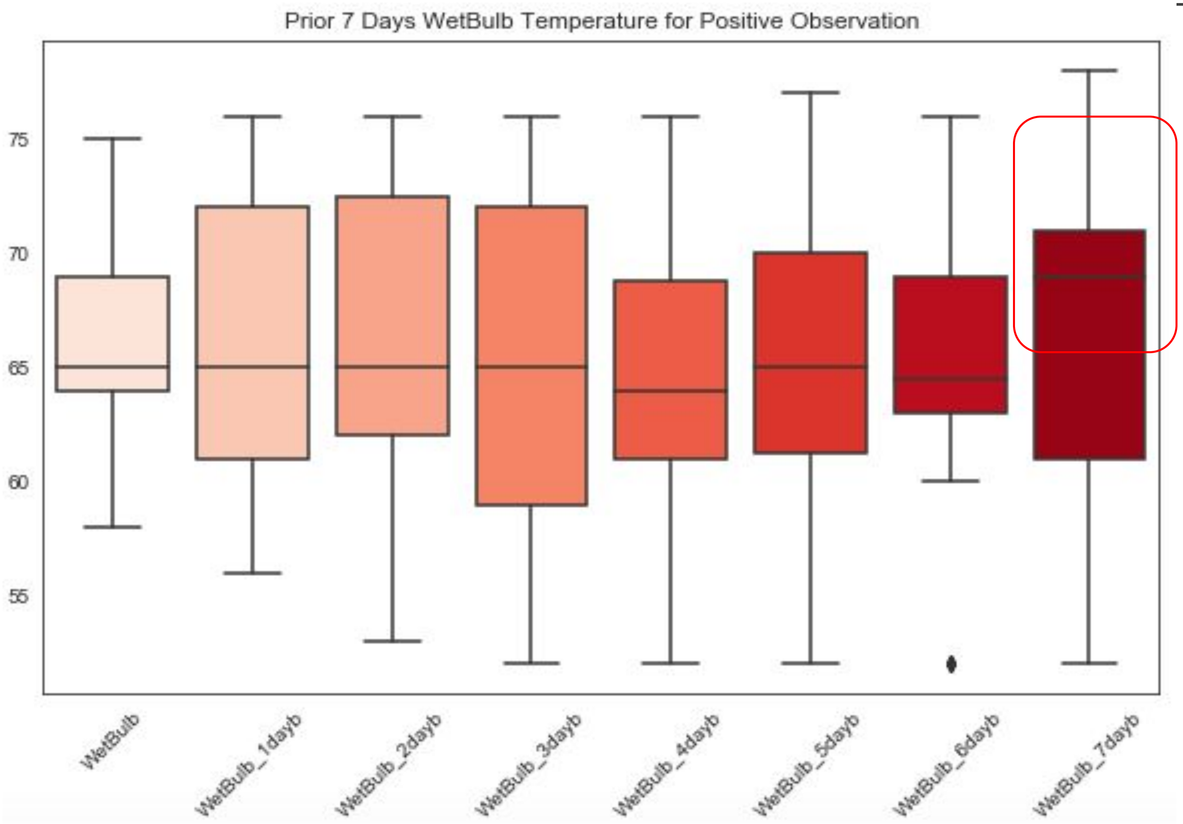


with WnV

