Exam of
Foundational Principles of Machine Learning (FPML)
December 16, 2022 – 3h

DON'T RETURN THIS SHEET BEFORE YOU ARE ALLOWED TO DO SO!
(you can write your name on the blank papers in the meantime)

General advice:

- **Authorized documents: 6 pages of personal notes.**
- Do not hesitate to do the exercises in any order you like: start with the ones you feel are quick to deal with.
- When you are allowed to start, before you start the first exercise, go through the subject quickly. In each exercise, the most difficult question is not necessarily the last, please feel free to skip some questions. Don't hesitate to go and scrap off points where they are easy to take. (vous pouvez "aller grapiller les points")
- The grading points (scale) is indicative, if the exam is too long a correction factor will be applied. So don't panic in front of the length, what you do, do it right! Also, you may notice that points sum up to 21 (5+5.5+7.5+4) instead of 20, so, I will do *something*.
- **French:** Vous êtes autorisés à composer en Français. (Y compris en insérant des mots techniques comme overfitting ou regularization en anglais quand vous ne savez pas la traduction).
- **French:** si certains bouts de l'énoncé ne sont pas clairs, je peux les traduire ! N'hésitez pas à demander si vous n'êtes pas sûrs.
- Calculators not allowed (and useless). No electronic device allowed (cell phone, etc).
- At the end, we will collect your papers. You can leave after you have returned your paper.

DON'T RETURN THIS SHEET BEFORE YOU ARE ALLOWED TO DO SO!
(you can write your name on the copies in the meantime)

1 Lecture related and independent questions (5 points)

1. (0.25 pt) Does PCA takes into account the labels of data points?
   CORRECTION No, PCA does not use the labels, it is a fully unsupervised method.

2. (0.25 pt) What does SVM stand for? (2 answers accepted)
   CORRECTION SVM: Support Vector Machines. Also: Separators with Vast Margin (unofficial, but quite clear).

3. (0.5 pt) Let's assume we have data and a model we trained. How can we estimate whether more data would be helpful, when we do not yet have this additional data? **Better rephrasing for this question, which was poorly written, sorry (resulting in few people getting it right): *Let's assume we have finite amound of data and a given model in mind, that we can esily train on the data. What experiment can we do to estimate whether more data would be helpful?***
   CORRECTION We can plot the learning curve, i.e. performance (train and validation, typically) as a function of training set size (ideally, keeping the data present in the validation set constant.

4. (0.5 pt) In PCA, what is maximized, or minimized? (2 answers accepted). Be precise in your answer with words, or, write the answer mathematically.
   CORRECTION PCA maximizes the variance after projection on the PCA components (i.e. projection on a smaller subspace). Other possible answer: it minimizes the (Mean Squared) error of reconstruction, i.e. the sum of squares of reconstructed features (after compression+decompression).

5. (0.5 pt) What are the benefits or uses of cross-validation? (give two)
   CORRECTION Cross-Validation (CV) allows to better estimate the error, because if we have e.g. $K = 5$ splits, we can take the average or median value, which is more reliable than a random pick of one of the 5 values. A second advantage is that it allows to compute a confidence interval or uncertainty estimate. When comparing hyperparameter choices, one should plot both the median or average value, AND the error bar associated to our uncertainty (taking the min and max values, or the standard deviation, over K values). This allows to assess whether variations in performance are statistically significant, or just noise from the train sampling.
   Here I did not give points for answers about CV allowing to fine-tune the hyper-parameters, because it can be done with simple trainval split. The point was to say why CV is better than a single train/val split, not re-explain hyper-param tuning.

6. (1 pt) When you have overfitting, what are the things you can do? (assuming we keep the same family of models). Cite as many possible solutions as you know, and each time, quickly explain your choice (1-2 lines per "solution" to overfitting).
   CORRECTION The things we can do to fight overfitting are:

   - *If possible, increase training set size (usually not possible)*
   - *Add some kind of regularization: L2 or L1, prossibly dropout, etc*
   - *Simplify the very architecture of the model: less layers in an MLP, less depth in a decision tree, etc*
   - *Compress the input data, using PCA or an other (un)supervised algorithm*

7. (1 pt) Explain the idea of the SVM. Introduce the proper key concepts and explain what the SVM models maximize.
   CORRECTION This is a lecture question. The key ideas expected were margin definition, maximization of that margin, what are support vectors.

