# DATA 37200 HW2, Winter 2026

## Problem 1

Consider the ("non-parametric") regression model $Y_i = f^*(X_i) + \xi_i$ for $i = 1, \ldots, n$, where $\xi_i \sim \mathcal{N}(0, \sigma^2)$ are independent noise terms and $X_i \in \mathcal{X}$. We wish to estimate the unknown regression function $f^* \in \mathcal{F}$ using the Multiplicative Weights (MW) algorithm on a finite discretization of $\mathcal{F}$.

For any two functions $f, g$, we measure their distance on the data points $x_1, \ldots, x_n$ using the scaled Euclidean norm of the vector of differences:

$$\|f - g\|_n = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(f(x_i) - g(x_i))^2}$$

Let $\mathcal{F}_\epsilon$ be a finite set of functions such that for any $f \in \mathcal{F}$, there is a representative $f_j \in \mathcal{F}_\epsilon$ satisfying $\|f - f_j\|_n \leq \epsilon$. The size of this set is denoted by the covering number $N(\epsilon)$.

Assume we run the Multiplicative Weights/exponentiated gradient forecaster with the squared loss (which we covered in class) on the experts in $\mathcal{F}_\epsilon$.

(a) Starting from the standard regret bound for Multiplicative Weights with squared loss, show that the expected Mean Squared Error (MSE) of the average estimator $\bar{f}_n$ is bounded by:

$$E[\|\bar{f}_n - f^*\|_n^2] \leq \epsilon^2 + C\frac{\sigma^2 \log N(\epsilon)}{n}$$

where $C$ is a constant. In English, briefly explain the role of each term on the right-hand side: in particular, identify which term corresponds to the cost of "approximating" by a function in the net.

(b) Suppose the function class $\mathcal{F}$ has a metric entropy that grows at the rate $\log N(\epsilon) \asymp \epsilon^{-p}$ for some complexity parameter $p > 0$ (as $\epsilon \to 0$). Determine the optimal choice of the grid size $\epsilon_n$ as a function of the sample size $n$ that minimizes the upper bound derived in (a).

(c) Substitute your optimal $\epsilon_n$ back into the risk bound to derive the convergence rate for the estimator in terms of $n$ and $p$. (Assuming you do this correctly, the result you

get is close to minimax due to the Yang-Barron theorem, which you can read about if interested).

(d) Consider the specific case where $\mathcal{F}$ is the class of 1-Lipschitz functions on the interval $[0, 1]$. As you can check in Vershynin's *High-Dimensional Probability* (Exercise 8.9, you can take this fact as given), the metric entropy for this class scales as $\log N(\epsilon) \asymp \frac{1}{\epsilon}$. Using your result from part (c), what is the optimal convergence rate for learning 1-Lipschitz functions in 1D?

## Problem 2

In high-dimensional spaces, random vectors tend to be nearly orthogonal. We will quantify this using the hypercube.

Let $d$ be the dimension. Consider $N$ vectors $v_1, \ldots, v_N$ chosen independently and uniformly at random from the hypercube $\{\pm 1\}^d$. Recall **Hoeffding's Inequality**: Let $Z_1, \ldots, Z_d$ be independent random variables with $a_k \leq Z_k \leq b_k$. Then for $S_d = \sum_{k=1}^{d} Z_k$:

$$P(|S_d - E[S_d]| \geq t) \leq 2 \exp\left(-\frac{2t^2}{\sum_{k=1}^{d}(b_k - a_k)^2}\right)$$

(a) Fix two distinct indices $i \neq j$. Let $X = \langle v_i, v_j \rangle$. Show that $E[X] = 0$ and use Hoeffding's inequality to bound the probability $P(|\langle v_i, v_j \rangle| \geq \epsilon d)$ for some $\epsilon > 0$.

(b) Using the Union Bound and the specific probability bound you derived in part (a), determine the largest possible number of vectors $N$ (as a function of $d$ and $\epsilon$) such that we can guarantee that with probability at least $1 - \delta$, **all** pairs of vectors satisfy $|\langle v_i, v_j \rangle| < \epsilon d$.

(c) By linear algebra, the size of the largest exact orthogonal set in $\mathbb{R}^d$ is $d$. Why does the result you found in (b) not contradict this fact?

## Problem 3

In this problem, you will run a forecaster on a simple review dataset, where the vectors input to the forecaster are in 384 dimensions, given by the output of a BERT-style language model. The inputs come from rotten tomatoes and amazon reviews, and the forecasters goal is to predict whether the review is negative (0) or positive (1), with error measured in squared loss. Because this is an online setting, the reviews come in one-by-one and after the prediction is made, the true rating is revealed immediately afterward.

- The link to the code is below. The code for the forecaster is incomplete (see comments "YOUR CODE HERE"). Complete the code by implementing an online ridge forecaster. Do not change the dataset or random seed for shuffling.
- Report the "TOTAL AVG LOSS" for at least 3 different values of the ridge regularization parameter. (Ideally, make them fairly different so you can see how the regularization changes the performance.) With the dataset and seed as in the colab below, you should be able to get "TOTAL AVG LOSS" below 0.20.

Here is the link to a google colab:
`https://colab.research.google.com/drive/1nOxwAVJs6nwrFXEsErckrYjw1L32ODes?usp=sharing`

Because the focus of this class is not on coding, feel free to use AI assistance for the implementation. The main goal for this exercise is for you to see how the concepts in class reflect in closer to a real-life setting, so I encourage you to experiment further with this. The code should run on any personal computer fine, but you can use colab if you do not want to set python etc up locally.