Aim : Experiment to implement a regression model using Rapid Miner and Python.

To Do:

1. Preprocess data. Split data into train and test set
2. Build Regression model using inbuilt library function on training data
3. Calculate metrics based on test data using inbuilt function
4. Build Regression model using a function defined by a student(model to be built using the same training data).
5. Calculate metrics based on test data using inbuilt function
6. Compare the results of all three ways of implementation.(Rapid Miner, Python Library)

Theory:

Regression is a method for understanding the relationship between independent variables or features and a dependent variable or outcome. Outcomes can then be predicted once the relationship between independent and dependent variables has been estimated. Regression is a field of study in statistics which forms a key part of forecast models in machine learning. It's used as an approach to predict continuous outcomes in predictive modeling, so has utility in forecasting and predicting outcomes from data. Machine learning regression generally involves plotting a line of best fit through the data points. The distance between each point and the line is minimized to achieve the best fit line.

Types of Regression

There are various types of regressions which are used in data science and machine learning. Each type has its own importance on different scenarios, but at the core, all the regression methods analyze the effect of the independent variable on dependent variables. Here we are discussing some important types of regression which are given below:

1. Linear Regression
2. Logistic Regression
3. Polynomial Regression
4. Support Vector Regression
5. Decision Tree Regression
6. Random Forest Regression
7. Ridge Regression
8. Lasso Regression:

Preprocessing Data

Preprocessing data is an important step in preparing data for a regression analysis. Here are some common preprocessing steps for regression:

- Handling missing data
- Encoding categorical variables
- Feature scaling
- Removing outliers
- Feature selection
- Splitting data

Evaluating the performance of regression models:

Evaluating the performance of a regression model is crucial to determine its accuracy and usefulness in predicting outcomes. Here are some common techniques for evaluating regression models:

- Mean squared error (MSE): MSE measures the average squared difference between the predicted values and the actual values. The lower the MSE, the better the model.
- Root mean squared error (RMSE): RMSE is the square root of the MSE and provides a measure of the average error in the same units as the target variable.
- R-squared (R^2): R-squared measures the proportion of variance in the target variable that is explained by the model. It ranges from 0 to 1, with higher values indicating a better fit.
- Adjusted R-squared: Adjusted R-squared adjusts for the number of predictors in the model and is a more accurate measure of the model's goodness-of-fit.
- Residual plots: Residual plots can help identify patterns in the residuals (the differences between predicted and actual values) that may suggest the model is misspecified or that there is heteroscedasticity (unequal variances) in the errors.
- Cross-validation: Cross-validation involves dividing the data into training and validation sets multiple times and testing the model on different validation sets. This can help assess the model's generalization performance.

Overall, evaluating a regression model involves measuring its accuracy using metrics such as MSE, RMSE, R-squared, and adjusted R-squared, examining residual plots, and performing cross-validation. The choice of evaluation metrics will depend on the specific goals of the analysis and the nature of the data.

Rapid Miner

RapidMiner is a powerful and easy-to-use data science platform that allows users to perform various data mining tasks, including data preparation, data integration, predictive modeling, and machine learning. RapidMiner provides a graphical user interface (GUI) that enables users to create and customize workflows visually. RapidMiner has a drag-and-drop interface that allows users to easily build workflows, import data from various sources, and perform data analysis.. RapidMiner is widely used in various industries, including finance, healthcare, retail, and marketing. It has a large community of users and provides extensive documentation and support resources to help users get started with the platform. RapidMiner also offers enterprise-grade features such as collaboration, automation, and integration with cloud-based services.

Linear Regression

Linear regression is the type of regression that forms a relationship between the target variable and one or more independent variables utilizing a straight line. The given equation represents the equation of linear regression

$Y = a + b*X + e$.

Where,

a represents the intercept

b represents the slope of the regression line

e represents the error

X and Y represent the predictor and target variables, respectively.

