CSC411 Assignment 3

Yue Guo

December 4, 2017

1 20 News Group

1.1 Top 3 Algorithms

1.1.1 Neural Network

1. Hypterparameter: 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 feartures. When I increase the number of hidden layers, accuracy decreases. I think this is because it overfits.

1.1.2 Random forest

1. Hypterparameter: number of estimators

   I have tried Ensamble 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 ensamble 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. Hypterparameter

   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 10 and 19

0.0	70.0	75.0	73.0	66.0	76.0	71.0	77.0	79.0	78.0	80.0	77.0	74.0	77.0	75.0	79.0	45.0	57.0	9.0	68.0
70.0	0.0	5.0	3.0	4.0	6.0	1.0	7.0	9.0	8.0	10.0	7.0	4.0	7.0	5.0	9.0	25.0	13.0	79.0	138.0
75.0	5.0	0.0	2.0	9.0	1.0	4.0	2.0	4.0	3.0	5.0	2.0	1.0	2.0	0.0	4.0	30.0	18.0	84.0	143.0
73.0	3.0	2.0	0.0	7.0	3.0	2.0	4.0	6.0	5.0	7.0	4.0	1.0	4.0	2.0	6.0	28.0	16.0	82.0	141.0
66.0	4.0	9.0	7.0	0.0	10.0	5.0	11.0	13.0	12.0	14.0	11.0	8.0	11.0	9.0	13.0	21.0	9.0	75.0	134.0
76.0	6.0	1.0	3.0	10.0	0.0	5.0	1.0	3.0	2.0	4.0	1.0	2.0	1.0	1.0	3.0	31.0	19.0	85.0	144.0
71.0	1.0	4.0	2.0	5.0	5.0	0.0	6.0	8.0	7.0	9.0	6.0	3.0	6.0	4.0	8.0	26.0	14.0	80.0	139.0
77.0	7.0	2.0	4.0	11.0	1.0	6.0	0.0	2.0	1.0	3.0	0.0	3.0	0.0	2.0	2.0	32.0	20.0	86.0	145.0
79.0	9.0	4.0	6.0	13.0	3.0	8.0	2.0	0.0	1.0	1.0	2.0	5.0	2.0	4.0	0.0	34.0	22.0	88.0	147.0
78.0	8.0	3.0	5.0	12.0	2.0	7.0	1.0	1.0	0.0	2.0	1.0	4.0	1.0	3.0	1.0	33.0	21.0	87.0	146.0
80.0	10.0	5.0	7.0	14.0	4.0	9.0	3.0	1.0	2.0	0.0	3.0	6.0	3.0	5.0	1.0	35.0	23.0	89.0	148.0
77.0	7.0	2.0	4.0	11.0	1.0	6.0	0.0	2.0	1.0	3.0	0.0	3.0	0.0	2.0	2.0	32.0	20.0	86.0	145.0
74.0	4.0	1.0	1.0	8.0	2.0	3.0	3.0	5.0	4.0	6.0	3.0	0.0	3.0	1.0	5.0	29.0	17.0	83.0	142.0
77.0	7.0	2.0	4.0	11.0	1.0	6.0	0.0	2.0	1.0	3.0	0.0	3.0	0.0	2.0	2.0	32.0	20.0	86.0	145.0
75.0	5.0	0.0	2.0	9.0	1.0	4.0	2.0	4.0	3.0	5.0	2.0	1.0	2.0	0.0	4.0	30.0	18.0	84.0	143.0
79.0	9.0	4.0	6.0	13.0	3.0	8.0	2.0	0.0	1.0	1.0	2.0	5.0	2.0	4.0	0.0	34.0	22.0	88.0	147.0
45.0	25.0	30.0	28.0	21.0	31.0	26.0	32.0	34.0	33.0	35.0	32.0	29.0	32.0	30.0	34.0	0.0	12.0	54.0	113.0
57.0	13.0	18.0	16.0	9.0	19.0	14.0	20.0	22.0	21.0	23.0	20.0	17.0	20.0	18.0	22.0	12.0	0.0	66.0	125.0
9.0	79.0	84.0	82.0	75.0	85.0	80.0	86.0	88.0	87.0	89.0	86.0	83.0	86.0	84.0	88.0	54.0	66.0	0.0	59.0
68.0	138.0	143.0	141.0	134.0	144.0	139.0	145.0	147.0	146.0	148.0	145.0	142.0	145.0	143.0	147.0	113.0	125.0	59.0	0.0

