# Assignment 1 - Probability, Linear Algebra, & Computational Programming

Version 2: Updated 1/14/24 (Question 4 revised)

## *Jiechen Li*

Netid: jl1254

*Names of students you worked with on this assignment*: Yulei Xia; also ChatGPT for calcular concepts and formula clarification and grammar check.

Note: this assignment falls under collaboration Mode 2: Individual Assignment – Collaboration Permitted. Please refer to the syllabus for additional information.

Instructions for all assignments can be found here, and are also linked to from the course syllabus.

Total points in the assignment add up to 90; an additional 10 points are allocated to professionalism and presentation quality.

## Learning Objectives

The purpose of this assignment is to provide a refresher on fundamental concepts that we will use throughout this course and provide an opportunity to develop skills in any of the related skills that may be unfamiliar to you. Through the course of completing this assignment, you will...

- Refresh you knowledge of probability theory including properties of random variables, probability density functions, cumulative distribution functions, and key statistics such as mean and variance.
- Revisit common linear algebra and matrix operations and concepts such as matrix multiplication, inner and outer products, inverses, the Hadamard (element-wise) product, eigenvalues and eigenvectors, orthogonality, and symmetry.
- Practice numerical programming, core to machine learning, by loading and filtering data, plotting data, vectorizing operations, profiling code speed, and debugging and optimizing performance. You will also practice computing probabilities based on simulation.
- Develop or refresh your knowledge of Git version control, which will be a core tool used in the final project of this course

- Apply your skills altogether through an exploratory data analysis to practice data cleaning, data manipulation, interpretation, and communication

We will build on these concepts throughout the course, so use this assignment as a catalyst to deepen your knowledge and seek help with anything unfamiliar.

If some references would be helpful on these topics, I would recommend the following resources:

- Mathematics for Machine Learning by Deisenroth, Faisal, and Ong
- Deep Learning; Part I: Applied Math and Machine Learning Basics by Goodfellow, Bengio, and Courville
- The Matrix Calculus You Need For Deep Learning by Parr and Howard
- Dive Into Deep Learning; Appendix: Mathematics for Deep Learning by Weness, Hu, et al.

*Note: don't worry if you don't understand everything in the references above - some of these books dive into significant minutia of each of these topics.*

---

## Probability and Statistics Theory

*Note: for all assignments, write out equations and math using markdown and LaTeX. I recommend that you complete the work on paper before typing up the final version. For this assignment show your math for questions 1-4, meaning that you should include any intermediate steps necessary to understand the logic of your solution. Most can be completed in 3-4 steps. Being proficient in expressing yourself clearly, sometimes mathematically, is a valuable skill to have as a data scientist*

### 1

**[3 points]**

Let $f(x) = \begin{cases} 0 & x < 0 \\ \alpha x^2 & 0 \leq x \leq 2 \\ 0 & 2 < x \end{cases}$ For what value of $\alpha$ is $f(x)$ a valid probability density function?

**ANSWER**

$f(x) = \alpha x^2$

$\int_0^2 \alpha x^2 dx$ = 1

$\alpha \int_0^2 x^2 dx$ = 1

$\alpha \int_0^2 x^2 dx$ = $\alpha \left(\frac{1}{3}x^3\right)\Big|_0^2$ = 1

$\alpha \left(\frac{2^3}{3}\right) - \alpha \left(\frac{0^3}{3}\right)$ = 1

$\alpha \left(\frac{8}{3}\right)$ = 1

$\alpha$ = $\frac{3}{8}$

---

## 2

**[3 points]** What is the cumulative distribution function (CDF) that corresponds to the following probability distribution function? Please state the value of the CDF for all possible values of $x$.

$$f(x) = \begin{cases} \frac{1}{3} & 0 < x < 3 \\ 0 & \text{otherwise} \end{cases}$$

**ANSWER**

From $-\infty$ to 0, the density is 0.

From $0 < x < 3$, the desity is as follow:

$F(x) = \int_0^x \frac{1}{3}\, dx$

$F(x) = \int_0^x \frac{1}{3}\, dx$

$F(x) = \frac{1}{3}(x)\Big|_0^x$

$F(x) = \frac{1}{3}x - \frac{1}{3} \cdot 0$

$F(x) = \frac{1}{3}x$

From 3 to $\infty$, the desity is 1.

---

## 3

**[6 points]** For the probability distribution function for the random variable $X$,

$$f(x) = \begin{cases} \frac{1}{3} & 0 < x < 3 \\ 0 & \text{otherwise} \end{cases}$$

what is the (a) expected value and (b) variance of $X$. *Show all work*.

**ANSWER**

**(a)**

The expected value of a continuous random variable $X$ is defined as $E[X] = \int_{-\infty}^{\infty} xf(x)dx$. In our case, we have $E[X] = \int_0^3 x\left(\frac{1}{3}\right)dx$, so we do not need to consider from $-\infty$ to 0 or from 3 to $\infty$ because $f(x) = 0[othervise]$. In conclusion, the expected value consider the entire range where $f(x)$ is non-zero.

$$E[X] = \int_0^3 x\left(\frac{1}{3}\right)dx$$

$$E[X] = \int_0^3 x \cdot \frac{1}{3}\,dx = \left[\frac{x^2}{6}\right]_0^3$$

$$E[X] = \frac{3^2}{6} - \frac{0^2}{6} = \frac{3}{2}$$

**(b)**

In order to know $\mathrm{Var}(X)$, we need to calculate $E[X^2]$ as well:

$$E[X^2] = \int_0^3 x^2 \cdot \frac{1}{3}\,dx$$

$$E[X^2] = \frac{1}{3}\int_0^3 x^2\,dx = \frac{1}{3}\left[\frac{x^3}{3}\right]_0^3$$

$$E[X^2] = \frac{1}{3} \cdot \frac{27}{3} = 3$$

$$\mathrm{Var}(X) = E[X^2] - (E[X])^2$$

$$\mathrm{Var}(X) = 3 - (\tfrac{3}{2})^2$$

$$\mathrm{Var}(X) = 3 - 2.25 = 0.75$$

## 4

**[6 points]** You are given the training data below and asked to determine the probability that a sample of $x = 0.54$ comes from class 1, or equivalently, $P(Y = 1|X = 0.54)$. The feature, $x$, can take on real values between 0 and 1.

| $x$ **value range** | **Negative data samples (**$x$,$y = 0$**)** | **Positive data samples (**$x$,$y = 1$**)** |
|---|---|---|
| 0.0 - 0.1 | (0.05,0),(0.07,0) | None |
| 0.1 - 0.2 | (0.11,0),(0.13,0),(0.19,0) | (0.14,1) |
| 0.2 - 0.3 | (0.23,0) | (0.24,1) |

| $x$ **value range** | **Negative data samples (**$x$**,**$y=0$**)** | **Positive data samples (**$x$**,**$y=1$**)** |
|---|---|---|
| 0.3 - 0.4 | (0.35,0), (0.37,0) | (0.32,1) |
| 0.4 - 0.5 | (0.49,0) | (0.47,1) |
| 0.5 - 0.6 | (0.51,0) | (0.53,1) |
| 0.6 - 0.7 | None | (0.61,1) |
| 0.7 - 0.8 | None | (0.77,1) |
| 0.8 - 0.9 | None | (0.83,1) |
| 0.9 - 1.0 | None | (0.92,1),(0.98,1) |

Note: *You don't need to use these data directly*, but this provides an example of how the data could be distributed to form the empirical likelihoods and priors shown below.

