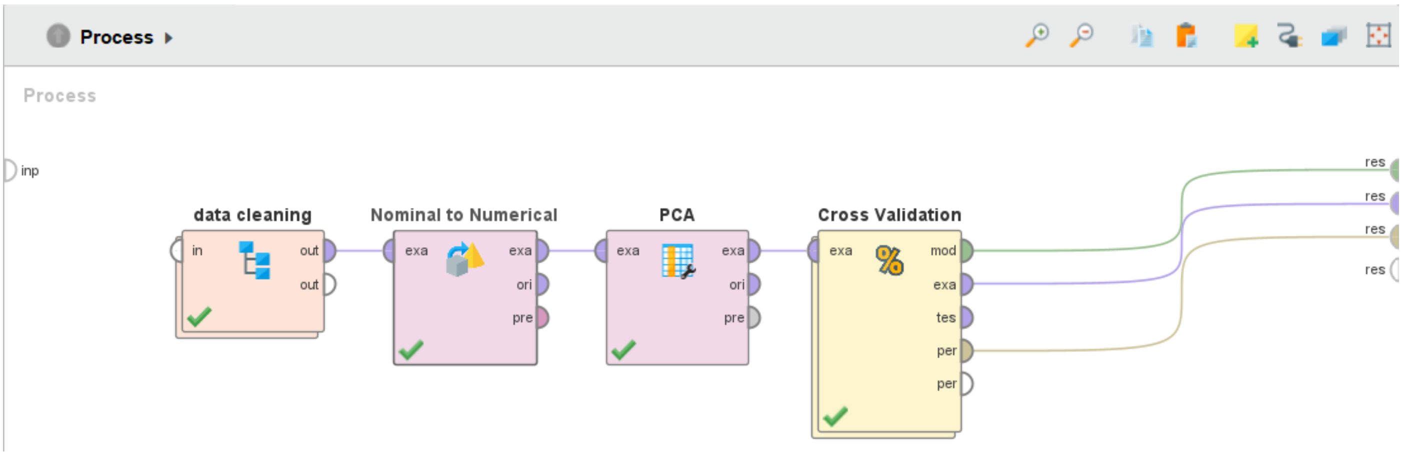
Titanic survival Prediction

Name: Jahnvi Rameshbhai Patel

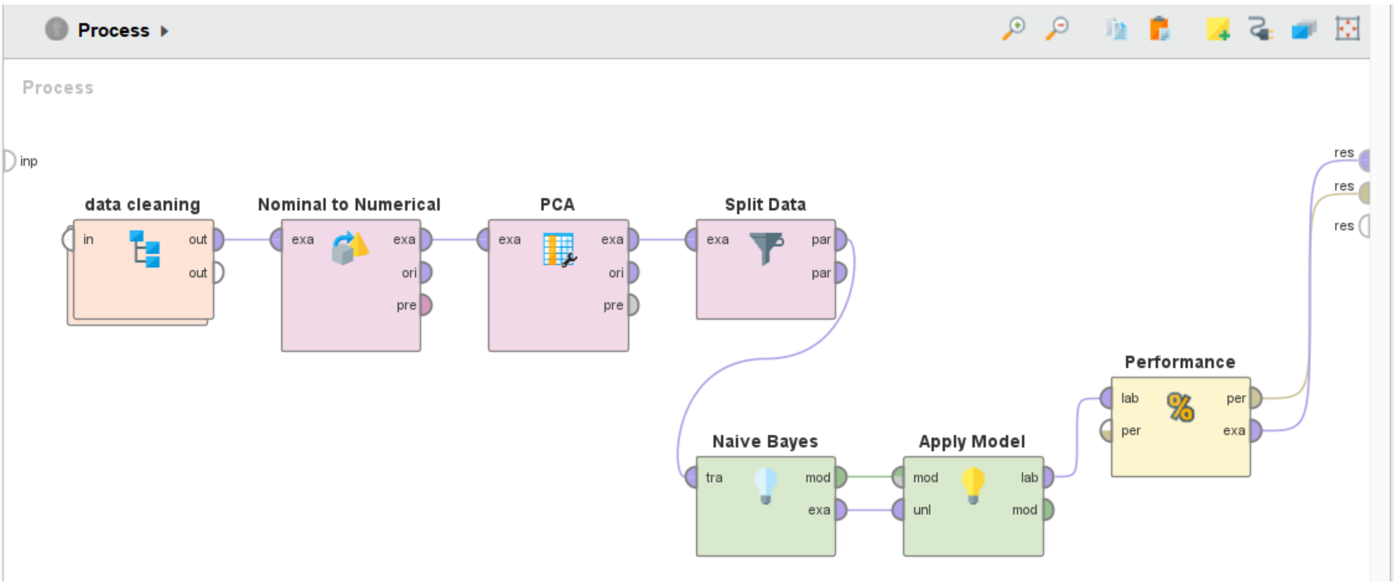
Model: Decision Tree (66.36%)



