Applied Mathematics for Computer Science: Experiment Report

Yuwen Xiong
3130000829

May 29, 2016

1 Homework 1

We generate a sin curve in $[0, 1]$ with $N(0, 0.1)$ gaussian noise by following code:

```
sample_x = np.linspace(0, 1, n)
sample_y = np.sin(sample_x * 2 * np.pi) + np.random.normal(0, 0.1, n)
```

We just calculate the close form $w = (X^T X + \lambda I)^{-1} X^T y$ to get the corresponding w, and plot it.

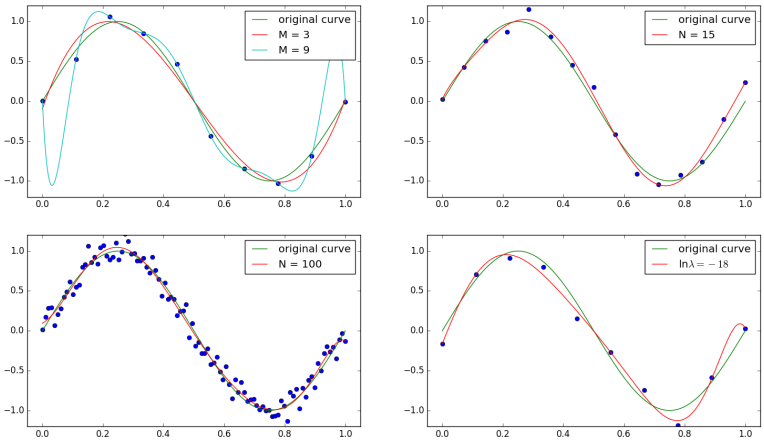

Figure 1: The result of homework 1

2 Homework 2

We use *optdigits.tes* as input, reduce its dimension from 64 to 2 by SVD decomposition and visualize them in 2D plot.

The key code as follows:

```
m = feas.mean(axis = 0)
feas = (feas - m)
U, S, VT = np.linalg.svd(feas, full_matrices = False)
res = np.dot(U[:, 0:2], np.diag(S[0:2]))
```

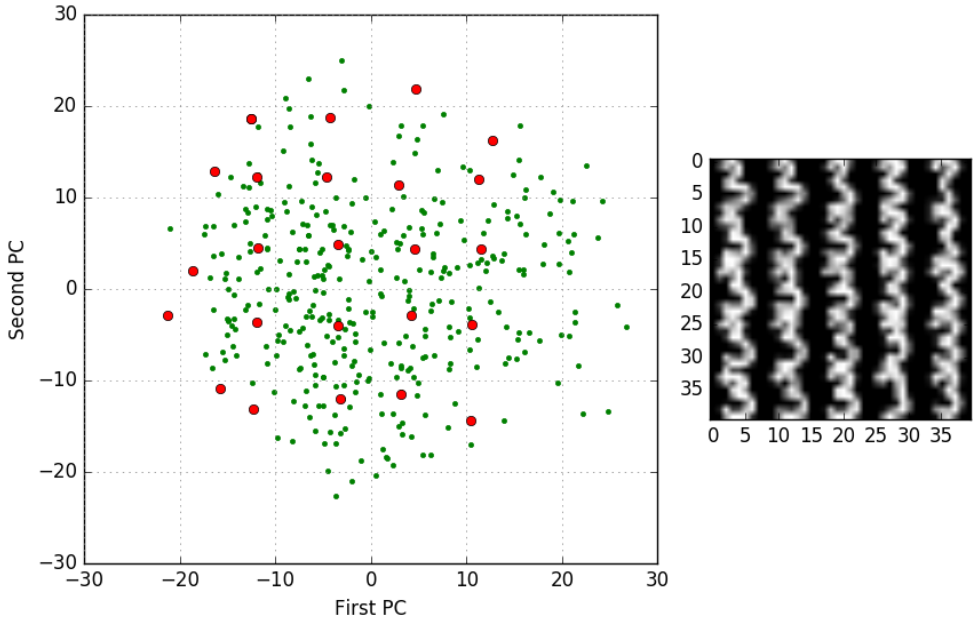

Figure 2: The result of homework 2

3 Homework 3

We use *numpy.random.multivariate_normal* as follows to generate three 2-D multi-variable gaussian clusters, and implement a mog class with EM algorithm.

```
mean1 = [3, 0]
mean2 = [-1, -3]
mean3 = [4, 5]
sigma1 = np.diag((1, 4))
sigma2 = np.diag((4, 3))
sigma3 = np.array([[2, 1], [1, 2]])
```

```
norm1 = np.random.multivariate_normal(mean1, sigma1, 300)
norm2 = np.random.multivariate_normal(mean2, sigma2, 300)
norm3 = np.random.multivariate_normal(mean3, sigma3, 300)
```

We also implement *regularization term support* and we think our code could also handle *higher dimensional* data(though we haven't tested it).

