Capstone Project

Car evaluation dataset

Machine Learning Engineer Nanodegree

Suthraye swamynath Rao

February 10th, 2019

1.Definition

Domain Background:

History:

Basically these are the steps for evaluating a car.by using this we can judge whether the car is worthy or not.

Step 1: Research the Car's VIN. Every car has a unique vehicle identification number (VIN). ...

Step 2: Informal Inspection. ...

Step 3: Test Drive. ...

Step 4: Professional Inspection

While coming to technical aspects these are the past usage

Car Evaluation Database was derived from a simple hierarchical decision model originally developed for the demonstration of DEX, M. Bohanec, V. Rajkovic: Expert system for decision making. Sistemica 1(1), pp. 145-157, 1990.).

M. Bohanec and V. Rajkovic: Knowledge acquisition and explanation for multi-attribute decision making. In 8th Intl Workshop on Expert Systems and their Applications, Avignon, France. pages 59-78, 1988.

Database reference:https://archive.ics.uci.edu/ml/datasets/Car+Evaluation

Problem Statement:

The problem statement is To evaluate a car based on a target concept (CAR), the model includes three intermediate concepts: PRICE, TECH, COMFORT

The target variable is CAR(car acceptability) which means whether the car is worthy or not.This can be done by intermediate concepts which I discribed below.By

The model evaluates

cars according to the following concept structure:

CAR	car acceptability
. PRICE	overall price
. . buying	buying price
. . maint	price of the maintenance
. TECH	technical characteristics
. . COMFORT	comfort
. . . doors	number of doors
. . . persons	capacity in terms of persons to carry
. . . lug_boot	the size of luggage boot
. . safety	estimated safety of the car

.By using this intermediate concepts we will make judgement about that car.Based on these concepts we will get a conclusion whether the car is good or bad.

Metrics

I want to use accuracy score as an evaluation metric for the prediction of credit approval. In the data set the class labels (+,-) are very closely balanced so we can use accuracy score as the evaluation metric. Here I am predicting the accuracy score of the selected models. We will select a model whose accuracy score is greater than all the other models and we treat it as the best.

For finding out the accuracy we have the following

formula: Accuracy Score= TP+TN/TP+FP+FN+TN

True Positives: are the values which are correctly predicted as positives True Negatives: are the values which are correctly classified as negatives. False Positives: are the values which are wrongly classified as positives. These are also type-1 errors.

False Negatives: are the values which are wrongly classified as negatives. These are also called as type-2 errors.

II.Analysis

Data Exploration:

In this dataset I have used 6attributes and 1728 instances to evaluate accuracy score.the attributes are shown below

	buying	maint	doors	persons	lug_boot	safety	class
0	vhigh	vhigh	2	2	small	low	unacc
1	vhigh	vhigh	2	2	small	med	unacc
2	vhigh	vhigh	2	2	small	high	unacc
3	vhigh	vhigh	2	2	med	low	unacc
4	vhigh	vhigh	2	2	med	med	unacc

Attribute information:

buying v-high, high, med, low
maint v-high, high, med, low
doors 2, 3, 4, 5-more
persons 2, 4, more
lug_boot small, med, big
safety low, med, high

Class Distribution (number of instances per class):

class	N	N[%]
unacc	1210	(70.023 %)
acc	384	(22.222 %)
good	69	(3.993 %)
v-good	65	(3.762 %)

This describes the distribution of target class(CAR acceptabilty)..The total 1728instances are divided into 4 classes as mentioned above.
Here I am using a Multivariate dataset.

