Comparison of performance of Standard Perceptron and Second order Perceptron

Hardik Sahi
David R. Cheriton School of Computer Science
University of Waterloo
Waterloo, ON, N2L 3G1
h2sahi@uwaterloo.ca
CS 698: Intro to ML Fall 2017 Project Report

Abstract

In this report, I perform empirical evaluation of performance of Second order perceptron algorithm and compare it with Standard perceptron algorithm. Performance analysis is done based on mistake bound model for online learning. The shortcomings of standard perceptron and the way in which they are handled in second order perceptron are discussed. I try and replicate the results in paper [1] using synthetic data satisfying specific spectral or geometric properties.

1 Introduction

Online methods of machine learning are the ones in which the data is available to the learner sequentially and is used to update the best predictor for future data at each step. Such algorithms use a **mistake bound model** of learning which are discussed in [2]. This model is the one in which the learner is evaluated on the basis of number of mistakes it makes before converging to the correct hypothesis. For second order perceptron, the idea is that the algorithm under consideration must be able to sequentially classify the data points correctly ensuring that it does not make mistakes much greater than that made by the fixed/ referenced linear classifier on the same sequence S $\{(\mathbf{x}_1,y_1), (\mathbf{x}_2,y_2)..(\mathbf{x}_N,y_N)\}$ of input data. Advantage of the above approach of analysis is that the loss of learning algorithm, in terms of misclassified points with respect to reference classifier can be explained and nicely bound in terms of geometric properties of individual data sequences on which the algorithm is run.

As will be shown in the next section, the Perceptron learning algorithm **(PLA)** is a gradient descent algorithm whose mistake bound is determined by the trace information (first order information). We get the second order perceptron algorithm by enhancing Standard Perceptron to become sensitive to second order information by incrementally computing sparse data correlation matrix. In this way, the enhanced algorithm is able to exploit certain geometric properties of data which are missed by standard Perceptron.

A typical scenario under study in which the mistake bound of Second order perceptron is significantly smaller than that of standard Perceptron is when the data lies on a flat ellipsoid so that the eigen values of correlation matrix is significantly different along different axis whereas the trace will be determined by the largest eigen value.

In this report, I explain the properties of the enhanced perceptron and show how its behaviour depends on eigenvalues of data correlation matrix. Also, I will publish the results of application of

Submitted to Introduction to Machine Learning (CS698, 2017 Fall). Do not distribute.

Second order perceptron on an artificial dataset and compare it with standard perceptron.

2 Standard Perceptron

A set of input data points S $\{(\mathbf{x}_1,y_1)$, $(\mathbf{x}_2,y_2)..(\mathbf{x}_N,y_N)\}$, $\mathbf{x}_t \in \mathbb{R}^d$ is **linearly separable** if there is a hyperplane (which splits the input space into two half-spaces) such that all points of the first class are in one half-space and those of the second class are in the other half-space. Hence the instance vectors $\mathbf{x}_t$ are consistently labeled according to whether they lie on positive (y_t = +1) or the negative side (y_t=-1) of an unknown target hyperplane with normal vector $\mathbf{u} \in \mathbb{R}^d$.

The standard perceptron learning algorithm was discussed for the first time by **RosenBlatt** in [4] Given the set **S** of data points and starting from an initial weight vector, the PLA is ultimately able to yield a hyperplane (represented by weight vector which is normal to the hyperplane) that classifies all the examples correctly after a finite number of iterations for a linearly separable dataset.

The prediction made by PLA at time t is given by $\widehat{y}_t$ = SIGNUM $(\mathbf{w}^\top \mathbf{x}_t) \in \{-1,+1\}$. We say that the algorithm has made a mistake in making a prediction if $\widehat{y}_t \neq y_t$. Hence, to correct its mistake, the algorithm updates its internal state (maintained by weight vector **w**) following a simple additive rule: $\mathbf{w}_{t+1} = \mathbf{w}_t + y_t\mathbf{x}_t$. Geometrically speaking, the updated weight vector $\mathbf{w}_t$ moves along the direction of instance $\mathbf{x}_t$ for which the prediction was wrong. Hence the number of updates made in the weight vector is equal to the number of mistakes the algorithm makes in the process of learning.

