**TRAFFIC VOLUME ESTIMATION USING IBM WATSON MACHINE LEARNING**

Developed By : B. Rajashekar, P. Nuthan Kumar, P. Swathi, K. Nishitha

**Smart Bridge – Mini Project Report**

**1.INTRODUCTION**

**1.1 Overview**

Growth in the number of vehicles and degree of urbanization means that the annual cost of traffic jams is increasing in cities. This leads to a decrease in the quality of life among citizens through a considerable waste of time and excessive fuel consumption and air pollution in congested areas Traffic congestion has been one of the major issues that most metropolises are facing despite measures being taken to mitigate and reduce it. The safe and time-efficient movement of the people and goods is dependent on Traffic flow, which is directly connected to the traffic characteristics. Early analysis of congestion events and prediction of traffic volumes is a crucial step to identify traffic bottlenecks, which can be utilized to assist traffic management centres.

**1.2 Purpose**

To overcome the problem of traffic congestion, the traffic prediction using machine learning which contains regression model and libraries like pandas, os, numpy, matplotlib.pyplot are used to predict the traffic. It is implemented so that the traffic congestion is controlled and can be accessed easily. Users can collect the traffic information of the traffic flow and can also check the congestion flow from the start of the day till the end of the day with the time span of one hour data.

**2.LITERATURE SURVEY**

**2.1Existing problem and Existing approaches or methods**

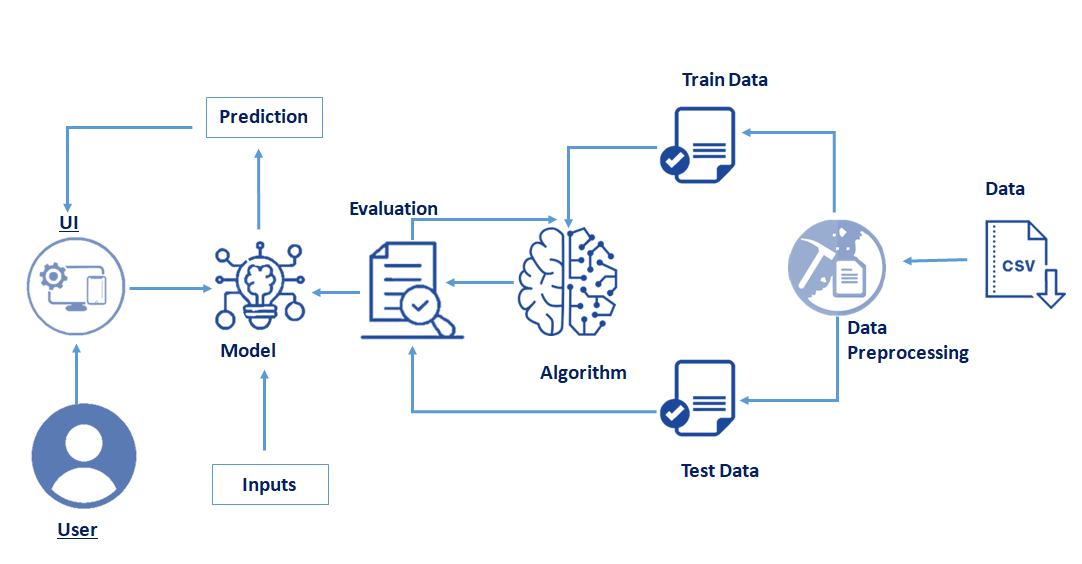
Traffic Volume Count can be done by various methods depending upon various factors like manpower available, budget, technology/instrument available, magnitude of traffic data required or to be collected which will then determine quality and type of vehicle classification to be adopted. Traffic counting falls in two main categories, namely: manual count and automatic count.Traffic data collection forms the integral part of traffic volume study as it provides the raw data and includes primary survey. The various types and methods used to collect traffic data not only provide a good and valuable coverage of the required traffic information.

**2.2 Proposed solution**

In this project we have shown use of Regression algorithms such as Linear Regression, Decision tree, Random forest, and xgboost to predict the count of traffic volume. We will train and test the data with these algorithms. From this best model is selected and saved in .pkl (Pickle) format. Once the model is saved, we integrate it with flask application and also deploy the model in IBM.

**3.THEORITICAL ANALYSIS**

**3.1 Block Diagram**



**3.2 Hardware and Software requirements**

Software Requirements:

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

Anaconda navigator

Python packages:

* numpy
* pandas.
* matplotlib.
* scikit-learn
* xgboost

Flask

Hardware Requirements

* Processor : Intel Core i3
* Hard Disk Space : Min 100 GB
* Ram : 4 GB
* Display : 14.1 “Color Monitor(LCD, CRT or LED)
* Clock Speed : 1.67 GHz

**4.EXPERIMENTAL INVESTIGATION**

For developing the project the team has completed several tasks:

**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

**5.FLOW CHAT**

Importing the necessary libraries

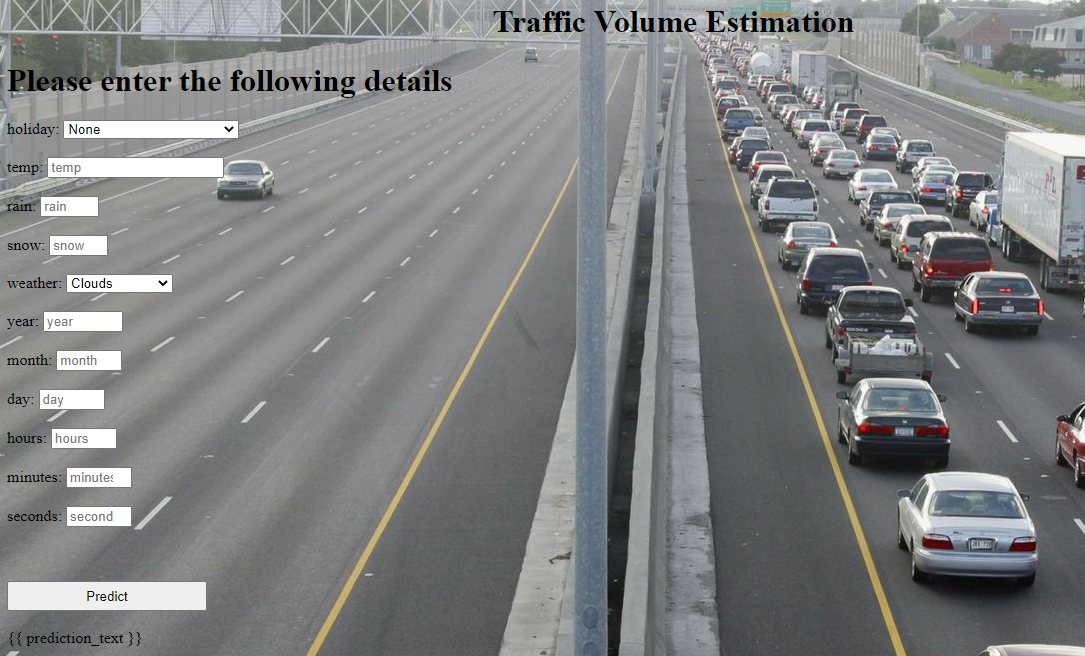
Loading the dataset

Training the model

Testing the model

**6. RESULTS**

This is the HTML page where the user has to enter the details:



Final output of the project:



**7.ADVANTAGES AND DISADVANTAGES**

**Advantages:**

* Time complexity has been reduced
* Suitable for capturing the underlying relationships among different variables in an environment of uncertainty

**Disadvantages:**

* Interpretability of input variables (“black box”)
* Only predict within bounds of training – no extrapolation

**8.APPLICATIONS**

* We can estimate the volume of traffic by using machine learning techniques, which

will cut down on the amount of time we spend in traffic.

