Scalable linear classifiers based on exponential loss function

Sandhya Tripathi
Indian Institute of Technology Bombay
Mumbai, India 400076
sandhya.tripathi@iitb.ac.in

N. Hemachandra
Indian Institute of Technology Bombay
Mumbai, India 400076
nh@iitb.ac.in

ABSTRACT

We first propose an empirical risk minimization based binary classification algorithm, ExpERM, with exponential function as surrogate loss function. Our ExpERM algorithm is scalable in both the dimension of feature space as well as in the number of data points as it is an unconstrained differentiable convex optimization problem or a convex optimization problem with a single constraint in the regularized ExpERM framework. Under mild assumption on data, we show that ExpERM classifier is unique. We implement ExpERM on wide collection (large-features, large-examples) of real datasets. We use Wilcoxon signed-rank test to show that these easily computable classifiers have good generalization property as their 5 or 10-fold cross validation error is not significantly different from those classifiers obtained by standard hinge loss based SVMs or AdaBoost classifiers. Using some large feature sized datasets, we make a statistical comparison of SVM and ExpERM and observe that ExpERM takes remarkably less time to train the model. Towards better understanding this simple and effective learning algorithm, we obtain PAC like sample complexity bounds for regularized ExpERM and high probability bounds for ExpERM based on Rademacher complexity and uniform stability notions. Our computational experience suggests that the regularization does not significantly improve the performance of the ExpERM based classifier. Due to repeated calls to be made to the binary classifier routines in implementation of combined multi-class algorithms, ExpERM scheme is expected to be significantly computationally cheaper and hence scalable. We statistically show that CV error of our binary ExpERM classifiers on many multi-class datasets is comparable to those obtained using computationally expensive binary class SVMs.

KEYWORDS

Exponential loss function; High probability bounds; Multi-class classification; Algorithmic stability; Sample complexity; ERM; Linear classifiers

ACM Reference format:
Sandhya Tripathi and N. Hemachandra. 2018. Scalable linear classifiers based on exponential loss function. In *Proceedings of The ACM India Joint International Conference on Data Science & Management of Data, Goa, India, January 11–13, 2018 (CoDS-COMAD '18),* 11 pages.
DOI: 10.1145/3152494.3152521

CoDS-COMAD '18, Goa, India

978-1-4503-6341-9/18/01…$15.00
DOI: 10.1145/3152494.3152521

1 INTRODUCTION

One important building block of learning algorithms is the loss function used to come up with a learner by algorithmically processing the given training data. So, loss functions also have a direct bearing on the generalization property of the learner that is being designed. Also of importance is the scalability of the learning algorithm both in terms of the data size and runtime requirements. Depending upon the underlying learning problem, these loss functions are required to possess different properties. We are interested in designing learners for classification problems, both binary class and multi-class.

The domain set or feature space and the label set is denoted by $\mathbf{X} \subseteq \mathbb{R}^n$ and $Y \in \{-1, 1\}$ respectively. Let $\mathbf{D}$ be a joint probability distribution over $\mathbf{Z} = \mathbf{X} \times Y$. The learner has access to the training data $S = (\mathbf{z}_1, \ldots, \mathbf{z}_m)$ where $\mathbf{z}_i \in \mathbf{Z}$. The learner is requested to output a prediction rule $f : \mathbf{X} \mapsto Y$ such that $f(\mathbf{x}) = \text{sign}(g(\mathbf{x}))$ where $g : \mathbf{X} \mapsto \mathbb{R}$ is a measurable function on $\mathbf{X}$. As is mostly common we restrict ourselves to the set of all linear classifiers $h(\mathbf{x})$ by considering $g(\cdot)$ as linear functionals. That is, our hypothesis class is of the form $\mathcal{H} = \{(\mathbf{w}, b) : \mathbf{w} \in \mathbb{R}^n, b \in \mathbb{R}\}$ so that $h(\mathbf{x}) = \mathbf{w}^{\mathbf{T}}\mathbf{x} + b$. For the sake of convenience, we incorporate the bias b into $\mathbf{w}$ as an extra coordinate at the end of the vector. The error of a prediction rule $f(\mathbf{x}) = \text{sign}(h(\mathbf{x}))$ is the probability that it does not predict the correct label on an instance $\mathbf{z} = (\mathbf{x}, y)$ generated by $\mathbf{D}$.

Multi-class classification is the problem of classifying instances into one of several possible classes; [14] and [18] provide a good introduction to this topic. We briefly describe the problem and popular approaches used to solve it. Our goal here is to learn a prediction rule $f : \mathbf{X} \mapsto Y$ where Y is a given finite set of classes. Without loss of generality, let us denote $Y = \{1, 2, \ldots, c\}$. There exist two broad classes of algorithms: *uncombined algorithms* which are specifically designed for multi-class problems and *combined algorithms* where the multi-class problem is solved by reduction to binary classification problem. In the later class of algorithms, two popular methods are One-versus-All (OVA) and One-versus-One (OVO) or Pairwise method.

In OVA, we learn c binary classifiers, $f_l = \text{sign}(h_l)$ derived from linear functions, $h_l : \mathbf{X} \mapsto \mathbb{R}$, for $l \in Y$. Given S, we construct c binary training sets $S_i = ((\mathbf{x}_1, (-1)^{\mathbf{1}[y_1 \neq i]}), \ldots, (\mathbf{x}_m, (-1)^{\mathbf{1}[y_m \neq i]}))$ such that each instance label is 1 if it belongs to class i, else -1. Then, the multi-class prediction rule $f : \mathbf{X} \mapsto Y$ defined by OVA method is given by:

$$\forall \mathbf{x} \in \mathbf{X},\ f(\mathbf{x}) = \underset{l \in Y}{\text{argmax}}\ h_l(\mathbf{x}). \tag{1}$$

In OVO, for each pair of distinct classes $(l, l') \in Y^2,\ l \neq l'$, a binary classifier $f_{ll'} : \mathbf{X} \mapsto Y$ is obtained by training a binary classification algorithm on sub-sample containing exactly the points

labeled with l or $l^{'}$, with 1 returned for class $l^{'}$ and -1 for class l. The final multi-class prediction rule f is defined by training $\binom{c}{2} = \frac{c(c-1)}{2}$ classifiers, combined via majority vote:

$$\forall \mathbf{x} \in \mathbf{X},\ f(\mathbf{x}) = \underset{l \in Y}{\operatorname{argmax}} |\{l : f_{ll'}(\mathbf{x}) = 1\}|. \tag{2}$$

Since a multi-class problem is reduced to a binary classification problem, we give a detailed analysis of binary classification problems. Given any set $\mathcal{H}$ and $\mathbf{Z}$, a loss function is defined as $l : \mathcal{H} \times \mathbf{Z} \mapsto \mathbb{R}_+$. The risk function, defined to be the expected loss of a classifier, $h \in \mathcal{H}$, with respect to the probability distribution $\mathbf{D}$ over $\mathbf{Z}$ is,

$$L_{\mathbf{D}}(h) := \underset{\mathbf{z} \sim \mathbf{D}}{E}[l(h, \mathbf{z})] \tag{3}$$

Let $\eta(\mathbf{x}) := P[y = 1|\mathbf{x}]$ be conditional in-class probability. Then, we can write

$$L_{\mathbf{D}}(h) = E_{\mathbf{X}}[\eta(\mathbf{x}) l(h(\mathbf{x}), (\mathbf{x}, 1)) + (1 - \eta(\mathbf{x})) l(h(\mathbf{x}), (\mathbf{x}, -1))]. \tag{4}$$

If the loss function is $0-1$ loss, defined as,

$$l_{0-1}(h, (\mathbf{x}, y)) := \begin{cases} 0 \text{ if } \operatorname{sign}(h(\mathbf{x})) = y \\ 1 \text{ if } \operatorname{sign}(h(\mathbf{x})) \neq y \end{cases} \tag{5}$$

then the risk function in (3) becomes,

$$\begin{aligned} L_{\mathbf{D}, 0-1}(h) &= \underset{\mathbf{z} \sim \mathbf{D}}{E}[l_{0-1}(h, \mathbf{z})] = \underset{\mathbf{z} \sim \mathbf{D}}{E}[\mathbf{1}_{\{\operatorname{sign}(h(\mathbf{x})) \neq y\}}] \\ &= \underset{\mathbf{z} \sim \mathbf{D}}{P}[\operatorname{sign}(h(\mathbf{x})) \neq y] \end{aligned} \tag{6}$$

and is called true risk or generalization error of h. The minimum value of true risk over all prediction rules is called Bayes risk. The population minimizer of $0-1$ loss function or minimizer of Bayes Risk, $\operatorname{sign}(2\eta(\mathbf{x}) - 1)$ is called *Bayes decision rule.*

The empirical risk, defined to be the expected loss over a given sample $S = (\mathbf{z}_i \ldots, \mathbf{z}_m) \in \mathbf{Z}^m$ is,

$$L_S(h) := \frac{1}{m} \sum_{i=1}^{m} l(h, \mathbf{z}_i) \tag{7}$$

Since, the only representation of the domain set we have is in the form of training sample S, the best classifier h can be found by minimizing $L_S(h)$, i.e.,

$$h_S^* \in \arg\min_{h \in \mathcal{H}} L_S(h).^1 \tag{8}$$

This learning paradigm is called Empirical Risk Minimization (ERM). However, solving the ERM problem with respect to the $0-1$ loss is known to be NP-hard ([18], Chapter 8). One popular approach is to upper bound the non-convex $0-1$ loss function by a convex surrogate function, say l_{sur}. In classification, most common surrogate loss functions are margin-based, i.e., there exists a function ϕ such that $l_{sur}(h(\mathbf{x}), (\mathbf{x}, y)) = \phi(yh(\mathbf{x}))$, $y \in Y$ and $h(\mathbf{x}) \in \mathbb{R}$. For example, hinge loss defined as $l_{hinge}(h, (\mathbf{x}, y)) = \max\{0, 1 - yh(\mathbf{x})\}$ is one such surrogate loss function used in SVM. Another surrogate for $0-1$ loss function is exponential loss defined as $l_{exp}(h, (\mathbf{x}, y)) = e^{-yh(\mathbf{x})}$. This loss function is relevant in AdaBoost, a popular boosting method; boosting algorithms are an important part of ensemble schemes whose paradigm is quite different from ERM paradigm.