As shown in [3] by **A. Novikoff**, the convergence of standard perceptron on linearly separable data set is guaranteed after a finite number of iterations. That is, it makes at most $(R/\gamma)^2$ mistakes on any number t of examples where,

$$R(Radius) = \max_{1 \leq s \leq t} \|\mathbf{x}_s\|$$

$$\gamma(Margin) = \min_{1 \leq s \leq t} | \mathbf{u}^\top \mathbf{x}_t |$$

A typical scenario in which standard perceptron would be extremely slow even for linearly separable data set would be when instance vector $\mathbf{x}_t$ would have a very small projection over target hyperplane (hence $\mathbf{x}_t$ would be almost orthogonal to **u**, making denominator extremely small) and has a large norm $\|\mathbf{x}_t\|$ (numerator becomes huge) [Figure 2].

3 Whitened Perceptron: Adding sensitivity to second order information.

The second order perceptron algorithm is defined to exploit the aforementioned spatial orientation of data points and hence uses second order information of data distribution as well. To be able to use it, I first introduce an intermediate algorithm called Whitened perceptron algorithm.

Whitened perceptron is a non incremental version of standard perceptron which means that all the instance vectors are known ahead of time, only the corresponding labels are unknown. Also, for simplicity it is assumed that span $\{\mathbf{x}_1 , \mathbf{x}_2 .. \mathbf{x}_N\} = \mathbb{R}^d$. This ensures that the data correlation matrix $C = \sum_{i=1}^{T} \mathbf{x}_i\mathbf{x}_i^T$ is full rank and hence its inverse exists (C^{-1} exists). Also, as C is positive semi definite matrix, $C^{-1/2}$ also exists. Given all the above, the whitened perceptron algorithm simply involves using standard perceptron algorithm on the whitened sequence represented by $\{(C^{-1/2}\mathbf{x}_1$, y1), $(C^{-1/2}\mathbf{x}_2$, y2) $..(C^{-1/2}\mathbf{x}_N, y_N)\}$.

Now, in order to show the advantage of whitening the data sequence and applying standard perceptron to it instead of simply applying the PLA directly to un-whitened data, I will show the

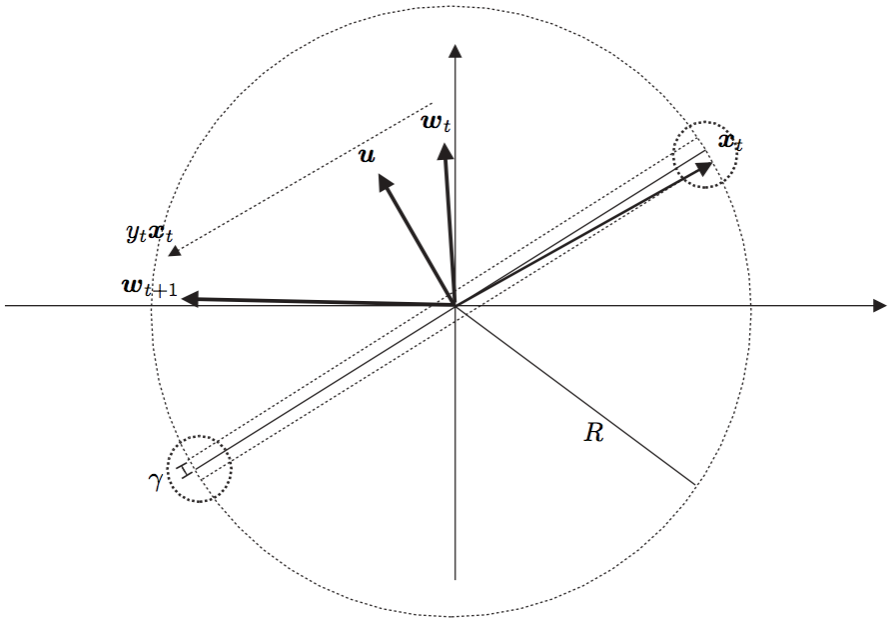


Figure 1: [1] A typical scenario in which perceptron convergence is very slow even for linearly separable dataset. **u** is the target normal vector, $\mathbf{w}_t$ is the weight vector that the algorithm maintains at the start of trial t. Because the instance vector $\mathbf{x}_t$ has a very small projection on **u** and length of $\mathbf{x}_t$ is R, following the simple additive rule of perceptron: $\mathbf{w}_{t+1} = \mathbf{w}_t + y_t\mathbf{x}_t$ updates the weight vector to $\mathbf{w}_{t+1}$ farther away from **u** than it originally was. Hence, this slows down the convergence of perceptron algroithm

mistake bound on such a whitened sequence under the assumption that the data sequence originally is linearly separable, and hence will converge after a maximum of $(R/\gamma)^2$ mistakes on any number t of examples.