* Smart cities can use this techniques.
* Structural design of roads.
* Planning and design of new streets.
* Planning, design and regulation for traffic.
* Establish properties and schedules for traffic improvements.

**9.CONCLUSION**

In the system, it has been concluded that we develop the traffic flow estimation system by using a machine learning algorithm and trained the model on IBM watson. By using regression model, the prediction is done. The public gets the benefits such as the current situation of the traffic flow, they can also check what will be the flow of traffic on the right after one hour of the situation. The weather conditions have been changing from years to years. The cost of fuel is also playing a major role in the transportation system. The forecasting or the prediction can help people or the users in judging the road traffic easier beforehand and even they can decide which way to go using their navigator and also this will prediction will be also helpful.

**10.FUTURE SCOPE**

These days, traffic prediction is extremely necessary for every part of the state and also worldwide. So, this method of prediction would be helpful in predicting the traffic earlier and decreasing the accidents which saves many lives. And it will also reduce the waste of time and excessive fuel consumption and air pollution in congested areas. For better congestion prediction, the grade and accuracy are prominent in traffic prediction. Within the future, the expectation are going to be the estimation of established order accuracy prediction with much easier and user-friendly methods. So people would find the prediction model useful and that they won’t be wasting their time and energy to predict the information.

**11. BIBILOGRAPHY**

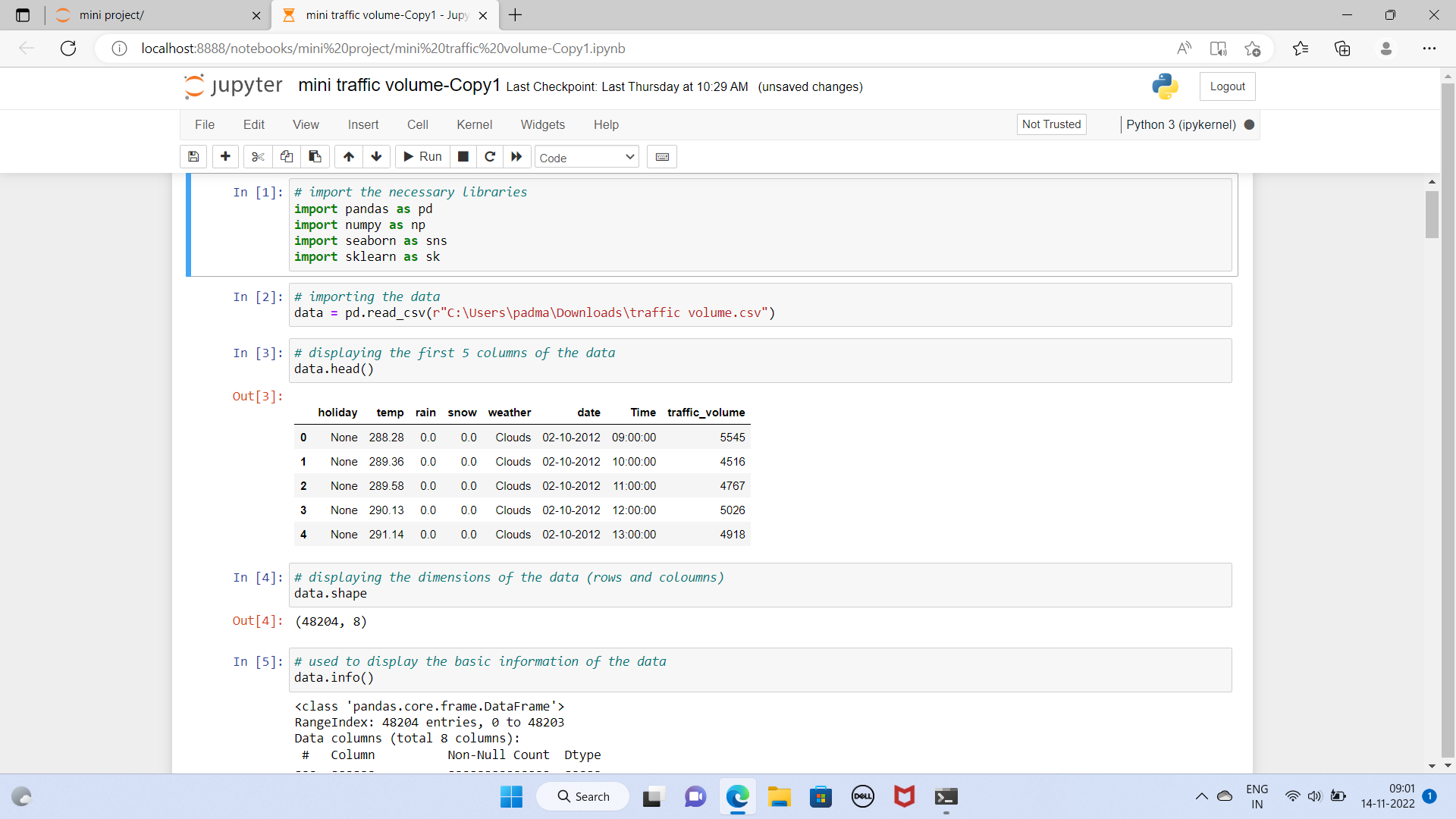
<https://www.researchgate.net/publication/332407206_Traffic_Prediction_Using_Machine_Learning>.

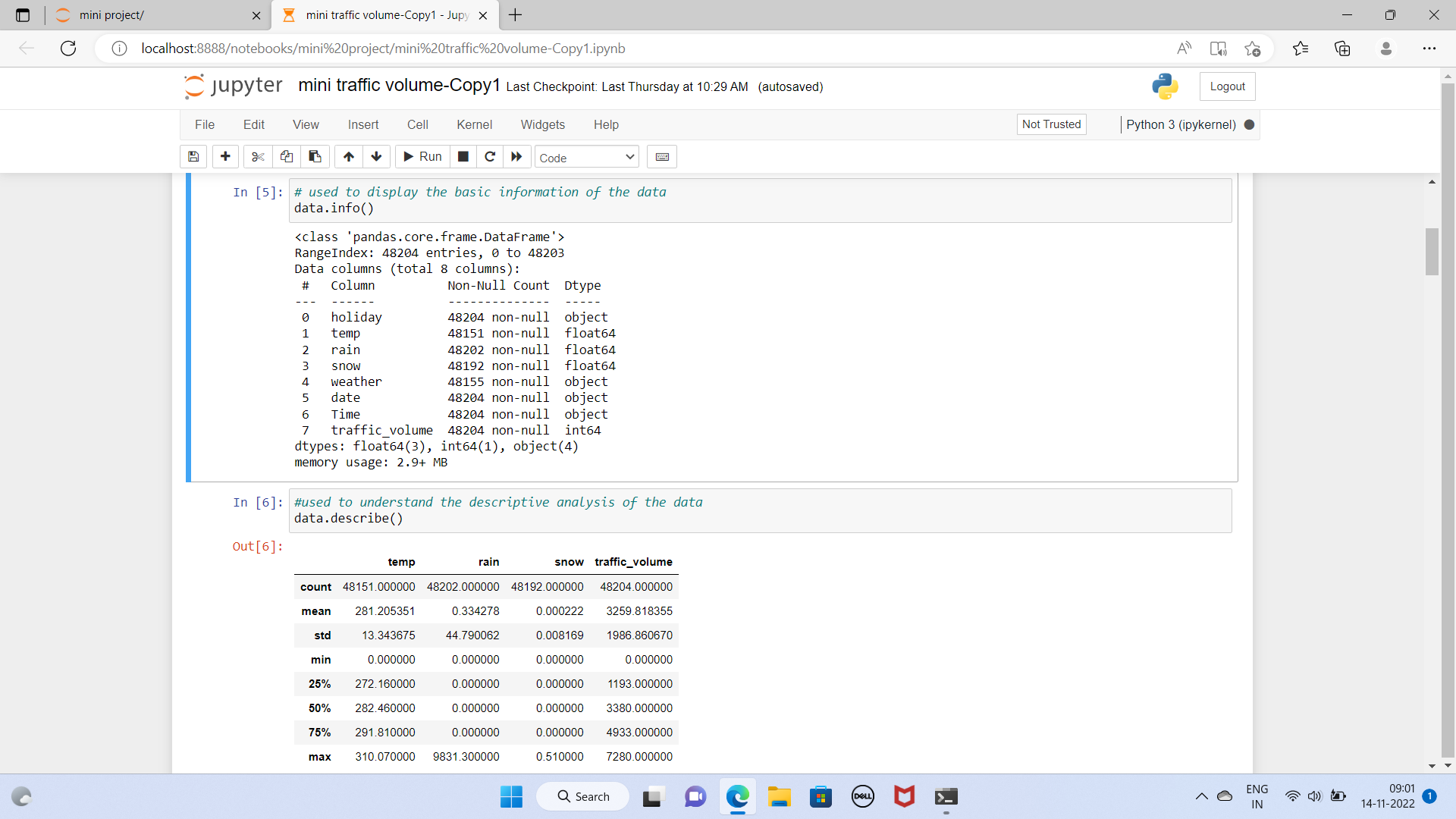
<https://www.ibm.com/docs/en/watson-studio-local/1.2.3?topic=model>.

