DS 3000 HW 3

Your Name: Diego Cicotoste

Due: Tuesday Oct 29 @ 11:59 PM EST

Extra Credit Deadline: Sunday Oct 27 @ 11:59 PM EST

Submission Instructions

Submit this `ipynb` file to Gradescope (this can also be done via the assignment on Canvas). To ensure that your submitted `ipynb` file represents your latest code, make sure to give a fresh `Kernel > Restart & Run All` just before uploading the `ipynb` file to gradescope. **In addition:**

- Make sure your name is entered above
- Make sure you comment your code effectively
- If problems are difficult for the TAs/Profs to grade, you will lose points

Tips for success

- Start early
- Make use of Piazza (also accessible through Canvas)
- Make use of Office Hours
- Remember to use cells and headings to make the notebook easy to read (if a grader cannot find the answer to a problem, you will receive no points for it)
- Under no circumstances may one student view or share their ungraded homework or quiz with another student (see also), though you are welcome to **talk about** (*not* show each other your answers to) the problems.

Part 1: Summarizing and Visualizing Data

For this part, you will use the `players_fifa23.csv` from Canvas to investigate the ratings for soccer players in the FIFA 23 video game. Make sure the `.csv` is in the same directory as this notebook file.

Note: You do not need to know anything about soccer or video games to complete this problem, only perhaps that a higher `Overall` rating is considered a good thing.

Part 1.1: Plotting Data (15 points)

Create a plotly scatter plot which shows the mean `Overall` rating for soccer players (rows) of a given `Nationality` for a particular `Age`. Focuse on three countries (`England`, `Germany`, `Spain`). In other words, your plot's x-axis should be `Age`, the y-axis should be `Overall`, and there should be three different colored points at each `Age`, one for each `Nationality`.

Export your graph as an html file `age_ratings_nationality.html`. You do not have to submit it with this homework, but the code should show that you did this.

Hints:

- There may be multiple ways/approaches to accomplish this task.
- One approach: you may use `groupby()` and boolean indexing to build these values in a loop which runs per each `Nationality`.
- `px.scatter()` will only graph data from columns (not the index). Some approaches may need to graph data from the index. You can use df.reset_index() to make your index a new column as shown in this example
- In some approaches you may need to pass multiple rows to df.append() if need be as shown in this example
- In some approaches you may need to go from "wide" data to "long" data by using df.melt() as discussed here
- The first few code cells below get you started with looking at the data set.

In [6]:
```
# use pandas to read in the data
import pandas as pd
import plotly.express as px

df_fifa = pd.read_csv('players_fifa23.csv', index_col = 'ID')
df_fifa.head()
```

Out[6]:

ID	Name	FullName	Age	Height	Weight	
165153	K. Benzema	Karim Benzema	34	185	81	https://cdn.sofifa.net/players/1(
158023	L. Messi	Lionel Messi	35	169	67	https://cdn.sofifa.net/players/15
231747	K. Mbappé	Kylian Mbappé	23	182	73	https://cdn.sofifa.net/players/2
192985	K. De Bruyne	Kevin De Bruyne	31	181	70	https://cdn.sofifa.net/players/19
188545	R. Lewandowski	Robert Lewandowski	33	185	81	https://cdn.sofifa.net/players/18

5 rows × 89 columns

In [7]:
```
df_fifa.Nationality.value_counts()
```

Out[7]:
```
Nationality
England        1652
Germany        1209
Spain          1054
France          936
Argentina       930
               ... 
Saint Lucia       1
Kazakhstan        1
Vietnam           1
Niger             1
Singapore         1
Name: count, Length: 161, dtype: int64
```

In [8]:
```
df_fifa.shape
```

Out[8]:
```
(18360, 89)
```

In [9]:
```
df_fifa['Age'].unique()
```

Out[9]:
```
array([34, 35, 23, 31, 33, 30, 36, 37, 28, 29, 27, 25, 32, 21, 26, 24, 19,
       22, 40, 20, 39, 38, 44, 17, 41, 18, 42, 43, 16])
```