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

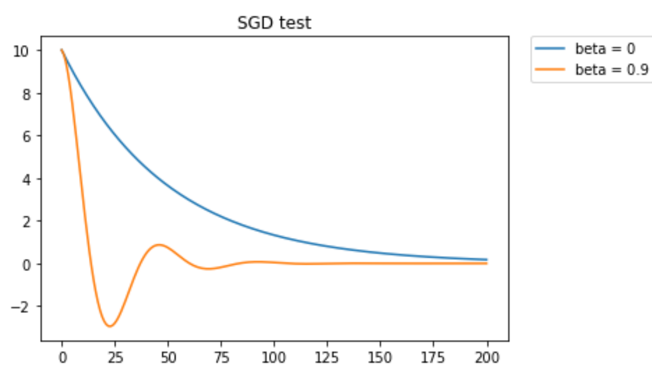

Figure 1: Plot of test SVM

2.2 SVM code

see code

2.3 SVM on MINIST

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

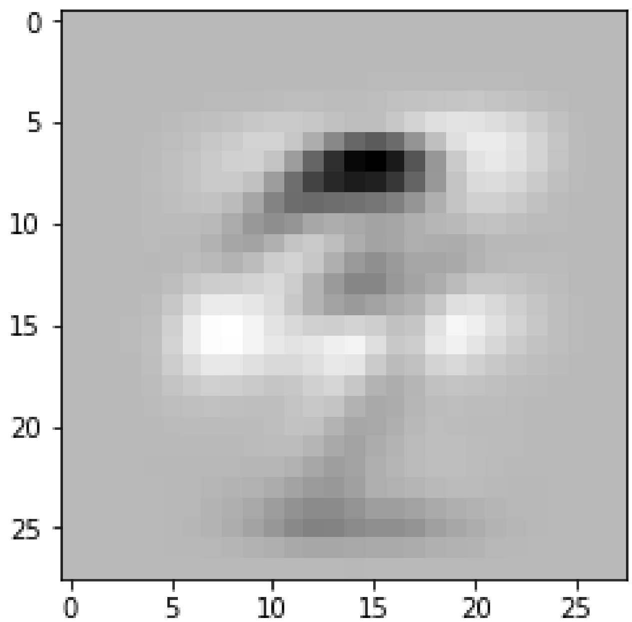

Figure 2: Plot momentum = 0

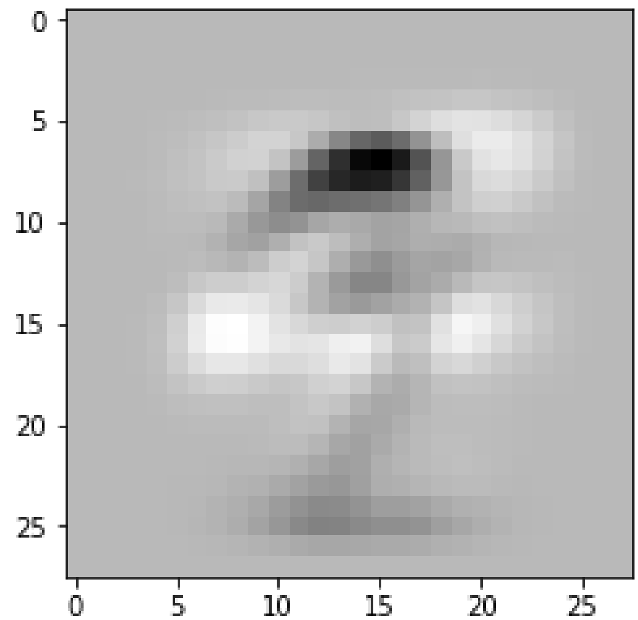

Figure 3: Plot momentum = 0.1

3 Kernels

3.1 Positive semidefinite and quadratic form

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

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

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

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

Then $x^T K x >= 0$ for all x in $\mathbb{R}^d$ iff K is postive semidefinite

3.2 Kernel properties

3.2.1 α

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 α. 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 $\phi(x) = f(x), \forall f : \mathbb{R}^d \to \mathbb{R}$
define $\phi(y) = f(y), \forall f : \mathbb{R}^d \to \mathbb{R}$
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 ϕ_1 be the mapping defined by k_1
We define a new mapping, ϕ for $k(x, y)$
We let $\phi(x) = \frac{\phi_1(x)}{\|\phi_1(x)\|}$

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

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