Project Report

BART Transit System Ridership Prediction

1. Definition

a. Project Overview

This project aims to predict the busiest day on BART Transit System. Number of BART Transit System users increase each year, and commute information is important for people. Ridership prediction may provide essential information for people who cares how busy is their commute today. This information shows people what they may expect on their route in terms of ridership which can help them to choose more appropriate commute.

Information about busiest day on BART Transit System may alert people in order they consider to use alternative transportation, which may save their commute time.

Related academic research and earlier work:

https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0202707 **[1]**
http://docs.trb.org/prp/16-0688.pdf **[2]**

Academic research above use Regression techniques for ridership prediction problems. A regression problem is when the output variable is a real value, such as "dollars", "weight" etc. This is a Regression analyses case which is part of Supervised Learning because the output variable is a real value – Throughput - capacity, number of passenger went b/n Origin and Destination stations. The goal is predict a number of passengers in each day of the week, and identify the busiest day on the BART.

Dataset source: https://www.kaggle.com/jonathanbouchet/bart-transit-system/data

b. Problem Statement

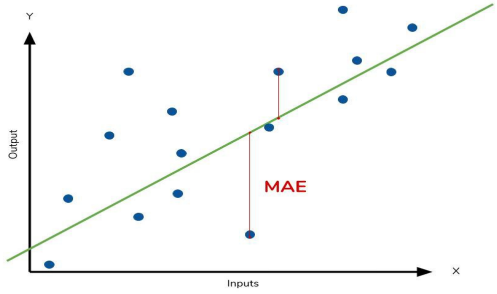
The goal is predict Throughput variable - capacity, number of passenger went between Origin and Destination stations, and identify the busiest day on the BART. Develop a Supervised Learning model using Logistic Regression algorithm to predict Ridership on BART Transit System using information from dataset.
This is the Regression problem, and I'm planing to use a Linear Regression. In Linear Regression the outcome (dependent variable) is continuous. It can have any one of an infinite number of possible values.
Below is brief overview of my machine learning pipeline:
1) Clean and pre-processed data;
2) Randomly split data into a training set and testing set (80/20);
3) Build a Linear Regression model to predict occupancies for each day of the week using all of the other variables as independent variables;
4) Identify busiest day of the week.

c. Metrics

The Mean Absolute Error will be a metric for this project.
The Mean Absolute Error (or MAE) is the sum of the absolute differences between predictions and actual values. It gives an idea of how wrong the predictions were.

$$MAE = 1N\sum i = 1N|\hat{\theta} i - \theta|$$

The picture below is a graphical description of the MAE. The green line represents the model's predictions, and the blue points represent the data.



MAE requires linear programming to compute the gradient. MAE is more robust to outliers which might be useful for this project.
I will use The Mean Absolute Error function from scikit-learn library.

2. Analysis

a. Data Exploration

The dataset includes *date-hour-soo-dest-2017.csv file which* contains daily inter- station ridership for (part of) 2017. There are over 2000 station-to-station combinations in the dataset.

date-hour-soo-dest-2017.csv is number of passengers (Throughput) that went between two stations of Origin and Destination in a given time (DateTime) in (part of) 2017, including:

NAME	DESCRIPTION
Origin	Short name of the origin station
Destination	Short name of destination station
Throughput	Capacity, number of passenger went b/n origin and destination stations
DateTime	Capacity timeframe.

Total: 3.31m rows and 4 columns.

Throughput is a target variable, this is number of passenger went b/n origin and destination stations.

b. Data Representation:

Head and the Tail of the Dataframe:

```
train=pd.read_csv('date-hour-soo-dest-2017.csv')

train.head()
```

	Origin	Destination	Throughput	DateTime
0	12TH	19TH	1	2017-01-01 00:00:00
1	12TH	24TH	2	2017-01-01 00:00:00
2	12TH	BAYF	1	2017-01-01 00:00:00
3	12TH	CIVC	5	2017-01-01 00:00:00
4	12TH	COLS	2	2017-01-01 00:00:00

```
train.tail()
```

	Origin	Destination	Throughput	DateTime
3313620	WSPR	MONT	1	2017-05-03 23:00:00
3313621	WSPR	NBRK	1	2017-05-03 23:00:00
3313622	WSPR	NCON	1	2017-05-03 23:00:00
3313623	WSPR	SANL	2	2017-05-03 23:00:00
3313624	WSPR	SHAY	4	2017-05-03 23:00:00

As you can see, I have the dataset for period from January 1, 2017 to May 3, 2017.

```
train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3313625 entries, 0 to 3313624
Data columns (total 4 columns):
Origin        object
Destination   object
Throughput    int64
DateTime      object
dtypes: int64(1), object(3)
memory usage: 101.1+ MB
```

I have a data with 3313625 entries and memory usage 101.1+ MB.