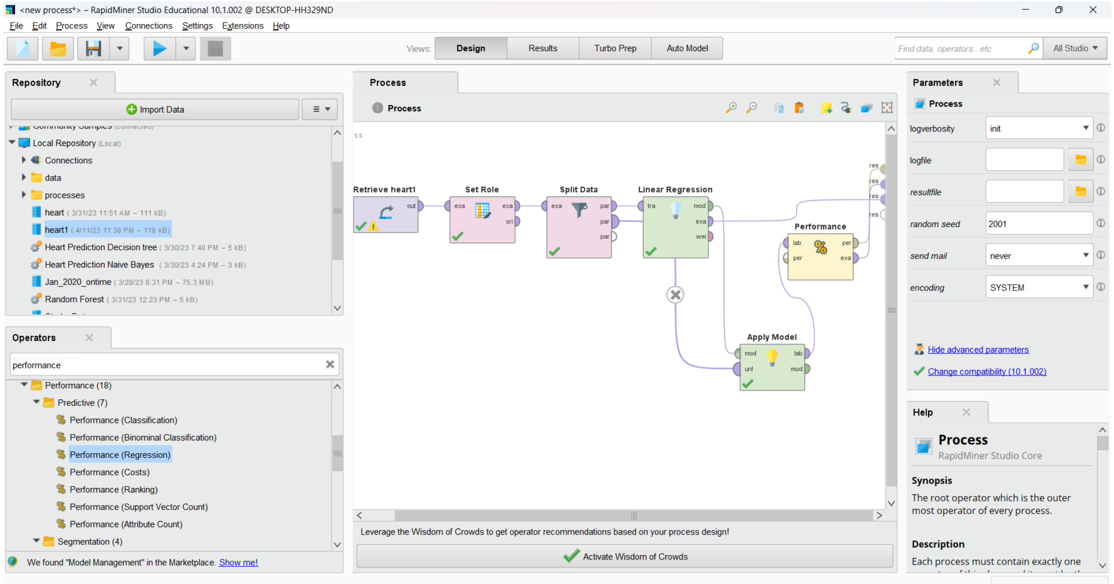
If X is made up of more than one variable, termed as multiple linear equations.

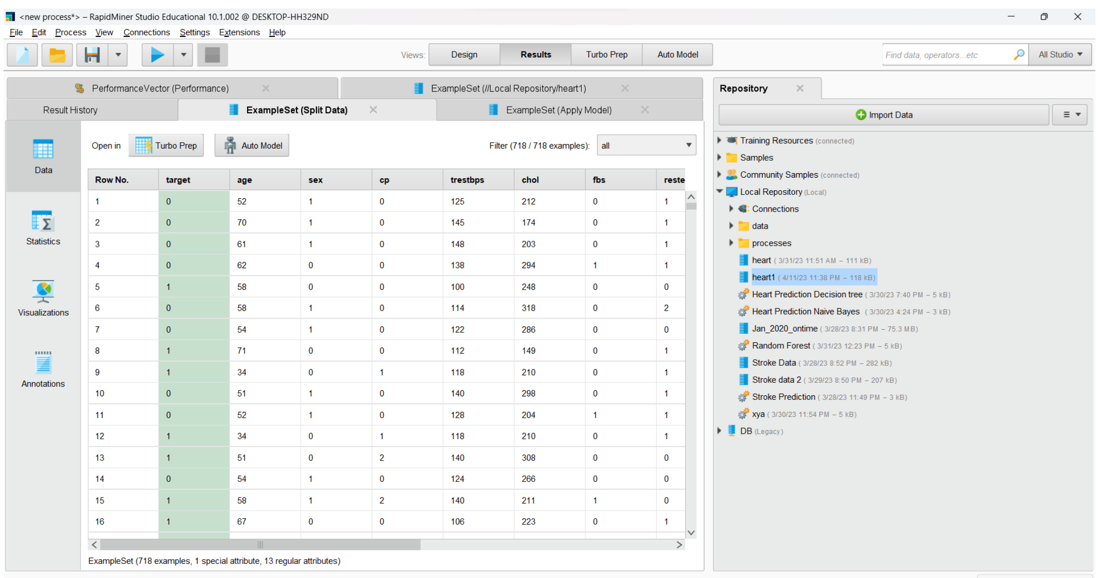
In linear regression, the best fit line is achieved utilizing the least squares method, and it minimizes the total sum of the squares of the deviations from each data point to the line of regression. Here, the positive and negative deviations do not get canceled as all the deviations are squared.

