RME-EP and Deep Learning

Advanced modeling methods are described in this paper. Examples will be used to develop a model based on the *Deep Learning* technique. Deep learning refers to *multi-level stacked neural networks*. It can enhance predictive modeling. In this paper, a two-level neural network model is described. The lower level consists of multiple neural networks. The upper level also consists of multiple *integration neural networks* that combine outputs of the lower level networks.

Multiple networks for the lower level can be developed based on various criteria. First, different network configurations produce multiple networks. For example, one network may have no internal hidden layers. This type of networks is very similar to *regression* models. Another network may use a single hidden layer with "auto" option in CMSR data miner. Third or fourth network may use smaller or larger number of hidden layer nodes.

Decision tree may be included. It can work as a segmentation model. Develop a decision tree without "Trim leaf nodes" of the "Options" in the decision tree dialog window. Prune tree nodes which have very low sample ratios manually. 0.02 (=2%) or 0.01 (=1%) for "Minimum support:" is recommended. Binary decision trees are recommended. For binary trees, select either "best and others" or "into two groups" for the "Branching" option of the "Options" tab.

Another approach for multiple networks is to decompose the problem into smaller networks, each performing simpler task with a smaller number of independent variables. For example for insurance, one model can be developed for accident claims. Another model is for theft claims. A third model is for breakdown claims. And so on. It is noted that large neural networks with large number of input and hidden nodes can learn very finely detailed information. This property is no good for predictive modeling as it will lead to *overfitting*. To avoid overfitting, simpler neural networks with a fewer number of relevant input and hidden nodes can be used.

In addition, CMSR neural networks provide two versions: running and best versions. This also provides multiple models.

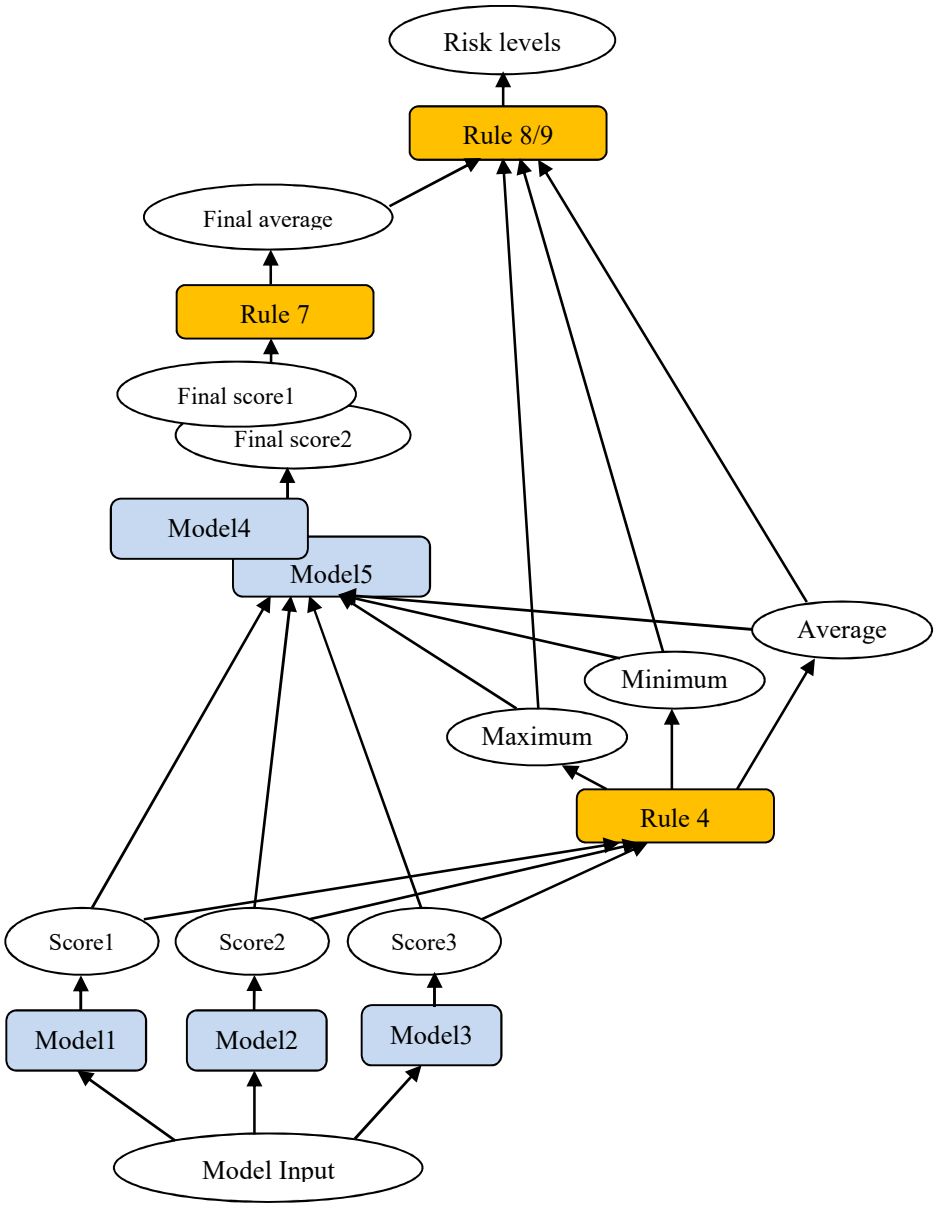
The upper level integration models can be developed using different hidden node sizes. For example, one network can use "auto" hidden layer configuration. Another network is with one node smaller hidden layer. Or simply use the best and the running versions of a single model.

