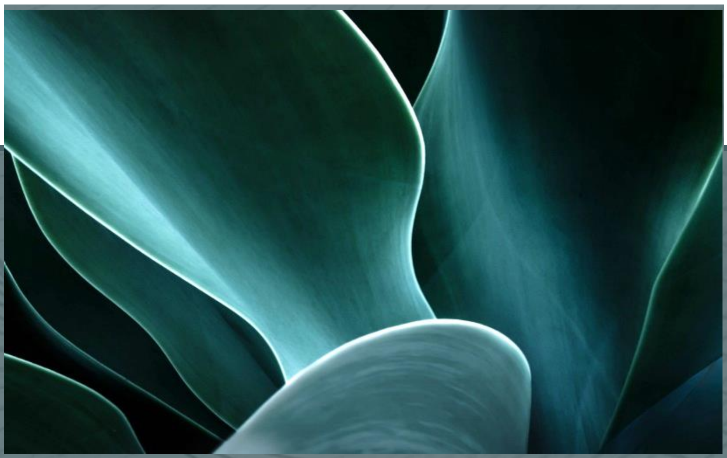
Project Report

Predicting Customer Churn

Shayan Lodi - 27002
Faariha Atiq – 26993
Uzair Arif – 27184
Kanza Rehan - 27182

Business Problem

One of the primary objectives for any business is to mitigate customer churn, as it directly impacts revenue and incurs additional costs in acquiring new customers. As a data scientist at a bank, the task at hand is to leverage historical data to forecast whether a customer will churn in the upcoming six months. This predictive model will enable the bank to strategize proactive engagement with customers, ensuring timely interventions to retain them effectively.



About the data

Variables

Each row in the dataset is uniquely identified by an ID. The age of each customer is recorded, along with their gender, which can be either male or female. The yearly income of the customer is also recorded. Additionally, the average quarterly balance of the customer's account and the length of time they have been associated with the bank, referred to as "Vintage," are included. The dataset indicates whether the customer has conducted any transactions in the past three months. Furthermore, it records the number of product holdings each customer has with the bank and whether they possess a credit card. The credit category of each customer, determined by their credit score, is also noted. Finally, the dataset specifies whether each customer is predicted to churn within the next six months.

Preprocessing Techniques

Normalisation

Normalization is a technique used to scale numerical features to a similar range. This ensures that features with different scales do not disproportionately influence the model's training process.

Discretization

Discretization involves converting continuous numerical features into categorical features by dividing them into intervals or bins. This can help simplify the model and make it more interpretable, especially when dealing with algorithms that require categorical input.

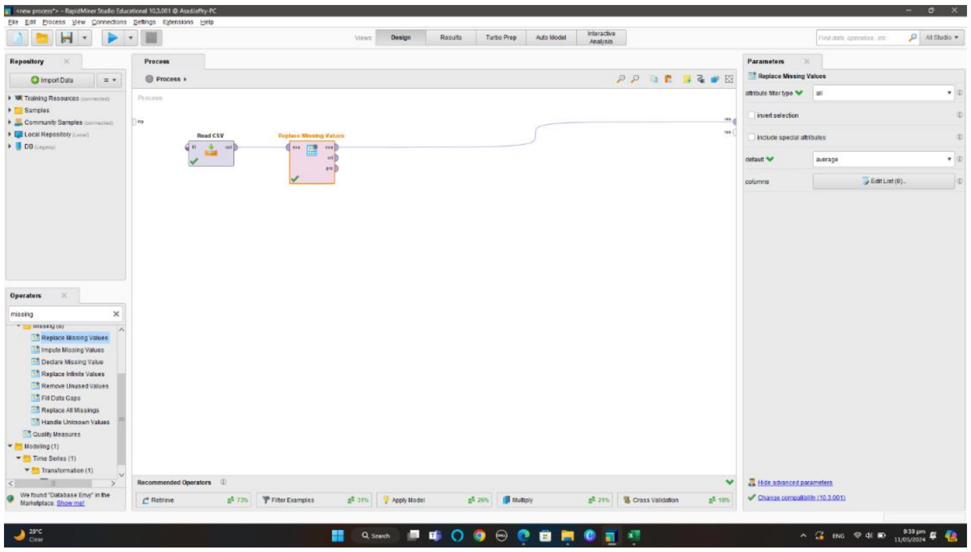
Feature Encoding

Feature encoding is the process of converting categorical features into a numerical representation that machine learning algorithms can understand. Feature encoding ensures that categorical data can be effectively utilized in the modeling process.

Cleaning the data

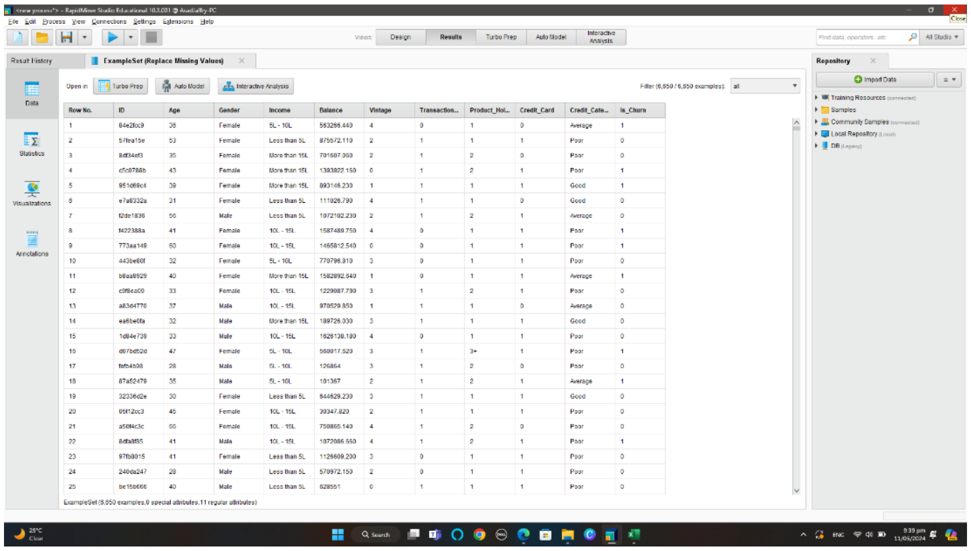
We start by first cleaning our data. Cleaning data before applying any data science techniques is essential to ensure accuracy, completeness, consistency, relevance, and quality of the dataset. It involves removing any errors, handling missing values, standardizing formats, and eliminating irrelevant variables. In essence, data cleaning is a critical preparatory step that enhances the integrity and usefulness of the data for subsequent data science tasks.

To handle missing values, we used the Replace Missing Values operator and applied average value option for the missing values.



Replace Missing Values
attribute filter type: all
invert selection
include special attributes
default: average
columns: Edit List (0)...

The result is as follows:



<process*> – RapidMiner Studio Educational 10.3.001 @ AsadJaffry-PC
File Edit Process View Connections Settings Extensions Help
Views: Design Results Turbo Prep Auto Model Interactive Analysis
Result History
ExampleSet (Replace Missing Values)
Open in Turbo Prep Auto Model Interactive Analysis
Data
Statistics
Visualizations
Annotations
Repository
Import Data
Training Resources
Samples
Community Samples
Local Repository
DB

Filter (6,650 / 6,650 examples): all

Row No.	ID	Age	Gender	Income	Balance	Vintage	Transaction...	Product_Hol...	Credit_Card	Credit_Cate...	Is_Churn
1	84e2fcc9	36	Female	5L - 10L	563266.440	4	0	1	0	Average	1
2	57fea15e	53	Female	Less than 5L	875572.110	2	1	1	1	Poor	0
3	8df34ef3	35	Female	More than 15L	701607.060	2	1	2	0	Poor	0
4	c5c0788b	43	Female	More than 15L	1393922.160	0	1	2	1	Poor	1
5	951d69c4	39	Female	More than 15L	893146.230	1	1	1	1	Good	1
6	e7a8332a	31	Female	Less than 5L	111026.790	4	1	1	0	Good	0
7	f2de1836	56	Male	Less than 5L	1072102.230	2	1	2	1	Average	0
8	f422388a	41	Female	10L - 15L	1587489.750	4	0	1	1	Poor	1
9	773aa149	60	Female	10L - 15L	1465812.540	0	0	1	1	Poor	1
10	443be80f	32	Female	5L - 10L	770796.810	3	0	1	1	Poor	0
11	b8aa8929	40	Female	More than 15L	1582892.640	1	0	1	1	Average	1
12	c9f8ea09	33	Female	10L - 15L	1229087.790	3	1	2	1	Poor	0
13	a83d4770	37	Male	10L - 15L	970529.850	1	1	1	0	Average	0
14	ea6be0fa	32	Male	More than 15L	189726.030	3	1	1	1	Good	0
15	1d84e739	33	Male	10L - 15L	1626138.180	4	0	1	1	Poor	0
16	d07bd52d	47	Female	5L - 10L	560017.620	3	1	3+	1	Poor	1
17	fefb4b98	28	Male	5L - 10L	126864	3	1	2	0	Poor	0
18	87a52479	35	Male	5L - 10L	101367	2	1	2	1	Average	1
19	32336d2e	30	Female	Less than 5L	644629.230	3	1	1	1	Good	0
20	05f12cc3	45	Female	10L - 15L	30347.820	2	1	1	1	Poor	0
21	a50f4c3c	66	Female	10L - 15L	750865.140	4	1	2	0	Poor	0
22	8dfa8f85	41	Male	10L - 15L	1072086.560	4	1	2	1	Poor	1
23	97fb8015	41	Female	Less than 5L	1126609.200	3	0	1	1	Poor	0
24	240da247	28	Male	Less than 5L	570972.150	2	0	1	1	Poor	0
25	be15b666	40	Male	Less than 5L	628551	0	1	1	1	Poor	0

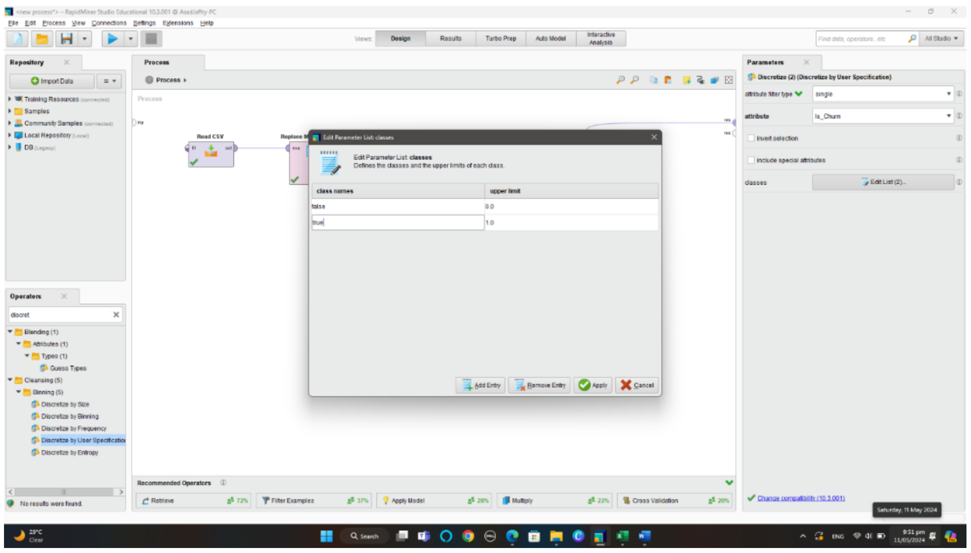
ExampleSet (6,650 examples,0 special attributes,11 regular attributes)

The target variable 'Is_Churn' is already normalized so we don't need to apply normalization.