In [10]:
```
df_filtered = df_fifa[df_fifa['Nationality'].isin(['England', 'Germany', 'Sp

df_grouped = df_filtered.groupby(['Nationality', 'Age'])['Overall'].mean().r
```

```
fig = px.scatter(df_grouped, x='Age', y='Overall', color='Nationality',
                 labels={'Overall': 'Mean Overall Rating', 'Age': 'Player Ag
```

In [11]:
```
fig.show()
```

Part 1.2: Numerical Summaries (10 points)

1. Calculate the sample mean and median of `Overall` for the entire data set. In a markdown cell, discuss what their relative values imply about the distribution of `Overall`, and then use the plot from 1.1 and these values to discuss whether you think English, German, and Spanish players are generally better rated than other country's players, and at what age do they become average players?
2. Calculate the `.group_by()` function to calculate the means and standard deviations of `Overall` for the three Nationalities in Part 1.1 (you will want to use the original data frame or a slightly modified version of it (the `.isin` function from pandas may help), **NOT** the data frame you used for the plot). What do these values tell you about the differences between English, German, and Spanish players?
3. Create a subset of the original data frame that includes only `Age`, `Height`, `Weight`, and `Overall`. Calculate the correlation matrix for these four features and discuss what the relationships seem to be and whether those relationships make sense to you.

In [12]:
```
mean = df_fifa['Overall'].mean()
median = df_fifa['Overall'].median()
print(f"mean {mean}")
print(f"median {median}")

df = df_fifa[df_fifa['Nationality'].isin(['England', 'Germany', 'Spain'])]
grouped_stats = df.groupby('Nationality')['Overall'].agg(['mean', 'median'])
print(f"grouped stats \n{grouped_stats}")

df_subset = df_fifa[['Age', 'Height', 'Weight', 'Overall']]
corr_matrix = df_subset.corr()
print(f"\nmatrix \n{corr_matrix}")
```

```
mean 65.83267973856209
median 66.0
grouped stats
  Nationality       mean  median
0     England  63.865617    63.0
1     Germany  65.635236    65.0
2       Spain  69.163188    68.0

matrix
              Age    Height    Weight   Overall
Age      1.000000  0.064194  0.212254  0.442932
Height   0.064194  1.000000  0.756610  0.031366
Weight   0.212254  0.756610  1.000000  0.124180
Overall  0.442932  0.031366  0.124180  1.000000
```

1. The median and the mean suggest that from the database and all the nationality, the mean and medium are in their mid 60s
2. By getting the specifict nationality we can break down the the mean and the median, by showing us england is the youngest, and spain is the oldest having germany in the middle, they all have around 2-3 in difference

Part 2: Vector Geometry Practice (20 points)

Use the vectors to below to compute the following quantities. You must show all math work/steps (no matter how trivial) to receive full credit. You may either use LaTeX typesetting within a Markdown cell, or do it by hand with pen and paper and embed the image in this .ipynb file, or submit a separate pdf file with your handwritten work. Round all decimals to three places.

After calculating the quantities by hand, use numpy in cells below to verify your answers.

$$a = \begin{bmatrix} 2 \\ -1 \\ 3 \end{bmatrix}$$

$$b = \begin{bmatrix} -4 \\ -2 \\ 0 \end{bmatrix}$$

$$c = \begin{bmatrix} 3 \\ 3 \\ -3 \end{bmatrix}$$

1. Compute $||b + c||$
2. Compute $2a + b$
3. Compute $c \cdot a$
4. Compute $||\frac{a}{2} - c||$
5. Compute $b \cdot (a + c)$

1.

$$b+c = \begin{bmatrix} -4 \\ -2 \\ 0 \end{bmatrix} + \begin{bmatrix} 3 \\ 3 \\ -3 \end{bmatrix}$$

$$= \begin{bmatrix} -1 \\ 1 \\ -3 \end{bmatrix}$$

$$= \sqrt{-1^2 + 1^2 + 3^2}$$

$$= \boxed{\sqrt{11}}$$

2.

