Some questions

1. What is Optimization?

The process of finding parameters that minimizing the loss function.

2. Three strategies to minimize loss function.

- Try out many different parameters and keep track of what works best.
- Start out with a random W and choose a random direction. Concretely, if loss at $W + \delta W$ is lower. We perform an update.
- Inspired by strategy two, we can actually choose the negative gradient direction. Analogous, this direction is the steepest in the hill. But why? Consider the first-order Taylor expansion of some continuously differentiable real-valued function $f : \mathbb{R} \to \mathbb{R}$.

  $$f(x + \epsilon) = f(x) + \epsilon f'(x).$$

  for fixed x we want to minimize $f(x + \epsilon)$, which is equivalent to minimizing $\epsilon f'(x) = |\epsilon||f'(x)|\cos\theta$. Apparently, we can set θ to negative 1, which means the direction of ϵ is the negative direction of $f'(x)$. Thus the decline in the negative gradient direction is the most.

3. What is Mini-batch gradient descent and why we use it?

- Computing gradient over batches of the training data, we usually set hyperparameter to the power of 2, e.g. 32, 64, or 128.
- It is wasteful to compute the full loss function over the entire training set in order to perform only a single parameter update.

4. What is SGD(Stochastic Gradient Descent)?

The mini-batch only contains one single example.

5. What is cross-validation?

We usually split raw data to two parts, training set and test set. Test set can't be used to set parameters, since it may cause our final model to overfit. One way to solve this is to split some date from training set as validation set. This set is used for tuning the parameters. In case the training set is small, **cross-validation** is used. The idea is that instead of choosing some data as validation set. We split training set to 5 parts. And choose one part as validation set, the rest as training set. Then iterate over which fold is the validation fold, evaluate the performance, and finally average the performance across the different folds.

- In practice, people prefer to avoid cross-validation in favor of having a single validation split, since cross-validation can be computationally expensive.
- Typical number of folds you can see in practice would be 3-fold, 5-fold or 10-fold cross-validation.
- There is a question of whether you should use the full training set with the best hyperparameters, since the optimal hyperparameters might change if you were to fold the validation data into your training set (since the size of the data would be larger). In practice it is cleaner to not use the validation data in the final classifier and consider it to be burned on estimating the hyperparameters. **Here is a problem, what about cross-validation?**