**EXPERIMENT 7**

**AIM:** To implement a regression model using Rapid Miner and Python.

**THEORY:**

Linear regression is a type of regression analysis used to model the relationship between a dependent variable and one or more independent variables. In simple linear regression, there is only one independent variable, while in multiple linear regression, there are multiple independent variables.

The goal of linear regression is to find a linear equation that can best fit the relationship between the dependent variable and independent variable(s).

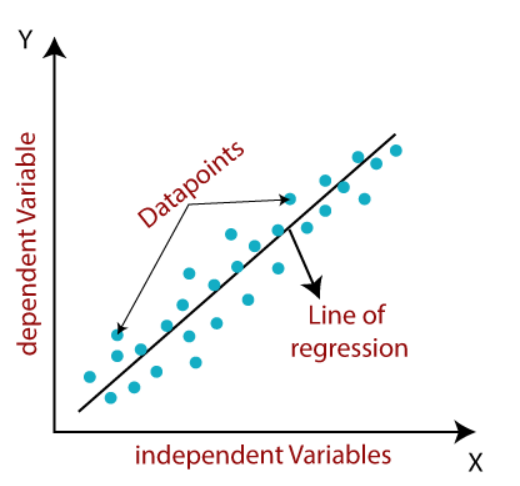
Linear regression algorithm shows a linear relationship between a dependent (y) and one or more independent (y) variables, hence called linear regression. Since linear regression shows the linear relationship, which means it finds how the value of the dependent variable is changing according to the value of the independent variable.

The linear regression model provides a sloped straight line representing the relationship between the variables.

This equation takes the form of:

y = b0 + b1x1 + b2x2 + ... + bn\*xn

where y is the dependent variable, x1, x2, ..., xn are the independent variables, and b0, b1, b2, ..., bn are the coefficients of the linear equation.



**Advantages:**

1. Simplicity: Linear regression is a simple and easy-to-understand method. The linear equation can be easily interpreted and used to make predictions or to understand the relationship between the dependent variable and independent variable(s).
2. Flexibility: Linear regression can be applied to a wide range of data types, including continuous, discrete, binary, and categorical data.
3. Efficiency: Linear regression is computationally efficient and can handle large datasets. It is also relatively fast to compute, making it suitable for real-time applications.
4. Interpretable results: The coefficients of the linear equation can be easily interpreted to determine the direction and strength of the relationship between the dependent variable and independent variable(s).

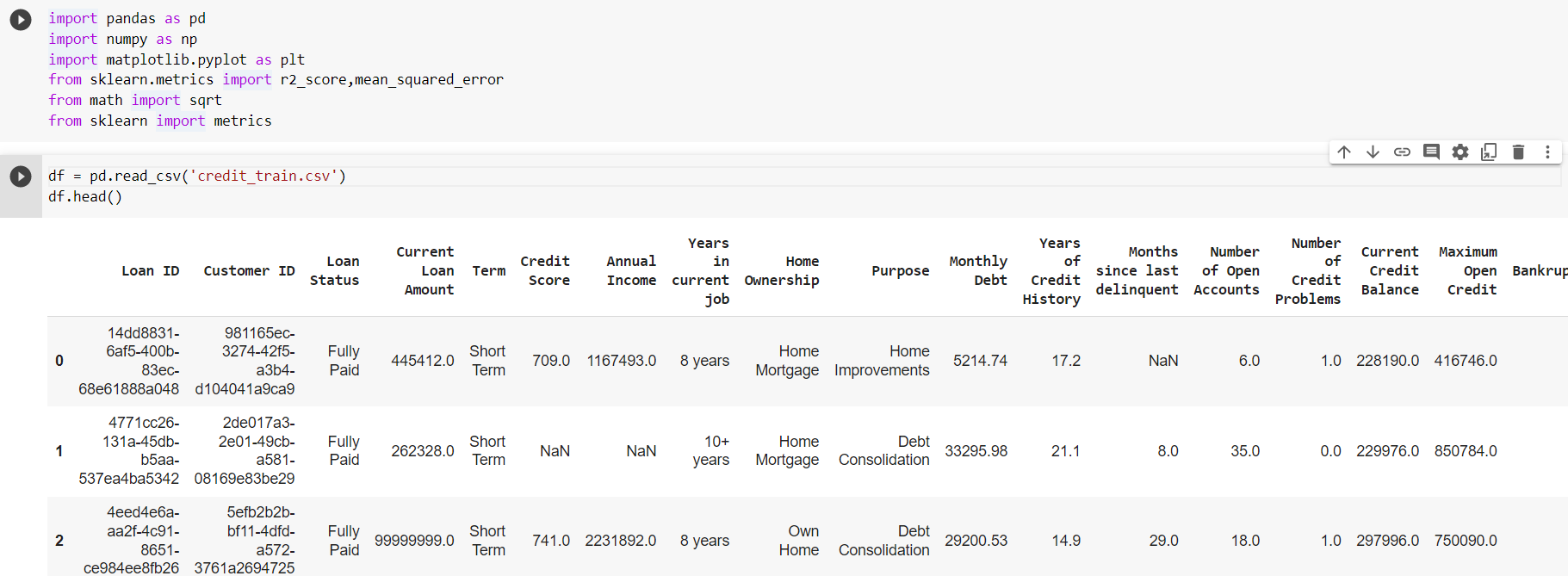
**Disadvantages:**

1. Limited flexibility: Linear regression assumes a linear relationship between the dependent variable and independent variable(s). If this assumption is not met, the results may be inaccurate.
2. Sensitivity to outliers: Linear regression is sensitive to outliers, which can distort the results of the analysis. Outliers can have a large impact on the regression line and can lead to incorrect conclusions.
3. Multicollinearity: Multicollinearity can occur when there is a high correlation between two or more independent variables. Multicollinearity can make it difficult to interpret the coefficients of the independent variables and can lead to incorrect conclusions.
4. Non-constant variance: Linear regression assumes that the variance of the residuals is constant across all levels of the independent variables. If this assumption is not met, the results may be inaccurate.

