CSC 411 Assignment 3

1. 20 Newsgroups predictions

You need to report the results of three different algorithms (you are encouraged to try out more and pick 3 out of them) and the baseline.

BaseLine:
BernoulliNB baseline train accuracy = 0.5987272405868835
BernoulliNB baseline test accuracy = 0.4579129049389272

Picked Models:
Multinomial Naïve Bayer's (MultinomialNB)
Logistic Regression
SVM with Linear Kernel (LinearSVC)

MultinomialNB train accuracy = 0.9589004772847799
MultinomialNB test accuracy = 0.7002124269782263
The two classes that classifier was most confused about: (15, 19)

Logistic Regression train accuracy = 0.9594307937069118
Logistic Regression test accuracy = 0.6895910780669146
The two classes that classifier was most confused about: (16, 18)

LinearSVC train accuracy = 0.9588120912144246
LinearSVC test accuracy = 0.7003451938396177
The two classes that classifier was most confused about: (16, 18)

Explain in your report how you picked the best hyperparameters.
The model I choose is mainly one parameter driven even though it comes with more parameters in the sklearn library. I used a lot of combinations and finally picked the most import one to tune, C(Inverse of regularization strength) in Logistic regression and C(Penalty parameter) in SVM, alpha in Naïve Bayer's Multinomial Model. I firstly found the default parameters as set in the sklearn. I increase and decrease it to see the improvement or degradation on the performance. And then I keep doing it for the benefit of higher performance. Then it comes out my best hyperparameters as shown in the code.

Explain why you picked these 3 methods, did they work as you thought? Why/Why not?
I did run a few other models which has been commented out in my code in order to see the difference between them. Honestly, I choose these three because they are the best compared to the rest.

Multinomial Naïve Bayer's (MultinomialNB):

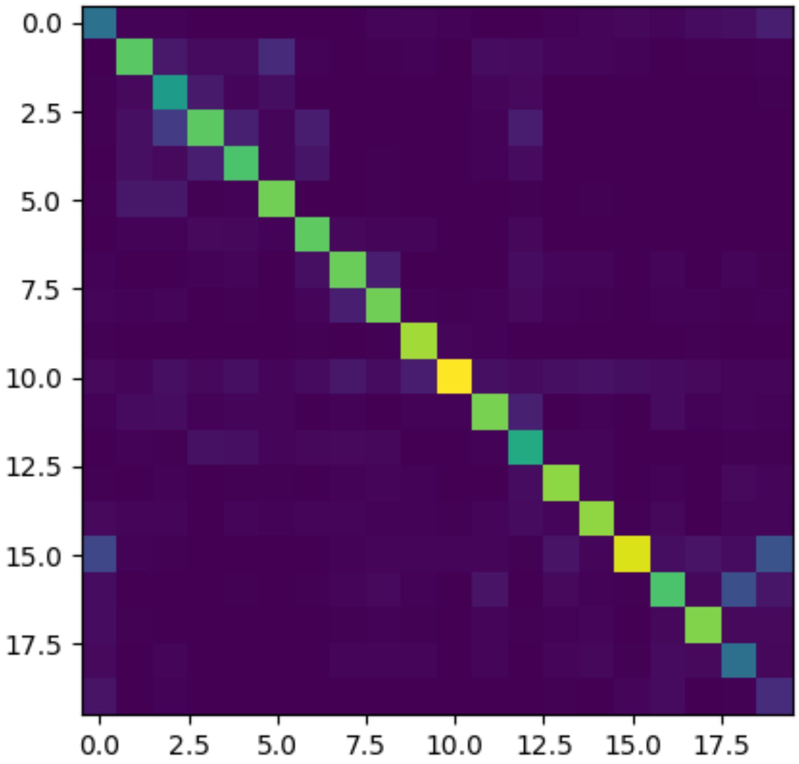
The one outperforms the others is the Naïve Bayer's with Multinomial Model. It has above 70% accuracy on the test set which is the highest among the all. The reason behind this is practically, this model works well with document classification with tf-idf applied. And also it is very fast to train. We have more than 100K features and 10K training data. By such large training set, the languages itself would reveal some sorts of patterns for the model to follow and fit, thus yielding the second-best result here.

Logistic Regression:
Logistic Regression is able to linearly separate data. It also requires less time to train. A fact that cannot be ignored that the newsgroups is not easily separable since articles would overlap with each other for many words even though their theme does not come across each other. Thanks to the large data set provided, it is able to gain more information from features and separate them out. More related words could be grouped together under this context to get a better prediction. Thus, the performance is third-best.

