**Business Intelligence Lab**

**Experiment 06**

**Aim:**

To implement any one of the classification algorithms (Decision tree / Naive Bayes) / Technique using python

**Theory:**

**Classification:**

Classification is the process of predicting the class of given data points. Classes are sometimes called targets/ labels or categories. Classification predictive modeling is the task of approximating a mapping function (f) from input variables (X) to discrete output variables (y).

For example, spam detection in email service providers can be identified as a classification problem. This is a binary classification since there are only 2 classes spam and not spam. A classifier utilizes some training data to understand how given input variables relate to the class. In this case, known spam and non-spam emails have to be used as the training data. When the classifier is trained accurately, it can be used to detect an unknown email.

Classification belongs to the category of supervised learning where the targets are also provided with the input data. There are many applications in classification in many domains such as credit approval, medical diagnosis, target marketing, etc.

There are two types of learners in classification:

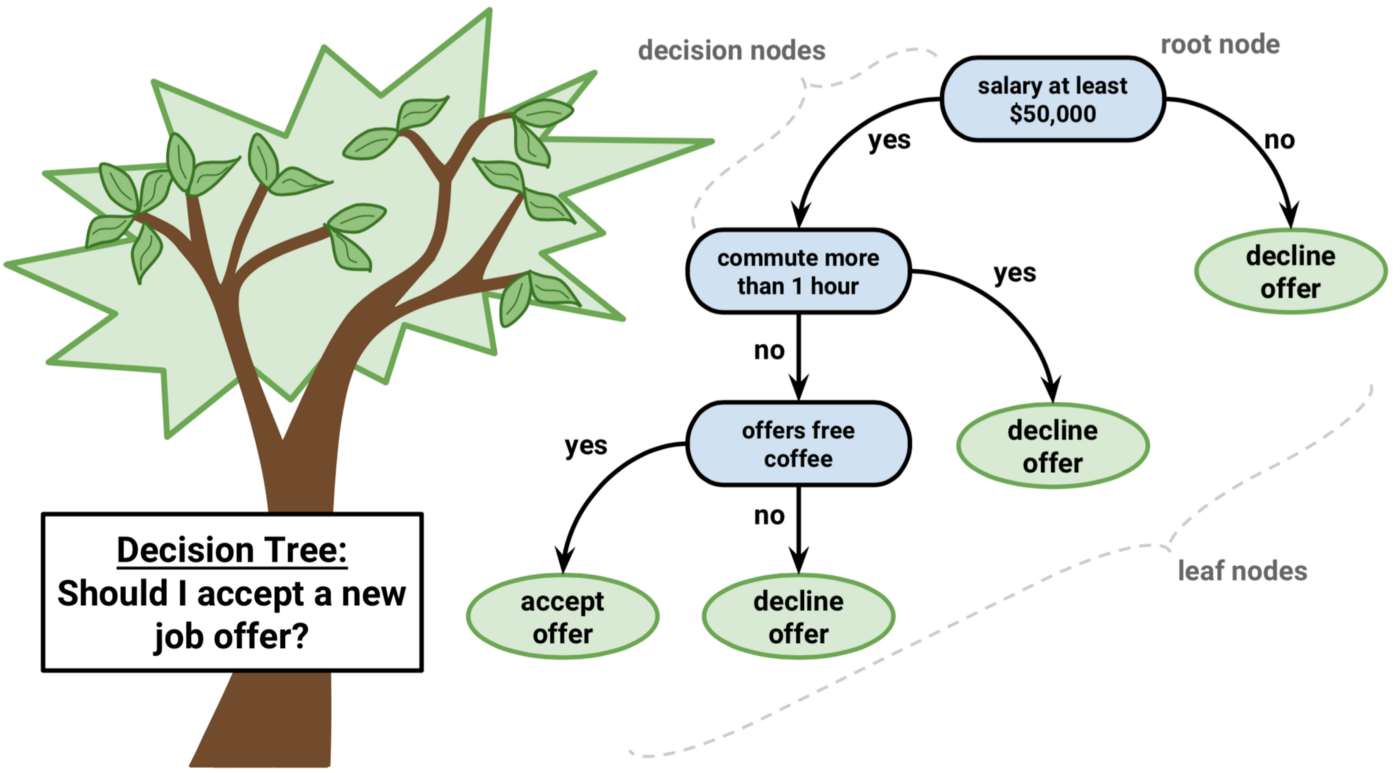
1. Lazy learners - Lazy learners simply store the training data and wait until testing data appears. When it does, classification is conducted based on the most related data in the stored training data. Compared to eager learners, lazy learners have less training time but more time in predicting. Example: k-nearest neighbor, Case-based reasoning.
2. Eager learners - Eager learners construct a classification model based on the given training data before receiving data for classification. It must be able to commit to a single hypothesis that covers the entire instance space. Due to the model construction, eager learners take a long time to train and less time to predict. Example: Decision Tree, Naive Bayes, Artificial Neural Networks.