Three or more networks are recommended for the lower level. Two or more networks are recommended for the upper level.

In the examples of this paper, we will use "**RISKCLASS**" and "**RISKFLAG**" database table columns. "RISKCLASS" values include "Risky" and "Safe". If training data were delinquent (or default or insurance claims), the value will be "Risky". Otherwise it is "Safe". Based on this, "RISKFLAG" value will be 0 if "RISKCLASS" is "Safe". 1 if "RISKCLASS" is "Risky". "RISKFLAG" will be the target variable for neural models.

(Note that in "Modeling Guide for Neural Network", we used "Risk" instead of "Risky". If you use "Risk" as the value in your data, interpret "Risky" as "Risk" in this document. In addition, notice that field names "RISKCLASS" and "RISKFLAG" are in upper case letters.)

The following is a schematic diagram that assumes three neural networks at the lower level ("Model1", "Model2" and "Model3"), and two integration neural networks at the upper level ("Model4" and "Model5"). Both levels can be extended with more neural networks, if required.



Input data is fed to three neural network models: "Model1", "Model2" and "Model3". Three models produce three values "Score1", "Score2" and "Score3", respectively. These scores are then passed to "Rule 4" to produce "Maximum", "Minimum" and "Average" score values. Then "Model4" and "Model5" take three model score values along with maximum, minimum and average values as input, and produces the "Final score1" and "Final score2" respectively. "Rule 7" then computes the average of two final scores.

"Rule 8" and "Rule 9" use the average final score as well as maximum, minimum and average scores to produce final risk level classifications: "Very high risk", "High risk", "Medium risk", "Low risk", etc.

Notice that to train "Model4" and "Model5" neural networks, a special intermediary training data is needed. This can be obtained using a temporary RME-EP model that computes three model values plus maximum, minimum values, and saving them into the database table using "Apply to Database". "Example 1" described in the next section can be used for this.

Example 1: Bagging Model

As noted in the previous page, “Model4” and “Model5” require special training data containing the following information;

- Model1 score
- Model2 score
- Model3 score
- Maximum score
- Minimum score
- Average score

To obtain this training data, create new columns in the database table of your training data. Name them as the followings. Note that all fields are of REAL data type;

- **“Model1Score”**
- “**Model2Score**”
- “**Model3Score**”
- “**MaximumScore**”
- “**MinimumScore**
- “**AverageScore**”

In addition, create the following columns as they will be needed for the examples described at the next sections;

- “**FinalScore1**”
- “**FinalScore2**”
- “**FinalAverage**”
- “**RiskLevel**”
- “**RiskLevel1**

Note that “**RiskLevel**” and “**RiskLevel1**” are character STRING type. The other columns are REAL data type.

