Perceptron_Shivani

October 17, 2021

[12]:
```
%load_ext autoreload
%autoreload 2
%matplotlib inline
```

```
The autoreload extension is already loaded. To reload it, use:
  %reload_ext autoreload
```

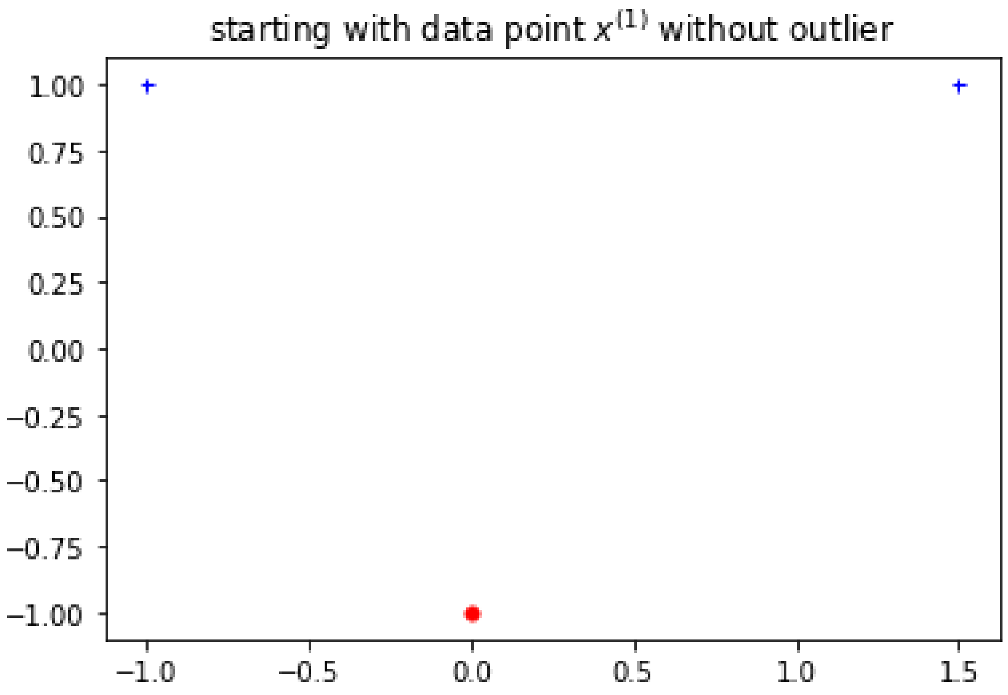
[128]:
```
import sys
from pathlib import Path
import matplotlib.pyplot as plt
```

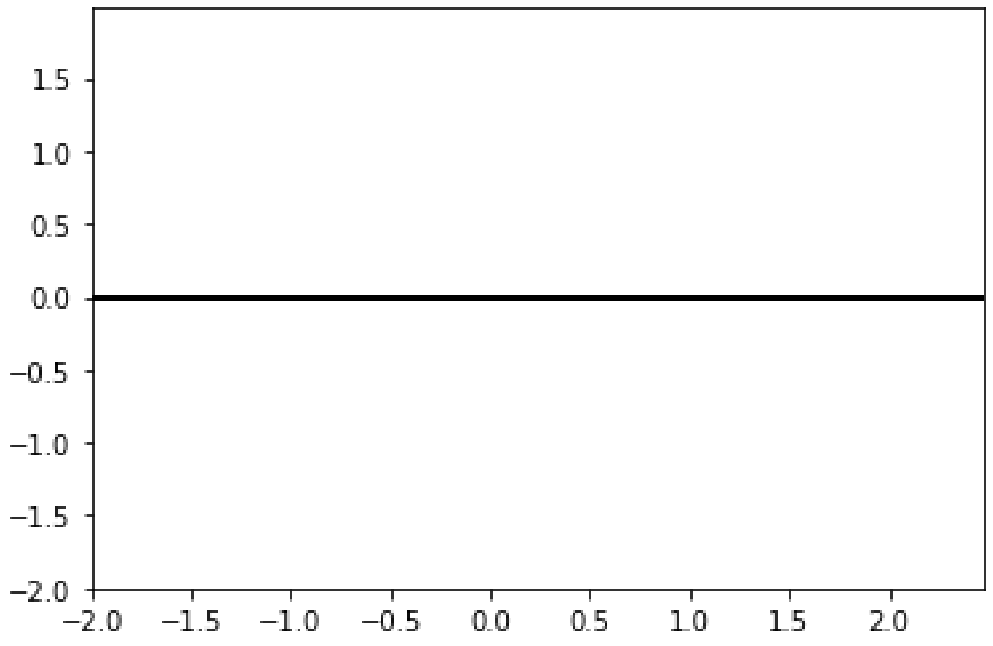
[14]:
```
if 'source' not in sys.path:
    sys.path.append('source')
```