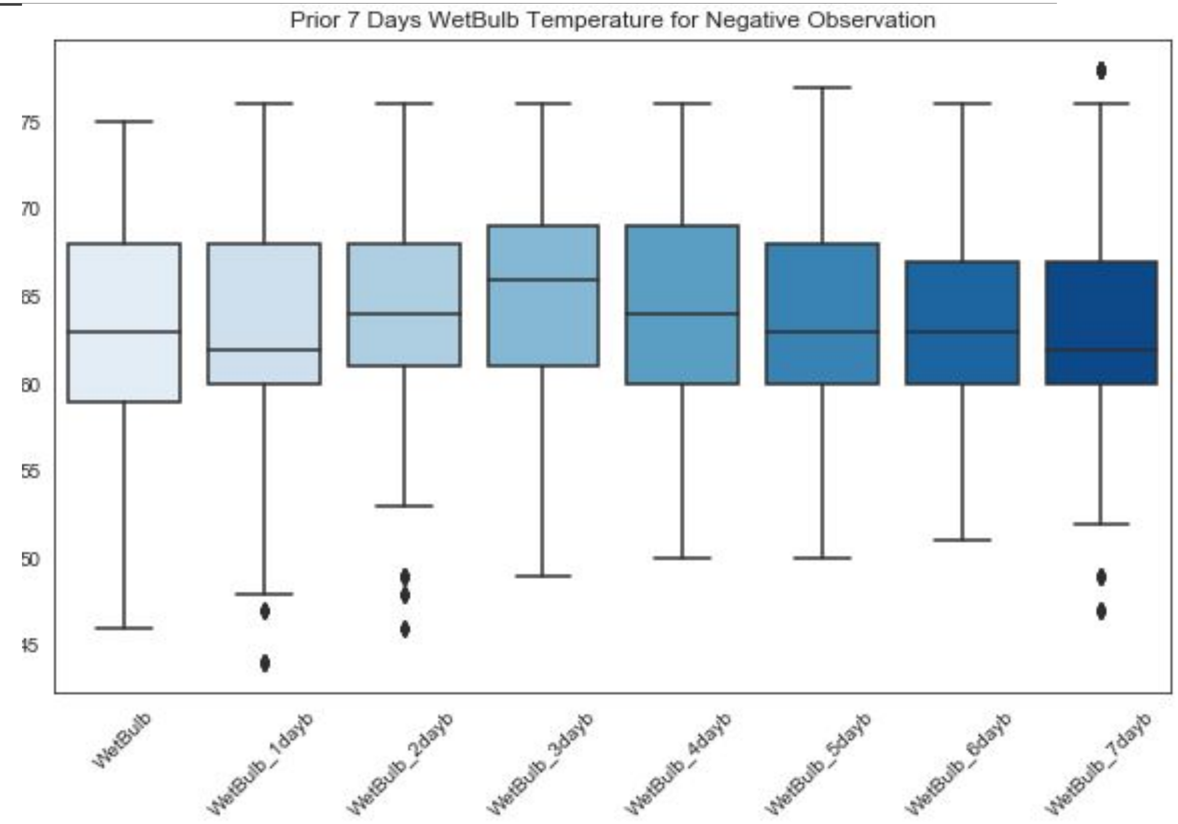


without WnV

Trap T900 (WetBulb Column)

