IE 613: Assignment 2

Manan Doshi
140100015

01 April 2018

Question 1: FoReL and WM Equivalence

In this question, we show that the Weighted Majority algorith we developed for the case of a finite number of experts is a special case of the FoReL algortihm when:

$$\begin{aligned}
R_w &= \frac{1}{\eta}\sum_{j=1}^{d} w_j \log w_j && R_w \text{ is the regularizer} \\
f_t(w) &= \langle w, v_t \rangle && f_t(w) \text{ is the linear loss function} \\
\sum_{j=1}^{D} w_j &= 1 && w \text{ does not span the entire space } \mathbb{R}_d
\end{aligned}$$

We show that this is analogous to the WMA with D experts where v_t is the loss vector associated with round t and w_t is the (regularized) weight of each expert.
For the FoReL algorithm:

$$\begin{aligned}
w_t &= \arg\min_w \left(\sum_{i=1}^{t-1} f_i(w) + R(w) \right) \\
&= \arg\min_w \left(\sum_{i=1}^{t-1} \langle w, v_i \rangle + \frac{1}{\eta}\sum_{j=1}^{D} w_j \log w_j \right) && \text{Substituting regularizer and loss terms}
\end{aligned}$$

We need to find the optimal value of w under the constraint $\sum_j w = 1$. We use the lagrange multiplier method.

$$\begin{aligned}
l(w) &= \left(\sum_{i=1}^{t-1} \langle w, v_i \rangle + \frac{1}{\eta}\sum_{j=1}^{D} w_j \right) && \text{We wish to minimise this function} \\
c(w) &= \sum_{j=1}^{D} w_j - 1 && \text{under the constraint that } c = 0 \\
\nabla_w(l(w)) &= \sum_{i=1}^{t-1} v_i + \frac{1}{\eta}(\mathbf{1} + \log \mathbf{w}) \\
\nabla_w(c(w)) &= \mathbf{1} \\
\mathcal{L}(w) &= l(w) - \lambda c(w) \\
\nabla_{w,\lambda}\mathcal{L}(w, \lambda) &= 0 && \text{Lagrange multiplier method} \\
\sum_{i=1}^{t-1} v_i + \frac{1}{\eta}(\mathbf{1} + \log \mathbf{w}) &= \lambda(\mathbf{1})
\end{aligned}$$

Solving for w,

$$w_t = \exp(\eta\lambda - 1)\exp(-\eta\sum_{i=1}^{t-1} v_i)$$

$$w_{j,t} = \frac{e^{-\eta\sum_{i=1}^{t-1} v_{i,j}}}{\sum_{j=1}^{D} e^{-\eta\sum_{i=1}^{t-1} v_{i,j}}} \qquad \text{This is because we choose } \lambda \text{ such that the constraint is satisfied}$$

This is the exact same expression obtained in WMA, where $v_{i,j}$ in the loss suffered by Expert j in round i. The η here is the same as the η in WMA. The optimal η will thus be

$$\eta^* = \sqrt{\frac{2\log D}{T}}$$

Question 2: Online Convex Optimisation

Following are the plots generated by running the FTL and FoReL algorithms on the given systems. Averages are taken over 30 paths and the 95% confidence interval is shown using the shaded region. Code for this question can by found here

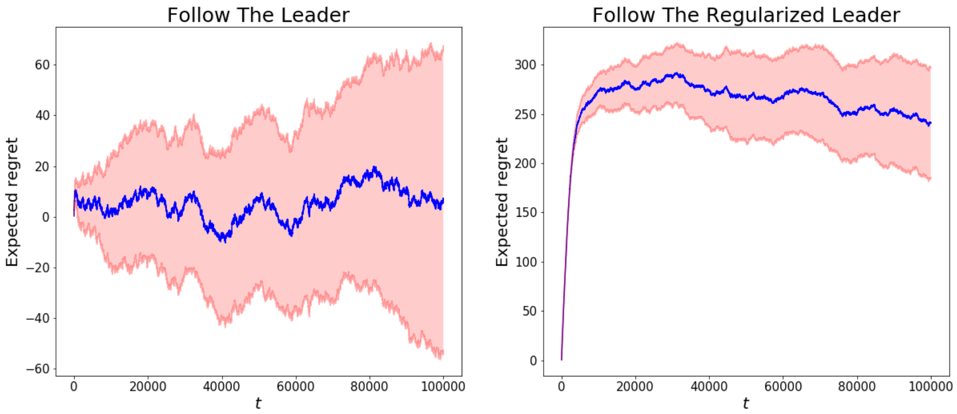

Figure 1: Expected regret as a function of times for FTL and FoReL algorithms working on the given system

It can be clearly seen that the confidence interval of FoReL is tighter. This is due to the stabilising effect of the regularisation term.

