Intro to AI

Sentiment Analysis

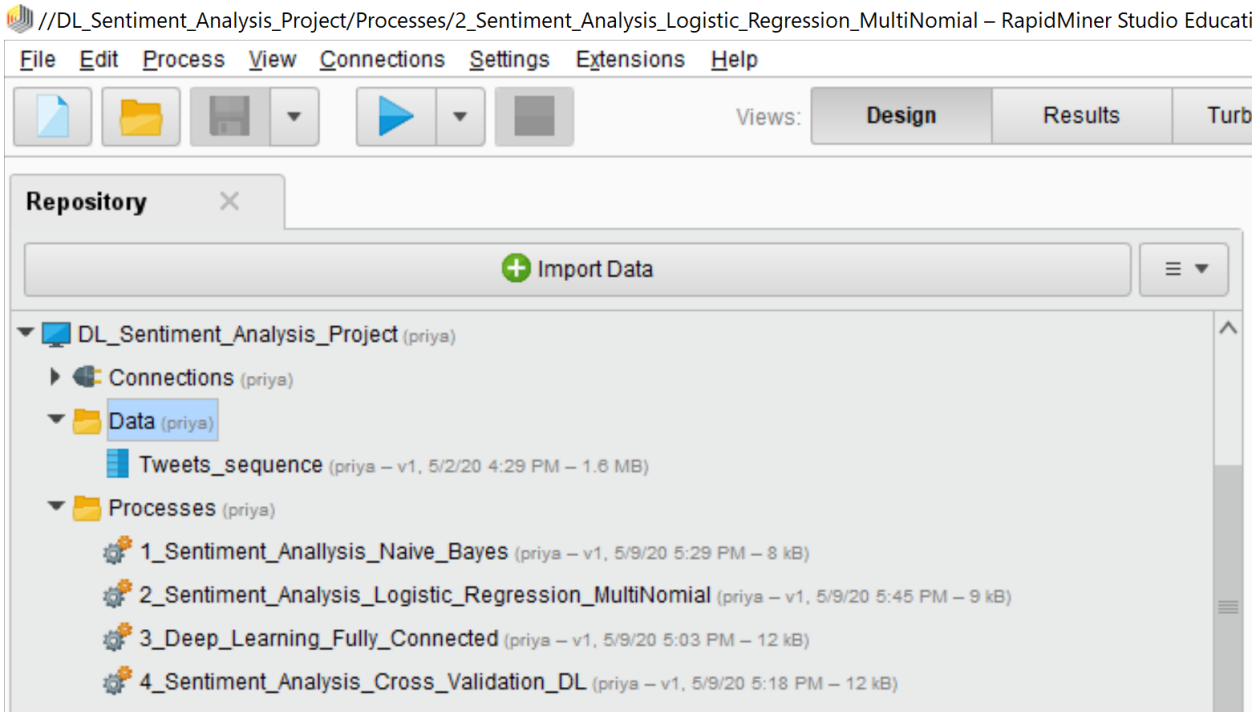
@Author: Priyal Nile(pan303)

RapidMiner Screenshots:

This document elaborates the architetetral designs of the networks for all the models implememtd using rapid minor along with necessary screenshots and justifications.

We implemented following 4 Models in RapidMiner:

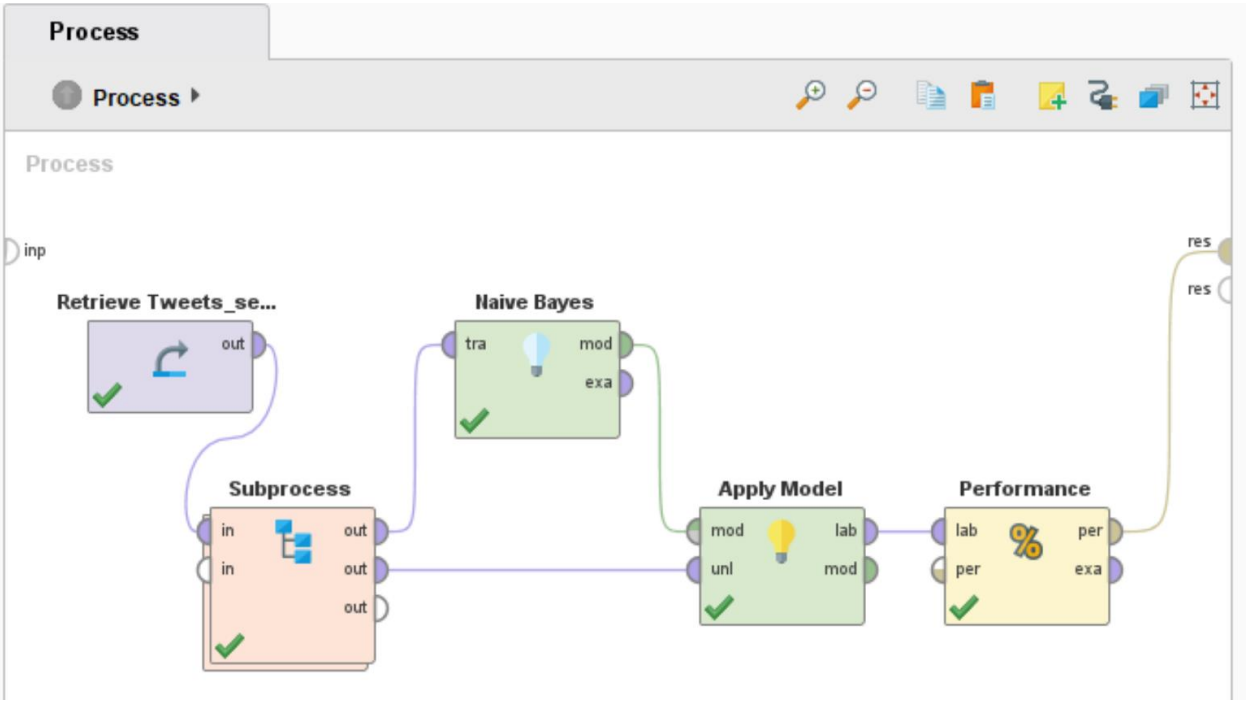
1. **Naïve Bayes**
2. **Multinomial Logistic Regression**
3. **DL: Fully Connected Layers with 80/20 Train-Test Split**
4. **DL: Fully Connected Layers with Cross Validation**

