HW2

1.

- **(a)** The target variable could be the score of reviews.
- **(b)** We could count the frequency of occurrence of words like “bad”, “good”, or “doesn’t work”, and grade the words.
- First, Set the judgement score as b, the numeric score of review as x, and the promotion rate as y.
- Second, set the reviews that without a judgement to b=0.
- Positive reviews with the words such as “good”, “great” will be graded to b=1.
- And the negative reviews with words like “doesn’t work” will be graded to b=-1.
- $\bar{b} = mean(b)$
- Thus we could get a linear model as
  $\bar{y} = \bar{x} + \bar{b}$
- **(c)** We could multiply our x with a coefficient k=0.5, for those who obtain a numeric score from 1 to 10.
- **(d)** Under the first situation, we keep the features same as above.
- For those whose rating is simply good or bad, we could only get the mean of b to rate the promotion. Thus, we could add the past sales to our system. That is to say, we could have a feature containing both the past sales as well as reviews.
- The past sales rate could be graded as a rate r that how many percent customers purchased this specific product among all customers who purchased the same kind of product.
- Thus the model would be modified to
  $\bar{y} = \bar{b} + r$
- At end, if there is no numeric rating at all, then we have to sort the promotion sequence with the past sales rate talked about above.
- **(e)** I think I’m gonna use the fraction of reviews with the word ‘good’ as a predictor, because each specific product got different amount of buyers. It is not fair if we only judge its quality with the total review numbers.

2.

- **(a)** $y = \beta_0 + \beta_1 x_1 + \beta_2 x_2$
  For instance, $y_i = 3(x_{i1} + x_{i2}) + 1$

- **(b)**

```
In [2]: import numpy as np
        A=np.array([[1,0,0],[1,0,1],[1,1,0],[1,1,1]])
        y=np.array([1,4,3,7])
        beta=np.linalg.lstsq(A,y)[0]
        beta

Out[2]: array([ 0.75,  2.5 ,  3.5 ])
```

3.

- **(a)** There are M+N+1 values of β in total

(b) $y = \begin{bmatrix} y_M \\ y_{M+1} \\ \vdots \\ y_{T-1} \end{bmatrix}$ $A = \begin{bmatrix} y_{M-1} & y_{M-2} & \cdots & y_0 & x_M & \cdots & x_{M-N} \\ y_M & & & \vdots & x_{M+1} & & x_{M-N+1} \\ \vdots & & & \vdots & \vdots & & \vdots \\ y_{T-2} & \cdots & & y_{T-M-1} & x_{T-1} & & x_{T-N-1} \end{bmatrix}$

(c) $A = [A_Y \; A_X]$ $A_Y = \begin{bmatrix} y_{M-1} & \cdots & y_0 \\ y_M & & y_1 \\ \vdots & & \vdots \\ y_{T-2} & & y_{T-M-1} \end{bmatrix}$ $A_X = \begin{bmatrix} x_M & \cdots & x_{M-N} \\ x_{M+1} & & x_{M-N+1} \\ \vdots & & \vdots \\ x_{T-1} & & x_{T-N-1} \end{bmatrix}$

$$\frac{1}{T}A^TA = \frac{1}{T}\begin{bmatrix} A_Y^TA_Y & A_Y^TA_X \\ A_X^TA_Y & A_X^TA_X \end{bmatrix}$$

$$\frac{1}{T}(A_Y^TA_Y)_{ij} = \frac{1}{T}\sum_k (A_Y^T)_{ik}(A_Y)_{kj}$$

$$= \frac{1}{T}\sum_k (A_Y)_{ki}(A_Y)_{kj} = \frac{1}{T}\sum_k Y_{m+k-i}Y_{m+k-j} = \frac{1}{T}\sum Y_k Y_{k+j-j}$$

$$\approx R_{yy}(i-j)$$

Similarly $\frac{1}{T}(A_Y^TA_X) \approx R_{yx}(i-j)$. $\frac{1}{T}(A_X^TA_X) \approx R_{xx}(i-j)$

$\frac{1}{T}(A_X^TA_Y) \approx R_{xy}(i-j)$

$\therefore \frac{1}{T}(A_Y^Ty) = \frac{1}{T}\sum_k Y_{m+k-i}Y_{m+k} \approx R_{yy}(M-i)$.

4. (a).

$$x_k \approx \sum_{l=1}^{L} a_l \cos(\Omega_l k) + b_l \sin(\Omega_l k).$$

$$\begin{bmatrix} x_0 \\ \vdots \\ x_{N-1} \end{bmatrix} = \begin{bmatrix} \cos(\Omega_1(0)) & \cdots & \cos(\Omega_L(0)) & \sin(\Omega_1(0)) & \cdots & \sin(\Omega_L(0)) \\ \cos(\Omega_1(1)) & & & & & \\ \vdots & & \vdots & \vdots & & \vdots \\ \cos(\Omega_1(N-1)) & & \cos(\Omega_L(N-1)) & \sin(\Omega_1(N-1)) & & \sin(\Omega_L(N-1)) \end{bmatrix} \cdot \begin{bmatrix} a_1 \\ \vdots \\ a_L \\ b_1 \\ \vdots \\ b_L \end{bmatrix}$$

(b) If the frequencies Ω_l were not taken, the model is no longer linear.