Auxiliary Uses of Decision Trees:

Optimal Binning of Nominal Variables with Many Levels Using Decision Trees in SAS E-Guide

Introduction

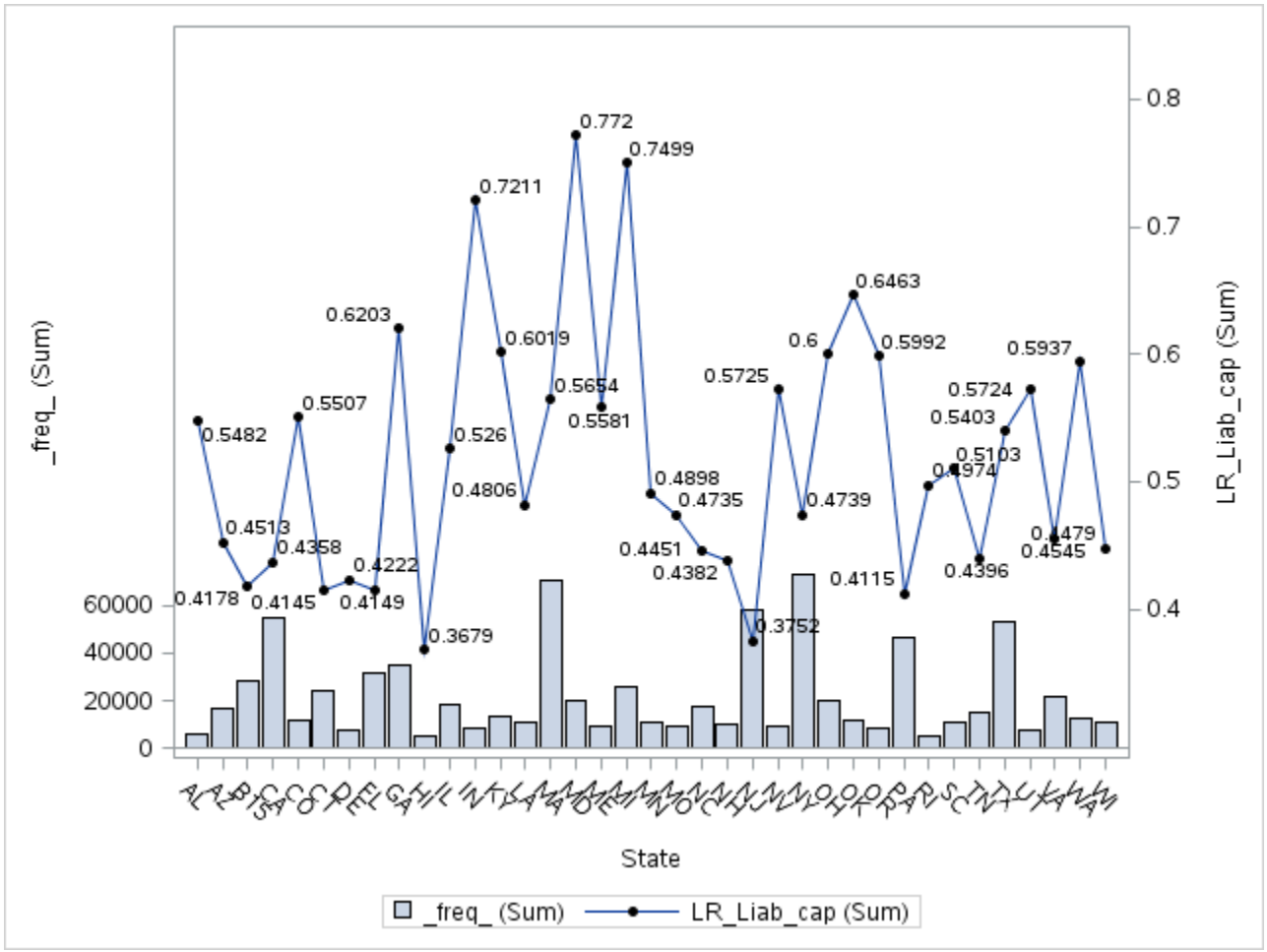
A persistent issue in data mining is dealing with highly dimensional data. Paradoxically adding more data (in terms of numbers of variables) does not necessarily improve the ability to develop a reliably predictive model, as this also increases the dimensions of the input vector. This problem has become known as the curse of dimensionality. Reducing the number of dimensions can in some situations help the modeler develop a more reliable result.

