CS6923 Machine Learning, Spring 2019
Prof. Linda Sellie, NYU School of Engineering

Homework 2[1]

Submit on NYU Classes by Thurs. Feb. 28 at 6:00 p.m. You may work together with one other person on this homework. If you do that, hand in JUST ONE homework for the two of you, with both of your names on it. You may *discuss* this homework with other students but YOU MAY NOT SHARE WRITTEN ANSWERS OR CODE WITH ANYONE BUT YOUR PARTNER.

IMPORTANT SUBMISSION INSTRUCTIONS: Please submit your solutions in 3 separate files: one file for your written answers to Part I, one file for your written answers/output for the questions in Part II, and one file with your code (a zip file if your code requires more than one file).

Part I: Written Exercises

1. A medical researcher wishes to evaluate a new diagnostic test for cancer. A clinical trial is conducted where the diagnostic measurement y of each patient is recorded along with attributes of a sample of cancerous tissue from the patient. Three possible models are considered for the diagnostic measurement:

   - Model 1: The diagnostic measurement y depends linearly only on the cancer volume.
   - Model 2: The diagnostic measurement y depends linearly on the cancer volume and the patient's age.
   - Model 3: The diagnostic measurement y depends linearly on the cancer volume and the patient's age, but the dependence (slope) on the cancer volume is different for two types of cancer – Type I and II. (Hint: Use a variable x_3 which is assigned the value 1 if the cancer is Type I, and x_3 has the value 0 if the cancer is of Type II.)

   (a) Define variables for the cancer volume, age and cancer type and write a linear model for the predicted value $\hat{y}$ in terms of these variables for models 1 & 2 above.

   (b) Do the same for model 3. For Model 3, you will want to use one-hot coding as mentioned above.

   (c) What are the number of parameters in model 1 & 2? Which model is the most complex?

   (d) Since the models in part (a) are linear, given training data, we should have $\hat{\mathbf{y}} = X\mathbf{w}$ where $\hat{\mathbf{y}}$ is the vector of predicted values on the training data, X is a design matrix (feature matrix) and $\mathbf{w}$ is the vector of parameters. To test the different models, data is collected from 100 patients. The records of the first three patients are shown below:

Patient ID	Measurement y	Cancer type	Cancer volume	Patient age
12	5	I	0.7	55
34	10	II	1.3	65
23	15	II	1.6	70
⋮	⋮	⋮	⋮	⋮

   For model 1 in part (a), based on this data, what are the first three rows of the matrix X?
   For model 2 in part (a), based on this data, what are the first three rows of the matrix X?

   For model 3 in part (a), based on this data, what are the first three rows of the matrix X?

[1] Some of these are modified from Prof. Rangan's questions.

(e) To evaluate the models, 10-fold cross validation is used with the following results.

Model	training MSE	test MSE
1	2.0	2.01
2	0.7	0.72
3	0.65	0.70

Which model should be selected?

2. Suppose you were interested in crop yields and you had collected data on the amount of rainfall, the amount of fertilizer, the average temperature, and the number of sunny days.

   How could you formalize this a as regression problem?

3. (The data in this problem is taken from `https://stats.libretexts.org/Homework_Exercises/General_Statistics/Exercises%3A_OpenStax/12.E%3A_Linear_Regression_and_Correlation_(Exercises)`)

   The following table gives the height (sidewalk to roof), and number of stories (beginning at street level) for notable buildings in the United States.

Height (in feet)	Stories
1,050	57
428	28
362	26
529	40
790	60
401	22
380	38
1454	110
1127	100
700	46

Suppose we want to predict the number of stories in a building, based on its height. To do this, apply (simple) linear regression to the above dataset, using the closed-form formula presented in class. Then answer the following questions.

(a) Give the equation for the linear function (line) produced using linear regression, in the form $g(x) = w_1 x + w_0$.

(b) Using this linear function, what is the predicted number of stories in a building that is 475 feet tall?

4. To apply k-NN to problems with real-valued attributes, it is important to scale the attributes beforehand. One standard approach to scaling the values of an attribute x_i is to replace each x_i by the following value:

$$\frac{x_i - x_i^{min}}{x_i^{max} - x_i^{min}}$$

Here x_i^{min} and x_i^{max} are the minimum and maximum values of x_i in the training set. This ensures that all values of x_i in the training set will be between 0 and 1.

(a) Show the result of applying the above scaling formula to the following dataset (showing two decimal places for each). You should do the scaling SEPARATELY for each attribute (so you should compute a different min and max value for each attribute x_i). To get you started, we have already scaled x_1, so you only need to scale x_2.

Original Dataset

x1	x2	label
2.5	42	+
3.8	51	+
-0.3	-1	+
0.7	3	-
1.6	26	-
2.3	41	-

Scaled Dataset

x1	x2	label
0.68		+
1.00		+
0.00		+
0.24		-
0.46		-
0.63		-

(b) When applying scaling of the attributes in k-NN, it is important to scale any new (test) examples using the same formula that was used to scale the attributes in the training set. Using the scaling formula you computed for the training set above, scale the new example $x = \begin{bmatrix} 3.9 \\ 4 \end{bmatrix}$

and classify it using k-NN with the Euclidean distance measure, for $k = 1$. (The scaled version of the new example can have attribute values that are NOT between 0 and 1.)

What is the predicted label for the example? Show your work.

5. Consider a linear model of the form,

$$y \approx \mathbf{w}x,$$

which is a linear model, but with the intercept forced to zero. This occurs in applications where we want to force the predicted value $\hat{y} = 0$ when $x = 0$. For example, if we are modeling $y =$ output power of a motor vs. $x =$ the input power, we would expect $x = 0 \Rightarrow y = 0$.

(a) Given data (x_i, y_i), write a cost function representing the residual sum of squares (RSS) between y_i and the predicted value $\hat{y}_i$ as a function of $\mathbf{w}$.

(b) Taking the derivative with respect to $\mathbf{w}$, find the $\mathbf{w}$ that minimizes the RSS.

6. Derive the closed form solution of ridge regression.

7. Suppose we have a binary (2-class) classification problem, where the classes are + and -. We learn a classifier, and then use it to predict the label of a test example x. Let r denote the correct label of example x, and let y denote the predicted label. There are four possibilities:

(a) True Positive: $r = +$, $y = +$.

(b) False Positive: $r = -$, $y = +$.

(c) True Negative: $r = -$, $y = -$.

(d) False Negative: $r = +$, $y = -$.

The words "true" and "false" here mean "correct" and "incorrect". For example, a False Positive prediction is one where you predict +, but the correct label is $-$.

The above terminology can also be used in describing the results of applying a spam classifier to a set of emails. The spam classifier labels an email as "positive" if it is being classified as spam, and "negative" otherwise.

The table below shows the results of the classifier on a small number of test emails. The labels (predictions) assigned by the spam classifier are given in the column labeled y. In column r, the true status of the email is shown, so if $r = +$ the email is actually spam and if $r = -$, the email is not spam.

From the column labeled y, we see that there is 1 example that was predicted to be positive, and 5 that were predicted to be negative. From the column labeled r, we see that there are 3 examples that are actually positive, and 3 that are actually negative. There are 1 false positive, 3 false negatives, 0 true positives, and 2 true negatives.

Table 1: Test Set with r and y values

Message Number	r	y
1	-	+
2	+	-
3	+	-
4	-	-
5	-	-
6	+	-

We can summarize the results in a *Confusion Matrix.* The format of a confusion matrix is shown in Table 2.

The columns of the confusion matrix correspond to the predicted labels (y), and the rows correspond to the correct labels (r). The entry in Row +, Column + should contain the total number of true positives in the test set (TP stands for true positives). The entry in Row +, Column - should contain the total number false negatives (FN stands for false negatives). Similarly, the entries in Row - should contain the total number of false positives (FP) and true negatives (TN).

The confusion matrix for the test set with predictions, given in Table 1, is shown in Table 3.

Important Note: In some confusion matrices, the *rows* will correspond to the predicted labels, and the *columns* will correspond to the correct labels. It is therefore important to always read the information that goes along with the confusion matrix, to determine the meaning of rows and columns, before trying to interpret the numbers in it.

(a) The True Positive Rate is equal to

$$\frac{\#\text{True Positive examples}}{\#\text{of examples whose correct label is } +}$$

which is equal to

$$\frac{\#\text{TP}}{\#\text{TP} + \#\text{FN}}$$

What is the true positive rate for the examples shown in Table 1?
(Note: True Positive Rate is sometimes called Sensitivity or Recall.)

Table 2: Confusion Matrix

		predicted (y)	
		+	-
correct (r)	+	TP	FN
	-	FP	TN

Table 3: Confusion Matrix

		predicted (y)	
		+	-
correct (r)	+	0	3
	-	1	2

(b) The False Positive Rate is equal to

$$\frac{\text{\#False Positive examples}}{\text{\#of examples whose correct label is -}}$$

which is equal to

$$\frac{\text{\#FP}}{\text{\#FP + \#TN}}$$

What is the false positive rate for the above results?

(Note: False Positive Rate is sometimes called the False Alarm Rate.)

(c) Table 4 is the confusion matrix for a hypothesis on a different test set.

Table 4: Confusion Matrix

		predicted	
		+	-
correct	+	56	2
	-	1	41

From the information in this confusion matrix, what was the *accuracy* of the hypothesis on the test set?

8. In practice, there can be costs (monetary or otherwise) to making prediction errors. The cost of a false positive can be greater than the cost of a false negative, or vice versa.

Suppose we are predicting whether an email is spam (+) or not spam (-). A false positive means that a non-spam email goes into the spam folder. Because people often don't check their spam folders, false positives can cause a user to miss important emails. A false negative means that a spam email ends up in the users regular email box, which is annoying to the user. It might also be harmful if the spam emails are, e.g., phishing emails.

Assume we are told the cost of making a false positive prediction, and the cost of making a false negative prediction. (Also assume that a correct prediction doesn't result in any cost.)

(a) If we estimate posterior probabilities $P(C|x)$, we can make the prediction that minimizes *expected risk*, meaning that we can predict whichever class will result in smaller expected cost.

If a false negative costs a units and a false positive costs b units, then predicting negative has an expected cost of $a * P(C = +|x)$, and predicting positive has an expected cost of $b * P(C = -|x)$.

Now consider a spam classification problem where we calculate two numerical attributes for an email, x_1 and x_2. Suppose we estimate the probability that the email is spam using the following formula: $P(spam|x_1, x_2) = \frac{1}{1+e^{-(3x_2-2x_1+1)}}$.

Consider a particular email with attribute values $x_1 = 3$, $x_2 = 2$. (pay attention to which one is x_1 and which is x_2, to avoid mistakes)

If a false negative prediction (predicting that an email is non-spam when it is spam) costs 5 units, and a false positive prediction costs 2 units, which classification has smaller expected risk: the spam classification, or the not spam classification? To answer this question, use the formula above for estimating the probability that the email is spam. Show your work.

9. Let D be a Gaussian distribution over the real numbers, and let $\mathcal{X}$ be an iid sample of size N from distribution D.

   The usual way of estimating the mean of D from $\mathcal{X}$ is to calculate the sample mean

$$\frac{\sum_{x \in \mathcal{X}} x}{N}$$

   Suppose we make a mistake in computing the size of $\mathcal{X}$, and we instead estimate the mean of D to be

$$\frac{\sum_{x \in \mathcal{X}} x}{N+1}$$

   What is the bias of this estimator?

10. Consider the following small labeled dataset for a binary classification problem (classes + and -). There are two real-valued attributes, x_1 and x_2.

