Mining the Data Mart: A Case Study with Stock Market Data

James A. Cox, Ph.D, SAS Institute Inc., Cary, NC
Tonya L. Etchison, Ph.D., SAS Institute Inc., Cary, NC

ABSTRACT

This paper provides a case study illustrating how to take your data from the data warehouse through the data mining process. It focuses on data preparation techniques which are applied before the data is imported into Enterprise Miner™ software as well as those that can be applied after the data has been imported into the product. The information is presented in the form of a case study using stock market data that is followed from the data warehouse through the entire data mining process.

INTRODUCTION

Typically, when people are mining data, they are looking for relationships among data in a data mart, but they may not have a good idea of exactly where to look. The data is often a complex mix of the useful and pertinent with the irrelevant and distracting. This paper focuses on how to get the data from your data warehouse into a form that is appropriate to be mined.

To make this paper as useful as possible, it is organized as a case study using stock market data. This data was selected as an example because it is widely known, often analyzed, and complex with subtle relationships that are often difficult to characterize.

ABOUT THE DATA

The data used for this paper was obtained via a Standard and Poor (S&P) COMSTOCK feed that SAS Institute licensed from 1994 to mid-1996. A nightly job created a SAS® data set that contained daily pricing information for about 3200 U.S. stocks. Another SAS data set was created on a bi-weekly basis containing fundamental data on the same stocks. Fundamental data is data that comes from required 10-Q (quarterly) and 10-K (annual) SEC filings by any organization with publicly traded stock. These filings typically contain information about cash flow (statement of operation) and financial condition (balance sheet), both of which are considered fundamental information about a stock.

PREPARING THE DATA

Data mining activities typically work on a data table organized as rows that represent observations and columns that represent variables. Most variables in the table can be placed into three categories:

1. **Target**: The target variable is the variable that is to be predicted. This variable is usually present in the data set used for data mining, but it may not be present when the results are applied to other data sets in the future. For example, a company trying to determine which customers should receive a mailing may have historical information about what factors have influenced customer response in the past. However, they may want to use that information to determine whether a new customer would respond to a mailing or not.

2. **Input**: The input variables are variables that may be used for predicting the target variable.

3. **Irrelevant**: The irrelevant variables in the data set are variables that may provide useful descriptive information, or they may just be noise, but they are not to be included when constructing predictive models for the target.

Before beginning the data mining process, you need to identify the target and input variables and determine whether they are in the proper form for use in predictive modeling. The question that needs to be answered here is how and when to prepare the data for analysis. For example, irrelevant variables need to be identified and excluded from the analysis. Also, input and target variables may need to be transformed or may contain outliers that need to be filtered before beginning the analysis.

There are actually two schools of thought about this process. One school says that the job should be done principally in preparing the data mart in the data warehouse. This method has a few advantages:

If you use this method, the filtering and transformation process can be automatically scheduled to be performed as part of the process of building the data mart. Therefore, the resulting data mart can be smaller and leaner, and the steps do not need to be replicated during the data mining process.

Also, the warehouse administrator may have a better understanding of the underlying data and how to manipulate it and therefore may be in a better position to know how to clean it up.

The other school of thought could be referred to as the "kitchen sink" approach: create a huge table containing every possible variable of interest and then toss that into the data mining enterprise, doing data preparation as part of the mining process. Proponents of this approach point out that often the person analyzing the data understands it best. Hence, transforming or filtering the data at the data warehouse stage may actually remove vital information that may not appear to be important at first glance but may turn out to be crucial to the analysis.

Enterprise Miner™ software is flexible enough to handle either approach. It has the tools necessary for data preparation but can use data which has already been prepared as well.

In this case study, the preparation of the data took place partly during the data mart extraction and partly inside Enterprise Miner™ software. A full description of the data preparation process will be given in the following section.

PREPARING THE DATA MART

Preparation of the data mart began with merging data from two sources: daily pricing history and fundamentals data. Daily pricing history data was included for two dates almost three months apart (April 3, 1996 and June 20, 1996). Fundamentals data was included for the first of the two dates (see Appendices 1 and 2). We decided to predict the price differential between the dates based on the fundamentals information available at the beginning of the period. Thus, the target variable (PRICEDIF) was created for each stock by dividing its price at the second date by its price at the first date. This data was then massaged to create the final data mart (see Appendix 3).

