ML Assignment

CWID: A20551908

0.1 Normal Equations

(1) The normal equation we used:

$$C = (A^T A)^{-1} A^T y$$

The pseudoinverse we used:

$$C = A^+ y$$

defined by $A^+ = V\Sigma^+ U^T$ where $A = U\Sigma V^T$ is singular decomposition of A, and Σ^+ is the diagonal matrix with positive elements $\{\sigma_j^{-1}\}$

The Single value Decomposition of a matrix:

SVD of mxn matrix A is given by $A = U\Sigma V^T$
where,

→ U is mxm matrix of the orthonormal vectors, i.e, orgthogonal matrix

→ V is nxn matrix of the orthonormal vectors, i.e, orthogonal matrix.

∴ V^T is transponse of a nxn matrix which is an orthogonal matrix.

→ Σ is diagonal matrix with positive elements.

It is given that Σ^+ is diagonal matrix with positive elements (σ_j^{-1})

$$\Sigma^+ = \begin{bmatrix} 1/\sigma_j & 0 & 0 & \cdots & 0 \\ 0 & 1/\sigma_j & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & \cdots & 1/\sigma_j \end{bmatrix}$$

As Σ is diagonal, we know using SVD of A.

$$\Sigma = \begin{bmatrix} \sigma_j & 0 & 0 & \cdots & 0 \\ \vdots & & & & \vdots \\ 0 & 0 & 0 & 0 \cdots 0 & 1 \end{bmatrix}$$

As Σ also has positive elements we can relate Σ^+ and Σ as follows.

$$\Sigma = \begin{bmatrix} \sigma_j & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 0 & 0 & \cdots & \sigma_j \end{bmatrix}$$ we get, $\Sigma^+ = \Sigma^{-1}$ $[\because (\sigma_j)^{-1} = 1/\sigma_j]$

we have to show,

$$A^+ = (A^T A)^{-1} \cdot A^T$$

RHS -

$(A^T \cdot A)^{-1} \cdot A^T$

lets first compute $A^T \cdot A$

we known $A = U\Sigma V^T$ — ①

Substituting ① we get

$$A^T \cdot A \Rightarrow (U\Sigma V^T)^T \cdot (U\Sigma V^T)$$

$$\Rightarrow (V\Sigma^T U^T)(U\Sigma V^T) \quad [(AB)^T = B^T A^T]$$

$$\Rightarrow V\Sigma^T (U^T U)\Sigma V^T$$

$$\Rightarrow V\Sigma^T (I)\Sigma V^T \quad [A^T \cdot A = I]$$

$$\Rightarrow V\Sigma^T \Sigma V^T \quad [I \cdot A = A]$$

$$\Rightarrow V(\Sigma^T \Sigma)V^T$$

we get $A^T \cdot A = V(\Sigma^T \Sigma)V^T$ — ②

By substituting ②, ① in $(A^T \cdot A)^{-1} \cdot A^T$

we get,

$$(A^T \cdot A)^{-1} \cdot A^T = (V(\Sigma^T \Sigma)V^T)^{-1} \cdot (U\Sigma V^T)^T$$

$$= (V(\Sigma^T \Sigma)^{-1} V^{-1})(V\Sigma^T U^T)$$

$$= V(\Sigma^T \Sigma)^{-1} \cdot \frac{1}{V} \cdot V\Sigma^T U^T$$

$$= V(\Sigma^T \Sigma)^{-1} \cdot \Sigma^T U^T$$

$$= V \cdot \frac{1}{\Sigma^T \cdot \Sigma} \cdot \Sigma^T U^T$$

$$= V \cdot \frac{1}{\Sigma \cdot \Sigma} \cdot \Sigma U^T$$ [as Σ is diagonal transpose of diagonal matrix is also diagonal $\therefore \Sigma^T = \Sigma$]

$$= V \frac{1}{\Sigma} U^T \quad = V \cdot (\Sigma\Sigma)^{-1} \cdot \Sigma U^T \quad [\Sigma^{-1} \cdot \Sigma = I]$$

$$= V\Sigma^{-1} U^T$$

we already known by relation between Σ and Σ^+ as

$$\Sigma^+ = \Sigma^{-1}$$

By substituting this

$$V\Sigma^{-1} U^T \Rightarrow V\Sigma^+ U^T = A^+$$

Hence RHS = LHS, proved

Q.2 Reformulate Ridge and LASSO

① The defined ridge regularization by the minimizer -

$$L(c) = \|y - Ac\|_2^2 + \lambda \|c\|^2$$

and constrained minimization problem

$$L(c) = \|y - Ac\|^2$$

$$\|c\|^2 \leq t$$

In constrained minimization problem, we introduce a Lagrange multiplier λ to enforce the constraint. The lagrangian is defined as:

$$L(c, \lambda) = \|y - Ac\|^2 + \lambda(\|c\|^2 - t)$$

we minimize the lagrangian with respect to c and λ to find optimal solution.

