1

when y = 1

$$ERM: \sum log(1+e^{-w^Tx})$$

$$NLL: -\sum \frac{1}{log(1+e^{-w^Tx})} = \sum log(1+e^{-w^Tx})$$

when y = 0, -1

$$ERM: log(1+e^{w^Tx})$$

$$NNL: -log(1-\frac{1}{1+e^{-w^Tx}}) = -log(\frac{e^{-w^Tx}}{1+e^{-w^Tx}})$$

$$= log(\frac{1}{e^{-w^Tx}}+1) = log(1+e^{w^Tx})$$

2

$$log\frac{P(y=1|x)}{P(y=0|x)} = log\frac{1}{1+e^{-x^Tw}}$$

becasue =$P(y-1|x) = P(y=0|x)$ ont the boundary

$$\frac{1}{1+e^{-x^Tw}} = 1 \Rightarrow -x^Tw = 0$$

3

$$\frac{\partial \ell^n}{\partial c} = \frac{\partial \ell^n}{\partial f^b}\frac{\partial f^n}{\partial c}$$

$$= (\frac{y^n}{f^n} - \frac{1-y^n}{1-f^n})\frac{\partial f^n}{\partial c}$$

$$= (\frac{y^n}{f^n} - \frac{1-y^n}{1-f^n})(f^n(1-f^n)x_i^n)$$

$$= (y^n - f^n)x_i^n$$

$$(y - \frac{1}{1+e^{1+e^{-cw^Tx}}})w^Tx^n$$

4

$$f(w) = log(1+exp(-yw^Tx))$$

$$f'(w) = \frac{1}{1+exp(-yw^Tx)}exp(-yw^Tx)(-yx)$$

$$= \frac{-yx}{exp(-yw^Tx)+1}$$

$$f''(w) = -yx\frac{(-exp(-yw^Tx)+1)dw}{(-exp(-yw^Tx)+1)^2} = y^2x^2\frac{exp(yw^Tx)}{(-exp(-yw^Tx)+1)^2} \geq 0$$

5

```
def f_objective(theta, X, y, l2_param=1):
    n = X.shape[0]
    o = np.logaddexp(0,(-np.dot(X,theta.T) * y))
    return (1/n) * np.sum(o) + l2_param * np.sum(theta**2)
```

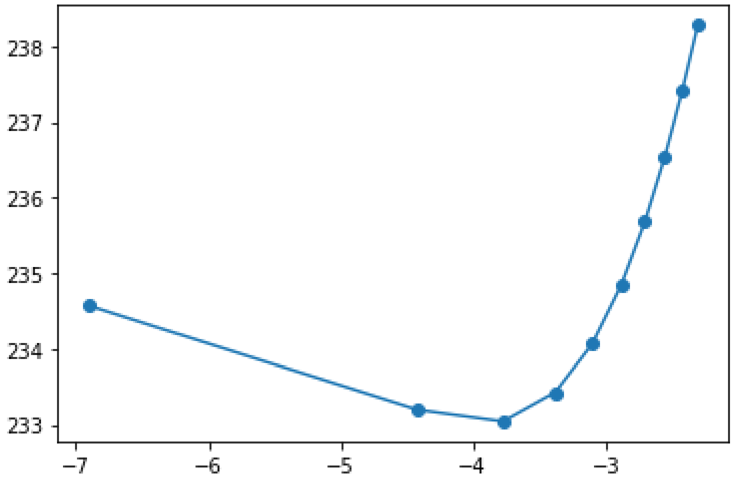
6

```
def fit_logistic_reg(X, y, objective_function, l2_param=1):
    w_0 = np.zeros(X.shape[1])
    return minimize(objective_function, w_0, (X, y, l2_param)).x
```

