International Journal of Engineering & Technology, 7 (4) (2018) 5318-5321

International Journal of Engineering & Technology

Website: www.sciencepubco.com/index.php/IJET
doi: 10.14419/ijet.v7i4.14099
Research paper

Study on Forecasting Public Revenue for Andhra Pradesh State Road Transport Using Regression Techniques

Reshma Gummadi[1]*, Sreenivasa Reddy Edara[2]

[1]Research Scholar, Acharya Nagarjuna University.Guntur, Andhra Pradesh, 522510
[2]Department of Computer Science & Engineering, Acharya Na-garjuna University, Guntur, Andhra Pradesh, 522510
**Corresponding author E-mail:reshma.gorripati@gmail.com*

Abstract

Forecasting public revenue of transport system indirectly shows the development of system with the growth of economy. Andhra Pradesh state road transport corporation (APSRTC) develops rapidly by this process .Therefore there is a need for developing model to forecast public revenue and identifying the relationship between transportation and economy of the system. The passenger traffic volume collected from the APSRTC for calculating public Revenue is based on the data from 2016 to 2017.Then different regression models are applied on the collected data for analysis purpose. The linear regression outperforms gradient boosting, decision trees and neural networks with a relative error less than 5% for predicting public revenue. The accuracy of models on revenue has been improved which helps for the development of transportation system.

Keywords:*Public Revenue; Linear Regression; Gradient Boosting.*

1. Introduction

Forecasting public Revenue based on number of passengers travelling on the bus helps for planning and development of transport system. Different forecast approaches including regression techniques applied on the data which helps in discovering the relationships among the variables [2].There are various regression techniques which help to forecast accurately. Regression techniques include linear regression, gradient boosting regression, and random forest regression and so on. In this paper general procedure for gradient booting algorithm and linear regression is explained and implemented in python to know the predicted values of public revenue. By plotting graphs linear regression and gradient boosting regression techniques were compared.
The relationship between Income and passenger flow is difficult to analyze. This paper forecast the revenue using regression techniques to improve accuracy. Linear regression gave better forecasting results compared to Random forest and gradient boosting.

2. Forecasting revenue using regression techniques

2.1. Data source

The Transit bus information of route wise data ,in this case from macherla is extracted from databases for the period of April 2016 to 31st December 2016.to forecast the income firstly the ticket fare of passengers paid on each date is calculated .so before applying regression techniques thetotal amount on that date paid byall passengers travelledisready.

#grouping the date field and calculating total amount for each date.

```
Dataset= data.groupby(['Date'],as_index=False)[['TotalAmount']].sum()
#declaration of temporary arrays to store the data.
x_data = []
y_data = []

for d in range(6,dataset.shape[0]):
print(d)
#The matrix will have columns one for each moth,prior to the one to be predicted, here start with the seventh month available.

x=dataset.iloc[d-6:d].values.ravel()
y = dataset.iloc[d].values [0]
x_data.append(x)
y_data.append(y)

x_data = np.array(x_data)
y_data = np.array(y_data)

Print (y_data)
```

