Section 04: Train-Test Splitting, Generalized Least Squares Regression, MAP as Regularization

1. Biased Test Error

Is the test error unbiased for these programs? If not, how can we fix the code so it is?

1.1. Program 1

```
# Given dataset of 1000-by-50 feature
# matrix X, and 1000-by-1 labels vector

mu = np.mean(X, axis=0)
X = X - mu

idx = np.random.permutation(1000)
TRAIN = idx[0:900]
TEST = idx[900::]

ytrain = y[TRAIN]
Xtrain = X[TRAIN, :]

# solve for argmin_w ||Xtrain*w - ytrain||_2
w = np.linalg.solve(np.dot(Xtrain.T, Xtrain), np.dot(Xtrain.T, ytrain))

b = np.mean(ytrain)

ytest = y[TEST]
Xtest = X[TEST, :]

train_error = np.dot(np.dot(Xtrain, w)+b - ytrain,
              np.dot(Xtrain, w)+b - ytrain ) / len(TRAIN)
test_error = np.dot(np.dot(Xtest, w)+b - ytest,
              np.dot(Xtest, w)+b - ytest ) / len(TEST)

print('Train error = ', train_error)
print('Test error = ', test_error)
```

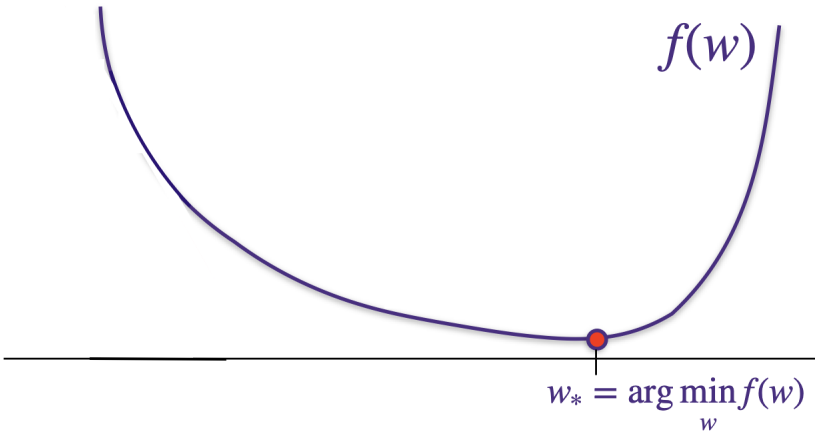
1.2. Program 2

```
# We are given: 1) dataset X with n=1000 samples and 50 features and 2) a vector y of length 1000 with labels.
# Consider the following code to train a model, using cross validation to perform hyperparameter tuning.

def fit(Xin, Yin, _lambda):
    w = np.linalg.solve(np.dot(Xin.T, Xin) + _lambda * np.eye(Xin.shape[1]), np.dot(Xin.T, Yin))
    b = np.mean(Yin) - np.dot(w, mu)
    return w, b

def predict(w, b, Xin):
    return np.dot(Xin, w) + b

idx = np.random.permutation(1000)
TRAIN = idx[0:800]
VAL = idx[800:900]
TEST = idx[900::]

ytrain = y[TRAIN]
Xtrain = X[TRAIN, :]
yval = y[VAL]
Xval = X[VAL, :]

# demean data
mu = np.mean(Xtrain, axis=0)
Xtrain = Xtrain - mu
Xval = Xval - mu

# use validation set to pick the best hyper-parameter to use
lambdas = [10 ** -5, 10 ** -4, 10 ** -3, 10 ** -2]
err = np.zeros(len(lambdas))

for idx, _lambda in enumerate(lambdas):
    w, b = fit(Xtrain, ytrain, _lambda)
    yval_hat = predict(w, b, Xval)
    err[idx] = np.mean((yval_hat - yval)**2)

lambda_best = lambdas[np.argmin(err)]

Xtot = np.concatenate((Xtrain, Xval), axis=0)
ytot = np.concatenate((ytrain, yval), axis=0)

w, b = fit(Xtot, ytot, lambda_best)

ytest = y[TEST]
Xtest = X[TEST, :]

# demean data
Xtest = Xtest - mu

ytot_hat = predict(w, b, Xtot)
train_error = np.mean((ytot_hat - ytot) **2)
ytest_hat = predict(w, b, Xtest)
test_error = np.mean((ytest_hat - ytest) **2)

print('Train error = ', train_error)
print('Test error = ', test_error)
```

2. Gradient Descent

Like we've seen in lecture, gradient descent is an important algorithm commonly used to train machine learning models, particularly useful for when there is no closed form solution for the minimum of a loss function. Here, we'll go through short introduction to the algorithm.

Consider some function $f(w)$, which has some w_* for which $w_* = \arg\min_w f(w)$:



Let w_0 be some initial guess for the minimum of $f(w)$. Gradient descent will allow us to improve this solution.

(a) For some w that is very close to w_0, give the Taylor series approximation for $f(w)$ starting at $f(w_0)$.

(b) Now, let us choose some $\eta > 0$ that is *very small*. With this very small η, let's assume that $w_1 = w_0 - \eta \left(\frac{df(w)}{dw} \Big|_{w=w_0} \right)$. Using your approximation from part (a), give an expression for $f(w_1)$.