accuracy: 66.36% +/- 2.89% (micro average: 66.37%)

	true false	true true	class precision
pred. false	529	279	65.47%
pred. true	20	61	75.31%
class recall	96.36%	17.94%	

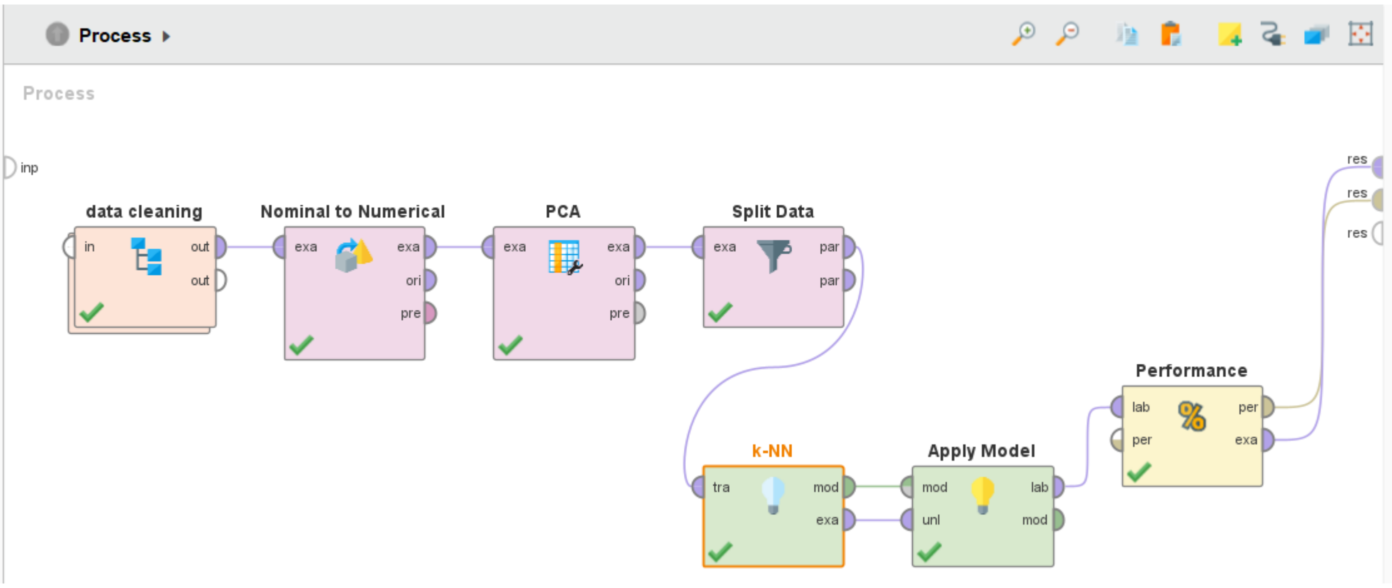
Model: Naïve Bayes (65.10%)



accuracy: 65.10%

	true false	true true	class precision
pred. false	317	174	64.56%
pred. true	12	30	71.43%
class recall	96.35%	14.71%	

Model: KNN (77.67%)



accuracy: 77.67%

	true false	true true	class precision
pred. false	290	80	78.38%
pred. true	39	124	76.07%
class recall	88.15%	60.78%	

Hence, comparing all three models I decided to use KNN model for Titanic dataset.