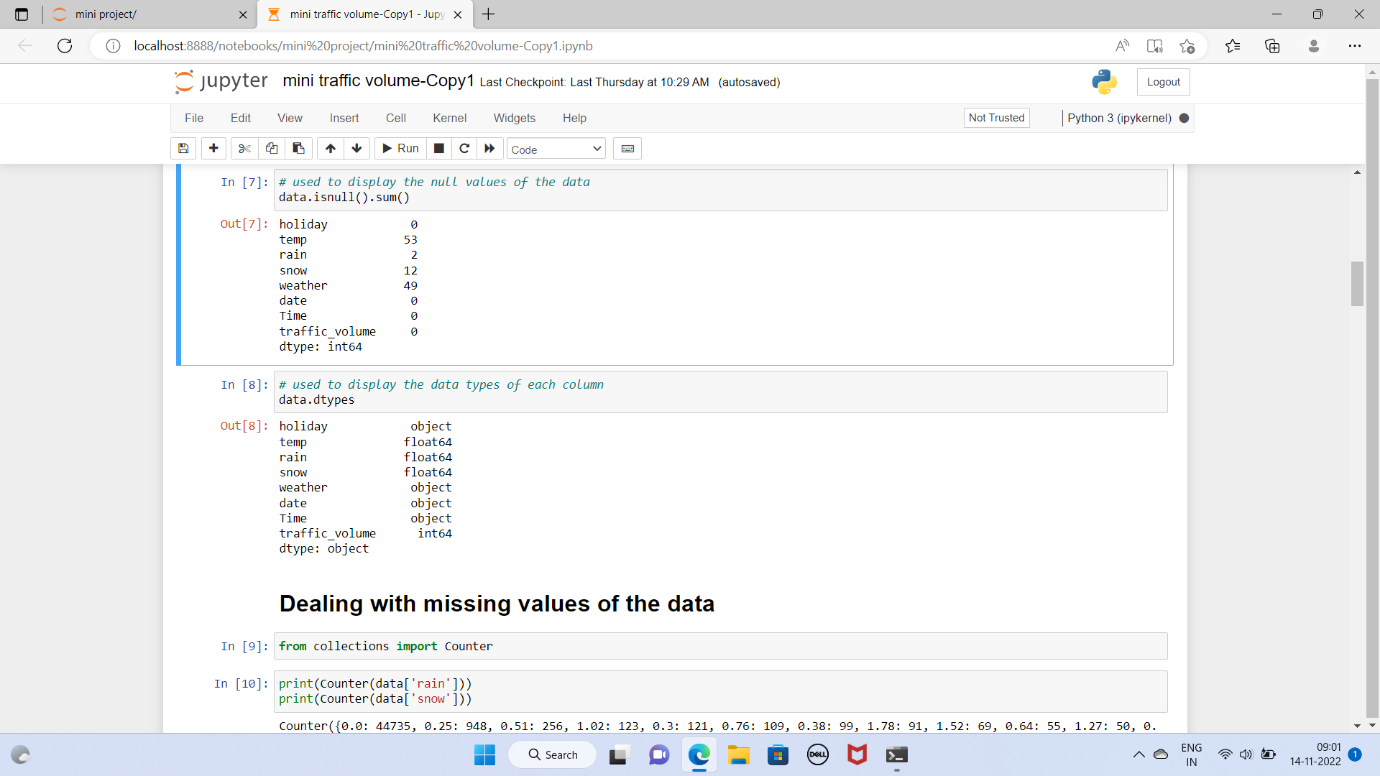
[www.engineeringenotes.com/transportation-engineering/traffic-engineering/traffic\_engineering](http://www.engineeringenotes.com/transportation-engineering/traffic-engineering/traffic_engineering).

