HOMEWORK 3

Mondo Jiang
gjiang25 <9085879535>

Instructions: Use this latex file as a template to develop your homework. Submit your homework on time as a single pdf file to Canvas. Late submissions may not be accepted. Please wrap your code and upload to a public GitHub repo, then attach the link below the instructions so that we can access it. You can choose any programming language (i.e. python, R, or MATLAB). Please check Piazza for updates about the homework.

https://github.com/MondoGao/uwm-cs760-hw3

1 Questions (50 pts)

1. (9 pts) Explain whether each scenario is a classification or regression problem. And, provide the number of data points (n) and the number of features (p).

   (a) (3 pts) We collect a set of data on the top 500 firms in the US. For each firm we record profit, number of employees, industry and the CEO salary. We are interested in predicting CEO salary with given factors.

type	Regression
number of data points	500
number of features	3, which are profit, number of employees, industry

   (b) (3 pts) We are considering launching a new product and wish to know whether it will be a success or a failure. We collect data on 20 similar products that were previously launched. For each product we have recorded whether it was a success or failure, price charged for the product, marketing budget, competition price, and ten other variables.

type	Classification
number of data points	20
number of features	13, which are price charged for the product, marketing budget, competition price, and ten other variables.

   (c) (3 pts) We are interesting in predicting the % change in the US dollar in relation to the weekly changes in the world stock markets. Hence we collect weekly data for all of 2012. For each week we record the % change in the dollar, the % change in the US market, the % change in the British market, and the % change in the German market.

type	Regression
number of data points	52
number of features	3, which are the % change in the US market, the % change in the British market, and the % change in the German market.

2. (6 pts) The table below provides a training data set containing six observations, three predictors, and one qualitative response variable.

X_1	X_2	X_3	Y
0	3	0	Red
2	0	0	Red
0	1	3	Red
0	1	2	Green
-1	0	1	Green
1	1	1	Red

Suppose we wish to use this data set to make a prediction for Y when $X_1 = X_2 = X_3 = 0$ using K-nearest neighbors.

(a) (2 pts) Compute the Euclidean distance between each observation and the test point, $X_1 = X_2 = X_3 = 0$.

X_1	X_2	X_3	D	Y
0	3	0	$\sqrt{3^2} = 3$	Red
2	0	0	$\sqrt{2^2} = 2$	Red
0	1	3	$\sqrt{1^2 + 3^2} = \sqrt{10}$	Red
0	1	2	$\sqrt{1^2 + 2^2} = \sqrt{5}$	Green
-1	0	1	$\sqrt{1^2 + 1^2} = \sqrt{2}$	Green
1	1	1	$\sqrt{1^2 + 1^2 + 1^2} = \sqrt{3}$	Red

(b) (2 pts) What is our prediction with $K = 1$? Why?

Prediction: **Green**

Reason: The nearest neighbor is the 5th ($\sqrt{2}$), which is Green.

(c) (2 pts) What is our prediction with $K = 3$? Why?

Prediction: **Red**

Reason: The nearest neighbors are the 5th ($\sqrt{2}$), 6th ($\sqrt{3}$), and 2nd ($\sqrt{4} = 2$), which are two Reds and 1 Green.

3. (12 pts) When the number of features p is large, there tends to be a deterioration in the performance of KNN and other local approaches that perform prediction using only observations that are near the test observation for which a prediction must be made. This phenomenon is known as the curse of dimensionality, and it ties into the fact that non-parametric approaches often perform poorly when p is large.

   (a) (2pts) Suppose that we have a set of observations, each with measurements on $p = 1$ feature, X. We assume that X is uniformly (evenly) distributed on [0, 1]. Associated with each observation is a response value. Suppose that we wish to predict a test observation's response using only observations that are within 10% of the range of X closest to that test observation. For instance, in order to predict the response for a test observation with $X = 0.6$, we will use observations in the range [0.55, 0.65]. On average, what fraction of the available observations will we use to make the prediction?

   0.1

   (b) (2pts) Now suppose that we have a set of observations, each with measurements on $p = 2$ features, $X1$ and $X2$. We assume that predict a test observation's response using only observations that $(X1, X2)$ are uniformly distributed on [0, 1] × [0, 1]. We wish to are within 10% of the range of $X1$ and within 10% of the range of $X2$ closest to that test observation. For instance, in order to predict the response for a test observation with $X1 = 0.6$ and $X2 = 0.35$, we will use observations in the range [0.55, 0.65] for $X1$ and in the range [0.3, 0.4] for $X2$. On average, what fraction of the available observations will we use to make the prediction?

