Affect Detection Models for Educational Data

LEARNING ANALYTICS REPORT

Building and comparing twenty models across four learner affective states, and choosing the metric that matters in a real school.

Yana Rulinsky

Instructional Designer · Online Learning in Higher Education

yanarulinsky.com

By Yana Rulinsky

In this final analysis, I will aim to develop the best possible model detector for each affect label (frustration, confusion, boredom, and concentration) and to arrive at the best possible model that I feel confident can be utilized in practice and real-world educational settings. As transparency and authenticity in my data modeling approaches are of paramount importance to me, I will list and explain my process steps during this assignment, my model choice and parameter settings, and the performance evaluation metrics I utilized to assess my models' performance. As part of this assignment, I will also include a table for each student affect listing performance metrics for each model explored. Lastly, I will interpret all my data modeling results as to my personal confidence level in these models being utilized to detect each of the four student affect states in real-world settings.

Task 0: Data Inspection and Statistical Examination

Step 1: Loading, Reading, and Examination of CSV Data

My analysis began with uploading of the CSV data into Altair AI (RapidMiner) via an "Import Wizard" function inside the Read CSV Operator and examining the raw dataset. First, I identified the non-numeric and numeric data columns (or attributes) and matched the attribute abbreviations to the column descriptions in the student_affect_dataset_column_descriptions.csv file to ensure my complete understanding of the data contained in the full dataset file student_affect_dataset.csv

Step 2: Filtering of CSV Examples

Next, I utilized the Filter Examples Operator to the Read CSV Operator and used it to filter out any invalid or missing data in the dataset.

Step 3: Statistical Examination of CSV Data

Following the filtering of the examples, I located and connected the Statistics Operator to the existing process and executed the current process.

While reviewing the output results of the Statistics Operator, I observed that the "Missing Data" for the four student affects (boredom, concentration, frustration, and confusion) demonstrated that over ninety percent of data for each affect was missing. This large gap in the dataset should be expected due to the short 20-second student observation intervals in which a research member directly observed each student.

In addition to reviewing the missing data in all distributions, I reviewed and considered the minimum, average, and maximum values for each attribute. I noted down the attributes with unusual minimum, average, or maximum values or values not consistent with my expectations, as well as extraordinarily high standard deviation values, which could indicate a presence of data

outliers and an overall large spread of the data. I also observed and noted a high variance in some of the features in my dataset.

Step 4: Correlation Matrix Operator

Next, I connected the Correlation Matrix to my main process and observed the correlation among the different features in my dataset. I noticed that several features with avg_, max_, sum_, and min_ exhibit a high level of correlation with each other indicating a possible feature multicollinearity. (Please see Figure 1-8 in the Appendix for dataset statistics.)

After observing and examining my dataset statistics, I proceeded to set up my models for each of the student affects of frustration, concentration, boredom, and confusion.

Task #1: Model Choices, Setup, and Parameter Setting

Step1: Affect Files, Select Attributes, and Filter Examples

I first created a different and separate RapidMiner File for each of the four affects and added operators utilized in the past three assignments, such as Read CSV and Filtering Examples operator, to the main process in each of the affect files. I then followed a similar procedure to that of previous assignments to set up five distinct models for each of the four affects.

Step 2: Set Role Operator

I introduced a Set Role Operator into each affect RapidMiner file main process to indicate to RapidMiner which columns are to be used use as predictors. I set the outcome label of each affect to “label” to indicate that this is the column that I want to predict and set the role of Fold feature to “batch” to indicate that the Fold column is to be used to split the data into batches.

Step 3: Numerical to Binomial Data Conversion

Following this, I introduced a Numerical to Binomial Operator to tell RapidMiner that my outcome variable is binomial or that exists in two different states. I checked the “include special attributes” checkbox to include special attributes and selected a chosen affect attribute to designate it as a binomial variable.

Step 4: Multiply Operator

After this, I introduced the Multiply Operator into my process to extend the above actions to multiple Cross Validation Operators.

Step 5: Model Selection Process and Cross Validation Operators

I chose five data models to predict each of my affects which are: Majority Class Model, Logistic Regression Model, Naïve Bayes Model, Decision Tree model, and Deep Learning model. My rationale for choosing these specific models is as follows:

First, these models are inherently built for classification tasks and can handle the binary data of each affect outcome, i.e., whether the student is frustrated (true) or not frustrated (false) or is bored or not bored. Specifically, as a linear regression model is built to predict continuous data and must be thresholded to create a binary outcome, a logistic regression model is very well suited to binary classification and outcome prediction. Additionally, logistic regression can be optimized by class weighting and threshold tuning so that it pays more attention to the minority class. As I previously observed a high imbalance between classes in my dataset during last assignment, this is important to keep in mind.

Although the Majority Class model is included as a baseline model, the Logistical Regression, Naïve Bayes, and a Decision Tree model are all designed to output probabilities or class assignments and are therefore well-suited for the purpose of this assignment. The Naïve Bayes model provides probability estimates of each class, helps professionals understand the certainty behind each prediction, and is said to perform well on imbalanced datasets according to various online sources. A Decision Tree Model does not assume any relationship between features and outcome variable and therefore can identify and capture non-linear relationships that may exist. Also, a decision tree model can handle both continuous and categorical data and can be modified via class weighting or cost-sensitive learning to give more importance to the minority class and therefore can handle an imbalance in my data. Thus, the Decision Tree is included in my set of models. Each of the three models above are tailored for predicting categorical outcomes and classification tasks, which makes them all well-suited for this assignment.

Lastly, I included a Deep Learning Model, a model that is inspired by the function and structure of a human brain and uses artificial neural networks to perform complex computations on large datasets and enable machines to learn from examples and make autonomous decisions. As my affect dataset is quite large, the deep learning model is well-equipped to be utilized in this assignment.

After selecting my models, I introduced into my process four Cross-Validation Operators that will help me look for patterns inside the dataset and will help me understand how well my model learns from my data. I then renamed each operator as per the model's name I am utilizing and checked "split on batch attributes" for each of the five operators to split my data according to the Fold Attribute column and not the default fold parameters specified in each Cross Validation Operator.

Step 6: Majority Class Model Setup and Discussion

The first model I utilized is the Majority Class Model that learns about the majority class in my data and displays class composition. This baseline model helps me understand the class composition of each student affect.

Inside the first Cross-Validation Operator, I introduced the Default Model into the Training Panel and added the Apply Model and Binomial Classification Performance Operators to the Testing

Panel. I selected “Mode” in parameters panel to help me determine the most frequent model encounters in my student dataset.

Step 7: Logistical Regression Model Setup and Discussion

The second model I added to my process was the Logistic Regression Model as per the reasons explained above.

Inside the Cross-Validation Operator, I added Logistic Regression Operator in the Training Panel and added the Apply Model and Binomial Classification Performance Operators to the Testing Panel. Not having studied the parameters setting for a Logistical Regression Model, I left the default model parameters intact.

Step 8: Naïve Bayes Model Setup and Discussion

The third model I added to my process was the Naïve Bayes Model as per the reasons explained above.

Inside the Cross-Validation Operator, I added Naïve Bayes to the Training Panel and added the Apply Model and Binomial Classification Performance Operators to the Testing Panel. Not having any parameters to be set inside the Naïve Bayes model, I left the default parameter intact.

Step 9: Decision Tree Model Setup and Discussion

The fourth model I added to my process was the Decision Tree Model as per the reasons explained above.

Inside the Cross-Validation Operator, I added Decision Tree to the Training Panel and added the Apply Model and Binomial Classification Performance Operators to the Testing Panel. As for the Decision Tree Model parameters, I decided to run the model with default parameters intact for my initial run and then adjust parameters such as minimum leaf size after evaluating model performance and performance metrics. I made a mental note to myself that increasing minimum leaf size helps protect against overfitting.

Step 10: Deep Learning Model Setup and Discussion

Inside the Cross-Validation Operator, I added Deep Learning Operator in the Training Panel and added Filtering Examples and Numerical to Binomial Operators to both the Training and Testing Panel. To set the Deep Learning Model parameters I followed the course guidance on parameter setting and chose TahnNwithDropout for activation, the hidden dropout ratios to 0.5, and the number of epochs = 50 to increase the number of training passes through the dataset. I also checked the “early stopping” box. (Please see Figure 9-13 in Appendix for visualization of models for each student affect.)