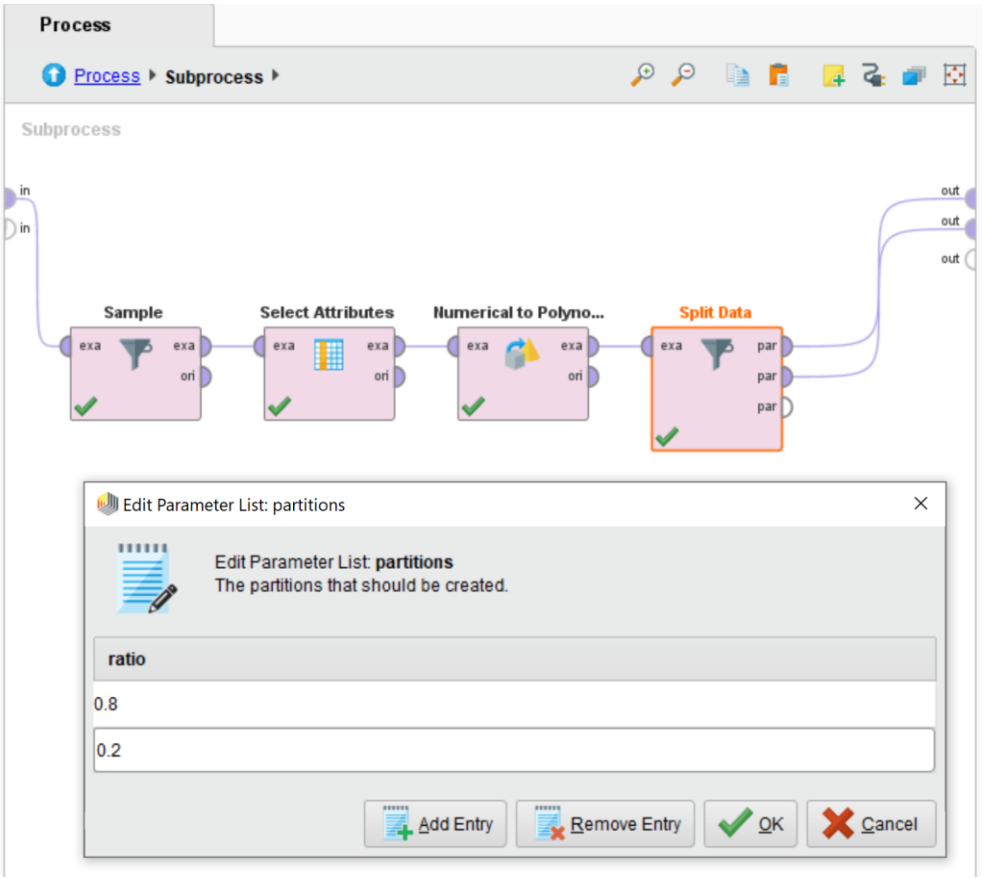


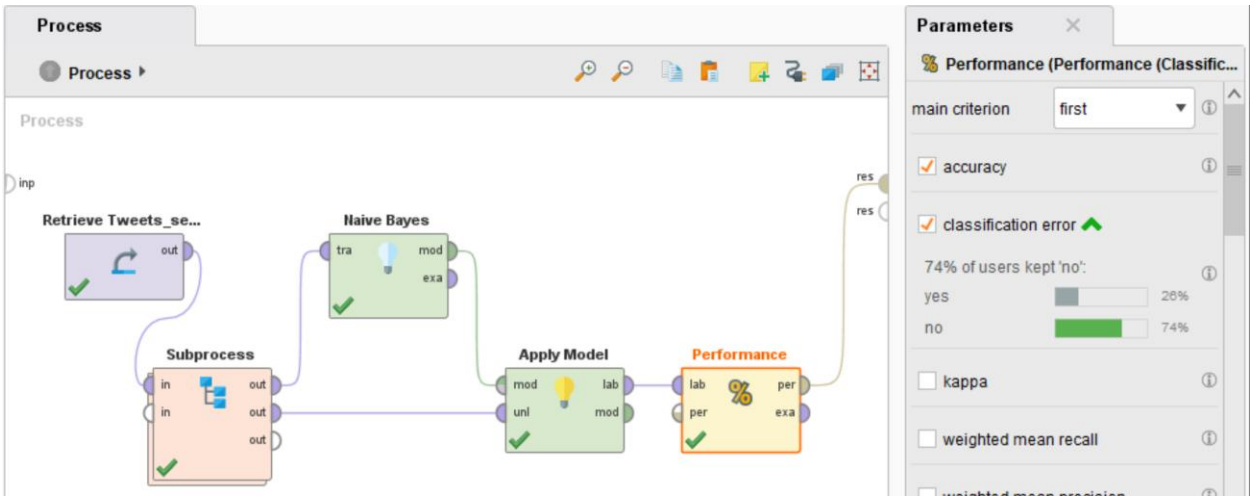
1. **Naïve Bayes: (Accuracy: 60.20)**

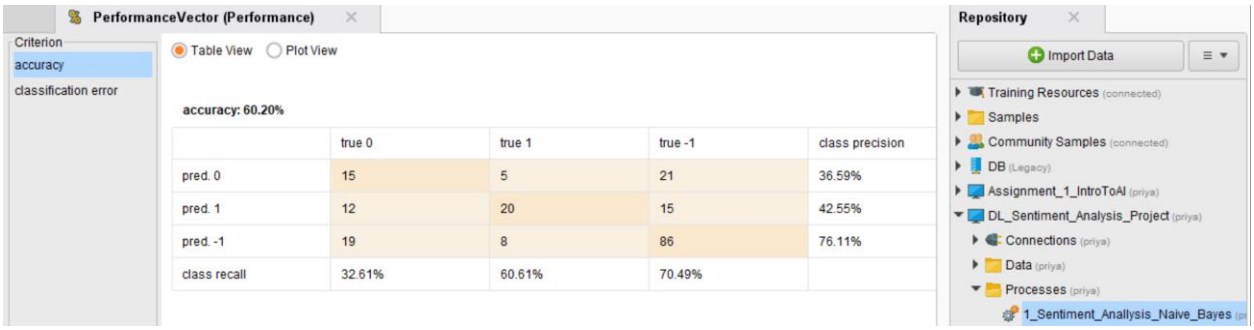
Process Design:

Processed data set is imported in RapidMiner, which is then passed to the Subprocess, where we select sentiment as the target label and set it as Polynimoial, rest of the attributes are also selected and are passed to Numerical to Polynomial operator. Then we split this data into training and test using 0.8/0.2 ratio. Further, this process is passed into the classification model- Naïve Bayes. Next, we use Apply Model to run the model and is connected to the subprocess and classifier both. It applies the trained model on the test data set and the result is passed through the Performance operator to calculate the accuracy of the trained model as depicted below.


Process
Process
Process
inp
Retrieve Tweets_se...
out
Naive Bayes
tra
mod
exa
Subprocess
in
in
out
out
out
Apply Model
mod
unl
lab
mod
Performance
lab
per
per
exa
res
res


Process
Process
Subprocess
Subprocess
in
in
Sample
exa
exa
ori
Select Attributes
exa
exa
ori
Numerical to Polyno...
exa
exa
ori
Split Data
exa
par
par
par
out
out
out
Edit Parameter List: partitions
Edit Parameter List: partitions
The partitions that should be created.
ratio
0.8
0.2
Add Entry
Remove Entry
OK
Cancel





	true 0	true 1	true -1	class precision
pred. 0	15	5	21	36.59%
pred. 1	12	20	15	42.55%
pred. -1	19	8	86	76.11%
class recall	32.61%	60.61%	70.49%	

PerformanceVector (Performance)