   $0.1 \times 0.1 = 0.01$

   (c) (2pts) Now suppose that we have a set of observations on $p = 100$ features. Again the observations are uniformly distributed on each feature, and again each feature ranges in value from 0 to 1. We wish to predict a test observation's response using observations within the 10% of each feature's range that is closest to that test observation. What fraction of the available observations will we use to make the prediction?

   0.1^{100}

   (d) (3pts) Using your answers to parts (a)–(c), argue that a drawback of KNN when p is large is that there are very few training observations "near" any given test observation.

   As we can see from (a) - (c), when p grows larger, the training observations we can use are decresing exponentially. Which means in high dimension space, there are fewer training observations we can use to make predictions, and the model's might not be able to perform well.

   (e) (3pts) Now suppose that we wish to make a prediction for a test observation by creating a p-dimensional hypercube centered around the test observation that contains, on average, 10% of the training observations. For p =1, 2, and 100, what is the length of each side of the hypercube? Comment what happens to the length of the sides as $\lim_{n\to\infty}$.

   i. When p = 1: 0.1
   ii. When p = 2: the hypercude is a square whose area is 0.1, so the length of each side is $\sqrt{0.1}$.
   iii. When p = 100: same logic, $\sqrt[100]{0.1}$
   iv. When $\lim_{p\to\infty}$: the length of each side ($\lim_{p\to\infty} \sqrt[p]{0.1}$) approaches 0.

4. (6 pts) Supoose you trained a classifier for a spam detection system. The prediction result on the test set is summarized in the following table.

		Predicted class	
		Spam	not Spam
Actual class	Spam	8	2
	not Spam	16	974

   Calculate

   (a) (2 pts) Accuracy $= \frac{TP+TN}{FP+FN+TP+TN} = \frac{8+974}{2+16+8+974} = \frac{982}{1000} = 0.982$

   (b) (2 pts) Precision $= \frac{TP}{TP+FP} = \frac{8}{8+16} = \frac{8}{24} = 0.333$

   (c) (2 pts) Recall $= \frac{TP}{TP+FN} = \frac{8}{8+2} = \frac{8}{10} = 0.8$

5. (9pts) Again, suppose you trained a classifier for a spam filter. The prediction result on the test set is summarized in the following table. Here, "+" represents spam, and "-" means not spam.

Confidence positive	Correct class
0.95	+
0.85	+
0.8	-
0.7	+
0.55	+
0.45	-
0.4	+
0.3	+
0.2	-
0.1	-

(a) (6pts) Draw a ROC curve based on the above table.

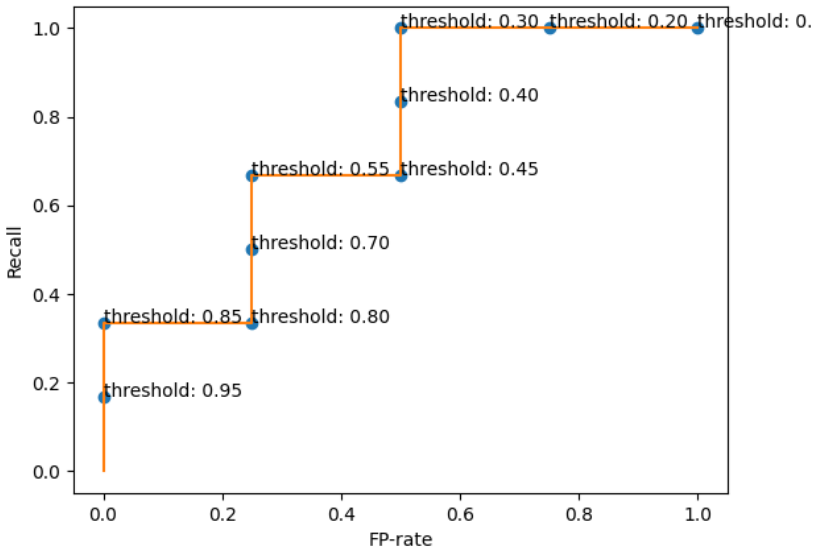
Say our classifier has a confidence positive threshold, when the confidence positive is above or equal to the threshold, we classify the email as spam (+).

For example, when the threshold is 0.5, we classify the first 5 emails as spam, and the last 5 emails as not spam. Then we can get a similar table with previous question:

		Predicted class	
		Spam	not Spam
Actual class	Spam	4	2
	not Spam	1	3

Recall: $= \frac{TP}{TP+FN} = \frac{4}{4+2} = \frac{2}{3}$
FP-rate: $= \frac{FP}{FP+TN} = \frac{1}{1+3} = \frac{1}{4}$

Repeat this process, we can get the following ROC curve:



(b) (3pts) (Real-world open question) Suppose you want to choose a threshold parameter so that mails with confidence positives above the threshold can be classified as spam. Which value will you choose? Justify your answer based on the ROC curve.