Descriptive Statistics:

```
train['Throughput'].value_counts().describe()
```

```
count       868.000000
mean       3817.540323
std       36809.852892
min           1.000000
25%           4.000000
```

```
50%          23.000000
75%         170.500000
max      825393.000000
Name: Throughput, dtype: float64
```

Descriptive statistics of the target vatiable - Throughput shows:

count(Number of non-null observations) = 868;

mean(Mean of Values) = 3817.540323;

std (Standard Deviation of the Values) = 36809.852892 is the amount of variation or dispersion of a set of data values;

min (Minimum Value) = 1;

25% (25% Value) = 4;

50% (50% Value) = 23;

75% (75% Value) = 170.5;

max (Maximum Value) = 825393.

b. Exploratory Visualization:

In this section, I will provide some form of visualization of the data.

```
%matplotlib inline
train['Day']=train.DateTime.dt.weekday_name

train.head()
```

	Origin	Destination	Throughput	DateTime	Day
0	12TH	19TH	1	2017-01-01	Sunday
1	12TH	24TH	2	2017-01-01	Sunday
2	12TH	BAYF	1	2017-01-01	Sunday
3	12TH	CIVC	5	2017-01-01	Sunday
4	12TH	COLS	2	2017-01-01	Sunday

Run the code above I can see DateTime information by days of the week. Since my goal to predict the busiest day of the week on BART this information is helpful.

Using matplotlib I visualize a Throughtput by day of the week

```
train.Day.value_counts().sort_index().plot()
```

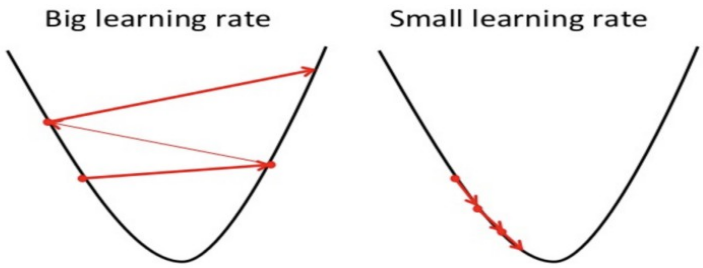
```
<matplotlib.axes._subplots.AxesSubplot at 0x1b501a250f0>
```

According to visualization above, it seems that the highest nember of Throughput is on Wednesday. Total Throughtput for Wednesday for period from January 1, 2017 to May 3, 2017 is around 520,000.

c. Algorithms and Techniques:

The most common solution to such problems is the method of Regression. There are some of the Regression methods such as:

In Linear Regression, the outcome (dependent variable) is continuous. It can have any one of an infinite number of possible values. In Logistic Regression , the outcome (dependent variable) has only a limited number of possible values.
I will use Linear regression which is useful for finding relationship between two continuous variables. One is predictor or independent variable and other is response or dependent variable. It looks for statistical relationship but not deterministic relationship. Relationship between two variables is said to be deterministic if one variable can be accurately expressed by the other. Linear Regression is gradient descent. Gradient descent is a method of updating a_0 and a_1 to reduce the cost function. Gradient descent helps us on how to change the values. We can use only discrete number of steps. If choose short steps at, it might take a longer time. If choose long steps, there is a chance that you could overshoot the bottom.



In the gradient descent algorithm, the number of steps is the learning rate. This decides on how fast the algorithm converges to the minima.

d. Benchmark

I will use the mean - average Throughput as the benchmark.
I will calculate the mean using Python as a sum of the elements divided by total number of elements.
I will use mean function from scikit-learn library.

3. Methodology

a. Data preprocessing

Data cleaning: In order to avoid Value Error with converting string to float with DateTime

column. Identify if I have missing data.

```
train.isnull().sum()
```

```
Origin         0
Destination    0
Throughput     0
DateTime       0
dtype: int64
```

As a part of preprocessing data, the table above shows that there is no missing data.

