Universal algorithms for multinomial logistic regression under Kullback-Leibler game

Raisa Dzhamtyrova[a], Yuri Kalnishkan[a,b]

[a] *Department of Computer Science, Royal Holloway, University of London*
[b] *Laboratory of Advanced Combinatorics and Network Applications, Moscow Institute of Physics and Technology*

Abstract

We consider the framework of competitive prediction, where one provides guarantees compared to other predictive models that are called experts. We propose a universal algorithm predicting finite-dimensional distributions, i.e. points from a simplex, under Kullback-Leibler game. In the standard framework for prediction with expert advice, the performance of the learner is measured by means of the cumulative loss. In this paper we consider a generalisation of this setting and discount losses with time. A natural choice of predictors for the probability games is a class of multinomial logistic regression functions as they output a distribution that lies inside a probability simplex. We consider the class of multinomial logistic regressions to be our experts. We provide a strategy that allows us to 'track the best expert' of this type and derive the theoretical bound on the discounted loss of the strategy. We provide the kernelized version of our algorithm, which competes with a wider set of experts from Reproducing Kernel Hilbert Space (RKHS) and prove a theoretical guarantee for the kernelized strategy. We carry out experiments on three data sets and compare the cumulative losses of our algorithm and multinomial logistic regression.

Keywords: online learning, sequential prediction, competitive prediction, universal algorithms, loss bounds, Kullback-Leibler game

---

Email addresses: `Raisa.Dzhamtyrova@rhul.ac.uk` (Raisa Dzhamtyrova), `Yuri.Kalnishkan@rhul.ac.uk` (Yuri Kalnishkan)

1. Introduction

In statistical learning, usually some assumptions are made about the data generation process, and guarantees are given for a method working under these assumptions. Unlike in statistical learning theory, we consider the adversarial setting, where no stochastic assumptions are made about the data. We consider an online sequential prediction protocol where at each trial $t = 1, 2, \ldots$ a learner observes x_t and attempts to predict an outcome y_t, which is shown to the learner later. We work in the framework of competitive prediction and look for performance guarantees relative to other predictive models called experts. In the standard setting of prediction with expert advice, the performance of a learner is measured by its cumulative loss. In this paper, we consider a generalisation of this setting and discount losses by some factor $\alpha_t \in (0, 1]$ revealed at the beginning of each trial. When all $\alpha_t = 1$, our framework reverts to the standard one.

An important class of games of prediction are probability forecasting games, where the predictions and outcomes are probability distributions on some finite set. In this paper, we consider the Kullback-Leibler game, which is one of the most important probability games. Our experts are a wide class of multinomial logistic regression functions. Each expert follows a particular strategy, which means that it uses some particular parameters of a logistic regression function. Our goal is to develop a merging strategy that suffers loss comparable to the retrospectively best expert. If we use weights decreasing for old data, we get a strategy that performs as well as the best expert on *recent* trials; this can be thought of as a way of tracking the best expert alternative to fixed share techniques.

In this paper we develop a universal algorithm for predicting finite- dimensional distributions, i.e., points from a simplex, under Kullback-Leibler loss. Related problems have been considered in the literature. Online convex optimization is a similar area where a decision-maker makes a sequence of decisions from a fixed feasible set. After each point is chosen, it encounters a convex cost functions. In [1] a logarithmic regret bound was obtained for α-convex cost functions, which have a lower bound on the second derivative; these bounds are not applicable here for the lack of such bound. A similar problem was considered in [2], where the authors proposed a general additive algorithm based on gradient descent and derived loss bounds that compare the loss of the resulting online algorithm to the best offline predictor from the relevant model class. They considered a softmax transfer function (Example 4 in [2]) and achieved a theoretical bound with a multiplicative coefficient of two in front of the loss of the best expert; whereas we achieved a multiplicative coefficient of one, which indicates that our theoretical bound is better for large losses.

Our approach is based on the Aggregating Algorithm (AA), which was first introduced in [3]. AA works as follows: we assign initial weights to experts and at each step the weights of experts are updated according to their current performance. The approach is similar to the Bayesian method, where the prediction is the average over all models based on the likelihood of the available data. For the case of finitely many experts, AA gives a guarantee ensuring that the learner's loss is as small as best expert's loss plus a constant.

We formulate AA for the case of discounted loss along the lines of [4]. Discounting allows us to give less importance to older losses, which is an important property for practical applications. Discounted loss in the online prediction framework was analyzed in [5] in the setting where only the losses of the experts are known. In [5] the authors noticed that in the context of prediction with expert advice, the discounted loss provides an alternative to 'tracking the best expert' framework of [6]. Indeed, if the best expert changes after some steps, an algorithm that competes under discounted loss will not take into account small losses of the old best expert because they are strongly discounted, and will switch to track the new best expert. An important special case is exponential discounting where $\alpha_t = \alpha \in (0,1)$, which is widely used in finance, time series analysis and other applications (see [7] for a review of exponential smoothing). Note that for the standard undiscounted loss $\alpha_t = 1$ at each step t and the algorithm coincides with AA.

Interesting results were achieved for the class of problems with an infinite decision pool of experts. Aggregating Algorithm for Regression was proposed in [8] for the case of linear experts under the squared loss. The generalisation for the case of discounted squared loss for linear regression was proposed in [4]. The case of generalised linear regression experts under log-loss was introduced in [9], and the case of the squared loss was considered in [10]. The multidimensional prediction problem was considered in [11], where the authors introduced an algorithm competitive with linear functions under the squared loss. One of the drawbacks of introducing linear experts for probability games is that predictions of linear experts could lie outside a probability simplex. In all the above cases the authors achieved the theoretical bounds which were logarithmic in the number of steps. In a recent paper [12] an algorithm was constructed for the case where outcomes and predictions are distributions on a finite set, the loss function is logarithmic, and competitors are linear functions with softmax applied on top of them. The paper contains an excellent survey of application domains. We propose an algorithm for the similar setting, but improve the regret term in the upper bound on the loss. Our regret bound has a lower growth rate w.r.t. the number of dimensions and does not contain the linear term on the number of steps. Asymptotically in T the regret is still of the order $C \ln T$, but our multiplicative constant C is lower.

In this paper we provide an explicit universal algorithm for predicting probability distributions, which can 'track the best expert' in terms of discounted cumulative Kullback-Leibler loss function. Kullback-Leibler game is one of the most important probability games ([8]). Kullback-Leibler divergence is a measure of discrepancy between two probability distributions ([13]), and it is widely used in different areas such as applied statistics, econometrics, risk management and machine learning. An excellent survey of application of entropy and divergence measures in econometrics can be found in [14]. Useful applications of the entropy and Kullback-Leibler divergence for studying income inequality and welfare economics are described in [15] and [16]. The Kullback-Leibler loss function is also used in optimal portfolio selection and solving portfolio diversification problem ([17]).

We apply AA with Discounting to multinomial logistic regression experts. Multinomial logistic regression predictors are a natural choice for probability games as they output predictions that lie inside a probability simplex. We provide a strategy that 'tracks the best expert' of this type and derive the theoretical bound on the discounted loss of the strategy. We generalise our algorithm to allow it to compete with wider set of experts from Reproducing Kernel Hilbert Space (RKHS) and prove the theoretical guarantee for the kernelized strategy.

Theoretical bounds obtained for Kullback-Leibler game are valid for logarithmic loss game. Indeed, Kullback-Leibler loss function is a generalisation of log-loss function where outcome space is the whole simplex instead of only vertices of the simplex as in the case of log-loss game. Therefore, theoretical bounds obtained for Kullback-Leibler game can be applied to the important problem of probabilistic multi-class classification under logarithmic loss function.

We conduct experiments to compare the performance of our algorithm with multinomial logistic regression. In our experiments we check that the theoretical bound for our algorithm is not violated. Our prediction algorithm is using Markov chain Monte Carlo (MCMC) method in a way which is similar to the algorithm introduced in [10], where AAR was applied to the generalised linear regression class of functions for making a prediction in a fixed interval. MCMC is only a method for evaluating the integral and it can be replaced by a different numerical method. Theoretical convergence of the Metropolis-Hastings method in this case follows from Theorems 1 and 3 in [18]. Estimating the convergence speed is more difficult. With the experiments provided we show that by tuning parameters online, our algorithm moves fast to the area of high values of the probability function and gives a good approximation of the prediction, and theoretical bounds are not violated.

2. Framework

A game of prediction contains three components: a space of outcomes Ω, a decision space Γ, and a loss function $\lambda : \Omega \times \Gamma \to \mathbb{R}$. We consider a probability game on some finite set $\Xi = \{1, \dots, d\}$, where space of outcomes $\Omega = \mathbb{P}(\Xi) = \{(y^{(1)}, \dots, y^{(d)}) : \sum_{i=1}^d y^{(i)} = 1,\ 0 \le y^{(i)} \le 1\}$, decision space $\Gamma = \mathbb{P}(\Xi) = \{(\gamma^{(1)}, \dots, \gamma^{(d)}) : \sum_{i=1}^d \gamma^{(i)} = 1,\ 0 \le \gamma^{(i)} \le 1\}$ are simplices in d-dimensional space, and for any $y \in \Omega$ and $\gamma \in \Gamma$ we define the Kullback-Leibler loss

$$\lambda(y, \gamma) = \sum_{i=1}^{d} y^{(i)} \ln \frac{y^{(i)}}{\gamma^{(i)}}, \tag{1}$$

where $y^{(i)}$ and $\gamma^{(i)}$ are the i-th coordinate of the respective vectors. As in ([13], Section 2.3) we assume that for $p, q > 0$ we have $0 \ln \frac{0}{q} = 0$, $p \ln \frac{p}{0} = +\infty$, and $0 \ln \frac{0}{0} = 0$. The loss function λ defined in this way is continuous in γ and satisfies Assumptions 1–4 from [19].