SVM with Linear Kernel (LinearSVC):
SVM with linear kernel is fast and can do achieve a good performance on large dataset and features. The reason we don't use non-linear kernels is it does not outperform the linear one and does take much time. It functions similarly when using linear kernel. So the same reasoning from Logistic Regression partially applies here. It is the best model of the three.

For your best classifier compute and show the confusion matrix, which is a k×k matrix where Cij is the number of test examples belonging to class j that were classified as i. This part you need to write yourselves.

```
[[141   4   4   0   0   0   0   1   6   5   4   1   1   4   5   8   5  12 16  31]
 [  1 278  26  11  11  46   3   1   2   3   0  12  11   6   7   3   0   2 2   3]
 [  2  10 205  27   7  14   1   1   1   0   0   5   9   0   1   1   0   0 0   2]
 [  2  16  65 279  33   7  30   1   1   0   0   3  28   1   1   0   1   1 0   1]
 [  1  16  10  32 268   6  21   0   2   0   0   3  11   0   0   1   0   0 0   0]
 [  2  24  24   2   3 293   0   0   2   1   1   1   2   0   2   1   1   1 0   0]
 [  0   4   3   9   8   4 280   8   5   5   0   1   8   1   0   0   1   0 1   0]
 [  3   0   1   4   6   0  15 289  28   0   1   0  12   7   6   0   6   1 5   2]
 [  5   3   5   0   2   0   7  31 292   4   2   4   9   3   2   1   3   4 2   4]
 [  2   1   0   0   0   1   2   0   2 321   5   3   1   0   1   1   1   2 1   1]
 [ 10   5  16   8  15   5  11  24  13  30 373  16  11  16  18  14  11  10 7   7]
 [  4  11  12   3   6   7   1   4   0   4   3 298  33   0   3   1  11   3 5   4]
 [  1   3   2  17  15   5   8   9   8   1   0   3 228   5   6   0   1   1 2   2]
 [  2   1   3   0   2   2   1   3   6   4   1   1  13 309   5   1   4   0 10   7]
 [ 10   7   7   0   6   3   6   6   4   3   2   5  11   6 311   2   8   0 7   5]
 [ 79   4   2   0   1   1   1   3   6   5   6   7   2  19   6 354  14  19 12  96]
 [ 12   0   0   0   2   1   2   6  10   3   1  19   0   9   4   2 267   8 91  22]
 [ 13   2   1   0   0   0   0   2   3   2   0   6   2   3   7   0   8 303 9   9]
 [ 10   0   5   0   0   0   1   7   6   6   0   7   1   5   8   2  11   9 138   8]
 [ 19   0   3   0   0   0   0   0   1   0   0   1   0   2   1   6  11   0 2  47]]
```



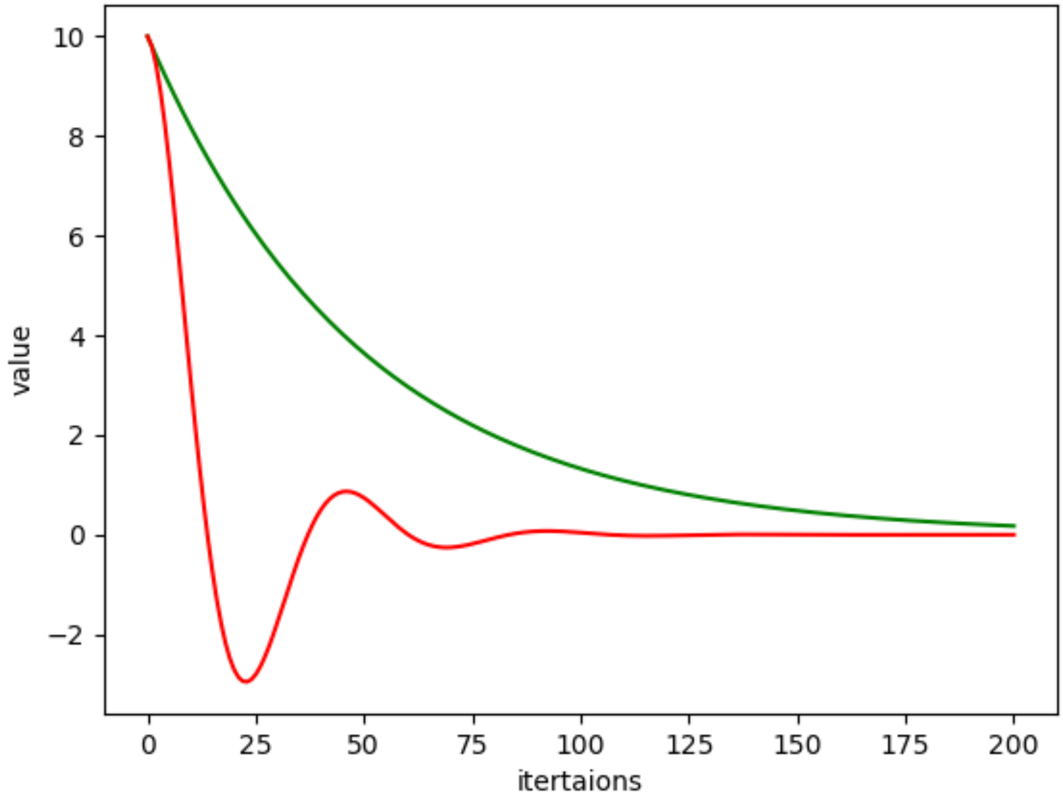
What were the two classes your classifier was most confused about?
(15, 19) as reported by MultinomialNB Model or (16,18) as reported both by Logistic Regression and LinearSVC Model.

