Problem 1

(25 points)
In this question you will build a spam e-mail filter using K-Nearest Neighbour algorithm. Each e-mail has two features: number of words in the email and total occurrences of the spam words "credit" and "dollars". Following graph shows the train data where filled circles are spam e-mails and unfilled circles are non-spam emails.

a. How would 1-Nearest Neighbour algorithm classify an e-mail with 10 spam words in a total of 90 words? Explain your answer.

b. How would 3-Nearest Neighbour algorithm classify an e-mail with 25 spam words in a total of 500 words? Explain your answer.

c. How would 5-Nearest Neighbour algorithm classify an e-mail with 5 spam words in a total of 600 words? Explain your answer.

d. How would you choose the best $k$ value to use in a KNN problem?

e. Draw an example 2D binary classification data set that a KNN algorithm with a small $k$ value would perform better than a high $k$ value. Explain your reasoning.

f. Draw an example 2D binary classification data set that a KNN algorithm with a high $k$ value would perform better than a small $k$ value. Explain your reasoning.
Problem 2

(25 points)
In class on Thursday, we talked about neural networks and saw examples of multi-layered neural networks. Here, we consider a 2-layer neural network that takes inputs of dimension $d$, has a hidden layer of size $m$, and produces scalar outputs. To see an illustration of this neural network, refer to section 20.1 in the textbook. The network’s parameters are $W$, $b_1$, $v$, and $b_2$. $W$ is a $m \times d$ matrix, $b_1$ is an $m$-dimensional vector, $v$ is an $m$-dimensional vector, and $b_2$ is a scalar.

For an input $x$, the output of the first layer of the network is:
\[ h = \sigma(Wx + b_1) \]

and the output of the second layer is:
\[ z = v \cdot h + b_2, \]

where $\sigma$ is an activation function. For this question, let $\sigma$ be the sigmoid activation function $\sigma_{\text{sigmoid}}$ (in the formula below, we apply it element-wise):
\[ \sigma_{\text{sigmoid}}(a) = \frac{1}{1 + e^{-a}} \]

We will be using the following loss function:
\[ L(z) = (z - y)^2, \]

where $y$ is a real-valued label and $z$ is the network’s output.

To train this network, we use gradient descent and backpropagation. The relevant section is 20.6 in the textbook. In order to do backpropagation, we need to calculate various partial derivatives. This will be useful for understanding backpropagation, which will come in handy when we code it in hw9!

In this problem, you will calculate the partial derivative of $L(z)$ with respect to each of the network’s parameters. Let $w_{ij}$ be the entry at the $i$th row and $j$th column of $W$. Let $v_i$ be the $i$th component of $v$. Let $b_{1i}$ be the $i$th component of $b_1$. (Note that $1 \leq i \leq m$ and $1 \leq j \leq d$.) (Hint: For each part of the problem, apply the chain rule.)

a. Calculate $\frac{\partial L(z)}{\partial b_2}$. Show your work.

b. Calculate $\frac{\partial L(z)}{\partial v_i}$. Show your work.

c. Calculate $\frac{\partial L(z)}{\partial b_{1i}}$. Show your work.

d. Calculate $\frac{\partial L(z)}{\partial w_{ij}}$. Show your work.