Analysis of Public Transportation for Efficiency

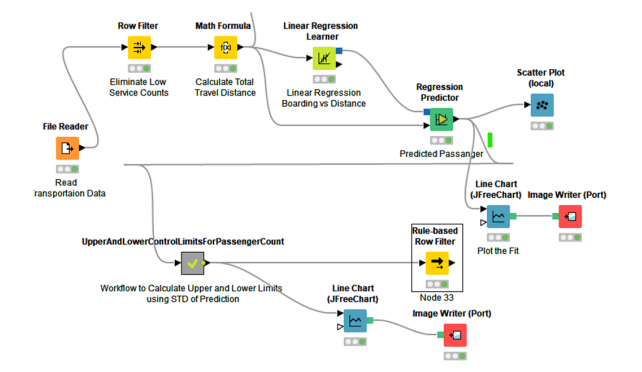
**Abstract:** Transit authorities have been searching for the indicators to measure transit service quality and the key factors to attract citizens who do not prefer public transport. The recent advent of data collection technologies such as AVL, APC, GPS and Smart Card (SC) promise opportunities for conducting comprehensive transit system performance measures, improving the quality of service while meeting passenger needs and reducing operation costs. The aim of this study is to propose metrics to improve transit service. The focus is on bus transportation since it is more flexible compared to rail transportation and widely preferred by the masses in cities. The primary data source of this study comes from the Department of Transportation for the City of Antalya. We load the complete boarding data of December 18,2019 which is a standard weekday. Most of the analysis is done using Knime software. As an outcome of this research, the analysis may propose to modify or eliminate inefficient routes, suggest new lines, identify inefficient bus stops, and potentially modify path of a route.

Keywords: Route Efficiency, Transportation Network, Bus Stop Analysis, Boarding, Clustering

1 Introduction :The subject of smart cities is a very new research area in the world. One of the most important issues for smart cities is the design of intelligent transportation systems. Transit authorities have been searching for the indicators to measure transit service quality and the key factors to attract citizens who do not prefer public transport. Various approaches and vast literature are available on evaluation of transit system performance. Quantitative performance indicators or measures are mentioned in two main dimensions: efficiency and effectiveness.

Efficiency refers to a system’s financial and productivity related performance dimensions; it is concerned with ”doing things right”. Effectiveness refers to the service’s social dimensions; it is concerned with ”doing the right things.” Early in literature, Fielding et al. [4] proposed various individual performance indicators as efficiency, effectiveness and overall indicators.

2 Methodology :In this study, the focus is on bus transportation since it is more flexible compared to rail transportation and widely preferred by the masses in cities. The primary data source of this study comes from the Department of Transportation for the City of Antalya. We load the complete boarding data of December 18,2019 which is a standard weekday. The boarding data consists of passenger Id, passenger’s boarding stop Id (origin), boarding time, bus Id, route Id (the direction on a particular line) . For a particular route, bus stop locations with GPS coordinates required to visualize path on a map, are available from both municipal and commercial websites (such as www.ulasimburada.com). The data set formed consist of 305 lines and 608 routes. A route consist of a sequential list of bus stops in either forward or backward (return) directions. Each line has opposite two directions except two lines which are omitted in analysis. On December 18, 2019, a total of 7347 trips (single direction services) were made and with these trips a total of 381962 passengers were carried. Efficiency of a transportation system as a whole requires individual busses which are limited in numbers run as efficient as possible while keeping residents happy through easy access to a bus service, quick access to destination, connectivity to 4 Kamer Ozg¨un and et.al ¨ transit network and comfort. Our approach in improving transportation system has 2 stages; 1- improve route efficiency, 2- improve customer satisfaction. However, in this study we solely focus on the route efficiency. In a followup study we aim to include customer satisfaction aspect of the transportation



**Steps to preprocessing the dataset**

**import pandas as pd**

**import scipy**

**import numpy as np**

**from sklearn.preprocessing import MinMaxScaler**

**import seaborn as sns**

**import matplotlib.pyplot as plt**

**df=pd.read\_csv('20140171.CSV')**

df.info()

<class 'pandas.core.frame.DataFrame'>

RangeIndex: 10857234 entries, 0 to 10857233

Data columns (total 6 columns):

# Column Dtype

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

0 TripID int64

1 RouteID object

2 StopID int64

3 StopName object

4 WeekBeginning object

5 NumberOfBoardings int64

dtypes: int64(3), object(3)

memory usage: 497.0+ MB

In [83]:

df.head(5)

Out[83]:

|  | **TripID** | **RouteID** | **StopID** | **StopName** | **WeekBeginning** | **NumberOfBoardings** |
| --- | --- | --- | --- | --- | --- | --- |
| **0** | 23631 | 100 | 14156 | 181 Cross Rd | 2013-06-30 00:00:00 | 1 |
| **1** | 23631 | 100 | 14144 | 177 Cross Rd | 2013-06-30 00:00:00 | 1 |
| **2** | 23632 | 100 | 14132 | 175 Cross Rd | 2013-06-30 00:00:00 | 1 |
| **3** | 23633 | 100 | 12266 | Zone A Arndale Interchange | 2013-06-30 00:00:00 | 2 |
| **4** | 23633 | 100 | 14147 | 178 Cross Rd | 2013-06-30 00:00:00 | 1 |

In [84]:

df.tail(5)

