Road Traffic Prediction and Analysis

Rakesh Muppala
(rmuppal)
Group P38
NC State University
rmuppal@ncsu.edu

Rishabh Agarwal
(ragarwa9)
Group P38
NC State University
ragarwa9@ncsu.edu

Shubham Bansal
(sbansal6)
Group P38
NC State University
sbansal6@ncsu.edu

Shubham Dua
(sdua2)
Group P38
NC State University
sdua2@ncsu.edu

1 Introduction and Background

Finding a way to dodge or avoid traffic is one of the most annoying things that we, as human beings, have to go through on a daily basis. The use of navigation apps to help find the fastest possible route has grown exponentially; almost 77% of smart-phone owners use navigation apps.

For this project we are going to find traffic trends and make predictions according to those trends for all the states across the United States. For our midterm report, we used 6904 different station ID's across the United States of America (according to their latitude and longitude coordinates), and predicted whether a particular road is a National Highway or not. We trained two simple classification models - SVM and KNN - on our data based on the hourly traffic volume at each station.

The raw dataset was preprocessed and its dimensionality was reduced by removing all the redundant attributes as well as all the attributes that do not have a meaningful impact on our predictions.

Once the data had been preprocessed, we used two classification models - SVM and KNN - to predict whether a given road is a National Highway or not.

2 Method

2.1 Technology Stack

2.1.1 Jupyter Notebooks

It is an open source web application, created by Fernando Pérez and Brian Granger. It is based on the server-client architecture. It provides an easy to use, interactive development environment (IDE).

2.1.2 Google Colab

Google Colab is a free cloud administration, in light of Jupyter Notebooks for AI learning and research. It extends a platform for runtime completely configured to deep learning and complimentary access to a robust GPU. The utility of Google Colab for this project:

- Free GPU support
- It provides common access to remote users and developers sharing Jupyter Notebooks and other files, likewise Google docs
- Major Python libraries, like TensorFlow, Scikit-Learn, Matplotlib, etc. are pre-installed
- It is developed on top of the Jupyter Notebook
- It allows training of deep-learning models free of cost from anywhere in the world

2.1.3 Programming Language (Python 3)

Python lets you integrate systems more effectively and work quickly. It is an interpreted, high-level, general purpose programming language. It supports various open source NLP and Machine Learning libraries. It has strong community support.

2.1.4 GitHub

GitHub is a web-based collaboration, version-control and virtual workspace platform for software developers. The underlying piece is Git, and it stores a project's source code, while keeping a track of all the changes. It helps avoid contradictory changes, by providing tools to developers collaborating on a project.

2.1.5 Python Libraries and Modules

Numpy: Numpy is a python programming library that helps handle huge dimensions of data, in the form of both arrays and matrices. It also supports many high-level statistical and mathematical functions to operate on these data forms.

Pandas: Pandas is a python programming library used for data analysis and manipulation. It also supports manipulating numerical tables and time series.

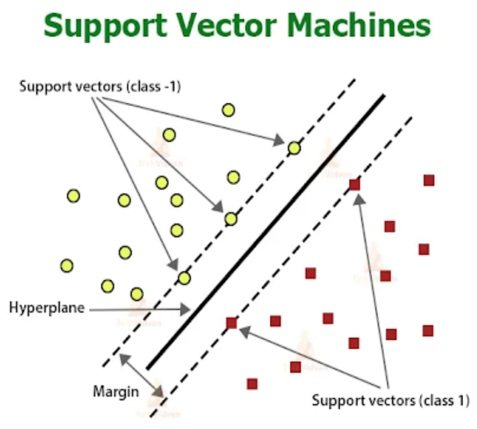
Matplotlib: Matplotlib is a python programming library that is used to plot numerical data for both python and numpy.

Sklearn: Scikit-Learn (also known as sklearn) is a python programming library and it is used for various regression, classification and clustering algorithms.

