Prediction or Guess? Decide by Looking at Two Images Generated by a “MATLAB MySQL” Algorithm

Carlos Rodríguez

University of Central Florida,
Institute for Simulation and Training,
3100 Technology Pkwy, Orlando, FL 32826
calirodriguez@knights.ucf.edu

Abstract. In the field of data mining, predictive modeling refers to the usage of a statistical model built on a training data set in order to make predictions about new prospects contained in the scoring data set. A model should not be used to predict when it encounters unseen data in the scoring set because such predictions would be a guess or a speculation. This paper proposes an algorithm that will produce two simple images and a “level of guessing” (LOG) pie chart. These images will tell the analyst whether or not it is appropriate to use a statistical predictive model to make predictions on a particular scoring set. The proposed algorithm will offer a solution to the scoring adequacy problem based on subsets of the original data. The algorithm will be implemented with a user interface built with MATLAB code, which acts on MySQL databases that contain the data.

Keywords: predictive modeling, data mining, scoring set, supervised learning, MATLAB, MATLAB GUI, MySQL.

1 Introduction

To my knowledge, there is no automated process or algorithm available that uses images to determine the adequacy of a statistical model to predict given new data. Even if a model shows very strong performance in the training data, this does not guarantee its applicability towards scoring new data since the scoring data might have values absent in the training data. The process presented here seeks to find differences between training and scoring data because such differences will impact the ability to make accurate predictions. The outcome of the process will send a message to the analyst highlighting these differences in the form of two simple images and a LOG pie chart.

It is often found in real world applications that a model built on a training data fails to produce accurate predictions in the scoring data. There are two reasons for that outcome:

i. The training set observations present different characteristics than those in the scoring set, hence a predictive model “speculates” when it classifies based on unseen data (for example, imagine that one of the important

S. Yamamoto (Ed.): HIMI 2014, Part I, LNCS 8521, pp. 87–97, 2014.

explanatory variables of the training set is OCCUPATION and its distinct values are A,B,C,D but in the scoring set OCCUPATION has the following values A,B,C,D,E,F,G,H,I,J; classifications based on F,G,H,I, J will be speculative and the analyst must know this caveat before making predictions in the scoring set).

ii. The training set observations look like those in the scoring set, but prospects with similar characteristics behave differently in the scoring set (for example, imagine that one of the important explanatory variables of the training set is OCCUPATION and its distinct values are A,B,C,D and in the scoring data OCCUPATION also has the values A,B,C,D but in the training set A and B responded to a marketing mailing campaign at a 30% rate, yet in the scoring set, once the outcome is known, A and B only respond to a mailing campaign at a 3% rate).

For situation (i) above, scoring should not be performed if the degree of "speculation" is considerable. The magnitude of "considerable" depends on the application or in the risk preference of the analyst given the implications of the decision. For situation (ii) scoring should occur because it is meant to predict a "similar" universe of prospects. There is no way around situation (ii) simply because patterns in old data did not hold in new data; given past information it is unrealistic to avoid this error.

Analysts cannot control the outcome of the target in the scoring set, but they can decide on the adequacy of a statistical predictive model to score new data. The proposed algorithm will produce two simple images plus a LOG pie chart that will inform the analyst about differences between the training set and the scoring set.

When predictions do not conform to the actual outcomes analysts are often not sure what caused the model to underperform. In order to reach an explanation, analysts may spend considerable amount of time trying to determine why their predictions were off. The images produced by this algorithm will answer that question quickly, indicating if the problem was different data between training and scoring universes (like situation (i)) or different individuals with similar data (like situation (ii)). Ideally the images produced by the proposed algorithm should be used before scoring in order to avoid speculation in the scoring process.

2 Proposed Algorithm for Determining Differences between Training Set and Scoring Set

The algorithm that follows will produce two images and a LOG pie chart that will provide information about the applicability of a statistical model M, built on a training set T, and used to make predictions on a scoring set S. The adequacy between training T and scoring S will be analyzed only for those variables found in T that are also found in S. In other words, if there is a missing explanatory variable in S, the algorithm will still find the adequacy of the remaining variables. Any model created in T will not work on S due to the missing variables required in S; any data modeling software will make you aware of this issue right away.

