Yaofei Zhou
Nuid:001151109

Final Report

Background:

In machine learning, multiclass or multinomial classification is the problem of classifying instances into more than two classes.

There are many approaches be able to do multiclass classification like:
Single classifier:

- Decision tree.
- AdaBoost-type algorithm.
- SVM-type algorithm.

Combination of binary classifiers:

- One-vs-all (e.g. naive Bayes)
- One-vs-one
- Error-correcting codes

Project Goal:

Implementing decision tree algorithm and then applying pruning decision tree strategy to car evaluation data set to classify cars of different conditions as acceptable, good, very good or unacceptable class.

Algorithm Implementation:

Decision tree algorithm

Decision tree learning uses a decision tree as a predictive model which maps observations about an item to conclusions about the item's target value. It is one of the predictive modelling approaches used in statistics, data mining and machine learning.

More descriptive names for such tree models are classification trees or regression trees. In these tree structures, leaves represent class labels and branches represent conjunctions of features that lead to those class labels.

Basic Review of Entropy

- Entropy (Information Theory)
  A measure of uncertainty (impurity) associated with a random variable.
  $H(Y) = -\sum_{i=1}^{m} p_i \log(p_i)$, where $p_i = P(Y = y_i)$

- Conditional Entropy

$$H(Y|X) \equiv \sum_{x \in \mathcal{X}} p(x)\, H(Y|X = x)$$

- Mutual Information (Information gain)

$$\begin{aligned} I(X;Y) &= H(X) - H(X|Y) \\ &= H(Y) - H(Y|X) \end{aligned}$$

Create Decision Tree

1. Tree is constructed in a top-down recursive divide-and-conquer manner.
2. At start, all the training examples are at the root
3. Attributes are categorical
4. Examples are partitioned recursively based on selected attributes
5. Test attributes are selected on the basis of a heuristic or statistical measure
   - ➢Information gain measure is biased towards attributes with a large number of values
   - ➢Using **information gain ratio** to overcome the drawback of information gain

Definition:

$$\textit{Information gain ratio} = \frac{Information\ Gain}{SplitInfo}$$

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

Conditions for stopping partitioning

1. All samples for a given node belong to the same class
2. There are no remaining attributes for further partitioning – **majority voting** is employed for classifying the leaf
3. There are no samples left

Tree Pruning Strategy

After decision tree has been created, there is an overfitting problem. The decision tree may have a perfect fit for training data, but when a test data come it could be diffcult for that overfitted decision tree to do classification.

To overcome the over-fitting problem, we need to do pruning, collapse some node into leaf node based on some strategy. Here I choose **Pessimistic Error Pruning**.

We use inequity for each non-leaf node, if it satisfy this inequity, we do pruning.