8. (0.5 pt) In ML, many learning algorithms consist essentially in Gradient Descent, a rather simple minimization algorithm. Explain with words why this is not such a bad idea, referring to the idea of train and test sets (we can omit hyper-parameters and the validation set in this explanation, for simplicity).
   CORRECTION Here I expected an explanation about how global minima (not reached by GD) are empirically observed to badly generalize, whereas flat-ish local minima, easily accessed by GD, are actually better at generalization. An algorithm that would be too good at finding deep minima would rely too much on the specifics of the train set, and is typically found to not generealize well.

9. (0.5 pt) Formalize the previous answer in terms of argmin, Losses, GD, and any notation you find useful.
   CORRECTION We can recall that the real goal is to obtain good performance on the validation set, i.e. minimize

$$\mathcal{L}_{val}(\theta = \theta^*_{train}, X_{val}, Y_{val}),$$
$$\text{where } \theta^*_{train} = argmin_\theta(\mathcal{L}_{train}(\theta, X_{train}, Y_{train}))$$

   If θ^*_{train} is a global minimum of $\mathcal{L}_{train}$, it may be bad at getting low values for $\mathcal{L}_{val}(\theta^*_{train}, X_{val}, Y_{val})$

2 Maximum A Posteriori (MAP) (5.5 points)

We want to compute the MAP estimates of the parameters μ, σ for a random variable X that we model as following a Gaussian law, $\rho(x) \sim \mathcal{N}(\mu, \sigma)$. We recall that $\mathcal{N}(\mu, \sigma) : \rho(x) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$.

We have access to a data set $\widetilde{X} = \{x_1, \ldots, x_N\}$ of empirical observations. We may refer to the empirical mean as $\overline{x} = \frac{1}{N}\sum_n^N x_n$ and to the empirical variance as $\overline{V}[\widetilde{X}] = \overline{\sigma}^2 = \frac{1}{N}\sum_n (x_n - \overline{x})^2$.

The event "I observe the data is $\widetilde{X}$" can be written $X = \widetilde{X}$.

1. (2 pt) At first we introduce a Gaussian prior for the parameter μ: $\rho(\mu) = \mathcal{N}(0, \tau)$. Do compute the MAP estimate μ_{MAP} from the start, i.e. from the definition, recalling the fundamental steps (that are general to all cases) as well as the computations that apply to this precise case.
   CORRECTION Let's first make notations more precise. Here we have decided to assume that:
   - for all n, $P(x = x_n|\mu, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x_n-\mu)^2}{2\sigma^2}}$. *This is our model of the data, which is already a kind of prior assumption.*
   - In addition, we have a prior on μ: $P(\mu) = \mathcal{N}(0, \tau) = \frac{1}{\tau\sqrt{2\pi}} e^{-\frac{(\mu)^2}{2\tau^2}}$
   Since the data is i.i.d, $P(X = \widetilde{X}|\mu) = \prod_n P(x = x_n|\mu)$. *The MAP estimate for* μ *is thus:*

$$\begin{aligned}
\mu^*_{MAP} &= argmax_\mu(P(\mu|X = \widetilde{X})) && (1)\\
&= argmax_\mu \left(\frac{P(X = \widetilde{X}|\mu)P(\mu)}{P(X = \widetilde{X})} \right) && (2)\\
&= argmax_\mu \left(\log P(X = \widetilde{X}|\mu) + \log P(\mu) \right) && (3)\\
&= argmax_\mu \left(\log \prod_n \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x_n-\mu)^2}{2\sigma^2}} + \log \frac{1}{\tau\sqrt{2\pi}} e^{-\frac{(\mu)^2}{2\tau^2}} \right) && (4)
\end{aligned}$$

   Where between 1st and 2nd line we used Bayes theorem, and between 2nd and third, used the fact that the evidence $P(X = \widetilde{X})$ *does not depend on* μ, *i.e. is a constant that is killed in the argmax, and used that* log *is monotincally increasing, and thus can be applied as well.*
   The rest is just computation. Calling $\mathbf{L}$ *the term inside the argmax, we want to find* μ *such that* $\nabla_\mu \mathcal{L} = 0$.