    (In practice, we wouldn't want to use a dataset this small to learn, but this is just for practice.)

x1	x2	label
2.7	4.8	+
3.2	5.1	+
-0.4	-0.3	+
0.6	0.5	-
1.8	2.8	-
2.1	4.3	-

    (a) One approach to learning with a dataset like this one is to use Gaussian Naive Bayes. Here, we consider a different approach: Fit one multivariate Gaussian to the data from the positive class, and another mulitivariate Gaussian to the data from the negative class.[2] To classify a new example, predict using the MAP hypothesis.

    (Since we have 2 variables, the two multivariate Gaussians are actually "bivariate" Gaussians. Sometimes people do not use matrix notation when describing the parameters of a bivariate Gaussian. However, we will use matrix notation here because it used for Gaussians in more than 2 dimensions, and we will also be using it later in the course.)

    In order to fit the two multivariate Gaussians, we need to compute estimates $\hat{\mu} = [\mu_1, \ldots, \mu_d]^T$ and $\hat{\mathbf{\Sigma}}$ of the mean and covariance matrix respectively, for each of the two distributions. (Note that the examples $\mathbf{x} \in \mathcal{X}$ are column vectors, and so is $\hat{\mu}$.)

    Calculate these estimates for the above dataset, using the ML estimates. The formulas for calculating the ML estimates of the mean and covariance matrix of a multivariate Gaussian, from a sample $\mathcal{X} = \{\mathbf{x}^t\}_{t=1}^N$, are as follows:

$$\hat{\mu} = \frac{\sum_{t=1}^N \mathbf{x}^t}{N}$$

[2] If you need to learn about or review the multivariate Gaussian distribution, I recommend watching Andrew Ng's Coursera video, `https://www.coursera.org/learn/machine-learning/lecture/Cf8DF/multivariate-gaussian-distribution` Or see `http://cs229.stanford.edu/section/gaussians.pdf`, a probability and statistics textbook, or Alpaydin, Section 5.4. This distribution is sometimes called the multivariate Normal distribution. You will not need to know much about this distributions – just enough to do the homework problems.

$$\hat{\mathbf{\Sigma}} = \frac{\sum_{t=1}^{N}(\mathbf{x}^t - \hat{\mu})(\mathbf{x}^t - \hat{\mu})^T}{N}$$

(Make sure to show which parameters belong to which distribution.)

(b) The pdf for a d-dimensional multivariate Gaussian is

$$\frac{1}{(2\pi)^{d/2}|\Sigma|^{1/2}} e^{-\frac{1}{2}(\mathbf{x}-\mu)^T\Sigma^{-1}(\mathbf{x}-\mu)}$$

Let $\mathbf{x} = \begin{pmatrix} 1.6 \\ 2.3 \end{pmatrix}$. Using the estimated multivariate Gaussians for the positive and negative classes, compute $\log p(\mathbf{x}|+)$ and $\log p(\mathbf{x}|-)$. Write down the computed values. Then predict the class of the example $\mathbf{x}$ using the ML hypothesis.

(c) Instead of using the approach above, which requires us to estimate separate covariance matrices for each class, we could assume that the covariance matrix for both classes is the same. Then, we could estimate a single covariance matrix using the entire dataset, rather than two covariances matrices. Give one reason it might be a good idea to do this, and one reason why it might not.

Part II: Programming Exercise

There are two programming exercises. In the first exercise, you will write the code to predict housing prices in Boston. In the second exercise, you will experiment with a sentiment analysis dataset.

Programming question 1

1. In this problem you will experiment with a linear regression problem based on real world data. The data is from the Boston Housing dataset in scikit-learn. Your task is to estimate the price of a house in Boston using 13 attributes. Your program should do the follow:

   (a) fit a linear regression model using the closed form solution presented in class

   (b) fit a ridge regression model using the closed solution from written question 6. Use cross validation to find the best λ

   Report the following for the training and test data:

   - RSS
   - TSS
   - R^2

2. Repeat the previous exercise, but this time, by creating a polynomial transformation of degree 2 on the features of the dataset.

