Math for Data Science

Introduction to Linear Algebra

Joanna Bieri DATA100

Important Information

- Email: joanna_bieri@redlands.edu
- Office Hours take place in Duke 209 unless otherwise noted – Office Hours Schedule

Today's Goals:

- What is Linear Algebra
- Vectors
- Addition, Magnitude, Scalar Multiplication, Subtraction
- K Nearest Neighbors - Machine Learning Classification

Linear Algebra

Linear algebra gives us a way to study linear systems. In my opinion Linear Algebra is one of the most important mathematical topics you can study to truly understand modern computation, data science, and machine learning. It is a fundamental tool behind the "black box" of many algorithms used in data science.

There are some scary terms used when talking about linear algebra: Vector, Matrix, Tensor, Eigenvalue, Inverse, Span, Basis, etc. I hope that by the end of this course these ideas feel more comfortable to you.

What is a Vector.

In the simplest words a vector is just an arrow in space with a specific direction and length. In two dimensions it is given by two numbers

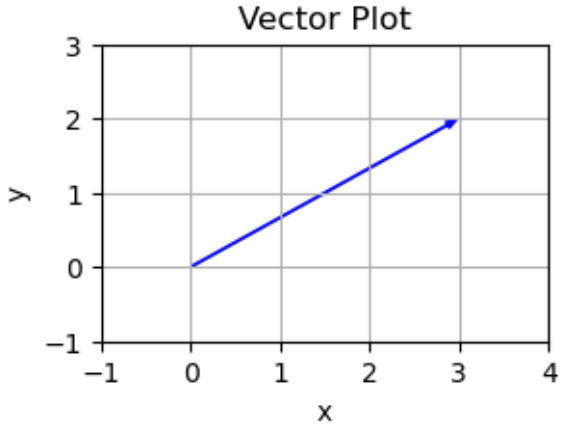
$$\vec{v} = \begin{bmatrix} a \\ b \end{bmatrix}$$

a and b are just numbers that tell me how far to go in each direction. Vectors exist independently of a coordinate system, although we imagine them with their tail at $(0, 0)$ and their point at (a, b)

Example:

$$\vec{v} = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

What is a Vector.



What is a Vector.

Here we can see that from $(0, 0)$ we go three steps in the x direction and two steps in the y direction.

What is a Vector.

There are other ways to input a vector in Python:

- Use Numpy if you want to do numerical calculations

```
v = np.array([3,2])
print(v)
```

```
[3 2]
```

- Use Sympy if you want symbolic results.

```
v = sp.Matrix([3,2])
v
```

$$\begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

You Try

Here are three more examples of vectors. First draw the vector by hand, then compare what what you get by using the code below.

1. $$\vec{v} = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

2. $$\vec{v} = \begin{bmatrix} -3 \\ 2 \end{bmatrix}$$

3. $$\vec{v} = \begin{bmatrix} -4 \\ -4 \end{bmatrix}$$

Why are Vectors Useful?

A vector can represent many different things:

- Navigation systems (GPS).
- Robotics (path planning).
- Physics (forces, velocities).
- Game development.
- Classification and Distances.
- Abstract systems.

Why are Vectors Useful?

Say you have information about the square footage of a home and it's sales price. We can put this information into a vector so that we can more easily do operations on it.

- Square footage $= 1500$
- Price $= 450000$
- $$\vec{v} = \begin{bmatrix} 1500 \\ 450000 \end{bmatrix}$$

In fact the inputs into machine learning models are represented as vectors.

Higher Dimensions

The beautiful thing about vectors is that you can develop intuition about them in just 2-dimensions but the ideas work no matter how many dimensions you have. We can still visualize three dimensions as an arrow in three dimensional space:

$$\vec{v} = \begin{bmatrix} 4 \\ 1 \\ 2 \end{bmatrix}$$

This vector starts at $(0, 0)$ then you go 4 steps in the x direction, 1 step in the y direction and 2 steps in the z direction.

Higher Dimensions

In many machine learning examples you will have multiple input features - columns in your data that you think help you predict something important. You will often put these features into a vector for each observation.

