Support Vector Regression

**Short note about Support Vector Regression**:

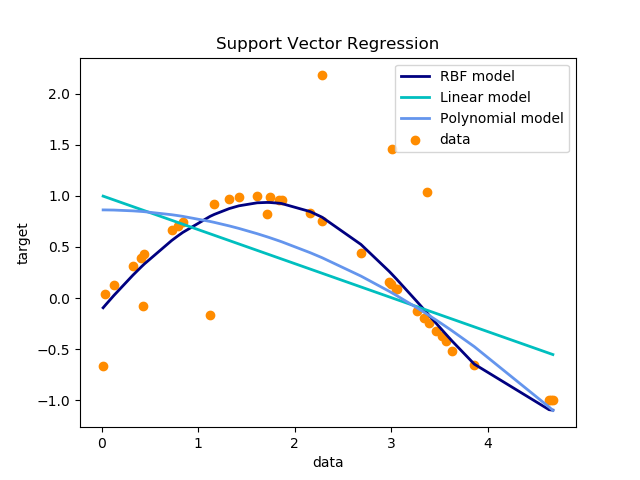
Support Vector Regression(SVR) is a supervised machine learning algorithm which can be used for both classification or Regression problems.

Usually used for classification.

**HyperPlane-**

Given 2 or more labeled classes of data,it acts as a discriminative classifier,which is known as optimal hyperplane that seperates all the classes.

Hyperplane is a linear decision surface that splits the space into two parts.



Decode Complex Algorithm

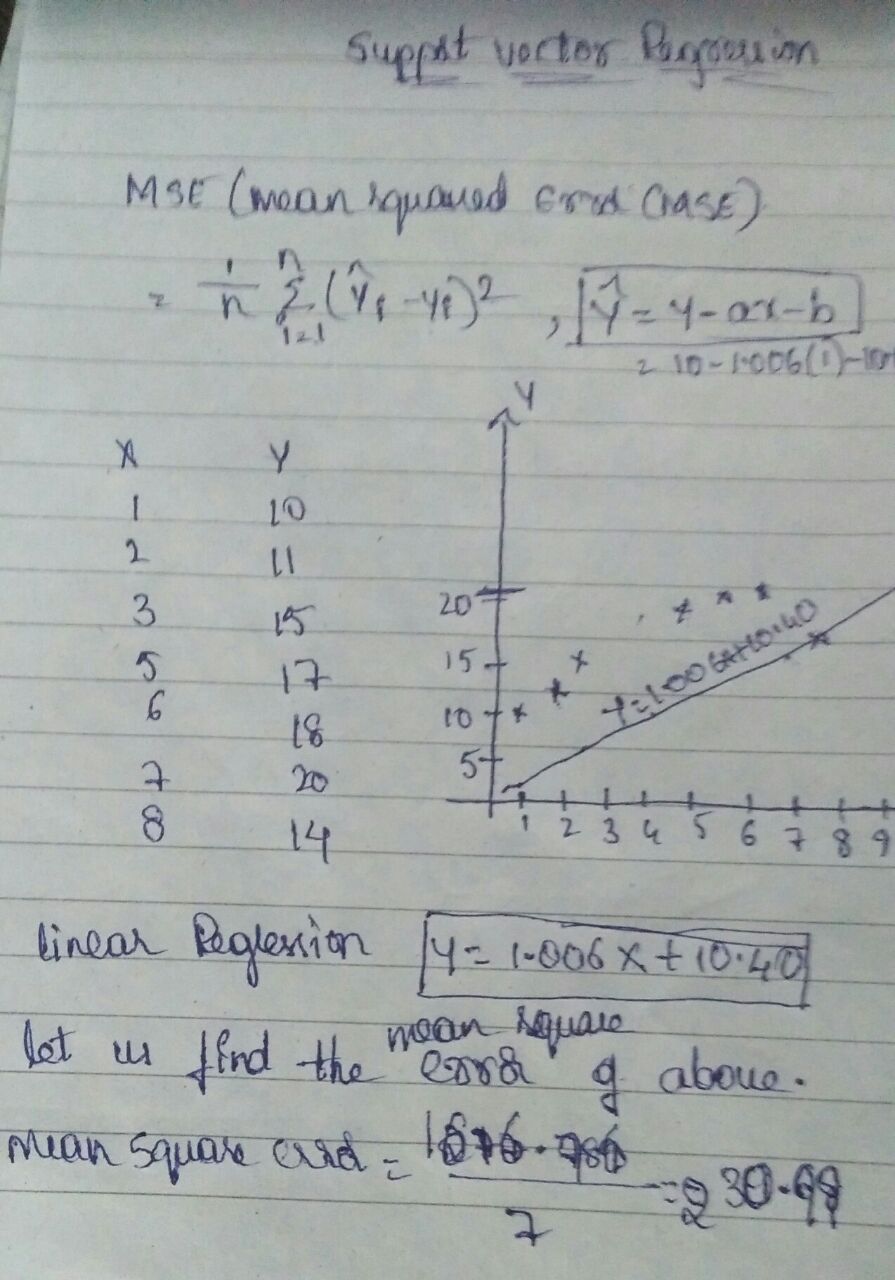
Step: 1(linear Model)