$$\begin{aligned}
\nabla_\mu \mathcal{L} = 0 &\Leftrightarrow \nabla_\mu \left(\sum_n \left[\log \frac{1}{\sigma\sqrt{2\pi}} \right] + \sum_n \left[\log e^{-\frac{(x_n-\mu)^2}{2\sigma^2}} \right] + \left[\log \frac{1}{\tau\sqrt{2\pi}} e^{-\frac{(\mu)^2}{2\tau^2}} \right] \right) = 0\\
&\Leftrightarrow \left(0 + \sum_n \nabla_\mu \left[-\frac{(\mu - x_n)^2}{2\sigma^2} \right] + \nabla_\mu \left[-\frac{(\mu)^2}{2\tau^2} \right] \right) = 0\\
&\Leftrightarrow \left(-\left[\sum_n \frac{(\mu - x_n)}{\sigma^2} \right] - \frac{\mu}{\tau^2} \right) = 0\\
&\Leftrightarrow \left(-\mu + \frac{1}{N}\sum_n x_n - \frac{\sigma^2}{N\tau^2}\mu \right) = 0\\
&\Leftrightarrow \mu = \frac{\overline{x}}{1 + \frac{\sigma^2}{N\tau^2}}, \text{ where } \overline{x} = \frac{1}{N}\sum_n x_n
\end{aligned}$$

   In conclusion, $\mu_{MAP} = \frac{\overline{x}}{1+\frac{\sigma^2}{N\tau^2}}$.

2. (0.25 pt) Without much computation, say what is σ_{MAP} (you can explain well or draft a few lines of computation)
   CORRECTION Since there is no prior on σ, *the MAP estimate of* σ *will be very similar to the MLE one, which is the empirical standard deviation,* $\sigma_{MLE} = \overline{\sigma} = \sqrt{\frac{1}{N}\sum_n (x_n - \mu)^2}$. *The only subtlety is that* μ *should be* μ_{MAP} *and not* μ_{MLE}: *so,* $\sigma_{MAP} = \sqrt{\frac{1}{N}\sum_n (x_n - \mu_{MAP})^2}$

3. (0.25 pt) Interpret (comment on) the limit $N \to \infty$.
   CORRECTION This relates of course to μ_{MAP}. *When* $N \to \infty$, *we recover* $\mu_{MAP} = \frac{\overline{x}}{1+0^+} = \mu_{MLE}$. *This is to be expected: as we get to see more and more data, we rely less and less on our prior assumptions about the data to estimate things about the data.*

4. (0.25 pt) Interpret (comment on) the limit $\tau \to 0$.
   CORRECTION The limit $\tau \to 0$ *corresponds to a very sharp Gaussian prior on* μ, *very centered on* 0.

In other words, we have a strong a priori belief. In that case, $\mu_{MAP} = \frac{\overline{x}}{1+\infty} \sim 0$: we expect the average to be basically 0, regardless of our data.

5. (0.25 pt) Interpret (comment on) the limit $\tau \to \infty$.
CORRECTION This is the opposite case, i.e. we have a flat prior, and so the correction to the data is minimal: $\mu_{MAP} = \frac{\overline{x}}{1+0^+} = \mu_{MLE}$. In other words, we had no a priori belief.

6. (2 pt) Now we forget this prior on μ and only assume a prior on σ. Precisely, we use an exponential prior for σ (which by definition, is positive), σ: $\rho(\sigma) = \lambda e^{-\lambda\sigma}$, where $\lambda > 0$. Remember that $\mathbb{E}[\rho_\lambda] = \int_0^\infty \sigma\ \lambda e^{-\lambda\sigma} \mathrm{d}\sigma = 1/\lambda$. Compute the MAP estimate σ_{MAP}. Computation is slightly heavier but conceptually not more difficult than in the first question.
CORRECTION

$$\sigma^*_{MAP} = argmax_\sigma(P(\sigma|X = \widetilde{X})) \tag{5}$$
$$= argmax_\sigma \left(\log P(X = \widetilde{X}|\sigma) + \log P(\sigma)\right) \tag{6}$$
$$= argmax_\sigma \left(\log \prod_n \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x_n-\mu)^2}{2\sigma^2}} + \log \lambda e^{-\lambda\sigma}\right) \tag{7}$$

Again, the rest is just computation. Calling $\mathbf{L}_\sigma$ the term inside the argmax, we want to find σ such that $\nabla_\sigma \mathcal{L} = 0$.

