| Serial no. | Kernel name | Parameters | Accuracy |
| --- | --- | --- | --- |
| 1 | Linear | \_ | 87.8868258179 |
| 2 | poly | Degree = 2 | 87.886825817860299 |
| 3 | Poly | Degree = 3 | 88.240495137 |
| 4 | Poly | Degree = 4 | 87.798408488063657 |
| 5 | poly | Degree = 5 | 87.3563218391 |
| 6 | rbf | Gamma = 2 | 87.3563218391 |
| 7 | rbf | Gamma = 3 | 87.3563218391 |

#code

"""

Created on Sun Jul 8 22:12:03 2018

@author: shubham b thorat

SVM on bank\_mutuak\_fund\_data

"""

#importing libraries

import numpy as np

import pandas as pd

import matplotlib.pyplot as plt

#importing datasets

dataset = pd.read\_csv("bank.csv", sep = ";")

X = dataset.iloc[:,:-1].values

Y = dataset.iloc[:,16].values

#using encoding

from sklearn.preprocessing import LabelEncoder ,OneHotEncoder

labelencoder\_X = LabelEncoder()

X[:,1] = labelencoder\_X.fit\_transform(X[:,1])

X[:,2] = labelencoder\_X.fit\_transform(X[:,2])

X[:,3] = labelencoder\_X.fit\_transform(X[:,3])

X[:,4] = labelencoder\_X.fit\_transform(X[:,4])

X[:,6] = labelencoder\_X.fit\_transform(X[:,6])

X[:,7] = labelencoder\_X.fit\_transform(X[:,7])

X[:,8] = labelencoder\_X.fit\_transform(X[:,8])

X[:,10] = labelencoder\_X.fit\_transform(X[:,10])

X[:,15] = labelencoder\_X.fit\_transform(X[:,15])

Y = labelencoder\_X.fit\_transform(Y)

onehotencoder = OneHotEncoder(categorical\_features= [1,2,3,4,6,7,8,10,15])

X = onehotencoder.fit\_transform(X).toarray()

#splitting dataset into training and testing dataset

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

X\_train, X\_test, Y\_train, Y\_test = train\_test\_split(X,Y, test\_size = .25, random\_state = 1)

#normalizing

from sklearn.preprocessing import StandardScaler

sc\_X = StandardScaler()

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

X\_test = sc\_X.fit\_transform(X\_test)

#fitting the SVM to the training set

from sklearn.svm import SVC #support vector classifier

classifier = SVC(kernel ='rbf',gamma=3,random\_state = 0)

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

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

accuracy = (cm[0,0]+cm[1,1])/(cm[0,0]+cm[1,1]+cm[0,1]+cm[1,0])\*100

#making confusion matrix

from sklearn.metrics import confusion\_matrix

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