

Machine Learning

Statistics Review

1. Mean (Average, μ):

- Definition: The sum of all values divided by the number of values.
- Formula: $\frac{\text{Sum of all values}}{\text{number of values}}$
- Example: For the dataset {2, 5, 8, 10}, the mean is calculated as $\frac{(2+5+8+10)}{4} = \frac{25}{4} = 6.25$.

2. Median:

- Definition: The middle value of a dataset when arranged in ascending or descending order.
- Formula: $\frac{n+1}{2}$
- Example: For the dataset {3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5}, first, sort the values in ascending order: {1, 1, 2, 3, 3, 4, 5, 5, 5, 6, 9}. the median is 4, as it is the middle value. $\frac{11+1}{2} = 6 = 6^{\text{th}}$ value represents 4.

3. Mode:

- Definition: The value that occurs most frequently in a dataset.
- Example: For the dataset {3, 1, 4, 1, 5, 9, 2, 6, 5, 3, 5}, the mode is 5, as it appears three times, more frequently than any other value. (5 * 3)

Frequency Tables:

- Definition: Tables that show the distribution of values and their frequencies in a dataset.
- Example:

Value (Can be like age interval)	Frequencies (number of occurrences)
10-19	1
20-29	2
30-39	48
40-49	158
50-59	236
60-69	262
70-79	174
80-89	50
90-99	3

Relative Frequency (Percentages):

- Definition: Relative frequency means the **number of times** a value appears in the **data** compared to the **total amount**. A **percentage** is a relative frequency.
- Example: For the dataset {2, 3, 3, 5, 5, 5}, the relative frequency of 3 is $\left(\frac{2}{6}\right) \times 100\% = 33.33\%$. **2** Represents number of occurrences and **6** total number of values within a given list.
- $Relative\ Frequency\ (\%) = \left(\frac{Occurrence}{Number\ of\ values\ in\ a\ list}\right) \times 100$
- In this example: **Total number** of winners = **934** and **frequencies** values were given in the previous list. So, we use the formula of dividing the **occurrence** by **934**.

Age Interval (Values)	Relative Frequency (%)
10-19	0.11%
20-29	0.21%
30-39	5.14%
40-49	16.92%
50-59	25.27%
60-69	28.05%
70-79	18.63%
80-89	5.35%
90-99	0.32%

Cumulative Frequency:

- Definition: Cumulative frequency counts up to a particular value.
- Example: Here are the cumulative frequencies of ages of Nobel Prize winners. Now, we can see how many winners have been younger than a certain age.

Age	Cumulative Frequency
Younger than 20	1
Younger than 30	3
Younger than 40	51
Younger than 50	209
Younger than 60	445
Younger than 70	707
Younger than 80	881
Younger than 90	931
Younger than 100	934

Percentiles:

Data = {15, 16, **18**, 19, **22**, 24, **29**, 30, 34}

Median (50th) = 22

25th percentile, since we have 10 values we can use $10 * 25 \div 100 = 2.5$ round that up to 3 because we have a .5 meaning it's the 3rd value in this case **18** What about **75th percentile**?

again use the formula $\frac{number\ of\ values\ in\ a\ list \cdot percentile}{100}$ in this case we end up with an **8** so that is **29** so half of the data is between **18** and **29**.

Standard Deviation & Variance

The **standard deviation** (σ or Σ) and **variance** are measures of how dispersed or spread out the data is.

We measure how far each datapoint is from the mean.

Let's look at our group of ages again:

{15, 16, 18, 19, 22, 24, 29, 30, 34}

Mean = 23

Let's calculate how far each value is from the mean. 15 is 8 away from the mean (since $23 - 15 = 8$). Each value - μ

Here's a list of all these distances:

{8, 7, 5, 4, 1, 1, 6, 7, 11}

We square these values and add them together. $\sigma^2 =$

$$\begin{aligned} &8^2 + 7^2 + 5^2 + 4^2 + 1^2 + 1^2 + 6^2 + 7^2 + 11^2 \\ &= 64 + 49 + 25 + 16 + 1 + 1 + 36 + 49 + 121 \\ &= 362 \end{aligned}$$

We divide this value by the total number of values and that gives us the **variance**.

$$\frac{\text{sum of all values squared}}{\text{total number of values}} = 362 / 9 = 40.22 \# \text{ or } \frac{362}{9}$$

To get the **standard deviation**, we just take the square root of **this number** and get: 6.34

$$\Sigma = 6.34$$

But ... there is a small change with Sample Data.