[15]:
```
import perceptron
from perceptron import main
```

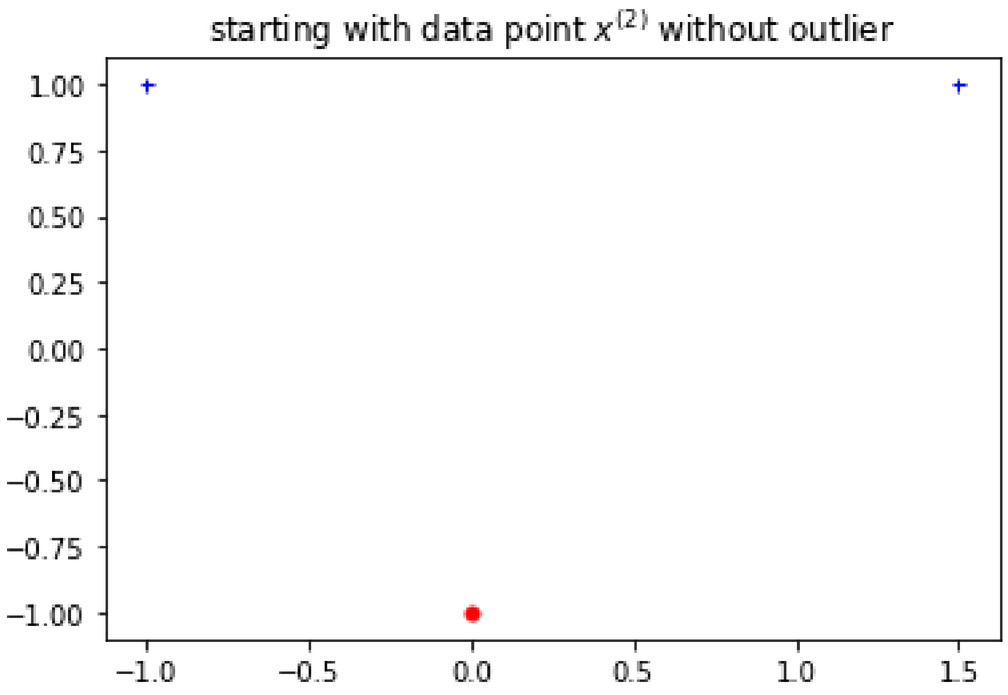
[167]:
```
main()
```

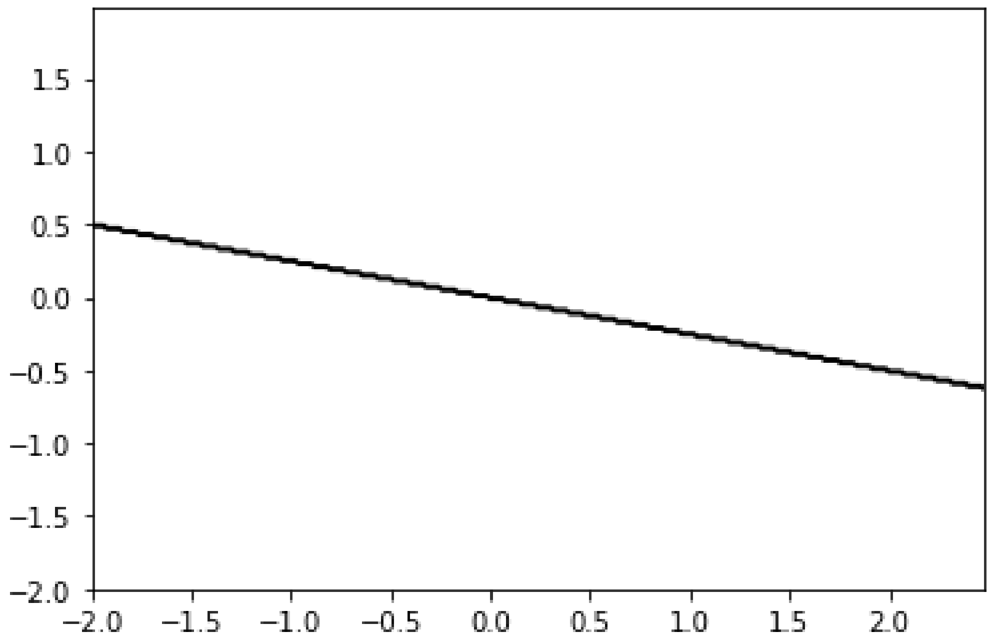
```
starting with data point $x^{(1)}$ without outlier
        coef = [0. 1.], mistakes = 1
```