You're likely familiar with Bayes' Rule, which for discrete random variables states that:

$$P(Y|X) = \frac{P(X|Y)P(Y)}{P(X)}$$

Additionally, $P(X) = \sum_y P(X|Y=y)P(Y=y)$

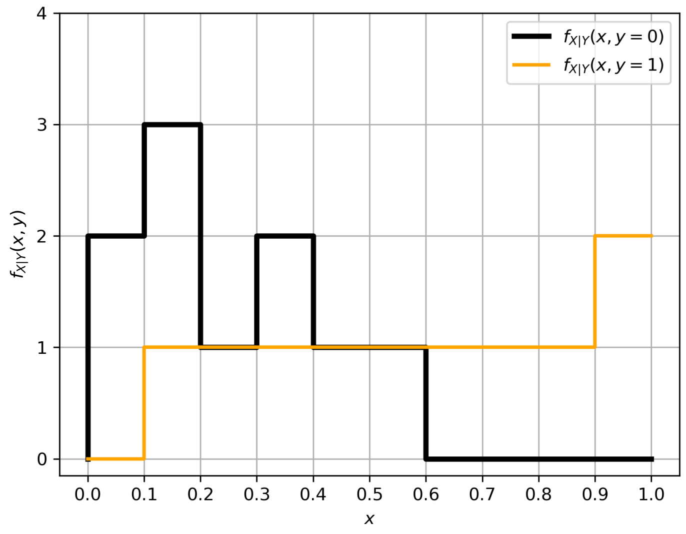
In our case, however, the variable Y is discrete but the variable X is continuous, so we express this function a bit differently. We can compute the poster probability, $P(Y|X)$, based on the likelihood function of the data conditioned on the class of the samples, $f_{X|Y}(x,y)$, the prior $P(Y)$ which is essentially the distribution of the class labels across all the data, and the evidence $f_X(x)$ which is the probability distribution function of the features, $X$, regardless of class labels:

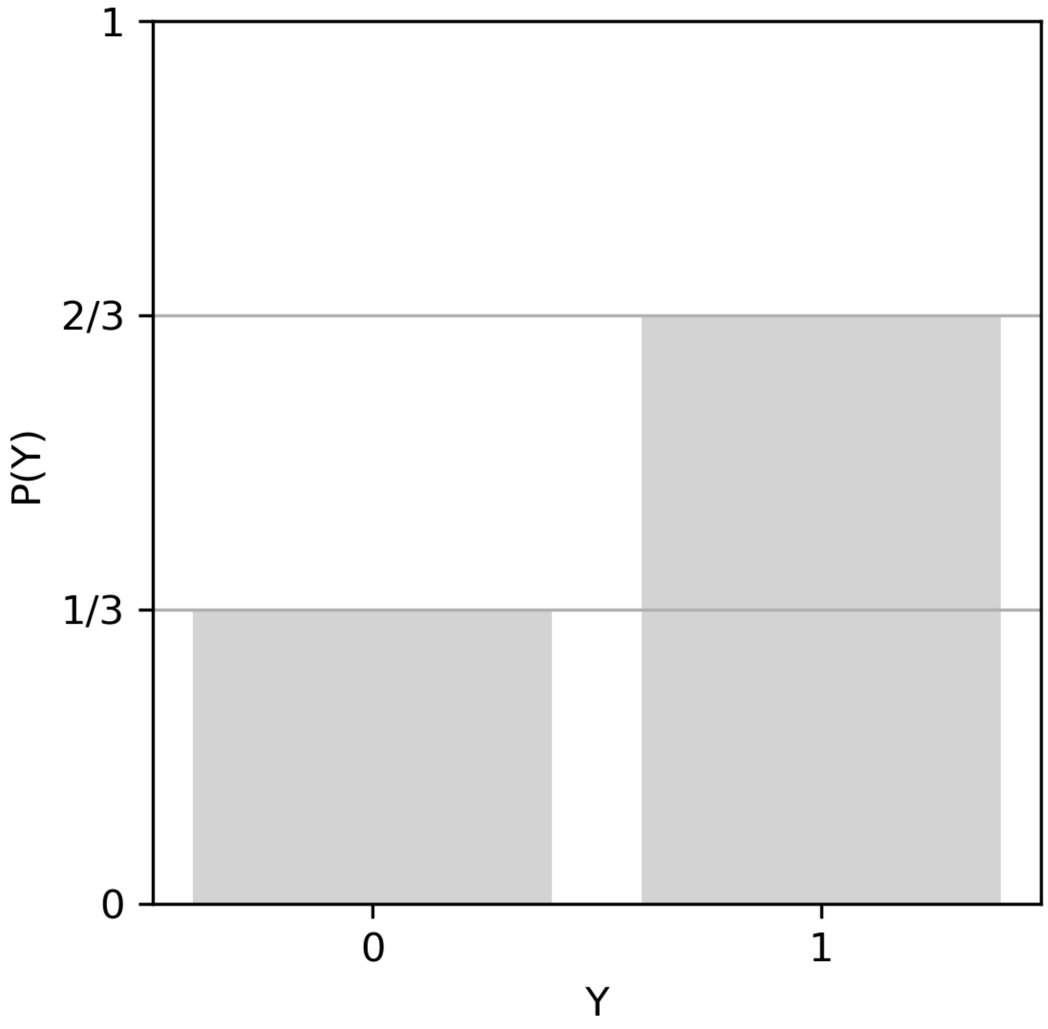
$$P(Y=y|X=x) = \frac{f_{X|Y}(x,y)P(Y=y)}{f_X(x)}$$

Also, note that $f_X(x)$ can be computed in this case (due to the discrete Y values of 0 and 1) as:

$$f_X(x) = f_{X|Y}(x,y=0)P(Y=0) + f_{X|Y}(x,y=1)P(Y=1)$$

Below are the prior and the likelihood functions based on the dataset above. Note that here we use $P(\cdot)$ to note a probability and $f(\cdot)$ to note a probability distribution function.





1. What is $f_{X|Y}(x = 0.54, y = 1)P(Y = 1)$?

2. What is $f_X(x = 0.54)$?
3. What is $P(Y = 1|X = 0.54)$?

*Show each value you use for each computation.*

**ANSWER**

**Question 1**

For $f_{X|Y}(x = 0.54, y = 1)$, I refer the orange line in the first graph at $x = 0.54$. For $P(Y = 1)$, I also refer to the first graph.

$$f_{X|Y}(X = 0.54|Y = 1)P(Y = 1) = 1 \times \frac{2}{3} = \frac{2}{3}$$

**Question 2**

To calculate $f_X(x = 0.54)$, I use the formula provided:
$f_X(x) = f_{X|Y}(x, y = 0)P(Y = 0) + f_{X|Y}(x, y = 1)P(Y = 1)$

$$f_{X|Y}(x = 0.54, y = 0)P(Y = 1) + f_{X|Y}(x = 0.54, y = 0)P(Y = 0) = 1 \times \frac{1}{3} = 1 \times \frac{2}{3}$$

**Question 3**

In order to compute $P(Y = 1|X = 0.54),\quad$ need to use Bayes' Rule provided:
$P(Y = 1|X = 0.54) = \frac{f_{X|Y}(x=0.54,y=1)P(Y=1)}{f_X(x=0.54)} = \frac{2}{3}$

---

# Linear Algebra

## 5

**[5 points]** A common task in machine learning is a change of basis: transforming the representation of our data from one space to another. A prime example of this is through the process of dimensionality reduction as in Principle Components Analysis where we often seek to transform our data from one space (of dimension $n$) to a new space (of dimension $m$) where $m < n$. Assume we have a sample of data of dimension $n = 4$ (as shown below) and we want to transform it into a dimension of $m = 2$.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

(a) What are the dimensions of a matrix, $\mathbf{A}$, that would linearly transform our sample of data, $\mathbf{x}$, into a space of $m = 2$ through the operation $\mathbf{Ax}$?

(b) Express this transformation in terms of the components of $\mathbf{x}$: $x_1$, $x_2$, $x_3$, $x_4$ and the matrix $\mathbf{A}$ where each entry in the matrix is denoted as $a_{i,j}$ (e.g. the entry in the first row and second column would be $a_{1,2}$). Your answer will be in the form of a matrix expressing result of the product $\mathbf{Ax}$.

*Note: please write your answers here in LaTeX*

**ANSWER**

**(a)**

In order to transform a 4 dimentioned column vector to $m = 2$, we need to make matrix $A$ has 2 rows and 4 conlumns.

$A = 2 \times 4$

**(b)**

The matrix multiplication $\mathbf{Ax}$ shows as:

$$\mathbf{A} = \begin{bmatrix} a_{1,1} & a_{1,2} & a_{1,3} & a_{1,4} \\ a_{2,1} & a_{2,2} & a_{2,3} & a_{2,4} \end{bmatrix}$$

$$\mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ x_3 \\ x_4 \end{bmatrix}$$

The product $\mathbf{Ax}$ shows as:

$$\mathbf{Ax} = \begin{bmatrix} a_{1,1}x_1 + a_{1,2}x_2 + a_{1,3}x_3 + a_{1,4}x_4 \\ a_{2,1}x_1 + a_{2,2}x_2 + a_{2,3}x_3 + a_{2,4}x_4 \end{bmatrix}$$

---

## 6

**[14 points] Matrix manipulations and multiplication**. Machine learning involves working with many matrices, so this exercise will provide you with the opportunity to practice those skills.

