Machine Learning Lab 5 Report

1. Naive Bayes Classifier

Using the Naive Bayes Classification method with a = b = 1 on the MNIST dataset, we get a Loss of 0.1361.

We observe that the loss is not affected much by the P(Y= i) factor in the Naive Bayes formula, the reason being the probability of the occurrence of a particular Y is constant, i.e. 1/10 in this case.

2. 5-fold cross-validation

After performing the Naive Bayes Classification with 5-fold cross-validation, we get the following losses corresponding to different values of a and b.

a: 1, b: 1, avg_loss: 0.15026307644692047
a: 1, b: 5, avg_loss: 0.15081708449396475
a: 1, b: 10, avg_loss: 0.15192974311358715
a: 1, b: 20, avg_loss: 0.1519281956050758
a: 1, b: 80, avg_loss: 0.15359950479727638
a: 1, b: 100, avg_loss: 0.15304240173320952
a: 1, b: 1000, avg_loss: 0.18309656453110493
a: 5, b: 1, avg_loss: 0.1530454967502321
a: 5, b: 5, avg_loss: 0.15360259981429897
a: 5, b: 10, avg_loss: 0.1569374806561436

a: 5, b: 20, avg_loss: 0.158050139275766

a: 5, b: 80, avg_loss: 0.1608325595790777

a: 5, b: 100, avg_loss: 0.15971990095945526

a: 5, b: 1000, avg_loss: 0.19477561126586196

a: 10, b: 1, avg_loss: 0.15638502011761063

a: 10, b: 5, avg_loss: 0.15527081398947695

a: 10, b: 10, avg_loss: 0.15749767873723303

a: 10, b: 20, avg_loss: 0.16139121015165583

a: 10, b: 80, avg_loss: 0.16472763850201175

a: 10, b: 100, avg_loss: 0.16472763850201178

a: 10, b: 1000, avg_loss: 0.20479727638502015

a: 20, b: 1, avg_loss: 0.1580547818012999

a: 20, b: 5, avg_loss: 0.15749767873723303

a: 20, b: 10, avg_loss: 0.1580578768183225

a: 20, b: 20, avg_loss: 0.15861343237387807

a: 20, b: 80, avg_loss: 0.16584648715567935

a: 20, b: 100, avg_loss: 0.16917982048901264

a: 20, b: 1000, avg_loss: 0.2253930671618694

a: 80, b: 1, avg_loss: 0.168628907458991

a: 80, b: 5, avg_loss: 0.16807180439492417

a: 80, b: 10, avg_loss: 0.16695914577530177

a: 80, b: 20, avg_loss: 0.16584184463014545

a: 80, b: 80, avg_loss: 0.16862426493345714

a: 80, b: 100, avg_loss: 0.17029402661714638

a: 80, b: 1000, avg_loss: 0.2626663571649644

a: 100, b: 1, avg_loss: 0.16807335190343548

a: 100, b: 5, avg_loss: 0.16807180439492417

a: 100, b: 10, avg_loss: 0.16862581244196845

a: 100, b: 20, avg_loss: 0.16973692355307954

a: 100, b: 80, avg_loss: 0.16862426493345714

a: 100, b: 100, avg_loss: 0.1736335499845249

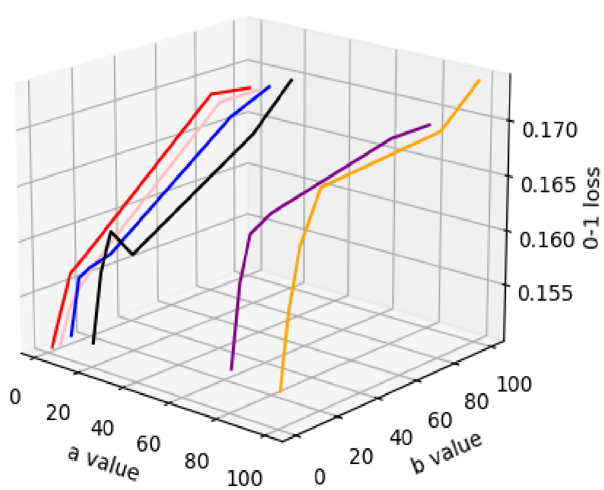
a: 100, b: 1000, avg_loss: 0.27101361807489943

a: 1000, b: 1, avg_loss: 0.18309192200557103
a: 1000, b: 5, avg_loss: 0.18142061281337046
a: 1000, b: 10, avg_loss: 0.1808635097493036
a: 1000, b: 20, avg_loss: 0.1808619622407923
a: 1000, b: 80, avg_loss: 0.1752955741256577
a: 1000, b: 100, avg_loss: 0.1786366450015475
a: 1000, b: 1000, avg_loss: 0.24429743113587127

Minimum loss: 0.15026307644692047
optimal a: 1
optimal b: 1

Observation:

For a given value of 'b', the loss generally increases when 'a' increases and vice-versa. The same happens when we fix 'a' i.e. for a fixed value of 'a', loss generally increases when 'b' increases.



0-1 loss for fixed 'a'

3. K Nearest Neighbours

After implementing K Nearest Neighbours for different k, we get the following losses:

K= 3, Loss = 0.0888

K= 5, Loss = 0.0861

Observation:

For random state = 42, the loss for K= 5 is less than the loss for K= 3.

However, for some random states, the loss for K= 5 is greater.