Task #2: Performance Metrics Choices and Metrics Definitions

Being already aware of some imbalance in my dataset prior to executing my Majority Class Model due to the previous assignment, I reminded myself that performance metrics can be very sensitive to an imbalance in data and can tell conflicting stories about the model based on metrics used, therefore, I decided to use multiple six metrics of Accuracy, Cohen's Kappa, AUC, Precision, Recall, and F1 measurement. As per the course lectures and the accompanying metrics table, five of these metrics are commonly used to evaluate categorical data. The Kappe and AUC metrics I believe are essential to evaluating models as they indicate above chance performance and level of differentiation between classes. Other metrics are also helpful in evaluating model predictiveness. The Accuracy metric will only be used for the Majority Class Model as it will only display the most frequent label in my dataset and will predict that label for all samples in my dataset. Accuracy metric will be deselected for the next four models and the other five performance metrics will be selected instead.

Performance Evaluation Metrics Definitions

Accuracy – accuracy metric will display the most frequent label in a dataset and will predict that label for all samples in the dataset. I will use the Accuracy metric only for my Majority Class Model to determine the imbalance between the majority and minority classes in my dataset.

Cohen's Kappa – Cohen's kappa measures the agreement between my model's predictions and actual classification, while adjusting for the agreement that could occur by chance. A kappa value of close to 1 shows a strong agreement and an above chance performance. A value of 0 shows that model's performance is no better than guessing. A negative value indicates that the model's agreement with actual data is worse than would be expected by chance and the model is not predictive.

AUC (Area Under the Curve) – AUC represent model's ability to distinguish between classes. An AUC of 1 indicates perfect discrimination between negative and positive classes and an AUC of 0.5 indicates no discriminative power between classes equal to random guessing. AUC above 0.5 indicates marginal ability and an increased ability of a model to differentiate between classes. This value could indicate whether the model has practical application or not.

Precision – precision value indicates the percentage of predicted values that are correct. For example, for an affect of frustration, this value will indicate that, when the model predicts that the student is frustrated, how many of these predictions are actually correct.

Recall – recall value indicates the percentage of actual label values that the model identifies as correct. For example, for an affect of frustration this would mean the percentage of frustrated students that model correctly identifies as being frustrated and the frustrated students that the model misses. As per the course guidance, recall is a good measurement of rarely occurring labels.

F1 Score – F1 is a performance metric for binary classification models that illustrates a balance between the values of precision and recall. F1 score computes the average of precision and

recall, where the relative contribution of both metrics is equal to F1 score. The best value of F1 score is 1 and the worst is 0.

Task #3: Original Affect Models' Performance Tables and Discussion

Below are the performance evaluation table metrics for each "original" model (i.e., model with no additional data or feature engineering applied) per student affect.

The best performer among the four data models was chosen based on the overall values of Cohen's Kappa and AUC, as the Kappa value indicates the model's above chance performance and AUC indicates how well the model can distinguish between classes.

The hard part in choosing the "best performing" model for this assignment rests on the fact that the applicability of a specific data model in a real-world educational setting does not necessarily correspond to the model's performance evaluation metrics such as Kappa and AUC values. The higher values of Kappa or AUC of a model do not necessarily indicate or imply that a given model can be confidently used to detect a specific affect in real-world educational settings but instead speak to the ability of the model to perform above a random guess and to distinguish between classes. The best performing model based on its Kappa and AUC values that was selected for additional data and feature engineering is highlighted in green in the tables below.

Model Name	Accuracy	Cohen's Kappa	AUC	Precision	Recall	F1
Major Class	96.25%	0	0.5	- -	0%	- -
Logistic Regression	**- -**	**0.097**	**0.667**	**11.52%**	**16%**	**13.04%**
Naïve Bayes	- -	0.069	0.640	8.65%	28%	11.87%
Decision Tree	- -	0.012	0.511	20%	0.83%	1.64%
Deep Learning		**0.096**	**0.706**	**11.62%**	**30.17%**	**13.80%**

Table 1. Frustration Affect Detection Model Performance Results

Model Name	Accuracy	Cohen's Kappa	AUC	Precision	Recall	F1
Major Class	82.24%	0	0.5	- -	0%	- -
Logistic Regression	- -	0.147	0.748	83.87%	97.17%	90.01%
Naïve Bayes	**- -**	**0.215**	**0.713**	**86.12%**	**86.04%**	**85.96%**
Decision Tree	- -	0.030	0.582	82.51%	99.47%	90.17%
Deep Learning		0.042	0.743	82.67%	98.63%	89.92%

Table 2. Concentration Affect Detection Model Performance Results

Model Name	Accuracy	Cohen's Kappa	AUC	Precision	Recall	F1
Major Class	89.36%	0	0.5	- -	0%	- -
Logistic Regression	**- -**	**0.258**	**0.748**	**30.82%**	**40.93%**	**34.63%**
Naïve Bayes	- -	0.083	0.710	14.64%	83.78%	24.73%
Decision Tree	- -	0.049	0.562	32.79%	3.68%	6.80%
Deep Learning		0.204	0.734	25.16%	40.36%	30.69%

Table 3. Boredom Affect Detection Model Performance Results

Model Name	Accuracy	Cohen's Kappa	AUC	Precision	Recall	F1
Major Class	96.01%	0	0.5	- -	0%	- -
Logistic Regression	**- -**	**0.066**	**0.680**	**8.38%**	**19.26%**	**11.43%**
Naïve Bayes	- -	0.028	0.617	5.53%	53.40%	9.51%
Decision Tree	- -	0.013	0.615	16.67%	0.83%	1.55%
Deep Learning		0.061	0.676	8.68%	22.14%	10.97%

Table 4. Confusion Affect Detection Model Performance Results

Task #4: Data and Feature Engineering Tasks Choice and Discussion

I chose five data engineering RapidMiner operators below to try and improve my best original affect model detectors. The rationale for choosing these specific RapidMiner operators is as follows:

SMOTE Operator:
The SMOTE operator in RapidMiner applies the Synthetic Minority Over-sampling Technique to generate synthetic examples for the minority class, thereby balancing class distributions in imbalanced datasets. Data science professionals can use a SMOTE operator to mitigate classifier bias and to balance classes. As an example, per my dataset statistics, the label of frustration is comprised of approximately 96% of false datapoints and 4% of true datapoints. The SMOTE operator will upsample the true datapoints and will thereby synthetically increase the minority class. Doing this should help my models achieve a higher recall metric and should improve my model's AUC metric. It should be noted however that the SMOTE operator can also replicate existing biases in datasets as it generates synthetic minority samples by interpolating existing samples. It is said in various online sources that Fair-SMOTE process can be used to ameliorate

the bias replication. Being unfamiliar with Fair-SMOTE, I will use a generic SMOTE operator for this assignment. (Please see Figure 14 in Appendix for SMOTE Operator visualization)

Normalize Operator:
The Normalize operator in RapidMiner will help prevent features with large variance from dominating the model and will scale numeric attributes to a common range or distribution (e.g., min–max scaling or Z-transformation), thus ensuring all features contribute equally to model training. Data professionals typically use normalization to eliminate scale disparities among variables. I am hoping that utilizing this operator in my process will lead to measurable gains in evaluation metrics and will result in higher accuracy, balanced precision/recall, and improved AUC. (Please see Figure 15 in Appendix for Normalize Operator visualization)

Forward Feature Selection Operator:
The Forward Selection Operator in RapidMiner starts with an empty set of attributes and iteratively adds the unused attribute that yields the greatest boost in model performance. Data science professionals use this operator to isolate the most predictive features, reducing dimensionality and preventing overfitting in both Logistic Regression and Naive Bayes models. After some experimentation with a feature parameter, I am setting my maximum features parameter to 10, as reducing or increasing the number of features from a number 10 did not in my practice lead to a model improvement. (Please see Figure 16-18 in Appendix for FFS Operator visualization)

PCA Operator:
As seen in the previous assignment, the PCA (Principal Component Analysis) operator in RapidMiner computes the covariance matrix of input attributes and projects data onto a set of principal components that capture the majority of variance. Data science professionals use PCA to reduce dimensionality and remove multicollinearity. Since I observed multicollinearity of features while examining the correlation matrix of my dataset, I decided to utilize this operator in my process. I am hoping that using this operator will help improve recall and F1 metrics of my models by reducing model complexity and mitigating overfitting. As determined based on the prior assignment and prior experimentation, I am setting the number of component parameter to a fixed number of 12. (Please see Figure 22-23 in Appendix for PCA Operator visualization)

K-Means Clustering Operator:
As seen in the previous assignment, K-means is a clustering algorithm that uses distance between data points to locate a centroid. K-means attempts to find clusters that maximize internal cohesion and external separation of data points. Clustering can help identify subpopulations and can be a valuable method for identifying groups of similar examples and thereby can help improve model performance. Data science professionals often use K-Means for feature engineering and as an additional attribute to enrich training data for supervised models like logistic regression and Naive Bayes. Based on my previous assignment and previous experimentation, I am setting the number of clusters to 10 in model parameters. I am hoping that by utilizing this operator I will see marked improvements in my models' performance evaluation metrics. (Please see Figure 19-21 in Appendix for K-Means Clustering Operator visualization)

After choosing my data engineering operators, I followed the processes outlined in previous class assignments to integrate these operators into my best performing Logistic Regression and Naïve Bayes models for each of the four student affects.

