CPSC 440/540 Machine Learning (January-April, 2022)
Assignment 2 (due Friday February 11th at midnight)

The assignment instructions are the same as for the previous assignment, but for this assignment you can work in groups of 1-2. However, please only hand in one assignment for the group.

1. Name(s):
2. Student ID(s):

1 Bernoulli Distributions

1.1 Inference

Consider a Bernoulli distribution with $\theta = 0.12$.

1. Suppose we generate 10 IID samples $\{x^1, x^2, ..., x^{10}\}$ according to this model. What is the probability that all 10 samples are equal to 0?
2. What is the probability that exactly one sample is equal to 1, and the other 9 are equal to 0?
3. Suppose we need to base decisions on samples from this dataset, and so we want collect a set of 1000 samples. Would it make sense to chose the decoding of the 1000 samples (the "most likely" samples to be seen) to make decisions? Explain why or why not?
4. Write a function that generates t IID examples from this distribution, that uses Julia's *rand()* function as the source of IID randomness (assuming that each call to *rand()* returns a number uniformly-distributed between 0 and 1). Hand in the code.

5. Consider a game where you get $1 if a sample from this distribution is 0, and lose $5 if the sample is 1. What is the expected $ value you get from playing this game?

6. A way to approximate the answer of the previous question is given by

$$\frac{1}{t}\sum_{i=1}^{t} f(x^i),$$

where each x^i is one of t IID sample and f gives the $ value for that sample (either 1 or -5). Implement this approximation, and hand in a plot the of value of this approximation as you vary the number of samples t from 1 to 100,000.

7. Suppose that you wanted to compute the expected number of samples from this Bernoulli that you would need to generate before seeing 10 values of 1. It possible to compute this expectation exactly, but suppose we really like sampling and we are too lazy to do the exact calculation. Give an approximation of this quantity based on generating samples, similar to the one used in Part 6.

8. Suppose that the *rand()* function was broken in the new version of Julia, in the sense that the numbers it returned were found to not look like uniform samples.[1] Also suppose that the *randn()* function, which generates samples from a standard normal distribution, worked well. Describe a way to generate samples from a Bernoulli distribution that relies on the *randn()* function instead of the *rand()* function. Hint: you may want to look up the CDF (and its inverse the "quantile" function) of the standard normal distribution.

[1] Some published research works have had to be revised due to bad random number generators.

1.2 Learning

1. Consider a dataset consisting of binary samples $\{x^1, x^2, \ldots, x^n\}$. Consider the binomial likelihood for such a dataset,

$$p(x^1, x^2, \ldots, x^n \mid \theta) = \binom{n}{k} \theta^k (1-\theta)^{n-k},$$

where $k = \sum_{i=1}^{n} x^i$ is the number of 1 values in the dataset and $\binom{n}{k} = \frac{n!}{k!(n-k)!}$ is the number o of combinations of k items that exist among n items. This likelihood measures the probability of seeing exactly k values of 1 in the dataset, as opposed to the Bernoulli likelihood which measures the likelihood of seeing the precise sequence x^1, x^2, through x^n. Derive the MAP estimate of θ for this model if we use a beta prior on θ with hyper-parameters α and β. You can assume that k and $(n-k)$ are both positive. You can also assume that the data is independent of the hyper-parameters if we condition on the parameters, which mathematically is equivalent to $p(X \mid \theta, \alpha, \beta) = p(X \mid \theta)$.

2. We often use Bernoulli distributions as parts of more-complicated distributions. In these cases we often to need to maximize a weighted log-likelihood of the form

$$f(\theta) = \sum_{i=1}^{n} v^i \log p(x^i \mid \theta),$$

where each v^i is a non-negative weight for example i. Derive the MLE for this model when we use a Bernoulli likelihood for $p(x^i \mid \theta)$.

2 Multivariate and Conditional Bernoullis

2.1 Naive Bayes

If you run the script *example_mnist35_naiveNaiveBayes*, it will load training and test MNIST digits 3 and 5. It will then discretize the features into binary values, and fit a "naive" naive Bayes model. In particular, the naive naive Bayes model is a generative classifier that assumes $p(x_1, x_2, \ldots, x_d, y)$ is product of Bernoullis. This model has a high test error, since the features do not affect the predictions.

1. The regular naive Bayes model discussed in class writes the joint probability of a data $\{X, y\}$ as

$$\begin{aligned} p(X, y) &= \prod_{i=1}^{n} \left[p(y^i) \prod_{j=1}^{d} p(x_j^i \mid y^i) \right] \\ &= \prod_{i=1}^{n} \left[\theta^{y^i} (1-\theta)^{1-y^i} \prod_{j=1}^{d} \theta_{jy^i}^{x_j^i} (1-\theta_{jy^i})^{1-x_j^i} \right], \end{aligned}$$

   where we have used a Bernoulli to parameterize $p(y^i)$ and a Bernoulli-like distribution to parameterize the conditionals $p(x_j \mid y)$. Show how to derive the MLE for a particular value of θ_{jc}.

