Mathematical Essay on random forest Classifier

Chaithanya Krishna Moorthy
Dept. of Physics
Indian Institute of Technology, Madras
Chennai, India
ph17b011@smail.iitm.ac.in

***Abstract*—In this assignment, I have used the random forest Classifier on a dataset containing the features of samples of a car and predict its degree of acceptance. The random forest model works very well and gives an accuracy of 98% on unseen data.**

***Index Terms*—Random Forests, Classification, Machine Learning, Supervised Learning**

I. Introduction

Classification models can be used in cases when gathering enough data points can be expensive. An example of this is judging if a car is acceptable in terms of safety. Performing multiple crash tests can be very expensive and hence, using a machine learning to classify can he helpful.

Random Forest Classifier is classification technique that builds an ensemble tree based on the decision principles learned from the training model. Random forests are very popular because of their interpretability, high performance and usability on large datasets.

We can use random forest for the classification task at hand - to build a classifier for quality acceptance of cars based on their features, such as safety degree, price, capacity and maintenance costs. Doing so, we can understand the criterion that decide if a car is of good quality or not and also understand which features have the most weight in this decision.

In this study, the a Random Forest is built for the classification of cars based on its features. Section II will be on the principles of random forests, section III will include Data Cleaning, Exploratory Data Analysis and applying the Random Forest Classifier on the cleaned data. Section IV will be conclusions and the study will end with references.

II. Principles of the Random Forest Classifier

Random Forest classifier builds an ensemble of decision trees. Individual trees are built by

- **Bagging** - From the total data examples, data points are sampled with replacement and are used to train each tree.
- **Feature Randomness** - from the set of all features, a subset of features are chosen to be input to each tree and train it

The above two methods decrease the correlation between any pair of trees in the forest. The uncorrelated trees give better performance.

Each tree gives a class as an output. The majority vote from all the trees in the ensemble is decided as the final output. Errors committed by a group of trees in the ensemble is nullified by another group, enhancing the performance of the model as a whole. An example of a random forest is shown in figure 1.

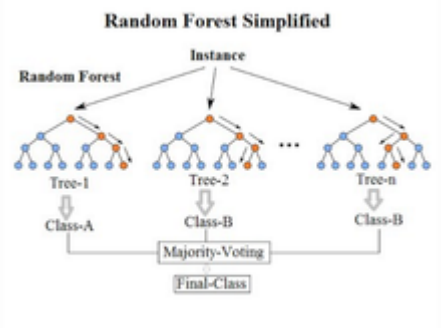

Fig. 1. Pictorial representation of a Random Forest Classifier [6]

For classification, the features are split accordingly to maximize the Gini index/Entropy. The splitting is stopped according to the criterion - the leaf nodes contain pure / majority class examples.

A. Advantages of Random Forests

- One of the best performing models among the current methods
- Work well on large datasets
- Can be used for both regression and Classification tasks
- Indicate importance of each feature in the model
- Usable in cases of missing data
- Do not overfit
- Can handle large number of features

The hyperparameters in Random Forests are number of leaf nodes, maximum depth of trees, node split criterion.

B. Metrics for Evaluation

Accuracy is defined as:

Accuracy = Number of correct predictions from the model/Total number of predictions

Accuracy is a good measure to see if our model is able to make correct predictions, implying that the input features influence the output features.

Feature	Description	Key
buying	buying price	vhigh, high, med, low
maint	Price of the maintenance	vhigh, high, med, low
doors	Number of doors	2, 3, 4, 5, more
persons	Capacity in terms of persons to carry	2, 4, more
lug_boot	The size of luggage boot	small, med, big
safety	Estimated safety of the car	low, med, high
Target	Target variable to predict	unacc, acc, good, vgood

TABLE I
FEATURES IN THE DATASET

param_max_depth	param_n_estimators	mean_test_score
15	200	0.972
50	100	0.972
15	500	0.972
50	50	0.971
50	500	0.971
15	100	0.971
15	500	0.971
50	200	0.970
15	200	0.970
15	100	0.970
15	50	0.967
15	50	0.961

TABLE II
GRIDSEARCH ON THE HYPERPARAMERS - NO. OF ESTIMATORS AND MAXIMUM DEPTH

III. USING RANDOM FOREST CLASSIFIER FOR THE DATASET

The dataset contains example cars with their features specified and are labeled if they are acceptable or not, and how good they are, if accepted (4 classes)[5]. The data has 1728 rows and contains the columns as is table 1. The dataset was divided in 80-20 ratio as training-test set to evaluate the performance of the model on new data.

1) Data cleaning and Exploratory Data Analysis: Pie plots of all the features that are categorical is as in figure 2. The data contains ordinal variables which were converted by OrdinalEncoder of scikit-learn module in Python to numerical values which are ordered (0,1,2..).

2) Correlations: The target features for the study is 'target', which indicates the degree of acceptance level of the car. Rest of the columns are taken as features.

The correlation matrix heatmap for every pair of columns is shown in figure 4. As seen in the figure, the correlations between the figures is very low.