3. Which model performed the best? State the parameters of that model. Using this model predict the price of a house with features: $[5, 0.5, 2, 0, 4, 8, 4, 6, 2, 2, 2, 4, 5.5]$

4. If you are given a choice of predicting future housing prices using one of the models you have learned above. Which one would you choose and why.

   The Boston housing data is one of the dataset in scikit-learn. Here is some code to help you get started:

```
import sklearn
from sklearn.model_selection import train_test_split
from sklearn.datasets import load_boston

boston_data = load_boston()
X = boston_data.data
y = boston_data.target.reshape(X.shape[0], 1)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# You may use the sklearn method to create a polynomial transformation of degree 2
poly_tranform = sklearn.preprocessing.PolynomialFeatures(degree=2)
Z = poly_tranform.fit_transform(X)

#To learn more about the Boston housing data  set type:
print(boston_data.DESCR)
```

Programming question 2 In this part, you will experiment with a sentiment analysis dataset.

This dataset is a version of a dataset downloaded from `http://nifty.stanford.edu/2016/manley-urness-movie-review-sentiment/`, which was previously used in a Stanford project and a Kaggle competition. (See the page referenced in the URL for further details.)

The dataset consists of comments (short movie reviews) from Rotten Tomatoes, where people wrote above movies that they watched. The problem here is to categorize comments as being "positive" (they liked the movie) or "negative" (they didn't like the movie).

An example in this dataset consists of a comment, which is a short piece of text. To simplify processing of these comments, capitalized letters have been converted to lower case, and each comment has been separated into "tokens". We define a token to be a non-empty sequence of non-whitespace characters, separated from the other tokens by whitespace characters or newlines. So, for example

```
a delightful , if minor , movie .
```

has 7 different tokens, and 8 total tokens (the comma token appears twice).

The training data appears in the file `reviewstrain.txt`. Each line of a file contains one comment (example). The first character in the line is the label of the example, which is either 1 (for positive) or 0 (for negative). It is NOT part of the example.

There is also a test file, `reviewstest.txt`. The first character in each line of the test file is the label of the example. It is NOT part of the example.

1. (a) Implement a version of k-Nearest Neighbor to classify the test examples, using the training examples.

   Use the following distance function. For two comments c_1 and c_2, define the distance between c_1 and c_2 to be $\frac{1}{|inter(c_1,c_2)|}$, where $inter(c_1, c_2)$ is the set of distinct tokens appearing in both c_1 and c_2.

   For example, if c_1 is `boy oh boy bad .` and c_2 is `oh boy this is great .`, then $inter(c_1, c_2)$ is { boy, oh, . }, which is a set of size 3. Therefore, the distance between c_1 and c_2 is $\frac{1}{3}$.

   Run your algorithm with $k = 1$ and with $k = 5$. For $k = 1$, if there is more than one "nearest neighbor" for a test example (because they are all at the same distance from the test example), predict the label that is most common among these nearest neighbors. If there is a tie for the most common label, predict 1. For example, suppose the closest neighbor is at distance 1/5, there are 4 neighbors at distance 1/5, and they have labels $0, 0, 1, 0$. Then predict 0. But if the labels are $0, 0, 1, 1$, there is a tie for the most common label, so predict 1.

For $k = 5$, do something analogous. Sort the training examples by distance from the test example, smallest to largest. If there are other neighbors at the same distance as the 5th one on this list, include them also, and predict the majority class. (If there is a tie for the majority class, predict 1) For example, if the 8 top examples in the sorted list are at distance 1/8, 1/8, 1/8, 1/7, 1/7, 1/7, 1/6, 1/6, then consider all 6 examples at distance 1/8 or 1/7 (since the 5th element in this list is at distance 1/7, we include all examples at distance 1/7). Predict the majority label among these examples, or 1 if there is a tie for the majority label.

Important: When you run your nearest neighbor algorithm, make sure that you do NOT include the labels when you compute the distance between two examples!

