Fall 2023
CSCE 320 - Principles of Data Science
Homework #2

Problem 1 – linear regression (30%)

In this exercise, you will gain experience developing and evaluating multivariate linear regression models. For this purpose, you will be using a synthetic dataset available on Canvas as h2.zip. The dataset includes a training set and a test set, each set consisting of (1) a matrix of predictors (i.e., matrix X in the lecture slides), and (2) a vector with the dependent variable (e.g., vector Y in the lecture slides).

You are to develop the ordinary least squares solution (i.e., pseudo-inverse). The problem has been carefully calibrated, so the correct solution is expected to yield R2 values in a tight range. For full transparency, Appendix I includes a snippet of the Matlab code we used to generate the data, though we used different hyper parameters from those shown on the snippet.

a. **15% credit**: Compute the OLS solution on the training set and apply it to the test set. Your results should include the R2 on test data, and a scatter plot of ground truth vs predictions. Please discuss your findings

b. **15% credit**: Merge the training and test sets ([x1; x2], [y1; y2]) and re-split them to generate new training and test partitions. Then repeat part a. multiple times and discuss the distribution of R2 values from split to split.

c. **5% extra credit**: Generate your own data by converting the code in Appendix I to Python. Then, experiment with different number of noisy channels (dd) and samples (n). How does R2 change as a function of these two parameters? Discuss this and other findings.

Problem 2 – dimensionality reduction (30%)

In this exercise, you will gain experience in using dimensionality-reduction techniques to explore a multivariate dataset from a real domain (nutrition). The dataset is also included in h2.zip. It includes a large number of food products from the ten different products shown on **Table 1** (see Appendix II). Each food item includes 46 descriptors, as shown on **Table 2**.

The dataset is organized as follows:

- Training set: Files x1.csv (one row per sample, one column per feature) and c1.csv (each row containing the class label of the corresponding row in x1.csv)
- Validation set: Files x2.csv and c2.csv, with similar structure as above.
- Test set: Files x3.csv with similar structure as above. The labels for the test data (i.e., c3.csv) are not included.

Detailed instructions:

a. **10% credit**. Perform PCA on the training set and generate a 2D scatterplot. To facilitate analysis, please color code each example according to its class label (even though PCA does not use class labels). Interpret the results. What information do the principal components capture? Is there any class separability? Is there any organization of classes into broader categories of food? …

b. **5% credit**. Generate a scree plot and interpret the results. How many PCs are required to capture 95% of the variance in the data? Does the scree plot suggest there is high or low collinearity between features? …

c. **10% credit**. Repeat part a. using Fisher's LDA. In this case, you will use the training data (x1.csv, c1.csv) to obtain the LDA eigenvector matrix (w1). Then, use matrix w1 to project the training examples (x1.csv), and generate a scatter plot. Finally, use matrix w1 to project the validation samples (x2.csv). As in problem 1, please color code the scatterplots according to class labels. Discuss your findings. Does the LDA scatter plot for training data show any structure? Describe that structure. Do these results hold when you project the (unseen) validation data? Please elaborate.

d. **5% credit**. Repeat part b. using Fisher's LDA.

e. **5% extra credit**. Experiment with a different type of dimensionality reduction, such as t-SNE, Isomap, or MDS (see manifold learning). Interpret your results.

Problem 3 – Classification (40%)

This problem will build upon the previous one. In this case, however, you will gain experience developing a classification model that can predict the class label of an unknown food item based on its nutritional descriptors.

a. **10% credit**. Use a 1-nearest neighbor (i.e., k=1) classifier to predict labels for the validation samples (x2.csv). Report classification performance by comparing ground truth labels (c2.csv) against predictions from the 1NN classifier. Generate a confusion matrix and interpret the results.

b. **20% credit**. Perform cross-validation to find the optimum number of neighbors (k) for the kNN classifier. For this purpose, merge the training and validation sets ([x1; x2], [c1; c2]), and then re-split them following the original proportions. For each split, use the training samples to classify the validation samples, and store the classification rate. Repeat the re-splitting process multiple times and compute the average performance. Start with k=1 neighbors, and repeat for k=2, 3, … Discuss your findings. Does classification performance improve as you increase the number of neighbors? Are changes in classification performance statistically significant (i.e., as indicated by ANOVA)? Generate a box plot for each value of k you evaluated.

c. **10% credit**. Repeat part a. using a different classifier from those discussed in lecture 8, e.g., nearest mean classifier, quadratic classifier[1]. You may incorporate prior probabilities to improve performance. NOTE: depending on the data split, the covariance matrix of each class may be ill-conditioned, so you may need to use a diagonal matrix (i.e., which assumes features are uncorrelated). Is this assumption valid?

d. **5% extra credit**. Write a Jupyter notebook that reads a test dataset (x3.csv) and generates predictions (c3_pred.csv). In this case, you will not be able to evaluate the performance of this classifier since the test labels (c3.csv) are not available to you. Instead, we will compare your predictions against the correct class labels (which we have kept aside). NOTE: to receive extra credit, we will need to run your code and generate the predictions file (c3_pred.csv) ourselves. In other words, you will not receive extra credits if you only submit the file c3_pred.csv.

[1] Note that scikit-learn uses the term "Linear Discriminant Analysis" to refer to both (1) the dimensionality reduction technique in Lecture 7 and (2) the Quadratic classifiers in Lecture 8. As you will recall, when you assume classes have equal covariance matrices ($\Sigma_i = \Sigma$) the decision boundaries become linear. In this case, the dimensionality reduction technique and the linear classifier are equivalent, except the former generates projections and the latter generates class labels.

Detailed submission instructions

Please submit a separate ZIP file for each problem:

1. Problem 1: p1.zip
2. Problem 2: p2.zip
3. …

Each ZIP file should include:

1. A PDF report (e.g., p1.pdf)
2. A Jupyter notebook (e.g., p1.ipynb)
3. Any data files your code needs in order to run (e.g., CSV files)

PDF reports:

- Each PDF report will be graded in isolation from the associated code. Thus, please make sure the PDF includes all the necessary figures.
- Each figure included in the PDF should be described and discussed in the write-up. To receive full credit, each figure should have its axes labeled and include a legend whenever possible.
- Please do not include the following, as they will be counted against your grade:
    - Snippets of your code
    - Screenshots of numerical results
    - Figures that are not described and discussed in the write up
- Please structure your PDF reports for readability. For example, use headings whenever appropriate and figure captions explaining the content of each figure. You may use the formatting of this document as a template.
- Please use language that is appropriate for a technical report. Informal language (e.g., something you would write in an email to a friend) is not appropriate. Neither is the use of hyperbole. Technical terms should also be used rigorously (e.g., the term 'significant' is only appropriate when supported by an appropriate test statistic).

Code:

- Please make sure your code runs before submitting it. This includes making sure the ZIP file has all the required data files.
- You will be allowed to make minor modifications (e.g., TypeError, NameError) to your code if it does not run initially. However, any modification to your submitted code will result in a 50% reduction on the code grade.
- Please add comments to your code to assist us in reviewing it.

You are free to use any code or notebooks that are available online, as long as you:

- Acknowledge the source,
- Provide a link to it (e.g., URL),
- Describe what the code does, and
- Describe how you modified it to serve your purpose

If you use existing code/notebooks without proper citations, your submission will be treated as a case of **plagiarism** (see Syllabus).