2.2 Literature Review

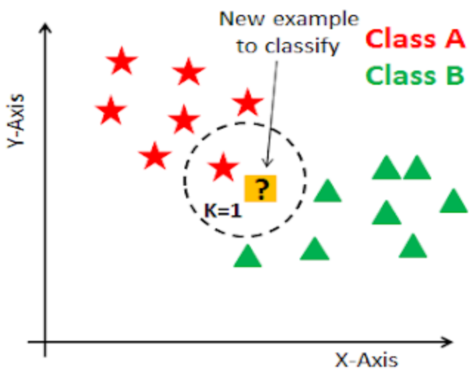
2.2.1 Support Vector Machines (SVM)

SVM's are a set of supervised learning algorithms; they are very widely used because they can be used for classification tasks, regression tasks, or even outlier detection. For binary classification, SVM's draw a decision boundary (called hyperplane) between the two classes in order to classify them.



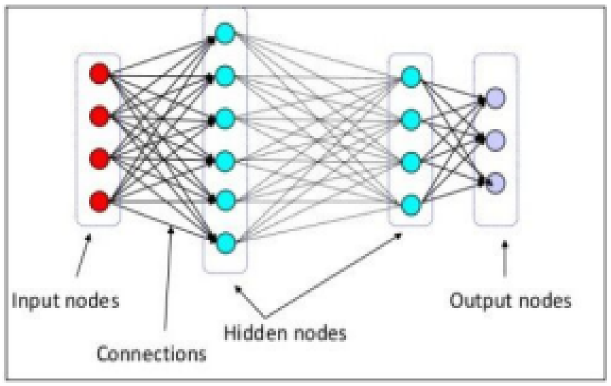
2.2.2 K-Nearest Neighbors (KNN)

KNN is a supervised learning algorithm. It is a very simple learning algorithm and is sometimes also called a lazy algorithm. To implement a KNN model, we calculate the distance between the point whose class we want to find with the rest of the data points. The KNN model makes the classification based on the K-nearest neighbors of that particular data point.



2.2.3 Artificial Neural Networks (ANN)

An Artificial Neural Network (also called Neural Network) is a supervised machine learning algorithm that is loosely based on the biological neural networks of human beings. ANNs have various nodes called artificial neurons that are based on the concept of the biological neurons present inside the human brain. Like the human brain has synapses, artificial neurons can also transmit or receive information from other neurons that are connected to it. The connections between these artificial neurons are known as edges and the importance of these connections are based on some weight given to these edges. As the Artificial Neural Network continues learning, it keeps on adjusting the weights along its edges. Generally neurons are branched together at different layers - the more number of neurons there are, the deeper is the Neural Network (that is, the ANN will have more number of layers of neurons). And deeper is the ANN, the more optimal result it will give after training. The main drawback of having a deep neural network, however, is that it takes a lot of time to train.



3 Experiment Setup

For this project, we took data from https://www.kaggle.com/jboysen/us-traffic-2015. The original data was in the form of two datasets; the first dataset gave information about the hourly traffic volume, the date of the recorded entry, the direction of travel (North, South, East or West) etc. for a given station ID and the second dataset gave information about the particular station ID's.

The first dataset has 38 columns - 38 input attributes for our binary classification task. We used data preprocessing to reduce the dimensionality of the dataset by dropping 11 attributes from the first dataset (that were not affecting our predictions much). These are the 11 attributes dropped after preprocessing - the day the data was recorded, the functional classification name, the direction of travel (North, South, East, West), the record type, the restrictions, the year, date, and month the data was recorded, the day of the week, the direction of travel name, and the lane of travel.

For the next step, we grouped the data together based on three attributes - the name of the state, the station ID, and the functional classification. The 'functional classification' attribute gives information about the type of the road, whether it is a rural road, an urban road, a state highway, an inter-state highway etc. The remaining 24 input attributes are the traffic volume attributes (one for each hour of the day). After grouping the data according to the three attributes above, we took the mean of the hourly traffic volume attributes to maintain the homogeneity of the dataset.

