2023-2024 **Third Year**

Machine Learning

Tutorial 3 – solution - Similarity-Based Learning

Exercise 1 (ex 2 book):

Email spam filtering models often use a **bag-of-words** representation for emails. In a bag-of-words representation, the descriptive features that describe a document (in our case, an email) each represent how many times a particular word occurs in the document. One descriptive feature is included for each word in a predefined dictionary. The dictionary is typically defined as the complete set of words that occur in the training dataset. The table below lists the bag-of-words representation for the following five emails and a target feature, SPAM, whether they are spam emails or genuine emails:

1. "*money, money, money*"
2. "*free money for free gambling fun*"
3. "*gambling for fun*"
4. "*machine learning for fun, fun, fun*"
5. "*free machine learning*"

	Bag-of-Words							
ID	MONEY	FREE	FOR	GAMBLING	FUN	MACHINE	LEARNING	SPAM
1	3	0	0	0	0	0	0	true
2	1	2	1	1	1	0	0	true
3	0	0	1	1	1	0	0	true
4	0	0	1	0	3	1	1	false
5	0	1	0	0	0	1	1	false

(a) What target label would a nearest neighbor model using **Euclidean distance** return for the following email: "*machine learning for free*"?

The bag-of-words representation for this query is as follows:

	Bag-of-Words							
ID	MONEY	FREE	FOR	GAMBLING	FUN	MACHINE	LEARNING	SPAM
Query	0	1	1	0	0	1	1	?

The table below shows the calculation of the Euclidean distance between the query instance and each of the instances in the training dataset:

	$(\mathbf{q}[i[-\mathbf{d}_j[i[)^2$							Euclidean
ID	MONEY	FREE	FOR	GAMBLING	FUN	MACHINE	LEARNING	Distance
1	9	1	1	0	0	1	1	3.6056
2	1	1	0	1	1	1	1	2.4495
3	0	1	0	1	1	1	1	2.2361
4	0	1	0	0	9	0	0	3.1623
5	0	0	1	0	0	0	0	1

Based on these distance calculations, the nearest neighbor to the query is instance $\mathbf{d}_5$, for which SPAM = *false*. Consequently, the model will return a prediction of SPAM = *false* for this query.

(b) What target label would a *k*-NN model with *k* = 3 and using **Euclidean distance** return for the same query?

Based on the distance calculations in part (a) of this question, the three nearest neighbors to the query are instances $\mathbf{d}_5$, $\mathbf{d}_3$, and $\mathbf{d}_2$. The majority of these three neighbors have a target value of SPAM = *true*. Consequently, the 3-NN model will return a prediction of SPAM = *true*.

(c) What target label would a **weighted *k*-NN** model with $k = 5$ and using a weighting scheme of the reciprocal of the squared Euclidean distance between the neighbor and the query, return for the query?

The weights for each of the instances in the dataset are

ID	Weights
1	$\frac{1}{3.6056^2} = 0.0769$
2	$\frac{1}{2.4495^2} = 0.1667$
3	$\frac{1}{2.2361^2} = 0.2$
4	$\frac{1}{3.1623^2} = 0.1$
5	$\frac{1}{1^2} = 1$

The total weight for the SPAM = *true* target level is $0.0769 + 0.1667 + 0.2 = 0.4436$. The total weight for the SPAM = *false* target level is $0.1 + 1 = 1.1$. Consequently, the SPAM = *false* has the maximum weight, and this is the prediction returned by the model.