**IMPLEMENTATION:**

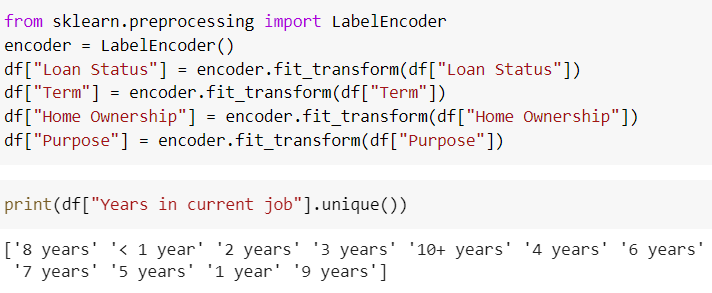
1. **USING AN IN-BUILT PYTHON FUNCTION**

**Data pre-processing**





Converting categorical variables into integers

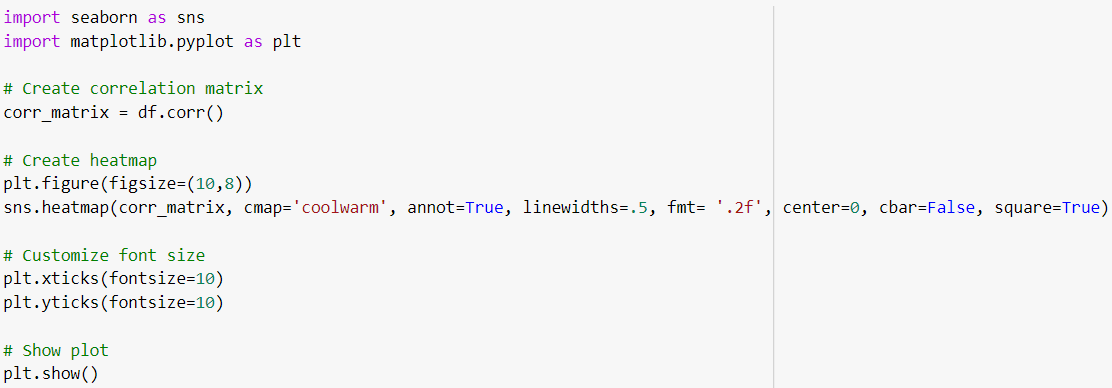


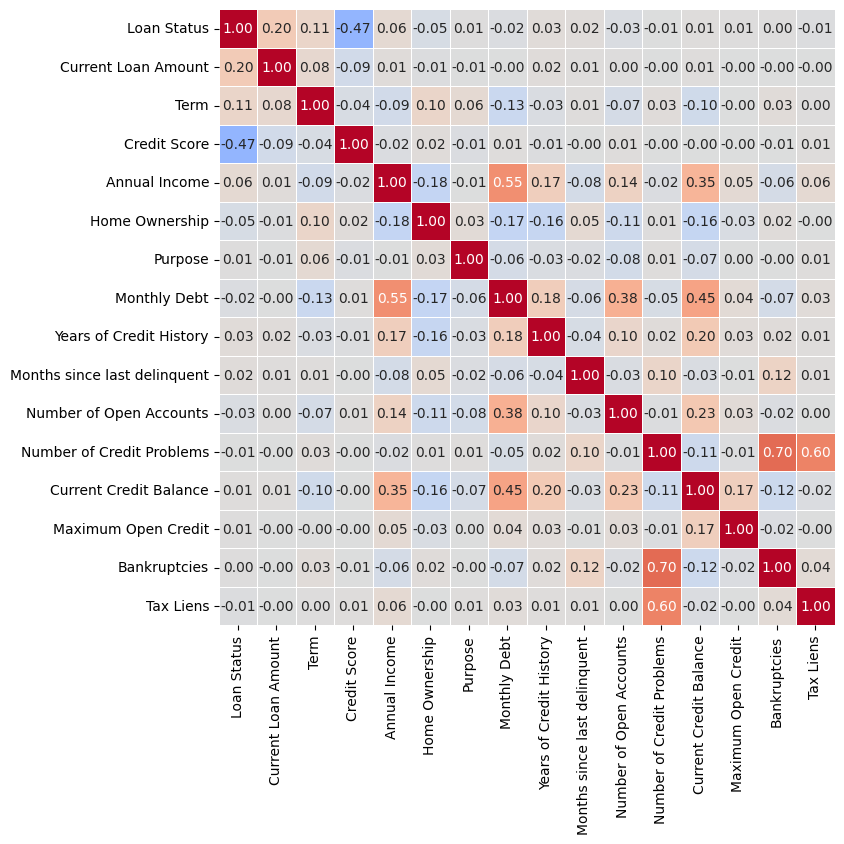


Dropping the useless columns

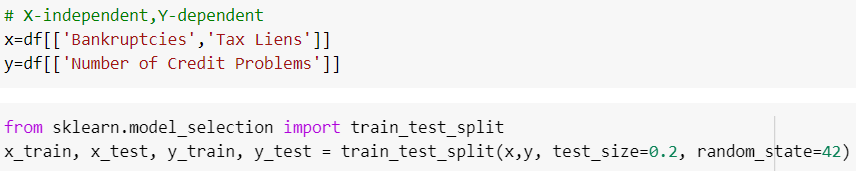


Plotting the heatmap to understand the correlation between variables



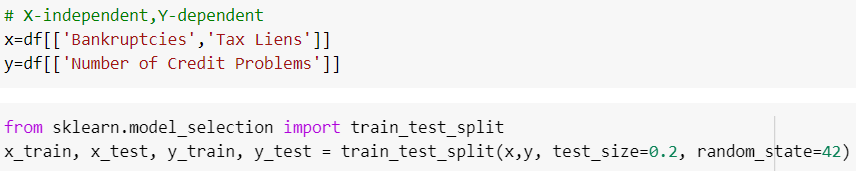


**Selecting the dependent variable and independent variables**

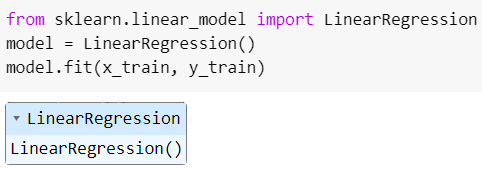


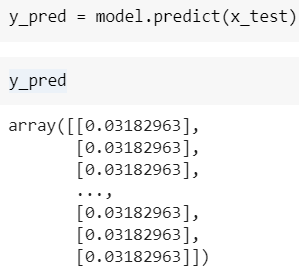
**Build the model**

Split the dataset into train and test datasets

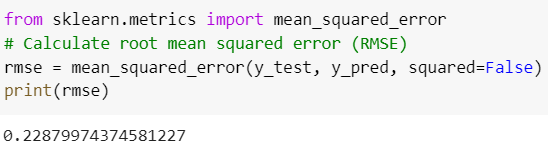


Fit to model





**Calculating the rmse value**



1. **USING USER-DEFINED FUNCTION**

**import pandas as pd**

**import numpy as np**

**class LinearRegression:**

**"""**

**Linear Regression implementation using the normal equation.**

**"""**

**def \_\_init\_\_(self):**

**self.theta = None**

**def fit(self, X, y):**

**"""**

**Train the linear regression model on the input data and targets.