[1] $l(h, \mathbf{z}_i)$ need not be strictly convex and hence, minimizer h_S^* need not be unique.

1.1 Relevant work and our contribution

There has been a lot of work on the role of loss functions, their properties and interpretation from various perspectives. It is known that a large family of margin-based loss functions are Fisher consistent, i.e., the population minimizer of the loss function leads to the Bayes optimal rule [12]. They also use connection between Fisher consistency and consistency in classification to derive consistency and rate of convergence results of classifiers based on margin-based loss functions. The convergence rate of the risk of a function that minimizes empirical risk over some fixed class $\mathcal{H}$ was studied in [22]. Under the assumption of low noise, it was shown that the risk converges to the minimum risk over class $\mathcal{H}$. Suppose the generic conditional $\phi-$risk is defined as $C_\eta(\alpha) = \eta\phi(\alpha) + (1 - \eta)\phi(-\alpha)$, then ϕ is classification calibrated if, for any $\eta \neq 1/2$, $\inf_{\alpha : \alpha(2\eta - 1) \leq 0} C_\eta(\alpha) > \inf_{\alpha \in \mathbb{R}} C_\eta(\alpha)$. Using a weaker condition, [3] shows that if a margin-based loss function is classification calibrated, then the convergence of excess risk due to l_{sur} implies convergence of excess risk due to l_{0-1}, i.e., for every sequence of measurable functions $\{h_i\}_{i \geq 1}$,

$$\text{if } L_{\mathbf{D}, l_{sur}}(h_i) \longrightarrow \min_{h \in \mathcal{H}} L_{\mathbf{D}, l_{sur}}(h)$$

$$\text{then, } L_{\mathbf{D}, l_{0-1}}(h_i) \longrightarrow \min_{h \in \mathcal{H}} L_{\mathbf{D}, l_{0-1}}(h)$$

A general theory on various loss functions is presented in [20]. Another approach to these classification problems is via function estimation where conditional in-class probabilities are estimated. A new loss function using the probability elicitation idea is developed in [13]. By selecting a risk and hypothesis class, they construct a loss l_{sur} and use the convexity of minimum conditional risk as a sufficient condition to obtain an optimal classifier. An extension of the work by [5] applied to composite losses in [13] is presented in [17]. They also generalize the notion of classification calibrated losses from [3]. An axiomatic framework for classifier design via coherent loss function is proposed in [23]. They derive a tractable upper bound for empirical misclassification error and claim that coherent loss approach has robustness properties. A family of coherence functions, as majorization to hinge loss is proposed in [25]. They show that using these coherence functions one can estimate the class probabilities and devise new large margin classifiers.

Various researchers have addressed the problem of comparing two or more classification algorithms by using standard parametric and non-parametric statistical tests. The appropriate test can be chosen depending upon the comparison, whether two or multiple algorithms on single dataset or multiple datasets. A good overview of this can be seen in [2], [10], [8]. We use these tests later.

We present an exponential loss function, $l_{exp}(h, (\mathbf{x}, y))$, based ERM framework in Section 2. We explore its properties like Lipschitzness and smoothness which help us in developing high probability bounds for the unknown risk. These properties are in addition to the already known properties like classification calibration, etc., of exponential loss function. We also reinterpret the exponential loss function as loss-reward function, which rewards *all* correctly classified examples and penalizes misclassified examples; maximization of this in ERM framework is equivalent to the above. We prove that under mild assumptions, the ExpERM classifier is unique.

In Section 3, we give some high probability bounds on generalization error by building upon the existing results. We obtain bounds based on algorithmic stability as ExpERM is shown to be uniformly stable. We also present some sample complexity bounds using the properties of exponential loss function shown in Section 2.

In Section 4, we give statistical evidence to validate the performance of ExpERM scheme. We perform Wilcoxon signed-rank test to make statistical comparisons of CV error and runtime between two algorithms. We show on a wide variety of real datasets that there is no significant difference between the CV error of ExpERM, SVM and Boosting based binary classifiers. It was seen that generalization error of the RegExpERM based binary classifiers is not significantly different from that of ExpERM based classifiers. In particular, we illustrate the scalability of ExpERM by comparing its runtime with SVM for some large feature sized datasets and observe that ExpERM is computationally efficient. For multi-class datasets with large number of classes, combined OVO/OVA based techniques make repeated calls to the binary classifier routines; for such instances ExpERM scheme is expected to be significantly computationally cheaper and hence scalable. To illustrate this, we implement ExpERM on multi-class datasets and observe that there is no statistically significant difference between the CV error of ExpERM based classifier and SVM classifier.

Section 5 concludes along with some promising directions of further work. Proofs of all results are presented in Appendix A.

2 EXPONENTIAL LOSS FUNCTION BASED EMPIRICAL RISK MINIMIZATION CLASSIFIER, ExpERM

In this section, we first consider the exponential loss function and a modified version of exponential loss. After showing some of its properties, we then define exponential loss function based ERM scheme.

2.1 Exponential loss function and its loss-reward interpretation

Exponential loss function is defined as $l_{exp}(h(\mathbf{x}),(\mathbf{x},y)) = e^{-yh(\mathbf{x})}$. Figure 1 shows the plot of $l_{exp}(h(\mathbf{x}),(\mathbf{x},y))$ along with $l_{hinge}(h(\mathbf{x}),(\mathbf{x},y))$ and $l_{0-1}(h(\mathbf{x}),(\mathbf{x},y))$. Clearly, l_{exp} is an upper bound on l_{0-1}. Also, $l_{exp}(yh(\mathbf{x})) : \mathbb{R} \mapsto \mathbb{R}$ is a positive and strict convex function. So, it is a valid candidate for surrogate function. For a data point $\mathbf{x}$, if $yh(\mathbf{x}) \leq 0$ (misclassification), then $l_{exp} \geq 1$ i.e., a large exponential loss is assigned. If $yh(\mathbf{x}) > 0$ (correct classification), then $l_{exp} \in (0,1]$. Therefore, unlike hinge loss $l_{hinge}(\cdot,\cdot)$ which penalizes *some* correctly classified examples along with the misclassified examples with a positive loss, exponential loss function, l_{exp} assigns a strictly positive loss to *all* correctly classified points. Also, for misclassified points, hinge loss penalization is only linear whereas exponential loss penalizes exponentially.

One property that any loss function is expected to have is that it should not give unbounded positive value to correctly classified examples. This is because inherently on a dataset any reasonable classifier will correctly classify a majority of the sample points. If the loss function gives arbitrarily large values to correct classifications, then the classifier obtained by optimizing over cumulative loss might not be finite. This motivates us to give a different perspective to the way a classifier is evaluated.

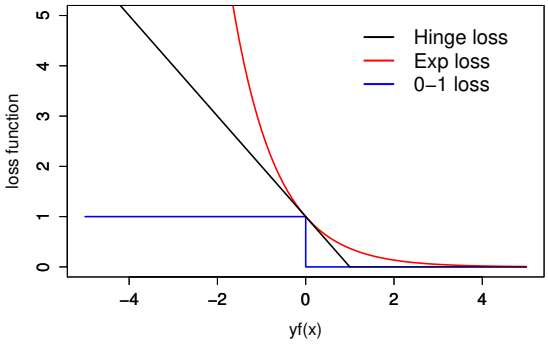

Figure 1: A plot showing $0-1$ loss function and its two convex surrogates; hinge loss and exponential loss. These functions are plotted as a function of the margin $yf(\mathbf{x})$.

We reinterpret the exponential loss function such that it rewards *all* the correctly classified examples and penalizes the misclassified examples. In particular, let this loss-reward function, $lr_{exp}(h(\mathbf{x}),\mathbf{z})$ be defined as follows:

$$\forall h \in \mathcal{H},\ \forall \mathbf{z} = (\mathbf{x},y),\ lr_{exp}(h(\mathbf{x}),\mathbf{z}) = 1 - e^{-yh(\mathbf{x})}. \tag{9}$$

$lr_{exp}(h(\mathbf{x}),\mathbf{z})$ is a smooth and concave function w.r.t. hypothesis $h = \mathbf{w} \in \mathbb{R}^n$. For correctly classified examples, $lr_{exp}(h(\mathbf{x}),(\mathbf{x},y)) \in [0,1)$ and for misclassified examples, $lr_{exp}(h(\mathbf{x}),(\mathbf{x},y)) \in (-\infty,0)$. This surrogate function incrementally rewards a sample point if it is correctly classified and exponentially penalizes it if it is misclassified, both based on the margin $yh(\mathbf{x})$. Figure 2 shows the plot of $lr_{exp}(h(\mathbf{x}),(\mathbf{x},y))$. This loss-reward interpretation holds for Boosting as it is also an exponential loss based classification algorithm. In Section 2.3, we use the fact that the performance of classifiers designed using l_{exp} and lr_{exp} is equivalent.

