**TrafficTelligence**

Advanced Traffic Volume Estimation with Machine Learning

**INTRODUCTION**

**Project Title:** TrafficTelligence: Advanced Traffic Volume Estimation with Machine Learning

**Team Size: 4**

**Team Leader :** Gona Pranitha

**Team** **member :**  Nagaraju Gari Susmitha

**Team** **member :** Yarragunta Likitha

**Team** **member :** RAnusha

**Description:**

TrafficTelligence is an advanced system that uses machine learning algorithms to estimate and predict traffic volume with precision. By analyzing historical traffic data, weather patterns, events, and other relevant factors, TrafficTelligence provides accurate forecasts and insights to enhance traffic management, urban planning, and commuter experiences.

**Scenario 1: Dynamic Traffic Management**

TrafficTelligence enables dynamic traffic management by providing real-time traffic volume estimations. Transportation authorities can use this information to implement adaptive traffic control systems, adjust signal timings, and optimize lane configurations to reduce congestion and improve traffic flow.

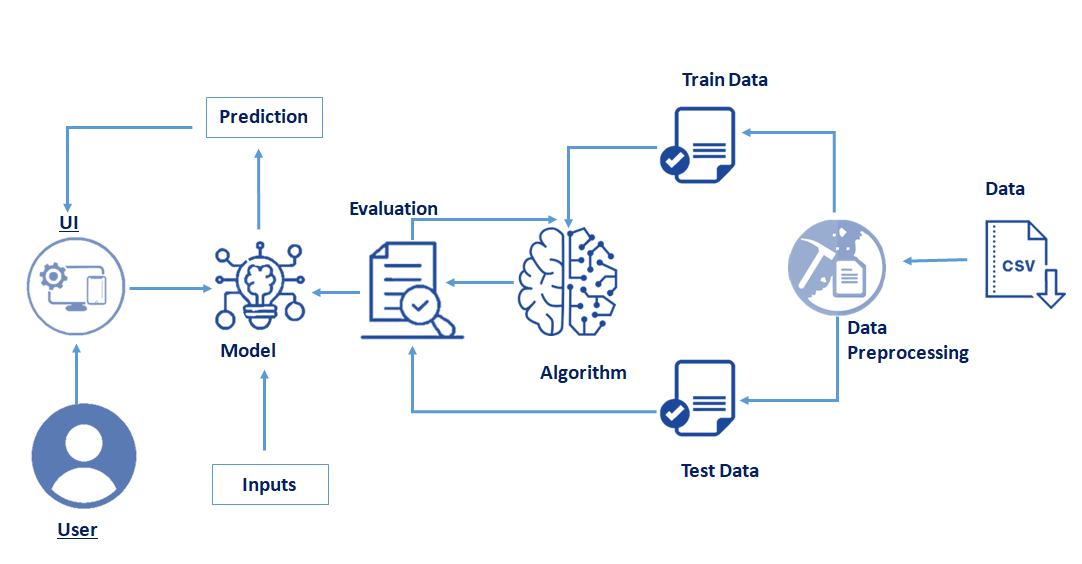
**Scenario 2: Urban Development Planning**

City planners and urban developers can leverage TrafficTelligence predictions to plan new infrastructure projects effectively. By understanding future traffic volumes, they can design road networks, public transit systems, and commercial zones that are optimized for traffic efficiency and accessibility.

**Scenario 3: Commuter Guidance and Navigation**

Individual commuters and navigation apps can benefit from TrafficTelligence's accurate traffic volume estimations. Commuters can plan their routes intelligently, avoiding congested areas and selecting optimal travel times based on predicted traffic conditions. Navigation apps can provide real-time updates and alternative routes to improve overall travel experiences.

**TECHNICAL ARCHITECTURE:**

****

**PREREQUISITES:**

To complete this project, you must require the following software’s,  concepts, and packages

**Python packages:**

Open anaconda prompt as administrator.

* Type “pip install numpy” and click enter.
* Type “pip install pandas” and click enter.
* Type “pip install matplotlib” and click enter.
* Type “pip install scikit-learn” and click enter.
* Type “pip install Flask” and click enter.
* Type “pip install xgboost” and click enter.