From **[1]**, it is realised that under the above conditions, whitened sequence is also linearly separable with the hyperplane $\mathbf{z}=C^{1/2}\mathbf{u}$. Also,

$$\gamma_{\text{white}} = \frac{\gamma}{\|C^{1/2}\mathbf{u}\|}.$$

Hence, maximum number of mistakes that the perceptron makes on whitened data sequence before convergence is given by,

$$(\frac{R_{\text{white}}}{\gamma_{\text{white}}})^2 = (\frac{1}{\gamma^2})(\max_t \|C^{-1/2}\mathbf{x}_t\|^2)\|C^{1/2}\mathbf{u}\|^2$$

Simplifying the above, we get,

$$(\frac{R_{\text{white}}}{\gamma_{\text{white}}})^2 = (\frac{1}{\gamma^2})(\max_t \mathbf{x}_t{}^\top C^{-1}\mathbf{x}_t)(\mathbf{u}^T C\mathbf{u})$$

Clearly, the term $\mathbf{x}_t{}^T C^{-1}\mathbf{x}_t$ can be associated with the extent of correlation in the input sequence S. As can be seen from the equation above, the maximum number of mistakes made will be reduced if the numerator is minimized. Consider the case in which the input sequence is highly correlated, which implies that the the term $\mathbf{x}_t{}^T C^{-1}\mathbf{x}_t$ is very small. Also, assume that the input instances have

very small projection over the target normal vector $\mathbf{u}$. In such a scenario the term $\mathbf{u}^{\mathrm{T}}\mathrm{C}\mathbf{u}$ is very small. Hence, the numerator term of the above equation is minimized following the aforementioned conditions. So, it can be concluded that the mistake bound for standard perceptron application on whitened sequence is significantly smaller than that of standard perceptron application on non-whitened sequence.

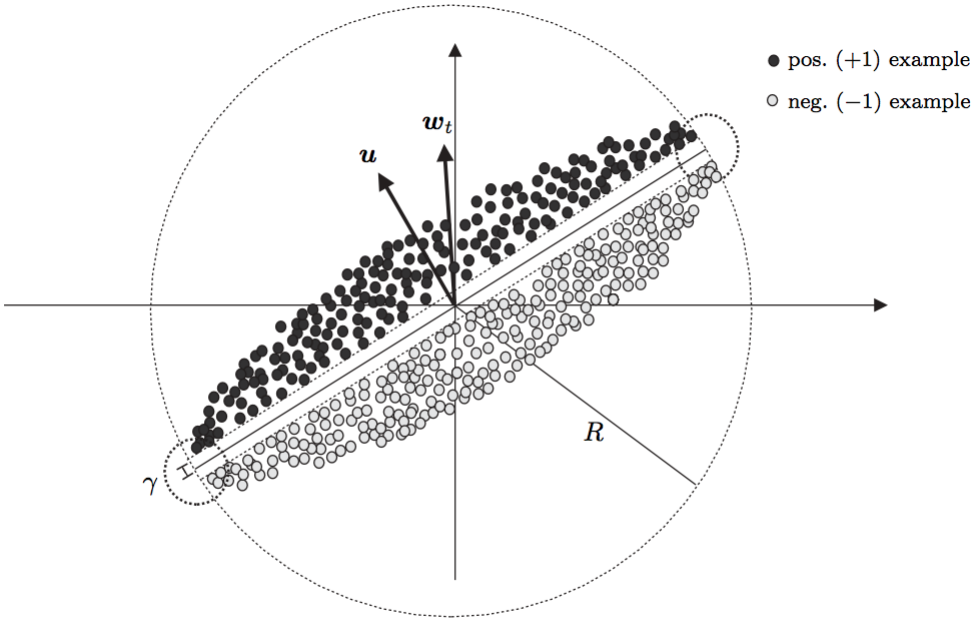


Figure 2: [1] The typical scenario which whitened perceptron can take advantage of but the standard perceptron algorithm struggles to converge.

