Numpy

1. array( )
2. average( )
3. str.replace(array, delimiter, to_replace)

a = [1, 2, 3, 4, 5]
b = [100, 200, 300, 400, 500]

4. np.column_stack([a, b])
   → [[a0, b0], [a1, b1] ...]

5. np.where(a, a, true_val, false_val) → (5 × 2)
   - value if condition is true
   - value if condition is false
   - list of all the index where the condition is satisfied.

6. np.set_printoptions(suppress = true, precision = true)
   - pretty print.

7. np.sort(arr) / np.argsort(arr)
   - returns sorted array.
   - returns indices according to the sorted array.

8. Matrix multiplication →
   - np.dot(a, b) → more flexible.
   - a @ b
   - np.matmul(a, b) → can operate on scalar values but avoid it for confusion.

reshape test →

a = [[1, 2, 3], [4, 5, 6]]

a.reshape(6,) → vector
a.reshape(6, 1) / a.reshape(6,) → 6 × 1 matrix.

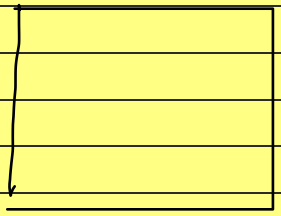
9) `np.vectorize(python fn)` → Vectorizes a normal python fn.
- now the fn can take np.array as input & operate on the elements of the array individually.
(i.e without a for loop. (classic np style).

10) `np.tile(a, x)`

11) `np.split(a, x)` (x number / list of indexes.

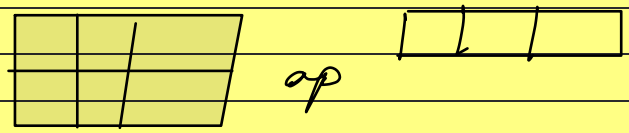
3/6/25 Lecture-2-notes

1) Sorting →



2) Broadcasting → Expanding smaller array b/w two arrays to perform operations b/w them smoothly.

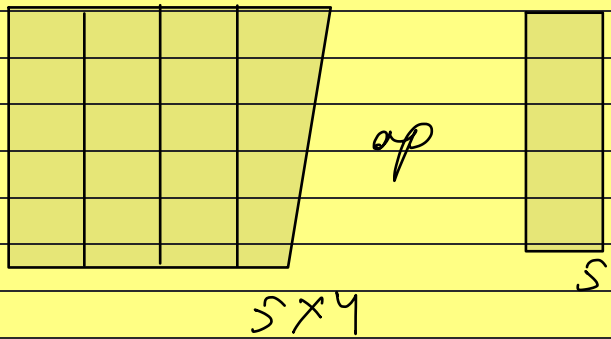
a) (2 x 3) op (1 x 3) op



✓

Row broadcasting.

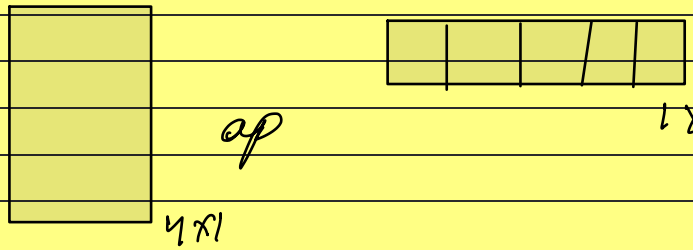
b)



✓

one dimension of the broadcasted vector should be 1.

c)



→ 4x5

Splitting →

np.split(array, axis = 0/1)

- 0 → vsplit → splits rows into equal sections.
- 1 → hsplit → splits columns into equal sections.

1	2	3	4
5	6	7	8

vsplit ↓ (line between rows)

hsplit → (line between columns)

polymorphism → np.split(arr, [list of indices])

↳ splits the array according to the indices provided.

Stacking →

np.hstack((...)), np.vstack((...))