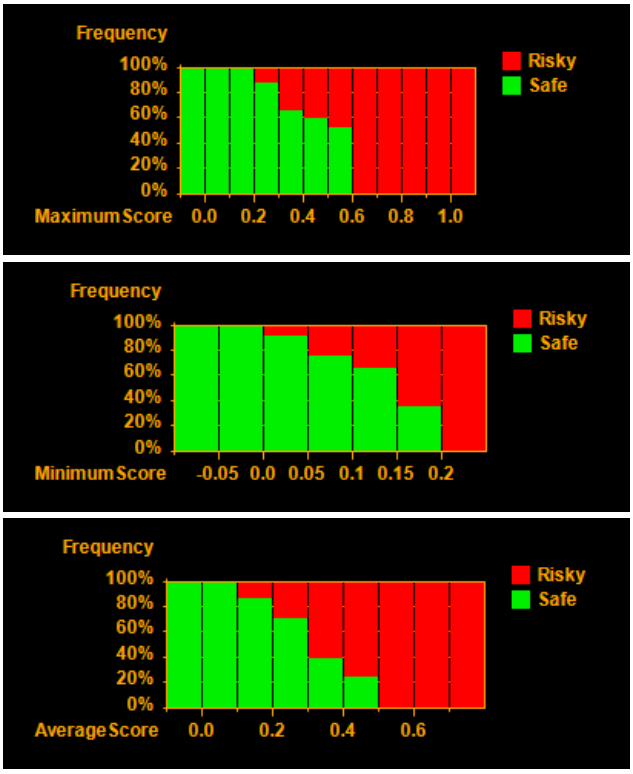
The following RME-EP model evaluates three neural network models and computes maximum, minimum and average scores. In the beginning, input and output variables are defined. Then “Rule 1”, “Rule 2” and “Rule 3” evaluate “Model1”, “Model2” and “Model3” respectively. Then “Rule 4” computes maximum, minimum and average score values.

```
// define input data fields and values in appearing order;
DECLARE Gender AS STRING INPUT VALUES IN GENDER OF RISKFLAG1;
DECLARE Race AS STRING INPUT VALUES IN RACE OF RISKFLAG1;
DECLARE AGEGROUP AS STRING INPUT VALUES IN AGEGROUP OF RISKFLAG1;
DECLARE Salary AS INTEGER INPUT;

// define output fields in appearing order;
DECLARE "Model1 score", "Model2 score", "Model3 score" AS REAL OUTPUT;
DECLARE "Maximum score", "Minimum score", "Average score" AS REAL OUTPUT;

RULE 1: // compute model 1 prediction;
IF TRUE THEN
                    SET "Model1 score" AS PREDICT(Model1) USING(
                                GENDER AS Gender,
                                RACE AS Race,
                                AGEGROUP AS  AGEGROUP,
                                SALARY AS Salary
                    )
END;
```

```
RULE 2: // compute model 2 prediction;
IF TRUE THEN
                    SET "Model2 score" AS PREDICT(Model2) USING(
                              GENDER AS Gender,
                              RACE AS Race,
                              AGEGROUP AS  AGEGROUP,
                              SALARY AS Salary
                    )
END;

RULE 3: // compute model 3 prediction;
IF TRUE THEN
                    SET "Model3 score" AS PREDICT(Model3) USING(
                              GENDER AS Gender,
                              RACE AS Race,
                              AGEGROUP AS  AGEGROUP,
                              SALARY AS Salary
                    )
END;

// compute max/min/avg;
RULE 4:  // evaluated after local rules evaluated
IF TRUE THEN
{
                    SET "Maximum score" AS MAX("Model1 score", "Model2 score", "Model3 score");
                    SET "Minimum score" AS MIN("Model1 score", "Model2 score", "Model3 score");
                    SET "Average score" AS AVG("Model1 score", "Model2 score", "Model3 score");
}
END;
```