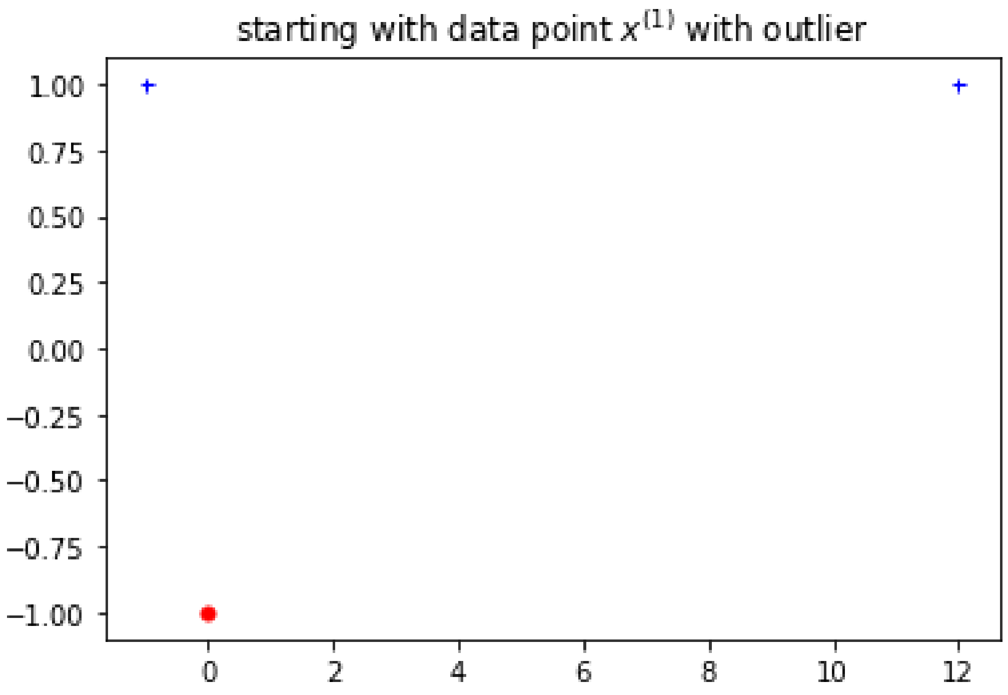


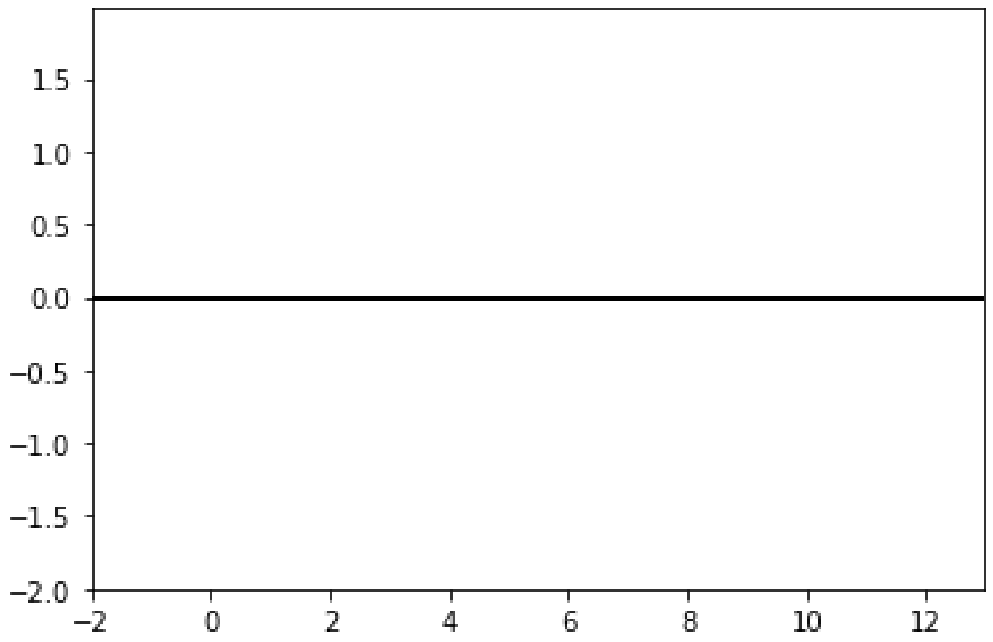
```
starting with data point $x^{(2)}$ without outlier
        coef = [0.5 2. ], mistakes = 2
```



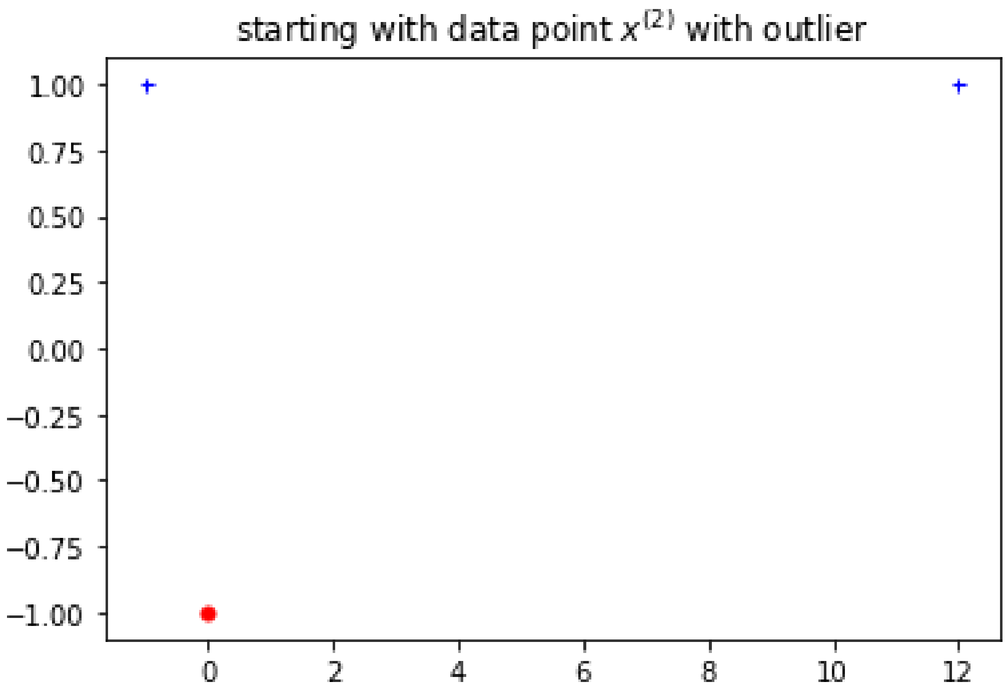


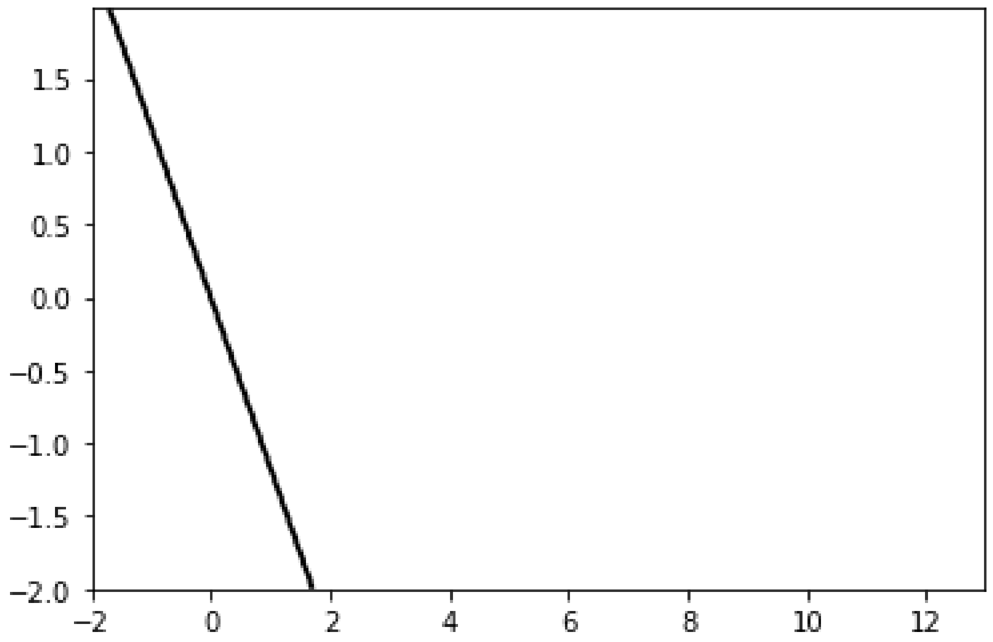
```
starting with data point $x^{(1)}$ with outlier
        coef = [0. 1.], mistakes = 1