Appendix I

```
%% Generate predictors

n   = 122;     % number of examples
d   = 7;       % number of predictors
dd  = 11;      % number of noisy channels

x = randn(n,d);         % normally distributed predictors (IVs)
x = [ones(n,1) x];      % add intercept
w = randn(d+1,1);       % forward model (from x to y)
w = w/sqrt(sum(w.^2)); % normalize regression coeffs (unit length)
y = x*w;                % generate dependent variable (DV)

std = .25;                      % standard dev of additive noise
y   = y + std*randn(size(y));   % add noise to DV

x = [x randn(n,dd)]; % Add noisy channels

ix = randperm(size(x,2));   % shuffle predictors and noise channels
x  = x(:,ix);               % predictors are no longer on columns 1:d+1

%% Split data

x1 = x(1:(n/2),:);   % training data
y1 = y(1:(n/2),:);

x2 = x((n/2+1):end,:); % test data
y2 = y((n/2+1):end,:);

csvwrite('x1.csv', x1); % save data on CSV format
csvwrite('x2.csv', x2);
csvwrite('y1.csv', clab1);
csvwrite('y2.csv', clab2);
```

Appendix II

The nutritional dataset is organized as follows:

- **Training set**: Files x1.csv (one row per sample, one column per feature) and c1.csv (each row containing the class label of the corresponding row in x1.csv)
- **Validation set**: Files x2.csv and c2.csv, with similar structure as above.
- **Test set**: Files x3.csv, with similar structure as above. In this case, however, the class labels c3.csv are not available to you, so we can ensure you do not use those examples for training.

Note: the predictor values on the CSV files have been z-score normalized (i.e., $z = (x - \mu)/\sigma$). If you wish to undo the normalization (i.e., $x = z\sigma + \mu$) and recover the original values (so they match the units in **Table 2**), the ZIP file includes the means (mu.csv) and standard deviations (sig.csv) of the original predictors. Your PCA projections will vary depending on whether you use the original values or the z-score values).

Table 1. Class descriptions

Class label	Description
1	Baked Products
2	Vegetables and Vegetable Products
3	Soups, Sauces, and Gravies
4	Sweets
5	Fast Foods
6	Fruits and Fruit Juices
7	Breakfast Cereals
8	Poultry Products
9	Beef Products
10	Lamb, Veal, and Game Products

Table 2. Feature descriptions

Feature	Description	Feature	Description
1	Water (g/100 g)	24	Vitamin B6 (mg/100 g)
2	Food energy (kcal/100 g)	25	Total Folate (μg/100 g)
3	Protein (g/100 g)	26	Folic acid (μg/100 g)
4	Total lipids (fat) (g/100 g)	27	Food Folate (μg/100 g)
5	Ash (g/100 g)	28	Folate (μg diet folate equivalent/100 g)
6	Carbohydrate (g/100 g)	29	Vitamin B12 (μg/100 g)
7	Total dietary fiber (g/100 g)	30	Vitamin A (IU/100 g)
8	Total sugars (g/100 g)	31	VitA (μg retinol activity equivalents/100g)
9	Calcium (mg/100 g)	32	Retinol (μg/100 g)
10	Iron (mg/100 g)	33	Vitamin E (alpha-tocopherol) (mg/100 g)
11	Magnesium (mg/100 g)	34	Vitamin K (phylloquinone) (μg/100 g)
12	Phosphorus (mg/100 g)	35	Alpha-carotene (μg/100 g)
13	Potassium (mg/100 g)	36	Beta-carotene (μg/100 g)
14	Sodium (mg/100 g)	37	Beta-cryptoxanthin (μg/100 g)
15	Zinc (mg/100 g)	38	Lycopene (μg/100 g)
16	Copper (mg/100 g)	39	Lutein + zeazanthin (μg/100 g)

17	Manganese (mg/100 g)	40	Saturated fatty acid (g/100 g)
18	Selenium (μg/100 g)	41	Monounsaturated fatty acids (g/100 g)
19	Vitamin C (mg/100 g)	42	Polyunsaturated fatty acids (g/100 g)
20	Thiamin (mg/100 g)	43	Cholesterol (mg/100 g)
21	Riboflavin (mg/100 g)	44	1st household weight from Weight file
22	Niacin (mg/100 g)	45	2nd household weight from Weight file
23	Pantothenic acid (mg/100 g)	46	Percent refuse