$$\nabla_\sigma \mathcal{L}_\sigma = 0 \Leftrightarrow \nabla_\sigma \left(\log \prod_n \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{(x_n-\mu)^2}{2\sigma^2}} + \log \lambda e^{-\lambda\sigma}\right) = 0 \tag{8}$$
$$\Leftrightarrow -N\nabla_\sigma \log(\sigma) - \left(\nabla_\sigma \frac{1}{2\sigma^2}\right) \sum_n (x_n - \mu)^2 + \nabla_\sigma(-\lambda\sigma) = 0 \tag{9}$$
$$\Leftrightarrow -N\frac{1}{\sigma} + \frac{2}{2\sigma^3} \sum_n (x_n - \mu)^2 - \lambda = 0 \tag{10}$$
$$\Leftrightarrow -\sigma^2 + \frac{1}{N} \sum_n (x_n - \mu)^2 - \frac{\sigma^3}{N}\lambda = 0 \tag{11}$$
$$\Leftrightarrow \sigma^2 \left(1 + \frac{\sigma}{N}\lambda\right) = \frac{1}{N} \sum_n (x_n - \mu)^2 = \overline{\sigma}^2 \tag{12}$$
$$\Leftrightarrow \sigma^2 = \overline{\sigma}^2 \frac{1}{\left(1 + \frac{\sigma}{N}\lambda\right)} \tag{13}$$

This equation is unfortunately a pain to reduce, but can be solved. For the discussion, it's enough to note that we can e.g. assume λ to be small, in which case $\sigma \approx \overline{\sigma}$ and the expression reduces to:

$$\sigma^2 \approx \overline{\sigma}^2 \frac{1}{\left(1 + \frac{\overline{\sigma}}{N}\lambda\right)} \tag{14}$$

To discuss limits, one should rely on the exact equation 13.

7. (0.5 pt) Now, we assume both priors simultaneously, i.e. we have the joint prior distribution:
$\rho(\mu, \sigma) = \frac{1}{\tau\sqrt{2\pi}} e^{-\frac{(\mu)^2}{2\tau^2}} \lambda e^{-\lambda\sigma}$.
Does μ_{MAP} change at all, and if so, how ?
Does σ_{MAP} change at all, and if so, how ?
(Note: I do not expect a lot of computation in this last question, just a bit of reasoning.)
CORRECTION Here, one has to notice first that when σ or μ appear in each other's MAP estimation, it's because those parameters are in the expression of our model for x, which was $\mathcal{N}(\mu, \sigma)$. That is, the values of μ, σ are not known, and especially, there is no reason to take them equal to the empirical estimates $\overline{\mu}, \overline{\sigma}$. Instead, they are estimated by whatever estimate we have, here, the MAP estimate.
μ_{MAP} adapts to become $\mu_{MAP} = \frac{\overline{x}}{1+\frac{\sigma^2_{MAP}}{N\tau^2}}$.
σ_{MAP} is computed with the empirical variance being defined as: $\overline{\sigma_{MAP}}^2 = \frac{1}{N}\sum_n (x_n - \mu_{MAP})^2$. Each expression depends on the other, but it's fairly easy to converge by decimation, esp. with large τ and small λ (i.e. reasonably weak priors wrt dataset size).

3 Binary classification with a few neurons (7.5 pts)

We have a dataset $X, Y = \{(\vec{x}_1, y_1), \ldots, (\vec{x}_N, y_N)\}$ where Y encodes the binary labels. The data is D dimensional, $\vec{x} \in \mathbb{R}^D$. To classify the data, we use the following model and Loss:

$$\hat{y}(\vec{x}_n) = f_\theta(\vec{x}_n) = \tanh(v\ ReLU(\vec{w} \cdot \vec{x}_n + a) + b) \tag{15}$$

$$\mathcal{L} = \frac{1}{N} \sum_n^N (\hat{y}(\vec{x}_n) - y_n)^2 \tag{16}$$

where ReLu is the so-called *Rectified Liner Unit*, i.e. $ReLU(z) = max(0, z)$. We may introduce the convenient notations:

$$f_\theta^{(1)}(\vec{x}_n) = \vec{w} \cdot \vec{x}_n + a \tag{17}$$

$$f_\theta^{(2)}(\vec{x}_n) = v\ ReLU(f_\theta^{(1)}(\vec{x}_n)) + b = v\ ReLU(\vec{w} \cdot \vec{x}_n + a) + b \tag{18}$$

$$\text{so that we have: } f_\theta(\vec{x}_n) = \tanh(f_\theta^{(2)}(\vec{x}_n)) \tag{19}$$

We recall that the hyperbolic tangent function is defined by: $\tanh(z) = \frac{e^z - e^{-z}}{e^z + e^{-z}}$. It goes from -1 at $z \sim -\infty$ to $+1$ at $z \sim +\infty$. Its derivative is (for a generic smooth function u): $\frac{\partial}{\partial z} \tanh(u(z)) = \frac{4u'(z)}{(\cosh(u(z)))^2}$, where $\cosh(z)$ is the hyperbolic cosine, $\cosh(z) = \frac{e^z + e^{-z}}{2}$. We know that $\cosh(z) \geq 1, \forall z$.