Logistic Regression

Logistic regression is a statistical method used for binary classification tasks, predicting the probability of an outcome based on predictor variables. It's chosen for its interpretability, probabilistic outputs, computational efficiency, and flexibility in handling nonlinear relationships. Logistic regression provides valuable insights into binary outcomes and is widely utilized in data analysis and predictive modeling.

To prepare the data for logistic regression, we use the Discretize by User Specification operator. This involves categorizing the data into distinct classes based on predefined criteria. In this case, we assign the label 'false' to values of 0 and 'true' to values of 1. This process simplifies the data, making it suitable for logistic regression analysis.



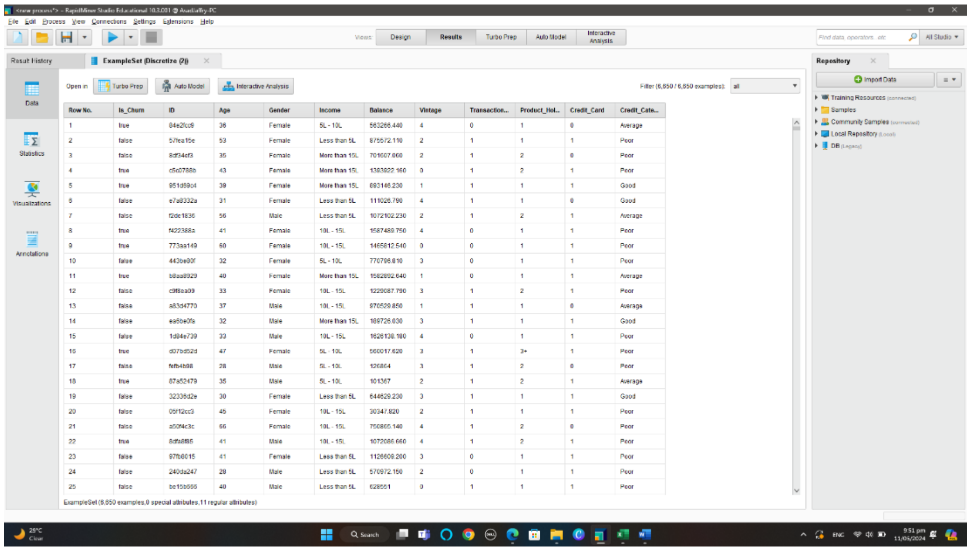
Edit Parameter List: classes
Defines the classes and the upper limits of each class.

class names	upper limit
false	0.0
true	1.0

Parameters
Discretize (2) (Discretize by User Specification)

attribute filter type	single
attribute	Is_Churn
invert selection	☐
include special attributes	☐
classes	Edit List (2)...

The result is as follows:



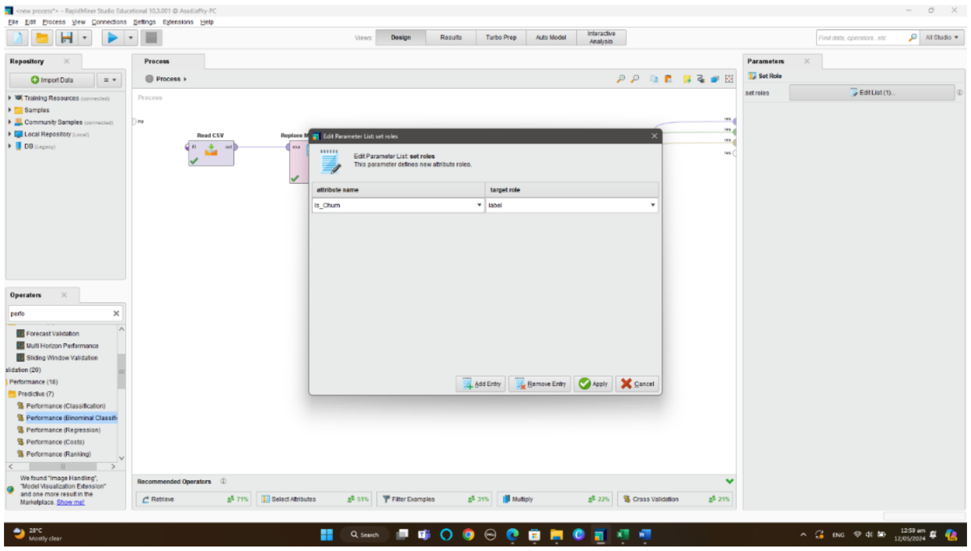
ExampleSet (Discretize (2))

Filter (6,650 / 6,650 examples): all

Row No.	Is_Churn	ID	Age	Gender	Income	Balance	Vintage	Transaction...	Product_Hol...	Credit_Card	Credit_Cate...
1	true	84e2fcc9	36	Female	5L - 10L	563266.440	4	0	1	0	Average
2	false	57fea15e	53	Female	Less than 5L	875572.110	2	1	1	1	Poor
3	false	8df34ef3	35	Female	More than 15L	701607.060	2	1	2	0	Poor
4	true	c5c0788b	43	Female	More than 15L	1393922.160	0	1	2	1	Poor
5	true	951d69c4	39	Female	More than 15L	893146.230	1	1	1	1	Good
6	false	e7a8332a	31	Female	Less than 5L	111026.790	4	1	1	0	Good
7	false	f2de1836	56	Male	Less than 5L	1072102.230	2	1	2	1	Average
8	true	f422388a	41	Female	10L - 15L	1587489.750	4	0	1	1	Poor
9	true	773aa149	60	Female	10L - 15L	1465812.540	0	0	1	1	Poor
10	false	443be80f	32	Female	5L - 10L	770796.810	3	0	1	1	Poor
11	true	b8aa8929	40	Female	More than 15L	1582892.640	1	0	1	1	Average
12	false	c9f8ea09	33	Female	10L - 15L	1229087.790	3	1	2	1	Poor
13	false	a83d4770	37	Male	10L - 15L	970529.850	1	1	1	0	Average
14	false	ea6be0fa	32	Male	More than 15L	189726.030	3	1	1	1	Good
15	false	1d84e739	33	Male	10L - 15L	1626138.180	4	0	1	1	Poor
16	true	d07bd52d	47	Female	5L - 10L	560017.620	3	1	3+	1	Poor
17	false	fefb4b98	28	Male	5L - 10L	126864	3	1	2	0	Poor
18	true	87a52479	35	Male	5L - 10L	101367	2	1	2	1	Average
19	false	32338d2e	30	Female	Less than 5L	644629.230	3	1	1	1	Good
20	false	05f12cc3	45	Female	10L - 15L	30347.820	2	1	1	1	Poor
21	false	a50f4c3c	66	Female	10L - 15L	750865.140	4	1	2	0	Poor
22	true	8dfa8f85	41	Male	10L - 15L	1072086.660	4	1	2	1	Poor
23	false	97fb8015	41	Female	Less than 5L	1126609.200	3	0	1	1	Poor
24	false	240da247	28	Male	Less than 5L	570972.150	2	0	1	1	Poor
25	false	be15b666	40	Male	Less than 5L	628651	0	1	1	1	Poor

ExampleSet (6,650 examples,0 special attributes,11 regular attributes)

Next, we use the Set Role operator to set ‘Is_Churn’ as ‘label’.

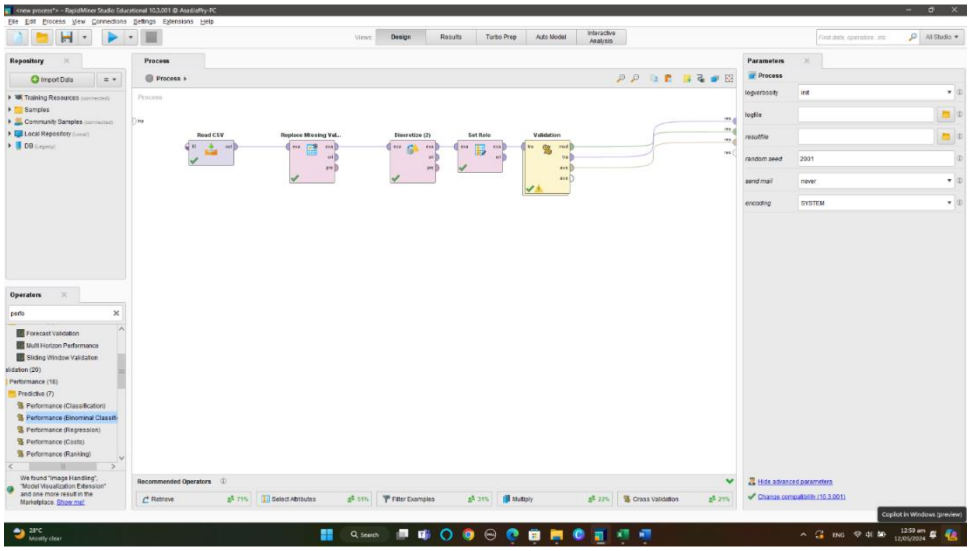


Edit Parameter List: set roles

Edit Parameter List: **set roles**
This parameter defines new attribute roles.

attribute name	target role
Is_Churn	label

We then use Split Validation to split our data into a 70:30 ratio using stratified sampling. Stratified sampling is a sampling technique used to ensure that the proportions of different classes or categories within the dataset are maintained in the sample. This is particularly useful when dealing with imbalanced datasets, where one class may be significantly more prevalent than others.


Parameters

Process

logverbosity: init

logfile:

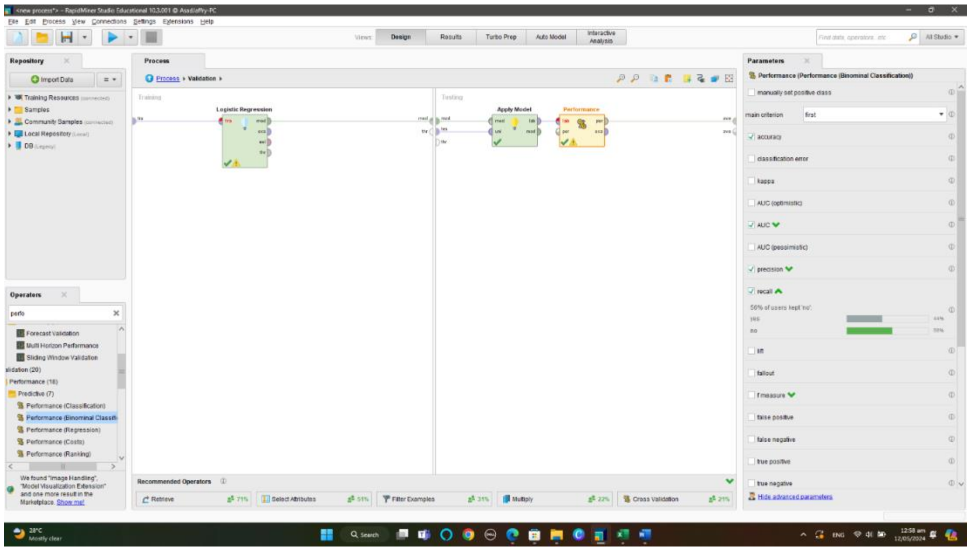
resultfile:

random seed: 2001

send mail: never

encoding: SYSTEM

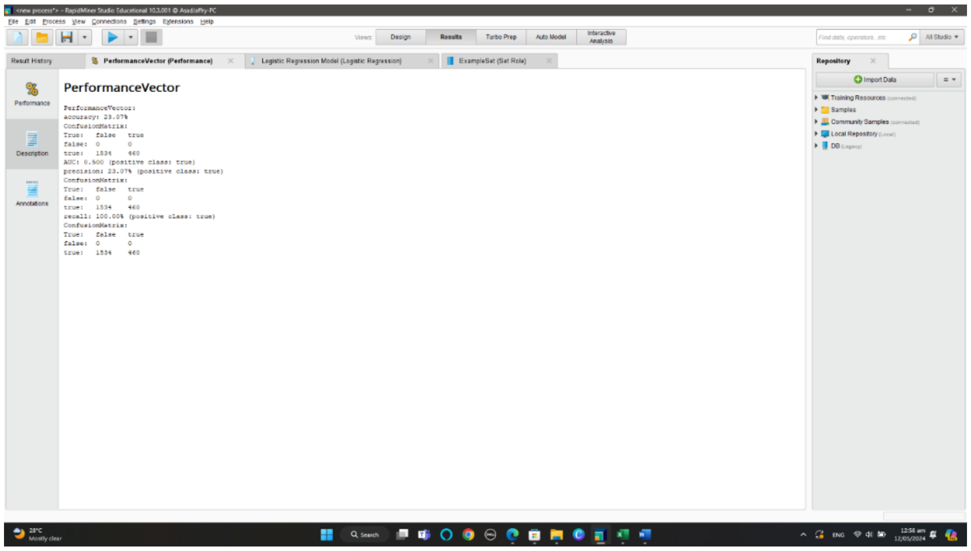
Double clicking the validation operator, we go to its subprocess. Here we add the Logistic Regression operator, Apply Model and Performance (Binomial Classification) operators. For performance, we choose accuracy, recall, precision, and AUC as shown below.


Parameters

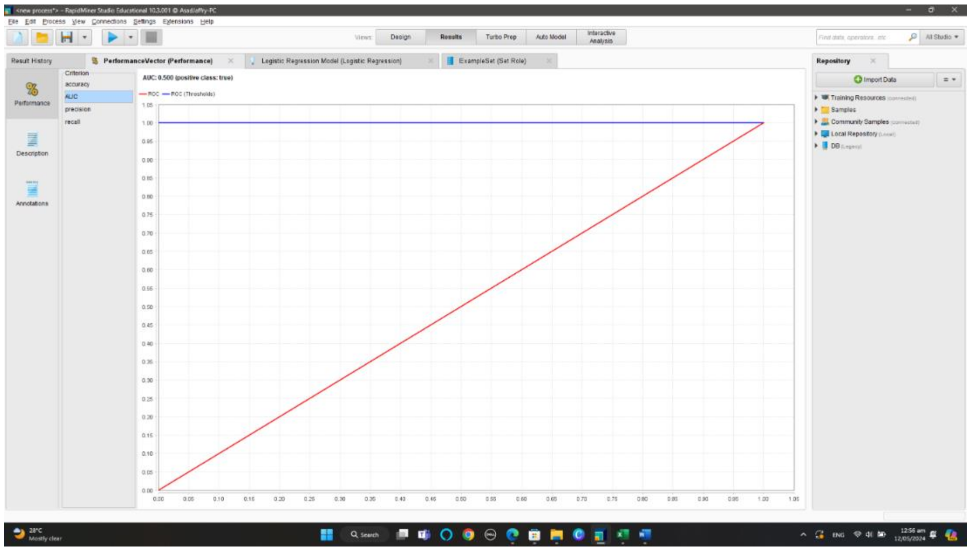
Performance (Performance (Binominal Classification))

- [ ] manually set positive class
- main criterion: first
- [✓] accuracy
- [ ] classification error
- [ ] kappa
- [ ] AUC (optimistic)
- [✓] AUC
- [ ] AUC (pessimistic)
- [✓] precision
- [✓] recall
- [ ] lift
- [ ] fallout
- [ ] f measure
- [ ] false positive
- [ ] false negative
- [ ] true positive
- [ ] true negative

The resultant Confusion Matrix and AUC curve is this:



```
PerformanceVector:
accuracy: 23.07%
ConfusionMatrix:
True:   false   true
false:  0       0
true:   1534    460
AUC: 0.500 (positive class: true)
precision: 23.07% (positive class: true)
ConfusionMatrix:
True:   false   true
false:  0       0
true:   1534    460
recall: 100.00% (positive class: true)
ConfusionMatrix:
True:   false   true
false:  0       0
true:   1534    460
```

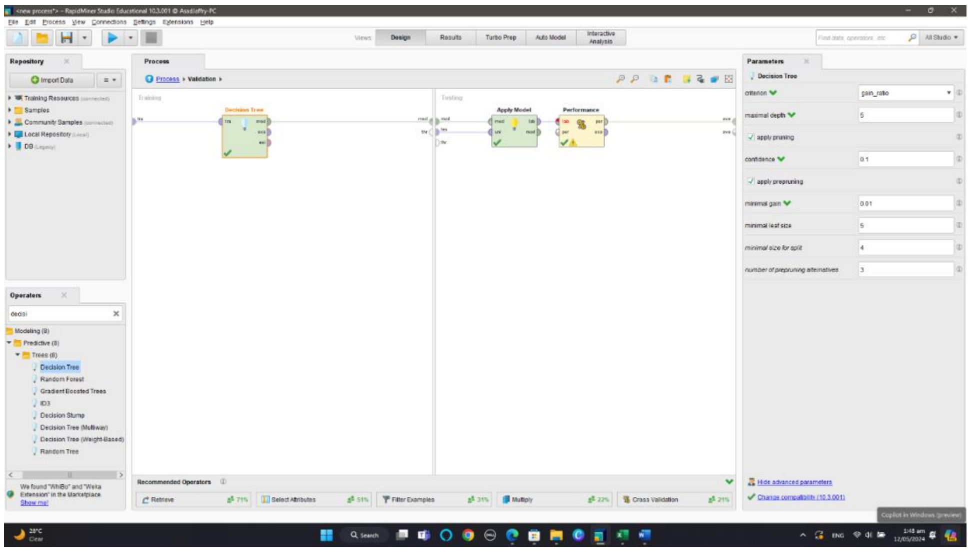


Accuracy comes out to be 23.07%, AUC = 0.5 (ideal), precision is 23.07%, and recall is 100%.

Decision Tree

A decision tree is a machine learning algorithm used for classification and regression tasks, organizing data into a tree-like structure based on predictor variables. We employ decision trees on our data for their interpretability, ability to capture nonlinear relationships, feature importance ranking, versatility, and scalability, providing actionable insights for decision-making tasks.

Using the Decision Tree Operator, with parameters 5,5, and 4 for maximal depth, minimal leaf size and minimal size for split respectively. The maximum depth parameter specifies the maximum number of levels allowed in the decision tree. The minimal leaf size parameter determines the minimum number of samples required to consider creating a leaf node. The minimal size for split parameter specifies the minimum number of samples required to split a node further.



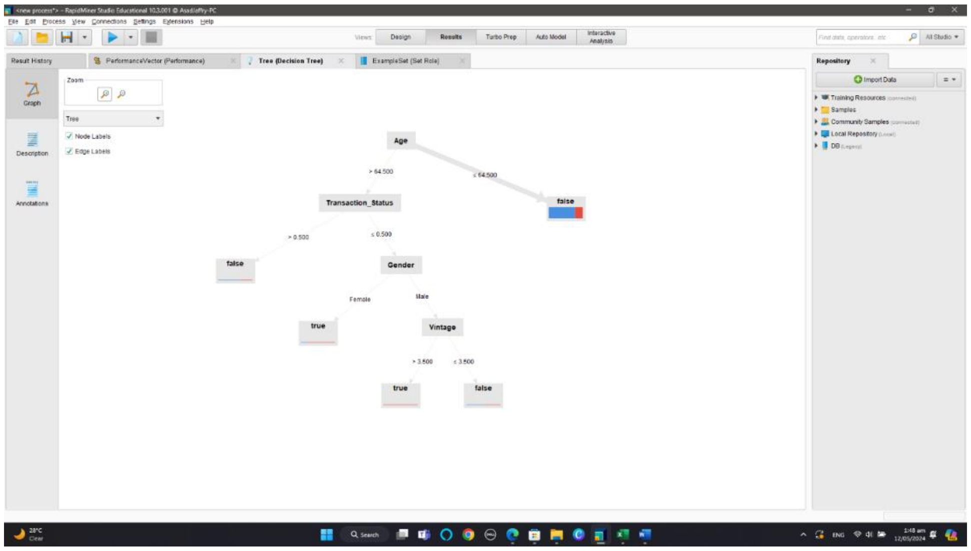
Parameters

Decision Tree

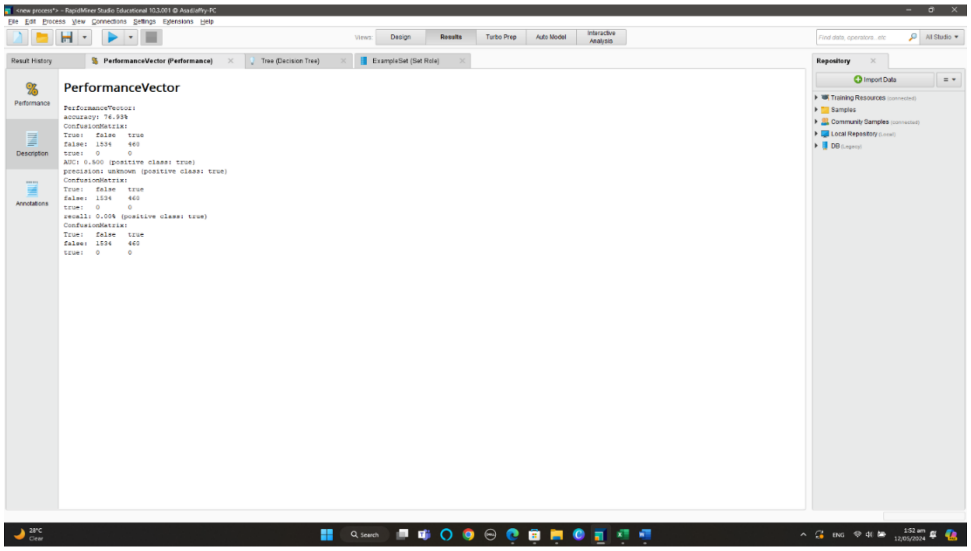
criterion	gain_ratio
maximal depth	5
✓ apply pruning	
confidence	0.1
✓ apply prepruning	
minimal gain	0.01
minimal leaf size	5
minimal size for split	4
number of prepruning alternatives	3

We get the following results.

