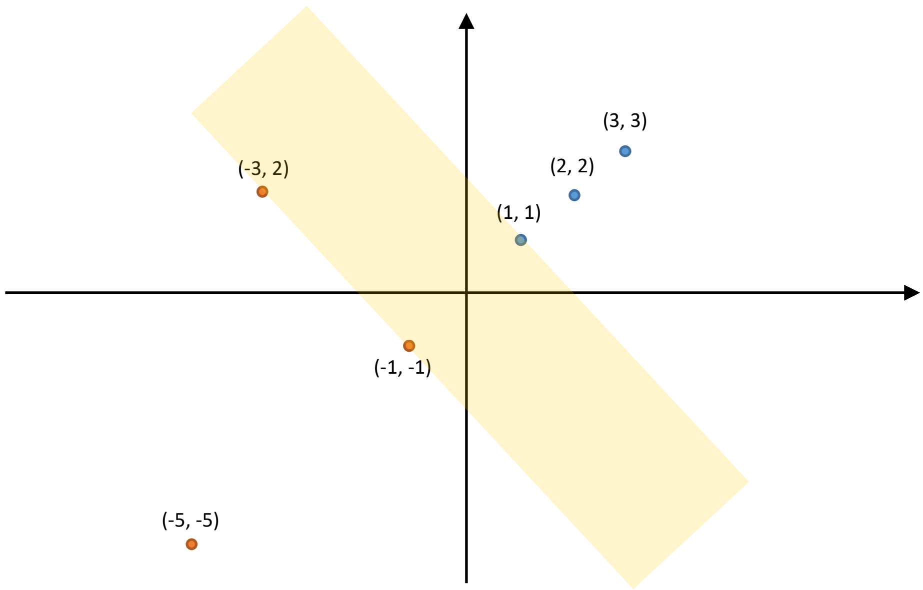
Machine Learning HW #6 answer

1.



$$\mathrm{S} = \left\{\begin{bmatrix}1\\1\end{bmatrix}, \begin{bmatrix}-1\\-1\end{bmatrix}, \begin{bmatrix}-3\\2\end{bmatrix}\right\} \quad \tilde{\mathrm{S}} = \left\{\begin{bmatrix}1\\1\\1\end{bmatrix}, \begin{bmatrix}-1\\-1\\1\end{bmatrix}, \begin{bmatrix}-3\\2\\1\end{bmatrix}\right\}$$

$$\widetilde{\mathrm{w}} = \alpha_1\widetilde{s_1} + \alpha_2\widetilde{s_2} + \alpha_3\widetilde{s_3}$$

$$\begin{cases} \alpha_1\widetilde{s_1}\widetilde{s_1} + \alpha_2\widetilde{s_2}\widetilde{s_1} + \alpha_3\widetilde{s_3}\widetilde{s_1} = +1 \\ \alpha_1\widetilde{s_1}\widetilde{s_2} + \alpha_2\widetilde{s_2}\widetilde{s_2} + \alpha_3\widetilde{s_3}\widetilde{s_2} = -1 \\ \alpha_1\widetilde{s_1}\widetilde{s_3} + \alpha_2\widetilde{s_2}\widetilde{s_3} + \alpha_3\widetilde{s_3}\widetilde{s_3} = -1 \end{cases}$$