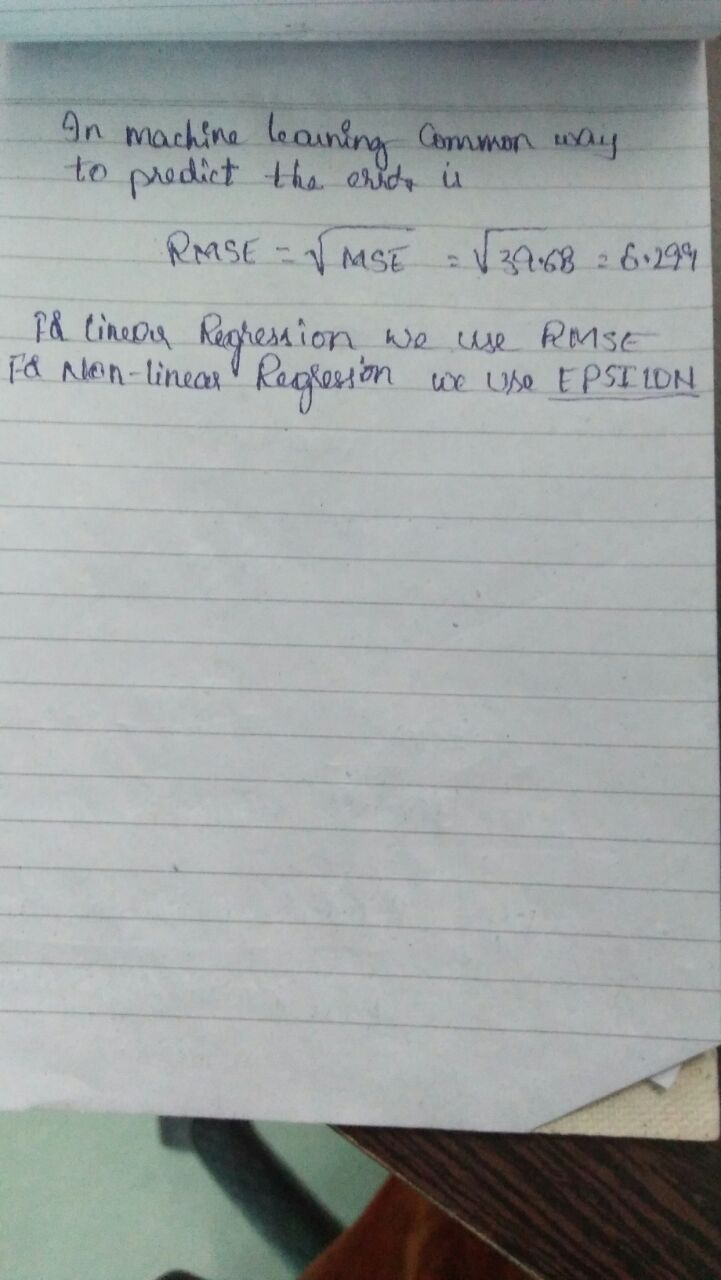
|  |  |
| --- | --- |
| **X** | **Y** |
| 1 | 10 |
| 2 | 11 |
| 3 | 15 |
| 5 | 17 |
| 6 | 18 |
| 7 | 20 |
| 8 | 14 |

