# Transport Suggester

Saurabh Burewar

B18CSE050

# Contents


1. Introduction 3
2. Objective 3
3. Problem Statement 3
4. Block diagram 4
5. Datasets 5
   5.1. Roadways Dataset 5
   5.2. Airways Dataset 5
   5.3. Railways Dataset 6
   5.4. Training Data for D-Tree 6
6. Application 7
   6.1. Input 7
   6.2. Search 8
   6.3. Details 8
   6.4. Making Decision 8
   6.5. Output 9
7. Additional Modules 10
   7.1. Graph 10
   7.2. Check heuristic 10
8. Analysis of the solution 10
9. Improvements 11
10. References 11

# 1. Introduction

This is an introductory project to Artificial Intelligence involving application of different concepts in a real-world example. The aim was to create an application that helps the user decide the best mode of transport to travel from point A to B. In this context, the term 'best' means the one that best satisfies the user's preferences.

The application first takes input from the user which include their current location, destination and a set of preferences. Then, it computes the required details for each mode of transport and suggests the one that best suits the user. It also displays the details for the user to see, so that they can refer and make their own decision.

# 2. Objective

- Create a dataset that forms the basis of the whole application. This includes data for each mode of transport and a dataset to train the application to make decisions.
- Implement A* search algorithm to calculate the best possible path from the source to destination.
- Construct a decision tree to predict the mode of transport that best suits the user preferences.

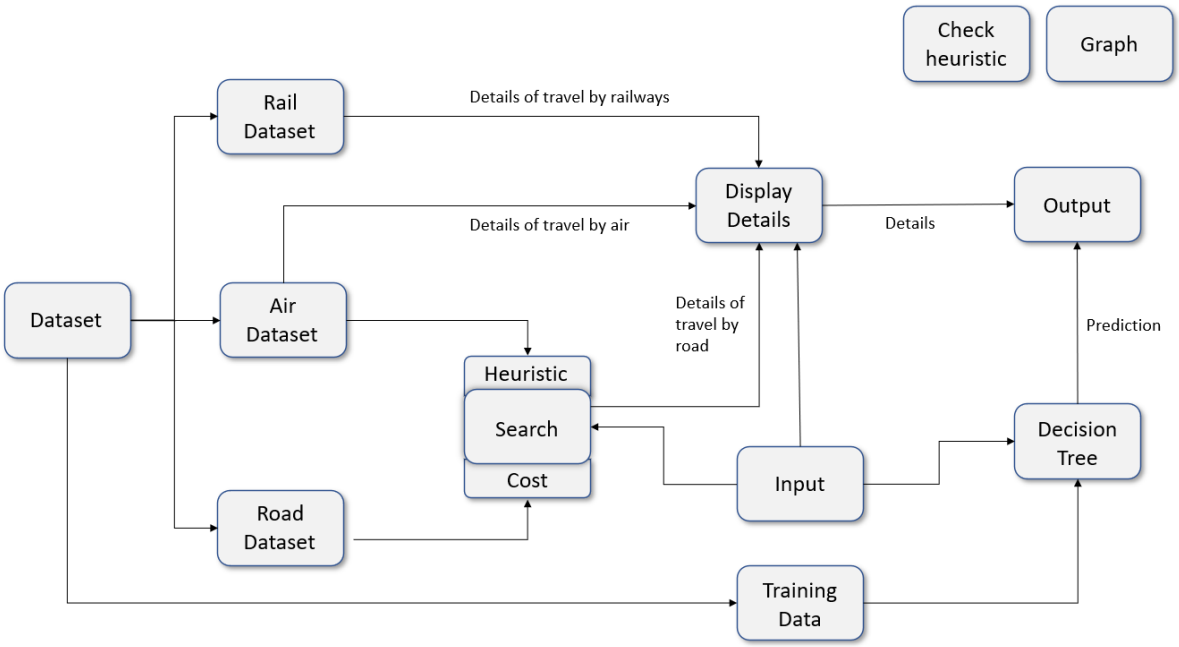
# 3. Problem Statement

Suggest a mode of transport that best suits the user's preferences. The user preferences are given as the input.

Input: Current location, destination, preferences.

Output: Best mode of transport for the user

# 4. Block diagram



# 5. Datasets

There is a total of four datasets used by the application. The datasets include entries for 20 cities of India; thus, the application supports those 20 cities only.

## 5.1. Roadways Dataset

This dataset contains the distance between two cities by road. The dataset is made according to the requirements of this project manually. The data for all the entries is taken from the travel website yatra.com.

Every city has a route to reach every other city but it is not necessary for that route to be the shortest. For example, there is a direct path from Pune to Bhopal that does not cross any of the other 18 cities but that is not the shortest path. The shortest is Pune->Indore->Bhopal.