**Prior Knowledge:**

Machine Learning concepts:

* Supervised learning
* Unsupervised learning
* Metrics

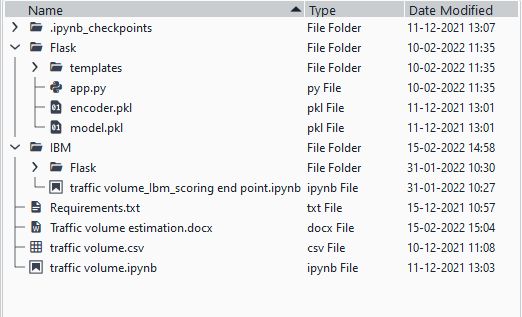
Flask Basics:

**Project Objectives :**

By the end of this project:

* You’ll be able to understand the problem to classify if it is a regression or a classification kind of problem.
* You will be able to know how to pre-process/clean the data using different data pre-processing techniques.
* You will able to analyze or get insights into data through visualization.
* Applying different algorithms according to a dataset and based on visualization.
* You will be able to know how to find the accuracy of the model.
* You will be able to know how to build a web application using the Flask framework.

**Project Structure:**



Create a Project folder that contains files as shown below

* Flask files consist of template folder which has HTML pages, app.py file and .pkl files which are used for application building
* IBM folder has flask files and scoring endpoint.ipynb- model training code file.
* We need the model which is saved and the saved model in this content is Traffic volume. Pkl
* Templates folder which contains index.HTML file, chance.HTML file, noChance.HTML file.
* Scale.pkl for scaling, encoder.pkl file for encoding the categorical data, imputer.pkl file for filling out the missing values

**Project Flow :**

* User interacts with the UI (User Interface) to enter the input values.
* Entered input values are analyzed by the model which is integrated.
* Once the model analyses the input the prediction is showcased on the UI.

To accomplish this, we have to complete all the activities and tasks listed below

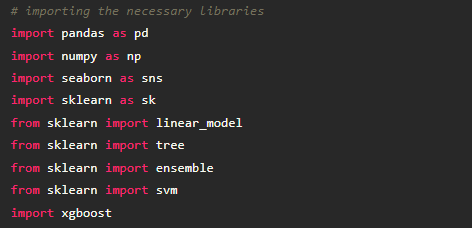
* **Data Collection.**
  + Collect the dataset or Create the dataset
* **Data Pre-processing.**
  + Import the Libraries.
  + Importing the dataset.
  + Checking for Null Values.
  + Data Visualization.
  + Taking care of Missing Data.
  + Feature Scaling.
  + Splitting Data into Train and Test.
* **Model Building**
  + Import the model building Libraries
  + Initializing the model
  + Training and testing the model
  + Evaluation of Model
  + Save the Model
* **Application Building**
  + Create an HTML file
  + Build a Python Code
  + Run the App

**Data Collection:**

ML depends heavily on data, without data, it is impossible for an “AI” model to learn. It is the most crucial aspect that makes algorithm training possible. In Machine Learning projects, we need a training data set. It is the actual data set used to train the model for performing various actions.

**Data Preprocessing:**

* **Import Necessary Libraries:**
* It is important to import all the necessary libraries such as pandas, NumPy, matplotlib.
* **Numpy**I-t is an open-source numerical Python library. It contains a multi-dimensional array and matrix data structures. It can be used to perform mathematical operations on arrays such as trigonometric, statistical, and algebraic routines.
* **Pandas**- It is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language.
* **Seaborn**- Seaborn is a Python data visualization library based on matplotlib. It provides a high-level interface for drawing attractive and informative statistical graphics.
* **Matplotlib**- Visualisation with python. It is a comprehensive library for creating static,animated, and interactive visualizations in Python
* **Sklearn** – which contains all the modules required for model building.

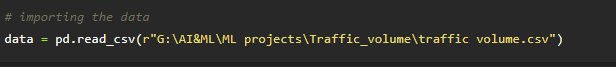


* **Importing the Dataset:**

You might have your data in .csv files, .excel files

Let’s load a .csv data file into pandas using read\_csv() function.We will need to locate the directory of the CSV file at first (it’s more efficient to keep the dataset in the same directory as your program).