Task #5: Affect Models' Performance Tables after Data Engineering Tasks

Below are the performance evaluation tables that list performance metrics for the "original" models as well as performance metrics for the best performing models after application of data engineering tasks. Considering the importance of positive labels of frustration, boredom, and confusion, I paid special attention to the Recall performance metric in identifying the best performing model for a real-world educational setting.

Model Name	Accuracy	Cohen's Kappa	AUC	Precision	Recall	F1
Major Class	96.25%	0	0.5	- -	0%	- -
Logistic Regression	**- -**	**0.097**	**0.667**	**11.52%**	**16%**	**13.04%**
Naïve Bayes	- -	0.069	0.640	8.65%	28%	11.87%
Decision Tree	- -	0.012	0.511	20%	0.83%	1.64%
Deep Learning		0.096	0.706	11.62%	30.17%	13.80%

Table 1. Frustration Affect Detection Models' Performance Results

Model Name	Accuracy	Cohen's Kappa	AUC	Precision	Recall	F1
Logistic Regression (SMOTE)		**0.055**	**0.681**	**6.60%**	**65.83%**	**11.91%**
LR with Normalize		0.095	0.661	11.16%	16%	12.88%
LR with Forward Feature Selection (10 max features)		0.060	0.701	8.73%	9.67%	9.71%
LR with PCA (12 components)		0.085	0.684	9.50%	18.67%	13.56%
LR with Clustering (10 Clusters)		0.111	0.669	12.36%	18.17%	14.53%

Table 1A. Frustration Models' Performance after Data Engineering Tasks

Model Name	Accuracy	Cohen's Kappa	AUC	Precision	Recall	F1
Major Class	82.24%	0	0.5	- -	0%	- -
Logistic Regression	- -	0.147	0.748	83.87%	97.17%	90.01%
Naïve Bayes	**- -**	**0.215**	**0.713**	**86.12%**	**86.04%**	**85.96%**

Decision Tree	- -	0.030	0.582	82.51%	99.47%	90.17%
Deep Learning		0.042	0.743	82.67%	98.63%	89.92%

Table 2. Concentration Affect Detection Models' Performance Results

Model Name	Accuracy	Cohen's Kappa	AUC	Precision	Recall	F1
Naïve Bayes (SMOTE)		0.190	0.694	88.17%	69.63%	76.35%
NB with Normalize		0.219	0.714	86.30%	85.51%	85.77%
NB with Forward Feature Selection (10 features max)		**0.275**	**0.742**	**88.39%**	**81.10%**	**84.54%**
NB with PCA (12 components)		0.166	0.718	84.39%	93.99%	88.90%
NB with Clustering (10 clusters)		0.215	0.713	86.12%	86.04%	85.96%

Table 2A. Concentration Models' Performance following Data Engineering Tasks

Model Name	Accuracy	Cohen's Kappa	AUC	Precision	Recall	F1
Major Class	89.36%	0	0.5	- -	0%	- -
Logistic Regression	**- -**	**0.258**	**0.748**	**30.82%**	**40.93%**	**34.63%**
Naïve Bayes	- -	0.083	0.710	14.64%	83.78%	24.73%
Decision Tree	- -	0.049	0.562	32.79%	3.68%	6.80%
Deep Learning		0.204	0.734	25.16%	40.36%	30.69%

Table 3. Boredom Affect Detection Model Performance Results

Model Name	Accuracy	Cohen's Kappa	AUC	Precision	Recall	F1
Logistic Regression (SMOTE)		**0.241**	**0.763**	**94.45%**	**78.57%**	**85.75%**
LR with Normalize		0.258	0.748	30.82%	40.93%	34.63%
LR with Forward Feature Selection (10 features maximum)		0.231	0.758	33.98%	28.91%	30.42%
LR with PCA (12 components)		0.234	0.746	27.58%	43.50%	33.26%
LR with Clustering (10		0.261	0.749	31.04%	41.23%	34.89%

clusters)						

Table 3A. Boredom Models' Performance following Data Engineering Tasks

Model Name	Accuracy	Cohen's Kappa	AUC	Precision	Recall	F1
Major Class	96.01%	0	0.5	- -	0%	- -
Logistic Regression	**- -**	**0.066**	**0.680**	**8.38%**	**19.26%**	**11.43%**
Naïve Bayes	- -	0.028	0.617	5.53%	53.40%	9.51%
Decision Tree	- -	0.013	0.615	16.67%	0.83%	1.55%
Deep Learning		0.061	0.676	8.68%	22.14%	10.97%

Table 4. Confusion Affect Detection Model Performance Results

Model Name	Accuracy	Cohen's Kappa	AUC	Precision	Recall	F1
Logistic Regression (SMOTE)		**0.064**	**0.694**	**7.29%**	**69.08%**	**13.15%**
LR with Normalize		0.066	0.680	8.38%	19.26%	11.43%
LR with Forward Feature Selection (10 features max)		0.063	0.618	10.08%	9.66%	9.36%
LR with PCA (12 components)		0.029	0.682	5.26%	21.17%	10.23%
LR with Clustering (10 clusters)		0.064	0.679	8.26%	19.26%	11.36%

Table 4A. Confusion Models' Performance following Data Engineering Tasks

Task #6: Real-World Application of Final Models Discussion

Model Name	Accuracy	Cohen's Kappa	AUC	Precision	Recall	F1
Frustration – LR with SMOTE		**0.055**	**0.681**	**6.60%**	**65.83%**	**11.91%**
Concentration – NB with Forward Feature Selection		**0.275**	**0.742**	**88.39%**	**81.10%**	**84.54%**
Boredom – LR with SMOTE		**0.241**	**0.763**	**94.45%**	**78.57%**	**85.75%**

Confusion – LR with SMOTE		**0.064**	**0.694**	**7.29%**	**69.08%**	**13.15%**

Table 5. Best Performing Affect Models as applied in Real-World Settings

The goal of this assignment was produce a model for each affect that I am confident can be applied in a real-world educational setting and NOT to produce a model with maximized Cohen's Kappa and AUC values, therefore, the best performing model to detect the affect of frustration, boredom, and confusion, in my mind, will be the model with the highest Recall metric value and an acceptable Kappa and AUC value. In real-world educational settings false negatives (under-identification of students) can carry a high cost and can delay critical student support and intervention, therefore, a recall performance metric or the percentage of the label values that the model identifies as correct I feel is the one of the most relevant and important model performance metrics in this case. A higher performing model displaying a higher Kappa and AUC values but having lower Recall values may under-identify students who are frustrated, bored, or confused which may result in less supportive intervention, therefore, I feel in a real-world educational setting, I would rather utilize a model that rings a false alarms overidentifies many students as being bored, frustrated, or confused, rather than use a model that under-identifies students that need help.

Since SMOTE synthetically increases the minority class, it is not surprising that three out four best performing models above have SMOTE components. Frustration and Confusion best models also have Kappa, AUC, and Precision values on the lower side, however, the trade-off of increased identification of truly frustrated and confused students is worth it.

For my best Concentration affect model, I selected the Naïve Bayes with Forward Feature Selection Model with a Kappa=0.275, AUC of 0.742, and Recall of 81%. Although by choosing this model I am identifying a slightly lesser percentage of students with an affect of concentration, I am achieving a model performance of approximately 28% above random guessing and a higher differentiation between classes as reflected in the highest AUC for the affect of Concentration.

