1. (a)

$$\mathbb{E}[\mathcal{L}(y = \text{keep}, t)] = 0.1 \cdot 1 + 0.9 \cdot 0 = 0.1$$
$$\mathbb{E}[\mathcal{L}(y = \text{keep}, t)] = 0.1 \cdot 0 + 0.9 \cdot 100 = 90$$

(b)

$$\begin{aligned}&\mathbb{E}[\mathcal{L}(y = \text{keep}, t) \,|\, x] \\ =&1 \cdot \Pr(t = \text{spam} \,|\, x) + 0 \cdot (1 - \Pr(t = \text{spam} \,|\, x)) \\ =&\Pr(t = \text{spam} \,|\, x)\end{aligned}$$

$$\begin{aligned}&\mathbb{E}[\mathcal{L}(y = \text{remove}, t) \,|\, x] \\ =&0 \cdot \Pr(t = \text{spam} \,|\, x) + 100 \cdot (1 - \Pr(t = \text{spam} \,|\, x)) \\ =&100 \cdot (1 - \Pr(t = \text{spam} \,|\, x))\end{aligned}$$

If $\Pr(t = \text{spam} \,|\, x) \leq 100 \cdot (1 - \Pr(t = \text{spam} \,|\, x))$, then choose $y_* = \text{keep}$, else $y_* = \text{remove}$.

(c) By Bayes rule, we know that

$$\Pr(t = \text{spam} \,|\, (x_1, x_2)) = \frac{\Pr(t = \text{spam}) \cdot \Pr(x_1, x_2 \,|\, t = \text{spam})}{\Pr(x_1, x_2)}$$

$$\Pr(t = \text{spam} \,|\, (0, 0)) = \frac{0.4 \cdot 0.1}{0.1 \cdot 0.4 + 0.9 \cdot 0.998} = 0.043$$
$$\Pr(t = \text{spam} \,|\, (0, 1)) = \frac{0.3 \cdot 0.1}{0.1 \cdot 0.3 + 0.9 \cdot 0.001} = 0.971$$
$$\Pr(t = \text{spam} \,|\, (1, 0)) = \frac{0.2 \cdot 0.1}{0.1 \cdot 0.2 + 0.9 \cdot 0.001} = 0.957$$
$$\Pr(t = \text{spam} \,|\, (1, 1)) = \frac{0.1 \cdot 0.1}{0.1 \cdot 0.1 + 0.9 \cdot 0} = 1$$

According to part (b), $y_* = \begin{cases} \text{remove} & (x_1, x_2) = (1, 1) \\ \text{keep} & \text{else} \end{cases}$

(d)

$$\begin{aligned}\mathbb{E}[\mathcal{L}(y_*), t] = &\mathcal{L}(\text{keep}, \text{spam}) \cdot \Pr(t = \text{spam} \,|\, (0,0)) \cdot \Pr(\mathbf{x} = (0,0)) \\ &+ \mathcal{L}(\text{keep}, \text{spam}) \cdot \Pr(t = \text{spam} \,|\, (0,1)) \cdot \Pr(\mathbf{x} = (0,1)) \\ &+ \mathcal{L}(\text{keep}, \text{spam}) \cdot \Pr(t = \text{spam} \,|\, (1,0)) \cdot \Pr(\mathbf{x} = (1,0)) \\ &+ \mathcal{L}(\text{remove}, \text{non-spam}) \cdot \Pr(t = \text{non-spam} \,|\, (1,1)) \cdot \Pr(\mathbf{x} = (1,1)) \\ = &\, 1 \cdot 0.043 \cdot 0.9382 + 1 \cdot 0.971 \cdot 0.0309 + 1 \cdot 0.957 \cdot 0.0209 + 0 \\ = &\, 0.09\end{aligned}$$

2. (a) Suppose the dataset is linearly separable, since $x = -1$ and $x = 3$ both have label 1, any point between them should also have label 1. In particular, $x = 1$ should also have label $t = 1$ which leads to contradiction. Therefore, the data is not linearly separable.