Let $\mathbf{A} = \begin{bmatrix} 1 & 2 & 3 \\ 2 & 4 & 5 \\ 3 & 5 & 6 \end{bmatrix}$, $\mathbf{b} = \begin{bmatrix} -1 \\ 3 \\ 8 \end{bmatrix}$, $\mathbf{c} = \begin{bmatrix} 4 \\ -3 \\ 6 \end{bmatrix}$, and $\mathbf{I} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$

Compute the following **using Python** or indicate that it cannot be computed. Refer to NumPy's tools for handling matrices. While all answers should be computer using Python, your response to whether each item can be computed should refer to underlying linear algebra. There may be circumstances when Python will produce an output, but based on the dimensions of the matrices involved, the linear algebra operation is not possible. **For the case when an operation is invalid, explain why it is not.**

When the quantity can be computed, please provide both the Python code AND the output of that code (this need not be in LaTex)

1. $\mathbf{A}\mathbf{A}$
2. $\mathbf{A}\mathbf{A}^T$
3. $\mathbf{A}\mathbf{b}$
4. $\mathbf{A}\mathbf{b}^T$
5. $\mathbf{b}\mathbf{A}$
6. $\mathbf{b}^T\mathbf{A}$
7. $\mathbf{b}\mathbf{b}$
8. $\mathbf{b}^T\mathbf{b}$
9. $\mathbf{b}\mathbf{b}^T$
10. $\mathbf{b} + \mathbf{c}^T$
11. $\mathbf{b}^T\mathbf{b}^T$
12. $\mathbf{A}^{-1}\mathbf{b}$
13. $\mathbf{A} \circ \mathbf{A}$
14. $\mathbf{b} \circ \mathbf{c}$

*Note: The element-wise (or Hadamard) product is the product of each element in one matrix with the corresponding element in another matrix, and is represented by the symbol "○".*

**ANSWER**

1. $\mathbf{A}\mathbf{A}$

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

A = np.array([[1, 2, 3], [2, 4, 5], [3, 5, 6]])
b = np.array([[-1], [3], [8]])
c = np.array([[4], [-3], [6]])
```

```
AA = np.dot(A, A)
print(AA)
```

```
[[14 25 31]
 [25 45 56]
 [31 56 70]]
```

2. $\mathbf{A}\mathbf{A}^T$

In [ ]:
```
np.dot(A, A.T)
```

Out[ ]:
```
array([[14, 25, 31],
       [25, 45, 56],
       [31, 56, 70]])
```

3. $\mathbf{A}\mathbf{b}$

In [ ]:
```
np.dot(A, b)
```

Out[ ]:
```
array([[29],
       [50],
       [60]])
```

4. $\mathbf{A}\mathbf{b}^T$

A 1 $\times$ 3 matrix cannot multiply with a 3 $\times$ 3 matrix. In this case, this matrix does not match the rule of linear algebra.

5. $\mathbf{b}\mathbf{A}$

1 for $\mathbf{A}$, and 3 for $\mathbf{b}$ do not match the rule of linear algebra.

6. $\mathbf{b}^T\mathbf{A}$

In [ ]:
```
np.dot(b.T, A)
```

Out[ ]:
```
array([[29, 50, 60]])
```

7. $\mathbf{b}\mathbf{b}$

A 3 $\times$ 1 vector cannot multiply with another 3 $\times$ 1 vector. In this case, this vector does not match the rule of linear algebra.

8. $\mathbf{b}^T\mathbf{b}$

In [ ]:
```
np.dot(b.T, b)
```

```
Out[ ]: array([[74]])
```

9. $\mathbf{b}\mathbf{b}^T$

```
In [ ]: np.dot(b, b.T)
```

```
Out[ ]: array([[ 1, -3, -8],
               [-3,  9, 24],
               [-8, 24, 64]])
```

10. $\mathbf{b} + \mathbf{c}^T$

$\mathbf{b}$ is a $3 \times 1$ vector, but $\mathbf{c}^T$ is a $1 \times 3$ matrix. In this case, this matrix does not match the rule of linear algebra.

11. $\mathbf{b}^T\mathbf{b}^T$

$\mathbf{b}^T$ is a $1 \times 3$ matrix, so $\mathbf{b}^T\mathbf{b}^T$ means $\mathbf{b}^T$ multiply themself. However, the first $\mathbf{b}^T$ has 3 columns, the second one has 1 row. In this case, this matrix does not match the key rule is that the number of columns in the first matrix must match the number of rows in the second matrix.

12. $\mathbf{A}^{-1}\mathbf{b}$

```
In [ ]: if np.linalg.det(A) != 0:
            result = np.dot(np.linalg.inv(A), b)
        else:
            None
        result
```

```
Out[ ]: array([[ 6.],
               [ 4.],
               [-5.]])
```

13. $\mathbf{A} \circ \mathbf{A}$

```
In [ ]: np.multiply(A, A)
```

```
Out[ ]: array([[ 1,  4,  9],
               [ 4, 16, 25],
               [ 9, 25, 36]])
```

14. $\mathbf{b} \circ \mathbf{c}$

```
In [ ]: np.multiply(b, c)
```

```
Out[ ]: array([[-4],
               [-9],
               [48]])
```

---

## 7

**[8 points] Eigenvectors and eigenvalues**. Eigenvectors and eigenvalues are useful for some machine learning algorithms, but the concepts take time to solidly grasp. They are used extensively in machine learning and in this course we will encounter them in relation to Principal Components Analysis (PCA), clustering algorithms, For an intuitive review of these concepts, explore this interactive website at Setosa.io. Also, the series of linear algebra videos by Grant Sanderson of 3Brown1Blue are excellent and can be viewed on youtube here. For these questions, numpy may once again be helpful.

1. Calculate the eigenvalues and corresponding eigenvectors of matrix $\mathbf{A}$ above, from the last question.
2. Choose one of the eigenvector/eigenvalue pairs, $\mathbf{v}$ and $\lambda$, and show that $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$. This relationship extends to higher orders: $\mathbf{A}\mathbf{A}\mathbf{v} = \lambda^2\mathbf{v}$
3. Show that the eigenvectors are orthogonal to one another (e.g. their inner product is zero). This is true for eigenvectors from real, symmetric matrices. In three dimensions or less, this means that the eigenvectors are perpendicular to each other. Typically we use the orthogonal basis of our standard x, y, and z, Cartesian coordinates, which allows us, if we combine them linearly, to represent any point in a 3D space. But any three orthogonal vectors can do the same. We will see this property is used in PCA to identify the dimensions of greatest variation in our data when we discuss dimensionality reduction.

**ANSWER**

```
In [ ]: # Question 1: Calculate the eigenvalues and corresponding eigenvectors of ma

        A = np.array([[1, 2, 3], [2, 4, 5], [3, 5, 6]])
        eig = np.linalg.eig(A)

        eigenvalues = eig[0]
        eigenvectors = eig[1]

        print("Eigenvalues:", eigenvalues)
        print("Eigenvectors:", eigenvectors)
```

```
Eigenvalues: [11.34481428 -0.51572947  0.17091519]
Eigenvectors: [[-0.32798528 -0.73697623  0.59100905]
 [-0.59100905 -0.32798528 -0.73697623]
 [-0.73697623  0.59100905  0.32798528]]
```

2. Verify $\mathbf{A}\mathbf{v} = \lambda\mathbf{v}$

```
In [ ]: # Question 2

        # choose the first eigenvector
        v = eigenvectors[:, 0]

        # corresponding eigenvalue
        lambda_ = eigenvalues[0]

        # verify
        left_side = np.dot(A, v)
        right_side = lambda_ * v

        # see whether or not left is approximately equal to right
        np.allclose(left_side, right_side)
```

```
Out[ ]: True
```

```
In [ ]: # Question 3: Show that the eigenvectors are orthogonal to one another

        v1 = eigenvectors[:, 0]
        v2 = eigenvectors[:, 1]
        v3 = eigenvectors[:, 2]

        # see whether or dot products are 0 (or close to 0)
        dot_product_12 = np.dot(v1, v2)
        dot_product_13 = np.dot(v1, v3)
        dot_product_23 = np.dot(v2, v3)

        np.allclose(dot_product_12, 0)
        np.allclose(dot_product_13, 0)
        np.allclose(dot_product_23, 0)
```