7

```
# y=0 to y=-1
for i in range(len(y_train)):
    if y_train[i] == 0:
        y_train[i] = -1

for i in range(len(y_val)):
    if y_val[i] == 0:
        y_val[i] = -1

# normalize
n = X_train.shape[0]
X_mean = np.mean(X_train, axis=0)
X_std = np.std(X_train, axis=0)

X_train_norm = (X_train-X_mean)/X_std
X_val_norm = (X_val-X_mean)/X_std

train_bias_term = np.ones(X_train.shape[0]).reshape(X_train.shape[0], 1)
val_bias_term = np.ones(X_val.shape[0]).reshape(X_val.shape[0], 1)
X_train = np.hstack((train_bias_term, X_train_norm))
X_val = np.hstack((val_bias_term, X_val_norm))

X_train.shape, y_train.shape, X_val.shape, y_val.shape
---
((1600, 21), (1600,), (400, 21), (400,))
```

```
# 7
lambda_list = np.linspace(0.001, 0.1, 10)
result_list = []
for l in lambda_list:
    theta = fit_logistic_reg(X_train, y_train, f_objective, l2_param=l)
    nll = (f_objective(theta, X_val, y_val, l) - l * theta.T @ theta) *
len(y_val)
    result_list.append(nll)
plt.plot(np.log(lambda_list), result_list, 'o-')
---
```

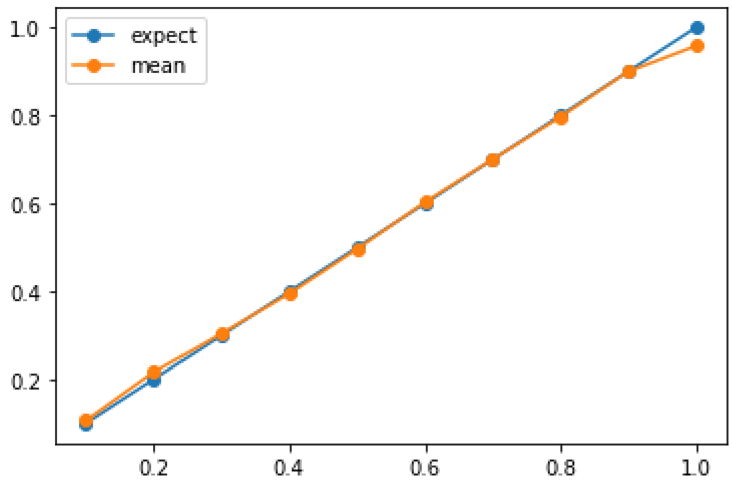



```
np.linspace(0.001, 0.1, 10)
---
array([0.001, 0.012, 0.023, 0.034, 0.045, 0.056, 0.067, 0.078, 0.089,       0.1
])
```

Choose 0.023

8

```
# 8
w = fit_logistic_reg(X_train, y_train, f_objective, l2_param=0.023)
val_prob = 1 / (1 + np.exp(-1 * X_val @ w))
bins = np.arange(0.05, 1.1, 0.1)
indices = np.digitize(val_prob, bins)
pred_prob = np.zeros(10)
count = np.zeros(10)
for i in range(len(indices)):
    pred_prob[indices[i] - 1] += val_prob[i]
    count[indices[i] - 1] += 1
mean_prob = np.zeros(10)
for i in range(10):
    mean_prob[i] = pred_prob[i] / count[i]
expect_prob = np.arange(0.1, 1.1, 0.1)
plt.plot(expect_prob, expect_prob, marker='o', label='expect')
plt.plot(expect_prob, mean_prob, marker='o', label='mean')
plt.legend()
plt.show()
```



9

$$
\begin{gathered}
P(x=H|\theta_1,\theta_2) \\
=P(X=H,Z=H|\theta_1,\theta_2)+P(X=H,Z=T|\theta_1,\theta_2) \\
=P(X=H|Z=H,\theta_2)P(Z=H|\theta_1)+P(X=H|Z=T,\theta_2)P(Z=T|\theta_1) \\
\xRightarrow{P(X=H|Z=T)=0} \theta_2\theta_1
\end{gathered}
$$

10

$$
\begin{gathered}
P(x=T|\theta_1,\theta_2) \\
=P(X=T,Z=T|\theta_1,\theta_2)+P(X=T,Z=H|\theta_1,\theta_2) \\
=P(X=T|Z=T,\theta_2)P(Z=T|\theta_1)+P(X=T|Z=H,\theta_2)P(Z=H|\theta_1) \\
(1-\theta_1)+(1-\theta_2)\theta_1=1-\theta_1\theta_2
\end{gathered}
$$

$$
\ell(D_r)=(\theta_1\theta_2)^{n_h}(1-\theta_1\theta_2)^{n_t}
$$

11

No, can only estimate $\theta_1\theta_2$, the result should be $\frac{n_h}{n_{total}}$