CSE 151A - Homework 2

Ashwin Tyagi. A16228593

Problem 1. *Lp norms*

Solution. $x = (1, 2, 3, 4)$

$$||x||_1 = |1 + 2 + 3 + 4|$$
$$||x||_1 = 10$$

$$||x||_2 = \sqrt{1^2 + 2^2 + 3^2 + 4^2}$$
$$||x||_2 = \sqrt{1 + 4 + 9 + 16}$$
$$||x||_2 = \sqrt{30}$$

$$||x||_\infty = max(x)$$
$$||x||_\infty = 4$$

Problem 2. *Comparing the ℓ_1, ℓ_2, ℓ_∞ norms.*

Solution.

(a) Given $||x||_\infty = 1$, the largest ℓ_1 and ℓ_2 possible occurs when $|x| = 1$.
Therefore, the largest ℓ_1 possible is $||x||_1 = d$.
The largest ℓ_2 possible is $||x||_2 = \sqrt{d}$.

(b) Given $||x||_2 = 1$, the largest ℓ_1 and ℓ_2 possible occurs when $x = \frac{1}{\sqrt{d}}$.
Therefore, the largest ℓ_1 possible is $||x||_1 = d \times \frac{1}{\sqrt{d}} = \sqrt{d}$.
The largest ℓ_∞ would be $||x||_\infty = \frac{1}{\sqrt{d}}$

Problem 3. *Is the distance function given by the table for $\mathcal{X} = (A, B, C, D)$ a metric.*

- *Non-negativity* - **yes**
  All the distances in the table are positive. Therefore there is no combination for $\mathcal{X}$ that can be non-negative.

- $d(x, y) = 0$ *iff* $x = y$ - **yes**
  The only distances of 0 occur on the diagonal of the table so that $d(A, A) = d(B, B) = d(C, C) = d(D, D) = 0$.

- *symmetry* - **yes**
  The table is symmetric across the diagonal so that $d(x, y) = d(y, x)$
- *Triangle Inequality* - **no**
  The triangle inequality states that $d(x, z) \leq d(x, y) + d(y, z)$. This does not hold.

  $$d(A, D) \leq d(A, C) + d(C, D)$$
  $$5 \leq 1 + 2$$
  $$5 \not\leq 3$$

  Since the distance from A to D is greater than the distance from A to C to D, the triangle inequality does not hold.

Problem 4. *Which of these distance functions are metrics*

Solution.

(a) Let $\mathcal{X} = \mathbb{R}$ and define $d(x, y) = x - y$. **This is not a metric.**

- *Non-negativity* - **no** If $x = 0$ and $y = 1$, then $d(x, y) = -1$, which breaks non-negativity
- *$d(x, y) = 0$ iff $x = y$* - **yes**
  By the rules of addition and subtraction, the only way for $d(x, y) = 0$ is if $x = y$
- *symmetry* - **no**
  Let $x = 0$ and $y = 1$.

  $$d(x, y) = d(y, x)$$
  $$0 - 1 = 1 - 0$$
  $$-1 \neq 1$$

  Therefore, the distance function is not symmetric.
- *Triangle Inequality* - **no**
  The triangle inequality states that $d(x, z) \leq d(x, y) + d(y, z)$. This does not hold Let $x = 0$, $z = 1$, and $y = -10$.

  $$d(x, z) \leq d(x, y) + d(y, z)$$
  $$0 - 1 \leq (0 - (-10)) + ((-10) - 1)$$
  $$-1 \not\leq -21$$

(b) Let Σ be a finite set and $\mathcal{X} = \Sigma^m$ The *Hamming distance* on $\mathcal{X}$ is $d(x, y) = \#$ of positions on which x and y differ. **This is a metric.**

- *Non-negativity* - **yes** The best case scenario for two strings of equal length is if they are the same. In this case the Hamming distance between x and y would be 0. It is impossible to have less than 0 differences between the two strings so non-negativity is maintained.

