Problem 2

[15]:
```
import numpy as np
spambase = np.loadtxt('spambase.data', delimiter=',')
```

[16]:
```
np.random.shuffle(spambase)
y = spambase[:, -1].astype(int)
X = np.zeros(np.array(spambase.shape) - np.array([0, 1]),␣
 ↪dtype=int)
X[spambase[:, :-1] > np.median(spambase[:, :-1], axis=0)[None, :]]␣
 ↪= 1
```

[17]:
```
X_train, y_train = X[:2000, :], y[:2000]
X_val, y_val = X[2000:, :], y[2000:]
```

[18]:
```
eta = y_train.sum() / 2000  # P[y=1]
theta = np.array([
    X_train[np.logical_not(y_train), :].sum(axis=0) / (eta*2000),
    # P[xi = 1 | y = 0] over i= 1,...,d
    X_train[y_train.astype(bool), :].sum(axis=0) / (eta*2000)
    # P[xi = 1 | y = 1] over i= 1,...,d
]).T
one_minus_theta = 1 - theta
# P[xi = 0 | y = 0], P[xi = 0 | y = 1]
```

[19]:
```
def y_hat(x):
    P = np.array([
        eta * theta[x.astype(bool), y].prod() * one_minus_theta[np.
 ↪logical_not(x), y].prod() for y in range(2)
    ])  # P[y] * prod(P[xi | y]) over y = 0, 1
    return P.argmax()
```

[20]:
```
y_test = np.array([
    y_hat(X[-i-1, :]) for i in range(X.shape[0] - 2000)
])[::-1]
error = np.abs(y_val - y_test).mean()
error
```

[20]:
```
0.08419838523644751
```

[21]:
```
y_naiv = int(y_train.mean() > 0.5)
error_naiv = np.abs(y_val - y_naiv).mean()
error_naiv
```

[21]:
```
0.4002306805074971
```