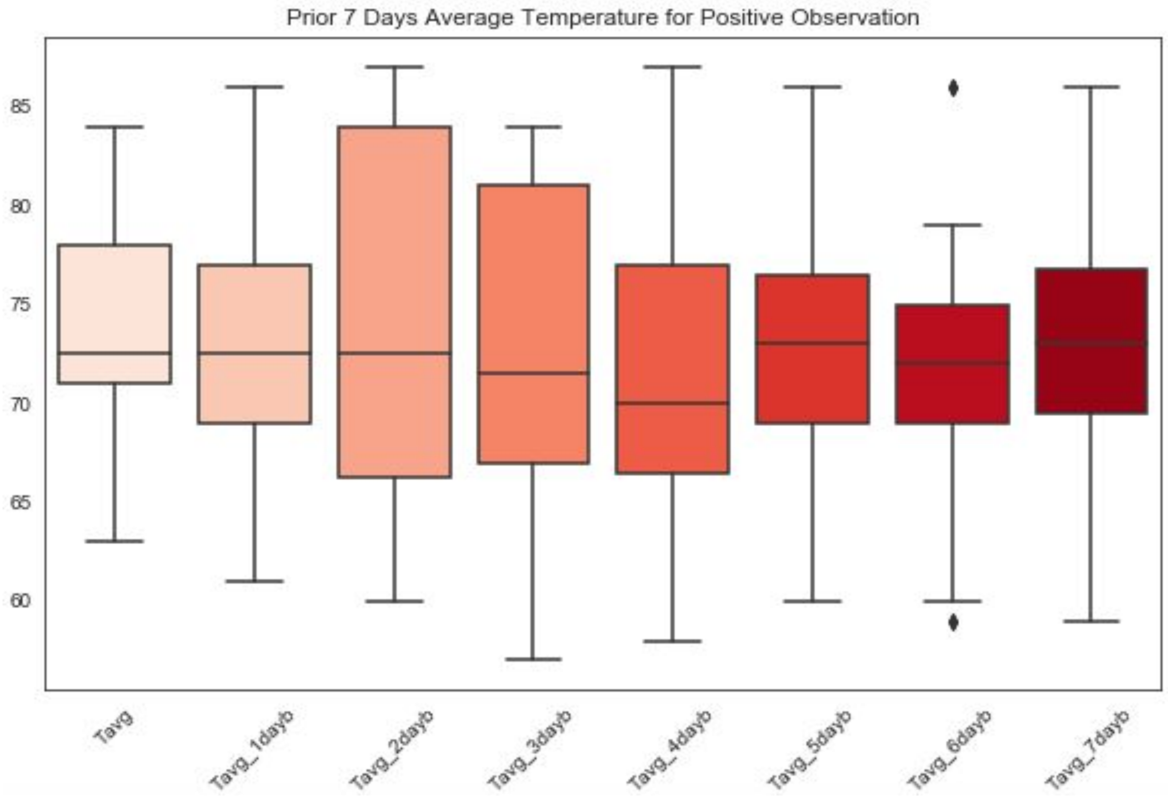


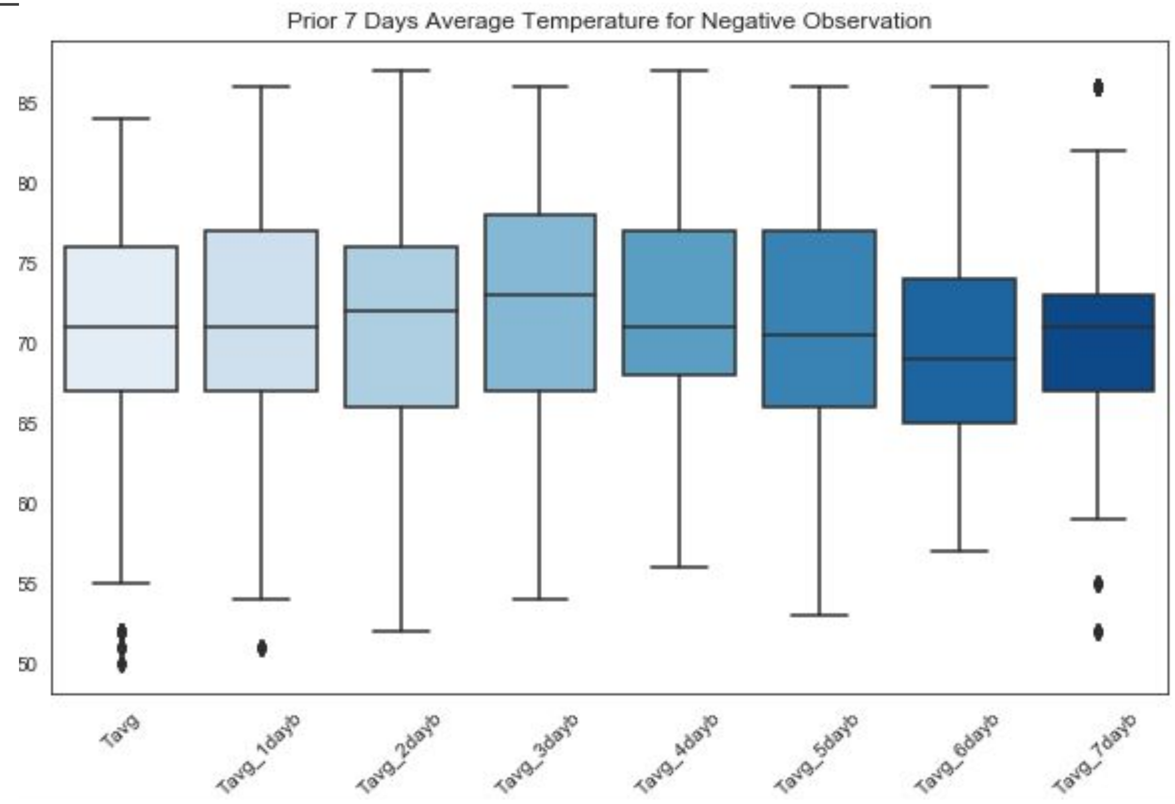
with WnV



without WnV

Trap T900 (Tavg Column)





