

# CSC411 Assignment 3

Yue Guo

December 5, 2017

## 1 20 News Group

### 1.1 Top 3 Algorithms

#### 1.1.1 Neural Network

1. Hyperparameter: number of layers

In the code `nn_news`, I have tried a single layer neural network vs multi-layered neural network. It turns out that the single neural network is the fastest and also most accurate.

2. Train/test loss

- Train accuracy 0.971451299275
- Test accuracy 0.632235793946

3. My expectations

This meets my expectation because NNs are good at working with a large dataset with many features. When I increase the number of hidden layers, accuracy decreases. I think this is because it overfits.

#### 1.1.2 Random forest

1. Hyperparameter: number of estimators

I have tried Ensemble with 10 to 150 estimators, and 150 works best after comparing results from cross validation.

2. Train/test loss

- Train accuracy 0.974721583878
- Test accuracy 0.59346787042

3. My expectations

This meets my expectations because Random Forest adds more randomness in each step, and ensemble method is more resistant to overfitting because it assigns weights to different features in each step. I also tried `decision_tree`, and it does not generalize well. The number of estimators learns better with a larger group of weak learners.

#### 1.1.3 SVM - Best Classifier

1. Hyperparameter: `rand_state`

In the code, I have tried different `rand_state` and cross validated each. It turns out `rand_state = 0` is the best

2. Train/test loss

- Train accuracy 0.972511932119
- Test accuracy 0.691980881572

### 3. Test confusion matrix

The most confused classes are class 16 talk.politics.guns and 18 talk.politics.misc

157.0	6.0	4.0	0.0	2.0	0.0	1.0	6.0	2.0	3.0	1.0	4.0	3.0	7.0	6.0	19.0	5.0	22.0	12.0	31.0
2.0	278.0	19.0	12.0	3.0	47.0	3.0	1.0	3.0	2.0	2.0	7.0	9.0	9.0	10.0	2.0	2.0	1.0	1.0	4.0
3.0	18.0	242.0	37.0	10.0	36.0	6.0	4.0	4.0	0.0	1.0	7.0	12.0	2.0	3.0	2.0	5.0	1.0	1.0	1.0
2.0	8.0	37.0	254.0	35.0	9.0	14.0	3.0	1.0	4.0	0.0	3.0	26.0	1.0	2.0	1.0	2.0	4.0	2.0	2.0
1.0	6.0	18.0	22.0	267.0	4.0	12.0	2.0	3.0	1.0	2.0	4.0	10.0	2.0	3.0	0.0	1.0	0.0	0.0	0.0
0.0	24.0	14.0	9.0	4.0	273.0	0.0	2.0	0.0	1.0	0.0	4.0	8.0	1.0	2.0	1.0	0.0	0.0	0.0	1.0
2.0	5.0	3.0	13.0	9.0	2.0	311.0	12.0	4.0	7.0	1.0	6.0	16.0	3.0	3.0	1.0	2.0	0.0	0.0	2.0
7.0	3.0	3.0	2.0	6.0	0.0	6.0	282.0	21.0	3.0	3.0	3.0	9.0	6.0	8.0	2.0	7.0	1.0	6.0	3.0
6.0	3.0	2.0	0.0	5.0	1.0	6.0	15.0	302.0	6.0	2.0	3.0	10.0	4.0	5.0	1.0	8.0	5.0	3.0	1.0
12.0	10.0	18.0	8.0	14.0	6.0	11.0	28.0	16.0	329.0	23.0	19.0	14.0	15.0	18.0	15.0	13.0	12.0	9.0	9.0
1.0	0.0	2.0	1.0	1.0	0.0	2.0	2.0	1.0	20.0	345.0	3.0	2.0	4.0	3.0	0.0	0.0	1.0	4.0	3.0
2.0	6.0	2.0	2.0	4.0	3.0	1.0	1.0	0.0	1.0	2.0	282.0	12.0	1.0	1.0	0.0	8.0	4.0	3.0	2.0
8.0	8.0	2.0	27.0	15.0	4.0	7.0	13.0	8.0	1.0	1.0	10.0	230.0	8.0	14.0	2.0	1.0	3.0	2.0	1.0
8.0	0.0	4.0	0.0	0.0	2.0	0.0	3.0	7.0	4.0	3.0	2.0	12.0	302.0	7.0	7.0	6.0	2.0	5.0	7.0
11.0	8.0	9.0	2.0	4.0	5.0	0.0	6.0	7.0	1.0	3.0	4.0	9.0	4.0	289.0	2.0	8.0	2.0	11.0	6.0
44.0	1.0	1.0	0.0	2.0	1.0	2.0	1.0	5.0	6.0	2.0	4.0	2.0	7.0	5.0	319.0	11.0	11.0	3.0	63.0
6.0	2.0	2.0	0.0	3.0	0.0	4.0	3.0	4.0	1.0	4.0	12.0	3.0	4.0	6.0	0.0	243.0	6.0	86.0	20.0
13.0	2.0	3.0	0.0	0.0	0.0	1.0	5.0	2.0	0.0	1.0	3.0	2.0	6.0	2.0	1.0	9.0	283.0	7.0	6.0
8.0	1.0	6.0	3.0	1.0	0.0	2.0	6.0	6.0	7.0	0.0	12.0	2.0	7.0	3.0	21.0	15.0	145.0	10.0	
26.0	0.0	3.0	0.0	0.0	2.0	1.0	1.0	2.0	0.0	3.0	4.0	2.0	3.0	0.0	20.0	12.0	3.0	10.0	79.0

