# HOMEWORK 1 - V0

STA 414/2104 Winter 2021

*University of Toronto*

Version history: V0 → V1:

- **Deadline:** Mon, Feb. 8, at 13:59.
- **Submission:** You need to submit your solutions through Crowdmark, including all your derivations, plots, and your code. You can produce the file however you like (e.g. LaTeX, Microsoft Word, etc), as long as it is readable. Points will be deducted if we have a hard time reading your solutions or understanding the structure of your code. For each question, you should append your code right after your answers to that question.

## 1. Probability and Calculus - 20 pts.

1.1. *Variance and covariance - 10 pts.* Let $X, Y$ be two independent random vectors in $\mathbb{R}^m$.

(a) Find their covariance.
(b) For a constant matrix $A \in \mathbb{R}^{m \times m}$, show the following two properties:

$$\mathbb{E}(X + AY) = \mathbb{E}(X) + A\mathbb{E}(Y)$$
$$\text{Var}(X + AY) = \text{Var}(X) + A\text{Var}(Y)A^T$$

(c) Using part (b), show that if $X \sim \mathcal{N}(\mu, \Sigma)$, then $AX \sim \mathcal{N}(A\mu, A\Sigma A^T)$.

*What to submit?*

*a) Here, simply stating the answer is sufficient.*
*b) Show that the equalities hold for each entry.*
*c) Here, you may use the fact that linear transformation of a Gaussian random vector is again Gaussian. Therefore you only need to compute the expected value and variance of $AX$.*

1.2. *Calculus - 10 pts.* Let $x, y \in \mathbb{R}^m$ and $A \in \mathbb{R}^{m \times m}$. In vector notation, what is

(a) the gradient with respect to $x$ of $x^T y$?
(b) the gradient with respect to $x$ of $x^T x$?
(c) the gradient with respect to $x$ of $\frac{1}{2} x^T A x$?
(d) the gradient with respect to $x$ of $\exp(x^T A x)$?

*What to submit?*

*a-d) Use the definition of gradient and find its each entry. Results should be in vector form.*

**2. Regression - 40 pts.** In this question, you will derive certain properties of linear regression.

2.1. *Linear regression - 20 pts.* Suppose that $\mathbf{\Phi} \in \mathbb{R}^{n \times m}$ with $n \geq m$ and $\mathbf{t} \in \mathbb{R}^n$, and that $\mathbf{t}|(\mathbf{\Phi}, \mathbf{w}) \sim \mathcal{N}(\mathbf{\Phi w}, \sigma^2 \mathbf{I})$. We know that the maximum likelihood estimate $\hat{\mathbf{w}}$ of $\mathbf{w}$ is given by

$$\hat{\mathbf{w}} = (\mathbf{\Phi}^T \mathbf{\Phi})^{-1} \mathbf{\Phi}^\top \mathbf{t}.$$

(a) Write the log-likelihood implied by the model above, and compute its gradient w.r.t. $\mathbf{w}$. By setting it equal to 0, derive the above estimator $\hat{\mathbf{w}}$.
(b) Find the distribution of $\hat{\mathbf{w}}$, its expectation and covariance matrix.

*What to submit?*

*a) Log-likelihood, its gradient, and your entire derivation.*
*b) Use the property in 1.1 c) of multivariate Gaussian random vectors, and find the distribution, and calculate its expectation and variance.*

2.2. *Ridge regression and MAP - 20 pts.* Suppose that we have $\mathbf{t}|(\mathbf{\Phi}, \mathbf{w}) \sim \mathcal{N}(\mathbf{\Phi w}, \sigma^2 \mathbf{I})$ and we place a normal prior on $\mathbf{w}|\mathbf{\Phi}$, i.e., $\mathbf{w} \sim \mathcal{N}(0, \tau^2 \mathbf{I})$. Recall from the first lecture (also in preliminaries.pdf) that MAP estimate of $\mathbf{w}$ is given as the maximum of the posterior density

$$\hat{\mathbf{w}}_{MAP} = \underset{\mathbf{w}}{\text{argmax}} \{p(\mathbf{w}|\mathbf{\Phi}, \mathbf{t}) \propto p(\mathbf{t}|\mathbf{\Phi}, \mathbf{w}) p(\mathbf{w}|\mathbf{\Phi})\}.$$