```
Out[ ]: True
```

---

# Numerical Programming

## 8

**[10 points]** Loading data and gathering insights from a real dataset

In data science, we often need to have a sense of the idiosyncrasies of the data, how they relate to the questions we are trying to answer, and to use that information to help us to determine what approach, such as machine learning, we may need to apply to achieve our goal. This exercise provides practice in exploring a dataset and answering question that might arise from applications related to the data.

**Data**. The data for this problem can be found in the `data` subfolder in the `assignments` folder on github. The filename is `a1_egrid2016.xlsx`. This dataset is the Environmental Protection Agency's (EPA) Emissions & Generation Resource Integrated Database (eGRID) containing information about all power plants in the United States, the amount of generation they produce, what fuel they use, the location of the plant, and many more quantities. We'll be using a subset of those data.

The fields we'll be using include:

| field | description |
|---|---|
| SEQPLT16 | eGRID2016 Plant file sequence number (the index) |
| PSTATABB | Plant state abbreviation |
| PNAME | Plant name |
| LAT | Plant latitude |
| LON | Plant longitude |
| PLPRMFL | Plant primary fuel |
| CAPFAC | Plant capacity factor |
| NAMEPCAP | Plant nameplate capacity (Megawatts MW) |
| PLNGENAN | Plant annual net generation (Megawatt-hours MWh) |
| PLCO2EQA | Plant annual CO2 equivalent emissions (tons) |

For more details on the data, you can refer to the eGrid technical documents. For example, you may want to review page 45 and the section "Plant Primary Fuel (PLPRMFL)", which gives the full names of the fuel types including WND for wind, NG for natural gas, BIT for Bituminous coal, etc.

There also are a couple of "gotchas" to watch out for with this dataset:

- The headers are on the second row and you'll want to ignore the first row (they're more detailed descriptions of the headers).
- NaN values represent blanks in the data. These will appear regularly in real-world data, so getting experience working with these sorts of missing values will be important.

**Your objective**. For this dataset, your goal is to answer the following questions about electricity generation in the United States:

**(a)** Which plant has generated the most energy (measured in MWh)?

**(b)** What is the name of the northern-most power plant in the United States?

**(c)** What is the state where the northern-most power plant in the United States is located?

**(d)** Plot a bar plot showing the amount of energy produced by each fuel type across all plants.

**(e)** From the plot in (d), which fuel for generation produces the most energy (MWh) in the United States?

**ANSWER**

In [ ]:
```
import pandas as pd

# pip install openpyxl
# load the dataset

file_path = "a1_egrid2016.xlsx"
data = pd.read_excel(file_path, header=1)
data.head()
```

Out[ ]:

| | SEQPLT16 | PSTATABB | PNAME | LAT | LON | PLPRMFL | CAPFAC | NA |
|---|---|---|---|---|---|---|---|---|
| **0** | 1 | AK | 7-Mile Ridge Wind Project | 63.210689 | -143.247156 | WND | NaN | |
| **1** | 2 | AK | Agrium Kenai Nitrogen Operations | 60.673200 | -151.378400 | NG | NaN | |
| **2** | 3 | AK | Alakanuk | 62.683300 | -164.654400 | DFO | 0.05326 | |
| **3** | 4 | AK | Allison Creek Hydro | 61.084444 | -146.353333 | WAT | 0.01547 | |
| **4** | 5 | AK | Ambler | 67.087980 | -157.856719 | DFO | 0.13657 | |

In [ ]:
```
# Question (a)

most_energy_plant = data.loc[data["PLNGENAN"].idxmax()]
print(most_energy_plant)
```

```
SEQPLT16              391
PSTATABB               AZ
PNAME          Palo Verde
LAT               33.3881
LON             -112.8617
PLPRMFL               NUC
CAPFAC            0.87801
NAMEPCAP           4209.6
PLNGENAN     32377476.999
PLCO2EQA              0.0
Name: 390, dtype: object
```

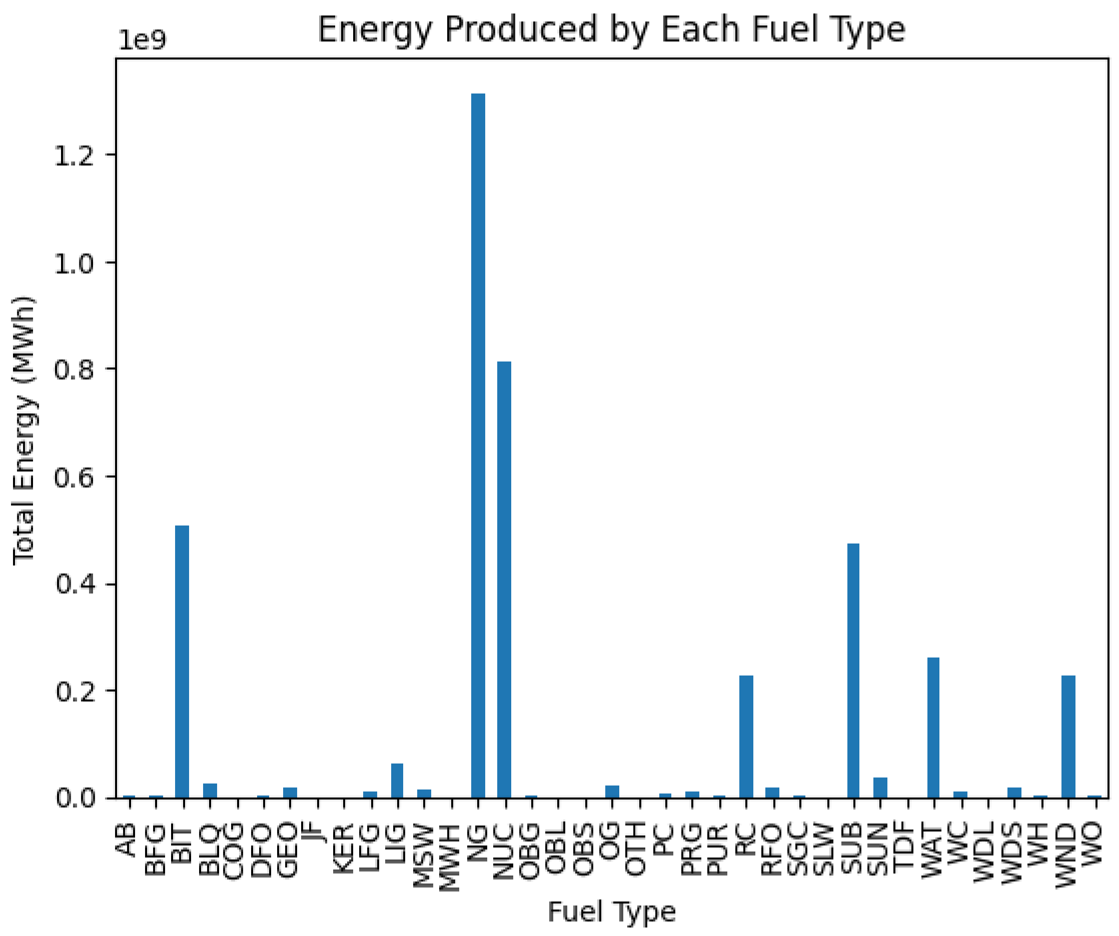
In [ ]:
```
# Question (b)

northernmost_plant = data.loc[data["LAT"].idxmax()]
print(northernmost_plant)
```

```
SEQPLT16           12
PSTATABB           AK
PNAME          Barrow
LAT            71.292
LON         -156.7786
PLPRMFL            NG
CAPFAC        0.28208
NAMEPCAP         20.3
PLNGENAN      50162.0
PLCO2EQA     44205.17
Name: 11, dtype: object
```

In [ ]:
```
# Question (c)

northernmost_state = northernmost_plant["PSTATABB"]
print(northernmost_state)
```

```
AK
```

In [ ]:
```
# Question (d)

import matplotlib.pyplot as plt

# groupby fuel type and sum energy generation
fuel_type_energy = data.groupby("PLPRMFL")["PLNGENAN"].sum()

# plot
fuel_type_energy.plot(kind="bar")
plt.title("Energy Produced by Each Fuel Type")
plt.xlabel("Fuel Type")
plt.ylabel("Total Energy (MWh)")
plt.show()
```



In [ ]:
```python
# Question (e)

max_energy_fuel = fuel_type_energy.idxmax()
print(max_energy_fuel)
```

```
NG
```

---

## 9

**[6 points]** *Vectorization*. When we first learn to code and think about iterating over an array, we often use loops. If implemented correctly, that does the trick. In machine learning, we iterate over so much data that those loops can lead to significant slow downs if they are not computationally efficient. In Python, vectorizing code and relying on matrix operations with efficient tools like numpy is typically the faster approach. Of course, numpy relies on loops to complete the computation, but this is at a lower level of programming (typically in C), and therefore is much more efficient. This exercise will explore the benefits of vectorization. Since many machine learning techniques rely on matrix operations, it's helpful to begin thinking about implementing algorithms using vector forms.

Begin by creating an array of 10 million random numbers using the numpy `random.randn` module. Compute the sum of the squares of those random numbers first in a for loop, then using Numpy's `dot` module to perform an inner (dot) product. Verify that your code produces the same output in each case. Time how long it takes to compute each and report the results and report the output. How many times faster is the vectorized code than the for loop approach? (Note - your results may vary from run to run).

Your output should use the `print()` function as follows (where the # symbols represent your answers, to a reasonable precision of 4-5 significant figures):

`Time [sec] (non-vectorized): ######`

`Time [sec] (vectorized):     ######`

`The vectorized code is ##### times faster than the nonvectorized code`

**ANSWER**

In [ ]:
```
import time

