Q1 (a) In the class we showed that the margin is $\frac{2c}{\|W\|}$ and we want to maximize this margin.

The magnitude of c is merely scales ω and b in below

$$\max \frac{2c}{\|W\|} \quad \text{s.t. } y^i(W^T x^i + b) \geq c \ \forall i$$

and because c is constant we need to find the optimum value for $\frac{2c}{\|W\|}$, we can omit c and "2".

hence we can find the statement can be replaced by

$$\frac{1}{\|W\|} \quad \text{s.t. } y^i(W^T x^i + b) \geq 1 \ \forall i$$

Q1 (b) the Lagrangian dual Problem is

$$L(w,a,b) = \frac{1}{2} W W^T + \sum_{i=1}^{m} a_i \left(1 - y^i (W^T x^i + b)\right)$$

$$\frac{\partial L(w,a,b)}{\partial W} = W + 0 - \sum_{i=1}^{m} a_i y^i x^i + 0 = 0$$

$$= W - \sum_{i=1}^{m} a_i y^i x^i = 0 \Rightarrow \boxed{\omega = \sum_{i=1}^{m} a_i y^i x^i}$$

in the HW Question $n = m$ & $y^i = y_i$ & $x^i = x_i$.

Q2

(a):

x_i number of times word V_i appears in x

$n = 15$ total vocabulary in V

x is of length 15. this is the input data

$y=0$ when input data is NOT spam

$y=1$ when input data is spam

$P(y=0) = P(\text{NOT Spam}) = \frac{4}{7}$ $\begin{matrix} \rightarrow \text{NOT Spam} \\ \rightarrow \text{total data} \end{matrix}$

these are the prior

$P(y=1) = P(\text{Spam}) = \frac{3}{7}$

(b):

million dollar offer [0,1,0,0,0,0,0,1,1,0,0,0,0,0,0]

secret offer today [1,1,0,0,0,0,1,0,0,0,0,0,0,0,0]

seclet is secret [2,0,0,0,0,0,0,0,0,0,1,0,0,0,0]

low price for valued customer [0,0,1,1,1,1,0,0,0,0,0,1,0,0,0]

Play secret sports today [1,0,0,0,0,0,1,0,0,1,0,0,1,0,0]

sports is healthy [0,0,0,0,0,0,0,0,0,1,1,0,0,1,0]

low Price pizza [0,0,1,1,0,0,0,0,0,0,0,0,0,0,1]

(C) According to bayes rule $P(y|x) = \dfrac{P(x|y)\,P(y)}{P(x)}$

Based on naive bayes rule $\Rightarrow \underbrace{P(y|x)}_{\text{Posterior}} = \underbrace{P(x|y)}_{\text{likelihood}} \overbrace{P(y)}^{\text{Prior}}$

if $P(y=i|x) > P(y=j|x) \Rightarrow \underline{y=i}$ else $\Rightarrow y=j$

Because we assume independance for all data incidence
Then we can write the probability for data X to
be $y=1$ (spam) or $y=0$ (ham) can be written as:

$$P(y|X) = P(X,y) = P(x^1,y^1)\cdot P(x^2,y^2)\cdot\dots\cdot P(x^m,y^m)$$

$\boxed{\text{m is total number of records of data}}$

$$= \prod_{i=1}^{m} P(x^i,y^i)$$

$\boxed{\text{Likelihood}}$

$$\underset{\text{naive}}{=} \prod_{i=1}^{m} P(X^i|y^i)\,P(y^i)$$

(why in next page)*

$$= \prod_{i=1}^{m} P(y^i) \prod_{k=1}^{n} \theta_{c,k}^{x_k^i}$$

$\boxed{\text{we want to maximiz this equation. - this is constained optim}}$

for simplification we take the log (ln) from both sides.
log is monotonic so the maximization result will hold

the constrained is: $\sum_{k=1}^{n} \theta_{c,k} = 1 \quad \forall c$

* the answer to why from previous page:
we are looking for maximizing the $P(X|y)$ likelihood.

to max $P(X|y) = \frac{P(y|X)}{P(y)}$ we can maximiz $P(y|X)$

because $0 < P(y) < 1$. And it will vanish when taking FOC of lagrange.

Now the optimization problem, by taking log:

$$\log P(X,y) = \log \prod_{i=1}^{m} \left(P(y^i) \prod_{k=1}^{n} \theta_{c,k}^{x_k^i}\right) = \sum_{i=1}^{m} \log \left(P(y^i) \prod_{k=1}^{n} \theta_{c,k}^{x_k^i}\right) =$$

$$= \sum_{i=1}^{m} \left\{ \log (P(y^i)) + \sum_{k=1}^{n} \log \theta_{c,k}^{x_k^i} \right\}$$

with constaint $\sum_{k=1}^{n} \theta_{c,k} = 1 \quad \forall c$

$\theta_{c,k}^{x_k^i}$ is the likelihood of word x_k from sentence i that belongs to category c.

y^i is the catigory indicator. Here c is the category here it can be c

c is variable for category. here is 1 or 0

the lagrangian optimization problem will be:

$$L(P(X,y)) = \sum_{i=1}^{m} \left\{ \log (P(y^i)) + \sum_{k=1}^{n} \log \theta_{c,k}^{x_k^i} \right\} + \lambda \left(\sum_{k=1}^{n} \theta_{c,k} - 1 \right)$$

we'll take derivative from $\underline{\underline{a}}$ $\theta_{C,k}$ which pertains to $\underline{\underline{a}}$ C
this will make the derivative easier to see.

$$L(P(X,y)) = \sum_{i=1}^{m} \{ \log(P(y^i = c)) + \sum_{k=1}^{n} \log \theta_{c,k}^{x^i_k} \} + \lambda(\sum_{k=1}^{n} \theta_{c,k} - 1)$$

$$\frac{\partial (P(X,y))}{\partial \theta_{c,k}} = \sum_{i=1}^{m} \{ 0 + \frac{1}{\theta_{c,k}} 1^{x^i_{ck}} \} + \lambda = 0$$

Because we are differentiating for a particular $\theta_{c,k}$ hence only one term from $\lambda \sum_{k=1}^{n} \theta_{c,k}$ can be differenciated to 1.
the rest are constants.

$1^{x^i_{ck}}$: indicates the presence of word **k** in group c in sentence i
so if the word k exist in sentence i group c the value is $\underline{1}$ otherwise it is $\underline{0}$.

$$\frac{\partial (P(X,y))}{\partial \theta_{c,k}} = \frac{1}{\theta_{c,k}} \sum_{i=1}^{m} 1^{x^i_{ck}} + \lambda = 0 \Rightarrow \theta_{c,k} = \frac{-\sum_{i=1}^{m} 1^{x^i_{ck}}}{\lambda} \quad (*)$$

$$\text{since } \sum_{k=1}^{n} \theta_{c,k} = 1 \Rightarrow \sum_{k=1}^{n} \theta_{c,k} = \frac{-\sum_{k=1}^{n} \sum_{i=1}^{M} 1^{x^i_{ck}}}{\lambda} = 1$$

$$\Rightarrow \lambda = - \sum_{k=1}^{n} \sum_{i=1}^{m} 1^{x^i_{ck}}$$

$\Rightarrow$ replace λ in $(*)$

$$\theta_{c,k} = \frac{\sum_{i=1}^{m} 1^{x^i_{ck}}}{\sum_{k=1}^{n}\sum_{i=1}^{m} 1^{x^i_{ck}}}$$

→ this says count all word k in the category c

→ this says count all words from all sentences with category c

$\theta_{0,1} = \frac{3}{9\ (y=0)} = \frac{1}{3}$ $\quad\theta_{1,1} = \frac{1}{15}$

$\theta_{0,7} = \frac{1}{9}$ $\quad\theta_{1,15} = \frac{1}{15}$

(d) $P(\text{today is secret} \mid \text{spam}) = P((1,0,0,0,0,0,1,0,0,0,1,0,0,0,0) \mid \text{spam})$

$= \frac{3}{9} \times (1 - \frac{3}{9}) \times 1 \times 1 \times 1 \times 1 \times 1/9 \times (1 - 1/9) \times (1 - 1/9) \times 1 \times 1/9 \times 1 \times 1 \times 1 \times 1$

$= 0.0025$

	Spam	ham
secret	3/9	1/15
after	3/9	0
Low	0	2/15
Price	0	2/15
valued	0	1/15
customer	0	1/15
today	1/9	1/15
dollar	1/9	0
million	1/9	0
sports	0	2/15
is	1/9	1/15
for	0	1/15
Play	0	1/15

	Spam	ham
healty	0	1/15
Pizza	0	1/15

$P(\text{today is secret} \mid \text{ham}) = \frac{1}{15} \times 1 \times (1 - \frac{2}{15}) \times (1 - \frac{2}{15})$
$\times (1 - \frac{1}{15}) \times (1 - 1/15) \times 1/15 \times \times 1 \times 1 \times (1 - \frac{2}{15}) \times \frac{1}{15}$
$\times (1 - \frac{1}{15}) \times (1 - \frac{1}{15}) \times (1 - \frac{1}{15}) (1 - \frac{1}{15}) = 0.00013$

$P(\text{sham} \mid \text{today is secret}) = \frac{0.0025 \times 4/7}{0.0025 \times 4/7 + 0.000113 \times \frac{3}{7}}$
$= \frac{0.0014}{0.001497} = \boxed{0.967}$

$P(\text{ham} \mid \text{today is secret}) = \frac{0.00005591}{0.001497} = 0.037$

$\boxed{\text{It is Spam}}$

Q3

Q3-a-i

When I ran the classification with the test data (20%), all three classifiers performed the same. As shown below.

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l2', random_state=None, solver='warn',
          tol=0.0001, verbose=0, warm_start=False)
              precision    recall  f1-score   support

         0.0       0.93      1.00      0.97        14
         1.0       1.00      0.95      0.97        20

   micro avg       0.97      0.97      0.97        34
   macro avg       0.97      0.97      0.97        34
weighted avg       0.97      0.97      0.97        34

[[14  0]
 [ 1 19]]
```

```
GaussianNB(priors=None, var_smoothing=1e-09)
              precision    recall  f1-score   support

         0.0       0.93      1.00      0.97        14
         1.0       1.00      0.95      0.97        20

   micro avg       0.97      0.97      0.97        34
   macro avg       0.97      0.97      0.97        34
weighted avg       0.97      0.97      0.97        34

[[14  0]
 [ 1 19]]
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=None, n_neighbors=5, p=2,
           weights='uniform')
              precision    recall  f1-score   support

         0.0       0.93      1.00      0.97        14
         1.0       1.00      0.95      0.97        20

   micro avg       0.97      0.97      0.97        34
   macro avg       0.97      0.97      0.97        34
weighted avg       0.97      0.97      0.97        34

[[14  0]
 [ 1 19]]
```

But then I ran the classification on the entire data, the I noticed that logistic regression had higher precision. I believe that data in the divorce dataset is linearly separable and therefore logistic regression performed the best and Gaussian Naïve Bayes and KNN performed slightly worse than logistic regression.

Logistic Regression

	precision	recall	f1-score	support
0.0	0.99	1.00	0.99	86
1.0	1.00	0.99	0.99	84
micro avg	0.99	0.99	0.99	170
macro avg	0.99	0.99	0.99	170
weighted avg	0.99	0.99	0.99	170

```
[[86  0]
 [ 1 83]]
```

Guassian Naïve Bayes

	precision	recall	f1-score	support
0.0	0.96	1.00	0.98	86
1.0	1.00	0.95	0.98	84
micro avg	0.98	0.98	0.98	170
macro avg	0.98	0.98	0.98	170
weighted avg	0.98	0.98	0.98	170

```
[[86  0]
 [ 4 80]]
```

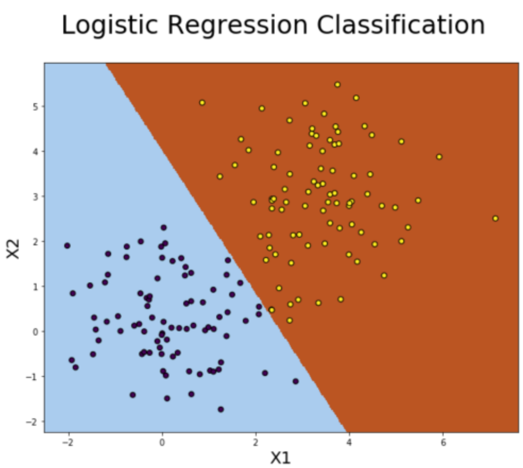
KNearest Neighbor

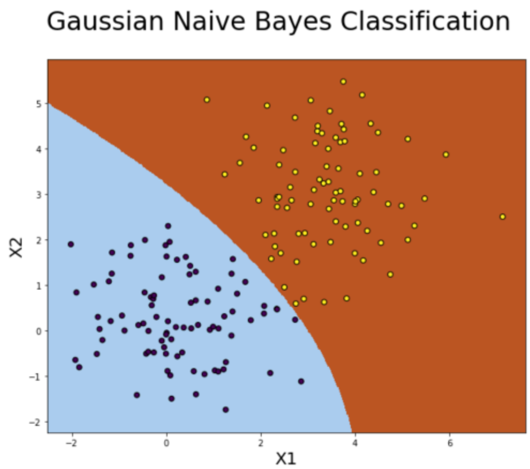
	precision	recall	f1-score	support
0.0	0.96	1.00	0.98	86
1.0	1.00	0.95	0.98	84
micro avg	0.98	0.98	0.98	170
macro avg	0.98	0.98	0.98	170
weighted avg	0.98	0.98	0.98	170

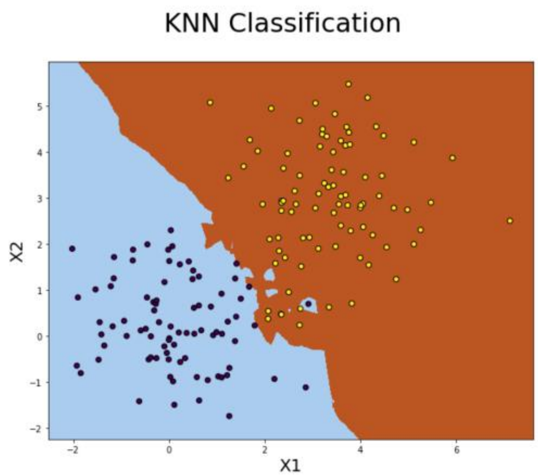
```
[[86  0]
 [ 4 80]]
```

Q3-a-ii

As can be seen from the figures below, logistic regression is classifiers works best for data that linearly separable. KNN shows more overfitting, and Naïve Bayes is more suitable for non- linearly separable data sets classification.







Q3-b

For the part b, we had more data in our test dataset. The results are more impressive than section before. As it can be seen in figure below, KNN performed the best. And the reason for this performance is because the MNIST data set is less linearly separable and KNN performed the best here.

```
LogisticRegression(C=1.0, class_weight=None, dual=False, fit_intercept=True,
          intercept_scaling=1, max_iter=100, multi_class='warn',
          n_jobs=None, penalty='l2', random_state=None, solver='warn',
          tol=0.0001, verbose=0, warm_start=False)
              precision    recall  f1-score   support

           0       0.98      0.98      0.98       201
           1       0.98      0.98      0.98       197

   micro avg       0.98      0.98      0.98       398
   macro avg       0.98      0.98      0.98       398
weighted avg       0.98      0.98      0.98       398

[[197   4]
 [  3 194]]
```

```
GaussianNB(priors=None, var_smoothing=1e-09)
              precision    recall  f1-score   support

           0       0.98      0.63      0.77       201
           1       0.72      0.99      0.84       197

   micro avg       0.81      0.81      0.81       398
   macro avg       0.85      0.81      0.80       398
weighted avg       0.85      0.81      0.80       398

[[126  75]
 [  2 195]]
```

```
KNeighborsClassifier(algorithm='auto', leaf_size=30, metric='minkowski',
           metric_params=None, n_jobs=None, n_neighbors=5, p=2,
           weights='uniform')
              precision    recall  f1-score   support

           0       1.00      1.00      1.00       201
           1       0.99      1.00      1.00       197

   micro avg       1.00      1.00      1.00       398
   macro avg       1.00      1.00      1.00       398
weighted avg       1.00      1.00      1.00       398

[[200   1]
 [  0 197]]
```