Information of dataset:

```
In [4]: data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 1728 entries, 0 to 1727
Data columns (total 7 columns):
buying      1728 non-null object
maint       1728 non-null object
doors       1728 non-null object
persons     1728 non-null object
lug_boot    1728 non-null object
safety      1728 non-null object
class       1728 non-null object
dtypes: object(7)
memory usage: 108.0+ KB
```

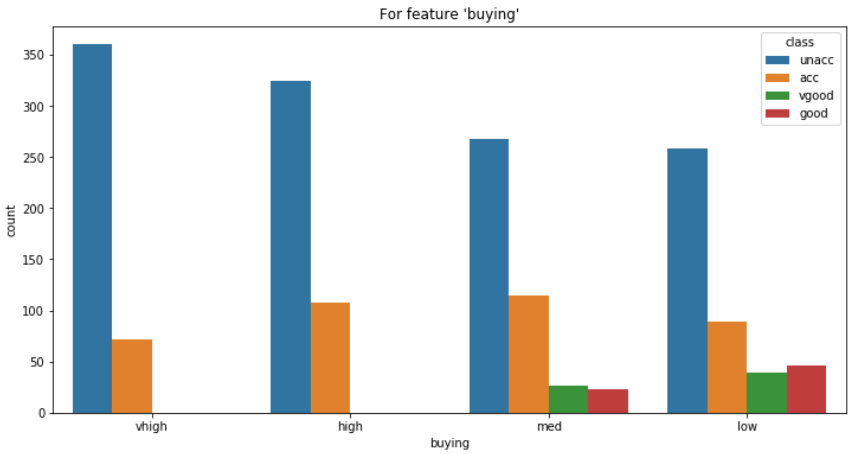
As all the columns are categorical, check for unique values of each column

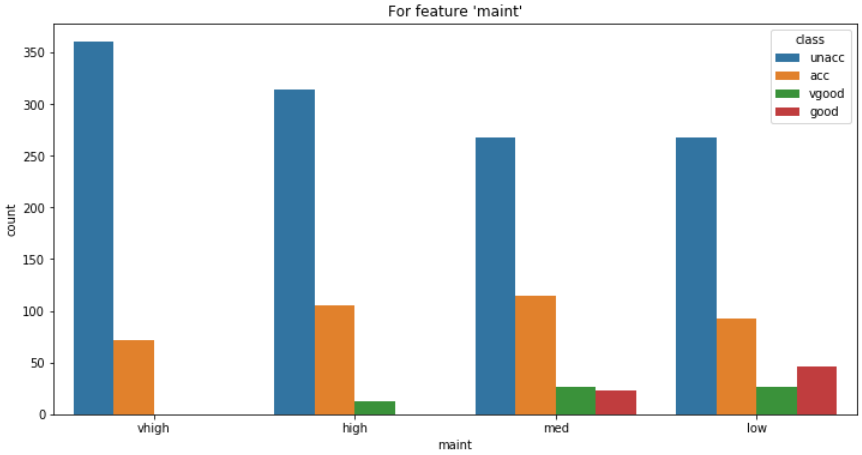
As I told above I have checked for unique values of each column because all the columns are categorical.

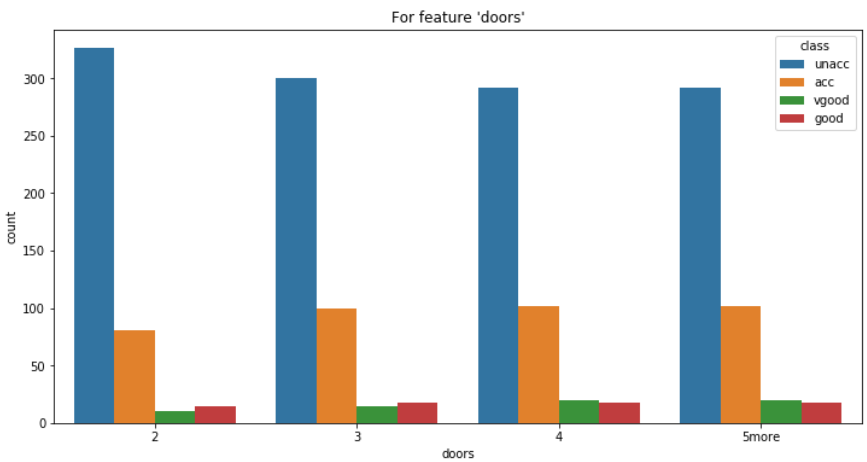
```
for i in data.columns:
    print(data[i].unique(),"\t",data[i].nunique())
```