Let's find relationship between λ and t,

To do so we take the derivative of the lagrangian with respect to c and set it to zero.

To differentiate $\|y - Ac\|^2$ with respect to c

let $r = y - Ac$. Then $\|r\|^2$ is given by the sum of the squares of the components of r:

$$\|r\|^2 = \sum_{i=1}^{n} r_i^2$$ [n is dimensionality of r (and thus also of c)]

Now, lets find derivative of $\|r\|^2$ w.r.t c, since $r = y - Ac$, we have:

$$\frac{\partial \|r\|^2}{\partial c} = \frac{\partial}{\partial c} \sum_{i=1}^{n} r_i^2$$

using chain rule,

$$\frac{\partial \|r\|^2}{\partial c} = \sum_{i=1}^{n} \frac{\partial (r_i^2)}{\partial r_i} \cdot \frac{\partial r_i}{\partial c}$$

Since $r_i = y_i - \sum_{j=1}^{m} A_{ij} c_j$, m is the number of features, we differentiate each term with respect to r_i and then c:

$$\frac{\partial (r_i)^2}{\partial r_i} = 2 r_i, \quad \frac{\partial (r_i)}{\partial c} = -A_{ij}$$

Substituting these in expression,

$$\frac{\partial \|r\|^2}{\partial c} = \sum_{i=1}^{n} 2 r_i \cdot (-A_{ij}) = -2 \sum_{i=1}^{n} r_i A_{ij} = -2 (A^T r)_j$$

∴ The derivation of $\|y - Ac\|^2$ with respect to c is:

$$\frac{\partial \|y - Ac\|^2}{\partial c} = -2 A^T (y - Ac)$$

$$-2A^T(y-Ac)+2\lambda c=0$$

$$-2A^Ty+2A^TAc+2\lambda c=0$$

$$2A^TAc+2\lambda c=2A^Ty$$

$$A^TA_c+\lambda c=A^Ty$$

$$A^TAc+\lambda Ic=A^Ty \quad [I \text{ is the identity matrix}]$$

$$c(A^TA+\lambda I)=A^Ty$$

$$c=A^Ty\,(A^TA+\lambda I)^{-1}$$

The constrain $\|c\|^2 \le t$ implies:

$$c^Tc \le t$$

Substituting value of c,

$$((A^Ty(A^TA+\lambda I)^{-1})^T\,(A^Ty(A^TA+\lambda I)^{-1}) \le t$$

Expand using matrix multiplication.

$$(y^TA\,(A^TA+\lambda I)^{-1})((A^TA+\lambda I)^{-1}A^Ty) \le t$$

simplifies to,

$$y^TA\,(A^TA+\lambda I)^{-2}A^Ty \le t$$

Now, let $A^TA=Q$ then the inequality,

$$y^T(A^TA+\lambda I)^{-2}AA^Tyy^T \le t$$

$$y^T(Q+\lambda I)^{-2}Qy \le t.$$

we see that λ and t are inversely related, as t increases, λ decreases. This ensures that as we impose a stricter constraint on the norm of c, we correspondingly reduce the impact of the regularization term $\lambda\|c\|^2$ in objective function. Therefore, the Lagrange multiplier λ and the constraint t are related in such a way that adjusting one affects the other, ensuring the equivalence between ridge regularization and the constrained minimization problem.