$2a+b$

$$2\begin{bmatrix} 2 \\ -1 \\ 3 \end{bmatrix} + \begin{bmatrix} -4 \\ -2 \\ 0 \end{bmatrix}$$

$$\begin{bmatrix} 4 \\ -2 \\ 6 \end{bmatrix} + \begin{bmatrix} -4 \\ -2 \\ 0 \end{bmatrix} = \boxed{\begin{bmatrix} 0 \\ -4 \\ 6 \end{bmatrix}}$$

3 c, a

$$\begin{bmatrix} 3 \\ 3 \\ -3 \end{bmatrix} \begin{bmatrix} 2 \\ -1 \\ 3 \end{bmatrix} = \begin{bmatrix} 6 \\ -3 \\ -9 \end{bmatrix}$$

$$6 + -3 + -9 = \boxed{-6}$$

4. $\left\| \frac{a}{2} - c \right\|$

$$\frac{1}{2}\begin{bmatrix}2\\-1\\3\end{bmatrix} - \begin{bmatrix}3\\3\\-3\end{bmatrix}$$

$$\begin{bmatrix}1\\-0.5\\1.5\end{bmatrix} - \begin{bmatrix}3\\3\\-3\end{bmatrix} = \begin{bmatrix}-2\\-3.5\\4.5\end{bmatrix}$$

$$= \sqrt{(-2)^2 + (3.5)^2 + (4.5)^2}$$

$$= \boxed{\sqrt{36.5}}$$

5.

$$\begin{bmatrix} -4 \\ -2 \\ 0 \end{bmatrix} \cdot \left(\begin{bmatrix} 2 \\ -1 \\ 3 \end{bmatrix} + \begin{bmatrix} 3 \\ 3 \\ -3 \end{bmatrix} \right)$$

$$\begin{bmatrix} -4 \\ -2 \\ 0 \end{bmatrix} \left(\begin{bmatrix} 5 \\ 2 \\ 0 \end{bmatrix} \right)$$

$$\begin{bmatrix} -20 \\ -4 \\ 0 \end{bmatrix} = \boxed{-24}$$

In [13]:
```
import numpy as np

a = np.array([2, -1, 3])
b = np.array([-4, -2, 0])
c = np.array([3, 3, -3])
```

In [14]:
```
b_plus_c = b + c
magnitude = np.linalg.norm(b_plus_c)
print("||b + c|| =", magnitude)

times_add = 2 * a + b
print("2a + b =", times_add)

dot= np.dot(c, a)
print("c x a =", dot)

a_div_2_minus_c = (a / 2) - c
magnitude = np.linalg.norm(a_div_2_minus_c)
print("||a/2 - c|| =", magnitude)

# 5. Compute b · (a + c)
dot = np.dot(b, a + c)
print("b · (a + c) =", dot)
```

```
||b + c|| = 3.3166247903554
2a + b = [ 0 -4  6]
c x a = -6
||a/2 - c|| = 6.041522986797286
b · (a + c) = -24
```

Part 3: Computation by Hand

For each of the sub-parts below, you must show all math work/steps (no matter how trivial) to receive full credit. You may either use LaTeX typesetting within a Markdown cell, or do it by hand with pen and paper and embed the image in this .ipynb file, or submit a separate pdf file with your handwritten work. Round all decimals to three places.

Part 3.1: Matrix Multiplication (10 points)

Using the below matrices, perform the following operations by hand, **then** perform the same operations in your notebook using `numpy`. If an operation cannot be done, still write the code but then comment it out before running and submitting your final .ipynb file.

$$A = \begin{bmatrix} -3 & 8 \\ 0 & 5 \end{bmatrix}$$

$$B = \begin{bmatrix} 2 & -7 \\ 6 & -1 \\ -9 & 4 \end{bmatrix}$$

$$C = \begin{bmatrix} -6 & 0 & 5 \\ 1 & 3 & -2 \\ 7 & -5 & -8 \\ 4 & 9 & -10 \end{bmatrix}$$

$$D = \begin{bmatrix} -4 & 0 & 3 \\ 8 & -2 & 5 \\ 6 & -3 & 1 \end{bmatrix}$$

$$e = \begin{bmatrix} 7 \\ -8 \\ 10 \\ -1 \end{bmatrix}$$

- AB^T
- CD
- DB
- Ce
- e^TC

3.1

$$B = \begin{bmatrix} 2 & -7 \\ 6 & -1 \\ -9 & 4 \end{bmatrix} \quad B^T = \begin{bmatrix} 2 & 6 & -9 \\ -7 & -1 & 4 \end{bmatrix}$$