2.2 Some properties of exponential loss function

In this section, we study some properties of the exponential loss function $l_{exp}(h(\mathbf{x}),(\mathbf{x},y)) = e^{-yh(\mathbf{x})}$ as a surrogate loss function. We assume through out this work that the domain set or the feature space is bounded, i.e., $\|\mathbf{x}\|_2 \leq R$ for a given $R > 0$. For practical applications, some knowledge about the value of R can be obtained from the domain experts. Following properties of $l_{exp}(\mathbf{w},(\mathbf{x},y)) = e^{-y(\mathbf{w}^T\mathbf{x}+b)}$ are used in deriving the performance guarantees for the classifier obtained from ExpERM. For the sake of convenience, let $\underline{\mathbf{x}}_i = (\mathbf{x}_i,1) \in \mathbb{R}^{n+1}$ where $\mathbf{x}_i \in \mathbb{R}^n$ such that $\|\underline{\mathbf{x}}\|_2 \leq \tilde{R}$. We also assume that hypothesis class is bounded and the norm includes bias term, i.e., $\|(\mathbf{w},b)\|_2 \leq B$ for a given $B > 0$.

The proofs of following results are given in the appendix.

Lemma 2.1. *For a given sample $\mathbf{z}_i = (\mathbf{x}_i,y_i)$, $l_{exp}((\mathbf{w},b),\mathbf{z}_i)$ is a convex function in $\mathbf{w}$ and b.*

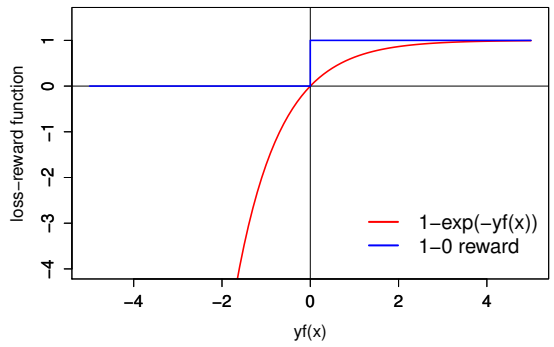

Figure 2: A plot showing $1-0$ reward function and its concave surrogate $lr_{exp}(h(\mathbf{x}),\mathbf{z}) = 1-e^{-yh(\mathbf{x})}$. *All* examples correctly classified by $lr_{exp}(\cdot,\cdot)$ are assigned positive reward within $[0,1)$ and misclassified examples are assigned negative reward within $(-\infty,0)$.

Lemma 2.2. *Given $\|\mathbf{w}\|_2 \leq B$, $\|\underline{\mathbf{x}}\|_2 \leq \tilde{R}$ and a sample $\mathbf{z}_i = (\mathbf{x}_i, y_i)$, $l_{exp}((\mathbf{w},b),\mathbf{z}_i)$ is a $\rho-$Lipschitz function with $\rho = \tilde{R}e^{B\tilde{R}}$.*

Lemma 2.3. *Given $\|\mathbf{w}\|_2 \leq B$, $\|\underline{\mathbf{x}}\|_2 \leq \tilde{R}$ and a sample $\mathbf{z}_i = (\mathbf{x}_i, y_i)$, $l_{exp}((\mathbf{w},b),\mathbf{z}_i)$ is a $\gamma-$smooth function $\gamma = \tilde{R}e^{B\tilde{R}}$.*

One more property of the loss function, later used in evaluating the algorithm based on it, is stability. In simple words, an algorithm is called stable if error due to hypothesis obtained using almost similar (differing by an instance) samples is very small. There have been various notions of stability ([4],[19]). We show that the exponential loss function based ERM is $\beta-$uniformly stable using the following definition of stability from [14]:

Definition 2.4 (Uniform stability). Let S and $S^{'}$ be any two training samples that differ by a single point. Then, a learning algorithm $\mathbf{A}$ is uniformly $\beta-$stable if the hypotheses it returns when trained on any such samples S and $S^{'}$ satisfy

$$\forall \mathbf{z} \in \mathbf{Z},\ |l(h_S,\mathbf{z}) - l(h_{S'},\mathbf{z})| \leq \beta \tag{10}$$

Lemma 2.5. *ExpERM is uniformly $\beta-$stable with $\beta = e^{B\tilde{R}}(e^{2B\tilde{R}}-1)$.*

Remark 1. *We already know some properties of exponential loss function like Fisher consistency and classification calibration. For a given $\mathbf{x}$, true risk of the classifier $h(\mathbf{x})$ trained by using $l_{exp}(h(\mathbf{x}),(\mathbf{x},y))$ is*

$$\eta(\mathbf{x})e^{-(1)h(\mathbf{x})} + (1-\eta(\mathbf{x}))e^{-(-1)h(\mathbf{x})}. \tag{11}$$

The population minimizer of exponential loss function, $l_{exp}(h(\mathbf{x}),(\mathbf{x},y))$, i.e., the decision rule which minimizes (11) is $f^{*,exp}(\mathbf{x}) = sign\left(\frac{1}{2}\ln\frac{\eta(\mathbf{x})}{1-\eta(\mathbf{x})}\right)$ and is same as Bayes decision rule. Hence, it is Fisher consistent. It is known that exponential loss function, $l_{exp}(h(\mathbf{x}),(\mathbf{x},y))$ is classification calibrated as defined in [3].

2.3 Binary ExpERM classifier

We obtain a binary classifier by using exponential loss function, $l_{exp}(h,\mathbf{z})$ as surrogate function in (7) and minimize the empirical risk (**ExpERM**), i.e.,

$$h_S^{exp} = \arg\min_{h\in\mathcal{H}} \frac{1}{m}\sum_{i=1}^{m} l_{exp}(h,\mathbf{z}_i) = \arg\min_{h\in\mathcal{H}} \frac{1}{m}\sum_{i=1}^{m} e^{-y_i h(\mathbf{x}_i)}. \tag{12}$$

For linear classifiers, it can be rewritten as follows:

$$h_S^{exp} = \arg\min_{\mathbf{w}\in\mathbb{R}^n, b\in\mathbb{R}} \frac{1}{m}\sum_{i=1}^{m} e^{-y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i+b)} \tag{13}$$

Another, very popular version of ERM is Regularized Empirical Risk Minimization. It is a learning rule in which one jointly minimizes the empirical risk and a regularization function $R:\mathbb{R}^d \to \mathbb{R}$ and the regularized loss minimization rule outputs a hypothesis,

$$\arg\min_{h\in\mathcal{H}}(L_{S,l_{sur}}(h) + R(h)) \tag{14}$$

We also give a regularized version of ExpERM with Tikhonov regularization function (**RegExpERM**) as follows, for a given $\lambda > 0$:

$$h_S^{exp(reg)} = \arg\min_{h\in\mathcal{H}} \frac{1}{m}\sum_{i=1}^{m} e^{-y_i h(\mathbf{x}_i)} + \lambda\|h\|_2^2 \tag{15}$$

The value of regularization function measures the complexity of the hypothesis space. The optimal $h_S^{exp(reg)}$ trades off between the empirical risk and hypothesis complexity. An interesting interpretation for λ in terms of the robustness properties of the learning rule or the classifier is presented in Chapter 10 of [21]. Another role of regularization term in RegExpERM is to avoid the unbounded objective condition.

We now obtain the ExpERM classifier using lr_{exp}. Consider maximizing the average of the loss-reward function over all examples:

$$\mathbf{W}_{max} = \arg\max_{\mathbf{w}\in\mathbb{R}^n, b\in\mathbb{R}} \frac{1}{m}\sum_{i=1}^{m}(1-e^{-y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i+b)}). \tag{16}$$

We observe that the classifiers h_S^{exp} and $\mathbf{W}_{max}$ are the same. The same equivalence holds for the regularized version with regularizer λ, i.e., $h_S^{exp(reg)}$ and $\mathbf{W}_{max}^{reg}$ defined as:

$$\mathbf{W}_{max}^{reg} = \arg\max_{\mathbf{w}\in\mathbb{R}^n, b\in\mathbb{R}} \frac{1}{m}\sum_{i=1}^{m}(1-e^{-y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i+b)}) - \lambda\|\mathbf{w}\|_2^2. \tag{17}$$

For our numerical experiments, we consider the datasets which are linearly inseparable. Then, the linear classifiers for such datasets based on our loss function, ExpERM (16), will be finite:

Theorem 2.6. *Consider the following optimization problem,*

$$(P) \qquad \max_{\mathbf{w}\in\mathbb{R}^n, b\in\mathbb{R}} \sum_{i=1}^{m}(1-e^{-y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i+b)})$$

(P) will have a finite solution $(\mathbf{w},b)$ if and only if the given data is linearly inseparable.

As we are now sure about the existence of ExpERM classifier, we are interested in its uniqueness. If the classifier is not unique, then one needs to decide out of the many classifiers with same total

empirical risk which one to choose for prediction as the generalization error of multiple optimal classifiers can differ. From Lemma 2.1, we know that $l_{exp}(h(\mathbf{x}),(\mathbf{x},y))$ is convex (but not necessarily strict). However, due to linear inseparability of data we expect that $\sum_{i=1}^{m} l_{exp}(h(\mathbf{x}_i),(\mathbf{x}_i,y_i))$ is strictly convex function of $(\mathbf{w},b)$. Under mild assumption on the data, we show in the following theorem that the optimal classifier obtained from ExpERM is unique. The proof is given in Appendix.

Theorem 2.7. *If the data matrix $A_{(n+1)\times m}$ (with $m > n+1$) defined as follows*

$$A := \begin{bmatrix} x_{11} & x_{21} & \dots & x_{m1} \\ x_{12} & x_{22} & \dots & x_{m2} \\ \vdots & \ddots & & \vdots \\ x_{1n} & x_{2n} & \dots & x_{mn} \\ 1 & 1 & 1 & 1 \end{bmatrix},$$

is full rank, then optimal classifier obtained from ExpERM is unique.