$$e'(t) \le e'(T_t) + S_e\big(e'(T_t)\big),$$

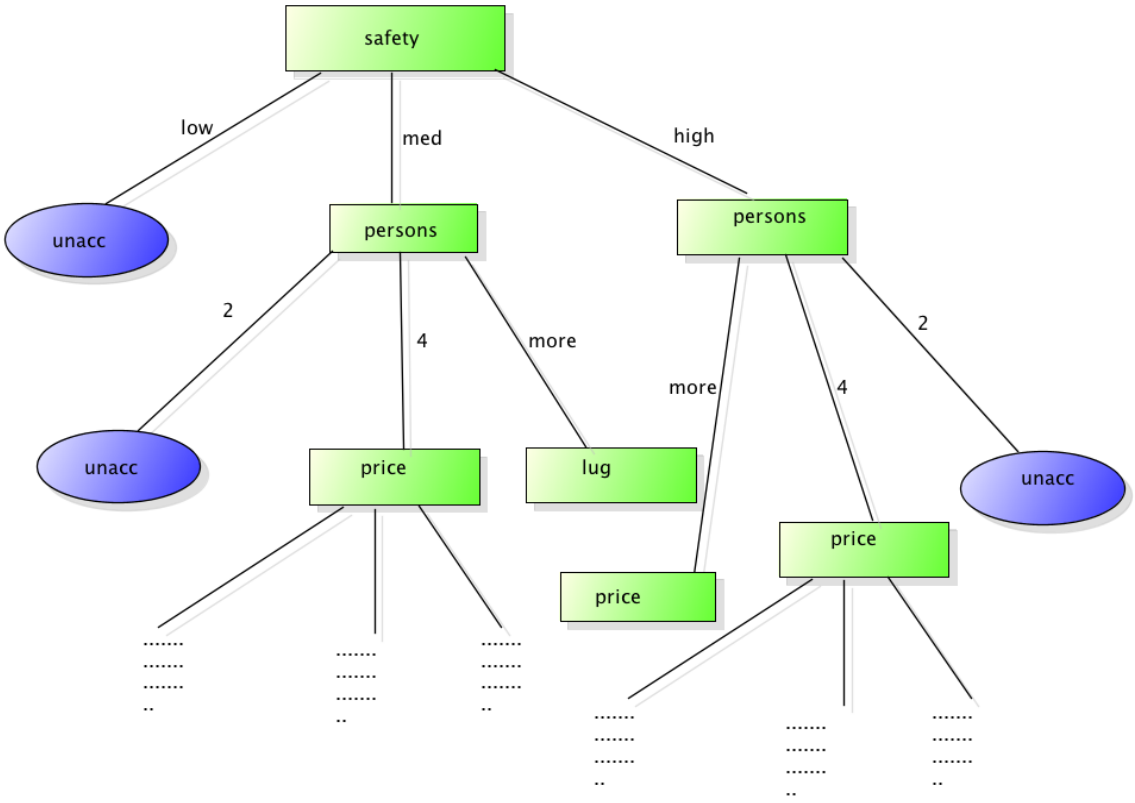
where $e'(t) = \left[e(t) + \frac{1}{2}\right]$,

$e'(T_t) = \sum e(i) + N_t/2$,

$S_e\big(e'(T_t)\big) = [e'(T_t)\frac{n(t) - e'(T_t)}{n(t)}]^{1/2}$

$e(t)$ is the error number at node t, e(i) is error number of leaf under subtree T_t. N_t is number of leaf in the subtree. n(t) is training instance in the node t.

A brief Structure



Dataset:

Car Evaluation Data Set (from UCI repository) http://archive.ics.uci.edu/ml/datasets/Car+Evaluation

Each record has the following attribute:

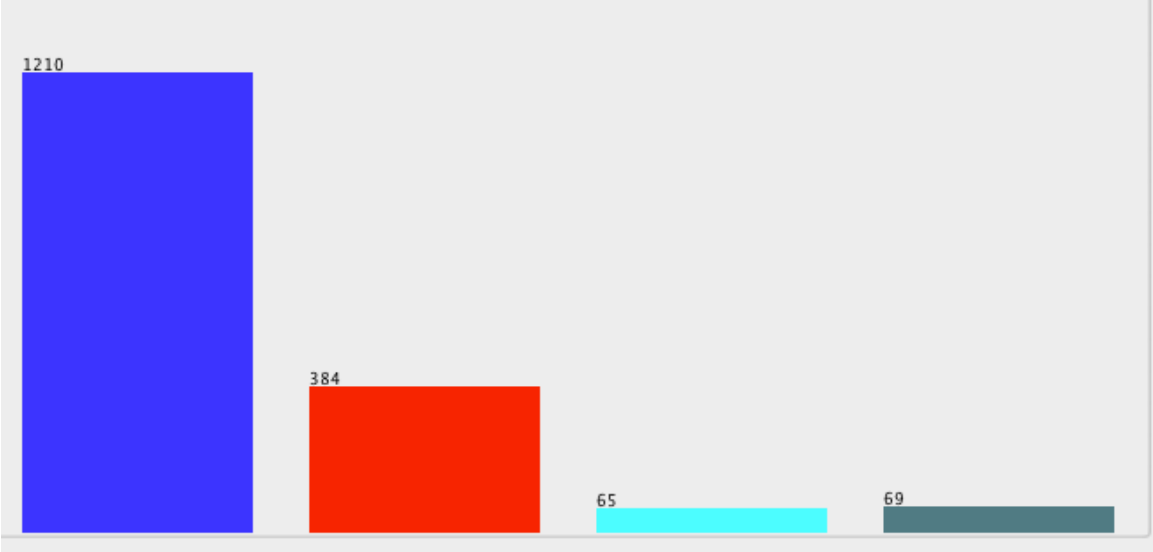
buying: vhigh, high, med, low.
maint: vhigh, high, med, low.
doors: 2, 3, 4, 5more.
persons: 2, 4, more.
lug_boot: small, med, big.
safety: low, med, high.