If your dataset is in some other location, Then



If the dataset is in the same directory of your program, you can directly read it, without giving raw as r.

Our Dataset weatherAus.csv contains the following Columns

Holiday - working day or holiday

Temp- temperature of the day

Rain and snow – whether it is raining or snowing on that day or not

Weather = describes the weather conditions of the day

Date and time = represents the exact date and time of the day

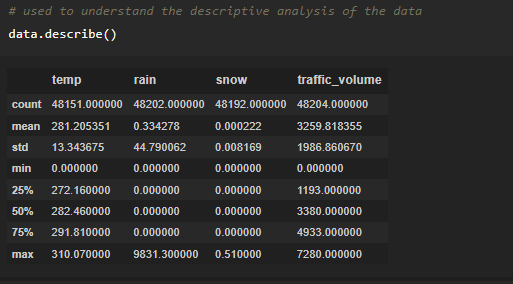
Traffic volume – output column

The output column to be predicted is Traffic volume.Based on the input variables we predict the volume of the traffic. The predicted output gives them a fair idea of the count of traffic

* **Analyse the Data:**
* **head()** method is used to return top n (5 by default) rows of a DataFrame or series.

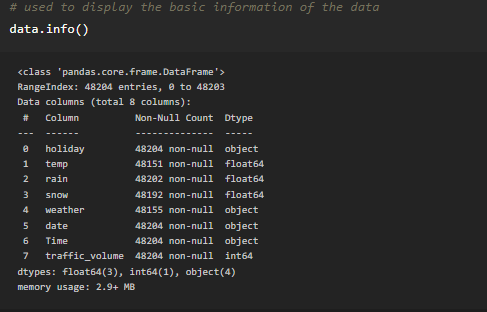


* **describe()** method computes a summary of statistics like count, mean, standard deviation, min, max, and quartile values.



From the data, we infer that there are only decimal values and no categorical values.

* **info()** gives information about the data - paste the image here.

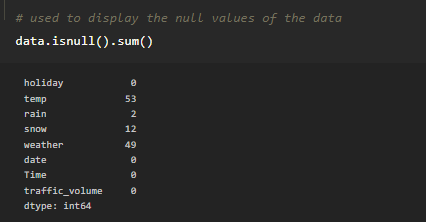


* **Handling Missing Values:**

1. The Most important step in data pre-processing is dealing with missing data, the presence of missing data in the dataset can lead to low accuracy.

2. Check whether any null values are there or not. if it is present then the following can be done.

There are missing values in the dataset, we will fill the missing values in the columns.



3. We are using mean and mode methods for filling the missing values

Columns such as temp, rain, and snow are the numeric columns, when there is a numeric column you should fill the missing values with the mean/median method. so here we are using the mean method to fill the missing values.

Weather column has a categorical data type, in such case missing data needs to be filled with the most repeated/ frequent value. Clouds are the most repeated value in the column, so imputing with clouds value.

* **Data Visualization:**

Before diving into the code, let's look at some of the basic properties we will be using when plotting.

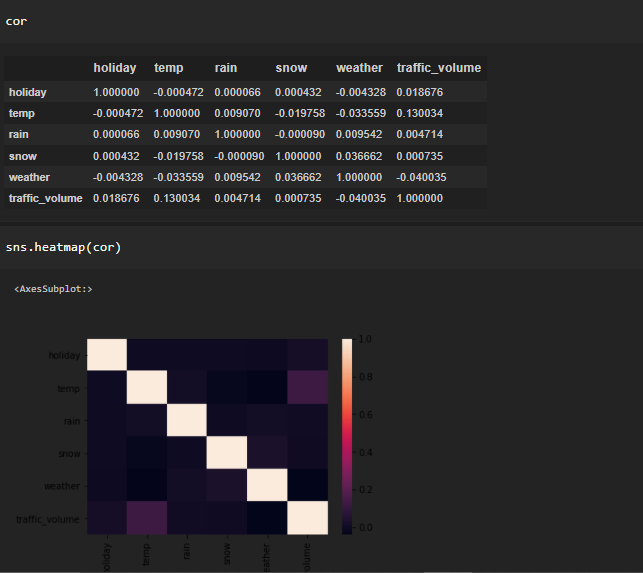
**xlabel:** Set the label for the x-axis.

