Supply Chain Management Case Study

Can you predict
product backorders ?

MyDataModels - Sophia Antipolis France
www.mydatamodels.com

Part backorders is a common supply chain problem. A backorder is a retailer's order for a part that is temporarily out of stock with the vendor.

This case study is about building a **predictive model** to identify parts at risk of backorder before the event occurs so to have time to react.

We used a dataset available on the **Kaggle** website. The initial dataset contained 1M parts and 22 variables for each part. Unlike most of the machine learning algorithms available today in the market, Databolics does not need big datasets to build accurate and compact predictive models. To prove it, we decided to run Databolics on **3 different datasets** : the 1M parts one, a subset of 100K parts and a subset of 30K parts. The models produced on these 3 different datasets were very much similar and equivalent in terms of accuracy/performance. Results below are those we got with the 100K parts dataset.

Dataset contained the historical data for 8 weeks prior to the week we were looking to predict. The data was taken as **weekly snapshots** at the start of each week. 22 columns/variables were defined as follows:

`sku` - Random ID for the product
`national_inv` - Current inventory level for the part
`lead_time` - Transit time for product (if available)
`in_transit_qty` - Amount of product in transit from source
`forecast_3_month` - Forecast sales for the next 3 months
`forecast_6_month` - Forecast sales for the next 6 months
`forecast_9_month` - Forecast sales for the next 9 months
`sales_1_month` - Sales quantity for the prior 1 month time period
`sales_3_month` - Sales quantity for the prior 3 month time period
`sales_6_month` - Sales quantity for the prior 6 month time period
`sales_9_month` - Sales quantity for the prior 9 month time period
`min_bank` - Minimum recommend amount to stock
`potential_issue` - Source issue for part identified
`pieces_past_due` - Parts overdue from source
`perf_6_month_avg` - Source performance for prior 6 month period
`perf_12_month_avg` - Source performance for prior 12 month period
`local_bo_qty` - Amount of stock orders overdue
`deck_risk` - Part risk flag
`oe_constraint` - Part risk flag
`ppap_risk` - Part risk flag
`stop_auto_buy` - Part risk flag
`rev_stop` - Part risk flag
`went_on_backorder` - Product actually went on backorder. This is the target value.

The dataset size was 8.3 Mbytes size and referenced 100,000 parts with 823 backorders. The dataset was highly unbalanced, the positive class (backorder) accounted for **0.823%** of all parts.

The dataset contained numerical input variables and text input variables. Lead time variable had some missing values, so we had to replace all of these by -1 to let Databolics interpret -1 has a missing value. No other pre-processing had to be made like outliers management as Databolics handles this automatically.

Feature 'Went_on_backorder' was the response variable and it had value “Yes” in case of backorder and “No” otherwise.

We used Databolics to produce a model which allowed to predict which part will be on backorder or not. Databolics automatically produced the best model – made actually of 2 mathematical equations - **without** any programming, any algorithm selection, any dataset splitting, etc…