$A \cdot B^T =$

$$\begin{bmatrix} -3 & 8 \\ 0 & 5 \end{bmatrix} \cdot \begin{bmatrix} 2 & 6 & -9 \\ -7 & -1 & 9 \end{bmatrix}$$

$(-3 \cdot 2) + (8 \cdot -7) = -62$
$(-3 \cdot 6) + (8 \cdot -1) = -26$
$(-3 \cdot -9) + (8 \cdot 4) = 59$
$(0 \cdot 2) + (5 \cdot -7) = -35$
$(0 \cdot 6) + (5 \cdot -1) = -5$
$(0 \cdot -9) + (5 \cdot 4) = 20$

$$\begin{bmatrix} -62 & -26 & 59 \\ -35 & -5 & 20 \end{bmatrix}$$

3.2

C

-35 -5 20

3.2

CD

$$C = \begin{bmatrix} -6 & 0 & 5 \\ 1 & 7 & -2 \\ 7 & -5 & -8 \\ 4 & 8 & -10 \end{bmatrix} \quad D = \begin{bmatrix} -4 & 0 & 3 \\ 8 & -2 & 5 \\ 6 & -3 & 1 \end{bmatrix}$$

row - collum multiply

$$\begin{bmatrix} 54 & -15 & -13 \\ 8 & 0 & 16 \\ -116 & 34 & -12 \\ -4 & 12 & 47 \end{bmatrix}$$

3.3

DB

3.3

DB

$$D = \begin{bmatrix} -4 & 0 & 3 \\ 8 & -2 & 5 \\ 6 & -3 & 1 \end{bmatrix} \quad B = \begin{bmatrix} 2 & -7 \\ 6 & -1 \\ -9 & 4 \end{bmatrix}$$

