COMP 3710 - Applied Artificial Intelligence

Vancouver Crime Prediction Machine Learning

AI Individual Project (Machine Learning)

Alexey Voropayev T00570110

11-20-2019

Alexey Voropayev T00570110

COMP 3710 - Applied Artificial Intelligence

Nov.20, 2019

AI Individual Project (Machine Learning)

Vancouver Crime Prediction Machine Learning

Introduction:

In modern world crimes are punishable acts that go against the law and need to be either prevented, stopped or, if happened, analyzed for patterns and evidences. A database of crime records and metadata of those records should be kept, maintained and analyzed for existing patterns for police to prevent or predict some of the future possible crimes. It is always good to keep every record in the database for the sake of the precision for crime prediction. However, for police officers sifting the data might take weeks and years to finally find some presence of patterns. Therefore, a machine learning algorithm should be developed to take crime data to predict possible future lawbreaking. The purpose of this project is to take a crime data of the city of Vancouver, which is collected from 2003 – 2017 years, and try to find any correlations between a type of crime and its time / location or get the possibility percentage for a certain crime to happen in a certain area and timespan subsequently.

Summary of what I did:

First, I extracted the dataset and observed it. I used the pandas library to insert additional columns to analyse the data easier via graph / plot representations. Second, I analyzed the data using the matplotlib library via constructing different sorts of plots and graphs to find the average number of crimes per day. Lastly, I prepared and preprocessed the data, trained my

machine on training set, and gave it my testing data to predict the possibilities with classification and regression models.

Background:

There have been done some similar works to mine, but with different datasets and different cities. Moreover, the datasets have been only analysed to find correlations, but were never fed to a machine to come up with any predictions. All the correlations have been done by a person and graphs. For example, one of the works has been done by Anand P. V. who analysed the crime dataset of North Caroline and found strong correlations between the crime rate, density of the population, urban areas, wage, and tax revenue per capita. Furthermore, the important theories and concepts to be taken into account while working with crime data are social disorganization theory (about A person's physical and social environments), politics, economics and etc. To explain, crimes are not always only driven by one factor. In fact, there are tons of influences behind each crime.

Project Details:

To begin, I used pandas library to analyse the dataset. It contains columns such as: 'TYPE' (type of crime), 'YEAR' (Year when the reported crime activity occurred), 'MONTH' (Month when the reported crime activity occurred), 'DAY' (Day when the reported crime activity occurred), 'HOUR' (Hour when the reported crime activity occurred), 'MINUTE' (Minute when the reported crime activity occurred), 'HUNDRED_BLOCK' (Generalized location of the report crime activity), 'NEIGHBOURHOOD' (Neighbourhood where the reported crime activity occurred), 'X' (Coordinate values projected in UTM Zone 10), 'Y' (Coordinate values projected in UTM Zone 10), 'Latitude' (Coordinate values converted to Latitude), 'Longitude' (Coordinate values converted to Longitude).

```
df = pd.read_csv('Z:\COMP 3710_01 - Applied Artificial Intelligence (Fall 19 Park
)\Project AI Crime\crime-in-vancouver\crimetrain.csv')
print(df.head())
```

```
/Project AI Crime/crime-in-vancouver/crimeVan.py"
         TYPE  YEAR  MONTH  DAY  HOUR  MINUTE     HUNDRED_BLOCK NEIGHBOURHOOD        X           Y   Latitude   Longitude
0  Other Theft  2003      5   12  16.0    15.0  9XX TERMINAL AVE    Strathcona  493906.5  5457452.47  49.269802 -123.083763
1  Other Theft  2003      5    7  15.0    20.0  9XX TERMINAL AVE    Strathcona  493906.5  5457452.47  49.269802 -123.083763
2  Other Theft  2003      4   23  16.0    40.0  9XX TERMINAL AVE    Strathcona  493906.5  5457452.47  49.269802 -123.083763
3  Other Theft  2003      4   20  11.0    15.0  9XX TERMINAL AVE    Strathcona  493906.5  5457452.47  49.269802 -123.083763
4  Other Theft  2003      4   12  17.0    45.0  9XX TERMINAL AVE    Strathcona  493906.5  5457452.47  49.269802 -123.083763
PS Z:\COMP 3710_01 - Applied Artificial Intelligence (Fall 19 Park)\Project AI Crime\crime-in-vancouver>
```