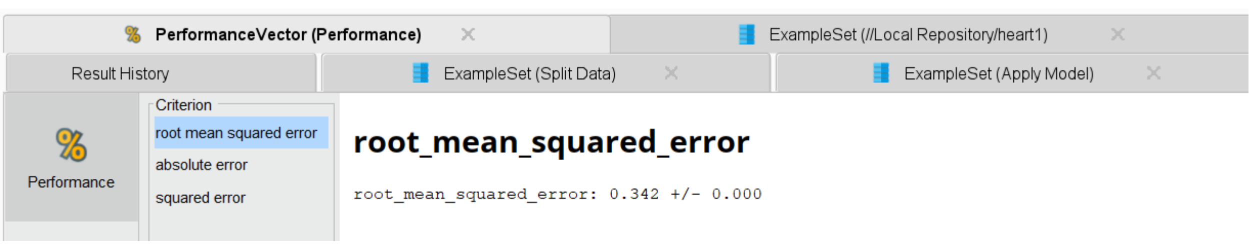
Implementation of Random Forest using Rapid Miner

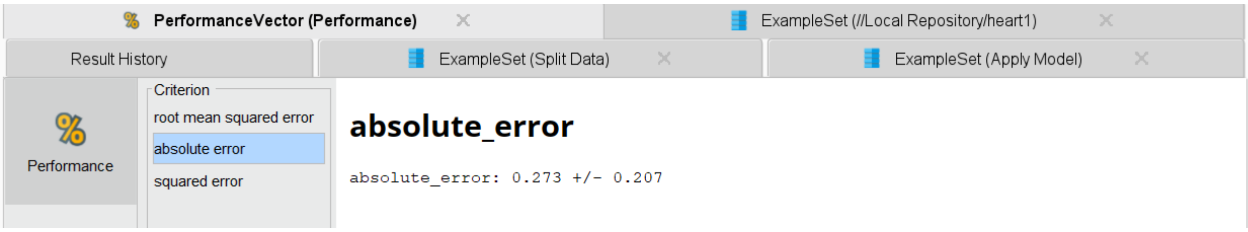
Step1: Import the required dataset
Step 2: Add set role operator by selecting the attribute as target
Step 3: Then we will split the data into train and test data as 70% and 30%
Step 4: Add Random forest operator and perform the given connections to apply model performance as shown below.

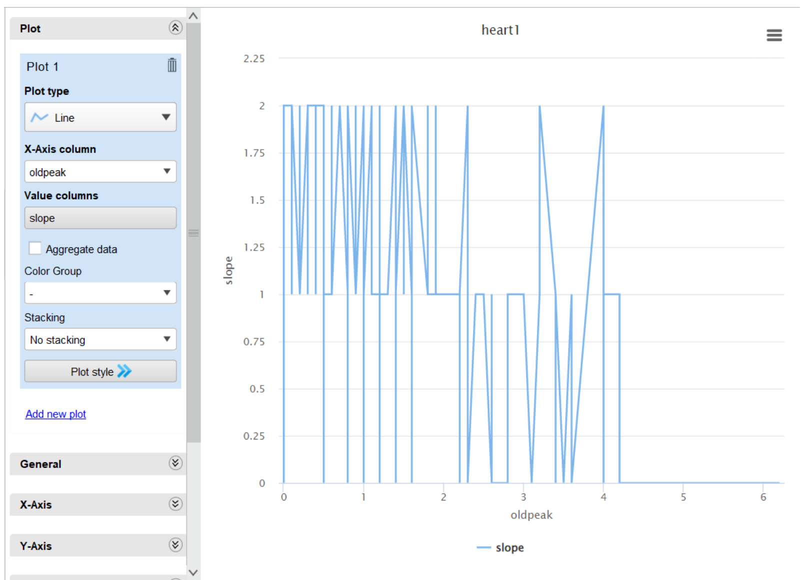
DMBI Exp-07 D15B-70











Implementation using Python

- Importing libraries and loading dataset

```python
#Import the libraries
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```python
#loading the dataset
data = pd.read_csv("heart.csv")
data
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0
4	62	0	0	138	294	1	1	106	0	1.9	1	3	2	0
...	...	...	...	...	...	...	...	...	...	...	...	...	...	...
1020	59	1	1	140	221	0	1	164	1	0.0	2	0	2	1
1021	60	1	0	125	258	0	0	141	1	2.8	1	1	3	0
1022	47	1	0	110	275	0	0	118	1	1.0	1	1	2	0
1023	50	0	0	110	254	0	0	159	0	0.0	2	0	2	1
1024	54	1	0	120	188	0	1	113	0	1.4	1	1	3	0

1025 rows × 14 columns

- Checking null values