In summary, given my limited knowledge and novice experience in the field of Learning Analytics, the models in Table 5 above I am confident represent the best model detectors for each of the four student affects. And, although some of these models (frustration and confusion) lack precision, perform slightly above random guessing, and exhibit lower performance metrics, they serve their purpose of capturing a greater number of students who may be frustrated or confused and therefore can be utilized in real-world educational settings. Also, as I did not assess the above algorithms as to the algorithmic bias, further fairness analyses will need to be conducted and possibly Fair-SMOTE process needs to be utilized prior to applying these algorithms in practice in urban, rural, or suburban educational settings.

Appendix:

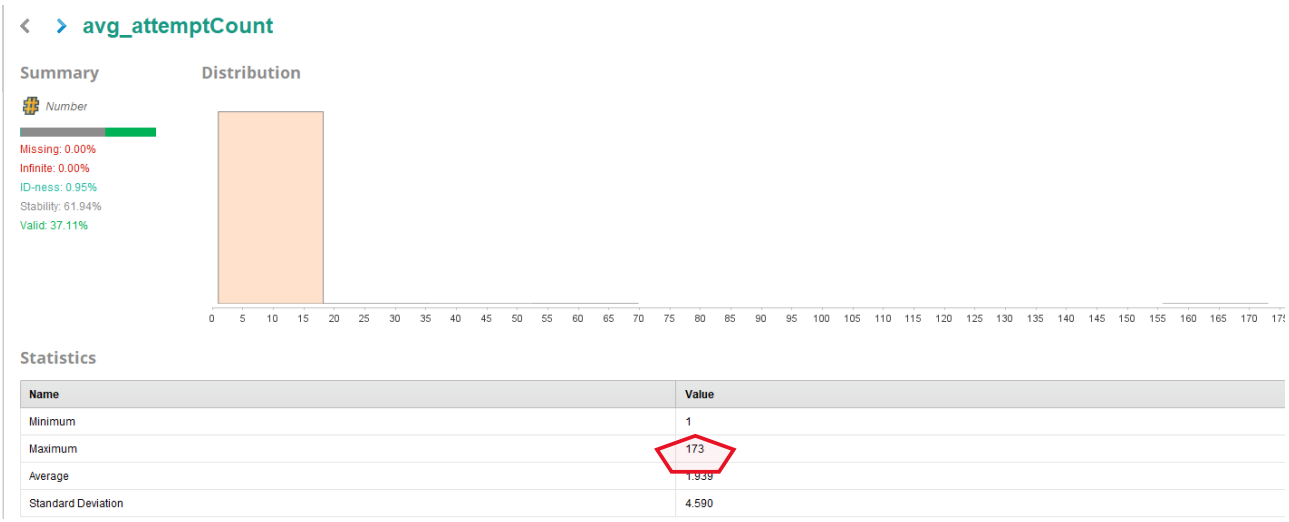

Figure 1. Dataset Statistics Example 1

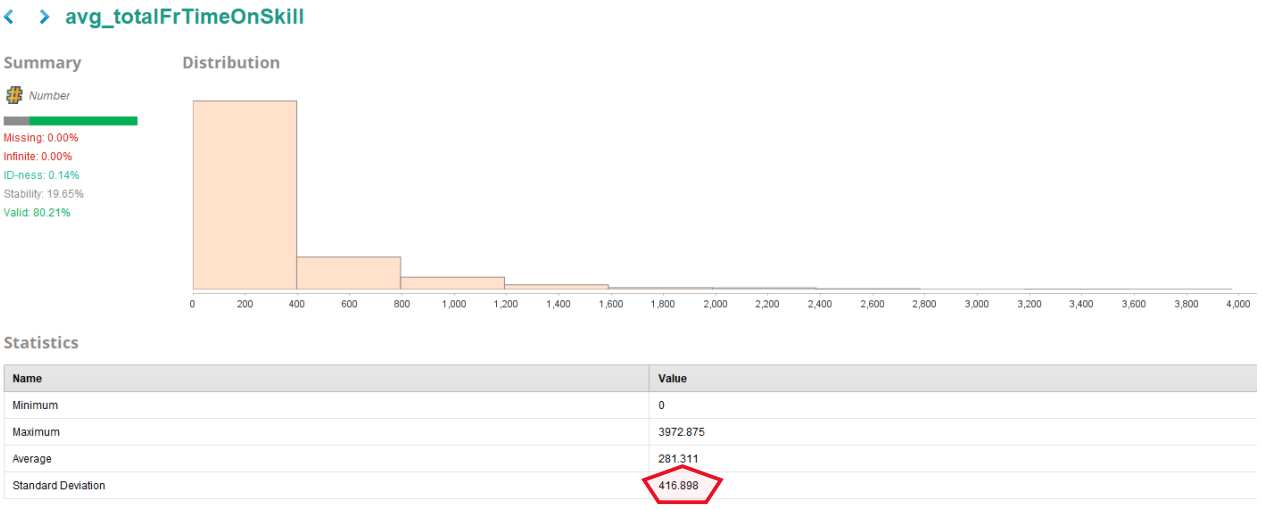

Figure 2. Dataset Statistics Example 2

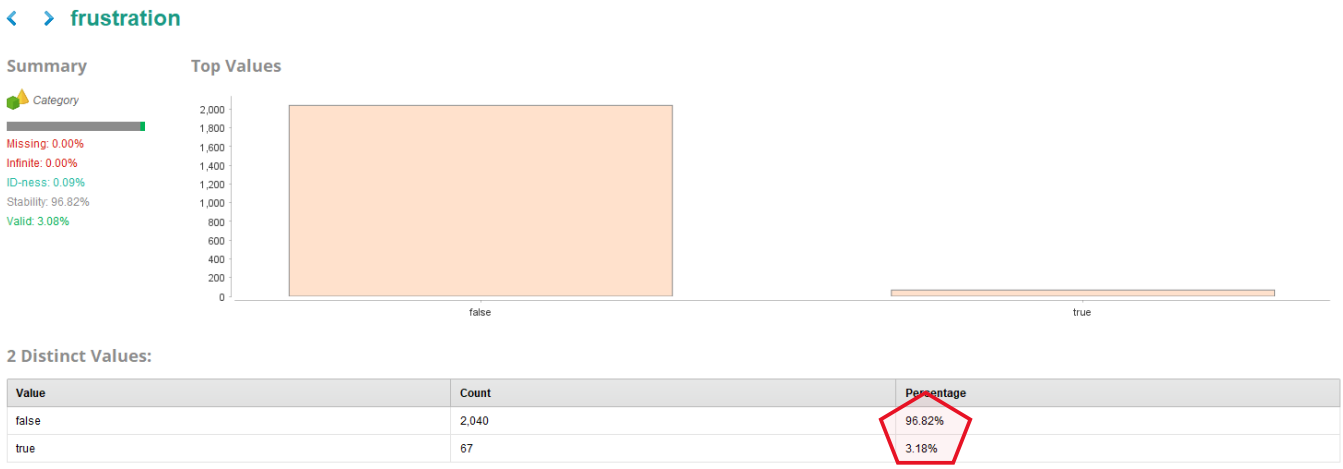

Figure 3. Frustration Statistics

Figure 4. Concentration Statistics

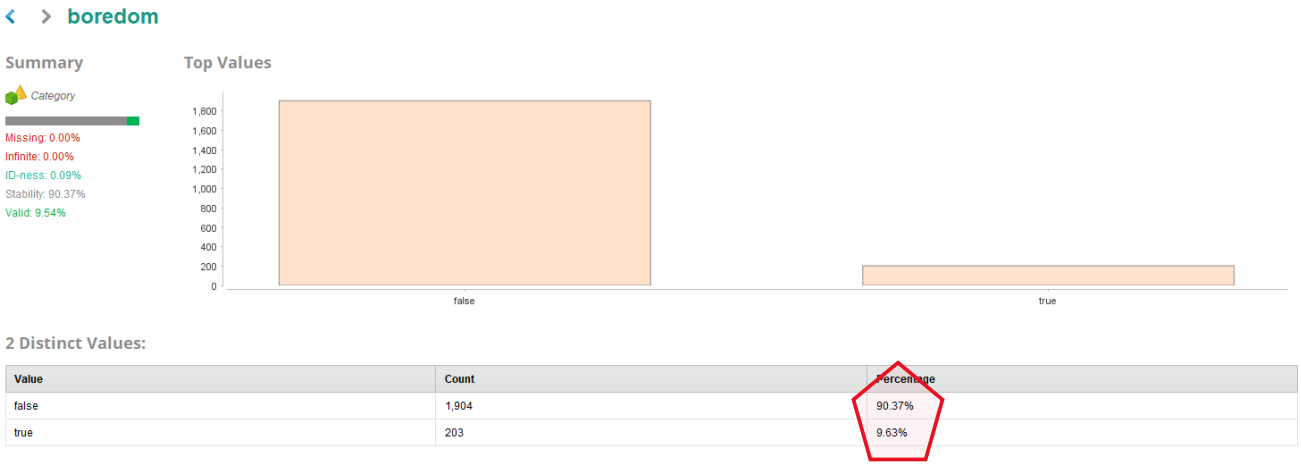

Figure 5. Boredom Statistics

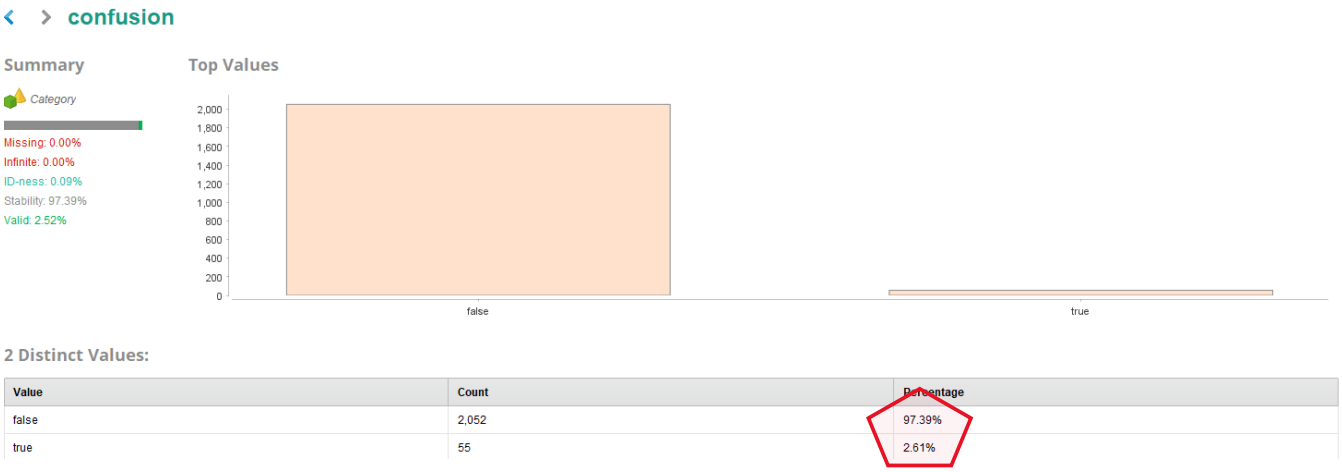

Figure 7. Confusion Statistics

Attributes	frustrati...	avg_attemptCount	avg_bot...	avg_cor...	avg_frIs...	avg_frP...	avg_frP...	avg_frP...	avg_frP...	avg_fr...	avg_hint	avg_hintCount
avg_timeSinceSkill	?	?	?	?	?	?	?	?	?	?	?	?
avg_timeTaken	0.009	-0.048	-0.062	0.050	0.029	-0.052	-0.087	-0.046	-0.085	0.073	-0.064	-0.114
avg_totalFrAttempted	-0.088	-0.034	0.001	0.070	0.024	0.024	-0.037	0.018	-0.038	-0.537	-0.027	-0.032