**

**"""**

**# Add a column of 1s for the bias term**

**X\_b = np.c\_[np.ones((len(X), 1)), X]**

**# Calculate the normal equation**

**theta = np.linalg.inv(X\_b.T.dot(X\_b)).dot(X\_b.T).dot(y)**

**self.theta = theta**

**def predict(self, X):**

**"""**

**Predict the target values for new input data.**

**"""**

**# Add a column of 1s for the bias term**

**X\_b = np.c\_[np.ones((len(X), 1)), X]**

**# Predict the target values**

**y\_pred = X\_b.dot(self.theta)**

**return y\_pred**

**# Split the data into features and target**

**X = df2[['Bankruptcies','Tax Liens']]**

**y = df2['Number of Credit Problems']**

**# Split the data into training and testing sets**

**from sklearn.model\_selection import train\_test\_split**

**X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.2, random\_state=42)**

**# Train the linear regression model**

**model = LinearRegression()**

**model.fit(X\_train, y\_train)**

**# Predict the target values for the testing set**

**y\_pred = model.predict(X\_test)**

**from sklearn.metrics import mean\_squared\_error**

**# Calculate root mean squared error (RMSE)**

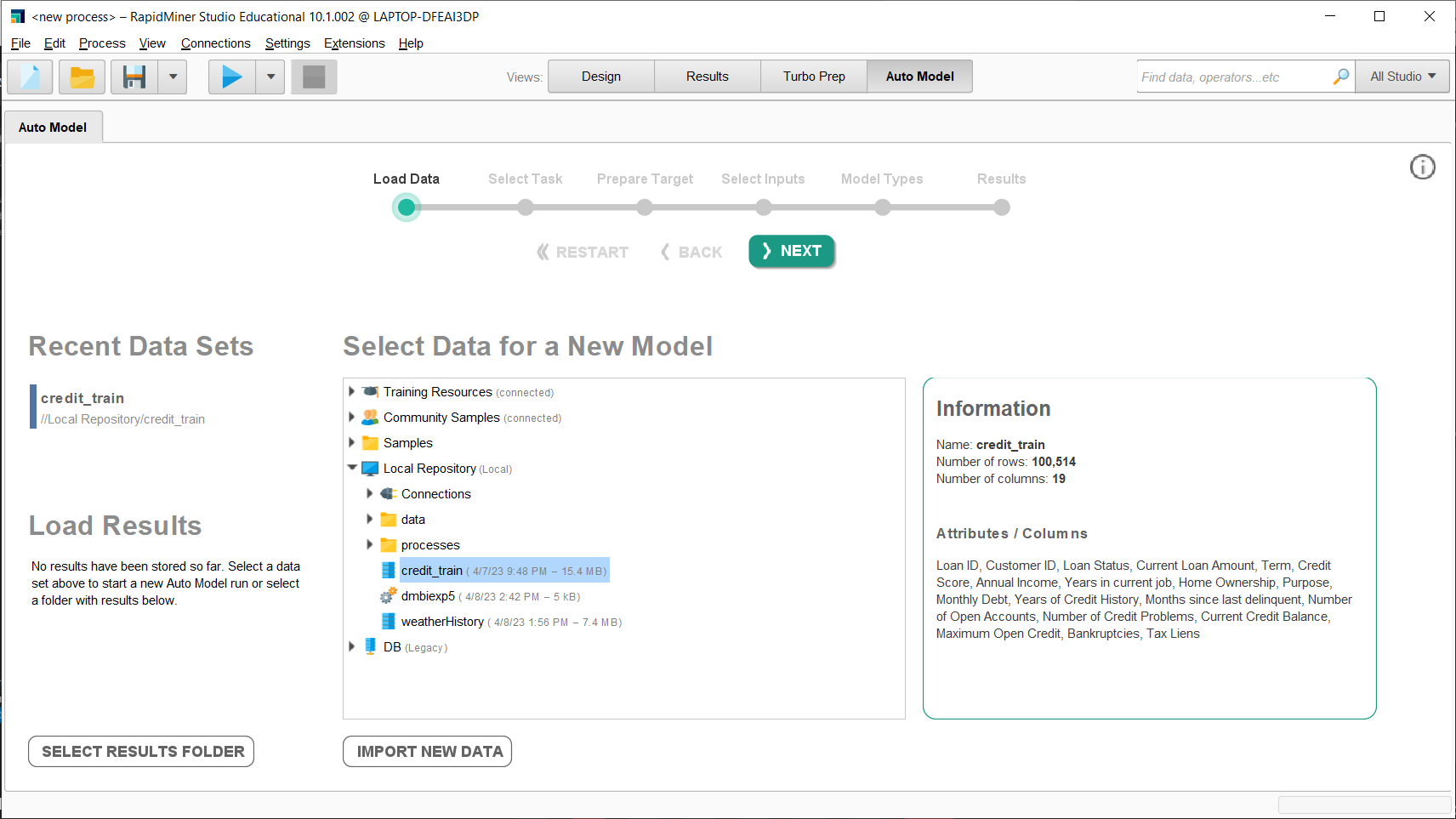
**rmse = mean\_squared\_error(y\_test, y\_pred, squared=False)**

**print(rmse)**

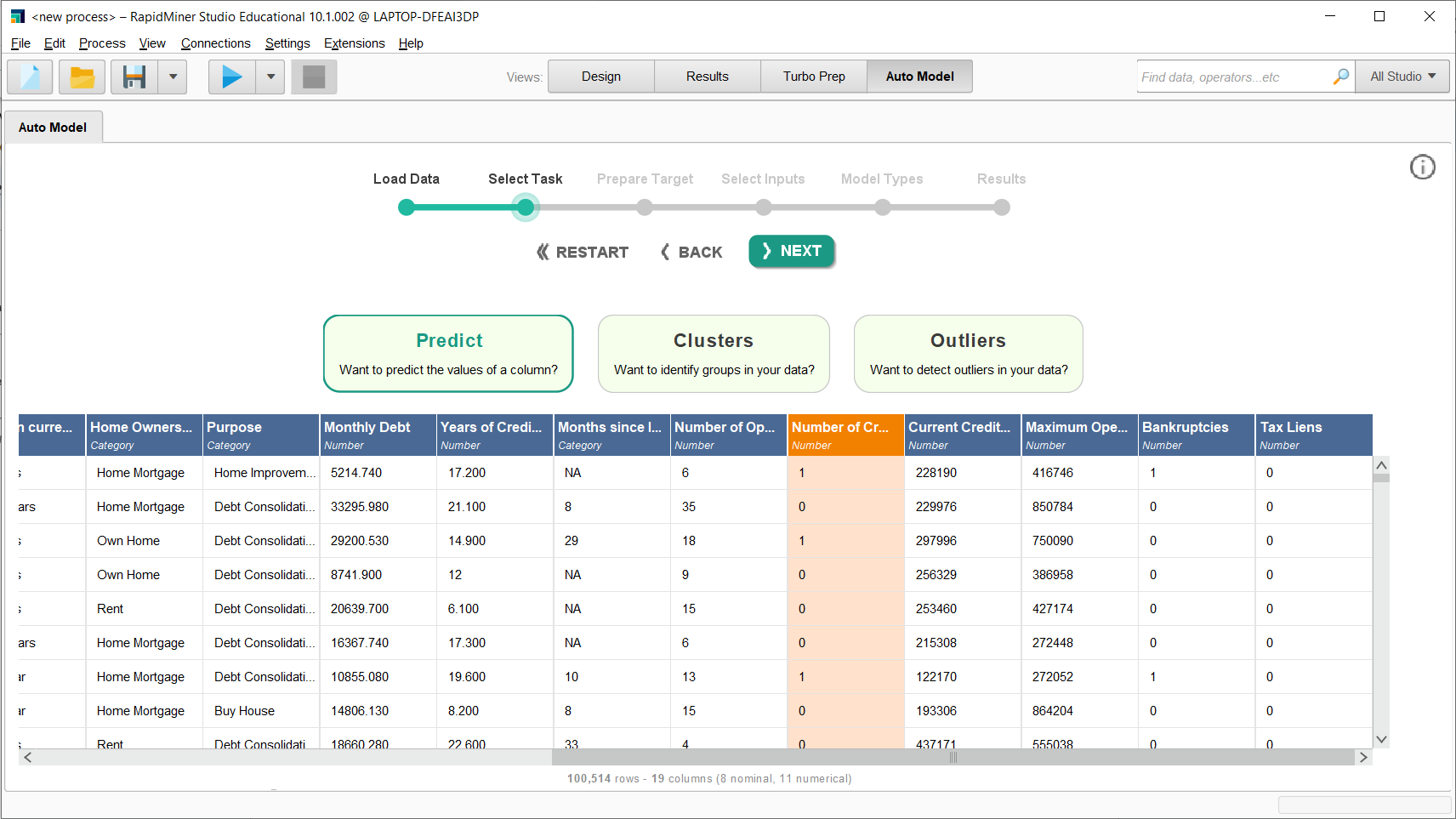
**OUTPUT:**



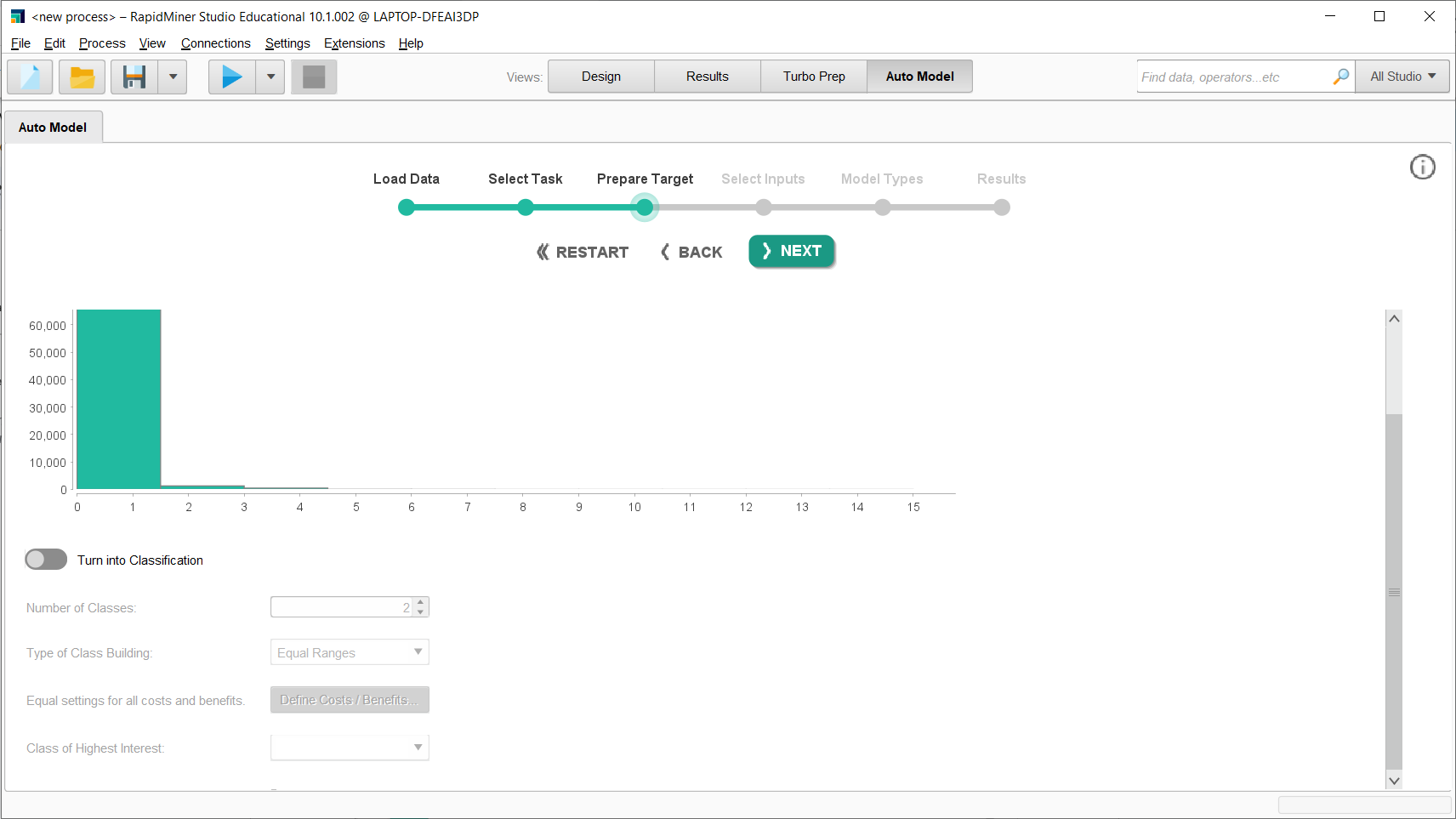
1. **USING RAPID MINER**
