Report On

CAR POPULARITY PREDICTION

ABSTRACT:

Car Popularity prediction has been a high in interest research area, as it requires noticeable effort and knowledge of the field expert. Considerable number of distinct attributes are examined for the reliable and accurate prediction. To build a model for predicting the popularity of cars, we applied three machine learning algorithms [OneVsRest Classifier(Logistic Regression), Decision Tree and Random Forest]. However, the mentioned techniques were applied to work as an ensemble. Respective performances of different algorithms were then compared to find one that best suits the available data set.. Furthermore, the model was evaluated using test data and the accuracy of 84.3% was obtained.

BY

Dheeraj Pranav

TOOLS / SKILS USED

==========

Language Used : Python

Editor : Jupyter

Libraries Used:

matplotlib : https://matplotlib.org/api/pyplot_api.html

Seaborn : https://seaborn.pydata.org/

pandas : https://pandas.pydata.org/

Random Forest : https://scikit-learn.org/stable/modules/generated/sklearn.ensemble.RandomForestClassifier.html

Logistic Regression : https://scikit-learn.org/stable/modules/generated/sklearn.linear_model.LogisticRegression.html

Decision Tree : https://scikit learn.org/stable/modules/generated/sklearn.tree.DecisionTreeClassifier.html

Metrics:

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.classification_report.html

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.accuracy_score.html

https://scikit-learn.org/stable/modules/generated/sklearn.metrics.confusion_matrix.html

Problem Statement :

INTRODUCTION

A car company has the data for all the cars that are present in the market. They are planning to introduce some new ones of their own, but first, they want to find out what would be the popularity of the new cars in the market based on each car's attributes. We have a dataset of cars along with the attributes of each car along with its popularity. Here task is to train a model that can predict the popularity of new cars based on the given attributes.

Car popularity prediction involves expert knowledge, because popularity (rating) usually depends on many distinctive features and factors. Typically, most significant ones such as safety rating and number of doors highly affect popularity of a car due to frequent changes.

Different features like maintenance cost, luggage boot size, buying price, and number of seats will also influence the car popularity.

DATA DESCRIPTION

Our train dataset contained 1628 rows and 7 columns, and test data contained 327 rows and 6 columns.

Data to predict – popularity (of cars) RATING of (1,2, 3 ,4)

Predictors – The following attributes were captured for each car

- buying_price: The buying_price denotes the buying price of the car, and it ranges from [1,2,3,4].
- maintenance_cost: The maintenance_cost denotes the maintenance cost of the car, and it ranges from [1,2,3,4].
- number_of_doors: The number_of_doors denotes the number of doors in the car,they are(2,5,4).
- number_of_seats: The number_of_seats denotes the number of seats in the car, and it consists of [2, 4, 5]
- luggage_boot_size: The luggage_boot_size denotes the luggage boot size, and it ranges from [1,2,3].
- safety_rating: The safety_rating denotes the safety rating of the car, and it ranges from [1,2,3].

The goal is to predict the popularity of the car based on these attributes.

IMPLEMENTATION

DATA PRE-PROCESSING (LOOKING INTO DATA)

To preprocess the data we have done finding all kind of plots , finding the number of null values , and feature engineering.

Checking for null values - As we see into data , we check for null values – checking at the whole datafound no null values in any column. So no missing values in this train set.

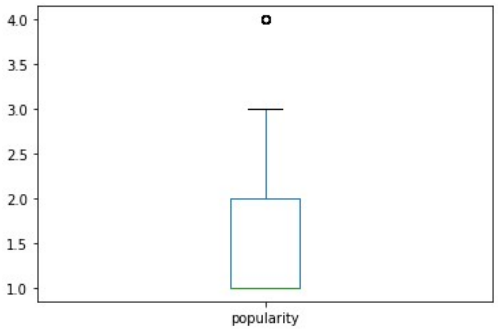
Checking for errors – As we see that the values can't be out of the range that is described for each and every figure, and for graph we noticed that dataset set does not contain any erroneous value.

Checking for outliers – As we see the boxplot of the popularity, we figure out that nothing can be judged as outlier.

```
In [7]: #check for null values

        df.isnull().sum()

Out[7]: buying_price        0
        maintainence_cost   0
        number_of_doors     0
        number_of_seats     0
        luggage_boot_size   0
        safety_rating       0
        popularity          0
        dtype: int64
```



We see that the popularity 4 may seem like an outlier but it is not, rather it is a data of great importance with less number of occurences.

As we have done the analysis of features so now we will be focusing on other parts.

