Classification using single layer perceptron

Report:

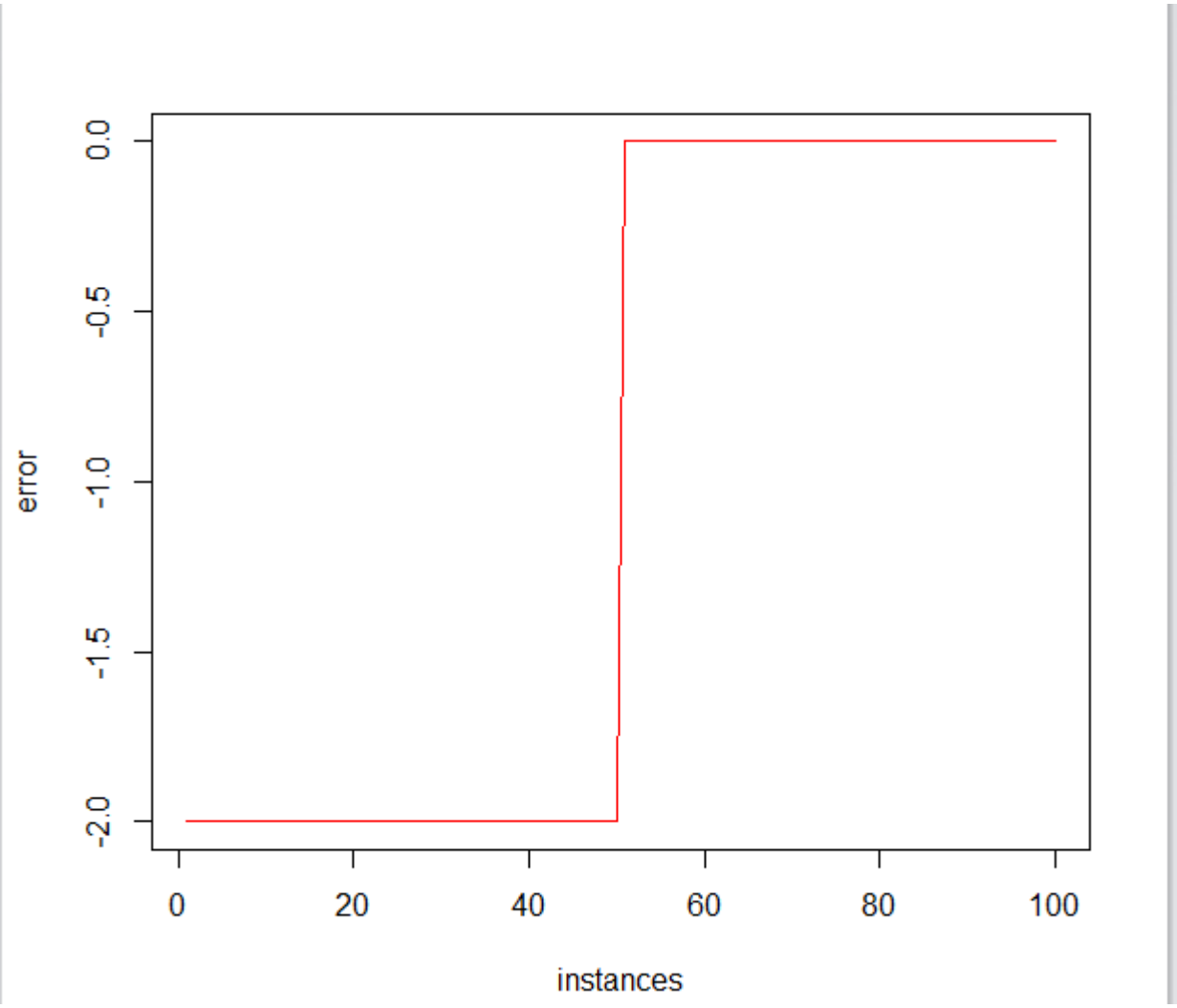
1-The perceptron structure:

Number of inputs = 2 “sepal length and petal length”
Number of outputs= 1 “species”
Number of weights = 2 “weight of sepal and weight of petal”

The structure is composed of a vector X representing the two inputs, sepal and petal length, one output Y representing the result for the summation of inputs multiplied by their weights and a vector weight of two values representing a weight for each input.

2-Training the perceptron algorithm using the provided dataset:

a- Plotting the value of error against the index of the training instance:



b- Observations on error value change:
The error function gave a result that is either value -2 or 0, which indicates that most probably the output was -1 while it should have been 1 and it was never the vice versa ,or there was no error meaning that the output was similar to the target.

3-Modifying the learning rate to take values (0.2, 0.4, 0.8, and 1):

a- Learning rate = 0.2:
Number of iterations = 7

```
Console C:/Users/Amira El-Ewady/Desktop/test/
+         error[j]= output -y[j]
+
+
+       }
+
+     }
+     print(w)
+     print(iterations)
+     return(error)
+
+ }
> run= perceptron(x,y,0.2)
    sepal petal
51   0.94 -2.22
[1] 7
>
```

b- Learning rate= 0.4:

Number of iterations = 13

```
Console C:/Users/Amira El-Ewady/Desktop/test/
+         error[j]= output -y[j]
+
+
+       }
+
+     }
+     print(w)
+     print(iterations)
+     return(error)
+
+ }
> run= perceptron(x,y,0.4)
    sepal petal
51   4.34 -8.86
[1] 13
>
```

c- Learning rate= 0.8:

Number of iterations= 10

```
Console C:/Users/Amira El-Ewady/Desktop/test/
-         error[j]= output -y[j]
-
-
-       }
-
-     }
-     print(w)
-     print(iterations)
-     return(error)
-
- }
> run= perceptron(x,y,0.8)
    sepal  petal
51   5.62 -14.06
[1] 10
>
```

d- Learning rate= 1:

Number of iterations= 10

```
Console C:/Users/Amira El-Ewady/Desktop/test/
+         error[j]= output -y[j]
+
+
+       }
+
+    }
+    print(w)
+    print(iterations)
+    return(error)
+
+ }
> run= perceptron(x,y,1)
   sepal petal
51   7.5 -17.9
[1] 10
>
```

Comment:
The number of iterations changes differently as the learning rate changes and accordingly the updated final weight changes as well.

4-Using the recall/testing mode:

a- Yes, the above instances were correctly classified using the perceptron output.

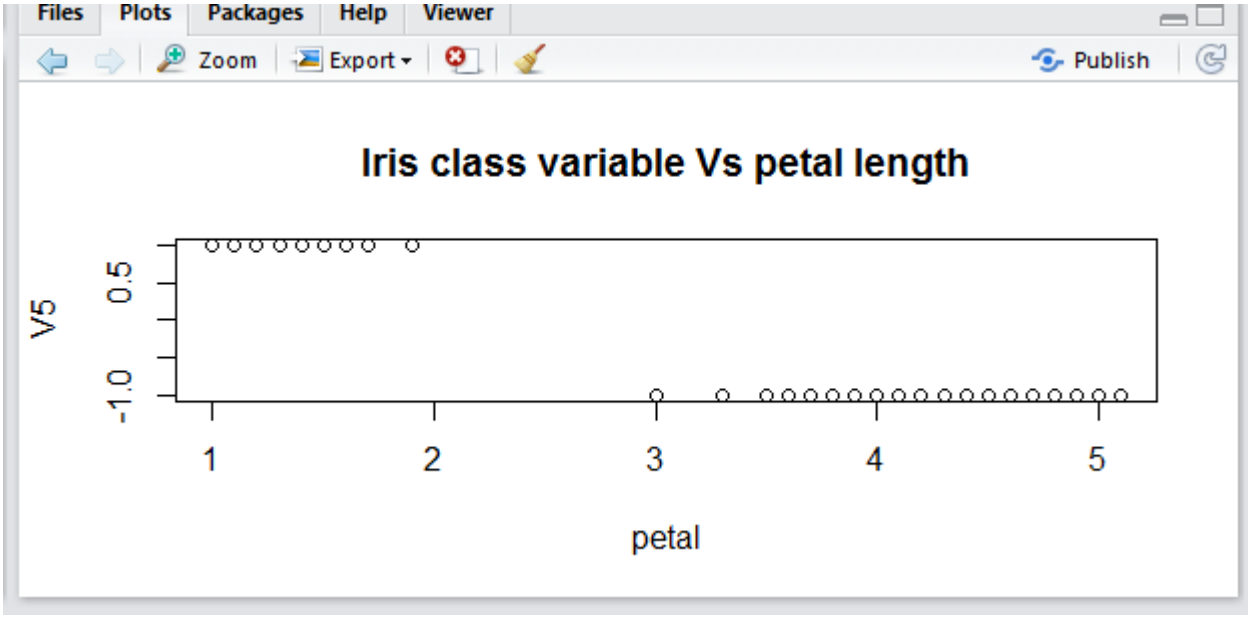
```
Console C:/Users/Amira El-Ewady/Desktop/test/
[1] "Species: -1"
> test = RecallMode(x,y,0.2,5.4,1.7)
   sepal petal
51  0.94 -2.22
[1] 7
   sepal
51 1.302
[1] 1
[1] "Species: 1"
> test = RecallMode(x,y,0.2,6.3,4.7)
   sepal petal
51  0.94 -2.22
[1] 7
    sepal
51 -4.512
[1] -1
[1] "Species: -1"
> test = RecallMode(x,y,0.2,5.1,3)
   sepal petal
51  0.94 -2.22
[1] 7
    sepal
51 -1.866
[1] -1
[1] "Species: -1"
>
```

