Predicting type of crime from location and time information

Zain Nofal, Tirth Joshi, Nicole Link

2025-12-07

Table of contents

Summary

In this project, we aim to predict what type of crime occurred in Vancouver based on when and where it happened. This is important because understanding crime patterns can help police departments allocate resources more effectively and plan better patrol routes. We use a dataset from the Vancouver Police Department with over 530,000 crime records from 2003-2017, covering 11 different crime types including theft, break-ins, and vehicle collisions.

We tested three machine learning models: K-Nearest Neighbors, Support Vector Machines, and Logistic Regression. After tuning, all three models performed similarly, achieving an accuracy around 0.5. While this shows that time and location do provide some useful information for predicting crime types, the moderate accuracy suggests there are limitations. The models only use when and where crimes happen, without other information like weather, economic conditions, or other factors that might matter.

Introduction

Background

Crime prediction is an important tool for police departments trying to figure out where to focus their resources. Vancouver, like most big cities, has many different types of crime happening at different times and places. If we can predict what kind of crime is likely to happen based on patterns in the data, it could help with planning patrols and prevention efforts.

Research Question

Can we predict the type of crime based on when and where it happens?

We're comparing three different classification algorithms (K-NN, SVM, and Logistic Regression) to see if this is possible, and which model works best for this problem.

Methods

Data

We're using the Vancouver Crime Dataset from Kaggle, which originally came from the Vancouver Police Department (Osaku 2017). It has 530,652 crime records from 2003 to 2017, split into 11 crime types. The most common is "Theft from Vehicle" (over 172,000 cases) and the rarest is "Homicide" (220 cases). We selected the four most common types of crime, to ensure

that all of our target classes have a sufficient number of observations. The counts of these crimes in our training data are shown in Figure 1.

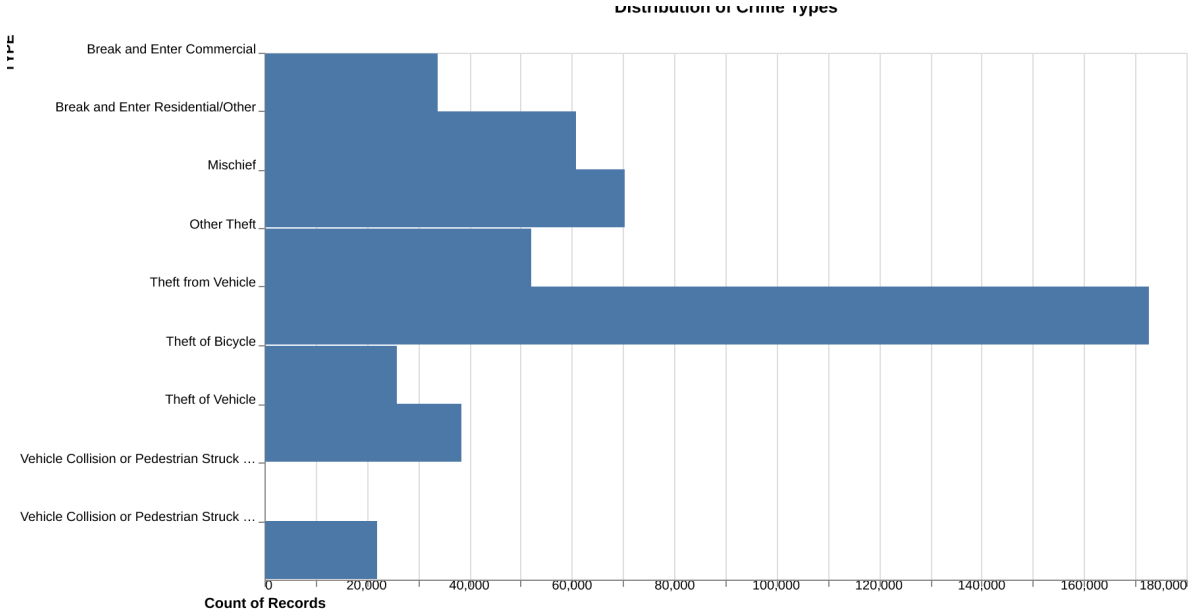

Figure 1: Observed counts of the four most frequent crime types in the training dataset.

