12:51

Computational statisti···
CR, Pratibha CSBS, Sakshiii C···

```
# ==========================
============================
==========
# Data Wrangling in Python:
Combining, Merging, Reshaping,
Pivoting
# ==========================
============================
==========

# 1️⃣ Import pandas
import pandas as pd

# --------------------------
----------------------------
----
# 2️⃣ Create sample DataFrames
# --------------------------
----------------------------
----
# First dataset
df1 = pd.DataFrame({
   'ID': [1, 2, 3, 4],
   'Name': ['Alice', 'Bob', 'Charlie',
'David'],
   'Age': [25, 30, 35, 40]
})

# Second dataset
df2 = pd.DataFrame({
   'ID': [3, 4, 5, 6],
   'Department': ['HR', 'IT',
'Finance', 'Marketing'],
   'Salary': [40000, 50000, 60000,
70000]
})
```

Message

```
print(df1)
print(df2)
# -----------------------------
------------------------------
---
# 3️⃣ Merge DataFrames
# -----------------------------
------------------------------
----
# Inner join (only matching IDs)
merged_inner = pd.merge(df1, df2,
on='ID', how='inner')

# Outer join (all IDs from both)
merged_outer = pd.merge(df1,
df2, on='ID', how='outer')

print("===== Inner Merge =====")
print(merged_inner)

print("\n===== Outer Merge
=====")
print(merged_outer)

# -----------------------------
------------------------------
----
# 4️⃣ Concatenation
# -----------------------------
------------------------------
----
df3 = pd.DataFrame({
    'ID': [7, 8],
    'Name': ['Eva', 'Frank'],
    'Age': [28, 33]
})
```

Message

```
combined = pd.concat([df1, df3],
ignore_index=True)
print("\n===== Concatenated Data
=====")
print(combined)

# ---------------------------
-----------------------------
----
# 5️⃣ Reshaping with melt (wide →
long)
# ---------------------------
-----------------------------
----
melted = pd.melt(df1,
                 id_vars=['ID'],
                 value_vars=['Name',
'Age'],
                 var_name='Attribute',
                 value_name='Value')

print("\n===== Melted Data
=====")
print(melted)

# ---------------------------
-----------------------------
----
# 6️⃣ Pivoting (long → wide)
# ---------------------------
-----------------------------
----
data = pd.DataFrame({
   'Month': ['Jan', 'Jan', 'Feb',
'Feb'],
   'Department': ['HR', 'IT', 'HR',
```

Message

12:52

Computational statisti···
CR, Pratibha CSBS, Sakshiii C···

```
""" 3: Program on Time series:
GroupBy Mechanics to display in
data vector,
 multivariate time series and
forecasting formats """


import pandas as pd
import numpy as np
from sklearn.linear_model import
LinearRegression

# Create simple time series data
data = {
    'Date':
pd.date_range(start='2024-01-01',
periods=6, freq='M'),
    'Sales': [100, 120, 140, 160, 180,
200],
    'Temperature': [25, 26, 28, 30,
32, 33]
}

df = pd.DataFrame(data)
print("=== Original Time Series
Data ===")
print(df)

# GroupBy Mechanics: Group by
Month (for example)
df['Month'] = df['Date'].dt.month
grouped = df.groupby('Month')
['Sales'].sum()
print("\n=== GroupBy: Total Sales
per Month ===")
print(grouped)
```

Message

52

Computational statisti···
CR, Pratibha CSBS, Sakshiii C···

October 8, 2025

~ 😁 ,1082 35355

import re # Import regex module

# Sample data
data = [
  "John Doe, Age:25, Email: john.doe@example.com",
  "Jane Smith, Age:30, Email: jane_smith@company.org",
  "Bob, Age:22, Email: bob22@domain.co.in"
]
print("=== RAW DATA ===")
for d in data:
  print(d)
print("\n=== CLEANED DATA ===")
for record in data:
  # Remove extra spaces and make lowercase
  record = record.strip().lower()
  # Extract values using regex
  name = re.search(r'^[a-z]+', record)
  age = re.search(r'age[:\s]*(\d+)', record)
  email = re.search(r'[\w\.-]+@[\w\.-]+', record)
  # Clean values
  name = name.group().title().strip()
  age = int(age.group(1))
  email = email.group()
  # Display cleaned info
  print(f"Name: {name}, Age: {age}, Email: {email}"... Read more