**ylabel:** Set the label for the y-axis.

**title:**Set a title for the axes.

**Legend:** Place a legend on the axes.

* 1. **data.corr()** gives the correlation between the columns

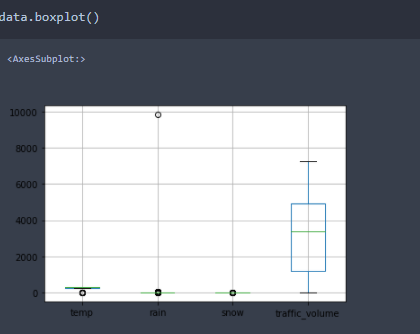


**2.Code:**

Sns.pairplot(data)

****

**3.Box Plot:**

****

* **Splitting the Dataset Into Dependent And Independent Variable:**

With this in mind, we need to split our dataset into the matrix of independent variables and the vector or dependent variable. Mathematically, Vector is defined as a matrix that has just one column.

To read the columns, we will use iloc of pandas (used to fix the indexes for selection) which takes two parameters — [row selection, column selection].

* **Feature Scaling:**
* There is a huge disparity between the x values so let us use feature scaling.
* Feature scaling is a method used to normalize the range of independent variables or features of data.
* After scaling the data will be converted into an array form
* Loading the feature names before scaling and converting them back to data frame after standard scaling is applied
* **Splitting The Data into Train and Test:**
* The train-test split is a technique for evaluating the performance of a machine learning algorithm.
* Train Dataset: Used to fit the machine learning model.
* Test Dataset: Used to evaluate the fit machine learning model.
* In general you can allocate 80% of the dataset to the training set and the remaining 20% to test.
* Now split our dataset into train set and test using train\_test\_split class from sci-kit learn library.
* from sklearn import model\_selection
* x\_train,x\_test,y\_train,y\_test=model\_selection.train\_test\_split(x,y,test\_size=0.2,random\_state =0)

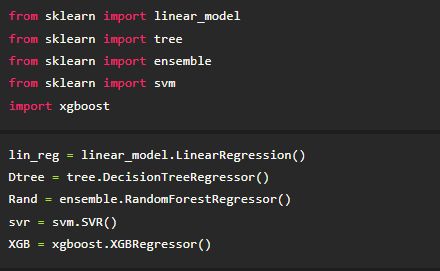
**Model Building:**

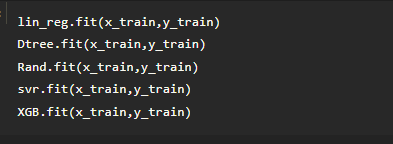
* **Training and Testing the Model:**

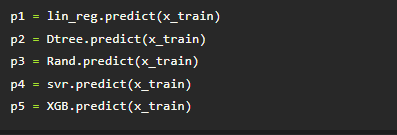
Once after splitting the data into train and test, the data should be fed to an algorithm to build a model.

There are several Machine learning algorithms to be used depending on the data you are going to process such as images, sound, text, and numerical values. The algorithms that you can choose according to the objective that you might have it may be Classification algorithms are Regression algorithms.  
1. Linear Regression  
2. Decision Tree Regressor  
3.Random Forest Regressor  
4.KNN  
5.svm  
6.xgboost

**Initialize the model -**



**Fit the models with x\_train and y\_train -**

**Predict the y\_train values and calculate the accuracy -**  
****

We’re going to use the x-train and y-train obtained above in the train\_test\_split section to train our Random forest regression model. We’re using the fit method and passing the parameters as shown below.

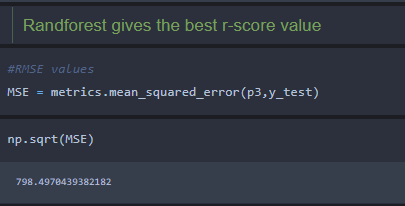
 We are using the algorithm from Scikit learn library to build the mode

* **Model Evaluation:**

**Regression Evaluation Metrics:**

These model evaluation techniques are used to find out the accuracy of models built in the Regression type of machine learning models. We have three types of evaluation methods.