Each record includes:

- Time info: year, month, day, hour, minute
- Location info: neighborhood, street block, coordinates

There's some missing data - about 10% of records don't have time information and 11% are missing neighborhood data. We filled in missing times with the most common values and labeled missing neighborhoods as "Unknown."

Analysis

We selected three models to use to build a classification model to predict the crime type for each incident in the Vancouver crime dataset: a k-Nearest Neighbors (k-NN) algorithm, a linear Support Vector Machine (SVM) classifier, and a multinomial Logistic Regression model (LogReg). For all tested models, all spatial, temporal, numeric, and categorical features were included. The categorical variables were one-hot encoded, and numeric variables were standardized using a column transformer immediately before model fitting. The crime dataset was split using an 80/20 stratified train test split, in order to preserve class proportions in both sets. A baseline model was created for each model type, using the default hyperparameter

values. Then, hyperparameter optimization was performed on all models, using a 15,000-observation stratified subset of the training data, in order to decrease computational load. The best fit models found from these optimizations were then scored on the test data set. The analysis was conducted in Python using python (Van Rossum and Drake 2009), numpy (Harris et al. 2020), pandas (team 2024), matplotlib (Hunter 2007), tqdm (Costa-Luis 2019), click (Team 2020), and scikit-learn (Pedregosa et al. 2011). The code used to perform this analysis is available at: https://github.com/nicolelink33/Vancouver_Crime_Predictor.

Results & Discussion

Model Performance

KNN

The baseline k-NN model, using k = 5, achieved an accuracy of 0.44. After hyperparameter optimization, the optimal value was found to be k = 85, yielding a final accuracy of 0.51 on the test set. The hyperparameter optimization results are shown in Figure 2.

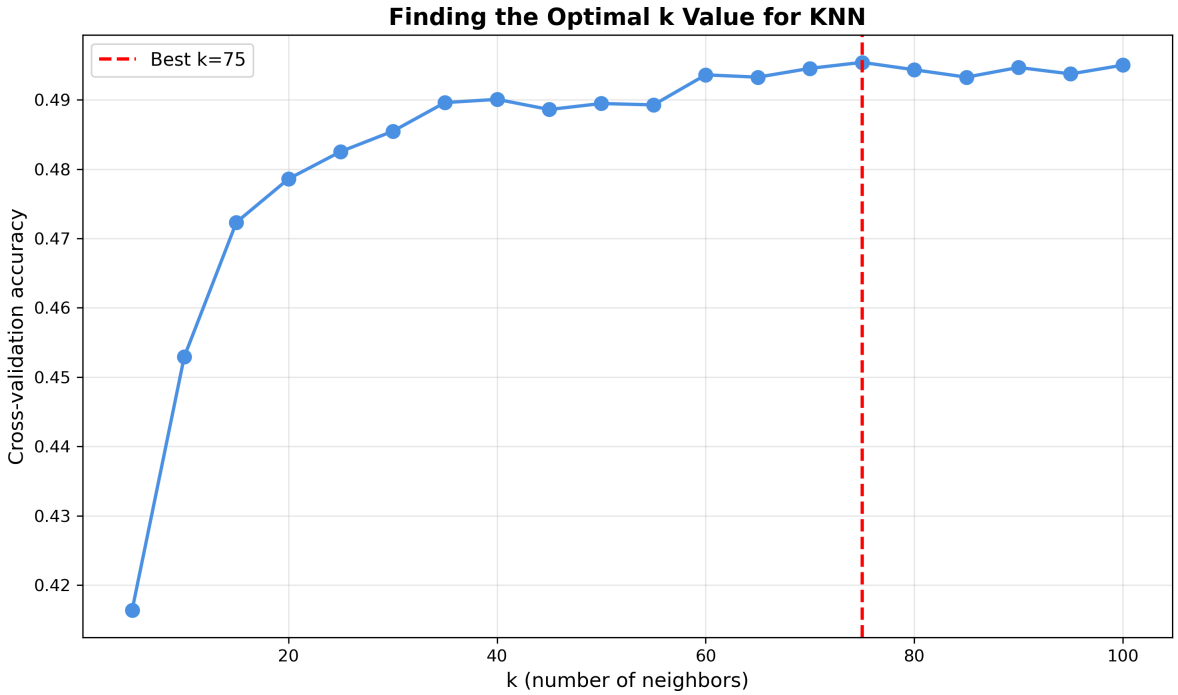

Figure 2: Hyperparameter optimization of the k value for k-NN model.