For verifying the point that the whitened perceptron algorithm has significantly smaller mistake bound as compared to that of standard perceptron algorithm for the input sequence with starkly different eigen values along different axis such that the data is maximally squashed along the direction of $\mathbf{u}$ (degenerate scenario), we perform the following analysis:

Because of symmetric distribution of data, we have the following eigenstructure of matrix C: λ_{min} and λ_{max} are the minimal and maximal eigenvalues of C respectively. As per our aforementioned degenerate scenario, λ_{min} is aligned along the target normal vector $\mathbf{u}$ and λ_{max} is aligned orthogonal to $\mathbf{u}$ (represented by $\mathbf{u}^{\perp}$).

$$
\begin{aligned}
\lambda_{\text{min}} &= \mathbf{u}^{\top} C \mathbf{u} \\
&= \mathbf{u}^{\top} (\sum_{i=1}^{T} \mathbf{x}_{\text{i}} \mathbf{x}_{\text{i}}^{\mathrm{T}}) \mathbf{u} \\
&= \sum_{i=1}^{T} (\mathbf{u}^{\top} \mathbf{x}_{\text{i}})^2 \\
&= \gamma^2 T
\end{aligned}
$$

Since $\lambda_{\text{min}} + \lambda_{\text{max}} = \sum_{i=1}^{T} \|\mathbf{x}_{\text{i}}\|^2 \implies (\mathbf{u}^{\mathrm{T}}\mathbf{x})^2 + ((\mathbf{u}^{\perp})^{\mathrm{T}}\mathbf{x})^2 = \|\mathbf{x}\|^2$

$$\mathbf{x_t}^\top C^{-1}\mathbf{x_t} = \frac{(\mathbf{u}^{\mathrm{T}}\mathbf{x_t})^2}{\lambda_{\min}} + \frac{((\mathbf{u}^{\perp})^{\mathrm{T}}\mathbf{x_t})^2}{\lambda_{\max}}$$
$$= \frac{\gamma^2}{\gamma^2 T} + \frac{\|\mathbf{x_t}\|^2 - \gamma^2}{\sum_{i=1}^{T}\|\mathbf{x_i}\|^2 - \gamma^2 T}$$

Hence,

$$(\frac{R_{\text{white}}}{\gamma_{\text{white}}})^2 = (\frac{1}{\gamma^2})(\max_t \mathbf{x_t}^\top C^{-1}\mathbf{x_t})(\mathbf{u}^T C\mathbf{u})$$
$$= 1 + \frac{R^2 T - \gamma^2 T}{\sum_{i=1}^{T}\|\mathbf{x_i}\|^2 - \gamma^2 T}$$

It can be clearly seen that the above bound for maximum number of mistakes approaches 2 (which is a constant) as norm of instance vectors $\mathbf{x_t}$ approaches R, which is independent of both margin γ and radius R of the ball containing data. Hence it proves that the mistake bound is extremely small in the degenerate scenario as shown above.

4 Second order perceptron.

Second order perceptron algorithm is viewed as an incremental version of Whitened perceptron discussed above which involves iterative computation of sparse correlation matrix based only on a small subset of data points observed so for. The pseudocode for Second order perceptron is described below.

Initialize: $X_0=\varnothing$, $\mathbf{v}_0 = \mathbf{0}$, k=1.
Repeat t = 1,2...

1. Access instance vector $\mathbf{x_t}$;
2. Set $S_t=[X_{k-1}\ \mathbf{x_t}]$;
3. Predict $\widehat{y}_t$ = SIGNUM $(\mathbf{w}^\top\mathbf{x_t})\in\{-1,+1\}$, where $(S_t S_t^{\mathrm{T}})^+\ \mathbf{v}_{k-1}$;
4. Get actual label corresponding to input, y_t;
5. Update the internal state of the algorithm if $y_t \neq \widehat{y}_t$ as follows:

$$\mathbf{v_k}=\mathbf{v}_{k-1} + y_t\mathbf{x_t}$$
$$X_k=S_t$$
$$k=k+1$$

Clearly we can see that the number of columns in X_k matrix is equal to the number of mistakes that the algorithm makes on the dataset. Hence the second order perceptron is also mistake bound like the standard perceptron algorithm.

Clearly the algorithm iterates over the data points over and over again. It converges in case the system of data points is linearly separable but stays in loop if the set is not linearly separable, just like the case with standard perceptron algorithm.But the number of mistakes it makes before convergence is smaller than that made by standard perceptron algorithm.