Before importing the data into SAS Enterprise Miner™, the following modifications were made:

- A number of irrelevant variables (e.g., balance sheet date, company name, cusip number, earnings period date, etc.) were removed.
- Several other variables were transformed to make them more useful. For example:
  - ◊ In the source data, the S&P dividend and earnings ranking was stored in character format as A+, B, C, etc. This variable was converted to numeric and used as an ordinal variable for the analysis.
  - ◊ All of the ratings for earnings per share, revenues per share, etc. for each of the past four years were removed. These were replaced by earnings and revenue growth ratings for the last year, the last quarter, the last 5 years, etc.
  - ◊ Several variables that are often used by stock market analysts were created (e.g., return on equity, debt as a percentage of equity, and current ratio).
  - ◊ Earnings yields (earnings per share divided by price) were computed instead of the more commonly used price to earnings ratio (price divided by earnings per share). This index was used because it is more easily analyzed for negative earnings. For example, if a share priced at $10 had earnings of -$0.01, then the price to earnings ratio would be -1000. However, if it has earnings of -$5, then the ratio would be -2. Three different earnings yields were computed for each stock: actual earnings from the past 12 months, actual earnings from the past 6 months with projected earnings for the next 6 months, and projected earnings for the next 12 months.

After the data was imported into Enterprise Miner™ software, additional modifications were made to the data to prepare it for analysis. Table 1 lists the variables that were present in the data set that was passed from the data mart into Enterprise Miner™ software.

Table 1 Variables in the TRAFUND Data Set

Variable Name	Variable Label
BETA	Volatility Indicator
CAPITAL	Capitalization (shares out * price)
CURPEYLD	Last 6 mo. + Next 6 mo. Earnings/Price
CURRATIO	Ratio of Current Assets to Current Liabilities
DEBTEQ	Long Term Debt/Equity
DIVYIELD	% Dividend Yield
EPS1QRT	Earnings Growth Last Qtr.
EPS1YR	EPS Growth Last Year
FUTGROW	% Growth in Earnings Next Year
GRWTH5YR	Annualized Earnings Growth Last 5 Years
INSTITUT	% Institutional Ownership
NXTPEYLD	Next 12 mo. Earnings/Price
ORGPRICE	Price as of April 1, 1996
PBOOK	Price/Assets per Share
PRICEDIF	3-Month Price Differential (Ratio)
PRICSALE	Price/Revenues
PROFMARG	Net Profit Margin
REV1YR	Revenue Growth Last Year
REV3YR	Annualized Revenue Growth Last 3 Years
ROE	Return on Equity
SPRANKING	Converted S&P Div. And Earnings Ranking
SPSTARS	S&P Stars (1-4)
SYMBOL	Ticker Symbol

TRAILYLD	Last 12 Months Earnings/Price

The data set was imported into Enterprise Miner^{TM} software through the Input Data Source node. The first thing that was obvious was that several of the variables had a large percentage of missing values. For example, NXTPEYLD and FUTGROW each have 72% missing values. Only PRICEDIF and ORGPRICE did not have any missing values at all. Missing values will be dealt with later in the case study.

To begin the second phase of data preparation, we examined the data graphically and discovered that there are several extreme values present in this data set (See Figure 1).

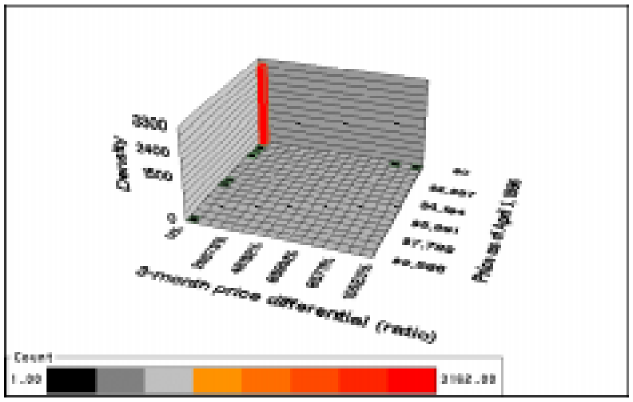


Figure 1 Visualization of Price Differential Data

You can see, for example, that there are two observations that have a price differential of greater than 85,000%. While these may be valid observations, they may severely affect the ability to predict the price differential for the remaining observations. If you re-define the range of values to be displayed in the graph, you can get a better understanding of the relationship between original price and price differential. The modified graph is shown in Figure 2.

