10-701 Machine Learning, Spring 2011: Homework 2

Due: Friday Feb. 4 at 4pm in Sharon Cavlovich's office (GHC 8215)

Instructions There are 3 questions on this assignment. The last question involves coding. Please submit your writeup as 3 **separate** sets of pages according to TAs, with your name and userid on each set. If you choose to work with a partner for question 3, you (as a team) should submit one set of pages for that question, labeled with both names/userids. You should still submit the solutions to questions 1-2 separately.

1 Probability [Xi Chen, 30 points]

1. This problem studies the relationship between entropy, conditional entropy, mutual information, conditional independence, and expected values.

   Consider random variables X and Y with joint probability density $p(X, Y)$. The expected value of any function $f(X, Y)$ of these variables is defined as $\mathbb{E}_p f(X, Y) = \int p(X, Y) f(X, Y) dxdy$ (i.e., it is the value of $f(X, Y)$ averaged over the different values X and Y can take on, weighted by their probabilities.) The expected value of a quantity is also sometimes called its "expectation".

   The entropy, joint entropy and conditional entropy can be expressed as the following expectations:

   - Entropy: $H(X) = -\mathbb{E}_p \ln p(X) = -\int p(X) \ln p(X) dx = -\int p(X, Y) \ln p(X) dxdy$
   - Joint entropy: $H(X, Y) = -\mathbb{E}_p \ln p(X, Y) = -\int p(X, Y) \ln p(X, Y) dxdy$
   - Conditional entropy: $H(X|Y) = -\mathbb{E}_p \ln p(X|Y) = -\int p(X, Y) \ln p(X|Y) dxdy$

   (a) In class we defined mutual information as

   $$I(X, Y) \triangleq H(X) - H(X|Y).$$

   Please use the linearity property of the expectation (i.e. $\mathbb{E}_p(X + Y) = \mathbb{E}_p(X) + \mathbb{E}_p(Y)$) to express $I(X, Y)$ as the expected value of a specific quantity. [2pt]

   (b) Use the linearity property of the expectation to prove the chain rule for entropy[1] [3pt]:

   $$H(X, Y) = H(Y) + H(X|Y).$$

   Notice that given this chain rule for entropy, we can easily derive that $I(X, Y) = H(X) + H(Y) - H(X, Y)$.

   (c) Recall the *conditional mutual information* between random variables X and Y given Z is defined by:

   $$I(X, Y|Z) \triangleq H(X|Z) - H(X|Y, Z).$$

   Express $I(X, Y|Z)$ as the expected value of a specific quantity. Also, state a conditional independence assumption that will guarantee $I(X, Y|Z) = 0$ [5pt].

[1] For your own interest, you may compare the chain rule for entropy to the chain rule in probability theory: $P(X, Y) = P(Y)P(X|Y)$

★ **SOLUTION:**

(a) Using the linearity property of the expectation:

$$\begin{aligned} I(X,Y) &= H(X) - H(X|Y) \\ &= -\mathbb{E}_p \ln p(X) + \mathbb{E}_p \ln p(X|Y) \\ &= -\mathbb{E}_p \ln \frac{p(X)}{p(X|Y)} \\ &= -\mathbb{E}_p \ln \frac{p(X)p(Y)}{p(X,Y)} \end{aligned}$$

(b)

$$\begin{aligned} H(Y) + H(X|Y) &= -\mathbb{E}_p \ln p(Y) - \mathbb{E}_p \ln p(X|Y) \\ &= -\mathbb{E}_p \left(\ln p(Y) + \ln p(X|Y)\right) \\ &= -\mathbb{E}_p \ln \left(p(X|Y)p(Y)\right) \\ &= -\mathbb{E}_p \left(\ln p(X,Y)\right) = H(X,Y) \end{aligned}$$

(c)

$$\begin{aligned} I(X,Y|Z) &= H(X|Z) - H(X|Y,Z) \\ &= -\mathbb{E}_p \ln p(X|Z) + \mathbb{E}_p \ln p(X|Y,Z) \\ &= -\mathbb{E}_p \ln \frac{p(X|Z)}{p(X|Y,Z)} \\ &= -\mathbb{E}_p \ln \frac{p(X|Z)p(Y|Z)}{p(X|Y,Z)p(Y|Z)} \\ &= -\mathbb{E}_p \ln \frac{p(X|Z)p(Y|Z)}{p(X,Y|Z)} \end{aligned}$$

If X and Y are *conditionally independent* given Z (i.e. $p(X|Z)p(Y|Z) = p(X,Y|Z)$), $I(X,Y|Z) = 0$.

2. Given two random variables X and Y, let $\mathbb{E}X$ and $\mathbb{E}Y$ denote the means (i.e., expected values) of X and Y and let σ_X and σ_Y denote the standard deviations of X and Y. Here are three quantities that are commonly used to characterize the relationship between X and Y:

- Covariance: $\text{cov}(X,Y) = \mathbb{E}[(X - \mathbb{E}X)(Y - \mathbb{E}Y)] = \mathbb{E}(XY) - \mathbb{E}X\mathbb{E}Y$
- Correlation: $\rho_{XY} = \frac{\text{cov}(X,Y)}{\sigma_X \sigma_Y}$
- Mutual Information: $I(X,Y) = H(X) - H(X|Y) = KL\left(p(X,Y)||p(X)p(Y)\right)$
where $KL(p||q) \equiv -\int p(x) \ln \frac{q(x)}{p(x)} dx$ is the Kullback-Leibler(KL) distance [2].

(a) Recall the Cauchy-Schwarz inequality: for any two non-degenerate [3] random variables X and Y:

$$\{\mathbb{E}(XY)\}^2 \leq \mathbb{E}(X^2)\mathbb{E}(Y^2),$$

where the equality holds if and and only if $P(X = aY) = 1$ for some non-zero constant a.
Use the Cauchy-Schwarz inequality to prove that for any two non-degenerate random variables X and Y, the absolute value of the correlation between X and Y is less than or equal to 1, i.e. $|\rho_{XY}| \leq 1$. [3pt]

(b) Show conditions under which we have $\rho_{XY} = 1$ and the conditions under which $\rho_{XY} = -1$. [2pt]

[2] As shown in homework 1, the definitions of mutual information based on KL divergence and entropy are equivalent.

[3] A random variable that can only take on one value (i.e., a constant) is called a degenerate random variable.