```
data = data.fillna(0)
data
```

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	52	1	0	125	212	0	1	168	0	1.0	2	2	3	0
1	53	1	0	140	203	1	0	155	1	3.1	0	0	3	0
2	70	1	0	145	174	0	1	125	1	2.6	0	0	3	0
3	61	1	0	148	203	0	1	161	0	0.0	2	1	3	0

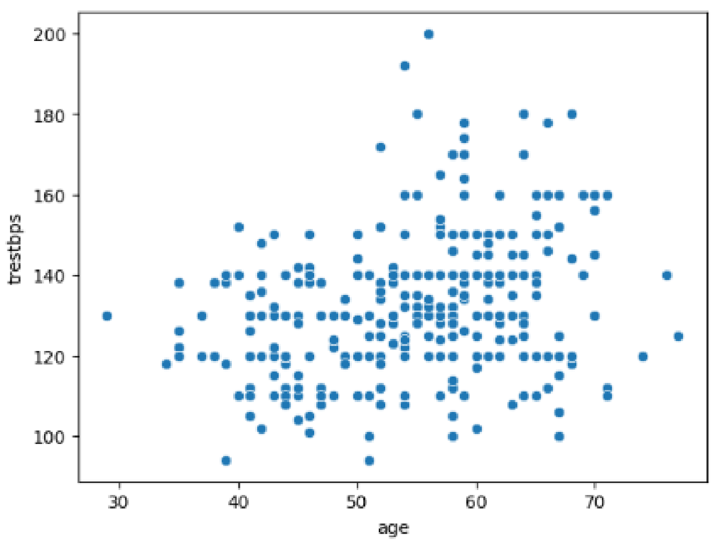
```
[8] data.isnull().sum()
```

```
age         0
sex         0
cp          0
trestbps    0
chol        0
fbs         0
restecg     0
thalach     0
exang       0
oldpeak     0
slope       0
ca          0
thal        0
target      0
dtype: int64
```

- Scatter plot for age and trestbps

```
sns.scatterplot(data=data,x='age',y='trestbps')
```

```
<Axes: xlabel='age', ylabel='trestbps'>
```



- Training the model

```
[14] from sklearn.linear_model import LinearRegression
     from sklearn.linear_model import Ridge
     from sklearn.ensemble import RandomForestRegressor
     models = {
         "Linear Regression": LinearRegression(),
         "Ridge": Ridge(),
         "Random Forest": RandomForestRegressor()

     }
```

```
[16] import pandas as pd
     import numpy as np
     from sklearn.model_selection import train_test_split
     from sklearn.tree import DecisionTreeClassifier
     from sklearn.metrics import accuracy_score
     from sklearn import tree
```

```
X=data[['age','chol','trestbps','thalach','fbs','exang','sex','slope','ca','thal','oldpeak',]]
y=data['target']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

\

- Mean square error for this model

```
[11] clf1=LinearRegression()
     clf1.fit(X_train,y_train)

     ▾ LinearRegression
     LinearRegression()
```

```
[12] from sklearn.metrics import mean_squared_error
     y_pred1=clf1.predict(X_test)
     score=mean_squared_error(y_pred1,y_test)
     print(score)

     0.1612465565963523
```

Implementation using Self defined function

```
from sklearn.base import BaseEstimator, RegressorMixin
from sklearn.metrics import mean_squared_error

class LinearRegression(BaseEstimator, RegressorMixin):
  def __init__(self, parameter1=1, parameter2=2):
    self.parameter1 = parameter1
    self.parameter2 = parameter2
```

```
  def fit(self, X, y):
    # Implement your own fitting method here
    # This example just calculates the mean of y
    self.y_mean = np.mean(y)
    return self
  def predict(self, X):
    # Implement your own prediction method here
    # This example just returns the mean of y for all instances
    y_pred = np.full((X.shape[0],), self.y_mean)
    return y_pred
  def score(self, X, y):
    # Implement your own scoring method here
    # This example just calculates the mean squared error
    y_pred = self.predict(X)
    mse = mean_squared_error(y, y_pred)
    return -mse # return negative MSE for use with GridSearchCV's default
scoring
```

```
[15] custom_Reg = LinearRegression(parameter1=0.5,parameter2=0.1)
     custom_Reg.fit(X_train,y_train)

     y_pred = custom_Reg.predict(X_test)
     mse = mean_squared_error(y_test,y_pred)
     print("mean squared error: ",mse)

     mean squared error:  0.250174003569304
```

Comparison for Linear Regression :-

Method used	**Mean Squared Error**
Rapid Miner	0.12
Python	0.1612
Self Defined function	0.250

Conclusion: Thus we compared the mean square error of each method. The performance of the Rapid Miner method is the best followed by Self Defined and Python since Rapid Miner has the least mean squared error. We successfully implemented a Linear regression model using Rapid Miner and Python.