```





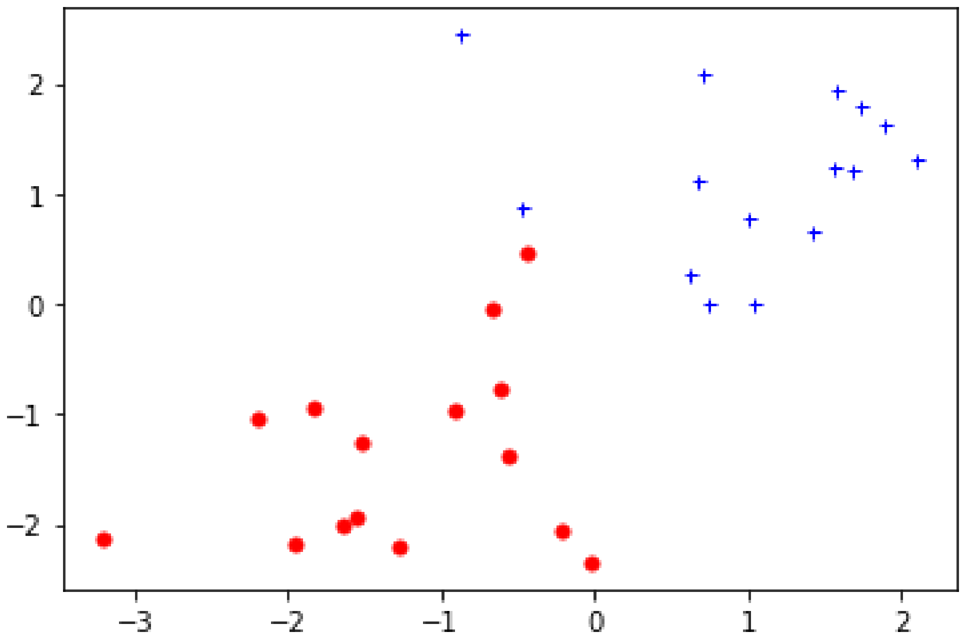
```
starting with data point $x^{(2)}$ with outlier
        coef = [7. 6.], mistakes = 6
```