$(-4 \cdot 2) + (0 \cdot 6) + (3 \cdot -9) = -35$

$(-4 \cdot -7) + (0 \cdot -1) + (3 \cdot 4) = 40$

$(8 \cdot 2) + (-2 \cdot 6) + (5 \cdot -9) = -41$

$(8 \cdot -7) + (-2 \cdot -1) + (5 \cdot 4) = -34$

$(6 \cdot 2) + (-3 \cdot 6) + (1 \cdot -9) = -15$

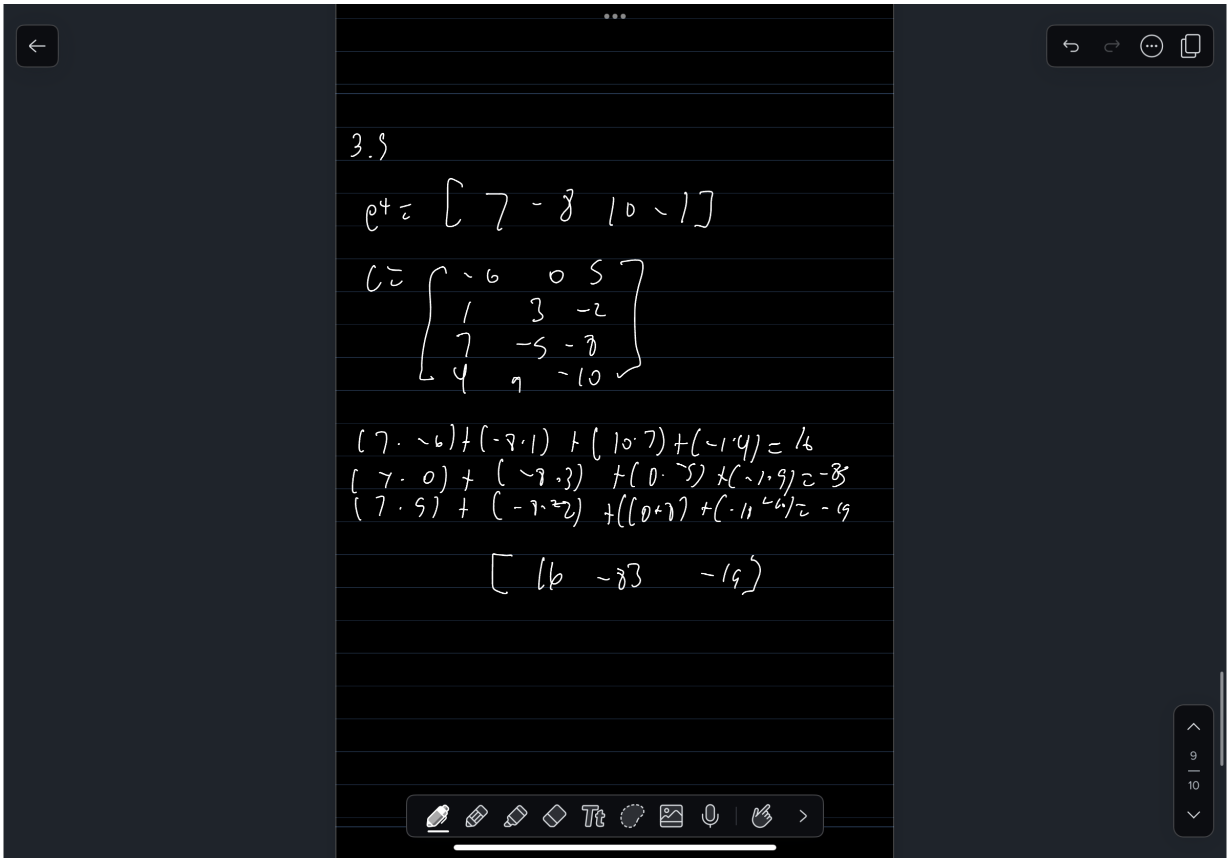
$(6 \cdot -7) + (-3 \cdot -1) + (1 \cdot 4) = -35$

$$\begin{bmatrix} -35 & 40 \\ -41 & -34 \\ -15 & -35 \end{bmatrix}$$

8

10

4. is not possible the number of rows need to match the number of columns



In [15]:

```python
A = np.array([[-3, 8], [0, 5]])
B = np.array([[2, -7], [6, -1], [-9, 4]])
C = np.array([[-6, 0, 5], [1, 3, -2], [7, -5, -8], [4, 9, -10]])
D = np.array([[-4, 0, 3], [8, -2, 5], [6, -3, 1]])
e = np.array([[7], [-8], [10], [-1]])

ABT = np.dot(A, B.T)
CD = np.dot(C, D)
DB = np.dot(D, B)
# Ce = np.dot(C, e)
eTC = np.dot(e.T, C)

print(f"ABT: {ABT} \n")
print(f"CD: {CD} \n")
print(f"Ce NOT POSSIBLE \n")
print(f"DB: {DB} \n")
print(f"eTC: {eTC} \n")
```

```
ABT: [[-62 -26  59]
 [-35  -5  20]]

CD: [[  54  -15  -13]
 [   8    0   16]
 [-116   34  -12]
 [  -4   12   47]]

Ce NOT POSSIBLE

DB: [[-35  40]
 [-41 -34]
 [-15 -35]]

eTC: [[ 16 -83 -19]]
```

Part 3.2: Spans, Linear In/Dependence, Orthogonality (10 points)

Based on the following three vectors, answer the ensuing questions, making sure to write out all supporting work with by hand or in a markdown cell. You may use `numpy` to help you check some of your answers, if you wish, but all work must be done by hand and provided for full credit.

$$a = \begin{bmatrix} 8 \\ -2 \end{bmatrix}$$

$$b = \begin{bmatrix} 1 \\ 4 \end{bmatrix}$$

$$c = \begin{bmatrix} -3 \\ .75 \end{bmatrix}$$

- What is the span of a and b?
- What is the span of a and c?
- Are the vectors a and b linearly independent or dependent?
- Is the set of all three vectors linearly independent or dependent?
- Which vectors are orthogonal to each other?