Our **example** has been for a **Population** (the 9 ages are the only ages we are interested in).

But if the data is a **Sample** (a selection taken from a bigger **Population**, say 15 instead of 9), then the calculation changes!

When you have "N" data values that are:

The Population: divide by N when calculating Variance (like we did).

A Sample: divide by N-1 when calculating Variance.

A **population** is the entire group that you want to draw conclusions about. A **sample** is the specific group that you will collect data from. The size of the **sample** is always less than the total size of the **population**. In research, a population doesn't always refer to people.

Python Implementation (Pt. 1)

```
statisticsreview.py > ...
1  import numpy as np
2
3  list = [2, 2, 3, 3, 8, 12, 16, 18, 20]
4
5  print(list)
6  print("Mean:", round(np.mean(list)))
7  print("Median:", round(np.median(list)))
8
9  def mode(lst):
10     count_dict = {}
11     max_count = 0
12     mode_value = None
13     mode_count = 0
14     modes = []
15
16     for value in lst:
17         count_dict[value] = count_dict.get(value, 0) + 1
18
19         if count_dict[value] > max_count:
20             max_count = count_dict[value]
21             mode_value = value
22             mode_count = 1
23             modes = [value] # Start a new list for the new mode
24
25         elif count_dict[value] == max_count:
26             mode_count += 1
27             modes.append(value) # Add the value to the list of modes
28
29     if mode_count == 1:
30         return "Unimodal", modes
31     elif mode_count == 2:
32         return "Bimodal", modes
33     elif mode_count == 3:
34         return "Trimodal", modes
35     else:
36         return f"{mode_count}-modal", modes
37
38     print("Mode:", mode(list))
39     print("Percentile (25%):", round(np.percentile(list, 25)))
40     print("Percentile (75%):", round(np.percentile(list, 75)))
41
42     print("Variance:", round(np.var(list)))
43     print("Standard Deviation:", round(np.std(list)))
44
45
```

Output:

[2, 2, 3, 3, 8, 12, 16, 18, 20]

Mean: 9

Median: 8

Mode: ('Bimodal', [2, 3])

Percentile (25%): 3

Percentile (75%): 16

Variance: 48

Standard Deviation: 7

Pandas

Python, a very infamous programming language, comes with handy libraries that can be used for Machine Learning.

What's cool about pandas is that you can take in data and view it as a table that's human readable, but it can also be interpreted numerically so that you can do lots of computations with it.

We call the table of data a **DataFrame**.

We need to start by importing Pandas. It's standard practice to nickname it **pd** so that it's faster to type later.

```
import pandas as pd
```

We will be working with a dataset of Titanic passengers. For each passenger, we'll have some data on them as well as whether or not they survived the crash.

Our data is stored as **CSV** (comma-separated values) file. The titanic.csv file is below. The first line is the header and then each subsequent line is the data for a single passenger.

```
Survived, Pclass, Sex, Age, Siblings/Spouses, Parents/Children, Fare
0, 3, male, 22.0, 1, 0, 7.25
1, 1, female, 38.0, 1, 0, 71.2833
1, 3, female, 26.0, 0, 0, 7.925
1, 1, female, 35.0, 1, 0, 53.1
```

We're going to pull the data into pandas so we can view it as a DataFrame.

The **read_csv** function takes a file in csv format and converts it to a Pandas DataFrame.

```
df = pd.read_csv('titanic.csv')
```

The **object df** is now our pandas dataframe with the Titanic dataset. Now we can use the **head** method to look at the data.

The **head** method returns the first 5 rows of the DataFrame.

```
print(df.head())
```

Code:

```
import pandas as pd
df = pd.read_csv('https://sololearn.com/uploads/files/titanic.csv')
print(df.head())
```

Summarize the Data

Usually, our data is much too big for us to be able to display it all.