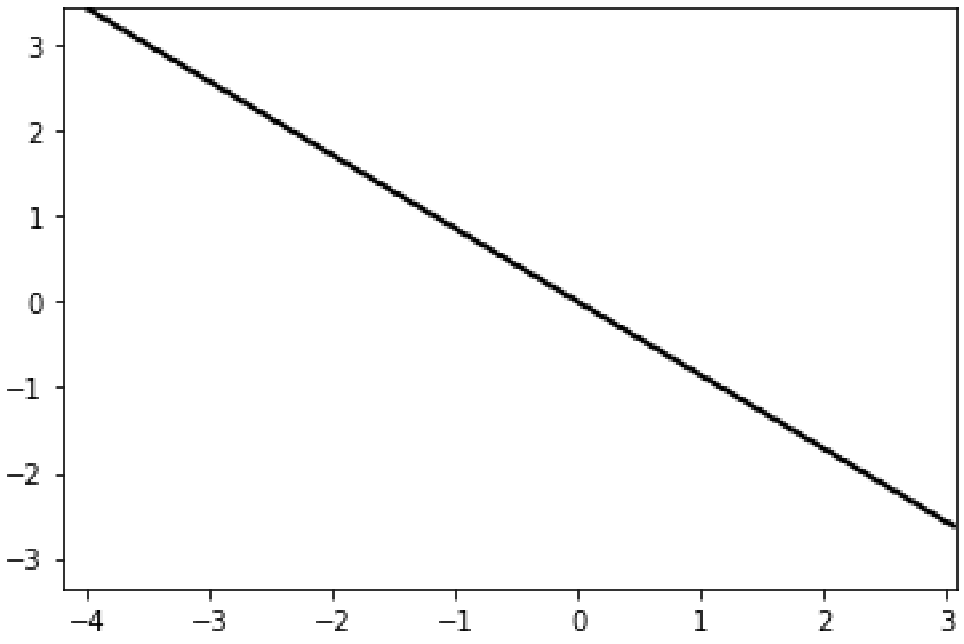

starting with data point $x^{(2)}$ with outlier
1.00
0.75
0.50
0.25
0.00
−0.25
−0.50
−0.75
−1.00
0
2
4
6
8
10
12