We denote $H(z)$ the Heaviside function (step function) : $H(z) = \begin{cases} 1 \text{ if } z \geq 0 \\ 0 \text{ if } z < 0 \end{cases}$.

You can introduce other notations if you find them useful.

1. (0.5 pt) This question is independent from the next ones. What is the maximal Loss that a single example can contribute to the Loss? What is its minimal value? Considering this observation, do you think this model is very sensitive to outliers?
   CORRECTION Question 9 should have come first... sorry. Since estimates $\hat{y}$ are in $[-1, 1]$ (output of tanh is in this range, obviously the labels' encoding should be in$\{+1, -1\}$. The maximal difference is thus $2^2 = 4$. This model will not penalize outliers strongly.

2. (0.25 pt) List the parameters θ of this model, and for each, say what space it is in ($\mathbb{R}, \mathbb{R}^D$, etc).
   CORRECTION Parameters are $a, b, v, \vec{w}$, the first three are scalar (real) values, the last one is the same dimension as an example $\vec{x}_n$, i.e. $\vec{w} \in \mathbb{R}^D$

3. (0.25 pt) Compute the derivative $\frac{\partial}{\partial z} ReLU(u(z))$, for a generic function $u(z)$ (assumed derivable).
   CORRECTION Since $ReLU(u(z)) = max(0, u(z))$, the derivative is 0 before 0 and linear in $u(z)$ for $u(z) > 0$, i.e. we have (applying the chain rule: $\nabla_z ReLU(u(z)) = H(u(z)).\nabla_z u(z)$

4. (0.75 pt) Compute $\nabla_b \mathcal{L}$. Ideally, try to derive the steps in a generic way as far as you can in the computation (this is to help you for the next questions).
   CORRECTION For the first case, we do it in detail. The trick is to try to not write lots of terms when they can be simply written as $f_\theta^{(1)}(\vec{x}_n)$ or $f_\theta^{(2)}(\vec{x}_n)$.

$$\nabla_b \mathcal{L} = \nabla_b \frac{1}{N} \sum_n (\hat{y}_n - y_n)^2 \tag{20}$$

$$= \frac{2}{N} \sum_n (\hat{y}_n - y_n)^1 \nabla_b \hat{y}_n \tag{21}$$

$$= \frac{2}{N} \sum_n (\hat{y}_n - y_n) \nabla_b tanh(f_\theta^{(2)}(\vec{x}_n)) \tag{22}$$

$$= \frac{2}{N} \sum_n (\hat{y}_n - y_n) \frac{4}{cosh^2(f_\theta^{(2)}(\vec{x}_n))} \nabla_b f_\theta^{(2)}(\vec{x}_n) \tag{23}$$

$$= \frac{8}{N} \sum_n (\hat{y}_n - y_n) \frac{1}{cosh^2(f_\theta^{(2)}(\vec{x}_n))} \nabla_b (... + b) \tag{24}$$

$$= \frac{8}{N} \sum_n (\hat{y}_n - y_n) \frac{1}{cosh^2(f_\theta^{(2)}(\vec{x}_n))} 1 \tag{25}$$

Note that all the steps up to and including eq. 23 are completely general.

5. (0.75 pt) Compute $\nabla_v \mathcal{L}$ (re-use previous question's result).
CORRECTION We re-use the steps above up to Eq. 23 to get:

$$\nabla_v \mathcal{L} = \frac{8}{N}\sum_n (\hat{y}_n - y_n)\frac{1}{cosh^2(f_\theta^{(2)}(\vec{x}_n))}\nabla_v f_\theta^{(2)}(\vec{x}_n) \tag{26}$$

$$= \frac{8}{N}\sum_n (\hat{y}_n - y_n)\frac{1}{cosh^2(f_\theta^{(2)}(\vec{x}_n))}[ReLU(f_\theta^{(1)}(\vec{x}_n)) + 0] \tag{27}$$