Learner and experts work according to the following protocol:

Protocol 1.

```
$L_0 := 0$
$L_0^\theta := 0$
for $t = 1, 2, \dots$
    Accountant announces $\alpha_{t-1} \in (0, 1]$
    Nature announces $x_t \in X \subseteq \mathbb{R}^n$
    Experts output $\xi_t(\theta),\ \theta \in \Theta$
    Learner outputs $\gamma_t \in \Gamma \subseteq \mathbb{R}^d$
    Nature announces $y_t \in \Omega \subseteq \mathbb{R}^d$
    $L_t^\theta := \alpha_{t-1} L_{t-1}^\theta + \lambda(y_t, \xi_t(\theta)),\ \theta \in \Theta$
    $L_t := \alpha_{t-1} L_{t-1} + \lambda(y_t, \gamma_t)$
end for
```

The learner in the game of prediction plays against experts θ from some pool Θ, and also accountant and nature. The aim of the learner is to keep his total loss L_t small as compared to the total losses L_t^θ of all experts $\theta \in \Theta$.

In the standard framework of online learning the performance of learners is evaluated by means of cumulative loss. The standard protocol of prediction with expert advice is a special case of Protocol 1 where accountant always announces $\alpha_t = 1$ at each step. In this paper, we consider the generalisation where we discount the previous losses with the discount factor which is announced at each time step.

The cumulative losses of the learner are discounted with a factor $\alpha_t \in (0, 1]$ at each step. If L_{T-1} is the discounted cumulative loss of the learner at step $T-1$, then the discounted cumulative loss of the learner at step T is defined by

$$L_T := \alpha_{T-1} L_{T-1} + \lambda_T(y_T, \gamma_T) = \sum_{t=1}^{T-1} \left(\prod_{j=t}^{T-1} \alpha_j \right) \lambda_t(y_t, \gamma_t) + \lambda_T(y_T, \gamma_T). \quad (2)$$

If L^{θ}_{T-1} is the discounted cumulative loss of the prediction strategy θ at the step $T-1$, then the discounted cumulative loss of the prediction strategy θ at the step T is defined by

$$L^{\theta}_T := \alpha_{T-1} L^{\theta}_{T-1} + \lambda_T(y_T, \xi_T(\theta)) = \sum_{t=1}^{T-1} \left(\prod_{j=t}^{T-1} \alpha_j \right) \lambda_t(y_t, \xi_t(\theta)) + \lambda_T(y_T, \xi_T(\theta)). \quad (3)$$

In the beginning the losses L_0, L^{θ}_0 are initialized to zero. If all the discount factors are the same, i.e. $\alpha_1 = \cdots = \alpha_T = \alpha$, then we have a case of exponential smoothing. At each step the dependence on the loss at the previous steps exponentially decreases: the initial loss is discounted by α^{T-1} at the step T.

We want to find a strategy which is capable of competing in terms of cumulative losses with all prediction strategies which at step t output $\xi_t(\theta) = (\xi^1_t(\theta), \ldots, \xi^d_t(\theta))$:

$$\xi^i_t(\theta) = \sigma_i(\theta, x_t), \ i = 1, \ldots, d, \quad (4)$$

where $\sigma_i(\theta, x_t)$ is multinomial logistic regression function:

$$\sigma_i(\theta, x_t) = \frac{e^{\theta'_i x_t}}{\sum_{j=1}^{d-1} e^{\theta'_j x_t} + 1}, \ i = 1, \ldots, d-1, \quad (5)$$

$$\sigma_d(\theta, x_t) = \frac{1}{\sum_{j=1}^{d-1} e^{\theta_j' x_t} + 1}, \tag{6}$$

and $\theta = (\theta_1', \ldots, \theta_{d-1}')' \in \mathbb{R}^{n(d-1)}$, $\theta_i = (\theta_{i,1}, \ldots, \theta_{i,n})' \in \mathbb{R}^n$.

3. Theoretical Bounds

Theorem 1. *Let $a > 0$. There exists a prediction strategy for Learner such that for every positive integer T, every sequence of outcomes of length T and every sequence $\alpha_t \in (0,1]$, $t = 1, \ldots, T$, and every $\theta \in \mathbb{R}^{n(d-1)}$ the cumulative loss L_T of the Learner satisfies*

$$L_T \le L_T^\theta + a\|\theta\|^2 + \frac{d-1}{2} \ln\det\left(I + \frac{d-1}{8a} X' W_T X\right), \tag{7}$$

where X is the matrix with rows $x_1', \ldots, x_T'$, and $W_T = \operatorname{diag}(w_{1,T}, \ldots, w_{T,T})$, where $w_{t,T} = \prod_{j=t}^{T-1} \alpha_j$. If in addition $\|x_t\|_\infty \le B$ for all t then

$$L_T \le L_T^\theta + a\|\theta\|^2 + \frac{n(d-1)}{2} \ln\left(1 + \frac{d-1}{8a} B^2 \sum_{t=1}^T w_{t,T}\right). \tag{8}$$

Note that for the undiscounted losses we have:

Corollary 1. *Let $a > 0$. There exists a prediction strategy for Learner such that for every positive integer T, every sequence of outcomes of length T, and every $\theta \in \mathbb{R}^{n(d-1)}$ the cumulative loss L_T of Learner satisfies*

$$L_T \le L_T^\theta + a\|\theta\|^2 + \frac{d-1}{2} \ln\det\left(I + \frac{d-1}{8a} \sum_{t=1}^T x_t x_t'\right). \tag{9}$$

If in addition $\|x_t\|_\infty \le B$ for all t then

$$L_T \le L_T^\theta + a\|\theta\|^2 + \frac{n(d-1)}{2} \ln\left(1 + \frac{d-1}{8a} B^2 T\right). \tag{10}$$

4. Aggregating Algorithm

We will use prediction with expert advice to create the strategy for Kullback-Leibler game. In the framework of prediction with expert advice we have access to experts' predictions at each time step and the learner has to make a prediction based on experts' past performances. We use an approach based on the AA. The AA is given a parameter η and an initial distribution on experts $P_0(d\theta)$. After each step t it updates the experts' weights according to their losses:

$$P_t(d\theta) = e^{-\eta\lambda(y_t,\xi_t(\theta))} P_{t-1}(d\theta). \tag{11}$$

The weights of experts which suffer large loss at some step will have a smaller importance for making further predictions.[1]

First, we introduce the Aggregating Pseudo-Algorithm (APA) which at step t outputs *generalised prediction*

$$g_t(y) = -\frac{1}{\eta} \ln \int_\Theta e^{-\eta\lambda(y,\xi_t(\theta))} P^*_{t-1}(d\theta), \tag{12}$$

where $P^*_{t-1}(d\theta)$ are normalized weights:

$$P^*_{t-1}(d\theta) = \frac{P_{t-1}(d\theta)}{P_{t-1}(\Theta)},$$

where Θ is a *parameter space*, i.e. experts $\theta \in \Theta$ can output prediction $\xi_t(\theta) \in \Gamma$ at time t.

The generalised prediction can be seen as a weighted average of the experts' predictions in a way which is similar to the Bayesian method.

The AA is obtained from the APA by replacing each generalised prediction g_t by *a permitted prediction* $\Sigma(g_t)$, where the *substitution function* Σ maps every generalised prediction $g : \Omega \to [0, \infty]$ into a permitted prediction $\Sigma(g) \in \Gamma$ satisfying

$$\forall y : \lambda(y, \Sigma(g)) \le g(y). \tag{13}$$

Let us define $P(\Theta)$ as the set of all probability measures over Θ. If a substitution function satisfying (13) for any distribution $P^*_{t-1}(d\theta) \in P(\Theta)$ exists, we say that the loss function is *η-mixable*. The loss function is *mixable* if it is η-mixable for some $\eta > 0$. The game is called *mixable* if the loss function of it is mixable in the setting of the game.

[1]For $t = 1$ we have $P_{t-1}(d\theta) = P_0(d\theta)$, which is an arbitrary distribution. Thus, any distribution can appear as $P_{t-1}(d\theta)$.

Lemma 1. (Lemma 4 in [8]). *The Kullback–Leibler game is 1-mixable. The AA for the Kullback–Leibler game with learning rate 1 coincides with the Bayesian mixture.*

In Section 7 we will obtain this result by reducing the Kullback-Leibler game to the logarithmic game.

5. Aggregating Algorithm with Discounting

In this section, we formulate AA for the case of discounted loss. It is essentially equivalent to the method in [4]. The Aggregating Algorithm with Discounting (AAD) differs from AA only by the use of the weights in the computation of generalised prediction g_t and the update of the weights.