- $d(x, y) = 0$ *iff* $x = y$ - **yes**
  The only case in which $d(x, y) = 0$ is if the two strings x and y are exactly the same.

- *symmetry* - **yes**
  Since we are only counting the number of differences between two strings, it does not matter in which order we compare them. Therefore $d(x, y) = d(y, x)$ no matter what.

- *Triangle Inequality* - **yes**
  The triangle inequality states that $d(x, z) \leq d(x, y) + d(y, z)$.
  Let $x = 0000$, $z = 1111$, and $y = 0101$.

$$d(x, z) \leq d(x, y) + d(y, z)$$
$$[0000, 1111] \leq [0000, 0101] + [0101, 1111]$$
$$4 \leq 2 + 2$$
$$4 \leq 4$$

  This holds true for any three strings.

(c) Squared Euclidean distance on $\mathbb{R}^m$, that is, $d(x, y) = \sum_{i=1}^{m}(x_i - y_i)^2$. (Let's assume $m = 1$)

- *Non-negativity* - **yes** Since we are squaring the difference between x_i and y_i, the summation equation will only ever sum positive values. Therefore, Euclidean distance will always be positive.

$$d(-4, -3) = \sum_{i=1}^{1}(-4 - (-3))^2$$
$$d(-4, -3) = (-1)^2$$
$$d(-4, -3) = 1 > 0$$

- $d(x, y) = 0$ *iff* $x = y$ - **yes**
  Euclidean distance can be thought of as physical distance. Therefore, it only makes sense that the distance between two points will only be 0 if the two points are in the same physical location, ie are the same.

- *symmetry* - **yes**
  Assume $m = 1$.

$$d(x, y) = d(y, x)$$
$$(x - y)^2 = (y - x)^2$$
$$(x - y)^2 = (-(x - y)^2$$
$$(x - y)^2 = (-1)^2(x - y)^2$$
$$(x - y)^2 = (x - y)^2$$

  Therefore, the Euclidean distance is reversible.

- *Triangle Inequality* - **yes**
  By definition, Euclidean distance returns the shortest distance between two points. Since it is a "physical" measure, the addition of a third point can at the very most match the distance.

Problem 5. *KL divergence between vectors p and q*

Solution. KL divergence equation: (use base 2)

$$d(p,q) = \sum_{x \in \mathcal{X}} p(x) \log \frac{p(x)}{q(x)}$$

$$p = (\frac{1}{2}, \frac{1}{4}, \frac{1}{8}, \frac{1}{16}, \frac{1}{16}), \quad q = (\frac{1}{4}, \frac{1}{4}, \frac{1}{6}, \frac{1}{6}, \frac{1}{6})$$

$$d(p,q) = \frac{1}{2} \log \frac{1/2}{1/4} + \frac{1}{4} \log \frac{1/4}{1/4} + \frac{1}{8} \log \frac{1/8}{1/6} + \frac{1}{16} \log \frac{1/16}{1/6} + \frac{1}{16} \log \frac{1/16}{1/6}$$

$$d(p,q) = \frac{1}{2} \log 2 + \frac{1}{4} \log 1 + \frac{1}{8} \log \frac{3}{4} + \frac{1}{16} \log \frac{3}{8} + \frac{1}{16} \log \frac{3}{8}$$

$$d(p,q) = \frac{1}{2}(1) + \frac{1}{4}(0) + \frac{1}{8}(-0.415) + \frac{1}{16}(-1.415) + \frac{1}{16}(-1.415)$$

$$d(p,q) = \frac{1}{2} + \frac{1}{8}(-0.415) + \frac{1}{8}(-1.415)$$

$$d(p,q) = \frac{1}{2} + \frac{1}{8}(-1.83)$$

$$d(p,q) = 0.5 - 0.229$$

$$d(p,q) = 0.271$$

Problem 6. *Classification vs Regression*

Solution.