Question 3

Below is the variation of the final expected regret for the FoReL algorithm with varying regularization coefficient. The obtained plot is pretty much the same as the one obtained for WMA.

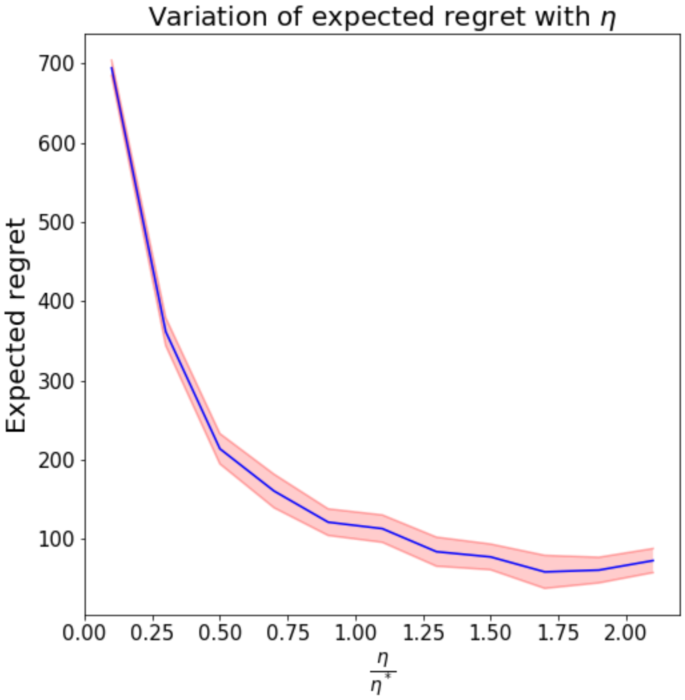

Figure 2: Variation of expected regret with η for the FoReL algorithm

Question 4: Online Classifier

A bias term was first to the feature set. The Perceptron algorithm was applied as-is. Since the Winnow algorithm requires the equation of the seperationg plane to have positive coefficients, the feature set was modified based on the weights obtained by the perceptron algorithm. The feature set was also shifted to make the seperating plane obtained by the perceptron algorithm to pass through the origin. The winnow algorithm was then applied to this modified dataset. Below are the results obtained.