Input → List of arrays.

output →

- hstack: If (# of rows are ==) → array (rows, col₁ + col₂), else error
- vstack: If (# of cols ==) → array (r₁ + r₂, col), else error.

Indexing →

$$x = \begin{bmatrix} 1, & 2, & 3 \\ 4, & 5, & 6 \\ 7, & 8, & 9 \end{bmatrix}$$

(3×3)

x[: , 0] → (3,) (row vector)

[1, 4, 7]

x[: , [0]] → (3, 1) (col vector)

[[1],
[4],
[7]]

D-3 Pandas →

1) df.info() ← 0 is default & Pandas will get confused.
2) df.drop(columns, axis=1, inplace=True) ← False is default.
3) unique() / nunique()
4) df[col].value_counts()
5) Indexing → Implicit & Explicit.
6) df.duplicated()
7) df.drop_duplicate(keep='first')

---

1) column.unique() → Returns a list of unique values in a column.
2) column.nunique() → Returns an integer, # of unique values in the column.
3) column.value_counts() → Returns a dictionary of unique values & their corresponding counts.

Indexing →

1) Implicit index → Default index of integers.
2) Explicit index → Custom index assigned by programmer, usually values of a different column.

tools for indexing →

.iloc[ ] → For implicit.
.loc[ ] → Explicit.

Handling Duplicates →

1) df.duplicated(keep = first/last/False) → Returns true for duplicate rows.

24 df.drop_duplicate(keep, subset = [cols])

Drops rows based on duplication in a specific column.
→ Drops duplicate rows & returns a new df.

In pandas → I0/nan = nan. ← not a number.

D-4

Amazon

1) `Pd.merge(df1, df2, on= , how= )`
   - join df1 & df2.
   - The column to join on.
   - inner/outer/left/right.

2) `df.groupby(col/[cols..])` groups dataframe based on common values in column/columns.

Ex: `grouped_df = df.groupby(col)`

`gdf!`

`gdf.apply(fn)` → applies the fn to the entire group.

`gdf.agg({'col':[aggregate fns]})` → applies a list of aggregate fns to the columns provided in the dictionary as keys.

Advanced groupby →

1) groupby returns an iterable of tuples containing group & key
df & value.

Ex: `for group, frame in df.groupby('category'):`
`# do some operation.`

2) Passing a fn to groupby →

`df.set_index(col)`
`for grp, frame in df.groupby(custom_fn)`

operates on index if col is not provided.

groupby is a two step process
1) Split the data to groups.
2) Apply something to each group.

Split the data frame

- Splitting by default is done on index.
  - for multi index → groupby(levels = (0,1))
    - Index of each column.

Processing on the split dataframe

- Three data processing categories on a groupby object
  1. Aggregation → gdf.agg(dictionary) → Returns a single #.
  2. Transformation → gdf[col].transform(fn/list of fn) → Returns a dataframe, same size as the original dataframe.
  3. Filtering

Scales

- Different types of data
  1. Nominal (categorical) → Data is split into categories with no particular order between the categories. ex: Nationality.
  2. Ordinal → Data is split into categories but there is order between the categories.
     ex: Academic grades.

3) **Interval** → units in the data are equally spaced but there is no true zero.

ex: Temperature, compass. 0th unit has a non-null value.

4) **Ratio Scale** → - units are equally spaced.
- There is a true zero.

ex: Height, weight, length, time.

Scale	operations
1) Nominal	—
2) Ordinal	</> (greater than / less than)
3) Interval	+/−
4) Ratio	+/−/×/÷

→ Pd.CategoricalDTypes([list of categories], ordered = True)

Pd.melt(id_vars = [ ], value_vars = [ ], var_name = , value_name)

id_vars → Cols for the wide format
value_vars → Cols for the long format
var_name → name of new column variable.
value_name → name of value col name.

- Converts a table from wide format to long.
  - Suppresses two or more columns to a single column.
- Creates two columns
  1) To hold column names.
  2) To hold corresponding values.