2. Training SVM with SGD

2.1 SGD With Momentum

1. Implement SGD with momentum.
As shown in q2.py

2. To verify your implementation find the minimum of $f(w) = 0.01w^2$ using gradient descent. Take w0 = 10.0 and set your learning rate to $\alpha = 1.0$. Plot w_t for 200 time-steps using $\beta = 0.0$ and $\beta = 0.9$ on the same graph.



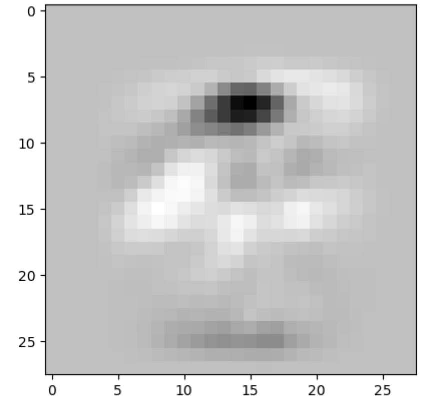
2.2 Training SVM

Code is as shown in q2.py

2.3 Apply on 4-vs-9 digits on MNIST

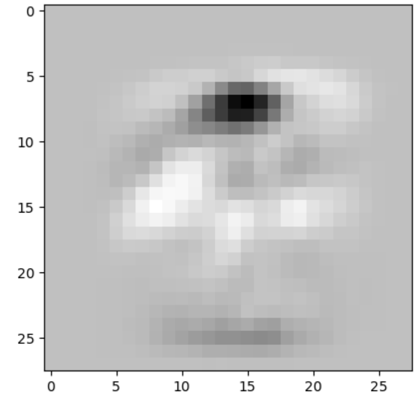
Beta = 0:

1. The training loss: 0.606868786012
2. The test loss: 0.610163921886
1. The training loss (hinge loss average): 0.396922546385
2. The test loss (hinge loss average): 0.400217682259
3. The classification accuracy on the training set: 0.9138321995464853
4. The classification accuracy on the test set: 0.9147624229234675
5. Plot w as a 28 × 28 image.



Beta = 0.1:

1. The training loss: 0.588971779765
2. The test loss: 0.589853506564
1. The training loss (hinge loss average): 0.373546602765
2. The test loss (hinge loss average): 0.374428329564
3. The classification accuracy on the training set: 0.9251700680272109
4. The classification accuracy on the test set: 0.9249183895538629
5. Plot w as a 28 × 28 image.