with WnV

without WnV

Include all 28 new features for modeling

Modeling / Conclusion

Modelling

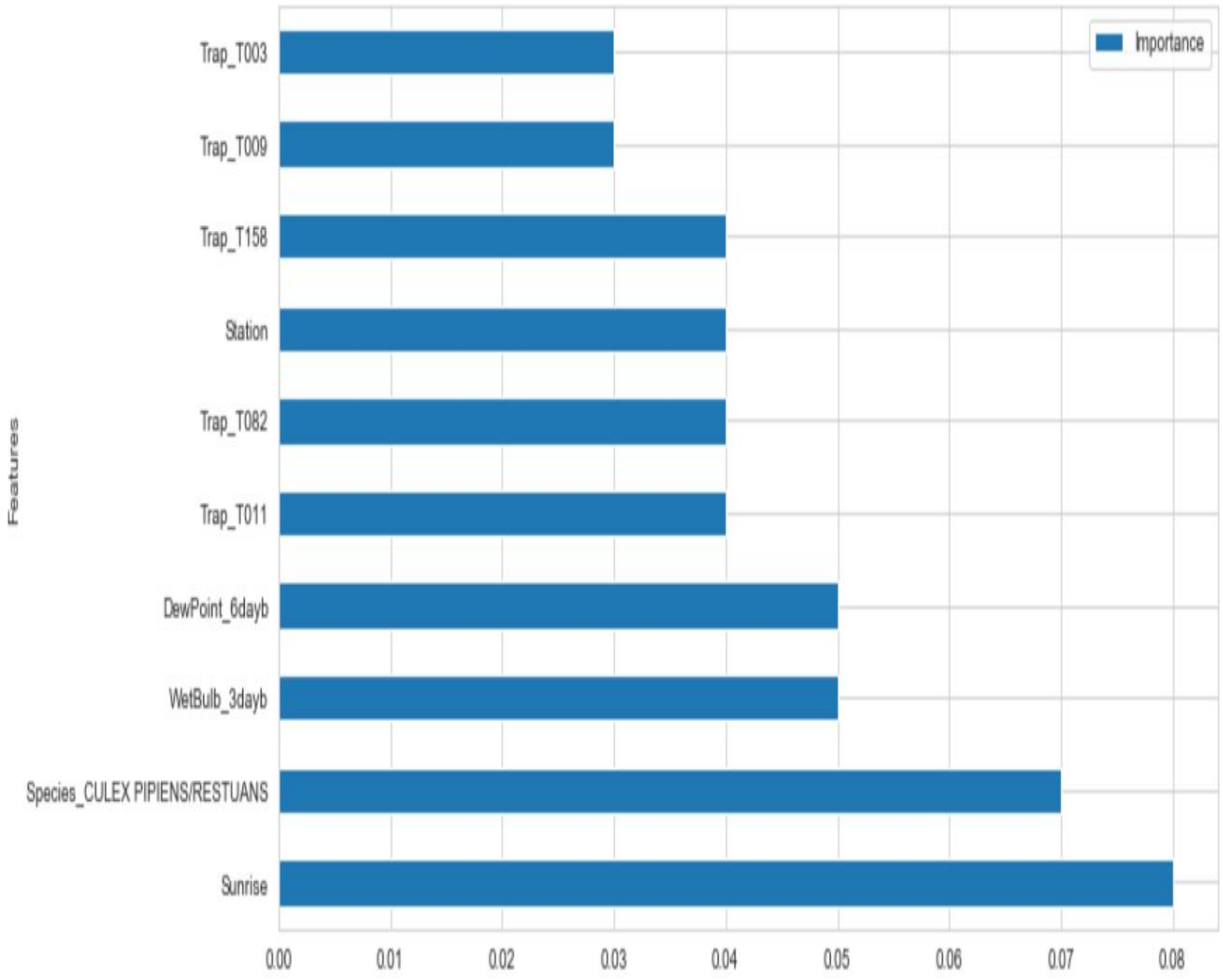
- Decision Tree
- Random Forest
- Ada Boost
- Logistic Regression
- Grid search was performed to find the best hyperparameters
- Model was fitted with dataset that was oversampled and not oversampled and submitted to Kaggle for scoring

	precision	recall	f1-score	support
0	0.90	0.81	0.85	2983
1	0.83	0.91	0.87	2990
accuracy			0.86	5973
macro avg	0.86	0.86	0.86	5973
weighted avg	0.86	0.86	0.86	5973