5 Pros and Cons of Second order perceptron algorithm.

As will be clear from the simulations on artificial data set in the following section, the second order perceptron algorithm exploits the spectral properties of data distribution in the sense that it takes advantage of the maximally squashed data distribution, something that the standard perceptron fails to and hence converges with fewer mistakes as compared to standard perceptron.
On the downside, the implentation of second order perceptron involves taking pseudo-inverse of the data correlation matrix iteratively which makes the algorithm slower as compared to standard perceptron algorithm.

6 Running the algorithms on artificial datasets.

I have tried to replicate the results provide in paper [1]. In order to do so, I tried creating an artificial dataset that is linearly separable and also satisfies the spectral properties that enhanced perceptron can take advantage of.

In order to create an artificial dataset I sampled data from a multivariate Gaussian distribution making sure that the correlation matrix has single dominant eigen vector perpendicular to the direction of the maximally squashed data distribution. As will be evident by the result of the simulation, the number of mistakes made by second order perceptron would be smaller than those made by standard perceptron algorithm.

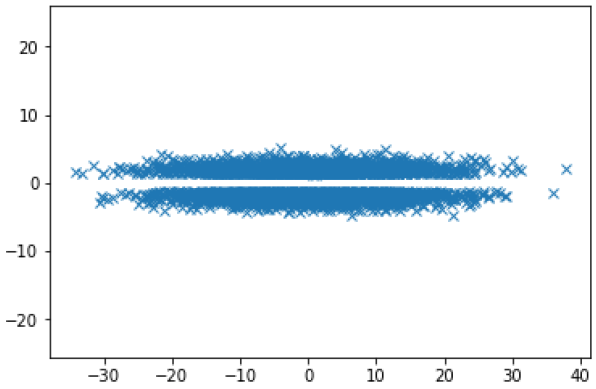
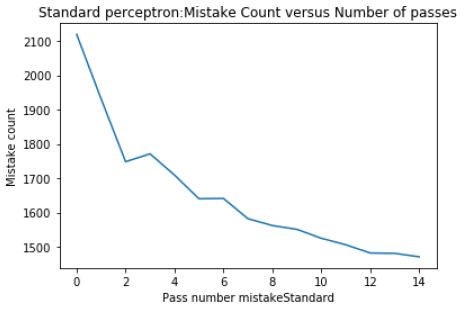
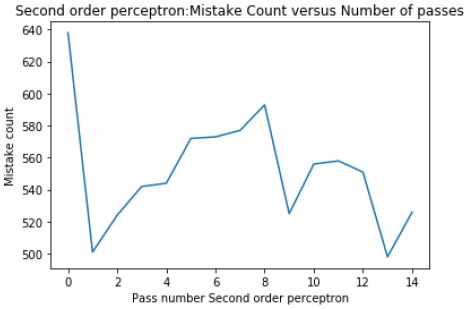


Figure 3: Distribution of the artificial dataset corresponding to the 2 major attributes represented along X and Y axis. The data is sampled from multivariate Gaussian distribution with significantly different eigen values along the 2 axis.



(a) Running standard Perceptron algorithm



(b) Running second order perceptron algorithm on data set

Figure 4: Comparison of mistake count of the standard and second order perceptron. Note the significantly lower mistake count made by second order perceptron.

Algorithm	**Mistake percent**
Standard Perceptron	35.81%
Second order perceptron	11.99%

Table 1: Mistake count comparison for the two algorithms.

7 Conclusion.

As evident by the results of the simulation above, it is clear that second order perceptron algorithm has significantly lower bound in case of specific geometric distribution of data. Having said that, the second order perceptron is slower as compared to standard one as it involves taking the pseudo-inverse of the incrementally computed data correlation matrix.

References

[1] Nicolò Cesa-Bianchi, Alex Conconi, and Claudio Gentile. A second-order perceptron algorithm. In *Proceedings of the 15th Annual Conference on Computational Learning Theory*, COLT '02, pages 121–137, London, UK, UK, 2002. Springer-Verlag.

[2] Nick Littlestone. Learning quickly when irrelevant attributes abound: A new linear-threshold algorithm. *Mach. Learn.*, 2(4):285–318, April 1988.

[3] A. B. Novikoff. On convergence proofs on perceptrons. In *Proceedings of the Symposium on the Mathematical Theory of Automata*, volume 12, pages 615–622, New York, NY, USA, 1962. Polytechnic Institute of Brooklyn.

[4] F. Rosenblatt. The perceptron: A probabilistic model for information storage and organization in the brain. *Psychological Review*, 65(6):386–408, 1958.