**Black Box Technique -SVM**

**Name: Upadhyay Sachin Naresh**

**Batch Id: Data science\_08032021**

**Problem 1:**

**1.Business Problem:**

**Objective:**

To predict We need to predict whether he will get greater than 50k or less than 50k.

To overcome this, they consult an analytics firm and would like to get insights on how densely the area is populated and different level of income group people.

**2. Work on each feature of the dataset to create a data dictionary as displayed in the below image:**

|  |  |  |  |
| --- | --- | --- | --- |
| NAME OF FEATURE | Description | Type | Relevance |
| index | index | quantitative | Irrelevant, it doesn’t have any info |
| age | age | quantitative | Relevant |
| Workclass | Workclass | quantitative | Relevant |
| education | education | quantitative | Relevant |
| educationno | educationno | quantitative | Relevant |
| Marital status | Marital status | quantitative | Relevant |
| Occupation | Occupation | quantitative | Relevant |
| Relationship | Relationship | quantitative | Relevant |
| Race | Race | quantitative | Relevant |
| Sex | Sex | quantitative | Relevant |
| Capitalgain | Capitalgain | quantitative | Relevant |
| capitalloss | capitalloss | quantitative | Relevant |
| Hoursperweek | Hoursperweek | quantitative | Relevant |
| native | native | quantitative | Relevant |
| salary | salary | quantitative | Relevant |

**3. SVM model building:**

**R Code:**

Step 1 load the library

library(readr)

library(kernlab)

step 2 load the data set and check the dataset

step 3 Data should be done partition to the train and test

step 4 Training the model on the data

checking the structure of the data

str(salary\_train)

**'data.frame': 30161 obs. of 14 variables:**

**$ age : int 39 50 38 53 28 37 49 52 31 42 ...**

**$ workclass : chr " State-gov" " Self-emp-not-inc" " Private" " Private" ...**

**$ education : chr " Bachelors" " Bachelors" " HS-grad" " 11th" ...**

**$ educationno : int 13 13 9 7 13 14 5 9 14 13 ...**

**$ maritalstatus: chr " Never-married" " Married-civ-spouse" " Divorced" " Married-civ-spouse" ...**

**$ occupation : chr " Adm-clerical" " Exec-managerial" " Handlers-cleaners" " Handlers-cleaners" ...**

**$ relationship : chr " Not-in-family" " Husband" " Not-in-family" " Husband" ...**

**$ race : chr " White" " White" " White" " Black" ...**

**$ sex : chr " Male" " Male" " Male" " Male" ...**

**$ capitalgain : int 2174 0 0 0 0 0 0 0 14084 5178 ...**

**$ capitalloss : int 0 0 0 0 0 0 0 0 0 0 ...**

**$ hoursperweek : int 40 13 40 40 40 40 16 45 50 40 ...**

**$ native : chr " United-States" " United-States" " United-States" " United-States" ...**

**$ Salary : chr " <=50K" " <=50K" " <=50K" " <=50K" ...**

Step 5 check whether the it is having the na values omit in there.

Step 6 converting the required columns into the factor because svm moel support the factor

**str(salary\_train)**

**'data.frame': 30161 obs. of 11 variables:**

**$ age : int 39 50 38 53 28 37 49 52 31 42 ...**

**$ workclass : Factor w/ 7 levels " Federal-gov",..: 6 5 3 3 3 3 3 5 3 3 ...**

**$ education : Factor w/ 16 levels " 10th"," 11th",..: 10 10 12 2 10 13 7 12 13 10 ...**

**$ maritalstatus: Factor w/ 7 levels " Divorced"," Married-AF-spouse",..: 5 3 1 3 3 3 4 3 5 3 ...**

**$ occupation : Factor w/ 14 levels " Adm-clerical",..: 1 4 6 6 10 4 8 4 10 4 ...**

**$ sex : Factor w/ 2 levels " Female"," Male": 2 2 2 2 1 1 1 2 1 2 ...**

**$ capitalgain : int 2174 0 0 0 0 0 0 0 14084 5178 ...**

**$ capitalloss : int 0 0 0 0 0 0 0 0 0 0 ...**

**$ hoursperweek : int 40 13 40 40 40 40 16 45 50 40 ...**

**$ native : Factor w/ 40 levels " Cambodia"," Canada",..: 38 38 38 38 5 38 22 38 38 38 ...**

**$ Salary : Factor w/ 2 levels " <=50K"," >50K": 1 1 1 1 1 1 1 2 2 2 ...**

Step 7 creating the salary\_classifier model on test data

summary(salary\_classifier)