**Classification algorithms**

There are a lot of classification algorithms available now but it is not possible to conclude which one is superior to the other. It depends on the application and nature of the available data set. For example, if the classes are linearly separable, the linear classifiers like Logistic regression, Fisher’s linear discriminant can outperform sophisticated models and vice versa.

**Decision Tree**

Decision trees build classification or regression models in the form of a tree structure. It utilizes an if-then rule set which is mutually exclusive and exhaustive for classification. The rules are learned sequentially using the training data one at a time. Each time a rule is learned, the tuples covered by the rules are removed. This process is continued on the training set until meeting a termination condition.

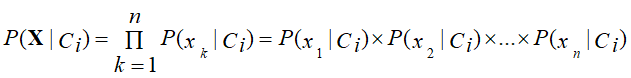


The tree is constructed in a top-down recursive divide-and-conquer manner. All the attributes should be categorical. Otherwise, they should be discretized in advance. Attributes in the top of the tree have more impact in the classification and they are identified using the information gain concept.

A decision tree can be easily over-fitted generating too many branches and may reflect anomalies due to noise or outliers. An over-fitted model has a very poor performance on the unseen data even though it gives an impressive performance on training data. This can be avoided by pre-pruning which halts tree construction early or post-pruning which removes branches from the fully grown tree.

**Naive Bayes**

Naive Bayes is a probabilistic classifier inspired by the Bayes theorem under a simple assumption that the attributes are conditionally independent.

