Forecasting Seattle Crime

Laura Borton

Project Design:

The Open Data Program makes the data generated by the City of Seattle openly available to the public. Included in this program is information from public safety agencies, specifically the Seattle Fire and Police Departments. By using crime data, I wanted to predict the total number of offenses for the week ahead. This information could help law enforcement allocate resources for the next seven days.

Tools and Data:

The Crime Data file lists unique events where at least one criminal offense was reported by a member of the community or detected by an officer in the field. The “Occurred Date” and “Crime Subcategory” features were selected for dates ranging from January 6, 2018 to August 26, 2018, containing over 488,00 incidents.

To determine if weather information had any affect on the number of crimes, temperature and precipitation data from the National Oceanic and Atmospheric Information (NOAA) was also gathered for the date range above.

Standard Python packages were used for preprocessing the data. Keras (Tensorflow backend), a neural network library, and Facebook’s Prophet were two of four Python packages used for time series modeling. Matplotlib and Tableau were used for visualization.

Preprocessing:

After the data was loaded into Python, all crimes were summed together by week, specifically from Sunday at midnight to Saturday at 11:59pm. The average weekly temperature and precipitation were merged into the crime file by date.

Predictive Modeling:

At first, it was not clear how to group the data. Many trials were run varying the crime, neighborhood, date range, train/test split, and time grouping (day/week/month). In the end, all crimes were grouped together by week for all of Seattle for ten years to make the analysis more manageable. The data was split into train/test sets with a 70/30 ratio.

Baseline -- Average:

The simplest model was calculated as the average of the training data.

Baseline -- Naive (Persistence):

The baseline model uses the last time step value to predict the next time step value. This appears to be the standard baseline model for time series.

Linear Regression:

Before the decision was made to group the data by week, OLS and Stationarity tests were run. The regression models treated every day as an independent variable. Day of week, day of month, and month were used as additional variables. Day of week and month seemed to have significance so PCA was tested for dimensionality reduction. The results were not convincing so the days of week were grouped into weekday and weekend, and the months were grouped into summer, holiday (November and December), and other months in order to reduce the dimensionality. Temperature and precipitation were also tested and a few polynomials were used to fit the data. At this point, the OLS method was abandoned in favor of other methods.

SARIMAX:
The SARIMAX models were also run before settling on weekly Seattle data. Both the Box-Jenkins Methodology and the ARIMA model in Python's statsmodel library were tested on crime-specific data. Because a range of p,d,q variables can be evaluated quickly with statsmodel ARIMA, it was chosen as the preferred model. It was not clear how to incorporate exogenous variables; thus the "X" (exogenous) parts of SARIMAX was ignored. Yearly plots were stacked vertically to observe seasonality. None was found, and the "S" (seasonal) part of SARIMAX was eliminated, leaving just the ARIMA model.

Prophet:
Facebook's Prophet model is very fast and easy to use. However, it is a bit of a "black-box" and requires accepting some default parameters. For example, the gradient descent algorithm is not obvious and it is not clear if it can be changed. Most improvements came by altering the changepoint prior scale and season variability parameters. Adding temperature and precipitation as variables was easy.

LSTM:
Most time was spent on LSTM trials. Preparing data for input to this neural network is quite tedious, but it allows for tuning many parameters. The following tests were run:

1. Use last time step value to predict the next time step value (similar to Persistence model)
    a. Training data shuffled
    b. Training data unshuffled
    c. Training data shuffled with stationarity
    d. Training data unshuffled with stationarity
2. Use last three time steps to predict the next value
    a. Shuffled
    b. Shuffled with stationarity
    c. Unshuffled
3. Use last three time steps sequentially as one time step to predict the next value
    a. Shuffled
    b. Shuffled with stationarity
    c. Unshuffled
4. Use last three time steps sequentially with memory (Stateful LSTM) -- all LSTMs with memory are unshuffled
    a. Non-stationarity
    b. Stationarity
5. Use last time step sequentially with memory
    a. Adam
    b. RMSProp
6. Use last time step sequentially with stacked memory
    a. Non-stationarity
    b. Stationarity
7. Use last two time steps with stacked memory -- non-stationarity

8. Use last three time steps sequentially with memory
9. Use last three time steps sequentially with stacked memory (additional layer)
    a. Non-stationarity
    b. Stationarity
10. Multivariate (precipitation and weather added) with memory
    a. Non-stationarity
    b. Stationarity

In addition, a handful of tests varying the number of batch-sizes, neurons, and epochs were run before choosing 1 batch-size, 100 epochs and 1 neuron.

It seems there is debate if neural nets that learn long-term dependencies (LSTM) need to be stationary. Because of this, the best result, whether stationary or not, was used.

For all models above, the RMSE was used as the metric. On the baseline and final best model, MAPE was also calculated.

Summary:
Because input data sets changed, most of the test results can not be considered equal--but they were used for comparison. LSTM provided the best results, with an RMSE of 57.17 crimes per week, and a MAPE of just under 5%. The naive baseline model had an RMSE of 68 crimes/per week and a MAPE of almost 7%. It is important to note that Prophet’s results were almost the same, and the model is much faster and easier to manipulate.

Future Work:
There is an opportunity to vary the input to crimes per neighborhood instead of crimes for all Seattle. This would not only help allocate the number of officers, it would help where to assign them. The model can be changed to daily instead of weekly for granularity, and time would be a nice feature for shift scheduling.