The distance in this dataset is taken as the cost of each edge during the search.

```
Delhi,Patna,1142
Delhi,Vadodara,1025
Delhi,Ludhiana,320
Mumbai,Chennai,1334
Mumbai,Kolkata,1977
Mumbai,Bangalore,982
Mumbai,Hyderabad,700
Mumbai,Ahmedabad,522
Mumbai,Surat,278
Mumbai,Pune,148
Mumbai,Jaipur,1180
```

*Figure 1: Road dataset (RoadData.csv)*

## 5.2. Airways Dataset

This dataset contains the distance between two cities by air and the average ticket price of the respective flight. Again, it is made manually using data from yatra.com.

The distance in this dataset is taken as the heuristic during search.

```
Delhi,Patna,853,3500
Delhi,Vadodara,809,3500
Delhi,Ludhiana,286,3500
Mumbai,Chennai,1033,4000
Mumbai,Kolkata,1655,5000
Mumbai,Bangalore,845,3500
Mumbai,Hyderabad,621,2500
Mumbai,Ahmedabad,440,3000
Mumbai,Surat,233,4000
Mumbai,Pune,120,5000
Mumbai,Jaipur,921,3500
```

*Figure 2: Air dataset (AirData.csv)*

## 5.3. Railways Dataset

This dataset contains the distance between two cities by rail and the average ticket price of the respective train. The data is taken from yatra.com.

```
Delhi,Patna,1142,972
Delhi,Vadodara,1025,431
Delhi,Ludhiana,320,872
Mumbai,Chennai,1334,1196
Mumbai,Kolkata,1977,620
Mumbai,Bangalore,982,755
Mumbai,Hyderabad,700,744
Mumbai,Ahmedabad,522,433
Mumbai,Surat,278,820
Mumbai,Pune,148,623
Mumbai,Jaipur,1180,431
```

*Figure 3: Rail dataset (RailData.csv)*

## 5.4. Training Data for D-Tree

This dataset contains a record of past decisions for different combinations of inputs. The decision tree is trained on this data. More details about this in further sections.

```
Distance,Env_conscious,Budget_conscious,Avoid,Mode
100,Yes,Yes,Rail,Road
100,Yes,Yes,Road,Rail
100,Yes,Yes,Air,Rail
100,Yes,No,Rail,Road
100,Yes,No,Road,Rail
100,Yes,No,Air,Rail
100,No,Yes,Rail,Road
100,No,Yes,Road,Rail
100,No,Yes,Air,Road
100,No,No,Rail,Road
100,No,No,Road,Rail
100,No,No,Air,Road
200,Yes,Yes,Rail,Air
200,Yes,Yes,Road,Rail
```

*Figure 4: Training dataset (dTreeData.csv)*

# 6. Application

## 6.1. Input

The application takes user input to know the current location and destination.

- Current city (Source)
- Destination city

It also takes some inputs to understand the user preferences. These include –

- Whether the user wants to be environment-friendly?
- Whether the user prefers budget-friendly?
- Whether the user wants to avoid any mode of transport?

```
================================================================================================

                        Welcome to Transport Suggester

================================================================================================

Please enter source and destination

As of now you have 20 choices:
Ahmedabad       Bangalore       Bhopal          Chennai         Delhi
Hyderabad       Indore          Jaipur          Kanpur          Kolkata
Lucknow         Ludhiana        Mumbai          Nagpur          Patna
Pune            Surat           Thane           Vadodara        Visakhapatnam

Source: Pune

Destination: Bhopal

Would you like to keep things environment-friendly?
        For Yes, enter '1'
        For No, enter '0'
Answer: 1

Want it cheap?
        For Yes, enter '1'
        For No, enter '0'
Answer: 0

Is there anything you would like to avoid?
        Avoid the roads: enter '0'
        Avoid travel by air: enter '1'
        Avoid railways: enter '2'
Answer: 0
```

## 6.2. Search

The application performs an A* search on the graph generated from the datasets.

Each city is taken as a node and an edge denotes a path between the two cities. The cost of each path or the edge weight in the graph is the distance between the cities by road as given in roadways dataset (RoadData.csv). Every node has a heuristic value which is the flight distance between that node and the destination node. These values are taken from the airways dataset (AirData.csv).

The datasets are parsed and all the necessary information is stored in a dictionary which is then used in search. Since this is a small dataset, it is possible to use dictionary, but in case of very large sets we will have to find another way like streaming small sets of data.

The application takes the source (current city) and destination city as input and the search function returns the shortest path. Along with the shortest path, the function also returns a list of cities that can be visited during the road trip.