Formulae-From linear regression we get the value of

|  |
| --- |
| Y=ax+b |
| Y^=y-ax-b | |
| Error value(mean square error)=TOTAL SUM((Y^-Y)^2)/n | |
| Where n=no.of samples | |
| Root mean square error =sqrt(mean square error) | |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **X** | **Y** | **Y^** | **y^-y(Error)** | **(y^-y)2** |
| 1 | 10 | -1.406 | -11.406 | 130.0968 |
| 2 | 11 | -1.412 | -12.412 | 154.0577 |
| 3 | 15 | 1.582 | -13.418 | 180.0427 |
| 5 | 17 | 1.57 | -15.43 | 238.0849 |
| 6 | 18 | 1.564 | -16.436 | 270.1421 |
| 7 | 20 | 2.558 | -17.442 | 304.2234 |
| 8 | 14 | -4.448 | -18.448 | 340.3287 |
| Total Mean Square Error | | | | 1616.976 |

w

****

**Case Study-**

**UseCases-**

1. Finding Age and immunity
2. Finding temperature vs. Number of cones sold at ice cream store
3. Finding Population vs Food consumption
4. Finding quantity with yield
5. Finding the response of product based on features.
6. Finding the chances to get king while playing cards.
7. Finding obese person with adiposity prone to heart disease
8. Finding Product Price with respect to other vendors
9. Determining the chances to win cricket match .
10. Determining the chances of getting Jobs after Completing Graduation.
11. Speed and distance relationship
12. Finding rate of growth of the economy of a Institution

**R Code**-

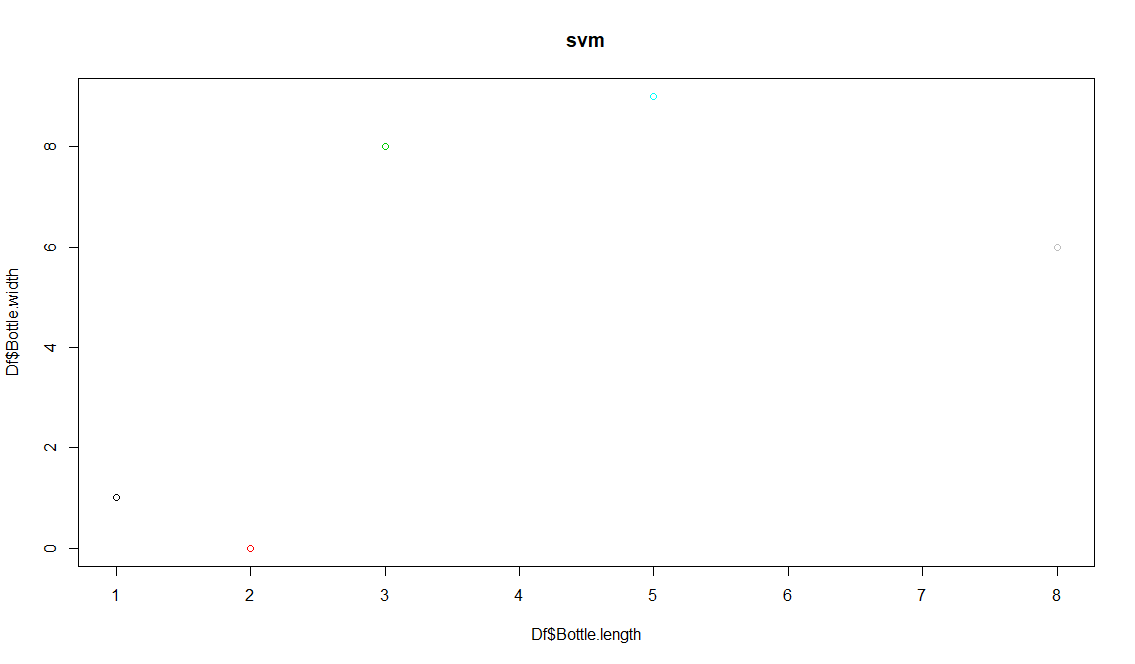
Df=data.frame(Bottle.length=c(1,2,3,5,8),Bottle.width=c(1,0,8,9,6))