(c) If $I(X,Y)=0$, can we conclude that $\rho_{XY}=0$ (or equivalently, $\text{cov}(X,Y)=0$) ? If so, please give the proof; if not, please find the two random variables X and Y such that $I(X,Y)=0$ but $\rho_{XY}\neq 0$. [5pt]

(d) If $\rho_{XY}=0$, can we conclude that $I(X,Y)=0$? If so, please give the proof; if not, please find the two random variables X and Y such that $\rho_{XY}=0$ but $I(X,Y)\neq 0$. [10pt]

★ **SOLUTION:**

(a) We want to show that $|\rho_{XY}|\leq 1$. Plugging in the definition of ρ_{XY}, this is equivalent to showing that:

$$(\text{cov}(X,Y))^2 \leq \text{var}(X)\text{var}(Y),$$

which is further equivalent to showing that

$$\{\mathbb{E}\left((X-\mathbb{E}X)(Y-\mathbb{E}Y)\right)\}^2 \leq E\left((X-\mathbb{E}X)^2\right)E\left((Y-\mathbb{E}Y)^2\right).$$

Now note that $X-\mathbb{E}X$ and $Y-\mathbb{E}Y$ are themselves random variables, so the Cauchy-Schwarz inequality directly applies to them, and proves our statement $|\rho_{XY}|\leq 1$.

(b) By the Cauchy-Schwarz inequality, we get that $(\text{cov}(X,Y))^2\leq\text{var}(X)\text{var}(Y)$ (i.e. $|\rho_{XY}|=1$) if and only if

$$P(X-\mathbb{E}X=a(Y-\mathbb{E}Y))=1,$$

for some real a. Therefore, if X is linear in Y, i.e. $X=aY+b$, we have $|\rho_{XY}|=1$. Plug this condition into $\text{cov}(X,Y)$:

$$\begin{aligned}\text{cov}(X,Y) &= \mathbb{E}(XY)-\mathbb{E}X\mathbb{E}Y\\ &= \mathbb{E}(X(aX+b))-\mathbb{E}X\mathbb{E}(aX+b)\\ &= a\mathbb{E}(X^2)-a(\mathbb{E}X)^2\\ &= a\text{var}(X)\end{aligned}$$

Since $\text{var}(X)>0$, if $a>0$, we have $\text{cov}(X,Y)>0$ so $\rho_{XY}>0$. Since $|\rho_{XY}|=1$, $\rho_{XY}=1$. In contrast, if $a<0$, we have $\text{cov}(X,Y)<0$ so $\rho_{XY}<0$. Since $|\rho_{XY}|=1$, $\rho_{XY}=-1$.

In summary, if $X=aY+b$ and $a>0$, $\rho_{XY}=1$; if $X=aY+b$ and $a<0$, $\rho_{XY}=-1$.

(c) If $I(X,Y)=0$, X and Y are statistically independent, i.e. $p(X,Y)=p(X)p(Y)$. We have

$$\begin{aligned}\text{cov}(X,Y) &= \mathbb{E}(XY)-\mathbb{E}X\mathbb{E}Y\\ &= \int\int xyp(x,y)dxdy-\int xp(x)dx\int yp(y)dy\\ &= \int\int xyp(x)p(y)dxdy-\int xp(x)dx\int yp(y)dy\\ &= \int xp(x)dx\int yp(y)dy-\int xp(x)dx\int yp(y)dy\\ &= 0.\end{aligned}$$

In summary, if $I(X,Y)=0$, we can conclude that $\rho_{XY}=0$.

(d) The conclusion is that if $\rho_{XY}=0$, X and Y may not be independent, and hence $I(X,Y)$ may not be zero. Now we provide two uncorrelated random variables X and Y which are dependent. You may come up with your own example and will get full remark if your example is correct.

Let $X\sim N(0,1)$ follow a standard normal distribution and $Y=X^2$. From the definition of independence, X and Y are not independent. Since $x^3p(x)$ is an odd function, $\mathbb{E}(XY)=\mathbb{E}(X^3)=\int x^3p(x)=0$. Since $\mathbb{E}X=0$, we have

$$\begin{aligned}\text{cov}(X,Y) &= \mathbb{E}(XY)-\mathbb{E}X\mathbb{E}Y\\ &= \mathbb{E}(X^3)-\mathbb{E}X\mathbb{E}(X^2)\\ &= 0-0=0\end{aligned}$$

Therefore, we have $\rho_{XY}=0$ but X and Y are dependent.

2 Generative and Discriminative Classifiers: Gaussian (Naive) Bayes and Logistic Regression [Yi Zhang, 30 points]

Recall that a generative classifier estimates $P(\mathbf{X}, Y) = P(Y)P(\mathbf{X}|Y)$, while a discriminative classifier directly estimates $P(Y|\mathbf{X})$[4]. For clarity, we highlight $\mathbf{X}$ in bold to emphasize that it usually represents a vector of multiple attributes, i.e., $\mathbf{X} =< X_1, X_2, \dots, X_n >$. However, this question does **not** require students to derive the answer in vector/matrix notation.

In class we have observed an interesting relationship between a discriminative classifier (logistic regression) and a generative classifier (Gaussian naive Bayes): the form of $P(Y|\mathbf{X})$ derived from the assumptions of **a specific class** of Gaussian naive Bayes classifiers is precisely the form used by logistic regression. The derivation can be found in the required reading, **Mitchell: Naive Bayes and Logistic Regression, Section 3.1 (page 8 - 10)**. In that reading, we made the following assumptions for Gaussian naive Bayes classifiers to model $P(\mathbf{X}, Y) = P(Y)P(\mathbf{X}|Y)$ (rephrased from the required reading, with a few comments):

1. Y is a boolean variable following a Bernoulli distribution, with parameter $\pi = P(Y = 1)$ and thus $P(Y = 0) = 1 - \pi$.

2. $\mathbf{X} =< X_1, X_2, \dots, X_n >$, where each attribute X_i is a continuous random variable. For each X_i, $P(X_i|Y = k)$ is a Gaussian distribution $N(\mu_{ik}, \sigma_i)$. Note that σ_i is the standard deviation of the Gaussian distribution (and thus σ_i^2 is the variance), which does **not** depend on k.

3. For all $i \neq j$, X_i and X_j are conditionally independent given Y. This is why this type of classifier is called "naive".

We say this is **a specific class** of Gaussian naive Bayes classifiers because we have made an assumption that the standard deviation σ_i of $P(X_i|Y = k)$ does not depend on the value k of Y. This is **not** a general assumption for Gaussian naive Bayes classifiers.

2.1 General Gaussian naive Bayes Classifiers and Logistic Regression [15 points]

Let's make our Gaussian naive Bayes classifiers a little more general by removing the assumption that the standard deviation σ_i of $P(X_i|Y = k)$ does not depend on k. As a result, for each X_i, $P(X_i|Y = k)$ is a Gaussian distribution $N(\mu_{ik}, \sigma_{ik})$, where $i = 1, 2, \dots, n$ and $k = 0, 1$. Note that now the standard deviation σ_{ik} of $P(X_i|Y = k)$ depends on both the attribute index i and the value k of Y.

Question: is the new form of $P(Y|\mathbf{X})$ implied by this more general Gaussian naive Bayes classifier still the form used by logistic regression? Derive the new form of $P(Y|\mathbf{X})$ to prove your answer.