This model performed best on “Theft from Vehicle”, while categories such as “Mischief” and “Break and Enter” were more challenging. k-NN model performance by crime type is shown in the produced confusion matrix, shown in Figure 3.

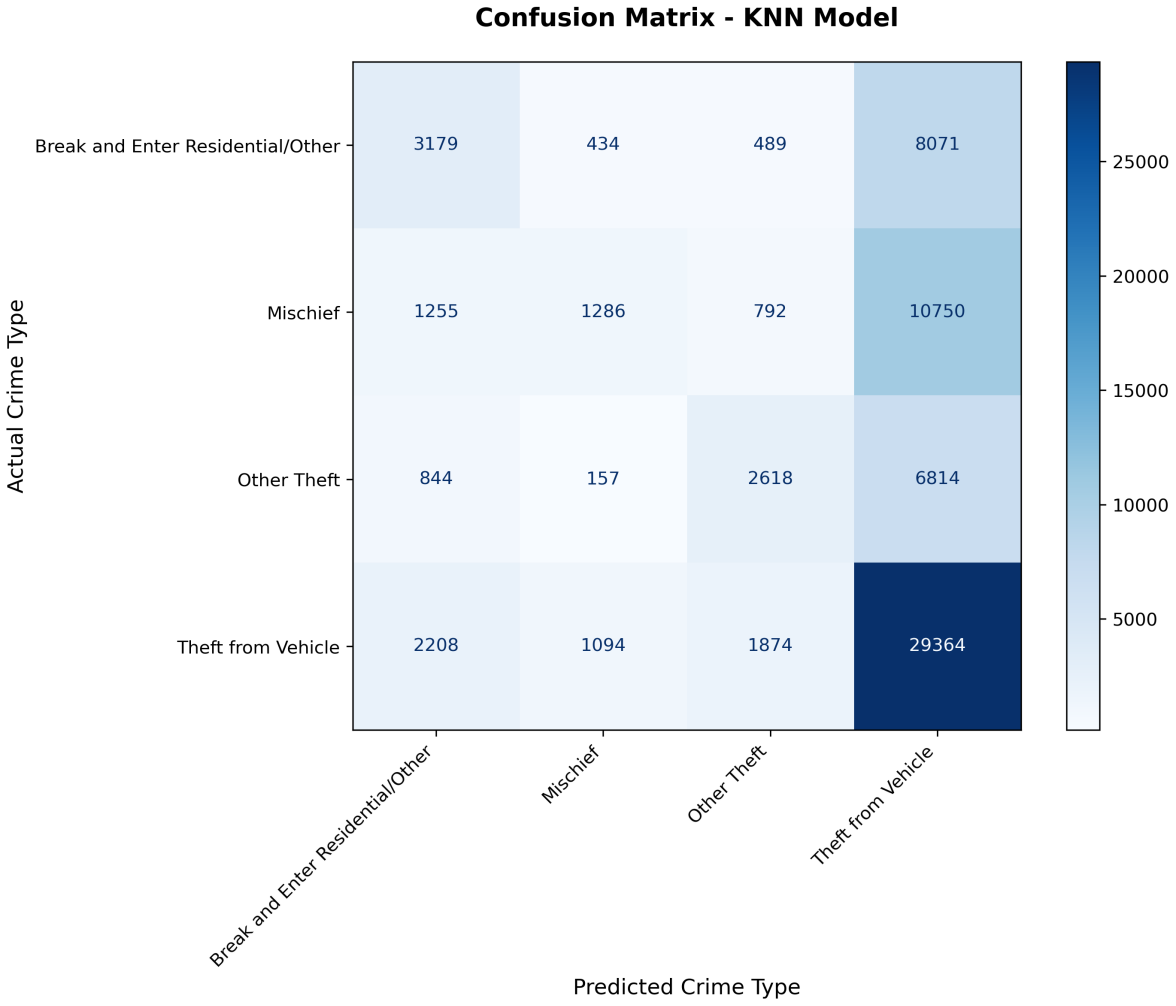

Figure 3: Confusion matrix for best fit k-NN model, showing predicted crime type versus actual crime type.

