Salford Predictive Modeler®

Introducing Generalized PathSeeker® (GPS)

This guide provides a brief introduction to GPS as well as a guide insight for model interpretation.

Minitab

Background and Motivation

GPS or Generalized PathSeeker® is a highly specialized and flexible regression (and logistic regression) procedure developed by Jerome Friedman (the co-creator of CART® and the developer and inventor of MARS® and TreeNet®, among several other major contributions to data mining and machine learning). GPS is a "regularized regression" procedure meaning that it is designed to handle modeling challenges that are difficult or impossible for everyday regression. For example, GPS is well suited to build models when:

- there are many more columns (predictors) than rows (observations);
- the predictors available may be extremely highly correlated with each other; or
- the goal is to find the most compact model yielding acceptable performance.

GPS strengths include high speed processing of huge numbers of predictors such as are found in bioinformatics and text mining, and increasingly in models of consumer behavior leveraging web activity and social network data. Predictive models using such data must be able to deftly handle from tens of thousands to possibly millions of predictors. GPS is also a capable variable selection machine, sifting a possible small subset of predictors from a huge pool of candidates.

Friedman's highly technical paper introducing GPS ("Fast Sparse Regression and Classification" (2008)) can be downloaded here.

Before using GPS as a data mining engine it is important to understand its limitations:

- GPS is ultimately a regression procedure that builds a linear model that is additive in its predictors. This means that GPS cannot on its own discover nonlinearities or interactions in data unless you prepare special predictors embodying such flexibility of the model.
- GPS cannot on its own handle missing values and will enforce row deletion of data when missing values are encountered.

In both of these senses GPS has limitations that are identical to that of traditional regression; GPS on its own shares the rigidity that makes linear regression so often unsuitable for modern data analytics. We do have some good news regarding GPS: when properly combined with TreeNet, where TreeNet first pre-processes the data, handles the missing values, and automatically discovers the nonlinearity and interactions embedded in the model, GPS becomes a superb tool in a true data mining context. We discuss this combination of TreeNet and GPS in the sections dedicated to model compression and rule learning. For technical details on GPS refer to the later sections of this discussion.

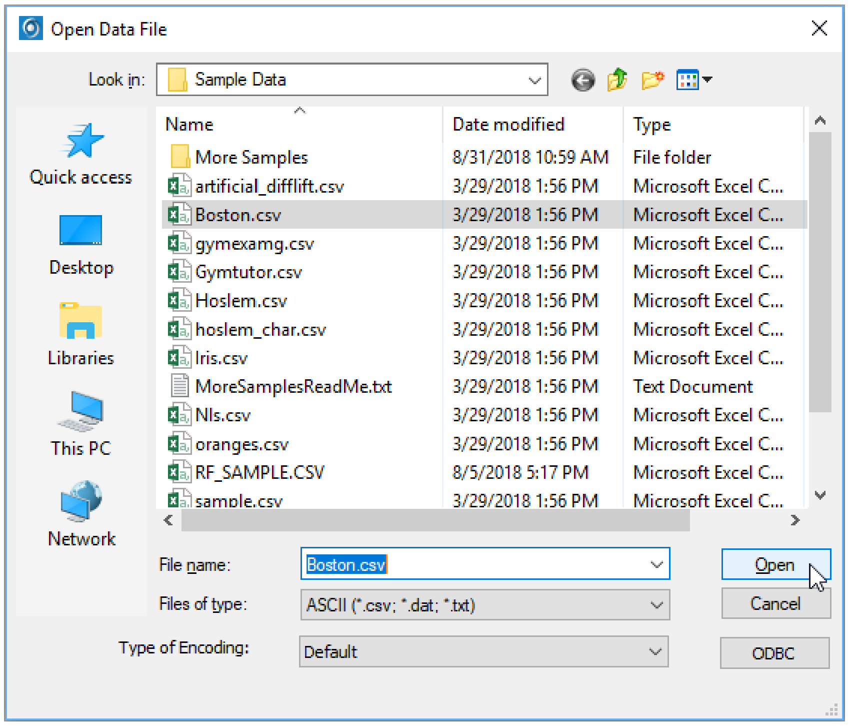
GPS is a forward stepping model building procedure. It is far different than traditional forward stepwise regression however. In the famous, and much maligned, forward stepwise statistical model one searches for the best variable to introduce (or add) to the model and then a traditional regression model is built; at each stage the variable that improves R-Squared the most (on training data) is added and a conventional regression is then run using just those variables. At each stage one more variable enters the model. In GPS we use a form of "partial" introduction of a variable into the model by permitting the variable to be given a very small coefficient (much smaller than a traditional regression would want to assign, perhaps as little as 1/10 of one percent of the everyday regression coefficient). Once the chosen variable has been added in this partial way the data is re-analyzed to determine what to do next. For example, include a new variable or update the coefficient on a variable already in the model.

Getting Started

Here we turn to a hands on tour of getting starting with GPS and review the essential model setup dialogs and the interpretation of the outputs. Please refer to the sample data sets included with your installation to locate the BOSTON.CSV file in order to follow along with our next examples.

Setting up a GPS Model

Click the **File—Open>Data File…** menu item:



Look in: Sample Data

Name	Date modified	Type
More Samples	8/31/2018 10:59 AM	File folder
artificial_difflift.csv	3/29/2018 1:56 PM	Microsoft Excel C...
Boston.csv	3/29/2018 1:56 PM	Microsoft Excel C...
gymexamg.csv	3/29/2018 1:56 PM	Microsoft Excel C...
Gymtutor.csv	3/29/2018 1:56 PM	Microsoft Excel C...
Hoslem.csv	3/29/2018 1:56 PM	Microsoft Excel C...
hoslem_char.csv	3/29/2018 1:56 PM	Microsoft Excel C...
Iris.csv	3/29/2018 1:56 PM	Microsoft Excel C...
MoreSamplesReadMe.txt	3/29/2018 1:56 PM	Text Document
Nls.csv	3/29/2018 1:56 PM	Microsoft Excel C...
oranges.csv	3/29/2018 1:56 PM	Microsoft Excel C...
RF_SAMPLE.CSV	8/5/2018 5:17 PM	Microsoft Excel C...
sample.csv	3/29/2018 1:56 PM	Microsoft Excel C...

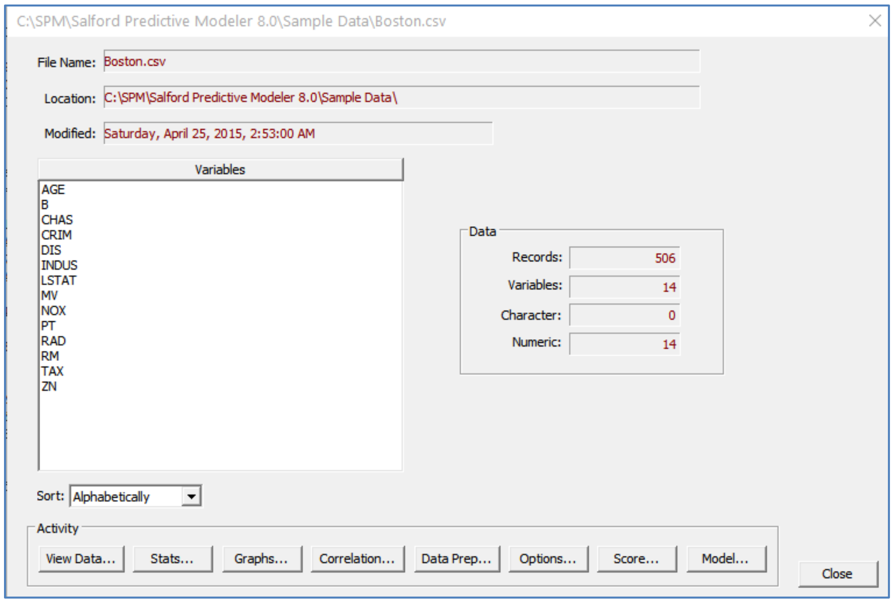
File name: Boston.csv

Files of type: ASCII (*.csv; *.dat; *.txt)

Type of Encoding: Default

Navigate to the *Sample Data* folder, select BOSTON.CSV dataset, and press the **[Open]** button. This will open the **Activity Window** shown below.

✓ If your system does something else you may use the **Edit** menu to select **Options-General** tab and then change what SPM does upon opening a data file.



C:\SPM\Salford Predictive Modeler 8.0\Sample Data\Boston.csv

File Name: Boston.csv

Location: C:\SPM\Salford Predictive Modeler 8.0\Sample Data\

Modified: Saturday, April 25, 2015, 2:53:00 AM

Variables: AGE, B, CHAS, CRIM, DIS, INDUS, LSTAT, MV, NOX, PT, RAD, RM, TAX, ZN

Data

Records:	506
Variables:	14
Character:	0
Numeric:	14

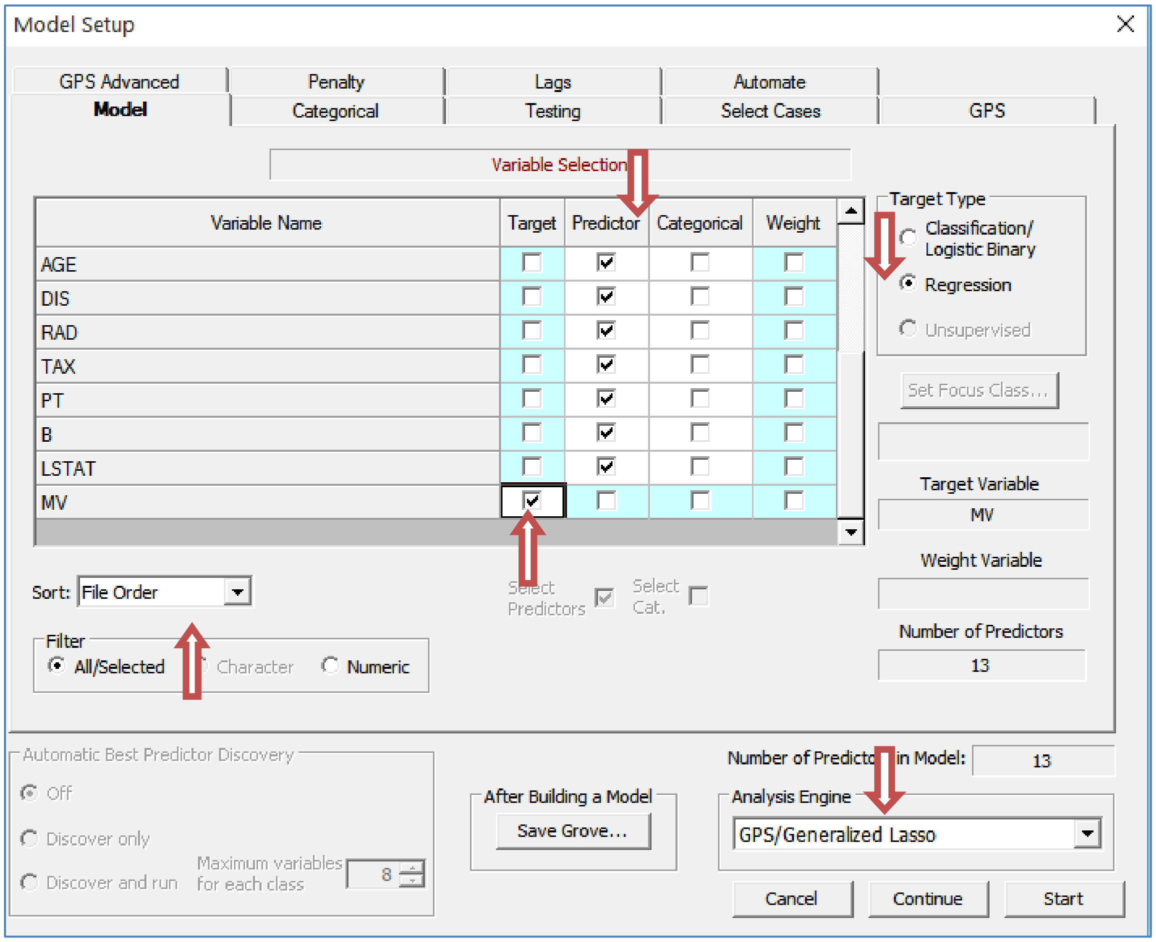
Sort: Alphabetically

Activity
View Data... Stats... Graphs... Correlation... Data Prep... Options... Score... Model...
Close

We see a confirmation screen listing the variables available, number of records and variables, and some other descriptive information on the file. This display is intended to allow you to be sure that you have indeed opened the file you wanted.

We can now move onto a number of follow-up actions represented by the lower row of buttons. Choose the **[Model…]** button to reveal the **Model Setup** dialog which is where we select our dependent variable, predictors, data mining engine and analysis type. On the **Model** tab in the **Model Setup** window:

- Set **Analysis Engine** to **GPS/Generalized Lasso**
- Change **Sort** into **File Order**
- Check **MV** in the **Target** column – this specifies the dependent variable for the current modeling round. In this data set **MV** stands for the median value of owner-occupied homes in Boston in 1970 for each of 506 Census tracts. (See here (link doesn't work)for details in the context of the econometric study that made this data famous.)
- Check all remaining variables in the **Predictor** column – thus all remaining variables will be used as potential predictors
- Set **Target Type** selection to **Regression** as we are predicting a continuous dependent variable



Variable Name	Target	Predictor	Categorical	Weight
AGE	☐	✓	☐	☐
DIS	☐	✓	☐	☐
RAD	☐	✓	☐	☐
TAX	☐	✓	☐	☐
PT	☐	✓	☐	☐
B	☐	✓	☐	☐
LSTAT	☐	✓	☐	☐
MV	✓	☐	☐	☐

Sort: File Order

Filter: [✓ All/Selected] Character Numeric

Target Type: Classification/ Logistic Binary [✓ Regression] Unsupervised

Target Variable: MV

Weight Variable:

Number of Predictors: 13

Automatic Best Predictor Discovery: [✓ Off] Discover only Discover and run; Maximum variables for each class: 8

After Building a Model

Number of Predictors in Model: 13

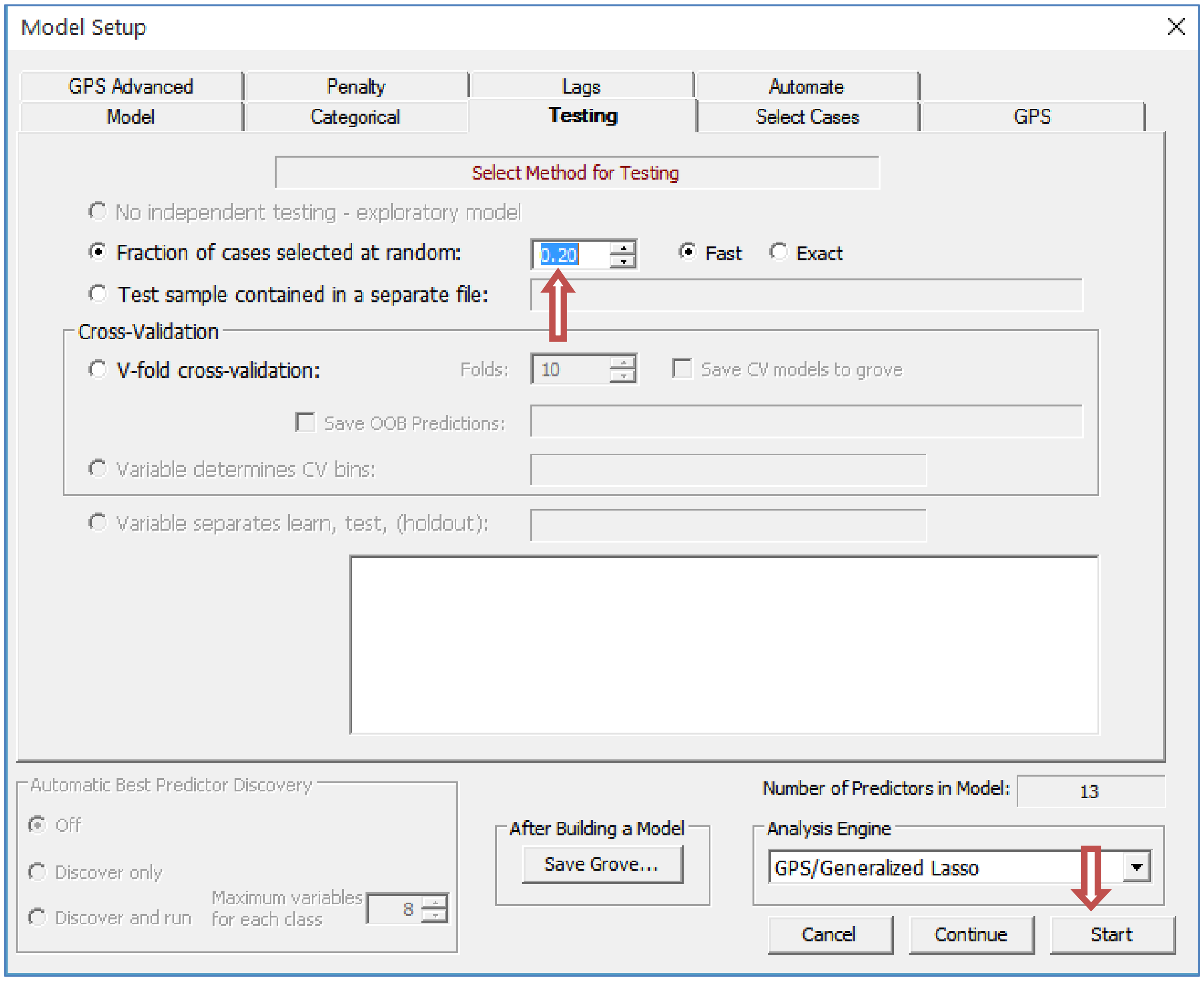
Analysis Engine: GPS/Generalized Lasso

- ✓ Observe that this is all we need to do get started with our analysis: open a data set, select a dependent variable, select an analysis method, and an analysis type. Clicking the **[Start]** button will launch the analysis using intelligent defaults for all other control options.