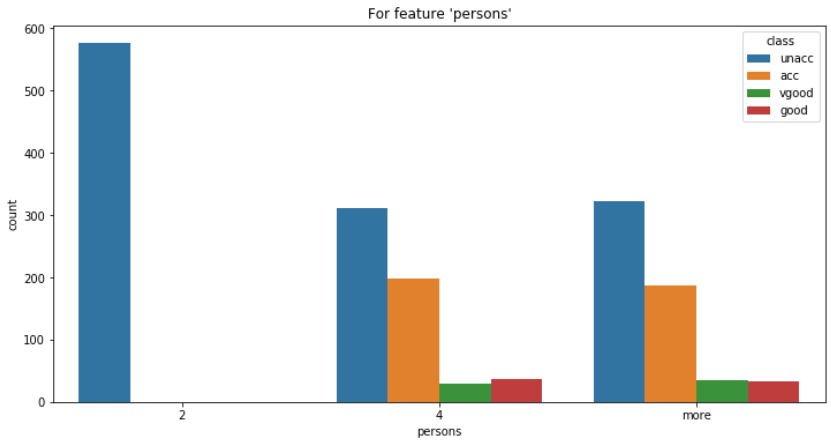
```
['vhigh' 'high' 'med' 'low']     4
['vhigh' 'high' 'med' 'low']     4
['2' '3' '4' '5more']    4
['2' '4' 'more']         3
['small' 'med' 'big']    3
['low' 'med' 'high']     3
['unacc' 'acc' 'vgood' 'good']   4
```

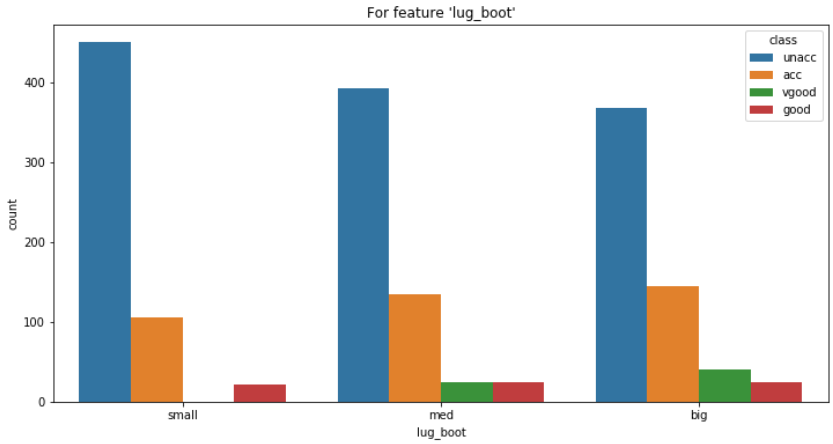
DATA Visualization:

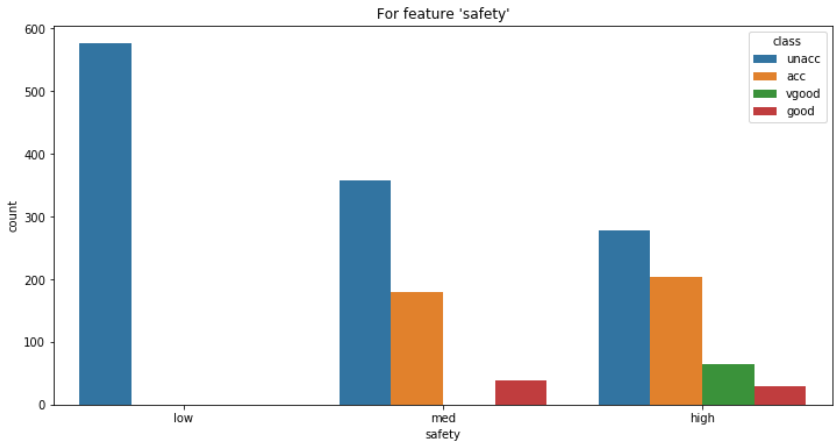

For feature 'buying'
class
unacc
acc
vgood
good
count
buying
vhigh
high
med
low
0
50
100
150
200
250
300
350