2. Even though the naive Bayes likelihood has many parameters, the MLE for a particular parameter θ_{jc} does not depend on θ or any any $\theta_{j'c'}$ with $j \neq j'$ and $c' \neq c$. Explain why these terms do not appear in the MLE. (You should not use the word "independence" in your answer. Instead explain how the form of the log-likelihood makes them not depend on each other.)

3. Modfy the naive naive Bayes code to implement the standard naive Bayes classifier. Hand in your code and report the the test error that you obtain. Hint: start from the function *naiveNaiveBayes* and replace the $d \times 1$ variable *p_x* with a $d \times 2$ variable *p_xy* that will represnt θ_{jc}. Hint: you can look at subsequent questions to get an idea of what the final error should be.

4. Instead of assuming that all variables are independent given the class label, a less-naive model called "chain-augmented naive Bayes" (CANB) assumes that variables follow a Markov chain given the class label. This leads to a joint likelihood of the form

$$p(x_1^i, x_2^i, \dots, x_d^i, y^i) = p(y^i)p(x_1^i \mid y^i) \prod_{j=2}^{d} p(x_j^i \mid x_{j-1}^i, y^i),$$

where features 2:d depend on not only on the class label but also on the value of the feature that comes "before" it in the ordering 1:d. This captures the fact that adjacent pixels in the image tend to be correlated with each other. Using MLE to estimate the parameters of this model leads to test error of 0.24, while if you add Laplace smoothing it achieves a better test error of 0.14 (which is still worse than naive Bayes). On the other hand, adding Laplace smoothing to naive Bayes does not change the test error for this data. Explain why Laplace smoothing improves the test error of CANB, while for naive Bayes it does not affect the test error.

5. With Laplace smoothing, the CANB from the previous question obtains a much-higher test-set likelihood than naive Bayes. In other words, $p(\tilde{X}, \tilde{y})$ for new data is much higher for the CANB model than naive Bayes. Explain why the CANB model might give wosre test error even though it gives a much-better test set likelihood.

2.2 Vector-Quantized Naive Bayes

If you run the script *example_mnist35_logistic*, it will fit a logistic regression model that acheives a lower test error on the MNIST 3 and 5 digits than naive Bayes does. In this question we will consider one way to make naive Bayes less naive in order to improve its performance.

1. It seems reasonable that there would be clusters in the classes. For example, there might several different ways to draw the digit '3'. Instead of assuming that the features are independent given the class label, we might instead assume they are independent given the cluster that they are in. Consider a *vector-quantized naive Bayes* (VQNB) that implements this idea:

   - It clusters the *examples associated with each digit* into k clusters, using k-means clustering (so there will be k clusters for each class and $2k$ clusters total). We will use z^i as the cluster number of example i. The value z^i will be from 1 to k, with y^i determining whether we are considering the k "3" clusters or the k "5" clusters.
   - Since we do not know z^i (it is a "latent" variable), we need to use the marginalization rule to consider its possible values. Using the marginalization and then the product rule, we can write the joint probability as

$$\begin{aligned} p(x_1^i, x_2^i, \dots, x_d^i, y^i) &= \sum_{z=1}^{k} p(x_1^i, x_2^i, \dots, x_d^i, y^i, z) \\ &= \sum_{z=1}^{k} p(x_1^i, x_2^i, \dots, x_d^i \mid y^i, z) p(y^i, z) \\ &= \sum_{z=1}^{k} p(x_1^i, x_2^i, \dots, x_d^i \mid y^i, z) p(z \mid y^i) p(y^i) \\ &= p(y^i) \sum_{z=1}^{k} p(z \mid y^i) \prod_{j=1}^{d} p(x_j^i \mid y^i, z), \end{aligned}$$

     where the last line is using the independence of the features given the cluster.

   Make a version of your naive Bayes code that implements the VQNB method. Hand in your code and the test error you obtain with this model for $k = 2$ through $k = 5$.. Hints:

   - You can use the variable *p_y* as before. You will need a new variable *p_zy* that is $k \times 2$ representing $p(z = c \mid y^i = b)$. The MLE for an element *p_zy[c,b]* of the matrix is (number of times that $y^i = b$ and $z^i = c$)/(number of times $y^i = b$).[2]
   - You can replace *p_xy* with *p_xyz* which will be a d by 2 by k array giving the value $p(x_j^i = 1 \mid y^i = b, z^i = c)$. The MLE for the value *p_xyz[j,b,c]* is (number of times $y^i = b$ and $z^i = c$ and $x_j^i = 1$)/(number of times $y^i = b$ and $z^i = c$).
   - To help with debugging, note that you should get the naive Bayes model in the special case of $k = 1$. Further, with this type of model you usually see the biggest performance gain when going from $k = 1$ to $k = 2$.
   - You may not be able to exactly match the performance of logistic regression.