2. Select the ‘auto model’ option and import data.

****

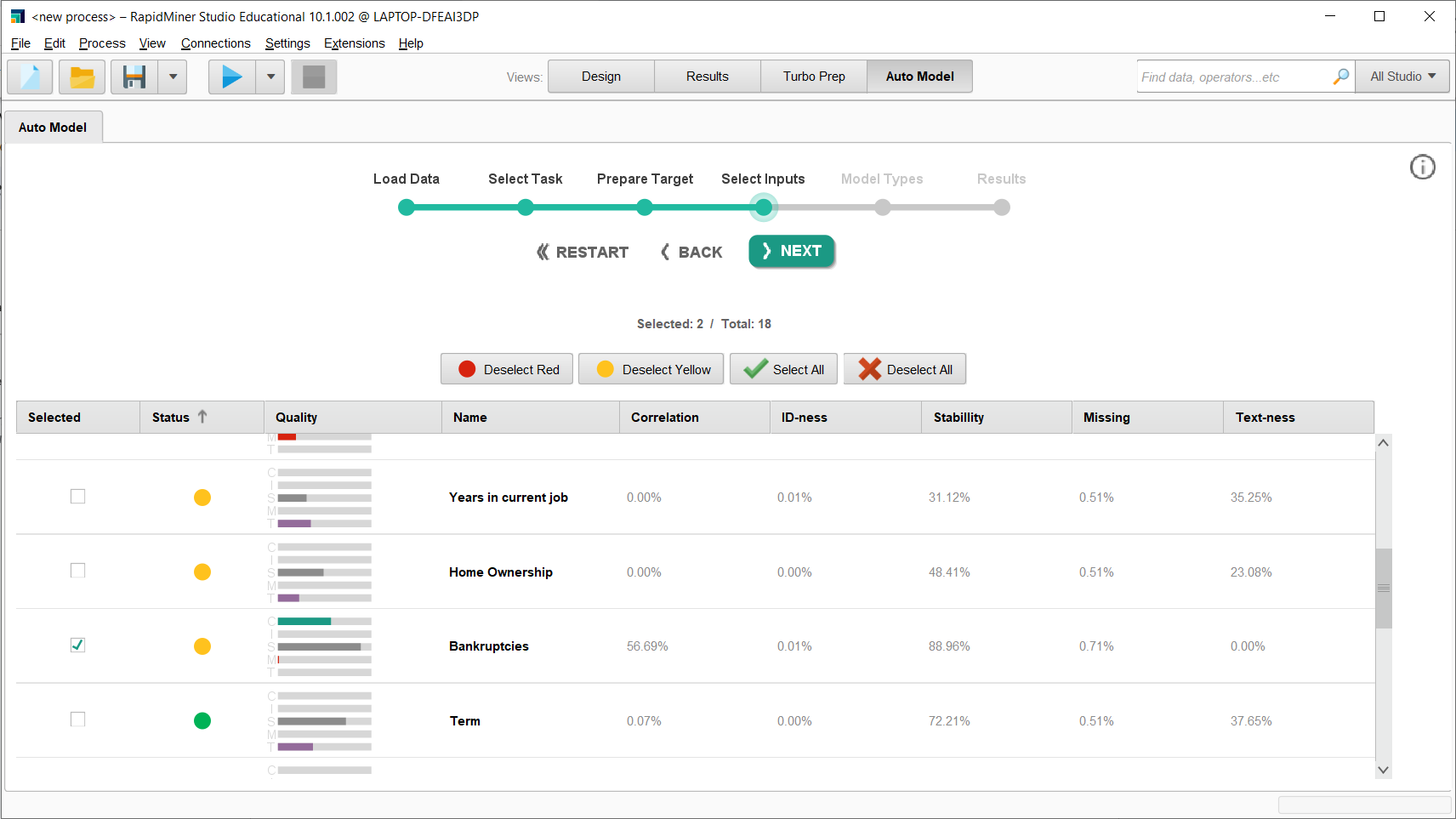
1. Select the type of task to be performed and the feature to be predicted.