Looking at the first few rows is the first step to understanding our data, but then we want to look at some summary statistics.

In pandas, we can use the **describe** method. It returns a table of statistics about the columns.

```
print(df.describe())
```

We add a line in the code below to force python to display all 6 columns. Without the line, it will abbreviate the results.

```
pandasData.py > ...
1 import pandas as pd
2 pd.options.display.max_columns = 6
3 df = pd.read_csv("M:\Downloads\Titanic.csv")
4 print(df.describe())
```

	Survived	Pclass	Age	Siblings/Spouses	Parents/Children	\
count	4.00	4.000000	4.00	4.00	4.00	4.0
mean	0.75	2.000000	30.25	0.75	0.0	0.0
std	0.50	1.154701	7.50	0.50	0.0	0.0
min	0.00	1.000000	22.00	0.00	0.0	0.0
25%	0.75	1.000000	25.00	0.75	0.0	0.0
50%	1.00	2.000000	30.50	1.00	0.0	0.0
75%	1.00	3.000000	35.75	1.00	0.0	0.0
max	1.00	3.000000	38.00	1.00	0.0	0.0

	Fare
count	4.000000
mean	34.889575
std	32.389078
min	7.250000
25%	7.756250
50%	30.512500
75%	57.645825
max	71.283300

For each column we see a few statistics. Note that it only gives statistics for the numerical columns.

Let's review what each of these statistics means:

Count: This is the number of **rows** that have a value. In our case, every passenger has a value for each of the **columns**, so the value is 4 (the total number of passengers).

Mean: Recall that the mean is the standard average.

Std: This is short for standard deviation. This is a measure of how dispersed the data is.

Min: The smallest value

25%: The 25th percentile

50%: The 50th percentile, also known as the median.

75%: The 75th percentile

Max: The largest value

Selecting Multiple Columns in Pandas DataFrame

Select Multiple Columns in a Pandas Dataframe

Below are the ways by which we can select multiple columns in a Pandas Dataframe:

- Using Basic Method
- Using loc[]
- Using iloc[]

Select Multiple Columns in a Pandas DataFrame using **Basic Method**

Select Name and Qualification columns:

```
1 import pandas as pd
2
3 data = {
4     "Name": ["John", "Peter", "Loner", "Depressed"],
5     "Age": [21, 18, 23, 18],
6     "Address": ["Heaven", "USA", "Hell", "Void"],
7     "Qualifications": ["College Degree", "Graduated from a Bootcamp", "Failed social classes", "Passed University"]
8 }
9
10 df = pd.DataFrame(data)
11
12 print(df[["Name", "Qualifications"]])
13
```

PROBLEMS OUTPUT DEBUG CONSOLE TERMINAL PORTS

```
PS M:\Pygame> & C:/Users/iHunter/AppData/Local/Programs/Python/Python310/python.exe m:/Pygame/geeksforgeeks.py
PS M:\Pygame> & C:/Users/iHunter/AppData/Local/Programs/Python/Python310/python.exe m:/Pygame/geeksforgeeks.py
   Name      Qualifications
0   John      College Degree
1  Peter  Graduated from a Bootcamp
2  Loner      Failed social classes
3 Depressed      Passed University
```

Select Second to fourth column

```
print(df[df.columns[1:4]])
```

`df.columns[1:4]` to obtain the column names at indices 1 to 3 (exclusive).

`df[df.columns[1:4]]` displays the columns at given indices and including their rows.

Select Multiple Columns in a Pandas Dataframe using loc[] function

```
df.loc[1:3, ["Name", "Qualification"]]
```

Another example

```
df.loc[0, :] # is more flexible than basic df.columns method.
```

Output: First column, all rows.

```
# select all rows
```

```
# select first two columns
```

```
df.iloc[:, 0:2]
```

```
# iloc[row slicing, column slicing]
```

```
df.iloc[0:2, 1:3]
```

Reinforcement Learning

Reinforcement learning is a type of **machine learning algorithm** where an **agent** learns to make decisions by **interacting** with its **environment**. The **agent** then receives a **reward** after interacting with its environment. But there are 4 main aspects of this, and these include:

- **Environment (e.g., Chess):** The environment that the agent must interact with.