Here, $\propto$ notation means *proportional to*, and is used since we dropped the term $p(\mathbf{t}|\mathbf{\Phi})$ in the denominator as it doesn't have $\mathbf{w}$ in it, thus it doesn't contribute to the maximization problem.

Show that the MAP estimate of $\mathbf{w}$ given $(\mathbf{t}, \mathbf{\Phi})$ in this context is

$$\hat{\mathbf{w}}_{MAP} = (\mathbf{\Phi}^\top \mathbf{\Phi} + \lambda \mathbf{I})^{-1} \mathbf{\Phi}^\top \mathbf{t} \tag{2.1}$$

where $\lambda = \sigma^2 / \tau^2$.

*What to submit?*

*a) Submit all your derivations.*

**3. Cross validation - 40 pts.** In this problem, you will write a function that performs $K$-fold cross validation (CV) procedure to tune the penalty parameter $\lambda$ in Ridge regression. CV procedure is one of the most commonly used methods for tuning hyperparameters. *In this question, you shouldn't use the package* `scikit-learn` to perform CV. *You should implement all of the below functions yourself. You may use* `numpy` *and* `scipy` *for basic math operations such as linear algebra, sampling etc.*

In class we learned *training*, *test*, and *validation* procedures which assumes that you have enough data and you can set aside a validation set and a test set to use it for assessing the performance of your machine learning algorithm. However in practice, this may be problematic since we may not have enough data. A remedy to this issue is K-fold cross- validation which uses a part of the available data to fit the model, and a different part to test it. K-fold CV procedure splits the data into K equal-sized parts; for example, when K = 5, the scenario looks like this:

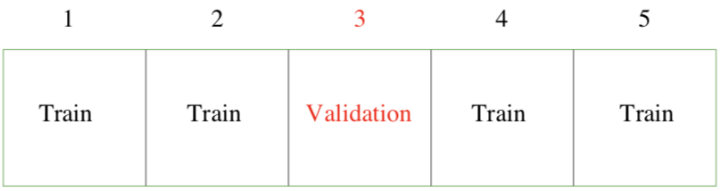


Fig 1: credit: Elements of Statistical Learning

1. We first set aside a test dataset and never use it until the training and parameter tuning procedures are complete. We will use this data for final evaluation. In this question, test data is provided to you as a separate dataset.
2. CV error estimates the test error of a particular hyperparameter choice. For a particular hyperparameter value, we split the training data into $K$ blocks (See the figure), and for $k = 1, 2, ..., K$ we use the $k$-th block for validation and the remaining $K - 1$ blocks are for training. Therefore, we train and validate our algorithm $K$ times. Our CV estimate for the test error for that particular hyperparameter choice is given by the average validation error across these $K$ blocks.
3. We repeat the above procedure for several hyperparameter choices and choose the one that provides us with the smalles CV error (which is an estimate for the test error).

Below, we will code the above procedure for tuning the regularization parameter in linear regression which is a hyperparameter. Your `cross_validation` function will rely on 6 short functions which are defined below along with their variables.