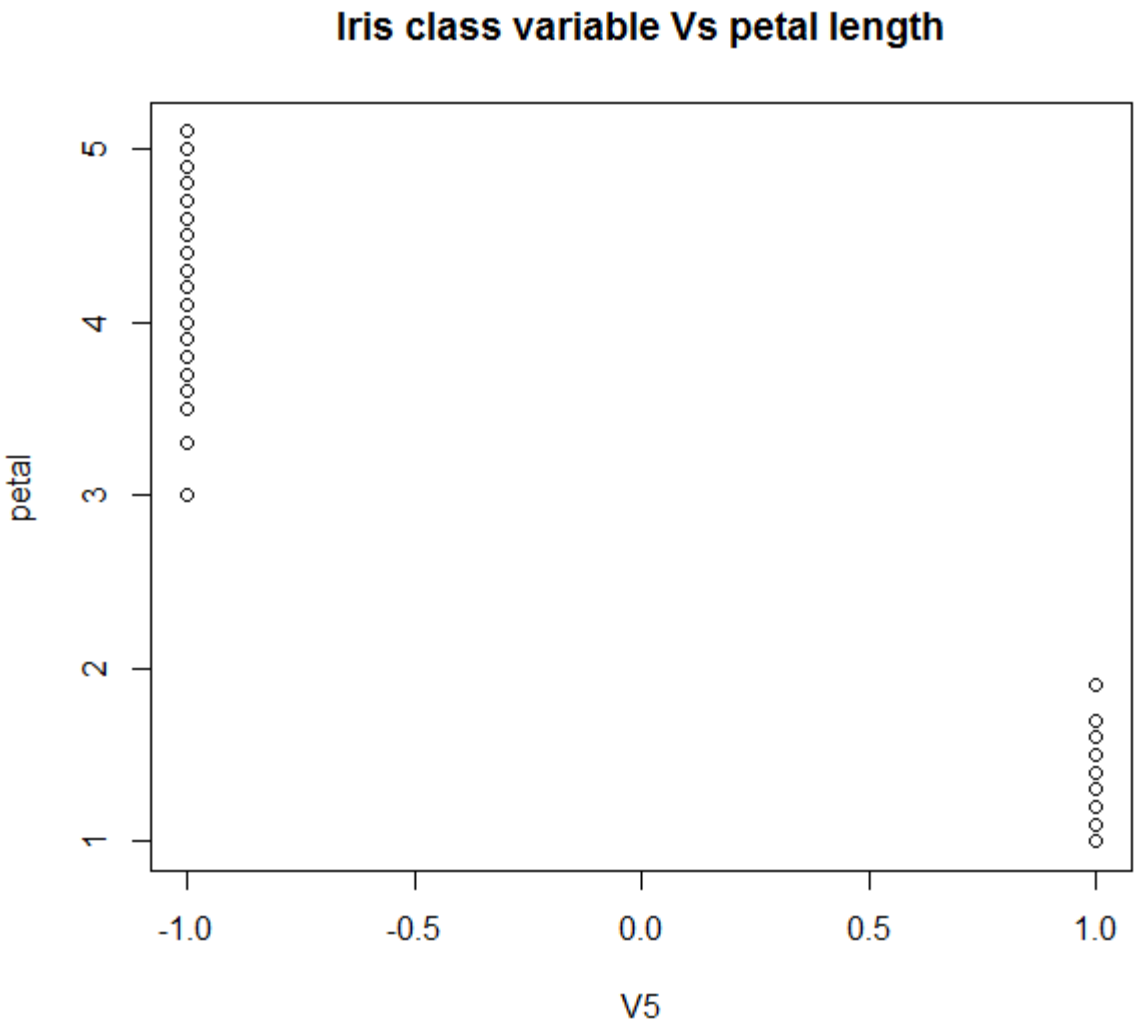
b- Since the testing on the three given instances were correctly classified and also random testing was applied giving correct classifying, then the perceptron classification can be regarded accurate.

c- I tried different random instances and they all gave correct classifications so I didn't really account for wrong classifications.