# create a random array of 10 million numbers
random_nums = np.random.randn(10000000)

# sum of squares using a for loop
# start
start_time = time.time()
sum_squares_loop = 0
for number in random_nums:
    sum_squares_loop += number**2

# end
end_time = time.time()

# calculate elapsed time
time_non_vectorized = end_time - start_time

print("Time [sec] (non-vectorized):", time_non_vectorized)
```

```
Time [sec] (non-vectorized): 1.0065598487854004
```

In [ ]:
```
# sum of squares using vectorized code (dot product)

# start
start_time = time.time()
sum_squares_vectorized = np.dot(random_nums, random_nums)

# end
end_time = time.time()

# Calculate elapsed time
```

```
time_vectorized = end_time - start_time

print("Time [sec] (vectorized):    ", time_vectorized)
```

```
Time [sec] (vectorized):     0.014971971511840820
```

In [ ]:
```
# compare the speeds
speedup = time_non_vectorized / time_vectorized
print("The vectorized code is", speedup, "times faster than the non-vectoriz
```

```
The vectorized code is 67.22961287959616 times faster than the non-vectorize
d code
```

In [ ]:
```
# verify that both results are the same
print(
    "Are the results the same:", np.allclose(sum_squares_loop, sum_squares_v
)
```

```
Are the results the same: True
```

---

## 10

**[10 points]** This exercise will walk through some basic numerical programming and probabilistic thinking exercises, two skills which are frequently used in machine learning for answering questions from our data.

1. Synthesize $n = 10^4$ normally distributed data points with mean $\mu = 2$ and a standard deviation of $\sigma = 1$. Call these observations from a random variable $X$, and call the vector of observations that you generate, $\mathbf{x}$.
2. Calculate the mean and standard deviation of $\mathbf{x}$ to validate (1) and provide the result to a precision of four significant figures.
3. Plot a histogram of the data in $\mathbf{x}$ with 30 bins
4. What is the 90th percentile of $\mathbf{x}$? The 90th percentile is the value below which 90% of observations can be found.
5. What is the 99th percentile of $\mathbf{x}$?
6. Now synthesize $n = 10^4$ normally distributed data points with mean $\mu = 0$ and a standard deviation of $\sigma = 3$. Call these observations from a random variable $Y$, and call the vector of observations that you generate, $\mathbf{y}$.
7. Create a new figure and plot the histogram of the data in $\mathbf{y}$ on the same axes with the histogram of $\mathbf{x}$, so that both histograms can be seen and compared.
8. Using the observations from $\mathbf{x}$ and $\mathbf{y}$, estimate $E[XY]$

**ANSWER**

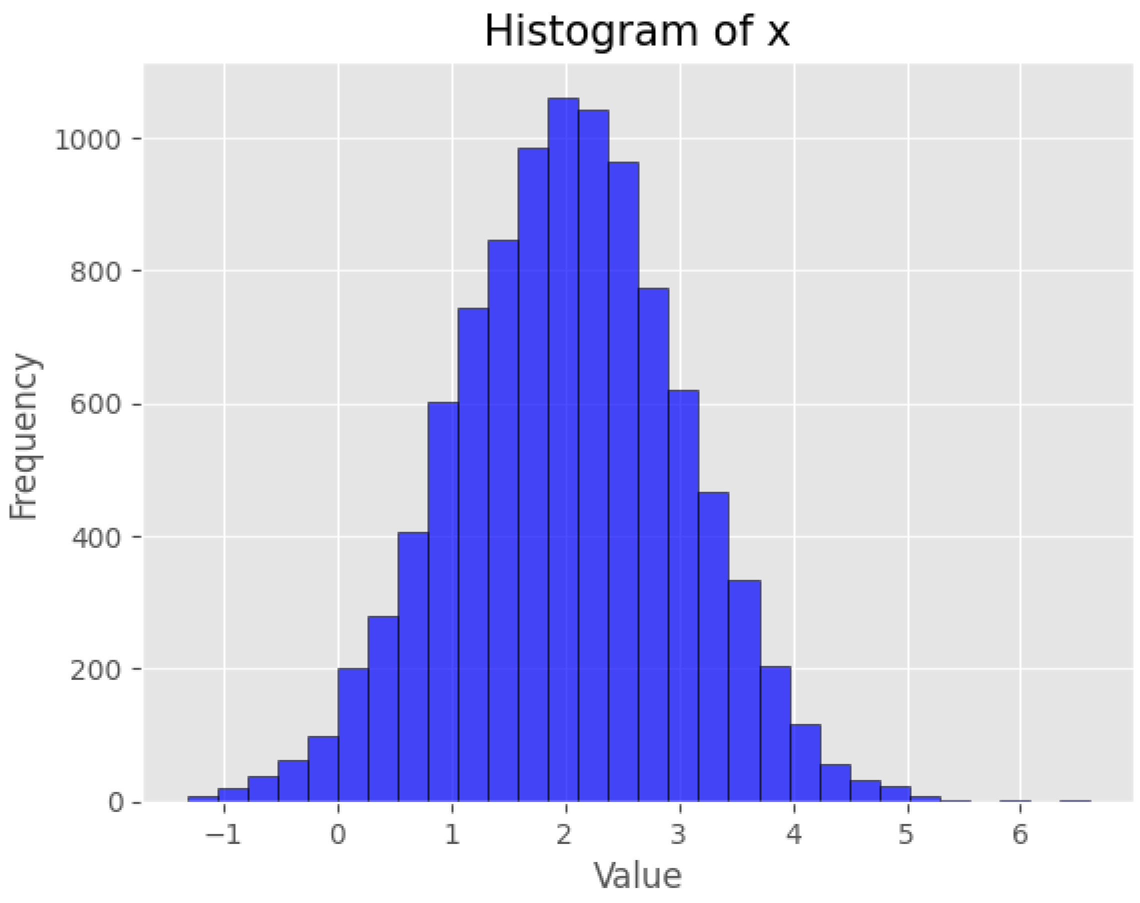
In [ ]:
```
# Question 1

n = 10000
mu_x = 2
```

```
sigma_x = 1
x = np.random.normal(mu_x, sigma_x, n)
```

In [ ]:
```
# Question 2

mean_x = np.mean(x)
std_dev_x = np.std(x)

print(round(mean_x, 4))
print(round(std_dev_x, 4))
```

```
2.014
1.0031
```

In [ ]:
```
# Question 3

# set a style
plt.style.use("ggplot")

# create the histogram
plt.hist(x, bins=30, color="blue", edgecolor="black", alpha=0.7)

# add grid, title, and labels
plt.grid(axis="y", alpha=0.75)
plt.title("Histogram of x", fontsize=15)
plt.xlabel("Value", fontsize=12)
plt.ylabel("Frequency", fontsize=12)