D-5

// pd.cut(col, bins=[ ], labels=[ ])

Continuous variable → Boundary values to split col to categories.

labels → what to name each category.

2// df.isna() / isnull() → Returns a boolean mask.

True → value missing.
False → not missing value.

Visualization →

// plt.axvline(val, color, linestyle) → Prints a vertical line on figure at position = val.

Univariate data →

Data Type

Continuous → histogram

Discrete → bargraph.

Q→ Plot average ratings by month-year.

Bivariate Data →

X	Y	Plot-type
Continuous	vs Continuous	Scatter

Day - 6

1) plt.grid (visible, color, linestyle, linewidth)

— To plot grid lines in the figure.

Sachin Tendulkar's Career Analysis :-

1) df.info().

2) Check if all the columns are in the right format.

Probability Theory :-

Intersection :- Inner join.

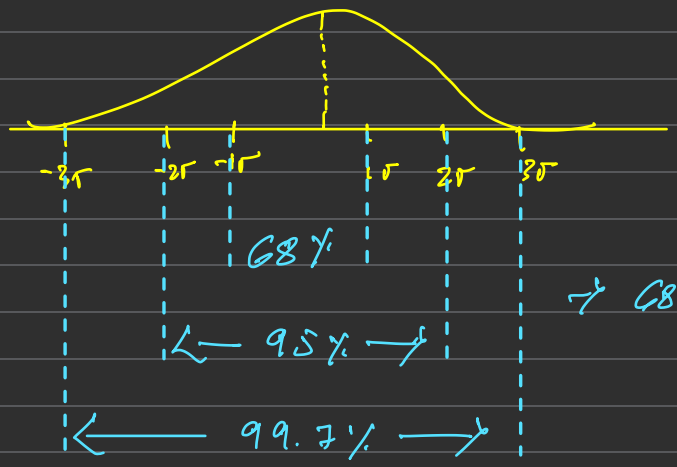
union :- concat() + drop_duplicates()

Statistics :-

Measure of Central Tendency	Measure of spread
1) Mean (μ)	1) Range
2) Median (m)	2) Variance
3) Q_1, Q_2.	3) Standard Deviation.

Properties of normal distribution :-

$$f(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

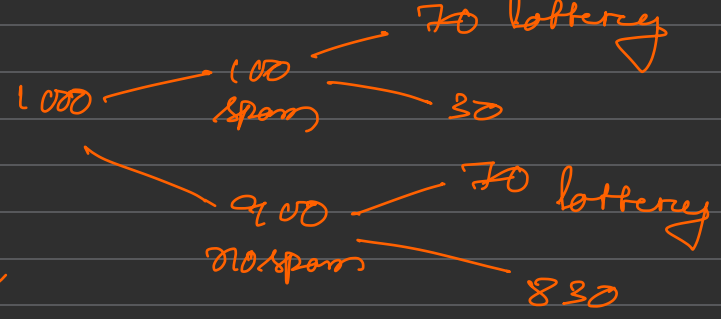


→ 68-95-97 rule.

Q:

$P(\text{lottery}) = 0.14$

$P(\text{lottery} \mid \text{spam}) = 0.7$



$$P(\text{spam} \mid \text{lottery}) = \frac{P(\text{lottery} \mid \text{spam})\, P(\text{spam})}{P(\text{lottery})}$$

$$= 0.7 \times 0.1 / 0.14 = 0.5$$

Bayes Theorem :-

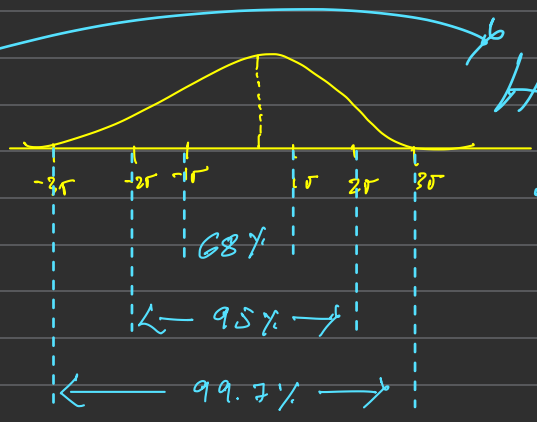
$$P(a/b) = \frac{P(b/a)\,P(a)}{P(b)}$$

Posterior ← Joint probability. ← Prior ← Class probability.