* R-square\_score
* RMSE – root mean squared error



* **Save the Model:**

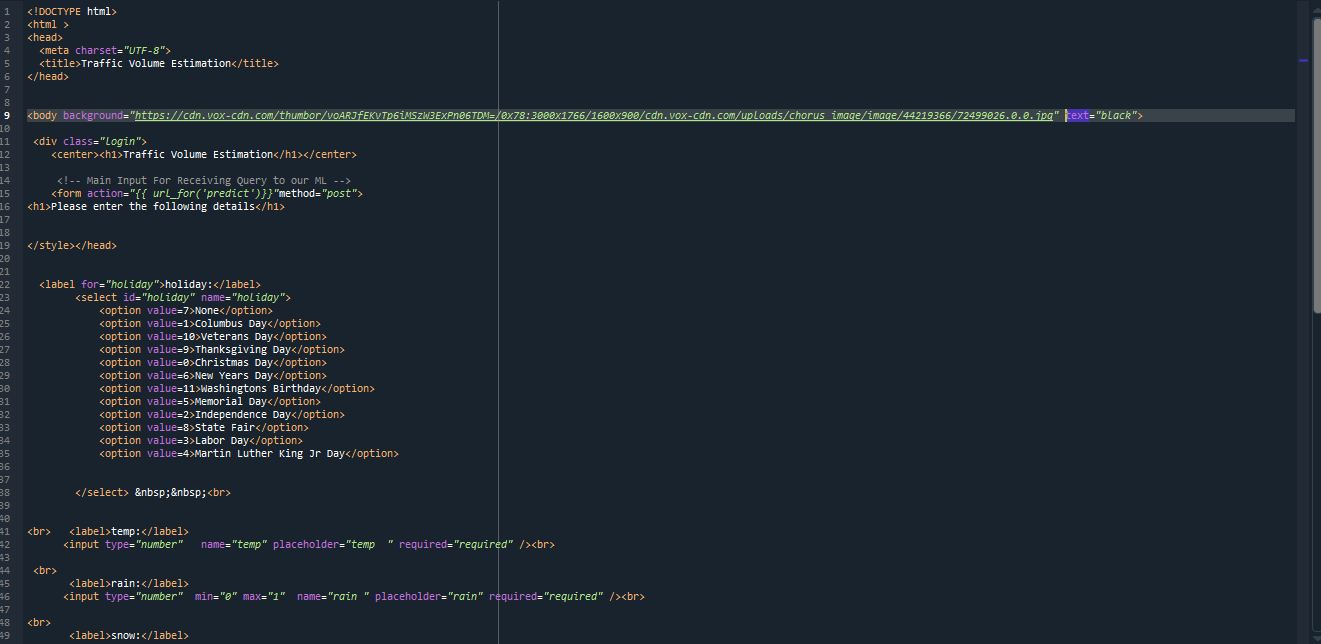
After building the model we have to save the model.

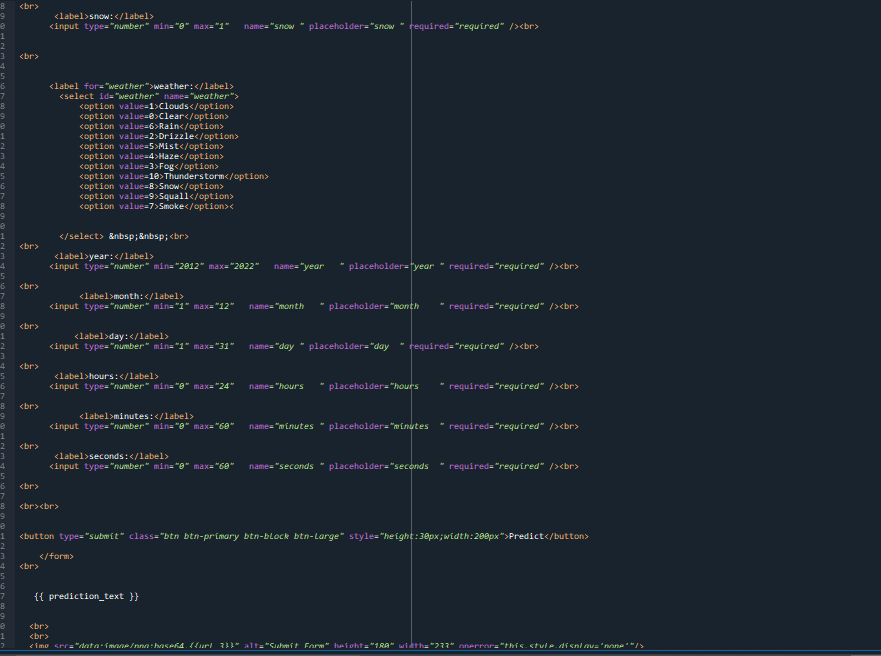
Pickle in Python is primarily used in serializing and deserializing a Python object structure. In other words, it's the process of converting a Python object into a byte stream to store it in a file/database, maintain program state across sessions or transport data over the network. wb indicates write method and rd indicates read method.