Highly dimensional data is frequently encountered, very often in dealing with nominal values that are used as inputs to models. A nominal value has one of a finite number of discrete values, such as a product code or a region code. Typically, character variables used as inputs in data mining are nominal variables. Numeric variables may be nominal as well; a classic example being a zip code.

Identifying factor levels can be arbitrary, judgmental, or optimality driven. Decision trees are transparent, intuitive, non-parametric, and robust to influential values, outliers, and missing values; therefore, they can be used to find optimal bins, or factor levels.

Example

The target variable in our model is the Loss Ratio. A typical risk factor is the State. In the following graph we can see the relationship between the Loss Ratio and the different states:



This variable has too many level in the context of the current model. Therefore, we would like to collapse the different states in only few levels. We can use decision trees to accomplish this task in an optimal way.

Implementation Using the HPSPLIT Procedure

We can use the new HPSPLIT procedure in SAS STAT 13.2 (THE current version of SAS) to create decision trees in SAS Enterprise Guide 7.1 (the current version of SAS Enterprise Guide). Therefore, we don't need to leave the SAS Enterprise Guide context in order to create our decision trees in SAS Enterprise Miner.

When we use a decision tree to create an optimal binning for a categorical variable with many levels, several other decisions must also be made during this process.

- How many leaves should have the decision tree?
- What is the maximum tree depth?
- What should the level of significance for factor splits be?
- What splitting criterion should be used?
- Should there be a minimum number of members in a node (policies or claims) to create a split?
- How Specifies how to handle missing values in an input variable?

The answers to the above question are the following:

- The level are conditioned by the number of observations in our dataset. In a typical context of data mining with several million of observation a subjective criteria many times applied is 16.
- The maxims tree depth should be one. Here we are using a decision tree such an auxiliary tool in our modeling process, therefore, we would like only to collapse the variable in few levels.
- The significant level for this kind of exercises usually is large, that is 0.1
- We should prevent post-pruning.
- The splitting criteria in the context of this example with at continuous target should be Variance or F-Test.
- The number of minimum observations in a node should be a credible exposure.
- The HPSPLIT is able to handle the missing values based on three options:
    - BRANCH requests that missing values be assigned to their own branch.
    - POPULARITY requests that missing values be assigned to the most popular, or largest, branch.
    - SIMILARITY requests that missing values be assigned to the branch they are most similar to (using the chisquare or F test criterion)

An implementation of the above criteria using the HPSPLIT procedure in the context of our above example is the following SAS code:

```
ods graphics on;
proc hpsplit data=Auto_DS_BT maxdepth=1 maxbranch=9  leafsize=5000 alpha=0.1;
criterion ftest ; /*default for interval target VARIANCE or FTEST */
prune none;
target LR_liab_cap / level = int;
input STATE / level = nom ;
output nodestats=stat;
run;

title 'Tree visualization';
proc print data=stat noobs;
run;
```

Where:

- The dataset is Auto_DS_BT
- The maximum depth of the tree to be grown equals one
- The MAXBRANCH option specifies the maximum number of children per node in the tree. PROC HPSPLIT tries to create this number of children unless it is impossible (for example, if a split variable does not have enough levels). The default is the number of target levels. The maximum number of branches in our example is 9