5-Exploratory Data analysis:

a- The Iris class variable versus the petal length:

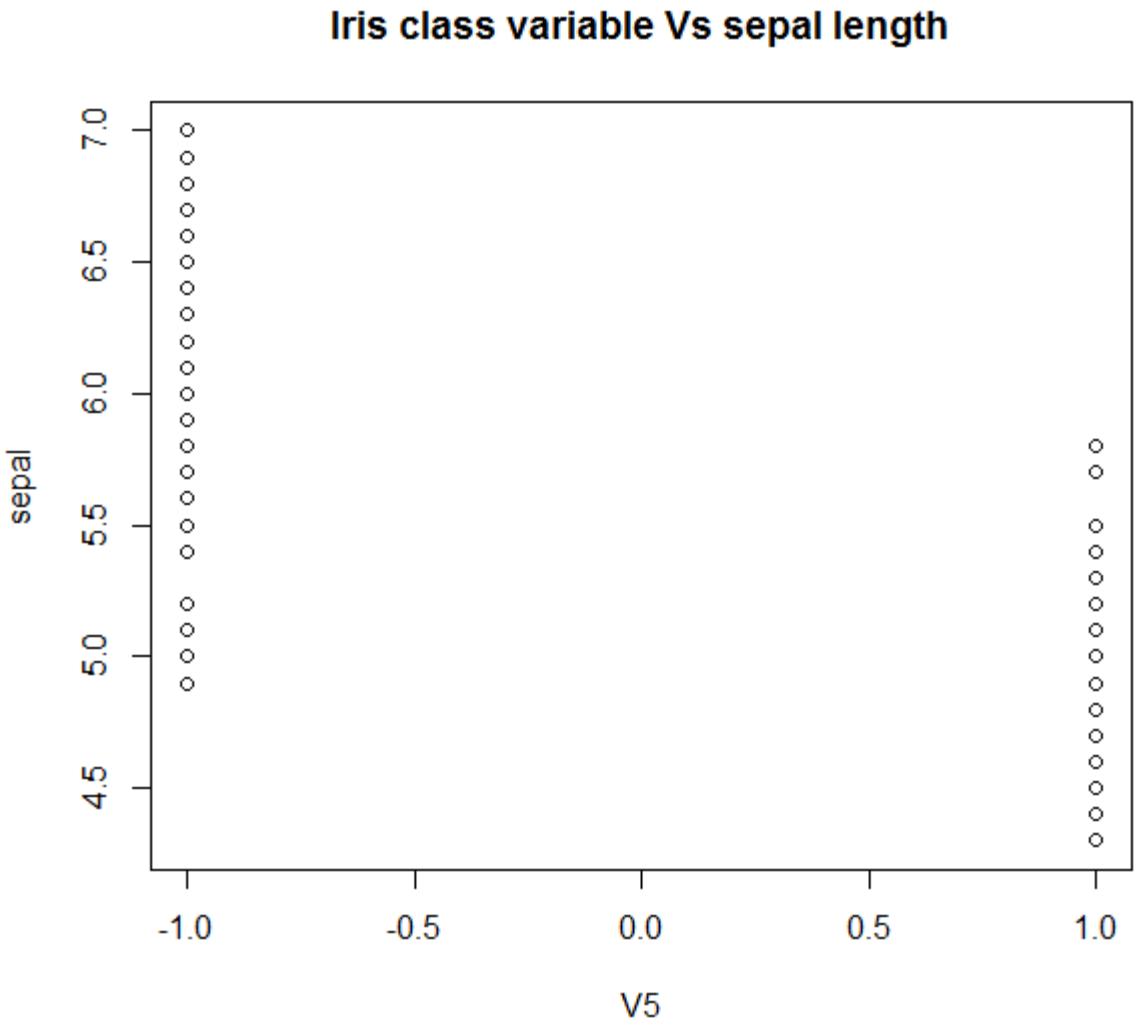

Files
Plots
Packages
Help
Viewer
Zoom
Export
Publish
Iris class variable Vs petal length
V5
0.5
-1.0
1
2
3
4
5
petal



b- Covariance and Correlation:

```
> covariance = cov(as.numeric(V5),as.numeric(petal))
> print(paste0("Covariance = ",covariance))
[1] "Covariance = -1.41313131313131"
> correlation =cor(as.numeric(V5),as.numeric(petal))
> print(paste0("Correlation = ",correlation))
[1] "Correlation = -0.969990231486349"
>
```

c- The Iris class variable versus the sepal length:



Covariance and correlation:

```
> covariance = cov(as.numeric(V5),as.numeric(sepal))
> print(paste0("Covariance = ",covariance))
[1] "Covariance = -0.46969696969697"
>
> correlation =cor(as.numeric(V5),as.numeric(sepal))
> print(paste0("Correlation = ",correlation))
[1] "Correlation = -0.728290148746214"
>
```

d- The relationship between the three variables is a – ve correlation as well as – ve covariance which means that the variables are not related, they are inversely proportional to each other, which can mean that the training of the

perceptron can be easier and faster than the case where the inputs are positively correlated.