I'll choose 0.85, which has a 0.33 recall and a 0 FP-rate.

Reason: I don't want to classify any not spam emails as spam. So I want to choose a threshold that has a lowest FP-rate. From the ROC curve, we can see 0.85 has the hignest recall when FP-rate = 0.

6. (8 pts) In this problem, we will walk through a single step of the gradient descent algorithm for logistic regression. As a reminder,

$$\hat{y} = f(x, \theta)$$
$$f(x; \theta) = \sigma(\theta^\top x)$$
$$\text{Cross entropy loss } L(\hat{y}, y) = -[y \log \hat{y} + (1-y)\log(1-\hat{y})]$$
$$\text{The single update step } \theta^{t+1} = \theta^t - \eta \nabla_\theta L(f(x;\theta), y)$$

(a) (4 pts) Compute the first gradient $\nabla_\theta L(f(x;\theta), y)$.

$$\begin{aligned} L(\hat{y}, y) &= -[y \log \hat{y} + (1-y)\log(1-\hat{y})] \\ &= -[y \log \sigma(\theta^\top x) + (1-y)\log(1-\sigma(\theta^\top x))] \end{aligned}$$

$$\begin{aligned} \frac{\partial L}{\partial \theta} &= -\left[y \cdot \frac{1}{\hat{y}} \cdot \frac{\partial \hat{y}}{\partial \theta} + (1-y) \cdot \frac{1}{1-\hat{y}} \cdot \frac{\partial (1-\hat{y})}{\partial \theta}\right] \\ &= -\left[y \left(\frac{1}{\sigma(\theta^\top x)}\right) \sigma(\theta^\top x)\left(1-\sigma(\theta^\top x)\right) x + (1-y)\left(-\frac{1}{1-\sigma(\theta^\top x)}\right)\sigma(\theta^\top x)\left(1-\sigma(\theta^\top x)\right) x\right] \\ &= -\left[y(1-\sigma(\theta^\top x))x - (1-y)\sigma(\theta^\top x)x\right] \\ &= (\hat{y} - y) \cdot x \end{aligned}$$

(b) (4 pts)

$$\text{Initial parameters : } \theta^0 = [0, 0, 0]$$
$$\text{Learning rate } \eta = 0.1$$
$$\text{data example : } x = [1, 3, 2], y = 1$$

Compute the updated parameter vector θ^1 from the single update step.

$$\begin{aligned} \hat{y} = \sigma(\theta^T x) &= \frac{1}{1 + e^{-\theta^T x}} = \frac{1}{1+e^0} \\ &= 0.5 \end{aligned}$$

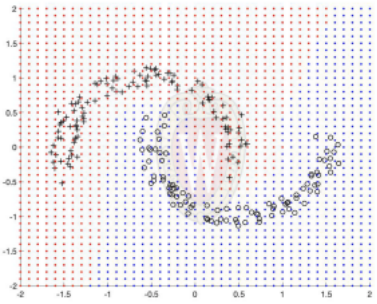
$$\begin{aligned} \nabla_\theta L(\hat{y}, y) &= (\hat{y} - y) \cdot x \\ &= (0.5 - 1) \cdot [1, 3, 2] \\ &= (-0.5) \cdot [1, 3, 2] \\ &= [-0.5, -1.5, -1] \end{aligned}$$

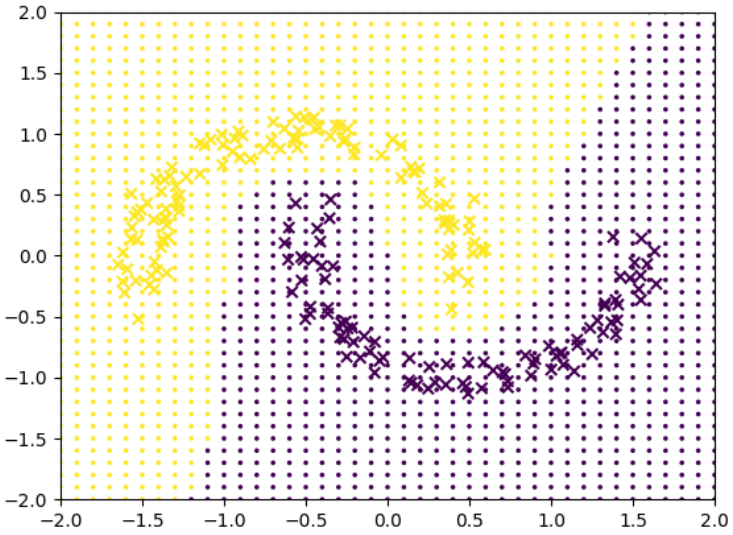
$$\begin{aligned} \theta_1 &= \theta_0 - \eta \nabla_\theta L(\hat{y}, y) \\ &= [0, 0, 0] - 0.1 \cdot [-0.5, -1.5, -1] \\ &= [0.05, 0.15, 0.1] \end{aligned}$$