When I checked for data completeness using df.head() it showed that some cells do not include data due to personal privacy protection or the information was not simply provided. I then filled out some dummy information to escape error.

```
'''
 'RangeIndex: 530652 entries, 0 to 530651'
 'HOUR 476290 non-null float64'
 'NEIGHBOURHOOD 474028 non-null object'
 'HUNDRED_BLOCK 530639 non-null object'
  tells us that the data is incomplete in these areas.

  Need to fill in data identifies as 'Not given'
'''
df['HOUR'].fillna(0000, inplace = True)
df['NEIGHBOURHOOD'].fillna('Not Given', inplace = True)
df['HUNDRED_BLOCK'].fillna('Not Given', inplace = True)
```

To easier analyze the data, I then inserted two additional columns “FULL_DATE” and “DAY_OF_WEEK”.

```
df['FULL_DATE'] = pd.to_datetime({'year': df['YEAR'], 'month': df['MONTH'],
'day':df['DAY']})

''' use pandas.day_name() to determine the day of a crime / use .dayofweek for
 (Monday = 0, Tuesday = 1, etc.) '''
df['DAY_OF_WEEK'] = df['FULL_DATE'].dt.weekday_name
```

```
/Project AI Crime/crime-in-vancouver/crimeVan.py"
         TYPE  YEAR  MONTH  DAY  HOUR     HUNDRED_BLOCK NEIGHBOURHOOD        X           Y   Latitude   Longitude FULL_DATE DAY_OF_WEEK
0  Other Theft  2003      5   12  16.0  9XX TERMINAL AVE    Strathcona  493906.5  5457452.47  49.269802 -123.083763 2003-05-12      Monday
1  Other Theft  2003      5    7  15.0  9XX TERMINAL AVE    Strathcona  493906.5  5457452.47  49.269802 -123.083763 2003-05-07   Wednesday
2  Other Theft  2003      4   23  16.0  9XX TERMINAL AVE    Strathcona  493906.5  5457452.47  49.269802 -123.083763 2003-04-23   Wednesday
3  Other Theft  2003      4   20  11.0  9XX TERMINAL AVE    Strathcona  493906.5  5457452.47  49.269802 -123.083763 2003-04-20      Sunday
4  Other Theft  2003      4   12  17.0  9XX TERMINAL AVE    Strathcona  493906.5  5457452.47  49.269802 -123.083763 2003-04-12    Saturday
PS Z:\COMP 3710_01 - Applied Artificial Intelligence (Fall 19 Park)\Project AI Crime\crime-in-vancouver>
```

For subsequent data representation with matplotlib library I put the ‘FULL_DATE’ as the index of the dataset.

```
df.index = pd.DatetimeIndex(df['FULL_DATE'])
```

```
/Project AI Crime/crime-in-vancouver/crimeVan.py"
                TYPE  YEAR  MONTH  DAY  HOUR     HUNDRED_BLOCK NEIGHBOURHOOD        X          Y   Latitude   Longitude  FULL_DATE DAY_OF_WEEK
FULL_DATE
2003-05-12  Other Theft  2003      5   12  16.0  9XX TERMINAL AVE    Strathcona  493906.5  5457452.47  49.269802 -123.083763 2003-05-12      Monday
2003-05-07  Other Theft  2003      5    7  15.0  9XX TERMINAL AVE    Strathcona  493906.5  5457452.47  49.269802 -123.083763 2003-05-07   Wednesday
2003-04-23  Other Theft  2003      4   23  16.0  9XX TERMINAL AVE    Strathcona  493906.5  5457452.47  49.269802 -123.083763 2003-04-23   Wednesday
2003-04-20  Other Theft  2003      4   20  11.0  9XX TERMINAL AVE    Strathcona  493906.5  5457452.47  49.269802 -123.083763 2003-04-20      Sunday
2003-04-12  Other Theft  2003      4   12  17.0  9XX TERMINAL AVE    Strathcona  493906.5  5457452.47  49.269802 -123.083763 2003-04-12    Saturday
/Project AI Crime/crime-in-vancouver/crimeVan.py"
Categories of Crime and thier number of occurrences
Break and Enter Commercial                                982
Break and Enter Residential/Other                        1531
Mischief                                                  994
Offence Against a Person                                  836
Other Theft                                               409
Theft from Vehicle                                        508
Theft of Vehicle                                         2912
Vehicle Collision or Pedestrian Struck (with Fatality)     24
Vehicle Collision or Pedestrian Struck (with Injury)     1803
Name: TYPE, dtype: int64
```