The pseudocode for AAD is given below

The Aggregating Algorithm with Discounting

```
Initialize prior distribution on experts $P_0(d\theta)$, $\theta \in \Theta$.
Initialize discounted weights of experts $\tilde{P}_0^*(\theta) = \tilde{P}_0(\theta) = 1$.
for $t = 1, 2, \dots$ do
    Get discount $\alpha_{t-1} \in (0, 1]$.
    Get experts' predictions $\xi_t(\theta)$, $\theta \in \Theta$.
    $g_t(y) = -\frac{1}{\eta} \ln \int_{\theta\in\Theta} P_0(d\theta) \left(\tilde{P}_{t-1}^*(\theta)\right)^{\alpha_{t-1}} e^{-\eta\lambda(y,\xi_t(\theta))}$, for all $y \in \Omega$.
    Output $\gamma_t := \Sigma(g_t) \in \Gamma$.
    Update the weights $\tilde{P}_t(\theta) = \left(\tilde{P}_{t-1}(\theta)\right)^{\alpha_{t-1}} e^{-\eta\lambda(y_t,\xi_t(\theta))}$, $\theta \in \Theta$.
    Normalize the weights $\tilde{P}_{t-1}^*(\theta) = \frac{\tilde{P}_{t-1}(\theta)}{\int_{\theta\in\Theta} P_0(d\theta)\tilde{P}_{t-1}(\theta)}$.
end for
```

Lemma 2. *For any learning rate $\eta > 0$, prior P_0 and $T = 1, 2, \dots,$*

$$L_T(AAD(\eta, P_0)) \le -\frac{1}{\eta} \ln \int_\Theta e^{-\eta L_T^\theta} P_0(d\theta). \tag{14}$$

Proof The weights update for AAD is

$$\tilde{P}_t(\theta) = \left(\tilde{P}_{t-1}(\theta)\right)^{\alpha_{t-1}} e^{-\eta\lambda(y_t,\xi_t(\theta))} = e^{-\eta L_t^\theta}. \tag{15}$$

We will prove (14) by induction. At step $t+1$ we can re-write inequality (13) as follows

$$e^{-\eta\lambda(y_{t+1},\gamma_{t+1})} \geq \int_\Theta P_0(d\theta)\left(\tilde{P}_t^*(\theta)\right)^{\alpha_t} e^{-\eta\lambda(y_{t+1},\xi_{t+1}(\theta))}$$
$$= \int_\Theta P_0(d\theta)\frac{e^{-\eta\alpha_t L_t^\theta}}{\left(\int_\Theta P_0(d\theta)e^{-\eta L_t^\theta}\right)^{\alpha_t}} e^{-\eta\lambda(y_{t+1},\xi_{t+1}(\theta))}. \quad (16)$$

Suppose that (14) is true for the step t. By putting the inequality (14) for step t in the power $0 < \alpha_t \leq 1$ we obtain

$$e^{-\eta\alpha_t L_t} \geq \left(\int_\Theta P_0(d\theta)e^{-\eta L_t^\theta}\right)^{\alpha_t}.$$

By putting the last inequality in the denominator of (16) we obtain

$$e^{-\eta\lambda(y_{t+1},\gamma_{t+1})} \geq \frac{\int_\Theta e^{-\eta\lambda(y_{t+1},\xi_{t+1}(\theta))-\eta\alpha_t L_t^\theta}P_0(d\theta)}{e^{-\eta L_t\alpha_t}}.$$

By multiplying by the denominator we have

$$e^{-\eta L_{t+1}} \geq \int_\Theta e^{-\eta L_{t+1}^\theta}P_0(d\theta).$$

By taking a natural logarithm of both parts and multiplying by $-\frac{1}{\eta}$ we obtain (14).

6. Proof of Theoretical Bounds

In this section we provide the proof of Theorem 1. We choose the normal initial distribution of parameters

$$P_0(d\theta) = (a/\pi)^{n(d-1)/2}\exp(-a\|\theta\|^2)d\theta \quad (17)$$

for some $a > 0$.

Applying Lemma 2 for initial distribution (17) and putting $\eta = 1$ from Lemma 1 for Kullback- Leibler loss function we obtain

$$\sum_{t=1}^T w_{t,T}\sum_{i=1}^d y_t^i \ln\frac{y_t^i}{\gamma_t^i} \leq -\ln\left((a/\pi)^{n(d-1)/2}\int_\Theta e^{-J(\theta)}d\theta\right), \quad (18)$$

where

$$w_{t,T} = \prod_{j=t}^{T-1} \alpha_j, \;\; w_{T,T} = 1$$

and

$$J(\theta) := \sum_{t=1}^{T} w_{t,T} \sum_{i=1}^{d} y_t^i \ln \frac{y_t^i}{\sigma_i(\theta, x_t)} + a\|\theta\|^2.$$

We use Taylor expansion (Section 1.7c in [20]) of $J(\theta)$ at the point θ_0 where $\min J(\theta)$ is obtained:

$$J(\theta) = J(\theta_0) + \frac{1}{2}(\theta - \theta_0)' H(\phi)(\theta - \theta_0),$$

where ϕ is a convex combination of θ_0 and θ, and H is the Hessian matrix of $J(\theta)$.

The second partial derivative of $J(\theta)$ by the l-th, j-th components of θ_k and θ_m respectively is expressed as follows:

$$\begin{aligned}\frac{\partial^2 J(\theta)}{\partial\theta_{k,l}\partial\theta_{m,j}} = 2a\delta_l^j \delta_k^m + \sum_{t=1}^{T} w_{t,T} \sum_{i=1}^{d} \Big(& y_t^i \frac{1}{\sigma_i^2(\theta, x_t)} \frac{\partial \sigma_i(\theta, x_t)}{\partial\theta_{k,l}} \frac{\partial \sigma_i(\theta, x_t)}{\partial\theta_{m,j}} \\ & - y_t^i \frac{1}{\sigma_i(\theta, x_t)} \frac{\partial^2 \sigma_i(\theta, x_t)}{\partial\theta_{k,l}\partial\theta_{m,j}} \Big), \end{aligned} \tag{19}$$

where

$$\delta_k^m = \begin{cases} 1, & \text{if } k = m \\ 0, & \text{if } k \neq m \end{cases}$$

is Kronecker delta.

The first and second partial derivatives of the function $\sigma_i(\theta, x_t)$ are as follows:

$$\frac{\partial \sigma_i(\theta, x_t)}{\partial\theta_{k,l}} = x_{t,l}\sigma_i(\theta, x_t)(\delta_i^k - \sigma_k(\theta, x_t)),$$

$$\begin{aligned}\frac{\partial^2 \sigma_i(\theta, x_t)}{\partial\theta_{k,l}\partial\theta_{m,j}} = x_{t,l}x_{t,j}\sigma_i(\theta, x_t)\Big(& \delta_k^m - \delta_k^m \sigma_k(\theta, x_t) - \delta_i^k \sigma_m(\theta, x_t) - \delta_i^m \sigma_k(\theta, x_t) \\ & + 2\sigma_k(\theta, x_t)\sigma_m(\theta, x_t)\Big).\end{aligned}$$

Expression (19) can be re-written as follows:

$$\frac{\partial^2 J(\theta)}{\partial\theta_{k,l}\partial\theta_{m,j}} = 2a\delta_l^j \delta_k^m + \sum_{t=1}^{T} w_{t,T} x_{t,l} x_{t,j} f_{k,m}(\theta, x_t), \tag{20}$$

where

$$f_{k,m}(\theta, x_t) = \sigma_k(\theta, x_t)(\delta_k^m - \sigma_m(\theta, x_t)).$$

We denote $W_T = \mathrm{diag}(w_{1,T}, w_{2,T}, \dots, w_{T,T})$ the diagonal matrix T by T. Let X be the $T \times n$ matrix with the rows $x'_1, \dots, x'_T$ and $\Gamma_{k,m}(\phi)$ be the diagonal $T \times T$ matrix that has $f_{k,m}(\phi, x_1, y_1), \dots, f_{k,m}(\phi, x_T, y_T)$ on the diagonal. Let Z be the block matrix as follows:

$$Z = \begin{pmatrix} X_{1,1} & \dots & X_{1,d-1} \\ \vdots & \ddots & \vdots \\ X_{d-1,1} & \dots & X_{d-1,d-1} \end{pmatrix},$$

where

$$X_{k,m} = \begin{cases} \sqrt{W_T}X, & \text{if } k = m \\ O, & \text{if } k \neq m \end{cases}$$

Let $\Gamma(\phi)$ to be a block matrix as follows:

$$\Gamma(\phi) = \begin{pmatrix} \Gamma_{1,1}(\phi) & \dots & \Gamma_{1,d-1}(\phi) \\ \vdots & \ddots & \vdots \\ \Gamma_{d-1,1}(\phi) & \dots & \Gamma_{d-1,d-1}(\phi) \end{pmatrix}.$$

Then Hessian matrix of $J(\theta)$ can be written in the matrix form:

$$H(\phi) = 2aI + Z'\Gamma(\phi)Z. \tag{21}$$

Since $\Gamma(\phi)$ is a symmetric matrix, we can see (Theorem 21.5.6 in [21]) that:

$$\psi'\Gamma(\phi)\psi \leq \psi'\lambda_{\max}(\Gamma(\phi))\psi,$$

for any $\psi \in \mathbb{R}^{T(d-1)}$ where $\lambda_{\max}(\Gamma(\phi))$ is the supremum over maximum eigenvalues of $\Gamma(\phi)$.

We will now show that matrix $\Gamma(\phi)$ is positive definite. The absolute value of the diagonal element of $\Gamma(\phi)$ is

$$|f_{k,k}(\theta, x_t)| = \sigma_k(\theta, x_t)(1 - \sigma_k(\theta, x_t)),$$

the sum of the absolute values of non-diagonal elements on the row is

$$\sum_{m\neq k} |f_{k,m}(\theta, x_t)| = \sum_{m\neq k} |-\sigma_k(\theta, x_t)\sigma_m(\theta, x_t)|$$
$$= \sigma_k(\theta, x_t) \sum_{m\neq k} \sigma_m(\theta, x_t) = \sigma_k(\theta, x_t)(1 - \sigma_k(\theta, x_t) - \sigma_d(\theta, x_t)).$$

As $|f_{k,k}(\theta, x_t)| > \sum_{m\neq k} |f_{k,m}(\theta, x_t)|$, for all $k = 1, \ldots, d-1$, $t = 1, \ldots, T$, then by Diagonal Dominance Theorem 6.1.10 in ([22]) $\Gamma(\phi)$ is positive definite.