****

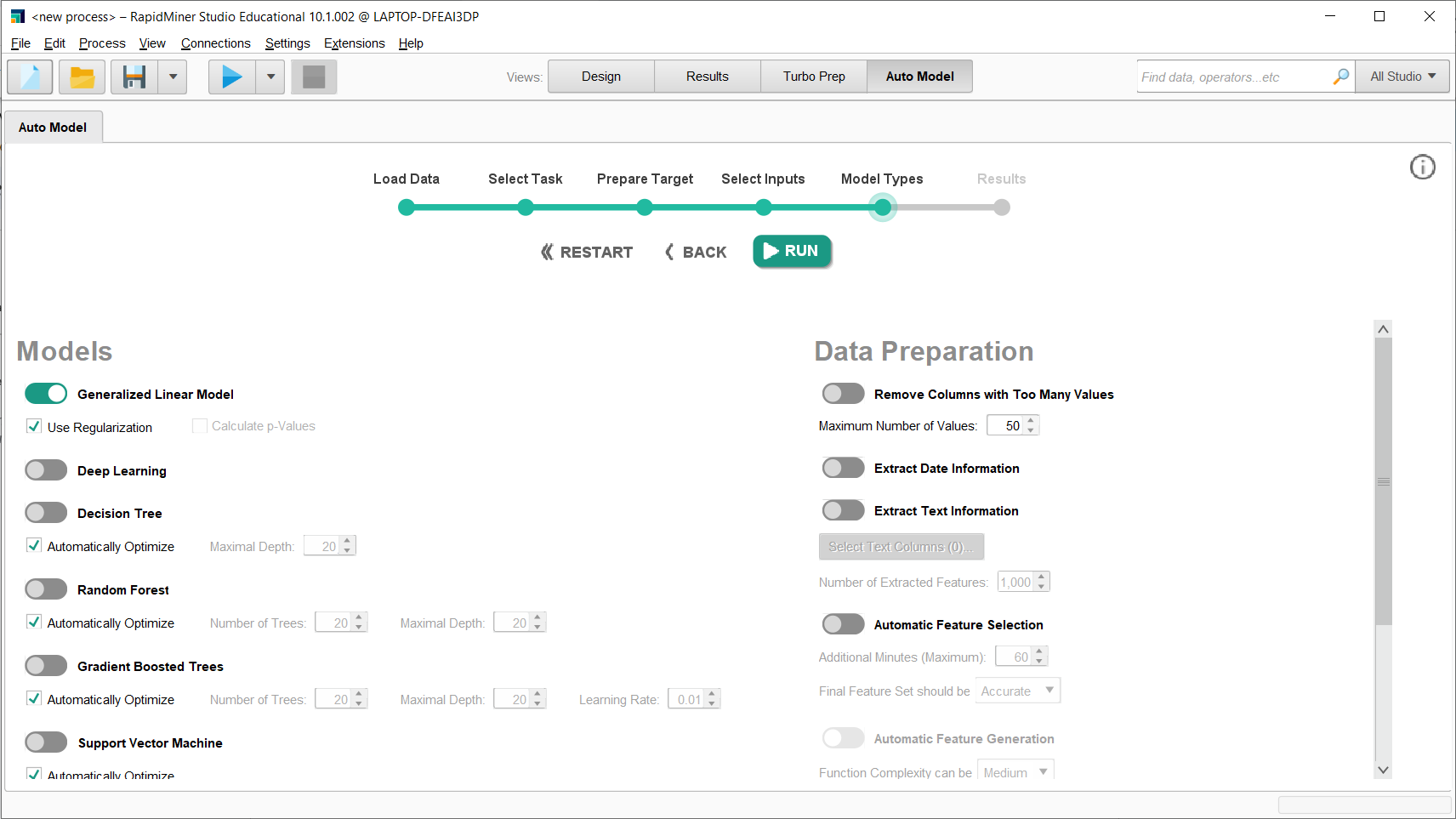
1. Analyze the feature.