Since car collisions do not really count as crimes unless intentionally committed, we exclude this category from the dataset with a function:

```
''' a function to sort out crime types on most occuring ones and others '''

def type(crime_type):
    if 'Theft' in crime_type:
        return 'Theft'
    elif 'Break' in crime_type:
        return 'Break and Enter'
    elif 'Homicide' in crime_type:
        return 'Homicide'
    elif 'Collision' in crime_type:
        return 'Vehicle Collision'
    else:
        return 'Others'
```

And create a new data frame without the car collisions:

```
car_coll = df[df['CATEGORY'] == 'Vehicle Collision']

crime = df[df['CATEGORY'] != 'Vehicle Collision']
```

Furthermore, I tried to predict / come up with the possibility prediction value for crimes.

I used different models for classification prediction and regression as well as I used different sets of features to predict either time, date or ‘everything’. First, I prepare the data for each of the by removing some of the unnecessary columns as such:

```
new_crime = crime.drop(['CATEGORY', 'DAY_OF_WEEK', 'YEAR', 'MONTH', 'DAY',
'HOUR', 'X', 'Y'], axis=1)
```

I preprocess the data using LabelEncoder() and fit_transform() methods.

```
le = preprocessing.LabelEncoder()
new_crime['TYPE'] = le.fit_transform(new_crime['TYPE'])
new_crime['HUNDRED_BLOCK'] = le.fit_transform(new_crime['HUNDRED_BLOCK'])
new_crime['NEIGHBOURHOOD'] = le.fit_transform(new_crime['NEIGHBOURHOOD'])
new_crime['Latitude'] = le.fit_transform(new_crime['Latitude'])
new_crime['Longitude'] = le.fit_transform(new_crime['Longitude'])
new_crime['FULL_DATE'] = le.fit_transform(new_crime['FULL_DATE'])
```

I assigned the sample and target variable as well as separated them into training and testing variables as such:

```
''' Preparing data and target vars '''

cols = [col for col in new_crime.columns if col not in ['TYPE']]
data = new_crime[cols]
target = new_crime['TYPE']

''' Splitting the data into training and testing sets '''
data_train, data_test, target_train, target_test = train_test_split(data, target,
 test_size = 0.33, random_state = 42)
```

At the end I applied Naïve-Bayes model for classification:

```
''' Naive-Bayes model for prediction'''

gnb = GaussianNB()
pred = gnb.fit(data_train, target_train).predict(data_test)
print("Naive-
Bayes accuracy : ",accuracy_score(target_test, pred, normalize = True))
```

The Lasso Model for regression:

```
'''  Lasso Model '''

y_pred_lasso = lasso.fit(data_train, target_train).predict(data_test)
r2_score_lasso = r2_score(target_test, y_pred_lasso)
print(lasso)
print("r^2 on test data : %f" % r2_score_lasso)
```

And The Elastic Net for regression:

```
''' Elastic Net '''

y_pred_enet = enet.fit(data_train, target_train).predict(data_test)
r2_score_enet = r2_score(target_test, y_pred_enet)
print(enet)
print("r^2 on test data : %f" % r2_score_enet)
```

I did the same steps for other predictions with the only features as ‘Latitude’, ‘Longitude’, ‘X’, ‘Y’ for place prediction, where I also used regression model SGD (Stochalistic Gradient Descent) for classification and regression:

```
''' Stochalistic Gradient Descent Classification'''
clf = SGDClassifier(loss="hinge", penalty="l2", shuffle=True, max_iter=10,)
clf.fit(data_train2, target_train2)
print('SGD classification val: ', clf.predict(data_test2))

''' Stochalistic Gradient Descent Prediction'''
clf2 = SGDClassifier(loss="log", penalty="l2", shuffle=True, max_iter=10,).fit(da
ta_train2, target_train2)
print('SGD prediction val: ', clf2.predict_proba(data_test2))
```

and prediction of the date of a crime with only features 'MONTH' and 'DAY'. There I used Lasso model for regression and SGD (Stochalistic Gradient Descent) for classification and regression.