SVM

The baseline LinearSVC model with C = 1 achieved an accuracy of 0.5. A best fit model was then found through hyperparameter optimization of C, which gave a final accuracy of 0.5 and an F1 score of 0.39, only slightly higher than the baseline, indicating that model performance

was not highly sensitive to the value of C. The confusion matrix for the best fit linear SVM model is shown in Figure 4.

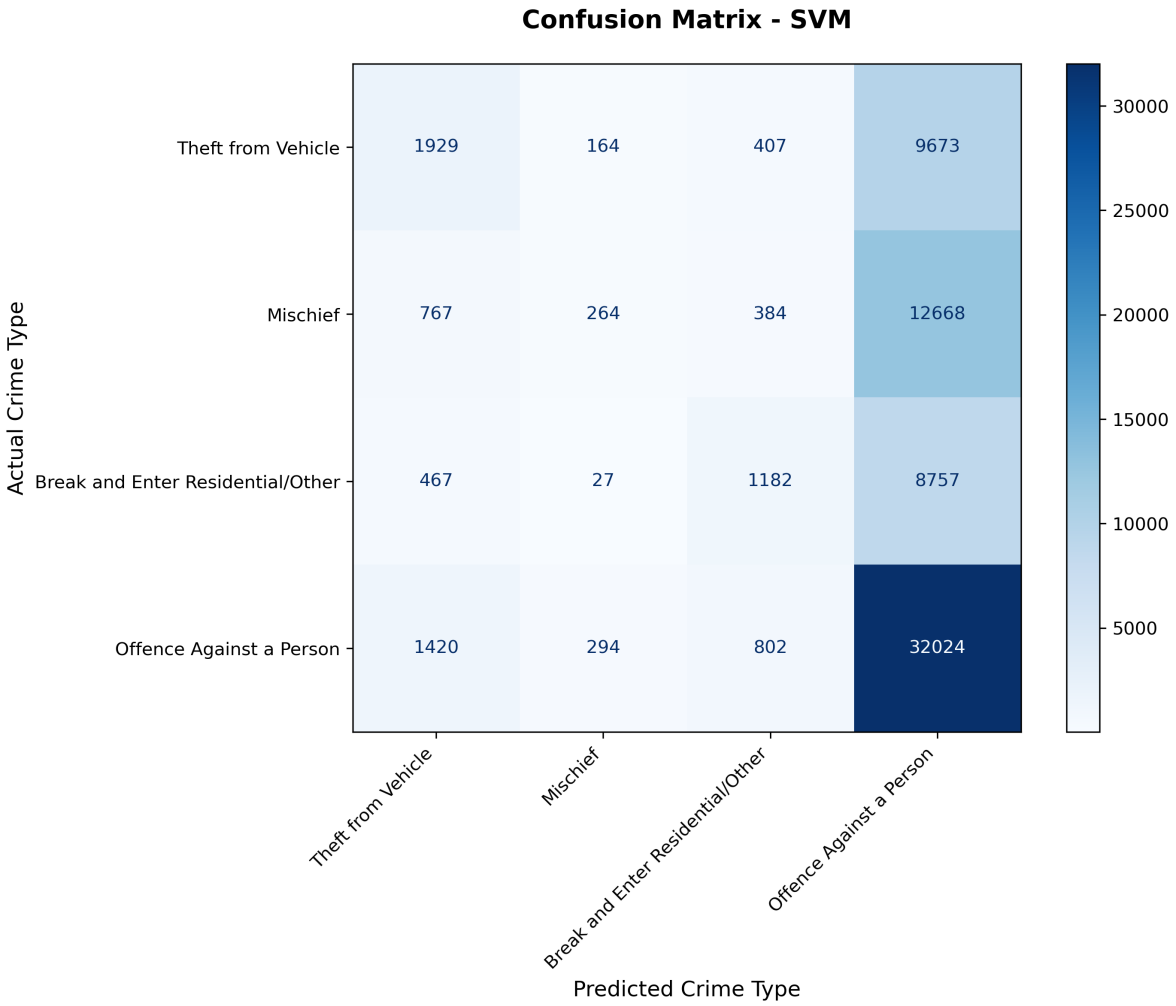

Figure 4: Confusion matrix for best fit SVM model, showing predicted crime type versus actual crime type.