Let T be a n x $(k+1)$ array representing a training dataset for which a statistical model M has been built. T can be broken as a dependent variable array Y_{nx1} and an array of k final explanatory variables X_{nxk}. Array X_{nxk} can have numeric, date and class variables. M explains Y_{nx1} as a function of X_{nxk}. M will be used to predict $\hat{Y}_{rx1}$ given a new array of explanatory variables $\hat{X}_{rxk}$, where r is the number of records or rows in the scoring set. The algorithm treats class variables differently than it treats numeric and date variables.

2.1 Class Explanatory Variables

Let v be the number of class variables out of the k explanatory variables. X_{nxv} is a n x v non numerical array of class variables with each class explanatory variable represented by a nx1 non numerical array called x_{Tci}, i=1 … v. Each non numerical array x_{Tci} will have L_i number of levels or distinct values, so $L_i \le n$. Let τ_i={all levels of class variable i in the training set T}, i=1 … v, so the set τ_i has L_i elements.

If v class variables were used in the creation of the final statistical model in T, then the same v class variables must be present in the scoring set S represented by $\hat{X}_{rxk}$; otherwise the scoring process will fail. Then the set $\tilde{X}_{rxv}$ is the subset from the scoring set $\hat{X}_{rxk}$ that contains the class variables from $\hat{X}_{rxk}$ and k $\geq v$. Let x_{Sci} be a rx1 non numerical array, i=1 … v, so all the x_{Sci} arranged together next to each other make up array $\tilde{X}_{rxv}$. Each non numerical array x_{Sci} will have P_i number of levels or distinct values, so $P_i \le r$. Let ρ_i={all levels of class variable i in the scoring set S}, i=1 … v, so the set ρ_i has P_i elements.

For the scoring process to be completely valid the condition {$\rho_i \in \tau_i$ for all i, i=1 … v } must hold because otherwise the prediction will not be completely based on prior knowledge leading to guessing or speculation. This is condition 1.

In real world applications there might be small violations to condition 1. An analyst might consider that the violations are not significant and score using model M. By using the proposed algorithm, the decision of whether to score using model M will be well informed and based on calculated risks. For instance, the analyst might conclude that it is better to remove the unseen levels from the scoring set before scoring or add more records to the training data to account for the unseen levels from S.

If condition 1 does not hold, that is, if there are elements in ρ_i not found τ_i, which is the same as saying that there are elements in the scoring set S not found in the training set T, then it is important to quantify the magnitude of the violation of condition 1. Such magnitude will help the analyst decide if the differences between training and scoring set are significant or if they are negligible for practical purposes.

For example, if we go back to the first situation why a predictive model performs poorly, recall that in the example it was argued that classifications based on F,G,H,I and J would be speculative, because levels F,G,H,I and J were not found in the training data (i.e. {F,G,H,I,J} $\in \rho_i$ and {F,G,H,I,J} *NOT* $\in \tau_i$). Let q_i be the number of rows out of the r rows of class array x_{Sci} for which condition 1 is violated, i=1…v. In other words, q_i represents the number of rows from the scoring set S with a level not found in the training set T for class variable i, i=1 … v. Then we can define

$$\boldsymbol{\alpha}= \frac{\sum_{i=1}^{v} q_i}{vr} \tag{1}$$

0≤ $\boldsymbol{\alpha}$ ≤1. $\boldsymbol{\alpha}$ is the overall percentage of violation of condition 1 for all class explanatory variables. The number α provides information on the severity of violation of condition 1. Large values of α indicate that there is considerable difference between training set T and scoring set S. The larger α, the more evidence against using a statistical model built on training set T to predict data found in scoring set S.

2.2 Numeric Explanatory Variables and Date Explanatory Variables Converted to Numbers

Let d be the number of numerical variables out of the k explanatory variables (date variables can be converted to numbers using the YYYYMMDD format so they are considered here). X_{nxd} is a n x d matrix with each variable represented by a nx1 vector called $x_{T\eta i}$, i=1…d. X_{nxd} is a matrix composed by the numerical variables from array X_{nxk} and arranged as vectors $x_{T\eta i}$. Each $x_{T\eta i}$ vector will have a maximum and minimum value labeled MINT_i and MAXT_i. Let θ_i={MINT_i, MAXT_i}, i=1…d, so each set θ_i has 2 elements.