3.2.1

$a = \begin{bmatrix} 8 \\ -2 \end{bmatrix} \quad b = \begin{bmatrix} 1 \\ 4 \end{bmatrix}$

$$\alpha \begin{bmatrix} 8 \\ -2 \end{bmatrix} + \beta \begin{bmatrix} 1 \\ 4 \end{bmatrix} = \begin{array}{c} \begin{bmatrix} 8\alpha + \beta \\ -2\alpha + 4\beta \end{bmatrix} \end{array} \begin{array}{c} x \\ y \end{array}$$

the span is the entire 2D plane

3.2.2

$a = \begin{bmatrix} 8 \\ -2 \end{bmatrix}$ $c = \begin{bmatrix} -3 \\ .75 \end{bmatrix}$

$\alpha \begin{bmatrix} 8 \\ -2 \end{bmatrix} + \beta \begin{bmatrix} -3 \\ .75 \end{bmatrix} = \begin{bmatrix} 8\alpha - 3\beta \\ -2\alpha + .75\beta \end{bmatrix}$

Since it is a multiple it is a line

3. Since a is not a scalar multiple of b, they are linearly independent.

4. delta a + beta b + alpha c = 0. Due to it != it is lineraly independent

5. a*b

   a⋅b=8×1+(−2)×4=8−8=0 There for it is **orthogonal**

a*c

```
a⋅c=8×(−3)+(−2)×0.75=−24−1.5=−25.5
There for it is **not orthogonal**
```

b*c

```
b⋅c=1×(−3)+4×0.75=−3+3=0
There for it is **orthogonal**
```

Part 3.3: Projections (5 points)

By hand, find the point in the span of $a = \begin{bmatrix} -1 \\ 3 \end{bmatrix}$ that is closest to $b = \begin{bmatrix} 0 \\ -3 \end{bmatrix}$. Make sure to show **all** work by hand, even if you use `numpy` to verify your answer. **Also, draw a rough sketch** of the operation, including it either as an embedded image in this notebook or in your separate .pdf file.

3.3

$a = \begin{bmatrix} -1 \\ 3 \end{bmatrix} \quad b = \begin{bmatrix} 0 \\ -3 \end{bmatrix}$

$proj(b) = \frac{b \cdot a}{a \cdot a} a$

$b \cdot a = \begin{bmatrix} 0 \\ -3 \end{bmatrix} \cdot \begin{bmatrix} -1 \\ 3 \end{bmatrix} = (0)(-1) + (-3)(3)$