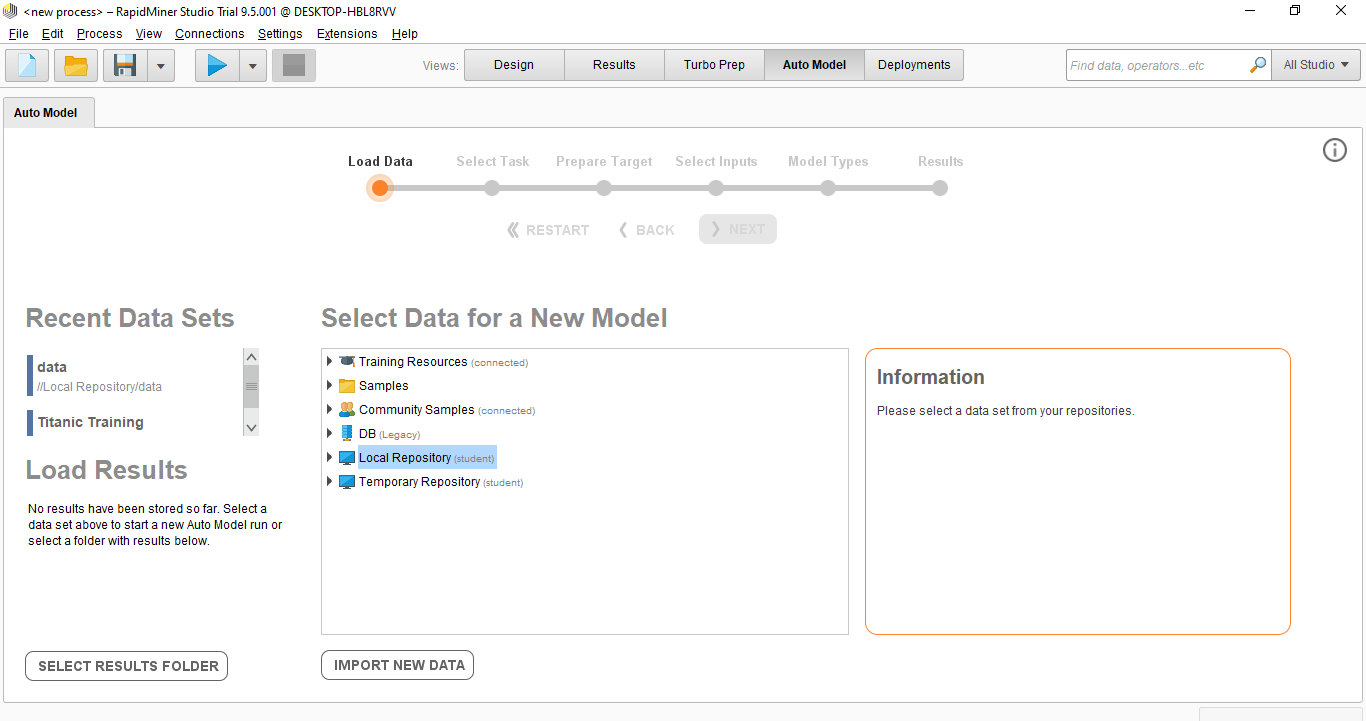


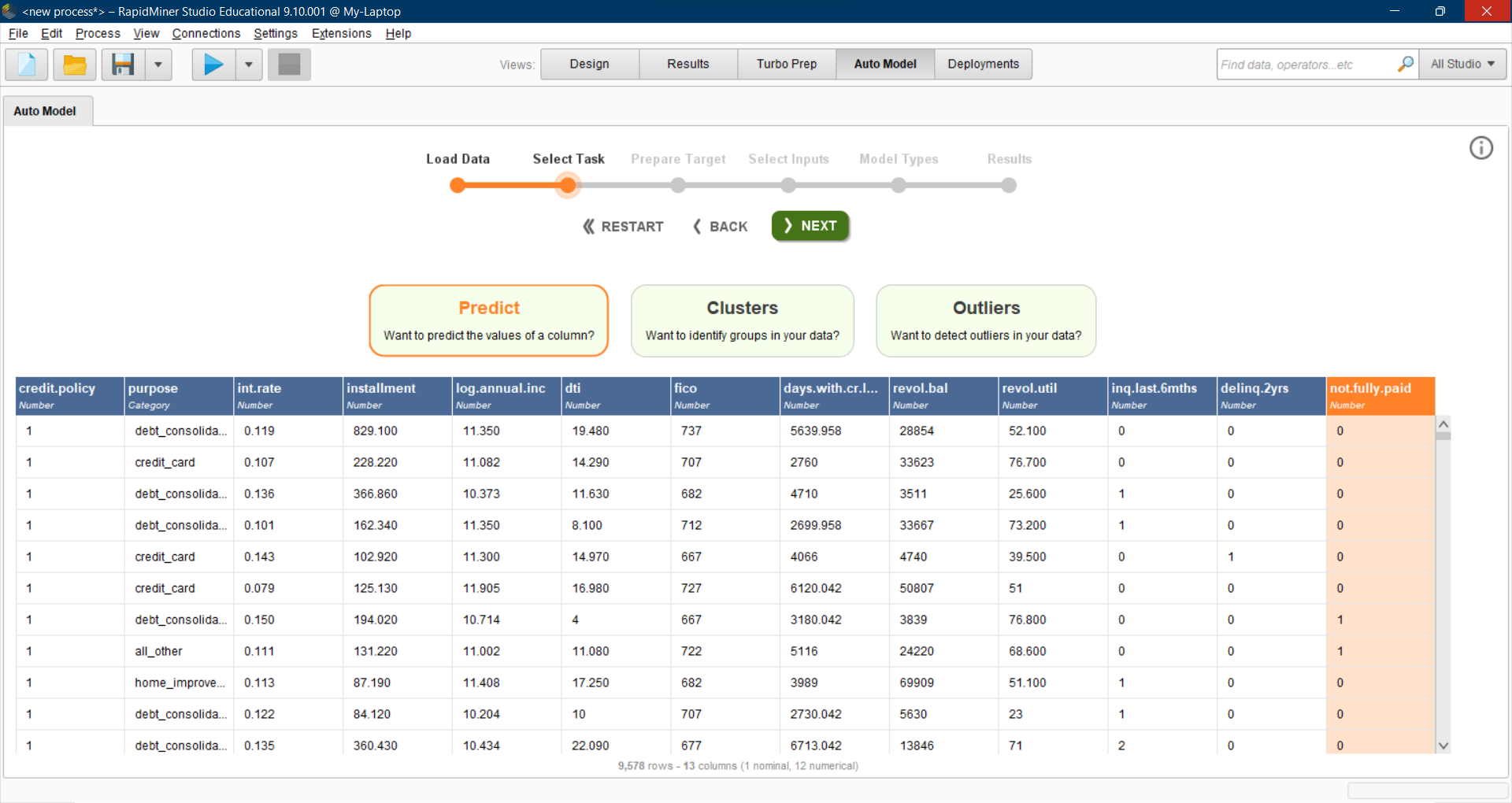
The classification is conducted by deriving the maximum posterior which is the maximal P(Ci|X) with the above assumption applying to Bayes theorem. This assumption greatly reduces the computational cost by only counting the class distribution. Even though the assumption is not valid in most cases since the attributes are dependent, surprisingly Naive Bayes was able to perform impressively.