- `data` is a variable and refers to a $(\mathbf{t}, \mathbf{\Phi})$ pair (can be test, training, or validation) where $\mathbf{t}$ is the target (response) vector, and $\mathbf{\Phi}$ is the feature matrix.
- `model` is a variable and refers to the coefficients of the trained model, i.e. $\hat{\mathbf{w}}_\lambda$.
- `data_shf = shuffle_data(data)` is a function and takes `data` as an argument and returns its randomly permuted version along the samples. Here, we are considering a uniformly random permutation of the training data. Note that $\mathbf{t}$ and $\mathbf{\Phi}$ need to be permuted the same way preserving the target-feature pairs.
- `data_fold, data_rest = split_data(data, num_folds, fold)` is a function that takes `data`, number of partitions as `num_folds` and the selected partition `fold` as its arguments and returns the selected partition (block) `fold` as `data_fold`, and the remaining data as `data_rest`. If we consider 5-fold cross validation, `num_folds`=5, and your function splits the data into 5 blocks and returns the block `fold` ($\in \{1, 2, 3, 4, 5\}$) as the validation fold and the remaining 4 blocks as `data_rest`. Note that `data_rest` $\cup$ `data_fold` = `data`, and `data_rest` $\cap$ `data_fold` $= \emptyset$.
- `model = train_model(data, lambd)` is a function that takes `data` and `lambd` as its arguments, and returns the coefficients of ridge regression with penalty level $\lambda$. For simplicity, you may ignore the intercept and use the expression in equation (2.1).
- `predictions = predict(data, model)` is a function that takes `data` and `model` as its arguments, and returns the `predictions` based on `data` and `model`.

- `error = loss(data, model)` is a function which takes `data` and `model` as its arguments and returns the average squared `error` loss based on `model`. This means if `data` is composed of $\mathbf{t} \in \mathbb{R}^n$ and $\mathbf{\Phi} \in \mathbb{R}^{n \times p}$, and model is $\hat{\mathbf{w}}$, then the return value is $\|\mathbf{t} - \mathbf{\Phi}\hat{\mathbf{w}}\|^2/n$.
- `cv_error = cross_validation(data, num_folds, lambd_seq)` is a function that takes the training `data`, number of folds `num_folds`, and a sequence of $\lambda$'s as `lambd_seq` as its arguments and returns the cross validation error across all $\lambda$'s. Take `lambd_seq` as evenly spaced 50 numbers over the interval [0.02, 1.5]. This means `cv_error` will be a vector of 50 errors corresponding to the values of `lambd_seq`. Your function will look like:

```
data = shuffle_data(data)
for i = 1,2,...,length(lambd_seq)
       lambd = lambd_seq(i)
       cv_loss_lmd = 0.
       for fold = 1,2, ...,num_folds
            val_cv, train_cv = split_data(data, num_folds, fold)
            model = train_model(train_cv, lambd)
            cv_loss_lmd += loss(val_cv, model)
       cv_error(i) = cv_loss_lmd / num_folds
return cv_error
```

Download the dataset from the course webpage `hw1_data.zip` and place and extract in your working directory, or note its location `file_path`. For example, file path could be `/Users/yourname/Desktop/`

- In `Python`:

```
import numpy as np
data_train = {'X': np.genfromtxt('data_train_X.csv', delimiter=','),
              't': np.genfromtxt('data_train_y.csv', delimiter=',')}
data_test = {'X': np.genfromtxt('data_test_X.csv', delimiter=','),
             't': np.genfromtxt('data_test_y.csv', delimiter=',')}
```

  Here, the design matrix $\mathbf{\Phi}$ is loaded as `data_??['X']`, and target vector $\mathbf{t}$ is loaded as `data_??['t']`, where `??` is either `train` or `test`.

(a) Write the above 6 functions, and identify the correct order and arguments to do cross validation.

(b) Find the training and test errors corresponding to each $\lambda$ in `lambd_seq`. This part does not use the `cross_validation` function but you may find the other functions helpful.

(c) Plot training error, test error, and 5-fold and 10-fold cross validation errors on the same plot for each value in `lambd_seq`. What is the value of $\lambda$ proposed by your cross validation procedure? Comment on the shapes of the error curves.

*What to submit?*

*a) The functions your wrote.*
*b) Report the errors you find.*
*c) The plot containing 4 curves: i) training ii) test, iii) 5-fold CV iv) 10-fold CV errors, where x axis is lambda.*
*d) Your entire code should be attached to the end of your answers.*