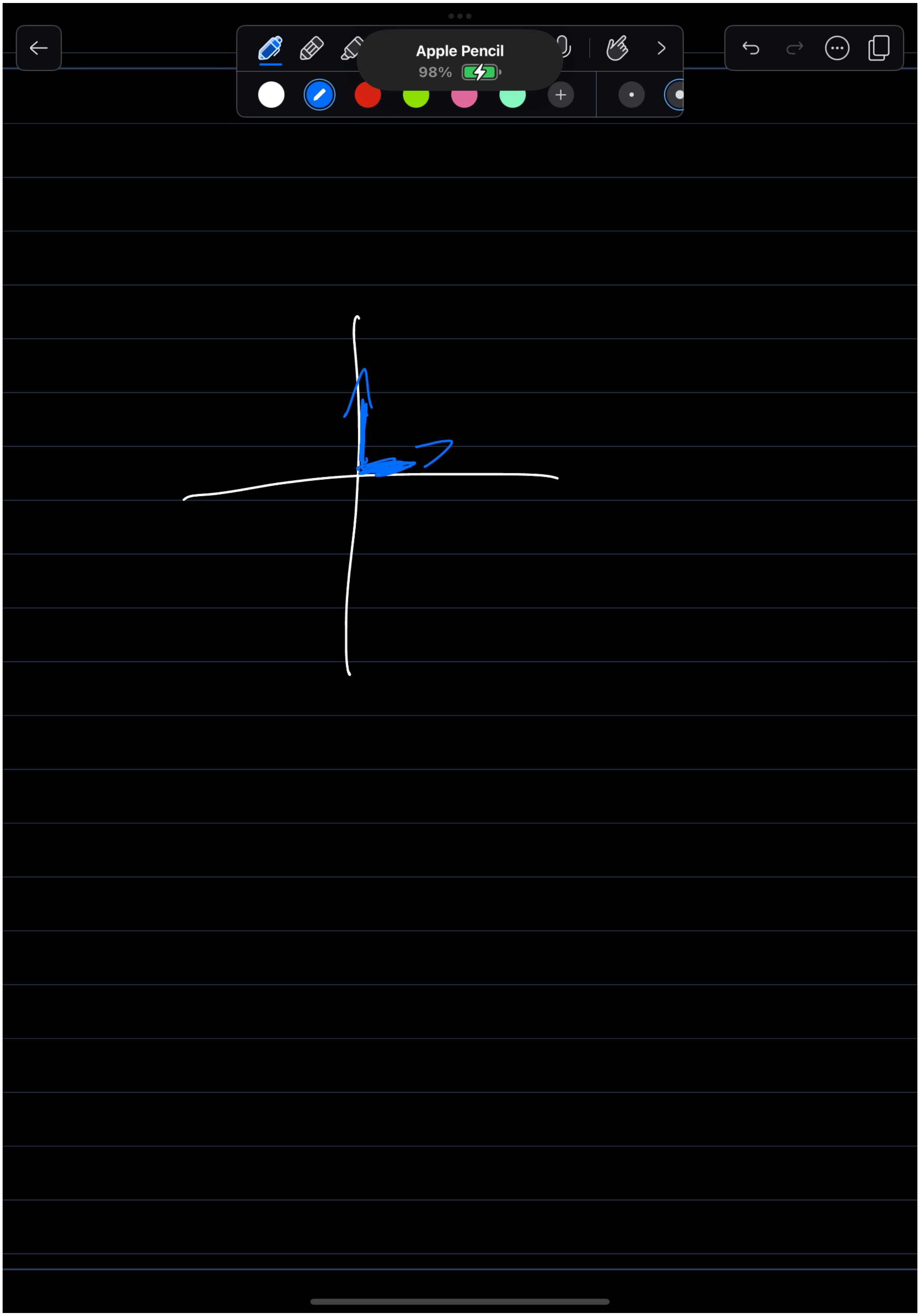
$= 3$

$a \cdot a = \begin{bmatrix} -1 \\ 3 \end{bmatrix} \cdot \begin{bmatrix} -1 \\ 3 \end{bmatrix} = (-1)(-1) + (3)(3)$

$= 10$

$-\frac{9}{10} \begin{bmatrix} -1 \\ 3 \end{bmatrix} = \begin{bmatrix} \frac{9}{10} \\ -\frac{27}{6} \end{bmatrix}$

$= \begin{bmatrix} .9 \\ -2.7 \end{bmatrix}$


Apple Pencil
98%

In [16]:
```
import numpy as np

a = np.array([-1, 3])
b = np.array([0, -3])

projection = np.dot(b, a) / np.dot(a, a) * a

projection
```

Out[16]:
```
array([ 0.9, -2.7])
```

Part 3.4: Line of Best Fit (10 points)

You are interested in if there is a relationship within your friend group between how many siblings they have and how many dates they've been on. You collect the following data from five of your friends:

siblings	dates
0	3
2	9
1	3
0	2
4	6

Find the line of best fit, by hand, for the relationship treating number of siblings as the x feature and number of dates as the y feature. Be sure to include an intercept term. You may verify your answer using `numpy`, but must show all work by hand.

3.4

$x = 0+2+1+0+4 = 4$

$y = 3+9+3+2+6 = 23$

$xy = 0+18+3+0+24 = 45$

$x^2 = 0+4+1+0+16 = 21$

$$\frac{5(45) - 7(23)}{5(21) - (7)^2} = \frac{225 - 161}{105 - 49} = \frac{64}{56}$$

$$= 1.14$$

$$\frac{23 - 1.14(7)}{5} = \frac{23 - 7.98}{5} = \frac{15.02}{5}$$

$$y = 1.14x + \frac{15.02}{5}$$

a# Part 4: Eigenvalues and Eigenvectors (20 points)

Show all math work/steps (no matter how trivial) to receive full credit. You may either use LaTeX typesetting within a Markdown cell, or do it by hand with pen and paper and embed the image in this .ipynb file, or submit a separate pdf file with your handwritten work. Round all decimals to three places.