- The minimum number of polices in each and every leaf is 5000. This option specifies the minimum number of observations that a split must contain in the training data set in order for the split to be considered. By default, LEAFSIZE=1.
- The split criterion used is the F-Test. By default, CRITERION = variance
- The alpha option specify the maximum p-value for a split to be considered, in this case 0.1. By default is 0.3. This option is only considered if you specify the FTEST criterion
- The prune option is not used
- The target variable is LR_Liab_cap. We used the level = int option to indicate that the target variable is continuous
- The explanatory variable is STATE. We used the level = nom option to indicate that the explanatory variable is nominal
- The output of the decision tree calculation is the SAS dataset called STAT
- The option for missing handle is not implemented in the current STAT product. Therefore, by default the criteria is POPULARITY.

Finally, I used the PRINT procedure to visualize the output:

The HPSPLIT Procedure

Performance Information	
Execution Mode	Single-Machine
Number of Threads	4

Data Access Information			
Data	Engine	Role	Path
WORK.AUTO_DS_BT	V9	Input	On Client

Page Break

Tree visualization

DEPTH	N	ID	SPLITVAR	DECISION	ALLTEXT	PARENT	PREDICTEDVALUE	TREENUM	CRITERION	LINKWIDTH	LEAF	INSPLITVAR	P_PRED	STR_ID	P_LR_liab_cap	Level0
0	1423669	0	State		P_LR_liab_cap=0.32297027	.	0.32297	1	Variance	10.0000	.		0.32297	0	0.32297	0.323
1	230858	1		AL,B15,ME,NH,NJ,NV,WI	P_LR_liab_cap=0.26285332	0	0.26285	1	Variance	1.6216	0	State	0.26285	1	0.26285	0.263
1	211471	2		AZ,MO,NY,VA	P_LR_liab_cap=0.28479162	0	0.28479	1	Variance	1.4854	1	State	0.28479	2	0.28479	0.285
1	162782	3		CA,CO,DE,IL	P_LR_liab_cap=0.31955784	0	0.31956	1	Variance	1.1434	2	State	0.31956	3	0.31956	0.32
1	245980	4		CT,FL,IN,NC,PA,SC or Missing	P_LR_liab_cap=0.30162387	0	0.30162	1	Variance	1.7278	3	State	0.30162	4	0.30162	0.302
1	187166	5		GA,RI,TX,WA	P_LR_liab_cap=0.39741393	0	0.39741	1	Variance	1.3147	4	State	0.39741	5	0.39741	0.397
1	181415	6		HI,KY,LA,MN,OH,OK,OR,TN,UT	P_LR_liab_cap=0.3377586	0	0.33776	1	Variance	1.2743	5	State	0.33776	6	0.33776	0.338
1	123699	7		MA	P_LR_liab_cap=0.43156662	0	0.43157	1	Variance	0.8689	6	State	0.43157	7	0.43157	0.432
1	35236	8		MD	P_LR_liab_cap=0.48012815	0	0.48013	1	Variance	0.2475	7	State	0.48013	8	0.48013	0.48
1	45062	9		MI	P_LR_liab_cap=0.14924092	0	0.14924	1	Variance	0.3165	8	State	0.14924	9	0.14924	0.149

Table 15.5 NODESTATS= Data Set Variables

Variable	Target Type	Description
ALLTEXT	Either	Text that describes the split
CRITERION	Either	Which of the three criteria was used
DECISION	Either	Values of the parent variable's split to get to this node
DEPTH	Either	Depth of the node
ID	Either	Node number
LEAF	Either	Leaf number
LINKWIDTH	Either	Fraction of all training observations going to this node
N	Either	Number of training observations at this node
NVALID	Either	Number of validation observations at this node
PARENT	Either	Parent's node number
PREDICTEDVALUE	Either	Value of target predicted at this node
SPLITVAR	Either	Variable used in the split
TREENUM	Either	Tree number (always 1)
P_VARLEV	Nominal	Proportion of training set at this leaf that has target VAR = LEV
V_VARLEV	Nominal	Proportion of validation set at this leaf that has target VAR = LEV
P_VAR	Interval	Average value of target VAR in the training set
V_VAR	Interval	Average value of target VAR in the validation set