Each numeric variable from the scoring array $\hat{X}_{rxk}$ is represented by a rx1 vector called $x_{S\eta i}$ i=1…d. Each vector $x_{S\eta i}$ will have a maximum and minimum value labeled MINS_i and MAXS_i. Let λ_i={MINS_i, MAXS_i}, i=1…d, so each set λ_i has 2 elements.

For the scoring process to be valid the condition { $\text{MINT}_i \le \text{MINS}_i \le \text{MAXT}_i$ and $\text{MINT}_i \le \text{MAXS}_i \le \text{MAXT}_i$ } must hold because otherwise the predictions will be based on extrapolation at least one time. Equivalently $\text{MINS}_i \in [\text{MINT}_i,\text{MAXT}_i]$ and $\text{MAXS}_i \in [\text{MINT}_i,\text{MAXT}_i]$, i=1…$d$. This is condition 2.

Let b_i be the number of rows out of the r rows of vector $x_{S\eta i}$ for which a value of is not in the interval [MINT_i,MAXT_i], i=1…d. In other words b_i represents the number of rows where extrapolation will be needed. Then we can define

$$\boldsymbol{\beta}= \frac{\sum_{i=1}^{d} b_i}{dr} \tag{2}$$

0≤$\boldsymbol{\beta}$≤1. $\boldsymbol{\beta}$ is the percentage of violation of condition 2 for all numerical (and date converted to numerical) explanatory variables. Large values of β indicate that there is considerable difference between training set T and scoring set S. The larger β, the more evidence against using the statistical model built on training set T to predict data from the scoring set S.

Initially, the proposed algorithm will indicate the violation of either condition in the form of two images. The images will be a reflection of the capacity of the model to predict new data. Before defining the rationale for the generation of the images, at this point there is already enough information on α and β to reach an answer to the main question: should I use this model? If α and β are close to 0 then the model built on T is adequate to make predictions in the scoring set S. Large values of α or β is a signal that making predictions using model M involves a considerable degree of speculation or guessing.

3 Generation of the First Two Images to Determine the Adequacy of a Statistical Model

In this section I will detail how to produce the first two images that will determine the adequacy of a statistical model M to make predictions on a scoring set S. The first image will cover the information detailed by α and β. This first image is called **αβ Binary Spectrum** and represents the first visual representation of adequacy. If this image indicates adequacy, then no further visual inspections are needed and scoring should be applied to the scoring set S. If the αβ Binary Spectrum indicates a problem coming from α or from β or both, then a second image can be used as reference to find out where the problems are. The second image is called **Detailed Metadata Chart (DMC)**. DMC needs to be generated because the information provided by the αβ Binary Spectrum is not enough to indicate the percentage of data found in S that was not found in T. αβ Binary Spectrum indicates that there is a problem but does not tell where the problem is. The DMC shows the variables with problems and the magnitude of the problem. After reviewing the DMC the analyst can make a more informed decision on whether to score with model M or modify training set T in order to make it more consistent with the values found in scoring set S. The LOG pie chart is an overall summary of the findings of the process and it will be discussed later.

3.1 αβ Binary Spectrum

It is a simple image of the vector $\phi = \begin{bmatrix} K\alpha \\ K\beta \end{bmatrix}$ where the entries of the vector represent the colors of the image and K is a constant that determines sensitivity to unseen data (higher K means more sensitive to unseen data; here K=150). Digital images can be created directly from a matrix, as images are numerical arrays. The farther ϕ_{11} and ϕ_{21} are from 0, the less adequate the statistical model will be. Because ϕ has two entries, the αβ binary spectrum will have two bands. The left band of the αβ binary spectrum is the class variables' adequacy and the right band is the numerical variables adequacy. Our imaging technique must define what color 0 takes and the farther from that color, the adequacy worsens. In this write up, 0 is represented by color black. Fig.1 shows the visual scale to determine adequacy for α and β, where leftmost is better and rightmost is worst in terms of adequacy:

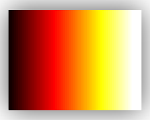


Fig. 1. Visual scale for the αβ binary spectrum

The ideal αβ binary spectrum is shown in Fig.2 and it indicates that α=0 and β=0

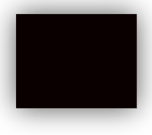


Fig. 2. Ideal αβ binary spectrum representing full scoring adequacy

Fig.3 shows other examples of αβ binary spectrums

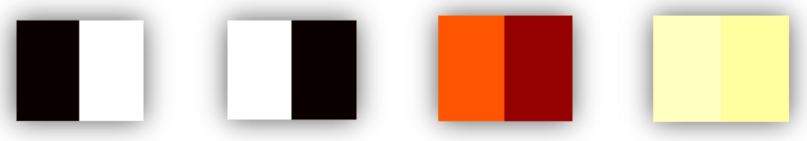


Fig. 3. Different examples of αβ binary spectrum

From left to right in Fig.3, the first αβ binary spectrum indicates a perfect adequacy in the class explanatory variables (black) and full non adequacy in the numerical variables (white). This means that numerical explanatory variables in the scoring set S are outside the range found in the training set T; if predictions are made in the scoring set S, there will be a high degree of extrapolation or speculation. In the second αβ binary spectrum from Fig.3 the numerical variables are adequate to make predictions but the class variables are not adequate. The first two αβ binary spectrums shown in Fig.3 are the extremes as they involve black and white and making a decision under these conditions is not difficult. If the αβ binary spectrum looked more like the two rightmost αβ binary spectrum from Fig.3, then the decision would not be as straight forward. In order to help the analyst make a decision, the proposed algorithm will produce a second image called Detailed Metadata Chart (DMC).

3.2 Detailed Metadata Chart (DMC)

The detailed metadata chart is a bar chart that indicates the percentage of records in the scoring set S with levels not found in the training set T for each explanatory variable. Depending on whether the variable i is numerical or class, the height of the bar that represents variable i in the DMC will be $\frac{b_i}{r}$ or $\frac{q_i}{r}$ respectively. Only positive values of $\frac{b_i}{r}$ and $\frac{q_i}{r}$ will be in the chart.

DMC helps identify where the adequacy problem is coming from. Based on the information of DMC the analyst should be able to make a better decision on how to score the new data or even a decision of not scoring at all. A hypothetical example of a DMC is shown in Fig.4.

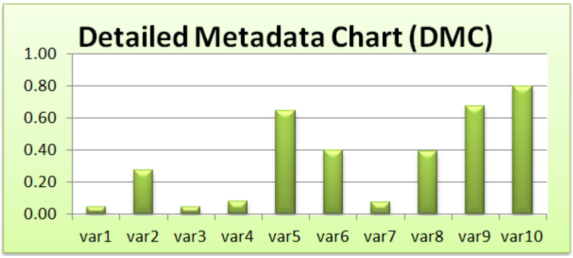


Fig. 4. Detailed Metadata Chart (DMC)

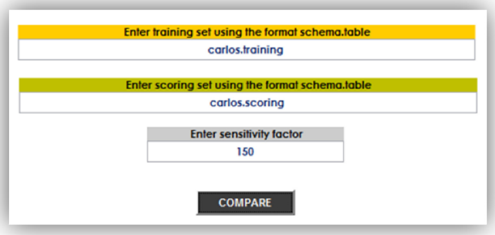
The way to read the sample DMC presented in Fig.4 is the following (for 2 of the 10 variables):

- 40% of the records in the scoring set S had a *level* for **var6** not found in the training set T.
- 80% of the records in the scoring set S had a *value* for **var10** outside the range analyzed in the training set T.

… where var6 is a class variable and var10 is numeric. The sample DMC shown above indicates that hypothetical explanatory variables var2, var5, var6, var9 and var10 would cause serious problems if used to make predictions on a scoring set S using a model M created on a training set T. DMC will only display variables with problems, so variables with all levels contained in the training data will not be displayed.