By Theorem 3 (see Appendix), we can upper bound $\lambda_{\max}(\Gamma(\phi))$, the maximum eigenvalue of $\Gamma(\phi)$, by the sum of maximum eigenvalues of diagonal blocks $\Gamma_{i,i}(\phi)$:

$$\lambda_{\max}(\Gamma(\phi)) \leq \sum_{i=1}^{d-1} \lambda_{\max}(\Gamma_{i,i}(\phi)).$$

Since $\Gamma_{i,i}(\phi)$ is diagonal then:

$$\lambda_{\max}(\Gamma_{i,i}(\phi)) = \sup_{\theta\in\mathbb{R}^{n(d-1)},\ x_t\in\mathbb{R}^n} f_{i,i}(\theta, x_t)$$
$$= \sup_{\theta\in\mathbb{R}^{n(d-1)},\ x_t\in\mathbb{R}^n} \sigma_i(\theta, x_t)(1 - \sigma_i(\theta, x_t)) = \frac{1}{4}.$$

By taking $\psi = Z(\theta - \theta_0)$ and using (21):

$$J(\theta) \leq J(\theta_0) + (\theta - \theta_0)'(aI + \frac{d-1}{8} Z'Z)(\theta - \theta_0).$$

We can obtain the lower bound on the integral in (18):

$$\int_\Theta e^{-J(\theta) d\theta} \geq e^{-J(\theta_0)} \int_\Theta e^{-(\theta-\theta_0)'(aI + \frac{d-1}{8} Z'Z)(\theta-\theta_0)} d\theta.$$

The integral in the right-hand side can be calculated analytically (see Section 15.12 in [21]):

$$\int_\Theta e^{-(\theta-\theta_0)'(aI + \frac{d-1}{8} Z'Z)(\theta-\theta_0)} d\theta = \frac{\pi^{\frac{n(d-1)}{2}}}{\sqrt{\det(aI + \frac{d-1}{8} Z'Z)}}$$

After putting this expression in (18) we obtain the upper bound:

$$L_T \leq -\ln\left(e^{-J(\theta_0)}\left(\frac{a}{\pi}\right)^{\frac{n(d-1)}{2}}\pi^{\frac{n(d-1)}{2}}\frac{1}{\sqrt{\det(aI+\frac{d-1}{8}Z'Z)}}\right)$$
$$= J(\theta_0)+\frac{1}{2}\ln\det(I+\frac{d-1}{8a}Z'Z) = L_T^{\theta_0}+a\|\theta_0\|^2$$
$$+\frac{d-1}{2}\ln\det\left(I+\frac{d-1}{8a}X'W_TX\right).$$

If $\|x_t\|_\infty \leq B$ the determinant of a symmetric positive definite matrix is upper bounded by the product of its diagonal elements (see Chapter 2, Theorem 7 in [23]):

$$\det\left(I+\frac{d-1}{8a}X'W_TX\right) \leq \left(1+\frac{d-1}{8a}B^2\sum_{t=1}^{T}w_{t,T}\right)^n.$$

7. Prediction Strategy

In this section we will provide a strategy for calculating predictions for Kullback-Leibler game. First, we will define a log-loss game. A log-loss game of prediction contains three components: a space of outcomes Ω, a decision space Γ, and a loss function $\lambda : \Omega \times \Gamma \to \mathbb{R}$. We consider the problem of probabilistic classification with d classes $\Xi = \{1, \ldots, d\}$. As the outcome space Ω we take d-dimensional unit vectors $e_1, \ldots, e_d$, i.e. the distributions concentrated on the vertices of the simplex. The vectors $e_1, e_2, \ldots, e_d$ form the standard basis in $\mathbb{R}^d$; the vector e_i has one at the i-th position and zeros elsewhere. The decision space $\Gamma = \mathbb{P}(\Xi) = \{(\gamma^{(1)}, \ldots, \gamma^{(d)}) : \sum_{i=1}^d \gamma^{(i)} = 1,\ 0 \leq \gamma^{(i)} \leq 1\}$ is a simplex in d-dimensional space, and for any $y \in \Omega$ we define the logarithmic loss function

$$\lambda(y,\gamma) = -\sum_{i=1}^{d} y^{(i)}\ln\gamma^{(i)},$$

where $y^{(i)}$ and $\gamma^{(i)}$ are the i-th coordinates of the respective vectors.

Kullback-Leibler game is a generalisation of log-loss game where outcome space is the whole simplex instead of only vertices of the simplex as in the case of log-loss game. Therefore, theoretical bounds obtained in Theorem 1 and Corollary 1 are valid for the problem of multi-class classification under logarithmic loss game.

Lemma 3. *Let $\gamma \in \Gamma$ is a permitted prediction for log-loss game, i.e. $\lambda(e_i, \gamma) \le g_T(e_i)$, for $i = 1, \dots, d$. Then γ is a permitted prediction for Kullback-Leibler game, i.e. $\lambda(y, \gamma) \le g_T(y)$ for all $y \in \mathbb{P}(\Xi)$.*

Proof. Let $\gamma \in \Gamma$ be a permitted prediction for log-loss game. From (1) Kullback-Leibler loss function is

$$\lambda(y, \gamma) = \sum_{i=1}^{d} y^{(i)} \ln \frac{y^{(i)}}{\gamma^{(i)}} = \sum_{i=1}^{d} y^{(i)} \ln y^{(i)} - \sum_{i=1}^{d} y^{(i)} \ln \gamma^{(i)}$$
$$= \sum_{i=1}^{d} y^{(i)} \ln y^{(i)} + \sum_{i=1}^{d} y^{(i)} \lambda(e_i, \gamma).$$

Generalised prediction (12) for Kullback-Leibler game is

$$\begin{aligned} g_T(y) &= -\frac{1}{\eta} \ln \int_\Theta e^{-\eta\lambda(y,\gamma)} P^*_{T-1}(d\theta) \\ &= \sum_{i=1}^{d} y^{(i)} \ln y^{(i)} - \frac{1}{\eta} \ln \int_\Theta e^{-\eta \sum_{i=1}^{d} y^{(i)} \lambda(e_i,\gamma)} P^*_{T-1}(d\theta) \\ &= \sum_{i=1}^{d} y^{(i)} \ln y^{(i)} - \frac{1}{\eta} \ln \int_\Theta \prod_{i=1}^{d} e^{-\eta y^{(i)} \lambda(e_i,\gamma)} P^*_{T-1}(d\theta) \\ &\ge \sum_{i=1}^{d} y^{(i)} \ln y^{(i)} - \frac{1}{\eta} \ln \prod_{i=1}^{d} \int_\Theta \left(e^{-\eta \lambda(e_i,\gamma)} P^*_{T-1}(d\theta) \right)^{y^{(i)}} \\ &= \sum_{i=1}^{d} y^{(i)} \ln y^{(i)} + \sum_{i=1}^{d} y^{(i)} \left(-\frac{1}{\eta} \ln \int_\Theta e^{-\eta\lambda(e_i,\gamma)} P^*_{T-1}(d\theta) \right) \\ &= \sum_{i=1}^{d} y^{(i)} \ln y^{(i)} + \sum_{i=1}^{d} y^{(i)} g_T(e_i) \\ &\ge \sum_{i=1}^{d} y^{(i)} \ln y^{(i)} + \sum_{i=1}^{d} y^{(i)} \lambda(e_i, \gamma) = \lambda(y, \gamma). \end{aligned}$$

The first inequality follows from the generalised Hölder inequality (this follows from the version of the inequality in Section 9.3 of [24] by induction). The second inequality follows from the fact that γ is a permitted prediction for log-loss game. We showed that prediction γ satisfies the inequality (13) for any $y \in \mathbb{P}(\Xi)$. Therefore, γ is a permitted prediction for Kullback-Leibler game. $\square$

We calculate generalised prediction for AAD from unnormalised weights (15) and taking initial parameter distribution (17)

$$G_T(e_k) = -\frac{1}{\eta}\ln\int_\Theta P_0(d\theta)\left(\tilde{P}_{T-1}(\theta)\right)^{\alpha_{T-1}} e^{-\eta\lambda(e_k,\xi_T(\theta))}$$
$$= -\frac{1}{\eta}\ln\int_\Theta P_0(d\theta)e^{-\eta L_T(e_k,\xi_T(\theta))}$$
$$= -\frac{1}{\eta}\ln\left(a/\pi\right)^{n(d-1)/2}\int_\Theta (\xi_T^k(\theta))^\eta e^{-\eta\sum_{t=1}^{T-1}(\prod_{j=t}^{T-1}\alpha_j)\sum_{i=1}^d y_j^i\ln\frac{y_j^i}{\xi_j^i(\theta)}-a\|\theta\|^2}d\theta,$$
$$k = 1,\dots,d. \quad (22)$$

Generalised prediction (22) calculated from unnormalized weights will differ from the generalised prediction (12) calculated from normalized weights by only an additive constant.

By putting $\eta = 1$ from Lemma 1 and applying substitution function $e^{-(\cdot)}$ prediction at step T predicts is expressed as follows

$$\gamma_T^k = \int_\Theta \xi_T^k(\theta)q_{T-1}^*(\theta)d\theta,\ k = 1,\dots,d, \quad (23)$$

where

$$q_T^*(\theta) = Cq_T(\theta) = C\exp\Big(-\sum_{t=1}^{T-1}\left(\prod_{j=t}^{T-1}\alpha_j\right)\sum_{i=1}^d y_t^i\ln\frac{y_t^i}{\xi_t^i(\theta)} - a\|\theta\|^2\Big), \quad (24)$$

and C is the normalising constant ensuring that $\int_\Theta q_T^*(\theta)d\theta = 1$.

Integral (23) is a Bayesian mixture, where function $\xi_T^k(\theta)$ needs to be integrated with respect to the normalized distribution $q_T^*(\theta)$. It is possible to avoid the calculation of normalization constant C as it is a computationally inefficient operation, and integrate function $\xi_T^k(\theta)$ from the unnormalized distribution $q_T(\theta)$. In order to calculate the integral (23), it is possible to use MCMC algorithms. MCMC techniques are often applied to solve integration and optimisation problems in large dimensional spaces. MCMC is a strategy for generating samples while exploring the state space using a Markov chain mechanism. This mechanism is constructed so that the chain spends more time in the most important regions. The good introduction of MCMC for Machine Learning can be found in [25].