Length Class Mode

1 ksvm S4

Step 8 from the cross table

**Predicted | <=50K | >50K | Row Total |**

**-------------| -----------|-----------|-----------|**

**<=50K | 10620 | 1580 | 12200 |**

**| 0.935 | 0.427 | |**

**-------------| -----------|-----------|-----------|**

**>50k | 740 | 2120 | 2860 |**

**| 0.065 | 0.573 | |**

**-------------| -----------|-----------|-----------|**

**Column Total| 11360| 3700 | 15060 |**

**| 0.754 | 0.246 | |**

**------------ | -----------|-----------|-----------|**

Step 9 test accuracy is 0.84 or 84%

Step 10 model on the train data

table(letter\_predictions, salary\_test$Salary)

**letter\_predictions <=50K >50K**

**<=50K 10620 1580**

**>50K 740 2120**

Step 11 train accuracy is 0.84

Both the test and train accuracy is 0.84

**Python code:**

step 1 import the packages

import pandas as pd

import numpy as np

import seaborn as sns

step 2 load the both train and test data and check the difference between them

step 3 decribing the data

age educationno capitalgain capitalloss hoursperweek

count 30161.000000 30161.000000 30161.000000 30161.000000 30161.000000

mean 38.438115 10.121316 1092.044064 88.302311 40.931269

std 13.134830 2.550037 7406.466611 404.121321 11.980182

min 17.000000 1.000000 0.000000 0.000000 1.000000

25% 28.000000 9.000000 0.000000 0.000000 40.000000

50% 37.000000 10.000000 0.000000 0.000000 40.000000

75% 47.000000 13.000000 0.000000 0.000000 45.000000

max 90.000000 16.000000 99999.000000 4356.000000 99.000000

step 4 import the package pre processing and do the preprocessing

step 5 creating the svm model for the calculatin the linear, poly, rbf and sigmod

step 6 creating the classification model kernel = linear

model\_linear = SVC(kernel = "linear")

model\_linear.fit(salary\_train\_x,salary\_train\_y)

Out: SVC(kernel='linear')

pred\_test\_linear = model\_linear.predict(salary\_test\_x)

np.mean(pred\_test\_linear==salary\_test\_y)

**# Accuracy = 80.41%**

Step 7 creating classification model kernel = ploy

model\_poly = SVC(kernel = "poly")

model\_poly.fit(salary\_train\_x,salary\_train\_y)

Out:SVC(kernel='poly')

pred\_test\_poly = model\_poly.predict(salary\_test\_x)

np.mean(pred\_test\_poly==salary\_test\_y)

**#accuracy =82%**

step 8 creating classification model kernel = rbf

model\_rbf = SVC(kernel = "rbf")

model\_rbf.fit(salary\_train\_x,salary\_train\_y)

pred\_test\_rbf = model\_rbf.predict(salary\_test\_x)

np.mean(pred\_test\_rbf==salary\_test\_y)

**#accuracy = 81%**

Step 9 construction of the confusion matrix

confusion\_matrix(salary\_test\_y,pred\_test\_linear)

confusion\_matrix(salary\_test\_y,pred\_test\_poly)

confusion\_matrix(salary\_test\_y,pred\_test\_rbf)

step 10 kernel = ploy is having the best accuracy of 82%

**4. CONCLUSION:**

Hence the accuracy of the train and the test is 0.84 or 84%.

kernel = ploy is having the best accuracy of 82%.

**Problem 2:**

**1.Business Problem:**

**Objective:**

To predict like to predict the size burned area in forest fires annually so that they can be better prepared in future calamities.

**2. Work on each feature of the dataset to create a data dictionary as displayed in the below image:**

|  |  |  |  |
| --- | --- | --- | --- |
| NAME OF FEATURE | Description | Type | Relevance |
| index | index | quantitative | Irrelevant, it doesn’t have any info |
| Month | Month | quantitative | Relevant |
| Day | Day | quantitative | Relevant |
| FFMC | FFMC | quantitative | Relevant |
| DMC | DMC | quantitative | Relevant |
| DC | DC | quantitative | Relevant |
| ISI | ISI | quantitative | Relevant |
| Temp | Temp | quantitative | Relevant |
| RH | RH | quantitative | Relevant |
| Wind | Wind | quantitative | Relevant |
| Air | Air | quantitative | Relevant |
| Area | Area | quantitative | Relevant |
| Dayfri | Dayfri | quantitative | Relevant |
| Daymon | Daymon | quantitative | Relevant |
| Daysat | Daysat | quantitative | Relevant |
| Daysun | Daysun | quantitative | Relevant |
| Daythu | Daythu | quantitative | Relevant |
| Daytue | Daytue | quantitative | Relevant |
| Daywed | Daywed | quantitative | Relevant |
| Monthapr | Monthapr | quantitative | Relevant |
| monthaug | monthaug | quantitative | Relevant |
| Monthdec | Monthdec | quantitative | Relevant |
| Monthfeb | Monthfeb | quantitative | Relevant |
| Monthjan | Monthjan | quantitative | Relevant |
| Monthjul | Monthjul | quantitative | Relevant |
| monthjun | monthjun | quantitative | Relevant |
| Monthmar | Monthmar | quantitative | Relevant |
| Monthmay | Monthmay | quantitative | Relevant |
| Monthnov | Monthnov | quantitative | Relevant |
| Monthoct | Monthoct | quantitative | Relevant |
| Monthsept | Monthsept | quantitative | Relevant |
| Size\_category | Size\_category | quantitative | Relevant |

**3. SVM model building:**

**R Code:**

Step 1 load the library

library(readr)

library(kernlab)

step 2 load the data set and check the dataset

step 3 the data which is finding the structure of the dataset

str(fire)

'data.frame': 517 obs. of 30 variables:

$ FFMC : num 86.2 90.6 90.6 91.7 89.3 92.3 92.3 91.5 91 92.5 ...