Adding Vectors

Adding vectors allows us to combine the movements of two vectors into a single instruction. Let look at an example:

$$\vec{v} = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

$$\vec{w} = \begin{bmatrix} 2 \\ -1 \end{bmatrix}$$

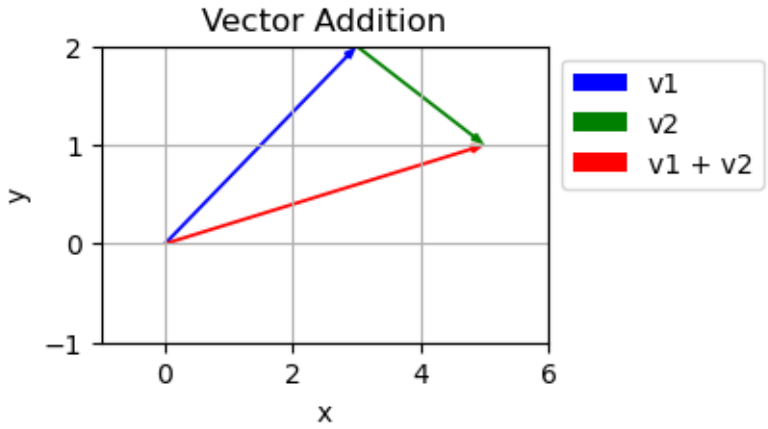
Adding Vectors

When you add vectors you add their components

$$\vec{v} + \vec{w} = \begin{bmatrix} 3 \\ 2 \end{bmatrix} + \begin{bmatrix} 2 \\ -1 \end{bmatrix} = \begin{bmatrix} 3+2 \\ 2+(-1) \end{bmatrix} = \begin{bmatrix} 5 \\ 1 \end{bmatrix}$$

Adding Vectors

Here is a picture of this




```
[5 1]
```

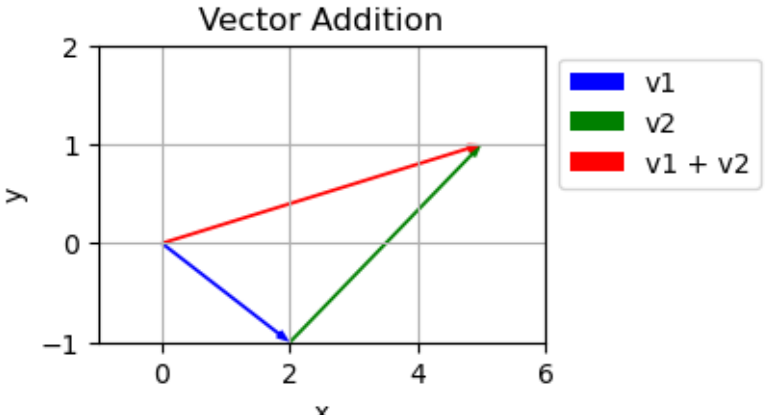
Adding Vectors

We can think about what this addition means by the fact that we can get to the final location either by following the blue and then the green vector OR by just following the red vector. The addition is combining these steps into a single instruction.

Adding Vectors

Also, we could have done this in any order! We get the same result

$$\vec{v} + \vec{w} = \vec{w} + \vec{v}$$



Adding Vectors

The code for adding vectors in numpy is

```
v = np.array([3,2])
w = np.array([2,-1])

v+w
```

```
array([5, 1])
```

You Try

Consider the vectors:

$$\vec{v} = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

$$\vec{u} = \begin{bmatrix} -3 \\ 2 \end{bmatrix}$$

$$\vec{w} = \begin{bmatrix} -4 \\ -4 \end{bmatrix}$$

Find the following sums. First do the calculation and drawing by hand then use the code below to check your answer.

1. $\vec{v} + \vec{u}$
2. $\vec{u} + \vec{v}$
3. $\vec{v} + \vec{u} + \vec{w}$

Magnitude and Direction of a Vector

The magnitude of a vector is a measure of it's length and it's direction is a measure of the angle between it and the x-axis. To get these values we can use trigonometry!

Magnitude and Direction of a Vector

Magnitude - use Pythagorean Theorem

We know the bottom side is 3 units and the right side is 2 units so the hypotenuse - or the length is

$$||\vec{v}|| = \sqrt{3^2 + 2^2} = \sqrt{9 + 4} = \sqrt{13}$$

We can also use the python code

```
np.linalg.norm()
```

Magnitude and Direction of a Vector

Angle - use Tangent

We know that in a right triangle the tangent of the angle is the opposite side over the adjacent side:

$$\tan(\theta) = opp/adj$$

so for our vector

$$\tan(\theta) = \frac{2}{3}$$

and solving for the angle

$$\theta = \arctan\left(\frac{2}{3}\right) = \tan^{-1}\left(\frac{2}{3}\right)$$

You Try

Consider the vectors:

$$\vec{v} = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

$$\vec{u} = \begin{bmatrix} -3 \\ 2 \end{bmatrix}$$

$$\vec{w} = \begin{bmatrix} -4 \\ -4 \end{bmatrix}$$

1. Find the magnitude of each vector write out the formula by hand. You can use python to check your work, see the code below, but you should draw the vector and write out the Pythagorean formula.

Scaling Vectors

Now that we can find the magnitude of a vector, we might want to shrink or extend the vector, without changing the direction. This is called scaling - or **scalar multiplication**

What is a **scalar**, this is just a number like we are use to. It has only a single value.

Scaling Vectors

Example

$$\vec{v} = \begin{bmatrix} 3 \\ 2 \end{bmatrix}$$

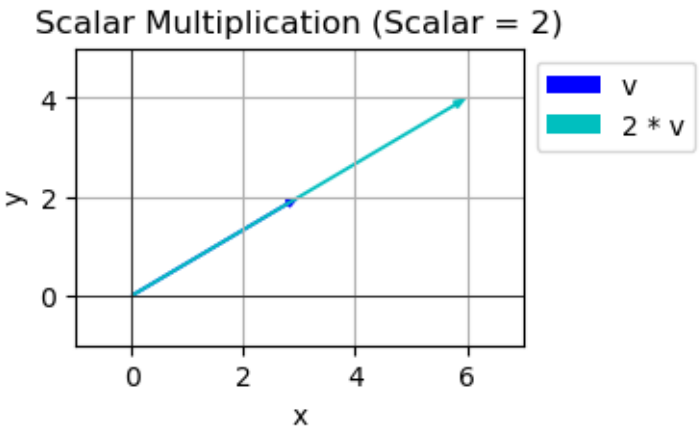
then

$$2\vec{v} = 2 \begin{bmatrix} 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 2 * 3 \\ 2 * 2 \end{bmatrix} = \begin{bmatrix} 6 \\ 4 \end{bmatrix}$$

In this case we multiply each component of the vector by the scalar.

Scaling Vectors

Here is a plot of this




```
array([6, 4])
```

You Try

Consider the vectors:

$$\vec{v} = \begin{bmatrix} 3 \\ 4 \end{bmatrix}$$

$$\vec{u} = \begin{bmatrix} -3 \\ 2 \end{bmatrix}$$

$$\vec{w} = \begin{bmatrix} -4 \\ -4 \end{bmatrix}$$

1. Find $4\vec{v}$
2. Find $-1\vec{w}$
3. Find $\frac{1}{2}\vec{u}$
4. Do the vectors change direction in any of these cases?

For each of these do the calculation by hand, draw the picture, and then check your results using the code below.

K-nearest Neighbors Classifier

Here is a real world example where vectors are used in Data Science and Machine Learning!

The K-Nearest Neighbors (K-NN) algorithm is a popular Machine Learning algorithm used mostly for solving classification problems. It uses a set of discrete steps to decide how to classify a test point based on existing data.

K-nearest Neighbors Classifier

Given a new data point

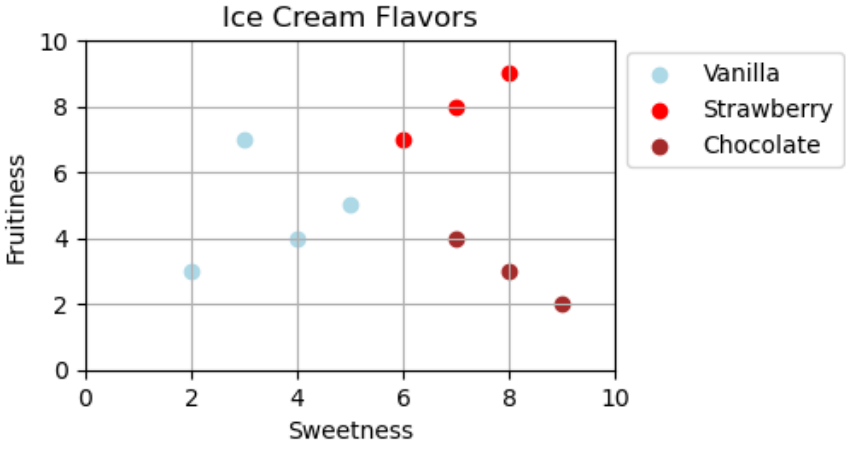
1. Choose how many neighbors you want to use (K)
2. Calculate the distance between the new data point and all other points in the data set. (vector magnitude).
3. Find the K nearest neighbors to the new data point based in the distances.
4. Assign the new data entry to the class that is in the majority of nearest neighbors. If there is a tie - come up with tie breaker rule! Often we increase or decrease K.

Data

Let's say you gather the following data about a group of people

	Sweetness	Fruitiness	Flavor
0	2	3	Vanilla
1	8	9	Strawberry
2	9	2	Chocolate
3	7	8	Strawberry
4	3	7	Vanilla
5	5	5	Vanilla
6	6	7	Strawberry
7	8	3	Chocolate
8	4	4	Vanilla
9	7	4	Chocolate

K-nearest Neighbors Classifier



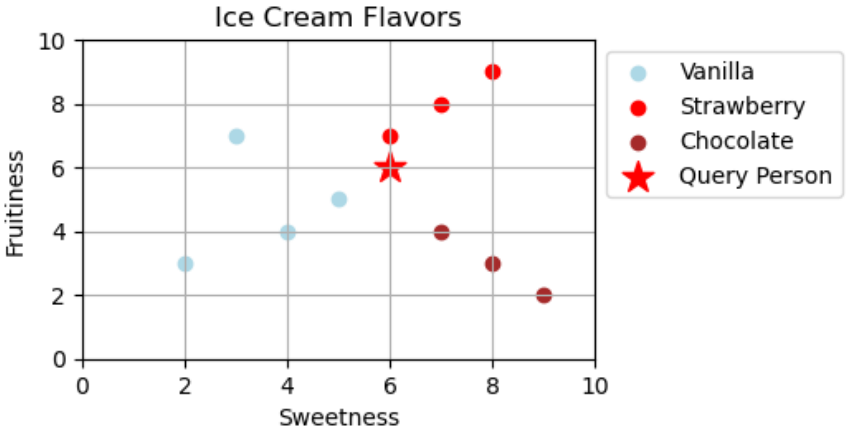
K-nearest Neighbors Classifier

Now imagine that you have a new person that who has the data

```
query_vector = np.array([6, 6])
```

$$\vec{q} = \begin{bmatrix} Sweetness \\ Fruitiness \end{bmatrix} = \begin{bmatrix} 6 \\ 6 \end{bmatrix}$$

K-nearest Neighbors Classifier



K-nearest Neighbors Classifier

1. We will choose $K = 4$ to start, but this is just a choice!
2. Calculate the distance between the new data point and all other points in the data set. (vector magnitude).

Now in this case the distance between two points is the magnitude of the vector between the two arrow points. To see how far apart my query vector and another data vector are we can subtract them and then find the magnitude

$$\vec{q} = \begin{bmatrix} 6 \\ 6 \end{bmatrix}$$

$$\vec{d}_1 = \begin{bmatrix} 2 \\ 3 \end{bmatrix}$$

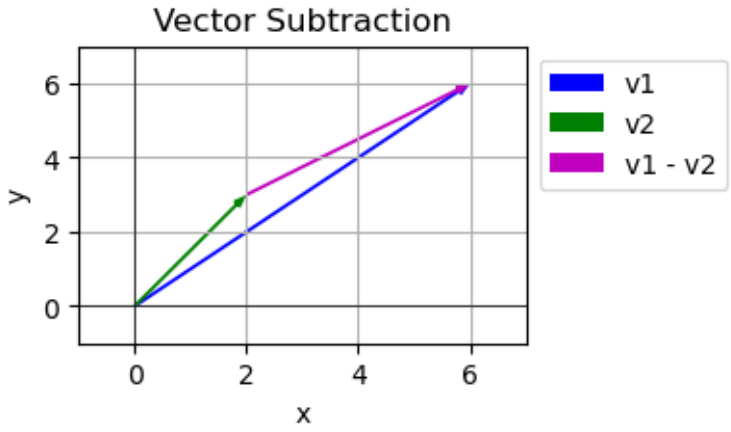
K-nearest Neighbors Classifier

$$\vec{q} - \vec{d_1} = \vec{q} + (-\vec{d_1}) = \begin{bmatrix} 6 \\ 6 \end{bmatrix} + \begin{bmatrix} -2 \\ -3 \end{bmatrix} = \begin{bmatrix} 4 \\ 3 \end{bmatrix}$$

then find the magnitude

K-nearest Neighbors Classifier

Here is a picture of this subtraction



5.0

K-nearest Neighbors Classifier

But we would need to do this for every point!

```
[(5.0, 'Vanilla'),
 (3.605551275463989, 'Strawberry'),
 (5.0, 'Chocolate'),
 (2.23606797749979, 'Strawberry'),
 (3.1622776601683795, 'Vanilla'),
 (1.4142135623730951, 'Vanilla'),
 (1.0, 'Strawberry'),
 (3.605551275463989, 'Chocolate'),
 (2.8284271247461903, 'Vanilla'),
 (2.23606797749979, 'Chocolate')]
```

K-nearest Neighbors Classifier

3. Get the K nearest points

```
[(1.0, 'Strawberry'),
 (1.4142135623730951, 'Vanilla'),
 (2.23606797749979, 'Chocolate'),
 (2.23606797749979, 'Strawberry')]
```

It looks like we should suggest that this person tries Strawberry Ice Cream!

KNN in Python!

Python Sklearn has a KNN function!

```
from sklearn.neighbors import KNeighborsClassifier

data = DF[['Sweetness','Fruitiness']].values
labels = list(DF['Flavor'])

knn_4 = KNeighborsClassifier(n_neighbors=4) # Update the numbe
knn_4.fit(data,labels)
```

```
KNeighborsClassifier(n_neighbors=4)
```

KNN in Python!

Use the model to make a prediction

```
query_vector = np.array([6,6])

query = query_vector.reshape(1, -1)
suggestion = knn_4.predict(query)
print(suggestion)
```

```
['Strawberry']
```

You can see that we got the prediction we expected! You can play around with this code and see what happens for different queries.

You Try

Consider the video game data below. Imagine you survey 10 video game enthusiast and ask them to say what kind of game they prefer and rank the amount of action and story content they like to have in a game. Now we are going to use this data to build a system that will recommend a game type to a new player based on their scores for action and story.

New player (query) vector:

$$\vec{q} = \begin{bmatrix} Action \\ Story \end{bmatrix} = \begin{bmatrix} 6 \\ 4 \end{bmatrix}$$

You Try

Please do the following:

1. Choose one of the data vectors and write it down in vector form.
2. Find the vector that is difference between to two $\vec{q} - \vec{data}$
3. Plot these vectors using the code below.
4. Find the distance between the two - magnitude of the difference.
5. Using Sklearn KNeighborsClassifier, build a KNN with K=3.
6. Predict the type of game your new player would want to play.

You Try

	Amt Action	Amt Story	Type
0	3	9	RPG
1	9	4	Action
2	5	8	RPG
3	10	2	Action
4	6	7	RPG
5	2	6	RPG
6	9	3	Action
7	4	7	RPG
8	8	5	Action
9	7	3	Action

You Try