avg_totalFrPastWrongC...	-0.013	0.048	0.061	-0.117	0.025	0.055	0.129	0.055	0.128	-0.281	0.084	0.100
avg_totalFrPercentPast...	-0.043	-0.115	-0.113	0.144	0.078	-0.034	-0.036	-0.038	-0.031	-0.088	-0.117	-0.206
avg_totalFrSkillOpportu...	0.026	0.242	0.095	-0.090	0.045	0.070	-0.001	0.069	-0.001	-0.240	0.118	0.163
avg_totalFrTimeOnSkill	-0.011	0.062	0.064	-0.044	-0.012	0.049	0.057	0.055	0.056	-0.187	0.040	0.070
max_attemptCount	0.218	0.978	0.108	-0.172	-0.012	0.012	-0.005	0.011	-0.005	0.044	0.116	0.236
max_bottomHint	0.086	0.119	0.821	-0.290	0.128	0.196	-0.034	0.190	-0.034	0.001	0.618	0.615
max_correct	-0.164	-0.130	-0.128	0.883	-0.213	-0.045	-0.278	-0.055	-0.277	-0.085	-0.336	-0.141
max_frIsHelpRequest	0.001	-0.004	0.212	-0.280	0.818	0.358	-0.073	0.348	-0.072	-0.033	0.486	0.278
max_frPast5HelpRequest	0.003	0.015	0.256	-0.185	0.296	0.833	-0.063	0.813	-0.063	-0.051	0.431	0.341
max_frPast5WrongCount	0.057	-0.012	0.013	-0.457	-0.070	-0.046	0.869	-0.041	0.863	-0.045	0.043	-0.077
max_frPast8HelpRequest	0.002	0.014	0.254	-0.189	0.293	0.827	-0.058	0.833	-0.057	-0.048	0.427	0.337
max_frPast8WrongCount	0.057	-0.012	0.012	-0.456	-0.069	-0.046	0.872	-0.041	0.870	-0.044	0.043	-0.077
max_frWorkingInSchool	0.080	0.052	0.042	-0.050	-0.004	-0.022	-0.013	-0.016	-0.011	0.996	0.049	0.047
max_hint	0.108	0.127	0.593	-0.452	0.335	0.274	-0.006	0.265	-0.007	0.018	0.941	0.682
max_hintCount	0.070	0.237	0.545	-0.320	0.147	0.226	-0.089	0.218	-0.089	0.020	0.693	0.958
max_hintTotal	0.167	0.240	0.311	-0.377	0.134	0.174	-0.010	0.166	-0.008	0.201	0.553	0.652