- **Agent:** in our example, this can be a computer that plays the chess.



- **Action:** Playing the game, like moving the pieces.



- **Reward (e.g., For winning and losing):** Feedback from the environment that describes the agent's actions. Providing positive reinforcement for desirable actions (winning), and negative reinforcement for undesirable actions (losing)

Supervised vs Unsupervised Learning

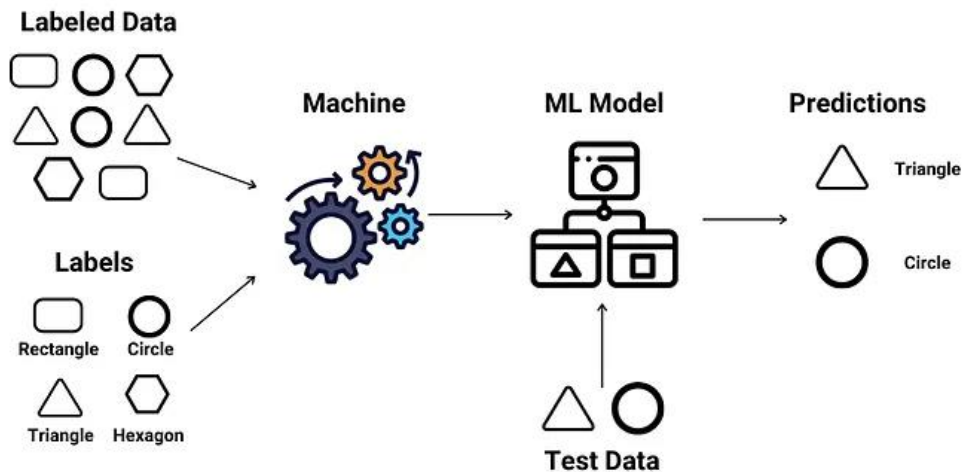
Supervised Learning:

Machine Learning Algorithm makes the prediction based on **labelled data**. Like if we tell it, that these are pictures of **shapes (Annotation)**, it will predict the unseen pictures before by recognizing its patterns and tries to make a prediction based on that. It's called supervise, because there is an **intervention**.

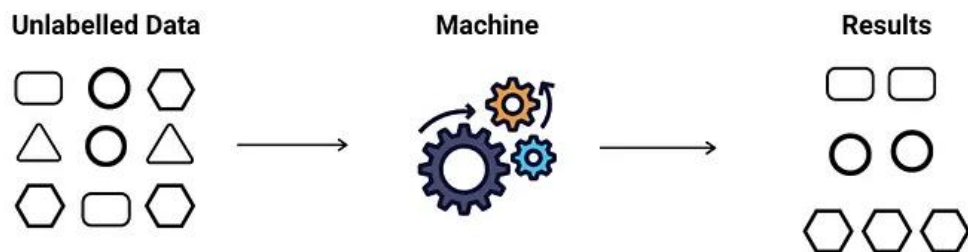
Unsupervised Learning:

Machine Learning Algorithm divides the data into categories since they **aren't labelled**, it's called unsupervised, because the **machine learning algorithm** tries to figure it out on its own.

Supervised Learning



Unsupervised Learning



Types Of Supervised Learning

Classification	Regression
Classification is about predicting a class or discrete values e.g.: <i>Male or Female; True or False.</i>	Regression is about predicting a quantity or continues values e.g.: Price; Salary; Age.

Classification:

Let's say we have an image of a dog and a cat, and we want our machine learning model to predict those images, it will “**classify**” them by giving the output as “**dog**” or “**cat**”. No integer or any other numerical values are displayed.

Regression:

A specific value, let's say I asked you “What's the temperature today?” You would probably give a **numerical** answer, a **value**. That is why it's called “Regression”.

Algorithms

Classification:

1. Decision Tree Classification
2. Random Forest Classification
3. K-nearest Neighbor

Regression:

1. Logistic Regression
2. Polynomial Regression
3. Support Vector Machines

Types Of Unsupervised Learning