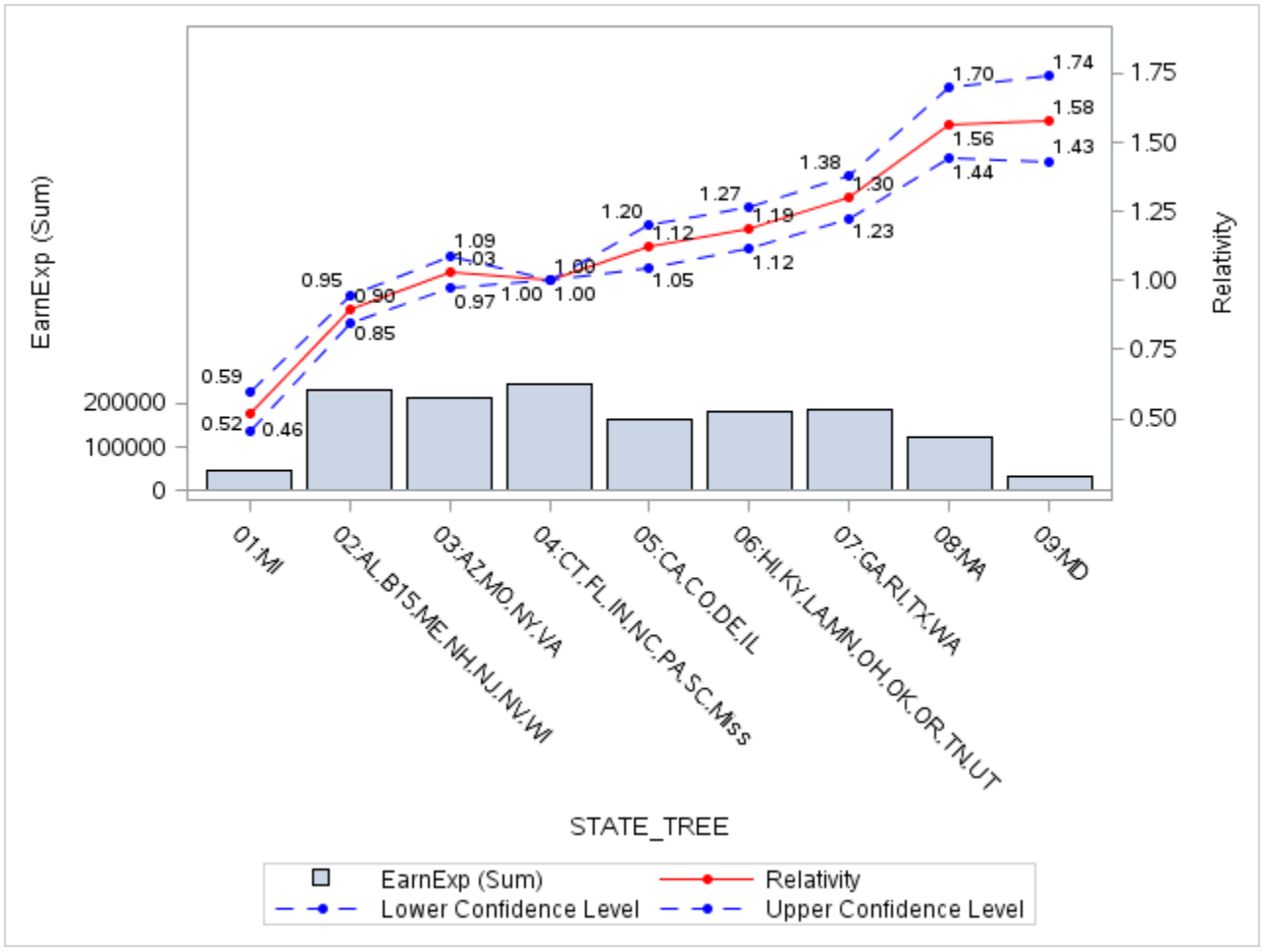
Thankfully to the decision tree we have a derived new variable that groups the States in an optimal way:

```
if STATE in('AL','B15','ME','NH','NJ','NV','WI') then STATE_TREE = '02:AL,B15,ME,NH,NJ,NV,WI';
else if STATE in('AZ','MO', 'NY','VA') then STATE_TREE = '03:AZ,MO,NY,VA';
else if STATE in('CA','CO', 'DE','IL') then STATE_TREE = '05:CA,CO,DE,IL';
else if STATE in('CT','FL', 'IN','NC','PA','SC','') then STATE_TREE = '04:CT,FL,IN,NC,PA,SC,Miss';
else if STATE in('GA','RI', 'TX','WA') then STATE_TREE = '07:GA,RI,TX,WA';
else if STATE in('HI','KY', 'LA','MN','OH','OK','OR','TN','UT') then STATE_TREE =
'06:HI,KY,LA,MN,OH,OK,OR,TN,UT';
else if STATE in('MA')  then STATE_TREE = '08:MA';
else if STATE in('MD') then STATE_TREE = '09:MD';
else if STATE in('MI') then STATE_TREE = '01:MI';
```