plt.show()
```



In [ ]:
```
# Question 4

percentile_90_x = np.percentile(x, 90)
print(percentile_90_x)
```

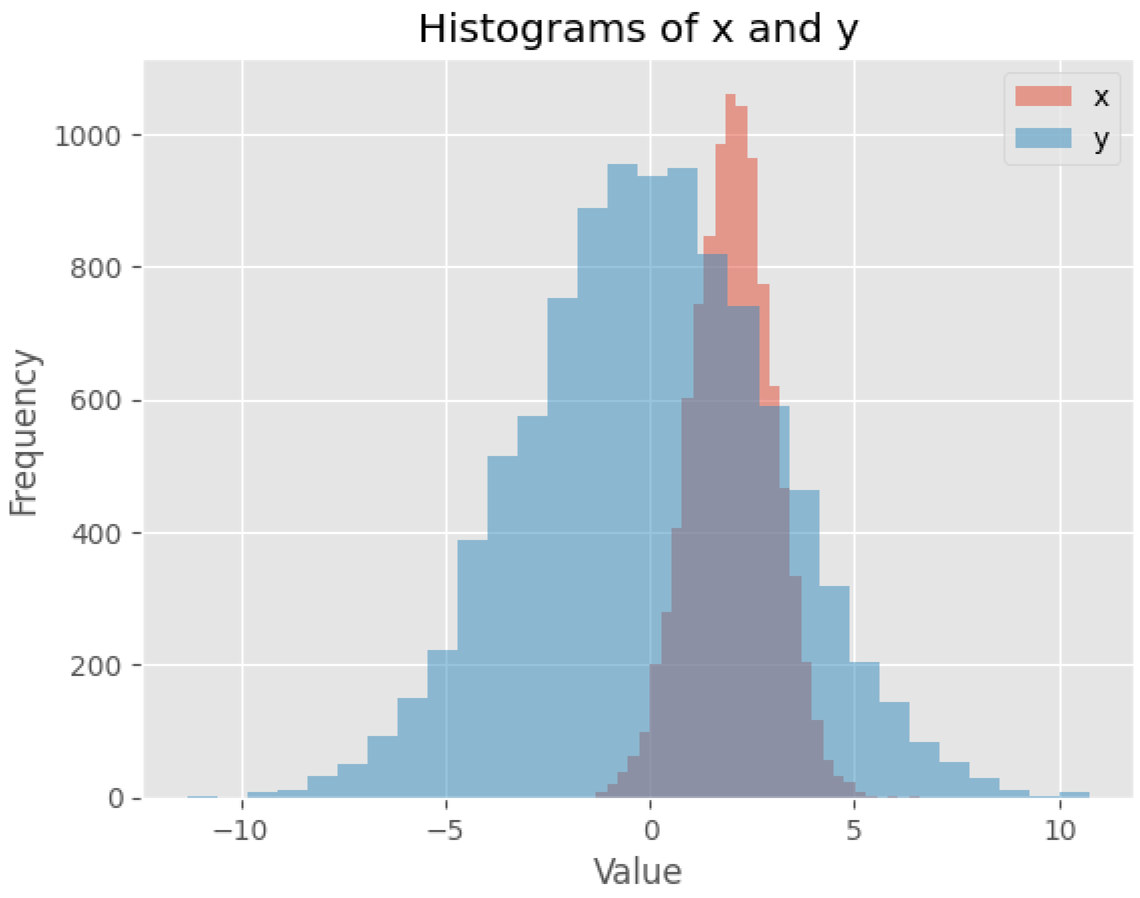
```
3.301093891177147
```

In [ ]:
```
# Question 5

percentile_99_x = np.percentile(x, 99)
print(percentile_99_x)
```

```
4.310929321940595
```

In [ ]:
```
# Question 6

mu_y = 0
sigma_y = 3
y = np.random.normal(mu_y, sigma_y, n)
```

In [ ]:
```
# Question 7

plt.hist(x, bins=30, alpha=0.5, label="x")
plt.hist(y, bins=30, alpha=0.5, label="y")
plt.title("Histograms of x and y")
plt.xlabel("Value")
plt.ylabel("Frequency")
plt.legend()
plt.show()
```



In [ ]:
```
# Question 8

E_XY = np.mean(x * y)
print(E_XY)
```

```
-0.004947970935483045
```

---

# Version Control via Git

## 11

**[4 points]** Git is efficient for collaboration, and expectation in industry, and one of the best ways to share results in academia. You can even use some Git repositories (e.g. Github) as hosts for website, such as with the course website. As a data scientist with experience in machine learning, Git is expected. We will interact with Git repositories (a.k.a. repos) throughout this course, and your project will require the use of git repos for collaboration.

Complete the Atlassian Git tutorial, specifically the following listed sections. Try each concept that's presented. For this tutorial, instead of using BitBucket as your remote repository host, you may use your preferred platform such as Github or Duke's Gitlab.

1. What is version control
2. What is Git
3. Install Git
4. Setting up a repository
5. Saving changes
6. Inspecting a repository
7. Undoing changes
8. Rewriting history
9. Syncing
10. Making a pull request
11. Using branches
12. Comparing workflows

I also have created two videos on the topic to help you understand some of these concepts: Git basics and a step-by-step tutorial.

As an additional resource, Microsoft now offers a git tutorial on this topic as well.

For your answer, affirm that you *either* completed the tutorials above OR have previous experience with ALL of the concepts above. Confirm this by typing your name below and selecting the situation that applies from the two options in brackets.

**ANSWER**

*I, [**Jiechen Li**], affirm that I have [**I have previous experience that covers all the content in this tutorial**]*

---

# Exploratory Data Analysis

## 12

**[15 points]** Here you'll bring together some of the individual skills that you demonstrated above and create a Jupyter notebook based blog post on your exploratory data analysis. Your goal is to identify a question or problem and to work towards solving it or providing additional information or evidence (data) related to it through your data analysis. Below, we walk through a process to follow for your analysis. Additionally, you can find an example of a well-done exploratory data analysis here from past years.

1. Find a dataset that interests you and relates to a question or problem that you find intriguing.
2. Describe the dataset, the source of the data, and the reason the dataset was of interest. Include a description of the features, data size, data creator and year of creation (if available), etc. What question are you hoping to answer through exploring the dataset?
3. Check the data and see if they need to be cleaned: are there missing values? Are there clearly erroneous values? Do two tables need to be merged together? Clean the data so it can be visualized. If the data are clean, state how you know they are clean (what did you check?).
4. Plot the data, demonstrating interesting features that you discover. Are there any relationships between variables that were surprising or patterns that emerged? Please exercise creativity and curiosity in your plots. You should have at least 4 plots exploring the data in different ways.
5. What insights are you able to take away from exploring the data? Is there a reason why analyzing the dataset you chose is particularly interesting or important? Summarize this for a general audience (imagine your publishing a blog post online) - boil down your findings in a way that is accessible, but still accurate.

Here your analysis will evaluated based on:

1. Motivation: was the purpose of the choice of data clearly articulated? Why was the dataset chosen and what was the goal of the analysis?
2. Data cleaning: were any issues with the data investigated and, if found, were they resolved?
3. Quality of data exploration: were at least 4 unique plots (minimum) included and did those plots demonstrate interesting aspects of the data? Was there a clear purpose and takeaway from EACH plot?
4. Interpretation: Were the insights revealed through the analysis and their potential implications clearly explained? Was there an overall conclusion to the analysis?

**ANSWER**

## Education EDA

**Name of Dataset**: Number of Participating Students with Disabilities in Reading and Mathematics Assessment With and Without Accommodations

**Source of Dataset**: North Carolina Department of Public Instruction, an educational official website of the States of North Carolina

Motivation:

- The purpose of the choosen dataset is clearly articulated. For the 2022–23 school year, schools continued to deal with student learning loss due to the COVID pandemic. Data from the 2022–23 school year should be reviewed in these contexts and comparisons to prior years' results should be made with caution. With the backdrop of the COVID pandemic and its effect on education, this analysis aims to provide insights into how different accommodations may influence student participation and performance.

- The goal of this analysis is to :

  1): understand the impact of accommodations on the participation of students with disabilities in reading and mathematics assessments across various grade levels in North Carolina;

  2): identify any significant trends or disparities in the participation rates with and without accommodations across different districts and schools. This could highlight areas where additional resources or focus might be needed to support students with disabilities;

  3): explore whether there are particular grade levels or subjects where students with disabilities participate more or less with accommodations, providing a deeper understanding of the educational landscape in the wake of the pandemic.

- Ultimately, this EDA seeks to inform educational policy and support mechanisms by pinpointing the effectiveness of accommodations and identifying potential areas for improvement in educational strategies for students with disabilities.

### Load the Dataset

In [ ]:
```
file_path_2 = "2023 SWD Accommodations Report.csv"
swd_data = pd.read_csv(file_path_2)

swd_data.head()
```

Out[ ]:

| | Reporting year | District code | District name | School code | District or school name | Subject | Grade level | Participating with accommodations | ac |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 2023 | NC | State of North Carolina | NC-SEA | State of North Carolina | RD | 3 | 8,065 | |
| **1** | 2023 | NC | State of North Carolina | NC-SEA | State of North Carolina | RD | 4 | 8,874 | |
| **2** | 2023 | NC | State of North Carolina | NC-SEA | State of North Carolina | RD | 5 | 8,961 | |
| **3** | 2023 | NC | State of North Carolina | NC-SEA | State of North Carolina | RD | 6 | 9,110 | |
| **4** | 2023 | NC | State of North Carolina | NC-SEA | State of North Carolina | RD | 7 | 9,383 | |

In [ ]:
```
# check for missing values
swd_data.isnull().sum()
```

Out[ ]:
```
Reporting year                            0
District code                             0
District name                             0
School code                               0
District or school name                   0
Subject                                   0
Grade level                               0
Participating with accommodations         0
Participating without accommodations      0
Unnamed: 9                            17477
dtype: int64
```

In [ ]:
```
# check for duplicates
swd_data.duplicated().sum()
```

Out[ ]:
```
0
```

## Data Cleaning

In [ ]:
```
# convert to numeric
swd_data["Participating with accommodations"] = pd.to_numeric(
    swd_data["Participating with accommodations"]
    .str.replace(",", "")
    .str.replace("*", ""),
    errors="coerce",
)
```