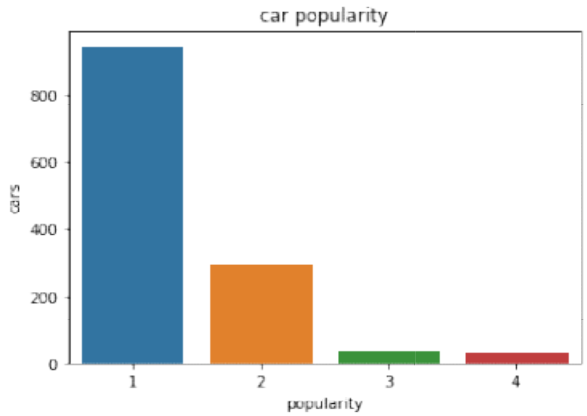
Encoding Features –

We see that in the dataset the features are in numeric form and have precedence amongst them(even when they are categorical) so no need of using one hot encoding.

VISUALIZATIONS

Bar chart representation of popularity meter

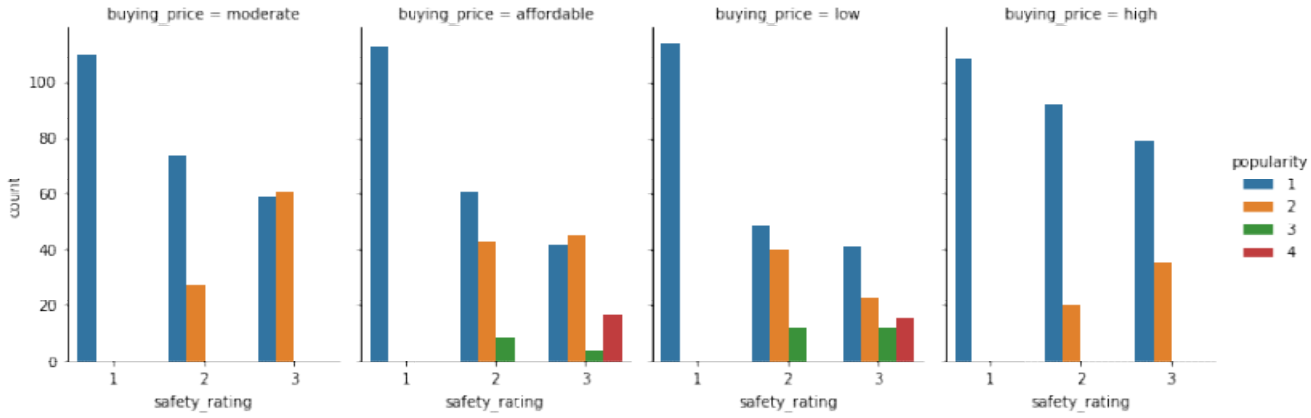
Checking data distribution – Popularity

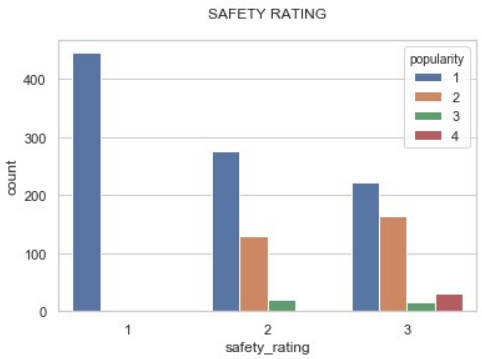



```
In [101]: train.popularity.describe()
Out[101]:
count    1628.000000
mean        1.348280
std         0.654766
min         1.000000
25%         1.000000
50%         1.000000
75%         2.000000
max         4.000000
Name: popularity, dtype: float64
```

So we see that maximum of our input data is of the popularity range 1 , and very few of the data are of popularity 3 or 4. By describing our column is also similar to this.