Example 2: Risk Classification Model

In this example, a classification scheme based on maximum, minimum and average score values will be introduced to Example 1. To this end, update "MaximumScore", "MinimumScore" and "AverageScore" of your training-data database table using values of "Maximum score", "Minimum score" and "Average score" of Example 1 model (using "Apply to database"). Then, plot histogram charts for each of "MaximumScore", "MinimumScore" and "AverageScore". Use "RISKCLASS" as category. Then you will see the following charts;



The first figure shows customers scoring maximum value 0.6 or over have 100% risk rate. The second figure indicates customers scoring minimum value 0.2 or over have 100% risk rate. The third figure shows that customers scoring average value 0.5 or over have 100% risk rate. From these observations, we can code such that if customers score Maximum value >= 0.6 OR Minimum value >= 0.2 OR Average value >= 0.5 are labeled as "Very high risk".

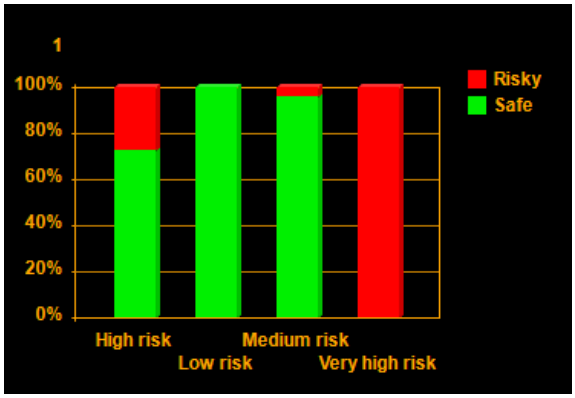
The second group of customers scoring Maximum value >= 0.3 OR Minimum value >= 0.05 OR Average value >= 0.2 is labeled as "High risk". The third group of customers scoring Maximum value >= 0.2 OR Minimum value >= 0.0 OR Average value >= 0.1 are labeled as "Medium risk". The rest are labeled as "Low risk".

The following is a coding example for this. The lines marked as red are the addition to previous Example 1 to include this classification scheme. First, the variable "Risk level1" is defined. Then "Rule 9" is added for this classification.

```
// define input data fields and values in appearing order;
DECLARE Gender AS STRING INPUT VALUES IN GENDER OF RISKFLAG1;
DECLARE Race AS STRING INPUT VALUES IN RACE OF RISKFLAG1;
DECLARE AGEGROUP AS STRING INPUT VALUES IN AGEGROUP OF RISKFLAG1;
DECLARE Salary AS INTEGER INPUT;

// define output fields in appearing order;
DECLARE "Model1 score", "Model2 score", "Model3 score" AS REAL OUTPUT;
DECLARE "Maximum score", "Minimum score", "Average score" AS REAL OUTPUT;
DECLARE "Risk level1" AS STRING OUTPUT; // (NEW)
```

```
RULE 1: // compute model 1 prediction;
IF TRUE THEN
                    SET "Model1 score" AS PREDICT(Model1) USING(
                              GENDER AS Gender,
                              RACE AS Race,
                              AGEGROUP AS  AGEGROUP,
                              SALARY AS Salary
                    )
END;

RULE 2: // compute model 2 prediction;
IF TRUE THEN
                    SET "Model2 score" AS PREDICT(Model2) USING(
                              GENDER AS Gender,
                              RACE AS Race,
                              AGEGROUP AS  AGEGROUP,
                              SALARY AS Salary
                    )
END;

RULE 3: // compute model 3 prediction;
IF TRUE THEN
                    SET "Model3 score" AS PREDICT(Model3) USING(
                              GENDER AS Gender,
                              RACE AS Race,
                              AGEGROUP AS  AGEGROUP,
                              SALARY AS Salary
                    )
END;

// compute max/min/avg;
RULE 4:  // evaluated after local rules evaluated
IF TRUE THEN
{
                    SET "Maximum score" AS MAX("Model1 score", "Model2 score", "Model3 score");
                    SET "Minimum score" AS MIN("Model1 score", "Model2 score", "Model3 score");
                    SET "Average score" AS AVG("Model1 score", "Model2 score", "Model3 score");
}
END;

RULE  9:   // classify risk levels;  // (NEW)
CASE
WHEN "Maximum score" >= 0.6 OR "Minimum score" >= 0.2 OR "Average score" >= 0.5 THEN
                    SET "Risk level1" AS 'Very high risk'
WHEN "Maximum score" >= 0.3 OR "Minimum score" >= 0.05 OR "Average score" >= 0.2 THEN
                    SET "Risk level1" AS 'High risk'
WHEN "Maximum score" >= 0.2 OR "Minimum score" >= 0.0 OR "Average score" >= 0.1 THEN
                    SET "Risk level1" AS 'Medium risk'
ELSE
                    SET "Risk level1" AS 'Low risk'
END;
```

