Introduction to learning and analysis of big data
Exercise 3

Dr. Sivan Sabato

Fall 2020/21

Submission guidelines, **please read and follow carefully**:

- Submit the exercise in pairs.
- Submit using the submission system.
- The submission should be a zip file named "ex3.zip".
- The zip file should include **only the following files in the root - no subdirectories please**.
- The files in the zip file should be:
  1. A file called "answers.pdf" - The answers to the questions, including the graphs.
  2. An ".m" file for each of the requested Matlab functions.

  Anywhere in the exercise where Matlab is mentioned, you can use the free software Octave instead.
- For questions use the course Forum, or if they are not of public interest, send them via the course requests system.
- Grading: Q1: 15, Q2: 16, Q3: 14, Q4: 14, Q5: 12, Q6: 15, Q7: 14.

Question 1. Implement the naive-Bayes algorithm we learned in class in MATLAB, for the general (non-symmetric) case, for binary classification with labels $\mathcal{Y} = \{+1, -1\}$. Create two functions, with the following signatures:

```
function [allpos,ppos,pneg] = bayeslearn(Xtrain, Ytrain)

function Ypredict = bayespredict(allpos, ppos, pneg, Xtest)
```

The first function estimates the conditional probabilities to use in the Bayes predictor. The input parameters are:

- `Xtrain` - a 2-D matrix of size $m \times d$ describing the training set, where m is the size of the training sample and d is the number of features in each example. All entries in the matrix are binary (in $\{0, 1\}$).
- `Ytrain` - a column vector of length m with entries in $\{-1, 1\}$ that provides the labels of the examples in `Xtrain`.

The output parameters are:

- `allpos`: A scalar in $[0, 1]$, which indicates the fraction of positive labels in the training sample, the empirical plug-in estimate of $\mathbb{P}[Y = 1]$.
- `ppos,pneg`: Vectors of length d, where `ppos`(i) is the empirical plug-in estimate of the conditional probability $\mathbb{P}[X(i) = 1 \mid Y = 1]$, and `pneg`$(i)$ is the empirical plug-in estimate of $\mathbb{P}[X(i) = 1 \mid Y = -1]$. Note that some of the values could be undefined, if no example with $Y = 1$ or $Y = -1$ exists in the training sample, since in this case, trying to calculate the conditional probabilities results in division by zero. In this case, the relevant value in the relevant vector should be `NaN`. You will need to ignore such coordinates when predicting the label in the prediction function.

The second function uses these probability estimates to predict labels of a test sample. The input parameters are:

- `allpos,ppos,pneg` - the outputs of the learning function
- `Xtest` - a 2-D matrix of size $m' \times d$ representing a test set, where m' is the number of test examples in the test set, and d is the dimension.

The output of the second function is the vector `Ypredict`, where `Ypredict`(i) is the predicted label for the i'th test example, where the prediction is based on the naive-Bayes classification rule based on `allpos, ppos,pneg`. If the classifier cannot decide which label to choose, it should output 0.

Note: the naive-Bayes classification rule is based on choosing the Bayes-optimal label as showed in class for the naive-Bayes assumption. This formula includes products of many probabilities. The best approach for calculating the winning label using this formula is to take the `log` of this formula and to compare sums instead of products. This avoids getting into very small numbers that are too close to machine precision.

Submit the files `bayeslearn.m` and `bayespredict.m` which implement the functions.

Question 2. Test your naive Bayes implementation on the MNIST data set. To translate every example to a binary vector with features in $\{0, 1\}$, set coordinate i of example x to 1 if pixel i of the MNIST image has value larger than 128, and to 0 otherwise. Test two binary classification problem: differentiating 0 from 1, and differentiating 3 from 5.

(a) Train your naive-Bayes classifier on training sizes $1000, 2000, \ldots, 10000$ for both binary classification problems, and submit a (single) plot of the test errors of each of the problems as a function of the sample size.

(b) Describe in words the performance of the method on each of the binary classification problems, and the effect of the sample size on the accuracy.

(c) Generate and submit two `HeatMap` plots, which show `ppos` and `pneg` that were generated for the classification problem of differentiating 0 and 1, for the case of a sample size of 10000. Make sure to plot the heat maps as a 28×28 image. Explain in words what the heat maps look like, and why.