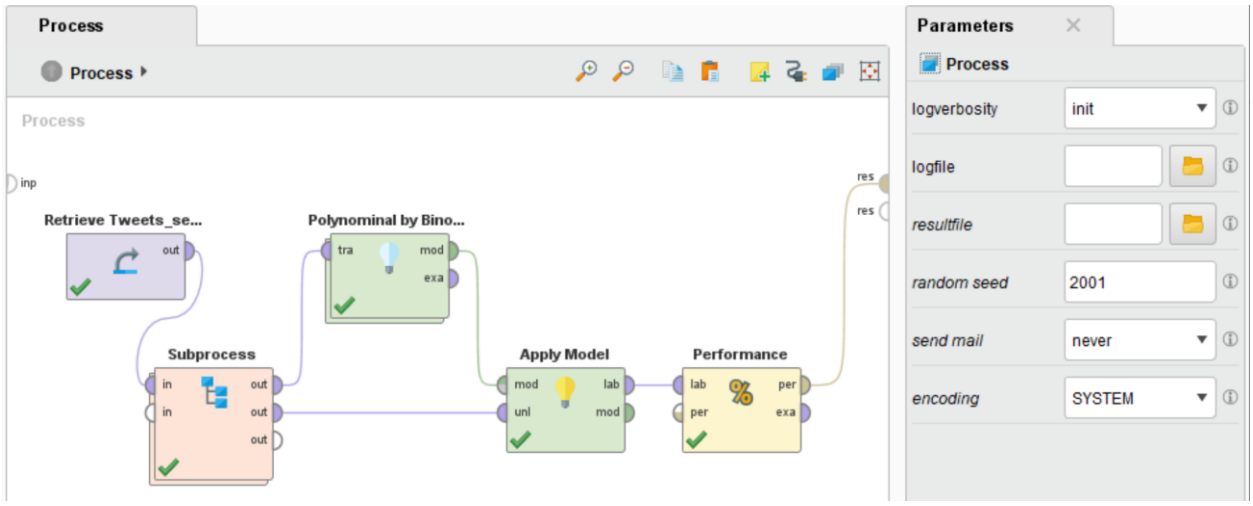
PerformanceVector

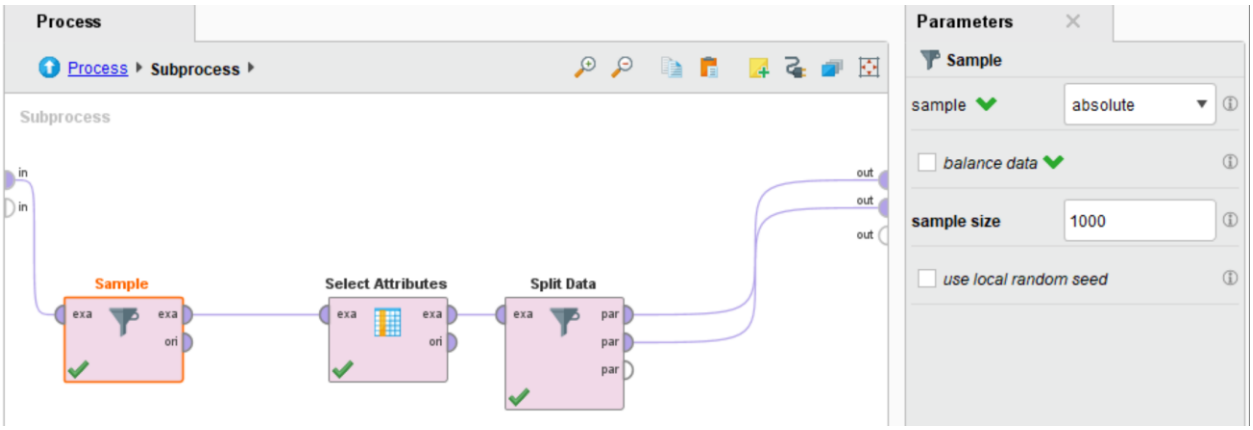
```
PerformanceVector:
accuracy: 60.20%
ConfusionMatrix:
True:    0       1       -1
0:       15      5       21
1:       12      20      15
-1:      19      8       86
classification_error: 39.80%
ConfusionMatrix:
True:    0       1       -1
0:       15      5       21
1:       12      20      15
-1:      19      8       86
```

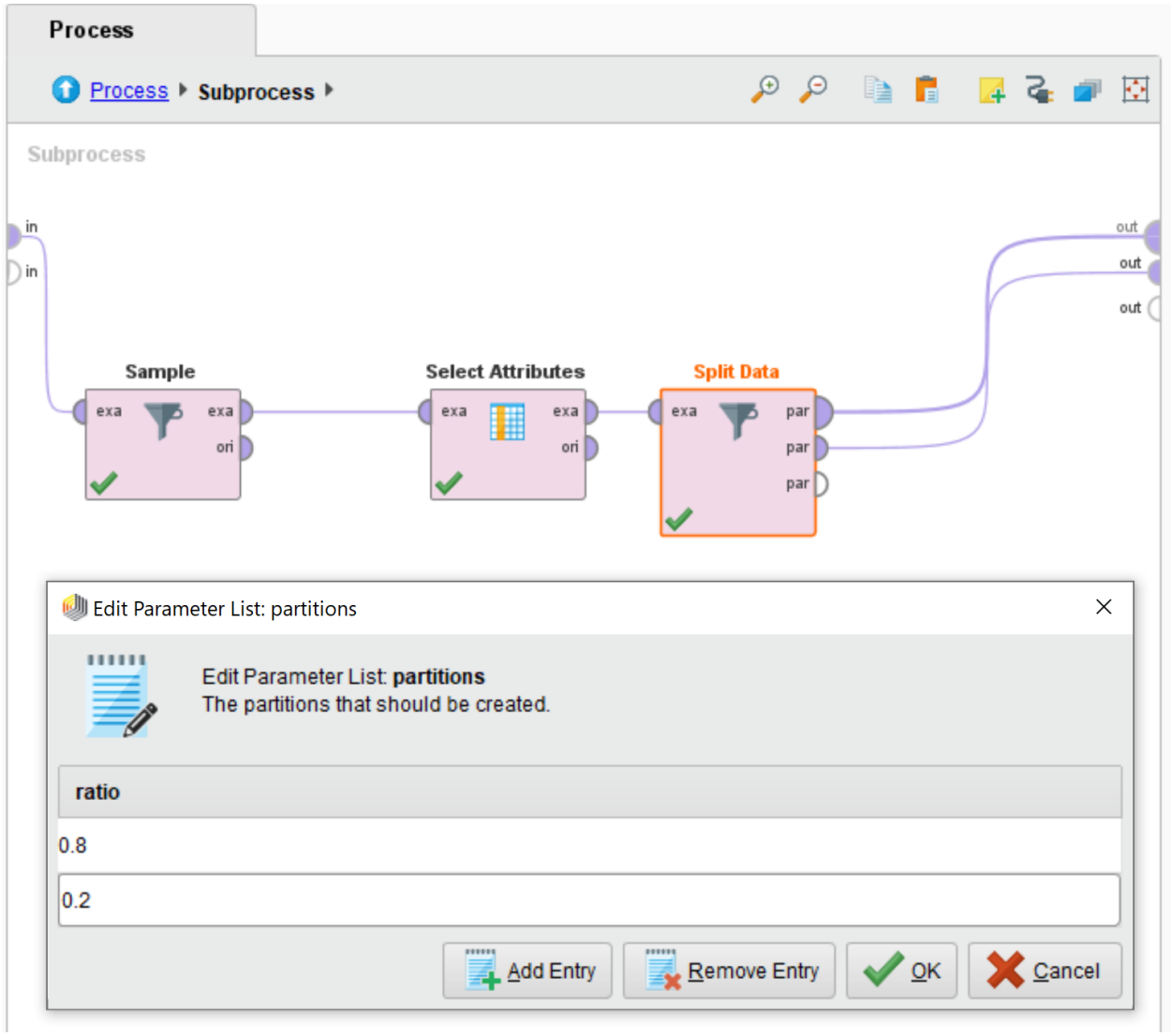
2. Multinomial Logistic Regression: (Accuracy: 61.69)

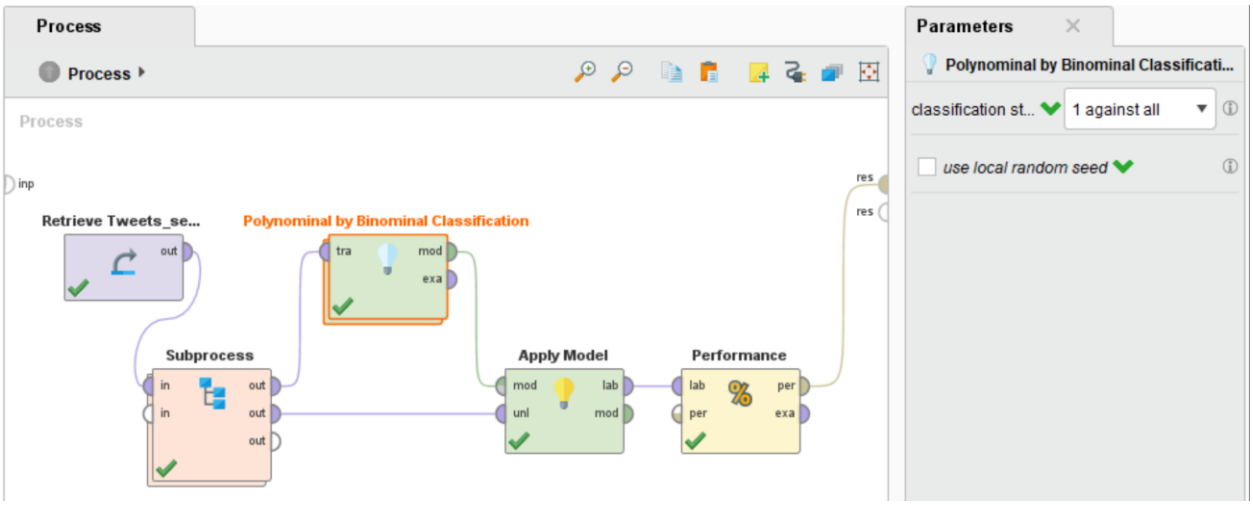
Process Design:

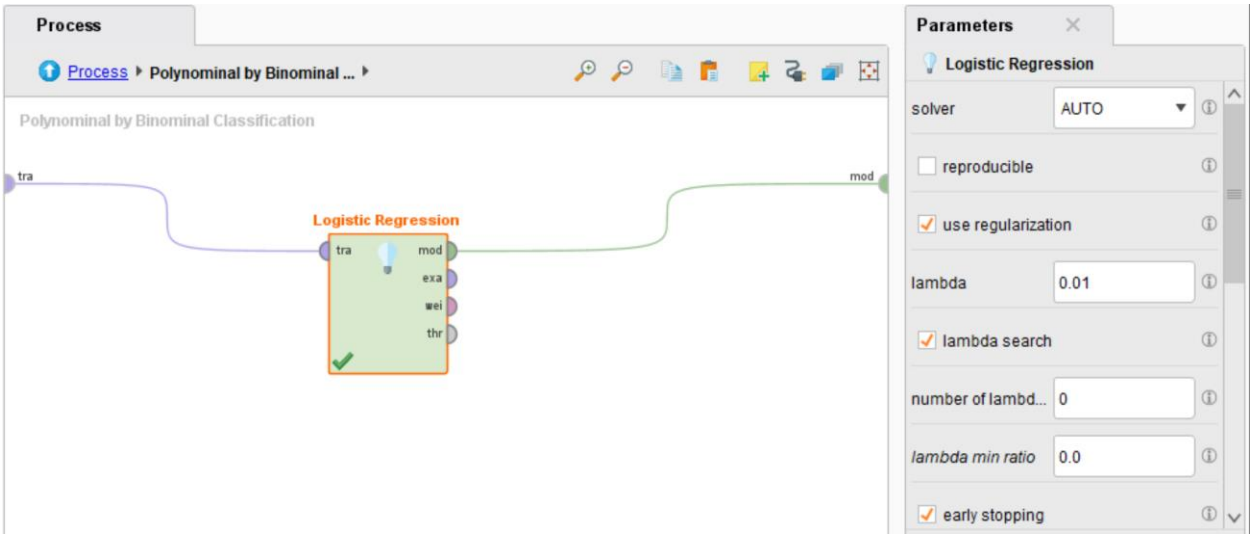
With the loaded Processed data set in RapidMiner, we passed it to subprocess, where we selected attributes and split this data into training and test as before. Further, this process is passed into Polynomial to Binomial Classification model to implement Logistic Regression for Multi-class Classification. Same as above, we then use Apply Model and Performance operator connected to the output to determine accuracy.






Process
Process
Subprocess
Subprocess
in
in
out
out
out
Sample
Select Attributes
Split Data
exa
exa
ori
exa
exa
ori
exa
par
par
par
Edit Parameter List: partitions
Edit Parameter List: partitions
The partitions that should be created.
ratio
0.8
0.2
Add Entry
Remove Entry
OK
Cancel


Process
Process
Process
inp
res
res
Retrieve Tweets_se...
out
Polynominal by Binominal Classification
tra
mod
exa
Subprocess
in
in
out
out
out
Apply Model
mod
unl
lab
mod
Performance
lab
per
per
exa
Parameters
Polynominal by Binominal Classificati...
classification st...
1 against all
use local random seed



PerformanceVector (Performance)

Table View | Plot View

accuracy: 61.69%

	true 0	true 1	true -1	class precision
pred. 0	7	5	6	38.89%
pred. 1	2	5	4	45.45%
pred. -1	37	23	112	65.12%
class recall	15.22%	15.15%	91.80%	

PerformanceVector (Performance)

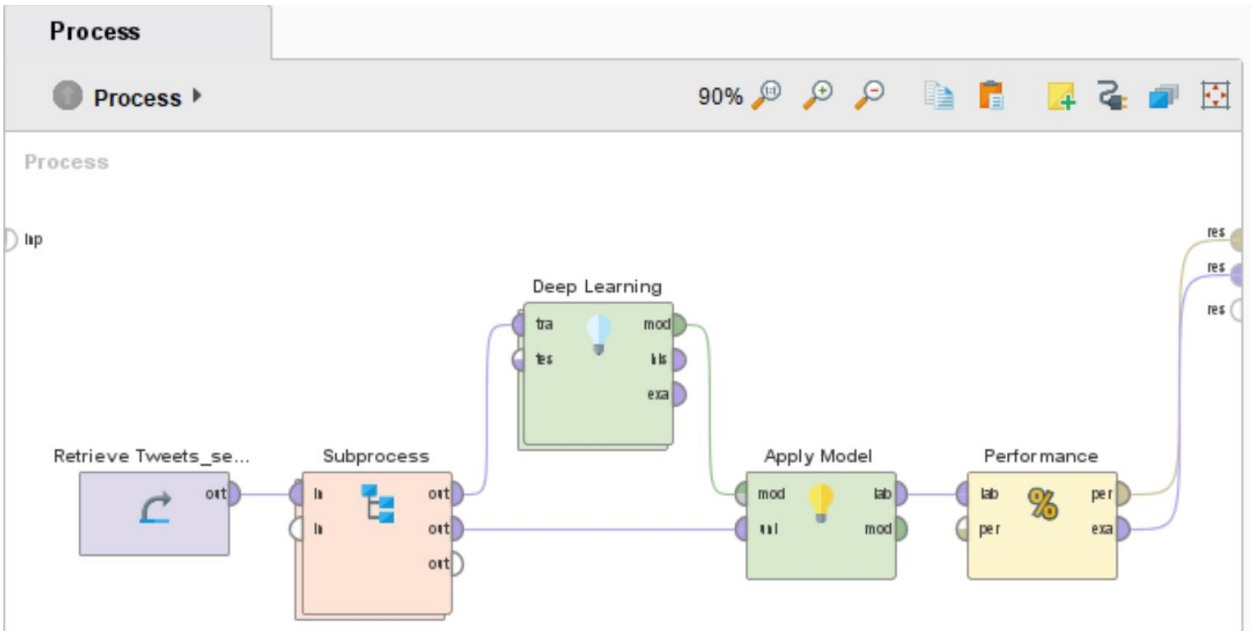
PerformanceVector

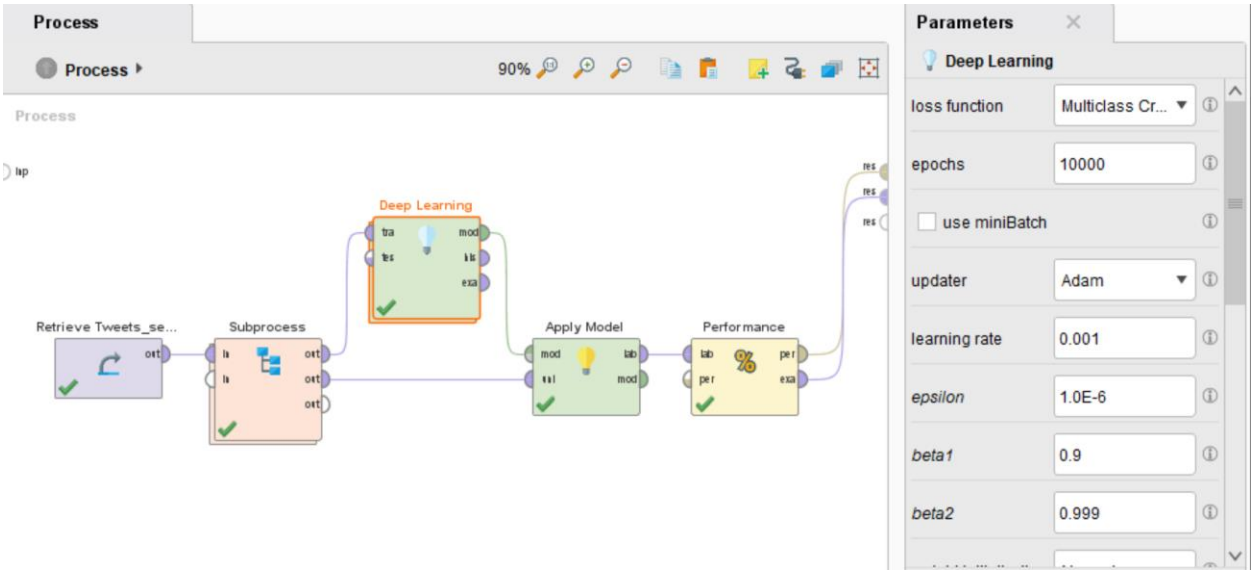
```
PerformanceVector:
accuracy: 61.69%
ConfusionMatrix:
True:    0       1       -1
0:       7       5       6
1:       2       5       4
-1:      37      23      112
classification_error: 38.31%
ConfusionMatrix:
True:    0       1       -1
0:       7       5       6
1:       2       5       4
-1:      37      23      112
```

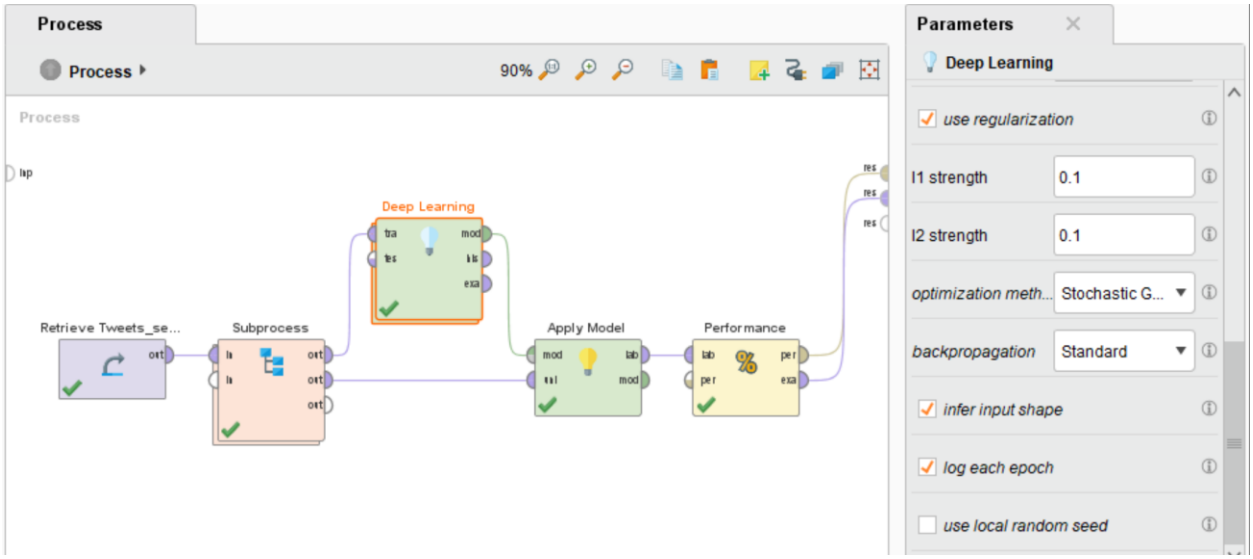
3. DL: Fully Connected Layers with 80/20 Train-Test Split: (Accuracy: 63.30)

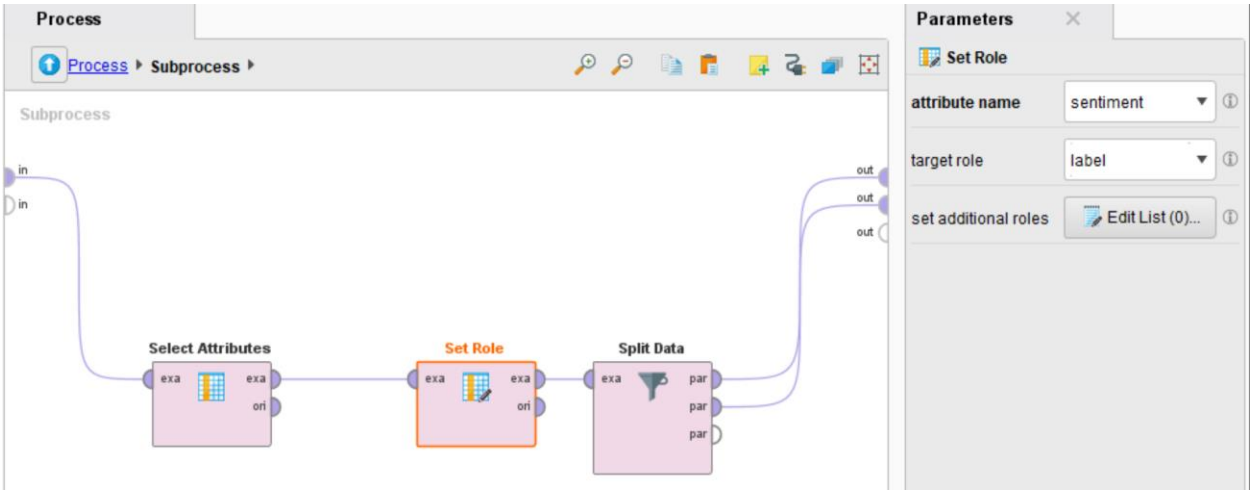
Process Design:

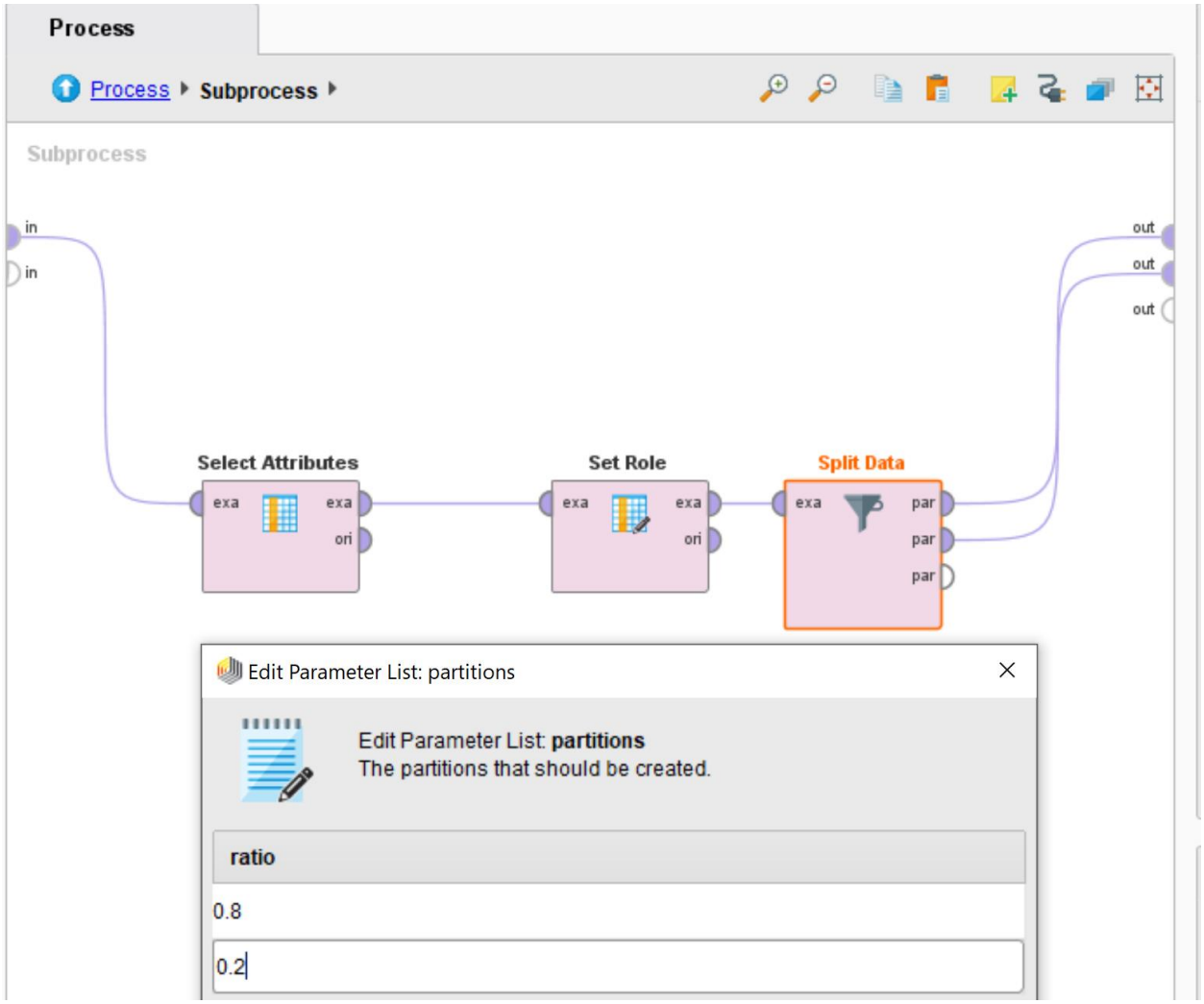
In building this we followed the standard process by passing data set to Subprocess and Set the target Role of attribute. Subsequently, we used a deep learning model with Fully-connected layer (16 neurons + ReLU activation) + Fully-connected layer (8 neuron + ReLU activation) + Fully-connected layer (4 neurons + ReLU activation) + Fully-connected layer (3 neurons + Softmax activation) and set the epochs to 1000. We've then used primitive Apply model and Performance operators to run and evaluate the accuracy of the model.

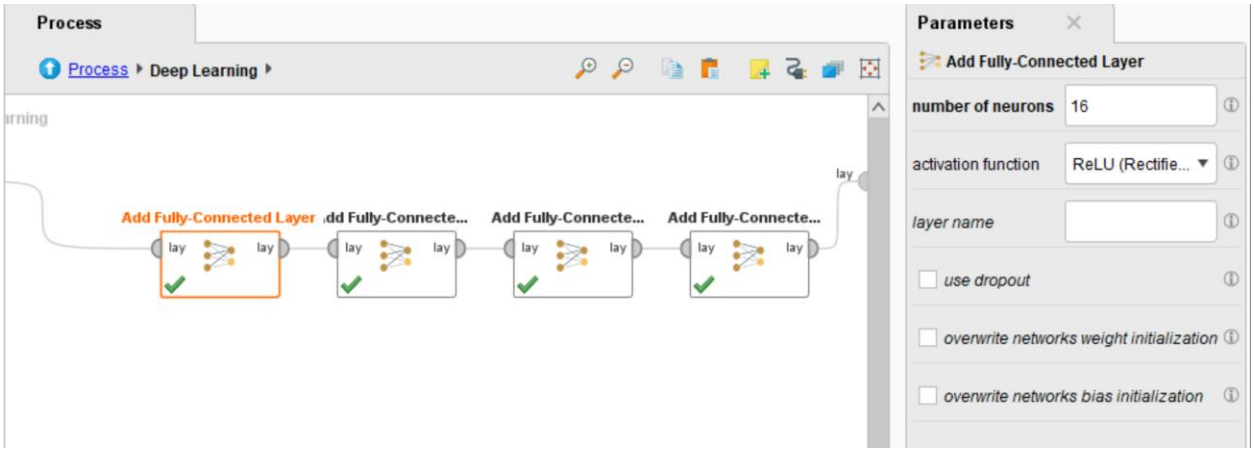


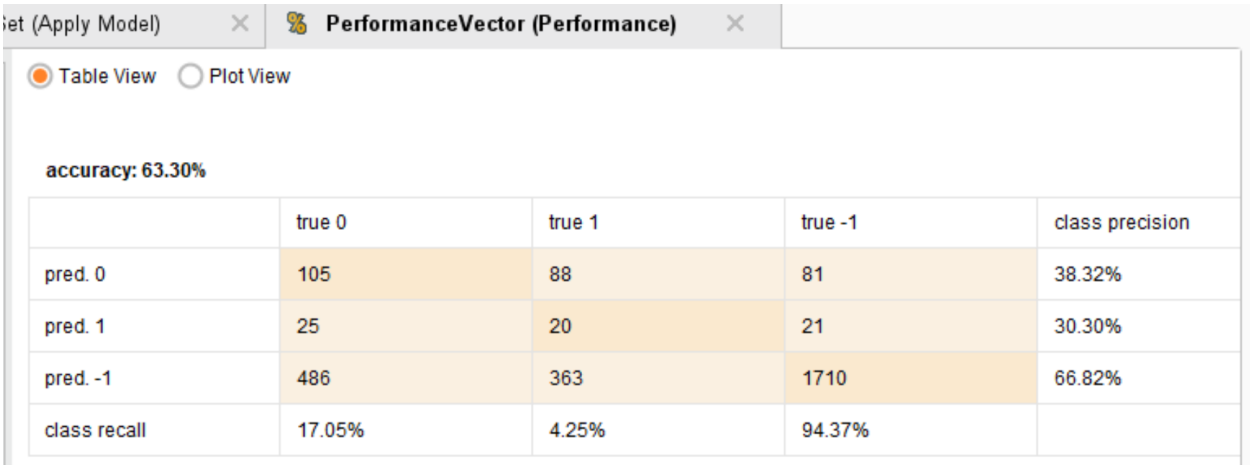



Process
Process
90%
Process
Deep Learning
Retrieve Tweets_se...
Subprocess
Apply Model
Performance
Parameters
Deep Learning
use regularization
l1 strength
0.1
l2 strength
0.1
optimization meth...
Stochastic G...
backpropagation
Standard
infer input shape
log each epoch
use local random seed

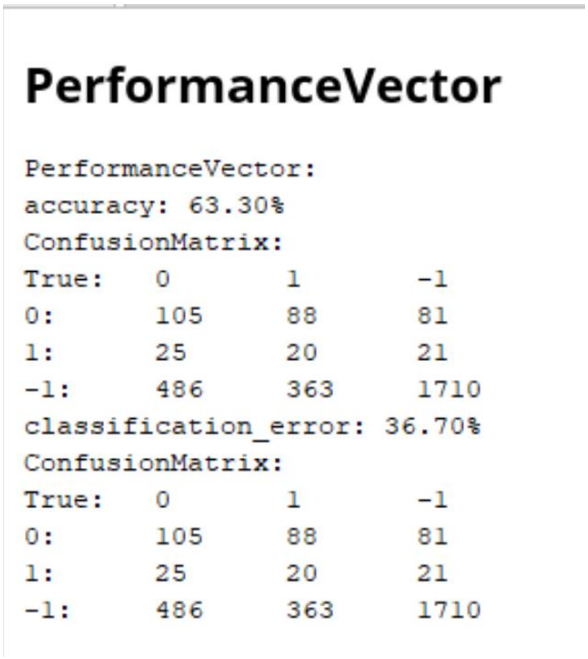

Process
Process
Subprocess
Subprocess
Select Attributes
Set Role
Split Data
Parameters
Set Role
attribute name
sentiment
target role
label
set additional roles
Edit List (0)...


Process
Process
Subprocess
Subprocess
in
in
out
out
out
Select Attributes
exa
exa
ori
Set Role
exa
exa
ori
Split Data
exa
par
par
par
Edit Parameter List: partitions
Edit Parameter List: partitions
The partitions that should be created.
ratio
0.8
0.2


Process
Process
Deep Learning
Add Fully-Connected Layer
Add Fully-Connecte...
Add Fully-Connecte...
Add Fully-Connecte...
lay
Parameters
Add Fully-Connected Layer
number of neurons
16
activation function
ReLU (Rectifie...
layer name
use dropout
overwrite networks weight initialization
overwrite networks bias initialization



	true 0	true 1	true -1	class precision
pred. 0	105	88	81	38.32%
pred. 1	25	20	21	30.30%
pred. -1	486	363	1710	66.82%
class recall	17.05%	4.25%	94.37%	



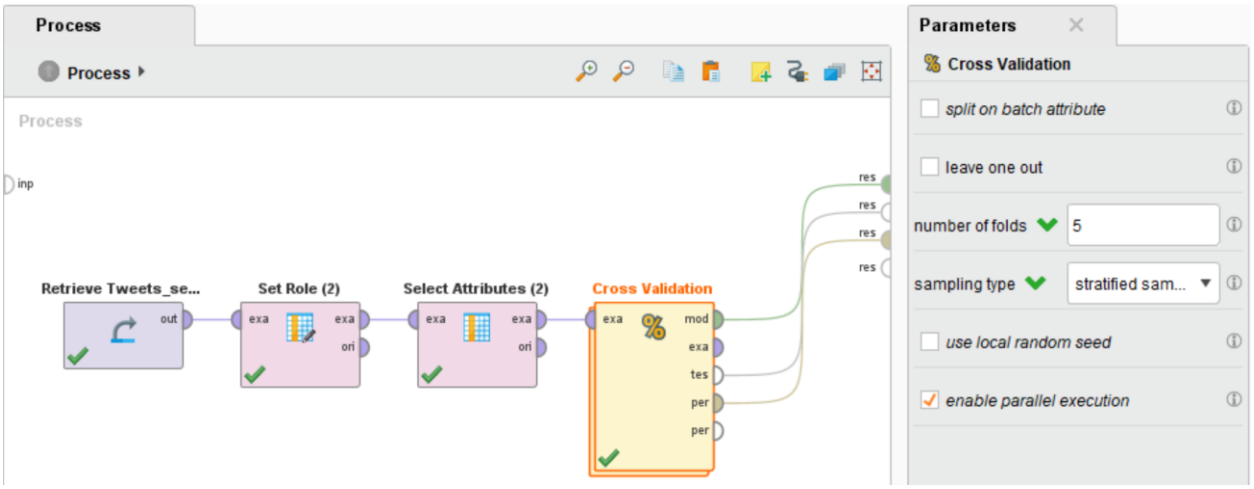
PerformanceVector

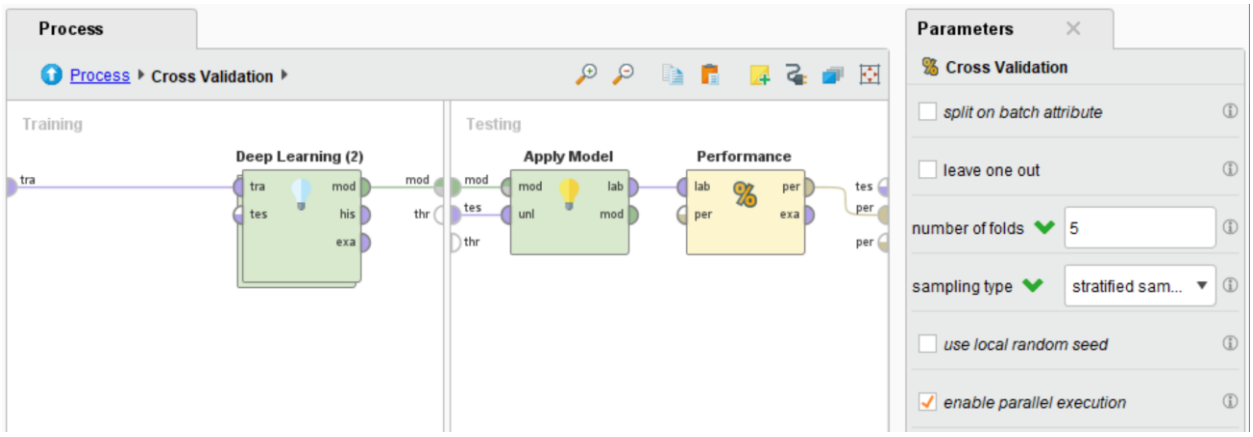
```
PerformanceVector:
accuracy: 63.30%
ConfusionMatrix:
True:    0       1       -1
0:       105     88      81
1:       25      20      21
-1:      486     363     1710
classification_error: 36.70%
ConfusionMatrix:
True:    0       1       -1
0:       105     88      81
1:       25      20      21
-1:      486     363     1710
```

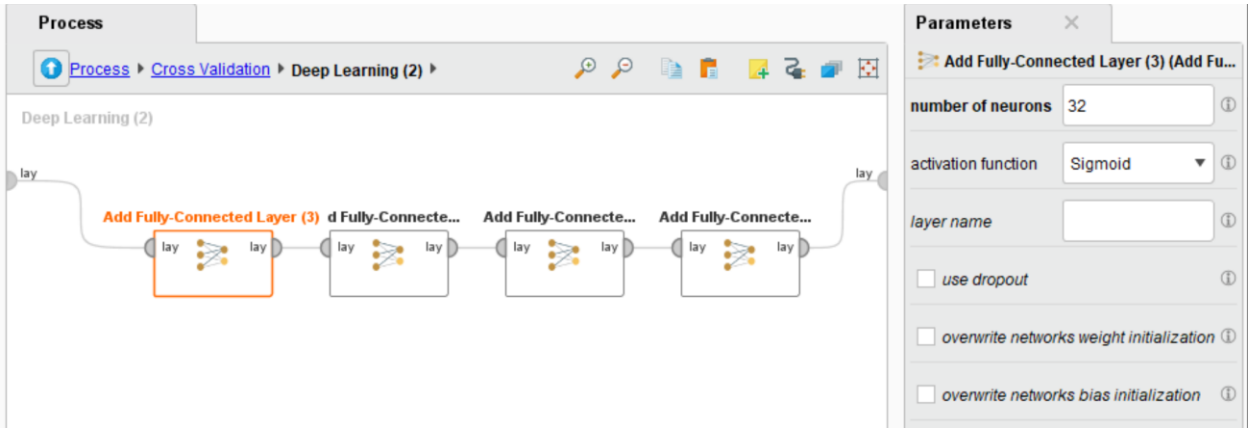
4. **DL: Fully Connected Layers with Cross Validation: Best Performance (Accuracy: 64.64)**

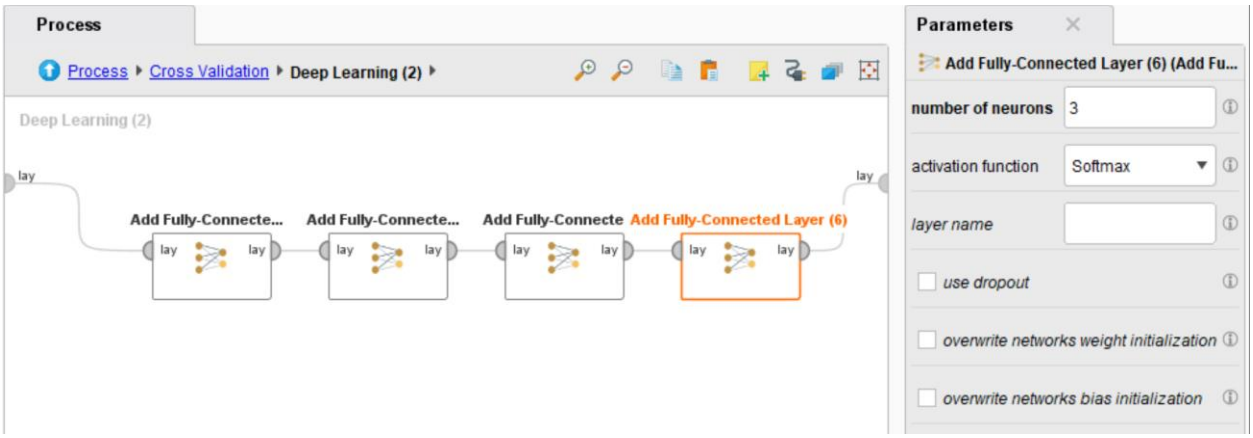
   Process Design:

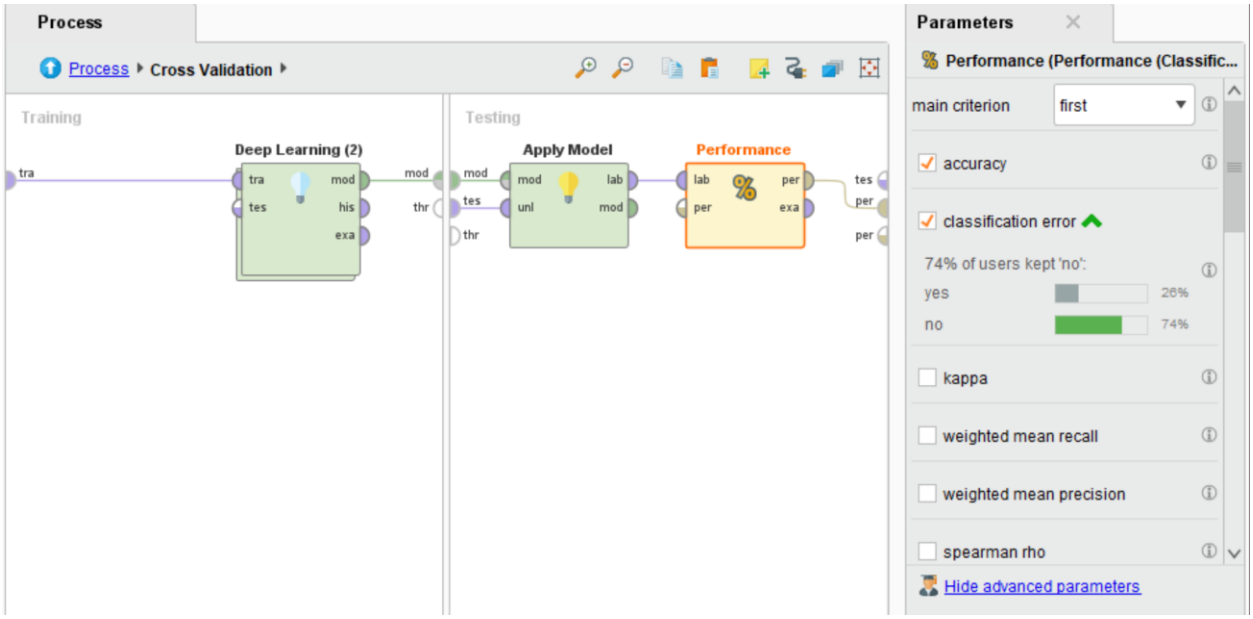
   In addition to above described process, this involves adding cross-validation to the sequence. We applied this validation by splitting the data as 80% training and rest 20% as test data. We applied 5 iteration to get the best result as illustrated.

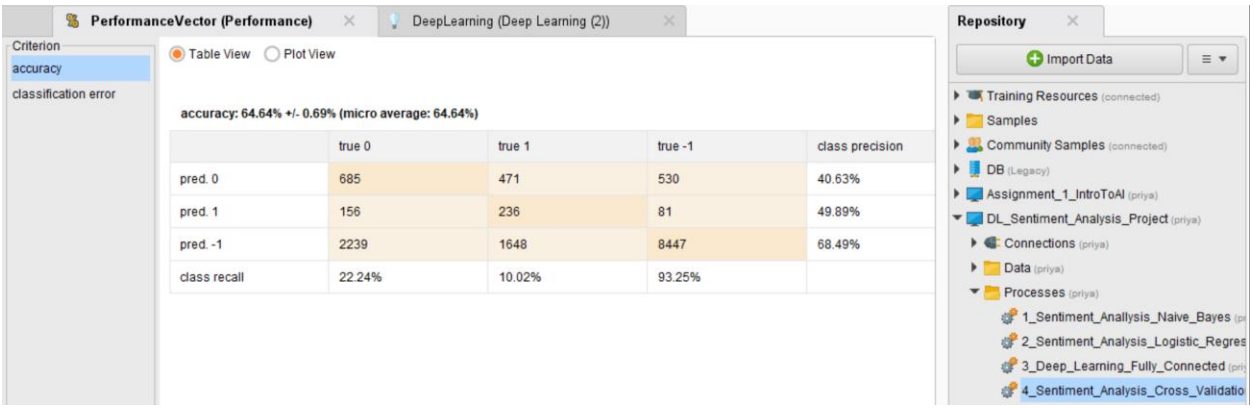

Process
Process
Process
inp
Retrieve Tweets_se...
Set Role (2)
Select Attributes (2)
Cross Validation
out
exa
ori
mod
tes
per
res
Parameters
Cross Validation
split on batch attribute
leave one out
number of folds
5
sampling type
stratified sam...
use local random seed
enable parallel execution

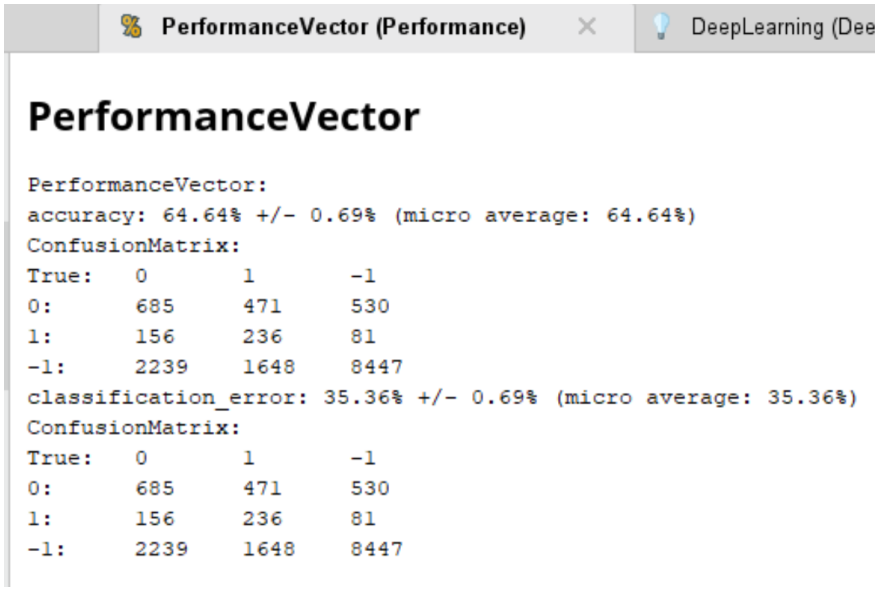

Process
Process
Cross Validation
Training
Testing
Deep Learning (2)
Apply Model
Performance
tra
mod
tes
thr
his
unl
lab
per
exa
Parameters
Cross Validation
split on batch attribute
leave one out
number of folds
5
sampling type
stratified sam...
use local random seed
enable parallel execution


Process
Process
Cross Validation
Deep Learning (2)
Deep Learning (2)
lay
Add Fully-Connected Layer (3)
d Fully-Connecte...
Add Fully-Connecte...
Add Fully-Connecte...
Parameters
Add Fully-Connected Layer (3) (Add Fu...
number of neurons
32
activation function
Sigmoid
layer name
use dropout
overwrite networks weight initialization
overwrite networks bias initialization


Process
Process ▸ Cross Validation ▸ Deep Learning (2) ▸
Deep Learning (2)
lay
Add Fully-Connecte...
Add Fully-Connecte...
Add Fully-Connecte
Add Fully-Connected Layer (6)
Parameters
Add Fully-Connected Layer (6) (Add Fu...
number of neurons
3
activation function
Softmax
layer name
use dropout
overwrite networks weight initialization
overwrite networks bias initialization


Process
Process ▸ Cross Validation ▸
Training
Testing
Deep Learning (2)
Apply Model
Performance
tra
mod
tes
his
exa
lab
unl
per
thr
Parameters
Performance (Performance (Classific...
main criterion
first
accuracy
classification error
74% of users kept 'no':
yes
26%
no
74%
kappa
weighted mean recall
weighted mean precision
spearman rho
Hide advanced parameters


PerformanceVector (Performance)
DeepLearning (Deep Learning (2))
Criterion
accuracy
classification error
Table View
Plot View
accuracy: 64.64% +/- 0.69% (micro average: 64.64%)
true 0
true 1
true -1
class precision
pred. 0
685
471
530
40.63%
pred. 1
156
236
81
49.89%
pred. -1
2239
1648
8447
68.49%
class recall
22.24%
10.02%
93.25%
Repository
Import Data
Training Resources (connected)
Samples
Community Samples (connected)
DB (Legacy)
Assignment_1_IntroToAI (priya)
DL_Sentiment_Analysis_Project (priya)
Connections (priya)
Data (priya)
Processes (priya)
1_Sentiment_Anallysis_Naive_Bayes
2_Sentiment_Analysis_Logistic_Regres
3_Deep_Learning_Fully_Connected
4_Sentiment_Analysis_Cross_Validatio


PerformanceVector (Performance)
DeepLearning (Dee
PerformanceVector
PerformanceVector:
accuracy: 64.64% +/- 0.69% (micro average: 64.64%)
ConfusionMatrix:
True: 0 1 -1
0: 685 471 530
1: 156 236 81
-1: 2239 1648 8447
classification_error: 35.36% +/- 0.69% (micro average: 35.36%)
ConfusionMatrix:
True: 0 1 -1
0: 685 471 530
1: 156 236 81
-1: 2239 1648 8447