Find the eigenvalues and eigenvectors for the following matrices by hand, **then** find them in your notebook using `numpy`:

$$A = \begin{bmatrix} -6 & 3 \\ 4 & 5 \end{bmatrix}$$

$$B = \begin{bmatrix} 5 & 6 \\ 2 & 1 \end{bmatrix}$$

In [17]:
```
from sympy import *

λ = symbols('λ')

A_sym = Matrix([[-6, 3], [4, 5]])
B_sym = Matrix([[5, 6], [2, 1]])

char_poly_A = A_sym.charpoly(λ)
eigenvalues_A_sym = solve(char_poly_A.as_expr(), λ)

char_poly_B = B_sym.charpoly(λ)
eigenvalues_B_sym = solve(char_poly_B.as_expr(), λ)

eigenvectors_A_sym = [A_sym.eigenvects()[i][2][0] for i in range(len(A_sym.e

eigenvectors_B_sym = [B_sym.eigenvects()[i][2][0] for i in range(len(B_sym.e

eigenvalues_A_sym
eigenvectors_A_sym

eigenvalues_B_sym
eigenvectors_B_sym
```

Out[17]:
```
[Matrix([
 [-1],
 [ 1]]),
 Matrix([
 [3],
 [1]])]
```

4. $A = \begin{bmatrix} -6 & 3 \\ 4 & 5 \end{bmatrix}$ $I = \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$

$\det(A - \lambda I) = 0$

$\begin{bmatrix} -6 & 3 \\ 4 & 5 \end{bmatrix} - \lambda \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix} = \begin{bmatrix} -6-\lambda & 3 \\ 4 & 5-\lambda \end{bmatrix}$

$= (-6-\lambda)(5-\lambda) - (3)(4)$

$0 = -\lambda^2 + 11\lambda + 18$

$\lambda^2 - 11\lambda - 18 = 0$

$$\frac{-(-11) \pm \sqrt{(-11)^2 - 4(1)(-18)}}{2}$$

$$\lambda = \frac{11 \pm \sqrt{193}}{2}$$

For eigenvector 6:

$$A - 6I = \begin{bmatrix} -12 & 3 \\ 4 & -1 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$-12x + 3y = 0$$
$$y = 4x$$

$$v = \begin{bmatrix} 1 \\ 4 \end{bmatrix} \in \mathbb{R}$$

For eigenvector -7:

$$A + 7I = \begin{bmatrix} 1 & 3 \\ 4 & 12 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$x + 3y = 0$$
$$-3y = x$$

$$v = \begin{bmatrix} -3 \\ 1 \end{bmatrix} \in \mathbb{R}$$

$$\det(B - \lambda I) = (5-\lambda)(1-\lambda) - 12$$
$$0 = \lambda^2 - 6\lambda - 7$$

$$\frac{-(-6) \pm \sqrt{(-6)^2 - 4(1)(-7)}}{2}$$

$$\lambda = \frac{6 \pm 8}{2}$$

$$\lambda = 7, -1$$

For eingnvector 7

$$B - 7I = \begin{bmatrix} -2 & 6 \\ 2 & -6 \end{bmatrix} \begin{bmatrix} x \\ y \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}$$

$$-2x + 6y = 0$$
$$\Rightarrow 3y = x$$

$$\begin{bmatrix} 3 \\ 1 \end{bmatrix} \in \mathbb{R}$$

for eingavector

$$B + I = \begin{bmatrix} 6 & 6 \\ 2 & 2 \end{bmatrix}$$

$$6x + 6y = 0$$

$$y = -x$$

$$\begin{bmatrix} -1 \\ 1 \end{bmatrix} \in \mathbb{R}$$