Sample:

vhigh,vhigh,2,2,small,low,unacc
vhigh,vhigh,2,2,small,med,unacc
vhigh,vhigh,2,2,small,high,unacc

vhigh,vhigh,2,2,med,low,unacc
.
.
.
.

Historgram show the distribution of class label:

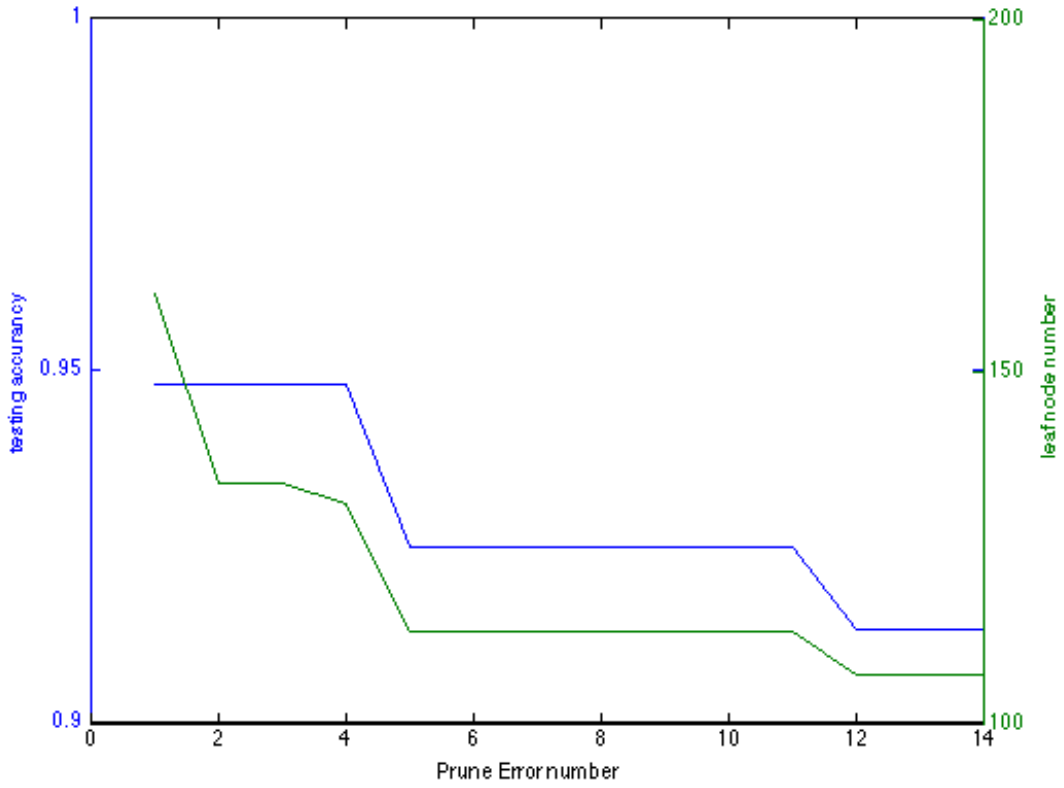


Analysis:

Testing Accuracy with different pruning error number tolerance. When doing pruning, we can set a threshold for how many error number increase we can tolerate after collapsing a node.