Lastly, I tried to focus on fewer amount of data for better prediction of a specific field. To illustrate, I took only the 'Break and Enter Residential/Other' type of crime, month and hour of the crime to see whether its mostly occurrences happen during day or nighttime. At first, I remoced the rows where the 'HOUR' was not provided.

```
df = df[df.HOUR != -1.0]
```

Then, I wrote a function defining the time ranges for day and night and apply it to the 'HOUR' field.

```
''' define day and night ranges '''
def time_per(x):
    if (x >= 5.0) and (x < 18.0):
        return 'day'
    else:
        return 'night'

df['hour'] = df['HOUR'].apply(time_per)
```

After I chose the rows that only describe the 'Break and Enter Residential/Other' type of crime.

```
break_res = clock.loc[df['TYPE'] == 'Break and Enter Residential/Other']
```

I preprocessed the data the same way I did before and applied a classifier model – KneighborsClassifier.

```
''' KNN '''
knn = KNeighborsClassifier(n_neighbors=1)
knn.fit(data_train4, target_train4)
```

```
print('KNN val: ', knn.predict(data_test4))

a = knn.predict(data_test4)
unique, counts = np.unique(a, return_counts=True)

print('{0 - Day(5am to 7pm), 1 - Night(6pm to 4am)}\n',dict(zip(unique, counts)))
```

Then, I narrowed down the time range to see what part of a day is the most criminal when it comes to 'Break and Enter Residential/Other'. I once again defined the time ranges for each of the four-day parts:

```
''' 4 day-period check '''
def p(x):
    if (x >= 5.0) and (x <= 12.0):
        return 'Morning'
    elif (x > 12.0) and (x < 17.0):
        return 'Afternoon'
    elif (x >= 17.0) and (x < 21.0):
        return 'Evening'
    else:
        return 'Night'

df['DAY_PART'] = df['HOUR'].apply(p)
```

Chose only the 'Break and Enter Residential/Other' crime and used the KneighborsClassifier model to get the output.

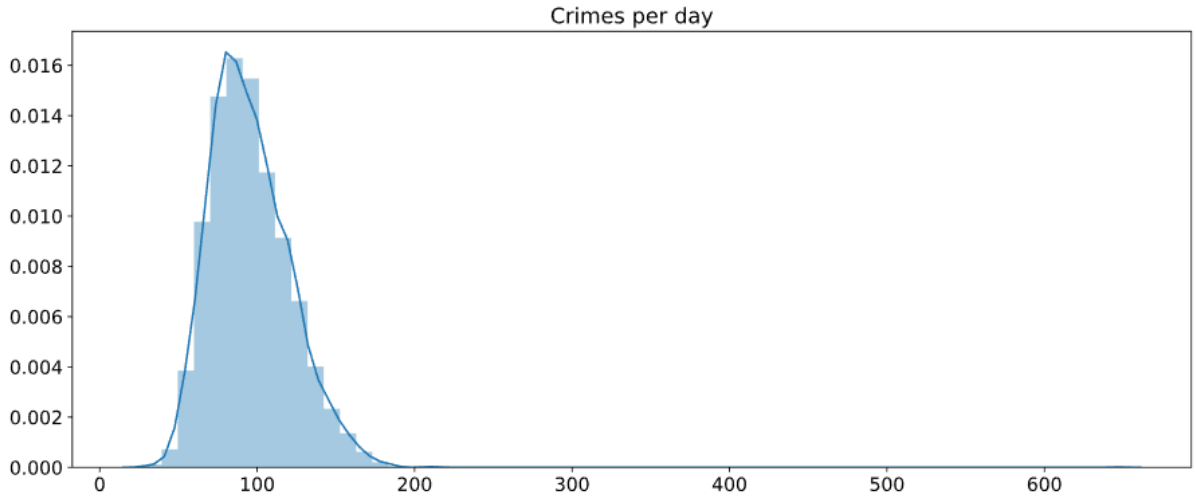
```
''' KNN '''
knn = KNeighborsClassifier(n_neighbors=1)
knn.fit(data_train5, target_train5)
print('KNN val: ', knn.predict(data_test5))

a = knn.predict(data_test5)
unique, counts = np.unique(a, return_counts=True)

print('{0 - Afternoon(12pm to 5pm), 1 - Evening(5pm to 9pm), 2 - Morning(5am to 1
2pm(noon)), 3 - Night(9pm to 4 am)}\n', dict(zip(unique, counts)))
```