Copyright © 2018Reshma Gummadi, Sreenivasa Reddy Edara. This is an open access article distributed under the Creative Commons Attribution License, which permits unrestricted use, distribution, and reproduction in any medium, provided the original work is properly cited.

```
113070.   190849.   178119.   203231.   149219.   168152.   157667.
132262.   159228.   143657.   195399.   130754.   157034.   208851.
201442.   179897.   239425.   252165.   190044.   248278.   173148.
239058.   162149.   171232.    98047.    97022.    79582.    63958.
 88719.    75031.    97524.    86714.    81023.    86459.   101145.
 89713.    63358.    83382.    88144.    82259.    85676.    67578.
 91247.    85391.    83518.    96857.    81551.    80153.    81757.
 89834.    71357.    98946.    86465.    92895.    89018.   142602.
211288.   159180.   127583.   210081.   193875.   148890.   204389.
196733.   181081.   153867.   270009.   170116.   182082.   220285.
196160.   147075.   190173.   191400.   181231.   167140.   188258.
213414.   157847.   166549.   202828.   141299.   181282.   153862.
229569.   182349.   157006.   105159.    16787.    82500.     6034.
 57898.   124530.   106935.   216163.   120180.   192490.   175527.
189661.   202753.   147409.   202009.   143809.   186989.   153814.
169222.   177927.   123408.   207669.   134231.   167765.   140070.
137562.   157748.   154717.   184852.    36372.   148005.   192656.
175699.   174938.   219836.   189229.   192336.   216501.   231946.
167519.   198813.   212645.   210317.   232728.   210489.   174579.
184168.   214461.   183377.   205831.   218379.   157405.   176944.
176734.   179241.   147809.   191538.   133794.   162929.   142825.
127112.   195618.   171829.   121764.   179851.   176200.   147733.
143363.    70776.   143398.   187028.   113127.   169996.   127554.
198133.   176449.   189279.   183108.   160940.   160967.    55476.
149335.   160776.   177472.   168410.   168535.   172342.   145394.
155143.   169418.3  194846.   209975.   173669.   179890.   171768.
193831.4  206084.   216621.   198425.   101593.   209954.   232477.
228491.   218107.   215310.   245249.   193555.   155402.   156053.
193437.   162165.   220555.   182343.   162966.   137158.   181855.
145186.   145433.   151462.   192274.   174840.   194400.   164034.
204949.   189842.   223572.   195731.   170164.   163602.   171618.
166646.   210099.   174838.   204074.   164910.   204547.   167845.
178870.   160982.   200681.   196235.   169752.   106322.   186558.
192923.   157980.   190022.   168245.   157795.   165151.   170836.
216117.   176415.   156228.8  203908.   170763.   159740.   196766.
190208.   142245.   212019.   170018.   149133.   183575.   177619.
180587.   183161.   166621.   175371.   163806.   153404.   180182.
156400.   166008.   185192.   154103.   153403.   195280.   114341.
185246.   130696.   157947. ]|
```

Fig. 1: Snapshot of Total Sum for Each Date Is Displayed.

2.2. Gradient boosting method

Gradient Boosting is a special type of boosting technique which reduces errors sequentially. In Boosting, Models are built in series. In each model, the heights are adjusted based on learning from previous models. After applying 1st and second model, the errors are identified and reduced in 3rd model. The term boosting itself means combining weak models to create powerful models. Decision tree is one of the most popular choices of ensemble models.

2.2.1. General procedure for gradient boosting

1) Develop a regression tree. For any leaf node, it gives an average value.
2) Find errors for all the observations of leaf nodes.

Error node1=x1-x_{avg}
Error node2=x2-x_{avg}

3) Consider the above errors as new observation values.
4) Develop second Tree to minimize these errors.
5) Repeat steps 3 to 4 for a specified number of iterations.

2.2.2. Gradient boosting in python

1) Load the Data
2) Splitting the datasets in to two sets, one is assigned as training and the other is for validating the set. The training dataset will be used to generate trees that will form in tothe final average model.
3) Fit a Gradient Tree Boosting model to the data using sckit-learn package. The number of trees and depth can be chosen at this stage.
4) Observe train errors if it goes down as more trees are added to the model. The test error remainsconstant and does not gain over fitting values. it will be one of the good features of the algorithm.
5) Identify the important variables and investigate how these variables interact between them. Plot the dependencies of the considered target variable with another one .Some variables have dependencies with target value.
6) Training data points can be drawn by picking an x coordinate randomly.
7) Fit a gradient boosting model to the training data and the approximation progresses are to be observed and based on that more trees can be added. The gradient boosting estimatesevaluate the model by using function as the number of trees. The function will return a generator that iterates over the predicted values as more number of decision trees are added.

The procedure of gradient boosting mainly involves
Learning a regression predictor, computing error residuals that can be done by the following formula. Actual target value-predicted target value(A1=x-x_predcited).later Learn to predict the residual thenFit a new model on error residuals as target variable with same input variables(A1-predicted).now Add the predicted residuals to the previous predictions(X_predcited2=x_predcited1+A1-predcited). Repeat this procedure until it starts over fitting or theconstant value occurs.