Note: You may be tempted to store the training examples in a matrix A, with one row per comment, one column for each possible token, where $A[i, j] = 1$ if token j appears in example i, and $A[i, j] = 0$ otherwise. This will result in a very large matrix, with many entries that are zeroes. It is better to use a sparse matrix (scipy.sparse), or to use other data structures that take less space and support more efficient computation.

Answer the following questions:

i. For $k = 1$, what is the predicted label for the following example in the test set: `It leaves little doubt that Kidman has become one of our best actors .` (This is line 18 of the test file.)

ii. What is the confusion matrix (on the test set) for $k = 1$?

iii. Report the accuracy, the true positive rate, and the false positive rate, on the test set for $k = 1$.

iv. For $k = 5$, what is the predicted label for the following example in the test set: `It leaves little doubt that Kidman has become one of our best actors .` (This is line 18 of the test file.)

v. What is the confusion matrix (on the test set) for $k = 5$?

vi. Report the accuracy, the true positive rate, and the false positive rate, on the test set for $k = 5$.

vii. What is the accuracy on the test set for $k = 5$?

viii. Suppose we used the very simple Zero-R classifier on this dataset, rather than k-NN. That is, we classify all examples in the test set as belonging to the class that is more common in the training set. What is the resulting confusion matrix (on the test set)?

(b) In the dataset we are using here, the examples (comments) are all fairly short, containing relatively few tokens. Suppose that we had a dataset consisting of documents of very different lengths, ranging from e.g., 10 tokens in length, to 10,000 tokens in length. If we applied k-NN to such a dataset, the distance function we are using here might not be a good choice. Why not?

(c) Implement 5-fold cross-validation on the training set to determine which of the following values of k works better in k-NN: 3, 7, 99. (When there are more than k possible nearest neighbors, because of multiple points at the same distance from the test set, handle this analogously to how you handled it in part 1.)

More particularly, to implement the 5-fold cross-validation, divide the training set into 5 sets of equal size. In practice you may want to randomly permute the data before dividing it into 5 sets, but for this assignment, just take the first 1/5 of the examples listed in the file, then the second 1/5, etc. (and DON'T PERMUTE) the examples first.

Then for each of the 3 values of k, do the following. (1) For each of the 5 sets, train on the examples in the other four sets, and test on the examples in the 5th set. The result is that a prediction has been made on each example in the training set. (2) Calculate the percentage of these predictions that were correct. This is the cross-validation accuracy (for this value of k).

i. For each of the 3 values of k, what is the cross-validation accuracy?

ii. Take the k that had the highest cross-validation accuracy. Run k-NN on the entire training set for this value of k, and then test on the test set. Give the confusion matrix and the accuracy (for the test set).

(d) Experiment with using a different distance function. Add an option to your program which allows the user to either choose the distance function above, or to use a second distance function that you define.

There are many different ways to define the distance function. You may want to explore ways of defining the tokens differently (e.g., by removing common "stop-words" or punctation from the examples), ways of giving tokens different weights depending on how common they are (e.g., tf-idf weighting), and/or some other standard ways of defining the distance between two vectors representing two documents (e.g., cosine similarity).

You can choose whatever distance function you like, but should choose something that you think might yield higher accuracy than the first distance function.

Run k-NN with your distance function, using the same training and test sets, to classify the examples in the test set.

i. Describe your distance function. How is the distance between two comments computed? Include an example in your explanation.

ii. Why did you think that your distance function would do better than the first one?

iii. What is the confusion matrix for $k = 1$?

iv. Report the accuracy, the true positive rate, and the false positive rate, on the test set for $k = 1$?

v. What is the confusion matrix for $k = 5$?

vi. Report the accuracy, the true positive rate, and the false positive rate, on the test set for $k = 5$?

vii. Did your distance function achieve higher accuracy (for $k = 1$ and $k = 5$) than the first distance function? If it didn't, what is a possible reason that it didn't?