Logistic Regression

The baseline logistic regression model achieved a test accuracy of 0.48. The best fit logistic regression model achieved a test accuracy of 0.48 and an F1 score of 0.32, indicating slightly lower performance than KNN and SVM. The confusion matrix for the best fit logistic regression model is shown in **?@fig-LR-cm** .

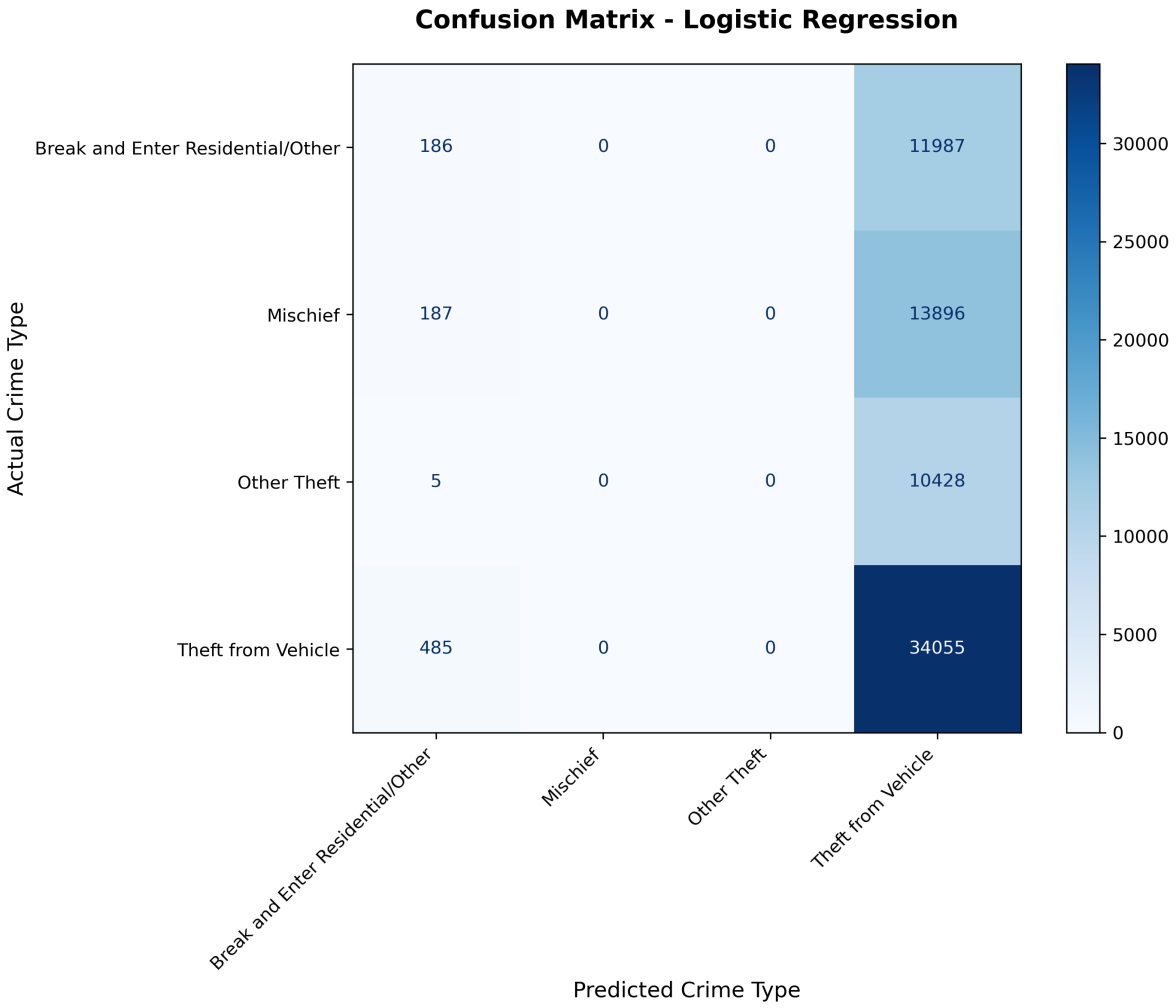

Figure 5: Confusion matrix for best fit Log Reg model, showing predicted crime type versus actual crime type.