Out[84]:

|  | **TripID** | **RouteID** | **StopID** | **StopName** | **WeekBeginning** | **NumberOfBoardings** |
| --- | --- | --- | --- | --- | --- | --- |
| **10857229** | 13346 | W91C | 14629 | 21 Cashel St | 2014-07-06 00:00:00 | 1 |
| **10857230** | 13346 | W91C | 14708 | 22 Cashel St | 2014-07-06 00:00:00 | 3 |
| **10857231** | 13346 | W91C | 13709 | 2 Greenhill Rd | 2014-07-06 00:00:00 | 1 |
| **10857232** | 13346 | W91C | 14029 | 10 East Av | 2014-07-06 00:00:00 | 1 |
| **10857233** | 13346 | W91C | 13824 | 6 Leader St | 2014-07-06 00:00:00 | 1 |

In [58]:

df.isnull

<bound method NDFrame.\_add\_numeric\_operations.<locals>.sum of TripID RouteID StopID StopName WeekBeginning NumberOfBoardings

0 False False False False False False

1 False False False False False False

2 False False False False False False

3 False False False False False False

4 False False False False False False

... ... ... ... ... ... ...

10857229 False False False False False False

10857230 False False False False False False

10857231 False False False False False False

10857232 False False False False False False

10857233 False False False False False False

[10857234 rows x 6 columns]>

df.describe()

|  | **TripID** | **StopID** | **NumberOfBoardings** |
| --- | --- | --- | --- |
| **count** | 1.085723e+07 | 1.085723e+07 | 1.085723e+07 |
| **mean** | 2.952100e+04 | 1.366132e+04 | 4.743737e+00 |
| **std** | 1.960938e+04 | 1.971760e+03 | 9.382286e+00 |
| **min** | 7.900000e+01 | 1.000100e+04 | 1.000000e+00 |
| **25%** | 1.191700e+04 | 1.231100e+04 | 1.000000e+00 |
| **50%** | 2.747900e+04 | 1.334600e+04 | 2.000000e+00 |
| **75%** | 4.885800e+04 | 1.491600e+04 | 4.000000e+00 |
| **max** | 6.553500e+04 | 1.871500e+04 | 9.770000e+02 |
| **corr = df.corr()**    **plt.figure(dpi=130)**  **sns.heatmap(df.corr(), annot=True, fmt= '.2f')**  **plt.show()** |  |  |  |

In [69]:

df.isnull().sum

<bound method NDFrame.\_add\_numeric\_operations.<locals>.sum of TripID RouteID StopID StopName WeekBeginning NumberOfBoardings

0 False False False False False False

1 False False False False False False

2 False False False False False False

3 False False False False False False

4 False False False False False False

... ... ... ... ... ... ...

10857229 False False False False False False

10857230 False False False False False False

10857231 False False False False False False

10857232 False False False False False False

10857233 False False False False False False

[10857234 rows x 6 columns]>

corr['StopID'].sort\_values(ascending **=** **False**)

Out[69]:

StopID 1.000000

TripID 0.105974

NumberOfBoardings 0.038397

Name: StopID, dtype: float64

In [77]:

X **=** df.drop(columns **=**['StopID'])

Y **=** df.StopID

In [68]:

**def** mean\_imputation(data, inplace **=** **False**):

data.fillna(data.mean(), inplace **=** inplace)

​

In [ ]:

scaler **=** StandardScaler()

scaler.fit(X\_train)

X\_train\_standardized **=** scaler.transform(X\_train)

X\_cv\_standardized **=** scaler.transform(X\_cv)

In [49]:

**import** imblearn

**from** imblearn.over\_sampling **import** RandomOverSampler

**from** imblearn.under\_sampling **import** TomekLinks

**from** imblearn.over\_sampling **import** SMOTE

**from** imblearn.under\_sampling **import** NearMiss

**def** sampler\_function(data\_x, data\_y, sampler **=** 0, random\_state **=** 101):

**if** sampler **==** 0:

sampler **=** RandomOverSampler(random\_state **=** random\_state)

**elif** sampler **==** 1:

sampler **=** TomekLinks()

**elif** sampler **==** 2:

sampler **=** SMOTE()

**else**:

sampler **=** NearMiss()

X\_transformed, y\_transformed **=** sampler.fit\_resample(data\_x, data\_y)

print('Original dataset shape:', Counter(data\_y))

print('Resample dataset shape:', Counter(y\_transformed))

**return** X\_transformed, y\_transformed

In [87]:

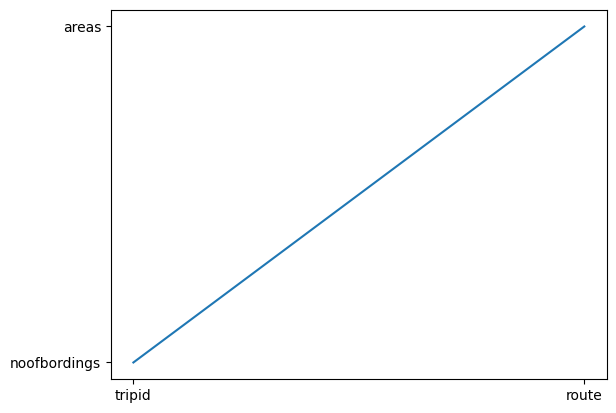
**from** matplotlib **import** pyplot **as** plt

x **=** ['tripid', 'route']

y **=** ['noofbordings', 'areas']

plt.plot(x, y)

plt.show()



In [89]:

from matplotlib import pyplot as plt

x = ['TripID','routeID','StopID']

y = ['StopName','weekbeginning','No of bordings']

plt.scatter(x, y)

plt.show()

plt.plot(x, y)

plt.show()