Remark 2. *As in the proof of Theorem 2.7, if data matrix A as above is full rank, using mid-point convexity (as the function is continuous) and AM-GM inequality, one can show that* (13)*, the empirical risk minimization involving exponential loss function, is in fact a strictly convex problem having a unique solution.*

It is known that objective function of SVM QP is strictly convex only in normal $\mathbf{w}$ and not in b. This leads to $\mathbf{w}$ being uniquely determined but the choice of b is not unique. The uniqueness of SVM classifiers for binary classification problem is studied in [6]. The authors give closed form expression for the threshold b, which is unique. Theorem 2.7 shows that ExpERM jointly determines the classifier uniquely i.e., both $\mathbf{w}$ and b are unique.

Remark 3. *Let B be the data matrix defined as below.*

$$B := \begin{bmatrix} x_{11} & x_{21} & \dots & x_{m1} \\ x_{12} & x_{22} & \dots & x_{m2} \\ \vdots & \ddots & & \vdots \\ x_{1n} & x_{2n} & \dots & x_{mn} \end{bmatrix}_{n\times m}$$

We argue that full rank assumption on B is not same as full rank assumption on A i.e. $rank(B) = n$ does not imply $rank(A) = n+1$. Let $B'_{n\times n}$ be the submatrix of B of rank n. Therefore, the rows of B' span the whole of $\mathbb{R}^n$. If we append a row $[1,\dots,1]$ to B', then for some $\{\beta_1,\dots,\beta_n\}$, we have $\sum_{j=1}^{n}\beta_j x_{ij} = 1,\ \forall i = 1,\dots,m$. If there is a $(n+1)\times(n+1)$ dimensional submatrix A' of matrix A such that $\sum_{j=1}^{n}\beta_j x_{(n+1)j} = 1$, then in such cases rank of A' and inturn of A continue to be n. In fact, given that rank of B is n, this condition on last column of A' can be verified easily to be both necessary and sufficient for A to have rank n.

3 PERFORMANCE GUARANTEES ON BINARY ExpERM CLASSIFIERS

In this section, we specialize various high probability bounds on the generalization error of the hypothesis. There have been a variety of generalization bounds based on different measures of the complexity of the hypothesis set $\mathcal{H}$ and some based on stability of the specific algorithm used.

3.1 Algorithmic stability bounds

Recall that some generalization error bounds exploit the algorithmic stability property to give algorithm dependent guarantees. As seen in Lemma 2.5, ExpERM is $\beta-$ stable with $\beta = e^{B\tilde{R}}(e^{2B\tilde{R}}-1)$. We see that the coefficient β is independent of m. Also, under the assumption on the boundedness of hypothesis class and feature space, loss function $l_{exp}(h(\mathbf{x}),(\mathbf{x},y))$ is bounded by$M = e^{B\tilde{R}}$. As all the assumptions and conditions of Theorem 11.1 from [14] are satisfied, we have the following:

Corollary 3.1. *Let ExpERM be implemented on a sample S of m points drawn i.i.d. according to distribution* $\mathbf{D}$*. Then, with probability at least $1-\delta$ over the sample S drawn, the following holds :*

$$L_{\mathbf{D}}(h_S^{exp}) \le L_S(h_S^{exp}) + e^{B\tilde{R}}\left[(e^{2B\tilde{R}}-1)\left(1+\sqrt{2m\log(\frac{1}{\delta})}\right)+\sqrt{\frac{\log(\frac{1}{\delta})}{2m}}\right] \quad (18)$$

Above result gives a data-dependent bound on the deviation of the unknown generalization error $L_{\mathbf{D}}(h_S^{exp})$ from the computable empirical error $L_S(h_S^{exp})$; note that this empirical error is obtained from a stable learning algorithm ExpERM.

3.2 Sample complexity type bounds

Here, we give PAC like bounds on the generalization error of the hypothesis obtained by solving regularized version of ExpERM. We consider the Convex-Lipschitz-Bounded $(\mathcal{H},\mathbf{Z},l_{exp},\rho)$ and Convex-Smooth-Bounded $(\mathcal{H},\mathbf{Z},l_{exp},\gamma)$ learning problems defined in [18]. These problems were found to be PAC learnable. Following are bounds on the generalization error of $h_S^{exp}(\mathbf{x})$ in terms of the generalization error of the optimal classifier $h^{*,exp} = \arg\min_{w\in\mathcal{H}} L_{\mathbf{D},l_{exp}}(w)$. The proofs are given in appendix.

Corollary 3.2. (α) *Suppose that* $\mathbf{D}$ *is a distribution over $\mathbf{Z} = \mathbf{X}\times Y$ such that with probability 1 we have that $\|\underline{\mathbf{x}}\|_2 \le \tilde{R}$ Consider the hypothesis $h_S^{exp(reg)}(\mathbf{x})$ obtained by implementing RegExpERM on $(\mathcal{H},\mathbf{Z},l_{exp},\rho)$. For any training set size m, let $\lambda = \sqrt{\frac{2\rho^2}{B^2m}}$. Then, RegExpERM rule* (15) *satisfies*

$$E_S[L_{\mathbf{D},l_{exp}}(h_S^{exp(reg)})] \le \min_{w\in\mathcal{H}} L_{\mathbf{D},l_{exp}}(w) + \rho B\sqrt{\frac{8}{m}}. \quad (19)$$

In particular, for every $\epsilon > 0$, if $m \ge \frac{8\rho^2B^2}{\epsilon^2}$ then for every distribution $\mathbf{D}$*, $E_S[L_{\mathbf{D},l_{exp}}(h_S^{exp(reg)})] \le \min_{w\in\mathcal{H}} L_{\mathbf{D},l_{exp}}(w)+\epsilon$.*

(β) *Also, RegExpERM satisfies,*

$$P[L_{\mathbf{D},l_{exp}}(h_S^{exp(reg)}) \le \min_{w\in\mathcal{H}} L_{\mathbf{D},l_{exp}}(w)+\epsilon] \ge 1-\delta \quad (20)$$

with $m \ge \frac{8\rho^2B^2}{\epsilon^2}$ and $\lambda = \sqrt{\frac{2\rho^2}{B^2m}}$.

Corollary 3.3. (α) *Suppose that* $\mathbf{D}$ *is a distribution over $\mathbf{Z} = \mathbf{X}\times Y$ such that with probability 1 we have that $\|\underline{\mathbf{x}}\|_2 \le \tilde{R}$.*

Consider the hypothesis $h_S^{exp(reg)}(\mathbf{x})$ *obtained by implementing RegExpERM on* $(\mathcal{H}, \mathbf{Z}, l_{exp}, \gamma)$ *For any* $\epsilon \in (0,1)$, *let* $m \geq \frac{192\gamma B^2}{\epsilon^2}$ *and set* $\lambda = \frac{\epsilon}{2B^2}$. *Then, for every distribution* $\mathbf{D}$,

$$E_S[L_{\mathbf{D},l_{exp}}(h_S^{exp(reg)})] \leq \min_{w\in\mathcal{H}} L_{\mathbf{D},l_{exp}}(w) + \epsilon \tag{21}$$

(β) *Also, RegExpERM satisfies,*

$$P[L_{\mathbf{D},l_{exp}}(h_S^{exp(reg)}) \leq \min_{w\in\mathcal{H}} L_{\mathbf{D},l_{exp}}(w) + \epsilon] \geq 1-\delta \tag{22}$$

with $m \geq \frac{192\gamma B^2}{\epsilon^2}$ *and* $\lambda = \frac{\epsilon\delta}{2B^2}$

Remark 4. *Since,* $l_{exp}(h(\mathbf{x}),(\mathbf{x},y))$ *is both* $\rho-$*Lipschitz and* $\gamma-$*smooth with* $\rho = \gamma$, *we observe that the bound in* (19) *(or* (21)*) is tighter if* $\gamma < 48$ *(or* $\gamma \geq 48$*).*

Remark 5. *Another type of bound on generalization error of linear predictors with bounded norm constraint as given in Chapter* 26 *of [18] is via Rademacher complexity. Since,* $l_{exp}(h(\mathbf{x}),(\mathbf{x},y))$ *satisfies all the assumptions and sufficient condition of Theorem* 26.12, *by substituting* $\rho = \tilde{R}e^{B\tilde{R}}$ *and* $c = e^{B\tilde{R}}$, *we get the following.*

For any $\delta \in (0,1)$, *with probability at least* $1-\delta$ *over the choice of an i.i.d. sample of size m,*

$$\forall h \in \mathcal{H},\ L_{\mathbf{D},l_{exp}}(h) \leq L_{S,l_{exp}}(h) + \frac{e^{B\tilde{R}}}{\sqrt{m}}\left(2B\tilde{R}^2 + \sqrt{2\ln(2/\delta)}\right) \tag{23}$$

4 COMPUTATIONAL EXPERIMENTS

In this section, we obtain linear classifiers by ExpERM and RegExpERM scheme described in Section 2 for a variety of binary and multi-class datasets taken from [1], [11] and [9]. Using Wilcoxon signed-rank test, we statistically compare 5 or 10 fold cross validation error (test error) and runtime due to ExpERM with methods like SVM and Boosting. We implement ExpERM and RegExpERM in *AMPL Version 20170616* and use *SNOPT 7.5-1.2* solver. SVM (with linear kernels) has been implemented in *R version 3.1.3 (2015-03-09)* and Boosting in *Python 2.7.13*. All the computations are performed on machine equipped with 4 Intel Xeon 2.13 GHz cores and 64 GB RAM. As shown in Subsection 2.3, minimization of loss function, $l_{exp}(h(\mathbf{x}),(\mathbf{x},y))$ is same as maximization of loss-reward function, $lr_{exp}(h(\mathbf{x}),(\mathbf{x},y))$. So, for numerical experiments, we solve (16) and (17) and do not include the bounded norm constraint. As part of reproducible research regime, we can make the code available, if needed.