4 Illustration of the Algorithm with an Application

In order to illustrate the algorithm described in this paper I have applied it to real world training and scoring data sets. In order to perform this illustration I have used MySQL and code programmed using MATLAB, including the user interface. MATLAB connects to MySQL using the MATLAB Database Toolbox. The training and scoring data are kept in MySQL and the algorithm can be run against them with the MATLAB code. The MATLAB user interface is shown in Fig.5 and requires the analyst to input the name of the training and scoring sets (carlos.training and carlos.scoring in this example) plus the level of sensitivity to differences between the two (150). The user then presses COMPARE in order to start the process:



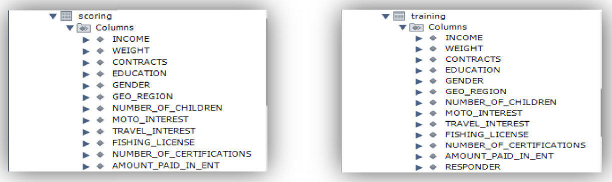
Enter training set using the format schema.table
carlos.training

Enter scoring set using the format schema.table
carlos.scoring

Enter sensitivity factor
150

Fig. 5. User interface that requires three inputs

The variables from the scoring set S and training data T used in this example are given in Fig.6.



scoring
Columns
INCOME
WEIGHT
CONTRACTS
EDUCATION
GENDER
GEO_REGION
NUMBER_OF_CHILDREN
MOTO_INTEREST
TRAVEL_INTEREST
FISHING_LICENSE
NUMBER_OF_CERTIFICATIONS
AMOUNT_PAID_IN_ENT

training
Columns
INCOME
WEIGHT
CONTRACTS
EDUCATION
GENDER
GEO_REGION
NUMBER_OF_CHILDREN
MOTO_INTEREST
TRAVEL_INTEREST
FISHING_LICENSE
NUMBER_OF_CERTIFICATIONS
AMOUNT_PAID_IN_ENT
RESPONDER

Fig. 6. Variables from sample training and scoring sets

As can be seen, the training data T contains the known target variable labeled as RESPONDER. The scoring set S does not have RESPONDER as we will try to predict it using a model M. There are 6 numerical explanatory variables and 6 class explanatory variables (so d=6 and v=6). A statistical model M was used to fit the data using all the "important" explanatory variables detailed in the training set T. M can be a regression, neural network, decision tree, etc.; the selection of the optimal modeling approach is a common functionality of data mining software such as SAS Enterprise Miner or SPSS Modeler based on statistical coefficients (i.e. AIC, BSC, ROC Area, etc.). But no matter what modeling approach is selected as the optimal, it will not work well if there are unseen values and/or levels in the scoring set S. In other words, how will a model classify or predict on something it has not seen?

The proposed algorithm will analyze all variables in the scoring set present in the training set (i.e. the intersection). After running the MATLAB code the resulting 2 images are shown in Fig.7.

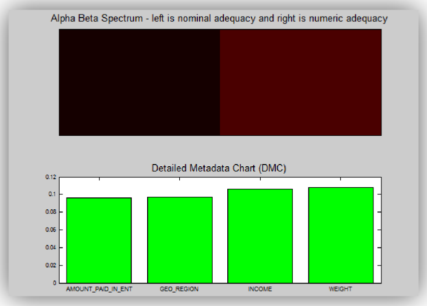


Fig. 7. αβ binary spectrum and DMC from MATLAB output

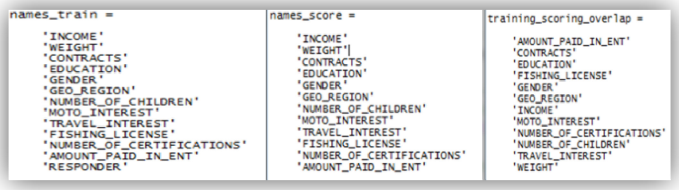
From the images generated by the algorithm we can see that the αβ binary spectrum suggests inadequacy in the numerical explanatory variables because the right side of the αβ binary spectrum departs considerably from color black (reddish appearance). The nominal side (left side) of the αβ binary spectrum is not completely black (brownish), so there are some inadequacies in the class variables as well. Because there is inadequacy we can revise the details in the DMC. The DMC confirms the information from the αβ binary spectrum and gives us the details of the inadequacy. We can see that extrapolation will be needed in the fields AMOUNT_PAID_ IN_ENT, INCOME and WEIGHT. Also, there are levels of GEO_REGION found in the scoring set S that were not found in the training set T. On each of the inadequate variables, the level of inadequacy is about 11%; another way to explain this is:

- For GEO_REGION, 11% of the records found in set S had levels not found in set T.
- For WEIGHT, 11% of the records found in set S had values outside the range used in set T.
- For INCOME, 11% of the records found in set S had values outside the range used in set T.
- For AMOUNT_PAID_IN_ENT, 10% of the records found in set S had values outside the range used in set T.