Data Cleaning

Missing value

Filter (12 / 12 attributes):

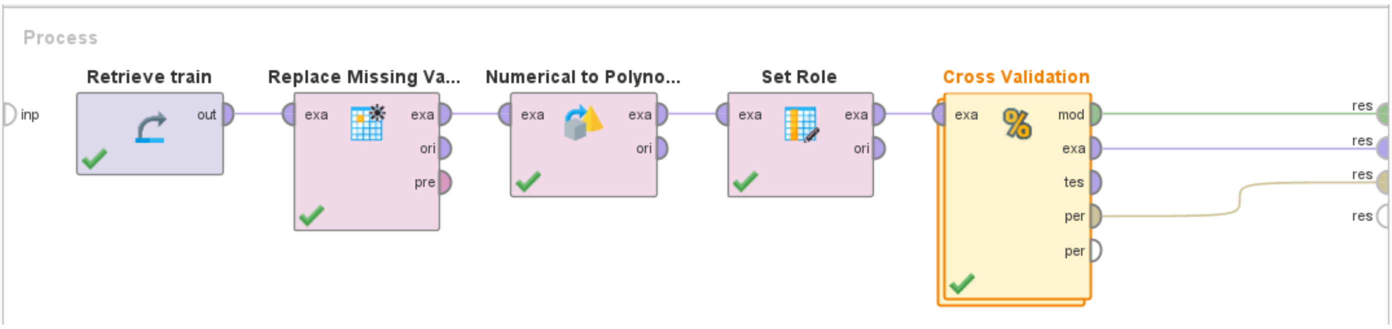
Name	Type	Missing	Statistics		
Name	Nominal	0	Least: van Melk [...] lemon (1)	Most: Abbing, Mr. Anthony (1)	Values: Abbing, Mr. Anthony (1), Abbott, [...]
Sex	Integer	0	Min: 0	Max: 1	Average: 0.648
Age	Real	177	Min: 0.420	Max: 80	Average: 29.699
SibSp	Integer	0	Min: 0	Max: 8	Average: 0.523
Parch	Integer	0	Min: 0	Max: 6	Average: 0.382
Ticket	Nominal	0	Least: W/C 14208 (1)	Most: 1601 (7)	Values: 1601 (7), 347082 (7), ...[679 more]
Fare	Real	0	Min: 0	Max: 512.329	Average: 32.204
Cabin	Nominal	687	Least: T (1)	Most: B96 B98 (4)	Values: B96 B98 (4), C23 C25 C27 (4), ...[14
Embarked	Nominal	2	Least: Q (77)	Most: S (644)	Values: S (644), C (168), ...[1 more]

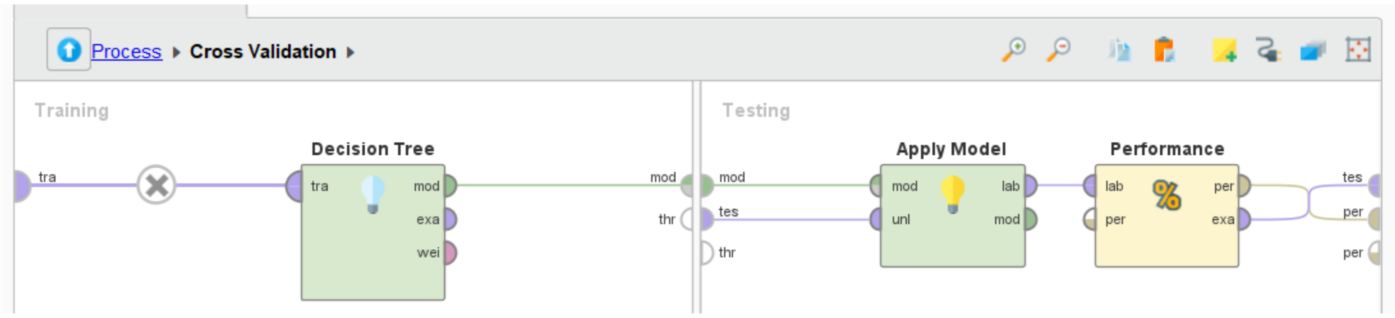
Since the "**Cabin**" feature in the Titanic dataset has a large number of missing values (687 out of 891), it may be difficult to use this feature effectively in a classification analysis. In general, it is best to avoid using features with a large number of missing values as this can lead to biased or unreliable results.

If you have a large number of missing values, as is the case with the "Age" column in the Titanic dataset (177 out of 891), then simply ignoring the missing values could lead to a loss of information and potentially bias your analysis.

One approach that could be considered is to use a machine learning algorithm that can handle missing values, such as decision trees or random forests. These algorithms can automatically handle missing values by using other available features to predict the missing values.