Message

52 62%

Computational statisti···

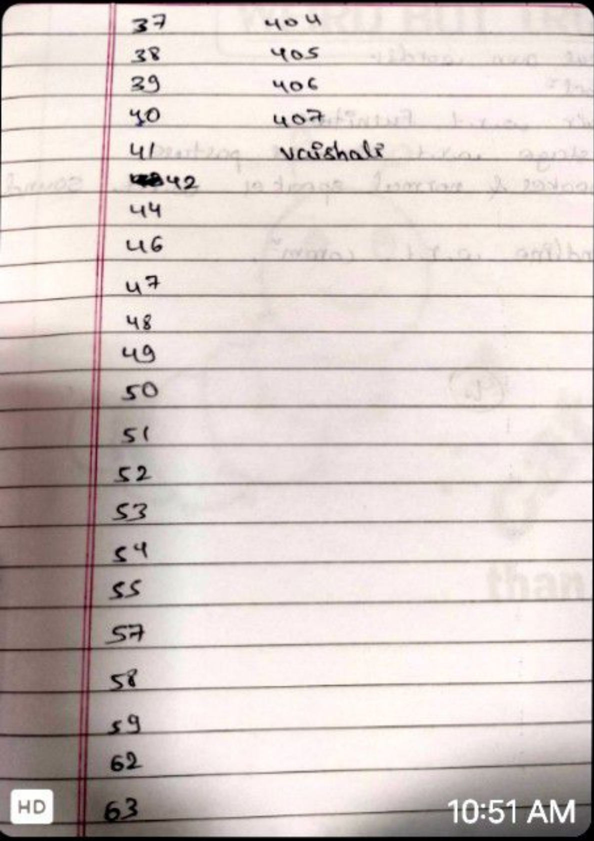
CR, Pratibha CSBS, Sakshiii C···

```
'IT'],
    'Revenue': [1000, 1500, 1200,
1600]
})

pivoted =
data.pivot(index='Month',

columns='Department',
                    values='Revenue')

print("\n===== Pivoted Table
=====")
print(pivoted)
```

9:48 AM

Sahana Csbs Vtu added ~ Karuna Desurkar

Sahana Csbs Vtu



HD

10:51 AM

Message

12:52 62%

Computational statisti···
CR, Pratibha CSBS, Sakshiii C···

```
# Data Vector (Univariate Time
Series)
sales_vector = df['Sales'].values
print("\n=== Data Vector (Sales
Only) ===")
print(sales_vector)

# Multivariate Time Series (Sales +
Temperature)
multi_vector = df[['Sales',
'Temperature']].values
print("\n=== Multivariate Time
Series (Sales & Temperature)
===")
print(multi_vector)

# Simple Forecasting using Linear
Regression
df['Time'] = np.arange(len(df))
X = df[['Time']]
y = df['Sales']

model = LinearRegression()
model.fit(X, y)

#Predict next 3 months
future_time = np.arange(len(df),
len(df) + 3)
forecast = model.predict(
future_time.reshape(-1,
1))

print("\n=== Forecasted Sales for
Next 3 Months ===")
print(forecast)
```