2.2.3. Implementation in python

The data collected from APSRTC consists of number of fields. For this purpose the date is grouped from entire data and sum of the amount paid by the passengers are taken in to consideration.

```
Dataset = data.groupby(['Date'],as_index=False)[['TotalAmount']].sum()

dataset.index=dataset.Date
dataset=dataset[['TotalAmount']]

# learning a regression predictor can be done by the following steps.
for i in range(30,end):

x_tr = x_data[:i,:]
y_tr = y_data[:i]

x_ts = x_data[i,:]
y_ts = y_data[i]


model = LinearRegression()
model.fit(x_tr,y_tr)

y_pred.append(model.predict(x_ts))
y_pred_last.append(x_ts[-1])
y_pred_ma.append(x_test.mean())
y_true.append(y_test)

# computing error residuals.
aa=list(y_pred)
bb=[list(x) for x in aa]
y_pred_1=[]
for i in bb:
#print(i)
y_pred_1.append(i[0])

# Fitting a model on error residuals.
model = GradientBoostingRegressor(n_estimators =10,)
model.fit(x_train,y_train)
for d in range(6,dataset_test.shape[0]):
x = dataset_test.iloc[d-6:d].values.ravel()
y = dataset_test.iloc[d].values[0]
x_testt.append(x)
y_testt.append(y)
y_pred11 = []

for i in x_testt:
y_pred11.append (model. Predict(i))
```

Repeat this process until it starts over fitting.

After plotting the graph the view is shown below in Figure 2

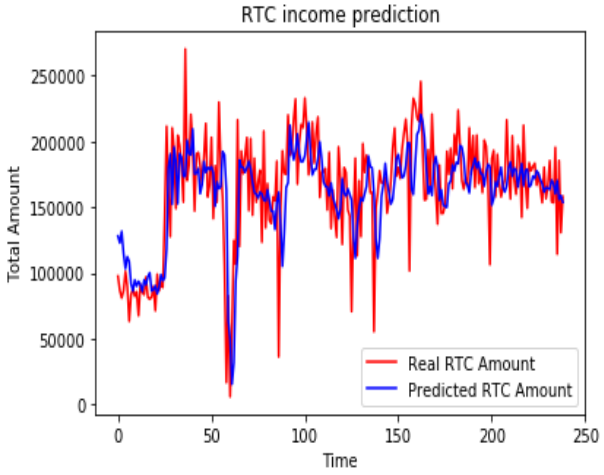


Fig. 2: Income Prediction Using Gradient Boosting.

3. Forecasting revenue based on linear regression

Linear regression is a linear model which a linear relationship can be estimated between input variables and single output variable when the input variable is a single variable then that model is called simple linear regression. The input variable is called independent variable and the output variable is called dependent variable [5]. The linear relationship can be defined as

Y=β0+β1x

Where β1 is called coefficient and β0 is called bias coefficient.
The coefficients can be found using the ordinary least square method and gradient descent approach. The ordinary least square method is used in simple linear regression. a good model will always have minimum error. A total error of this model is the sum of all errors of each point [5].

D=£di2

Di is the distance between line and where m is the total number of points.
The model can e evaluated using Root Mean squared Error and coefficient of determination.
RMSE is the square root of sum of all errors divided by number of values[5].

3.1. Implementation in python

1) Linear regression cannot be applied to the entire dataset; hence the data is spitted in to training data and test data. Now model can be trained on training data and testing can be performed on test data. The linear regression model can be fitted to the training data set.

```
y_pred11 = []

end1 = y_testt.shape[0]
for i in range(30,end1):

# x_train = x_data[:i,:]
# y_train = y_data[:i]

x_testt = x_testt[i,:]
y_testt = y_testt[i]

# model = LinearRegression(normalize=True)
# model.fit(x_train,y_train)

y_pred11.append(model.predict(x_testt))
y_pred_last11.append(x_testt[-1])
y_pred_ma11.append(x_testt.mean())
y_true11.append(y_testt)

model.predict([166621, 175371, 163806, 153404, 180182, 156400])

print(model.predict(x_testt))

[168395.69898104 163972.35456069 167979.48902895 168395.69898104
163636.20217048 166158.12370387 168395.69898104 163636.20217048
173082.38818986]
```