ML5 visualization of the matrix

Figure 8. Dataset Correlation Matrix

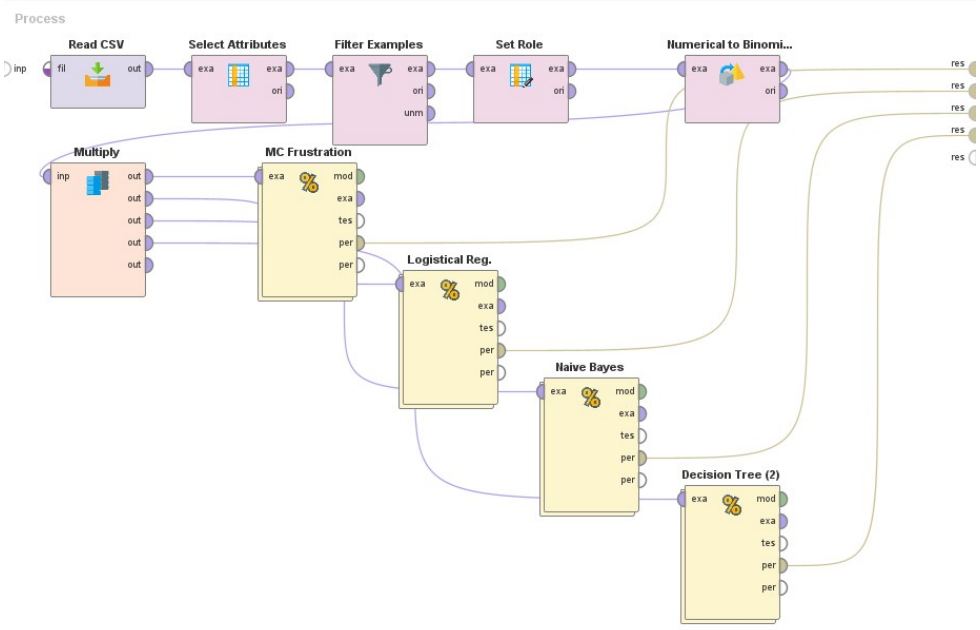

Figure 9. Frustration Affect Models

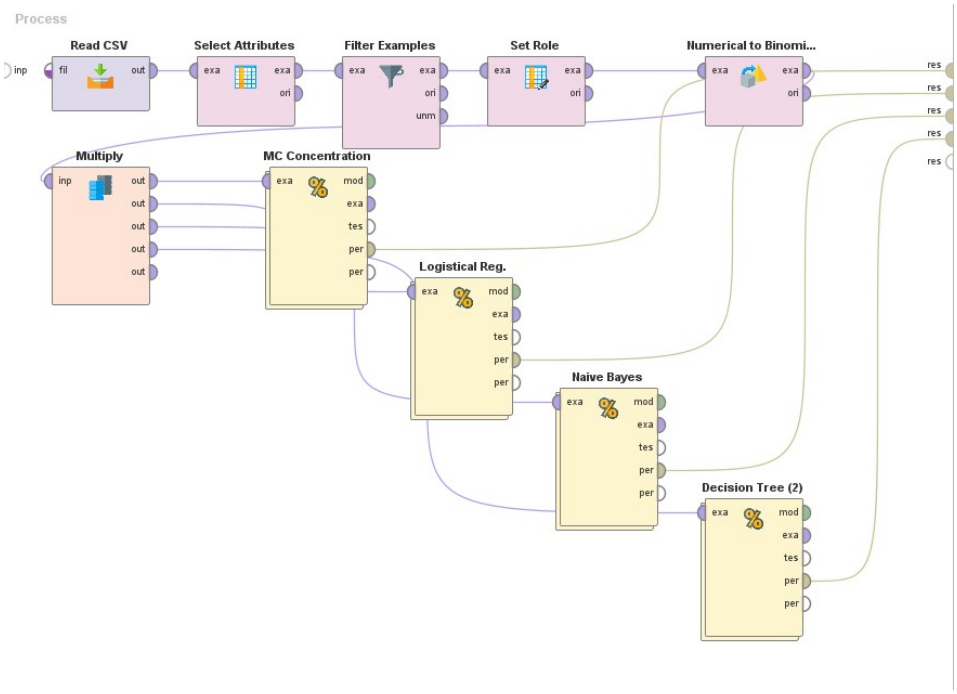

Figure 10. Concentration Affect Models

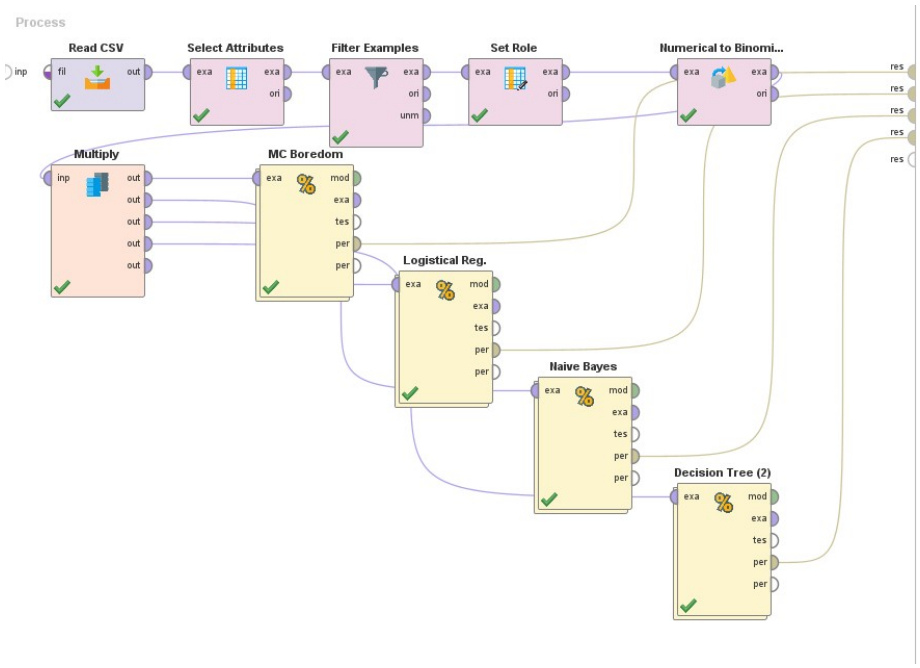

Figure 11. Boredom Affect Models

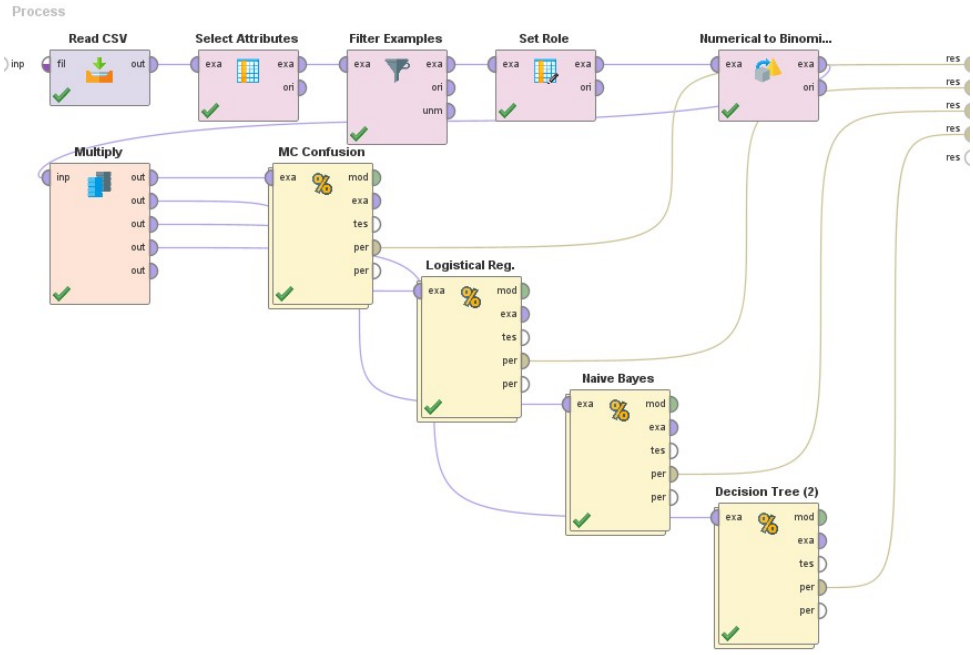

Figure 12. Confusion Affect Models

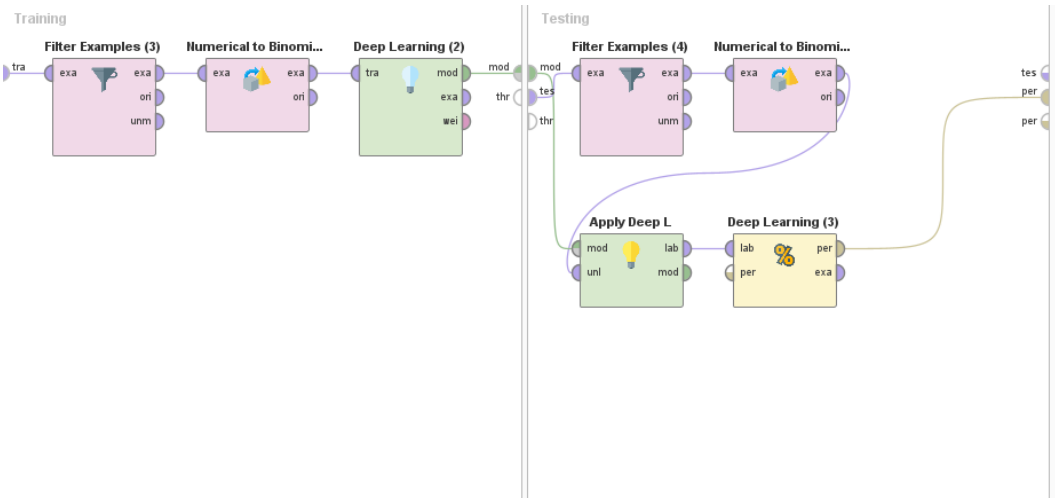

Figure 13. Inside Deep Learning Model Cross Validation Operator