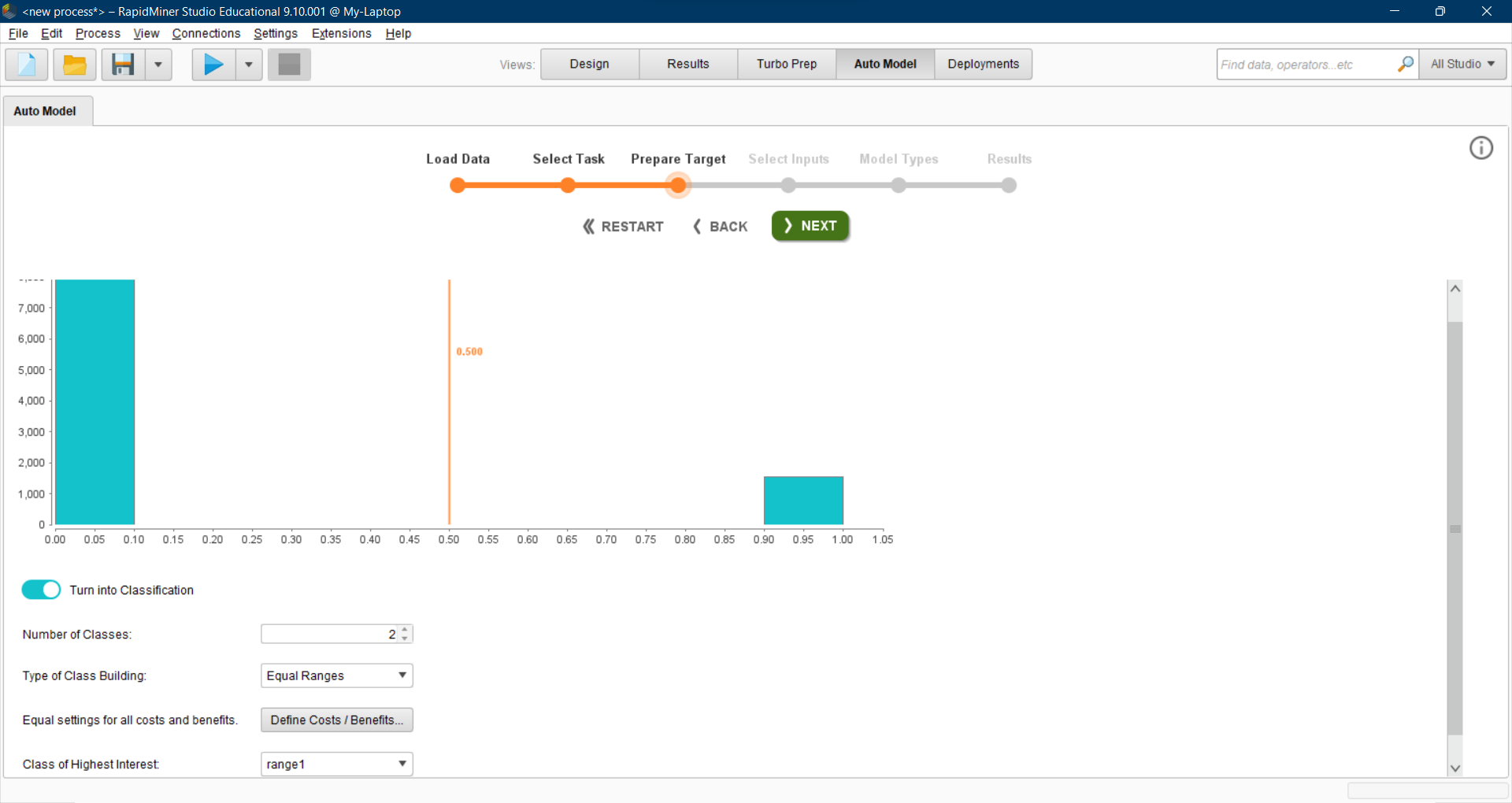
Naive Bayes is a very simple algorithm to implement and good results have been obtained in most cases. It can be easily scalable to larger datasets since it takes linear time, rather than by expensive iterative approximation as used for many other types of classifiers.