Although we could just click the **[Start]** button now we recommend visiting the **Testing** tab to select the test method best suited for your needs. As you probably already know modern data mining methods usually do not offer the classical statistical measures often used for model assessment and diagnosis. In data mining rather than study performance on training data we prefer to use test data to determine the quality of our models.

In the dialog below you can see the testing options available. We select 20% randomly chosen test partition. As a matter of policy we would probably favor cross-validation for a data set this small (only 506 rows) but it will be easier to illustrate some important concepts setting aside an explicit test partition.

- ✓ 10-fold Cross-Validation is the default option


Model Setup

GPS Advanced | Penalty | Lags | Automate
Model | Categorical | Testing | Select Cases | GPS

Select Method for Testing

( ) No independent testing - exploratory model
(●) Fraction of cases selected at random: 0.20 (●) Fast ( ) Exact
( ) Test sample contained in a separate file:

Cross-Validation
( ) V-fold cross-validation: Folds: 10 [ ] Save CV models to grove
[ ] Save OOB Predictions:
( ) Variable determines CV bins:

( ) Variable separates learn, test, (holdout):

Automatic Best Predictor Discovery
(●) Off
( ) Discover only
( ) Discover and run — Maximum variables for each class: 8

After Building a Model
Save Grove...

Number of Predictors in Model: 13

Analysis Engine
GPS/Generalized Lasso

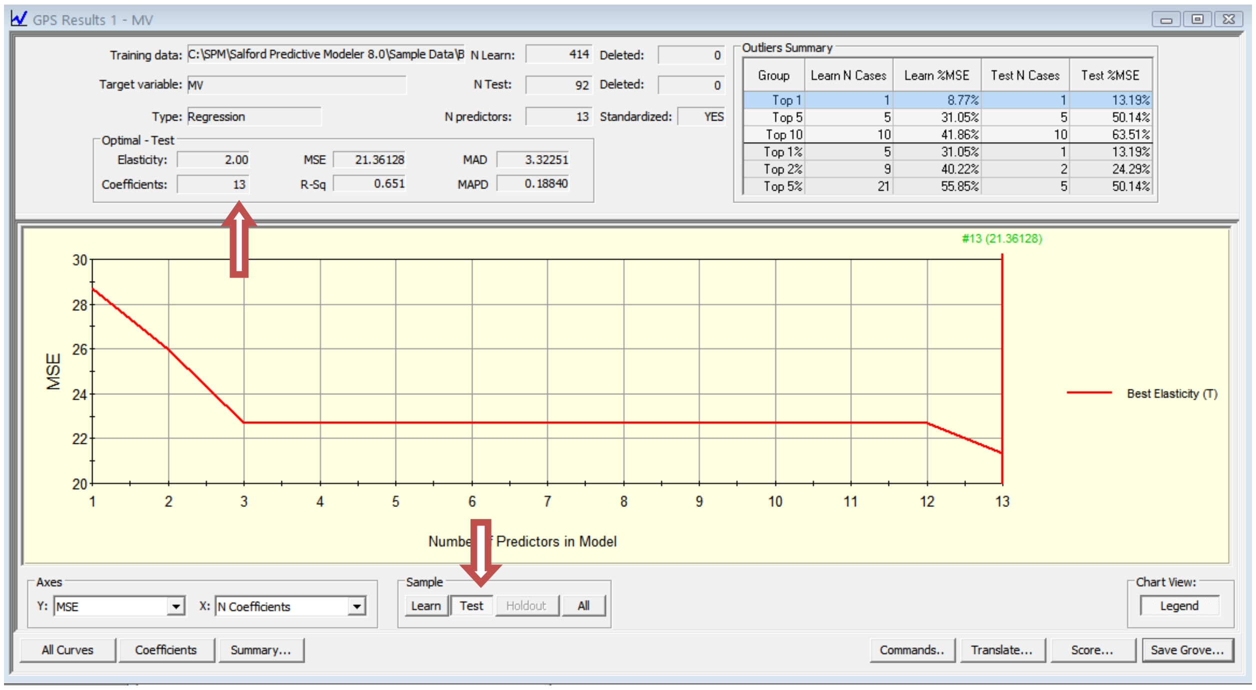
Now click the **[Start]** button and wait for the results, which will come back more or less instantaneously.

Viewing Model Results

The **GPS Results: Paths** window provides basic information on the results of running GPS regularized regression and is geared towards a practical user who wants to start using models right away and skip all of the inner details.

First observe the main graph area, showing model performance on the test sample in terms of *Mean Squared Error* (MSE) indexed by the model size in coefficients. Again, any GPS model is a linear regression model with coefficients selected as a result of some elaborate internal searching technique.

The upper part of the window shows performance results of the best model found. By default, the best (or optimal) model is the one that minimizes MSE on the test sample. The optimal model is also marked by the vertical line in the graph area. In this case, the model has 13 coefficients and achieves test MSE of 21.361. Other performance measures are also reported.



Training data: C:\SPM\Salford Predictive Modeler 8.0\Sample Data\B
Target variable: MV
Type: Regression
N Learn: 414 Deleted: 0
N Test: 92 Deleted: 0
N predictors: 13 Standardized: YES

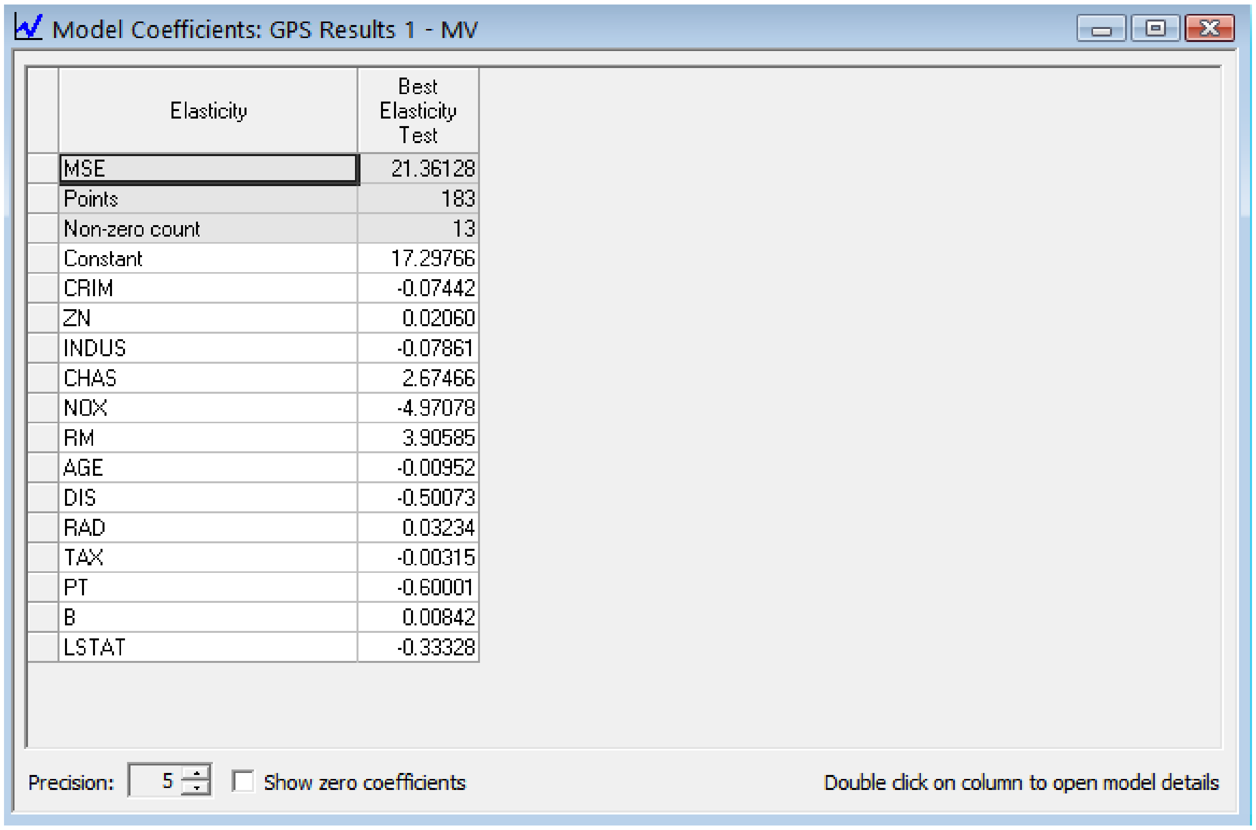
Optimal - Test
Elasticity: 2.00 MSE 21.36128 MAD 3.32251
Coefficients: 13 R-Sq 0.651 MAPD 0.18840

Y: MSE X: N Coefficients

Outliers Summary

Group	Learn N Cases	Learn %MSE	Test N Cases	Test %MSE
Top 1	1	8.77%	1	13.19%
Top 5	5	31.05%	5	50.14%
Top 10	10	41.86%	10	63.51%
Top 1%	5	31.05%	1	13.19%
Top 2%	9	40.22%	2	24.29%
Top 5%	21	55.85%	5	50.14%

Now click the **[Coefficients]** button to see what the model coefficients are:



Elasticity	Best Elasticity Test
MSE	21.36128
Points	183
Non-zero count	13
Constant	17.29766
CRIM	-0.07442
ZN	0.02060
INDUS	-0.07861
CHAS	2.67466
NOX	-4.97078
RM	3.90585
AGE	-0.00952
DIS	-0.50073
RAD	0.03234
TAX	-0.00315
PT	-0.60001
B	0.00842
LSTAT	-0.33328

Precision: 5 Show zero coefficients Double click on column to open model details

Use the following controls to change what is displayed in the main graph area:

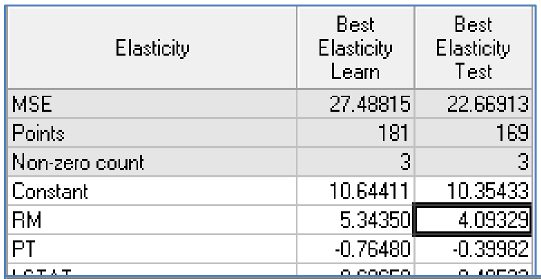
♦ In the **Axes – Y:** selection box, choose among different regression performance evaluation criteria: *R-squared*, *Mean Squared Error* (MSE), *Mean Absolute Deviation* (MAD), *Mean Relative Absolute Deviation* (MRAD), *Root Mean Squared Error* (RMSE), and a number of standard penalized error measures AIC, AICc, BIC.

Strictly speaking *R-Squared* is a learn sample only concept and most data miners prefer to look at test sample *Mean Squared Error*; however, the *R-Squared* measure is familiar, easily understood, and already normalized so we have extended the concept for convenience.

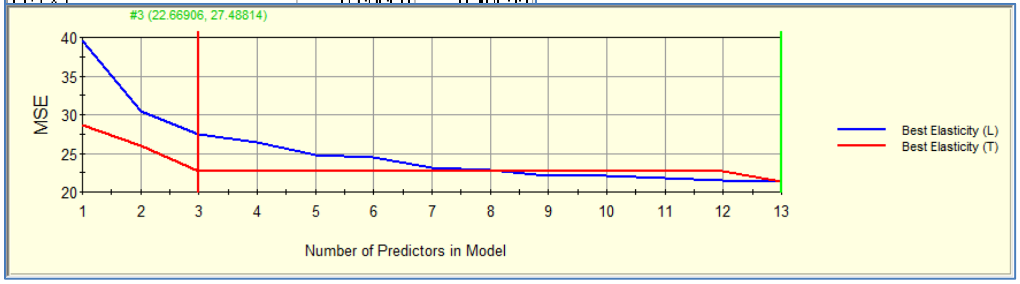
♦ Press the **[Learn]** button to view performance on the learn sample, the **[Test]** button to view performance on the test sample, or the **[All]** button to view both curves

♦ Observe that doing the above manipulations automatically changes the coefficients reported in the **Model Coefficients: GPS Results** window

So far you only looked at 13-coefficient models; however, you may freely navigate within the graph window viewing different models. For example, press the **[All]** button and then click on the 3-coefficient model in the graph area – it should now be marked with the red vertical line.



Elasticity	Best Elasticity Learn	Best Elasticity Test
MSE	27.48815	22.66913
Points	181	169
Non-zero count	3	3
Constant	10.64411	10.35433
RM	5.34350	4.09329
PT	-0.76480	-0.39982

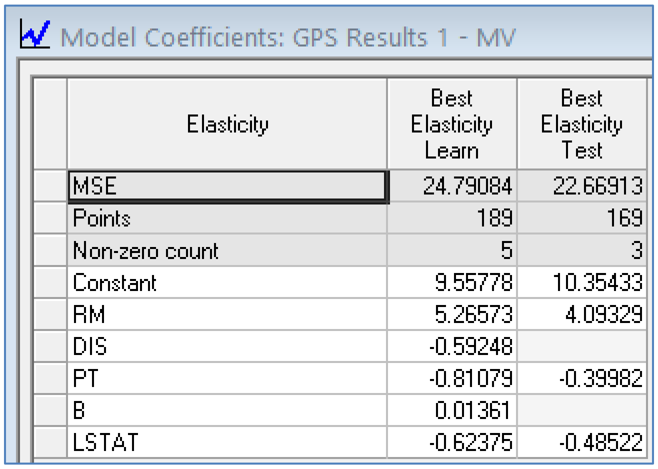


Observe the following:

- The **Model Coefficients** window reflects the new values of the coefficients.
- More importantly, the coefficients in the **Best Elasticity Learn** column differ from the coefficients in the **Best Elasticity Test** column thus displaying two different models. The former set optimizes performance on the learn sample, while the latter set optimizes performance on the test set. Internally GPS builds a potentially very large pool of linear models with varying number of coefficients and varying values. However, only the models with the best performance on the currently selected evaluation criterion, data partition, and model size are selected.

✓ The coefficients of the optimal test sample model have smaller magnitudes than the coefficients of the optimal learn sample model. This is an illustration of a well-known fact that shrinking OLS regression coefficients towards zero may improve model performance on a different test sample.