The decision about what do after this information is up to the analyst. Options include removing unseen data from the scoring set or add more data into the training set that includes the current missing levels and numeric values.

5 Sample MATLAB Output and MATLAB Code

The only inputs required by the algorithm are the training and scoring sets loaded into MySQL and defined in the user interface with the desired sensitivity. The algorithm "loops" thru each variable performing all the necessary computations. Variable identification from training and scoring sets plus the overlap between the two as shown in Fig.8.



```
names_train =

    'INCOME'
    'WEIGHT'
    'CONTRACTS'
    'EDUCATION'
    'GENDER'
    'GEO_REGION'
    'NUMBER_OF_CHILDREN'
    'MOTO_INTEREST'
    'TRAVEL_INTEREST'
    'FISHING_LICENSE'
    'NUMBER_OF_CERTIFICATIONS'
    'AMOUNT_PAID_IN_ENT'
    'RESPONDER'
```

```
names_score =

    'INCOME'
    'WEIGHT'
    'CONTRACTS'
    'EDUCATION'
    'GENDER'
    'GEO_REGION'
    'NUMBER_OF_CHILDREN'
    'MOTO_INTEREST'
    'TRAVEL_INTEREST'
    'FISHING_LICENSE'
    'NUMBER_OF_CERTIFICATIONS'
    'AMOUNT_PAID_IN_ENT'
```

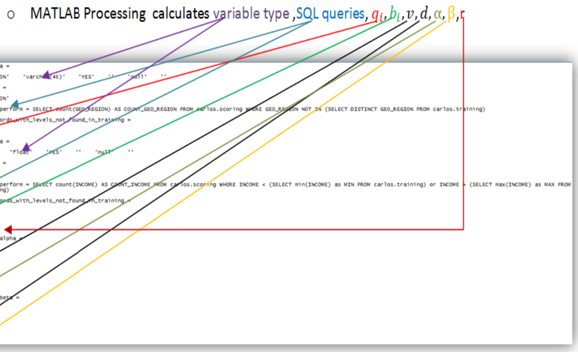
```
training_scoring_overlap =

    'AMOUNT_PAID_IN_ENT'
    'CONTRACTS'
    'EDUCATION'
    'FISHING_LICENSE'
    'GENDER'
    'GEO_REGION'
    'INCOME'
    'MOTO_INTEREST'
    'NUMBER_OF_CERTIFICATIONS'
    'NUMBER_OF_CHILDREN'
    'TRAVEL_INTEREST'
    'WEIGHT'
```

Fig. 8. MATLAB Output of the variables found in training and scoring

Fig.9 shows other key results of the MATLAB processing, including SQL queries.

The algorithm loops identifying variable type and processing them accordingly. For processing speed, it is better to keep in the training set only those variables used in the model M before running the code.

In addition to αβ binary spectrum and DMC, the algorithm also produces the LOG pie chart shown in Fig.10. This is a pie chart with the proportion of records out of the scoring set S with at least one variable having a value or level not found in the training data.


o MATLAB Processing calculates variable type ,SQL queries, q_i, b_i, v, d, α, β, r

```
Indicator_data =
    'GEO_REGION'    'varchar(45)'    'YES'    ''    'null'    ''
current_field =
    'GEO_REGION'
SQL_query_to_perform = SELECT count(GEO_REGION) AS COUNT_GEO_REGION FROM carlos.scoring WHERE GEO_REGION NOT IN (SELECT DISTINCT GEO_REGION FROM carlos.training)
Number_of_records_with_levels_not_found_in_training =
    [387]
Indicator_data =
    'INCOME'    'float'    'YES'    ''    'null'    ''
current_field =
    'INCOME'
SQL_query_to_perform = SELECT count(INCOME) AS COUNT_INCOME FROM carlos.scoring WHERE INCOME < (SELECT min(INCOME) as MIN FROM carlos.training) or INCOME > (SELECT max(INCOME) as MAX FROM carlos.training)
Number_of_records_with_levels_not_found_in_training =
    [424]
r =
        4000
numerator_of_alpha =
   387
v =
     6
alpha =
    0.0161
numerator_of_beta =
        1241
d =
     6
beta =
    0.0517
```