The relativities for the STATE_TREE variable in a multivariate model are (training dataset):

Relativities for STATE_TREE

level	Pr > ChiSq	Relativity	Lower Confidence Level	Upper Confidence Level	EarnExp
01:MI	<.0001	0.52	0.46	0.59	45062
02:AL,B15,ME,NH,NJ,NV,WI	0.0001	0.90	0.85	0.95	230858
03:AZ,MO,NY,VA	0.2916	1.03	0.97	1.09	211471
04:CT,FL,IN,NC,PA,SC,Miss	.	1.00	1.00	1.00	245980
05:CA,CO,DE,IL	0.0009	1.12	1.05	1.20	162782
06:HI,KY,LA,MN,OH,OK,OR,TN,UT	<.0001	1.19	1.12	1.27	181415
07:GA,RI,TX,WA	<.0001	1.30	1.23	1.38	187166
08:MA	<.0001	1.56	1.44	1.70	123699
09:MD	<.0001	1.58	1.43	1.74	35236



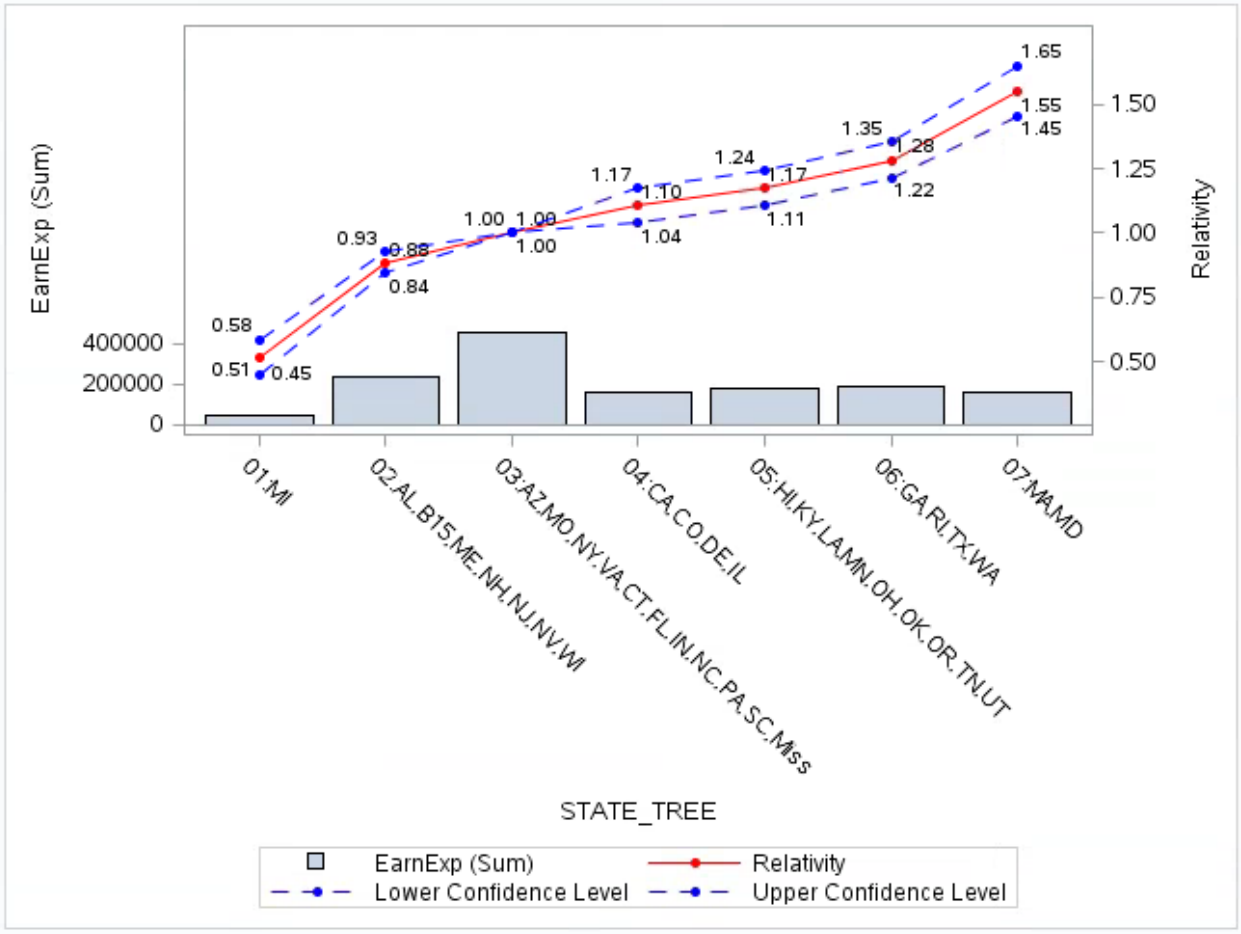
In a multivariable context some of the levels should be re-grouped again, such as 03 and 04 or 08 and 09

Relativities for STATE_TREE

level	Pr > ChiSq	Relativity	Lower Confidence Level	Upper Confidence Level	EarnExp
01:MI	<.0001	0.51	0.45	0.58	45062
02:AL,B15,ME,NH,NJ,NV,WI	<.0001	0.88	0.84	0.93	230858
03:AZ,MO,NY,VA,CT,FL,IN,NC,PA,SC,Miss	.	1.00	1.00	1.00	457451
04:CA,CO,DE,IL	0.0015	1.10	1.04	1.17	162782
05:HI,KY,LA,MN,OH,OK,OR,TN,UT	<.0001	1.17	1.11	1.24	181415
06:GA,RI,TX,WA	<.0001	1.28	1.22	1.35	187166
07:MA,MD	<.0001	1.55	1.45	1.65	158935

Page Break

Relativities for STATE_TREE



Summary

- The new HPSPLIT can be used as an auxiliary tool during the modeling process creating an optimal binning of nominal variable with too many levels.
- Identifying factor levels can be arbitrary, judgmental, or optimality driven. Decision trees are transparent, intuitive, non-parametric, and robust to influential values, outliers, and missing values.
- The process is very quickly and you don't need to leave the SAS E-Guide context to use SAS Enterprise Miner