Now pick the 5-coefficient model in the graph area, observe the following:



Model Coefficients: GPS Results 1 - MV

Elasticity	Best Elasticity Learn	Best Elasticity Test
MSE	24.79084	22.66913
Points	189	169
Non-zero count	5	3
Constant	9.55778	10.35433
RM	5.26573	4.09329
DIS	-0.59248	
PT	-0.81079	-0.39982
B	0.01361	
LSTAT	-0.62375	-0.48522

- The learn sample optimal model changed and now shows 5 non-zero coefficients.
- The test sample optimal model did not change and shows 3 non-zero coefficients as before. What this means is that among all internally constructed models GPS could not find a 4-coefficient or 5-coefficient model that outperforms the optimal 3-coefficient model in terms of test sample MSE. One

way to think about it is that the optimal 5-coefficient model on the test sample has two of the coefficients zeroed out.

- ♦ Therefore, the flat segment on the test sample performance graph between 3 and 12 means that the 3-coefficient model is the best MSE performing model on the test sample until all 13 variables are introduced at which point a better model is found.
- ✓ This is why displayed performance curves are always monotone, perhaps having some flat segments. The idea is that the only justification for a larger (in terms of non-zero coefficients) model is if that model shows better performance; otherwise, a small model is obviously a better choice.
- ✓ These interesting phenomena stem from the fact that GPS works with the learn sample to build the internal pool of models, the test sample is only that used to evaluate the performance of the resulting models. Therefore, while it is always guaranteed that a 4-variable model will almost always improve performance over the 3-variable model on the learn sample, it may no longer be the case on the test sample.
- ✓ This also explains why the learn performance curves, unlike test sample performance curves, tend not to have flat segments.

Thus, the overall finding so far is that the 3-variable model achieved robust optimal test sample performance over a wide range of models and can only be marginally improved by a much larger 13-variable model.

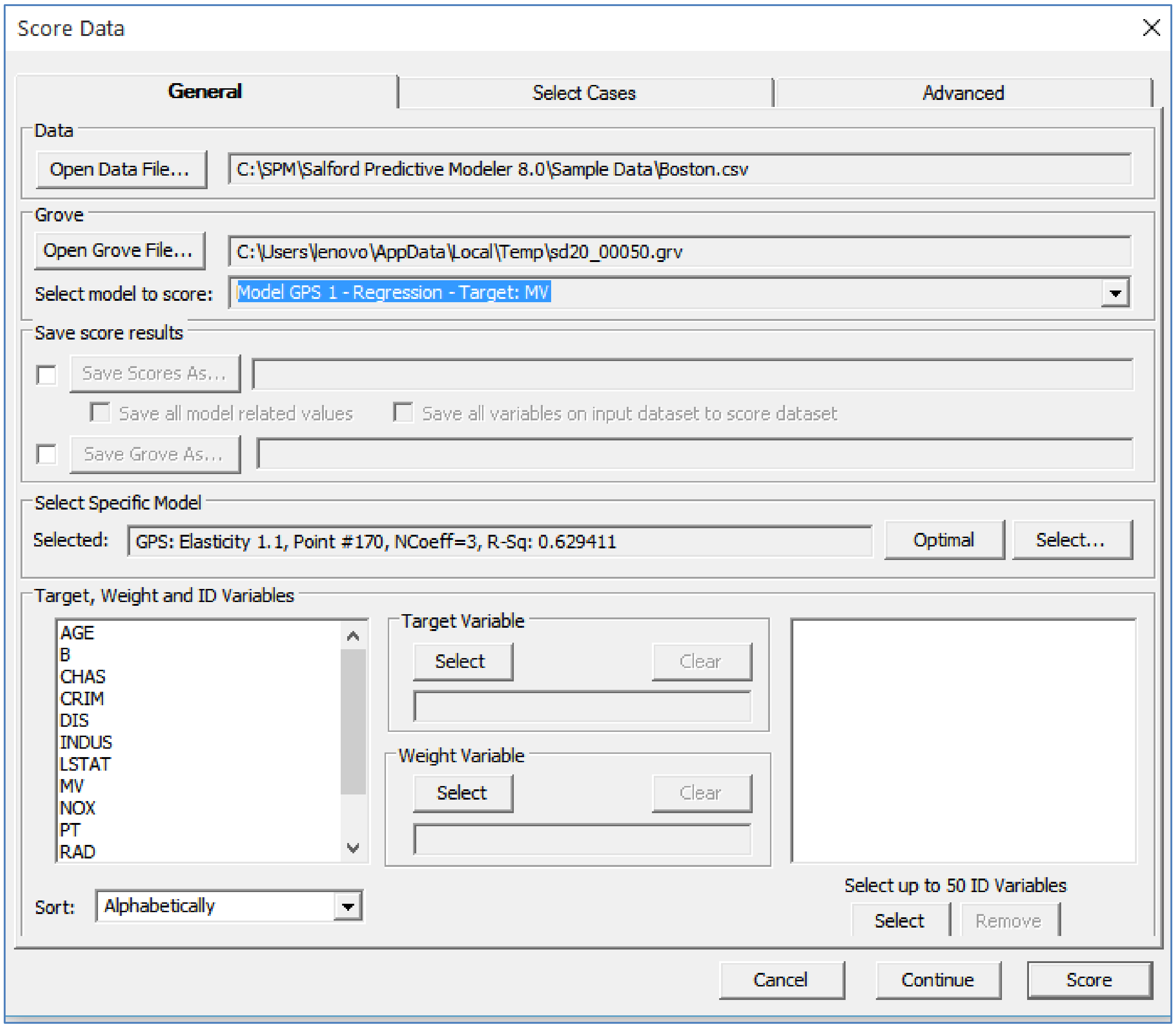
One can view detailed summary information on the currently selected model by pressing the **[Summary…]** button. It produces a standard set of reports (described earlier in this manual) SPM provides for any regression model generated by any of our engines (including, therefore, CART, MARS, Random Forests, TreeNet, and Regression).

One can also save the entire set of results into a grove file by clicking the **[Save Grove…]** button. This will create a binary file with extension GRV. The grove file can then be loaded at a different point in time (and perhaps on a different computer) using the **File/Open/Open Grove…** menu item.

Applying Model to New Data

We can now proceed with applying the 3-coefficient optimal test MSE model (or any other model of your choice) to a dataset to obtain predicted responses.

- ♦ Switch to the **GPS Results -MV** window, make sure that **MSE** is selected, the **[Test]** button is pressed, and the 3-variable model is highlighted by the vertical red line after clicking in the graph area.
- ♦ Press the **[Score…]** button – this will open the **Score Data** window.
- ♦ In the **Score Data** window, the current modeling dataset BOSTON.CSV is already selected for scoring.
- ♦ Click on the check box next to the **[Save Scores As…]** button and in the resulting window enter BOSTON_SCORE.CSV, this file will be created during model scoring process and will contain the prediction results.
- 💣 You may not have the write permission to the **Sample Data** folder; in this case simply pick any folder you do have write permissions for.
- ♦ Press the **[Score]** button to initiate the model scoring process.



Score Data

Data: C:\SPM\Salford Predictive Modeler 8.0\Sample Data\Boston.csv

Grove: C:\Users\lenovo\AppData\Local\Temp\sd20_00050.grv

Select model to score: Model GPS 1 - Regression - Target: MV

Save score results

☐ Save all model related values ☐ Save all variables on input dataset to score dataset

Select Specific Model

Selected: GPS: Elasticity 1.1, Point #170, NCoeff=3, R-Sq: 0.629411

Target, Weight and ID Variables

AGE
B
CHAS
CRIM
DIS
INDUS
LSTAT
MV
NOX
PT
RAD

Target Variable

Weight Variable

Select up to 50 ID Variables

Sort: Alphabetically

The **Score Results** window will appear containing basic scoring stats. More importantly, a new file BOSTON_SCORE.CSV is now created containing four variables:

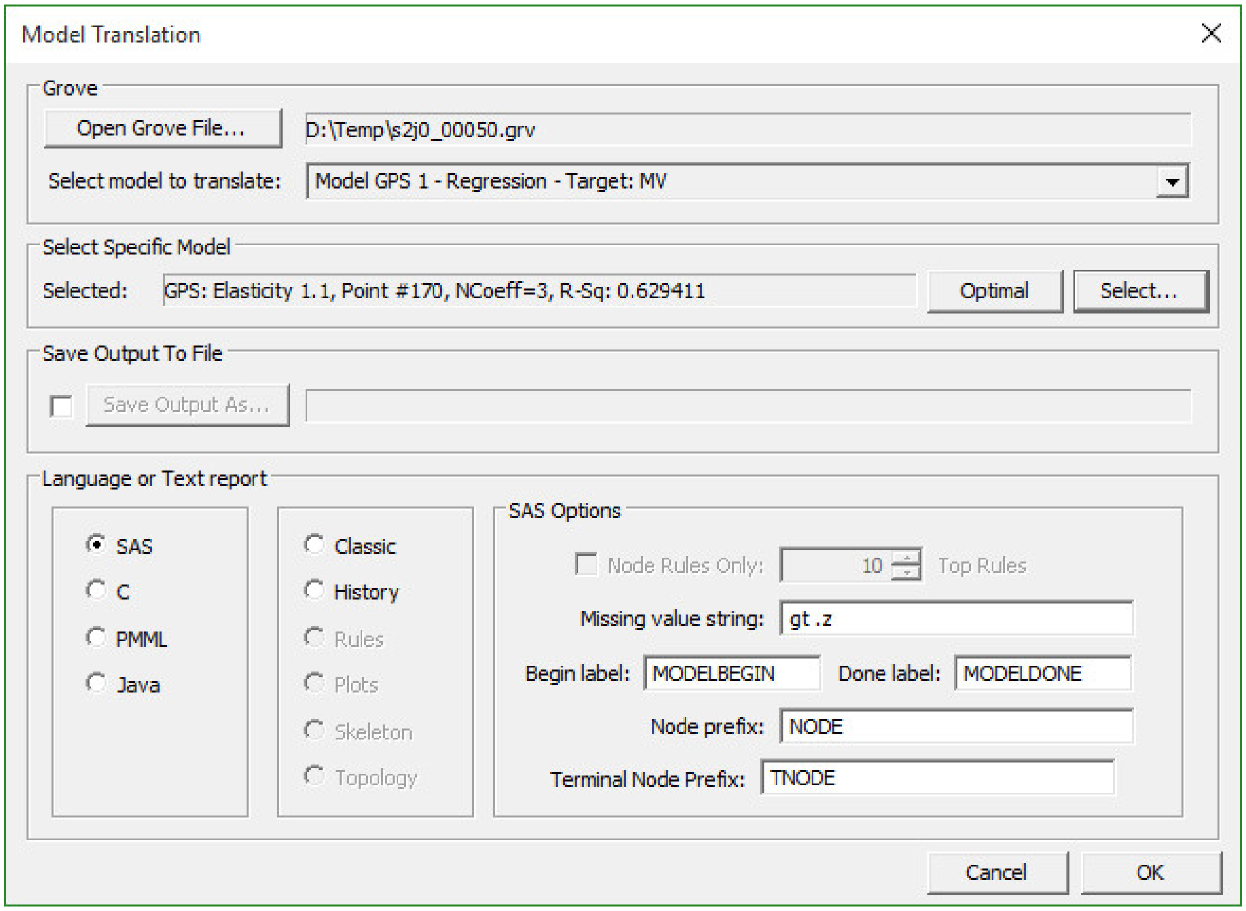
- **CASEID** – record number – 1, 2, 3, etc.
- **RESPONSE** – predicted response generated by the model you've just scored.
- **MV** – observed response available in the input dataset.
- **RESIDUAL** – predicted response minus the observed response.

✓ When the observed response is not available in the input dataset, the residuals and MV will not be available, only the RESPONSE will be produced.

Translating Model

The scoring just conducted relies on the SPM software inner mechanism. You may want to extract the model into a convenient representation for use outside of SPM. This is known as model translation:

- Switch to the **GPS Results- MV** window, make sure that **MSE** is selected, the **[Test]** button is pressed, and the 3-variable model is highlighted by the vertical red line after clicking in the graph area (same as before).
- Press the **[Translate…]** button – this will open the **Model Translation** window.


Model Translation

Grove
Open Grove File… D:\Temp\s2j0_00050.grv
Select model to translate: Model GPS 1 - Regression - Target: MV

Select Specific Model
Selected: GPS: Elasticity 1.1, Point #170, NCoeff=3, R-Sq: 0.629411
Optimal Select…

Save Output To File
Save Output As…

Language or Text report
[✓ SAS] C PMML Java
Classic History Rules Plots Skeleton Topology

SAS Options
Node Rules Only: 10 Top Rules
Missing value string: gt .z
Begin label: MODELBEGIN Done label: MODELDONE
Node prefix: NODE
Terminal Node Prefix: TNODE

Cancel OK

- Note that the model of interest is already selected
- Also note that you have a number of output language options available, we use SAS as an illustration.
- Click on the check box next to the **[Save Output As…]** button and in the resulting window enter BOSTON_MODEL.SAS, this file will be created during model scoring process and will contain the model code is SAS language.

You may not have the write permission to the Sample Data folder; in this case simply pick any folder you do have write permissions for.

- Press the **[OK]** button to initiate the model translation process.

A new file BOSTON_MODEL.SAS will be created containing model code in SAS-compatible language. Note the model simplicity – any GPS model is a simple linear combination of coefficients. However, it differs from OLS regression and was selected to have superior performance on the test sample.

This concludes the hands-on introduction into GPS regression engine. We used a simple dataset to illustrate the key practical steps a modeler would usually take in order to build a useful linear solution. Note that a similar set of steps is required to build a GPS logistic regression to model binary response. Simply switch to the **Classification**/**Logistic Binary Target Type** in the **Model** tab on the **Model Setup** window. The rest is pretty much the same with the exception of a different set of performance measures available for display (area under ROC curve, lift, etc.) and log-odds interpretation of the predicted response.

In the following sections we provide an in-depth look at the inner workings of the GPS engine. Those may dramatically expand your understanding of the key principles involved as well as help building superior models by proper adjustments to the fine engine control parameters.

A Practical Overview of GPS

GPS or *Generalized PathSeeker* is a highly specialized and flexible regression (and logistic regression) procedure developed by Jerome Friedman (the co-creator of CART and the developer and inventor of MARS and TreeNet, among several other major contributions to data mining and machine learning). GPS is a "regularized regression" procedure meaning that it is designed to handle modeling challenges that are difficult or impossible for everyday regression. For example, GPS is especially well suited to build models when:

- there are many more columns (predictors) than rows (observations);
- the predictors available may be extremely highly correlated with each other;
- the goal is to find the most compact model yielding acceptable performance.

GPS strengths include high speed processing of huge numbers of predictors such as are found in bioinformatics and text mining, and increasingly in models of consumer behavior leveraging web activity and social network data. Predictive models using such data must be able to deftly handle from tens of thousands to possibly millions of predictors. GPS is also a capable variable selection machine sifting a possible small subset of predictors from a huge pool of candidates.

Modern data mining is increasingly required to deal with the uncomfortably many predictors, both when the number of rows is less than the number of columns and when the number of rows is plentiful. In general, a challenge of dealing with such data is that the parameter estimates can be highly sensitive to the data used; working with a different sample thought to be from the same population and small changes in the existing training set can cause substantial changes in specific coefficient estimates. GPS provides a mechanism for dramatically reducing this variability, meaning that results will typically remain stable instead of volatile.

Before using GPS as a data mining engine it is important to understand its limitations:

- GPS is ultimately a regression procedure that builds a linear model that is additive in its predictors. This means that GPS cannot on its own discover nonlinearities or interactions in data unless you prepare special predictors embodying such model flexibility.
- GPS cannot on its own handle missing values and will enforce row deletion of data when missing values are encountered.

In both of these senses GPS has limitations that are identical to that of traditional regression; GPS shares the rigidity that makes linear regression so often unsuitable for modern data analytics.

Why do we offer GPS inside of Salford predictive modeler then? A better question might be: why did Jerome Friedman, one of the most important contributors to data mining, bother to develop a new procedure that is more like the old methods he has helped to displace, and more unlike the series of machine learning advances he has been making since 1974? The answer, quite simply, is that Friedman never really intended for GPS to be used directly on raw data the way regression is normally used. Instead he developed GPS as a second stage modeling method that is used to repackage and improve the results that come from his flagship methods of CART, MARS, and TreeNet. In other words, GPS was developed to work in tandem with machine learning methods and was never intended to be "just" a modern regression package.