- The cost of predicting a negative WNV when it turns out to be positive is higher

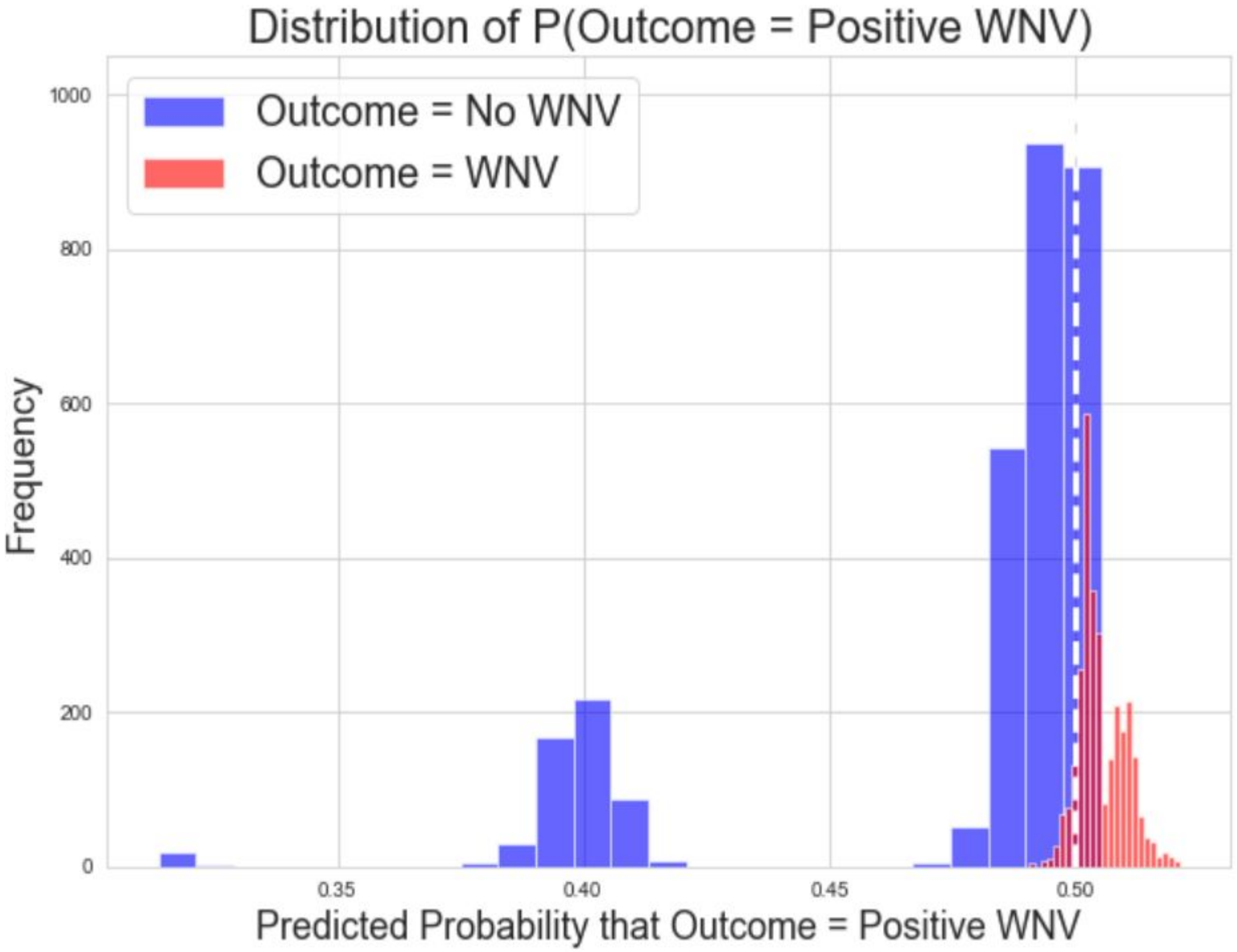
- Ada Boost with Oversampled Data has high recall and less false positive