(2) The initial definition of LASSO regression is:

$$L(c) = \|y - Ac\|_2^2 + \lambda \|c\|_1$$

we have to show its equivalence to the constrained minimization problem:

$$L(c) = \|y - Ac\|_2^2$$

such that $\|c\|_1 \leq t$

we can solve the problem using lagrange multipliers:

$$L(c, \alpha) = \|y - Ac\|_2^2 + \alpha(\|c\|_1 - t)$$

$\alpha \rightarrow$ lagrange multiplier

Now, we take derivation of L with respect to c and set it equal to zero.

$$\frac{\partial L}{\partial c} = -2A^T (y - Ac) + \alpha \cdot \text{sign}(c) = 0$$

we get,

$$A^T Ac - A^T y + \alpha \cdot \text{sign}(c) = 0$$

we see that solving directly for c is not affective as due to non-differentiability of l_1 norm. some, solution contains some elements set to zero. The non-zero elements will satisfy

$$c_i = \frac{A_i^T (y - Ac + \alpha/2)}{\|A_i\|_2^2}$$ 'A_i' $\rightarrow$ i-th column of A

considering l_1 norm constraint is active, $\|c\|_1 = t$, we get:

$$\sum_{i=1}^{n} |c_i| = t$$

Once we have the relationship between α and t, we can consider how t and λ are related from the original LASSO formulation:

$$\|y - Ac\|_2^2 + \lambda \|c\|_1$$

In constrained minimization problem, the lagrange multiplier α is related to constraint t by enforcing the condition that l_1 norm of c equals t.

$$\sum_{i=1}^{n} |c_i| = t$$

For establishing relationship between α and λ, we need to compare the effect of these parameters on the solution c.

* In constrained minimization problem, ~~at~~ larger magnitudes of c_i are possible in the restricted minimization problem because the l_1 norm constraint ~~loose~~ decreases as t grows. This shows that the solution c tends to have bigger absolute values as t grows.

* According to the LASSO formulation, the penality for greater absolute values of c_i grows as λ increases. As a result, the solution becomes increasingly sparse and more parts of c tends to be pushed towards zero.

As a result, we see that the solution tends to have more ~~absolute values~~ elements pulled towards ~~zero~~ zero as λ rises and absolute values of λ increases.

we can say that there exists a relationship between t and λ such that as t decreases, λ should increase to achieve similar effect on the sparsity of solution and vice versa. In practice, adjusting one parameter will often necessitate adjusting the other to maintain a similar level of regularization or sparsity.

0.3 Naive Bayes

① Key assumptions that make the Naive Bayes model naive are -

→ It presumes that predicators in Naive Bayes model to be conditionally independent, or not related to any of the other in the model.

→ It also presumes that all features contribute equally to the outcome.

② Given,

Benign Python scripts in data set = 9500
Malicious in data set = 500

Bengin scripts that import the ast library = 200
Malicious samples that import the ast library = 100

given that script is imports ast library, the probability that it is malicious } = ?

Let,

A → event that script is malicious
B → event that script imports the ast library

$$P(A) = \frac{500}{9500+500} = \frac{500}{10000} = 0.05$$

$$P(B) = \frac{(200+100)}{9500+500} = \frac{300}{10000} = 0.03$$

$$P(B|A) = \frac{100}{500} = 0.2$$ [Probability that script imports ast library given that it is malicious]

P(A|B) = probability of malicious given that script imports the ast library

using Baye's theorem,

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} = \frac{0.2 \times 0.05}{0.03} = \frac{0.01}{0.03} = 0.33$$

Probability of script malicious given it is imports ast library is ≃ 0.33 ≃ 33.3%