(a) Based on sensors in a person's cell phone, predict whether they are walking, sitting, or running.
Classification: There are only three outcomes to this problem. For each outcome, a classification algorithm can make a predictable guess as to whether a person is sitting, walking, or running. Furthermore, there is a "correct" and deterministic outcome. Therefore, a classification algorithm can be used.

(b) Based on sensors in a moving car, predict the speed of the car directly in front.
Classification: The speed of the car directly in front of the present car is deterministic. That means a classification algorithm can definitively guess the car's speed.

(c) Based on a student's high-school SAT score, predict their GPA during freshman year of college.
Regression: There is little direct correlation between a student's high-school SAT score and a student's freshman year GPA. It is possible for someone to fail the SAT and maintain a 4.0, since they are really bad at taking tests. Since the output is not guaranteed, this is a regression problem.

(d) Based on a student's high-school SAT score, predict whether or not they will complete college.
Regression: Similar to the last one, there is no direct correlation between a student's high-school SAT score and a student completing college. It is possible for someone to fail the SAT and graduate valedictorian from their college and vice versa. Since the output is not guaranteed, this is a regression problem.

Problem 7. *Covariance and Correlation*

Solution.

(a) Covariance

$$cov(X,Y) = E[XY] - E[X]E[Y]$$

$$E[X] = \sum_x xp(x)$$

$$E[X] = (-1)(\frac{1}{3}) + (0)(\frac{1}{3}) + (1)(\frac{1}{3}) = 0$$

$$E[Y] = (-1)(\frac{1}{3}) + (0)(\frac{1}{3}) + (1)(\frac{1}{3}) = 0$$

$$E[XY] = (-1)(-1)(\frac{1}{3}) + (0)(0)(\frac{1}{3}) + (1)(1)(\frac{1}{3})$$

$$E[XY] = \frac{1}{3} + \frac{1}{3} = \frac{2}{3}$$

$$cov(X,Y) = E[XY] - E[X]E[Y]$$

$$cov(X,Y) = \frac{2}{3} - (0)(0)$$

$$cov(X,Y) = \frac{2}{3}$$

(b) Correlation

$$corr(X,Y) = \frac{cov(X,Y)}{std(X)std(Y)}$$

$$std(X) = \sqrt{var(X)}$$
$$std(X) = \sqrt{E((X - E[X])^2)}$$
$$std(X) = \sqrt{E((X - 0)^2)}$$
$$std(X) = \sqrt{E(X^2)}$$
$$E(X^2) = (-1)^2 p(-1) + (0^2)p(0) + (1^2)p(1)$$
$$E(X^2) = (1)p(-1) + (1)p(1)$$
$$E(X^2) = \frac{1}{3} + \frac{1}{3} = \frac{2}{3}$$
$$std(X) = \sqrt{\frac{2}{3}}$$

$$std(Y) = \sqrt{var(Y)}$$
$$std(Y) = \sqrt{E((Y - E[Y])^2)}$$
$$std(Y) = \sqrt{E((Y - 0)^2)}$$
$$std(Y) = \sqrt{E(Y^2)}$$
$$E(Y^2) = (-1)^2 p(-1) + (0^2)p(0) + (1^2)p(1)$$
$$E(Y^2) = (1)p(-1) + (1)p(1)$$
$$E(Y^2) = \frac{1}{3} + \frac{1}{3} = \frac{2}{3}$$
$$std(Y) = \sqrt{\frac{2}{3}}$$

$$corr(X,Y) = \frac{cov(X,Y)}{std(X)std(Y)}$$
$$corr(X,Y) = \frac{\frac{2}{3}}{\sqrt{\frac{2}{3}}\sqrt{\frac{2}{3}}} = \frac{2/3}{2/3}$$
$$corr(X,Y) = 1$$

Problem 8. *Independence and Uncorrelatedness*

Solution.