$$=> \begin{cases} \alpha_1\begin{bmatrix}1 & 1 & 1\end{bmatrix}\begin{bmatrix}1\\1\\1\end{bmatrix} + \alpha_2\begin{bmatrix}-1 & -1 & 1\end{bmatrix}\begin{bmatrix}1\\1\\1\end{bmatrix} + \alpha_3\begin{bmatrix}-3 & 2 & 1\end{bmatrix}\begin{bmatrix}1\\1\\1\end{bmatrix} = 1 \\ \alpha_1\begin{bmatrix}1 & 1 & 1\end{bmatrix}\begin{bmatrix}-1\\-1\\1\end{bmatrix} + \alpha_2\begin{bmatrix}-1 & -1 & 1\end{bmatrix}\begin{bmatrix}-1\\-1\\1\end{bmatrix} + \alpha_3\begin{bmatrix}-3 & 2 & 1\end{bmatrix}\begin{bmatrix}-1\\-1\\1\end{bmatrix} = -1 \\ \alpha_1\begin{bmatrix}1 & 1 & 1\end{bmatrix}\begin{bmatrix}-3\\2\\1\end{bmatrix} + \alpha_2\begin{bmatrix}-1 & -1 & 1\end{bmatrix}\begin{bmatrix}-3\\2\\1\end{bmatrix} + \alpha_3\begin{bmatrix}-3 & 2 & 1\end{bmatrix}\begin{bmatrix}-3\\2\\1\end{bmatrix} = -1 \end{cases}$$

$$=> \begin{cases} 3\alpha_1 - \alpha_2 = 1 \\ -\alpha_1 + 3\alpha_2 + 2\alpha_3 = -1 \\ 2\alpha_2 + 14\alpha_3 = -1 \end{cases} => \alpha_1 = 0.26, \alpha_2 = -0.22, \alpha_1 = -0.04$$

$$\widetilde{\mathrm{w}} = 0.26\begin{bmatrix}1\\1\\1\end{bmatrix} + (-0.22)\begin{bmatrix}-1\\-1\\1\end{bmatrix} + (-0.04)\begin{bmatrix}-3\\2\\1\end{bmatrix} = \begin{bmatrix}0.6\\0.4\\0\end{bmatrix}$$

$$w = \begin{bmatrix}0.6\\0.4\end{bmatrix}, b = 0$$

2.

$thr_1 = [0.9, 1.1, 1.4, 2.1]$

	(1, 2.1)	(2, 1.1)	(1.3, 1)	(1, 1)	(2, 1)
Target	+1	+1	-1	-1	+1
If x<0.9 c=+1	-1	-1	-1	-1	-1
If x>0.9 c=+1	+1	+1	+1	+1	+1
If x<1.1 c=+1	+1	-1	-1	+1	-1
If x>1.1 c=+1	-1	+1	+1	-1	+1
If x<1.4 c=+1	+1	-1	+1	+1	-1
If x>1.4 c=+1	-1	+1	-1	-1	+1
If x<2.1 c=+1	+1	+1	+1	+1	+1
If x>2.1 c=+1	-1	-1	-1	-1	-1

$$\epsilon = \frac{1}{5} \quad \alpha = \frac{1}{2}\ln\left(\frac{1-0.2}{0.2}\right) \fallingdotseq 0.693$$

$$w = \begin{cases} \frac{1}{5}e^{0.693} \fallingdotseq 0.4 \quad c = false \\ \frac{1}{5}e^{-0.693} \fallingdotseq 0.1 \quad c = true \end{cases}$$

$$\mathrm{w} = [0.4, 0.1, 0.1, 0.1, 0.1] => [0.5, 0.125, 0.125, 0.125, 0.125]^T$$

second iteration:

	(1, 2.1)	(2, 1.1)	(1.3, 1)	(1, 1)	(2, 1)	w
Target	+1	+1	-1	-1	+1	
If x<0.9 c=+1			0.125	0.125		0.25
If x>0.9 c=+1	0.5	0.125			0.125	0.75
If x<1.1 c=+1	0.5		0.125			0.625
If x>1.1 c=+1		0.125		0.125	0.125	0.375
If x<1.4 c=+1	0.5					0.5
If x>1.4 c=+1		0.125	0.125	0.125	0.125	0.5
If x<2.1 c=+1	0.5	0.125			0.125	0.75
If x>2.1 c=+1			0.125	0.125		0.25

$$\epsilon = \frac{0.125+0.125}{0.5+0.125\times 4} = 0.25 \quad \alpha = \frac{1}{2}\ln\left(\frac{1-0.4}{0.4}\right) \fallingdotseq 0.203$$

$$w = \begin{cases} 0.25e^{0.203} \fallingdotseq 0.3 \quad c = false \\ 0.25e^{-0.203} \fallingdotseq 0.2 \quad c = true \end{cases}$$

$$\mathrm{w} = [0.5\times 0.2, 0.125\times 0.2, 0.125\times 0.3, 0.125\times 0.3, 0.125\times 0.2]$$

$$\mathrm{w} => [0.1, 0.025, 0.0375, 0.0375, 0.025]^T$$

3.