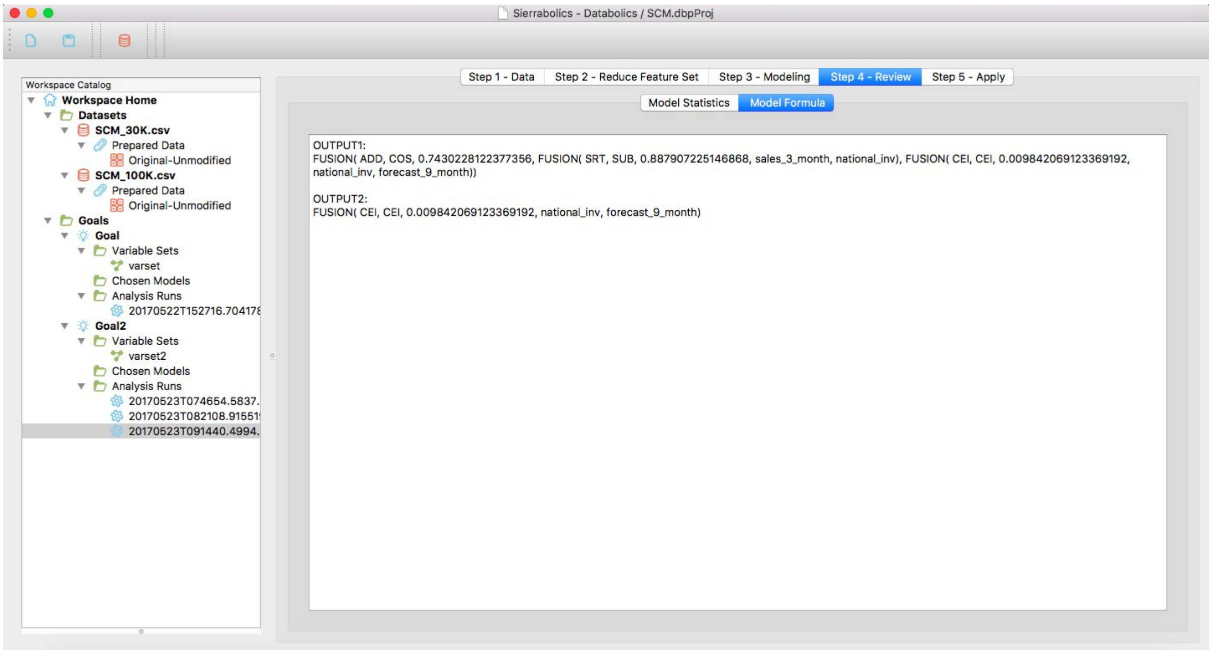
Preparation of the dataset file

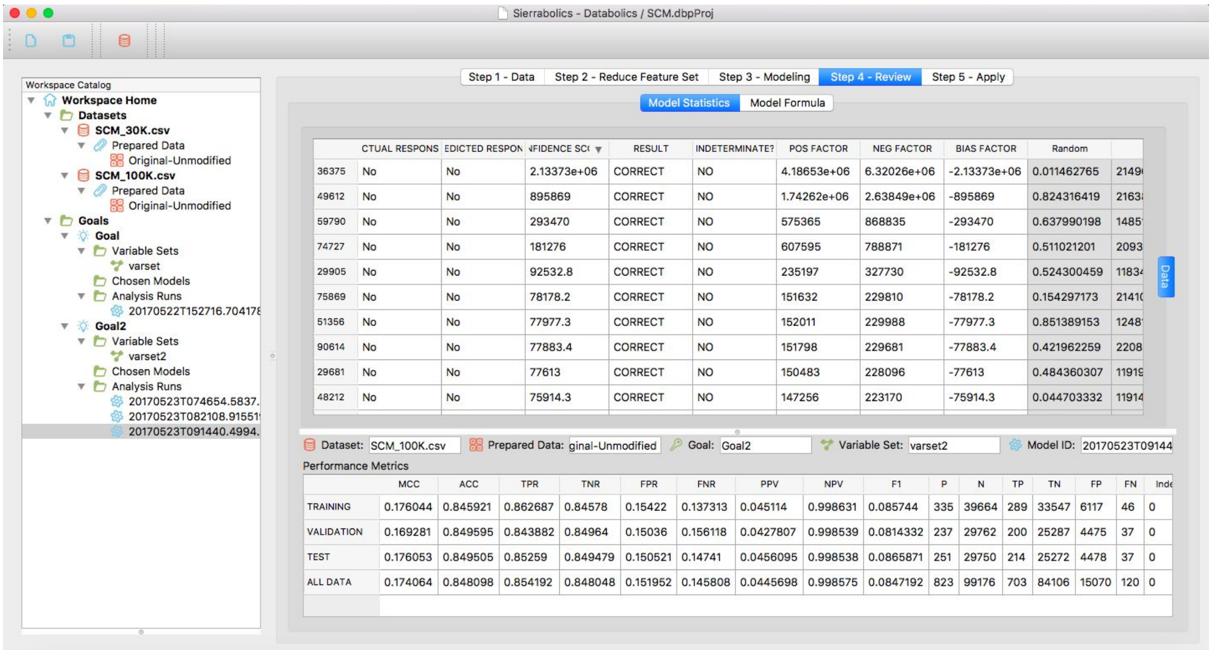
The response variable ‘Went_on_backorder’ (Yes or No) value was derived from the metadata. Rows were then randomized such that the order of samples in the rows was ensured to be random.