## 6.3. Details

The application provides details regarding each mode of transport in the results. These details include –

- The shortest path in the road details.

- The cost estimate for all modes
  For flights and trains, the cost estimate mainly includes the average ticket price. This, as said in section 3, is already available in the datasets.
  For roads, the cost is calculated as the gasoline expenditure and some other miscellaneous. The gasoline expenditure is according to mileage (kmpl) for an average car in India.

- The estimated $CO_2$ emissions for all modes
  This is calculated as per the distance. The average emissions by cars in India is released by ICCT (International Council on Clean Transportation). Similar data can be found on railways and airways where it is normally calculated as kg per passenger-km.

## 6.4. Making Decision

After searching the path and calculating all the details, the next thing is making a decision based on what is available. A decision tree is used to make this decision.

A decision tree algorithm is a supervised learning algorithm which learns using the available data and predicts the outcome of the query based on what it learnt. The training data discussed in section 3 is used to train and test the decision tree.

The user preferences taken as input are used as an input to the decision tree and it predicts an outcome for the given set of preferences.

Decision trees are not always correct. In this case, it can predict the outcome with an accuracy of 80%.

## 6.5. Output

The output of the application includes –

- The output from the details section
- The prediction from decision tree.

```
Pune => Bhopal (Road Travel)
=========================================================
Cities you can explore on the way       : ['Pune', 'Thane', 'Indore', 'Mumbai', 'Bhopal']
Number of cities that can be explored   : 5
=========================================================
The path I suggest you take is  : ['Pune', 'Indore', 'Bhopal']
Number of cities passed         : 3
=========================================================
Total distance          : 818 km
Cost estimate           : Rs. 5202.48
Estimated CO2 emission  : 98.98 kg
---------------------------------------------------------

Pune => Bhopal (Air Travel)
=========================================================
Estimate Cost   : Rs. 4000
Estimated CO2 emission  : 73.945 kg


---------------------------------------------------------

Pune => Bhopal (Rail Travel)
=========================================================
Estimate Cost   : Rs. 967
Estimated CO2 emission  : 6.418503 kg


---------------------------------------------------------

I think you should take a flight.

=========================================================================================================
                        Have a safe trip !!
```

# 7. Additional Modules

## 7.1. Graph

This module saves the graph that was created from the database in a PNG format. It can save both the road and the flight map with respective edge weights. As it is not part of the main aim of the application, hence it is not included in the main application. It can be called separately to save the graphs.

## 7.2. Check heuristic

This module is also a separately called module which checks if the heuristic used in search is admissible and consistent or not. When executed, it returns two Boolean values for whether the heuristic is admissible and consistent.

```
Admissible: True
Consistent: True
```

# 8. Analysis of the solution

The proposed A* search gives an optimal solution since the heuristic is consistent. This can be scaled for bigger projects by increasing the dataset and implementing heuristics by streaming, instead of using python dictionary.

In this case, implementation is done by dictionary which takes O(1) time to access element which makes it efficient. The time taken to travel the whole graph would be, in worst case, O(E) where E is number of edges in the graph because we would be travelling the whole graph for worst case.

The decision tree, on the other hand, doesn't always give the optimal solution. According to our training and testing, it has an accuracy of about 80%. So, it might not give the optimal solution.

For the decision tree, the time can be calculated in terms of the number of examples that the decision is based on. These are the examples in the training dataset. It also depends on the number of features being considered and the depth of the tree.

Let N be the number of examples, k be the features and d be the depth of tree. Here, the time can be given by O(Nkd). In this case, N is around 160, we are considering 3 features, so the time would be accordingly as O(c*d).

# 9. Improvements

There are a lot of possible improvements in this application –

- The cost and carbon emissions of a road trip can be calculated according to the specifications of the vehicle used. This would give a highly accurate estimation. But this would require a database containing information on hundreds or thousands of vehicles.

- Introduction of public road transport like buses, local taxis, auto rickshaws, cabs like uber and even car rental services like zoom car. This would again increase the data to be collected. This was not implemented because of large data requirement and limited time.

# 10. References


- ICCT records
  https://theicct.org/sites/default/files/publications/India_fuel_consumption_standards_20180925.pdf

- Carbon emissions
  https://www.carbonindependent.org/22.html
  https://www.linkedin.com/pulse/indian-railways-carbon-emission-debi-prasad-dash?articleId=6620101610364338176

- Geeks for Geeks
  https://www.geeksforgeeks.org/
  https://www.geeksforgeeks.org/decision-tree-implementation-python/

- GitHub
  https://github.com/rizanw/Romania-A-star-Algorithm/blob/master/astarRomania.py