★ **SOLUTION:** **No**, the new $P(Y|\mathbf{X})$ implied by this more general Gaussian naive classifier is no longer the form used by logistic regression. Here is the derivation.

As shown in page 8 of [Mitchell: Naive Bayes and Logistic Regression], we have:

$$\begin{aligned}
P(Y = 1|\mathbf{X}) &= \frac{P(Y = 1)P(\mathbf{X}|Y = 1)}{P(Y = 1)P(\mathbf{X}|Y = 1) + P(Y = 0)P(\mathbf{X}|Y = 0)} \\
&= \frac{1}{1 + \frac{P(Y=0)P(\mathbf{X}|Y=0)}{P(Y=1)P(\mathbf{X}|Y=1)}} \\
&= \frac{1}{1 + \exp(\ln \frac{P(Y=0)P(\mathbf{X}|Y=0)}{P(Y=1)P(\mathbf{X}|Y=1)})} \\
&= \frac{1}{1 + \exp(\ln \frac{1-\pi}{\pi} + \ln \frac{P(\mathbf{X}|Y=0)}{P(\mathbf{X}|Y=1)})} \\
&= \frac{1}{1 + \exp(\ln \frac{1-\pi}{\pi} + \sum_i \ln \frac{P(X_i|Y=0)}{P(X_i|Y=1)})}
\end{aligned}$$

[4] Note that certain discriminative classifiers are non-probabilistic: they directly estimate a function $f : \mathbf{X} \rightarrow Y$ instead of $P(Y|\mathbf{X})$. We will see such classifiers later this semester, but it is beyond the scope of this question.

Now the standard deviation σ_{ik} of $P(X_i|Y=k)$ depends on both the attribute index i and the value k of Y, so we have:

$$\begin{aligned}
\sum_i \ln \frac{P(X_i|Y=0)}{P(X_i|Y=1)} &= \sum_i \ln \frac{\frac{1}{\sqrt{2\pi\sigma_{i0}^2}} \exp(\frac{-(X_i-\mu_{i0})^2}{2\sigma_{i0}^2})}{\frac{1}{\sqrt{2\pi\sigma_{i1}^2}} \exp(\frac{-(X_i-\mu_{i1})^2}{2\sigma_{i1}^2})} \\
&= \sum_i \ln \frac{\sigma_{i1}}{\sigma_{i0}} + \sum_i (\frac{(X_i-\mu_{i1})^2}{2\sigma_{i1}^2} - \frac{(X_i-\mu_{i0})^2}{2\sigma_{i0}^2}) \\
&= \sum_i \ln \frac{\sigma_{i1}}{\sigma_{i0}} + \sum_i \frac{(\sigma_{i0}^2 - \sigma_{i1}^2)X_i^2 + 2(\mu_{i0}\sigma_{i1}^2 - \mu_{i1}\sigma_{i0}^2)X_i + \mu_{i1}^2\sigma_{i0}^2 - \mu_{i0}^2\sigma_{i1}^2}{2\sigma_{i0}^2\sigma_{i1}^2} \\
&= \sum_i (\ln \frac{\sigma_{i1}}{\sigma_{i0}} + \frac{\mu_{i1}^2\sigma_{i0}^2 - \mu_{i0}^2\sigma_{i1}^2}{2\sigma_{i0}^2\sigma_{i1}^2}) + \sum_i \frac{(\mu_{i0}\sigma_{i1}^2 - \mu_{i1}\sigma_{i0}^2)}{\sigma_{i0}^2\sigma_{i1}^2} X_i + \sum_i \frac{(\sigma_{i0}^2 - \sigma_{i1}^2)}{2\sigma_{i0}^2\sigma_{i1}^2} X_i^2
\end{aligned}$$

As a result, we have $P(Y=1|\mathbf{X})$ as follows:

$$\begin{aligned}
P(Y=1|\mathbf{X}) &= \frac{1}{1+\exp(\ln\frac{1-\pi}{\pi} + \sum_i \ln\frac{P(X_i|Y=0)}{P(X_i|Y=1)})} \\
&= \frac{1}{1+\exp(w_0 + \sum_i w_i X_i + \sum_i v_i X_i^2)}
\end{aligned}$$

where

$$\begin{aligned}
w_0 &= \ln\frac{1-\pi}{\pi} + \sum_i (\ln\frac{\sigma_{i1}}{\sigma_{i0}} + \frac{\mu_{i1}^2\sigma_{i0}^2 - \mu_{i0}^2\sigma_{i1}^2}{2\sigma_{i0}^2\sigma_{i1}^2}) \\
w_i &= \frac{(\mu_{i0}\sigma_{i1}^2 - \mu_{i1}\sigma_{i0}^2)}{\sigma_{i0}^2\sigma_{i1}^2} \\
v_i &= \frac{(\sigma_{i0}^2 - \sigma_{i1}^2)}{2\sigma_{i0}^2\sigma_{i1}^2}
\end{aligned}$$

In general we do not have $\sigma_{i1} = \sigma_{i0}$ so the quadratic term $v_i X_i^2$ in $P(Y=1|\mathbf{X})$ will not disappear. Therefore the form of $P(Y=1|\mathbf{X})$ is no longer the form of logistic regression.

2.2 Gaussian Bayes Classifiers and Logistic Regression [15 points]

Students in 10-701 are all smart, so clearly we will not be satisfied by only studying a "naive" classifier. In this part, we will turn our attention to a specific class of Gaussian Bayes classifiers (without "naive", yeah!). We consider the following assumptions for our Gaussian Bayes classifiers:

1. Y is a boolean variable following a Bernoulli distribution, with parameter $\pi = P(Y=1)$ and thus $P(Y=0) = 1-\pi$.

2. $\mathbf{X} =< X_1, X_2 >$, i.e., we only consider **two** attributes, where each attribute X_i is a continuous random variable. X_1 and X_2 are **not** conditionally independent given Y. We assume $P(X_1, X_2|Y=k)$ is a **bivariate Gaussian distribution** $N(\mu_{1k}, \mu_{2k}, \sigma_1, \sigma_2, \rho)$, where μ_{1k} and μ_{2k} are means of X_1 and X_2, σ_1 and σ_2 are standard deviations of X_1 and X_2, and ρ is the **correlation** between X_1 and X_2. Note that μ_{1k} and μ_{2k} depend on the value k of Y, but σ_1, σ_2, and ρ do **not** depend on Y. Also recall that the density of a bivariate Gaussian distribution, given $(\mu_{1k}, \mu_{2k}, \sigma_1, \sigma_2, \rho)$, is:

$$P(X_1, X_2|Y=k) = \frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}} \exp[-\frac{\sigma_2^2(X_1-\mu_{1k})^2 + \sigma_1^2(X_2-\mu_{2k})^2 - 2\rho\sigma_1\sigma_2(X_1-\mu_{1k})(X_2-\mu_{2k})}{2(1-\rho^2)\sigma_1^2\sigma_2^2}]$$

Question: is the form of $P(Y|\mathbf{X})$ implied by such not-so-naive Gaussian Bayes classifiers still the form used by logistic regression? Derive the form of $P(Y|\mathbf{X})$ to prove your answer.