An important way in which GPS resembles our other data mining engines is that it offers not just one model but an entire sequence of different models. One of these models will display best results on your preferred performance measure and details relevant to this model are presented by default. But you will clearly see the options available to go with a different model, for example, a simpler model involving fewer predictors with little loss of predictive accuracy. You might also be looking at both squared error and absolute error loss in a regression model and elect to go with a model that gives you an attractive balance on both.

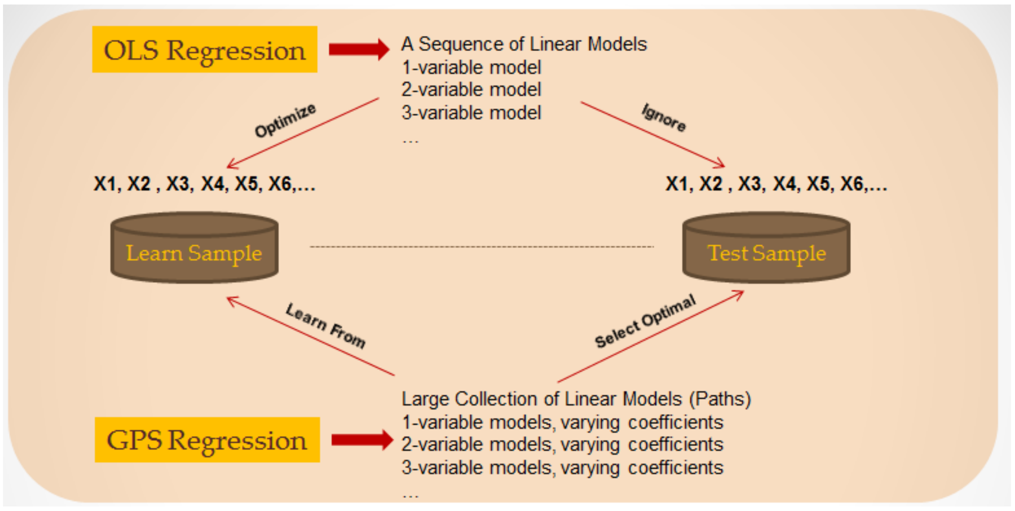
If your model is a regression then the model, however selected, will look just like the classical textbook regression of conventional statistics. The model is constructed as a weighted sum of your predictors with the weights being regression coefficients. Thus model will always look like:

$$Y = C_0 + C_1 X_1 + C_2 X_2 + \dots + C_k X_k$$

where the C_i are the estimated regression coefficients and the X_i are the predictors selected into the model.

✓ Unlike the textbook regression model we do not report standard errors and t-statistics for the coefficients; the coefficients are reported without further adornment and we rely on the fact that this model was ultimately selected by the learning machine as sufficient reason to stick with the details. You can get confidence intervals via bootstrap resampling which can be requested and set up on the **GPS Advanced** tab.

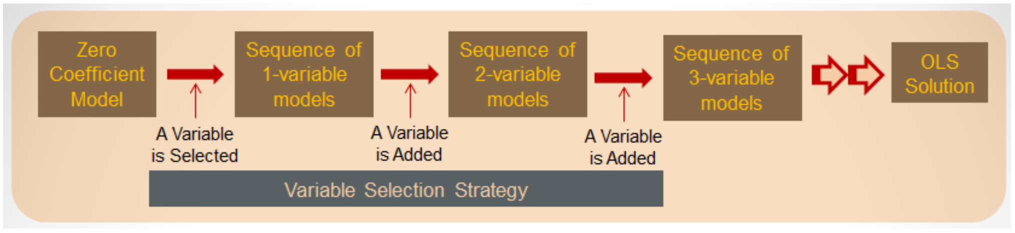
GPS (Generalized Path Seeker) can be viewed as a clever way to dramatically expand the pool of potential linear models based on the learn sample. Unlike conventional OLS regression, which can only build single models with the specified number of predictors already optimized for the learn sample while completely ignoring the test sample, GPS produces a vastly superior set of models by offering a much larger number of coefficient configurations.


OLS Regression
A Sequence of Linear Models
1-variable model
2-variable model
3-variable model
…
Optimize
Ignore
X1, X2 , X3, X4, X5, X6,…
X1, X2 , X3, X4, X5, X6,…
Learn Sample
Test Sample
Learn From
Select Optimal
Large Collection of Linear Models (Paths)
1-variable models, varying coefficients
2-variable models, varying coefficients
3-variable models, varying coefficients
…
GPS Regression

The test sample can therefore be used to identify the model configuration with the best performance given the user constraints on the model size and complexity.

✓ Other than the task of optimal model selection, the test sample is not used in the model construction thus alleviating the problems of over-fitting generally arising in OLS regression.

Internally, GPS constructs the collection of models by creating a path as a sequence of iterations (steps) in the space of coefficients. This can be visualized as follows:


Zero Coefficient Model
Sequence of 1-variable models
Sequence of 2-variable models
Sequence of 3-variable models
OLS Solution
A Variable is Selected
A Variable is Added
A Variable is Added
Variable Selection Strategy

The path therefore effectively connects the zero coefficients intercept-only model on one end with the fully developed OLS solution on the other end by gradually modifying values of the coefficients as well as introducing more and more variables.

At each step along the path, one and only one coefficient is adjusted. If it is an already non-zero coefficient, the model size remains the same; otherwise, when a previously zero coefficient is being adjusted a new variable is effectively introduced into the model.

The *Variable Selection Strategy* plays the key role in determining at which point along the path a new variable enters into the model and also how the coefficients are adjusted. Thus at the current step, a decision is made which coefficient will be modified next according to the currently imposed variable selection strategy. A number of different strategies can be conceptually introduced and there exists vast technical literature dedicated to the subject. The key differences arise in how to balance the stronger variables introduced earlier during the modeling process versus weaker variables introduced later. Some strategies result to very sparse solutions with only a handful of variables selected, others try to include all available predictors while simultaneously shrinking the coefficients towards zero to allow varying degree of regularization.

GPS uses the elastic family of penalty functions as the mathematical tool to impose different variable selection strategies. The actual theoretical details are not important for the current discussion; the only relevant part is that each member of the elastic penalty family is defined by a single real numeric parameter called **Elasticity**.

Elasticity can be set to any real number between 0 and 2 (inclusive) and mathematically imposes variable selection strategies with varying degree of sparseness on the resulting path solutions.

The following elasticity values can be considered as archetypal:

- Elasticity = 2 – fast approximation to the Ridge Regression, introduces variables as quickly as possible and then jointly varies the magnitude of coefficients. Produces the least sparse solutions favoring uniform shrinkage instead.
- Elasticity = 1 – fast approximation to the Lasso Regression, introduces variables sparingly letting the current active variables develop their coefficients. Produces reasonably sparse solutions.
- Elasticity = 0 – fast approximation to the Stepwise Regression, introduces new variables only after the current active variables were fully developed (basically meaning the conventional OLS solution). Produces the sparsest solutions, especially on the earlier parts of the path.

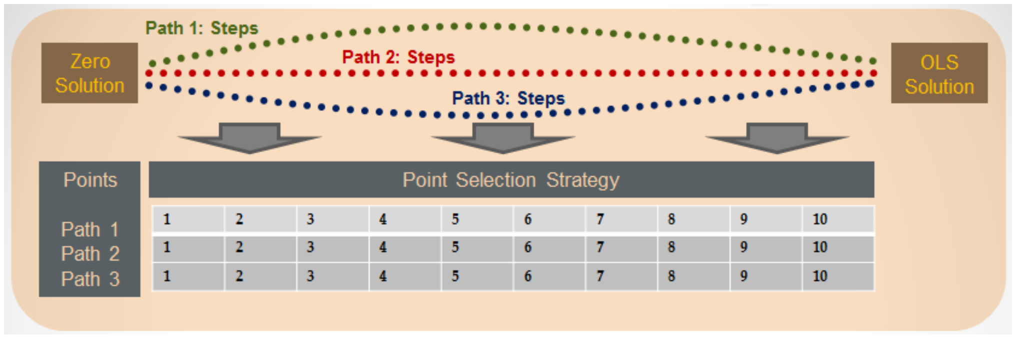
Regardless of the actual value of elasticity selected, any path will eventually reach (at least in theory) the OLS solution in the complete set of predictors. The key difference is how the path gets to that point and how aggressively the coefficients are introduced/adjusted in various variables.

✓ In practice, paths spawned by the sparser strategies may terminate prematurely due to user imposed limits on the number of steps (iterations), adjustment rate, as well as other factors, like presence of collinear predictors, numeric convergence issues, etc.

At each step, a coefficient is being adjusted based on fancy mathematical analysis of the current set of gradients. The actual amount of adjustment can be further controlled by **Learn Rate** – a user specified parameter between 0 and 1, such that the smaller values force smaller updates at each step. Thus smaller learn rates result to smoother paths in the parameter space but may require significantly larger number of steps until the convergence to the OLS solution is reached.

Typically, GPS will try several different elasticities based on user specified values. Because each path may have different numbers of actual steps completed (different elasticities will impose different values of updates along the path) additional measures are needed to facilitate model extraction and comparisons. Each path will typically include thousands of steps each being a potentially new regression solution. Reporting and storing all of the solutions may therefore impose unnecessary burden on the time, compute, and storage resources.

Therefore, GPS has an additional layer of model extraction called **Points**. Only a fixed user specified number of points (models, a.k.a. coefficient sets) will be extracted from each path for further processing. This can be visualized as follows:



For example, is the user sets 200 points (the default), only 200 models will be selected into the internal pool of models, even though the path itself may have an arbitrary number of steps.

✓ GPS employs clever interpolation techniques to guarantee the user specified number of points at all times, even when the number of steps along the path is smaller than the number of points requested.

✓ On the diagram above, only 10 points are requested while the dots represent individual steps, which could be many.

Aside from the fixed number of points requested, the *Point Selection Strategy* can be influenced by the positive **Skew** parameter:

♦ Skew = 1 – samples points on a uniform grid imposed on the learn sample accuracy in terms of MSE (regression) or Cross-Entropy (logistic regression).

♦ Skew > 1 – biases the sampling grid towards more accurate side, thus increasing the share of larger models (larger models always have better performance on the learn sample).

♦ Skew < 1 – biases the sampling grid towards less accurate side (on the learn sample), thus emphasizing reduced model complexity.

To summarize, GPS will construct a number of different paths based on the user supplied list of path building parameters (elasticity and learn rate), it will then extract a fixed user specified number of models (points and skew), these models will constitute the internal pool of candidate models referred to earlier, they will vary by size (but not necessarily contiguously) and magnitude of coefficients. Each model from the pool will then be evaluated based on a number of performance criteria (one set for regression and another set for logistic regression) based on both learn and test samples. The user can then use the flexible GUI interface to view, score, and translate different models from the pool to serve his particular needs (performance criterion of interest, model size versus model accuracy tradeoff, etc.).

Within SPM to run a GPS model you need to follow the same steps as you would for any of our analytical engines:

♦ Select an analysis data set arranged as rows (observations) and columns (potential predictors). With GPS you can theoretically work with many fewer rows than you require for a CART model. You could develop a model with as little as three rows of data although you could hardly have much confidence in the results.

♦ Select a target (dependent variable), almost always a numeric variable.

✓ If the target has more than two distinct values you will have no choice but to run a regression analysis. With two distinct values you can opt for a logistic regression model. The logistic option is preferable if you want to predict the probability of an event or outcome but it runs considerably slower than the regression option.

♦ Identify the set of predictors which GPS will be permitted consider for inclusion in the final model. As always, exclude inappropriate predictors such as ID variables, clones of the target, data derived from the future (unless you intend to do this). Categorical predictors are automatically expanded into a set of dummy indicators (a 0/1 variable for each level), and these indicators are then treated as unrelated variables.

♦ Choose a TEST method – in this respect GPS follows the mold of data mining models generally. We do not (fully) trust results derived from the training data and hence need to rely on test data to provide us with an honest estimate of true model performance. If you do not have sufficient data to be able to afford a true test partition you always have the option of using cross-validation.

GPS operates in two distinct modes depending on your selection of analysis method on the **Model Setup** tab. Your choices are either **Regression** or **Logistic Binary**. The former can be used for any numeric dependent variable including numeric variables with just two values (e.g. 0 and 1). The latter can be used with any variable with just two distinct values whether numeric or text.

✓ Thus a variable RESPONSE$ with possible values "YES" and "NO" can also be used as target for this mode of GPS.

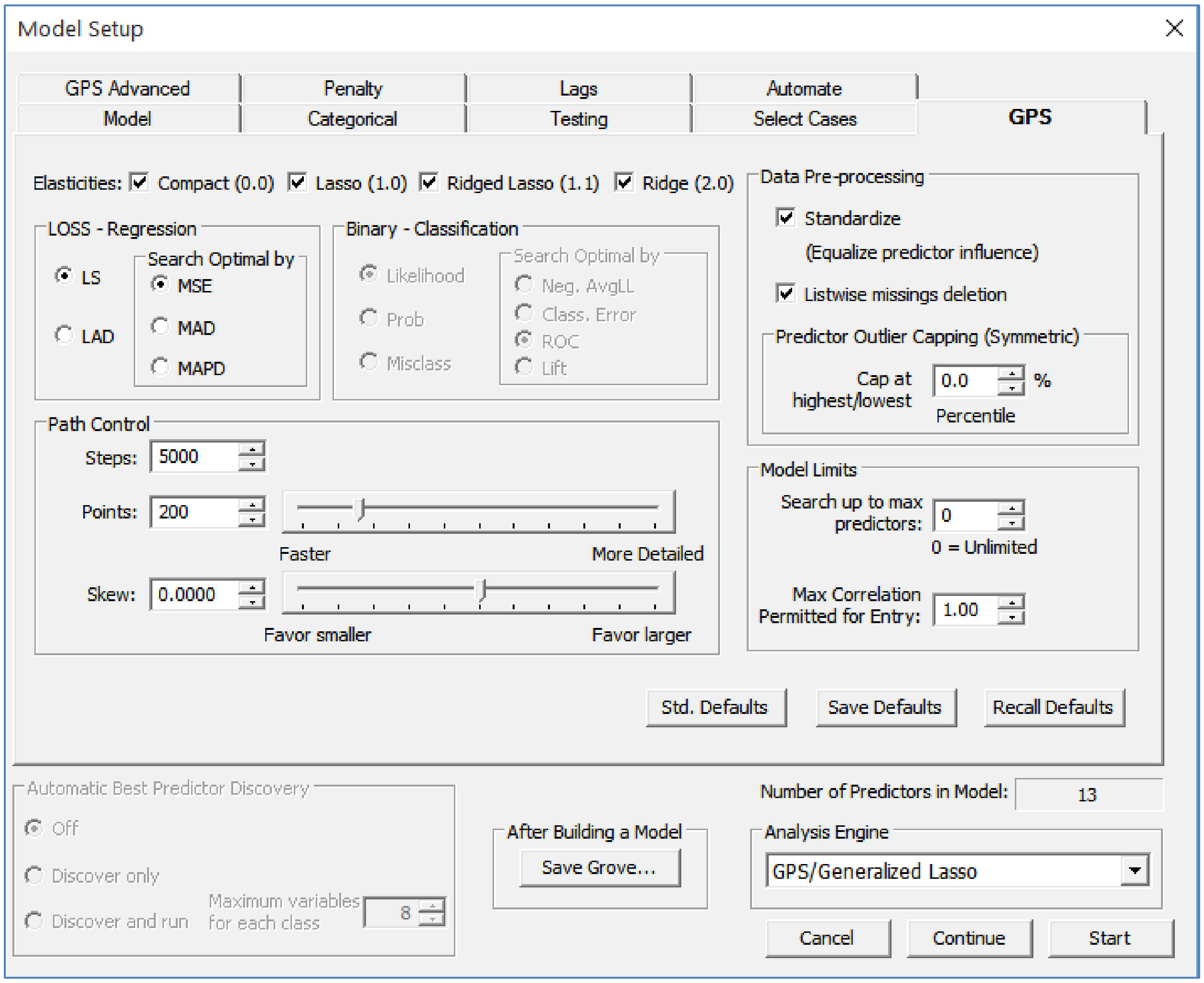
The logistic binary mode is the preferred analysis mode for binary targets and is essential when the less common class falls much below 30% of the data. You are likely to obtain very unsatisfactory results for a rare target if you treat it as an everyday regression.