Model: Decision tree





Open in Turbo Prep Auto Model

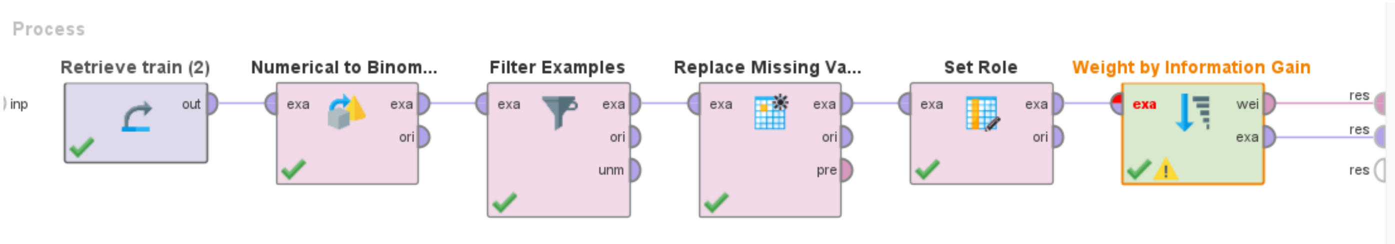
Filter (891 / 891 examples): all

Row No.	Age	PassengerId	Survived	Pclass	Name	Sex	SibSp	Parch	Ticket	Fare
3	26	3	1	3	Heikkinen, Mis...	0	0	0	STON/O2. 31...	7.925
4	35	4	1	1	Futrelle, Mrs. J...	0	1	0	113803	53.100
5	35	5	0	3	Allen, Mr. Willia...	1	0	0	373450	8.050
6	29.699	6	0	3	Moran, Mr. Ja...	1	0	0	330877	8.458
7	54	7	0	1	McCarthy, Mr. ...	1	0	0	17463	51.862
8	2	8	0	3	Palsson, Mast...	1	3	1	349909	21.075
9	27	9	1	3	Johnson, Mrs. ...	0	0	2	347742	11.133
10	14	10	1	2	Nasser, Mrs. ...	0	1	0	237736	30.071
11	4	11	1	3	Sandstrom, Mi...	0	1	1	PP 9549	16.700
12	58	12	1	1	Bonnell, Miss. ...	0	0	0	113783	26.550
13	20	13	0	3	Saundercock, ...	1	0	0	A/5. 2151	8.050
14	39	14	0	3	Andersson, Mr...	1	1	5	347082	31.275
15	14	15	0	3	Vestrom, Miss....	0	0	0	350406	7.854
16	55	16	1	2	Hewlett, Mrs. (...	0	0	0	248706	16
17	2	17	0	3	Rice, Master. ...	1	4	1	382652	29.125
18	29.699	18	1	2	Williams, Mr. ...	1	0	0	244373	13
19	31	19	0	3	Vander Planke...	0	1	0	345763	18
20	29.699	20	1	3	Masselmani, M...	0	0	0	2649	7.225

ExampleSet (891 examples, 1 special attribute, 11 regular attributes)

Model replaced the missing values with the average of the Age.