★ **SOLUTION:** **Yes**, the new $P(Y|\mathbf{X})$ implied by this not-so-naive Gaussian Bayes classifier is still the form used by logistic regression.

$$\begin{aligned}
P(Y=1|\mathbf{X}) &= \frac{P(Y=1)P(\mathbf{X}|Y=1)}{P(Y=1)P(\mathbf{X}|Y=1)+P(Y=0)P(\mathbf{X}|Y=0)} \\
&= \frac{1}{1+\frac{P(Y=0)P(\mathbf{X}|Y=0)}{P(Y=1)P(\mathbf{X}|Y=1)}} \\
&= \frac{1}{1+\exp(\ln\frac{P(Y=0)P(\mathbf{X}|Y=0)}{P(Y=1)P(\mathbf{X}|Y=1)})} \\
&= \frac{1}{1+\exp(\ln\frac{1-\pi}{\pi}+\ln\frac{P(\mathbf{X}|Y=0)}{P(\mathbf{X}|Y=1)})} \\
&= \frac{1}{1+\exp(\ln\frac{1-\pi}{\pi}+\ln\frac{P(X_1,X_2|Y=0)}{P(X_1,X_2|Y=1)})}
\end{aligned}$$

Note that we must work on the joint distribution $P(X_1,X_2|Y=0)$ and $P(X_1,X_2|Y=1)$ because X_1 and X_2 are no longer conditionally independent given Y. We proceed by focusing on the term $\ln\frac{P(X_1,X_2|Y=0)}{P(X_1,X_2|Y=1)}$:

$$\begin{aligned}
\ln\frac{P(X_1,X_2|Y=0)}{P(X_1,X_2|Y=1)} &= \ln\frac{\frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}}}{\frac{1}{2\pi\sigma_1\sigma_2\sqrt{1-\rho^2}}}+\ln\exp[(*)] \\
&= \ln\exp[(*)] \\
&= (*)
\end{aligned}$$

where (*) is the following formulation, obtained as the difference between the exponential parts of the bivariate Gaussian densities $P(X_1,X_2|Y=0)$ and $P(X_1,X_2|Y=1)$:

$$\begin{aligned}
(*) &= \frac{\sigma_2^2(X_1-\mu_{11})^2+\sigma_1^2(X_2-\mu_{21})^2-2\rho\sigma_1\sigma_2(X_1-\mu_{11})(X_2-\mu_{21})}{2(1-\rho^2)\sigma_1^2\sigma_2^2} \\
&\quad -\frac{\sigma_2^2(X_1-\mu_{10})^2+\sigma_1^2(X_2-\mu_{20})^2-2\rho\sigma_1\sigma_2(X_1-\mu_{10})(X_2-\mu_{20})}{2(1-\rho^2)\sigma_1^2\sigma_2^2}
\end{aligned}$$

It turns out, by a few steps of derivations, that all quadratic terms X_1^2, X_2^2 and X_1X_2 are canceled in the above (*) and therefore we end up with the following:

$$\begin{aligned}
(*) &= \frac{2\sigma_2^2(\mu_{10}-\mu_{11})+2\rho\sigma_1\sigma_2(\mu_{21}-\mu_{20})}{2(1-\rho^2)\sigma_1^2\sigma_2^2}X_1 \\
&\quad +\frac{2\sigma_1^2(\mu_{20}-\mu_{21})+2\rho\sigma_1\sigma_2(\mu_{11}-\mu_{10})}{2(1-\rho^2)\sigma_1^2\sigma_2^2}X_2 \\
&\quad +\frac{\sigma_2^2(\mu_{11}^2-\mu_{10}^2)+\sigma_1^2(\mu_{21}^2-\mu_{20}^2)+2\rho\sigma_1\sigma_2(\mu_{10}\mu_{20}-\mu_{11}\mu_{21})}{2(1-\rho^2)\sigma_1^2\sigma_2^2}
\end{aligned}$$

As a result, we have:

$$\begin{aligned}
P(Y=1|\mathbf{X}) &= \frac{1}{1+\exp(\ln\frac{1-\pi}{\pi}+\ln\frac{P(X_1,X_2|Y=0)}{P(X_1,X_2|Y=1)})} \\
&= \frac{1}{1+\exp(\ln\frac{1-\pi}{\pi}+(*))} \\
&= \frac{1}{1+\exp(w_0+w_1X_1+w_2X_2)}
\end{aligned}$$

where

$$
\begin{aligned}
w_0 &= \ln\frac{1-\pi}{\pi} + \frac{\sigma_2^2(\mu_{11}^2-\mu_{10}^2)+\sigma_1^2(\mu_{21}^2-\mu_{20}^2)+2\rho\sigma_1\sigma_2(\mu_{10}\mu_{20}-\mu_{11}\mu_{21})}{2(1-\rho^2)\sigma_1^2\sigma_2^2}\\
w_1 &= \frac{2\sigma_2^2(\mu_{10}-\mu_{11})+2\rho\sigma_1\sigma_2(\mu_{21}-\mu_{20})}{2(1-\rho^2)\sigma_1^2\sigma_2^2}\\
w_2 &= \frac{2\sigma_1^2(\mu_{20}-\mu_{21})+2\rho\sigma_1\sigma_2(\mu_{11}-\mu_{10})}{2(1-\rho^2)\sigma_1^2\sigma_2^2}
\end{aligned}
$$

This form of $P(Y=1|\mathbf{X})$ is still the form represented by logistic regression.

SPECIAL NOTES: we intentionally choose only two attributes $\mathbf{X} =< X_1, X_2 >$ because the density of *bivariate* Gaussian distribution can be expressed using scalars instead of vectors and matrices. This way, students do not need to derive the answer using vector/matrix algebra. However, if you are comfortable with vector and matrix algebra, please feel free to use following **alternative assumptions** and derive your answer in vector/matrix notation (which may turn out to be less time-consuming):

1. Y is a boolean variable following a Bernoulli distribution, with parameter $\pi = P(Y=1)$ and thus $P(Y=0) = 1-\pi$.
2. $\mathbf{X} =< X_1, X_2, \ldots, X_n >$ are **not** conditionally independent given Y, and $P(\mathbf{X}|Y=k)$ follows a **multivariate normal distribution** $N(\boldsymbol{\mu}_k, \boldsymbol{\Sigma})$. Note that $\boldsymbol{\mu}_k$ is the $n \times 1$ mean vector depending on the value of Y, and $\boldsymbol{\Sigma}$ is the $n \times n$ **covariance** matrix, which does **not** depend on Y. Also, you should be familiar with the density of multivariate normal distribution in vector/matrix notation.