****

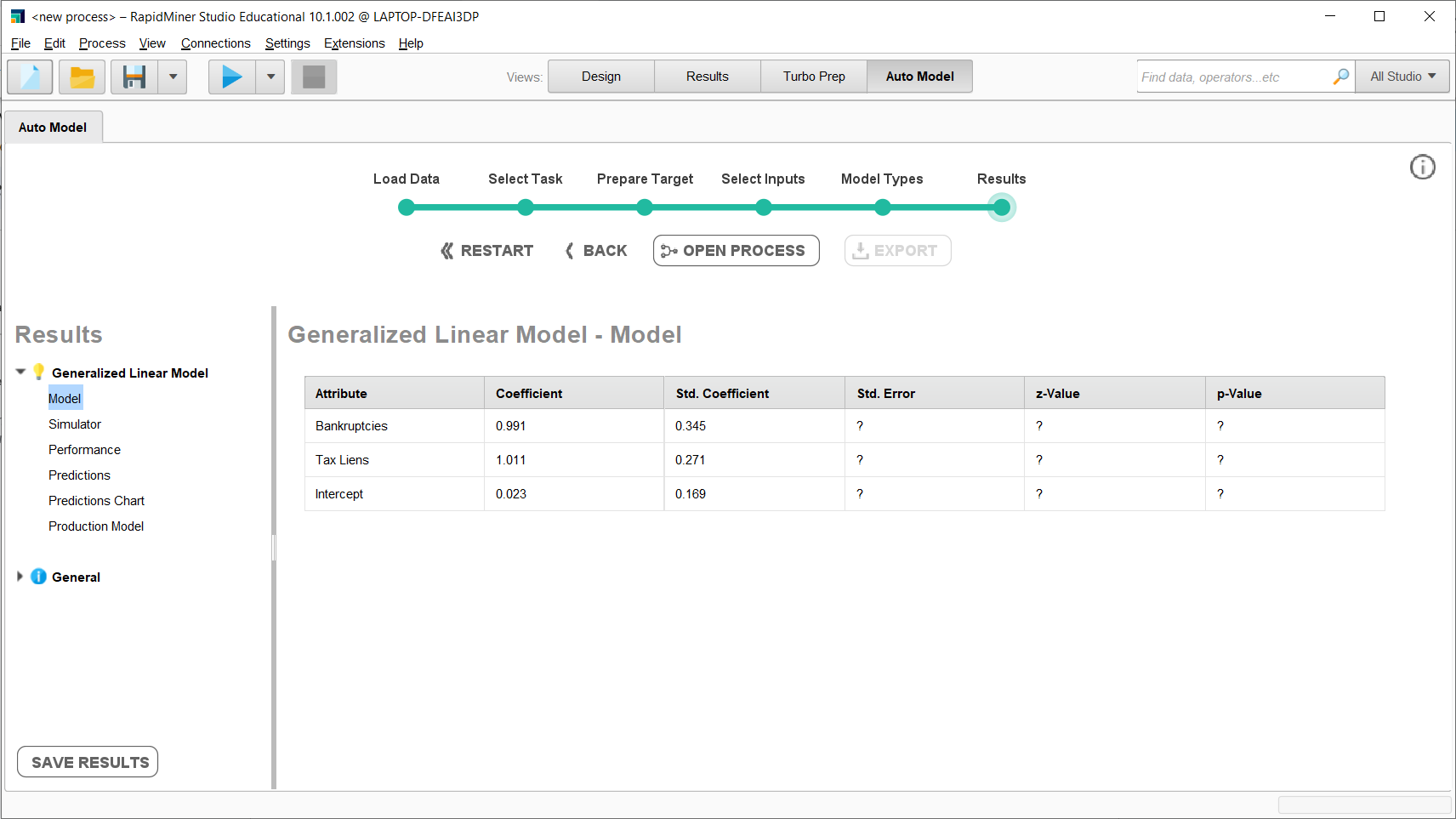
1. Select the input features by considering the factors involved.