1.5
1.0
0.5
0.0
−0.5
−1.0
−1.5
−2.0
−2
0
2
4
6
8
10
12

```
For  [0, 0]
```

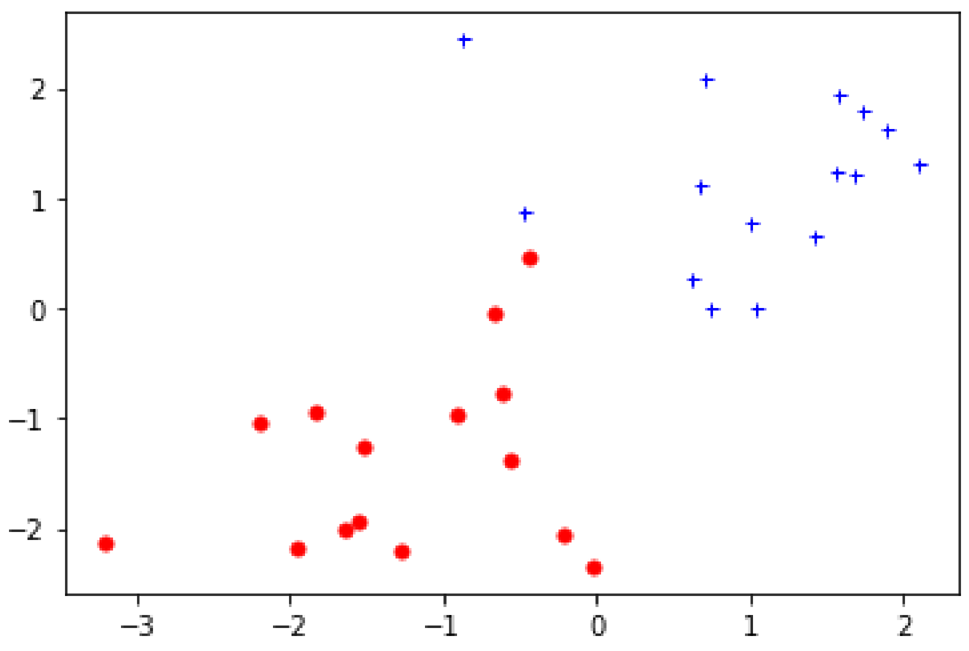
```
Inital Coef:  [0, 0]
        coef = [6. 7.], mistakes = 7
```

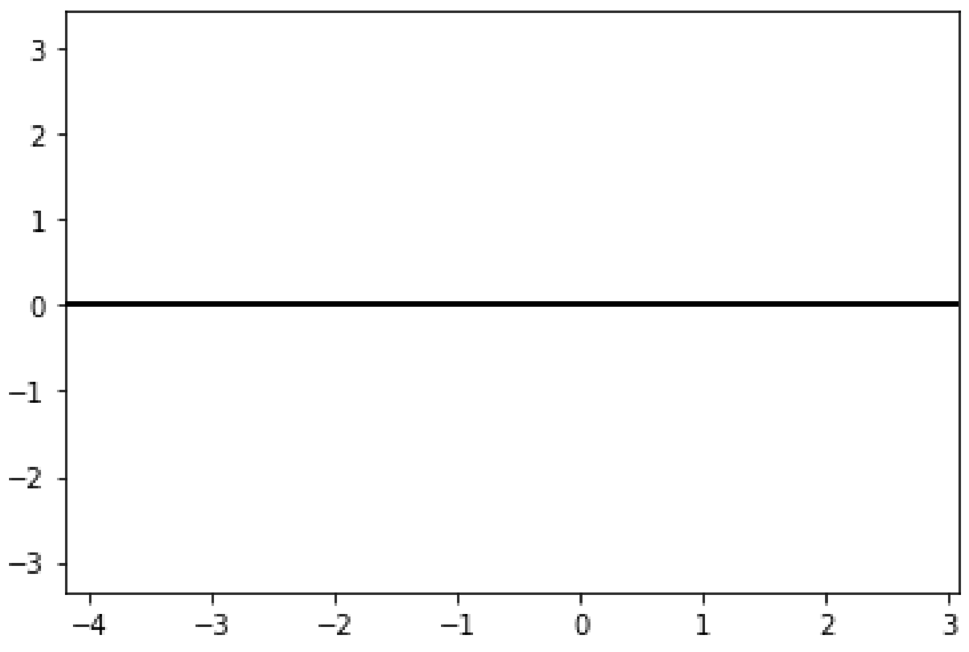




```
For  [1, 0]
Inital Coef:  [1, 0]
```

```
coef = [0. 1.], mistakes = 1
```





```
[ ]: 2. For Initial coef = [0, 0], the trained coeffecients are [6. 7.]
        For Initial coef = [1, 0], the trained coeffecients are [0. 1.]
```

```
The 2 training procedures is not converging to the same solution, becasuse of␣
↪the different initial coeffcient.
No the performance is different, 1st initial coef does 7 mistakes and 2nd␣
↪initial coef does 1 mistake.
On the held out data set, it depends on the initail coef that we start with.
```