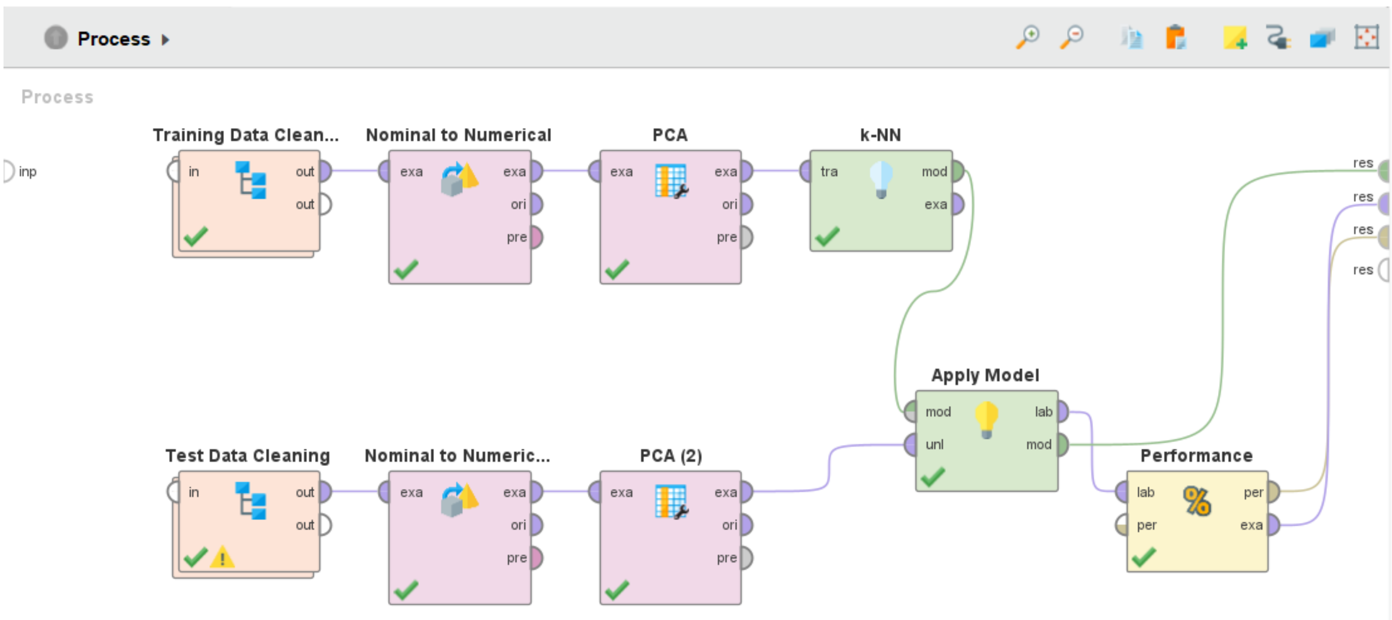
By weight



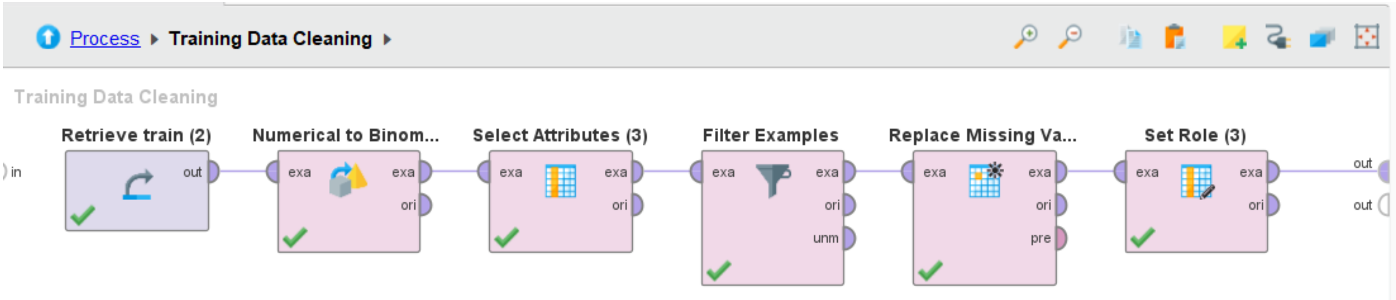
attribute	weight ↓
Name	0.960
Ticket	0.825
Cabin	0.732
Sex	0.216
Pclass	0.075
Fare	0.068
Embarked	0.021
Age	0.017
Parch	0.016
SibSp	0.010
PassengerId	0.003

After considering the table shown in the left, I found "Name" & "Ticket" to be irrelevant even though they have the highest weightage. "Cabin" has the 3rd highest weightage, but the missing value is way too high to consider it in the classification.