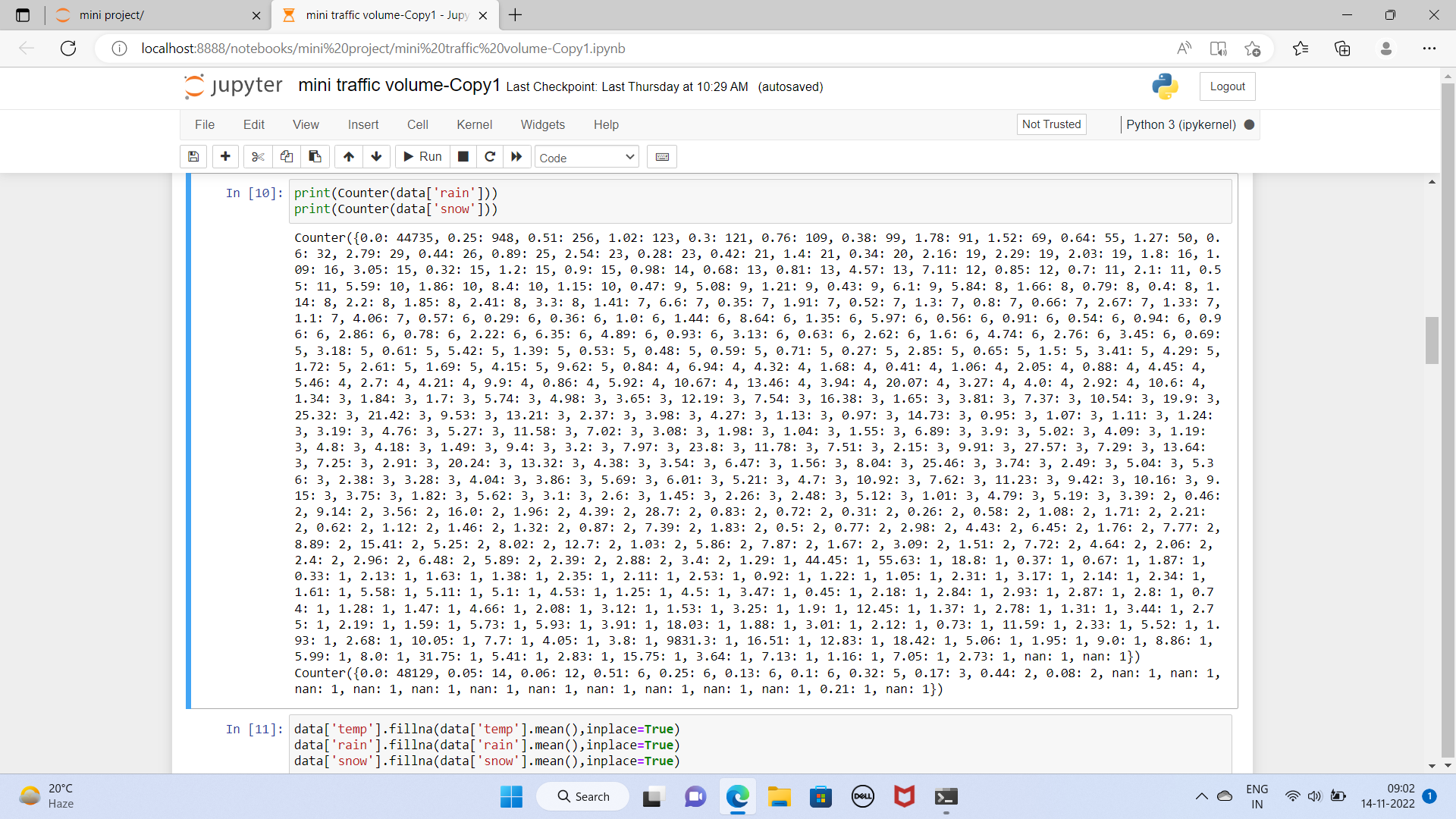
**APPENDIX**

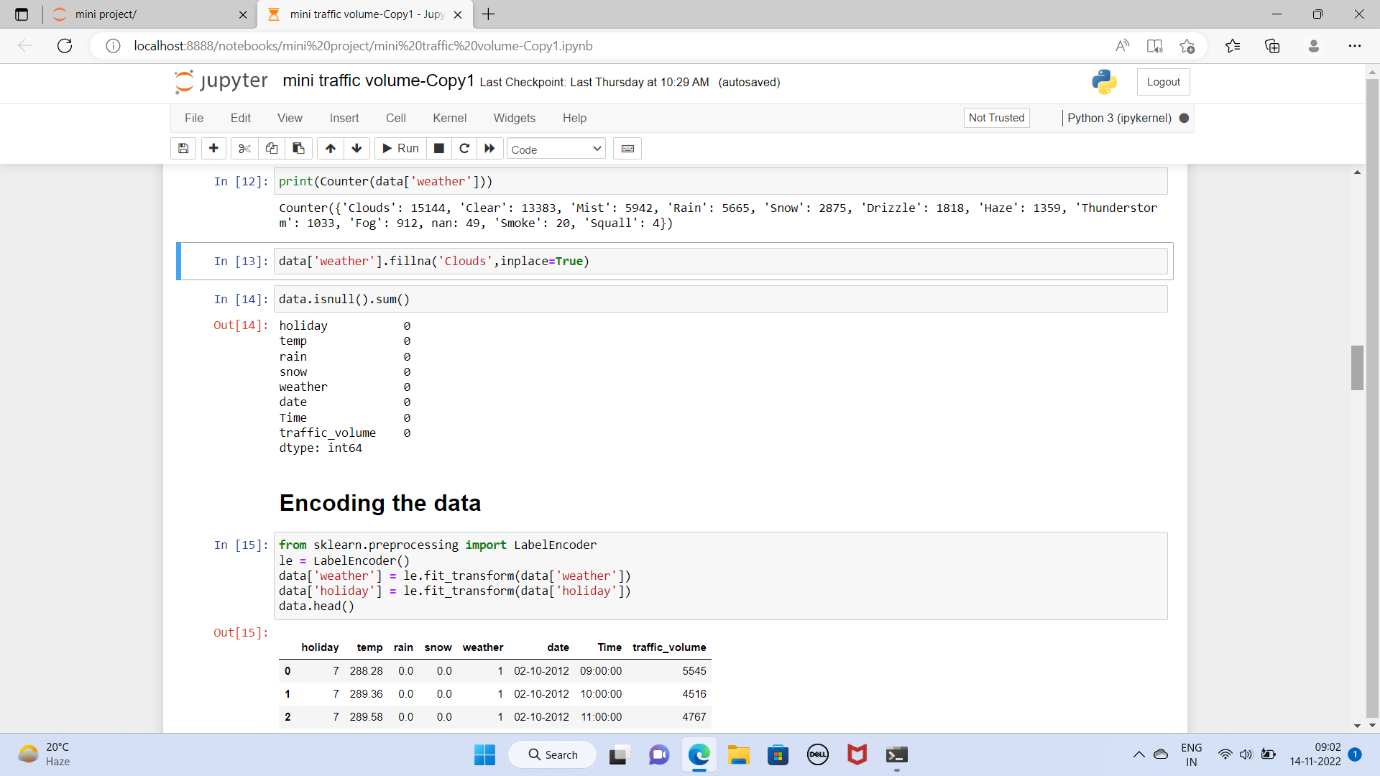
**Source Code**

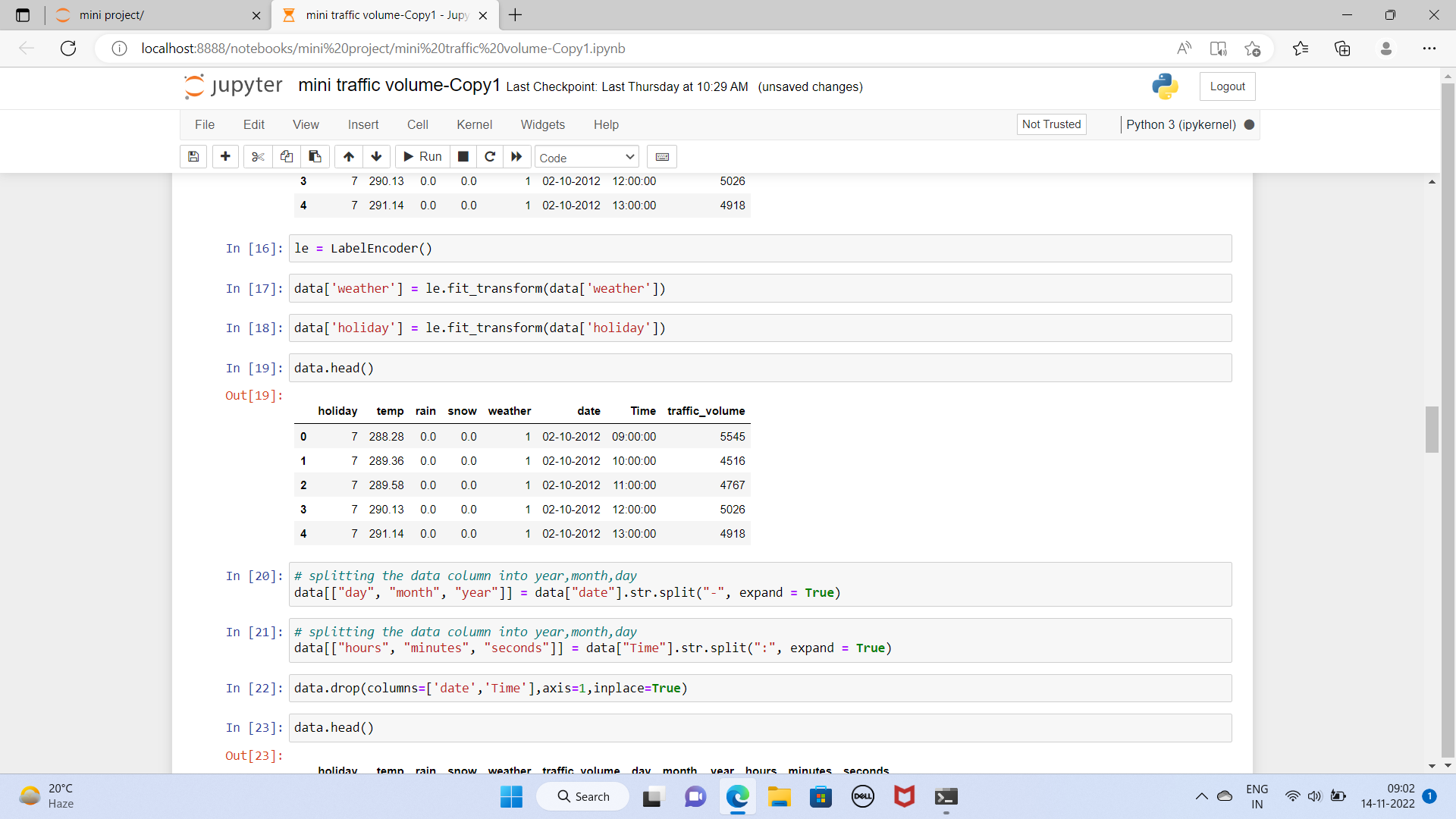


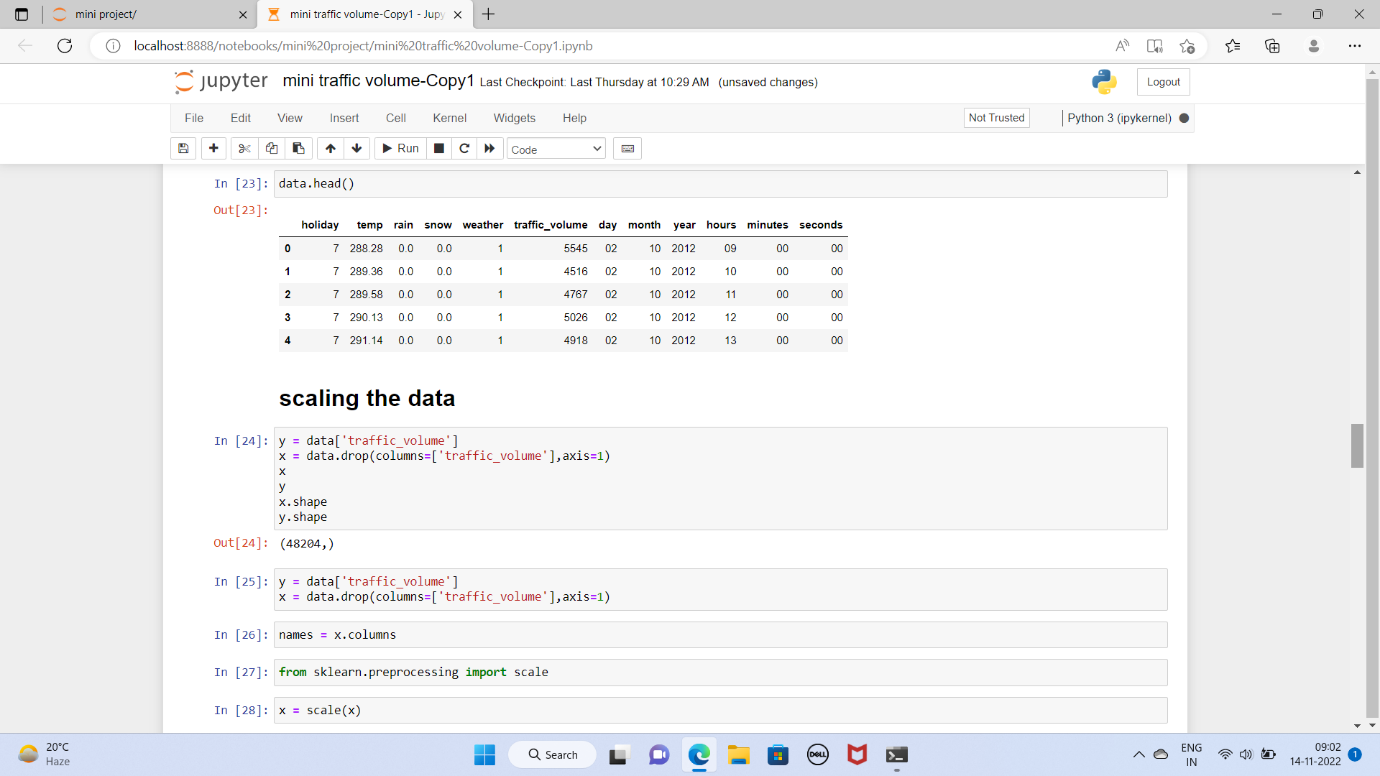


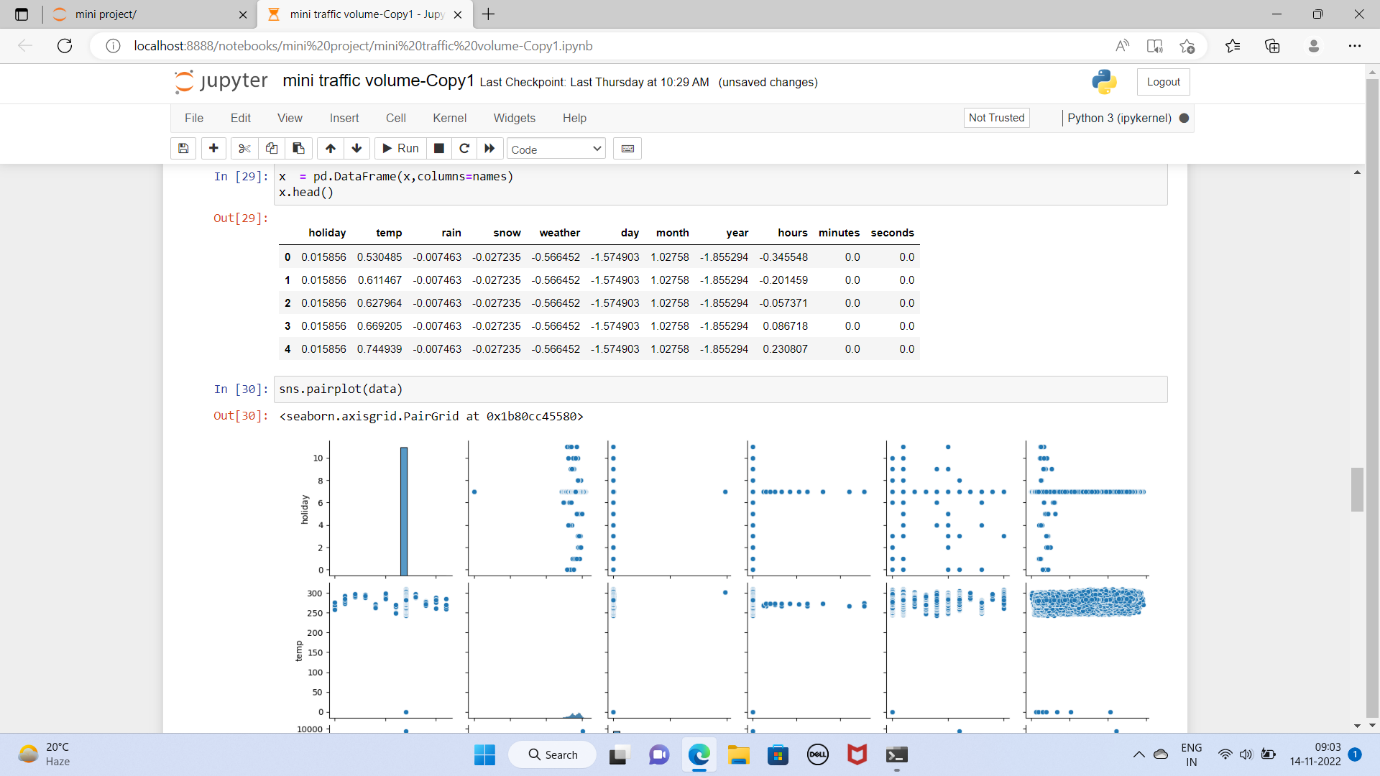


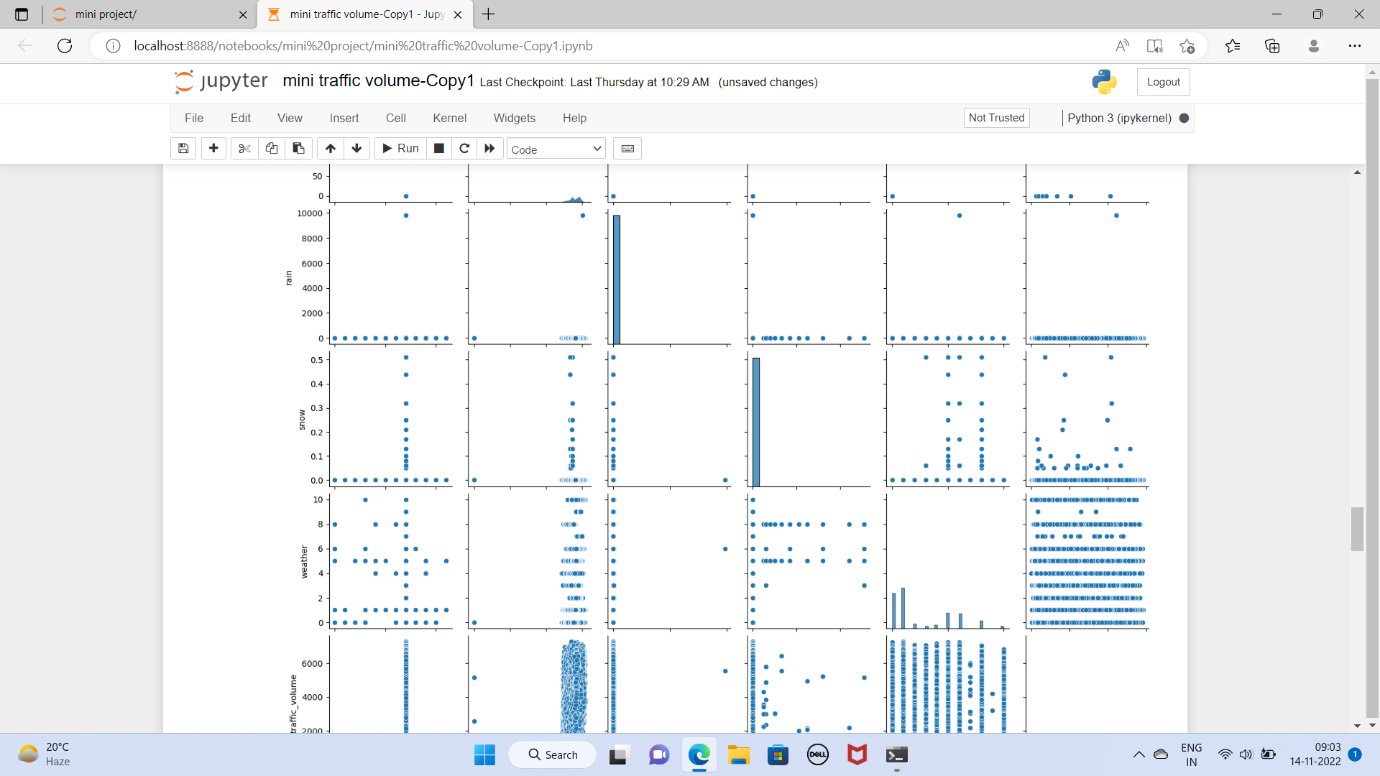


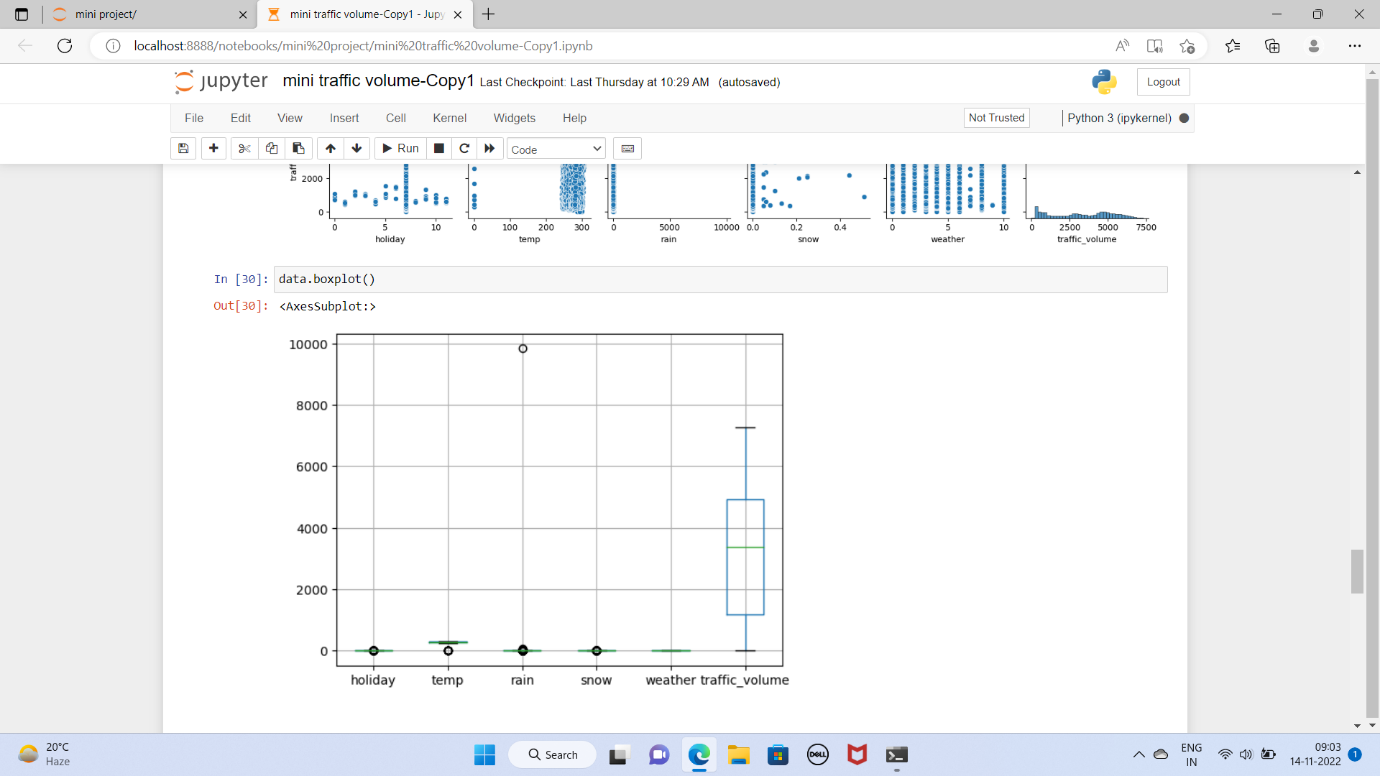


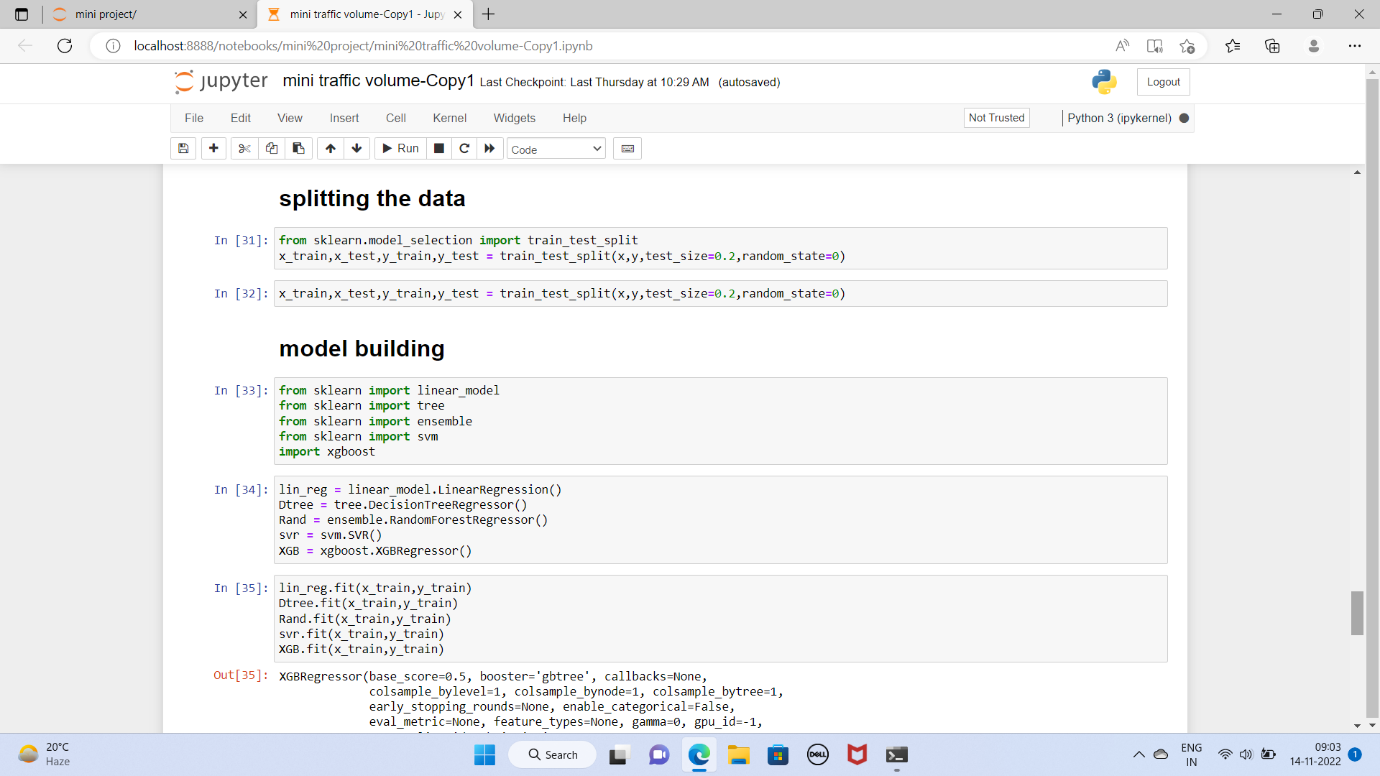