It is important to test whether classifying customers into “Very high risk”, “High risk”, “Medium risk” and “Low risk” is valid. To verify this, apply this RME-EP model to your training-data database table. Update “Risk level1” labels to “RiskLevel1” of your training-data database table. Produce a CMSR bar chart as the following figure shows;



This chart shows what you expect from this RME-EP model! Customers classified by the RME-EP model as “Very high risk” are all was risk. “High risk” also has substantial risk involved. “Medium risk” has some risk. Low risk has no risk.

Note that this chart was produced as follows. From the bar chart dialog window, select “RiskLevel1” for “Item:” and “RISKCLASS” for “Category:”. Then select “1” for “Values(s):”. Then from the bar chart, select “Categoric proportion” for “Percent” and “Stacked/Cumulative” for “Stacking”.

Example 3: Deep Learning Model

As noted, “Model4” and “Model5” requires special training data. To obtain training data, update your training-data database table with the following information. Use the Example 1 model to update the following table columns. Then import the updated data into CMSR Data Miner. (Note that you may have done this already with Example 2 in the previous section. You can skip this.)

- “Model1Score”
- “Model2Score”
- “Model3Score”
- “MaximumScore”
- “MinimumScore”
- “AverageScore”

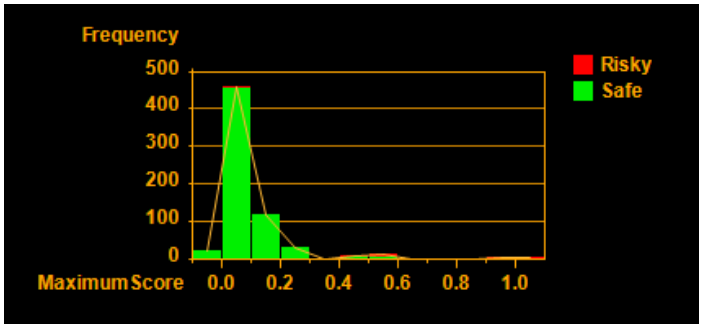
Create the integration models “Model4” and “Model5”. Train them and test fully.

Now you are ready to create a new RME-EP model that uses “Model4” and “Model5”. Import “Model1”, “Model2”, “Model3”, “Model4” and “Model5” into your new RME-EP model. Then use the following codes. Note that this is derived from Example 1. Red lines are added. Output variables “Final score1”, “Final score2” and “Final average” are added. “Rule 5” evaluates “Model4” and “Rule 6” evaluates “Model5”. “Rule 7” computes the “Final average”.

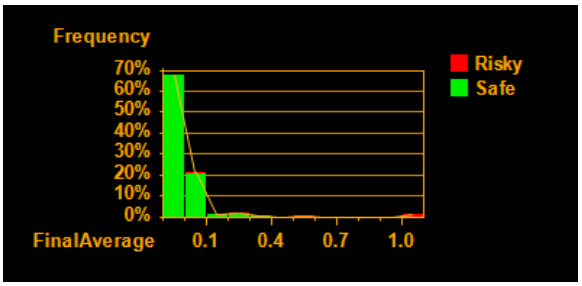
```
// define input data fields and values in appearing order;
DECLARE Gender AS STRING INPUT VALUES IN GENDER OF RISKFLAG1;
DECLARE Race AS STRING INPUT VALUES IN RACE OF RISKFLAG1;
DECLARE AGEGROUP AS STRING INPUT VALUES IN AGEGROUP OF RISKFLAG1;
DECLARE Salary AS INTEGER INPUT;

// define output fields in appearing order;
DECLARE "Model1 score", "Model2 score", "Model3 score" AS REAL OUTPUT;
DECLARE "Maximum score", "Minimum score", "Average score" AS REAL OUTPUT;
DECLARE "Final score1", "Final score2", "Final average" AS REAL OUTPUT;

RULE 1: // compute model 1 prediction;
IF TRUE THEN
                        SET "Model1 score" AS PREDICT(Model1) USING(
                                GENDER AS Gender,
                                RACE AS Race,
                                AGEGROUP AS  AGEGROUP,
                                SALARY AS Salary
                        )
END;

RULE 2: // compute model 2 prediction;
IF TRUE THEN
                        SET "Model2 score" AS PREDICT(Model2) USING(
                                GENDER AS Gender,
                                RACE AS Race,
                                AGEGROUP AS  AGEGROUP,
                                SALARY AS Salary
                        )
END;
```