For feature 'maint'
class
unacc
acc
vgood
good
count
maint
vhigh
high
med
low
0
50
100
150
200
250
300
350


For feature 'doors'
class
unacc
acc
vgood
good
count
doors
2
3
4
5more
0
50
100
150
200
250
300







These shows the co-relation between features and taget class.Each feature will be in relation with target class diffrently

Heat map:

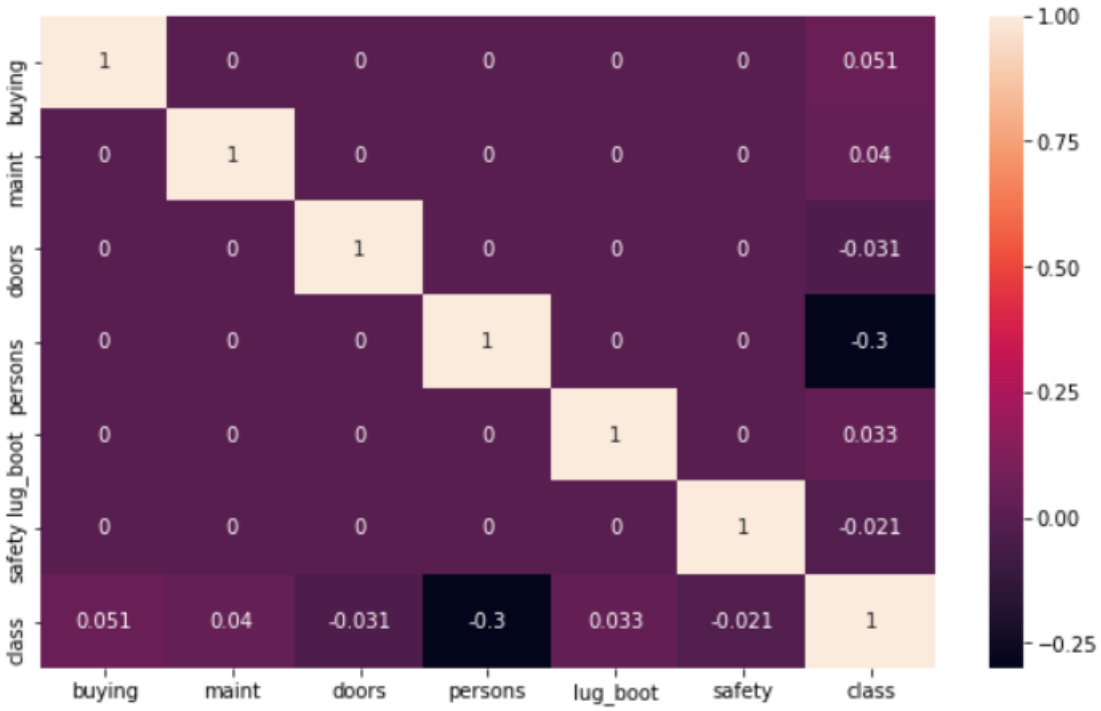
The heat map is a 2-D representation of data in which values are represented by

colours. A simple heat map provides the immediate visual summary of information. More elaborate heat maps allow the user to understand complex data.

By using the below code we will generate the heat map between the attributes and try to deduce the correlation between the attributes. We will use the following code to do so.

```
fig=plt.figure(figsize=(10,6))
sns.heatmap(data.corr(),annot=True)
```

By executing the above code we get the following heat map:



Ignoring the diagonal values, it can be seen that most of the columns shows very weak correlation with 'class'. 'persons' column is showning a weak relation with 'class'. Other columns except 'class' shows no correlation with each other.

Modelling and Predicting with Machine Learning:

The main goal of the entire project is to evaluate car based on features with the highest accuracy. In order to achieve this, we will test several classification algorithms. This section includes all results obtained from the study and introduces the best performer according to accuracy metric. I have chosen several algorithms typical for solving supervised learning problems throughout classification methods.

First of all, let's equip ourselves with a handy tool that benefits from the cohesion of SciKit Learn library and formulate a general function for training our models. The reason for displaying accuracy on both, train and test sets, is to allow us to evaluate whether the model overfits or underfits the data (so-called bias/variance tradeoff).