### 4. My expectations

SVM is the best out of all. I think it is because it has a linear decision boundary and does not overfit on training data. The test accuracy is close to single neuron neural net, but still higher.

#### 1.1.4 Bernoulli Baseline

##### 1. Train/test loss

- Train accuracy 0.598727240587
- Test accuracy 0.457912904939

## 2 SVM

### 2.1 SVM test

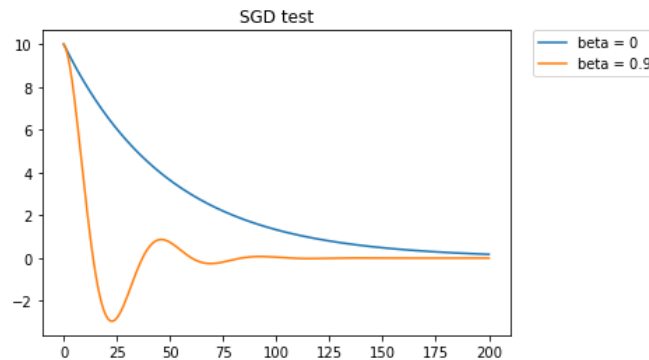


Figure 1: Plot of test SVM

### 2.2 SVM code

see code

## 2.3 SVM on MNIST

### 2.3.1 without momentum

1. Train loss= 0.400699029921
2. Test loss= 0.37243523202
3. classification accuracy on training set = 0.826985854189
4. classification accuracy on testing set = 0.818503401361

### 2.3.2 with momentum

1. classification accuracy on training set = 0.817555313747
2. classification accuracy on testing set = 0.809977324263