We will use Metropolis-Hastings algorithm for sampling parameters θ from the posterior distribution $\mathcal{P}$. As a proposal distribution we chose Gaussian distribution $\mathcal{N}(0, \sigma^2)$ with some parameter σ. We start with some initial parameter θ_0 and at each step m we update:

$$\theta^m = \theta^{m-1} + \mathcal{N}(0, \sigma^2),\ m = 1, \dots, M,$$

where M is a maximum number of iterations in MCMC method.

The update parameter θ^m at step m is accepted with probability $\min\left(1, \frac{f_{\mathcal{P}}(\theta^m)}{f_{\mathcal{P}}(\theta^{m-1})}\right)$, where $f_{\mathcal{P}}(\theta)$ is the density function for the distribution $\mathcal{P}$ at point θ. At each step by accepting and rejecting the updates of parameters θ we move closer to the maximum of the density function. At the beginning it is common to use 'burn-in' stage when the integral is not calculated till we will not reach the area of high values of density function $f_{\mathcal{P}}$. Thus, we perform integration only from the area with high density of $\mathcal{P}$. Some values of θ are accepted even when the calculated probability is less than 1, it allows the algorithm to move away from local minimum of the density function. Because we are interested only in the ratio of density functions of generated parameters, we can generate new parameters θ from the unnormalized posterior distribution $q_T(\theta)$ and avoid the weights normalization at each step, which is more computationally efficient.

At time $t = 0$ the algorithm starts with the initial estimate of the parameters $\theta_0 = 0$. At each iteration $t > 0$ we start with parameter θ^M calculated at the previous step $t - 1$. It allows the algorithm to converge faster to the correct location of the main mass of the distribution.

Algorithm

```
**Parameters**: number $M > 0$ of MCMC iterations,
                standard deviation $\sigma > 0$,
                regularization coefficient $a > 0$
$\eta := 1$
initialize $\theta_0^M := 0 \in \Theta$
```

```
define $q_0(\theta) := \exp(-a\eta\|\theta\|^2)$
for $t = 1, 2, \dots$ do
    $\gamma_t^i := 0,\ i = 1, \dots, d$
    read $x_t \in \mathbb{R}^n$
    initialize $\theta_t^0 = \theta_{t-1}^M$
    for $m = 1, 2, \dots, M$ do
        $\theta^* := \theta_t^{m-1} + \mathcal{N}(0, \sigma^2 I)$
        flip a coin with success probability
                $\min\left(1, q_{t-1}(\theta^*)/q_{t-1}(\theta_t^{m-1})\right)$
            if success then
                $\theta_t^m := \theta^*$
            else
                $\theta_t^m := \theta_{t-1}^m$
            end if
        $\gamma_t^i := \gamma_t^i + \eta\xi_t^i(\theta_t^m),\ i = 1, \dots, d$
    end for
    output predictions $\gamma_t^i = \gamma_t^i/M,\ i = 1, \dots, d$
end for
```

8. Kernelized Algorithm

In this section we kernelize the algorithm and prove upper bounds on the Kullback-Leibler loss of the algorithm competing with wider class of experts.

We start with the following lemma restating Theorem 1 in the dual form.

Lemma 4. *Under the conditions of Theorem 1, the cumulative loss L_T of the Learner satisfies*

$$L_T \le L_T^\theta + a\|\theta\|^2 + \frac{d-1}{2} \ln\det\left(I + \frac{d-1}{8a}\sqrt{W_T} X_T' X_T \sqrt{W_T}\right), \tag{25}$$

where $\sqrt{W_T} = \mathrm{diag}(\sqrt{w_{1,T}}, \sqrt{w_{2,T}}, \dots, \sqrt{w_{T,T}})$.

Proof. The lemma follows from (7) and the Sylvester identity (Lemma 7). □

The lemma opens the way for the kernelization of the loss bound along the usual lines, but one should be careful. We do not have an explicit formula for the universal algorithm and cannot state it in the dual form straightforwardly.

We will now define the kernel form of the algorithm. Our starting point is the representation given by (23).

Lemma 5. *Let J be an orthogonal $(n\times n)$-matrix. If all vectors x_t are replaced by Jx_t, $t=1,2,\ldots,T$, the value of γ_t^k given by (23) will not change.*

Proof. Vectors x_t appear in the integral of (23) only in scalar products $x_t'\theta_i$. Let us replace all x_t by Jx_t. We have $(Jx_t)'\theta_i = x_t'(J'\theta_i)$. The substitution $\tilde\theta_i = J'\theta_i$ reduces the integral to the same form as before because $\|\tilde\theta\|^2 = \sum_{i=1}^{d-1}\theta_i'J'J\theta_i = \sum_{i=1}^{d-1}\|\theta_i\|^2 = \|\theta\|^2$ and $|\det(\mathrm{diag}(J,J,\ldots,J))| = 1$. $\square$

Lemma 6. *Let all x_t, $t=1,2,\ldots,T$, belong to an m-dimensional subspace S_m of $\mathbb{R}^n$, $m<n$. Then the integral in (23) can be taken over $\Theta_m = (S_m)^{d-1}$, so that*

$$\gamma_T^k = \int_{\Theta_m} \xi_T^k(\theta)\tilde q_{T-1}^*(\theta)d\theta,$$

with

$$\tilde q_{T-1}^*(\theta) = \tilde C \exp\left(-\sum_{t=1}^{T-1}\left(\prod_{j=t}^{T-1}\alpha_j\right)\sum_{i=1}^{d} y_t^i \ln\frac{y_t^i}{\xi_t^i(\theta)} - a\|\theta\|^2\right), \qquad (26)$$

where $\tilde C$ is such that $\int_{\Theta_m}\tilde q_T^*(\theta) = 1$.

Proof. Lemma 5 implies that without restricting the generality we can assume that vectors in S_m have their last $n-m$ coordinates equal to 0. We can then split θ_i as $\theta_i' = (\tilde\theta_i', \hat\theta_i')$, where $\tilde\theta_i$ has m and $\hat\theta_i$ has $n-m$ coordinates ($i=1,2,\ldots,d-1$), and take $\tilde\theta = (\tilde\theta_1',\ldots,\tilde\theta_{d-1}')'$ and $\hat\theta = (\hat\theta_1',\ldots,\hat\theta_{d-1}')'$. Since $x_t\in S_m$, one can split them as $x_t' = (\tilde x_t', 0)$, where 0 is of dimension $n-m$. We have $x_t'\theta_i = \tilde x_t'\tilde\theta_i$, and therefore $\xi_T^k((\tilde\theta_1',\hat\theta_1',\ldots,\tilde\theta_{d-1}',\hat\theta_{d-1}')') = \xi_T^k((\tilde\theta_1',0,\ldots,\tilde\theta_{d-1}',0)')$.

We have

$$\gamma_T^k = \int_\Theta \xi_T^k(\theta) q_{T-1}^*(\theta) d\theta =$$

$$= \int_{\mathbb{R}^{m(d-1)}} \int_{\mathbb{R}^{(n-m)(d-1)}} \xi_T^k((\tilde{\theta}_1', 0, \ldots, \tilde{\theta}_{d-1}', 0)') \times C\tilde{q}_{T-1}^*(\tilde{\theta}) \exp\left(-a\sum_{i=1}^{d-1} \|\hat{\theta}_i\|\right) d\tilde{\theta} d\hat{\theta}, \quad (27)$$

where C is such that

$$C\int_\Theta \tilde{q}_{T-1}^*(\tilde{\theta}) \exp\left(-a\sum_{i=1}^{d-1} \|\hat{\theta}_i\|\right) d\theta = 1.$$

An application of Fubini's theorem to 27 completes the proof. □

Let $k : \mathbf{X} \times \mathbf{X} \to \mathbb{R}$, where $\mathbf{X}$ is some domain, be a kernel and let $\mathcal{F}$ with the scalar product $\langle\cdot,\cdot\rangle_\mathcal{F}$ and norm $\|\cdot\|_F$ be the corresponding RKHS (see [26] for definitions). Let $\Phi : \mathbf{X} \to \mathcal{F}$ be the feature mapping given by $\Phi(x) = k(x, \cdot)$.

Consider the kernelized modification of Protocol 1 with nature outputting $x_t \in \mathbf{X}$. We want to compete with predictors of the following kind. Take an array of $d-1$ functions $\mathbf{f} = (f_1, f_2, \ldots, f_{d-1}) \in \mathcal{F}^{d-1}$. At step t array $\mathbf{f}$ outputs $\xi_t(\mathbf{f}) = (\xi_t^1(\mathbf{f}), \ldots, \xi_t^d(\mathbf{f}))$ such that

$$\xi_t^i(\mathbf{f}) = \sigma_i(\mathbf{f}, x_t),\ i = 1, \ldots, d, \quad (28)$$

where $\sigma_i(\mathbf{f}, x_t)$ are multinomial logistic regression functions:

$$\sigma_i(\mathbf{f}, x_t) = \frac{e^{f_i(x_t)}}{\sum_{j=1}^{d-1} e^{f_j(x_t)} + 1},\ i = 1, \ldots, d-1, \quad (29)$$

$$\sigma_d(\mathbf{f}, x_t) = \frac{1}{\sum_{j=1}^{d-1} e^{f_j(x_t)} + 1}. \quad (30)$$

The discounted cumulative loss $L_t^\mathbf{f}$ is defined similar to (3).

We will now construct a universal algorithm working according to the kernelized Protocol 1. The algorithm works as follows.

On step T let $\mathcal{F}_T \subseteq \mathcal{F}$ be the span of $\Phi(x_1), \Phi(x_2), \ldots, \Phi(x_T)$. It is a space of finite dimension $T' \leq T$ and it is isomorphic to $\mathbb{R}^{T'}$. We define γ_T^k

by (23) with $\Theta = \mathbb{R}^{T'(d-1)}$ and $x'_t \in \mathbb{R}^{T'}$ being the values corresponding to $\Phi(x_t)$, $t = 1, 2, \ldots, T$. Lemma 5 implies that the algorithm is well-defined and independent of the choice of a linear isomorphism.

The values of γ_T^k can be computed by evaluating the integral in (23) as follows. Each θ in Θ corresponds to a predictor $(\sigma_1(\mathbf{h}, x), \ldots, \sigma_d(\mathbf{h}, x))$, where $\mathbf{h} = (h_1, h_2, \ldots, h_{d-1}) \in \mathcal{F}^{d-1}$ is defined by $h_i(x) = \sum_{t=1}^T a_t^i k(x_t, x)$, where $a_1^i, a_2^i, \ldots, a_T^i \in \mathbb{R}$ are some constants. The density

$$q_T^*(\theta) \propto \exp\left(-\sum_{t=1}^{T-1}\left(\prod_{j=t}^{T-1}\alpha_j\right)\sum_{i=1}^d y_t^i \ln\frac{y_t^i}{\xi_t^i(\mathbf{h})} - a\sum_{i=1}^{d-1}\|\mathbf{h}_i\|_{\mathcal{F}}^2\right), \tag{31}$$

where $\|\mathbf{h}_i\|_{\mathcal{F}}^2 = \sum_{t_1,t_2=1}^T a_{t_1}^i a_{t_2}^i k(x_{t_1}, x_{t_2})$, may be evaluated (up to a multiplicative constant) once we know $a_1^i, a_2^i, \ldots, a_T^i$. Therefore we can use MCMC doing a random walk over the space of coefficients a_t^i, i.e., $\mathbb{R}^{T(d-1)}$.

Theorem 2. *Let $a > 0$. There exists a prediction strategy S for the learner such that for every positive integer T, for every sequence of outcomes of the length T, and every sequence $\alpha_t \in (0, 1]$, $t = 1, \ldots, T$, and any $\mathbf{f} = (f_1, \ldots, f_{d-1}) \in \mathcal{F}^{d-1}$, the loss L_T of the learner satisfies*

$$L_T \le L_T^{\mathbf{f}} + a\sum_{i=1}^{d-1}\|f_i\|_{\mathcal{F}}^2 + \frac{d-1}{2}\ln\det\left(I + \frac{d-1}{8a}\sqrt{W_T}\mathbf{K}_T\sqrt{W_T}\right), \tag{32}$$

where

$$\mathbf{K}_T = \begin{pmatrix} k(x_1, x_2) & \ldots & k(x_1, x_T) \\ \vdots & \ddots & \vdots \\ k(x_T, x_1) & \ldots & k(x_T, x_T) \end{pmatrix},$$

and $W_T = \operatorname{diag}(w_{1,T}, w_{2,T}, \ldots, w_{T,T})$, where $w_{t,T} = \prod_{j=t}^{T-1}\alpha_j$.

Proof. Fix a positive integer T. The distribution γ_t output by our algorithm is constructed using $\mathcal{F}_T$ of dimension T' isomorphic to some $\mathbb{R}^{T'}$. For $t < T$ the construction relies on a different $\mathcal{F}_t$ isomorphic to $\mathbb{R}^{t'}$. However, $\mathcal{F}_t \subseteq \mathcal{F}_T$ and $\mathcal{F}_t$ is isomorphic to a subspace of $\mathbb{R}^{T'}$. Lemmas 5 and 6 imply that γ_t could be calculated by integration over the same $\mathbb{R}^{T'(d-1)}$ with the same $x'_1, \ldots, x'_t$.

Let $f_1, f_2, \dots, f_{d-1} \in \mathcal{F}_T$. Then each is isomorphic to a θ_i with the same norm and the theorem follows from Lemma 4.

Let $f_1, f_2, \dots, f_{d-1} \in \mathcal{F}$ be arbitrary functions from the RKHS. Using the Representer Theorem argument, we can project each f_i on $\mathcal{F}_T$ and write $f_i = f_i^{\|} + f_i^{\perp}$, where $f_i^{\|} \in \mathcal{F}_T$ and $f_i^{\perp}$ is orthogonal to $\mathcal{F}_T$. By the construction of $\mathcal{F}_T$, we have $f_i(x_t) = f_i^{\|}(x_t)$ $i = 1, 2, \dots, d-1$ and $t = 1, 2, \dots, T$, but $\|f_i^{\|}\|_{\mathcal{F}} \le \|f_i\|_{\mathcal{F}}$. Thus the orthogonal component does not affect predictions but increases the norm. The theorem follows. □

Note that for the undiscounted losses we have:

Corollary 2. *Let $a > 0$. There exists a prediction strategy S for the learner such that for every positive integer T, for every sequence of outcomes of the length T, and any $\mathbf{f} = (f_1, \dots, f_{d-1}) \in \mathcal{F}^{d-1}$, the loss L_T of the learner satisfies*

$$L_T \le L_T^{\mathbf{f}} + a \sum_{i=1}^{d-1} \|f_i\|_{\mathcal{F}}^2 + \frac{d-1}{2} \ln \det\left(I + \frac{d-1}{8a} \mathbf{K}_T \right), \tag{33}$$

where

$$\mathbf{K}_T = \begin{pmatrix} k(x_1, x_2) & \dots & k(x_1, x_T) \\ \vdots & \ddots & \vdots \\ k(x_T, x_1) & \dots & k(x_T, x_T) \end{pmatrix}.$$

The order of the regret term in 33 may vary. However, we show that it has the order $O(\sqrt{T})$ in many cases. We will use the notation $c_{\mathcal{F}}^2 = \sup_{z \in X} k(z, z)$ and assume $c_{\mathcal{F}}^2 < \infty$.

Corollary 3. *Under the conditions of Corollary 2 and if the number of steps T is known in advance, the kernelized algorithm with $a = c_{\mathcal{F}}\sqrt{T}$ achieves loss satisfying*

$$L_T \le L_T^{\mathbf{f}} + \left(\sum_{i=1}^{d-1} \|f_i\|_{\mathcal{F}}^2 + \frac{(d-1)^2}{16} \right) c_{\mathcal{F}} \sqrt{T}. \tag{34}$$

Proof. The determinant of a symmetric positive definite matrix is upper bounded by the product of its diagonal elements (see Chapter 2, Theorem 7 in

[23]) and thus

$$\ln\det\left(I+\frac{d-1}{8a}\mathbf{K}_T\right)\leq T\ln\left(1+\frac{(d-1)c_{\mathcal{F}}^2}{8a}\right)\leq T\frac{(d-1)c_{\mathcal{F}}^2}{8a}.$$

If we know the number of steps T in advance, then we can choose a specific value $a=c_{\mathcal{F}}\sqrt{T}$.

In a case when the number of trials is not known in advance, it is still possible to use a suitable initial weights distribution over the parameter a to achieve a similar bound using the AA (see [27]). $\square$

9. Experiments

In this section we apply our algorithm on three data sets and compare its performance with the multinomial logistic regression. For simplicity we apply our algorithm for multi-class classification problems and put $\alpha_t = 1$ for all $t=1,\ldots,T$. We obtained the best parameters of multinomial logistic regression by using function 'multinom' from library 'nnet' in R. [2]

9.1. Synthetic Data Set

We generated the synthetic 'Smiley' data set that consists of two Gaussian eyes, a trapezoid nose and a parabola mouth. The function for generating this data set was taken from R library 'mlbench'. Figure 1 shows the data set which contains 1000 observations with two features and four classes: left eye, right eye, nose, and mouth. We divide our data in a way that each class will have half of its observations in training and test data sets. Figure 2 illustrates the generated training data set. The split of the training and test data set is not random to show that sometimes the training data set does not describe the 'underlying nature' of the data. The training data set is obtained so that there are infinite number of linear classifiers that could classify the training data set correctly.

First, we will run our algorithm and train the multinomial logistic regression on training data set and compare their performance. We run our algorithm for

[2]The code written in R is available at https://github.com/RaisaDZ/LogisticRegression.

the number of MCMC iterations $M = 3000$ and 'burn-in' period $M_0 = 1000$ for different parameters of regularization a and standard deviation σ.

Table 1 shows the total loss of our algorithm on training data set. Low values of losses are achieved with small regularization parameters a and large standard deviation σ. Very small values of σ lead to big losses as the algorithm is not able to reach the area of high values of density function $f_{\mathcal{P}}$.

Table 2 illustrates the acceptance ratio of new sampling parameters of our algorithm. Large values of σ and large values of regularization parameter a result in low acceptance ratios. With large values of σ we move faster to the area of high values of density function while smaller values of σ can lead to more expensive computations as our algorithm would require more iterations to find the optimal parameters. Figure 3 illustrates logarithm of parameters likelihood $q(\theta)$ defined in (24) for $a = 0.001$ and $\sigma = 0.1$ and 1.5. We can see from the graphs that for $\sigma = 1.5$ the algorithm reachs maximum value of log-likelihood quite fast while for $\sigma = 0.1$ it still tries to find maximum value after 3000 iterations. It is important to keep track on the acceptance ratio of the algorithm, as high acceptance ratio means that we move too slowly and need more iterations and larger 'burn-in' period to find the optimal parameters.

Now we want to demonstrate the 'power' of online learning compared to batch learning. We train the multinomial logistic regression on training data set and will compare its performance with our algorithm applied to test data set. We choose parameters of algorithm to be $M = 3000$, 'burn-in' period $M_0 = 1000$, regularization parameter $a = 0.001$ and standard deviation $\sigma = 1.5$. Note, that even we though use the prior knowledge about optimal parameters of our algorithm using results on training data set, we do not actually train our algorithm, and start with initial value $\theta_0 = 0$. Figure 4 shows the difference between cumulative losses of the multinomial logistic regression and our algorithm $L_T^{\theta^*} - L_T$ on test data set, where θ^* was obtained by multinomial logistic regression model on training data set. We can see from the graph that our algorithm needs a little time to train and after a few steps it becomes better than multinomial logistic regression trained on training data set. It is obvious from

Figure 2 that there are infinite number of linear classifiers that could classify data correctly as training data set contains linearly separable classes. Training data set does not describe the 'underlying nature' of the generated data. As a result, retrespectively best model that was trained on training data set does not perform good on test data set.