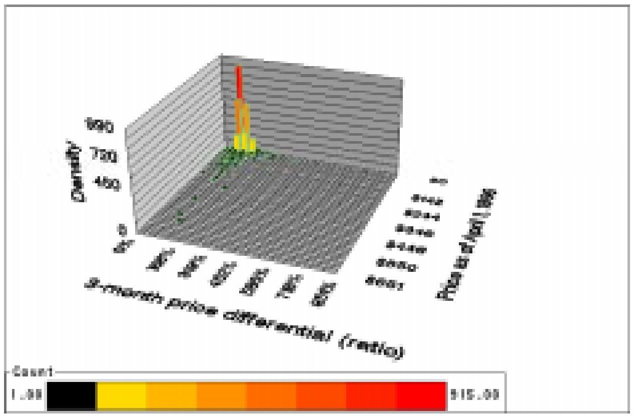


Figure 2 View of Price Differential After Restricting Data Range to Remove Extreme Values

Based on our graphical exploration of the data, we decided to exclude the extreme values of price differential from our predictive modeling exercise. This task was accomplished using the Filter Outliers node.

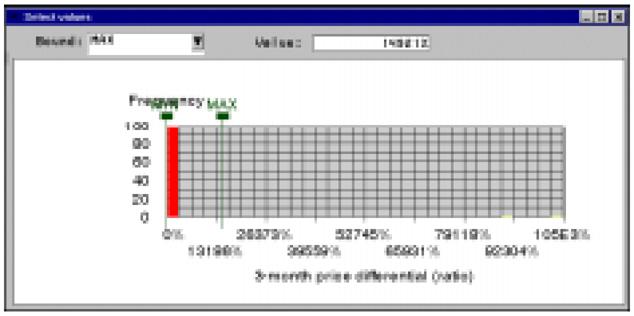


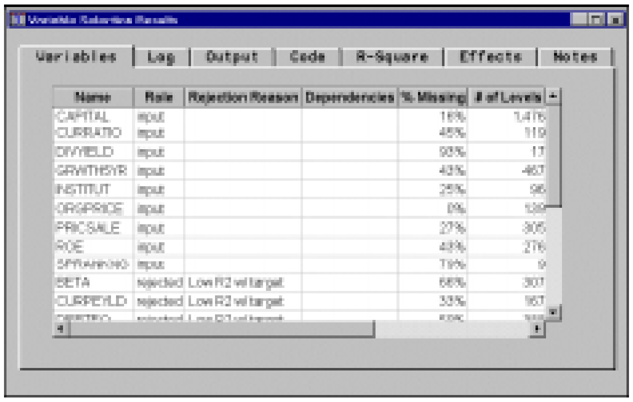
Figure 3 Filtering Observations Using the Filter Outliers Node

After filtering the extreme values for price differential from the data set, we needed to examine the variables to determine which ones were the most important for our analysis and which ones could be excluded from further consideration. Enterprise Miner™ software provides tools to assist you in identifying important variables for your analysis.

For interval target variables, such as price differential, the Variable Selection node provides access to PROC DMINE which gives a quick preliminary assessment of which variables are associated with the target based on a linear models framework.

However, before we could use the Variable Selection node, we needed to impute the missing values in the data set. We did this using the Data Replacement node. The Data Replacement node provides several options for imputing missing values. We tried both mean and median imputation. Since neither method seemed to produce superior results, we used the default mean imputation method for our analysis.

After the missing values were imputed, we ran the variable selection node to get a preliminary assessment of which variables were important for predicting price differential. The variables that were selected are shown in Figure 4.



Name	Role	Rejection Reason	Dependencies	% Missing	# of Levels
CAPITAL	input			16%	1,476
CURRATIO	input			45%	119
DIVYIELD	input			93%	17
GRWTH5YR	input			43%	467
INSTITUT	input			25%	96
ORGPRICE	input			0%	139
PRICSALE	input			27%	305
ROE	input			48%	276
SPRANKNG	input			79%	9
BETA	rejected	Low R2 w/ target		66%	307
CURREYLD	rejected	Low R2 w/ target		33%	167

Figure 4 Variables Chosen by Variable Selection Node

In addition to identifying which variables are important, the Variable Selection node gives us an assessment of how well the chosen regression model is performing. For this particular example, the R^2 value was only 0.02. That is, the linear regression model only explained about 2% of the variability in price differential. This result was probably due to the fact that the relationship between the variables and the target is not linear.

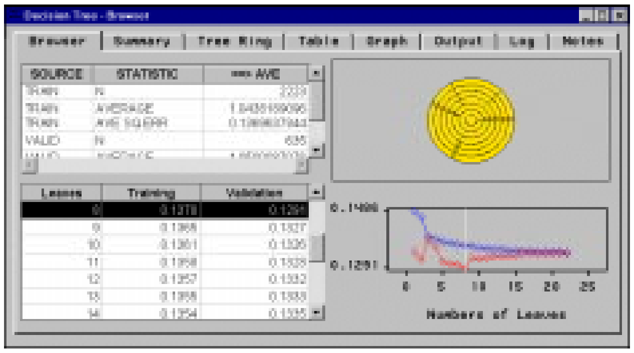
This explanation of this result was substantiated by further graphical exploration of the data and by using the ANOVA16 option in the Variable Selection node. The ANOVA16 option buckets all interval variables into 16 equally spaced buckets and tests the significance of these bucketed variables in the model. In almost all cases, the ANOVA16 variables were preferred to their interval counterparts, indicating that the relationship between the variables exhibits some degree of nonlinearity.

At this point, we needed to make a decision about which modeling technique to use. We could have used neural networks to try to capture the nonlinear relationship between the inputs and the target. However, our goal was to determine which variables were most important for predicting price differential. While neural networks are powerful predictive models, they do not provide a good explanation of which variables were driving the prediction.

Another factor in our decision was the large percentage of missing values present in the data. We decided to use a decision tree model to analyze the data because it can handle the nonlinear relationships between inputs and target and it automatically uses missing values in the modeling process.

The Decision Tree node in Enterprise Miner™ software gives you the option to perform both CHAID and CART types of analyses for interval, nominal, and binary targets. We tried both options and found that we got the best model results when we used Variance Reduction as the criterion for constructing the tree. We also specified that the minimum number of observations that could be contained in any leaf was five. In the Data Partition node, we put 70% of the data into a training data set, 20% into a validation data set, and 10% into a test data set. The validation data set was used to select the final decision tree.

The results from running the Decision Tree node are displayed in Figure 5.



SOURCE	STATISTIC	AVE
TRAIN	N	2223
TRAIN	AVERAGE	1.0435159096
TRAIN	AVE SQ ERR	0.1289837943
VALID	N	635

Leaves	Training	Validation
8	0.1378	0.1291
9	0.1365	0.1327
10	0.1361	0.1326
11	0.1358	0.1328
12	0.1357	0.1332
13	0.1355	0.1333
14	0.1354	0.1335

Figure 5 Decision Tree Results Browser

Using the graph in the lower right corner of the screen, we chose the decision tree that corresponded to the point where the assessment values for the training and validation data sets were approximately equal. We then viewed the actual tree that was created and found that Capitalization (CAPITAL) was the most important variable for predicting price differential. The first split is shown in Figure 6, and you can see that stocks with a capitalization of less than $269 million went into the left branch and stocks with a capitalization of greater than or equal to $269 million went into the right branch.

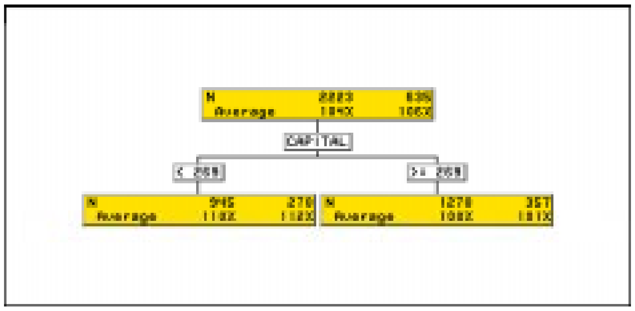


Figure 6 First Split in the Decision Tree

If you follow the left branch of the tree, you see that the original price of the stock (ORGPRICE) and the ratio of current assets to current liabilities (CURRATIO) were also important (Figure 7).

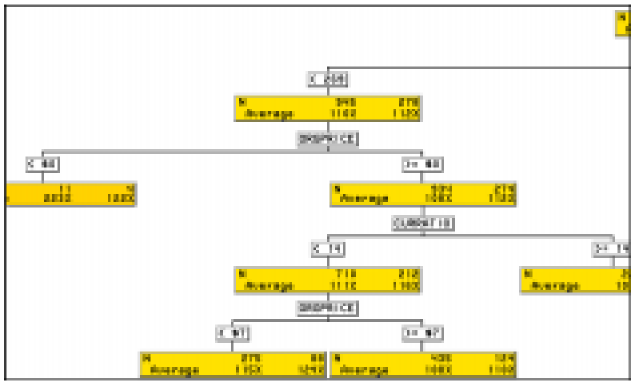


Figure 7 Left Branch of the Decision Tree

If you follow right branch of tree, you find that the price book ratio (PBOOK) is also an important variable (Figure 8).

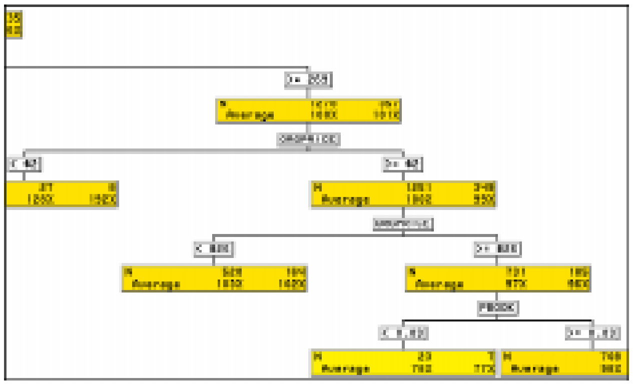


Figure 8 Right Branch of the Decision Tree

The results from the decision tree correlated nicely with the results from the variable selection tool. All of the variables that were chosen as being important by the Decision Tree were also chosen as being important by the Variable Selection tool. Based on this data mining exercise, we could say that the top four most important variables for predicting price differential are capitalization, original price, ratio of current assets to current liabilities, and price per assets per share.

So, for example, this decision tree allows you to make statements like the following:

If a stock has a capitalization of greater than $269 million, and its original price as of April 1, 1996 was greater than $26, and its price per assets per share was less than 0.03, then on average, the price differential for that stock will be about 78%.

This rule illustrates the type of stock that you want to avoid for this time period. The stocks that performed well, on the other hand, were any stocks with low capitalization (less than $269 million), low price (less than $7 per share), and a low current ratio. This scenario can be easily understood by recognizing that the period in question was a good period for the market as a whole. In such a situation, small, low-priced stocks tend to exhibit greater price movement than do stocks of larger, more established companies.

EXPLANATION OF RESULTS

This paper is not intended to show you how to determine what the next best “killer” investment strategy should be. We just examined one time period while any true analysis of the stock market should consider many different time periods and ascertain which factors were important based on different economic conditions, etc. Instead, our intent was to illustrate how to apply Enterprise Miner™ software to a data set beginning with the data mart and ending with the analysis of the data.