Population & Sample :-

Sampling Distribution :- (Distribution of sample means).

Z-Score :- How much std. deviation the data point is away from the mean.



$Z = P(x \leq X) \rightarrow \frac{|x-\mu|}{\sigma/\sqrt{3}}$ → Z score → Consult Z table for $P(x \leq X)$.

→ Standard Error.

Q-:- What should be the size of sample?

Central Limit Theorem :-

1) For any distribution, the sampling distribution of sample means has a normal distribution.

2) Sample size α fit of the sample dist of sample means.

Standard Error :- It is the measure of how much away the mean of the sampling dist of sample means is from the true population mean.

$$S.E = \sigma/\sqrt{\text{sample size}}$$

or can be considered standard deviation of sample dist.

$S.E \propto 1/\text{sample size}$ are measure of how well the sampling dist of sample means fits the population distribution.

Confidence Interval → Converting from % of area to data-point values through z-score.

↳ An estimated range that could include the true population statistic (mean) with a certain probability.

- Relation with hypothesis testing →

- using confidence interval we can get a range within which the true population parameter must be present with certain confidence.

Ex → medical_data →

import scipy.stats as stats ↳ Confidence interval.

stats.norm.ppf((1+0.9)/2)

↳ fn to return the z-critical value.

lower_limit = mean - (std.error × z-critical)

if sample mean between lower & upper

fail to reject h_0.

else

reject h_0

Probability → Binomial distribution →

from scipy.stats import binom

binom(n, k, p)
↳ success probability.

Poisson → Rate of occurance of an event is given.
→ To find the p(# of occurance) of an event within an interval.

$$P(x = k) = \frac{\lambda^k (e)^{-\lambda}}{k}$$

p(# of occurances of an event) given the rate of occurance of event.

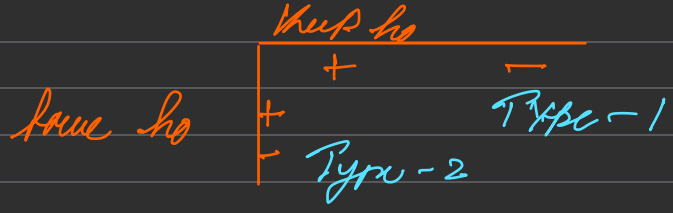
Q-Q plot → To check if the data is normally distributed or not.

Hypothesis Testing

null → no change.
alternate → claimed change.

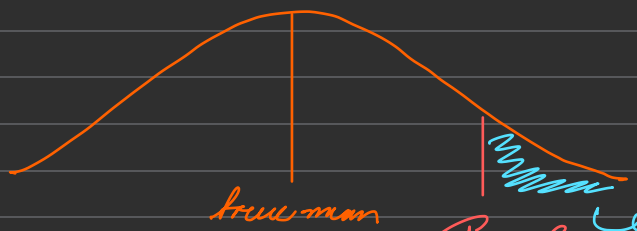
P(alternate / null) → Conditional probability.

↳ low → reject null hypothesis.



Ex: Mean

Weight ≤ 200g → alternate
Weight > 200g → null



P-value = 1 − P(Z ≤ alone)

P(alternate / null)

P-value	Conclusion
P(alternate/null) high	occurrence of alternate hypothesis is common. − Keep h_0. − alternate had less contribution.
low	occurrence of alternate is more w.r.t null. − reject h_0. − Null had a lot of contribution to this rarity.