6. (0.75 pt) Compute $\nabla_a \mathcal{L}$ (re-use previous question's result).
CORRECTION We re-use the steps above up to Eq. 23 to get:

$$\nabla_a \mathcal{L} = \frac{8}{N}\sum_n (\hat{y}_n - y_n)\frac{1}{cosh^2(f_\theta^{(2)}(\vec{x}_n))}\nabla_a f_\theta^{(2)}(\vec{x}_n) \tag{28}$$

$$= \frac{8}{N}\sum_n (\hat{y}_n - y_n)\frac{1}{cosh^2(f_\theta^{(2)}(\vec{x}_n))} v.\nabla_a(ReLU(f_\theta^{(1)}(\vec{x}_n))) + 0 \tag{29}$$

$$= \frac{8}{N}\sum_n (\hat{y}_n - y_n)\frac{1}{cosh^2(f_\theta^{(2)}(\vec{x}_n))} vH(f_\theta^{(1)}(\vec{x}_n))\nabla_a f_\theta^{(1)}(\vec{x}_n) \tag{30}$$

$$= \frac{8}{N}\sum_n (\hat{y}_n - y_n)\frac{1}{cosh^2(f_\theta^{(2)}(\vec{x}_n))} vH(f_\theta^{(1)}(\vec{x}_n))1 \tag{31}$$

Here we used the result shown in question 2.

7. (0.75 pt) Compute $\vec{\nabla}_{\vec{w}} \mathcal{L}$ (re-use previous question's result).
CORRECTION We re-use the steps above up to Eq. 30 to get:

$$\nabla_{\vec{w}} \mathcal{L} = \frac{8}{N}\sum_n (\hat{y}_n - y_n)\frac{1}{cosh^2(f_\theta^{(2)}(\vec{x}_n))} vH(f_\theta^{(1)}(\vec{x}_n))\nabla_{\vec{w}} f_\theta^{(1)}(\vec{x}_n) \tag{32}$$

$$= \frac{8}{N}\sum_n (\hat{y}_n - y_n)\frac{1}{cosh^2(f_\theta^{(2)}(\vec{x}_n))} vH(f_\theta^{(1)}(\vec{x}_n))\vec{x}_n \tag{33}$$

8. (0.5 pt) Write down $\vec{\nabla}_{\vec{w}} \mathcal{L}$ in terms of $\nabla_b \mathcal{L}$ (making it appear explicitly).
CORRECTION Sorry, the question was poorly written. Here I wanted to simpy make sure you stress out the similarity between the explicit forms. In the end I gave up to 0.25 when the idea was there, or if it was niticed that it's impossible, and 0 otherwise

9. (0.25 pt) What should be our choice of encoding of the binary labels y_n? In other words, which are the 2 values the y_n's should take?
CORRECTION As said above, $y_n \in \{-1, +1\}$

10. (0.25 pt) What should be the decision function (readout), i.e. what is the expression of y_{predict} as a function of $\hat{y}$?
CORRECTION It needs to map $\hat{y}$ to $\{-1, +1\}$, so the good choice is $sign(\hat{y})$.

11. (0.5 pt) Compute $\nabla_b\Big(\frac{1}{N}\sum_n^N (y_{\text{predict}}(\vec{x}_n) - y_n)^2\Big)$? What is the problem with this loss ?
CORRECTION This Loss is almost always zero, because the derivative of $sign(u(z)$ is 0 everywhere except in 0, where it could be assigned to whatever we want (pseudo gradient). Denoting it 0, we have:*

$$\nabla_b \mathcal{L} = \nabla_b \frac{1}{N}\sum_n (y_n^{pred} - y_n)^2 \tag{34}$$

$$= \frac{2}{N}\sum_n (y_n^{pred} - y_n)^1 \nabla_b y_n^{pred} \tag{35}$$

$$= \frac{2}{N}\sum_n (y_n^{pred} - y_n)^1 \nabla_b sign(\hat{y}_n) \tag{36}$$

$$= \frac{2}{N}\sum_n (y_n^{pred} - y_n) 0 * \nabla_b \hat{y}_n \tag{37}$$

$$= 0 \text{ } \textit{(almost always)} \tag{38}$$

12. Priors. We now have the idea that the data comes from the following process:

$$y_n = \tanh(v\ ReLU(\vec{w} \cdot \vec{x}_n + a) + b) + \varepsilon_n \tag{39}$$

where ε_n are noise terms, each ε_n is an i.i.d. Gaussian variable: $\rho(\varepsilon) = \mathcal{N}(0, \sigma)$, i.e. $\rho(\varepsilon) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{\varepsilon^2}{2\sigma^2}}$. Additionally, we have a Gaussian prior on v: $\rho(v) = \mathcal{N}(0, \lambda)$.