Now we will train multinomial logistic regression on test data set to find retrospectively the best model with parameters θ^*. Figure 5 shows the difference between cumulative losses of retrospectively best expert θ^* on test data set and the cumulative loss of our algorithm. We also plot the theoretical bound for our algorithm. The initial large gap corresponds to the value $-a\|\theta^*\|^2$, which gives the initial start to Learner on expert θ^*. As time increases, we add an additional value $-\frac{n(d-1)}{2}\ln(1+\frac{d-1}{8a}X^2T)$ to the bound. We can see from the graph that initially the loss difference is decreasing fast which means that loss of our algorithm becomes larger compared to the loss of multinomial logistic regression model. The initial start $-a\|\theta^*\|^2$ gives us some time for training. After the initial training time passes, the difference between cumulative losses becomes smoother and behaves in a similar way with the theoretical bound of our algorithm which is decreasing logarithmically with the number of steps.

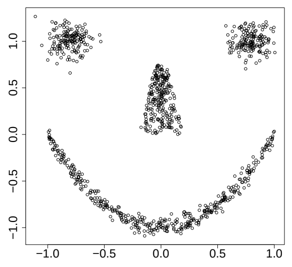

Figure 1: Smiley data set

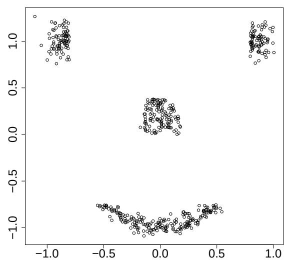

Figure 2: Training data set

9.2. Glass Identification Data Set

We conduct similar experiments on Glass Identification data set which is the part of the library 'mlbench' in R or could be downloaded from UCI Machine

Table 1: Total Losses of our algorithm on training set

$a \setminus \sigma$	0.1	0.5	1.0	1.5	3.0
0.001	218.52	2.11	0.82	0.68	1.10
0.005	220.23	2.31	2.24	2.05	1.44
0.010	219.93	2.93	2.95	3.18	3.89
0.050	207.49	9.99	10.06	10.21	7.65
0.100	226.51	16.72	16.50	22.15	7.30
0.500	214.10	54.74	54.05	71.29	307.79
0.700	207.18	69.12	73.36	65.90	312.76
1.000	222.63	86.55	99.73	79.63	278.83

Table 2: Acceptance ratio of our algorithm on training set

$a \setminus \sigma$	0.1	0.5	1.0	1.5	3.0
0.001	0.82	0.74	0.58	0.38	0.03
0.005	0.81	0.75	0.39	0.10	0.01
0.010	0.81	0.70	0.29	0.05	0.00
0.050	0.82	0.56	0.08	0.01	0.00
0.100	0.80	0.46	0.05	0.01	0.00
0.500	0.80	0.24	0.01	0.01	0.00
0.700	0.82	0.21	0.01	0.01	0.00
1.000	0.82	0.16	0.01	0.00	0.00

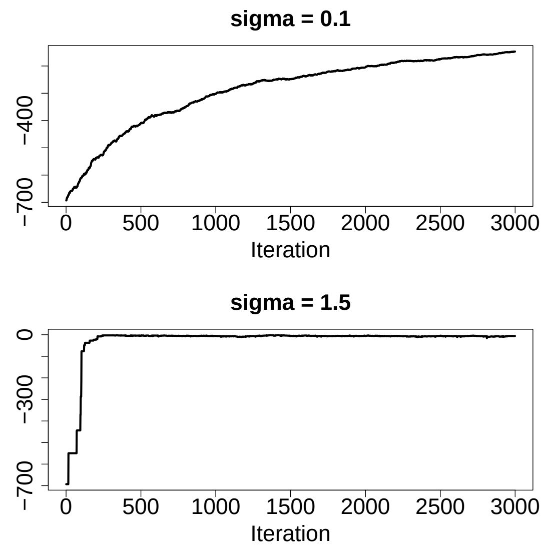

Figure 3: Log-likelihood of parameters depending on iteration step

Learning Repository. The goal is to classify the six type of glasses. The study of classification of types of glass was motivated by criminological investigation. At the scene of the crime, the glass left can be used as evidence. The data set contains nine features and total 214 observations. As there were no timestamps in the data sets, observations were randomly shuffled, and this order was used as a time. We normalise all the features between -1 and 1 and add addition bias 1 to all observations.

Similar to the previous experiment, we find retrospectively the best multinomial logistic regression with parameters θ^* using the whole data set. We want to compare the performance of retrospectively best expert θ^* with the performance of our algorithm.

Now we will show how the performance of our algorithm and the behavior of the loss bound depend on the different parameters of regularization a. We choose number of steps $M = 3000$, 'burn-in' period $M_0 = 1000$ and $\sigma = 0.1$. First, we run our algorithm for small regularization $a = 0.001$. Figure 6 shows the difference between cumulative losses of multinomial logistic regression and

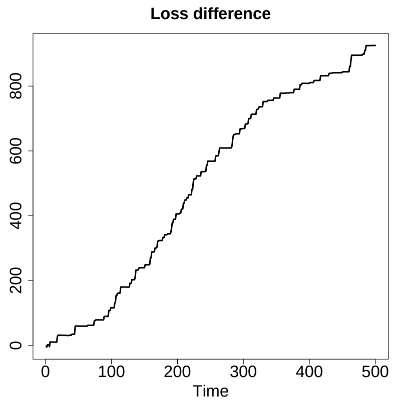

Figure 4: Comparison with logistic regression trained on training set

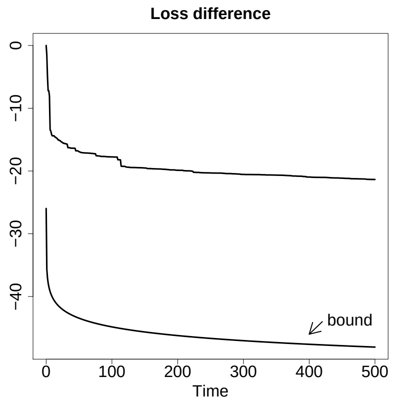

Figure 5: Comparison with retrospectively best logistic regression

our algorithm. Small values of regularization gives small start on the initial parameters $-a\|\theta^*\|^2$ at time $t = 0$. However, the theoretical bound will grow faster with time $-\frac{n(d-1)}{2}\ln(1 + \frac{d-1}{8a}X^2T)$ as it is inversely proportional to the logarithm of the regularizalion parameter a.

We will conduct the second experiment for larger regularization $a = 0.01$. Figure 7 shows the difference between cumulative losses of logistic regression and our algorithm $L_T^{\theta^*} - L_T$. For larger regularization we allow larger initial start on the parameters $-a\|\theta^*\|^2$. Hovewer, the theoretical bound decreases slower with time compared to the previous experiment.

The choice of the regularization parameter a is important as it affects the behaviour of the theoretical bound of our algorithm. Larger parameters of regularization gives larger start on the parameters of the best model, however the theoretical bound will have smaller growth rate as time increases.

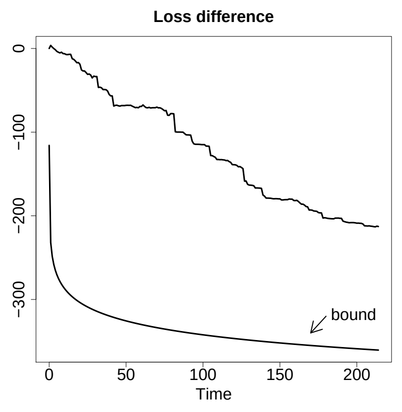

Figure 6: Glass data set, $a = 0.001$

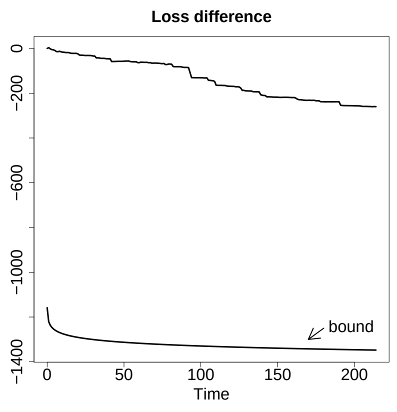

Figure 7: Glass data set, $a = 0.01$

9.3. Football Data Set

The third data set was compiled from historical information on football matches and bookmakers odds [3]. The data set covers three seasons, 2014/2015, 2015/2016 and 2016/2017 of the English Premier League and total 1140 matches. Each match can have three outcomes: 'home win', 'draw', or 'away win'. The data contains the historical information such as total number of goals, shots, corners, yellow and red cards after half-time and full-time and bookmakers' odds from different providers. For each team we generated features such as average number of games won / lost, average number of goals scored / conceded, average number of shots during the first-half, etc. In addition, we combined the odds of different bookmakers provided for the current match. There were total 46 generated features. The first two seasons were used for the training of multinomial logistic regression and the last season was left for test. We want to check if our algorithm could perform close to the model of logistic regression that will be trained in online mode. We choose the parameters of our algorithm $M = 2000$, 'burn-in' period $M_0 = 500$, regularization parameter $a = 0.05$ and standard deviation $\sigma = 0.2$. At the initial step multinomial logistic regression uses the

[3]Available at http://football-Data.co.uk

parameters of the model that was trained on the first two seasons. After that, we add data sequentially and re-train the model after each match. Figure 8 illustrates the difference between cumulative losses of multinomial logistic regression trained online and our algorithm $L_T^{\theta^*} - L_T$. Initially our algorithm performes much worse than logistic regression in online mode as the difference of cumulative losses decreases fast. However, after around 200 steps the difference of cumulative losses stabilizes and becoming more 'flattened' which indicates that the performance of our algorithm becomes close to the performance of logistic regression.