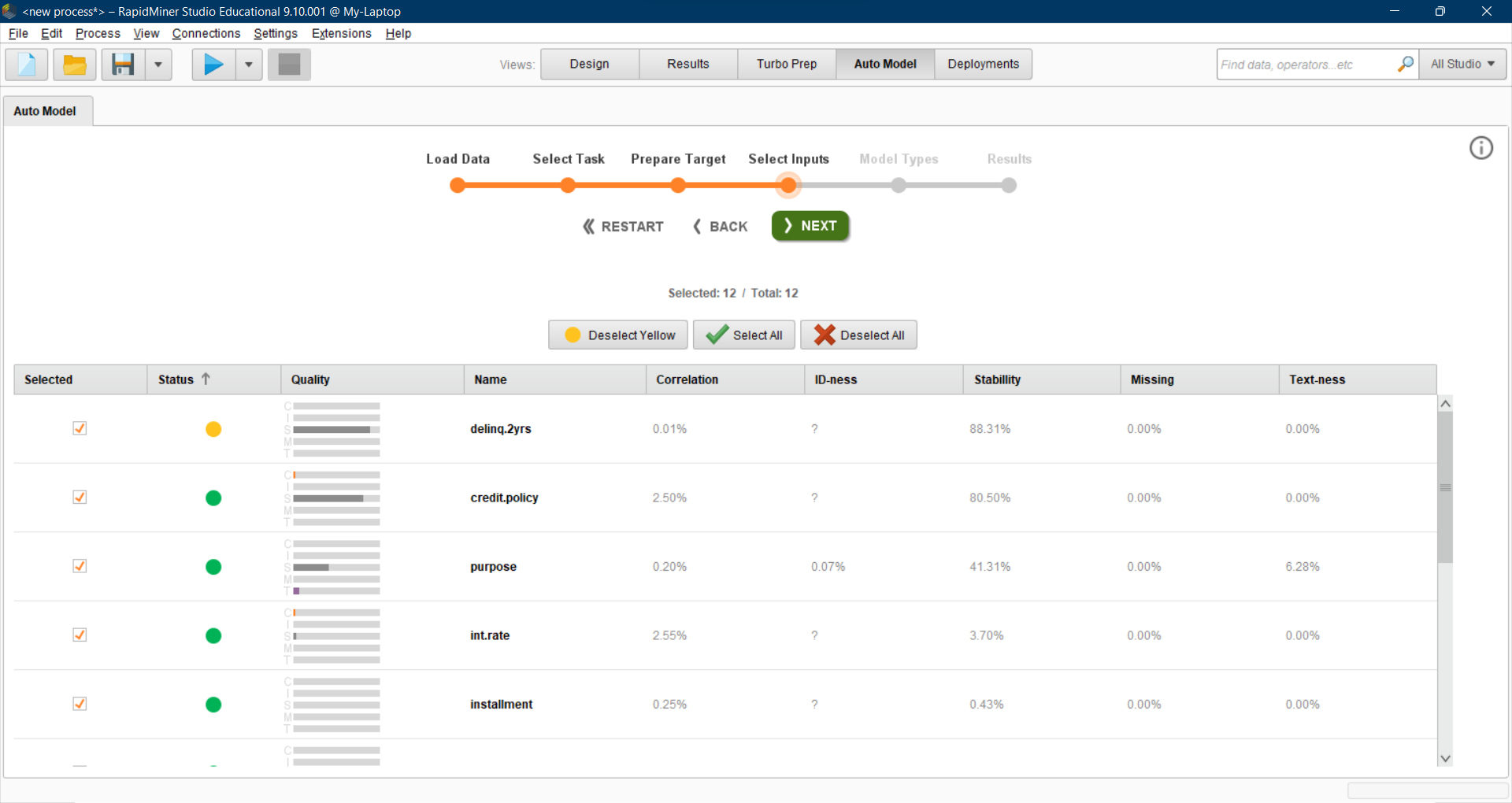
Naive Bayes can suffer from a problem called the zero probability problem. When the conditional probability is zero for a particular attribute, it fails to give a valid prediction. This needs to be fixed explicitly using a Laplacian estimator.