Fig. 9. MATLAB calculation of different coefficients and SQL queries

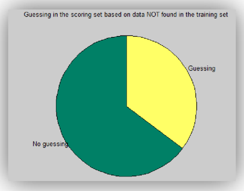


Fig. 10. LOG Pie Chart

6 Proposed Solution to the Scoring Adequacy Problem

The algorithm then computes a new column called ADEQUACY_IND in the scoring set that indicates how many values are not in the training set for each record of the scoring table. Finally, the algorithm solves the scoring adequacy problem by creating the following new data sets in MySQL:

i. A new scoring set with those records for which there will be no guessing in the original scoring set. This is called scoring set "OK" because it contains the rows from the original data that were completely found in the training data. In other words, "OK" contains records where ADEQUACY_IND=0.

ii. A new training set that will be used to produce a new model to score the problematic records. This training set is called training set "ALT" because it is an alternative training set where new models will be built. "ALT" does not contain variables found in the DMC.

iii. A new scoring set with the problematic records, which can be scored with the model built on training "ALT". This scoring set is called scoring set "ALT" because it will be scored, without any guessing, using models built in training set "ALT". Adequacy between scoring set "ALT" and training set "ALT" is perfect by construction. Scoring "ALT" contains records from original scoring set S where ADEQUACY_IND>0.

7 Conclusion

This paper presents a algorithm to determine, by using two images and a "level of guessing" (LOG) pie chart, the adequacy of a statistical predictive model M created on a training set T to predict new data from a scoring set S. Often in practice we observe that predictive models do not perform as expected and this can happen because of two reasons: the first one, we train on certain characteristics in T and the individuals to be predicted in S have different set of characteristics; the second one, individuals with same characteristics behaved differently in T than they do in S. In the second reason we just have to accept the errors as previous patterns in the data did not hold true in new data. The proposed algorithm addresses situations where data in T does not contain values and levels found in S. Making predictions when scoring data S has values not found in training data T is a guess or a speculation because model M has never seen such values, hence does not really know how to classify or predict based on them.

The first image is the αβ binary spectrum and it presents two rectangles, the left one represents the inadequacy coming from class data and the right one represents the inadequacy coming from numerical and date data. The more these rectangles approach the color black, the better, meaning that levels found in the scoring set S were also found in the training set T. The further away these rectangles are from black, the worst the adequacy of models built on T to score S. The second image is called Detailed Metadata Chart (DMC) and it shows the percentage of inadequacy for each explanatory variable. DMC summarizes the percentage of records with levels or values found in the scoring set S that were not found in the training set T. If both rectangles of the αβ binary spectrum are black, there is no need to focus on DMC but if any of the rectangles depart from color black, the DMC will detail where the inadequacy is coming from. Finally, the LOG pie chart summarizes the findings.

The user interface and sample MATLAB output were provided. The only inputs required by the algorithm are the training and scoring sets plus the sensitivity factor. The algorithm offers a solution to the scoring adequacy problem by breaking the original scoring set S in two subsets, "OK" and "ALT", where "OK" will be scored with the original models built in the original training set T and "ALT" will be scored with new models built in new training set "ALT", that contains only those variables that guarantee full scoring adequacy in the problematic records.

References

1. Kutner, M., Nachtsheim, C., Neter, J.: Applied linear statistical models. Irwin, Chicago (2004)
2. Kuhn, M., Johnson, K.: Applied predictive modeling. Springer, New York (2013)
3. Attaway, S.: MATLAB: A practical introduction to programming and problem solving. Butterworth-Heinemann, Waltham (2011)
4. MySQL, A., MySQL Administrator's Guide and Language Reference. MySQL, Indianapolis (2006)
5. Coulson, L.: MATLAB Programming (e-book). Global Media, Chandni Chowk (2009), Available from: eBook Collection (EBSCOhost), Ipswich, MA (accessed October 29, 2013)
6. Cerrito, P.: SAS I. Introduction to Data Mining Using SAS Enterprise Miner (e-book). SAS Institute, Cary (2006); Available from: eBook Collection (EBSCOhost), Ipswich, MA (accessed October 30, 2013)