Prune Error number	Testing Accuracy	Leaf Node Number
1	0.947977	161
2	0.947977	134
3	0.947977	134
4	0.947977	131
5	0.924855	113
6	0.924855	113
7	0.924855	113
8	0.924855	113
9	0.924855	113
10	0.924855	113
11	0.924855	113
12	0.913295	107

13	0.913295	107
14	0.913295	107



As we can see, when prune error number increases, the testing accuracy and leaf nodes of decision tree deceases.

Confusion Matrix & Measurement

A Confusion Matrix is a very practical and intuitive way of seeing such a distribution. Given test examples and a classifier, an example of a Confusion Matrix for four classes {unacc, acc, vgood, good} might be as follows:

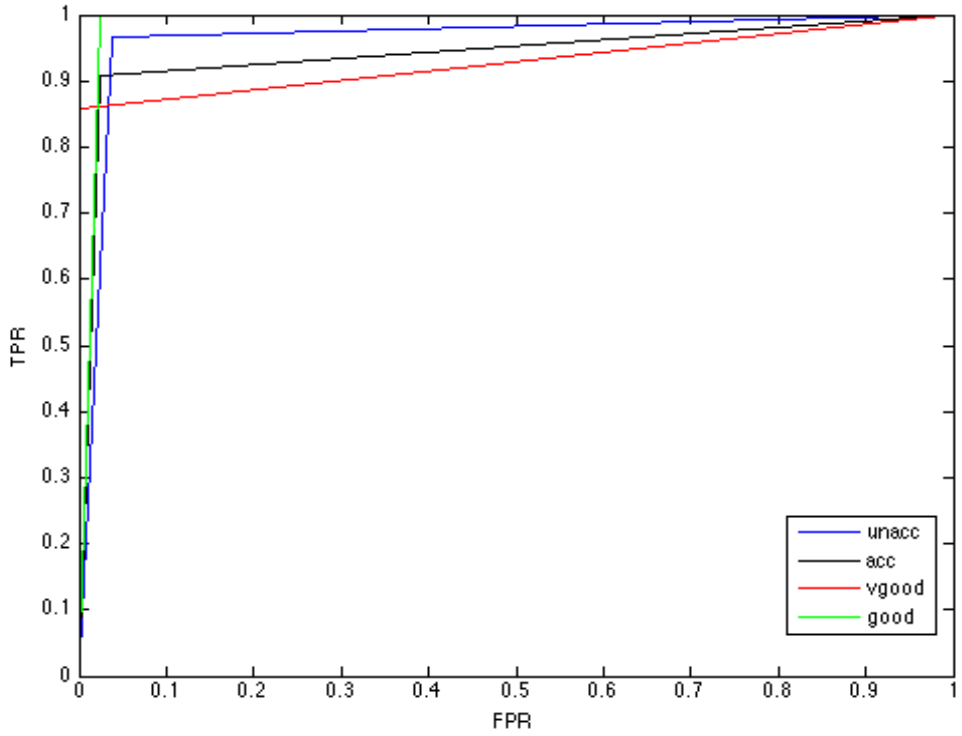
Prune Error number =1

unacc	acc	vgood	good	<-Actual	
115	2	0	0	unacc	<-Classify
3	39	0	0	acc	
0	0	6	0	vgood	
1	2	1	4	good	

Calculation TPR,FPR,Precision,Recall,AUC

Classifier	TPR	FPR	Precision	Recall	AUC
unacc	0.96639	0.03704	0.98291	0.96639	0.9647
acc	0.90698	0.02308	0.92857	0.90698	0.9419
vgood	0.85714	0.00000	1.00000	0.85714	0.9286
good	1.00000	0.02367	0.50000	1.00000	0.9882