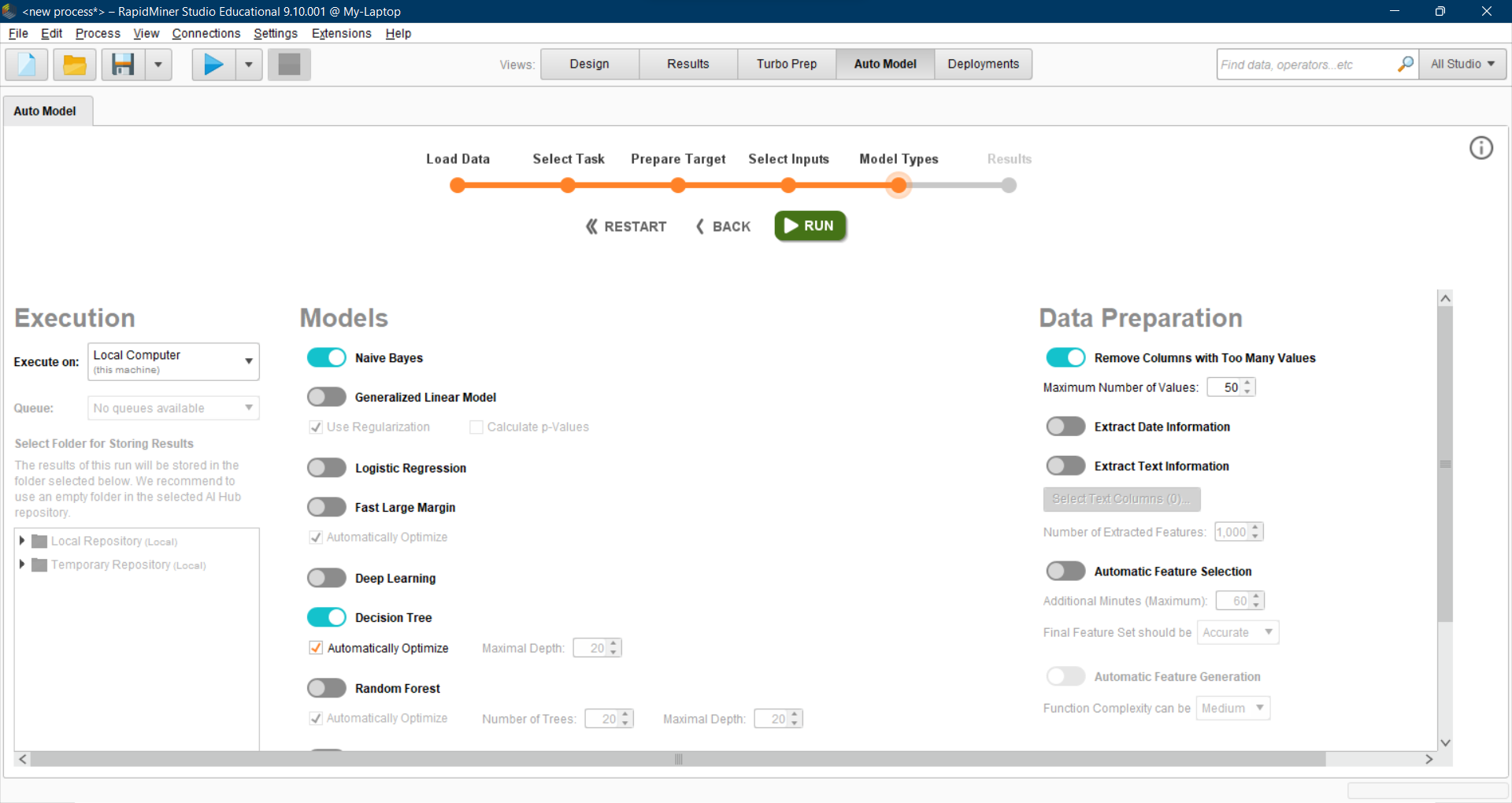
**Classification Model using RapidMiner**











**Metrics and Comparison of Models**