3) Applying Random Forest Classifier: A random forest classifier model was built using sklearn.ensemble's RandomForestClassifier module. Hyperparameter tuning was done with features 'n_estimators' and 'max_depth' to get the highest accuracy possible on the Cross-Validation set (5-fold Cross-Validation was performed to choose the best model). The results are summarized in table 2. The weight of each feature in the final model is shown in figure 3. The accuracy on the training model obtained is 1.00

The model was used on test data and the accuracy 0.97

4) Observations:

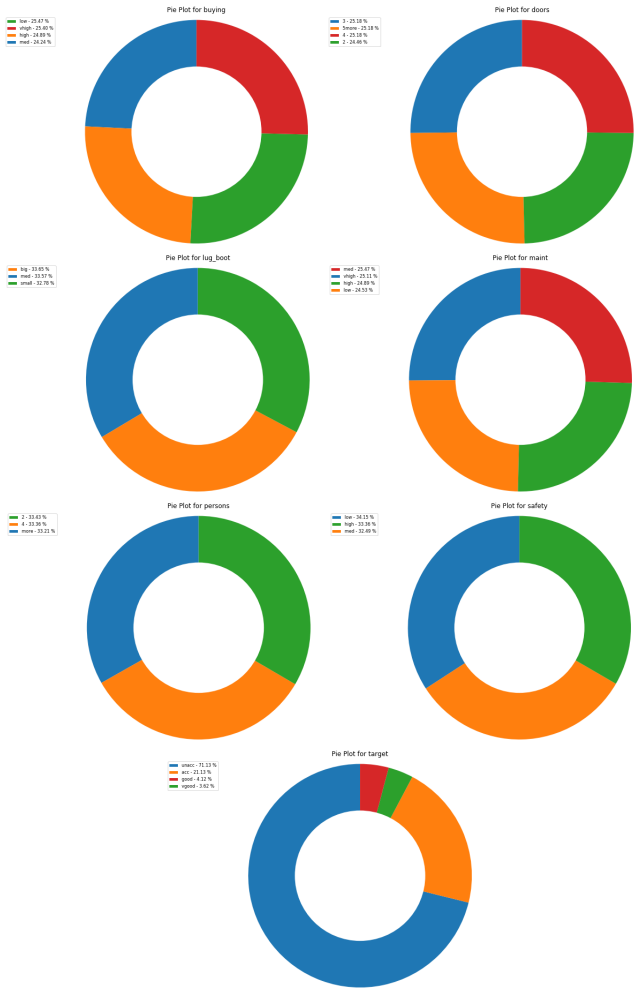

Fig. 2. Pie charts of categorical features and target variable, showing their size in the total dataset

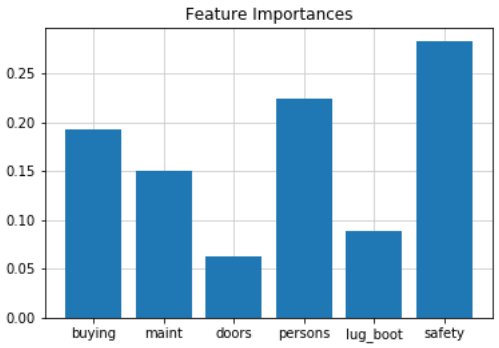

Fig. 3. Feature importance of each figure input to the model, in the final model

- As we can see in figure 2, the fractions of nominal values in the features is almost equal, but there is some imbalance in the target variable's class frequencies.
- The accuracy obtained by the random forest implementation is slightly lesser than the accuracy obtained using Decision Tree model.

IV. Conclusions

From the model that was built, we can see that the random forest classifier performed exceptionally well for this dataset, giving accuracy of 100% on the the training and 97% on the test dataset. This is slightly lesser than the one obtained from Decision Tree Classifier. A possible explanation for this can be that since the dataset used is relatively simple, the simpler Decision Tree model works better.

V. References

[1] Waskom, L.: seaborn: statistical data visualization, *Journal of Open Source Software, v6* [2] Pedregosa, F. and Varoquaux, G. and Gramfort, A. and Michel, V. and Thirion et al.: Scikit-learn: Machine Learning in Python, *Journal of Machine Learning Research, v12* [3] Hunter J. D: Matplotlib: A 2D graphics environment, *Computing in Science & Engineering, v9*

[4] Andreas C. Müller and Sarah Guido: Introduction to Machine Learning with Python: A Guide for Data Scientists [5] Sources: (a) Creator: Marko Bohanec (b) Donors: Marko Bohanec (marko.bohanec@ijs.si) Blaz Zupan (blaz.zupan@ijs.si) [6] https://en.wikipedia.org/wiki/Random_forest [7] https://towardsdatascience.com/understanding-random-forest-58381e0602d2

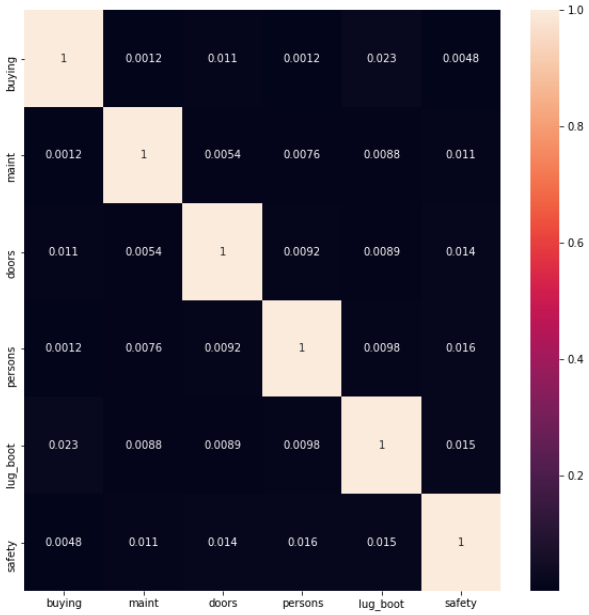

Fig. 4. Correlation between all features