14.14

$z = 10/14.14 = 7.07$ / 2.23

$P(\text{alt}|\text{null}) = 0.012 \qquad \alpha = 0.01$

– keep H_0.

Q7 $\quad \mu = 65$, $\sigma = 2.5$ | $n = 20$, $\bar{x} = 64.5$ $\quad$ 0.05

$$z = \frac{64.5 - 65}{2.5/\sqrt{20}} = -0.89$$

$P(z) = 0.186 \qquad 0.05$

Two sample test ↓

$$z = \frac{\mu_1 - \mu_2}{\sqrt{\sigma_1^2/n_1 - \sigma_2^2/n_2}}$$

μ_1 ↓ mean
σ_1 ↓ std. dev
n_1 ↓ sample size
} sample 1

μ_2 ↓ mean
σ_2 ↓ std. dev
n_2 ↓ sample size
} sample 2

Proportion Test ↓

$$z = \frac{p - \bar{p}}{\sqrt{P(1-P)/n}}$$

z score for proportions. } ↓ standard error.

Two sample ↓

$$z = \frac{\hat{p}_1 - \hat{p}_2}{\sqrt{\hat{p}(1-\hat{p})(1/n_1 + 1/n_2)}}$$

$\hat{p}_1$ ↓ old proportion. $\quad \hat{p}_2$ ↓ new proportion.

Combined proportion.

T-test

- Assumptions -

1) Sample size ≤ 30.

- although can be used for larger sample size.

2) σ is unknown.

↳ Population std. deviation.

Paired T Test → Dependent

ttest_rel(df[group1], df[group2])

import ttest_rel from teststats

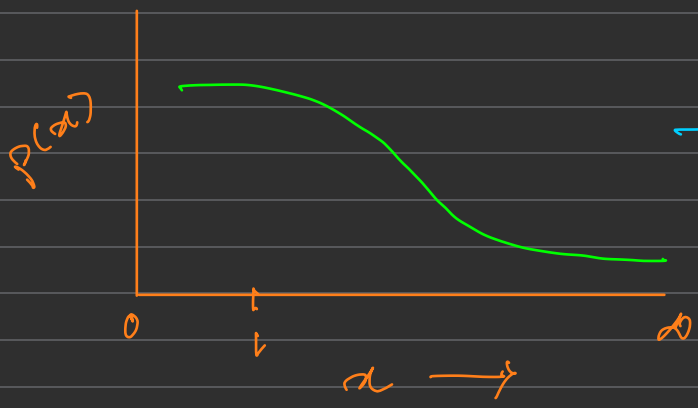
χ^2 - test → Check if the actual values differ significantly from expected values.

	H	T	
Coin toss	25	25	expected
	28	22	actual

- no std. deviation.
- uses frequency.

H_0 : Coin is fair
H_1 : coin not fair.

$$\chi^2 = \sum \frac{(\text{actual}_i - \text{expected}_i)^2}{\text{expected}_i}$$



- right skewed.

→ formula implementation.

from scipy.stats import chisquare, chi2 → pdf & cdf.

chi_stat, p_value = chisquare(actual, expected)

p_val = 1 - chi2.cdf(chi_stat , df = 1)

- Compute Expected values using marginal values.

χ^2 - two category → If one category affects the other.

Ex-4

	M	W		M	W
offline	527	72	599 (66%)	484 (66% of 733)	115 (66% of 184)
online	206	102	308 (34%)	269	59
	733	174	907		
	Actual			Expected	

```
from scipy.stats import chi2_contingency
chi_stat, p_val, dof, exp = chi2_contingency(actual_values)
                                  # expected frequency
                                  # degree of freedom
```

Co-relation ⇒

X, Y ⇒ one/two numerical categories.
— used for numbers.

χ^2 ⇒ one/two categorical categories.
(non-numerical)

Correlation ⇒ Two numeric categories.
— how two values are related to each other.