Appendix 1: Contents of the original daily pricing history data set and fundamentals data set

Daily Price History Data:

```
         -----Alphabetic List of Variables and Attributes-----
   #    Variable    Type    Len    Pos    Format          Informat
 -----------------------------------------------------------------
   3    DATETIME    Num       8      9    DATETIME16.     DATETIME16.
  11    DAY         Num       8     73    DATE7.          DATE7.
   7    HIGH        Num       8     41
   6    LAST        Num       8     33
   8    LOW         Num       8     49
   4    OPEN        Num       8     17
   5    OPENINT     Num       8     25
   2    SECTYPE     Char      1      8
   1    SYMBOL      Char      8      0
   9    VOLUME      Num       8     57
  10    YVOLUME     Num       8     65
```

Fundamentals Data:

```
         -----Alphabetic List of Variables and Attributes-----

                  #    Variable    Type    Len    Pos
                 ------------------------------------
                 32    ASSETS      Num       8    368
                 31    BETA        Num       8    360
                 34    BSDATE      Char      8    384
                 20    COMMEARN    Char     40    160
                 38    COMMFINC    Char     40    416
                 36    COMMNOUT    Num       8    400
                 26    COMPNAME    Char     64    264
                 18    CURNTEPS    Num       8    144
                 24    CUSIPNUM    Char      9    250
                  3    DAILYVOL    Num       8     16
                  9    DIVAMNT     Num       8     64
                 10    DIVRATE     Num       8     72
                 48    DIVREFYR    Num       8    528
                 49    DIVYEAR1    Num       8    536
                 50    DIVYEAR2    Num       8    544
                 51    DIVYEAR3    Num       8    552
                 52    DIVYEAR4    Num       8    560
                  8    DIVYIELD    Num       8     56
                 13    EARNPERD    Char     16     96
                 53    EPSREFYR    Num       8    568
                 54    EPSYEAR1    Num       8    576
                 55    EPSYEAR2    Num       8    584
                 56    EPSYEAR3    Num       8    592
                 57    EPSYEAR4    Num       8    600
                 58    EQUITYR1    Num       8    608
                 59    EQUITYR2    Num       8    616
                 60    EQUITYR3    Num       8    624
                 61    EQUITYR4    Num       8    632
                 23    FISCLEND    Char      2    248
                 22    GRWTH5YR    Num       8    240
```

```
-----Alphabetic List of Variables and Attributes-----

         #    Variable    Type    Len    Pos
         -----------------------------------

         4    HI52        Num       8     24
        39    HILOREF     Num       8    456
         6    HIYEAR      Num       8     40
        40    HIYEAR1     Num       8    464
        42    HIYEAR2     Num       8    480
        44    HIYEAR3     Num       8    496
        46    HIYEAR4     Num       8    512
        21    ICOMEARN    Char     40    200
        29    INSTITUT    Num       8    344
        14    LATSTEPS    Num       8    112
        33    LIABILTS    Num       8    376
         5    LO52        Num       8     32
         7    LOYEAR      Num       8     48
        41    LOYEAR1     Num       8    472
        43    LOYEAR2     Num       8    488
        45    LOYEAR3     Num       8    504
        47    LOYEAR4     Num       8    520
        16    LST12EPS    Num       8    128
        35    LTRMDEBT    Num       8    392
        66    NETINYR1    Num       8    672
        67    NETINYR2    Num       8    680
        68    NETINYR3    Num       8    688
        69    NETINYR4    Num       8    696
        19    NEXTEPS     Num       8    152
        28    OPTNSYMB    Char      8    336
        25    PAD         Char      5    259
        11    PAYDATE     Char      8     80
         2    PERATIO     Num       8      8
        37    PREFDOUT    Num       8    408
        15    PRIOREPS    Num       8    120
        17    REFYEAR     Num       8    136
        62    REVNUYR1    Num       8    640
        63    REVNUYR2    Num       8    648
        64    REVNUYR3    Num       8    656
        65    REVNUYR4    Num       8    664
        70    SPLTFAC1    Char     16    704
        71    SPLTFAC2    Char     16    720
        72    SPLTFAC3    Char     16    736
        27    SPRANK      Char      8    328
        30    SPSTARS     Num       8    352
         1    SYMBOL      Char      8      0
        12    XDIVDATE    Char      8     88
```