install.packages("e1071")

library(e1071)

plot(Df)

plot(Df$Bottle.length,Df$Bottle.width,col=Df$Bottle.length,main="svm")



s=sample(1,1)

> svm.fit=svm(Df$Bottle.length ~Df$Bottle.width,data=Df,kernal="linear",cost=0.1,scale=FALSE,main="SVM")

> plot(svm.fit,Df)

> install.packages("e1071")

> s=sample(9,6)

> s

[1] 2 9 4 8 5 1

> svmfit=svm(Df$Bottle.width ~.,data=Df,kernal="linear",cost=10,scale=FALSE)

> plot(svmfit,Df)

> svmfit$index

[1] 1 2 3 4 5

> summary(svmfit)

Call:

svm(formula = Df$Bottle.width ~ ., data = Df, kernal = "linear", cost = 10,

scale = FALSE)

Parameters:

SVM-Type: eps-regression

SVM-Kernel: radial

cost: 10

gamma: 1

epsilon: 0.1

Number of Support Vectors: 5

**Python**

# Importing the libraries

import numpy as np

import matplotlib.pyplot as plt

import pandas as pd

from sklearn.cross\_validation import train\_test\_split

# Importing the dataset

SVR = pd.read\_csv("C:\\Users\\Rama\\Documents\\bank data\\bank.csv")

print(SVR.head())

print(SVR.tail())

SVR

X = SVR.iloc[0:50,:1].values

y = SVR.iloc[0:50,8].values

print(X)

# Splitting the dataset into the Training set and Test set

from sklearn.cross\_validation import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size = 0.25)

# Feature Scaling

from sklearn.preprocessing import StandardScaler

sc = StandardScaler()

X\_train = sc.fit\_transform(X\_train)

X\_test = sc.transform(X\_test)

# Fitting K-NN Regression to the Training set

from sklearn.neighbors import KNeighborsClassifier

classifier = KNeighborsClassifier(n\_neighbors = 2, metric = 'minkowski', p = 2)

classifier.fit(X\_train, y\_train)

#Fitting SVR to the current Data

from sklearn.svm import SVR

regressor=SVR(kernel='rbf')

regressor.fit(X,y)

# Predicting the Test set results

y\_pred = classifier.predict(X\_test)

#with warnings.catch\_warnings():

#import warnings

#from sklearn.exceptions import DataConversionWarning

#warnings.filterwarnings(action='ignore', category=DataConversionWarning)

#y\_predict=regressor.predict(6.5)

#null\_counts =SVR.isnull().sum()

# Making the Confusion Matrix

from sklearn.metrics import confusion\_matrix

cm = confusion\_matrix(y\_test, y\_pred)

#visualise the SVR results

plt.scatter(X,y,color='red')

plt.plot(regressor.predict(X),color='blue')

plt.title('SVR')

plt.xlabel('age')

plt.ylabel('balance')