   (b) From the feature map, we get

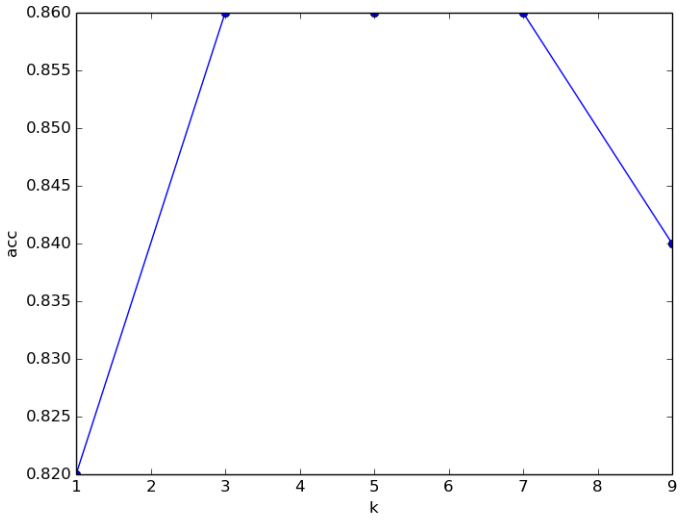
$$\psi(-1) = \begin{pmatrix} -1 \\ 1 \end{pmatrix}, \psi(1) = \begin{pmatrix} 1 \\ 1 \end{pmatrix}, \psi(3) = \begin{pmatrix} 3 \\ 9 \end{pmatrix}$$

   the constraint on w_1, w_2 is

$$\begin{cases} -w_1 + w_2 \geq 0 \\ w_1 + w_2 < 0 \\ 3w_1 + 9w_2 \geq 0 \end{cases}$$

   $\Rightarrow (w_1, w_2) = (-2, 1)$ correctly classify all the examples.

3.1. (a) here is the plot on validation set:



(b) The performance of knn on this data set is reasonably well, for many choice of k, the classification rate is over 85%.
My choice is $k^* = 5$, here is the test classification rate:

```
when k = 3, the test classification rate is 0.92
when k = 5, the test classification rate is 0.94
when k = 7, the test classification rate is 0.94
```

It's easy to see that the test performance of these values of k is better than the validation performance.

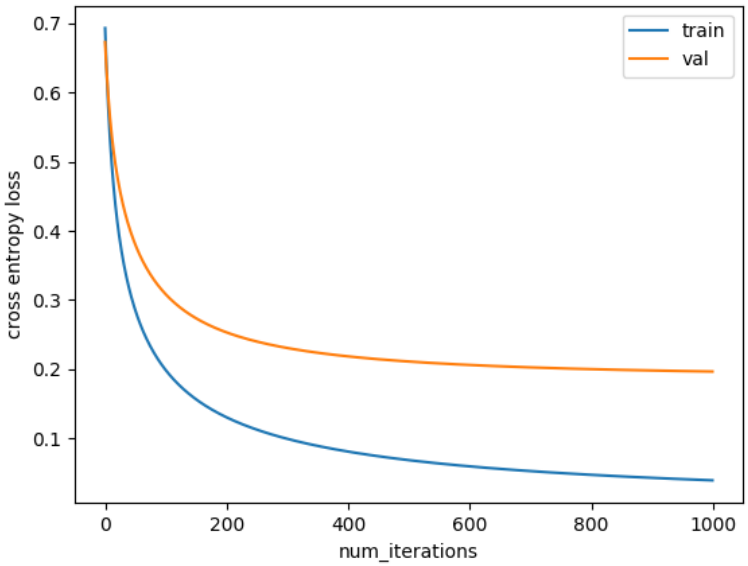
3.2. (b) For data set `mnist_train`:

```
{'learning_rate': 0.05, 'weight_regularization': 0.0, 'num_iterations': 1000}
On training set, the classification error is 0.00 with loss 0.04
On validation set, the classification error is 0.12 with loss 0.20
On test set, the classification error is 0.08 with loss 0.20
```

For data set `mnist_train_small`:

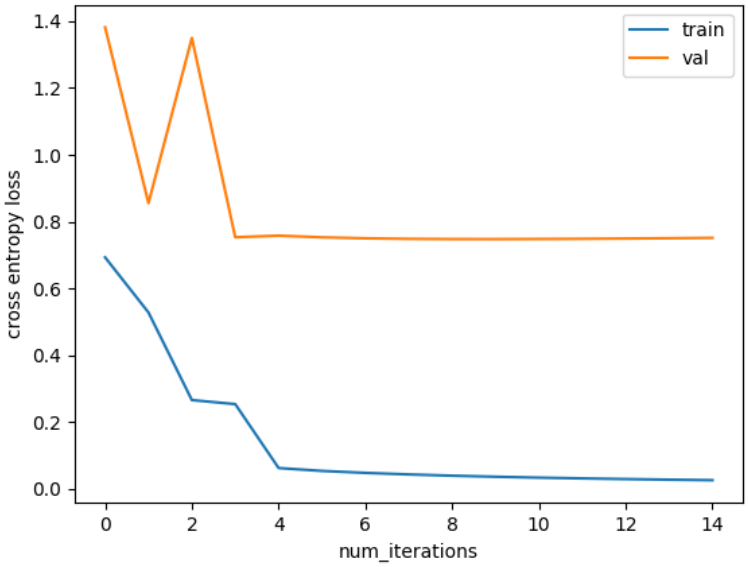
```
{'learning_rate': 0.5, 'weight_regularization': 0.0, 'num_iterations': 15}
On training set, the classification error is 0.00 with loss 0.02
On validation set, the classification error is 0.34 with loss 0.75
On test set, the classification error is 0.22 with loss 0.63
```

(c) For data set `mnist_train`:



`learning_rate = 0.05`, `iterations = 1000`
The higher the learning rate, the faster the graph converges. However, if the learning rate is too high, it might cause higher cross entropy loss. Low learning rate will take more iterations to converges.

For data set `mnist_train_small`:



`learning_rate = 0.5`, `iterations = 15`
Since it's a small data set, when the iterations is too large (over 20), the validation loss is increasing while the training loss is decreasing which indicates overfitting. If the learning rate is too high (over 2), it will lead numerical instability.

4. (a) Let $f = \frac{1}{2}\sum_{i=1}^{N} a^{(i)}(y^{(i)} - \mathbf{w}^T\mathbf{x}^{(i)})^2 + \frac{\lambda}{2}\sum_{i=1}^{N} w_i^2$, differentiate wrst. w_j

$$\begin{aligned}
\frac{\partial f}{\partial w_j} &= \sum_{i=1}^{N} a^{(i)}(y^{(i)} - \mathbf{w}^T\mathbf{x}^{(i)})(-x_j^{(i)}) + \lambda w_j \\
&= \sum_{i=1}^{N} a^{(i)}x_j^{(i)}(\sum_{k=1}^{D} w_k x_k^{(i)} - y^{(i)}) + \lambda w_j \\
&= \sum_{i=1}^{N} a^{(i)}x_j^{(i)} \sum_{k=1}^{D} w_k x_k^{(i)} - \sum_{i=1}^{N} a^{(i)}x_j^{(i)}y^{(i)} + \lambda w_j \\
&= \sum_{k=1}^{D} w_k \sum_{i=1}^{N} a^{(i)}x_j^{(i)}x_k^{(i)} - \sum_{i=1}^{N} a^{(i)}x_j^{(i)}y^{(i)} + \lambda w_j \\
&= \sum_{k=1}^{D} w_k(\lambda I(j=k) + \sum_{i=1}^{N} a^{(i)}x_j^{(i)}x_k^{(i)}) - \sum_{i=1}^{N} a^{(i)}x_j^{(i)}y^{(i)}
\end{aligned}$$

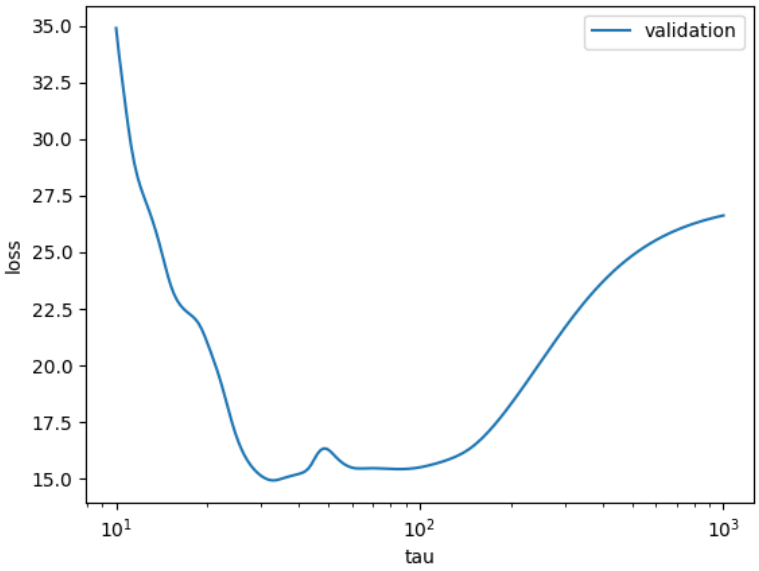
Let $A_{jk} = \lambda I(j=k) + \sum_{i=1}^{N} a^{(i)}x_j^{(i)}x_k^{(i)}$ and $c_j = \sum_{i=1}^{N} a^{(i)}x_j^{(i)}y^{(i)}$. Similar to A1 Q3, by observation,

$$\frac{\partial f}{\partial w_j} = \mathbf{B}\mathbf{w} - \mathbf{c} \overset{set}{=} 0$$

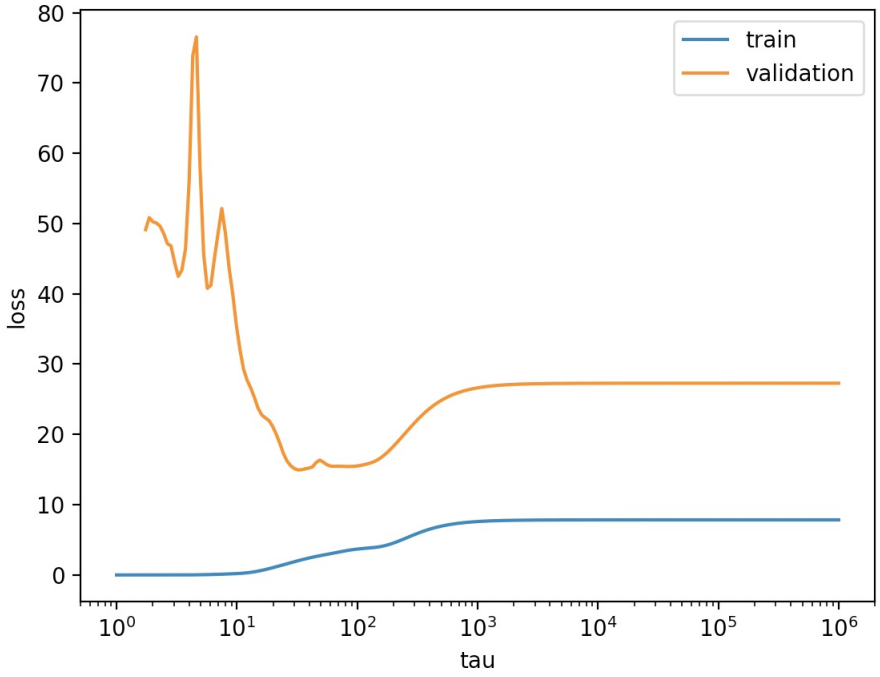
where $\mathbf{B} = \mathbf{X}^T\mathbf{A}\mathbf{X} + \lambda\mathbf{I}$ and $\mathbf{c} = \mathbf{X}^T\mathbf{A}\mathbf{y}$. Therefore,

$$\mathbf{w}^* = (\mathbf{X}^T\mathbf{A}\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{A}\mathbf{y}$$

(c) Here is the validation losses as a function of τ



(d) The validation loss will be high as $\tau \to \infty$ and $\tau \to 0$. When $\tau \to \infty$, the validation loss converges to a constant number. As $\tau \to \infty, a^{(i)} \to 1/N$ and the model becomes a linear regression model which is invariant of τ. As $\tau \to 0$, the training loss is almost 0 which indicates overfitting and causes high loss.



The plot, as τ ranges from 10^0 to 10^6, also supports this argument.