Appendix: Regression Trees, Partition Criteria and Heteroscedasticity

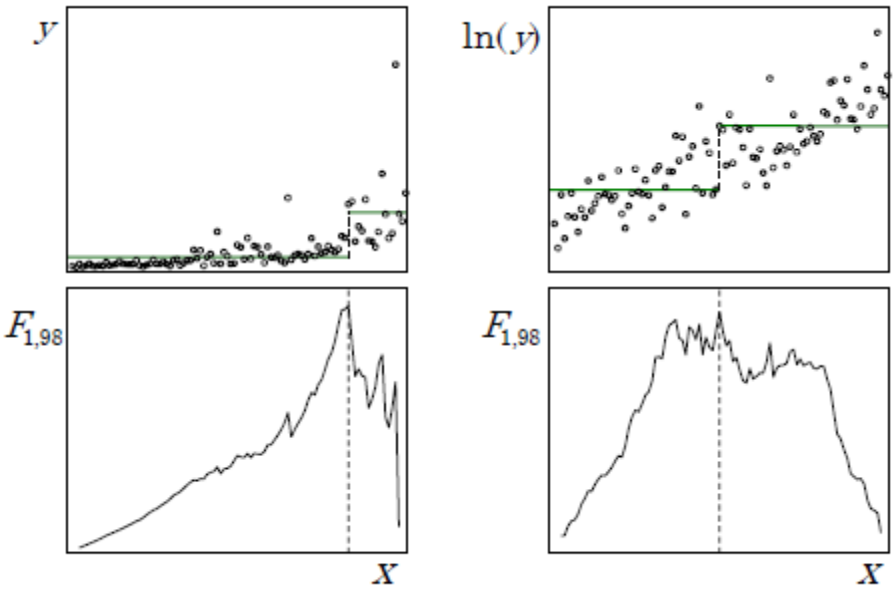
There are two splitting criteria used in regression tress:

- The first criterion is based on impurity function of a node. When the target distribution is continuous, the sample variance is the obvious measure of impurity.

- The second criterion is based on a statistical test for one-way ANOVA, the F-Test.

Some differences between the two above criteria:

- The F test is better than (sample) variance reduction as it has P-value adjustment number of branches.
- F test is relatively robust to departures from normality assumption
- However, F test is sensitive to departures from non-constant variance



The F test has many optimal properties when the distribution of the target is independently and normally distributed with constant variance. The F test is relatively robust to departures from the normality assumption. However, variance heterogeneity (heteroscedasticity) can have disastrous effects. For example, the F test is too liberal (overstates the significance of the effect) when small nodes have larger variance. Consider the common case of a nonnegative target with variance increasing with the mean. Using the F test as the splitting criterion will tend to favor small splits of the largest values. Decision trees are usually regarded as robust and nonparametric. However, regression trees are not robust to heteroscedasticity. Like classical regression models, finding a suitable variance stabilizing transformation can improve the model. For example, apply a logarithm or a squared root transformation to the target variable. A typical context where a logarithm transformation is useful happens when the severity is the target variable.

References

- De Ville, Barry, and Padraic Neville. 2013. Decision Trees for Analytics Using SAS® Enterprise Miner. Cary, NC: SAS Institute Inc.
- The Anti-Curse: Creating an Extension to SAS® Enterprise Miner™ Using PROC ARBORETUM Andrew Cathie, SAS Institute (NZ) Ltd, Auckland, New Zealand
- Decision Tree Modeling Course Notes was developed by William J.E. Potts and revised by Lorne Rothman. Technical review was provided by Bob Lucas and Michael J. Patetta. 2006 by SAS Institute Inc. Chapter 4 – Auxiliary Uses of Trees. Section 4.3. - Collapsing Levels
- SAS/STAT® 13.2 User's Guide High-Performance Procedures - Chapter 15. The HPSPLIT Procedure
- SAS Enterprise Miner 13.2 High Performance Data Mining Nodes Help