Pearson ⇒ x_i, y_i

$$cov(x,y) = \frac{\sum (x_i - \bar{x})(y_i - \bar{y})}{n}$$

Correlation ⇒ $corr(x,y) = \frac{cov(x,y)}{\sigma(x,y)}$

Hypothesis Testing Revision

Z-test → To test if data from samples differ significantly from a prior belief/claim.

↳ single-group

1) one-sample → To test if sample measurements differ significantly from a known population measurement.

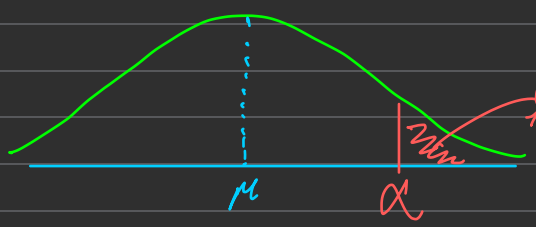
$$Z = \frac{\bar{x} - \mu}{\sigma/\sqrt{n}}$$ ← standard-error.

2) Two-Sample → To test if the mean of two groups for a test statistic differ significantly or not.

$$Z = \frac{\mu_1 - \mu_2}{\sqrt{\sigma_1^2/n_1 + \sigma_2^2/n_2}}$$

Null & Alternate Hypothesis →

null → (H_0) → A natural belief about data.
alternate → (H_{alt}) → A claim on the data.



Z-Score → # of std. deviations away from the mean.

Z-score	P-val (P(alternate/null))	reject H_0
high	low	Yes, as the observed data is too far away from expected data.
low	high	No, observed data is close to expected data.

2) **T-test** → Similar to Z-test but uses sample statistics instead of population.

use when

1) Population std. deviation is unknown.
2) Sample size is comparatively small. ($n \leq 30$)
3) Samples are independent (for two sample).

a) **one-sample** → To check if the sample mean for a measurement is similar to population mean.

- population std. dev (σ) is unknown.

$$t\text{-score} = \frac{\bar{x} - \mu}{s/\sqrt{n}}$$

s → sample std. error.
s → Sample std. deviation.

b) **Two-Sample** → To check if the mean for a given statistic/measurement is similar b/w two groups.

- σ is unknown for both the groups.
- two groups are independent.

$$t\text{-score} = \frac{\mu_1 - \mu_2}{\sqrt{s_1^2/n_1 + s_2^2/n_2}}$$

s → Sample std. error.

3) **Proportions test** → To check if the ratios for a binary-measurement are similar for sample & population.

requirements →
1) Binary variable.
2) Independent-observations.
3) **Sample-Size adequacy** → 1) one sample → $n(P_0) \geq 10$ && $n(1-P_0) \geq 10$
2) Two Sample → $n_1 P_1 \geq 10$ && $n_1(1-P_1) \geq 10$
$n_2 P_2 \geq 10$ && $n_2(1-P_2) \geq 10$

P → Population proportion.

a) **one-Sample z** → To check if the proportions in a binary measurement is similar to claimed proportion or not.

$$z\text{-score} = \frac{\hat{p} - p_0}{\sqrt{p_0(1-p_0)/n}}$$

$\hat{p}$ → Sample proportion. p_0 → Population proportion. n → Sample size.

b) **Two-Sample z**

$$p_1 = x_1/n_1$$

$$p_2 = x_2/n_2$$

$$p = \frac{x_1 + x_2}{n_1 + n_2}$$

$$\frac{p_1 - p_2}{\sqrt{p(1-p)\left(1/n_1 + 1/n_2\right)}}$$

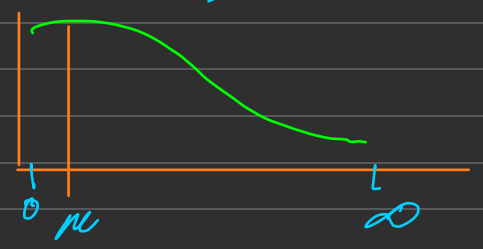
(p → Combined proportion.)

c) **χ^2 test** → (Chi square)

$$\chi^2 = \sum \frac{(\text{expected}_i - \text{actual}_i)^2}{\text{expected}_i}$$

χ^2	P-val
low	high
high	low

chi-square dist.