Modeling - Evaluation of Model



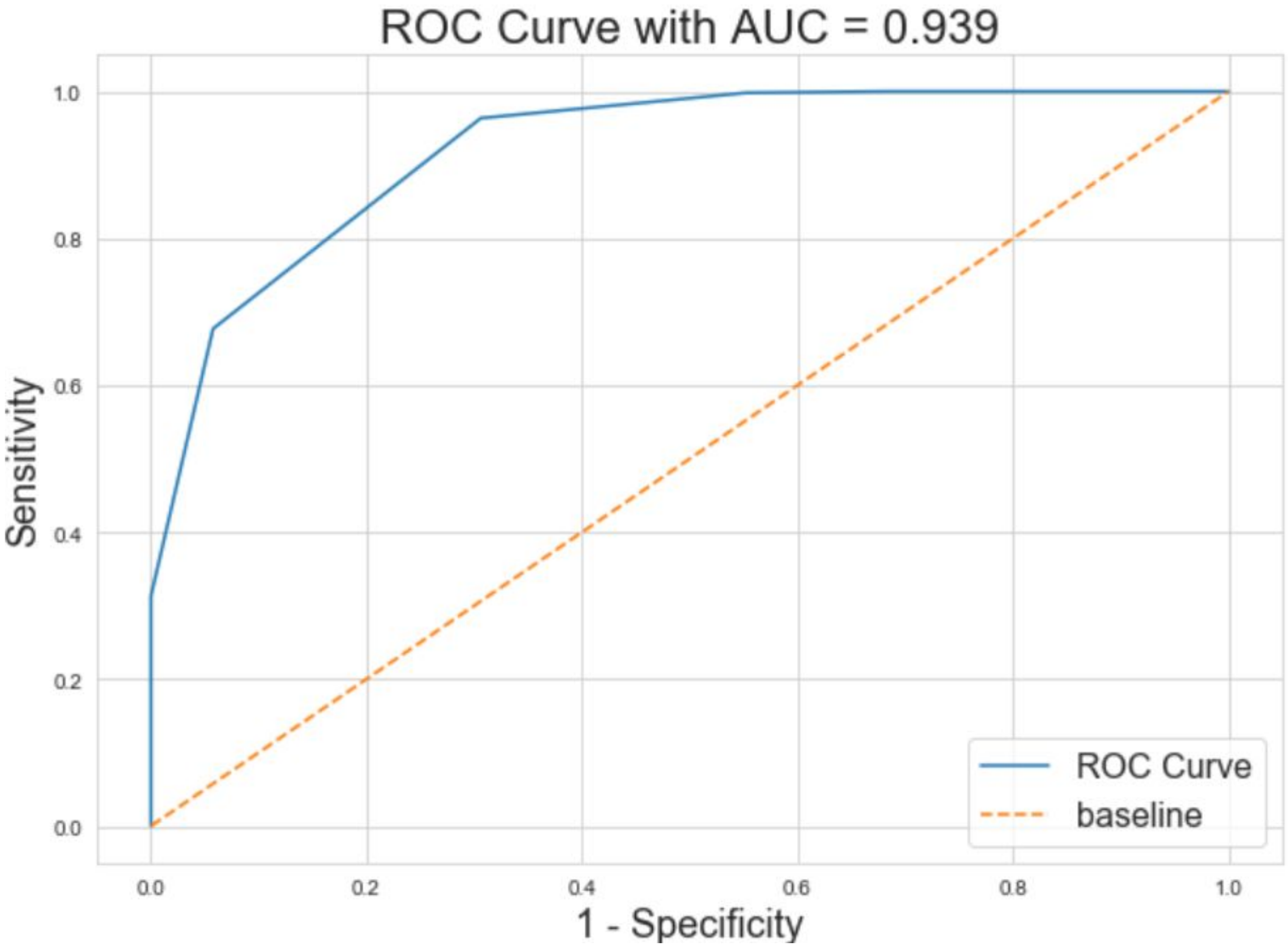
- This shows the top 10 features based on Importance
- We can see that some traps are considered as top features
- Some other features includes
    - Station
    - DewPoint
    - WetBulb
    - Sunrise
    - Species Culux Pipien and Restuan