Appendix 2: Code to create merged data

```
data aprprice(keep=symbol orgprice);
   set sastray.trashis (where=(day = '03APR96'd));
   orgprice = last;
   run;
proc sort;
   by symbol;

data junprice(keep=symbol endprice);
   set sastray.trashis (where=(day = '20JUN96'd));;
   endprice = last;
   run;
proc sort;
   by symbol;


data sasuser.trafund(drop=paydate i commfinc);
   merge sastrai.trafund aprprice junprice;
   by symbol;
   if orgprice and endprice;
   divdate = input(paydate, mmddyy8.);
   i = index(commfinc, '$');
   if i then bookval = input(substr(commfinc, i), dollar8.);

   run;
```

Appendix 3: Creation of the final data mart

```
data sasuser.trafund3 (keep= symbol pricedif orgprice divyield
                       curpeyld nxtpeyld trailyld  grwth5yr sprankng
                       spstars beta pbook roe curratio debteq capital pricsale
                       profmarg futgrow eps1qrt eps1yr rev1yr rev3yr
                       institut);
   set sasuser.trafund;
   label
      symbol = 'Ticker Symbol'
      pricedif = '3-month price differential (ratio)'
      orgprice = 'Price as of April 1, 1996'
      divyield = '% dividend yield'
      curpeyld = 'last 6 mo + next 6 mo earnings/price'
      trailyld = 'last 12 mo earnings/price'
      nxtpeyld = 'next 12 mo earnings/price'
      eps1qrt = 'earnings growth last quarter'
      grwth5yr = 'annualized earnings growth last 5 years'
      sprankng = 'converted s&p div. and earnings ranking'
      spstars = 's&p stars 1-4'
      beta = 'volatility indicator'
      pbook = 'price/assets per share'
      roe = 'return on equity'
      curratio = 'ratio current assets to current liab.'
      debteq = 'long term debt / equity'
      capital = 'capitalization (shares out x price)'
      pricsale = 'price/revenues'
      profmarg = 'net profit margin'
      futgrow = '% growth in earnings next year'
      eps1yr = 'eps growth last year'
      rev1yr = 'revenue growth last year'
      rev3yr = 'annualized revenue growth last 3 years'
      institut = '% institutional ownership';

   format
      pricedif percent8.
      orgprice dollar8.
      divyield
      curpeyld
      trailyld
      nxtpeyld
      eps1qrt
      grwth5yr percent8.
      sprankng 3.1
      spstars 1.
      beta 4.2
      pbook 4.1
      roe percent8.
      curratio 3.1
      debteq percent8.
      capital comma12.
      pricsale 4.1
      profmarg percent8.
      futgrow percent8.
      eps1yr
      rev1yr
      rev3yr percent8.
      institut percent8.
      ;

   divyield = divyield / 100;
```

```
grwth5yr = grwth5yr / 100;
institut = institut / 100;

select (sprank);
   when ('A+') sprankng = 4.3;
   when ('A', 'AA') sprankng = 4;
   when ('A-') sprankng = 3.7;
   when ('B+','BB+') sprankng = 3.4;
   when ('B') sprankng = 3;
   when ('B-', 'BBB') sprankng = 2.7;
   when ('C', 'CCC') sprankng = 2;
   when ('D') sprankng = 1;
   when ('LIQ') sprankng = 0;
   otherwise sprankng = .;
   end;

pricedif = endprice / orgprice;
curpeyld  = curnteps / orgprice;
trailyld = lst12eps / orgprice;
nxtpeyld = nexteps / orgprice;
if bookval = . then bookval = equityr1;
pbook = bookval / orgprice;
roe = curnteps / bookval;
curratio = assets / liabilts;
debteq = ltrmdebt / (bookval * commnout);
capital = commnout * orgprice / 1000000;
revshr = revnuyr1 * 1000000 / commnout;
pricsale = orgprice / revshr;
profmarg = netinyr1 / revnuyr1;

eps1qrt = latsteps / prioreps - 1;
futgrow = nexteps / lst12eps - 1;
eps1yr = curnteps / epsyear1 - 1;
rev1yr = revnuyr1 / revnuyr2 - 1;
rev3yr = (revnuyr1 / revnuyr4) ** 0.33333333 -1;
run;
```