$$J(w) = |z_k d_k| - z_k d_k$$

$$=> J(w) = \begin{cases} 0 & , if \ \ z_k d_k > 0 \\ -2 z_k d_k & , if \ \ z_k d_k < 0 \end{cases}$$

But $z_k = {x_k}^T \times w_k$

$$=> \nabla J(w) = -2 x_k d_k$$

The updating equation becomes (stochastic gradient descent)

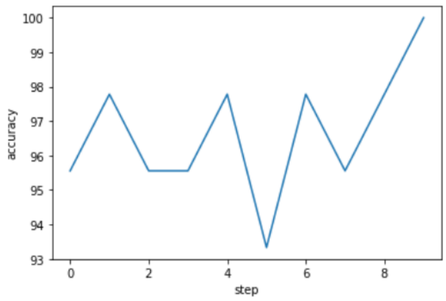
$$=> w_{k+1} = w_k + \eta x_k d_k \ \ , \text{if} \ \ z_k d_k < 0$$

4.

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn import datasets

iris_dataset = datasets.load_iris()

df_X = iris_dataset.data[:, :4]
df_y = iris_dataset.target
```

```
accuracy=[]

for i in range(10):
    X_train, X_test, y_train, y_test = train_test_split(df_X, df_y, test_size=0.3)
    classifier = SVC(kernel='rbf')
    classifier.fit(X_train, y_train)
    accuracy.append(round(classifier.score(X_test, y_test) * 100, 3))

plt.plot(accuracy)
plt.xlabel('step')
plt.ylabel('accuracy')
plt.show()
```

```
print(f"SVM accuracy = ", accuracy)
print(f"SVM accuracy average: {np.mean(accuracy)}")
```



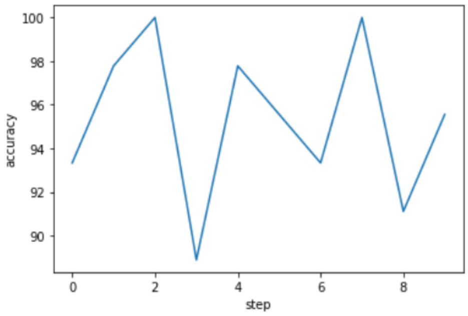

```
SVM accuracy =  [95.556, 97.778, 95.556, 95.556, 97.778, 93.333, 97.778, 95.556, 97.778, 100.0]
SVM accuracy average: 96.66690000000001
```

5.

```python
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from sklearn.ensemble import AdaBoostClassifier
from sklearn import datasets

iris_dataset = datasets.load_iris()

df_X = iris_dataset.data[:, :4]
df_y = iris_dataset.target
```

```python
accuracy=[]

for i in range(10):
    X_train, X_test, y_train, y_test = train_test_split(df_X, df_y, test_size=0.3)
    classifier = AdaBoostClassifier()
    classifier.fit(X_train, y_train)
    accuracy.append(round(classifier.score(X_test, y_test) * 100, 3))

plt.plot(accuracy)
plt.xlabel('step')
plt.ylabel('accuracy')
plt.show()
```

```python
print(f"Adaboost accuracy = ", accuracy)
print(f"Adaboost accuracy average: {np.mean(accuracy)}")
```




```
Adaboost accuracy =  [93.333, 97.778, 100.0, 88.889, 97.778, 95.556, 93.333, 100.0, 91.111, 95.556]
Adaboost accuracy average: 95.33340000000001
```

In this experimental result, SVM is better than Adaboost.