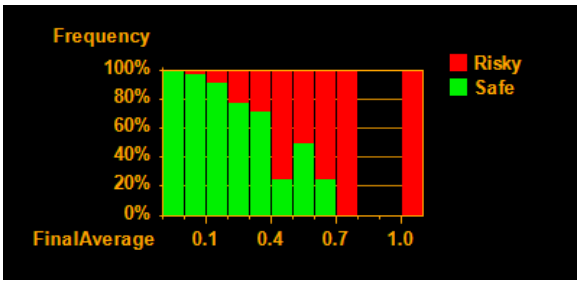
```
RULE 3: // compute model 3 prediction;
IF TRUE THEN
                    SET "Model3 score" AS PREDICT(Model3) USING(
                              GENDER AS Gender,
                              RACE AS Race,
                              AGEGROUP AS  AGEGROUP,
                              SALARY AS Salary
                    )
END;

// compute max/min/avg;
RULE 4:  // evaluated after local rules evaluated
IF TRUE THEN
{
                    SET "Maximum score" AS MAX("Model1 score", "Model2 score", "Model3 score");
                    SET "Minimum score" AS MIN("Model1 score", "Model2 score", "Model3 score");
                    SET "Average score" AS AVG("Model1 score", "Model2 score", "Model3 score");
}
END;

RULE 5: // compute final score1;
IF TRUE THEN
                    SET "Final score1" AS PREDICT(Model4) USING(
                              Model1Score AS "Model1 score",
                              Model2Score AS "Model2 score",
                              Model3Score AS "Model3 score",
                              MaximumScore AS "Maximum score",
                              MinimumScore  AS  "Minimum score",
                              AverageScore AS  "Average score"
                    )
END;

RULE 6: // compute final score2;
IF TRUE THEN
                    SET "Final score2" AS PREDICT(Model5) USING(
                              Model1Score AS "Model1 score",
                              Model2Score AS "Model2 score",
                              Model3Score AS "Model3 score",
                              MaximumScore AS "Maximum score",
                              MinimumScore  AS  "Minimum score",
                              AverageScore AS  "Average score"
                    )
END;

/ compute the final average;
RULE 7:
IF TRUE THEN
                    SET "Final average score" AS AVG("Final score1", "Final score2");
END;
```

The “MaximumScore” has following score distribution. “MinimumScore” and “AverageScore” also have similar distribution patterns.



When this RME-EP model is applied to your training-data database table updating “FinalAverage”, the following histograms show “FinalAverage” score distribution patterns.





The left chart shows improvement over maximum (, minimum and average) score distribution shown in the previous example. Samples are now more concentrated below 0.0 and over 1.0. So the classification rule “Rule 6” can be modified as follows. If final average is 0.7 or over, then classify as “Very high risk”. If final average is less than 0.0, classify as “Low risk. If final average is less than 0.1, classify as “Medium risk. Otherwise the rules of the previous example based on maximum, minimum and average will apply. In the following code, new added part is colored in red.

```
RULE  8:   // classify risk levels with Final average;
CASE
WHEN "Final score" >= 0.7 THEN
                        SET "Risk level" AS 'Very high risk'
WHEN "Final score" < 0.0 THEN
                        SET "Risk level" AS 'Low risk'
WHEN "Final score" < 0.1 THEN
                        SET "Risk level" AS 'Medium risk'
WHEN "Maximum score" >= 0.6 OR "Minimum score" >= 0.2 OR "Average score" >= 0.5 THEN
                        SET "Risk level1" AS 'Very high risk'
WHEN "Maximum score" >= 0.3 OR "Minimum score" >= 0.05 OR "Average score" >= 0.2 THEN
                        SET "Risk level" AS 'High risk'
WHEN "Maximum score" >= 0.2 OR "Minimum score" >= 0.0 OR "Average score" >= 0.1 THEN
                        SET "Risk level" AS 'Medium risk'
ELSE
                        SET "Risk level" AS 'Low risk'
END;
```

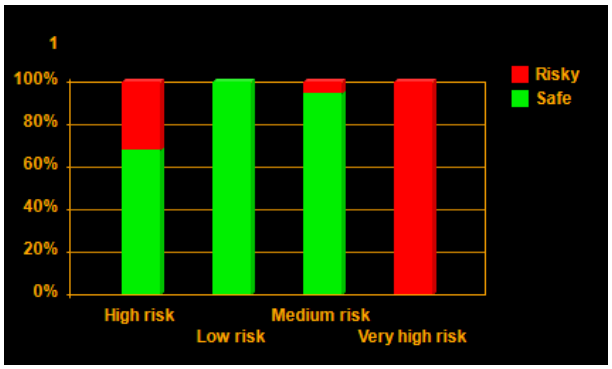
The final updated version of this example is as follows. Note that this version includes the classification rule (Rule 9) introduced in Example 2.

```
// define input data fields and values in appearing order;
DECLARE Gender AS STRING INPUT VALUES IN GENDER OF RISKFLAG1;
DECLARE Race AS STRING INPUT VALUES IN RACE OF RISKFLAG1;
DECLARE AGEGROUP AS STRING INPUT VALUES IN AGEGROUP OF RISKFLAG1;
DECLARE Salary AS INTEGER INPUT;

// define output fields in appearing order;
DECLARE "Model1 score", "Model2 score", "Model3 score" AS REAL OUTPUT;
DECLARE "Maximum score", "Minimum score", "Average score" AS REAL OUTPUT;
DECLARE "Final score" AS REAL OUTPUT;
DECLARE "Risk level", "Risk level1" AS STRING OUTPUT;

RULE 1: // compute model 1 prediction;
IF TRUE THEN
                SET "Model1 score" AS PREDICT(Model1) USING(
                        GENDER AS Gender,
                        RACE AS Race,
                        AGEGROUP AS  AGEGROUP,
                        SALARY AS Salary
                )
END;

RULE 2: // compute model 2 prediction;
IF TRUE THEN
                SET "Model2 score" AS PREDICT(Model2) USING(
                        GENDER AS Gender,
                        RACE AS Race,
                        AGEGROUP AS  AGEGROUP,
                        SALARY AS Salary
                )
END;

RULE 3: // compute model 3 prediction;
IF TRUE THEN
                SET "Model3 score" AS PREDICT(Model3) USING(
                        GENDER AS Gender,
                        RACE AS Race,
                        AGEGROUP AS  AGEGROUP,
                        SALARY AS Salary
                )
END;

// compute max/min/avg;
RULE 4:  // evaluated after local rules evaluated
IF TRUE THEN
{
                SET "Maximum score" AS MAX("Model1 score", "Model2 score", "Model3 score");
                SET "Minimum score" AS MIN("Model1 score", "Model2 score", "Model3 score");
                SET "Average score" AS AVG("Model1 score", "Model2 score", "Model3 score");
}
END;
```

```
RULE 5: // compute final score;
IF TRUE THEN
                    SET "Final score" AS PREDICT(Model4) USING(
                              Model1Score AS "Model1 score",
                              Model2Score AS "Model2 score",
                              Model3Score AS "Model3 score",
                              MaximumScore AS "Maximum score",
                              MinimumScore  AS  "Minimum score",
                              AverageScore AS  "Average score"
                    )
END;

RULE 6: // compute final score2;
IF TRUE THEN
                    SET "Final score2" AS PREDICT(Model5) USING(
                              Model1Score AS "Model1 score",
                              Model2Score AS "Model2 score",
                              Model3Score AS "Model3 score",
                              MaximumScore AS "Maximum score",
                              MinimumScore  AS  "Minimum score",
                              AverageScore AS  "Average score"
                    )
END;

// compute the final average;
RULE 7:
IF TRUE THEN
                    SET "Final average score" AS AVG("Final score1", "Final score2");
END;

RULE  8:   // classify risk levels with final average';
CASE
WHEN "Final score" >= 0.7 THEN
                    SET "Risk level" AS 'Very high risk'
WHEN "Final score" < 0.0 THEN
                    SET "Risk level" AS 'Low risk'
WHEN "Final score" < 0.1 THEN
                    SET "Risk level" AS 'Medium risk'
WHEN "Maximum score" >= 0.6 OR "Minimum score" >= 0.2 OR "Average score" >= 0.5 THEN
                    SET "Risk level1" AS 'Very high risk'
WHEN "Maximum score" >= 0.3 OR "Minimum score" >= 0.05 OR "Average score" >= 0.2 THEN
                    SET "Risk level" AS 'High risk'
WHEN "Maximum score" >= 0.2 OR "Minimum score" >= 0.0 OR "Average score" >= 0.1 THEN
                    SET "Risk level" AS 'Medium risk'
ELSE
                    SET "Risk level" AS 'Low risk'
END;
```

```
RULE  9:   // classify risk levels;
CASE
WHEN "Maximum score" >= 0.6 OR "Minimum score" >= 0.2 OR "Average score" >= 0.5 THEN
                        SET "Risk level1" AS 'Very high risk'
WHEN "Maximum score" >= 0.3 OR "Minimum score" >= 0.05 OR "Average score" >= 0.2 THEN
                        SET "Risk level1" AS 'High risk'
WHEN "Maximum score" >= 0.2 OR "Minimum score" >= 0.0 OR "Average score" >= 0.1 THEN
                        SET "Risk level1" AS 'Medium risk'
ELSE
                        SET "Risk level1" AS 'Low risk'
END;
```