Comparison

All three models achieved similar performance on the test set. The table below summarizes the accuracy and F1 scores for each model:

Model	Accuracy	F1 Score
Baseline K-NN (k=5)	0.44	0.44
Optimized K-NN (k=85)	0.51	0.45
Optimized SVM	0.5	0.39
Optimized Logistic Regression	0.48	0.32

The optimized K-NN model performed best overall, achieving the highest F1 score of 0.45. However, the differences between models were relatively small, with all achieving accuracies around 50%. The models performed well on common crimes like "Theft from Vehicle" but struggled with rarer or more ambiguous categories. We also observed that crime in Vancouver decreased from 2003 to 2011, then started increasing again after 2012.

Discussion

These results are somewhat expected given the nature of crime classification. Different crime types often occur in similar locations at similar times, making them difficult to distinguish based solely on temporal and spatial features. For example, various types of theft might all peak during nighttime hours in the same neighborhoods. Our models are limited to using only time and location information, without access to potentially important factors such as weather conditions, economic indicators, or other contextual variables that might influence crime patterns.

The fact that all three algorithms performed similarly suggests that the limiting factor is not the choice of algorithm, but rather the inherent predictability of crime types using only "when" and "where" information. This indicates that while these features provide some useful signal, they are insufficient on their own for highly accurate crime type prediction.

Despite the moderate accuracy of approximately 0.5, these models could still provide value for law enforcement planning. They can help identify which crime types are more predictable from spatiotemporal patterns, inform patrol allocation decisions, and highlight the need for additional features beyond basic time and location data to improve prediction accuracy.

Limitations and Assumptions

Our analysis has several important limitations. First, we only use time and location features, which means we're missing other information that could be important like weather, economic

conditions, or proximity to bars and schools. This limited feature set probably explains why our accuracy isn't higher.

Second, there's a class imbalance problem - some crime types are way more common than others. Even though we focused on the four most common types, the imbalance still exists and might make our models biased toward predicting the more frequent crimes. We also had missing data for about 10% of time values and 11% of neighborhood values, which we filled in with simple methods that might not perfectly represent what actually happened.

Third, our dataset only goes up to 2017, so our models might not work as well for current crime patterns, especially if things have changed in Vancouver over the past few years. Finally, we're assuming that the reported crimes accurately represent what actually occurred, but we know that reporting rates can vary by crime type and neighborhood, which could introduce bias into our models.

Future Work

Several avenues exist for improving upon this work. First, incorporating additional features could significantly enhance model performance. Weather conditions, day of week patterns, proximity to specific locations such as bars or schools, and economic indicators could all provide valuable predictive information. Second, exploring more sophisticated modeling approaches such as Random Forests or neural networks might better capture complex non-linear relationships in the data. Third, rather than predicting specific crime types, it may be more practical to predict crime severity levels (minor, moderate, or serious), which could reduce the classification complexity while still providing actionable insights. Fourth, updating the analysis with more recent data would be valuable, as this dataset ends in 2017 and crime patterns may have evolved. Finally, applying techniques specifically designed to handle class imbalance, such as SMOTE or class weighting, could improve performance on rarer crime types that are currently poorly predicted.

References

Costa-Luis, Casper da. 2019. "Tqdm – a Fast, Extensible Progress Bar for Python and CLI." https://github.com/tqdm/tqdm.

Harris, Charles R., K. Jarrod Millman, Stéfan J. van der Walt, Ralf Gommers, Pauli Virtanen, David Cournapeau, Eric Wieser, et al. 2020. "Array Programming with NumPy." *Nature*. https://doi.org/10.1038/s41586-020-2649-2.

Hunter, J. D. 2007. "Matplotlib: A 2D Graphics Environment." *Computing in Science & Engineering*. https://doi.org/10.1109/MCSE.2007.55.

Osaku, Wilian. 2017. "Crime in Vancouver." https://www.kaggle.com/datasets/wosaku/crime-in-vancouver.

Pedregosa, F., G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, M. Blondel, et al. 2011. "Scikit-Learn: Machine Learning in Python." *Journal of Machine Learning Research* 12: 2825–30.

Team, Pallets. 2020. *Click*. https://click.palletsprojects.com/.

team, The pandas development. 2024. "Pandas-Dev/Pandas: Pandas." Zenodo. https://doi.org/10.5281/zenodo.10537285.

Van Rossum, Guido, and Fred L. Drake. 2009. *Python 3 Reference Manual*. Scotts Valley, CA: CreateSpace.