Wilcoxon signed-rank test is a non-parametric test which uses both sign and magnitude of the difference between the observations [15]. If the null hypothesis is $H_0 : \mu = \mu_0$, then depending upon the statement which a person is interested in testing, alternative hypothesis could be two-sided ($H_1 : \mu \neq \mu_0$) or one-sided ($H_1 : \mu \lessgtr \mu_0$). The idea of this test is to rank the performance differences in absolute value in ascending order and then calculate the sum of ranks for positive (W^+) and negative (W^-) differences separately. Let $w^*_{d,\alpha}$ be the critical values of W for d datasets at α level of significance. For two sided tests, if the observed value of statistic $w = \min(W^+, W^-) \leq w^*_\alpha$, the null hypothesis is rejected. For one-sided tests, if the alternative is $H_1 : \mu > \mu_0$, reject H_0 if $W^- \leq w^*_{d,\alpha}$; and if the alternative is $H_1 : \mu < \mu_0$, reject H_0 if $W^+ \leq w^*_{d,\alpha}$.

4.1 Binary classification problems

First, we consider moderate sized binary classification datasets with both balanced and imbalanced classes. Of the datasets described in Table 1, 5 datasets (Appendicitis, Vehicle Silhouettes, Image Segmentation, Winconsin, Page Blocks) have imbalanced classes. Table 1 shows the details of datasets and comparison of ExpERM with SVM ($C = 1$) and Boosting technique in terms of the CV error. We determined C in SVM through cross validation. We only report SVM CV error with $C = 1$ (Column 5 of Table 1) as it is within 1% of the above. We have used decision trees as base classifiers for Boosting. We iterate over a list $L = \{5, 10, \ldots, 145, 150\}$ denoting the maximum number of base classifiers at which Boosting is terminated and average over it to compute the final Boosting CV error.

We performed Wilcoxon signed-rank test [15] on the CV error of ExpERM (μ_E) and SVM (μ_S) based classifier to test if there is any significant difference between them. The null hypothesis $H_0^S : \mu_S = \mu_E$ is tested against the alternative $H_1^S : \mu_S < \mu_E$. The sum of positive ranks is $W_S^+ = 108$ and sum of negative ranks is $W_S^- = 28$. The observed Wilcoxon test statistic value, W_S^- is more than tabulated value of $W_{16,0.01} = 23$. So, we cannot reject H_0^S at 1% level of significance. Hence, we can say that there is no significant difference between the CV error of ExpERM and SVM based binary classifiers. Similarly, for comparing the CV error of ExpERM and Boosting, consider the null hypothesis $H_0^B : \mu_B = \mu_E$ against the alternative $H_1^B : \mu_B < \mu_E$. The sum of positive ranks is $W_B^+ = 86$ and sum of negative ranks is $W_B^- = 50$. The observed Wilcoxon test statistic value, W_B^- is more than tabulated value of $W_{16,0.01} = 23$. So, we cannot reject H_0^B at 1% level of significance. Hence, we can say that there is no significant difference between the CV error of ExpERM and Boosting based binary classifiers.

It is observed that there is no significant improvement after regularizing, i.e., ExpERM and RegExpERM are performing almost same. Table 2 to Table 9 show the comparison of RegExpERM with SVM for various regularization parameter λ on some of the above datasets given in Table 1.

For some large feature sized datasets, we implemented ExpERM and SVM. In Table 10, we report test error and runtime of both ExpERM and SVM. We performed Wilcoxon signed-rank test on the runtime of ExpERM (t_E) and SVM (t_S) based classifier to test if there is any significant difference between the runtime of these algorithms. The null hypothesis $H_0^{tS} : t_S = t_E$ is tested against the alternative $H_1^{tS} : t_E < t_S$. The sum of positive ranks is $W_{tS}^+ = 0$ and sum of negative ranks is $W_{tS}^- = 45$. The observed Wilcoxon test statistic value, W_{tS}^+ is less than tabulated value of $W_{9,0.01} = 3$. So, we reject H_0^{tS} at 1% level of significance. Hence, we can say that runtime of ExpERM is significantly less than that of SVM.

4.2 Multi-class classification problems

We implemented ExpERM on some multi-class datasets and compared its CV error with SVM. Table 11 gives details of the datasets and the number of folds, k and compares the results of ExpERM and SVM. As the CV error of SVM with different values of C was almost same, for the sake of consistency we work with $C = 1$. The SVM module in *Kernlab* package in R for multi-class classification

S.no	Dataset name (# of examples, # of features)	k	ExpERM CV error	SVM CV error (C=1)	Boosting CV error
1	Spambase (4601,57)	10	0.082374	0.07064	0.0685
2	Blood transfusion service center (748,4)	5	0.217971	0.236564	0.235284
3	Appendicitis (106,7)	5	0.150600	0.132468	0.141125
4	Default of Credit Card (30000,23)	10	0.199062	0.190767	0.182633
5	Pima Indians Diabetes(768,8)	10	0.226594	0.230388	0.256642
6	Sonar, Mines vs Rocks (208,60)	5	0.278630	0.2499	0.276030
7	Magic Gamma Telescope (19020,10)	10	0.253733	0.2084	0.167777
8	COIL : Insurance (9822,85)	10	0.060578	0.059864	0.060442
9	Vehicle Silhouettes (846,18)	5	0.033087	0.030734	0.039826
10	Image Segmentation (2308,19)	5	0.003466	0.002601	0.003554
11	Winconsin (683,9)	5	0.039513	0.032335	0.045207
12	Page Blocks (5472,10)	10	0.059395	0.049	0.047025
13	Banana (5300,2)	10	0.448302	0.448302	0.275038
14	Haberman (306,3)	5	0.261449	0.268059	0.351451
15	Titanic (2201,3)	10	0.223970	0.223998	0.221732
16	Skin Segmentation (245057,4)	10	0.107171	0.070979	0.0725

Table 1: Comparison of ExpERM with SVM and Boosting methods based on their CV errors

λ	RegExpERM CV error	SVM CV error
0.05	0.081739	**0.067814**
0.1	**0.081522**	0.069766
0.5	0.081739	0.072162
1	0.082374	0.071287

Table 2: Spambase dataset

λ	RegExpERM CV error	SVM CV error
0.1	**0.226594**	0.229242
0.5	0.230498	**0.222664**
1	0.230524	0.227909
2	0.240939	0.234454

Table 3: Pima dataset

λ	RegExpERM CV error	SVM CV error
0.05	0.217971	**0.237888**
7	0.213770	0.23932
20	0.210001	0.237942
100	**0.207579**	0.237969

Table 4: Blood Transfusion dataset

λ	RegExpERM CV error	SVM CV error
0.03	0.216260	0.283275
0.05	**0.211498**	**0.216144**
0.09	0.221022	0.268873
0.1	0.221022	0.255633

Table 5: Sonar dataset

λ	RegExpERM CV error	SVM CV error
1	**0.035122**	0.035133
1.8	0.035133	**0.027834**
2.5	0.036604	0.030754
5	0.040983	0.038117

Table 6: Wisconsin dataset

λ	RegExpERM CV error	SVM CV error
0.1	0.030721	0.039025
0.5	0.029544	0.033122
0.8	0.028361	**0.029558**
2.5	**0.027184**	0.034278

Table 7: Vehicle Silhouettes dataset

λ	RegExpERM CV error	SVM CV error
0.01	0.003466	0.149817
1.2	0.003466	0.00953
2.5	**0.002600**	**0.003465**
15	0.002600	0.006067

Table 8: Image Segmentation dataset

λ	RegExpERM CV error	SVM CV error
0.01	0.059395	0.155346
0.1	0.059760	0.065789
0.5	0.059029	0.057749
2	**0.058481**	**0.048611**

Table 9: Page Blocks dataset

S.no	Dataset name (# of examples (train + test), # of features, k)	ExpERM Test error	SVM Test error (C=1)	ExpERM time (in s)	SVM time (in s)
1	Arcene ((200+100),10000)	0.18	0.17	0.72	2.05
2	Dexter ((300+300),20000)	0.09	0.07	2.09	4.09
3	Dorothea ((800+350), 100000)	0.06	0.06	28.78	53.06
4	Gisette ((6000+1000),5000)	0.022	0.024	11.79	72.59
5	Madelon ((2000+600),500)	0.41	0.42	1.03	124.78
6	LSVT_Voice _Rehabilitation ((101+25),310)	0.28	0.32	0.036	0.06
7	Colon Cancer ((50+12),2000)	0.33	0.25	0.10	0.31
8	Duke Breast Cancer ((38+4),7129)	0	0.25	0.27	1.04
9	Leukamia ((38+34),7129)	0.08	0.117	0.28	1.06

Table 10: Comparison of ExpERM with SVM in terms of test error and time taken to train the model.

uses OVO method. For datasets with large number of classes (Letter Recognition dataset), OVO performs better than OVA. To compare the results of ExpERM using OVO method (μ_E^m) and SVM (μ_S^m) classifiers, we performed Wilcoxon signed ranks test on their CV errors. The null hypothesis $H_0^{S(m)} : \mu_S^m = \mu_E^m$ is tested against the alternative $H_1^{S(m)} : \mu_S^m < \mu_E^m$. Since, for one dataset the difference between CV error is 0, only 6 datasets are considered for comparison. The sum of positive ranks is $W_{S(m)}^+ = 18$ and sum of negative ranks is $W_{S(m)}^- = 3$. The observed Wilcoxon test statistic value, $W_{S(m)}^-$ is more than tabulated value of $W_{6,0.05} = 2$. So, we cannot reject $H_0^{S(m)}$ at 5% level of significance. Hence, we have shown that there is no significant difference between the CV error of ExpERM and SVM based binary classifiers used for multi-class problem.