Used for →

1) **goodness of fit →** Check if the freq count of two categorical variables differ significantly or not.

ex → Test if a coin is fair by comparing the observed freqs to natural frequencies.

2) **Test for independence →** To test if two categorical variables are independent.

ex → Test if political views were related to demographics or genders. → Categorical variable.

5) Anova → Analysis of variance.

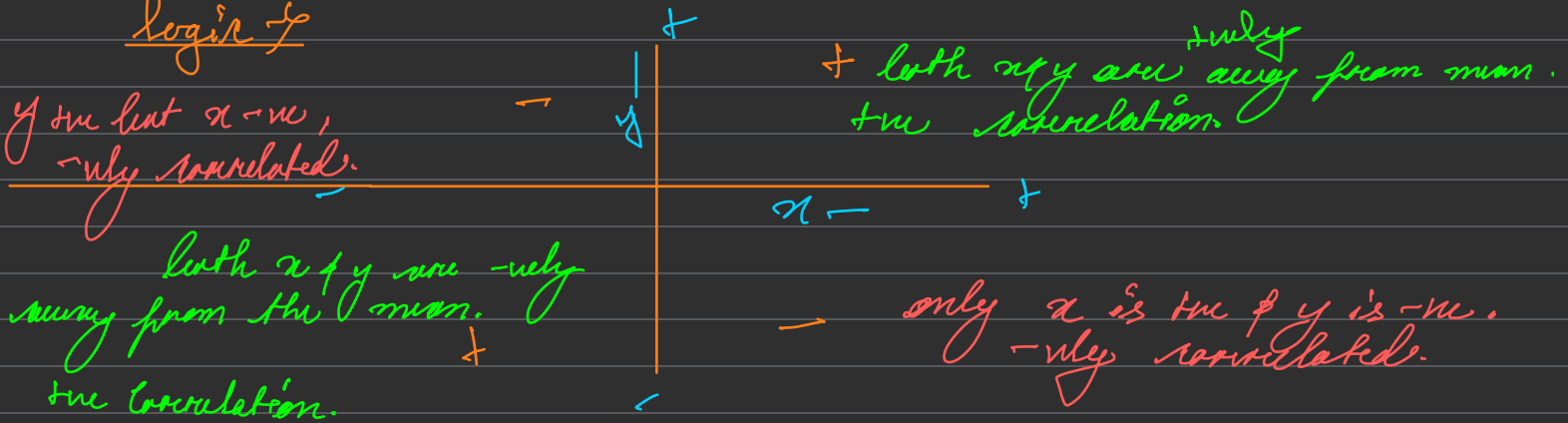
To check if the mean of multiple categorical variables differ with one another or not.

Mobile	fruit
apple	x
samsung	y
oppo	z

→ To check if the means of the three fruit categories vary with eachother are not.

6) Correlation →

Login →



y increases, x → inc, +vely correlated.

+ both x,y are +vely away from mean. → +ve correlation.

both x & y are -vely away from the mean. → +ve correlation.

only x is inc & y is -ve. → -vely correlated.

$$\text{corr}(x,y) = \frac{\text{cov}(x,y)}{\sigma_x \sigma_y}$$

$\sigma_{x,y}$ → joint std. dev.

$$\text{cov}(x,y) = \frac{\sum (x_i - \bar{x}) + (y_i - \bar{y})}{n-1}$$

Variable
- Numerical
  - Two Categories
    - $n \geq 30$ & known → z-test
    - σ & unknown → T-test
  - $\geq$ Two Categories → Anova
- Categorical
  - Two Category → χ^2-test.
  - $\geq$ Two Category → ?