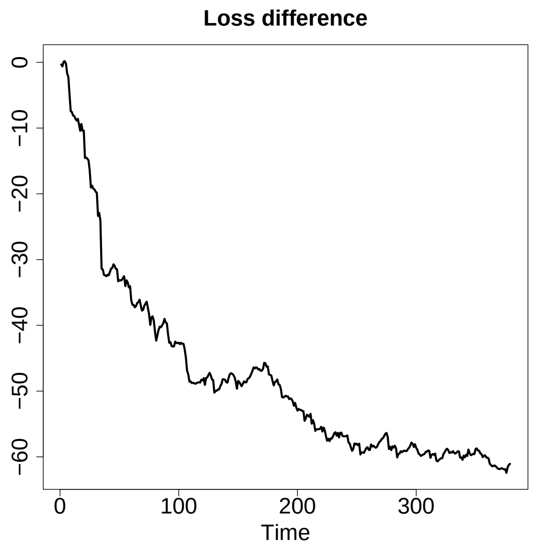

Figure 8: Comparison with multinomial logistic regression trained in online mode

9.4. Conclusions

We carry out the experiments on artificial data set to evaluate the performance of our algorithm. Results show that our algorithm could outperform the model of retrospectively best model of multinomial logistic regression trained on training data set. We also compare the difference between the cumulative losses of retrospectively best multinomial logistic regression trained on the test data set and our algorithm, and we check that the theoretical bound of our algorithm is not violated.

One of the disadvantages of our algorithm is that it might perform much worse with non-optimal input parameters of regularization a and standard deviation σ. If no prior knowledge is available, one can start with some reasonable values of input parameters and keep track on the acceptance ratio of new generated θ. If the acceptance ratio is too high it might indicate that the algorithm moves too slowly to the area of high values of the probability function of θ, and standard deviation σ should be increased. Another option is to take very large number of steps and larger 'burn-in' period. The choice of the regularization parameter a is important as it affects the behaviour of the theoretical bound of our algorithm. Larger parameters of regularization gives larger start on the parameters θ^* of the best model, however the theoretical bound will have smaller growth rate as time increases. The choice of the regularization parameter depends on the particular task and goals that desired to be achieved. Another disadvantage of our algorithm against the competitors is in its training speed. Increasing the training speed of our algorithm is an interesting area of future research.

References

[1] E. Hazan, Introduction to Online Convex Optimization, nowpublishers.com, 2016.

[2] J. Kivinen, M. Warmuth, Relative loss bounds for multidimensional regression problems, Advances in Neural Information Processing Systems 17 (2001) 301–329.

[3] V. Vovk, Aggregating strategies, in: Proceedings of the 3rd Annual Workshop on Computational Learning Theory, Morgan Kaufmann, San Mateo, CA, 1990, pp. 371–383.

[4] A. Chernov, F. Zhdanov, Prediction with expert advice under discounted

loss, in: Proccedings of ALT 2010, Vol. LNAI 6331, Springer, 2010, pp. 255–269, see also arXiv:1005.1918 [cs.LG].

[5] Y. Freund, D. Hsu, A new hedging algorithm and its application to inferring latent random variables, Technical report, arXiv:0806.4802 [cs.GT], arXiv.org e-Print archive.

[6] M. Herbster, M. K. Warmuth, Tracking the best expert, Machine Learning 32 (1998) 151–178.

[7] E. S. Gardner, Exponential smoothing: The state of the art – part II, International Journal of Forecasting (2006) 22:637–66.

[8] V. Vovk, Competitive on-line statistics, International Statistical Review 69 (2) (2001) 213–248.

[9] S. M. Kakade, A. Y. Ng, Online bounds for bayesian algorithms, Advances in Neural Information Processing Systems 17 (2005) 641–648.

[10] F. Zhdanov, V. Vovk, Competitive online generalized linear regression under square loss, in: European Conference on Machine Learning and Principles and Practice of Knowledge Discovery in Databases, 2010, pp. 531–546.

[11] F. Zhdanov, Y. Kalnishkan, Linear probability forecasting, in: Artificial Intelligence Applications and Innovations, AIAI-2010, Proceedings, Vol. 339 of IFIP Advances in ICT, Springer, 2010, pp. 4–11.

[12] D. G. Foster, S. Kale, H. Luo, M. Mohri, K. Sridharan, Logistic regression: The importance of being improper, Proceedings of Machine Learning Research vol, 75 (2018) 75: 1–42.

[13] T. M. Cover, J. A. Thomas, Elements of information theory, John Wiley and Sons, Inc., Second Edition, 2006.

[14] A. Ullah, Entropy, divergence and distance measures with econometric applications, Journal of Statistical Planning and Inference, Elsevier, 1996.

[15] H. Theil, Economics and information theory, 1967.

[16] E. Maasoumi, The measurement and decomposition of multidimensional inequality, Econometrica, 1986 54:991–997.

[17] A. K. Bera, S. Y. Park, Optimal portfolio diversification using the maximum entropy principle, Econometric Reviews, 2008.

[18] G. O. Roberts, A. F. M. Smith, Simple conditions for the convergence of the Gibbs sample and Metropolis-Hastings algorithms, Stoch. Processes Appl., 49, 1993.

[19] V. Vovk, A game of prediction with expert advice, Journal of Computer and System Sciences 56 (1998) 153–173.

[20] R. Courant, F. John, Introduction to Calculus and Analysis, Springer, New York, 1989.

[21] D. A. Harville, Matrix Algebra From a Statistician's Perspective, Springer, 1997.

[22] R. A. Horn, C. R. Johnson, Matrix analysis, Cambridge University Press, 1985.

[23] E. F. Beckenbach, R. Bellman, Inequalities, Springer, Berlin, 1961.

[24] M. Loève, Probability Theory I, 4th Edition, Springer, 1977.

[25] C. Andrieu, N. de Freitas, A. Doucet, M. I. Jordan, An introduction to MCMC for machine learning, Machine Learning Journal, 50.

[26] B. Schölkopf, A. J. Smola, Learning with kernels: Support vector machines, regularization, optimization, and beyond, MIT Press, Cambridge, MA, USA.

[27] V. Vovk, On-line regression competitive with reproducing kernel hilbert spaces, Technical report, arXiv:cs/0511058 [cs.LG].

Appendix

Theorem 3. *Let A is positive symmetric semidefinite block matrix such as*

$$A = \begin{pmatrix} A_{1,1} & A_{1,2} & \dots & A_{1,d} \\ A_{2,1} & A_{2,2} & \dots & A_{2,d} \\ \vdots & \vdots & \ddots & \vdots \\ A_{d,1} & A_{d,2} & \dots & A_{d,d} \end{pmatrix},$$

where $A_{i,i},\ i = 1, \dots, d$ are square matrices. Then $A_{i,i},\ i = 1, \dots, d$ are positive semidefinite and $\lambda_{\max}(A) \le \sum_{i=1}^{d} \lambda_{\max}(A_{i,i})$.

Proof. Let A be an $n \times n$-matrix and $A_{i,i}$ be an $n_i \times n_i$-matrix, $i = 1, \dots, d$. Every vector $x \in \mathbb{R}^n$, $\|x\| = 1$ can be partitioned as $x = (x_1', \dots, x_d')'$, where $x_i \in \mathbb{R}^{n_i},\ i = 1, \dots, d$. Define c_i and u_i by $x_i = \|x_i\| \cdot \frac{x_i}{\|x_i\|} = c_i u_i$, where $\sum_{i=1}^{d} c_i^2 = \sum_{i=1}^{d} \|x_i\|^2 = 1$, and $\|u_i\| = \frac{x_i}{\|x_i\|} = 1$ for $i = 1, \dots, d$. If $x_i = 0$ we put $c_i = 0$ and u_i be any vector such that $\|u_i\| = 1$ for $i = 1, \dots, d$.

We have

$$\lambda_{\max}(A) = \max_{\|x\|=1} x'Ax$$

and

$$x'Ax = \sum_{i,j: x_i, x_j \neq 0} x_i' A_{i,j} x_j = \sum_{i,j} c_i' u_i' A_{i,j} u_j c_j = \sum_{i,j} c_i' \tilde{a}_{i,j} c_j, \tag{35}$$

where $\tilde{a}_{i,j} = u_i' A_{i,j} u_j$ and

$$\tilde{A} = \begin{pmatrix} \tilde{a}_{1,1} & \dots & \tilde{a}_{1,d} \\ \vdots & \ddots & \vdots \\ \tilde{a}_{d,1} & \dots & \tilde{a}_{d,d} \end{pmatrix}.$$

Matrices $A_{i,i},\ i = 1, \dots, d$ are positive semidefinite (by Observation 7.1.2 in [22]) and $\tilde{A}$ is positive semidefinite by (35). Then

$$\sum_{i,j} c_i' \tilde{a}_{i,j} c_j \le \lambda_{\max}(\tilde{A}) \le \operatorname{tr} \tilde{A} = \sum_{i=1}^{d} \tilde{a}_{i,i} = \sum_{i=1}^{d} u_i' A_{i,i} u_i \le \sum_{i=1}^{d} \lambda_{\max}(A_{i,i}).$$

□

Lemma 7. (Sylvester Identity) *For any $n \times m$ matrix B, any $m \times n$ matrix C, and any number a*

$$\det(aI_n + BC) = \det(aI_m + CB),$$

where I_n, I_m are unit matrices $n \times n$ and $m \times m$, respectively.

Proof. *It follows from matrix multiplication rules that*

$$\begin{pmatrix} I_n & B \\ O & I_m \end{pmatrix} \begin{pmatrix} aI_n + BC & O \\ -C & aI_m \end{pmatrix} = \begin{pmatrix} aI_n & aB \\ -C & aI_m \end{pmatrix}$$
$$= \begin{pmatrix} aI_n & O \\ -C & aI_m + CB \end{pmatrix} \begin{pmatrix} I_n & B \\ O & I_m \end{pmatrix}.$$

Taking the determinant of both sides and using rules of taking the determinant of block matrices we get the statement of the lemma. □