(a) Are X and Y Independent? - **NO**

$$p(XY) = p(X)p(Y)$$
$$p(-1 \cap -1) = p(X = -1)p(Y = -1)$$
$$\frac{1}{6} = (\frac{1}{3})(\frac{1}{3})$$
$$\frac{1}{6} \neq \frac{1}{9}$$

(b) Are X and Y uncorrelated? - **YES**

$$E[XY] = E[X]E[Y]$$
$$E[XY] = (-1)(-1)(\frac{1}{6}) + (-1)(1)(\frac{1}{6}) + (0)(0)(\frac{1}{3}) + (1)(-1)(\frac{1}{6}) + (1)(1)(\frac{1}{6})$$
$$E[XY] = \frac{1}{6} - \frac{1}{6} - \frac{1}{6} + \frac{1}{6} = 0$$
$$E[X] = (-1)(\frac{1}{3}) + (0)(\frac{1}{3}) + (1)(\frac{1}{3}) = 0$$
$$E[Y] = (-1)(\frac{1}{3}) + (0)(\frac{1}{3}) + (1)(\frac{1}{3}) = 0$$
$$E[XY] = E[X]E[Y]$$
$$0 = 0$$

Problem 9. *Binary Classification with* $+, -$.

Solution. The Generative Approach bases itself around probability. Therefore, if there are more $+$ in the data set, the chance of returning a $+$ is higher. In the case where $+$ is returned a majority of the time, it is likely because there are very few $-$'s or significantly more $+$'s.

Problem 10. *Winery Classification*

Solution. The class probabilities are $\pi_1 = 0.33$, $\pi_2 = 0.39$, and $\pi_3 = 0.28$. Optimal prediction is for label j from $\max(\pi_j p_j(x))$

(a) 12.0 - **label: 2**

$$\pi_1 p_1(j) = 0.33 * 0.0 = 0$$
$$\pi_2 p_2(j) = 0.39 * 0.7 = 0.273$$
$$\pi_3 p_3(j) = 0.28 * 0.05 = 0.014$$

(b) 12.5 - **label: 2**

$$\pi_1 p_1(j) = 0.33 * 0.0 = 0$$
$$\pi_2 p_2(j) = 0.39 * 0.6 = 0.234$$
$$\pi_3 p_3(j) = 0.28 * 0.4 = 0.112$$

(c) 13 - **label: 3**

$$\pi_1 p_1(j) = 0.33 * 0.3 = 0.099$$
$$\pi_2 p_2(j) = 0.39 * 0.3 = 0.117$$
$$\pi_3 p_3(j) = 0.28 * 0.75 = 0.21$$

(d) 13.5 - **label: 1**

$$\pi_1 p_1(j) = 0.33 * 0.6 = 0.198$$
$$\pi_2 p_2(j) = 0.39 * 0.0 = 0.0$$
$$\pi_3 p_3(j) = 0.28 * 0.6 = 0.168$$

(e) 14 - **label: 1**

$$\pi_1 p_1(j) = 0.33 * 0.9 = 0.297$$
$$\pi_2 p_2(j) = 0.39 * 0.0 = 0.0$$
$$\pi_3 p_3(j) = 0.28 * 0.2 = 0.056$$

Problem 11. *Cross-validation for nearest neighbor classification*

Solution.

(a) LOOCV code, accuracy, and Confusion Matrix

```
[6]: import numpy as np
     import matplotlib.pyplot as plt
```

```
[13]: from ucimlrepo import fetch_ucirepo

      # fetch dataset
      wine = fetch_ucirepo(id=109)

      # data (as pandas dataframes)
      raw_X = wine.data.features
      raw_Y = wine.data.targets

      X = raw_X.to_numpy()
      Y = raw_Y.to_numpy()
```

```
[99]: def squared_dist(x,y):
          return np.sum(np.square(x-y))

      def find_NN(x,k, train_data):
          # for Leave one out cross validation range will be length of data sets - 1
          distances = [squared_dist(x,train_data[i]) for i in range(len(train_data))]

          return np.argmin(distances)
```

```
[184]: #Build Confusion Matrix
       confusion = [[0,0,0],[0,0,0],[0,0,0]]
       index = 0
       errors = 0
       for x in X:
           training_set = [item for i, item in enumerate(X) if i != index]
           nn_ind = find_NN(x, len(X), training_set)

           conf_act = Y[index][0] - 1
           conf_pred = Y[nn_ind][0] - 1

           if(conf_act != conf_pred):
               errors += 1

           confusion[conf_act][conf_pred] += 1

           index+=1

       accuracy_rate =  1- errors/len(X)
       print(accuracy_rate)
       for v in confusion:
           print(*v)
```