1. Decision tree



2. Confusion Matrix

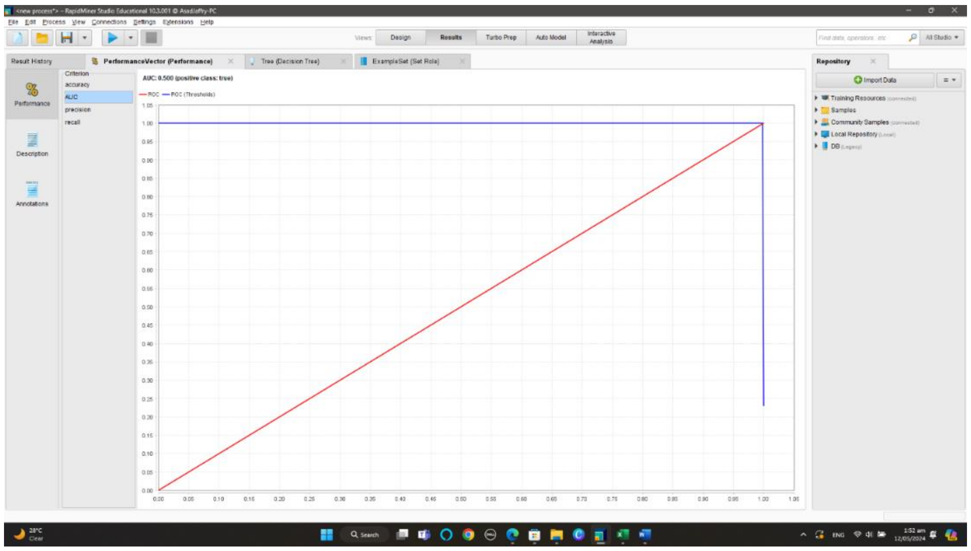


PerformanceVector

```
PerformanceVector:
accuracy: 76.93%
ConfusionMatrix:
True:   false   true
false:  1534    460
true:   0       0
AUC: 0.500 (positive class: true)
precision: unknown (positive class: true)
ConfusionMatrix:
True:   false   true
false:  1534    460
true:   0       0
recall: 0.00% (positive class: true)
ConfusionMatrix:
True:   false   true
false:  1534    460
true:   0       0
```

Accuracy has now increased to 76.93% compared to logistic regression. The increase in accuracy from logistic regression to decision tree suggests that the decision tree model might be better capturing the underlying patterns and relationships in the data compared to logistic regression. This improvement could be attributed to the decision tree's ability to handle nonlinear relationships and interactions between variables more effectively.

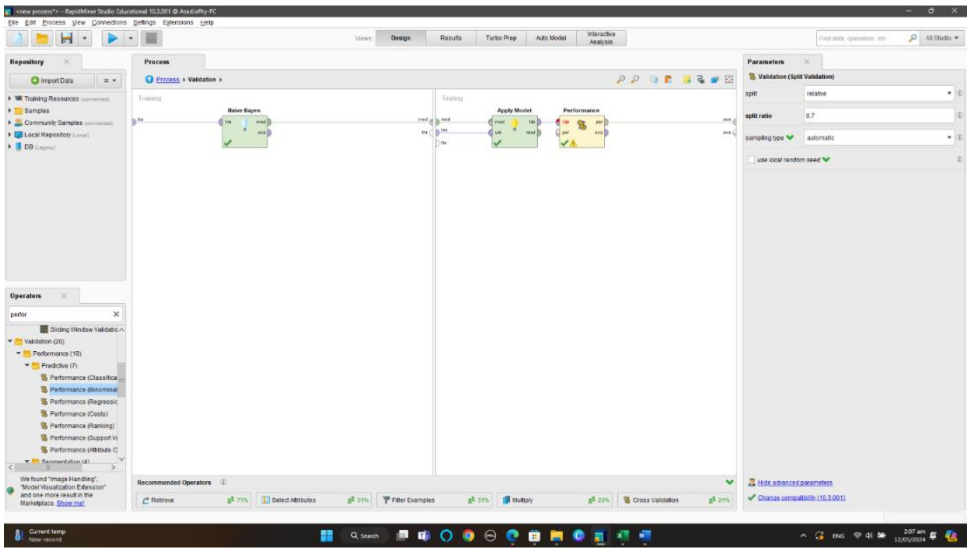
3. AUC



AUC equals 0.5, which is ideal.

Naïve Bayes

Naive Bayes is a probabilistic machine learning algorithm utilized primarily for classification tasks. It operates on the assumption of feature independence given the class variable, making it computationally efficient and easy to interpret. We apply Naive Bayes to our data due to its simplicity, efficiency, and versatility in handling both numerical and categorical data. Its scalability and suitability for large datasets make it particularly useful for real-time applications and big data scenarios. Additionally, Naive Bayes serves as a baseline model for comparing the performance of more complex algorithms. Overall, Naive Bayes is a valuable tool in data analysis and predictive modeling, offering a balance of simplicity and effectiveness.



Validation (Split Validation)

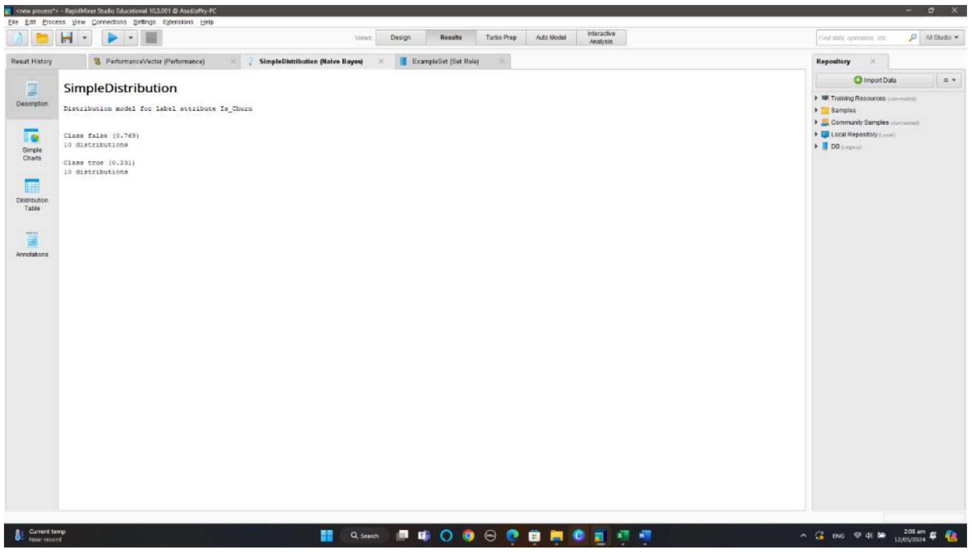
split: relative

split ratio: 0.7

sampling type: automatic

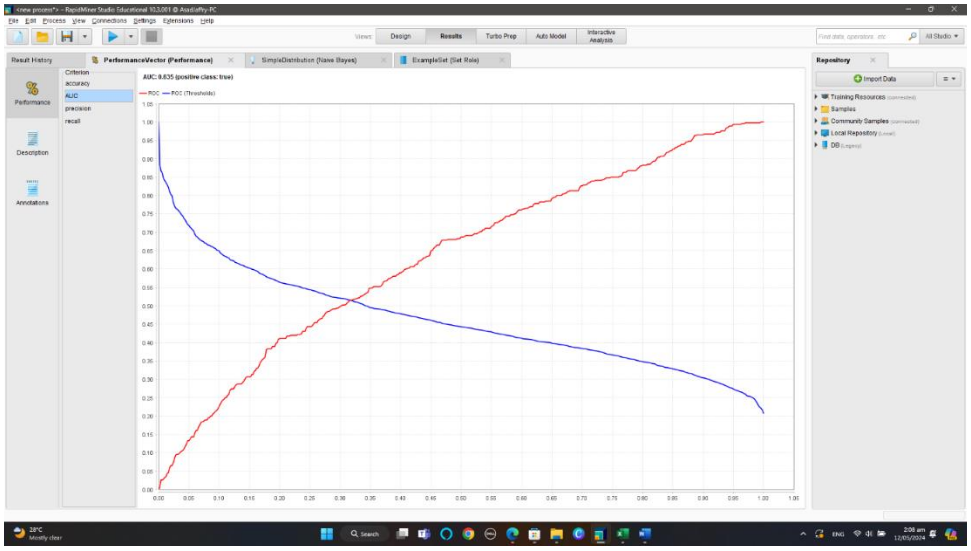
use local random seed

Result:



SimpleDistribution

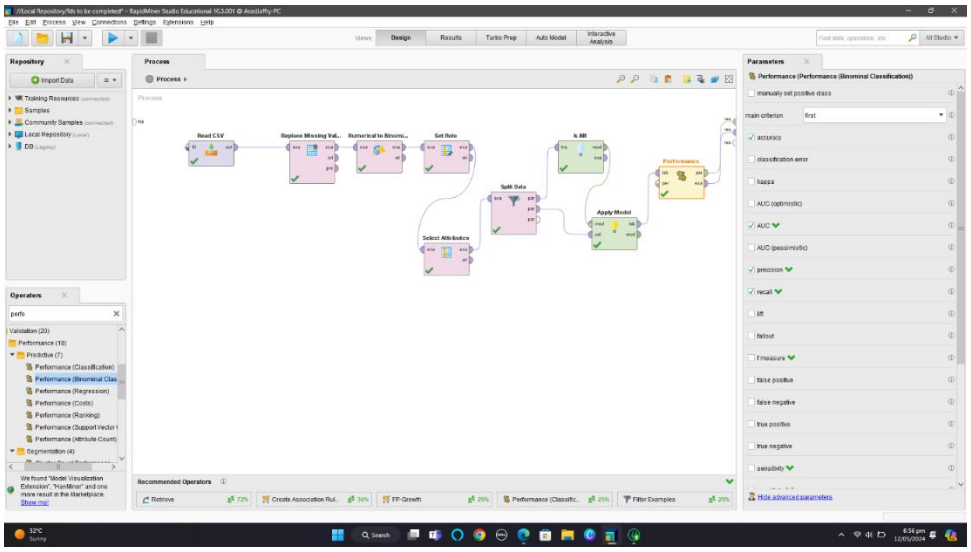
```
Distribution model for label attribute Is_Churn

Class false (0.769)
10 distributions

Class true (0.231)
10 distributions
```



AUC for Naïve Bayes is 0.635, higher than the logistic regression and decision tree. The higher AUC indicates that Naive Bayes is better at distinguishing between the positive and negative classes in the dataset. This suggests that Naive Bayes may have learned the underlying patterns and relationships in the data more effectively, resulting in improved predictive performance.