Plot four classes in ROC space



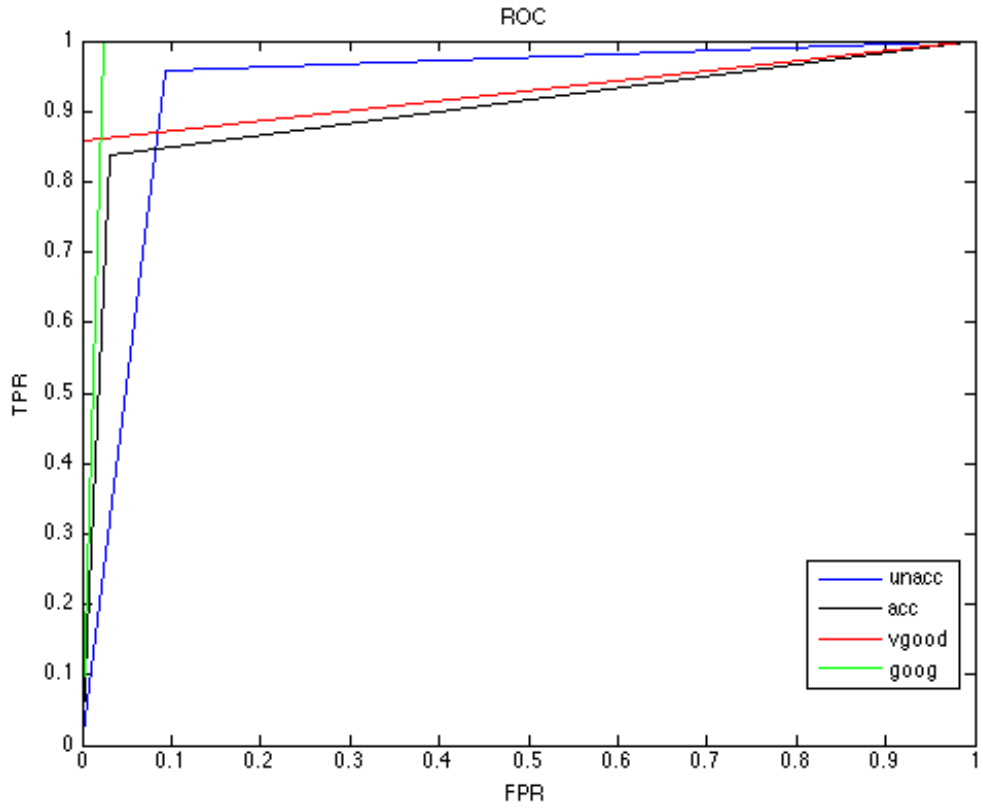
Prune Error number =5

unacc	acc	vgood	good	<-Actual	
114	5	0	0	unacc	<-Classify
4	36	0	0	acc	
0	0	6	0	vgood	
1	2	1	4	good	

Calculation of TPR,FPR,Precision,Recall,AUC

Classifier	TPR	FPR	Precision	Recall	Auc
unacc	0.95798	0.09259	0.95798	0.95798	0.9327
acc	0.83721	0.03077	0.90000	0.83721	0.9032
vgood	0.85714	0.00000	1.00000	0.85714	0.9286
good	1.00000	0.02367	0.50000	1.00000	0.9882