(a) (0.5 pt) Can you guess the result of a MAP estimate for v ?
CORRECTION It will most likely be an L2-regularization term on v, i.e. the loss will have an additional term $\propto \frac{1}{\lambda}|v|^2$

(b) (1 pt) Prove your guess, i.e. perform the MAP reasonning, being as rigorous as possible.
Note: you cannot and are not asked to compute v_{MAP} explicitly, you should just write the problem that needs to be solved numerically to get v_{MAP}, and notice to what it corresponds to.
CORRECTION This is a classical computation that was done in class. We quickly sketch it here: $v_{MAP} = argmax(logP(X = x, Y = y|v) + logP(v))$.
Here $P(X = x, Y = y|v) \sim \rho(\varepsilon) = \mathcal{N}(0, \sigma) = \frac{1}{\sigma\sqrt{2\pi}} e^{-\frac{\varepsilon^2}{2\sigma^2}}$. *Recalling that* $\varepsilon_n = \hat{y}_n - y_n$, *we get:*

$$v_{MAP} = argmax_v \left(\log P(X = x, Y = y|v) + \log P(v)\right) \tag{40}$$

$$= argmax_v \left(const + \frac{1}{N} \sum_n \frac{-(\hat{y}_n - y_n)^2}{2\sigma^2} + \log(\rho(v)) \right) \tag{41}$$

$$= argmax_v \left(\frac{1}{N} \sum_n -(\hat{y}_n - y_n)^2 - 2\sigma^2 \frac{v^2}{2\lambda^2} \right) \tag{42}$$

Which ends up being the usual MSE term + a term in v^2. *That is, the usual L2 regularization.*

(c) (0.5) Guess again what happens if we have a Gaussian prior on each component w_d of the vector $\vec{w}$.
CORRECTION Again, it will come up as an additional term $\propto ||\vec{w}||^2$ *in the loss.*

Note: in deep neural networks, typically, people use such priors.
CORRECTION Actually, the bias terms are usually not regularized !! It makes sense to let the average output of a layer be non-zero, which is what the bias term allows to do, essentially. To avoid the overall activation values to be too large, one can use Batch-Norm. But regularizing the bias terms is unusual (here for pedagogical reasons it was practical to consider the regularization on v).

4 Lasso Regularization (simple) (4 pts)

The first two questions are not about Lasso, but are here as preliminary, to help.

1. (0.5) Solve the minimization problem: $\text{argmin}_w(\frac{1}{2}(w-a)^2 + \lambda w^2)$, where $w \in \mathbb{R}, a \in \mathbb{R}$
CORRECTION In this question, like the next ones, we can find the minimum of the function by exact computation. We apply the theorem that says that extrema of $J(w)$ *are where* $nabla_{\vec{w}} J(\vec{w}) = \vec{0}$. *We do not need to compute the second derivative exactly, the fact that the extremum (we typically find only one or two) is a minimum is obvious, each time.*

$$\nabla_w J_1 = 0 \Leftrightarrow \nabla_w \Big(\frac{1}{2}(w-a)^2 + \lambda w^2 \Big) \tag{43}$$

$$\Leftrightarrow (w-a) + 2\lambda w = 0 \tag{44}$$

$$\Leftrightarrow w = \frac{a}{1+2\lambda} \tag{45}$$

2. (0.5) Solve the minimization problem: $\text{argmin}_{\vec{w}}(\frac{1}{2}||\vec{w} - \vec{a}||^2 + \lambda||\vec{w}||^2)$, where $\vec{w} \in \mathbb{R}^D, \vec{a} \in \mathbb{R}^D$. Take the time to first find the solution for the component w_1, explicitating the scalar products (or norms) as sums, and only then, generalize.
CORRECTION Following the steps suggested, we compute only $\partial_{w_1} J_2$

$$\partial_{w_1} J_2 = 0 \Leftrightarrow \partial_{w_1} \Big(\frac{1}{2}||\vec{w} - \vec{a}||^2 + \lambda||\vec{w}||^2 \Big) \tag{46}$$

$$\Leftrightarrow \partial_{w_1} \Big(\frac{1}{2}\big[(w_1 - a_1)^2 + (w_2 - a_2)^2 + \dots\big] + \lambda(w_1^2 + w_2^2 + \dots) \Big) \tag{47}$$

$$\Leftrightarrow \frac{1}{2} 2(w_1 - a_1) + 0 + 2\lambda w_1 + 0 \tag{48}$$

$$\Leftrightarrow w_1 = \frac{a_1}{1+2\lambda} \tag{49}$$

Which generalizes to all $d \neq 1$, so we get: $\vec{w}^* = \vec{a}\frac{1}{1+2\lambda}$
We could also have computed it directly:

$$\nabla_{\vec{w}} J_2 = \vec{0} \Leftrightarrow \frac{1}{2}2(\vec{w} - \vec{a}) + 2\lambda\vec{w} \tag{50}$$