💣 Be aware though that the binary logistic is substantially slower and noticeably so with large data sets.

We now provide a quick summary of every GPS model setup parameter available to the user in SPM.

GPS Model Setup

The **GPS** tab of the **Model Setup** window provides the key engine settings.


Model Setup

Tabs: GPS Advanced | Penalty | Lags | Automate | Model | Categorical | Testing | Select Cases | **GPS**

Elasticities: [✓] Compact (0.0) [✓] Lasso (1.0) [✓] Ridged Lasso (1.1) [✓] Ridge (2.0)

LOSS - Regression: [✓ LS] LAD
Search Optimal by: [✓ MSE] MAD MAPD

Binary - Classification: [✓ Likelihood] Prob Misclass
Search Optimal by: Neg. AvgLL Class. Error [✓ ROC] Lift

Data Pre-processing
[✓] Standardize (Equalize predictor influence)
[✓] Listwise missings deletion
Predictor Outlier Capping (Symmetric): Cap at highest/lowest 0.0 % Percentile

Path Control
Steps: 5000
Points: 200 (Faster – More Detailed)
Skew: 0.0000 (Favor smaller – Favor larger)

Model Limits
Search up to max predictors: 0 (0 = Unlimited)
Max Correlation Permitted for Entry: 1.00

Std. Defaults | Save Defaults | Recall Defaults

Automatic Best Predictor Discovery: [✓ Off] Discover only Discover and run
Maximum variables for each class: 8

After Building a Model: Save Grove...

Number of Predictors in Model: 13

Analysis Engine: GPS/Generalized Lasso

Cancel | Continue | Start

Standard Elasticities group

Located on the top of the window and includes four suggested archetypal **Elasticities** to try: Compact, Lasso, Ridged Lasso, and Ridge.

✓ Ridged Lasso eliminates some of the potential problems of the traditional Lasso related to the sharp singularities of the penalty function while preserving the general Lasso variable selection strategy.

Path Control group

Organizes key controls influencing path construction and points extraction:

♦ **Steps** – the maximum number of iterations for the path construction.

✓ It is possible for a path to finish (converge to the OLS solution) before the maximum number of steps is reached.

- Setting steps to a very small number may terminate path development prematurely and produce inferior models.
- **Points** – the fixed number of coefficient sets to be extracted from each path into the internal pool of available models. The slider on the right allows quick and easy adjustments to the value.
- Each additional point consumes time and resources, use your judgment to find the optimal balance.
- **Skew** – skews the point selection towards the beginning or the end of the path (see the earlier discussion).

Data Preprocessing group

Organizes controls responsible for modeling data preparation immediately before it is seen by the GPS engine:

- **Standardize** – standardizes all available predictors (mean of zero and variance of one). Because the internal GPS implementation is based on generalized gradient descent in the coefficient space, standardizing variables "evens the play field", thus allowing individual predictors equal chances of being selected. When the standardization is turned off, GPS will tend to favor large variance predictors, which is typically undesirable.
- **Listwise missings deletion** – unlike other engines available in SPM, GPS has no built-in ability to handle missing values. With this option turned on, all records with missing values in at least one modeling variable will be deleted. Alternatively, (when this option is off), GPS will use median imputation for all missing values.
- **Predictor Outlier Capping** – replaces a fixed percentage of outliers in each predictor (symmetrically) by the corresponding percentile. This may improve computation robustness of the model building step.

LOSS – Regression group

Used when modeling continuous response (regression), influences the default optimal model selection done automatically by the GPS engine. The **LS/LAD** control is used by the engine when the slow performance search algorithm is turned off, while the **Search Optimal by** control is used by the engine when the slow performance search algorithm is turned on.

- ✓ This control is mostly relevant to automated runs relying on default optimal model automatically suggested by the engine. In the GUI, the user may select any model of interest interactively thus bypassing the engine recommendation.
- ✓ This rather confusing feature will be streamlined in the next engine update

Binary – Classification group

Used when modeling binary response (logistic regression), influences the default optimal model selection done automatically by the GPS engine. The **Likelihood | Prob | Misclass** control is used by the engine when the slow performance search algorithm is turned off, while the **Search Optimal by** control is used by the engine when the slow performance search algorithm is turned on.

- ✓ This control is mostly relevant to automated runs relying on default optimal model automatically suggested by the engine. In the GUI, the user may select any model of interest interactively thus bypassing the engine recommendation.
- ✓ This rather confusing feature will also be streamlined in the next engine update

Model Limits group

Imposes additional simple restrictions on the variables that are allowed to enter into a model:

- **Search up to max predictors** – sets the upper limit on the total number of non-zero coefficients allowed in the model. As soon as the number of active predictors on the path reaches this limit, GPS stops looking at additional predictors and only focuses on the selected set.

✓ The default value of 0 imposes no restriction, the final number of variables selected may still be smaller than the complete set because of the natural path termination after the maximum number of steps is reached.

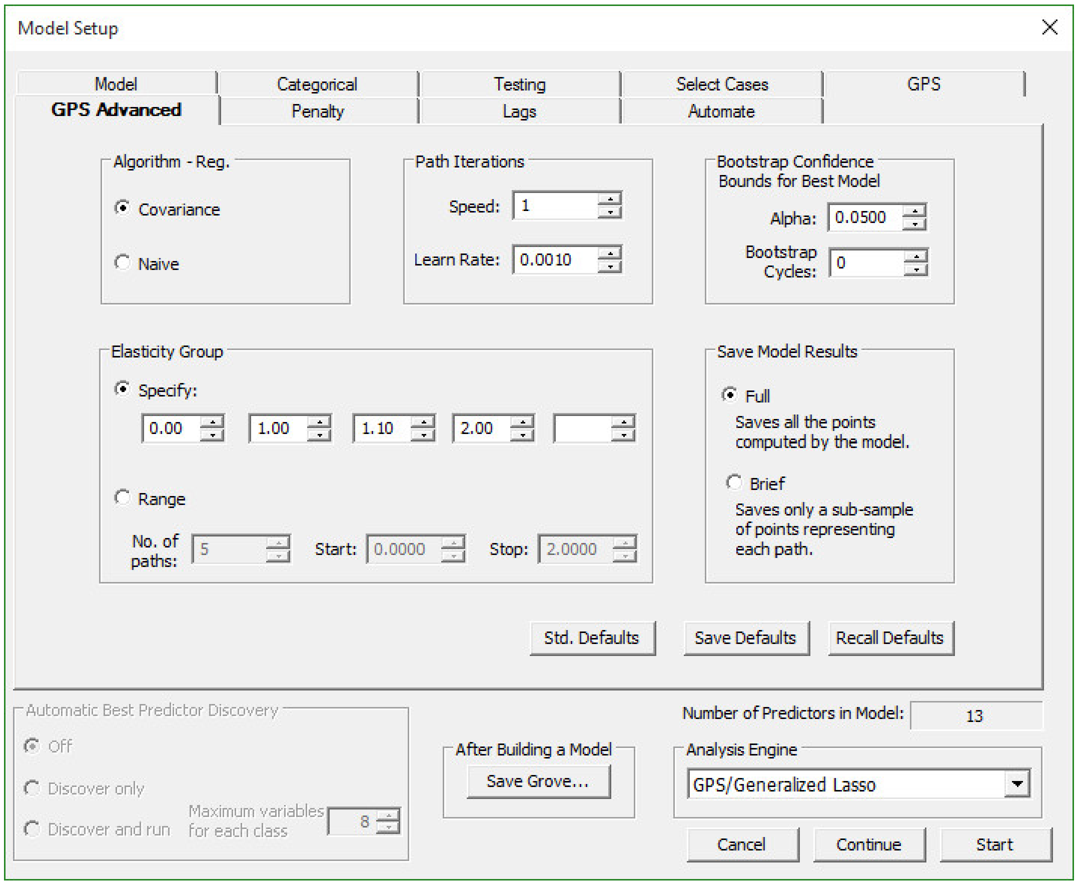
- **Max Correlation Permitted for Entry** – once a new variable is introduced along the path, all variables that have correlating with the variable above the specified limited are excluded from further analysis. This ensures that the active variables selected by GPS are correlated up to the specified threshold. The best way to work with this control is simply to experiment over a range of values, or to use the **AUTOMATE** tab feature for the **MAXCORR** option if you have access to SPM PRO EX.

✓ The default value of 1.0 imposes no restrictions.

Be aware that this control can absolutely prevent the entry of some variables thus limiting the maximum number of possible variables that can make it into the model.

GPS Advanced Model Setup

The **GPS Advanced** tab of the **Model Setup** window provides additional engine settings for extra modeling flexibility.


Model Setup

Tabs: Model, Categorical, Testing, Select Cases, GPS, GPS Advanced, Penalty, Lags, Automate

Algorithm - Reg.: [✓ Covariance] Naive

Path Iterations: Speed: 1; Learn Rate: 0.0010

Bootstrap Confidence Bounds for Best Model: Alpha: 0.0500; Bootstrap Cycles: 0

Elasticity Group: [✓ Specify:] 0.00, 1.00, 1.10, 2.00; Range: No. of paths: 5, Start: 0.0000, Stop: 2.0000

Save Model Results: [✓ Full] Saves all the points computed by the model. Brief: Saves only a sub-sample of points representing each path.

Std. Defaults | Save Defaults | Recall Defaults

Automatic Best Predictor Discovery: [✓ Off] Discover only; Discover and run; Maximum variables for each class: 8

After Building a Model: Save Grove...

Number of Predictors in Model: 13

Analysis Engine: GPS/Generalized Lasso

Cancel | Continue | Start

Algorithm – Regression group

Switches between two different ways to compute internal updates during regression path construction:

- ♦ **Covariance** – uses fast update algorithm based on full covariance matrix, runs very fast but increases memory footprint of the process.
- ♦ **Naive** – uses slow update algorithm but has a small memory footprint.
- ✓ Other than memory/speed tradeoff, the algorithm selection should make no difference on the actual paths constructed.
- ✓ This control is only available for regression, there is no known fast update algorithm for logistic regression which always uses the naïve approach.

Path Iterations group

Provide additional fine controls over path construction:

- ♦ Speed – forces GPS to consider introducing additional variables into the path only at the specific milestones. For example, speed of 10 would only allow a new variable to enter at every 10th step. This allows bypassing some of the expensive internal calculations of the gradients thus speeding the process up. On the other hand, the path itself will change and may produce different results.
- ✓ The default speed of 1 is the slowest.
- ♦ Learn Rate – controls the amount of update at each step. Smaller learn rates will generally require more steps for the path fully develop.

Elasticity group

Provide fine controls over the requested elasticities, way beyond the default four archetypes. One can either type in an arbitrary set of additional elasticity values or generate a uniform grid of elasticities.

- ✓ Up to 100 different elasticities are allowed which is more than enough for all practical purposes.

Save Model Results

Select **Full** to save all the points computed by the model, else **Brief** to save only a sub-sample of points representing each path.

Bootstrap Confidence group

This is experimental albeit powerful feature to run bootstrapping process around the GPS model builder in order to obtain bootstrap bounds on the values of the coefficients.

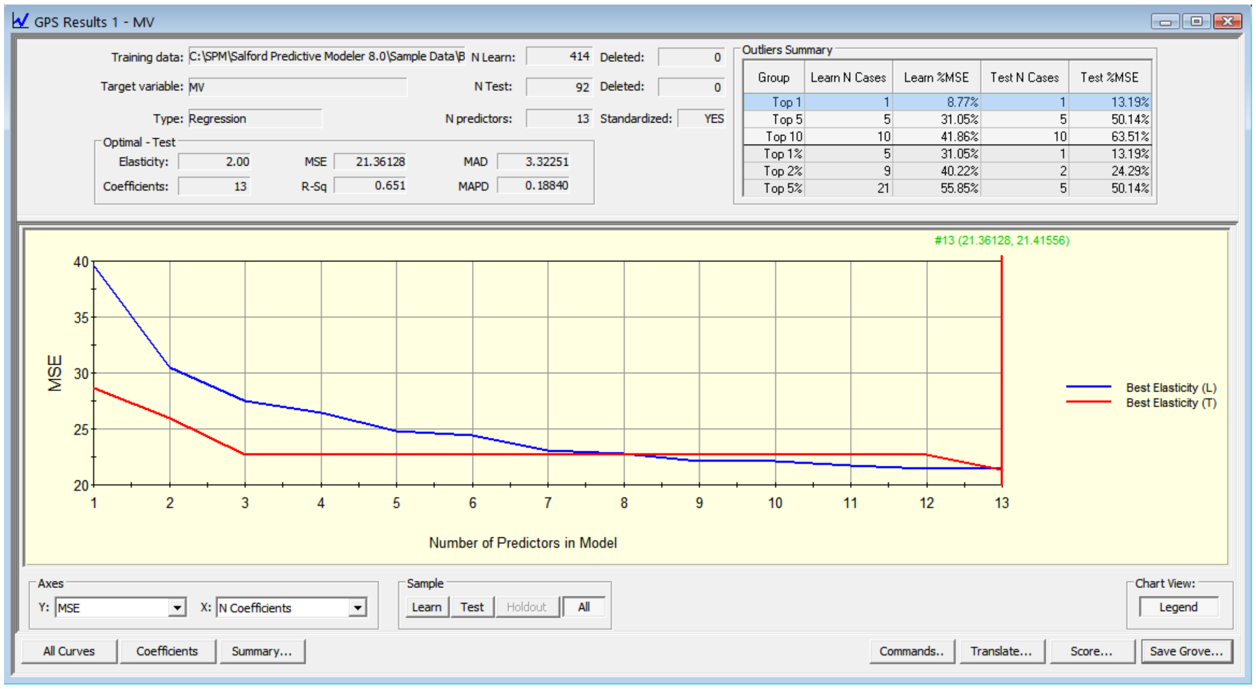
- ♦ **Alpha** – sets the fraction (symmetric) of the extreme values to be truncated in order to obtain the bootstrap confidence intervals for each coefficient.
- ♦ **Bootstrap Cycles** – specifies the number of bootstrap iterations, at least 50 or more will be required to obtain meaningful results.
- 💣 Bootstrapping process can be extremely time and resource consuming endeavor.

This concludes the description of all of the GPS engine control options available in the SPM GUI. Additional controls may be available in the command line. One may type in “HELP GPS” in the command prompt in order to see the full list of available commands and options.

GPS Output Displays

In what follows we provide detailed descriptions of various displays produced as the result of running the GPS engine.

GPS Results: Paths


GPS Results 1 - MV

Training data: C:\SPM\Salford Predictive Modeler 8.0\Sample Data\B N Learn: 414 Deleted: 0
Target variable: MV N Test: 92 Deleted: 0
Type: Regression N predictors: 13 Standardized: YES

Optimal - Test
Elasticity: 2.00 MSE 21.36128 MAD 3.32251
Coefficients: 13 R-Sq 0.651 MAPD 0.18840

Outliers Summary

Group	Learn N Cases	Learn %MSE	Test N Cases	Test %MSE
Top 1	1	8.77%	1	13.19%
Top 5	5	31.05%	5	50.14%
Top 10	10	41.86%	10	63.51%
Top 1%	5	31.05%	1	13.19%
Top 2%	9	40.22%	2	24.29%
Top 5%	21	55.85%	5	50.14%

Basic Stats

Basic stats are located in the upper left part of the window

- **Training data** – points to the modeling dataset
- **Target Variable** – name of the variable used as the target
- **Type** – *regression* for continuous targets or *binary* for classification targets
- **N Learn** – number of learn records in the sample, after possible piece-wise deletion of missing value records
- **N Test** – number of test sample records
- **N predictors** – number of predictors potentially used in the model

Optimal Model Stats

Optimal model stats are assembled under the **Optimal – Test** or **Optimal – Learn** group right below the basic stats.

✓ **Optimal – Learn** is only displayed when the [Learn] button is pressed, otherwise **Optimal – Test** is displayed.

The optimal model is suggested by the engine based on the user-supplied optimality criterion during the Model Setup.