$ DMC : num 26.2 35.4 43.7 33.3 51.3 ...

$ DC : num 94.3 669.1 686.9 77.5 102.2 ...

$ ISI : num 5.1 6.7 6.7 9 9.6 14.7 8.5 10.7 7 7.1 ...

$ temp : num 8.2 18 14.6 8.3 11.4 22.2 24.1 8 13.1 22.8 ...

$ RH : int 51 33 33 97 99 29 27 86 63 40 ...

$ wind : num 6.7 0.9 1.3 4 1.8 5.4 3.1 2.2 5.4 4 ...

$ rain : num 0 0 0 0.2 0 0 0 0 0 0 ...

$ area : num 0 0 0 0 0 0 0 0 0 0 ...

$ dayfri : int 1 0 0 1 0 0 0 0 0 0 ...

$ daymon : int 0 0 0 0 0 0 1 1 0 0 ...

$ daysat : int 0 0 1 0 0 0 0 0 0 1 ...

$ daysun : int 0 0 0 0 1 1 0 0 0 0 ...

$ daythu : int 0 0 0 0 0 0 0 0 0 0 ...

$ daytue : int 0 1 0 0 0 0 0 0 1 0 ...

$ daywed : int 0 0 0 0 0 0 0 0 0 0 ...

$ monthapr : int 0 0 0 0 0 0 0 0 0 0 ...

$ monthaug : int 0 0 0 0 0 1 1 1 0 0 ...

$ monthdec : int 0 0 0 0 0 0 0 0 0 0 ...

$ monthfeb : int 0 0 0 0 0 0 0 0 0 0 ...

$ monthjan : int 0 0 0 0 0 0 0 0 0 0 ...

$ monthjul : int 0 0 0 0 0 0 0 0 0 0 ...

$ monthjun : int 0 0 0 0 0 0 0 0 0 0 ...

$ monthmar : int 1 0 0 1 1 0 0 0 0 0 ...

$ monthmay : int 0 0 0 0 0 0 0 0 0 0 ...

$ monthnov : int 0 0 0 0 0 0 0 0 0 0 ...

$ monthoct : int 0 1 1 0 0 0 0 0 0 0 ...

$ monthsep : int 0 0 0 0 0 0 0 0 1 1 ...

$ Size\_cateogary.small: num 1 1 1 1 1 1 1 1 1 1 ...

$ Size\_cateogary.large: num 0 0 0 0 0 0 0 0 0 0 ...

Step 4 doing partition the data into the train and the test

Step 5 training the svm model on the data

Import the libray (kernel)

Using simple linear svm

Step 6 evaluation the model performance prediction on the testing data

fire\_predictions<- predict(fire\_classifier,fire\_test)

table(fire\_predictions, fire\_test$area)

agreement <- fire\_predictions == fire\_test$area

table(agreement)

prop.table(table(agreement))

step 7 falsea aggrement value is 104 this a best fit model

**Python code:**

Step 1 import the required packages

import pandas as pd

import numpy as np

import seaborn as sns

step 2 load the datset and check the dataset

step 3 removing the columns which are not useful in the dataset

step 4 doing the prerocessing of the data and remving na values if available

step 5 now splitting the data into the train and test

step 6 Now making the SVM classification model of the linear, poly, rbf

step 7 kernel = linear model

model\_linear = SVC(kernel = "linear")

model\_linear.fit(train\_x,train\_y)

pred\_test\_linear = model\_linear.predict(test\_x)

np.mean(pred\_test\_linear==test\_y)

**# Accuracy = 99.23%**

Step 8 now kernel = ploy

model\_poly = SVC(kernel = "poly")

model\_poly.fit(train\_x,train\_y)

pred\_test\_poly = model\_poly.predict(test\_x)

np.mean(pred\_test\_poly==test\_y)

**# accuracy =80.76%**

Step 9 kernel =rbf

model\_rbf = SVC(kernel = "rbf")

model\_rbf.fit(train\_x,train\_y)

pred\_test\_rbf = model\_rbf.predict(test\_x)

np.mean(pred\_test\_rbf==test\_y)

**# Accuracy = 78.46%**

Step 10 hence the linear kernel model is performing well

**4.Conclusion:**

Hence the linear kernel is performing well with the accuracy of 99.23%