**Application Building**

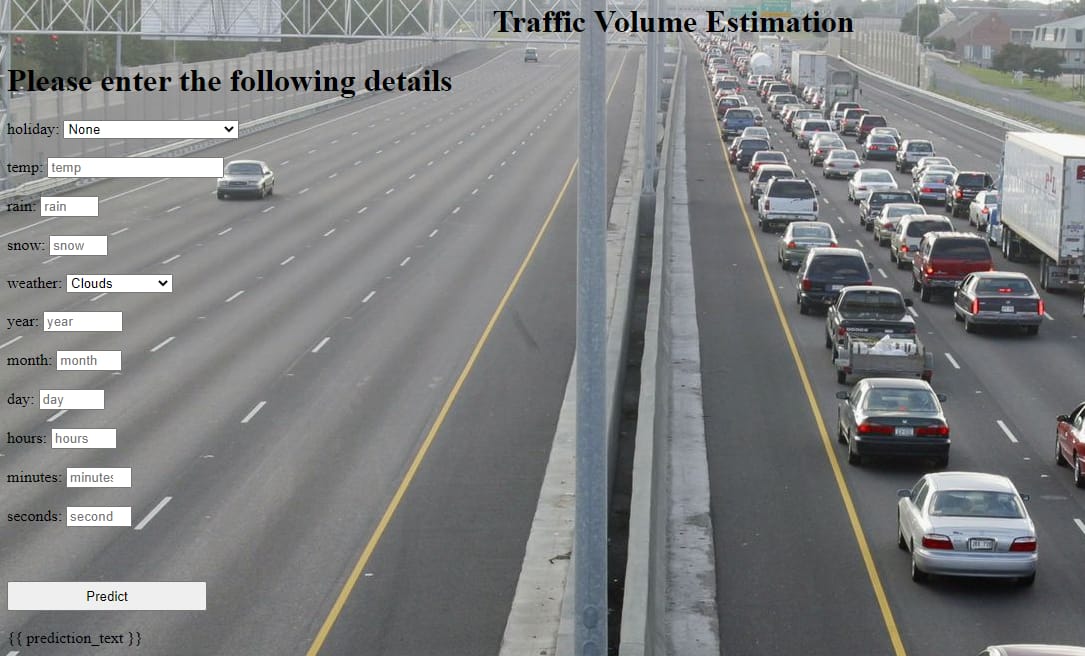
* **Build HTML Code**

**1.index.html - paste the image**





* 1. **The HTML page looks like this-**



**3.It will display all the input parameters and the prediction text will display the output value of the data given by the user.**



* **Main Python Script**
* **Run the App**

Open anaconda prompt  from the start menu

* Navigate to the folder where your python script is.
* Now type the “python app.py” command

Navigate to the localhost where you can view your web page, Then it will run on **local host:5000**

* **Output**
* Copy the HTTP link and paste it in google link tab, it will display the form page
* Enter the values as per the form and click on predict button
* It will redirect to the page based on prediction output
* The output will be displayed in the prediction text as Estimated Traffic volume is in units.



**The Demo of the App Available at:**

"C:\Users\LENOVO\Desktop\likki.mp4"