2 Programming (50 pts)

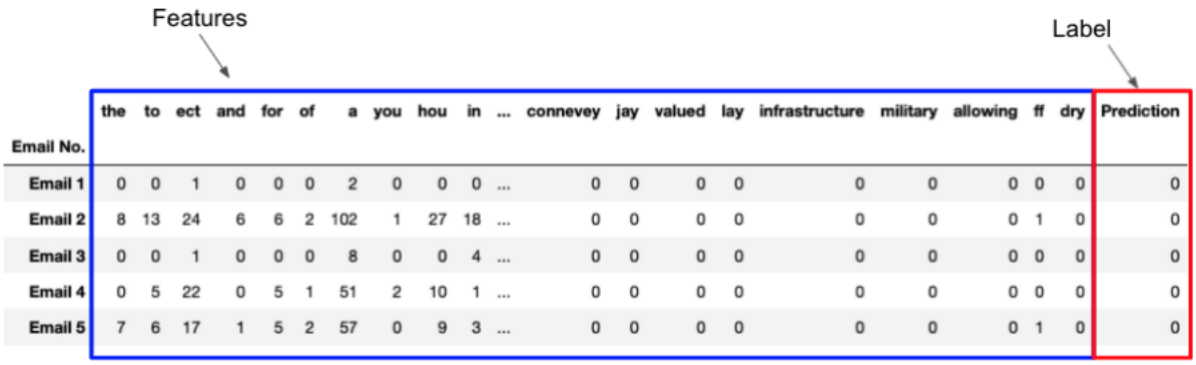
1. (10 pts) Use the whole D2z.txt as training set. Use Euclidean distance (i.e. $A = I$). Visualize the predictions of 1NN on a 2D grid $[-2 : 0.1 : 2]^2$. That is, you should produce test points whose first feature goes over $-2, -1.9, -1.8, \ldots, 1.9, 2$, so does the second feature independent of the first feature. You should overlay the training set in the plot, just make sure we can tell which points are training, which are grid.

   The expected figure looks like this.





Spam filter Now, we will use 'emails.csv' as our dataset. The description is as follows.



Email No.	the	to	ect	and	for	of	a	you	hou	in	...	connevey	jay	valued	lay	infrastructure	military	allowing	ff	dry	Prediction
Email 1	0	0	1	0	0	0	2	0	0	0	...	0	0	0	0	0	0	0	0	0	0
Email 2	8	13	24	6	6	2	102	1	27	18	...	0	0	0	0	0	0	0	1	0	0
Email 3	0	0	1	0	0	0	8	0	0	4	...	0	0	0	0	0	0	0	0	0	0
Email 4	0	5	22	0	5	1	51	2	10	1	...	0	0	0	0	0	0	0	0	0	0
Email 5	7	6	17	1	5	2	57	0	9	3	...	0	0	0	0	0	0	0	1	0	0

- Task: spam detection
- The number of rows: 5000
- The number of features: 3000 (Word frequency in each email)
- The label (y) column name: 'Predictor'
- For a single training/test set split, use Email 1-4000 as the training set, Email 4001-5000 as the test set.
- For 5-fold cross validation, split dataset in the following way.
  - Fold 1, test set: Email 1-1000, training set: the rest (Email 1001-5000)
  - Fold 2, test set: Email 1000-2000, training set: the rest
  - Fold 3, test set: Email 2000-3000, training set: the rest
  - Fold 4, test set: Email 3000-4000, training set: the rest
  - Fold 5, test set: Email 4000-5000, training set: the rest

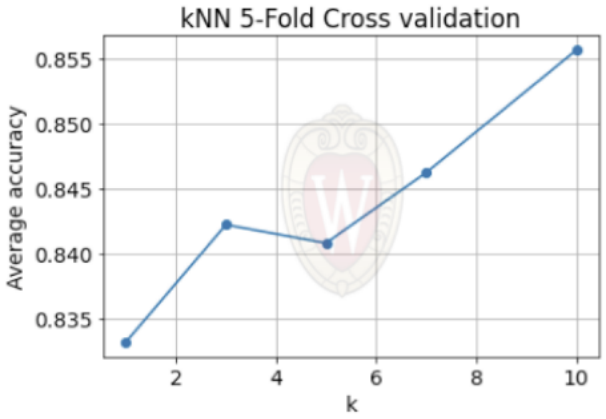
2. (8 pts) Implement 1NN, Run 5-fold cross validation. Report accuracy, precision, and recall in each fold.