3.1 I. A symmetric matrix $K \in R^{d \times d}$ is positive semidefinite $\Rightarrow \forall x \in R^d, x^T K x \geq 0$

By properties of symmetric matrix, we have

$K = QAQ^T$ (Q is orthogonal matrix, A is diagonal matrix with entries to be eigenvalues of K)

$\forall x \in R^d$, $x^T k x = x^T Q A Q^T x \quad \cdots ①$

Let $P = x^T Q$, then $① = PAP^T = P \begin{bmatrix} \lambda_1 & \cdots & \\ \vdots & \lambda_2 & \vdots \\ \vdots & \cdots & \lambda_d \end{bmatrix} P^T = \sum_{i=1}^{d} \lambda_i P_i^2 \quad \cdots ②$

And from the definition of positive semidefinite, we have $\lambda_i > 0$, and we also know $P_i^2 \geq 0$, then we know $② \geq 0 \Rightarrow x^T k x \geq 0$, proof done

II. $\forall x \in R^d, x^T k x \geq 0 \Rightarrow$ A symmetric matrix $K \in R^{d \times d}$ is positive semidefinite

Above is equal to prove its contrapositive, which is

$\forall$ symmetric matrix $K \in R^{d \times d}$ is not positive semidefinite $\Rightarrow \exists x \in R^d, x^T k x < 0$

Then we have $\lambda < 0$ such that $\exists x \in R^d, \lambda x = k x \ (x \neq \vec{0}) \cdots ①$

$① \Rightarrow x^T \lambda x = x^T k x \Rightarrow \lambda (x^T x) = x^T k x$

since $\lambda < 0$, $x^T x > 0$, then we have $x^T k x = \lambda (x^T x) < 0$

Contrapositive is true, then II. is true, proof done.

3.2.1 $\forall x$, let $\phi(x) = \sqrt{2}$ then we have $k(x,y) = \langle \phi(x) . \phi(y) \rangle = \sqrt{2} . \sqrt{2} = 2$

2. for all $f : R^d \to R$, we can construct $W = [f(x^{(1)}), f(x^{(2)}), \dots, f(x^{(n)})]$ such that $K = W^T W$, and $\forall y \in R^d$, we have $Y^T K Y = Y^T W^T W Y = (WY)^2 \geq 0$ From what we have from 3.1.1 we show that k is positive semedefinite, then $k(x,y) = f(x) . f(y)$ is a kernel for all $f : R^d \to R$

3. $k_1(x,y) = \langle \phi_1(x), \phi_1(y) \rangle$ $\quad k_2(x,y) = \langle \phi_2(x), \phi_2(y) \rangle$

$ak_1(x,y) + bk_2(x,y) = \langle \sqrt{a}\phi_1(x), \sqrt{a}\phi_1(y) \rangle + \langle \sqrt{b}\phi_2(x), \sqrt{b}\phi_2(y) \rangle$

$= \langle [\sqrt{a}\phi_1(x), \sqrt{b}\phi_2(x)], [\sqrt{a}\phi_1(y), \sqrt{b}\phi_2(y)] \rangle$

(properties of concatenation of feature maps)

Then we can let $\phi(x) = [\sqrt{a}\phi_1(x), \sqrt{b}\phi_2(x)]$

$\Rightarrow k(x,y) =$ ~~$ak_1(x,y) + bk_2(x,y)$~~ $\langle \phi(x), \phi(y) \rangle$

$= \langle [\sqrt{a}\phi_1(x), \sqrt{b}\phi_2(x)], [\sqrt{a}\phi_1(y), \sqrt{b}\phi_2(y)] \rangle$

$= a \cdot k_1(x,y) + b\, k_2(x,y)$ is a kernel

4. $k(x,y) = \dfrac{k_1(x,y)}{\sqrt{k_1(x,x)}\sqrt{k_1(y,y)}} = \dfrac{<\phi_1(x), \phi_1(y)>}{\sqrt{<\phi_1(x), \phi_1(x)>}\sqrt{<\phi_1(y), \phi_1(y)>}}$

$$= \frac{<\phi_1(x), \phi_1(y)>}{\sqrt{\phi_1(x_1)^2 + \cdots + \phi_1(x_d)^2}\sqrt{\phi_1(y_1)^2 + \cdots \phi_1(y_d)^2}}$$

$$= \frac{<\phi_1(x), \phi_1(y)>}{\|\phi_1(x)\| \cdot \|\phi_1(y)\|}$$

then we can let $\phi(x) = \dfrac{\phi_1(x)}{\|\phi_1(x)\|}$, we have

$$<\phi(x), \phi(y)> = \frac{<\phi_1(x), \phi_1(y)>}{\|\phi_1(x)\| \cdot \|\phi_1(y)\|}$$

$\Rightarrow k(x,y) = \dfrac{<\phi_1(x), \phi_1(y)>}{\|\phi_1(x)\| \cdot \|\phi_1(y)\|} = <\phi(x), \phi(y)>$ is a kernel