Deadline: Fr. June 14, 14:00 Drop your printed or legible handwritten submissions into the boxes at Samelsonplatz, or a `.pdf` file via LearnWeb.

1 Mixture of Experts (Theory) (8+2 points)

In this exercise, we want to fit a mixture of experts

$$p(y \mid x) = \sum_c p(y \mid x, c) p(c \mid x) \tag{1}$$

using Linear Regression models and softmax gating:

$$p(y \mid x, c) = \mathcal{N}(y \mid \beta_c^T x, \sigma^2) = \frac{\exp\left(-\frac{(y-\beta_c^T x)^2}{2\sigma^2}\right)}{\sqrt{2\pi\sigma^2}} \qquad p(c \mid x) = s_c(x) = \frac{\exp\left(\gamma_c^T x\right)}{\sum_{c'} \exp\left(\gamma_{c'}^T x\right)}$$

We derive the necessary formulas to perform the EM-algorithm.
In the E-step, we need to compute the weights

$$w_{n,c} = p(c \mid y_n, x_n) \tag{2}$$

In the M-step we need to minimize the *expected negative complete data log likelihood*

$$\mathscr{L} = -\log \ell = -\sum_{n=1}^{N} \sum_{c=1}^{C} w_{n,c} \left(\log p(y_n | x_n, c) + \log p(c | x_n) \right) \tag{3}$$

with respect to the parameters $(\beta, \gamma, \sigma^2)$ of our model.

A. [2p] Show that the weight update is given by

$$w_{n,c} = \frac{\exp\left(\gamma_c^T x_n - \frac{1}{2\sigma^2}(y_n - \beta_c^T x_n)^2\right)}{\sum_{c'} \exp\left(\gamma_{c'}^T x_n - \frac{1}{2\sigma^2}(y_n - \beta_{c'}^T x_n)^2\right)} \tag{4}$$

Hint: Bayes Theorem

B. [2p] Show that the optimal β_c satisfies the normal equation

$$(X^T W_c X)\beta_c = X^T W_c y \tag{5}$$

where $W_c = \operatorname{diag}(w_c)$, $w_c = (w_{n,c})_n$. I.e. the optimal choice of parameters for the c-th expert is precisely given by finding the optimal parameters w.r.t. the weighted $L2$ loss induced by the gating mechanism.

C. [2p] Show that the optimal σ^2 is given by

$$\sigma^2 = \sum_{c=1}^{C} \frac{1}{N} \|w_c \odot (y - X\beta_c)\|_2^2 \tag{6}$$

D. [2p] Show that the Gradient of $\mathscr{L}$ w.r.t. γ_c is given by

$$\nabla_{\gamma_c} \mathscr{L} = X^T (w_c - s_c(X)) \tag{7}$$

Hint: Use the Lemma $\frac{\partial}{\partial \gamma_i} s_j(x) = (\delta_{ij} - s_i(x)) s_j(x) \mathbf{x}$

E. (Bonus) [2p] Show that the Hessian of $\mathscr{L}$ w.r.t. γ_c is given by

$$\nabla^2_{\gamma_c}\mathscr{L} = -X^T S X \tag{8}$$

where $S = \operatorname{diag}(s_c(X) \odot (1 - s_c(x)))$.
Hint: This formula should remind you of the Newton update for Logistic Regression.

2 Mixtures of Experts (Applied) (8 points)

A. [4p] Implement the EM-algorithm using the formulas from part 1. Use Newton-steps to update the γ parameters.

B. [2p] Fit a mixture of two linear regression experts to the dataset `heights1.dat`. Provide the learned parameters of the experts and the gating function. Plot the weighted average of the expert models $\hat{y}(x) = \mathbb{E}[p(y \mid x)] = \sum_c \hat{y}_c(x) s_c(x)$

C. [2p] The dataset `heights2.dat` contains age and median height of both male and female subjects. Unfortunately the indicator variable is missing. Predict the median height of a (wo-)man at age 21 by fitting a mixture of 3 linear regression experts to the data and checking what the different experts individually predict. Provide the learned parameters of the experts and the gating function.

Hints:

- To avoid singular matrices, use L^2-regularization, i.e. add λI to $X^T W_c X$ and $X^T S X$
- A lot of the involved formulas can be vectorized using Einstein Summation. For example all β_c can be computed simultaneously via

```
XWX = np.einsum('ni, cn, nj -> cij', X, W, X)  # CxMxM tensor
XWy = np.einsum('ni, cn,  n -> ci' , X, W, Y)  # CxM tensor
Beta = solve(XWX+lam*np.eye(m), XWy)  # CxM tensor
```

- It is theoretically guaranteed that $\mathscr{L}$ decreases after each iteration. If if doesn't then there is a bug in your code!
- Your algorithm may end up converging to a sub-optimal local minimum. Tweak the initialization in this case and try multiple restarts. If nothing works, initialize with an educated guess!
- If the Newton Method does not converge reduce the learn-rate. If it converges too slow one can speed up by using the Armijo rule for step size selection.

3 Variable Dependence (6 points)

Given data $(x, y)_{1:N}$ generated from $y = f(x, z) + \epsilon$ with $\epsilon \overset{\text{iid}}{\sim} \mathcal{N}(0, \sigma)$, with features $x \in \mathbb{R}^n$ and $z \in \mathbb{R}$. Consider the partial dependence plot (pdp) w.r.t. z, i.e. the function $g(z) = \frac{1}{N} \sum_{i=1}^{N} f(x_i, z)$.

A. [3p] Show that:

1. If f is independent of z, then g is constant
2. If f depends linearly on z, then g is linear
3. If f depends non-linearly on z, then g is not necessarily non-linear

B. [3p] On the converse, what can we conclude about f's dependence on z, if g is

1. constant
2. linear (but not constant)
3. non-linear

Note: All the above statements should be understood in an approximate sense, i.e. when say g is linear we mean that it is approximately linear, up to some small error.