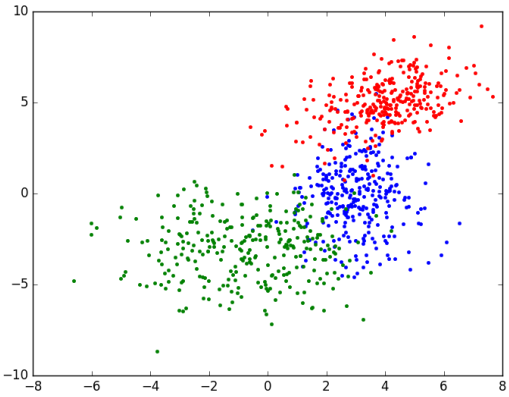

Figure 3: The original data

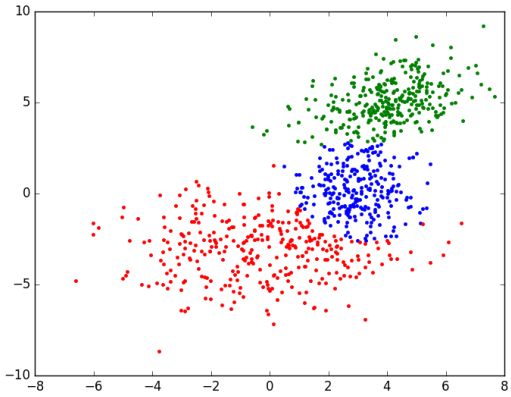

Figure 4: The result generated with reg= 0

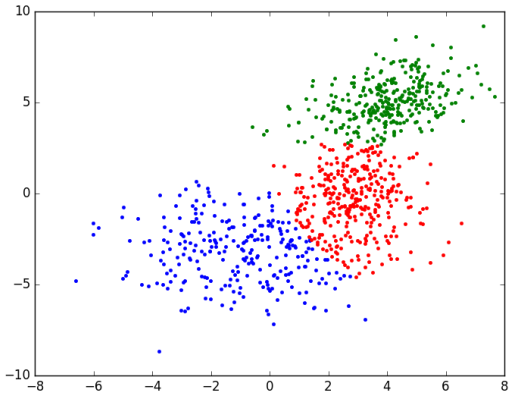

Figure 5: The result generated with reg= 0.2

As we can see, reg=0.2 produce a better result than reg = 0.

4 Homework 4

We implement the Levenberg-Marquardt method to solve the extremum problem.

We use $\sin(x)+\cos(y)$ and $\frac{1}{4}x^4-\frac{1}{3}x^3+\frac{1}{2}x^2-x+1$ as our target function, and visualize the first one.