You may choose to derive your answer for **either** the bivariate normal version using scalars **or** the multivariate normal version using vector/matrix notation. Derive both versions will **not** gain extra credits.

★ **SOLUTION:** In fact, deriving the answer in matrix and vector notation is much simpler and elegant. Now $\mathbf{X}$, μ_1 and μ_2 are $n \times 1$ vectors, and Σ is the $n \times n$ matrix. Similarly, we start with:

$$
\begin{aligned}
P(Y=1|\mathbf{X}) &= \frac{P(Y=1)P(\mathbf{X}|Y=1)}{P(Y=1)P(\mathbf{X}|Y=1)+P(Y=0)P(\mathbf{X}|Y=0)}\\
&= \frac{1}{1+\frac{P(Y=0)P(\mathbf{X}|Y=0)}{P(Y=1)P(\mathbf{X}|Y=1)}}\\
&= \frac{1}{1+\exp(\ln\frac{P(Y=0)P(\mathbf{X}|Y=0)}{P(Y=1)P(\mathbf{X}|Y=1)})}\\
&= \frac{1}{1+\exp(\ln\frac{1-\pi}{\pi}+\ln\frac{P(\mathbf{X}|Y=0)}{P(\mathbf{X}|Y=1)})}
\end{aligned}
$$

Then we focus on the term $\ln\frac{P(\mathbf{X}|Y=0)}{P(\mathbf{X}|Y=1)}$:

$$
\begin{aligned}
\ln\frac{P(\mathbf{X}|Y=0)}{P(\mathbf{X}|Y=1)} &= \ln\frac{\frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}}}{\frac{1}{(2\pi)^{n/2}|\Sigma|^{1/2}}} + \ln\exp[(*)]\\
&= \ln\exp[(*)]\\
&= (*)
\end{aligned}
$$

where, again, (*) is the formulation obtained as the difference between the exponential parts of two *multivariate* Gaussian densities $P(\mathbf{X}|Y=0)$ and $P(\mathbf{X}|Y=1)$:

$$
\begin{aligned}
(*) &= \frac{1}{2}[(X-\mu_1)^T\Sigma^{-1}(X-\mu_1)-(X-\mu_0)^T\Sigma^{-1}(X-\mu_0)]\\
&= (\mu_0^T-\mu_1^T)\Sigma^{-1}X + \frac{1}{2}\mu_1^T\Sigma^{-1}\mu_1 - \frac{1}{2}\mu_0^T\Sigma^{-1}\mu_0
\end{aligned}
$$

As a result, we have:

$$\begin{aligned} P(Y=1|\mathbf{X}) &= \frac{1}{1+\exp(\ln\frac{1-\pi}{\pi}+\frac{1}{2}\mu_1^T\Sigma^{-1}\mu_1-\frac{1}{2}\mu_0^T\Sigma^{-1}\mu_0+(\mu_0^T-\mu_1^T)\Sigma^{-1}X)} \\ &= \frac{1}{1+\exp(w_0+\mathbf{w}^T\mathbf{X})} \end{aligned}$$

where $w_0 = \ln\frac{1-\pi}{\pi}+\frac{1}{2}\mu_1^T\Sigma^{-1}\mu_1-\frac{1}{2}\mu_0^T\Sigma^{-1}\mu_0$ is a scalar and $\mathbf{w}=\Sigma^{-1}(\mu_0-\mu_1)$ is an $n\times 1$ parameter vector.

This is still the form of logistic regression (in vector and matrix notation).

3 Naive Bayes Document Classifier [Carl Doersch, 40 points]

In this question, you will implement the Naive Bayes document classifier and apply it to the classic 20 newsgroups dataset[5]. In this dataset, each document is a posting that was made to one of 20 different usenet newsgroups. Our goal is to write a program which can predict which newsgroup a given document was posted to.

For this question, you may write your code and solution in teams of at most 2. If you decide to do this, you should submit one copy of your solutions to question 3 (both code and answers to questions) per team. This copy should be clearly marked with the names of both team members.

3.1 Model

Say we have a document D containing n words; call the words $\{X_1,...,X_n\}$. The value of random variable X_i is the word found in position i in the document. We wish to predict the label Y of the document, which can be one of m categories, We could use the model:

$$P(Y|X_1...X_n)\propto P(X_1...X_n|Y)P(Y)=P(Y)\prod_i P(X_i|Y)$$

That is, each X_i is sampled from some distribution that depends on its position X_i and the document category Y. As usual with discrete data, we assume that $P(X_i|Y)$ is a multinomial distribution over some vocabulary V; that is, each X_i can take one of $|V|$ possible values corresponding to the words in the vocabulary. Therefore, in this model, we are assuming (roughly) that for any pair of document positions i and j, $P(X_i|Y)$ may be completely different from $P(X_j|Y)$.

Question 3.1: In your answer sheet, explain in a sentence or two why it would be difficult to accurately estimate the parameters of this model on a reasonable set of documents (e.g. 1000 documents, each 1000 words long, where each word comes from a 50,000 word vocabulary). **[3 points]**

★ **SOLUTION:** In this model, each position in a given document is assumed to have its own probability distribution. Each document gives has only one word at each position, so if there are M documents then we must estimate the parameters a roughly 50,000-dimensional distribution using only M samples from that distribution. In only a thousand documents, there will not be enough samples.

To see it another way, the fact that a word w appeared at the i'th position of the document gives us information about the distribution at another position j. Namely, in English, it is possible to rearrange the words in a document without significantly altering the document's meaning, and therefore the fact that w appeared at i means that it is more likely to appear at position j. Thus, it would be *statistically inefficient* to not to make use of the information in estimating the parameters of the distribution of X_j.

To improve the model, we will make the additional assumption that:

$$\forall i,j \quad P(X_i|Y)=p(X_j|Y)$$

Thus, in addition to estimating $P(Y)$, you must estimate the parameters for the single distribution $P(X|Y)$, which we define to be equal to $P(X_i|Y)$ for all X_i. Each word in a document is assumed to be an *iid* draw from this distribution.

[5] http://people.csail.mit.edu/jrennie/20Newsgroups/

3.2 Data

The data file (available on the website) contains six files:

1. **vocabulary.txt** is a list of the words that may appear in documents. The line number is word's id in other files. That is, the first word ('archive') has wordId 1, the second ('name') has wordId 2, etc.

2. **newsgrouplabels.txt** is a list of newsgroups from which a docment may have come. Again, the line number corresponds to the label's id, which is used in the .label files. The first line ('alt.atheism') has id 1, etc.

3. **train.label** Each line corresponds to the label for one document from the training set. Again, the document's id (docId) is the line number.

4. **test.label** The same as train.label, except that the labels are for the test documents.

5. **train.data** Specifies the counts for each of the words used in each of the documents. Each line is of the form "`docId wordId count`", where `count` specifies the number of times the word with id `wordId` in the training document with id `docId`. All word/document pairs that do not appear in the file have count 0.

6. **test.data** Same as train.data, except that it specified counts for test documents.

If you are using matlab, the functions `textread` and `sparse` will be useful in reading these files.

3.3 Implementation

Your first task is to implement the Naive Bayes classifier specified above. You should estimate $P(Y)$ using the MLE, and estimate $P(X|Y)$ using a MAP estimate with the prior distribution $Dirichlet(1+\alpha, ..., 1+\alpha)$, where $\alpha = 1/|V|$ and V is vocabulary.

Question 3.2: In your answer sheet, report your overall testing accuracy (Nubmer of correctly classified documents in the test set over the total number of test documents), and print out the confusion matrix (the matrix C, where c_{ij} is the number of times a document with ground truth category j was classified as category i). [**7 points**]

★ **SOLUTION:** The final accuracy of this classifier is **78.52%**, with the following confusion matrix:

		Predicted Class																			
		alt.atheism	**comp.graphics**	**comp.os.ms-windows.misc**	**comp.sys.ibm.pc.hardware**	**comp.sys.mac.hardware**	**comp.windows.x**	**misc.forsale**	**rec.autos**	**rec.motorcycles**	**rec.sport.baseball**	**rec.sport.hockey**	**sci.crypt**	**sci.electronics**	**sci.med**	**sci.space**	**soc.religion.christian**	**talk.politics.guns**	**talk.politics.mideast**	**talk.politics.misc**	**talk.religion.misc**
True Class	**alt.atheism**	249	0	0	0	0	1	0	0	1	0	0	2	0	3	3	24	2	3	4	26
	comp.graphics	0	286	13	14	9	22	4	1	1	0	1	11	8	6	10	1	2	0	0	0
	comp.os.ms-windows.misc	1	33	204	57	19	21	4	2	3	0	0	12	5	10	8	3	1	0	5	3
	comp.sys.ibm.pc.hardware	0	11	30	277	20	1	10	2	1	0	1	4	32	1	2	0	0	0	0	0
	comp.sys.mac.hardware	0	17	13	30	269	0	12	2	2	0	0	3	21	8	4	0	1	0	1	0
	comp.windows.x	0	54	16	6	3	285	1	1	3	0	0	5	3	6	4	0	1	1	1	0
	misc.forsale	0	7	5	32	16	1	270	17	8	1	2	0	7	4	6	0	2	1	2	1
	rec.autos	0	3	1	2	0	0	14	331	17	0	0	1	13	0	4	2	0	0	6	1
	rec.motorcycles	0	1	0	1	0	0	2	27	360	0	0	0	3	1	0	0	1	1	0	0
	rec.sport.baseball	0	0	0	1	1	0	2	1	2	352	17	0	1	3	3	5	2	1	5	1
	rec.sport.hockey	2	0	1	0	0	0	2	1	2	4	383	0	0	0	0	1	2	0	1	0
	sci.crypt	0	3	0	3	4	1	0	0	0	1	1	362	2	2	2	0	9	0	5	0
	sci.electronics	3	20	4	25	7	4	8	11	6	0	0	21	264	9	7	1	3	0	0	0
	sci.med	5	7	0	3	0	0	3	5	4	1	0	1	8	320	8	7	6	5	8	2
	sci.space	0	8	0	1	0	3	1	0	1	0	1	4	6	5	343	3	2	1	12	1
	soc.religion.christian	11	2	0	0	0	2	1	0	0	0	0	0	0	2	0	362	0	1	2	15
	talk.politics.guns	1	1	0	0	0	1	1	2	1	1	0	4	0	5	2	1	303	5	23	13
	talk.politics.mideast	12	1	0	1	0	0	1	2	0	2	0	2	1	0	0	6	3	326	18	1
	talk.politics.misc	6	1	0	0	1	1	0	0	0	0	0	5	0	10	6	2	63	6	196	13
	talk.religion.misc	39	3	0	0	0	0	0	0	1	1	0	1	0	2	6	27	10	3	7	151

Question 3.3: Are there any newsgroups that the algorithm confuses more often than others? Why do you think this is? [**2 points**]

★ **SOLUTION:** From the confusion matrix, it is clear that newsgroups with a similar topics are confused frequently. Notably, those related to computers (eg comp.os.ms-windows.misc and comp.sys.ibm.pc.hardware), those related to politics (e.g. talk.politics.guns and talk.politics.misc), and those related to religion (alt.atheism and talk.religon.misc). Newsgroups with similar topics have similar words that identify them. For example, we would expect the computer-related groups to all use computer terms frequently.

3.4 Priors and Overfitting

In your initial implementation, you used a prior $Dirichlet(1+\alpha, ..., 1+\alpha)$ to estimate $P(X|Y)$, and I told you set $\alpha = 1/|V|$. Hopefully you wondered where this value came from. In practice, the choice of prior is a difficult question in Bayesian learning: either we must use domain knowledge, or we must look at the performance of different values on some validation set. Here we will use the performance on the testing set to gauge the effect of α [6].

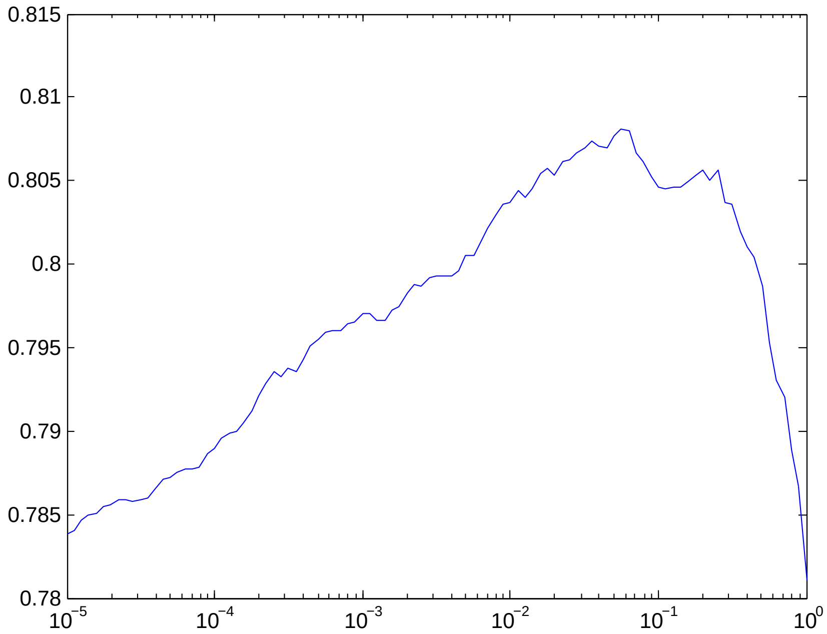
Question 3.4: Re-train your Naive Bayes classifier for values of α between .00001 and 1 and report the accuracy over the test set for each value of α. Create a plot with values of α on the x-axis and accuracy on the y-axis. Use a logarithmic scale for the x-axis (in Matlab, the `semilogx` command). Explain in a few sentences why accuracy drops for both small and large values of α [**5 points**]

★ **SOLUTION:** For very small values of α, we have that the probability of rare words not seen during training for a given class tends to zero. There are many testing documents that contain words seen only in one or two training documents, and often these training documents are of a different class than the test document. As α tends to zero, the probabilities of these rare words tends to dominate.

A number of students attributed the poor performance at small values of α to 'overfitting'. While this is strictly speaking correct (the classifier estimates $P(X|Y)$ to be smaller than is realistic simply because that was the case in the data), simply attributing this to overfitting is not a sophisticated answer. Different classifiers overfit for different reasons, and understanding the differences is an important goal for you as students.

For large values of α, we see a classic underfitting behavior: the final parameter estimates are closer tend toward the prior as α increases, and the prior is just something we made up. In particular, the classifier tends to underestimate the importance of rare words: for example, if α is 1 and we see only one occurrence of the word w in the category C (and we see the same number of words in each category), then the final parameter estimates are 2/21 for category C and 19/21 that it would be something else. Furthermore, the most informative words tend to be relatively uncommon, and so we would like to rely on these rare words more.

[6] It is tempting to choose α to be the one with the best performance on the testing set. However, if we do this, then we can no longer assume that the classifier's performance on the test set is an unbiased estimate of the classifier's performance in general. The act of choosing α based on the test set is equivalent to training on the test set; like any training procedure, this choice is subject to overfitting.



3.5 Identifying Important Features

One useful property of Naive Bayes is that its simplicity makes it easy to understand why the classifier behaves the way it does. This can be useful both while debugging your algorithm and for understanding your dataset in general. For example, it is possible to identify which words are strong indicators of the category labels we're interested in.

Question 3.5: Propose a method for ranking the words in the dataset based on how much the classifier 'relies on' them when performing its classification (hint: information theory will help). Your metric should use only the classifier's estimates of $P(Y)$ and $P(X|Y)$. It should give high scores to those words that appear frequently in one or a few of the newsgroups but not in other ones. Words that are used frequently in general English ('the', 'of', etc.) should have lower scores, as well as words that only appear appear extremely rarely throughout the whole dataset. Finally, your method this should be an overall ranking for the words, not a per-category ranking.[**3 points**]

★ **SOLUTION:** First of all, a handful of people whined about not liking the open-endedness of this problem. I hate to say it, but nebulous problems like this are common in machine learning–this problem was actually inspired by something I worked on last summer in industry. The goal was to design a metric for finding documents similar to some query document, and part of the procedure involved classifying words in the query document into one of 100 categories, based on the word itself and the word's context. The algorithm initially didn't work as well as I thought it should have, and the only path to improving its performance was to understand what these classifiers were 'relying on' in order to do their classification–some way of understanding the classifiers' internal workings, and even I wasn't sure what I was looking for. In the end I designed a metric based on information theory and, after looking at hundreds of word lists printed from these classifiers, I eventually found a way to fix the problem. I felt this experience was valuable enough that I should pass it on to all of you.

There were many acceptable solutions to this question, but probably the most successful ones studied the distribution of indicator variables $X_i = (X == the\_i'th\_word\_in\_V)$. A common error was the use of imprecise notation: note that $H(X|Y)$ and $I(X,Y)$ are just single numbers; these expressions have different meanings than the expressions $H(X_i|Y)$ and $I(X_i, Y)$.

The first measure is $H(Y|X_i = True)$, the entropy of the label given a document with a single word w_i. Intuitively, this value will be low if a word appears most of the time in a single class, because the distribution $P(Y|X_i = True)$ will be highly peaked. More concretely (and abbreviating $True$ as T),

$$\begin{array}{rcl} H(Y|X_i = T) & = & -\sum_k P(Y = y_k|X_i = T) log(P(Y = y_k|X_i = T) \\ & = & -E_{P(Y|X_i=T)} log(P(Y = y_k|X_i = T) \\ & = & -E_{P(Y|X_i=T)} log\frac{P(X_i=T|Y=y_k)P(Y=y_k)}{P(X_i=T)} \\ & = & -E_{P(Y|X_i=T)} log\frac{P(X_i=T|Y=y_k)}{P(X_i=T)} - E_{P(Y|X_i=T)} log(P(Y = y_k)) \end{array}$$

Note that

$$log\frac{P(X_i = T|Y = y_k)}{P(X_i = T)}$$

is exactly what gets added to Naive Bayes' internal estimate of the posterior probability $log(P(Y))$ at each step of the algorithm (although in implementations we usually ignore the constant $P(X_i = T)$). Furthermore, the expectation is over the posterior distribution of the class labels given the appearance of word w_i. Thus, the first term of this measure can be interpreted as the expected change in the classifier's estimate of the log-probability of the 'correct' class given the appearance of word w_i. The second term tends to be very small relative to the first term since P(Y) is close to uniform; I found that the word list is the same with or without it.

Another popular measure was $I(X_i, Y)$, which Prof. Mitchell said was quite useful in fMRI data. Intuiively, this measures the amount of information we learn by observing X_i. An issue with this measure is that Naive Bayes only really learns from X_i in the event that $X_i = True$, and essentially ignores this variable when $X_i = False$ (thus, the issue was introduced because we're computing our measure on X_i rather than on X). Note that this is not the case in fMRI data (i.e. you compute the mutual information directly on the features used for classification), which explains why mutual information works better in that domain. Note that $X_i = False$ most of the time for informative words, so in the formula:

$$I(X_i, Y) = H(X_i) - H(X_i|Y) =$$

$$-\sum_{x_i \in \{T,F\}} P(X_i = x_i) \left[log(X_i = x_i) - \sum_k P(Y = y_k|X_i = x_i) log(P(Y = y_k|X_i = x_i)) \right]$$

We see that the term for $x_i = F$ tends to dominate even though it is essentially meaningless.

Another disadvantage of this metric is that it's more difficult to implement; the majority of people who tried made some mistake. Most notably, quite a few people forgot to sum over the possible values of x_i.

Question 3.6: Implement your method, set α back to $1/|V|$, and print out the 100 words with the highest measure. [**2 points**]

★ **SOLUTION:** For the metric $H(Y|X_i = True)$:

```
‘‘nhl’’, ‘‘stephanopoulos’’, ‘‘leafs’’, ‘‘alomar’’, ‘‘wolverine’’, ‘‘crypto’’, ‘‘lemieux’’,
‘‘oname’’, ‘‘rsa’’, ‘‘athos’’, ‘‘ripem’’, ‘‘rbi’’, ‘‘firearm’’, ‘‘powerbook’’, ‘‘pitcher’’,
‘‘bruins’’, ‘‘dyer’’, ‘‘lindros’’, ‘‘lciii’’, ‘‘ahl’’, ‘‘fprintf’’, ‘‘candida’’, ‘‘azerbaijan’’,
‘‘baerga’’, ‘‘args’’, ‘‘iisi’’, ‘‘gilmour’’, ‘‘clh’’, ‘‘gfci’’, ‘‘pitchers’’, ‘‘gainey’’, ‘‘clemens’’,
‘‘dodgers’’, ‘‘jagr’’, ‘‘sabretooth’’, ‘‘liefeld’’, ‘‘hawks’’, ‘‘hobgoblin’’, ‘‘rlk’’, ‘‘adb’’,
‘‘crypt’’, ‘‘anonymity’’, ‘‘aspi’’, ‘‘countersteering’’, ‘‘xfree’’, ‘‘punisher’’, ‘‘recchi’’,
‘‘cipher’’, ‘‘oilers’’, ‘‘soderstrom’’, ‘‘azerbaijani’’, ‘‘obp’’, ‘‘goalie’’, ‘‘libxmu’’, ‘‘inning’’,
‘‘xmu’’, ‘‘sdpa’’, ‘‘argic’’, ‘‘serdar’’, ‘‘sumgait’’, ‘‘denning’’, ‘‘ioccc’’, ‘‘obfuscated’’,
‘‘umu’’, ‘‘nsmca’’, ‘‘dineen’’, ‘‘ranck’’, ‘‘xdm’’, ‘‘rayshade’’, ‘‘gaza’’, ‘‘stderr’’, ‘‘dpy’’,
‘‘cardinals’’, ‘‘potvin’’, ‘‘orbiter’’, ‘‘sandberg’’, ‘‘imake’’, ‘‘plaintext’’, ‘‘whalers’’,
‘‘moncton’’, ‘‘jaeger’’, ‘‘uccxkvb’’, ‘‘mydisplay’’, ‘‘wip’’, ‘‘hicnet’’, ‘‘homicides’’, ‘‘bontchev’’,
‘‘canadiens’’, ‘‘messier’’, ‘‘bure’’, ‘‘bikers’’, ‘‘cryptographic’’, ‘‘ssto’’, ‘‘motorcycling’’,
‘‘infante’’, ‘‘karabakh’’, ‘‘baku’’, ‘‘mutants’’, ‘‘keown’’, ‘‘cousineau’’
```

For the metric $I(X_i, Y)$:

```
‘‘windows’’, ‘‘god’’, ‘‘he’’, ‘‘scsi’’, ‘‘car’’, ‘‘drive’’, ‘‘space’’, ‘‘team’’, ‘‘dos’’, ‘‘bike’’,
‘‘file’’, ‘‘of’’, ‘‘that’’, ‘‘mb’’, ‘‘game’’, ‘‘key’’, ‘‘mac’’, ‘‘jesus’’, ‘‘window’’, ‘‘dod’’,
‘‘hockey’’, ‘‘the’’, ‘‘graphics’’, ‘‘card’’, ‘‘image’’, ‘‘his’’, ‘‘gun’’, ‘‘encryption’’, ‘‘sale’’,
‘‘apple’’, ‘‘government’’, ‘‘season’’, ‘‘we’’, ‘‘games’’, ‘‘israel’’, ‘‘disk’’, ‘‘files’’,
‘‘ide’’, ‘‘controller’’, ‘‘players’’, ‘‘shipping’’, ‘‘chip’’, ‘‘program’’, ‘‘was’’, ‘‘cars’’,
‘‘nasa’’, ‘‘win’’, ‘‘year’’, ‘‘were’’, ‘‘they’’, ‘‘turkish’’, ‘‘motif’’, ‘‘people’’, ‘‘armenian’’,
‘‘play’’, ‘‘drives’’, ‘‘bible’’, ‘‘use’’, ‘‘widget’’, ‘‘pc’’, ‘‘clipper’’, ‘‘offer’’, ‘‘jpeg’’,
‘‘baseball’’, ‘‘bus’’, ‘‘my’’, ‘‘nhl’’, ‘‘software’’, ‘‘is’’, ‘‘db’’, ‘‘server’’, ‘‘jews’’,
‘‘os’’, ‘‘israeli’’, ‘‘output’’, ‘‘data’’, ‘‘system’’, ‘‘who’’, ‘‘league’’, ‘‘armenians’’,
‘‘for’’, ‘‘christian’’, ‘‘christians’’, ‘‘entry’’, ‘‘mhz’’, ‘‘ftp’’, ‘‘price’’, ‘‘christ’’,
‘‘guns’’, ‘‘thanks’’, ‘‘church’’, ‘‘color’’, ‘‘teams’’, ‘‘privacy’’, ‘‘condition’’, ‘‘launch’’,
‘‘him’’, ‘‘com’’, ‘‘monitor’’, ‘‘ram’’
```

(Note the presence of the words ‘car’, ‘of’, ‘that’, etc.

Question 3.7: If the points in the training dataset were not sampled independently at random from the same distribution of data we plan to classify in the future, we might call that training set *biased*. Dataset bias is a problem because the performance of a classifier on a biased dataset will not accurately reflect its future performance in the real world. Look again at the words your classifier is ‘relying on’. Do you see any signs of dataset bias? [**3 points**]

★ **SOLUTION:** While I don’t know exactly how this dataset was collected, it is certain that the dataset was collected over some finite time period in the past. That means our classifier will tend to rely on some words that are specific to this time period. For the first word list, ‘stephanopolous’ refers to a politician who may not be around in the future, and ‘whalers’ refers to the Connecticut hockey team that was actually being desolved at the same time as this dataset was being collected. For the second list, 'ghz' has almost certainly replaced 'mhz' in modern computer discussions, and the controversy regarding Turkey and Armenia is far less newsworthy today. As a result, you should expect the classification accuracy on the 20-newsgroups testing set to significantly overestimate the classification accuracy your algorithm would have on a testing sample from the same newsgroups taken today.

Sadly, there is a lot of bad machine learning research that has resulted from biased datasets. Researchers will train an algorithm on some dataset and find that the performance is excellent, but then apply it in the real world and find that the performance is terrible. This is especially common in vision datasets, where there is a tendency to always photograph a given object in the same environment or in the same pose. In your own research, make sure your datasets are realistic!

3.6 Hand-in for Question 3

Print out the code that you used to solve problem 3. This should include the code for the Naive Bayes classifier, the code for modifying α, and the code for identifying important features. Submit this code in class on the due date, along with your answers to questions 3.1-3.7. [**15 points**]

★ **SOLUTION:** The code for this question is provided in the solution code file on the webpage.