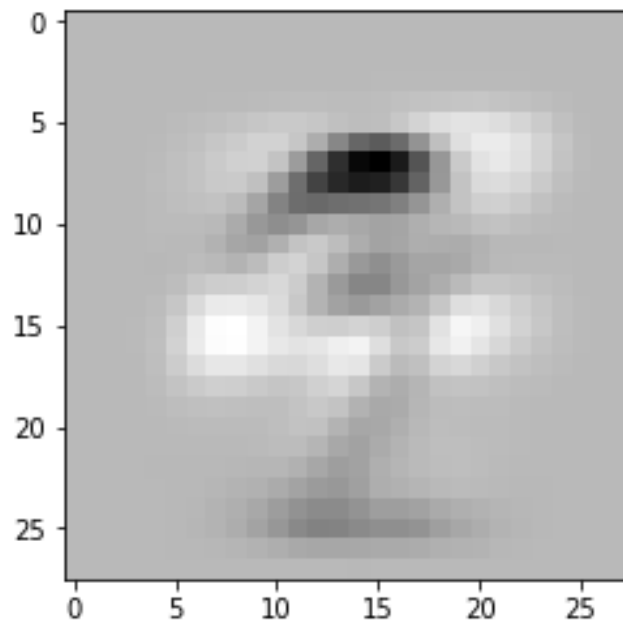


Figure 2: Plot momentum = 0

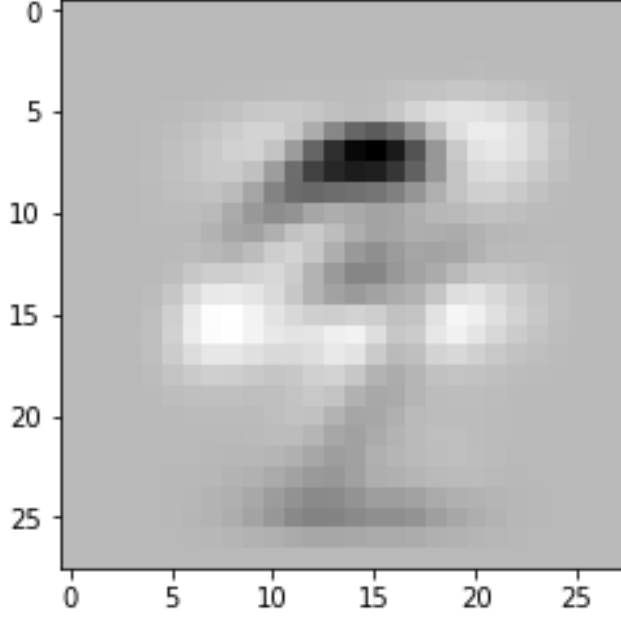


Figure 3: Plot momentum = 0.1

### 3 Kernels

#### 3.1 Positive semidefinite and quadratic form

Assume  $K$  is symmetric, we can decompose  $K$  into  $U\Lambda U^T$

$$x^T K x = x^T (U \Lambda U^T) x = (x^T U) \Lambda (U^T x)$$

$\Lambda$  has the eigenvalues  $\lambda_i$ , and if  $K$  is positive, and all  $\lambda_i > 0$ ,

$$x^T K x = \sum_{i=1}^d \lambda_i ([x^T U_i])^2 \geq 0$$

Then  $x^T K x \geq 0$  for all  $x$  in  $\mathbb{R}^d$  iff  $K$  is positive semidefinite

#### 3.2 Kernel properties

##### 3.2.1 $\alpha$

Define mapping  $\phi(x) = \sqrt{\alpha}$ ,  $\alpha > 0$ , and the kernel  $\langle \phi(x), \phi(y) \rangle = \alpha$ . The resulting matrix  $K$  has item  $K_{ij} = \alpha$ , the matrix  $K$  has equal number of row and columns, and element is  $\alpha$ . Since  $\alpha > 0$ , and all elements are equal,  $K$  is positive semidefinite

##### 3.2.2 $f(x), f(y)$

$$K_{ij} = \langle \phi(x), \phi(y) \rangle,$$

define  $\phi(x) = f(x), \forall f : \mathbb{R}^d \rightarrow \mathbb{R}$

define  $\phi(y) = f(y), \forall f : \mathbb{R}^d \rightarrow \mathbb{R}$

Since  $f(x)$  and  $f(y)$  produce a scalar,  $\langle \phi(x), \phi(y) \rangle = f(x) \cdot f(y)$

### 3.2.3 k1 and k2

If the gram matrix,  $K_1$  of kernel k1 and gram matrix,  $K_2$  of kernel k2 are positive semidefinite, by scaling them and adding each element, the new gram matrix of  $a \cdot k_1(x, y) + b \cdot k_2(x, y)$ , call it  $K$ , each element of  $K$  is positive since  $a, b > 0$ .

$K$  is also symmetric because  $K_1$  and  $K_2$  are symmetric with the same dimension, and element wise addition and linear combination preserve the symmetric property.

### 3.2.4

$$k(x, y) = \frac{k_1(x, y)}{\sqrt{k_1(x, x)}\sqrt{k_1(y, y)}}$$

Let  $\phi_1$  be the mapping defined by  $k_1$

We define a new mapping,  $\phi$  for  $k(x, y)$

We let  $\phi(x) = \frac{\phi_1(x)}{\|\phi_1(x)\|}$

$$\begin{aligned} k(x, y) &= \langle \phi(x), \phi(y) \rangle \\ &= \frac{\phi_1(x)}{\|\phi_1(x)\|} \cdot \frac{\phi_1(y)}{\|\phi_1(y)\|} \\ &= \frac{\phi_1(x)}{\sqrt{\phi_1(x) \cdot \phi_1(x)}} \cdot \frac{\phi_1(y)}{\sqrt{\phi_1(y) \cdot \phi_1(y)}} \\ &= \frac{\phi_1(x)}{(\sqrt{\phi_1(x) \cdot \phi_1(y)})} \cdot \frac{\phi_1(y)}{(\sqrt{\phi_1(x) \cdot \phi_1(y)})} \\ &= \frac{\phi_1(x)}{\sqrt{\phi_1(x) \cdot \phi_1(y)}} \cdot \frac{\phi_1(y)}{\sqrt{\phi_1(x) \cdot \phi_1(y)}} \\ k(x, y) &= \frac{k_1(x, y)}{\sqrt{k_1(x, x)}\sqrt{k_1(y, y)}} \end{aligned}$$

Therefore, there is a new mapping  $\phi(x)$  that supports  $k(x, y)$  and it is a kernel because  $\phi(x)$  is the product of two kernel mappings