5 DISCUSSION AND ONGOING RESEARCH

Exponential loss function has been popular in frameworks like Adaboost. We propose its use in the ERM setting. ExpERM based classifiers obtained using such a convex loss function inherit uniqueness

S.no	Dataset name (# of examples, # of features, # of classes)	k	ExpERM CV error (OVA)	ExpERM CV error (OVO)	SVM CV error (C=1)
1	Iris(150,4,3)	5	0.04	0.04	0.04
2	Balance(625,4,3)	5	0.128	0.12	0.0832
3	Hayes-Roth (160,4,3)	5	0.46875	0.4625	0.4125
4	NewThyroid (215,5,3)	5	0.023255	0.018605	0.027907
5	Page-Blocks (5472,10,5)	10	0.050073	0.068532	0.03783
6	White wine quality (4898,11,7)	10	0.467326	0.467941	0.478367
7	Letter Recognition (20000,17,26)	10	0.305448	0.169250	0.14645

Table 11: Comparison of ExpERM with SVM for multi-class datasets based on their CV errors

and some desirable statistical properties like classification calibration and consistency. We show some additional properties of these learners in terms of high probability bounds on their generalization error.

We have considered a wide range (small or large features, small or large examples, balanced and imbalanced classes) of datasets and used Wilcoxon signed-rank test to statistically test if there is any difference between the performance of ExpERM and other existing algorithms in terms of CV error (test error) and runtime. We have demonstrated on many datasets that ExpERM scheme outputs a binary classifier with generalization error comparable to existing methods like hinge-loss based SVMs and Boosting. We have also analyzed regularized version of ExpERM. We observe that in a large number of datasets, regularization has no significant improvement on the generalization error. Note that our scheme involving unconstrained differentiable convex minimization problem is easy and scalable [16]. For some large datasets (with both large number of examples and features), we show that the computational time of ExpERM is significantly less than that of SVM. After reducing the multi-class classification problem into many binary ones, our initial computations show that CV error of ExpERM scheme is statistically comparable to the existing SVM based combined class of algorithms. We believe that viewing a modified version of exponential loss function as a loss-reward function gives another explanation for our ExpERM classifiers having performances comparable to some popular classifiers, but, obtained at much less computational cost.

Existing SVM packages (based on LIBSVM) use some fine tuned and tailor made algorithm for the hinge loss based SVM QP. However, we are only using a generic non-linear solver (SNOPT) to solve ExpERM optimization problem. An algorithm exploiting properties of exponential loss function to solve ExpERM can lead to an improvement in its computational time performance. Besides runtime, it was observed that AMPL is taking a significant amount of time to read the specific format data file, particularly for large datasets. This aspect of ExpERM can be improved upon. One interesting question that we are currently looking at is, if ExpERM is implicitly performing some form of regularization. Also, some more theoretical guarantees on the classifier for multi-class classification problems can help in using ExpERM for developing an uncombined multi-class classifier.

ACKNOWLEDGMENTS

We sincerely thank Reviewers for their comments which helped us improve the presentation of this paper. This work was partially supported by a Teaching Assistantship offered by Government of India, through MHRD.

A PROOFS

A.1 Proof of Lemma 2.1

Proof. We will show that $l_{exp}((\mathbf{w}, b), \mathbf{z}_i)$ is a convex function by showing its Hessian to be a positive semi-definite matrix. The first and second partial derivatives of $l_{exp}((\mathbf{w}, b), \mathbf{z}_i)$ are as follows:

$$\begin{aligned}
\frac{\partial l_{exp}((\mathbf{w}, b), \mathbf{z}_i)}{\partial w_j} &= -y_i x_{ij} e^{-y_i(\mathbf{w}^T \mathbf{x}_i + b)} \\
\frac{\partial l_{exp}((\mathbf{w}, b), \mathbf{z}_i)}{\partial b} &= -y_i e^{-y_i(\mathbf{w}^T \mathbf{x}_i + b)} \\
\frac{\partial^2 l_{exp}((\mathbf{w}, b), \mathbf{z}_i)}{\partial {w_j}^2} &= (y_i x_{ij})^2 e^{-y_i(\mathbf{w}^T \mathbf{x}_i + b)} \\
\frac{\partial^2 l_{exp}((\mathbf{w}, b), \mathbf{z}_i)}{\partial w_j \partial b} &= y_i^2 x_{ij} e^{-y_i(\mathbf{w}^T \mathbf{x}_i + b)} \\
\frac{\partial^2 l_{exp}((\mathbf{w}, b), \mathbf{z}_i)}{\partial b^2} &= y_i^2 e^{-y_i(\mathbf{w}^T \mathbf{x}_i + b)} \\
\frac{\partial^2 l_{exp}((\mathbf{w}, b), \mathbf{z}_i)}{\partial w_j \partial w_k} &= y_i^2 x_{ij} x_{ik} e^{-y_i(\mathbf{w}^T \mathbf{x}_i + b)}
\end{aligned}$$

The eigenvalues of the Hessian H in terms of the above partials are $\left\{ y_i^2 e^{-y_i(\mathbf{w}^T \mathbf{x}_i + b)} \left(\sum_{j=1}^{n} x_{ij}^2 + 1 \right), 0, \ldots, 0 \right\}$. This implies that it is a positive semi-definite matrix and in turn, $l_{exp}((\mathbf{w}, b), \mathbf{z}_i)$ is a convex function. □

A.2 Proof of Lemma 2.2

Proof. A sufficient condition for a function to be $\rho-$Lipschitz is that the gradient of the function is uniformly bounded. Since, $\|\mathbf{w}\|_2 \leq B$ and $\|\underline{\mathbf{x}}\|_2 \leq \tilde{R}$, using Cauchy Schwartz inequality, we obtain $|\mathbf{w}^T \underline{\mathbf{x}}| = \|\mathbf{w}^T \underline{\mathbf{x}}\| \leq B\tilde{R}$.

$$\begin{aligned}
\|\nabla l_{exp}((\mathbf{w}, b), \mathbf{z})\| &= \left[\sum_{j=1}^{n+1} x_{ij}^2 e^{-2y_i(\mathbf{w}^T \mathbf{x}_i + b)} \right]^{1/2} \\
&\leq \left[\sum_{j=1}^{n+1} x_{ij}^2 e^{|2y_i(\mathbf{w}^T \mathbf{x}_i + b)|} \right]^{1/2} \\
&\leq \left[\sum_{j=1}^{n+1} x_{ij}^2 e^{2B\tilde{R}} \right]^{1/2} \\
&\leq \tilde{R} e^{B\tilde{R}}
\end{aligned}$$

Therefore, the gradient of $l_{exp}((\mathbf{w}, b), \mathbf{z})$ is uniformly bounded with Lipschitz constant $\rho = \tilde{R}e^{B\tilde{R}}$. □

A.3 Proof of Lemma 2.3

Proof. A sufficient condition for a function to be γ-smooth is that the second derivative or Hessian of the function is uniformly bounded. The induced matrix norm is the spectral norm given by the largest singular value of A, i.e., square root of the largest eigenvalue of $A * A$.

$$\|A\|_2 = \max_j \sigma_j$$

The Hessian H of $l_{exp}((\mathbf{w}, b), \mathbf{z}_i)$ is a positive semi-definite matrix with only non-zero eigenvalue $y_i^2(e^{-y_i(\mathbf{w}^T\mathbf{x}_i+b)}(\sum_{j=1}^{n} x_{ij}^2 + 1))$. So, by definition, the spectral norm of H can be bounded as follows :

$$\begin{aligned} \|H\|_2 &= y_i^2(e^{-y_i(\mathbf{w}^T\mathbf{x}_i+b)}(\sum_{j=1}^{n} x_{ij}^2 + 1)) \\ &= \|\underline{\mathbf{x}}_i\|_2 e^{-y_i(\mathbf{w}^{\mathrm{T}}\underline{\mathbf{x}}_i)} \\ &\leq \tilde{R}e^{B\tilde{R}} \end{aligned}$$

Therefore, $l_{exp}((\mathbf{w}, b), \mathbf{z})$ is $\gamma-$smooth with $\gamma = \tilde{R}e^{B\tilde{R}}$. □

A.4 Proof of Lemma 2.5

Proof. Let $h_S = (\mathbf{w}^1, b^1)$ and $h_{S'} = (\mathbf{w}^2, b^2)$. Without loss of generality, we can write, $\begin{bmatrix}\mathbf{w}^2 \\ b^2\end{bmatrix} = \begin{bmatrix}\mathbf{w}^1 \\ b^1\end{bmatrix} + \begin{bmatrix}\mathbf{a}^1 \\ a^2\end{bmatrix}$ For proving $\beta-$stability we need to uniformly upper bound $|l_{exp}(h_S, \mathbf{z}) - l_{exp}(h_{S'}, \mathbf{z})|$ for all $\mathbf{z} \in \mathbf{Z}$.

