December 3, 2017

Vibhavi Peiris

Student Number: 1000597687

CSC411

Assignment 3

***Part 1: Newsgroups***

1. I used Logistic Regression, random Forest & multinomial Naive Bayes. I tried to use Svm and neural nets but it was taking too long.
2. The accuracies for each model including baseline.

Baseline: BernoulliNB train accuracy = 0.598727240587

Baseline: BernoulliNB test accuracy = 0.457912904939

MultinomialNB train accuracy = 0.811384125862

MultinomialNB test accuracy = 0.606213489113

Random Forest train accuracy = 0.790701785399

Random Forest test accuracy = 0.535050451407

Logistic train accuracy = 0.895704436981

Logistic test accuracy = 0.67750929368

1. The zero-one losses for each model including baseline.

Baseline: BernoulliNB train zero-one loss = 0.401272759413

Baseline: BernoulliNB test zero-one loss = 0.542087095061

MultinomialNB train zero-one loss = 0.188615874138

MultinomialNB test zero-one loss = 0.393786510887

Random Forest train zero-one loss = 0.209298214601

Random Forest test zero-one loss = 0.464949548593

Logistic train zero-one loss = 0.104295563019

Logistic test zero-one loss = 0.32249070632

1. For Logistic Regression & multinomial Naive Bayes I used the default parameters, which got accuracies above the baseline. However, for Random forest I had to use grid search to tune and find the best parameters in order to get an accuracy above the baseline.
2. I picked these methods because the where fast and would have an accuracy at least equal to the baseline.
3. Confusion matrix for Logistic regression: The model with the best accuracy

[[141 1 1 0 2 4 2 4 9 18 1 2 6 10 17 63 13 10 5 10]

[ 5 272 19 8 7 16 8 3 5 8 0 5 14 3 11 3 0 1 1 0]

[ 3 25 240 33 17 11 5 1 3 24 1 4 2 4 11 1 3 2 3 1]

[ 1 13 38 243 28 4 12 5 1 8 0 2 34 0 2 0 0 0 1 0]

[ 0 7 8 27 256 4 12 9 7 15 3 2 25 0 8 1 1 0 0 0]

[ 0 46 29 7 4 271 11 1 1 6 0 2 6 2 4 4 0 0 1 0]

[ 0 3 2 17 12 0 313 10 6 11 0 1 8 1 4 1 0 1 0 0]

[ 2 2 1 1 0 2 13 278 14 29 1 1 26 3 9 0 5 3 5 1]

[ 3 1 0 1 0 0 8 24 310 19 0 0 9 6 6 2 5 0 4 0]

[ 7 3 0 0 1 1 7 0 6 330 26 0 3 2 3 5 0 0 3 0]

[ 5 1 0 1 0 1 0 1 4 26 346 1 2 4 2 1 1 3 0 0]

[ 3 7 5 2 5 6 4 6 9 27 0 261 10 10 6 2 20 4 8 1]

[ 2 15 10 20 8 3 21 10 11 16 1 16 238 10 8 2 1 1 0 0]

[ 6 11 1 1 1 0 11 10 6 15 3 0 9 301 6 3 1 4 7 0]

[ 4 11 2 0 1 1 6 9 3 21 2 0 17 10 290 2 4 2 9 0]

[ 22 2 3 0 0 1 2 1 5 19 1 0 3 11 4 313 2 2 3 4]

[ 6 1 2 1 2 1 4 7 9 21 1 5 3 4 7 9 255 4 18 4]

[ 22 2 1 0 0 1 3 2 11 14 1 3 4 4 2 9 8 276 13 0]

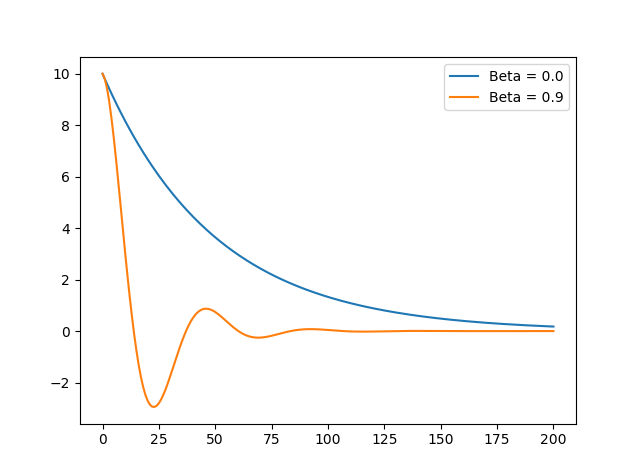
[ 18 1 0 0 0 1 2 7 10 13 4 1 3 7 11 4 91 8 127 2]

[ 45 4 2 0 0 0 6 5 7 10 2 1 1 15 5 67 21 11 7 42]]

1. The 2 classes where most confused from (classes 1…20)
   1. class 17 being mistaken for 19: 91 times
   2. class 16 being mistaken for 20: 67 times

***Part 2: Train SVM***

1. Plot for weights history



1. Code in the file
2. Digit classification
   1. Model 1 with beta 0.0:

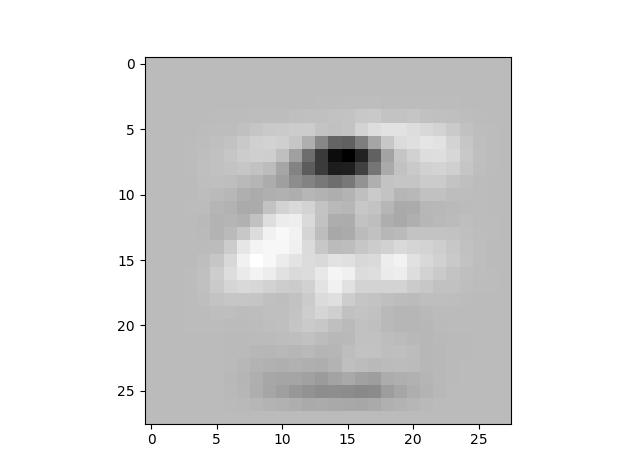
SVM Train with lr = 0.05 & beta = 0.0

Train accuracy = 0.915011337868

Test accuracy = 0.916938701487

Train hinge loss = 0.39833051008

Test hinge loss = 0.401406119389



* 1. Model 2: with beta 0.1

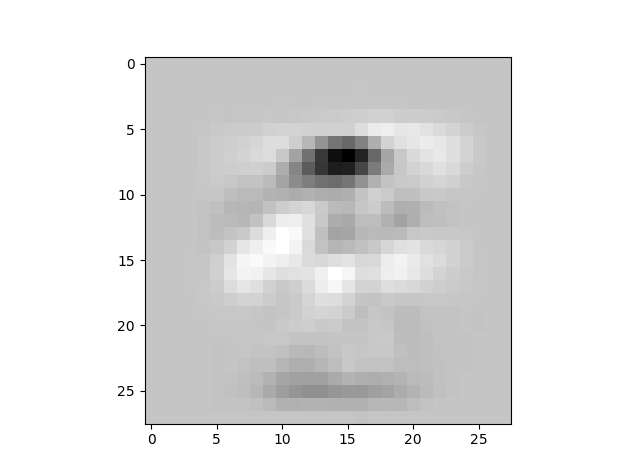
SVM Train with lr = 0.05 & beta = 0.1

Train accuracy = 0.89306122449

Test accuracy = 0.894087776569

Train hinge loss = 0.363258049103

Test hinge loss = 0.350357622194



***Part 3: Kernals***

Question 3.1:



Question 3.2.1:



Question 3.2.2:



Question 3.2.3:



Question 3.2.4:

