1. **Pset 2 problem 2**

   The Naive Bayes assumption says that the features are independent given the class, i.e. $p(\text{feature1, feature2, feature3}|\text{class}) = p(\text{feature1}|\text{class})p(\text{feature2}|\text{class})p(\text{feature3}|\text{class})$.

   This does not mean that $p(\text{class}|\text{feature1, feature2, feature3}) = p(\text{class}|\text{feature1})p(\text{class}|\text{feature2})p(\text{class}|\text{feature3})$.

   So the conditional probabilities of the form $p(\text{feature}|\text{class})$ are the ones that you needed to calculate based on the data that was given.

   Also the calculation for Naive Bayes needs to include the prior probability of the class. Otherwise you could end up selecting a very unlikely class for which the features have a high probability instead of a very likely class with slightly lower probabilities for the features.

2. **Pset2 problem 3**

   You’re asked to give the equation for the expected loss for predicting class $j$ for data point $d$. Since $d$ and $j$ are known, and the true class $k$ is unknown, it must be $k$ and only $k$ that should be summed over. In general, the expected value is the sum of each possible value (or integral for the continuous case) weighted by the probability of the value. In this case, $E_k[L_j] = \sum_k P(k|d)L_{kj}$

3. **HMMs**

   See the lecture slides for the relevant equations for the forward and backward algorithms, viterbi algorithm, and Baum-Welch.

   An HMM consists of a set of states with initial state probabilities $\pi$, transition probabilities $A$, and emission probabilities $\phi$. We’ll call our observed data $X = x_1 \ldots x_n$ and the corresponding hidden variables are the states $Z = z_1 \ldots z_n$.

   In graphical model notation it looks like this:
We can write the complete-data likelihood by simple reading the dependencies in the diagram:

\[
p(X, Z|\pi, A, \phi) = p(z_1|\pi)p(x_1|z_1, \phi)p(z_2|z_1, A)p(x_2|z_2, \phi) \cdots p(z_n|z_{n-1}, A)p(x_n|z_n, \phi) \tag{1}
\]

\[
= p(z_1|\pi) \prod_{i=2}^{n} p(z_i|z_{i-1}, A) \prod_{j=1}^{n} p(x_j|z_j, \phi) \tag{2}
\]

In English, this is the product starting with the probability of starting with the initial state \(z_1\), then the probability of emitting \(x_1\) from state \(z_1\), then the probability of transitioning from \(z_1\) to \(z_2\) and emitting \(x_2\) from \(z_2\), and so on all the way through transitioning to \(z_n\) and emitting \(x_n\).

In posterior decoding we ask what is the probability of being in a particular state \(k\) at time/position \(z_i\) given the data \(X\). Intuitively this must be based on all the possible ways of emitting \(x_1\) to \(x_i\) and ending in state \(k\) at time \(i\), combined with all of the possible ways of emitting \(x_{i+1}\) to \(x_n\) starting from state \(k\) at time \(i\). These are exactly the quantities calculated by the forward and backward algorithms and assigned the notation \(\alpha\) and \(\beta\), respectively.

We could use the posterior decoding to determine the most likely state at each time, but the resulting sequence of states might require transitions that have zero probability.

As an alternative we can use the Viterbi algorithm to determine the state sequence that is most likely in its entirety. In this case, analogous to finding the best sequence alignment,
the best entire state sequence must consist of a combination of the best sequence up to time \( n - 1 \) ending in some state \( j \) combined with the transition to a state \( k \) at time \( n \) and the emission of \( x_n \) from \( k \). In turn, the best sequence up to time \( n - 1 \) must be the combination of the best sequence up to time \( n - 2 \) combined with a transition to a state for time \( n - 1 \) and an emission from that state.

Finally, we can also use EM to learn the parameters for our HMM, called Baum-Welch training. The parameters are the initial state probabilities, transition probabilities, and emission probabilities, and the hidden variables are the states. So, in the E-step we compute the posterior probability of each hidden state given the data and the current guess at the parameters. These posterior probabilities are the same ones computed using the forward and backward algorithms in posterior decoding. In the M-step we compute new parameters based on the current estimates of the hidden states.