Algorithms and techniques:

Here mainly we will use three algorithms.

1. Logistic Regression(Benchmark Model)
2. KNN classifier.
3. Random Forest.

Logistic Regression:

Logistic Regression is a Machine Learning classification algorithm that is used to predict the probability of a categorical dependent variable. In logistic regression, the dependent variable is a binary variable that contains data coded as 1 (yes, success, etc.) or 0 (no, failure, etc.). In other words, the logistic regression model predicts P(Y=1) as a function of X.

Advantages:

Because of its efficient and straightforward nature, doesn't require high computation power, easy to implement, easily interpretable, used widely by data analyst and scientist. Also, it doesn't require scaling of features. Logistic regression provides a probability score for observations.

Disadvantag:

Logistic regression is not able to handle a large number of categorical features/variables. It is vulnerable to over fitting. Also, can't solve the non-linear problem with the logistic regression that is why it requires a transformation of non-linear features. Logistic regression will not perform well with independent variables that are not correlated to the target variable and are very similar or correlated to each other.

Sample code:

```
from sklearn.model_selection importlearning_curve,cross_val_score,validation_curve

param_range=[0.0001,0.001,0.1,1]

curve=validation_curve(logreg,X_train,y_train,cv=5,param_name='C',

param_range=param_range,n_jobs=-1,)
```

The application is classification oriented . So, techniques that are used are taken from Classification techniques

1. K-Nearest Neighbours (KNN):

K-Nearest Neighbours algorithm is a non-parametric method used for classification and regression. The principle behind nearest neighbour methods is to find a predefined number of training samples closest in distance to the new point and predict the label from these.

Sample code:

```
knn=KNeighborsClassifier(n_jobs=-1)
knn.fit(X_train,y_train)
pred=knn.predict(X_test)
knn.score(X_test,y_test)
```

Advantages:

The K-Nearest Neighbor (KNN) Classifier is a very simple classifier that works well on basic recognition problems.

Disadvantages:

The main disadvantage of the KNN algorithm is that it is a *lazy learner*, i.e. it does not learn anything from the training data and simply uses the training data itself for classification.

To predict the label of a new instance the KNN algorithm will find the *K* closest neighbors to the new instance from thetraining data, the predicted class label will then be set as the most common label among the *K* closest neighboringpoints.

The algorithm must compute the distance and sort all the training data at each prediction, which can be slow if there are a large number of training examples.

Another disadvantage of this approach is that the algorithm does not learn anything from the training data, which can resultin the algorithm not generalizing well and also not being robust to noisy data.

Random Forest

Tree models are known to be high variance, low bias models. In consequence, they are prone to over fit the training data. This is catchy if we recapitulate what a tree model does if we do not prune it or introduce early stopping criteria like a minimum number of instances per leaf node. Well, it tries to split the data along the features until the instances are pure regarding the value of the target feature, there are no data left, or there are no features left to spit the dataset on. If one of the above holds true, we grow a leaf node. The consequence is that the tree model is grown to the maximal depth and therewith tries to reshape the training data as precise as possible which can easily lead to over fitting. Another drawback of classical tree models like the (ID3 or CART) is that they are relatively unstable. This instability can lead to the situation that a small change in the composition of the dataset leads to a completely different tree model.

Sample code:

```
from sklearn.ensemble import RandomForestClassifier

rfc=RandomForestClassifier(n_jobs=-1,random_state=51)


from sklearn.metrics import f1_score

rfc.fit(X_train,y_train)

print(rfc.score(X_test,y_test))

print(f1_score(y_test,rfc.predict(X_test),average='macro'))
```

Advantages:

1. Reduction in over fitting: by averaging several trees, there is a significantly lower risk of over fitting.
2. Less variance: By using multiple trees, you reduce the chance of stumbling across a classifier that doesn't perform well because of the relationship between the train and test data.