Fold	Accuracy	Precision	Recall
1	0.825	0.6544943820224719	0.8175438596491228
2	0.853	0.6857142857142857	0.8664259927797834
3	0.862	0.7212121212121212	0.8380281690140845
4	0.851	0.7164179104477612	0.8163265306122449
5	0.775	0.6057441253263708	0.7581699346405228

3. (12 pts) Implement logistic regression (from scratch). Use gradient descent (refer to question 6 from part 1) to find the optimal parameters. You may need to tune your learning rate to find a good optimum. Run 5-fold cross validation. Report accuracy, precision, and recall in each fold.

Fold	Accuracy	Precision	Recall
1	0.86975	0.8047619047619048	0.7278208440999139
2	0.87675	0.8414141414141414	0.7125748502994012
3	0.8875	0.7951907131011609	0.8253012048192772
4	0.79025	0.9086161879895561	0.3020833333333333
5	0.891	0.7988115449915111	0.8254385964912281

4. (10 pts) Run 5-fold cross validation with kNN varying k (k=1, 3, 5, 7, 10). Plot the average accuracy versus k, and list the average accuracy of each case.
   Expected figure looks like this.



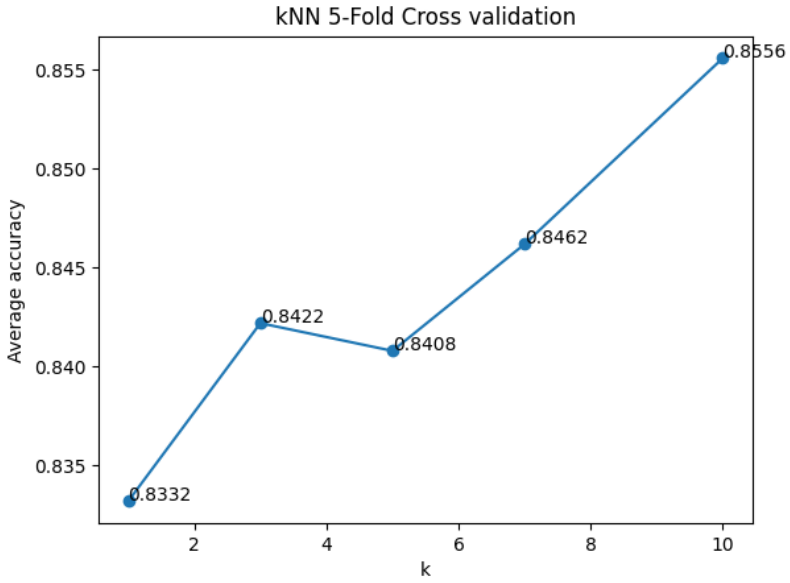
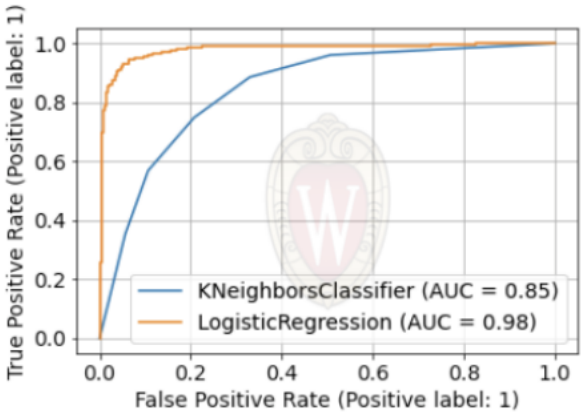


figure result from skilearn[1]

[1] I used skilearn's result to draw this figure, and the result is slightly different from my own implementation. I do implement two versions of the algorithm (q2-4.py and q2-4_ski.py), and I give my algorithm the same interface with skilearn, but it just run too slow. When I test the first fold, I get a same result from both algorithm, so I decide to put data from skilearn here.

k	Accuracy
1	0.8332
3	0.8422
5	0.8408
7	0.8462
10	0.8556

5. (10 pts) Use a single training/test setting. Train kNN (k=5) and logistic regression on the training set, and draw ROC curves based on the test set.
Expected figure looks like this. Note that the logistic regression results may differ.



Traning Set: Email 1-2500
Test Set: Email 2501-5000