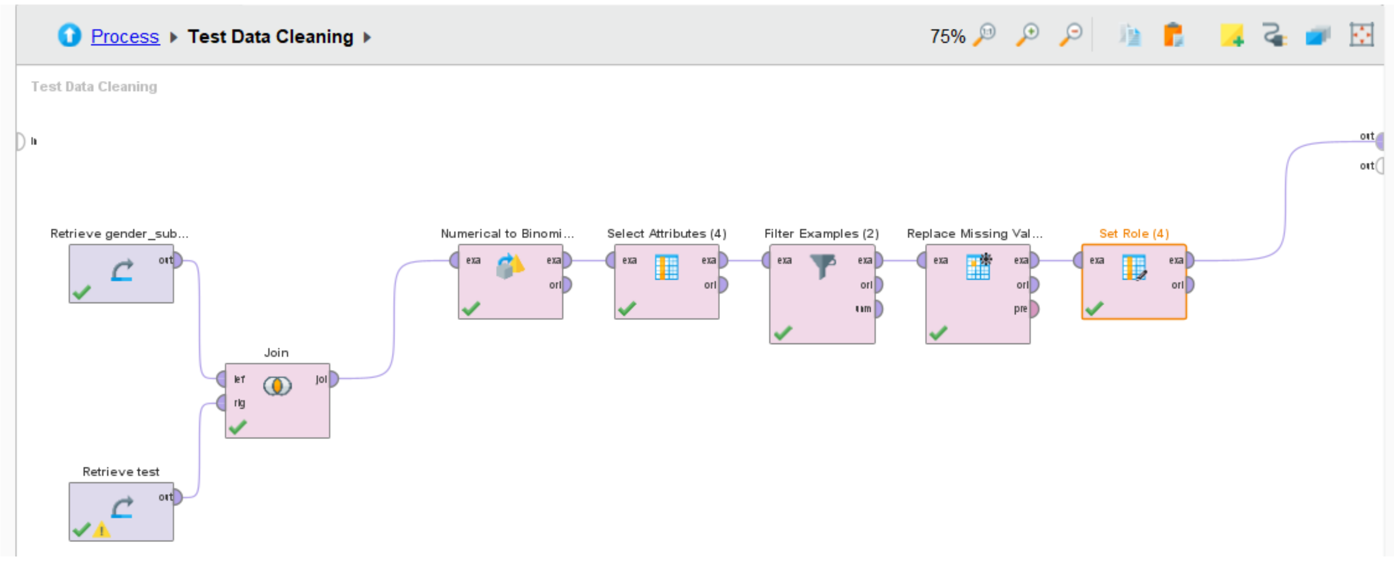
Final Model: KNN



Training Subprocess:



Test Subprocess:



Accuracy:

accuracy: 62.59%

	true false	true true	class precision
pred. false	200	91	68.73%
pred. true	65	61	48.41%
class recall	75.47%	40.13%	

F-measure:

f_measure: 43.88% (positive class: true)

	true false	true true	class precision
pred. false	200	91	68.73%
pred. true	65	61	48.41%
class recall	75.47%	40.13%	

Example set:

Open in Turbo Prep Auto Model Filter (417 / 417 examples): all

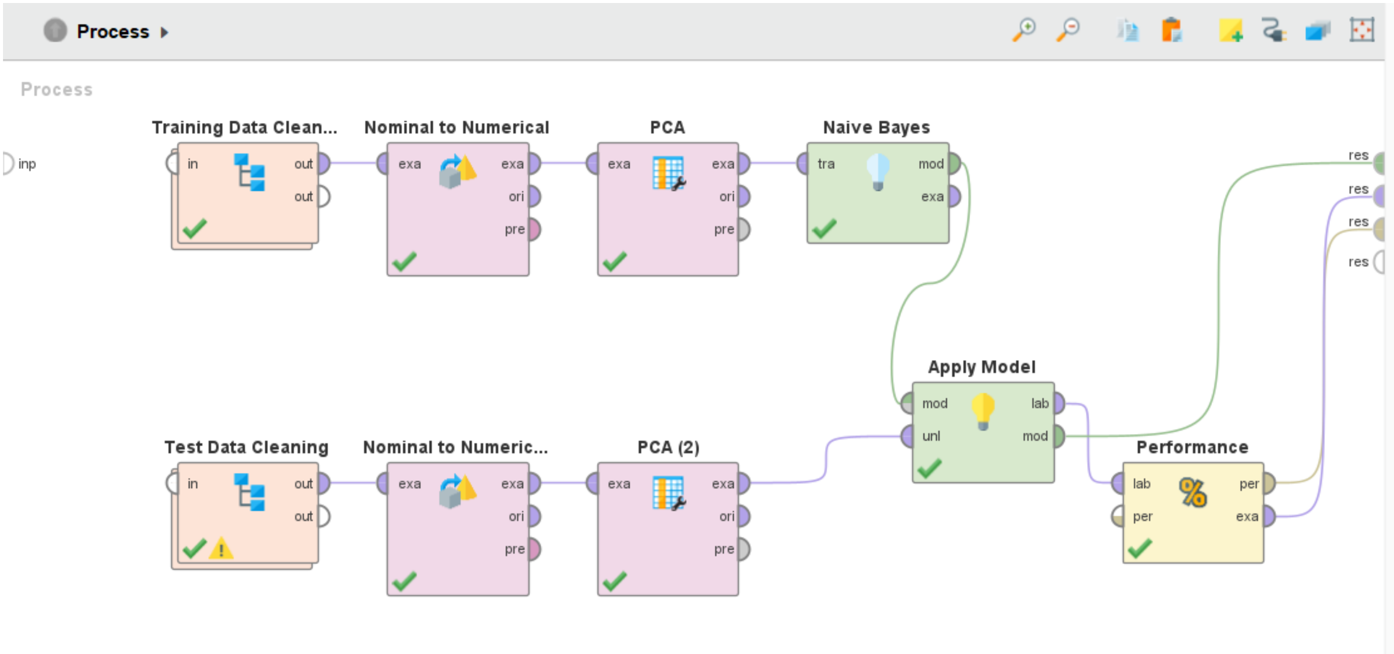
Row No.	PassengerId	Survived	prediction(S...	confidence(f...	confidence(t...	pc_1	pc_2
1	892	false	false	1	0	-27.388	6.449
2	893	true	false	1	0	-27.245	18.963
3	894	false	false	0.586	0.414	-23.400	33.736
4	895	false	false	0.795	0.205	-27.137	-1.096
5	896	true	false	1	0	-23.907	-6.366
6	897	false	true	0.402	0.598	-27.582	-14.098
7	898	true	false	1	0	-27.932	1.973
8	899	false	false	0.621	0.379	-6.930	-3.649
9	900	true	false	1	0	-29.255	-9.956
10	901	false	false	0.593	0.407	-12.161	-8.277
11	902	false	false	1	0	-27.656	2.134
12	903	false	false	0.607	0.393	-8.366	16.538
13	904	true	false	0.563	0.437	45.955	-10.752
14	905	false	false	1.000	0	-7.061	33.464
15	906	true	true	0.395	0.605	26.783	14.805
16	907	true	false	0.611	0.389	-8.353	-5.544
17	908	false	false	1	0	-22.833	6.616
18	909	false	false	0.805	0.195	-29.030	-6.960