Disadvantages:

1. It takes more time to train samples.

Benchmark Model:

For this problem we will choose Logistic regression model as the benchmark model. By applying this logistic regression we achieved an accuracy of 0.718 that is 71.8%. Now we will try and achieve better accuracy than this model by using the above

mentioned classification models.

III. Methodology

Data Pre-processing:

In this step we will pre process the data. Data pre-processing is considered to be the first and foremost step that is to be done before starting any process. We will read the data by using read_csv. Then we will know the shape of the data. After that we will know the unique class attributes by using unique (). And by using the info () we will know the information of the attributes. Then we will check whether there are any null values by using isnull ().

LabelEncoder can be used to normalize labels. We will import LabelEncoder from

sklearn.preprocessing we will also use fit_transform(y) for Fit label encoder and return encoded labels. After doing that we will use le.transform () for Transform labels to normalized encoding.

After that we will divide the whole data into training and testing data. We will assign 70% of the data to the training data and the remaining 30% of the data into testing data. We will do this by using train_test_split from sklearn.model_selection.

Implementation

Out of the chosen algorithms we will first take KNN classifier model. We will take a classifier and fit the training data. After that we will predict that by using predict (X_train). Now we will predict the accuracy of the testing data by using accuracy_score (y_test, pred).

By doing so for the Logistic regression which is the benchmark will give us the accuracy of 0.71.

We will now choose the algorithm which will give us the better score than the benchmark model out of KNN classifier and Random Forest . By following the same procedure above that is fitting, predicting and finding the accuracy score we will get the accuracy score as bellow.

KNN classifier:0.90

Random Forest: 0.984

From the above reports Random Forest seems to be performing well.

Refinement

I found out random forest as the best classifier out of the chosen classifiers. Now we will perform tuning of random forest classifier in order to achieve the better accuracy. For this we will use GridSearchCV.

For doing refinement we will just tune the parameter. Here we will assign n_estimators=[50] and 'criterion': ['gini', 'entropy']. Now will find out the new accuracy. The new accuracy will be 0.984. This tuned accuracy will be a little bit more than the untuned accuracy.

IV. Result

Model evaluation and validation

The final model we have chosen is tuned random forest which gave us more accuracy that is 0.984. In order to achieve this accuracy we assigned n_estimators=[50] and 'criterion': ['gini', 'entropy'] but without using the n_estimators and criterion we achieved the accuracy of only 0.977 which is a bitless than the tuned value. Here we can say that the solution is reasonable because we are getting much more less accuracy while using other models.

The final model that is tuned random forest has been tested with various inputs

to evaluate whether the model generalises well. This model is also robust enough for the given problem.

Our data already has less features and even if we drop the least important feature, then also the accuracy is reducing to 93.64%

Justification

My final model's solution is better than the benchmark model.

Tuned	random forest	Benchmark model
Accuracy	0.984	0.71

From the above we can conclude that the results for the final model are stronger than the benchmark model.

Hence we can say that tuned random forest provides the significant to solve the problem of predicting the outcome of the car evaluation

V. Conclusion

The goal of the project was to compare different machine learning algorithms and predict the car is worthy or not l by using different

featureslike' buying,maintenance,doors,persons,lug_boot,safety,class

ACCURACY

Logistic regression	0.71
KNN	0.90

Random Forest	0.984

Form the results we can easily see that the Random forestl has the highest accuracy score of 0.9917 compared with the other models. By producing decent results, simpler methods proved to be useful as well

Reflection:

1. I have learnt how to visualize and understand the data.

2. I have learnt that the data cleaning place a very vital role in data analytics.

3. Removing the data features which are not necessary in evaluating model is very important.

4. I got to know how to use the best technique for the data using appropriate ways

5. I got to know how to tune the parameters in order to achieve the best score.

6. On a whole I learnt how to graph a dataset and applying cleaning techniques on it and to fit the best techniques to get best score.

Improvement:

The process which I have followed can be improved to classify not only for cars but can be extended to automobile industry which consists large vehicles also. Some Features will be added based on that vehicle.As we can say that there has never been an end in the machine learning there will be many more models to learn. By taking more amount of datasets, we can get the model more generalized and optimized to get better accurate result