KNN

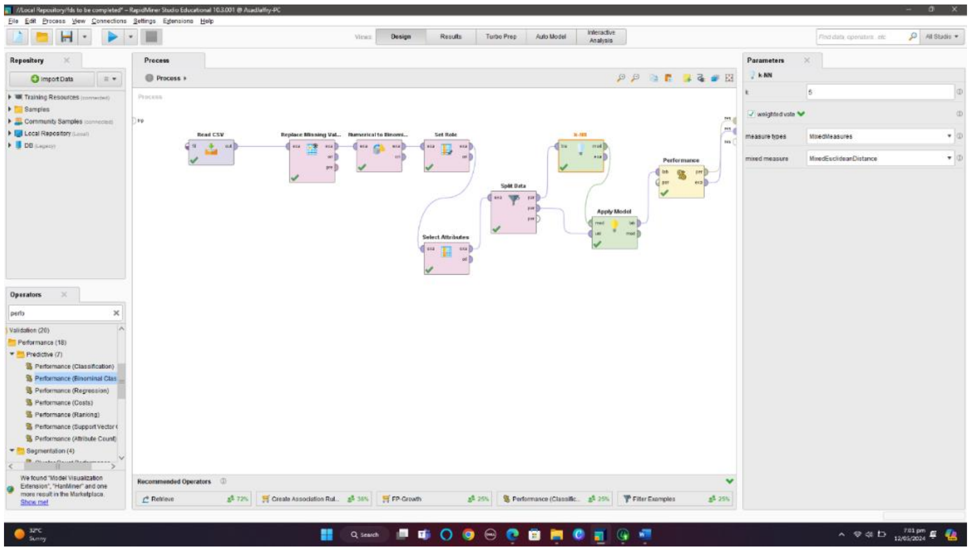
K-Nearest Neighbors (KNN) is a versatile supervised machine learning algorithm utilized for classification and regression tasks. Its simplicity and ease of implementation make it a popular choice, particularly for initial exploration of datasets. KNN operates by assigning a new data point to the majority class (for classification) or computing the average of its nearest neighbors (for regression) based on a specified number, K, of closest data points in the feature space. We perform KNN on our data for its flexibility in handling various types of predictive modeling problems, as well as its ability to capture complex relationships between variables without making assumptions about the underlying data distribution. Additionally, the interpretability of KNN results, coupled with its lack of a training phase, makes it an appealing option for quick insights and decision-making tasks in data analysis.



Performance (Performance (Binominal Classification))

manually set positive class	☐
main criterion	first
accuracy	✓
classification error	☐
kappa	☐
AUC (optimistic)	☐
AUC	✓
AUC (pessimistic)	☐
precision	✓
recall	✓
lift	☐
fallout	☐
f measure	☐
false positive	☐
false negative	☐
true positive	☐
true negative	☐
sensitivity	☐

Parameters for KNN; k=5. What this means is that the algorithm considers the five nearest neighbors when making predictions for a new data point. These neighbors are determined based on Euclidean distance, and the majority class among the nearest neighbors is used as the predicted class for classification tasks.



Parameters

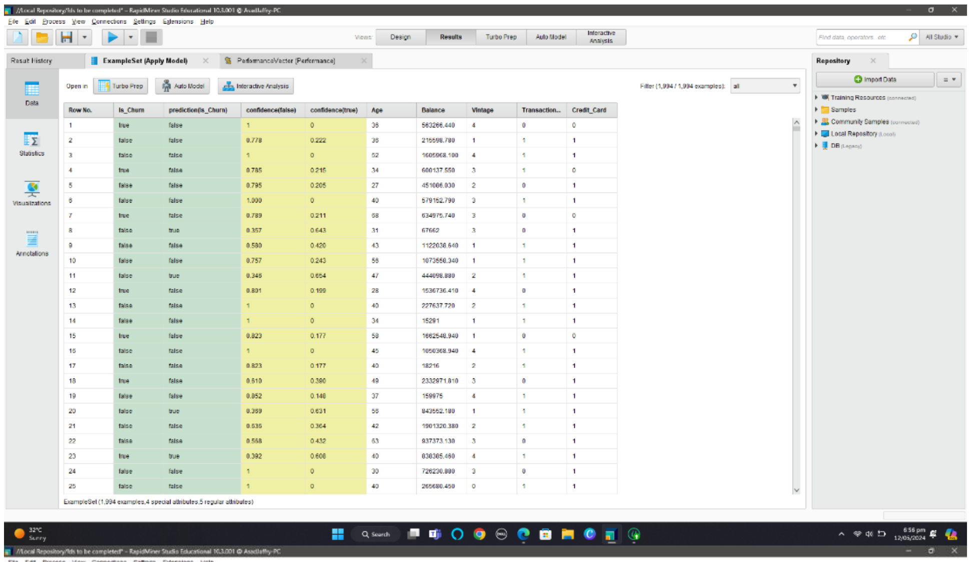
k: 5

weighted vote ✓

measure types: MixedMeasures

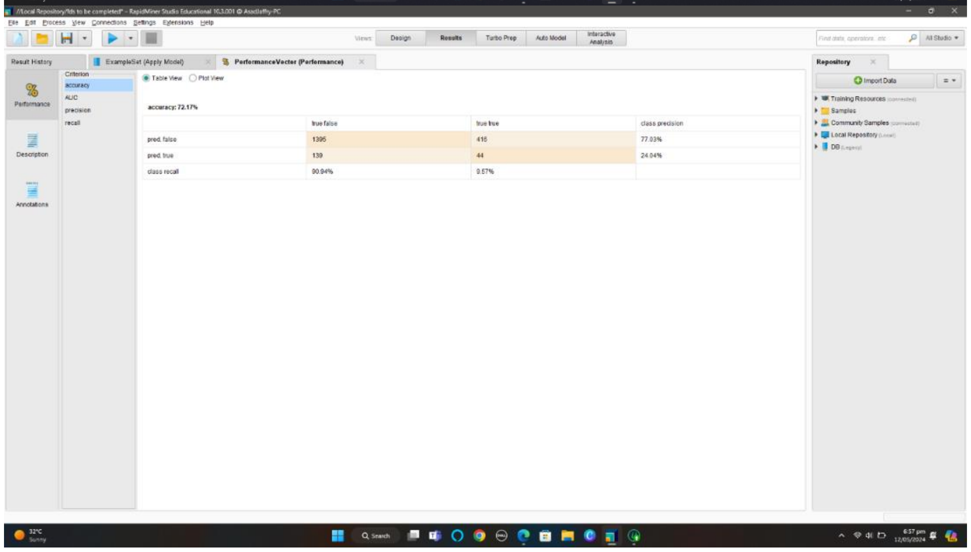
mixed measure: MixedEuclideanDistance

Result:



Row No.	Is_Churn	prediction(Is_Churn)	confidence(false)	confidence(true)	Age	Balance	Vintage	Transaction...	Credit_Card
1	true	false	1	0	36	563266.440	4	0	0
2	false	false	0.778	0.222	36	215598.780	1	1	1
3	false	false	1	0	52	1605968.100	4	1	1
4	true	false	0.785	0.215	34	600137.550	3	1	0
5	false	false	0.795	0.205	27	451006.030	2	0	1
6	false	false	1.000	0	40	579152.790	3	1	1
7	true	false	0.789	0.211	68	634975.740	3	0	0
8	false	true	0.357	0.643	31	67662	3	0	1
9	false	false	0.580	0.420	43	1122038.640	1	1	1
10	false	false	0.757	0.243	56	1073558.340	1	1	1
11	false	true	0.346	0.654	47	444098.880	2	1	1
12	true	false	0.801	0.199	28	1536736.410	4	0	1
13	false	false	1	0	40	227637.720	2	1	1
14	false	false	1	0	34	15291	1	1	1
15	true	false	0.823	0.177	58	1662548.940	1	0	0
16	false	false	1	0	45	1050368.940	4	1	1
17	false	false	0.823	0.177	40	18216	2	1	1
18	true	false	0.610	0.390	49	2332971.810	3	0	1
19	false	false	0.852	0.148	37	159975	4	1	1
20	false	true	0.369	0.631	56	843562.180	1	1	1
21	false	false	0.636	0.364	42	1901320.380	2	1	1
22	false	false	0.568	0.432	63	937373.130	3	0	1
23	true	true	0.392	0.608	40	838385.460	4	1	1
24	false	false	1	0	30	726230.880	3	0	1
25	false	false	1	0	40	265680.450	0	1	1

ExampleSet (1,994 examples,4 special attributes,5 regular attributes)



Criterion: accuracy, AUC, precision, recall

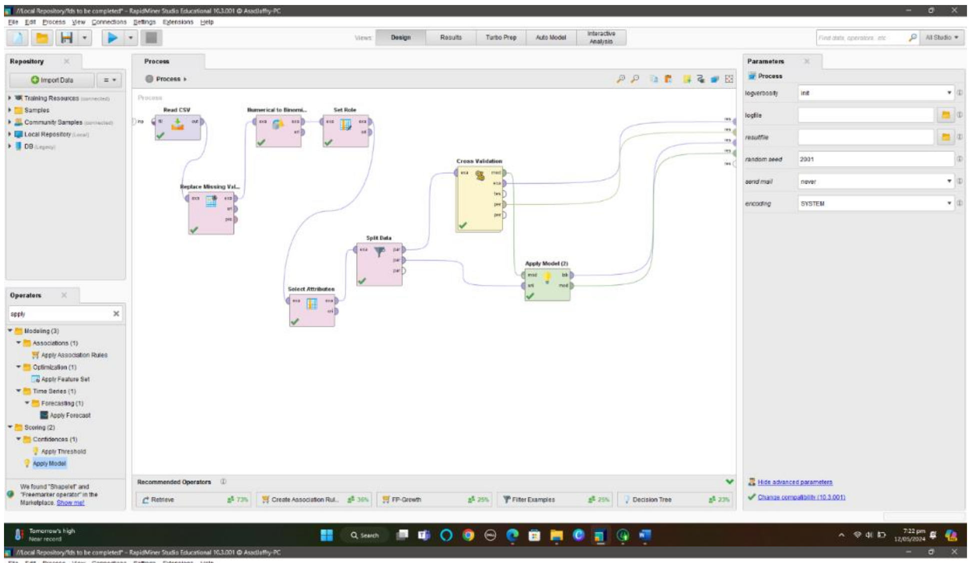
accuracy: 72.17%

	true false	true true	class precision
pred. false	1395	416	77.03%
pred. true	139	44	24.04%
class recall	90.94%	9.57%	

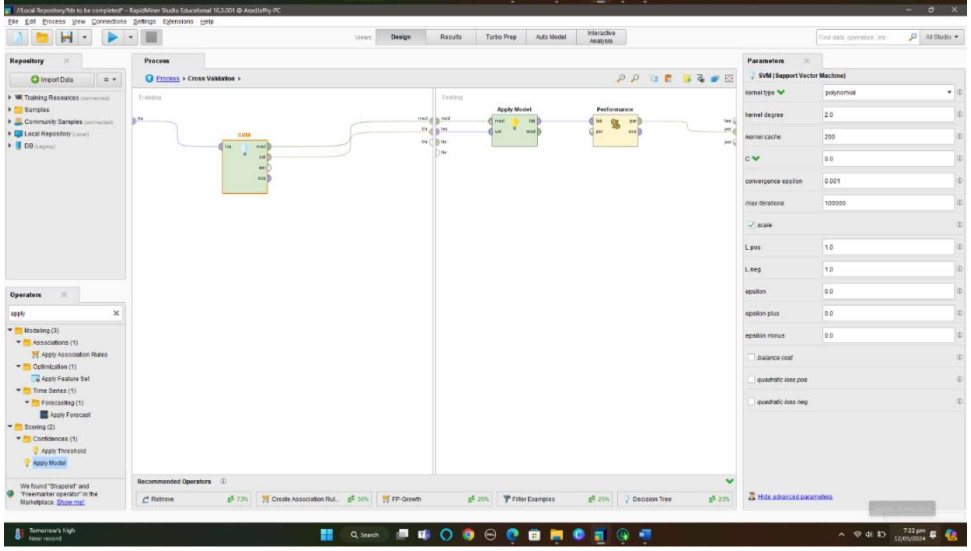
Accuracy for KNN is 72.17%, higher than the previous three machine learning models. AUC is 0.508, and the precision is 24.04%.