The following measures are reported for the regression:

- **Elasticity** – elasticity for the optimal model path.
- **Coefficients** – optimal model size in coefficients.
- **MSE** – Mean Squared Error of the optimal model predictions.
- **R-Sq** – R-squared of the optimal model.
- **MAD** – mean absolute deviation of the optimal model predictions.
- **MRAD** – mean relative absolute deviation of the optimal model predictions.

The following measures are reported for the binary classification:

- **Elasticity** – elasticity for the optimal model path.
- **Coefficients** – optimal model size in coefficients.
- **ROC** – area under the ROC curve.
- **Class. Error** – classification accuracy.
- **Lift** – lift in the top decile.
- **Cross Entropy** – cross-entropy (log-likelihood).

Outliers Summary

This table reports the top summary of the outliers' analysis described earlier in the manual.

Main Graph Window

The main graph window contains model performance curves for the currently selected performance criterion, sample, and labeling. The optimal model (as reported by the engine) is marked by the green line, while the currently user-selected model is marked by the red line.

The curves are always forced to be monotonic, thus redefining the larger model sizes in terms of smaller model sizes when those result to better performance. For example, if there is no 4-coefficient model with better performance than the 3-coefficient model, the 3-coefficient model is reported even when the 4-coefficient model is selected. Situations like this result to flat performance segments on the performance curves.

Simply click on the model of interest anywhere within the display area to change the currently selected model. Use the **[Left]** and **[Right]** arrow keys on the keyboard to navigate either left or right within the available models.

Axes Controls

Axes controls influence what is displayed in the main graph area

- **Y: -** this selection box specifies the performance measure in focus, nine performance measures are available for regression and four performance measures are available for classification.
- **X: -** switches between Coefficients-based and Points-based indexing along the horizontal axis. Recall that the internal pool of models if always a points based grid of fixed size (supplied by the user in the **Model Setup** window). The coefficients-based display allows the end user to focus only on the most interesting models from the points pool.

Sample Buttons

Sample control buttons select which sample is used for the performance measures shown on the graph as well as which models are shown in the **Coefficient** and **Summary** windows (see below).

- **Learn** – the train sample is used.
- **Test** – the test sample is used.
- **Holdout** – the holdout sample is used.
- **All** – all available samples are used.

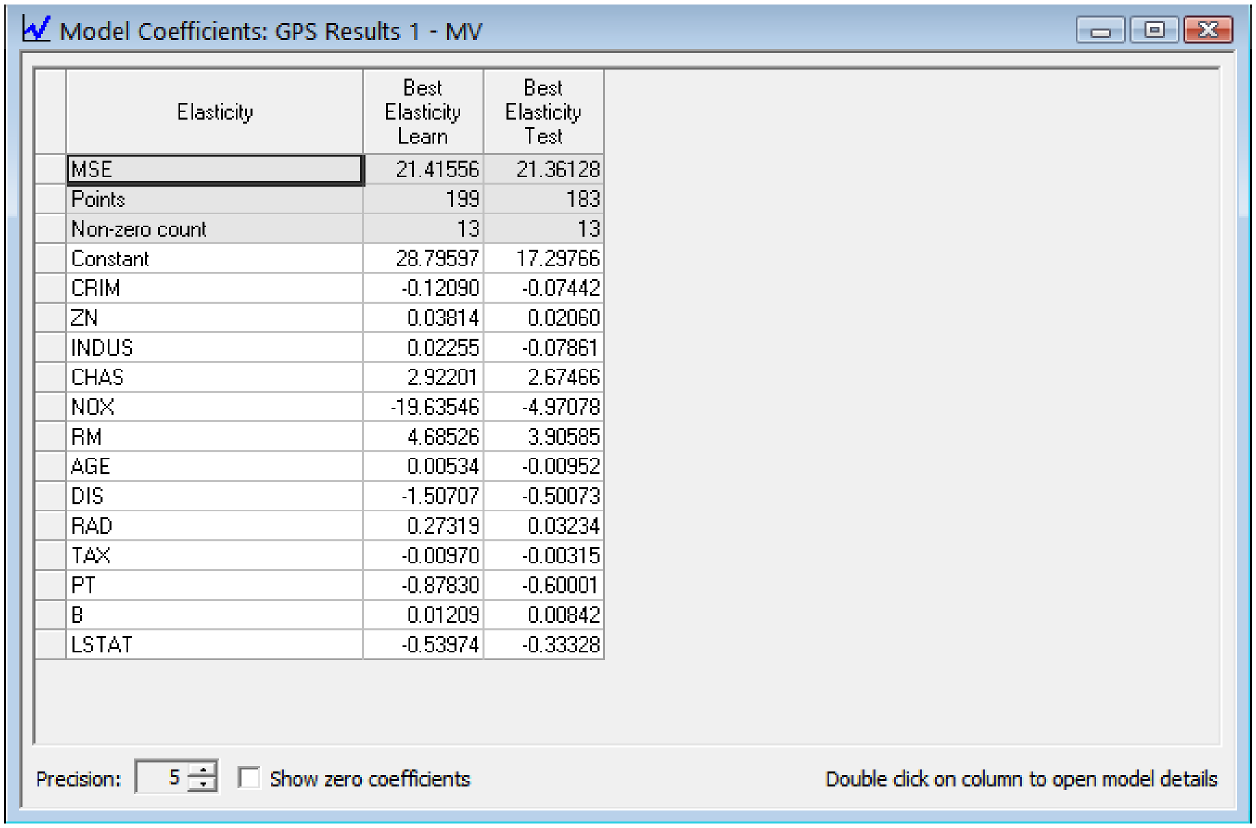
Main Control Buttons

The remaining buttons on the **GPS Results: Paths** window trigger further actions or open up additional reporting displays.

- **Chart View: [Legend]** – turns the graph legend on or off.
- **[All Curves]** – turns on the advanced display mode (see below).
- **[Coefficients]** – opens up the **Coefficients** window.
- **[Summary…]** - opens up the model **Summary** window.
- **[Commands]** – opens the **Command Log** window.
- **[Translate]** – opens the **Translate** window, to translate the model of interest into the language of choice.
- **[Score]** – opens the **Score** window, to score data using the model of interest.**[Save Grove…]** - opens the **Save Grove** window, to save the complete model output into a binary file for further use and portability.

GPS Results: Paths: Model Coefficients

This window opens after the user pressed on the **[Coefficients]** button in the **GPS Results: Paths** window, it reports the model coefficients.


Model Coefficients: GPS Results 1 - MV

Elasticity	Best Elasticity Learn	Best Elasticity Test
MSE	21.41556	21.36128
Points	199	183
Non-zero count	13	13
Constant	28.79597	17.29766
CRIM	-0.12090	-0.07442
ZN	0.03814	0.02060
INDUS	0.02255	-0.07861
CHAS	2.92201	2.67466
NOX	-19.63546	-4.97078
RM	4.68526	3.90585
AGE	0.00534	-0.00952
DIS	-1.50707	-0.50073
RAD	0.27319	0.03234
TAX	-0.00970	-0.00315
PT	-0.87830	-0.60001
B	0.01209	0.00842
LSTAT	-0.53974	-0.33328

Precision: 5 ☐ Show zero coefficients Double click on column to open model details

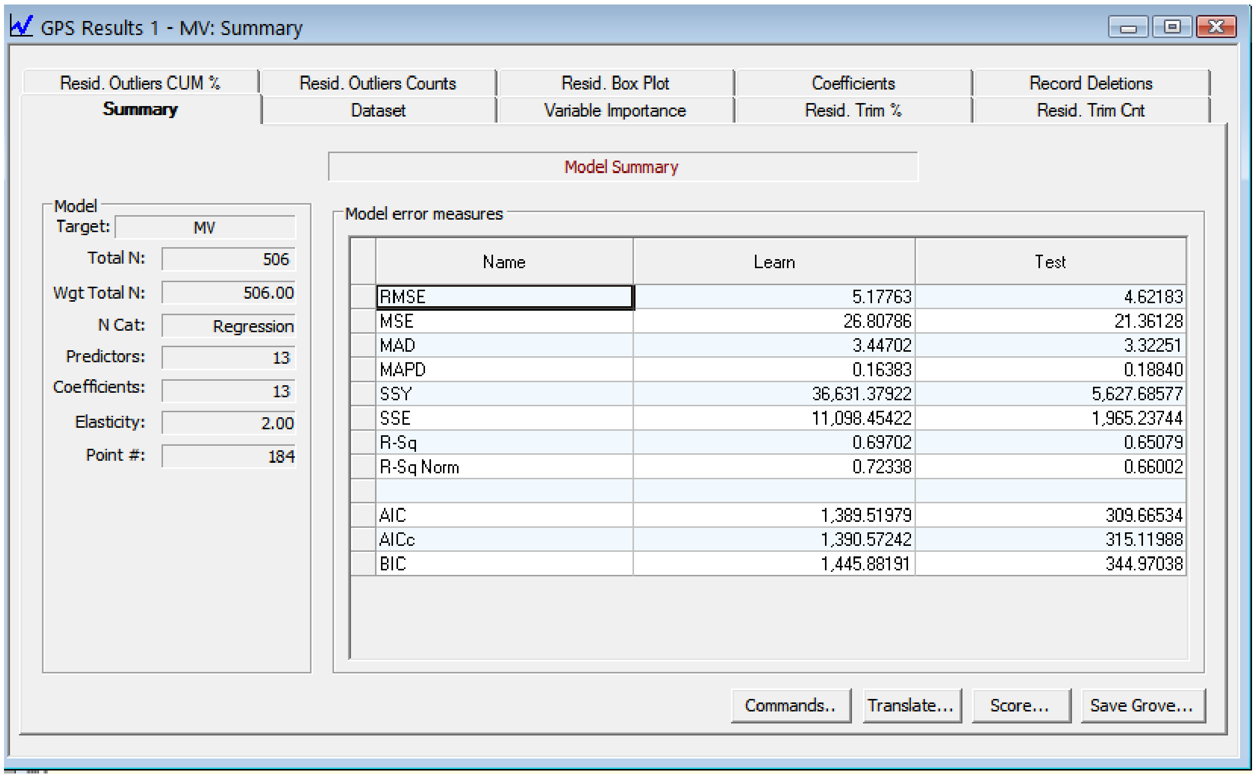
The coefficients are synchronized with the currently selected model, performance measure, and sample partition selected in the **GPS Results: Paths** window.

✓ The Non-zero count value may be smaller than the requested, this means that no model of the selected size was found that has a better performance than the smaller size model shown.

GPS Results: Summary

This window opens after the user pressed on the **[Summary…]** button in the **GPS Results: Paths** window, it reports various summary details of the currently selected model.

The details are synchronized with the currently selected model, performance measure, and sample partition selected in the **GPS Results: Paths** window.



Model Summary

Model

Target:	MV
Total N:	506
Wgt Total N:	506.00
N Cat:	Regression
Predictors:	13
Coefficients:	13
Elasticity:	2.00
Point #:	184

Model error measures

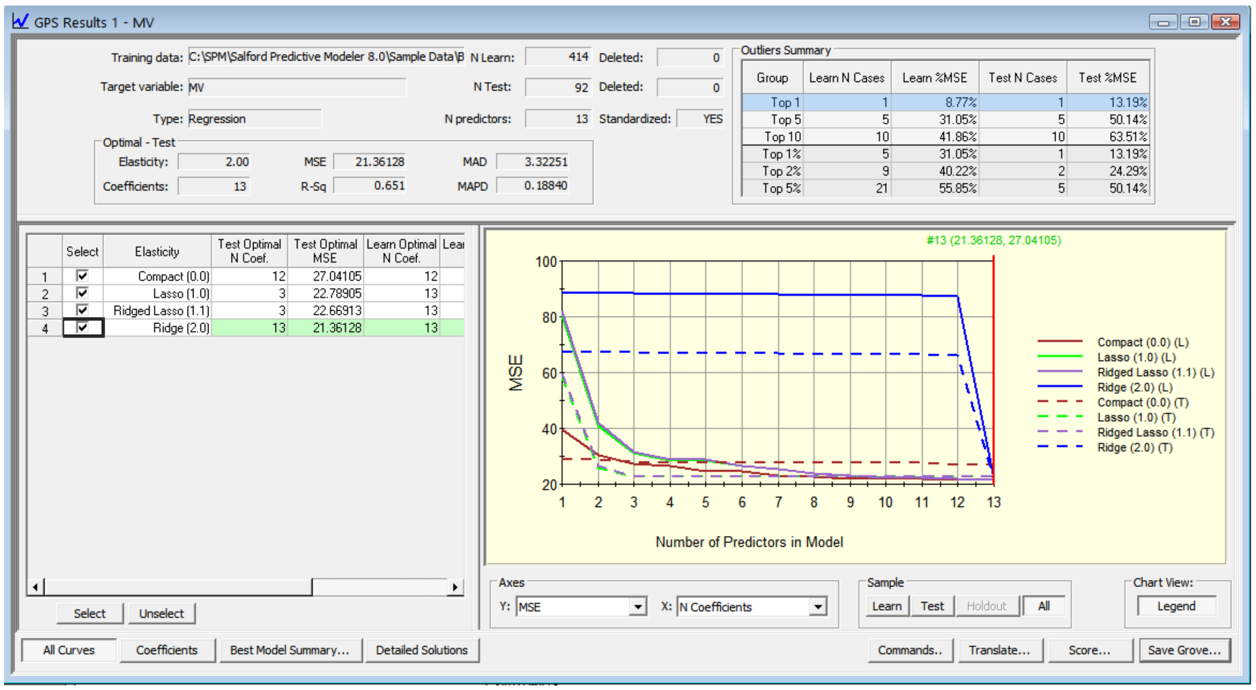
Name	Learn	Test
RMSE	5.17763	4.62183
MSE	26.80786	21.36128
MAD	3.44702	3.32251
MAPD	0.16383	0.18840
SSY	36,631.37922	5,627.68577
SSE	11,098.45422	1,965.23744
R-Sq	0.69702	0.65079
R-Sq Norm	0.72338	0.66002
AIC	1,389.51979	309.66534
AICc	1,390.57242	315.11988
BIC	1,445.88191	344.97038

Most summary tabs are generic and have already been described earlier in this or other manuals. The GPS engine specific tabs will be:

- **Coefficients** – this tab shows the values of the coefficients. Unlike the **GPS Results: Path: Model Coefficients** window, all zero coefficients are removed.
- **Record Deletions** – this tab shows details of the list-wise record deletion due to missing values (if any).

GPS Results: Paths – All Curves

Press the **[All Curves]** button in the **GPS Results: Paths** window to display fine details of the individual paths constructed by the **GPS** engine. This may provide further insights into the inner workings of the algorithm and is usually not necessary for the casual user.



Training data: C:\SPM\Salford Predictive Modeler 8.0\Sample Data\B
Target variable: MV
Type: Regression
N Learn: 414 Deleted: 0
N Test: 92 Deleted: 0
N predictors: 13 Standardized: YES

Optimal - Test

Elasticity:	2.00	MSE	21.36128	MAD	3.32251
Coefficients:	13	R-Sq	0.651	MAPD	0.18840

Outliers Summary

Group	Learn N Cases	Learn %MSE	Test N Cases	Test %MSE
Top 1	1	8.77%	1	13.19%
Top 5	5	31.05%	5	50.14%
Top 10	10	41.86%	10	63.51%
Top 1%	5	31.05%	1	13.19%
Top 2%	9	40.22%	2	24.29%
Top 5%	21	55.85%	5	50.14%

	Select	Elasticity	Test Optimal N Coef.	Test Optimal MSE	Learn Optimal N Coef.	Lea
1	☑	Compact (0.0)	12	27.04105	12	
2	☑	Lasso (1.0)	3	22.78905	13	
3	☑	Ridged Lasso (1.1)	3	22.66913	13	
4	☑	Ridge (2.0)	13	21.36128	13	

Axes Y: MSE X: N Coefficients

All of the original controls already described are still preserved.