2) Creating a difference transform of the dataset and make a prediction which give regression coefficients and lags

```
def difference (dataset):
diff = list()
for i in range(1, len(dataset)):
value = dataset[i] - dataset[i - 1]
diff.append(value)
return numpy.array(diff)

def predict1(coef, history):
y = coef[0]
for j in range(1, len(coef)):
y += coef[i] * history[-j]
return y
```

3) Split the dataset and train the data using auto regression. While during testing walk forward over time steps. The test MSE is obtained and plotted figure obtained as shown in Figure 3.

Test MSE: 938407405.243

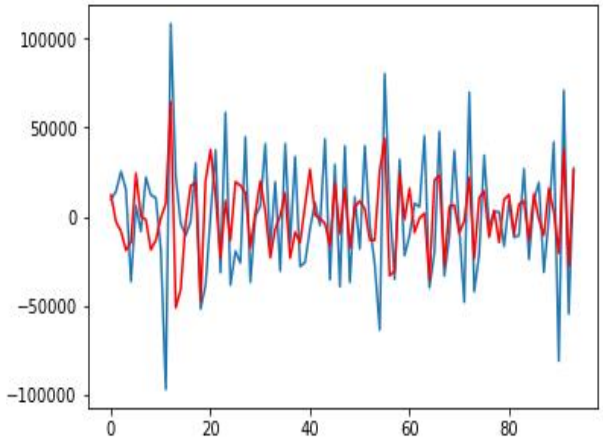


Fig. 3: Income Prediction Using Linear Regression.

4. Results and discussion

The following table shows the results obtained for various regression techniques applied on the given data. The actual test values are taken from the month of December 23 rd to 31st.linear regression technique goes well rather gradient boosting technique. The public revenue of the passengers also increases with respect to time more than 5%.

Table 1:ACTUAL Values and Predicted Values

Date	Actu-al_test_values	Linear Regres-sion_prediction	Gradientboost-ing predictions
12/23/2016	166008	164435.1661	168395.699
12/24/2016	185192	163435.5773	163972.3546
12/25/2016	154103	169703.8383	167979.489
12/26/2016	153403	168227.2649	168395.699
12/27/2016	195280	160182.3014	163636.2022
12/28/2016	114341	169248.7768	166158.1237
12/29/2016	185246	156614.6551	168395.699
12/30/2016	130696	157956.4619	163636.2022
12/31/2016	157947	153805.6969	173082.3882

5. Conclusion

The methods used for forecasting public revenue based on passenger data mainly focused on various regression techniques. In this paper we use linear regression and gradient boosting technique for forecasting. Linear regression gives better predicted values compared to gradient boosting technique. There is still need for refining the data and to achieve more accurate results for the development of transportation systems. The other techniques like random forest algorithm and neural networks can be applied and compared for better accurate results. The predicted data is in terms of revenue per year and by calculating the more accurate the data the prediction can be done effectively which helps in the analysis of the revenue of public transport system.

References

[1] Gharehchopogh, F.S., Mohammadi, P., & Hakimi, P. (2012). Application of Decision Tree Algorithm for Data Mining in Healthcare Operations: A Case Study. International Journal of Computer Applications, 52(6), 21-26. https://doi.org/10.5120/8206-1613.

[2] Draper, N. R., Smith, H., &Pownell, E. (1966). *Applied regression analysis* (Vol. 3). New York: Wiley.

[3] Han, Y. H., & Chen, C. (2011). Relationship between freight transportation and economic growth of China. *Journal of Beijing Institute ofTechnology, 20*, 101-106.

[4] Gurmu, K.Z.; Fan, W. 2014. Artificial Neural Network Travel Time Prediction Model for Buses using only GPS Data, Journal of Public Transportation, 17(2): 45-65. https://doi.org/10.5038/2375-0901.17.2.3.

[5] https://mubaris.com/2017/09/28/linear-regression-from-scratch.