TrafficTelligence:Advanced Traffic Volume Estimation with Machine Learning

By

(KRISHANMURTI KUMAR)

(RAJANI MUDIMADUGU)

(SHASHI KUMAR)

(VIKASH KUMAR UPADHAYA)

Guided by

Prof. Ms Raasha

A Dissertation Submitted to SRI VENKATESWARA COLLEGE OF ENGINEERING AND TECHNOLOGY, An Autonomous Institution affiliated to 'JNTU Ananthapur' in Partial Fulfilment of the Bachelor of Technology (***Computer Engineering***) with Specialization in ***Artificial Intelligence and Machine Learning.***

May 2024

SRI VENKATESWARA COLLEGE OF ENGINEERING AND TECHNOLOGY
R.V.S. Nagar Tirupathi Road, Andhra Pradesh– 517127

Model Building

The model building includes the following main tasks

- Training and testing the model
- Evaluation of Model
- Save the Model

➢ Training And Testing The Model

- Once after splitting the data into train and test, the data should be fed to an algorithm to build a model.
- There are several Machine learning algorithms to be used depending on the data you are going to process such as images, sound, text, and numerical values. The algorithms that you can choose according to the objective that you might have it may be Classification algorithms are Regression algorithms.
  1. Linear Regression
  2. Decision Tree Regressor
  3.Random Forest Regressor
  4.KNN
  5.svm
  5.xgboost

Steps in Building the model:-

Initialize the model -

```
from sklearn import linear_model
from sklearn import tree
from sklearn import ensemble
from sklearn import svm
import xgboost

lin_reg = linear_model.LinearRegression()
Dtree = tree.DecisionTreeRegressor()
Rand = ensemble.RandomForestRegressor()
svr = svm.SVR()
XGB = xgboost.XGBRegressor()
```

Fit the models with x_train and y_train -

```
lin_reg.fit(x_train,y_train)
Dtree.fit(x_train,y_train)
Rand.fit(x_train,y_train)
svr.fit(x_train,y_train)
XGB.fit(x_train,y_train)
```

Predict the y_train values and calculate the accuracy -

```
p1 = lin_reg.predict(x_train)
p2 = Dtree.predict(x_train)
p3 = Rand.predict(x_train)
p4 = svr.predict(x_train)
p5 = XGB.predict(x_train)
```

We're going to use the x-train and y-train obtained above in the train_test_split section to train our Random forest regression model. We're using the fit method and passing the parameters as shown below.

We are using the algorithm from Scikit learn library to build the model as shown below,

Once the model is trained, it's ready to make predictions. We can use the predict method on the model and pass x_test as a parameter to get the output as y_pred.

Notice that the prediction output is an array of real numbers corresponding to the input array.

➢ Model Evaluation:

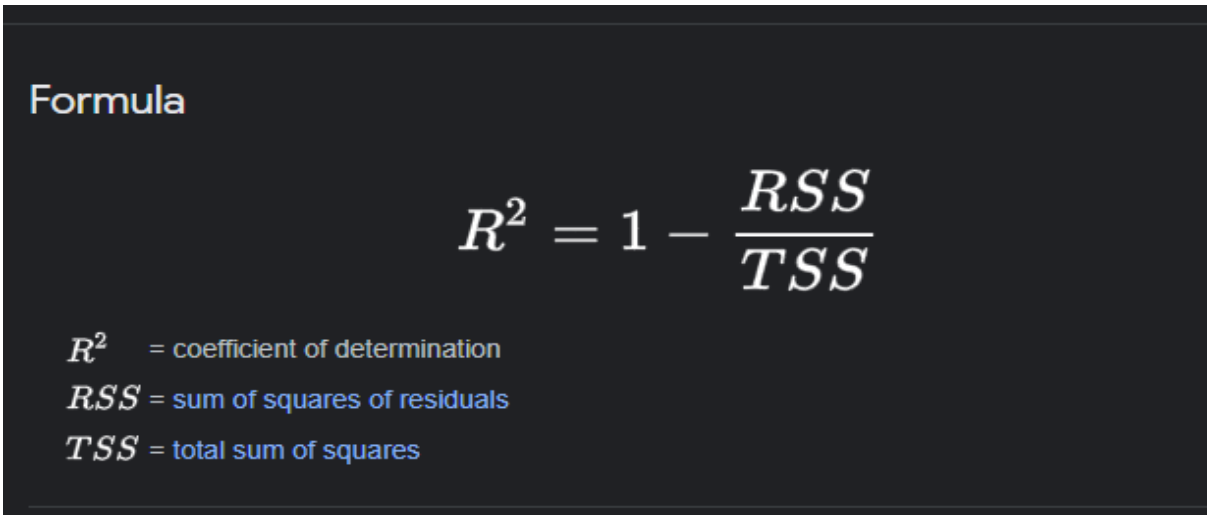
After training the model, the model should be tested by using the test data which is been separated while splitting the data for checking the functionality of the model.

Regression Evaluation Metrics:

These model evaluation techniques are used to find out the accuracy of models built in the Regression type of machine learning models. We have three types of evaluation methods.

- R-square_score
- RMSE – root mean squared error

1. R-squared _score -



It is the ratio of the number of correct predictions to the total number of input samples.

Calculating the r2 score value using for all the models.

```
from sklearn import metrics
```

```
print(metrics.r2_score(p1,y_train))
print(metrics.r2_score(p2,y_train))
print(metrics.r2_score(p3,y_train))
print(metrics.r2_score(p4,y_train))
print(metrics.r2_score(p5,y_train))
```

```
-5.517285423636865
1.0
0.9747969952887571
-12.188104231382285
0.8349874938269883
```

```
p2 = Dtree.predict(x_test)
p3 = Rand.predict(x_test)
p4 = svr.predict(x_test)
p5 = XGB.predict(x_test)
```

```
print(metrics.r2_score(p1,y_test))
print(metrics.r2_score(p2,y_test))
print(metrics.r2_score(p3,y_test))
print(metrics.r2_score(p4,y_test))
print(metrics.r2_score(p5,y_test))
```

```
-5.399396398322181
0.6920677009517378
0.8031828166614183
-11.972215715232434
0.7922184852381723
```

- After considering both r squared values of test and train we concluded that random forest regressor is giving the better value, it is able to explain the 97% of the data in train values.
- Random forest gives the best r2-score, so we can select this model.

RMSE –Root Mean Square Error

Formula

$$\mathrm{RMSD} = \sqrt{\frac{\sum_{i=1}^{N} (x_i - \hat{x}_i)^2}{N}}$$

RMSD = root-mean-square deviation
i = variable i
N = number of non-missing data points
x_i = actual observations time series
$\hat{x}_i$ = estimated time series

Randforest gives the best r-score value

```
#RMSE values
MSE = metrics.mean_squared_error(p3,y_test)
```

```
np.sqrt(MSE)
```

```
798.4970439382182
```

RMSE value for Random forest is very less when compared with other models, so saving the Random forest model and deploying using the following process

➢ Save The Model:

After building the model we have to save the model.

Pickle in Python is primarily used in serializing and deserializing a Python object structure. In other words, it's the process of converting a Python object into a byte stream to store it in a file/database,

maintain program state across sessions or transport data over the network. wb indicates write method and rd indicates read method.

This is done by the below code

```
import pickle

pickle.dump(Rand,open("model.pkl",'wb'))
pickle.dump(le,open("encoder.pkl",'wb'))
```