```
swd_data["Participating without accommodations"] = pd.to_numeric(
    swd_data["Participating without accommodations"]
    .str.replace(",", "")
    .str.replace("*", ""),
    errors="coerce",
)

# drop NA column
swd_data.drop(columns=["Unnamed: 9"], inplace=True)

# verify the changes
swd_data.head()
```

Out[ ]:

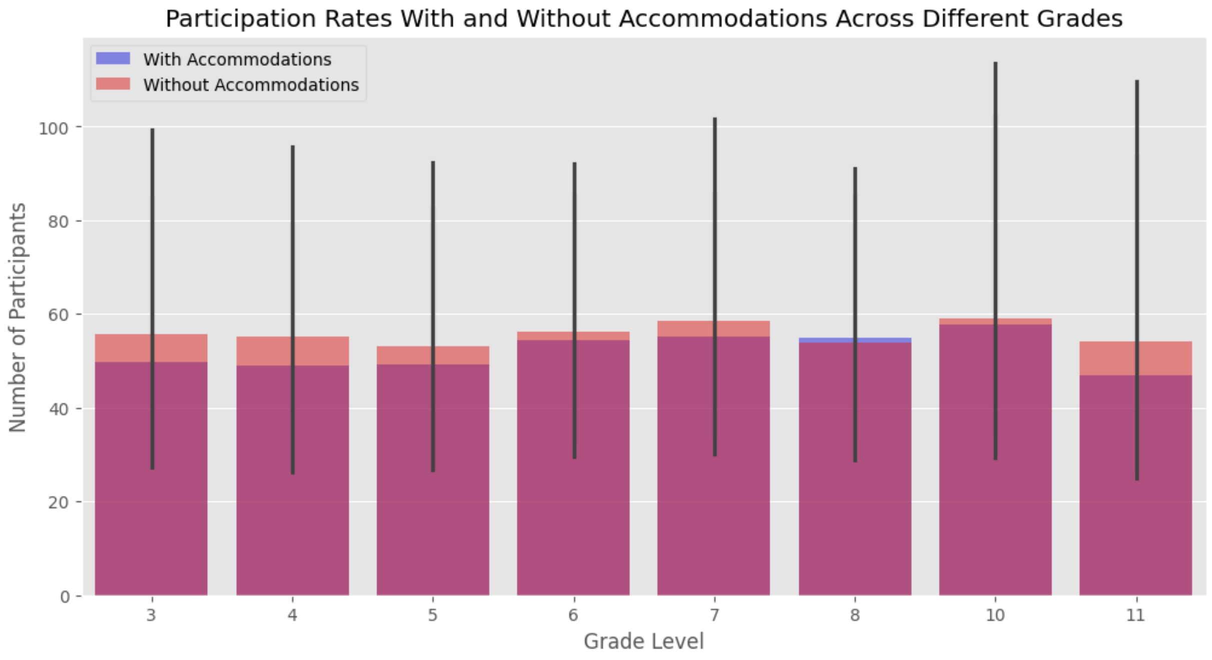
| | Reporting year | District code | District name | School code | District or school name | Subject | Grade level | Participating with accommodations | ac |
|---|---|---|---|---|---|---|---|---|---|
| **0** | 2023 | NC | State of North Carolina | NC-SEA | State of North Carolina | RD | 3 | 8065.0 | |
| **1** | 2023 | NC | State of North Carolina | NC-SEA | State of North Carolina | RD | 4 | 8874.0 | |
| **2** | 2023 | NC | State of North Carolina | NC-SEA | State of North Carolina | RD | 5 | 8961.0 | |
| **3** | 2023 | NC | State of North Carolina | NC-SEA | State of North Carolina | RD | 6 | 9110.0 | |
| **4** | 2023 | NC | State of North Carolina | NC-SEA | State of North Carolina | RD | 7 | 9383.0 | |

## EDA

### 1. Participation Rates Across Different Grades

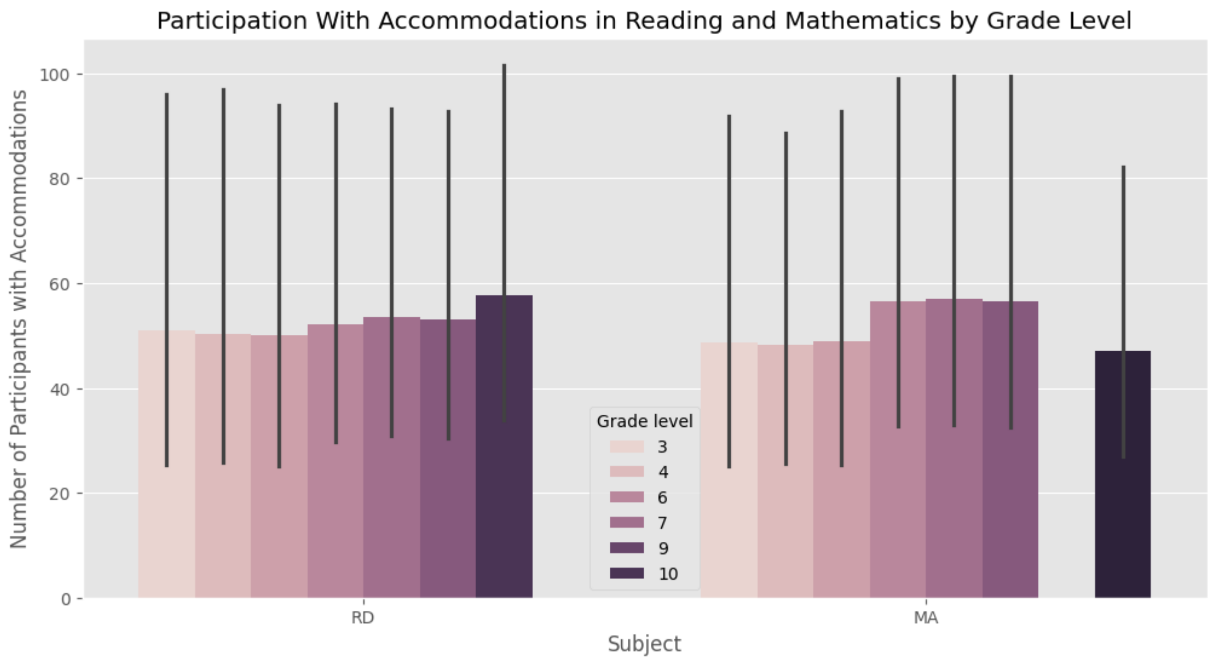
In [ ]:

```
import seaborn as sns
import warnings

warnings.filterwarnings("ignore")