SVM

Support Vector Machine (SVM) is a versatile supervised machine learning algorithm used for classification, regression, and outlier detection tasks. SVM identifies the hyperplane that best separates data points into different classes, aiming to maximize the margin between them. We utilize SVM on our data due to its versatility in handling both linear and nonlinear classification tasks through various kernel functions. SVM is robust to overfitting, particularly in high-dimensional spaces, making it suitable for datasets with many features. Its effectiveness in high-dimensional spaces, along with its focus on maximizing the margin between classes and robustness to outliers, makes SVM a valuable tool for various data analysis and predictive modeling tasks.

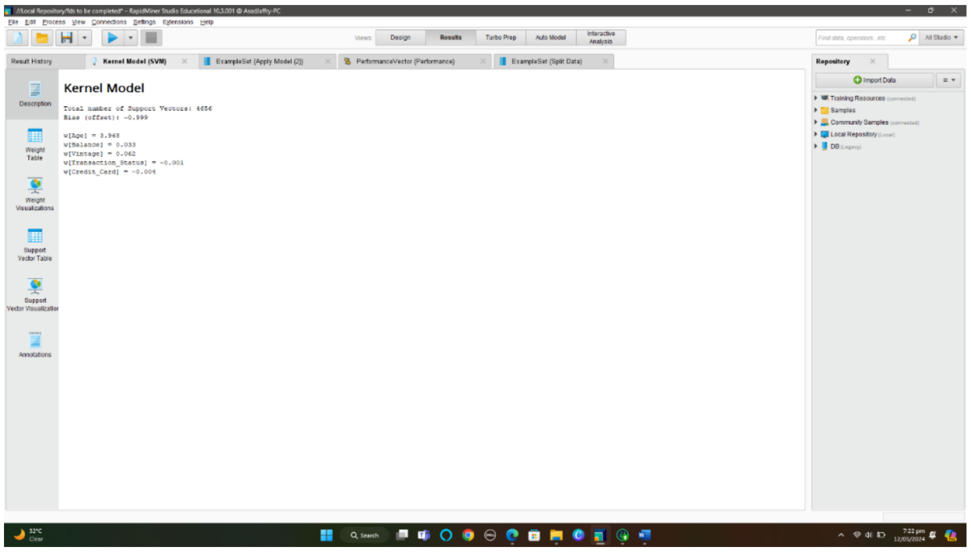


Parameters
Process
logverbosity: init
logfile:
resultfile:
random seed: 2001
send mail: never
encoding: SYSTEM



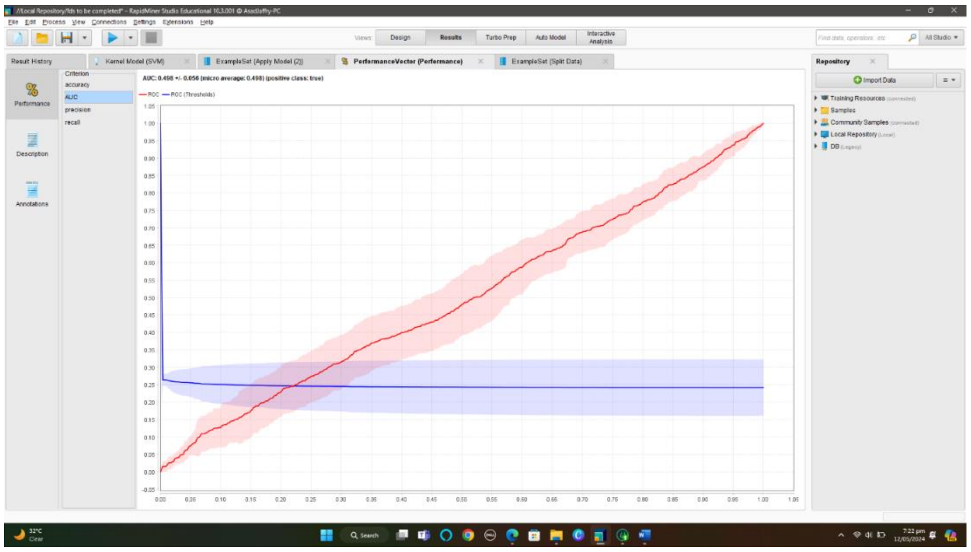
Parameters
SVM (Support Vector Machine)
kernel type: polynomial
kernel degree: 2.0
kernel cache: 200
C: 0.0
convergence epsilon: 0.001
max iterations: 100000
[✓] scale
L pos: 1.0
L neg: 1.0
epsilon: 0.0
epsilon plus: 0.0
epsilon minus: 0.0
[ ] balance cost
[ ] quadratic loss pos
[ ] quadratic loss neg

Kernel type is set to polynomial so to identify any complex, non-linear relationships between input features and the target variable. It allows the machine learning model to project the data into higher-dimensional spaces, where it becomes easier to find a hyperplane that can effectively separate the classes.
C is a hyperparameter known as the regularization parameter. It controls the trade-off between maximizing the margin and minimizing the classification error. Setting C to zero in SVM effectively removes the regularization term from the objective function. This means that the SVM algorithm will focus solely on maximizing the margin, without penalizing misclassifications.


//Local Repository/fds to be completed* – RapidMiner Studio Educational 10.3.001 @ AsadJaffry-PC
File Edit Process View Connections Settings Extensions Help
Result History
Kernel Model (SVM)
ExampleSet (Apply Model (2))
PerformanceVector (Performance)
ExampleSet (Split Data)
Description
Weight Table
Weight Visualizations
Support Vector Table
Support Vector Visualization
Annotations
Repository
Import Data

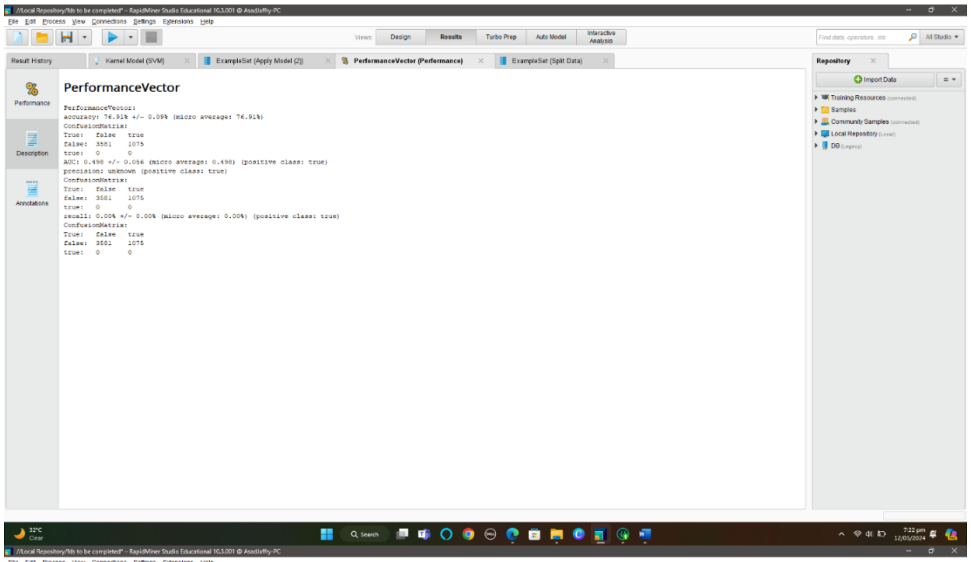
Kernel Model

```
Total number of Support Vectors: 4656
Bias (offset): -0.999

w[Age] = 3.968
w[Balance] = 0.033
w[Vintage] = 0.062
w[Transaction_Status] = -0.001
w[Credit_Card] = -0.004
```

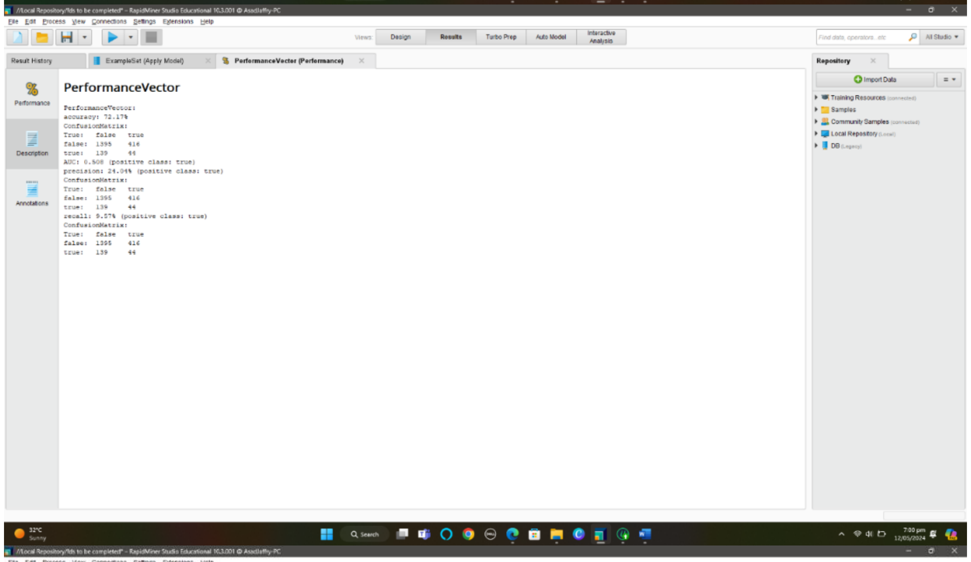


AUC: 0.498 +/- 0.056 (micro average: 0.498) (positive class: true)

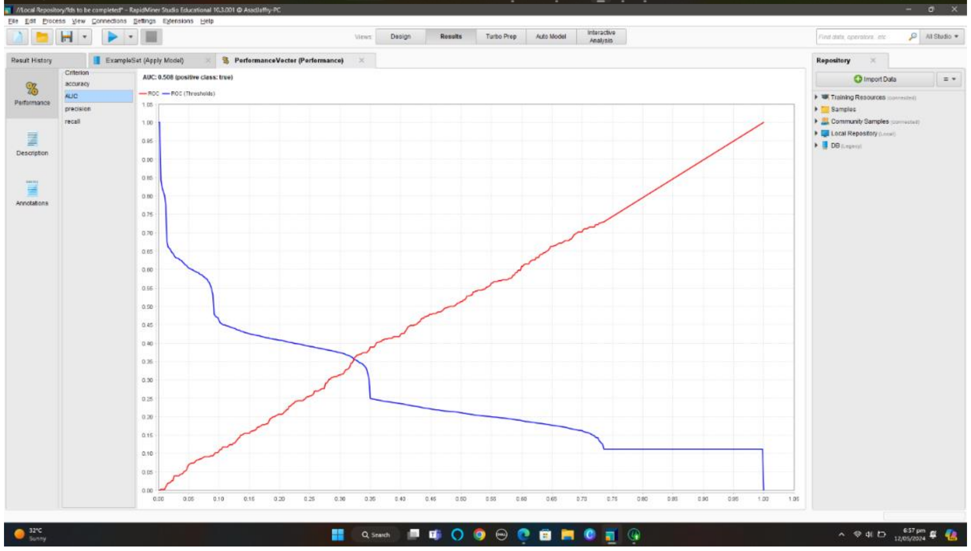
AUC for the kernel model comes out to be 0.498.



```
PerformanceVector

PerformanceVector:
accuracy: 76.91% +/- 0.09% (micro average: 76.91%)
ConfusionMatrix:
True:   false   true
false:  3581    1075
true:   0       0
AUC: 0.498 +/- 0.056 (micro average: 0.498) (positive class: true)
precision: unknown (positive class: true)
ConfusionMatrix:
True:   false   true
false:  3581    1075
true:   0       0
recall: 0.00% +/- 0.00% (micro average: 0.00%) (positive class: true)
ConfusionMatrix:
True:   false   true
false:  3581    1075
true:   0       0
```