Modeling - Evaluation of Model



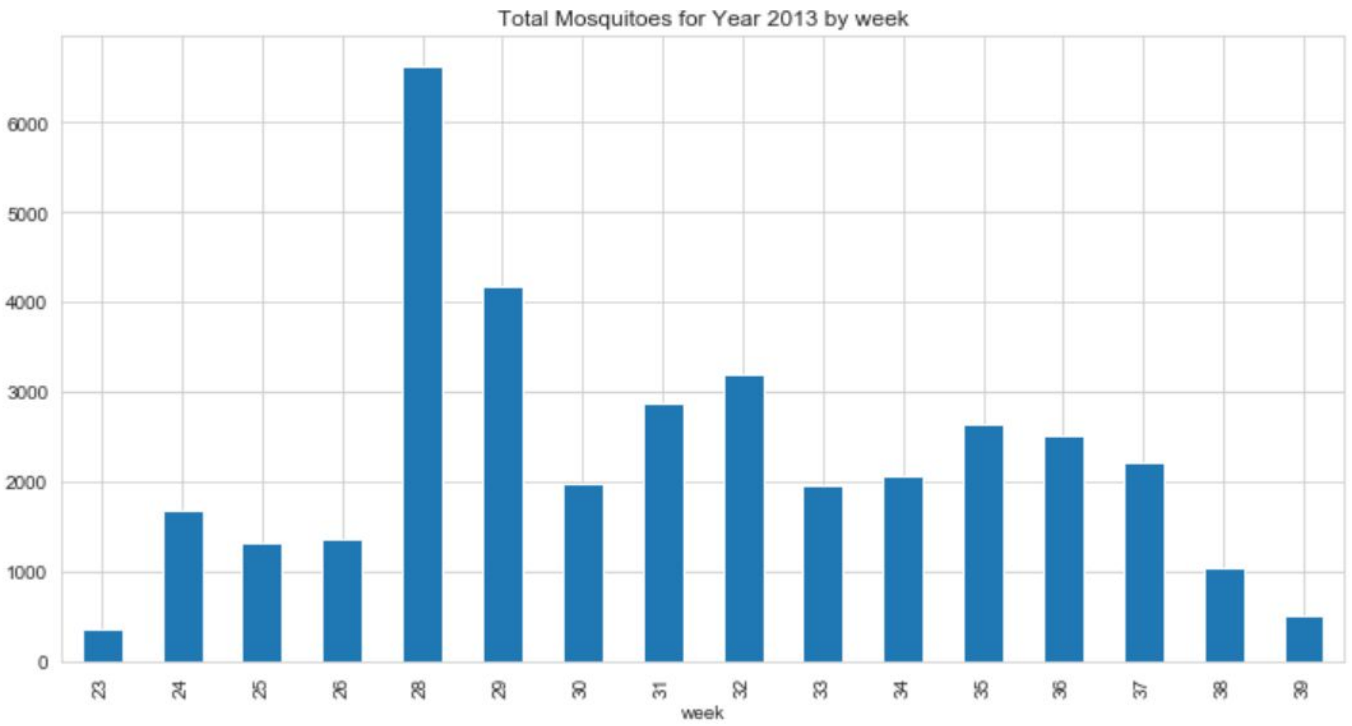
- Distribution of Predicted Probabilities
    - Blue are negative classification
    - Red are positive classification
- Misclassifications are blues to the right of the white line and red to the left
    - 576 False Positives
    - 263 False Negatives

Modeling - Evaluation of Model



- A good model's ROC AUC is above 0.5 and close to 1
- A ROC AUC of 0.5 means that our positive and negative population overlaps perfectly
- If the ROC AUC is below 0.5, it means that our model inversely classify the observations
- Our ROC AUC is 0.843

Cost Benefit Analysis



- Spray was performed on week 29
  - We can see that mosquito population decreased significantly
- Most people who become infected with West Nile virus have mild symptoms
- Typically less than 1% develop severe neroinvasive disease, according to CDC
- We can look into the cost of each spraying session and determine if the costs covers the risks or consider other alternative

source

Conclusion

- Model selection should be based on business needs
    - We would hope to be able to achieve zero Type II error
    - For this instance, recall should take precedence over other metrics
    - We can consider getting information on high risk bird species

- Its subjective to determine the ROI or perform Cost Benefit Analysis when life is involve
  - You cannot put a value on life
  - Different School of Thoughts