(d) Take the parameters that your `bayeslearn` function learned for the training size of 10000, and change the `allpos` value to be 0.75 instead of the value your function returned. Now, compare the labels predicted on the test set using this new `allpos` value with the labels predicted using the original `allpos` value. Report what percent of the test set examples had their label changed from -1 to 1, and what percent had it changed from 1 to -1. Explain your findings using the naive Bayes predictor formula.

Question 3. Let $\mathcal{X} = \mathbb{R}^d$, and $\mathcal{Y} = \{-1, +1\}$. Consider a neural network with the activation function $\sigma = \text{sign}$ and a single output neuron, that predicts a label in $\mathcal{Y}$ for each $x \in \mathcal{X}$, by taking the sign of the output neuron.

(a) Define such a neural network such that the hypothesis class it represents is equivalent to the class of homogeneous linear predictors. Prove your claim.

(b) Give an example of such a neural network such that the hypothesis class it represents includes the class of homogeneous linear predictors but is **not** equivalent to it. Prove your claim.

Hint: Construct a network over examples in $\mathbb{R}^3$, whose hypothesis class includes a function which is positive iff there are more positive coordinates than negative coordinates.

Question 4. Let $\mathcal{X} = \{0, 1\}^3$, $\mathcal{Y} = \{-1, 1\}$. Consider training sample in which each of the 8 possible values of examples in $\mathcal{X}$ is observed the same number of times, and $y_i = 1$ if and only if $x_i(1) = x_i(2)$.

(a) Prove that the smallest depth of a decision tree (with a single attribute in each node) which is sufficient to perfectly classify this sample is exactly 2.

(b) Prove that a greedy algorithm which uses one of the `Gain` functions that we saw in class might get an error of 50% after getting to a depth of 2.

Question 5. (Do this question after we learn about regression in class) Consider regression with $\mathcal{X} = \mathbb{R}^d$ and $\mathcal{Y} = \{-1, +1\}$. Suppose we have a training sample $S = ((x_1, y_1), \ldots, (x_m, y_m))$. We are told that there is a unique solution $w \in \mathbb{R}^d$ to the problem of linear regression with the squared loss on the given sample.

(a) What is the rank of the matrix X whose rows are the examples in the sample? explain.

(b) Suppose that we take each example x_i in the training set and add a coordinate to its end, and set its value to 0. Call the new example with the added coordinate $x'_i \in \mathbb{R}^{d+1}$. Let $S' = ((x'_1, y_1), \ldots, (x'_m, y_m))$. What is the set of optimal solutions to the linear regression problem on S'? State this set as a function of w, the solution to the original problem. Prove your claim.

Question 6. (Do this question after we learn about PCA in class) There are 4 sensors, each giving a real value as a reading. In an experiment, readings were taken from all of the sensors at times $t = 1, 2, 3$. The 4 readings of the sensors at time t were listed as a vector $x_t \in \mathbb{R}^4$. This created a data set $S = (x_1, \ldots, x_3)$, with $x_1 = (1, 2, 3, 4), x_2 = (3, 4, 1, -5), x_3 = (-10, 1, 4, 6)$. PCA was performed on the data set S to reduce its dimensionality from 4 to 2. Use Matlab or python to answer each of the following questions. In each of your answers, explain how you calculated the result.

Note: do not use functions specifically for PCA. Use only basic functions such as matrix/vector operations, calculating eigenvalues, eigenvectors etc.

(a) Calculate the distortion that is expected for the data set above based on the distortion formula we saw in class.

(b) Calculate the transformation matrix U^T that performs the optimal dimensionality reduction using the formula for the solution that we saw in class. Explain how you calculated it, and report the resulting U^T. Make sure that U^T is orthonormal!

(c) Perform dimensionality reduction on $x_1, \ldots, x_4$ using U^T, and then restore using U. Report the values of the restored vectors. Now, calculate the resulting distortion by comparing the restored vectors to the original vectors. Explain how you calculated it, and report the number you got. Is it similar to the answer you got in (a)? Explain.

Question 7. (Do this question after we learn about clustering in class) Consider the single-linkage clustering algorithm, with a "cluster distance" stopping condition. So the algorithm stops when the minimum distance between any two clusters is larger than r.

(a) Does the algorithm satisfy the scale-invariance axiom? Prove your claim.

(b) Does the algorithm satisfy the richness axiom? Prove your claim.