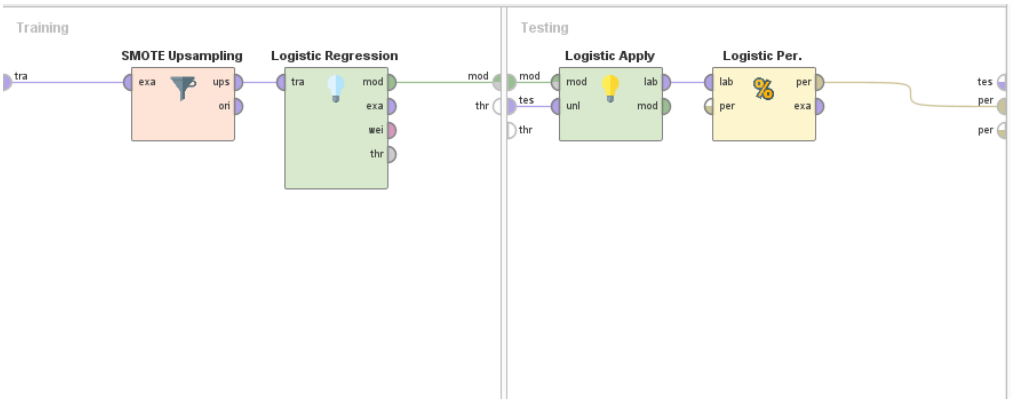

Figure 14. SMOTE Upsampling Operator

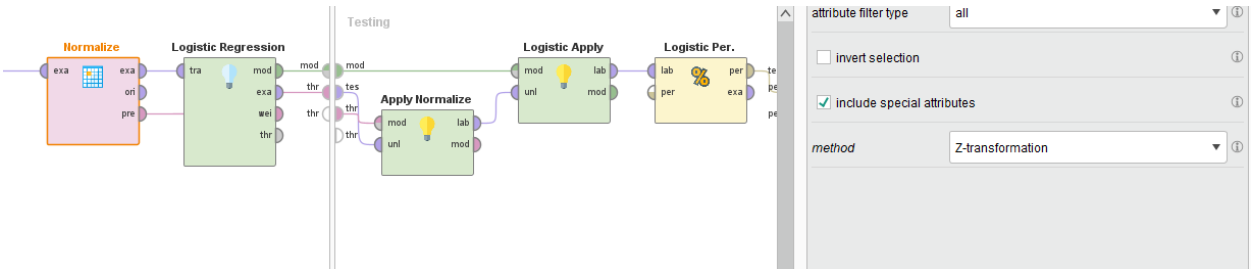

Figure 15. Normalize Features Operator

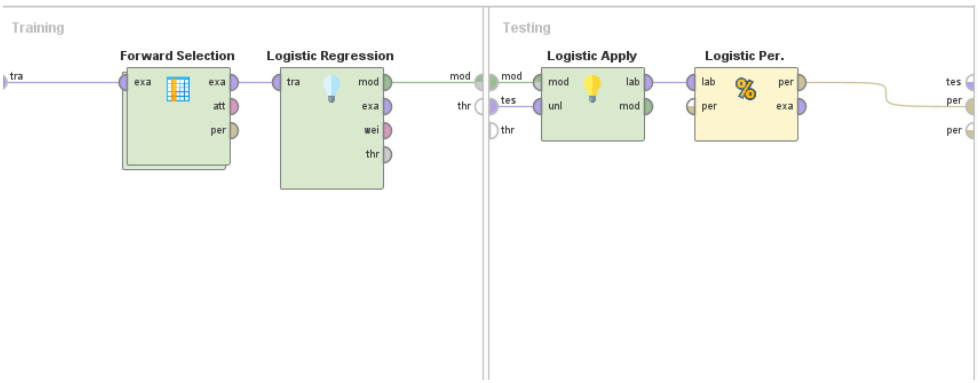

Figure 16. Logistic Regression with Forward Selection Operator

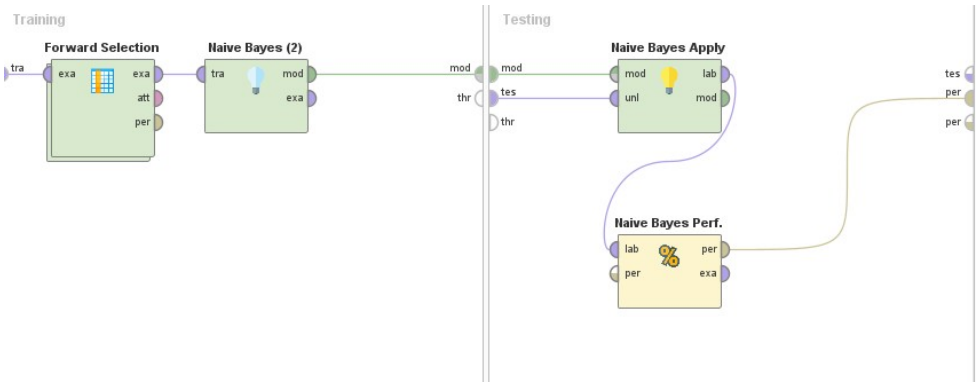

Figure 17. Naïve Bayes with Forward Selection Operator

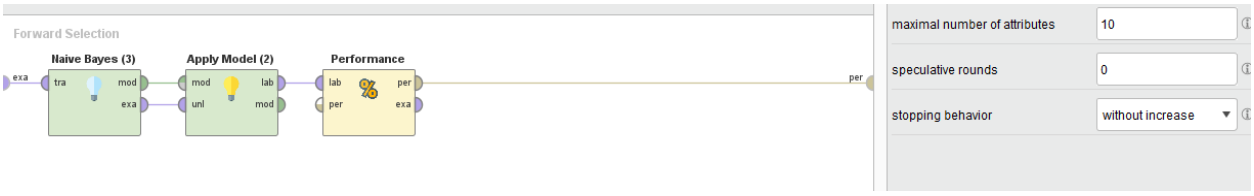

Figure 18. Inside NB with Forward Selection Operator