Plot four classes in ROC space



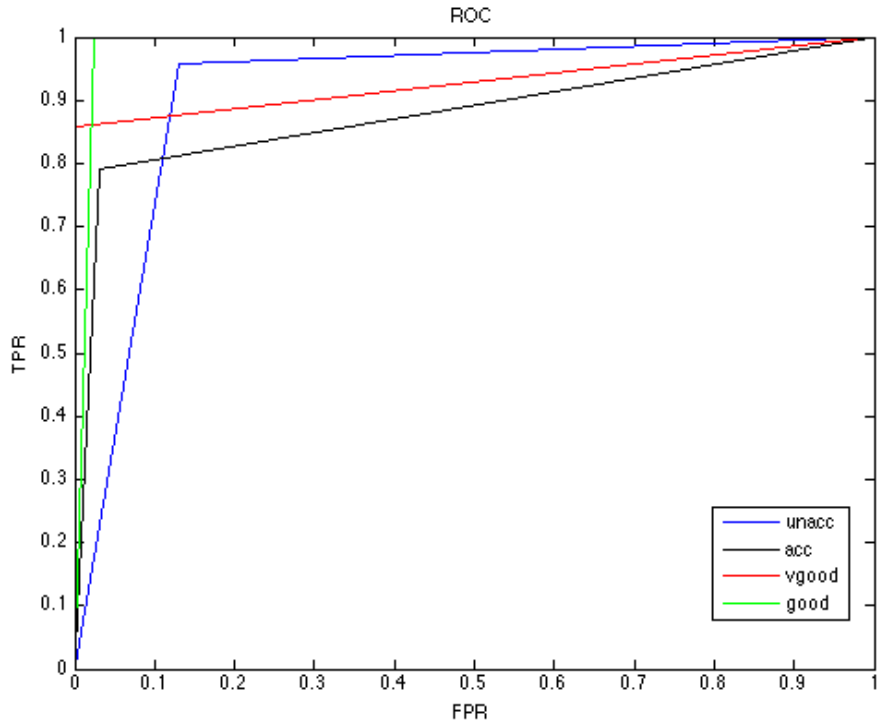
Prune Error number =12

unacc	acc	vgood	good	<-Actual	
114	7	0	0	unacc	<-Classify
4	34	0	0	acc	
0	0	6	0	vgood	
1	2	1	4	good	

Calculation of TPR,FPR,Precision,Recall,AUC

Classifier	TPR	FPR	Precision	Recall	AUC
unacc	0.95798	0.12963	0.94215	0.95798	0.9142
acc	0.79070	0.03077	0.89474	0.79070	0.8800
vgood	0.85714	0.00000	1.00000	0.85714	0.9286
good	1.00000	0.02367	0.50000	1.00000	0.9882

Plot four classes in ROC space



Cost Matrix

Given a classification problem domain, the information about the cost of correct or wrong classification can be given in very different degrees of detail. The most complete way to provide the information about misclassification costs is a Cost Matrix (also known as a Loss Matrix), which indicates the costs for correct and incorrect classifications.

In the car evaluation domain, the cost for misclassify a good, very good or acceptable car into unacceptable is huge since the dealer may lost a business opportunity. Also, misclassifying unacceptable car into acceptable, good or very good damage the reputation of car dealer.

We define a cost matrix as follow:

unacc	**acc**	**vgood**	**good**	**<-Actual**	
0	3.5	5	4.5	unacc	<-Classify
3	0	2	2.5	acc	
4.5	2	0	1	vgood	
3.5	1.5	1.5	0	good	

Generally, we use negative or zero for correct classification.
From this matrix and the confusion matrix it is very easy to compute the cost of a classifier for a given dataset, just as the 1 by 1 matrix product, given a Resulting Matrix:

$$R(i,j) = M(i,j) \cdot C(i,j)$$

And just summing all the elements of the matrix, we have the overall cost:

$$Cost = \sum_i \sum_j R(i,j)$$

Prune error number	Total cost
1	24
5	37.5
12	44.5

Conclusion

As we can see from the analysis, when set the prune error number is 1, we can the highest testing accuracy, highest average AUC value or lowest cost for wrong classification. But tree structure is more complicate than the prune error number 5 and 12. Since it has 161 leaf nodes, it is much more than 113 and 107 leaf nodes. Therefore, there is a trade-off. Vice versa, for prune error number is 12.

A reasonable way for choosing a pruned tree is selecting something in between. In this case, I believe we should choose the pruned decision tree with prune error number of 5.

Insight

During this project, I met several problems such as overfitting problem, a biased entropy formula problem and how to evaluation multi-class classification problem.

Luckily, I got some advice from professor and read some articles written by others and find a solution eventually.

Overall, insights I got from this project are listed below:

Using entropy gain ration rather than entropy gain to overcome the bias when choosing attribute.

Pruning the decision tree to get a general tree to solve the overfitting problem.

Calculate confusion matrix and cost matrix to evaluate multi-class classification.

Reference

Textbook: Machine Learning: A Probabilistic Perspective by Kevin Murphy ch16
http://users.dsic.upv.es/~flip/papers/rocking.pdf
http://en.wikipedia.org/wiki/Decision_tree
http://en.wikipedia.org/wiki/Receiver_operating_characteristic