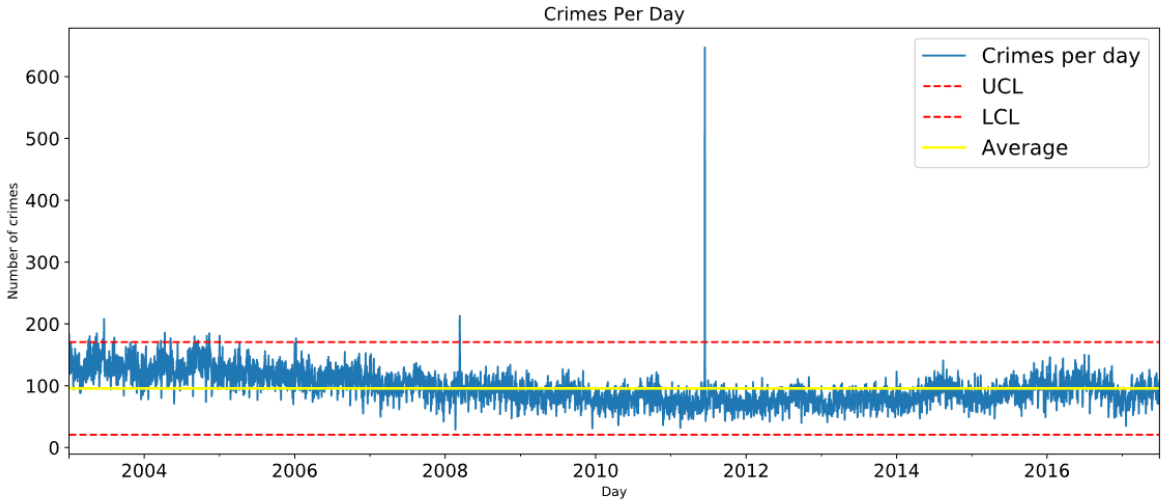
Results:

Exploring the dataset and illustrating it using matplotlib library, I mainly focused on visualizing 'Crimes per day'.