[2] We will derive this MLE for k-valued discrete variables in the next part of the course.

2. What is the cost of making a prediction with this model?

3. For a run of the method with $k = 5$, show the images obtained by plotting the estimates for $p(x_j = 1 \mid z, y)$ for all j as a 28 by 28 image, for each value of z and y (so there should be 10 images). Hint: if you want to view image i with *PyPlot*, you could use `imshow(reshape(X[i,:],28,28)',''gray'')`.

4. The logistic regression model and VQNB model obtain similar accuracies, and both give estimates of the probability for $p(y^i \mid x_1^i, x_2^i, \ldots, x_d^i)$. Give an example of something we could do with the VQNB model that we could not do with the logistic regression model.

3 Neural Networks

3.1 Neural Networks by Hand

Suppose that we train a neural network with sigmoid activations and one hidden layer and obtain the following parameters (assume that we don't use any bias variables):

$$W = \begin{bmatrix} -2 & 2 & -1 \\ 1 & -2 & 0 \end{bmatrix}, v = \begin{bmatrix} 3 \\ 1 \end{bmatrix}.$$

Assuming that we are doing regression, for a training example with features $(x^i)^T = \begin{bmatrix} -3 & -2 & 2 \end{bmatrix}$ what are the values in this network of the hidden units z^i, activations $a^i = h(z^i)$, and prediction $\hat{y}^i$?

3.2 Neural Network Tuning - Regression

The file `example_nnet.jl` runs a stochastic gradient method to train a neural network on the *basisData* dataset from a previous assignment. However, in its current form it doesn't fit the data very well. Modify the training procedure and model to improve the performance of the neural network. Hand in your plot after changing the code to have better performance, and list the changes you made.

Hint: there are many possible strategies you could take to improve performance. Below are some suggestions, but note that the some will be more effective than others:

- Changing the network structure (*nHidden* is a vector giving the number of hidden units in each layer).
- Changing the training procedure (you can change the stochastic gradient step-size, use decreasing step sizes, use mini-batches, run it for more iterations, add momentum, switch to *findMin*, use Adam, and so on).
- Transform the data by standardizing the feautres, standardizing the targets, and so on.
- Add regularization (L2-regularization, L1-regularization, dropout, and so on).
- Change the initialization.
- Add bias variables.
- Change the loss function or the non-linearities (right now it uses squared error and tanh to introduce non-linearity).
- Use mini-batches of data, possibly with batch normalization.

3.3 Neural Network Tuning - Classification

The file `example_mnist35_nnet.jl` runs a stochastic gradient method to train a neural network on the '3' and '5' images from the naive Bayes question (it uses the squared error for training and classifies by taking the sign of the prediction). Modify the training procedure and model so that the neural network has a better error on the test set than the 0.03 achieved by logistic regression. List the changes you made and the best test performance that you were able to obtain.

4 Very-Short Answer Questions

1. List 5 different inference tasks you might do with a (single variable) Bernoulli model?
2. What is a reason we might want to sample from a distribution?
3. Suppose we have $p(x) \propto \alpha g(x)$ for some continuous variable x, some positive constant α, and some non-negative function g. Give the form of the distribution if we do not use the $\propto$ sign to hide the proportionality constant.
4. In the CANB model (discussed in a previous question), you would need to estimate $p(x_j^i = 1 \mid x_{j-1}^i, y^i)$ for all values of x_{j-1}^i and y^i. We could consider a generalization of this where we estimate $p(x_j^i = 1 \mid x_{j-1}^i, x_{j-2}^i, \ldots, x_{j-k}^i, y^i)$ for some value k (to further relax the conditional independence assumption made by naive Bayes.) Assuming all values are binary we use tabular probabilities, how many parameters would this model need?
5. What is the advantage of using a discriminative model for supervised learning, rather than the generative approach of treating supervised learning as special case of density estimation.

6. Give a choice of the number of hidden units, the activation function, and likelihood in a neural network with a single hidden layer such that MLE in the model makes the same predictions as the MLE logistic regression model.

7. Describe an experimental setup where you might see a double descent curve when fitting a logistic regression model with L2-regularization. You can assume you have a way to generate new relevant features. Caution: the global minimum is unique in this model so the double descent would not come from choosing among multiple global minima.

8. Consder a neural network with L hidden layers, where each layer has at most k hidden units and we have c labels in the final layer (and the number of features is d). What is the cost of prediction with this network?

9. What is one advantage of using automatic differentiation and one disadvantage?

10. Convolutional networks seem like a pain... why not just use regular ("fully connected") neural networks for image classification?

11. Why do autoencoders have a bottleneck layer?

12. We discussed multi-label classification using neural networks and a product of Bernoullis model. The product of Bernoullis model assumes the class labels are independent, so why might this model make predictions that seem to capture dependencies between the random variables?

13. Why can fully-convolutional networks segment images of different sizes?