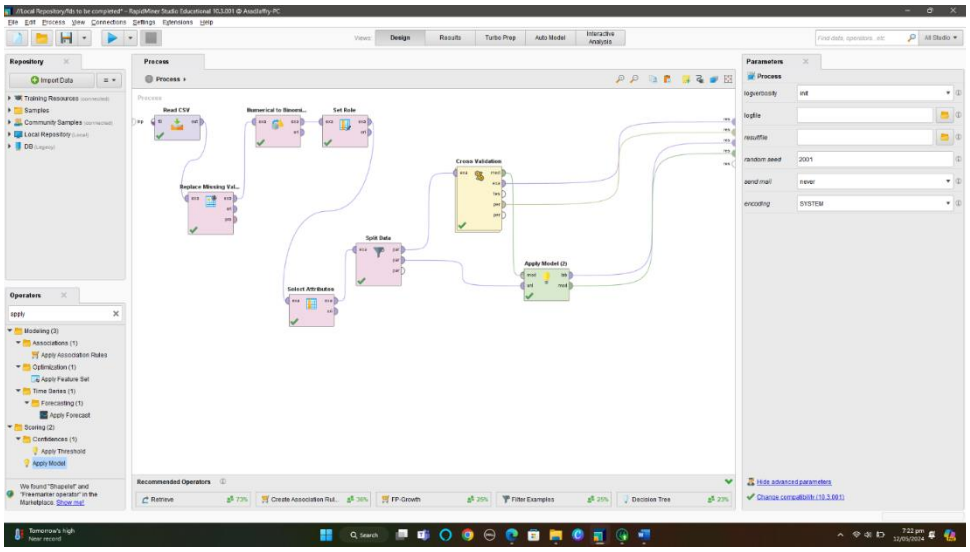
```
PerformanceVector

PerformanceVector:
accuracy: 72.17%
ConfusionMatrix:
True:   false   true
false:  1395    416
true:   139     44
AUC: 0.508 (positive class: true)
precision: 24.04% (positive class: true)
ConfusionMatrix:
True:   false   true
false:  1395    416
true:   139     44
recall: 9.57% (positive class: true)
ConfusionMatrix:
True:   false   true
false:  1395    416
true:   139     44
```



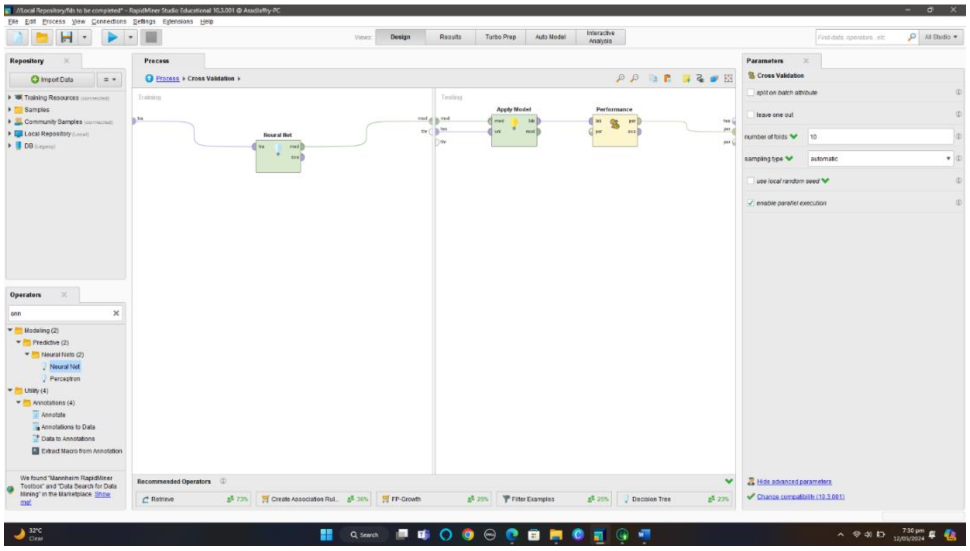
Accuracy for performance vector is 76.91%. AUC is 0.498

ANN

Artificial Neural Networks (ANNs) are sophisticated machine learning models inspired by the structure and function of the human brain. Comprising interconnected nodes organized into layers, ANNs learn complex patterns and relationships in data through training, adjusting weights and biases based on input-output pairs. We employ ANNs on our data due to their capability to capture nonlinear relationships, automate feature learning, and achieve state-of-the-art performance across various tasks such as classification, regression, and unsupervised learning.



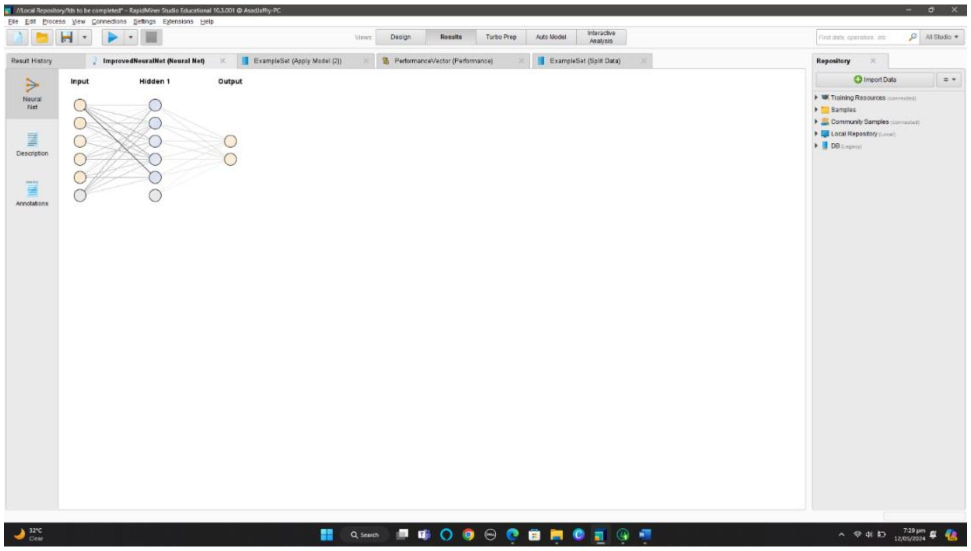
Parameters
Process
logverbosity: init
random seed: 2001
send mail: never
encoding: SYSTEM

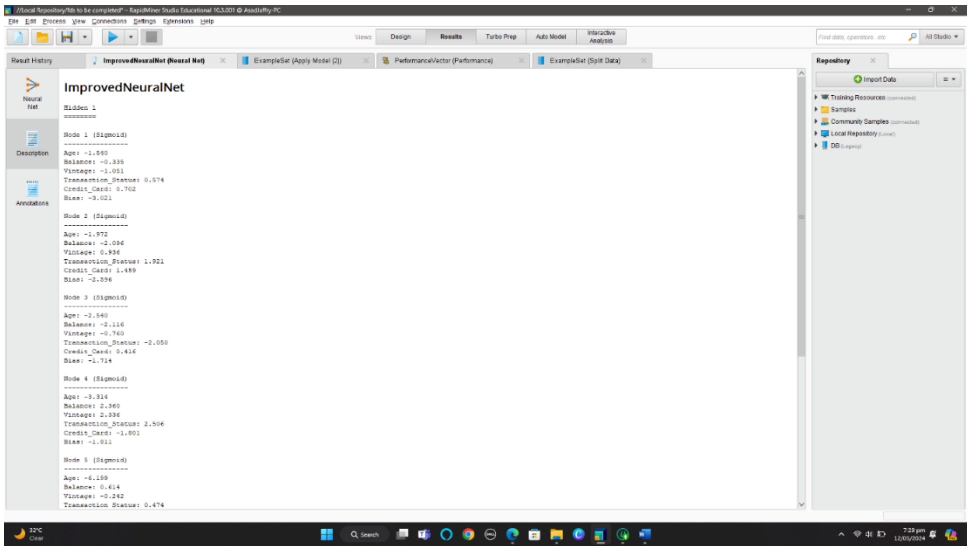


Parameters
Cross Validation
☐ split on batch attribute
☐ leave one out
number of folds: 10
sampling type: automatic
☐ use local random seed
✓ enable parallel execution

Setting the sampling type to "automatic" for Artificial Neural Networks (ANNs) indicates that the algorithm will automatically handle the sampling strategy based on the dataset characteristics and the specific implementation of the ANN algorithm being used. This simplifies model selection and potentially improves performance.

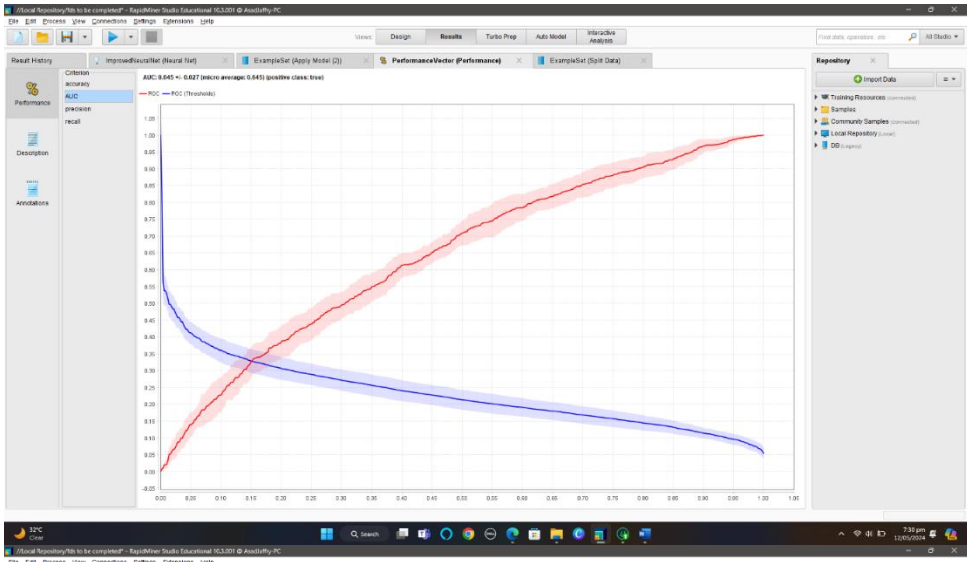
Result:

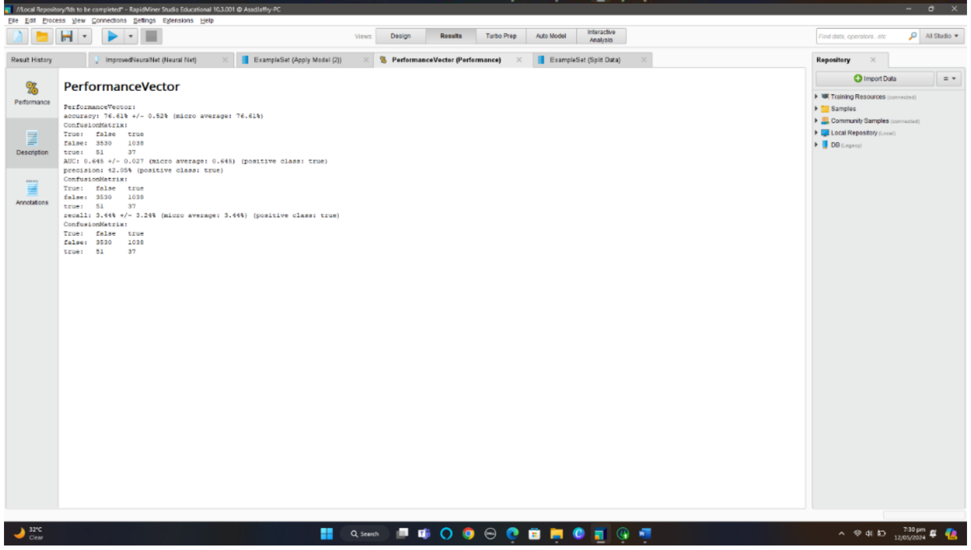




ImprovedNeuralNet

```
Hidden 1
========

Node 1 (Sigmoid)
----------------
Age: -1.540
Balance: -0.335
Vintage: -1.051
Transaction_Status: 0.574
Credit_Card: 0.702
Bias: -3.021

Node 2 (Sigmoid)
----------------
Age: -1.972
Balance: -2.096
Vintage: 0.936
Transaction_Status: 1.921
Credit_Card: 1.459
Bias: -2.596

Node 3 (Sigmoid)
----------------
Age: -2.540
Balance: -2.116
Vintage: -0.760
Transaction_Status: -2.050
Credit_Card: 0.416
Bias: -1.714

Node 4 (Sigmoid)
----------------
Age: -3.314
Balance: 2.360
Vintage: 2.336
Transaction_Status: 2.506
Credit_Card: -1.801
Bias: -1.811

Node 5 (Sigmoid)
----------------
Age: -6.199
Balance: 0.614
Vintage: -0.242
Transaction_Status: 0.474
```





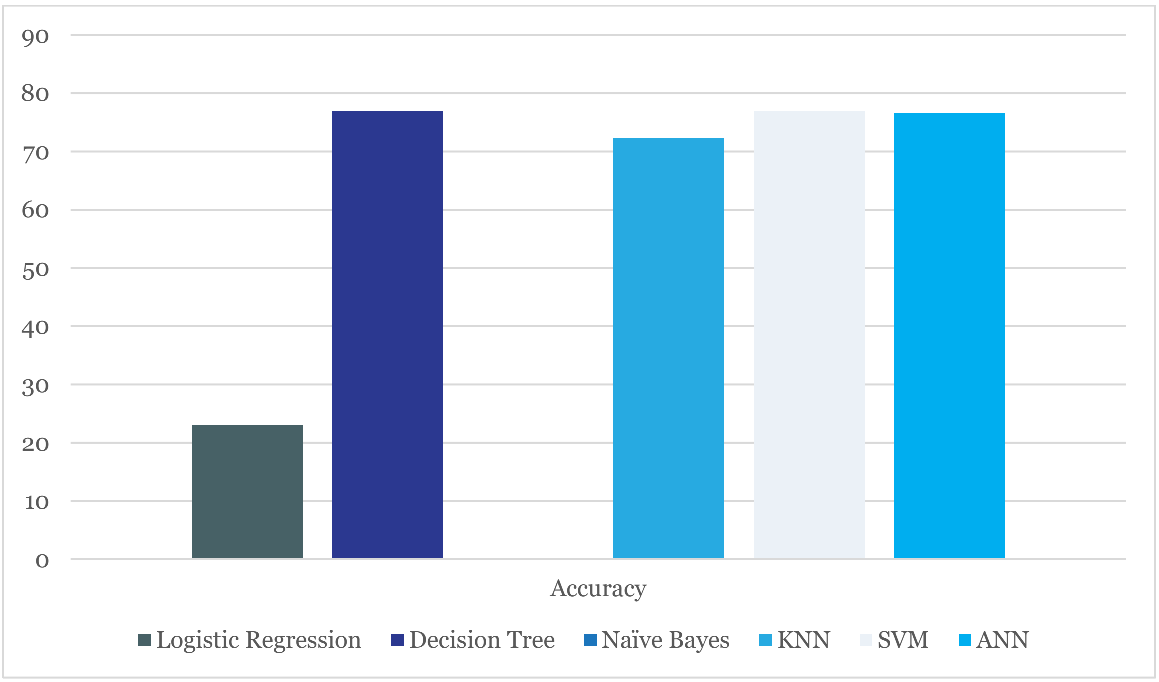
PerformanceVector

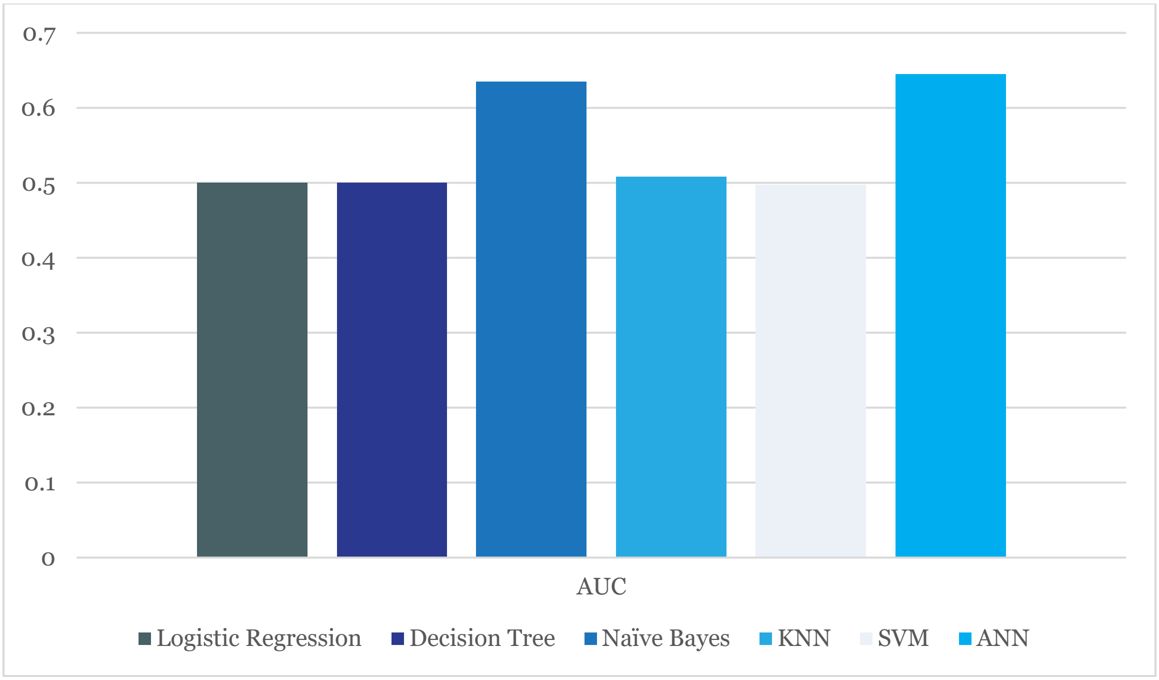
```
PerformanceVector:
accuracy: 76.61% +/- 0.52% (micro average: 76.61%)
ConfusionMatrix:
True:   false   true
false:  3530    1038
true:   51      37
AUC: 0.645 +/- 0.027 (micro average: 0.645) (positive class: true)
precision: 42.05% (positive class: true)
ConfusionMatrix:
True:   false   true
false:  3530    1038
true:   51      37
recall: 3.44% +/- 3.24% (micro average: 3.44%) (positive class: true)
ConfusionMatrix:
True:   false   true
false:  3530    1038
true:   51      37
```

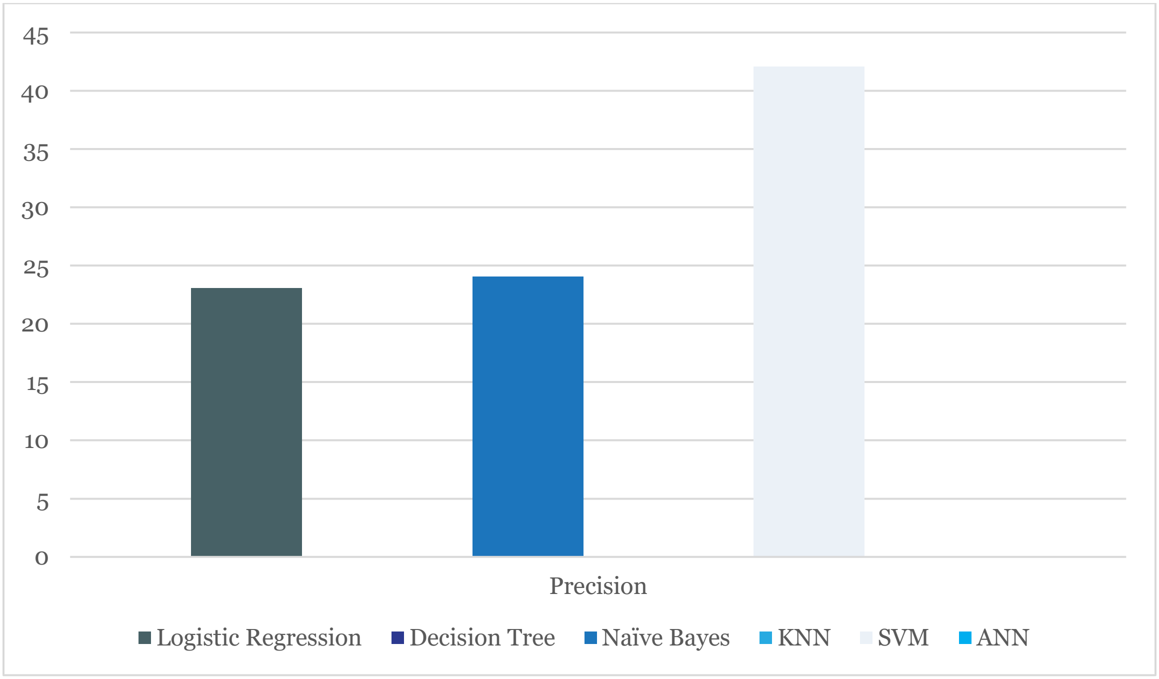
Accuracy for ANN model is 76.61%. AUC comes out to be 0.645 and precision is 42.05%.

Comparison between models

Comparing the performance of machine learning models is essential for evaluating their effectiveness in classification tasks. Accuracy measures the overall correctness of predictions, precision quantifies the proportion of correctly predicted positive cases among all predicted positive cases, and recall assesses the proportion of correctly predicted positive cases among all actual positive cases. By visualizing these metrics using bar charts, we can gain insights into how different models perform across multiple evaluation criteria. This comparison allows us to identify the strengths and weaknesses of each model and make informed decisions about which model is best suited for the specific task at hand.







The results show that Decision Tree, SVM, and ANN outperform Logistic Regression, Naïve Bayes, and KNN in terms of accuracy. However, it's essential to note that Decision Tree has a recall of 0%, indicating potential issues with correctly identifying positive cases. Naïve Bayes demonstrates the highest AUC, suggesting its better performance in distinguishing between positive and negative cases. Despite having a relatively low recall, ANN exhibits the highest precision, indicating its capability to correctly classify positive cases.