$$\Leftrightarrow \vec{w} = \vec{a}\frac{1}{1+2\lambda} \tag{51}$$

3. (2 pts) Solve the minimization problem: $\text{argmin}_w(\frac{1}{2}(w-a)^2 + \lambda|w|)$, where $w \in \mathbb{R}, a \in \mathbb{R}$, and $|.|$ is the absolute value.
You should assume that your pseudo-gradient $\frac{\partial}{\partial w}|w|$ is in the range $[-1, 1]$, and check that you find a solution for any value of a.
Hint: you should discuss the cases where the solution is $w \neq 0$ and where it is $w = 0$, separately.
CORRECTION We first assume that the solution w^* will be non-zero. In that case, we have $\nabla_w|w| = sign(w)$.

$$\partial_{w_1} J_3 = 0 \Leftrightarrow \nabla_w\Big(\frac{1}{2}(w-a)^2 + \lambda|w|\Big) \tag{52}$$

$$\Leftrightarrow w - a + \lambda sign(w) = 0 \tag{53}$$

$$\Leftrightarrow w = a - \lambda sign(w) \tag{54}$$

We should check that this is self-consistent. To do this, we do the two sub-cases, $w < 0$ and $w > 0$.

$$w = a - \lambda sign(w)$$
$$w > 0 \Leftrightarrow a - \lambda sign(w) > 0 \Leftrightarrow a - \lambda > 0 \Leftrightarrow a > \lambda > 0$$
$$w < 0 \Leftrightarrow a - \lambda sign(w) < 0 \Leftrightarrow a + \lambda < 0 \Leftrightarrow a < -\lambda < 0$$

Remember that $\lambda > 0$. We note that in both cases, we need to have $|a| > \lambda$.
And we note that the $sign(w) = sign(a)$
So we can summarize the case $w \neq 0$ as: if $|a| > \lambda$, then $w^* = a - \lambda sign(a) \neq 0$.
.
Second case: assuming $w^* = 0$. Then, $\nabla_w|w|$ is a pseudo-gradient, meaning concretely we can choose its value, within the range $[-1, 1]$.

$$\partial_{w_1} J_3 = 0 \Leftrightarrow \nabla_w\Big(\frac{1}{2}(w-a)^2 + \lambda|w|\Big) \tag{55}$$

$$\Leftrightarrow w - a + \lambda\nabla_w|w| = 0 \tag{56}$$

since we assumed $w = 0$, we replace w with 0: $\Leftrightarrow 0 - a + \lambda\nabla_w|w| = 0$ (57)

$$\Leftrightarrow \nabla_w|w| = \frac{a}{\lambda} \tag{58}$$

So the hypothesis $w = 0$ is self-consistent iff $\frac{a}{\lambda} \in [-1, 1]$. That is, when $|a| \leq \lambda$.
Summary: if $|a| \leq \lambda$, then $w^* = 0$.
.
We note that the two cases are perfectly complementary, we have a solution for any a, and that the solutions are continuous (at the change point when $|a| = \lambda$).

4. (1 pt) Solve the minimization problem: $\text{argmin}_{\vec{w}}(\frac{1}{2}||\vec{w} - \vec{a}||^2 + \lambda|\vec{w}|_1)$, where $|.|_1$ is the $L1$ norm, i.e. $|\vec{w}|_1 = \sum_d^D |w_d|$.
Hint: you can generalize from previous cases, not necessarily re-writing all explanations.
CORRECTION Here it is wiser to indeed compute only one component, to avoid silly indices mistakes. The point is that for each component, we have

$$\partial_{w_1} J_4 = 0 \Leftrightarrow \partial_{w_1}\Big(\frac{1}{2}||\vec{w} - \vec{a}||^2 + \lambda|\vec{w}|\Big) \tag{59}$$

$$\Leftrightarrow \partial_{w_1}\Big(\frac{1}{2}\big[(w_1 - a_1)^2 + (w_2 - a_2)^2 + \dots\big] + \lambda(|w_1| + |w_2| + \dots)\Big) \tag{60}$$

$$\Leftrightarrow (w_1 - a_1) + 0 + \dots + \lambda\partial_{w_1}|w_1| + 0 + \dots \tag{61}$$

$$\Leftrightarrow \partial_{w_1} J_3(w_1) = 0 \tag{62}$$

We notice that we are back to the question 3, with simply w_1 instead of w and a_1 instead of a.
So the solution is the same as in question 3, component by component:
$\forall d, w_d = 0$ if $|a_d| < \lambda$, else $w_d = a_d - \lambda sign(a_d)$