Finally, apply the classification variable “Risk level” to the “RiskLevel” column of the training-data database table. And produce the following bar chart;



There isn’t much difference at chart level, compared to Example 2. But a close examination of data numbers shows some improvement over the maximum, minimum and average method. (Note that you can see actual numbers from the “Data” tab of bar charts.)

Example 4: Deep Learning Model with Decision Tree

The Example 3 Deep Learning model can be extended to include a decision tree model. To make decision tree compatible with integration neural network models, a decision tree is developed as a *segmentation probability model*. For this purpose, develop a decision tree without "Trim leaf nodes" of the "Options" tab in the decision tree dialog window. Prune tree nodes which have very low sample ratios. 0.02 (=2%) or 0.01 (=1%) for "Minimum support:" is recommended. A binary decision tree is recommended. For a binary decision tree, select either "best and others" or "into two groups" for the "Branching" option of the "Options" tab.

An evaluation rule for decision tree can be coded as in the following codes. A variable "Model4 ratio" is declared. The evaluation rule will be contain PREDICT(Model4, 'Risky') to compute probability. Note that "Model" in this example is the decision tree. This will compute probability of 'Risky' portions.

```
DECLARE "Model4 ratio" AS REAL OUTPUT;
RULE r2d: // compute decision tree risk probability;
IF TRUE THEN
                    SET "Model4 ratio" AS PREDICT(Model4, 'Risky') USING(
                                ….  AS …,
                                ….  AS …,
                                ….  AS …
                    )
END;
```

The model "Deep Learning (ALL)" of the "Demo" project in CMSR is an implementation of this extended model. From Example 3, above decision tree evaluation rule "r2d" is added. In addition, two new rules for the new integration models which include output from the decision tree model are added (Rule "r4_2a" and "r4_2b"). Final average is computed and new risk classification rule (Rule "r6b") is added. Finally, the most popular vote from three risk classifications is determined at Rule "r7".