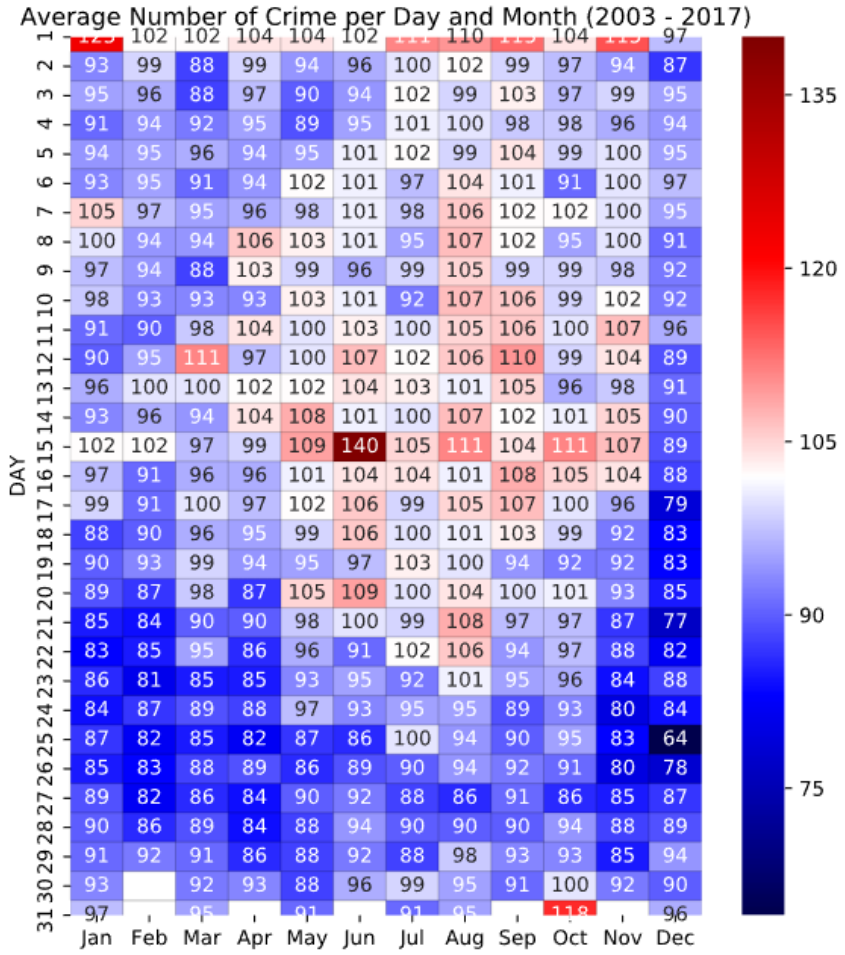


This char presents a usual number of crimes per day, which is around a 100.

After, I visualized the whole dataset with Upper and Lower bounds:



There present some outliers like for the date 2011-06-15 when Vancouver held the Stanley Cup's final game, which apparently led to way too many crimes that day. However, the overall graph looks clear and shows a slight reduction in crime rate over the past years. Then, I analyzed the data creating a heat map of 'Average Number of Crime per Day and Month (2003 - 2017)':



It shows us that most of the crimes happen on the day after the New Year's Eve and the average number of crimes occurs in the middle of a month from May-November.

When I tried to predict the locations and time for certain crimes to happen non of the classification / regression models were successful. This got logical backup since crimes are not that easy to predict and this field is one of the hardiest to deal with. Therefore, I took on fewer and only specific samples of data (i.e. 'TYPE', 'MONTH', 'HOUR') to see what part of the day-night is mostly susceptible to crime occurrences. The results showed that taking into account only day and night, the night crimes are as double bigger in amount than the day crimes.

```
KNN val:  [1 1 1 ... 1 1 0]
{0 - Day(5am to 7pm), 1 - Night(6pm to 4am)}
 {0: 6820, 1: 13234}
```

In addition, if we were to check what parts of the day are the most criminal the KNN classifier presents the output as:

```
KNN val:  [3 3 3 ... 3 1 2]
{0 - Afternoon(12pm to 5pm), 1 - Evening(5pm to 9pm), 2 - Morning(5am to 12pm(noon)), 3 - Night(9pm to 4 am)}
 {0: 3165, 1: 1663, 2: 3655, 3: 11571}
```

From this snapshot we can see that still the nighttime (9pm to 4 am) is the most criminal time for 'Break and Enter Residential/Other', and the second most criminal part is morning (5am to 12pm(noon)).

Discussion:

Working with crime datasets taught me that it is far not an easy case to apply different sorts of classification and regression models to come up with a solid and accurate prediction number. The reason for that is the nature of crimes. They can happen any where and any time. Motives are different as well as the crimes are different. To come to a truthful and realistic value a little bit closer it is better to focus on certain characteristics of a crime and narrow down the fields to work with. In my case, I got some of the accurate results by only working with the time and only one crime type to see any patterns. Thus, with the power of nowadays computers there are only a few specifically developed for this kind of business machines that can pretty accurately predict the future crimes. Most of the crime data available today is mostly suitable for analysis by humans to construct heatmaps, plots, and charts for better visualization of the crime patterns.

Future plan:

For the future work to make my code output better results, I would take an additional dataset (preferably a numeric one) to have more fields to work with. I believe that will make the final values more accurate and I would be able to find other crime characteristics to explore. To

add, I would try other libraries (like Deeplearning4j) for machine learning to work with this dataset.

Works Cited

Briggs, Steven, and Part of Criminology For Dummies Cheat Sheet. "Important Theories in Criminology: Why People Commit Crime." *Dummies*, https://www.dummies.com/education/psychology/important-theories-in-criminology-why-people-commit-crime/.

V, Anand P. "Can Machines Predict and Prevent Crimes?" *Medium*, Data Driven Investor, 1 Jan. 2019, https://medium.com/datadriveninvestor/can-machines-predict-and-prevent-crimes-9203f9e1524c.