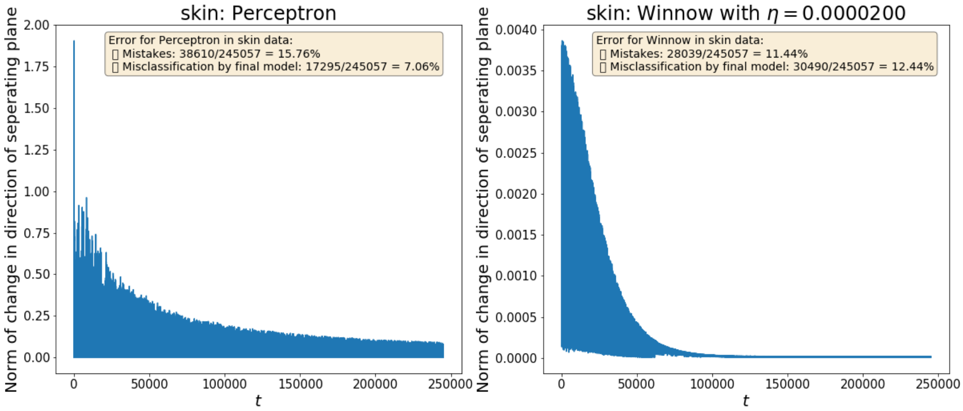

Figure 3: Performance on the two algorithms on the Skin Segmentation Dataset

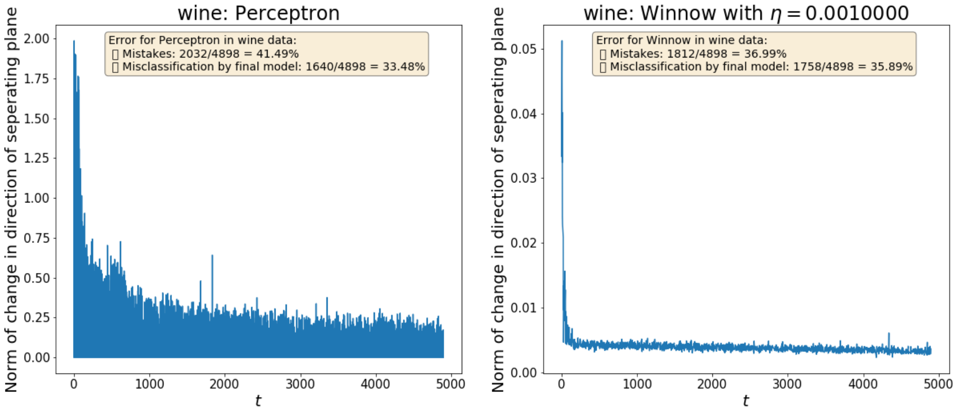

Figure 4: Performance on the two algorithms on the Wine Quality Dataset

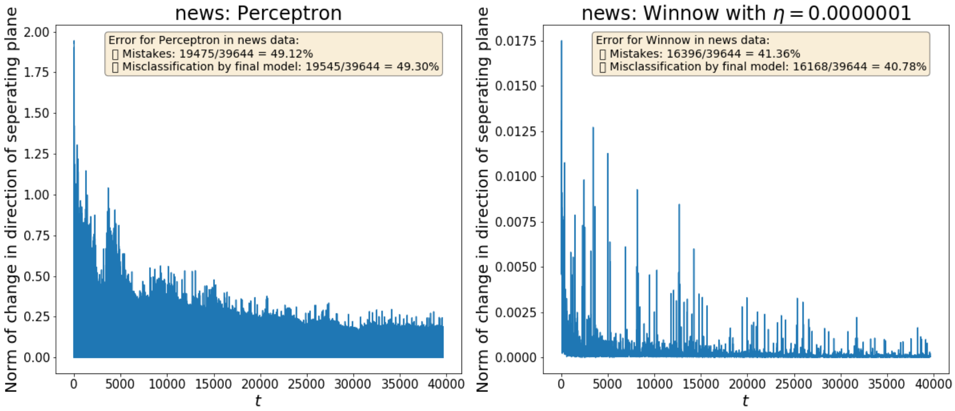

Figure 5: Performance on the two algorithms on the Online News Popularity Dataset

Question 5: Online to batch conversion

We first run our algorithm A on $\{x_i\}$ where x is sampled from $\mathcal{D}$.

$$\begin{aligned} \sum_{t=1}^{T} L_{\mathcal{D}}(h_t) &\leq M_A(H) && \text{Since } M_A(H) \text{ is the mistake bound} \\ \mathbb{E}_{\mathcal{D}}\left(\sum_{t=1}^{T} L_{\mathcal{D}}(h_t)\right) &\leq M_A(H) \end{aligned}$$

Since this relation is true for and sequence from the distribution, with the corresponding h_ts

$$\begin{aligned} T\mathbb{E}_{\mathcal{D},r}\left(L_{\mathcal{D}}(h_r)\right) &\leq M_A(H) && \text{We pick } h_r \text{ from a uniform distribution of } h_t\text{s} \\ \mathbb{E}_{\mathcal{D},r}\left(L_{\mathcal{D}}(h_r)\right) &\leq \frac{M_A(H)}{T} \end{aligned}$$

This means that if we predict the label for a new x chosen from $\mathcal{D}$ using a randomly chosen hypothesis from $\{h_t\}$ then the probability of making an error is less than $\frac{M_A(H)}{T}$

Question 6

Following are the seperating planes generated by the two algorithms for the same dataset

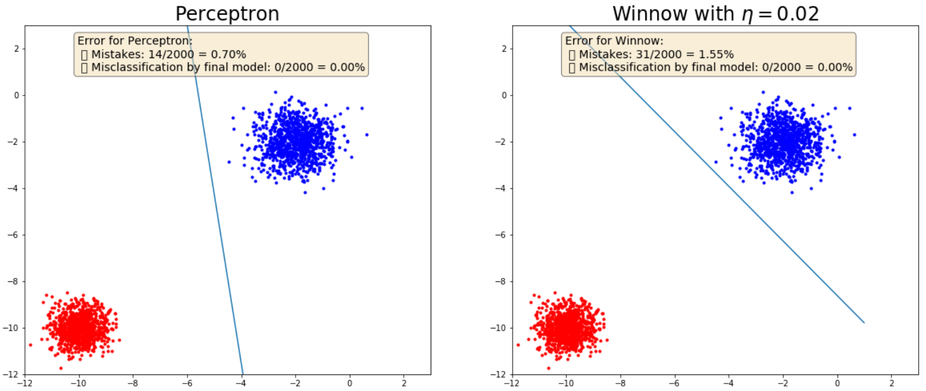

Figure 6: Performance on the two algorithms on the generated Dataset

An estimate of the margin is computed by creating multiple datasets and computing the margin for that dataset assuming the optimal seperating plane is parallel to $x + y + 12 = 0$. The estimated margin comes out to be 3.97. Similarly, the estimated value for the max norm comes out to be 16.11. Using this numbers, the Mistake bound for the perceptron algorithm is

$$
\begin{aligned}
M &\leq \left(\frac{R}{\gamma}\right)^2 \\
&= \left(\frac{16.11}{4.11}\right)^2 \\
M &\leq 16
\end{aligned}
$$

This is obeyed in our simulation.

Next we see the variation of the number of mistakes made ny the winnow algorithm as a function of the learning rate (with a 95% confidence interval. The system is regenerated and multiple runs are done). When the learning rate is very low, a high number of mistakes are made. This is expected since the classifier is very slow to respond to data. When the learning rate is very high, the confidence interval widens. The algorithm starts making more mistakes as it overshoots everytime there is a mistake.

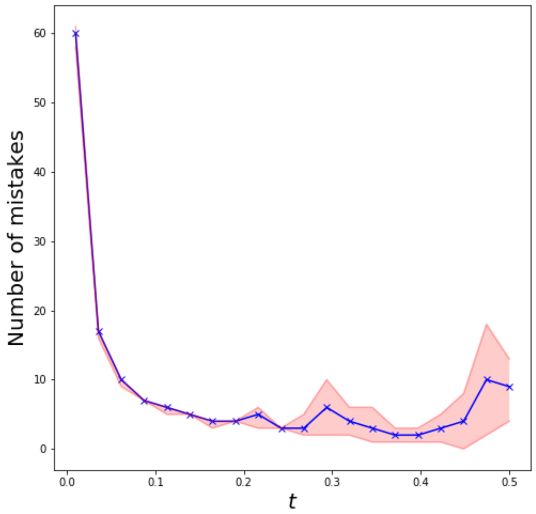

Figure 7: Performance of the winnow algorithm with various learning rates

Question 7

We compare the performance of OGD and OMD. OGD behaves as expected, making a lot of errors initially. The regret finally flattens out, implying that the algorithm is correct on most of the new data. The implementation of OMD is possibly incorrect. The 95% confidence intervals are plotted by doing multiple runs where a new dataset is generated and shuffled everytime.

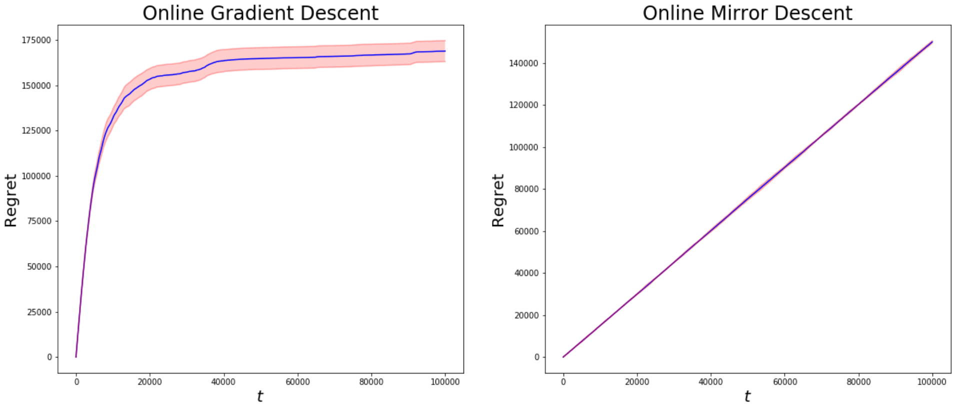

Figure 8: Performance of the OGD and OMD algorithms