We were only interested in five attributes from the second dataset - the name of the state, the station ID, the latitude and longitude for that station ID, and whether the given road is a highway or not. We merged the two datasets using an inner join on two attributes - the name of the state and the station ID. The attribute to check if a given road is a highway or not is the output attribute (the attribute to be predicted) for the binary classification task. The aim to combine the two datasets was just to map each station ID with its corresponding latitude and longitude coordinates. After removing all the duplicate entries, we have 6904 records as the data we gave to train our models. The input data (now consisting of 27 attributes) was normalized to transform the data points between the values of 0 and 1.

After preprocessing, we used two simple binary classification models - SVM and KNN - to calculate the accuracy of both the models.

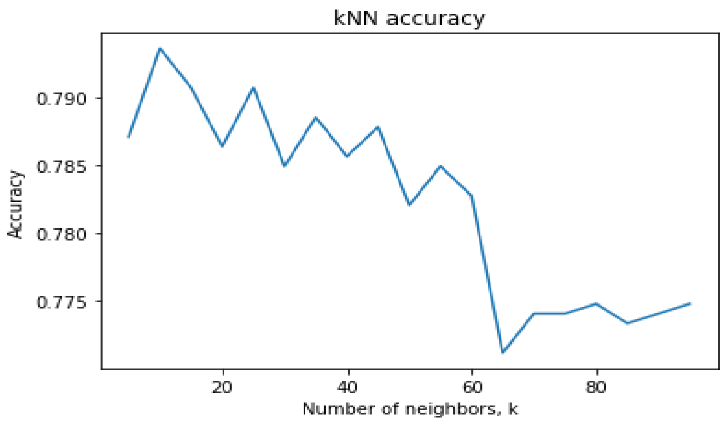
For now, we split the data into 80 percent training data and 20 percent test data. Our next step is to include a validation set to increase the accuracy of both these classification models. So we will split the data as 70 percent training data, 15 percent validation data and 15 percent test data. Then we will also try to use Cross-Validation to train our models even better. So we will compare the accuracy of the two models before using a validation set, after using a validation set (15 percent of the original data), and after applying cross-validation to the dataset.

Once we complete the binary classification task (using all the three splitting techniques), we will use a multi-input multi-output regression model to predict the hourly traffic volume given the latitude and longitude coordinates (that map to a station ID) among other features.

4 Results

We split the data into training and testing sets - 80

For the SVM model, we got an accuracy of 0.777. For the KNN model, we found the nearest neighbors using Euclidean distance. We calculated the KNN model for K=5 up to K=100, increasing K by 5 after every iteration. We plotted a graph below showing the accuracy vs the value of K.



5 Conclusion

There are two problem statements that we will try to complete by the end of this project.

- The first task is to implement binary classification using SVM and KNN. We split the data as 80 percent training data and 20 percent test data to get the above results. To improve the accuracy of the models, we will use a validation set (15 percent of the original data) and also use cross-validation to train the SVM and KNN models.
- The second task will be to implement regression to predict the hourly traffic volume from our data. For this, we will have to preprocess the original dataset once again to get good results.

6 References

1. Pan, B., Demiryurek, U. and Shahabi, C., 2012, December. Utilizing real-world transportation data for accurate traffic prediction. In 2012 IEEE 12th International Conference on Data Mining (pp. 595-604). IEEE.
2. Min, W. and Wynter, L., 2011. Real-time road traffic prediction with spatio-temporal correlations. Transportation Research Part C: Emerging Technologies, 19(4), pp.606-616.
3. Ishak, S. and Al-Deek, H., 2002. Performance evaluation of short-term time-series traffic prediction model. Journal of transportation engineering, 128(6), pp.490-498.

7 Appendix

7.0.1 Old Problem Statement

- Predicting traffic volume at a station for a particular date-time. Visualizing hourly traffic volume at a station/route/region for different days of the week and calculating peak congestion hours for different regions.
- Analysing traffic trends and discovering clusters of stations having similar traffic volume at a time, and discovering routes having high traffic congestion.

7.0.2 New Problem Statement

- Implement binary classification model to predict whether a given road is a National Highway or not.
- Implement regression model to find out the hourly traffic volume at any given latitude and longitude coordinates.