$$\begin{aligned} |l_{exp}(h_S, \mathbf{z}) - & l_{exp}(h_{S'}, \mathbf{z})| \\ &= |l_{exp}((\mathbf{w}^1, b^1), (\mathbf{x}, y)) - l_{exp}((\mathbf{w}^2, b^2), (\mathbf{x}, y))| \\ &= |e^{-y(\mathbf{w}^{1\mathrm{T}}\mathbf{x}+b^1)} - e^{-y(\mathbf{w}^{2\mathrm{T}}\mathbf{x}+b^2)}| \\ &= |e^{-y(\mathbf{w}^{1\mathrm{T}}\mathbf{x}+b^1)}(1 - e^{-y(\mathbf{a}^{1\mathrm{T}}\mathbf{x}+a^2)})| \\ &= |e^{-y(\mathbf{w}^{1\mathrm{T}}\mathbf{x}+b^1)}||(1 - e^{-y(\mathbf{a}^{1\mathrm{T}}\mathbf{x}+a^2)})|. \end{aligned}$$

Since, $\|\mathbf{w}\| \leq B$, we get $\|(\mathbf{a}^1, a^2)\| \leq 2B$. Also,

$$e^{-B\tilde{R}} \leq e^{-y\mathbf{w}^{\mathrm{T}}\mathbf{x}} \leq e^{B\tilde{R}}, \text{ and } e^{-2B\tilde{R}} \leq e^{-y(\mathbf{a}^{1\mathrm{T}}\mathbf{x}+a^2)} \leq e^{2B\tilde{R}}.$$

Therefore, we have $|(1 - e^{-y(\mathbf{a}^{1\mathrm{T}}\underline{\mathbf{x}}+a^2)})| \leq (e^{2B\tilde{R}} - 1)$. This is true because if $p \in [-r, r]$, then $|(1-e^{-p})| \leq (e^r - 1)$ as $(1-e^{-r}) \leq (e^r - 1)$. Using the above inequalities, we get $|l_{exp}(h_S, \mathbf{z}) - l_{exp}(h_{S'}, \mathbf{z})| \leq e^{B\tilde{R}}(e^{2B\tilde{R}} - 1)$. Hence, ExpERM is uniformly $\beta-$stable with $\beta = e^{B\tilde{R}}(e^{2B\tilde{R}} - 1)$. □

A.5 Proof of Theorem 2.6

Proof. We will first prove that (P) will have an objective value arbitrarily close to m if and only if the given data $(\mathbf{x}_i, y_i)$, $i = 1, \ldots, m$ is linearly separable and then take the negation of this to show that theorem statement is true.

Suppose the given data is linearly separable. Then, there exist $\mathbf{w} \in \mathbb{R}^n$ and b such that

$y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 0$ which implies $1 - e^{-y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i+b)} \geq 0 \; \forall i = 1, \ldots, m$

If $a_1 \in \mathbb{R}_+$ and $a_2 \in \mathbb{R}_+$ s.t. $a_1 < a_2$ then, $1 - e^{-a_1} < 1 - e^{-a_2}$. Hence, for a given $\mathbf{w} \in \mathbb{R}^n$ and b, we can always find a $\mathbf{w}' = \alpha\mathbf{w}$ and $b' = \alpha b$ for some $\alpha > 0$ s.t.

$$\sum_{i=1}^{m}(1 - e^{-y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i+b)}) < \sum_{i=1}^{m}(1 - e^{-y_i(\mathbf{w}'^{\mathrm{T}}\mathbf{x}_i+b')}).$$

This implies that, given a $(\mathbf{w}, b)$, one can always find $(\mathbf{w}', b')$ whose objective value is higher than that of the former. Therefore, (P) will have an objective value arbitrarily close to m.

Suppose (P) has an objective value arbitrarily close to m. We know that,

$$\sup_{\mathbf{w}, b} \sum_{i=1}^{m}(1 - e^{-y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i+b)}) = m$$

$$\text{which implies } \sup_{\mathbf{w}, b}(1 - e^{-y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i+b)}) = 1.$$

So, there exists a sequence $\{(\mathbf{w}^k, b^k)\}_k$ which makes $e^{-y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i+b)}$ arbitrarily close to 0. This can be possible only when $y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b)$ goes to ∞ in limit, i.e., $y_i(\mathbf{w}^{\mathrm{T}}\mathbf{x}_i + b) \geq 0 \; \forall \; i = 1, \ldots, m$. That is, all the examples are correctly classified and the data is linearly separable. □

A.6 Proof of Theorem 2.7

Proof. Suppose the optimal classifier is not unique i.e. there exist two distinct optimal classifiers $\underline{\mathbf{w}}^* := (\mathbf{w}^*, b^*)$ and $\underline{\tilde{\mathbf{w}}} := (\tilde{\mathbf{w}}, \tilde{b})$. For notational convenience, define $\underline{\mathbf{x}}_i := (\mathbf{x}_i, 1)$. Let loss incurred on the i^{th} data point by $\underline{\mathbf{w}}^*$ and $\underline{\tilde{\mathbf{w}}}$ is $l_i^{\underline{\mathbf{w}}^*} = e^{-y_i(\underline{\mathbf{w}}^{*T}\underline{\mathbf{x}}_i)}$ and $l_i^{\underline{\tilde{\mathbf{w}}}} = e^{-y_i(\underline{\tilde{\mathbf{w}}}^{\mathrm{T}}\underline{\mathbf{x}}_i)}$ respectively. Also, let the total loss for $\underline{\mathbf{w}}^*$ and $\underline{\tilde{\mathbf{w}}}$ be $L^{\underline{\mathbf{w}}^*} := \sum_{i=1}^{m} l_i^{\underline{\mathbf{w}}^*}$ and $L^{\underline{\tilde{\mathbf{w}}}} := \sum_{i=1}^{m} l_i^{\underline{\tilde{\mathbf{w}}}^T}$. Due to optimality, $L^{\underline{\mathbf{w}}^*} = L^{\underline{\tilde{\mathbf{w}}}}$.

Since, (12) is convex, so the set of optimal solution is also be convex. Hence, the total loss for any convex combination classifier $\underline{\mathbf{w}}^\alpha := (\alpha\mathbf{w}^* + (1-\alpha)\tilde{\mathbf{w}}, \alpha b^* + (1-\alpha)\tilde{b}) = \alpha\underline{\mathbf{w}}^* + (1-\alpha)\underline{\tilde{\mathbf{w}}}$, is same as that of $\underline{\mathbf{w}}^*$ and $\underline{\tilde{\mathbf{w}}}$. i.e. $L^{\underline{\mathbf{w}}^*} = L^{\underline{\tilde{\mathbf{w}}}} = L^\alpha$. Let $\alpha = 1/2$ Consider the loss $l_i^{1/2}$ on the i^{th} point due to $1/2$ convex combination classifier, $\underline{\mathbf{w}}^{1/2}$.

$$\begin{aligned} l_i^{1/2} &= e^{-y_i\left(\frac{\underline{\mathbf{w}}^* + \underline{\tilde{\mathbf{w}}}}{2}\right)^T\underline{\mathbf{x}}_i} \\ &= \left[e^{-y_i(\underline{\mathbf{w}}^{*T}\underline{\mathbf{x}}_i)}\right]^{1/2}\left[e^{-y_i(\underline{\tilde{\mathbf{w}}}^{\mathrm{T}}\underline{\mathbf{x}}_i)}\right]^{1/2} \end{aligned}$$

Now, using the inequality that $AM \geq GM$ we get,

$$l_i^{1/2} \leq \frac{\left[e^{-y_i(\underline{\mathbf{w}}^{*T}\underline{\mathbf{x}}_i)} + e^{-y_i(\underline{\tilde{\mathbf{w}}}^{\mathrm{T}}\underline{\mathbf{x}}_i)}\right]}{2} \tag{24}$$

Summing (24) over all data points we get,

$$\begin{aligned} L^{1/2} &= \sum_{i=1}^{m} l_i^{1/2} && (25) \\ &\leq \frac{\left[\sum_{i=1}^{m} l_i^{\underline{\tilde{\mathbf{w}}}^T} + \sum_{i=1}^{m} l_i^{\underline{\mathbf{w}}^*}\right]}{2} && (26) \\ &= L^{\underline{\mathbf{w}}^*} && (27) \end{aligned}$$

Now, we know that $L^{1/2} = L^{\underline{\mathbf{w}}^*}$. This implies that inequality in (24) should be strict for all i. The $AM \geq GM$ inequality is strict if and

only if the individual values are equal. Therefore, we should have, for all i,

$$\begin{aligned} l_i^{\underline{\mathbf{w}}^*} &= l_i^{\underline{\tilde{\mathbf{w}}}} \\ e^{-y_i(\underline{\mathbf{w}}^{*T}\underline{\mathbf{x}}_i)} &= e^{-y_i(\underline{\tilde{\mathbf{w}}}^{\mathbf{T}}\underline{\mathbf{x}}_i)} \\ (\underline{\mathbf{w}}^* - \underline{\tilde{\mathbf{w}}})^T\underline{\mathbf{x}}_i &= 0 \\ (\underline{\mathbf{w}}^* - \underline{\tilde{\mathbf{w}}})^T A &= 0 \end{aligned} \tag{28}$$

Next, we show that if A is full rank, then (28) has 0 as the only solution. Taking transpose on both sides of (28) we get,

$$A^T(\underline{\mathbf{w}}^* - \underline{\tilde{\mathbf{w}}}) = \mathbf{0} \tag{29}$$

$$\begin{bmatrix} A_1^T \\ A_2^T \end{bmatrix}(\underline{\mathbf{w}}^* - \underline{\tilde{\mathbf{w}}}) = \mathbf{0} \tag{30}$$

$$\begin{bmatrix} A_1^T(\underline{\mathbf{w}}^* - \underline{\tilde{\mathbf{w}}}) \\ A_2^T(\underline{\mathbf{w}}^* - \underline{\tilde{\mathbf{w}}}) \end{bmatrix} = \mathbf{0} \tag{31}$$