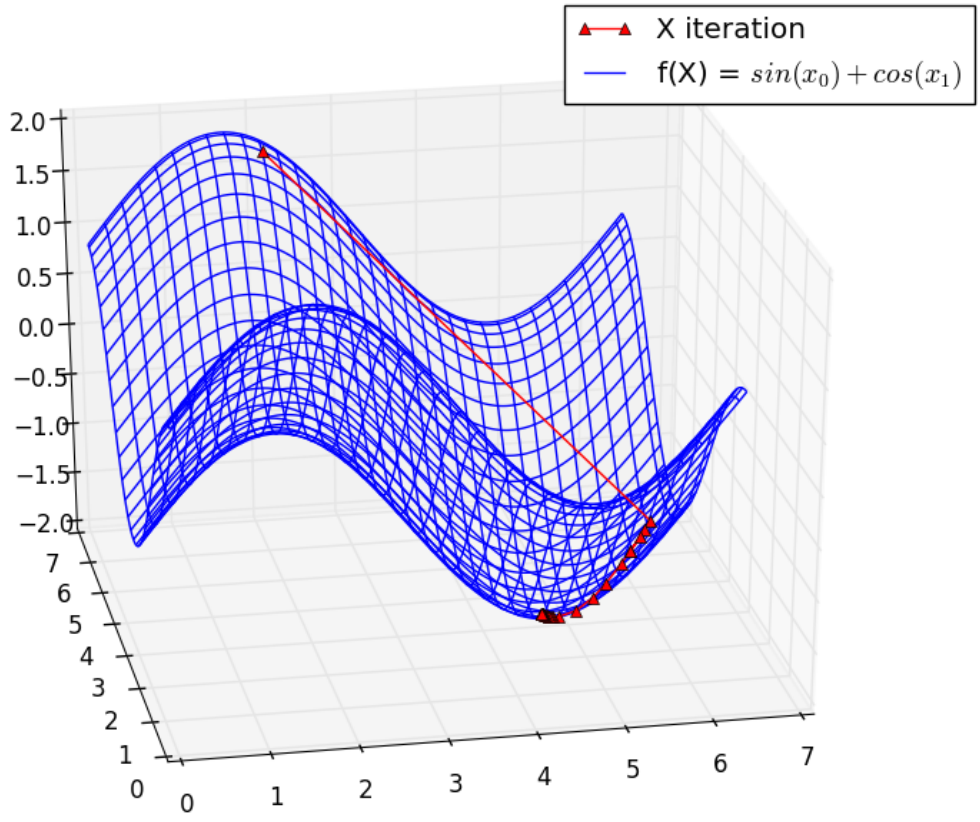

Figure 6: The result of homework 4

We observed a strange phenomenon: when we fix r = 1, it will converge much faster, we haven't figure it out till now.

5 Homework 5

We implement the primal SVM with active set method, since active set method needs a valid initial value, we use cvxopt to solve a linear programming to get a initial value.

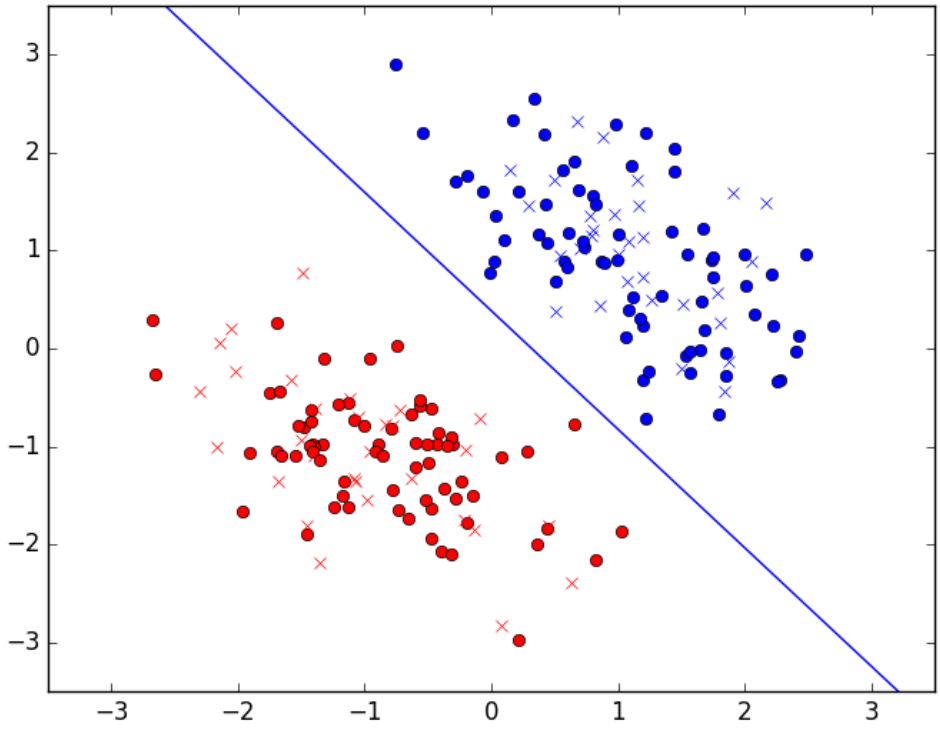

Figure 7: The result of homework 5