# plot participation rates across different grades
import matplotlib.pyplot as plt
import seaborn as sns

# Plotting participation rates across different grades
plt.figure(figsize=(12, 6))
sns.barplot(
    data=swd_data,
    x="Grade level",
    y="Participating with accommodations",
    color="blue",
    alpha=0.5,
    label="With Accommodations",
)
sns.barplot(
    data=swd_data,
    x="Grade level",
    y="Participating without accommodations",
    color="red",
    alpha=0.5,
    label="Without Accommodations",
)
plt.title("Participation Rates With and Without Accommodations Across Differ
plt.xlabel("Grade Level")
plt.ylabel("Number of Participants")
plt.legend()
plt.show()
```



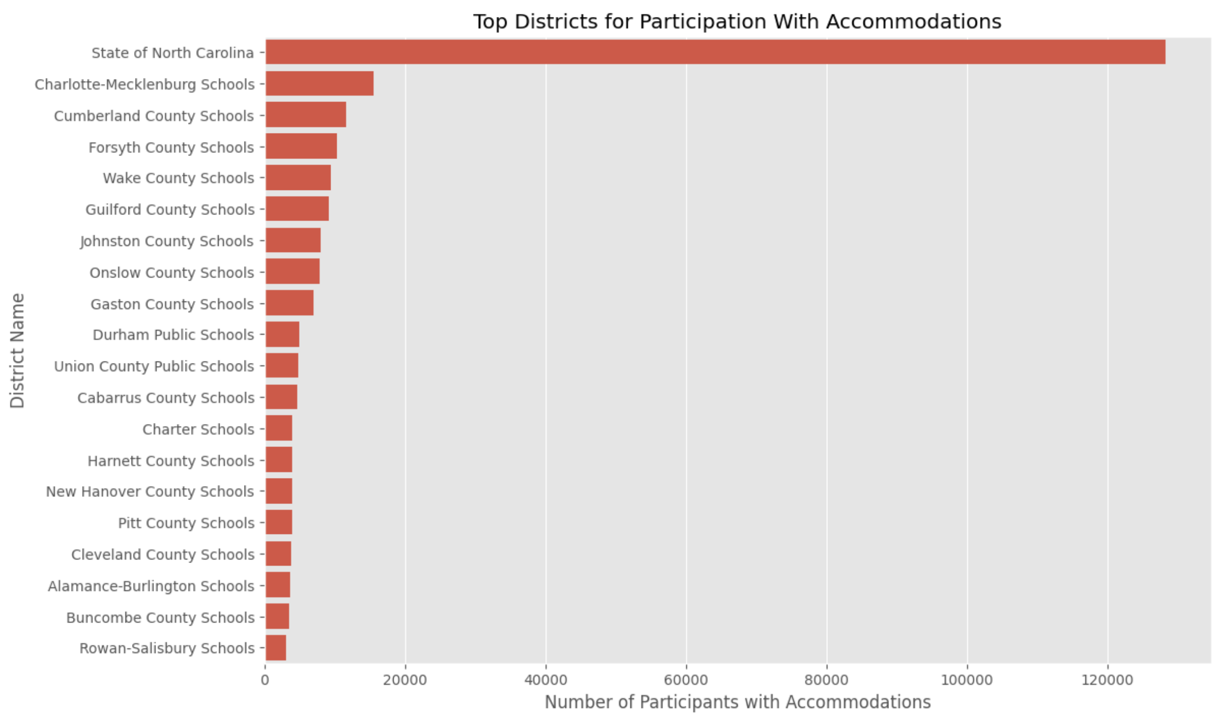
### 2. Comparing Participation in Reading and Mathematics

```
In [ ]: plt.figure(figsize=(12, 6))
        sns.barplot(
            data=swd_data, x="Subject", y="Participating with accommodations", hue="
        )
        plt.title("Participation With Accommodations in Reading and Mathematics by G
        plt.xlabel("Subject")
        plt.ylabel("Number of Participants with Accommodations")
        plt.show()
```



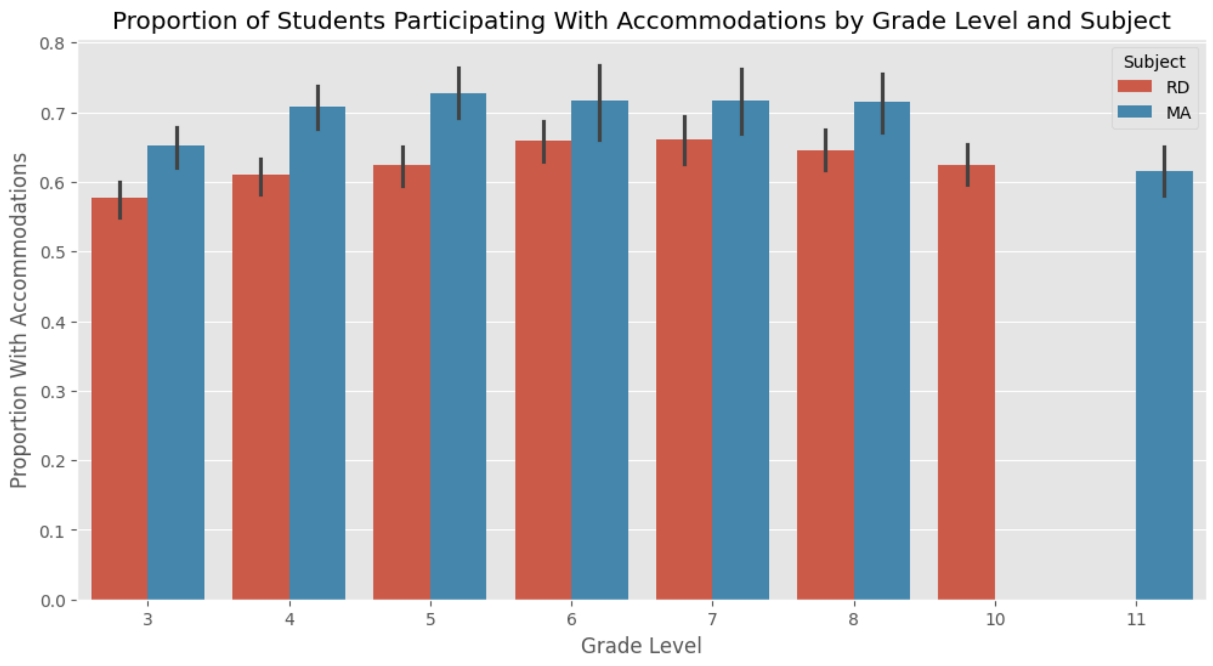
### 3. Disparities Across Different Districts

In [ ]:
```python
# calculating total participants with accommodations for each district
district_accommodations = (
    swd_data.groupby("District name")["Participating with accommodations"]
    .sum()
    .reset_index()
)

# sort top N districts
top_n_districts = district_accommodations.sort_values(
    by="Participating with accommodations", ascending=False
).head(20)

# plot the top 20 districts
plt.figure(figsize=(12, 8))
sns.barplot(
    data=top_n_districts,
    y="District name",
    x="Participating with accommodations",
    orient="h",
)
plt.title("Top Districts for Participation With Accommodations")
plt.xlabel("Number of Participants with Accommodations")
plt.ylabel("District Name")
plt.show()
```



### 4. Effectiveness of Accommodations

In [ ]:
```
swd_data["Total Participating"] = (
    swd_data["Participating with accommodations"]
    + swd_data["Participating without accommodations"]
)
swd_data["Proportion With Accommodations"] = (
    swd_data["Participating with accommodations"] / swd_data["Total Particip
)

plt.figure(figsize=(12, 6))
sns.barplot(
    data=swd_data, x="Grade level", y="Proportion With Accommodations", hue=
)
plt.title(
    "Proportion of Students Participating With Accommodations by Grade Level
)
plt.xlabel("Grade Level")
plt.ylabel("Proportion With Accommodations")
plt.show()
```



## Interpretation

The exploratory data analysis of the 2022-23 school year data reveals several key insights into the participation of students with disabilities in North Carolina's educational assessments:

**1. Participation with Accommodations**:

The analysis indicated that a significant number of students with disabilities participate in assessments with accommodations. This suggests that schools are recognizing and acting upon the unique needs of these students to provide them with the necessary support during assessments.

**2. Disparities Across Districts**:

There were noticeable disparities in participation rates with accommodations across different districts. Some districts showed higher participation rates with accommodations, which may indicate a more robust support system or a greater awareness of student needs. In contrast, districts with lower participation rates may require additional scrutiny to ensure that all students have equal access to accommodations.

**3.Grade-Level Trends**:

The data displayed varying participation rates across grade levels, with certain grades showing higher use of accommodations. This could be due to the varying academic challenges that come with different curricula or developmental stages. It's important for educators and policymakers to recognize these patterns to tailor their support strategies effectively.

**4. Subject-Specific Participation**:

The participation rates in reading and mathematics assessments showed that accommodations were utilized differently across subjects. This might reflect the nature of the assessments or the specific challenges faced by students with disabilities in those subjects.

**5. Impact of COVID-19**:

When considering the ongoing impact of the COVID pandemic, it is crucial to understand that these patterns may also be influenced by the disruptions in traditional learning environments. The pandemic has necessitated shifts in educational approaches, which could affect how accommodations are provided and utilized.

In conclusion, this EDA analysis has highlighted the importance of accommodations in supporting students with disabilities and underscored the need for equitable access to these accommodations. The findings suggest that while accommodations are widely used, there may be opportunities to improve their implementation across districts and grade levels. As schools continue to adapt to the aftermath of the pandemic, these insights could guide efforts to enhance the educational support system for students with disabilities, ensuring that all students have the opportunity to achieve their full academic potential.