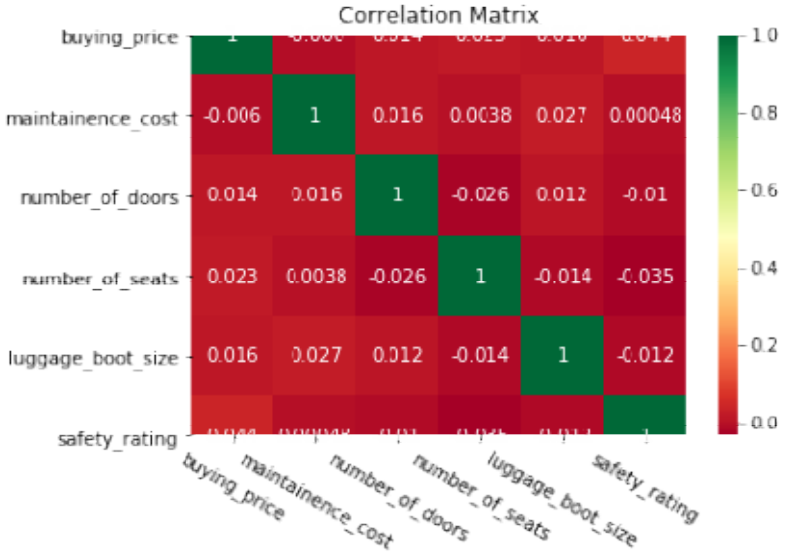
Relation of other features with popularity.





CORRELATION :

Clearly, “safety_rating" has a high correlation with “popularity” , compared to others.



SNIPPETS

Train Test Split & Modeling

The whole data set collected in this research has been split into training (80%) and testing (20%) subsets and Logistic Regression(oVr) , Decision Tree and Random Forest classifiers models were built.

```
l=['Logistic Regression','Decision Tree Classifier','RandomForestClassifier']
# prepare models
models = []
models.append(('LR', LogisticRegression(random_state=0, multi_class='multinomial',
models.append(('Decision Tree', DecisionTreeClassifier()))
models.append(('RandomForest',RandomForestClassifier(random_state = 8)))
# evaluate each model in turn
results = []
names = []
print('-------------ACCURACY SCORE-----------------')
for name, model in models:
    model.fit(X_train,Y_train)
    y_pred_class = model.predict(X_test)
    acc=metrics.accuracy_score(Y_test, y_pred_class)
    results.append(acc)
    names.append(name)
    print('='*40)
    print("%s: %f" %(name, acc))
print('\n')
```

```
In [19]: ovr.fit(X_train,Y_train)

Out[19]: OneVsRestClassifier(estimator=LogisticRegression(C=1.0, class_weight=None,
                                                           dual=False, fit_intercept=True,
                                                           intercept_scaling=1,
                                                           l1_ratio=None, max_iter=100,
                                                           multi_class='warn',
                                                           n_jobs=None, penalty='l2',
                                                           random_state=None,
                                                           solver='warn', tol=0.0001,
                                                           verbose=0, warm_start=False),
                                 n_jobs=None)

In [20]: y_train_pred=ovr.predict(X_train )
         y_test_pred=ovr.predict(X_test)
         from sklearn.metrics import *
```

```
# 20% of the data will be used for testing
X_train, X_test, Y_train, Y_test= train_test_split(X, Y, test_size=0.2,
```

```
logreg = LogisticRegression()
ovr = OneVsRestClassifier(logreg)
```

```
X_train.shape,Y_train.shape,X_test.shape,Y_test.shape
```

```
((1041, 6), (1041,), (261, 6), (261,))
```

CONCLUSION & RESULTS

This problem seems to be a classification problem as we have 4 classes, this seems to be a multi-class classification. Here we have got 6 features on the basis of which we have to select which which of the class out of the 4 options, this test data should fall into.

Looking at the data we can figure out that there would be a clear margin of separation between the points and here we have less number of training data so I chose One Vs rest Classifier (estimator = Logistic Regression) , since tree models Decision tree , Random Forest are having a chance of overfitting.

```
TEST DATA
=======================================
Accuracy: 0.842912
Precision: 0.40
Recall: 0.39
F1 score: 0.393636
---------------------------------------
              precision    recall  f1-score   support

           1       0.90      0.94      0.92       195
           2       0.69      0.62      0.65        58
           3       0.00      0.00      0.00         6
           4       0.00      0.00      0.00         2

    accuracy                           0.84       261
   macro avg       0.40      0.39      0.39       261
weighted avg       0.82      0.84      0.83       261

---------------------------------------
[[184   9   0   2]
 [ 20  36   2   0]
 [  1   5   0   0]
 [  0   2   0   0]]
```

```
-------------ACCURACY SCORE---------
===================================
LR: 0.865900
===================================
Decision Tree: 0.980843
===================================
RandomForest: 0.977011
```

FUTURE SCOPE

Although, this system has achieved astonishing performance in car popularity prediction problem our aim for the future research is to test this system to work successfully with various data sets. We will extend our test data with eBay [16] and OLX [17] used cars data sets and validate the proposed approach.

And also we can get a valid accuracy performance if we get other features (more) by using other algorithms such ANN etc.