To the left of the main graphing area one now finds an additional table where each row represents individual paths internally constructed by the engine:

- **Select** – allows to turn on or off individual paths for display, this is useful in studying specific solutions.
- **Elasticity** – the elasticity parameter of the path displayed on this row.
- **Test Optimal N Coef**. – test optimal model size, on this specific path only.
- **Test Optimal MSE** – optimal value of the selected performance criterion

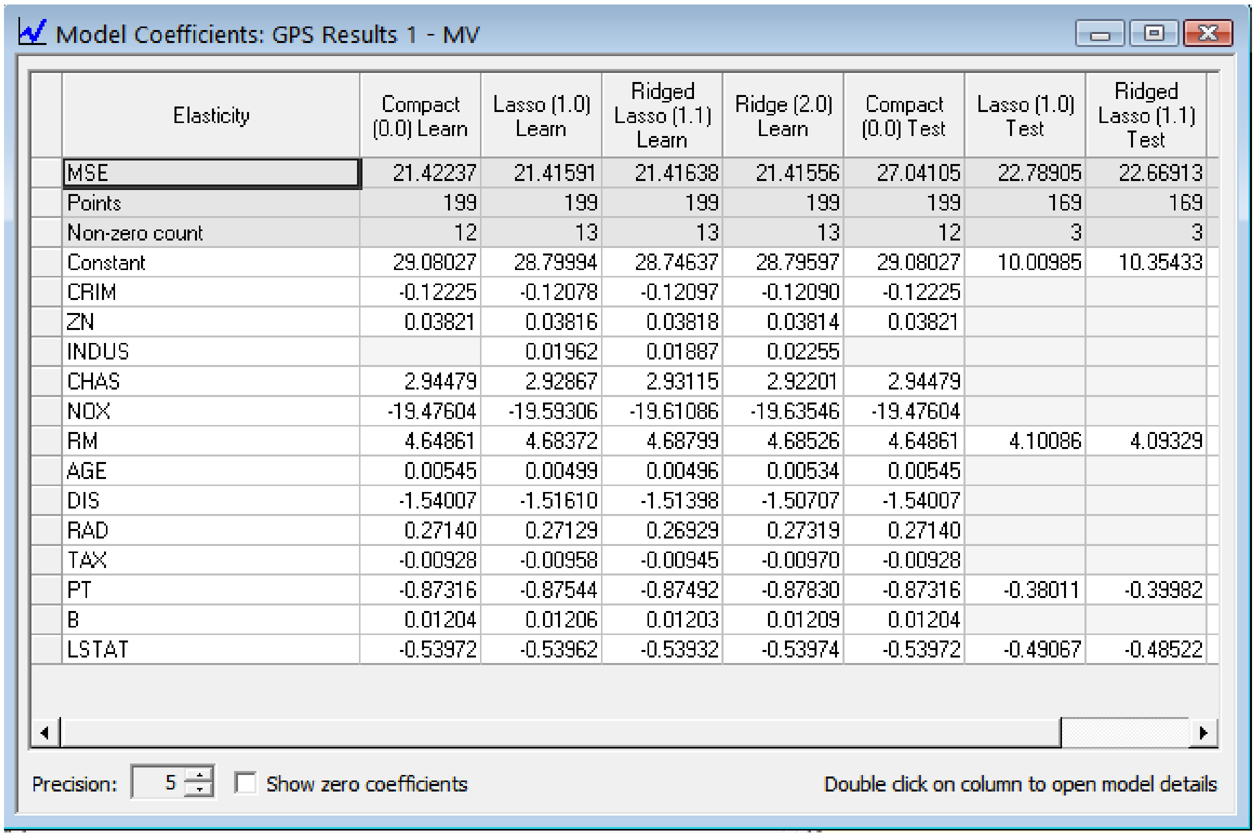
✓ The title of this column depends on the current selected performance measure in the **Axes – Y:** control.

- **Learn Optimal N Coef**. – learn optimal model size, on this specific path only.
- **Learn Optimal MSE** – optimal value of the selected performance criterion

✓ The title of this column depends on the current selected performance measure in the **Axes – Y:** control.

GPS Results: Paths: Model Coefficients – All Curves

This window opens after the user pressed on the **[Coefficients]** button in the **GPS Results: Paths** window, it reports the model coefficients.



Elasticity	Compact (0.0) Learn	Lasso (1.0) Learn	Ridged Lasso (1.1) Learn	Ridge (2.0) Learn	Compact (0.0) Test	Lasso (1.0) Test	Ridged Lasso (1.1) Test
MSE	21.42237	21.41591	21.41638	21.41556	27.04105	22.78905	22.66913
Points	199	199	199	199	199	169	169
Non-zero count	12	13	13	13	12	3	3
Constant	29.08027	28.79994	28.74637	28.79597	29.08027	10.00985	10.35433
CRIM	-0.12225	-0.12078	-0.12097	-0.12090	-0.12225		
ZN	0.03821	0.03816	0.03818	0.03814	0.03821		
INDUS		0.01962	0.01887	0.02255			
CHAS	2.94479	2.92867	2.93115	2.92201	2.94479		
NOX	-19.47604	-19.59306	-19.61086	-19.63546	-19.47604		
RM	4.64861	4.68372	4.68799	4.68526	4.64861	4.10086	4.09329
AGE	0.00545	0.00499	0.00496	0.00534	0.00545		
DIS	-1.54007	-1.51610	-1.51398	-1.50707	-1.54007		
RAD	0.27140	0.27129	0.26929	0.27319	0.27140		
TAX	-0.00928	-0.00958	-0.00945	-0.00970	-0.00928		
PT	-0.87316	-0.87544	-0.87492	-0.87830	-0.87316	-0.38011	-0.39982
B	0.01204	0.01206	0.01203	0.01209	0.01204		
LSTAT	-0.53972	-0.53962	-0.53932	-0.53974	-0.53972	-0.49067	-0.48522

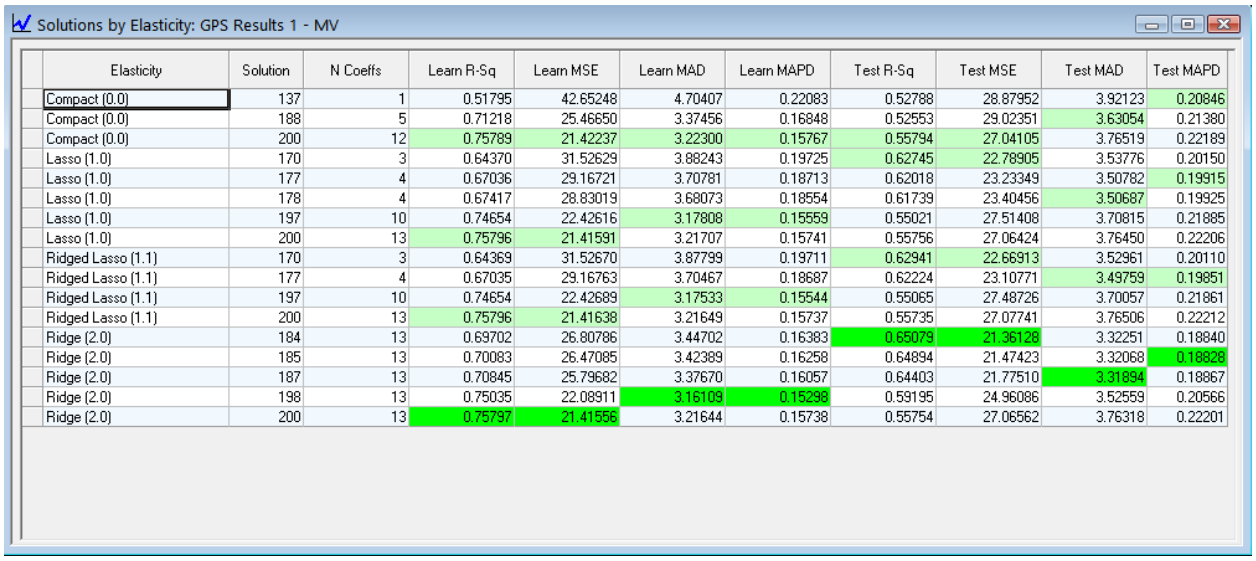
Precision: 5 ☐ Show zero coefficients Double click on column to open model details

Note that now the window reports coefficients by individual paths. By switching to the **X: Points** mode, pressing the **[Learn]** sample button, and then clicking on the various model sizes in terms of points in the main graphing area, one may quickly see how GPS variable selection strategy work under different elasticities. For example, the Compact elasticity tends to squeeze out existing variables before it adds a new variable, while the Ridge elasticity tends to grab all available predictors and work with varying degree of shrinkage on the values of the coefficients.

- ✓ In the **[Test]** sample display mode some model sizes and coefficient values may become “sticky”; this means that GPS can’t find a user-selected model size better than the one found already for small sizes.
- ✓ The table is updated interactively: you may click on the model of interest inside of the main graph to view different models. The currently selected model size is highlighted by red line.
- ✓ The table of coefficients displays the best model in terms of test or learn sample performance found so far, no exceeding the model size (in coefficients) specified on the graph. For example, when the red line on the graph points to the 10 coefficients, the **Lasso** column in the table shows only 3 coefficients – this means that any other configuration of coefficients (up to 10 non-zero ones) along the Lasso path would produce inferior performance.
- ✓ Similarly, you can press the **[Learn]** button to see which sets of coefficients optimize learn performance.
- ✓ You can even change the performance criterion of interest using the **Axes – Y:** selection box, this will automatically update the graph and the coefficient tables.

GPS Results: Solutions by Elasticity – All Curves

This window opens after the user pressed on the **[Detailed solution]** button in the **GPS Results: Paths** window, it reports the optimal models by path (elasticity) and by the performance criterion.



Solutions by Elasticity: GPS Results 1 - MV

Elasticity	Solution	N Coeffs	Learn R-Sq	Learn MSE	Learn MAD	Learn MAPD	Test R-Sq	Test MSE	Test MAD	Test MAPD
Compact (0.0)	137	1	0.51795	42.65248	4.70407	0.22083	0.52788	28.87952	3.92123	0.20846
Compact (0.0)	188	5	0.71218	25.46650	3.37456	0.16848	0.52553	29.02351	3.63054	0.21380
Compact (0.0)	200	12	0.75789	21.42237	3.22300	0.15767	0.55794	27.04105	3.76519	0.22189
Lasso (1.0)	170	3	0.64370	31.52629	3.88243	0.19725	0.62745	22.78905	3.53776	0.20150
Lasso (1.0)	177	4	0.67036	29.16721	3.70781	0.18713	0.62018	23.23349	3.50782	0.19915
Lasso (1.0)	178	4	0.67417	28.83019	3.68073	0.18554	0.61739	23.40456	3.50687	0.19925
Lasso (1.0)	197	10	0.74654	22.42616	3.17808	0.15559	0.55021	27.51408	3.70815	0.21885
Lasso (1.0)	200	13	0.75796	21.41591	3.21707	0.15741	0.55756	27.06424	3.76450	0.22206
Ridged Lasso (1.1)	170	3	0.64369	31.52670	3.87799	0.19711	0.62941	22.66913	3.52961	0.20110
Ridged Lasso (1.1)	177	4	0.67035	29.16763	3.70467	0.18687	0.62224	23.10771	3.49759	0.19851
Ridged Lasso (1.1)	197	10	0.74654	22.42689	3.17533	0.15544	0.55065	27.48726	3.70057	0.21861
Ridged Lasso (1.1)	200	13	0.75796	21.41638	3.21649	0.15737	0.55735	27.07741	3.76506	0.22212
Ridge (2.0)	184	13	0.69702	26.80786	3.44702	0.16383	0.65079	21.36128	3.32251	0.18840
Ridge (2.0)	185	13	0.70083	26.47085	3.42389	0.16258	0.64894	21.47423	3.32068	0.18828
Ridge (2.0)	187	13	0.70845	25.79682	3.37670	0.16057	0.64403	21.77510	3.31894	0.18867
Ridge (2.0)	198	13	0.75035	22.08911	3.16109	0.15298	0.59195	24.96086	3.52559	0.20566
Ridge (2.0)	200	13	0.75797	21.41556	3.21644	0.15738	0.55754	27.06562	3.76318	0.22201

The bright green entries report best overall performances by sample and performance evaluation criterion, regardless of the path. The **Solution** column references the internal point number. Recall, that the points supply a fixed sampling grid on the constructed paths in terms of models. This is set by the user during the Model Setup and defaults to 200.

For example, point 200 sampled from the Ridge path has the best overall Learn MSE of 21.416 while point 184 sampled from the Ridge path has the best overall Test MSE of 21.361.

The light green entries report best performances by sample and performance evaluation criterion, on the given path.

For example, point 170 sampled from the Lasso path has the best Test MSE of 22.79 along the lasso path. Even though this is the best Test MSE Lasso path model, there exists a better Test MSE model, which is the Ridge point 184 mentioned earlier.

Observe that the Compact variable selection strategy in our case does not perform nearly as well as the alternatives, while the Ridge elasticity dominates on every performance statistic.

✓ Results in other examples on other data sets could well be quite different!

GPS Discussion Topics

We conclude our description of the GPS engine by discussing a number of topics to provide further insights into its use and applications.

GPS "Elasticities"

There are three principal modeling strategies built into GPS: Compact, Lasso, and Ridge. The ridge regression mode is the oldest method, having been introduced into statistics in 1970. Its principal motivation was dealing with highly correlated predictors and was designed to "stabilize" the coefficients generated by "shrinking" all the coefficient estimates towards zero. One symptom of highly correlated predictors in conventional regression is the appearance of wildly large coefficient estimates, often hundreds or thousands of times larger (in absolute value) than most of the other coefficients. You might think of ridge regression as a way of preventing any coefficient from reaching unreasonable values while still retaining overall model quality (goodness of fit). The ridge machinery is not just confined to protecting against "wild" coefficients and it operates on all coefficients and even in contexts where none of the coefficients has any tendency to go wild.

A fascinating feature of the ridge regression is that it improves the predictive performance of a conventional regression. Ridge works by optimally "shrinking" coefficients, i.e. making them smaller in absolute value, or moving them towards 0. (We shrink the vector of coefficients as a whole, and not necessarily every element individually.) The beauty of the strategy makes eminent sense: when we fit a regression model to data we are literally making the fit as good as possible, or representing our training data as faithfully as possible. But our training data will almost certainly be different from future data even if both training data and future data are derived from the identical data generation process. By shrinking the regression coefficients, we "detune" the model and make it less reflective of the training data and this should yield better predictive accuracy. The only unanswered question is how much to shrink. Maximal shrinkage would move all the coefficients to 0 leaving us with no model and a prediction based on the mean of the target in the training data; no shrinkage would leave us with the conventional regression model. The official job of the ridge regression is to shrink optimally and this is best accomplished with the help of test data. An important characteristic of the ridge regression as implemented in traditional statistics is that it does not do any variable selection. If you provide the ridge regression with 500 predictors it will keep all 500. Later we will see that GPS goes about computing the ridge regression in a way that actually supports variable selection and this is a very useful extension of the original concept.

Some useful observations about the ridge methodology: if all the predictors are uncorrelated with each other then every coefficient is shrunk (reduced in absolute value) by the same percentage. Given that all the coefficients are reduced in absolute value the variability of the predictions generated must be less than that produced by the standard regression.

The second principal "elasticity" recognized by GPS is the "Lasso". The lasso was introduced into the statistical literature in 1996 and at first was used only by a small number of adventuring data analysts. However, over the last few years the lasso idea has gained considerable traction both within the statistical and machine learning communities. The lasso was explicitly described in the language originally used to motivate the ridge regression but with a slight modification that makes it far more attractive: the lasso shrinks coefficients following the lead of ridge regression but it also selects variables for inclusion or exclusion in the model. It accomplishes this by shrinking each regression coefficient by substantially different amounts and allows for some coefficients to be shrunk all the way to 0. A great advantage of the lasso is that it drives towards simpler models than the ridge regression. Whereas the ridge regression is intended to keep all available predictors the lasso is free to ultimately reject any number of them.