I'll change the type of the 'Day' column to the integer

```
train['DateTime'] = pd.to_datetime(train.DateTime)
train['Day']=train.DateTime.dt.dayofweek
train.dtypes
Origin                 object
Destination            object
Throughput              int64
DateTime       datetime64[ns]
Day                     int64
dtype: object
```

Test Train Split Data

The goal is to create a model that generalises well to new data. The test set serves as a proxy for new data. Trained data is the data on which we apply the linear regression algorithm. And finally I test that algorithm on the test data. The code for splitting is as follows:

```
from sklearn.cross_validation import train_test_split
X = train[['Day']]
y = train['Throughput']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
random_state=99)
```

From the above code snippet I infer 20% of the data which goes to the test data and the rest remains in the training set.

b. Implementation:

Machine Learning Pipline:

1) Import the numpy and pandas libraries and matplotlib visualization;
2) Read and display the data;
3) Data and Visual Exploration: show Descriptive statistics of the target vatiable – Throughput, Identify missing data, update type of the variables, visualization of a Throughtput by day of the week;
4) Train Test Split Data;
5) Training and Predicting by Linear Regression

Training and Predicting by Linear Regression

```
from sklearn.linear_model import LinearRegression
lm = LinearRegression()
lm.fit(X_train,y_train)
```

LinearRegression(copy_X=True, fit_intercept=True, n_jobs=1, normalize=False)

The Mean Absolute Error (Metric of the Project) is the sum of the absolute differences between predictions and actual values. It gives an idea of how wrong the predictions were.

```
from sklearn.metrics import mean_absolute_error
mean_absolute_error(y_test,predictions)
```

14.296556398102966

MAE compare true values to their estimates, but do it in a slightly different way.
MAE is an average sum of all absolute errors. Absolute values – have less influence on the result.

Calculate Mean (Benchmark). I calsulate mean of all days of the week in my dataset.

```
from statistics import mean
train['Throughput'].mean()
```

12.536420083745144

Mean	12.54
MAE	14.3

Prediction result is quite close to the Benchmark (Mean).

c. Refinement

Cross-Validation

With cross-validation the entire available dataset is divided into more-or-less equal sized subsets.

The major advantage of any form of cross-validation is that each result is generated using a classifier which was not trained on that result.

Using multiple train-test splits will result in more models being trained, and in turn, a more accurate estimate of the performance of the models on unseen data.

from sklearn.model_selection import TimeSeriesSplit

```
X = train.values
splits = TimeSeriesSplit(n_splits=3)
for train_index, test_index in splits.split(X):
    train = X[train_index]
    test = X[test_index]
    print(len(train) + len(test))
    print(len(train))
    print(len(test))
```

```
1656813
828407
828406
2485219
1656813
828406
3313625
2485219
828406
```

4. Results:

a. Model Evaluation and Validation

The Mean Absolute Error = 14.296556398102966
Mean = 12.536420083745144
Difference = 1.76

b. Justification

Parameters	Final model	Benchamrk	Difference
MAE	14.3	12.54	1.76

5. Conclusion

a. Free-from visualization

The most important feature is Throughput.

b. Reflection:

The project might be summarized as the sequence of following steps:

1. Searching for a dataset for chosen problem on Kaggle.com;
2. Download and preprocess the data;

3. Deciding the algorithms to be used to solve the problem;
4. Chosing a benchmark;
5. Applying an algorithm and review the result;

First of all, I'm excited by idea of the Project which might be useful for people in their daily commute routine. Data was decent but without missing features.
Visualizing dataset was useful and fun. It's easy to read and understand the data. Display ridership information by day of the week provide general pattern of the data.
Since it is a Regression problem, I use Linear Regression model and MAE as a evaluation metric. MAE was quite useful, it's simple and the same time convinient metric for this Project because it doesn't squared an errors that are farther away from the mean and provide more weighted result. The mean appears straightforward and steady benchmark.

c. Improvement:

Several ways to improve model:

- Get More Data (for whole year and more);
- Use Feature Engineering (add new relevant features);
- Random Search (sample algorithm parameters from a random distribution for a fixed number of iterations)
- Combine all previous ways to obtain more efficient model