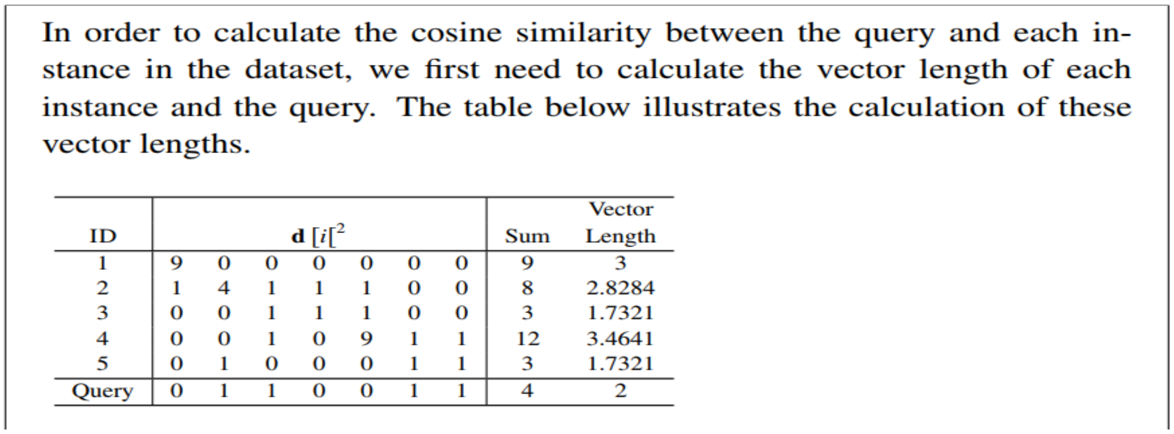
(d) What target label would a *k*-NN model with $k = 3$ and using **Manhattan distance** return for the same query?

The table below shows the calculation of the Manhattan distance between the query bag-of-words vector and each instance in the dataset:

	$abs(\mathbf{q}[i[-\mathbf{d}_j[i[)$							Manhattan
ID	MONEY	FREE	FOR	GAMBLING	FUN	MACHINE	LEARNING	Distance
1	3	1	1	0	0	1	1	7
2	1	1	0	1	1	1	1	6
3	0	1	0	1	1	1	1	5
4	0	1	0	0	3	0	0	4
5	0	0	1	0	0	0	0	1

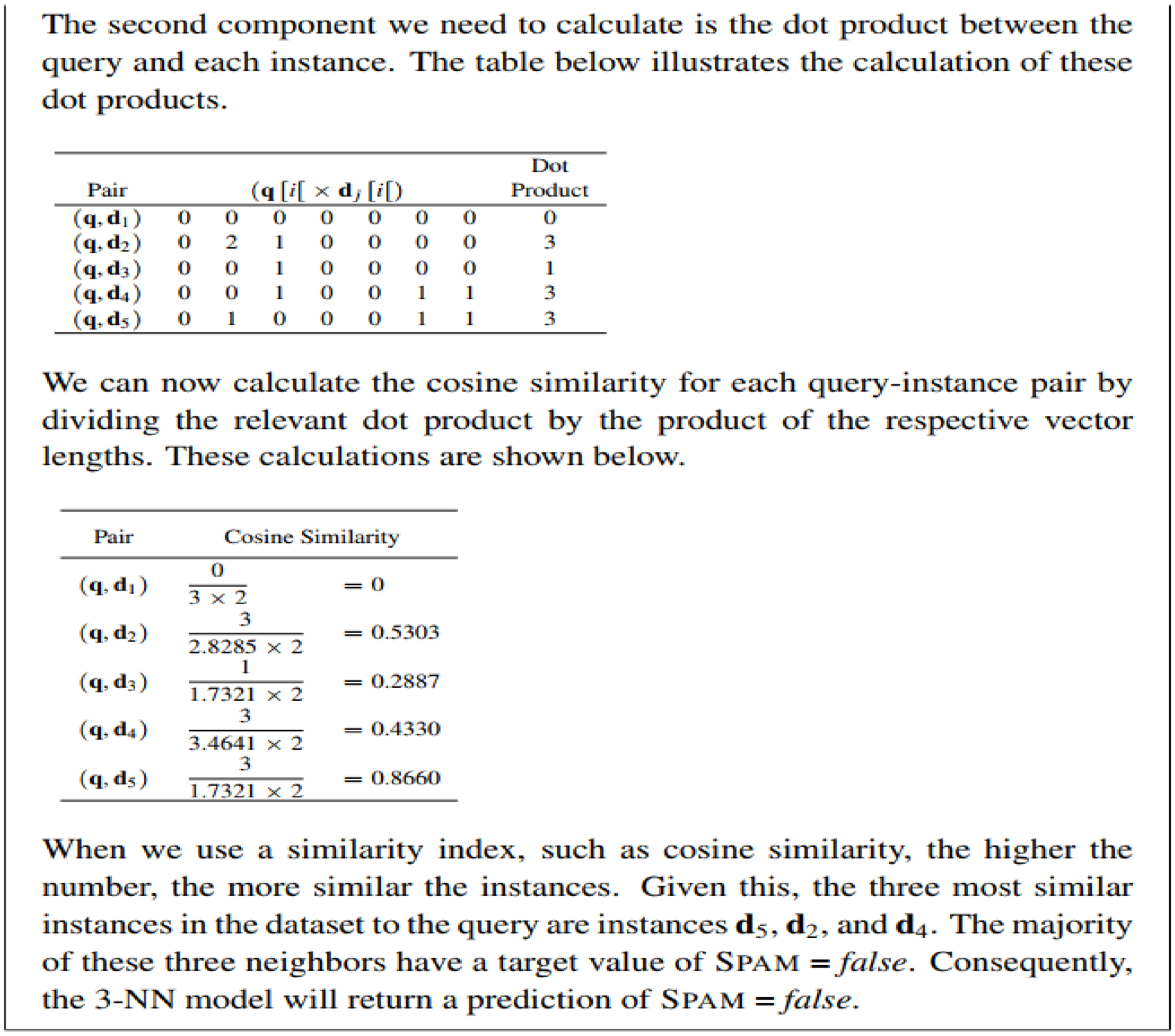
Based on these Manhattan distance calculations, the three nearest neighbors to the query are instances $\mathbf{d}_5$, $\mathbf{d}_4$, and $\mathbf{d}_3$. The majority of these three neighbors have a target value of SPAM = *false*. Consequently, the 3-NN model using Manhattan distance will return a prediction of SPAM = *false*.

(e) There are a lot of zero entries in the spam bag-of-words dataset. This is indicative of **sparse data** and is typical for text analytics. **Cosine similarity** is often a good choice when dealing with sparse non-binary data. What target label would a 3-NN model using cosine similarity return for the query?



In order to calculate the cosine similarity between the query and each instance in the dataset, we first need to calculate the vector length of each instance and the query. The table below illustrates the calculation of these vector lengths.

ID	$\mathbf{d}\,[i[^2$							Sum	Vector Length
1	9	0	0	0	0	0	0	9	3
2	1	4	1	1	1	0	0	8	2.8284
3	0	0	1	1	1	0	0	3	1.7321
4	0	0	1	0	9	1	1	12	3.4641
5	0	1	0	0	0	1	1	3	1.7321
Query	0	1	1	0	0	1	1	4	2



The second component we need to calculate is the dot product between the query and each instance. The table below illustrates the calculation of these dot products.

Pair	$(\mathbf{q}\,[i[\times \mathbf{d}_j\,[i[)$							Dot Product
$(\mathbf{q}, \mathbf{d}_1)$	0	0	0	0	0	0	0	0
$(\mathbf{q}, \mathbf{d}_2)$	0	2	1	0	0	0	0	3
$(\mathbf{q}, \mathbf{d}_3)$	0	0	1	0	0	0	0	1
$(\mathbf{q}, \mathbf{d}_4)$	0	0	1	0	0	1	1	3
$(\mathbf{q}, \mathbf{d}_5)$	0	1	0	0	0	1	1	3

We can now calculate the cosine similarity for each query-instance pair by dividing the relevant dot product by the product of the respective vector lengths. These calculations are shown below.

Pair	Cosine Similarity
$(\mathbf{q}, \mathbf{d}_1)$	$\frac{0}{3 \times 2} = 0$
$(\mathbf{q}, \mathbf{d}_2)$	$\frac{3}{2.8285 \times 2} = 0.5303$
$(\mathbf{q}, \mathbf{d}_3)$	$\frac{1}{1.7321 \times 2} = 0.2887$
$(\mathbf{q}, \mathbf{d}_4)$	$\frac{3}{3.4641 \times 2} = 0.4330$
$(\mathbf{q}, \mathbf{d}_5)$	$\frac{3}{1.7321 \times 2} = 0.8660$

When we use a similarity index, such as cosine similarity, the higher the number, the more similar the instances. Given this, the three most similar instances in the dataset to the query are instances $\mathbf{d}_5$, $\mathbf{d}_2$, and $\mathbf{d}_4$. The majority of these three neighbors have a target value of SPAM = *false*. Consequently, the 3-NN model will return a prediction of SPAM = *false*.

Exercise 2 (ex 4 book):

You have been given the job of building a recommender system for a large online shop that has a stock of over 100000 items. In this domain the behavior of customers is captured in terms of what items they have bought or not bought. For example, the following table lists the behavior of two customers in this domain for a subset of the items that at least one of the customers has bought.

ID	ITEM 107	ITEM 498	ITEM 7256	ITEM 28063	ITEM 75328
1	true	true	true	false	false
2	true	false	false	true	true

(a) The company has decided to use a similarity-based model to implement the recommender system. Which of the following three similarity indexes do you think the system should be based on?

$$\text{Russell-Rao(X,Y)} = \frac{CP(X,Y)}{P}$$

$$\text{Sokal-Michener(X,Y)} = \frac{CP(X,Y) + CA(X,Y)}{P}$$

$$\text{Jaccard(X,Y)} = \frac{CP(X,Y)}{CP(X,Y) + PA(X,Y) + AP(X,Y)}$$

In a domain where there are hundreds of thousands of items, co-absences aren't that meaningful. For example, you may be in a domain where there are so many items that most people haven't seen, listened to, bought, or visited that the majority of features will be co-absences. The technical term to describe a dataset where most of the features have zero values is **sparse data**. In these situations, you should use a metric that ignores co-absences. For a scenario such as this one, where the features are binary, the **Jaccard similarity index** is ideal as it ignores co-absences.

(b) What items will the system recommend to the following customer? Assume that the recommender system uses the similarity index you chose in the first part of this question and is trained on the sample dataset listed above. Also assume that the system generates recommendations for query customers by finding the customer most similar to them in the dataset and then recommending the items that this similar customer has bought but that the query customer has not bought.

ID	ITEM 107	ITEM 498	ITEM 7256	ITEM 28063	ITEM 75328
Query	true	false	true	false	false

Using a similarity metric, the higher the value returned by the metric, the more similar the two items are.

Assuming you chose the **Jaccard similarity index**, then the query customer is more similar to customer $\mathbf{d}_1$ than to customer $\mathbf{d}_2$:

- $Jaccard(\mathbf{q}, \mathbf{d}_1) = \frac{2}{2+1} = 0.6667$
- $Jaccard(\mathbf{q}, \mathbf{d}_2) = \frac{1}{4} = 0.25$

There is only 1 item that customer $\mathbf{d}_1$ has bought that the query customer has not bought, item 498. As a result, the system will recommend item 498 to the query customer.

It turns out that in this instance, no matter which of the three similarity metrics we use, customer $\mathbf{d}_1$ is more similar to the query customer than customer $\mathbf{d}_2$. The supporting calculations for Russell-Rao and Sokal-Michener are

- $\text{Russell-Rao}(\mathbf{q}, \mathbf{d}_1) = \frac{2}{5} = 0.4$
- $\text{Russell-Rao}(\mathbf{q}, \mathbf{d}_2) = \frac{1}{5} = 0.2$
- $\text{Sokal-Michener}(\mathbf{q}, \mathbf{d}_1) = \frac{4}{5} = 0.8$
- $\text{Sokal-Michener}(\mathbf{q}, \mathbf{d}_2) = \frac{2}{5} = 0.4$

So, the system will recommend item 498 regardless of which similarity metric is used.

Exercise 3 (ex 5 book):

You are working as an assistant biologist to Charles Darwin on the Beagle voyage. You are at the Galapagos Islands, and you have just discovered a new animal that has not yet been classified. Mr. Darwin has asked you to classify the animal using a nearest neighbor approach, and he has supplied you the following dataset of already classified animals.

ID	Births Live Young	Lays Eggs	Feeds Offspring Own Milk	Warm-Blooded	Cold-Blooded	Land and Water Based	Has Hair	Has Feathers	Class
1	true	false	true	true	false	false	true	false	mammal
2	false	true	false	false	true	true	false	false	amphibian
3	true	false	true	true	false	false	true	false	mammal
4	false	true	false	true	false	true	false	true	bird

The descriptive features of the mysterious newly discovered animal are as follows:

ID	Births Live Young	Lays Eggs	Feeds Offspring Own Milk	Warm-Blooded	Cold-Blooded	Land and Water Based	Has Hair	Has Feathers	Class
Query	false	true	false	false	false	true	false	false	?

(a) A good measure of distance between two instances with categorical features is the **overlap metric** (also known as the **hamming distance**), which simply counts the number of descriptive features that have *different* values. Using this measure of distance, compute the distances between the mystery animal and each of the animals in the animal dataset.

We can calculate the overlap metric between the query instance and each instance in the dataset by counting the number of feature values that are different.

ID	Class	Overlap Metric
1	mammal	6
2	amphibian	1
3	mammal	6
4	bird	2

(b) If you used a 1-NN model, what class would be assigned to the mystery animal?

The nearest neighbor to the mystery animal is $\mathbf{d}_2$. So the mystery animal would be classified as an *amphibian*.

(c) If you used a 4-NN model, what class would be assigned to the mystery animal? Would this be a good value for *k* for this dataset?

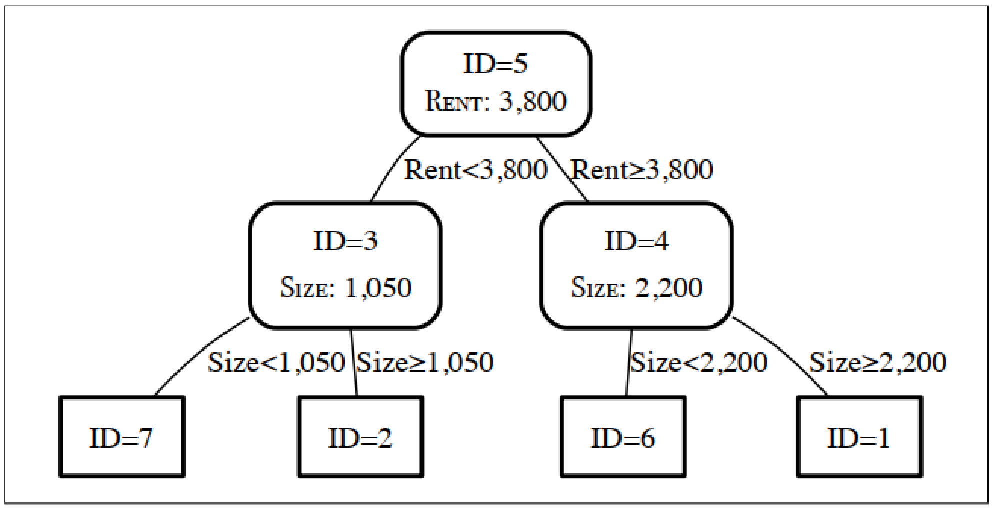
If you applied a 4-NN model to this dataset, the neighborhood defined around the query would include all the instances in the dataset irrespective of their distance from the query. As a result, any query would simply be assigned the majority class in the dataset, in this case *mammal*. So, for this dataset, a 4-NN model would massively underfit the dataset.

Exercise 4 (ex 6 book):

You have been asked by a San Francisco property investment company to create a predictive model that will generate house price estimates for properties they are considering purchasing as rental properties. The table below lists a sample of properties that have recently been sold for rental in the city. The descriptive features in this dataset are SIZE (the property size in square feet) and RENT (the estimated monthly rental value of the property in dollars). The target feature, PRICE, lists the prices that these properties were sold for in dollars.

ID	SIZE	RENT	PRICE
1	2,700	9,235	2,000,000
2	1,315	1,800	820,000
3	1,050	1,250	800,000
4	2,200	7,000	1,750,000
5	1,800	3,800	1,450,500
6	1,900	4,000	1,500,500
7	960	800	720,000

(a) Create a ***k-d* tree** for this dataset. Assume the following order over the features: RENT then SIZE.



(b) Using the *k-d* tree that you created in the first part of this question, find the nearest neighbor to the following query: SIZE = 1000, RENT = 2200.

The initial step in retrieving the nearest neighbor is to descend the tree to a leaf node. For this query, this descent will terminate at the node that stores instance $\mathbf{d}_7$. At this point, the *current best* variable is set to the instance stored at this node $\mathbf{d}_7$, and the *current best-distance* variable is set to the Euclidean distance between the query and $\mathbf{d}_7$:

$$current\ best = \mathbf{d}_7$$

$$current\ best\text{-}distance = Euclidean(\mathbf{q}, \mathbf{d}_7) = 1{,}400.5713$$

The retrieval algorithm then ascends the tree. The first node the algorithm will encounter is the node that stores instance $\mathbf{d}_3$. The Euclidean distance between the query and $\mathbf{d}_3$ is less than *current best-distance*. Consequently, *current best* and *current best-distance* are updated to reflect this:

$$current\ best = \mathbf{d}_3$$

$$current\ best\text{-}distance = Euclidean(\mathbf{q}, \mathbf{d}_3) = 951.3149$$

Because the difference between the splitting feature value at this node, SIZE = 1,050, and the query, SIZE = 1,000, is less than the *current best-distance*, the algorithm descends the other branch of the tree from this node. This descent will terminate at the node $\mathbf{d}_2$. The Euclidean distance between the query and $\mathbf{d}_2$ is less than the *current best-distance*. Consequently, *current best* and *current best-distance* are updated to reflect this:

$$current\ best = \mathbf{d}_2$$

$$current\ best\text{-}distance = Euclidean(\mathbf{q}, \mathbf{d}_2) = 509.1414$$

The algorithm will then ascend the tree; because it has already visited all the nodes on the path back to the root, it does not need to check for nodes closer than the *current best* until it gets back to the root. In this instance, the Euclidean distance between the query and the instance stored at the root node, $\mathbf{d}_3$, is greater than *current best-distance*, so neither *current best* nor *current best-distance* are updated when we reach the root node. Furthermore, because the difference between the splitting feature at the root, RENT = 3,800, and the query feature value, RENT = 2,200 is larger than the *current best-distance*, the algorithm can prune the other branch from the *k-d* tree and return $\mathbf{d}_2$ as the nearest neighbor, which would indicate that the property is worth approximately \$820,000.

Exercise 5 (ex 7 book):

A data analyst building a k-nearest neighbor model for a continuous prediction problem is considering appropriate values to use for k.

(a) Initially the analyst uses a simple average of the target variables for the *k* nearest neighbors in order to make a new prediction. After experimenting with values for *k* in the range 0-10, it occurs to the analyst that they might get very good results if they set *k* to the total number of instances in the training set. Do you think that the analyst is likely to get good results using this value for *k*?

If the analyst set *k* to the number of training examples all predictions would essentially be the average target value across the whole dataset. In other words, the model would return the average value for the target feature in the training data no matter what query was input into the model. This means that the model would be massively underfitting the data.

(b) If the analyst was using a distance weighted averaging function rather than a simple average for his or her predictions, would this have made the analyst's idea any more useful?

Yes, if distance weighted voting is used (particularly if a $\frac{1}{d^2}$ type distance weight is used) then examples that are far away from the query will have very little impact on the result and so the model will adjust the predictions it returns to the features in the query. It is worth highlighting that when distance weighted voting is used the value of *k* in *k*-NN classifiers is much less important.

Exercise 6 (ex 9 book):

A lecturer is about to leave for the airport to go on vacation when they find a script for a student they forgot to mark. They don't have time to manually grade the script before the flight, so they decide to use a k-nearest neighbor model to grade it instead. The model is designed to award a grade to a student on the basis of how similar they are to other students in the module in terms of their grades on other modules. The following table describes a set of students in terms of their grades out of 100 on two other modules (MODULE 1 and MODULE 2) and the GRADE they got in the lecturer's module: first-class honors, second-class honors, pass, or fail.

ID	MODULE 1	MODULE 2	GRADE
1	55	85	first
2	45	30	fail
3	40	20	fail
4	35	35	fail
5	55	75	pass
6	50	95	second

(a) Looking up the results on the other modules of the student whose script hasn't been corrected, the lecturer finds that the student got the following marks: MODULE 1=60, and MODULE 2=85. Assuming that the k-nearest neighbor model uses k=1 and Euclidean distance as its similarity metric, what GRADE would the model assign the student?

The Euclidean distance, rounded to 2 places of decimal, between the student and the other students in the sample are as follows:

ID	DISTANCE	GRADE
1	5.00	first
2	57.00	fail
3	68.00	fail
4	55.90	fail
5	11.18	pass
6	14.14	second

The student is most similar to student ID=1 and so the model will return a GRADE of *first*.

(b) Reviewing the spread of marks for the other two modules, the lecturer notices that there is a larger variance across students in the marks for Module 2 than there is for Module 1. So, the lecturer decides to update the k-nearest neighbor model to use the Mahalanobis distance instead of Euclidean distance as its similarity measure. Assuming that the inverse covariance matrix for the Module 1 and Module 2 results is

$$\sum^{-1} = \begin{bmatrix} 0.046 & -0.009 \\ -0.009 & 0.003 \end{bmatrix}$$

- what GRADE would the *k*-nearest neighbor model assign the student?

The calculations of the Mahalanobis distance between the student and each of the other students is as follows:

$$Mahalanobis(\mathbf{a}, \mathbf{b}) = \sqrt{[\mathbf{a}[1] - \mathbf{b}[1], \ldots, \mathbf{a}[m] - \mathbf{b}[m]] \times \sum^{-1} \times \begin{bmatrix} \mathbf{a}[1] - \mathbf{b}[1] \\ \vdots \\ \mathbf{a}[m] - \mathbf{b}[m] \end{bmatrix}}$$

$$Mahalanobis(\mathbf{q}, \mathbf{d}_1) = \sqrt{[5, 0] \times \begin{bmatrix} 0.046 & -0.009 \\ -0.009 & 0.003 \end{bmatrix} \times \begin{bmatrix} 5 \\ 1 \end{bmatrix}} = 1.072380529$$

$$Mahalanobis(\mathbf{q}, \mathbf{d}_2) = \sqrt{[15, 55] \times \begin{bmatrix} 0.046 & -0.009 \\ -0.009 & 0.003 \end{bmatrix} \times \begin{bmatrix} 15 \\ 55 \end{bmatrix}} = 2.138924964$$

$$Mahalanobis(\mathbf{q}, \mathbf{d}_3) = \sqrt{[20, 65] \times \begin{bmatrix} 0.046 & -0.009 \\ -0.009 & 0.003 \end{bmatrix} \times \begin{bmatrix} 20 \\ 65 \end{bmatrix}} = 2.770379035$$

$$Mahalanobis(\mathbf{q}, \mathbf{d}_4) = \sqrt{[25, 50] \times \begin{bmatrix} 0.046 & -0.009 \\ -0.009 & 0.003 \end{bmatrix} \times \begin{bmatrix} 25 \\ 50 \end{bmatrix}} = 3.708099244$$

$$Mahalanobis(\mathbf{q}, \mathbf{d}_5) = \sqrt{[5, 10] \times \begin{bmatrix} 0.046 & -0.009 \\ -0.009 & 0.003 \end{bmatrix} \times \begin{bmatrix} 5 \\ 10 \end{bmatrix}} = 0.374165739$$

$$Mahalanobis(\mathbf{q}, \mathbf{d}_6) = \sqrt{[10, -10] \times \begin{bmatrix} 0.046 & -0.009 \\ -0.009 & 0.003 \end{bmatrix} \times \begin{bmatrix} 10 \\ -10 \end{bmatrix}} = 2.588435821$$

When the model is updated to use Mahalanobis distance the student is most similar to **d**5 and so the model returns a GRADE of *pass*.