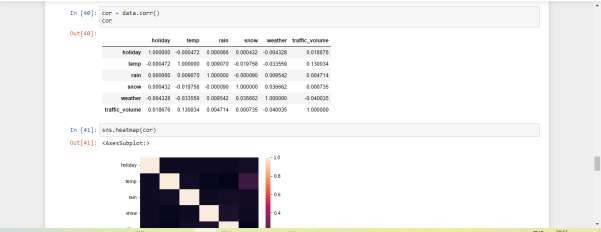


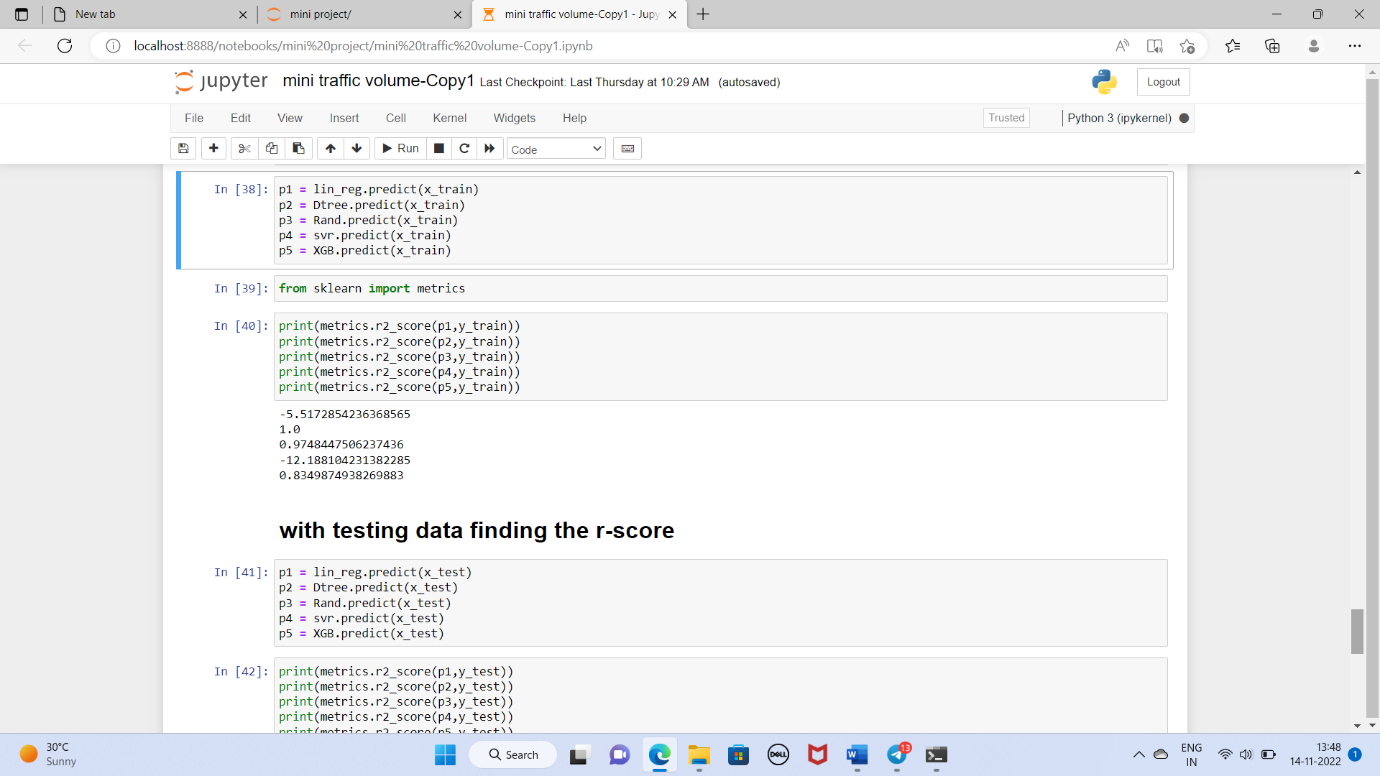


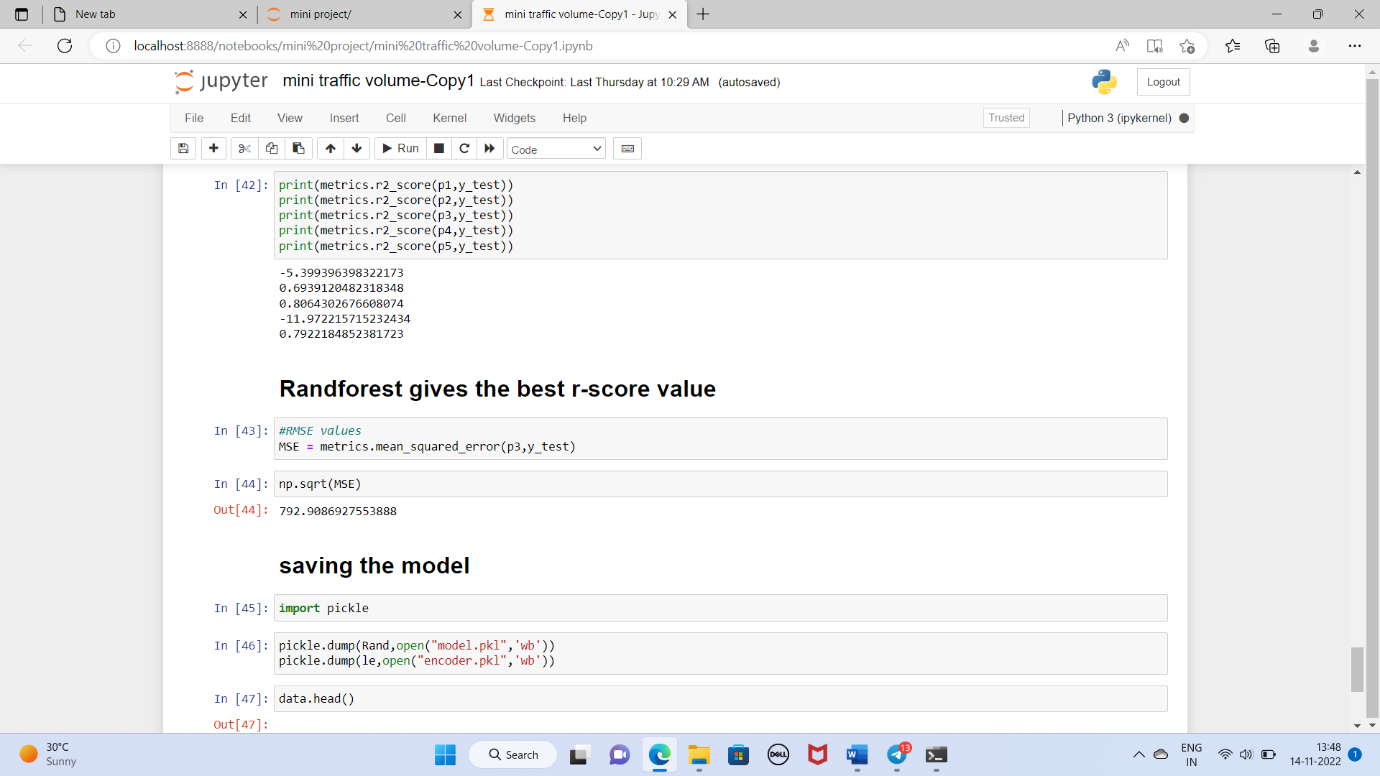


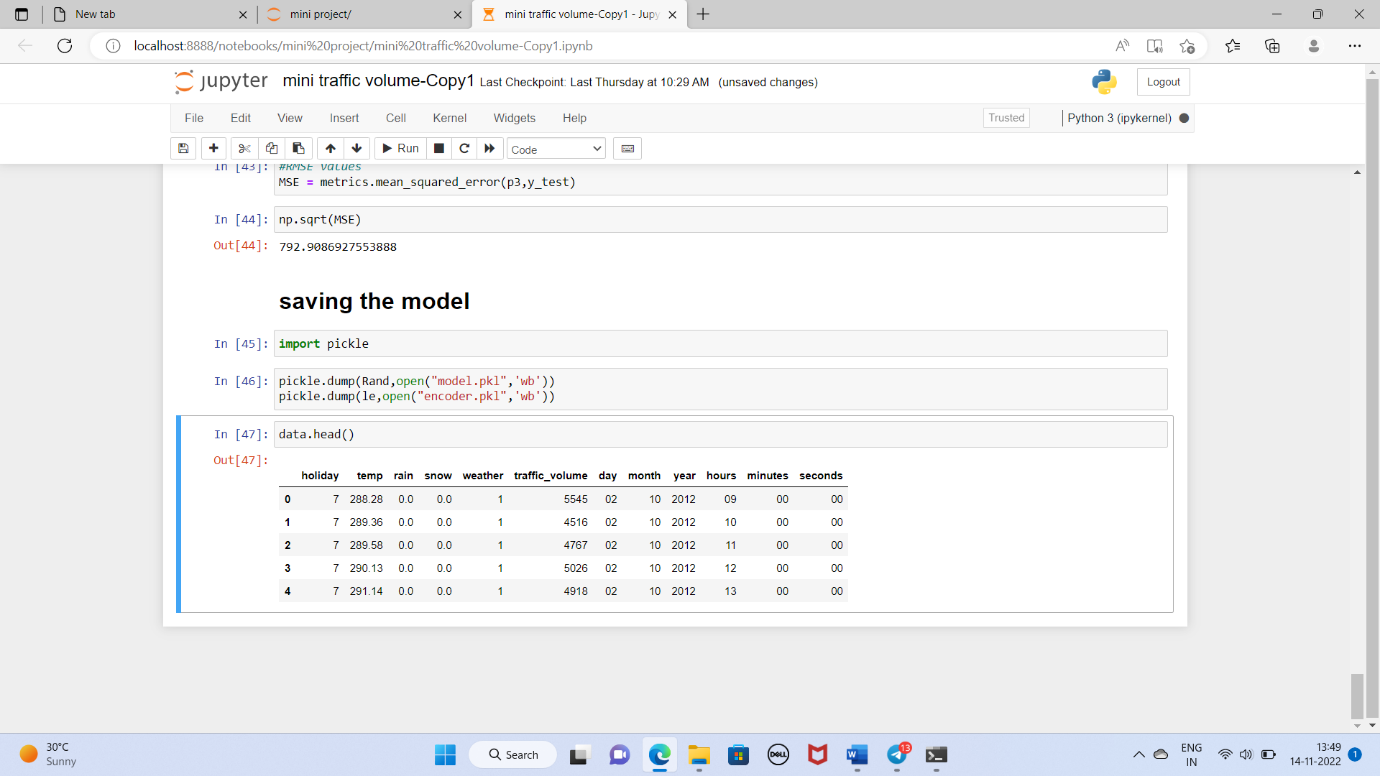












**app.py**

import numpy as np

import pickle

import time

import pandas

import os

from flask import Flask, request, render\_template

app = Flask(\_\_name\_\_)

model = pickle.load(open(r'model.pkl','rb'))

scale = pickle.load(open('encoder.pkl','rb'))

@app.route('/')# route to display the home page

def home():

return render\_template('index.html') #rendering the home page

@app.route('/predict',methods=["POST","GET"])# route to show the predictions in a web UI

def predict():

# reading the inputs given by the user

input\_feature=[float(x) for x in request. form. values() ]

features \_values=[np. array(input\_ feature)]

names = [['holiday', 'temp', 'rain', 'snow', 'weather', 'year', 'month', 'day', 'hours', 'minutes', 'seconds']]

data = pandas. Data Frame (features\_ values, columns=names)

# predictions using the loaded model file

prediction=model. predict(data)

print(prediction)

text = "Estimated Traffic Volume is :"

return render\_template("output.html",result = text + str(prediction) + "units")

# showing the prediction results in a UI

if \_\_name\_\_=="\_\_main\_\_":

# app.run(host='0.0.0.0', port=8000,debug=True) # running the app

port=int(os.environ.get('PORT',5000))

app.run(port=port,debug=True,use\_reloader=False)