(c) Given your expression for $f(w_1)$ from part (b), explain why, if η is small enough and if the function approximation is a good enough approximation, we are guaranteed to move in the "right" direction closer to the minimum w_*.

(d) Building from your answer in part (c), write a general form for the gradient descent algorithm.

3. Generalized Least Squares Regression

In class, we've seen linear regression and ridge regression. Here, we consider a problem that generalizes both of these. As a reminder, in linear regression, we seek a model that captures a linear relationship between input data and output data. The general case we consider imposes additional structure on the model.

Consider an experiment in which you have n data points $x_i \in \mathbb{R}^d$ and corresponding n observations y_i. We wish to come up with a model $\omega \in \mathbb{R}^d$ that satisfies the following properties: first, the error $\sum_{i=1}^n (x_i^\top \omega - y_i)^2$ should be small; second, we don't want small changes in training data resulting in large changes in solution; third, we want to put different weights in controlling the magnitude of different coordinates of ω. We therefore define

$$\widehat{\omega}_{\text{general}} = \arg\min_{\omega} \sum_{i=1}^{n} (y_i - x_i^\top \omega)^2 + \lambda \sum_{i=1}^{d} D_{ii} \omega_i^2.$$

Here, D is a diagonal matrix, with positive entries on the diagonal. Observe that when D is the identity matrix, we recover ridge regression, and when $\lambda = 0$, we recover least squares regression. Different weights on D_{ii} cause the magnitudes of ω_i to be controlled differently.

3.1. Closed form in the general case

Deduce the closed form solution for $\widehat{\omega}_{\text{general}}$. You should be comfortable with proofs in the "coordinate" form as well as the "matrix" form.

3.2. Special cases: linear regression and ridge regression

(a) In the simple least squares case ($\lambda = 0$ above), what happens to the resulting $\widehat{\omega}$ if we double all the values of y_i?

(b) In the simple least squares case ($\lambda = 0$ above), what happens to the resulting $\widehat{\omega}$ if we double the data matrix $X \in \mathbb{R}^{n \times d}$?

(c) Suppose $D = I$ (that is, it is the identity matrix). That is, this is the *ridge* regression setting. Explain why $\lambda > 0$ ensures that the solution exists and the matrix can be inverted.

4. MAP as Regularization

Recall the regularization techniques that were presented in class this week and ponder their objectives:

(a) **Ridge-Regression**: $\hat{w}_{ridge} = \arg\min_w \sum_{i=1}^{n} (y_i - x_i^\top w)^2 + \lambda \|w\|_2^2$

(b) **LASSO**: $\hat{w}_{LASSO} = \arg\min_w \sum_{i=1}^{n} (y_i - x_i^\top w)^2 + \lambda \|w\|_1$

Reminder: don't ever regularize your bias term. This term doesn't add any complexity to the model (since it just shifts), so we'd like it to take on any value that best fits our training data.

The two types of regularization above can be derived from a statistical perspective in which we assume some prior belief about what the weights of our model should be and then observe data to further update the belief.

More specifically, let w denote our weights and Y, X our data(Y represents the labels and X the inputs). As before, $p(X, Y|w)$ represents the **likelihood function**. We specify our belief of what the weights should be through a **prior distribution** over $p(w)$. Using Bayes' Rule, we can write our updated belief of what the weights ought to be after observing the data as:

$$p(w|X, Y) = \frac{p(X, Y|w)p(w)}{p(X, Y)} = \frac{p(X, Y|w)p(w)}{\int_{w'} p(X, Y|w')p(w')dw'}$$

where we call $p(w|X, Y)$ the **posterior distribution** and $p(X, Y)$ the **evidence**.

What **Maximum A Posteriori Estimation(MAP)** does is compute the weights which maximize the posterior distribution, $p(w|X, Y)$. This type of estimation differs from MLE (which maximizes the likelihood function $p(X, Y|w)$) by taking into account our prior belief of what the weights are, namely $p(w)$. More specifically, the MAP estimate is:

$$\begin{aligned}
\hat{w}_{MAP} &= \arg\max_w p(w|X, Y) \\
&= \arg\max_w \frac{p(X, Y|w)p(w)}{p(X, Y)} \\
&= \arg\max_w p(X, Y|w)p(w) \\
&= \arg\max_w \log p(X, Y|w) + \log p(w)
\end{aligned}$$

where we dispose of the denominator because it doesn't depend on w. Contrast this with the MLE which is:

$$\hat{w}_{MLE} = \arg\max_w p(X, Y|w)$$

Let us now study how we can obtain the Ridge and LASSO regression objectives from this perspective:

(a) Suppose the elements of w are independently distributed according to a Laplacian distribution:

$$p(w_i) = \frac{\lambda}{4\sigma^2} \exp(-|w_i| \frac{\lambda}{2\sigma^2}).$$

Show that under this prior on w, MAP estimation of the linear measurement model recovers the LASSO objective.

(b) Derive an expression for the prior on w that corresponds to the ridge regression objective. What is the significance of this result?