ExampleSet (417 examples, 5 special attributes, 2 regular attributes)

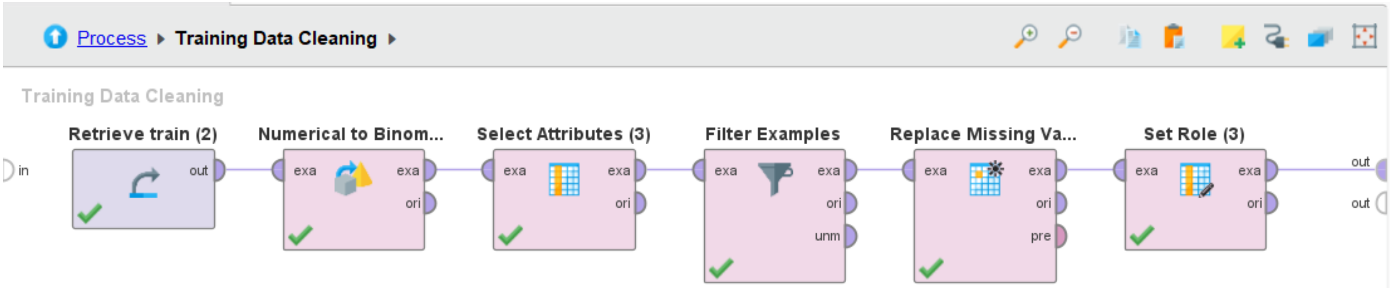
A model with 62.59% accuracy and 43.88% F-measure can be considered to have poor performance. The model may not be suitable for making accurate predictions for the given data set. It may be necessary to try other machine learning algorithms or techniques, or perform additional data preprocessing, feature selection, or hyperparameter tuning to improve the model's performance.

Since the accuracy is low I decided to try another model with training & test data.

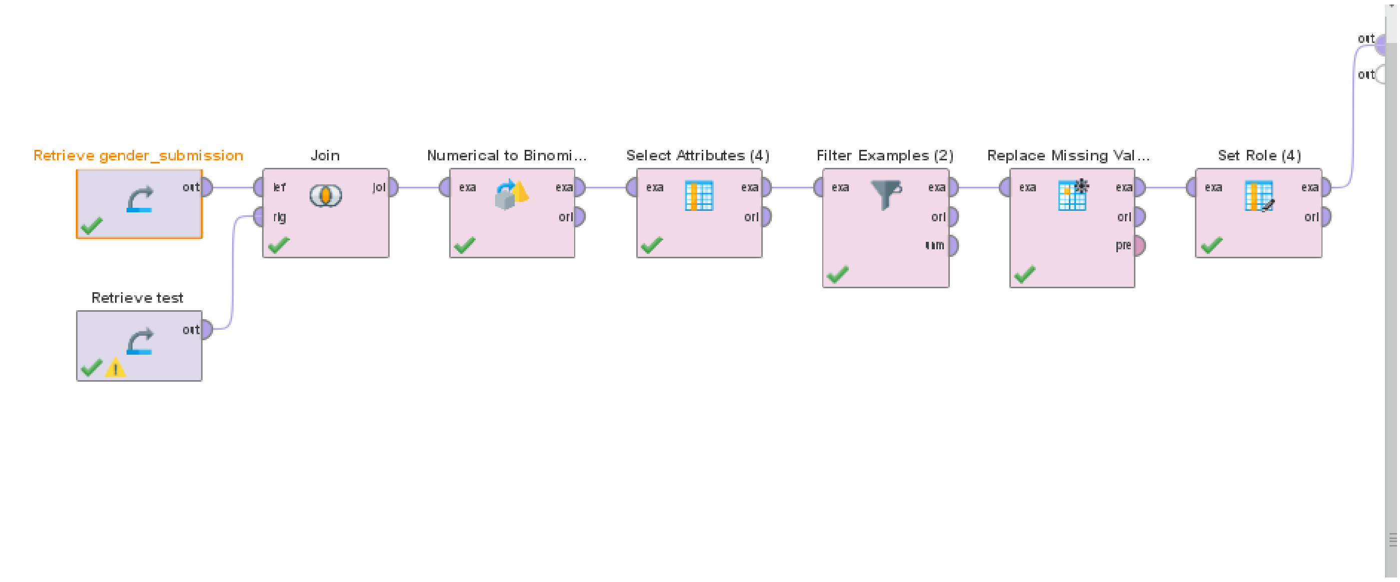
Model: Naïve Bayes



Training Subprocess:



Test Subprocess:



Example Set:

Open in Turbo Prep Auto Model Filter (417 / 417 examples): all

Row No.	PassengerId	Survived	prediction(S...	confidence(f...	confidence(t...	pc_1	pc_2
1	892	false	false	0.811	0.189	-27.388	6.449
2	893	true	false	0.816	0.184	-27.245	18.963
3	894	false	false	0.794	0.206	-23.400	33.736
4	895	false	false	0.797	0.203	-27.137	-1.096
5	896	true	false	0.783	0.217	-23.907	-6.366
6	897	false	false	0.743	0.257	-27.582	-14.098
7	898	true	false	0.803	0.197	-27.932	1.973
8	899	false	false	0.788	0.212	-6.930	-3.649
9	900	true	false	0.761	0.239	-29.255	-9.956
10	901	false	false	0.777	0.223	-12.161	-8.277
11	902	false	false	0.804	0.196	-27.656	2.134
12	903	false	false	0.818	0.182	-8.366	16.538
13	904	true	true	0.405	0.595	45.955	-10.752
14	905	false	false	0.790	0.210	-7.061	33.464
15	906	true	false	0.683	0.317	26.783	14.805
16	907	true	false	0.784	0.216	-8.353	-5.544
17	908	false	false	0.816	0.184	-22.833	6.616
18	909	false	false	0.775	0.225	-29.030	-6.960

ExampleSet (417 examples, 5 special attributes, 2 regular attributes)

Accuracy:

accuracy: 64.75%

	true false	true true	class precision
pred. false	245	127	65.86%
pred. true	20	25	55.56%
class recall	92.45%	16.45%	

F-measure:

f_measure: 25.38% (positive class: true)

	true false	true true	class precision
pred. false	245	127	65.86%
pred. true	20	25	55.56%
class recall	92.45%	16.45%	

Based on the given metrics, the KNN model has higher F-measure (43.88%) compared to the Naive Bayes model (25.38%), which means that the KNN model has a better balance between precision and recall. However, the Naive Bayes model has higher accuracy (64.75%) compared to the KNN model (62.59%), which means that the Naive Bayes model correctly predicted a higher proportion of cases in the test set.

It is important to note that the choice of which model to use depends on the specific problem being addressed and the relative costs of different types of errors. If the problem requires a higher level of precision, the KNN model may be more appropriate, while if the problem requires a higher level of recall, the Naive Bayes model may be more appropriate. Additionally, it may be necessary to analyze other metrics and consider other factors such as computational complexity, interpretability, and ease of implementation when selecting a model.