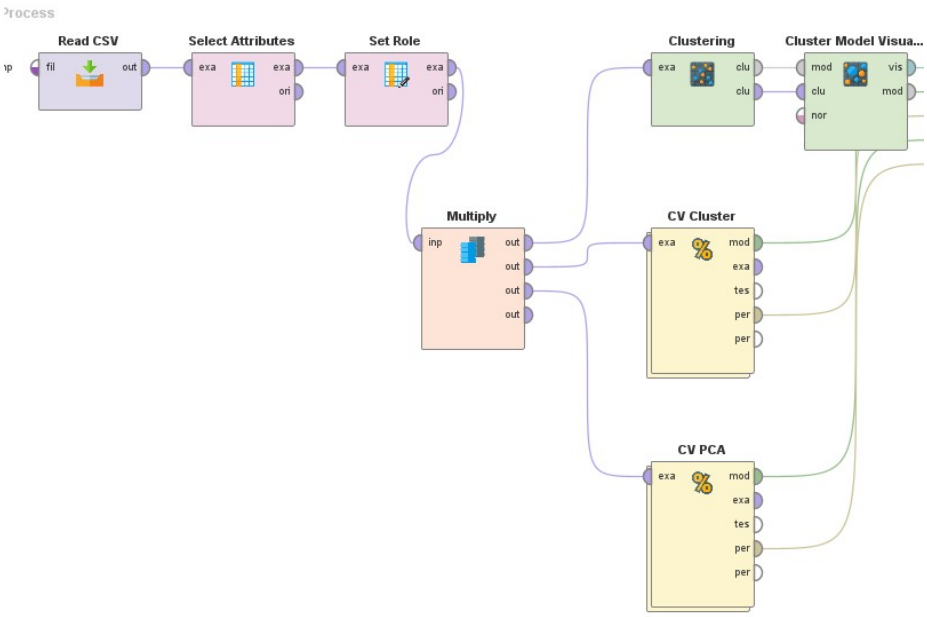

Figure 19. K-Means Clustering and PCA Process

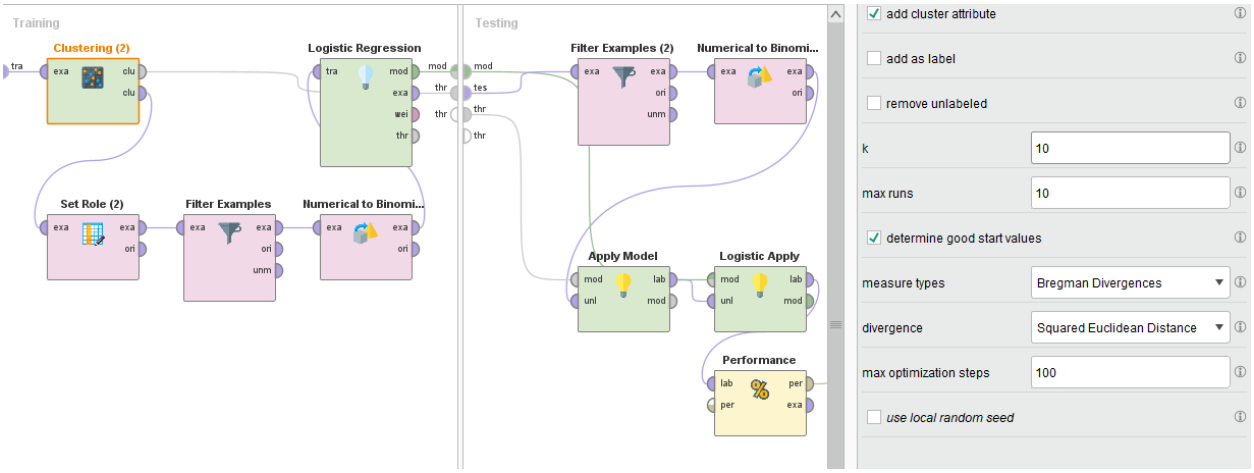

Figure 20. Logistic Regression Model with K-Means Clustering Process

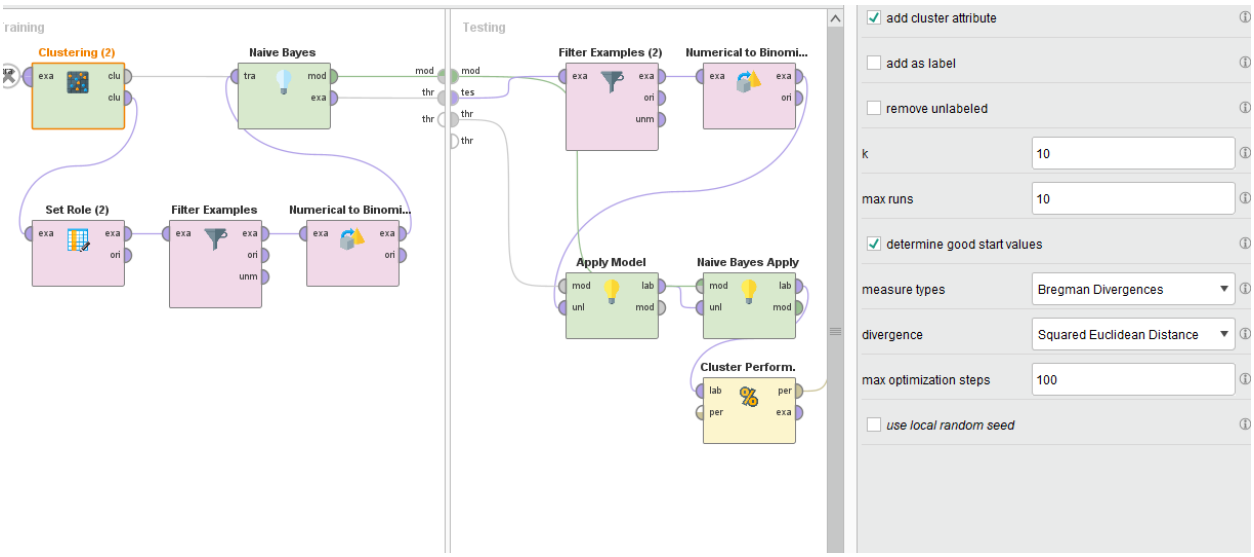

Figure 21. Naïve Bayes Model with K-Means Clustering Process

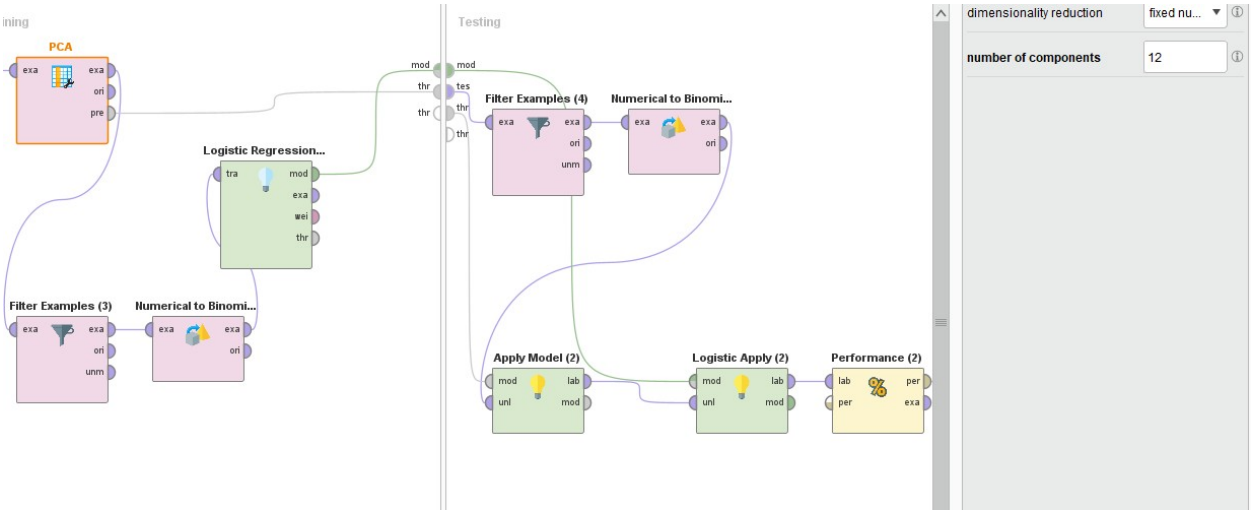

Figure 22. Logistic Regression Model with PCA Process

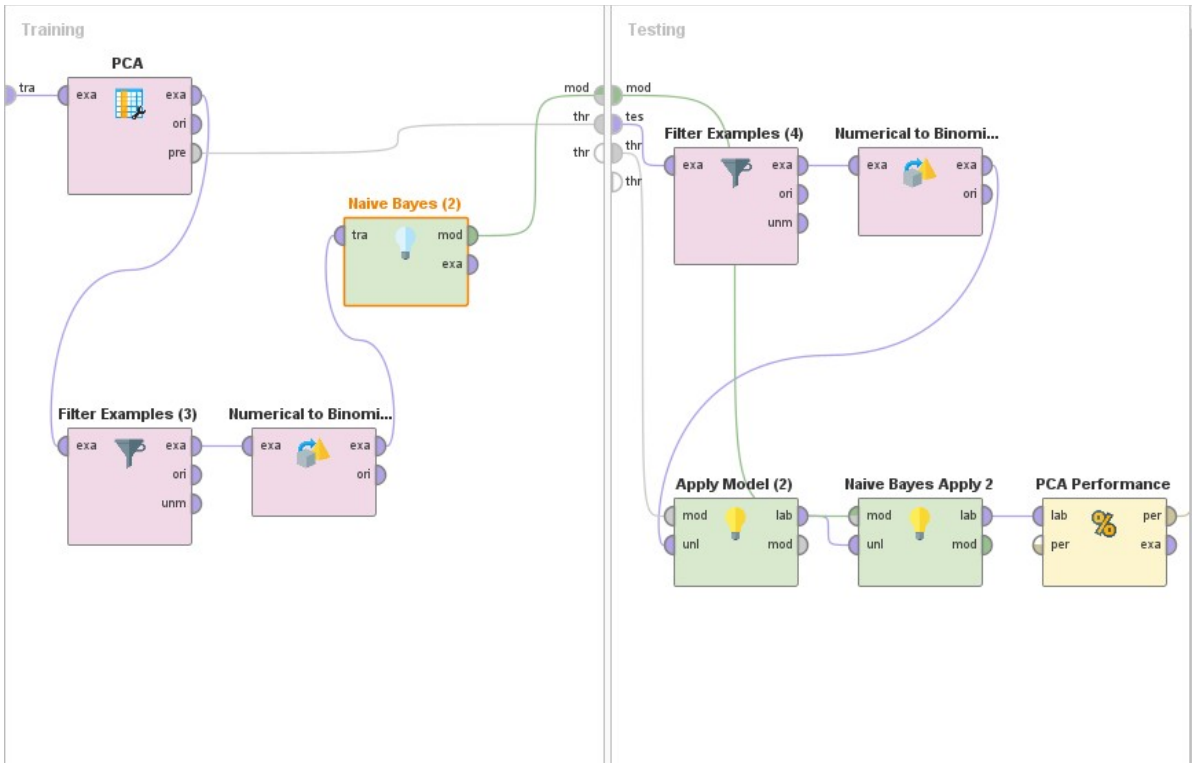

Figure 23. Naïve Bayes Model with PCA Process