HOMEWORK 4

$YangGao$
9083410275

Solution 1

Strategy 1 $E[1[\hat{x} \neq x]] = \sum_{x=1}^{k}(1 * P(\hat{x} \neq x) + 0 * P(\hat{x} \neq x))$
k = 1 in this case as there is only a single observation
$= 1 - P(\hat{x} \neq x) = 1 - \theta_{max}$
Strategy 2 $E[1[\hat{x} \neq x]] = \sum_{x=i}^{k} P(x = i, \hat{x} \neq i)$
by independence
$= \sum_{x=i}^{k} P(x = i)P(\hat{x} \neq i)$
$= \theta_i(1 - \theta_i)$

Solution 2

To miminize loss, minimize $\sum_{i=1}^{k} c_{ij}\theta_i$
$\rightarrow argmin(\sum_{i=1}^{k} c_{ij}\theta)$

Solution 3

Solution 3 (i)

If we knew the distribution, I would pick the either 0 or 1 at all times depending on which is closest to mean to minimize the squared difference. E.g. if mean is 0.6 I would pick 1, if mean is 0.3 I would pick 0 at times. Proof that mean minimizes squared error: $error = (x_t - y_t)^2$ The expectation of the error is (μ is the mean):
$E[(x_t - y_t)^2]$
$= E[(x_t - \mu + \mu - y_t)^2]$
$= E[(x_t - \mu)^2 + 2(\mu - y_t)(x_t - \mu) + (\mu - y_t)^2]$
$= (x_t - \mu)^2 + 2E[(\mu - x_t)(y_t - \mu)] + E[(\mu - y_t)^2]$
$= (x_t - \mu)^2 + 2(\mu - x_t)(E[y_t] - \mu) + E[(\mu - y_t)^2]$
$= (x_t - \mu)^2 + 0 + E[(\mu - y_t)^2]$
$= (x_t - \mu)^2 + E[(\mu - y_t)^2]$
Expectation of error is minimized when $x_t = \mu$ and the first term becomes 0.

Solution 3 (ii)

let n be the number of samples we have observed, then the strategy is choose one of the guesses and compute:
$L = \frac{1}{n}\sum_{i=1}^{n}(x_t - y_t)^2 - \sigma^2 n$

Solution 4

Solution 4.1

priors are [0.33, 0.33, 0.33]

Solution 4.2

order is space then alphabetical
$\theta_e =$
[0.1792499586981662,
0.0601685114819098,
0.011134974392863043,
0.021509995043779945,
0.021972575582355856,
0.1053692383941847,
0.018932760614571286,
0.017478936064761277,
0.047216256401784236,
0.055410540227986124,
0.001420783082768875,
0.0037336857756484387,
0.028977366595076822,
0.020518751032545846,
0.057921691723112505,
0.06446390219725756,
0.01675202378985627,
0.0005617049396993227,
0.053824549810011564,
0.06618205848339666,
0.08012555757475633,
0.026664463902197257,
0.009284652238559392,
0.015496448042293078,
0.001156451346439782,
0.013844374690236246,
0.0006277878737815959]

Solution 4.3

$\theta_j =$
[0.12344945665466997,
0.1317656102589189,
0.010866906600510151,
0.005485866033054963,
0.01722631818022992,
0.06020475907613823,
0.003878542227191726,
0.014011670568503443,
0.03176211607673224,
0.09703343932352633,
0.0023411020650616725,
0.05740941332681086,
0.001432614696530277,
0.03979873510604843,
0.05671057688947902,
0.09116321324993885,
0.0008735455466648031,
0.00010482546559977637,
0.04280373178657535,
0.0421747789929767,
0.056990111464411755,
0.07061742199238269,
0.0002445927530661449,
0.01974212935462455,

3.4941821866592126e-05,
0.01415143785596981,
0.00772214263251686]
$\theta_s =$
[0.16826493170115014,
0.10456045141993771,
0.008232863618143134,
0.03752582405722919,
0.039745922111559924,
0.1138108599796491,
0.00860287996053159,
0.0071844839813758445,
0.0045327001942585795,
0.049859702136844375,
0.006629459467793161,
0.0002775122567913416,
0.052943171656748174,
0.02580863988159477,
0.054176559464709693,
0.07249236841293824,
0.02426690512164287,
0.007677839104560451,
0.05929511886774999,
0.06577040485954797,
0.03561407295488884,
0.03370232185254849,
0.00588942678301625,
9.250408559711388e-05,
0.0024976103111220747,
0.007862847275754679,
0.0026826184823163022]

Solution 4.4

[498,
164,
32,
53,
57,
311,
55,
51,
140,
140,
3,
6,
85,
64,
139,
182,
53,
3,
141,
186,
225,
65,
31,
47,

4,
38,
2]

Solution 4.5

$\hat{p}(x|y=e) = e^{-7843}$
$\hat{p}(x|y=j) = e^{-8773}$
$\hat{p}(x|y=s) = e^{-8468}$

Solution 4.6

English: $e^{-7843}/(e^{-7843}+e^{-8773}+e^{-8468}) = 1$
Japanese: $e^{-8773}/(e^{-7843}+e^{-8773}+e^{-8468}) = 0$
Spanish:$e^{-8468}/(e^{-7843}+e^{-8773}+e^{-8468}) = 0$
Predicted class label: English

Solution 4.7

[[10, 0, 0],
[0, 10, 0],
[0, 0, 10]]

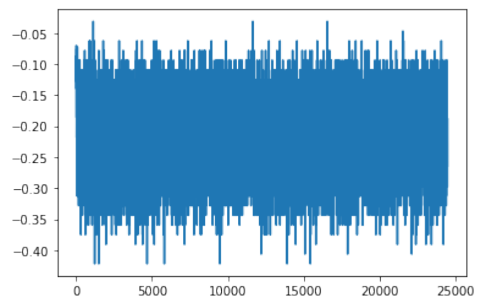
Solution 4.8

No, the order of the characters does not affect the prediction. The key mathematical step is making the bag-of-words count vector. Since this step is only counting the occurrence of characters, the information about order is not included in features fed into the classifier..

Solution 5

Pytorch Implementation

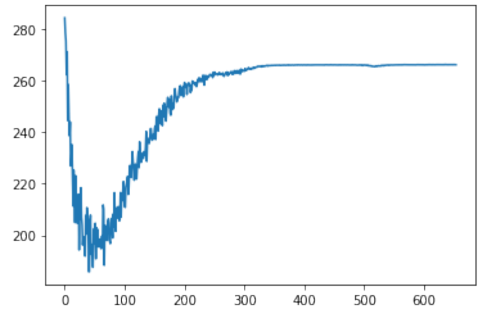
learning rate = 0.001 epochs = 20 batch size = 64
test loss = -0.2138 accuracy = 0.1067
The neural net did not converge



Training Loss Pytorch

Manual Implementation

My manual implementation did not converge as seen in the training loss graph. I tried clipping the data going into Sigmoid functions, it helped, but did not fix the convergence issue. I saw that my last layer weights were large, and perhaps the weights need to be regularized.



Training Loss Manual