In [4]:
```python
import numpy as np
import pandas as pd
from scipy.special import expit  # sigmoid function

# List of file names
files = ['data1.csv', 'data12.csv']

# Loop through each file
for i, files in enumerate(files, start = 1):
    print(f"Processing file {i}: {files}")

    df = pd.read_csv(files)  # Replace 'your_file.csv' with the actual file path
    info = df.loc[:, ~df.columns.str.contains('^Unnamed')]

    # X refers to features  y target values
    X = info.iloc[:, :-1].values
    y = info.iloc[:, -1].values

    # Add a column of ones to the features for the bias term
    X = np.c_[np.ones(X.shape[0]), X]

    # Initialize parameters
    coefficient = np.zeros(X.shape[1])
    lr = 0.001
    lambda_parameter = 0.01
    num_of_iterations = 1000

    # Logistic sigmoid function
    def sigmoid(z):
        return 1 / (1 + np.exp(-z))

    # Regularized log loss function
    def log_loss(coefficient, X, y, lambda_parameter):
        m = len(y)
        h = sigmoid(np.dot(X, coefficient))
        h = np.clip(h, 1e-15, 1 - 1e-15)
        loss = -np.sum(y * np.log(h) + (1 - y) * np.log(1 - h)) / m
        reg_term = (lambda_parameter) * np.sum(coefficient[1:]**2)  # Exclude bias term
        return loss + reg_term
```

```python
# Gradient of regularized log loss function
def gradient_ridge(coefficient, X, y, lambda_parameter):
    m = len(y)
    h = sigmoid(np.dot(X, coefficient)) # h= prediction, errors = h-y
    h = np.clip(h, 1e-15, 1 - 1e-15)
    gradient = np.dot(X.T, (h - y)) / m
    reg_term = 2 * (lambda_parameter) * np.r_[0, coefficient[1:]]  # Include 0 for bias term
    return gradient + reg_term

# Gradient descent for regularized log loss
def gradient_descent(X, y, coefficient, lr, lambda_parameter, num_of_iterations):
    for itr in range(num_of_iterations):
        gradient = gradient_ridge(coefficient, X, y, lambda_parameter)
        coefficient -= lr * gradient
        if itr % 100 == 0:
            loss = log_loss(coefficient, X, y, lambda_parameter)
            print(f"Iterations {itr}, Loss: {loss}")
    return coefficient, loss

def accuracy(X, coefficient):
    prob = sigmoid( np.dot(X, coefficient))
    return (prob >= 0.5).astype(int)

# Apply gradient descent
optimal_coefficient, loss = gradient_descent(X, y, coefficient, lr, lambda_parameter, num_of_iterations)


# Display the optimal parameters
print(f"Optimal Parameters for file {i}:")
print(optimal_coefficient)
print(loss)
predict = accuracy(X, optimal_coefficient )
accur = np.mean(predict == y) * 100
print("accuracy", accur)
```

```
Processing file 1: data1.csv
Iterations 0, Loss: 0.6921578814351956
Iterations 100, Loss: 0.6036875220827
Iterations 200, Loss: 0.5328792593561658
Iterations 300, Loss: 0.4757958298479238
Iterations 400, Loss: 0.4293254288542837
Iterations 500, Loss: 0.39108166475527806
Iterations 600, Loss: 0.35925860670069737
Iterations 700, Loss: 0.33249344569447037
Iterations 800, Loss: 0.30975420466501935
Iterations 900, Loss: 0.2902538888858622
Optimal Parameters for file 1:
[-0.00167865  0.62533039]
0.2902538888858622
accuracy 99.4
Processing file 2: data12.csv
Iterations 0, Loss: 0.6566374516953546
Iterations 100, Loss: 0.45326291531055024
Iterations 200, Loss: 0.43167547786865235
Iterations 300, Loss: 0.4211472314078763
Iterations 400, Loss: 0.41469246778214774
Iterations 500, Loss: 0.4102466396022887
Iterations 600, Loss: 0.4069559235034316
Iterations 700, Loss: 0.4044018176202778
Iterations 800, Loss: 0.4023538994011773
Iterations 900, Loss: 0.40067351132005397
Optimal Parameters for file 2:
[-4.23320799e-03 -1.46718463e-02 -2.59287926e-02  1.69443862e-02
  1.56859500e-02 -2.53574409e-02  1.65280461e-03  3.00100906e-02
  2.27598968e-02 -2.78708799e-02 -5.62695219e-04  2.26501377e-02
  1.19135994e-04  1.27860168e-02  4.23341293e-02 -3.36307222e-02
 -1.23978550e-01 -8.20457720e-04  4.34018311e-03  4.03055736e-02
 -3.70270967e-02 -5.15627061e-02  1.34981925e-02 -8.88870457e-02
  6.41628389e-02  2.04456469e-02 -9.64825263e-03 -3.62425371e-02
  1.06648160e-02 -2.05962143e-02 -4.42991593e-02 -3.60470639e-03
  9.72366020e-02  8.87165520e-03  6.98414452e-02  6.09968416e-02
  6.60976901e-03 -3.25728858e-02  3.83242123e-04 -6.41335733e-02
 -3.75143470e-02  5.37671181e-02 -2.04381461e-02 -8.38334853e-03
  2.62010169e-02 -1.08942136e-02 -1.55112891e-04 -3.77175700e-02
 -1.40653926e-02  1.76542966e-02  1.23180306e-02]
```

```
0.40067351132005397
accuracy 82.19999999999999
```

In [ ]: