Capstone 1 - Final Report

Problem and Client

The city of Phoenix, AZ is the 5th largest city in the United States. Over the last 6 years the population of Phoenix has increased. With the increase in population there is the potential for an increase in crime. This poses the question: 1) ***Does it make sense for the city of Phoenix to hire more police officers to support the population increase?; 2) Can we use historical data crime data to determine where crime hot spots will be to allocate resources more efficiently and not have to hire new officers?*** The clients for this project is the Phoenix Police department and City of Phoenix.. They can use the findings to allocate resources(officers, patrols routes, etc.), make budgetary decisions and develop community outreach programs to prevent crime.

Data

The city of Phoenix publishes an updated crime incident list at 11am daily (https://phoenixopendata.com/dataset/crime-data). The list contains crime incident data from November 1st, 2015 forward through 7 days prior to today's posting date. The data provided includes; Date/Time of Incident, UCR crime type, Block Address, Zip code and Premise Type. In addition, I will append demographic data for Phoenix to help answer our question which is also provided by the city of Phoenix.

Approach and Deliverables

The approach will be to use the historical crime data to show how accurately the **type of crime** can be predicted. Multiple machine learning technique will be utilized to determine which provides the most accurate prediction. The deliverables for this project will be all the code, a report detailing the work completed and a slide deck providing the story and answers.

Data Wrangling

The dataset being used for this project is from the City of Phoenix Open data portal. The dataset is the reported crimes for the City of Phoenix from officer responded incidents and online reported incidents. The data provided is from November 1st, 2015 forward through 7 days prior to the newest posting date. The data used for the capstone project is from 11/1/15 to 2/20/18 (the csv file was downloaded on 2/27/18). This dataset has 148468 rows, 7 columns and 2 data types: object(6) and float64(1).

Reviewing missing data showed 4 columns have missing data:

1. Date_occurred_on = 336
2. Date_occurred_to = 43719
3. Zip code = 3
4. Presmise_type = 805

To clean the missing values for the date columns they will be filled in with the date/time from the populated date column of that incident. A check was run to display the rows that were NaN for date_occurred_on and date_occurred to and it returned 0 rows where that condition is true (code shown below).

```
#Fill in missing date values with date_occurred_on or date_occurred_to
crime_clean.date_occurred_on.fillna(crime_clean.date_occurred_to, inplace=True)
print(crime_clean.iloc[248])
crime_clean.date_occurred_to.fillna(crime_clean.date_occurred_on, inplace=True)
print(crime_clean.iloc[24])
print(crime_clean[['date_occurred_on', 'date_occurred_to']].isna().sum())
```

Since we had missing values of 336 and 43719 we cannot omit those rows. The 43719 missing values are due to the victims knowing the exact date/time of the incident therefore the date_occurred_to field is left blank. The 3 zip code rows will be removed since 3 is a very insignificant number that will not affect the project when removed. The premise_type missing values will be classified as unknown, there is already an unknown variable entry. The 805 could not be removed since it is a large number of rows that could affect the project (code shown below).

```
#Premise type NaN convert to Unknown
crime_clean.premise_type = crime_clean.premise_type.fillna('UNKNOWN')
```

To make it easier for the model to learn and to provide meaningful data, the date columns were converted to datetime and broken out into individual features (code shown below.)

```
# Convert date columns to datetime
crime_clean['date_occurred_on'] = pd.to_datetime(crime_clean['date_occurred_on'],
format='%m/%d/%Y %H:%M')
crime_clean['date_occurred_to'] = pd.to_datetime(crime_clean['date_occurred_to'],
format='%m/%d/%Y %H:%M')
#crime_clean.info()

# Break down datetime columns into separate columns, new columns listed below
# date_occ_on ==> [occ_on_month, occ_on_day, occ_on_year, occ_on_hr, occ_on_min]
crime_clean['occ_on_month'] = crime_clean['date_occurred_on'].dt.month
crime_clean['occ_on_day'] = crime_clean['date_occurred_on'].dt.day
crime_clean['occ_on_year'] = crime_clean['date_occurred_on'].dt.year
crime_clean['occ_on_dayofweek'] = crime_clean['date_occurred_on'].dt.weekday_name
crime_clean['occ_on_time'] = crime_clean['date_occurred_on'].dt.time
crime_clean['occ_on_hr'] = crime_clean['date_occurred_on'].dt.hour
crime_clean['occ_on_min'] = crime_clean['date_occurred_on'].dt.minute

# date_occ_to ==> [occ_to_month, occ_to_day, occ_to_year, occ_to_hr, occ_to_min]
crime_clean['occ_to_month'] = crime_clean['date_occurred_to'].dt.month
crime_clean['occ_to_day'] = crime_clean['date_occurred_to'].dt.day
crime_clean['occ_to_year'] = crime_clean['date_occurred_to'].dt.year
crime_clean['occ_to_dayofweek'] = crime_clean['date_occurred_to'].dt.weekday_name
crime_clean['occ_to_time'] = crime_clean['date_occurred_to'].dt.time
crime_clean['occ_to_hr'] = crime_clean['date_occurred_to'].dt.hour
crime_clean['occ_to_min'] = crime_clean['date_occurred_to'].dt.minute
```

Taking the the newly created time date features the occured on time and occured on month were grouped into time buckets and seasons (code shown below).

```
# Classify incident times into 4hr bucket groups
# A: 0-3, B: 4-7, C: 8-11, D: 12-15, E: 16-19, F: 20-23
occ_on_hr_grp = []
for row in crime_clean.occ_on_hr:
    if row in [0,1,2,3]:
        occ_on_hr_grp.append('A')
    elif row in [4,5,6,7]:
        occ_on_hr_grp.append('B')
    elif row in [8,9,10,11]:
        occ_on_hr_grp.append('C')
    elif row in [12,13,14,15]:
        occ_on_hr_grp.append('D')
    elif row in [16,17,18,19]:
        occ_on_hr_grp.append('E')
    else:
        occ_on_hr_grp.append('F')

crime_clean['occ_on_hr_grp'] = occ_on_hr_grp
```

```
# Classify incident months into seasons
# Spring(3,4,5), Summer(6,7,8), Fall(9,10,11), Winter(12,1,2)
occ_on_season = []
for row in crime_clean.occ_on_month:
    if row in [3,4,5]:
        occ_on_season.append('Spring')
    elif row in [6,7,8]:
        occ_on_season.append('Summer')
    elif row in [9,10,11]:
        occ_on_season.append('Fall')
    else:
        occ_on_season.append('Winter')

crime_clean['occ_on_season'] = occ_on_season
```

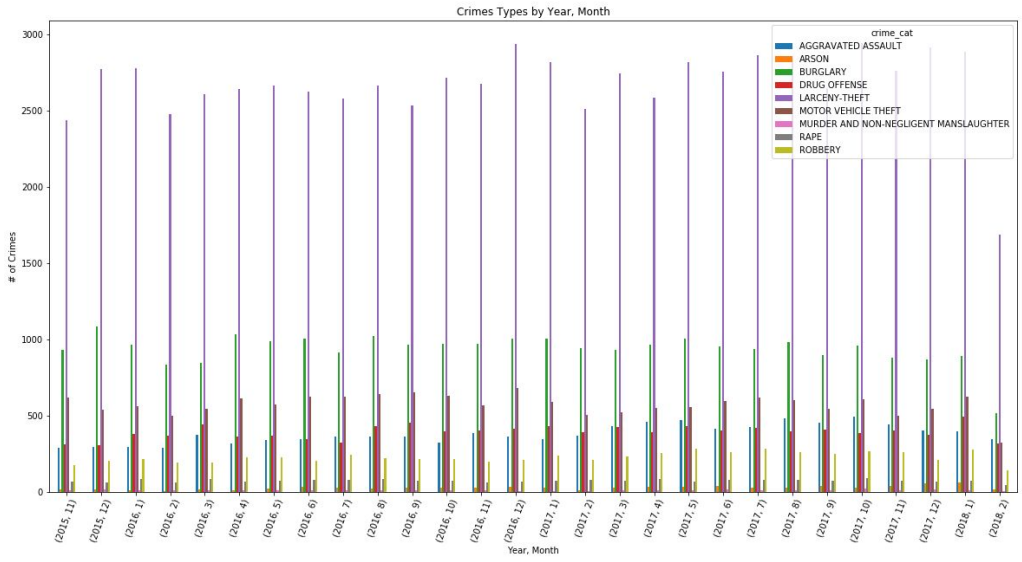
Exploratory Data Analysis

The crime data set is categorical type data set. The graphs created show counts of the groups within each variable based on different filters. To draw insights from the data using visualizations some initial questions were asked:

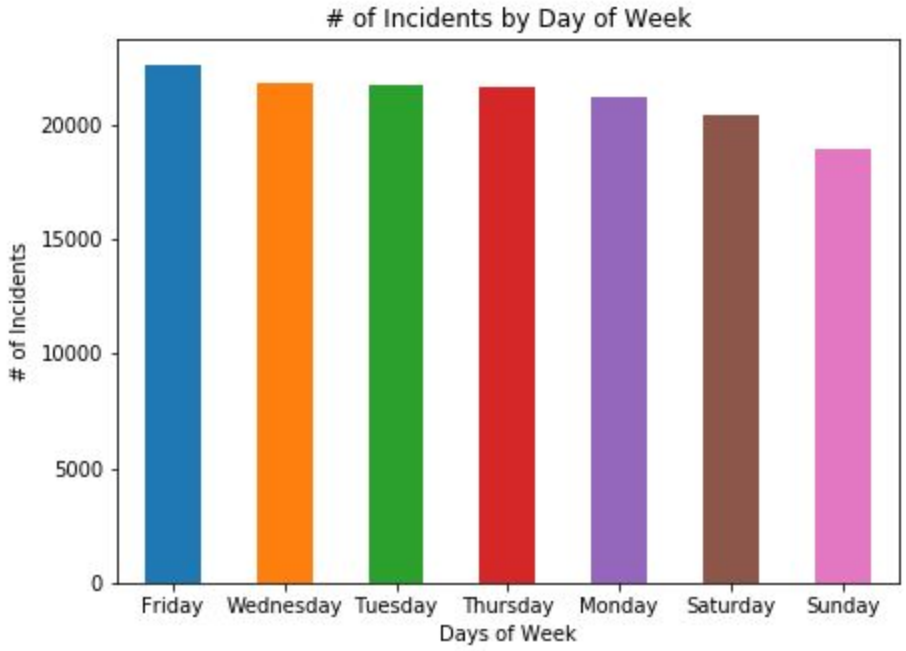
1. What are the top trending crimes?
2. When do crimes occur: Month, Day, Year, Season?
3. Do more crimes occur in certain zip codes?
4. What is the most common place a crime occurs e.g. house, business, etc.?

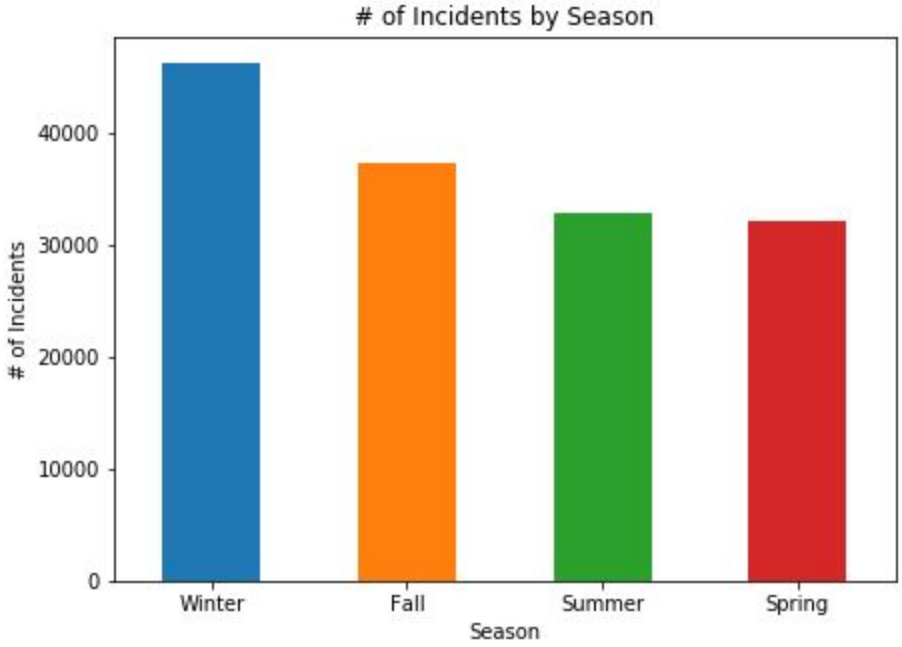
The answers to the questions above:

1. Top trending crimes are Larceny-Theft, Burglary, Motor Vehicle Theft, Drug Offenses

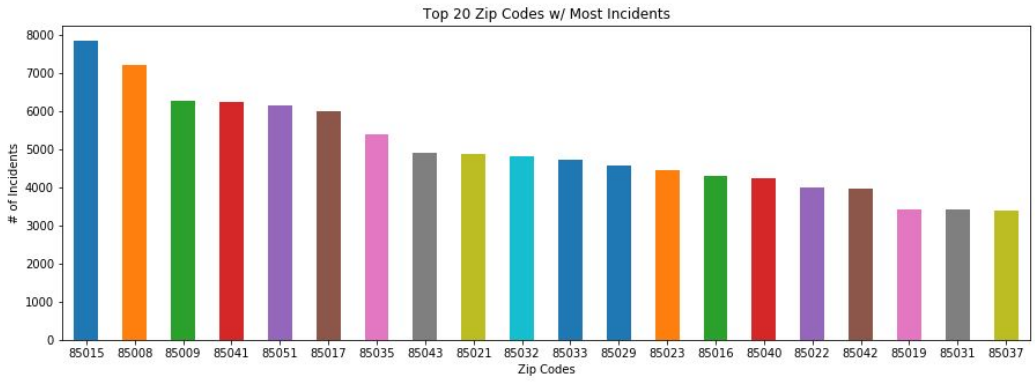


2. The most crimes occurred in the Winter months(Dec, Jan, Feb), Fridays and between the hours of 4-7pm.

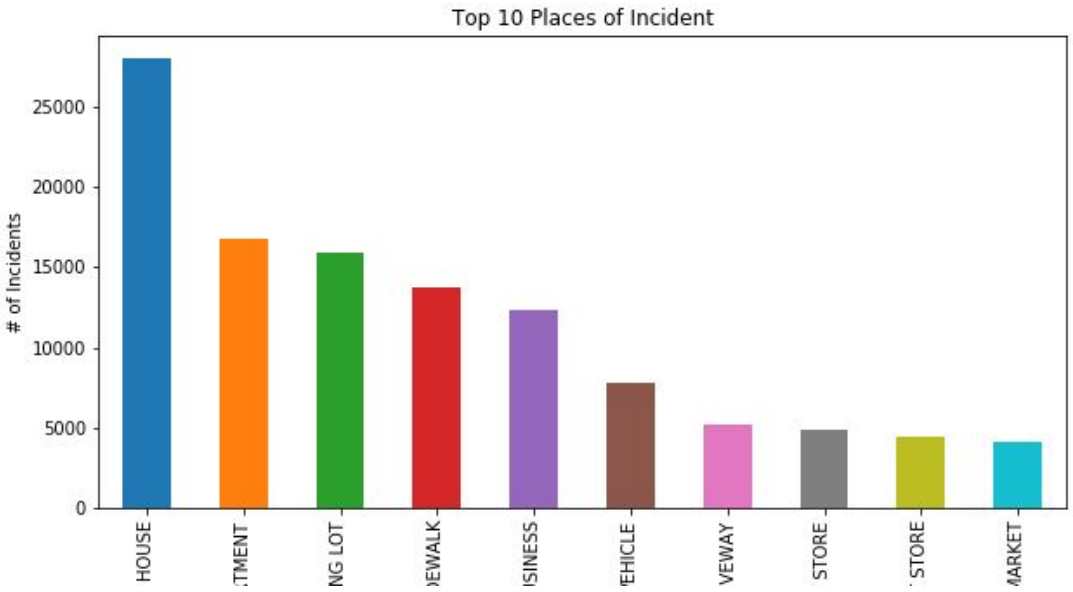




3. Two zip codes stood out with the most crimes 85015 and 85008.



4. Crimes typically took place at Single Family Homes, Apartments, and Parking Lots.



These answers provide us with good information to look at how these categories are related. A technique to help identify relationships or associations is Market Basket Analysis (MBA). MBA takes the data and determines what items are commonly associated e.g. people who purchase Milk & Bread also purchase cheese. To accomplish this we use the Apriori function from the Apyori library. The main outputs of the Apriori function are the **Frequent itemsets** and **Support metric**. **Frequent itemsets** are the item or sets of items most frequently occurring together and the **Support metric** is the fraction of transactions where item(s) occur divided by the total number transactions. The higher the support the more frequently the item(s) occur. Along with frequent itemsets and the support metric there are 2 additional key metrics **Confidence** and **Lift** which are known as Association rules. **Confidence** is the probability of seeing an itemset that contains an item of the itemset. **Lift** is used to measure how more often an itemset occurs than we would expect if the items in the itemset were statistically independent. A Lift score of of 1 indicates independence.

Using the Apriori function with the following minimums: min_support=0.05 and 0.003, min_confidence=0.2, a list of itemsets with support, confidence and lift were returned. The low Support and Confidence minimums were used to return more itemsets. What we are interested in seeing is what groupings appear most often for the different baskets. 5 baskets were created with different variables to see what items were associated most often. All 5 baskets had one variable in common, crime type. The results from the 5 baskets provided very little new information that couldn't be detected from visualizing the data. What it did show was the **Larceny-Theft** crime type was very prominent in the itemsets. This was expected due to it accounting for 50% of the crimes reported.

This was another reason the support levels needed to go so low to return helpful itemsets. The MBA was useful in returning frequent itemsets and showing that crime type might not be a good choice for the prediction variable due to the influence of one crime type.

Inferential Statistics

Using the information learned from the visualizations created and Market Basket Analysis, we created hypotheses to test on the population data. The data set is comprised of categorical variables therefore the Chi-Square test statistic was used. The Chi-Square test tests the strength of association between variables. 6 different hypotheses were tested and the results are listed below.

1. Compare Crime Category with Zip Code

- **Null Hypothesis:** There is no association between the crime type and zip code.
- **Alternate Hypothesis:** There is an association between the crime type and zip code.
- **Answer:** P-value = 0.0, null hypothesis is rejected

2. Compare Crime Category with Time Crime Occurred

- **Null Hypothesis:** There is no association between the crime type and time of day crime occurred.
- **Alternate Hypothesis:** There is an association between crime type and time of day crime occurred.
- **Answer:** P-value = 0.0, null hypothesis is rejected

3. Compare Crime Category with Season of the year

- **Null Hypothesis:** There is no association between crime type and season of the year.
- **Alternate Hypothesis** There is an association between crime type and season of the year.
- **Answer:** P-value = 3.4184825183899065e-07, null hypothesis is rejected

4. Compare Crime Category with Month

- **Null Hypothesis:** There is no association between crime type and premise type.
- **Alternate Hypothesis** There is an association between crime type and premise type.
- **Answer:** P-value = 1.6102836646351823e-18, null hypothesis is rejected

5. Compare Crime Category with Premise Type

- **Null Hypothesis:** There is no association between crime type and premise type.
- **Alternate Hypothesis** There is an association between crime type and premise type.
- **Answer:** P-value = 0.0, null hypothesis is rejected

From the hypothesis testing we see that Crime Type has significant associations with Zip Code, Time Crime Occurred, Premise Type, and Season of the year or Months.

Machine Learning

Before running our prediction models some additional data wrangling was required. We need to ensure that the training and test sets have comparable data. This means we need each set to have at least one of each of the classes within each variable. Initial attempts to split the data into training and test splits failed because of failed representation in the training and test sets. Upon investigation we had to remove rows where the zip code and premise type only appear once in the entire dataset. To ensure we have representation of the data in the training and test set Stratified Shuffle Split was used to accomplish this.

The dataset being used had 8 variables of data to feed into the models. To give our models even more data to learn from feature engineering was performed; 5 features were created. The data being used for the project is categorical and the features created were aggregations of the crime type, zip code and other important features (premise type, day of the week, season and time of day crime occurred). The target variable of crime type has 9 classes or 9 types of crime. The features were joined to the training and test sets to ensure the models can learn from the features. After joining the features to the test sets the categorical data including the target variable was encoded using Label Encoder.

To perform the machine learning we will use 3 methods to predict the target variable **Crime Type**; Logistic Regression, Naive Bayes and Decision Tree. Logistic Regression is the regression model used for categorical data and predicting a binary dependent variable. Naive Bayes is a easy to use model that works well with large datasets. Decision Tree is an algorithm used for classification that works well for categorical data. Using these 3 methods to predict the crime type we get the following accuracy scores:

	Logistic Regression	**Naive Bayes**	**Decision Tree**
CV Score Avg	0.6262	0.5971	0.6479
Accuracy Score	0.6619	0.6268	**0.9996**

The table shows that Decision Tree was very accurate with the test data. The classification report for Decision Tree shows the accuracy for each of the classes within the target variable.

	Precision	Recall	F1- Score	Support
0	1.00	1.00	1.00	2977
1	0.99	0.99	0.99	132
2	1.00	1.00	1.00	7691
3	1.00	1.00	1.00	3117
4	1.00	1.00	1.00	22225
5	1.00	1.00	1.00	4634
6	0.93	1.00	0.96	27
7	1.00	1.00	1.00	493
8	1.00	1.00	1.00	1783
Avg / Total	**1.00**	**1.00**	**1.00**	**43709**

The concern of overfitting did not come into play here because the accuracy scores for logistic regression and naive bayes were lower. If all three were in the 90 percent area then it could be thought overfitting was occurring.

Conclusion

In conclusion, using Decision Tree we are able to accurately predict the type of crime that will occur given a zip code and other factors. Our accuracy of 99% is very good almost too good. It appears that the features within the dataset work best with the decision tree technique. Further analysis could be performed to create more features to see if the accuracy goes down or stays the same and using different machine learning techniques.

The City of Phoenix can use this model to look at different zip codes around Phoenix to determine the types of crime that are most likely to occur in that zip code. They can use that information target along with additional analysis to narrow the target to specific hours of the day, neighborhood, and days of the week. If new information is recorded about the incidents that can be fed into the model to help the prediction. The accuracy obtained provide the City of Phoenix with a solid starting point to build upon to meet their needs in predicting crime to help protect the citizens of Phoenix.

The Jupyter Notebook containing all the code for the project can be found on GitHub (https://github.com/jpeterson28/DSCT_Capstone1)

References

1. City of Phoenix Open Data. 2017. *Crime Data*. [ONLINE] Available at: https://phoenixopendata.com/dataset/crime-data. [Accessed 27 February 2018].