****

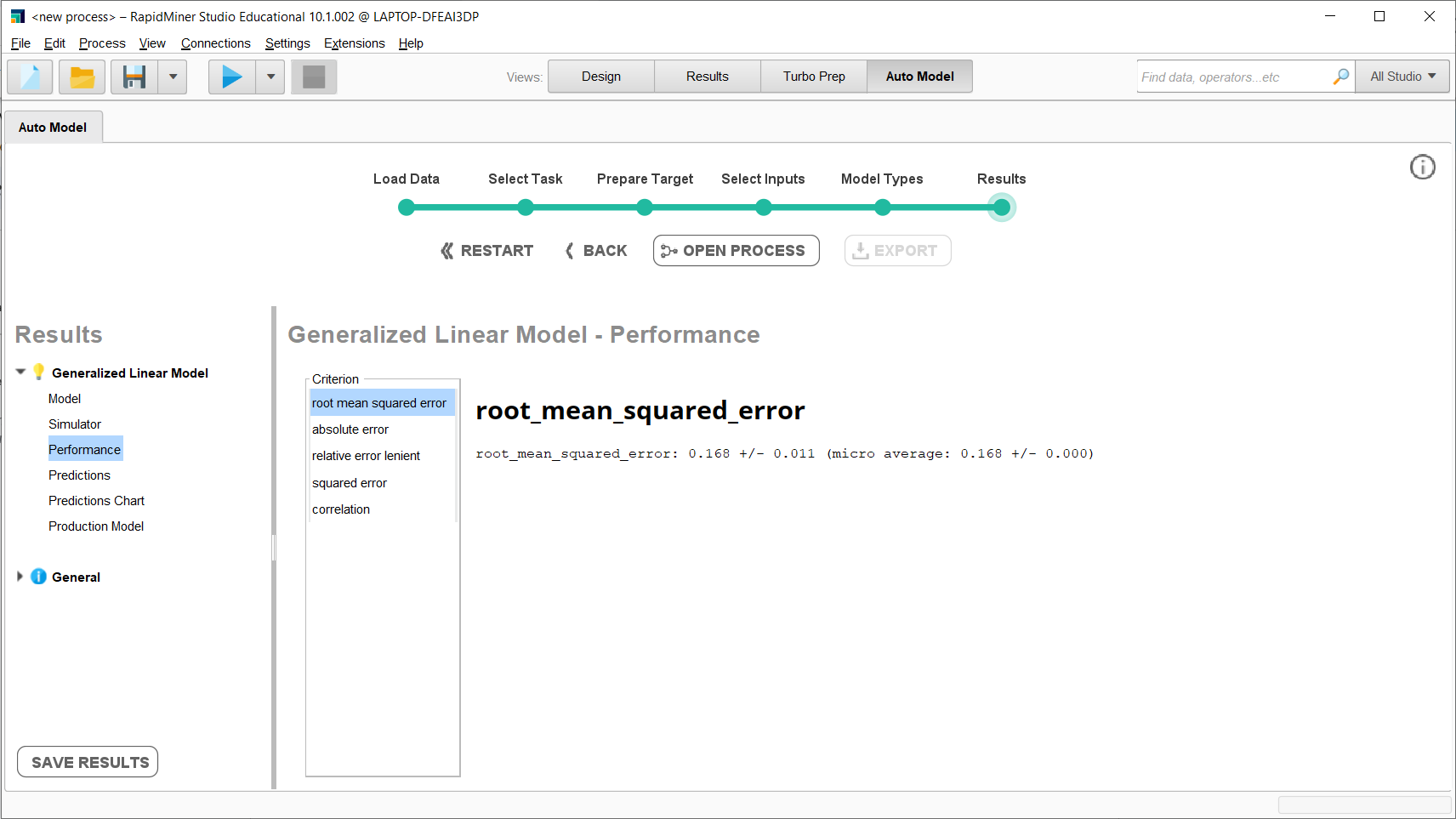
1. Select the model ‘Generalized Linear Model’.

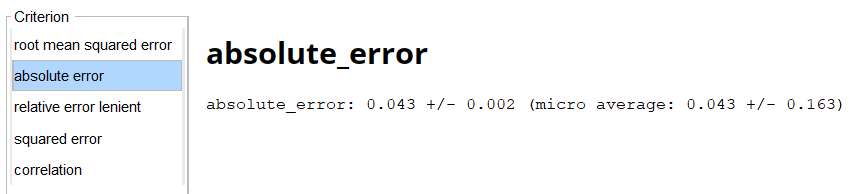
****

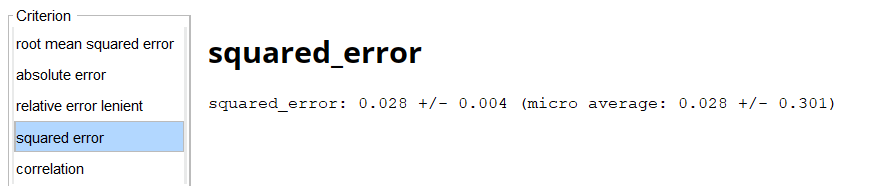
1. Analyze the results.

****

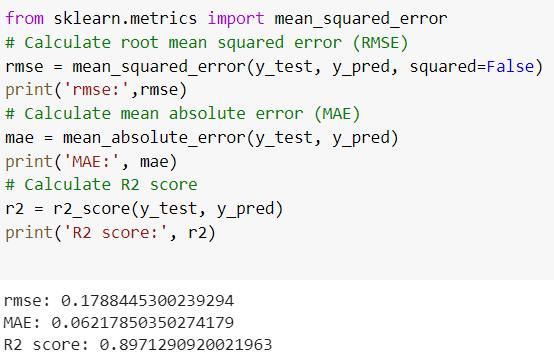
**Performance:**

****

****

****

**In Python:**

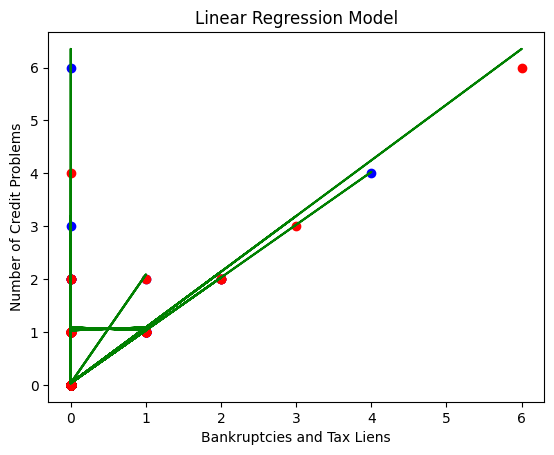


**COMPARISON OF ACCURACY OBTAINED BY ALL 3 METHODS**

|  | Inbuilt python function | User-defined function | Rapid Miner |
| --- | --- | --- | --- |
| RMSE | 0.228 | 0.228 | 0.168 |
| mae | 0.06 | 0.06 | 0.043 |
| r2\_score | 0.89 | 0.89 | 0.028 |

**Graph:**





**Equation of the line:**

**y = 0.0338 + 0.998\*Bankruptcies + 1.05294\*Tax Liens**

**CONCLUSION:**

In this experiment, we have implemented a Generalized Linear Model using Rapid Miner, Python Library, and a self-defined function, and the best accuracy was obtained by the Rapid Miner, followed by an inbuilt Python function and user-defined function.