The final elasticity resembles forward stepwise regression and we call it the "Compact" elasticity. Introduced by Jerome Friedman as the principal reason for creating the GPS engine it goes considerably further than the lasso in pushing towards models with as few included variables as possible. Friedman describes it as a method that attempts to replicate as closely as possible the "Best Subset Regression" which considers all possible configurations of predictors (in or out of the model). If you had three possible predictors then best subset regression would test 2^3 or 2 raised to the 3rd power different models, or eight models altogether. Best subset has long been thought to be impractical for all but the smallest problems because the number of possible subsets explodes as the number of predictors increases. Recall that 2 raised to the 32nd power is more than 4 billion, meaning that with 32 predictors we would need to consider 4 billion plus models to find the best subset. With Friedman's novel methodology in GPS we can use approximation to solve this problem with very little effort.

Considering the example of two very highly correlated predictors can help elucidate the differences between the differing modeling strategies. In this case ridge regression will tend to treat the two variables equally, control any wildness in the regression by preventing the coefficients from becoming too large, and ensure that each gets a very similar coefficient in absolute value. In the compact or best subsets model only one of the two variables will ever be included in any candidate best model. And in the lasso we cannot know in advance what will happen; we might get just one of the variables, or one with a large coefficient and the other with a very small coefficient, or perhaps both predictors with very similar coefficients.

Hybrid Elasticities

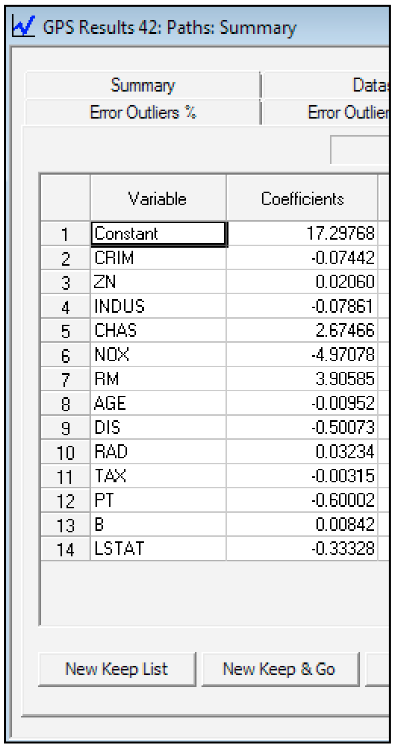
Starting from the three reference elasticities Compact, Lasso, and Ridge, which are represented by the elasticity values of 0, 1, and 2, respectively, GPS allows for creation of elasticities that are essentially a mixture or blend of reference elasticities. Thus, we allow for a special elasticity which we call the "Ridged Lasso" with an elasticity value of 1.1 (which is one tenth of the distance between the Lasso, at 1, and the Ridge, at 2). For the most part there is nothing special about one hybrid over another, but their availability allows us to tweak models to higher levels of performance. Since there is no way to tell which elasticity will work best for any given data set it is recommended that you ask GPS to conduct a grid search. By default, GPS uses the four values of 0.1, 1.1 and 2 but you can set up a more detailed search. For example, you might experiment with a grid of elasticities going from 0 to 2 in steps of 0.1. The intermediate elasticities do not have a special story describing what they are about but if the best predictive model uses one then you should consider using it instead of a named elasticity.

GPS versus Linear Regression

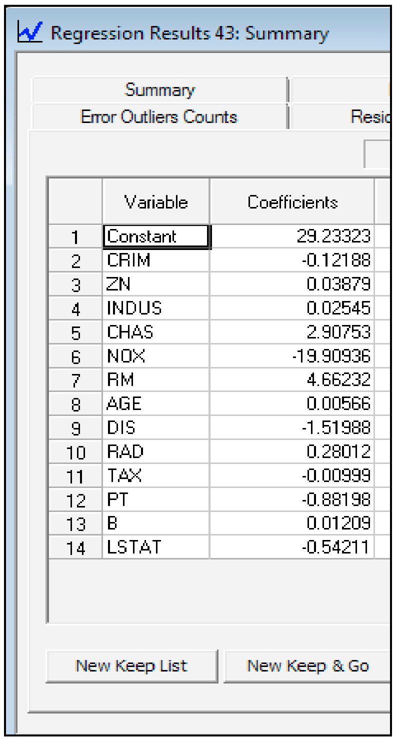
How do the GPS optimal model results compare with everyday regression? To investigate we ran an ordinary least squares regression in SPM (by selecting the “Regression” analysis method on the **Model Setup** dialog) and a standard GPS regression using Ridge elasticity and 200 points.

Below we show the coefficients for the Regression and Ridge Regression results side by side.

Ridge Regression **Conventional Regression**



	Variable	Coefficients
1	Constant	17.29768
2	CRIM	-0.07442
3	ZN	0.02060
4	INDUS	-0.07861
5	CHAS	2.67466
6	NOX	-4.97078
7	RM	3.90585
8	AGE	-0.00952
9	DIS	-0.50073
10	RAD	0.03234
11	TAX	-0.00315
12	PT	-0.60002
13	B	0.00842
14	LSTAT	-0.33328



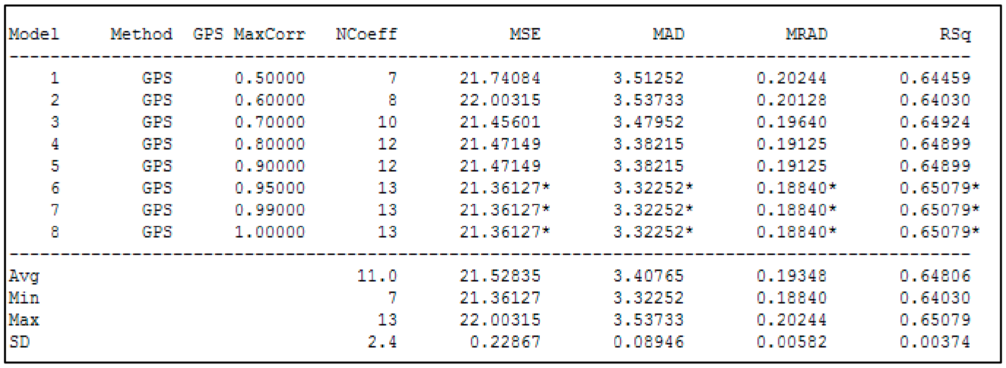
	Variable	Coefficients
1	Constant	29.23323
2	CRIM	-0.12188
3	ZN	0.03879
4	INDUS	0.02545
5	CHAS	2.90753
6	NOX	-19.90936
7	RM	4.66232
8	AGE	0.00566
9	DIS	-1.51988
10	RAD	0.28012
11	TAX	-0.00999
12	PT	-0.88198
13	B	0.01209
14	LSTAT	-0.54211

Looking at the ordinary regression and ridge regression results we can see the typical pattern of coefficient shrinkage, especially for the NOX (pollution) variable for which the regression coefficient goes from -19.91 to -4.97. Observe though that not every coefficient becomes smaller in absolute value and that the AGE coefficient changes sign. This is not an uncommon pattern of changes and in fact has been exploited from time to time when a regression model exhibits a “wrong” sign for a predictor; trying a ridge regression might “fix” the sign (but of course there is no guarantee). Observe also that the test sample MSE drops from 27.07 (ordinary regression) to 21.36 (ridge regression). This is a dramatic example of the benefits of shrinkage. This is not a fluke or some lucky draw of the test sample. Rather, the opposite is true. You would have to be unusually unlucky to see ridge model do worse on previously unseen data than ordinary regression. But you do need to either have a test set partition or use cross-validation to allow GPS to learn how to tune the ridge parameter.

GPS: Maximum Correlation Control

This control, set by default to 1.0, is intended to prevent the addition of new variables into the model when they are too highly correlated with the variables already in the model. Plausible values range from 0.5 to 0.999 and again it is not possible to know in advance which settings might work best for your data. The best way to work with this control is simply to experiment over a range of values, or to use the AUTOMATE MAXCORR option if you have access to SPM PRO EX.

In our example, MAXCORR=0.7 limits the largest model we can construct to 10 variables instead of the 13 used in the default setup. This model is very slightly worse on test MSE than the full 13 variable model. (See the screenshot for the next section). Below is a table of results using different MAXCORR threshold:



```
Model     Method  GPS MaxCorr   NCoeff          MSE          MAD          MRAD          RSq
-------------------------------------------------------------------------------------------
    1        GPS      0.50000        7     21.74084      3.51252       0.20244      0.64459
    2        GPS      0.60000        8     22.00315      3.53733       0.20128      0.64030
    3        GPS      0.70000       10     21.45601      3.47952       0.19640      0.64924
    4        GPS      0.80000       12     21.47149      3.38215       0.19125      0.64899
    5        GPS      0.90000       12     21.47149      3.38215       0.19125      0.64899
    6        GPS      0.95000       13     21.36127*     3.32252*      0.18840*     0.65079*
    7        GPS      0.99000       13     21.36127*     3.32252*      0.18840*     0.65079*
    8        GPS      1.00000       13     21.36127*     3.32252*      0.18840*     0.65079*
-------------------------------------------------------------------------------------------
Avg                               11.0     21.52835      3.40765       0.19348      0.64806
Min                                  7     21.36127      3.32252       0.18840      0.64030
Max                                 13     22.00315      3.53733       0.20244      0.65079
SD                                 2.4      0.22867      0.08946       0.00582      0.00374
```

In this example every best model happened to have been a ridge regression but the MAXCORR setting prevented some of the predictors from entering thus yielding models that contain less than the full set of 13 predictors. When working with larger collections of predictors the MAXCORR control is likely to have its biggest effect on which variables are ultimately included.

✓ MAXCORR control is simply a device that allows exploring other ways to tweak your models to better test sample performance.

GPS: Maximum Predictors Control

This is an important setting when working with very large numbers of predictors (e.g. tens of thousands, hundreds of thousands). When GPS starts it will examine every available predictor in searching for the first variable to enter into the model. It will then examine all available predictors again when considering whether to add a variable to the existing set of predictors (after the first step there is just one variable already in the model). To speed the process of actually getting to a final model and to keep demands for workspace memory from becoming excessive we offer an option to limit the number of variables that can finally enter the model. You are never required to set a limit and we regularly work with many thousands of predictors in this way. But if you run into either memory or run time problems using this control judiciously can be the difference between success and failure. Consider for example working with a 100,000 term vector in a text mining model (meaning you have 100,000 predictors available each representing a word or phrase). It could be very helpful to limit the model to just a few thousand predictors both to speed up the modeling process and also to be able to fit the required analytical workspace into your available RAM.

Getting A Bit More Technical

GPS is known as a "regularized" regression machine because it exerts some control over the details of the model that will be created. A standard regression simply solves some equations and spits out whatever the mathematical solution dictates. In the family of regularized regressions GPS offers we exert control (regulation) by limiting how large in absolute value any regression coefficient can become. While this sounds somewhat innocuous the core idea is related to the notion of variability of results. Consider first the situation in which all of our data has been "standardized", to have mean zero and standard deviation 1, which means that all coefficients can be reasonably compared by size.

The ridge regression is implemented by fitting a regression model to the data that minimizes the sum of squared errors (on the training data) subject to a penalty on the sum of the squared magnitudes of the coefficients. The standard linear regression is intended to minimize

$$\mathrm{SSE} = \Sigma(\mathrm{y}_i - \widehat{\mathrm{y}})^2 \;=\; \Sigma(\mathrm{y}_i - \beta_0 - \sum_k \beta_k \mathrm{x}_{ik})^2$$

on the training data. The ridge regression is intended to minimize

$$SSE + lambda \sum_k \beta_k{}^2$$

where *lambda* is a real positive number representing a fixed strength of the penalty. Recall that a "null" regression model contains only a constant and no regressors and generates as a prediction the mean value of the target in the training data. The model is null in the sense that it generates the same prediction for every record of every data set whether train, test, or holdout. We can always compute this null model easily, and in some cases it does represent the best that we can do. The R-Squared performance measure starts with the sum of squared errors for the null model and reflects the percentage reduction our non-null model achieves.

As soon as we introduce the ridge penalty things change because in addition to the sum squared errors we need to minimize we also have to take the sum of squared regression coefficients into account. These will be multiplied by the value of lambda and if lambda is large the sum of squared regression coefficients can become as large or larger than the sum of squared errors coming from the ordinary least squares regression coefficients. If lambda is very large, the penalty can become so great that the "best" model is indeed the null model where all regression coefficients are set to zero. At the other extreme, if the lambda value is set to zero then we have no penalty at all and are back to unregularized regression. So how should we set lambda?

As usual in data mining we prefer to be data driven and in this case we let the data (real test data or simulated test data via cross-validation) tell us. One way to run a ridge regression is to conduct a grid search over a range of possible lambda values and select the one that yields the best predictive accuracy on test data. With the small data sets typical of applied statistics in the 1970s when the ridge regression was first introduced such searches were practical and fast enough. But for modern data mining problems we need a better search algorithm, which is what GPS provides us. But the principle logic is still the same, with the proviso that we have some special tricks for dealing with very large numbers of predictors (to be discussed below).

Ridge regression produces a "biased" regression meaning that on its predictions for the target variable given a set of predictors will not be centered on the "truth". However, if estimated correctly, its predictions will tend to be closer to the truth on average than an ordinary regression. This is an example of the "bias variance tradeoff" so important in the creation and assessment of predictive models.

A practical matter relating to very large numbers of predictors

GPS allows you to set an upper bound on the number of predictors that will be considered for inclusion in any final model. This is a departure from traditional ridge regression in which all predictors are by definition included, but is essential in the development of practical models. If you happen to have, say, 1 million predictors you are not going to want a model with 1 million regression coefficients.

Most practitioners would start by trying to eliminate a very large number of the potential predictors in order to arrive at a manageable model but GPS will not force you to engage in such pre-filtering. Instead, you start with the specification of a maximum number of predictors that may be included in the model. (You can experiment with several such limits to see if you are on the right track). Recall that GPS works as follows:

It starts by examining all the available predictors and continues to work with all as it slowly builds up a provisional model. GPS works in a forward stepping mode meaning that it starts by including just one predictor in the model and gradually adds predictors. A step in GPS can either add a new variable or refine the coefficient on a variable already in the model. The stepping continues, always scanning all variables during every step until the maximum number of included variables has been reached. Beyond this point GPS stops looking at any variable not already selected into the model and simply refines the model working with this smaller set of variables.

Suppose for example you are working with 100,000 predictors but would like to consider models with fewer than 200 predictors. The GPS will proceed with its forward stepping scanning each of your 100,000 predictors at every step to determine what to do next (include a new predictor or refine an already included predictor). But once GPS reaches 200 included variables it stops scanning outside of these selected 200 predictors in any subsequent model refinement.

Should you set an upper bound on the number of allowed predictors?

If you are working with a modest number of predictors, there is absolutely no reason to impose any limit. However, GPS must maintain a correlation matrix for all the predictors in the model in memory, and if you allow for 10,000 predictors GPS will need to manage a 100 million element matrix (actually just the upper or lower half). In addition, if you allow for a very large number of predictors GPS will need to take the compute time to rescan the available predictors and this could lead to rather long run times. By setting a reasonable upper bound you economize on both workspaces required in RAM and compute time. Your guide to the upper bound might be accepted scientific knowledge about the process you are studying, your own experience, or experimentation with different values of the upper bound.

GPS as a Variable Selection Technique

In general, we would not favor GPS as a variable selection technique as it suffers from the limitations of any additive linear model when it comes to nonlinearity, interactions, and missing values. However, when we are working with trees and nodes extracted from trees these drawbacks disappear and indeed the ISLE procedure does look to be essentially a tree selection technique. In this context Friedman points out that we might want to consider building the optimal ISLE model in two stages:

- Run the two stage TreeNet/GPS model compression analysis to obtain a preferred size of model (number of trees)
- Now start the GPS model again restricting the analysis to just the selected trees and just run a ridge model (elasticity=2)

The reason for this is that GPS combines coefficient shrinkage and variable selection in lockstep. The only way to get smaller models is to increase the aggressiveness of the shrinkage mechanism, meaning that if we start with a large number of potential predictors we might end up over-shrinking the variables that survive. Running the GPS model a second time on a pre-selected set of predictors allows GPS to reconsider the shrinkage and allow for larger coefficients. Friedman actually suggests that the second model could be an un-regularized regression but we think that running the GPS model again in just ridge mode will give you the option to stick with the plain regression results or not. Naturally, you can experiment with this strategy following any GPS model.