Since A is full rank, say $n+1 = \min(n+1, m)$, A_1 ($(n+1) \times (n+1)$ dimensional matrix) is also full rank. This implies that $A_1^T(\underline{\mathbf{w}}^* - \underline{\tilde{\mathbf{w}}}) = \mathbf{0}$ has $(\underline{\mathbf{w}}^* - \underline{\tilde{\mathbf{w}}}) = \mathbf{0}$ as unique solution. Therefore, only solution to $A^T(\underline{\mathbf{w}}^* - \underline{\tilde{\mathbf{w}}}) = \mathbf{0}$ is $\mathbf{0}$ which implies that $\underline{\mathbf{w}}^* = \underline{\tilde{\mathbf{w}}}$. This contradicts our starting assumption of two distinct optimal classifiers. Hence, the optimal classifier obtained from ExpERM is unique. □

A.7 Proof of Corollary 3.2

Proof. (α) Clearly, from Lemma 2.1 and 2.2, $l_{exp}(h(\mathbf{x}), (\mathbf{x}, y))$ is convex and $\rho-$ Lipschitz with $\rho = \tilde{R}e^{B\tilde{R}}$. So, $l_{exp}(h(\mathbf{x}), (\mathbf{x}, y))$ along with the bounded hypothesis space constraint satisfies all the sufficient conditions given in Corollary 13.9 from [18]. Hence, (19) follows.

(β) We have, $E_S[L_{\mathbf{D},l_{exp}}(h_S^{exp(reg)})] \leq \min_{w\in H} L_{\mathbf{D},l_{exp}}(w) + \epsilon$ and hence, $E_S[L_{\mathbf{D},l_{exp}}(h_S^{exp(reg)}) - \min_{w\in\mathcal{H}} L_{\mathbf{D},l_{exp}}(w)] \leq \epsilon$. Since, the random variable $L_{\mathbf{D},l_{exp}}(h_S^{exp(reg)}) - \min_{w\in\mathcal{H}} L_{\mathbf{D},l_{exp}}(w)$ is always non-negative, we can apply Markov's inequality to get (20).

□

A.8 Proof of Corollary 3.3

Proof. (α) Clearly, from Lemma 2.1 and 2.3, $l_{exp}(h(\mathbf{x}), (\mathbf{x}, y))$ is convex and $\gamma-$smooth. From Theorem 13.2 and Corollary 13.7 of [18], we have,

$$E_S[L_{\mathbf{D},l_{exp}}(h_S^{exp(reg)}) - L_{S,l_{exp}}(h_S^{exp(reg)})] \leq \frac{48\gamma}{m\lambda}C.$$

Since, $l_{exp}(0, \mathbf{z}) = 1$, we have $C = 1$. Therefore, we get,

$$\begin{aligned} & E_S[L_{\mathbf{D},l_{exp}}(h_S^{exp(reg)})] \leq E_S[L_{S,l_{exp}}(h_S^{exp(reg)})] + \frac{48\gamma}{m\lambda} \\ \leq\ & E_S[L_{S,l_{exp}}(h_S^{exp(reg)}) + \lambda\|h_S^{exp(reg)}\|^2] + \frac{48\gamma}{m\lambda} \\ =\ & L_{\mathbf{D},l_{exp}}(h_S^{exp(reg)}) + \lambda\|h_S^{exp(reg)}\|^2 + \frac{48\gamma}{m\lambda} \end{aligned}$$

We optimize RHS for λ and get the following, $\inf_\lambda(\lambda B^2 + \frac{48\gamma}{m\lambda}) = \sqrt{\frac{192\gamma B^2}{m}}$ and get $\lambda = \sqrt{\frac{48\gamma}{mB^2}}$. If $\sqrt{\frac{192\gamma B^2}{m}} \leq \epsilon$, then we have $m \geq \frac{192\gamma B^2}{\epsilon^2}$. Using this value of m in λ, we get $\lambda \leq \frac{\epsilon}{2B^2}$.

(β) We have, $E_S[L_{\mathbf{D},l_{exp}}(h_S^{exp(reg)})] \leq \min_{w\in\mathcal{H}} L_{\mathbf{D},l_{exp}}(w) + \epsilon$ and hence, $E_S[L_{\mathbf{D},l_{exp}}(h_S^{exp(reg)}) - \min_{w\in\mathcal{H}} L_{\mathbf{D},l_{exp}}(w)] \leq \epsilon$. Since, the random variable $L_{\mathbf{D},l_{exp}}(h_S^{exp(reg)}) - \min_{w\in\mathcal{H}} L_{\mathbf{D},l_{exp}}(w)$ is always non-negative, we can apply Markov's inequality to get (22).

□

REFERENCES

[1] Jesús Alcalá-Fdez, Alberto Fernández, Julián Luengo, Joaquín Derrac, Salvador García, Luciano Sánchez, and Francisco Herrera. 2011. Keel data-mining software tool: data set repository, integration of algorithms and experimental analysis framework. *Journal of Multiple-Valued Logic & Soft Computing* 17 (2011).

[2] Ethem Alpaydin. 2010. *Introduction to Machine Learning* (2nd ed.). The MIT Press.

[3] Peter L Bartlett, Michael I Jordan, and Jon D McAuliffe. 2006. Convexity, classification, and risk bounds. *J. Amer. Statist. Assoc.* 101, 473 (2006), 138–156.

[4] Olivier Bousquet and André Elisseeff. 2002. Stability and Generalization. *J. Mach. Learn. Res.* 2 (March 2002), 499–526. DOI:http://dx.doi.org/10.1162/153244302760200704

[5] Andreas Buja, Werner Stuetzle, and Yi Shen. 2005. Loss functions for binary class probability estimation and classification: Structure and applications. *Working draft, November* (2005).

[6] Christopher J. C. Burges and David J. Crisp. 1999. Uniqueness of the SVM Solution. In *Advances in Neural Information Processing Systems 12, [NIPS Conference, Denver, Colorado, USA, November 29 - December 4, 1999]*. 223–229. http://papers.nips.cc/paper/1735-uniqueness-of-the-svm-solution

[7] CC Chang and CJ Lin. 2011. LIBSVM: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology* 2 (2011), 27:1–27:27.

[8] Janez Demšar. 2006. Statistical Comparisons of Classifiers over Multiple Data Sets. *J. Mach. Learn. Res.* 7 (Dec. 2006), 1–30. http://dl.acm.org/citation.cfm?id=1248547.1248548

[9] Rong-En Fan. 2011. LIBSVM Data: Classification (Binary Class). (2011). Retrieved August 12, 2017 from https://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/binary.html

[10] Peter Flach. 2012. *Machine Learning: The Art and Science of Algorithms That Make Sense of Data.* Cambridge University Press, New York, NY, USA.

[11] M. Lichman. 2013. UCI Machine Learning Repository. (2013). http://archive.ics.uci.edu/ml

[12] Yi Lin. 2004. A note on margin-based loss functions in classification. *Statistics & probability letters* 68, 1 (2004), 73–82.

[13] Hamed Masnadi-Shirazi and Nuno Vasconcelos. 2009. On the design of loss functions for classification: theory, robustness to outliers, and savageboost. In *Advances in neural information processing systems*. 1049–1056.

[14] Mehryar Mohri, Afshin Rostamizadeh, and Ameet Talwalkar. 2012. *Foundations of machine learning.* The MIT press.

[15] D.C. Montgomery and G.C. Runger. 2010. *Applied Statistics and Probability for Engineers, 5th Edition.* Wiley. https://books.google.co.in/books?id=baobAAAAQBAJ

[16] J. Nocedal and S. Wright. 2006. *Numerical Optimization.* Springer New York.

[17] Mark D Reid and Robert C Williamson. 2010. Composite binary losses. *Journal of Machine Learning Research* 11, Sep (2010), 2387–2422.

[18] Shai Shalev-Shwartz and Shai Ben-David. 2014. *Understanding Machine Learning: From Theory to Algorithms.* Cambridge University Press, New York, NY, USA.

[19] Shai Shalev-Shwartz, Ohad Shamir, Nathan Srebro, and Karthik Sridharan. 2010. Learnability, stability and uniform convergence. *Journal of Machine Learning Research* 11, Oct (2010), 2635–2670.

[20] Ingo Steinwart. 2007. How to compare different loss functions and their risks. *Constructive Approximation* 26, 2 (2007), 225–287.

[21] Ingo Steinwart and Andreas Christmann. 2008. *Support vector machines.* Springer Science & Business Media.

[22] Alexandre B Tsybakov. 2004. Optimal aggregation of classifiers in statistical learning. *Annals of Statistics* (2004), 135–166.

[23] Wenzhuo Yang, Melvyn Sim, and Huan Xu. 2014. The Coherent Loss Function for Classification. In *Proceedings of the 31st International Conference on Machine Learning (ICML-14)*, Tony Jebara and Eric P. Xing (Eds.). JMLR Workshop and Conference Proceedings, 37–45.
[24] Tong Zhang. 2004. Statistical behavior and consistency of classification methods based on convex risk minimization. *Annals of Statistics* (2004), 56–85.
[25] Zhihua Zhang, Dehua Liu, Guang Dai, and Michael I Jordan. 2012. Coherence functions with applications in large-margin classification methods. *Journal of Machine Learning Research* 13, Sep (2012), 2705–2734.