Results in less than 1 hour

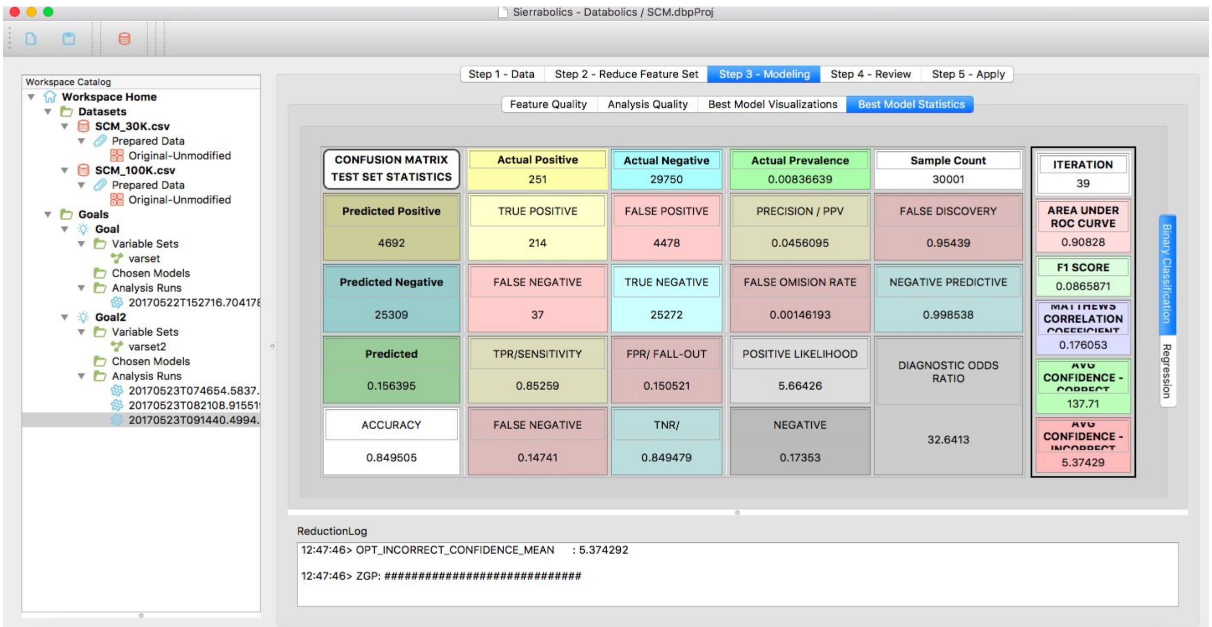
Generation of an explanatory model, took just under 1 hour on a simple MacBook Air notebook. When evaluated against an independent hold out set of 30,001 parts (actually the test subset derived from the 100,000 parts dataset, see **Fig 2** comments), the resulting explanatory model was 85% accurate with a True Negative Rate of 86%, and a True Positive Rate of 85%. True Negative Rate represents the percentage of parts predicted Negative (not on backorder in our case) and being actually Negative in the test subset. True Positive Rate represents the percentage of parts predicted Positive (on backorder in our case) and being actually Positive in the test subset.

Area under ROC curve was 0.91.




Sierrabolics - Databolics / SCM.dbpProj
Workspace Catalog
Workspace Home
Datasets
SCM_30K.csv
Prepared Data
Original-Unmodified
SCM_100K.csv
Goals
Goal
Variable Sets
varset
Chosen Models
Analysis Runs
Goal2
varset2
Step 1 - Data
Step 2 - Reduce Feature Set
Step 3 - Modeling
Step 4 - Review
Step 5 - Apply
Model Statistics
Model Formula
Dataset: SCM_100K.csv
Goal: Goal2
Variable Set: varset2
Performance Metrics
MCC ACC TPR TNR FPR FNR PPV NPV F1 P N TP TN FP FN
TRAINING 0.176044 0.845921 0.862687 0.84578 0.15422 0.137313 0.045114 0.998631 0.085744 335 39664 289 33547 6117 46 0
VALIDATION 0.169281 0.849595 0.843882 0.84964 0.15036 0.156118 0.0427807 0.998539 0.0814332 237 29762 200 25287 4475 37 0
TEST 0.176053 0.849505 0.85259 0.849479 0.150521 0.14741 0.0456095 0.998538 0.0865871 251 29750 214 25272 4478 37 0
ALL DATA 0.174064 0.848098 0.854192 0.848048 0.151952 0.145808 0.0445698 0.998575 0.0847192 823 99176 703 84106 15070 120 0

[illegible]


Sierrabolics - Databolics / SCM.dbpProj
Step 3 - Modeling
Feature Quality
Analysis Quality
Best Model Visualizations
Best Model Statistics
CONFUSION MATRIX
TEST SET STATISTICS
Actual Positive 251
Actual Negative 29750
Actual Prevalence 0.00836639
Sample Count 30001
ITERATION 39
Predicted Positive 4692
TRUE POSITIVE 214
FALSE POSITIVE 4478
PRECISION / PPV 0.0456095
FALSE DISCOVERY 0.95439
AREA UNDER ROC CURVE 0.90828
Predicted Negative 25309
FALSE NEGATIVE 37
TRUE NEGATIVE 25272
FALSE OMISION RATE 0.00146193
NEGATIVE PREDICTIVE 0.998538
F1 SCORE 0.0865871
CORRELATION 0.176053
Predicted 0.156395
TPR/SENSITIVITY 0.85259
FPR/ FALL-OUT 0.150521
POSITIVE LIKELIHOOD 5.66426
DIAGNOSTIC ODDS RATIO 32.6413
CONFIDENCE - 137.71
ACCURACY 0.849505
FALSE NEGATIVE 0.14741
TNR/ 0.849479
NEGATIVE 0.17353
CONFIDENCE - 5.37429
Binary Classification
Regression
ReductionLog
12:47:46> OPT_INCORRECT_CONFIDENCE_MEAN : 5.374292
12:47:46> ZGP: ##############################

[illegible]