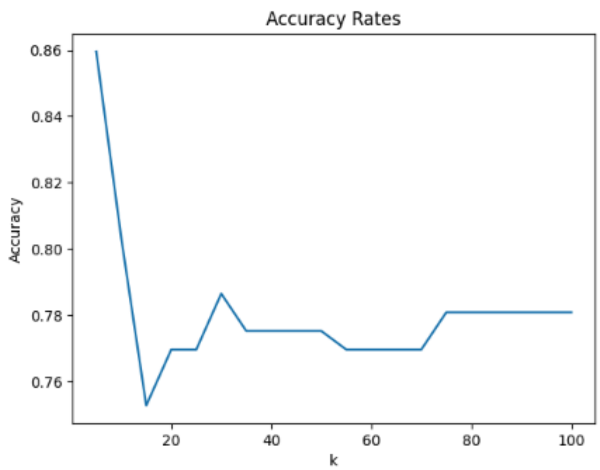
```
0.7752808988764045
52 3 4
5 55 11
3 14 31
```

(b) Unscaled Estimates

```
[173]: def find_kfold_error(k):
    index = 0
    per_fold = int(round(len(X) / k, 0))
    errors = 0

    while (index + per_fold) < len(X):

        test = [item for i, item in enumerate(X) if (i >= index and i < index+per_fold)]
        train = [item for i, item in enumerate(X) if (i < index or i >= index+per_fold)]

        i = index
        for data in test:
            nn_ind = find_NN(data, k, train)
            x_act = Y[nn_ind][0]
            x_pred = Y[i][0]

            if x_act != x_pred:
                errors+=1
            i+=1

        index+=per_fold

    return errors/len(X)

def find_kfold_error_scaled(k):
    index = 0
    per_fold = int(round(len(X) / k, 0))
    error_rates = []

    while (index + per_fold) < len(X):

        test = [item for i, item in enumerate(X) if (i >= index and i < index+per_fold)]
        train = [item for i, item in enumerate(X) if (i < index or i >= index+per_fold)]

        i = index
        errors = 0
        for data in test:
            nn_ind = find_NN(data, k, train)
            x_act = Y[nn_ind][0]
            x_pred = Y[i][0]

            if x_act != x_pred:
                errors+=1
            i+=1

        error_rates.append( errors/len(test) )

        index+=per_fold

    return sum(error_rates) / k
```

[180]:

```
#20 k values from 2-100 at an interval of 5
k_vals = []
for i in range(5, 101, 5):
    k_vals.append(i)


error_rates = []
for k in k_vals:
    error_rates.append(1-find_kfold_error(k))

figure, axis = plt.subplots()

axis.plot(k_vals, error_rates, label = 'Accuracy Rates')

axis.set_xlabel('k')
axis.set_ylabel('Accuracy')
axis.set_title('Accuracy Rates')

plt.show()
```



(c) Scaled Estimates

[189]:

```
error_rates = []
for k in k_vals:
    error_rates.append(1-find_kfold_error_scaled(k))

figure, axis = plt.subplots()

axis.plot(k_vals, error_rates, label = 'Accuracy Rates Scaled')

axis.set_xlabel('k')
axis.set_ylabel('Accuracy')
axis.set_title('Accuracy Rates Scaled')

plt.show()
```