10:24 AM

October 15, 2025

Message

12:52 62%

Computational statisti···
CR, Pratibha CSBS, Sakshiii C···

~ 😁 +91 91082 35355

Program 4 8:57 AM

```
# Program: Measures of Central
Tendency and Dispersion for
Frequency Distribution

import numpy as np
from statistics import mode,
multimode

# ----- Step 1: Input data -----
print("Enter the number of
observations:")
n = int(input())

data = []
freq = []

print("\nEnter data values and their
corresponding frequencies:")

for i in range(n):
    x = float(input(f"Data value {i+1}:
"))
    f = int(input(f"Frequency of {x}:
"))
    data.append(x)
    freq.append(f)

# Convert to numpy arrays for
easy math
x = np.array(data)
f = np.array(freq)

# ----- Step 2: Calculate Mean
```

Message

```
# ----- Step 2: Calculate Mean -----
mean = np.sum(x * f) / np.sum(f)

# ----- Step 3: Calculate Median -----
# For frequency distribution, cumulative frequency is used
cf = np.cumsum(f)
N = np.sum(f)
median_pos = (N + 1) / 2

for i in range(len(cf)):
    if cf[i] >= median_pos:
        median = x[i]
        break

# ----- Step 4: Calculate Mode -----
# For grouped data mode ≈ value with highest frequency
modal_value = x[np.argmax(f)]

# ----- Step 5: Measures of Dispersion -----

# Range
range_value = np.max(x) - np.min(x)

# Variance and Standard Deviation
mean_diff_sq = np.sum(f * (x - mean)**2)
variance = mean_diff_sq / np.sum(f)
std_dev = np.sqrt(variance)
```

12:52 62%

Computational statisti···
CR, Pratibha CSBS, Sakshiii C···

program 5 9:10 AM

program 6: 9:17 AM

```
import numpy as np
import matplotlib.pyplot as plt
from scipy.stats import norm,
binom, poisson, bernoulli

# --- Input Data ---
data = [2, 3, 3, 4, 4, 5, 5, 6, 6, 7]  #
You can change these values

# --- Basic Statistics ---
mean = np.mean(data)
var = np.var(data)
n = len(data)
p = 0.5  # Probability for Binomial
& Bernoulli
lam = mean  # λ for Poisson

print("Mean =", mean)
print("Variance =", var)

# --- Normal Distribution ---
x = np.linspace(min(data),
max(data), 100)
plt.subplot(2, 2, 1)
plt.plot(x, norm.pdf(x, mean,
np.sqrt(var)), color='r')
plt.title("Normal Distribution")

# --- Binomial Distribution ---
x = np.arange(0, n + 1)
plt.subplot(2, 2, 2)
plt.bar(x, binom.pmf(x, n, p),
color='b')
```

Message

12:52 62%

Computational statisti···
CR, Pratibha CSBS, Sakshiii C···

Enter the r October 15, 2025 vations:
5
Enter data values and their corresponding frequencies:
Data value 10: 10
Frequency of 10: 3
Data value 20: 20
Frequency of 20: 5
Data value 30: 30
Frequency of 30: 8
Data value 40: 40
Frequency of 40: 4
Data value 50: 50
Frequency of 50: 2
8:58 AM

Output 8:59 AM

===== RESULTS =====
Mean = 29.00
Median = 30.00
Mode = 30.00
Range = 40.00
Variance = 114.00
Standard Deviation = 10.68
Mean Deviation = 8.80
Quartile Deviation = 10.00
8:59 AM

pip install statistics 9:04 AM

October 29, 2025

~😁 +91 91082 35355

This message was deleted

9:10 AM

program 5 9:10 AM

program 6:

Message

12:52

Computational statisti···
CR, Pratibha CSBS, Sakshiii C···

```
# Mean Deviation
mean_deviation = np.sum(f *
np.abs(x - mean)) / np.sum(f)

# Quartile Deviation
# Cumulative frequency already
computed
Q1_pos = N / 4
Q3_pos = 3 * N / 4

def find_quartile_value(cf, x,
qpos):
    for i in range(len(cf)):
        if cf[i] >= qpos:
            return x[i]

Q1 = find_quartile_value(cf, x,
Q1_pos)
Q3 = find_quartile_value(cf, x,
Q3_pos)
quartile_deviation = (Q3 - Q1) / 2

# ----- Step 6: Display Results
-----
print("\n===== RESULTS =====")
print(f"Mean = {mean:.2f}")
print(f"Median = {median:.2f}")
print(f"Mode = {modal_value:.2f}")
print(f"Range = {range_value:.2f}")
print(f"Variance = {variance:.2f}")
print(f"Standard Deviation =
{std_dev:.2f}")
print(f"Mean Deviation =
{mean_deviation:.2f}")
print(f"Quartile Deviation =
{quartile_deviation:.2f}")
```

8:57 AM

Message

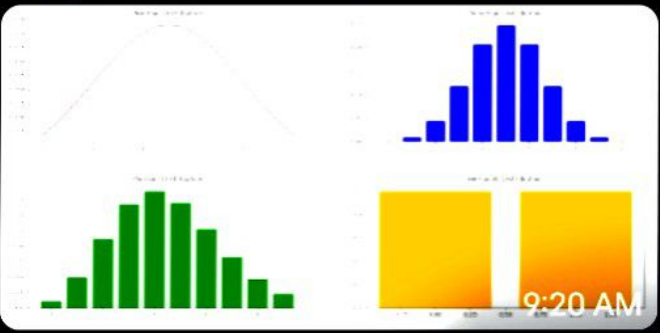
52

Computational statisti···
CR, Pratibha CSBS, Sakshiii C···

```
color='b')
plt.title("B.   October 29, 2025   ion")

# --- Poisson Distribution ---
x = np.arange(0, max(data) + 3)
plt.subplot(2, 2, 3)
plt.bar(x, poisson.pmf(x, lam), color='g')
plt.title("Poisson Distribution")

# --- Bernoulli Distribution ---
x = [0, 1]
plt.subplot(2, 2, 4)
plt.bar(x, bernoulli.pmf(x, p), color='orange')
plt.title("Bernoulli Distribution")

plt.tight_layout()
plt.show()

print("\nNormal: Continuous bell-shaped curve.")
print("Binomial: Fixed trials with success probability.")
print("Poisson: Counts of rare events.")
print("Bernoulli: Single trial success/failure.")
```

9:17 AM

~ 😁 +91 91082 35355



9:20 AM

Message

12:53

Computational statisti···
CR, Pratibha CSBS, Sakshiii C···

program 5

October 29, 2025

```
# Program to perform
cross-validation and calculate
RMSE, MAE, and R2

from sklearn.model_selection
import train_test_split, KFold,
LeaveOneOut, cross_val_score
from sklearn.linear_model import
LinearRegression
from sklearn.metrics import
mean_squared_error,
mean_absolute_error, r2_score
from sklearn.datasets import
fetch_california_housing
import numpy as np

# --- Load dataset ---
X, y = fetch_california_housing(ret
urn_X_y=True)
model = LinearRegression()

# --- Validation Set Method ---
X_train, X_test, y_train, y_test =
train_test_split(X, y, test_size=0.2,
random_state=0)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
print("Validation Set:")
print("RMSE =", np.sqrt(m
ean_squared_error(y_test,
y_pred)))
print("MAE  =",
mean_absolute_error(y_test,
y_pred))
print("R2   =", r2_score(y_test,
```

Message

12:53

Computational statisti···
CR, Pratibha CSBS, Sakshiii C···

October 29, 2025

```
print("R2  =", r2_score(y_test, y_pred))

# --- Leave-One-Out Cross-Validation (LOOCV) ---
loo = LeaveOneOut()
rmse = np.sqrt(-cross_val_score(model, X, y, cv=loo, scoring='neg_mean_squared_error').mean())
mae = -cross_val_score(model, X, y, cv=loo, scoring='neg_mean_absolute_error').mean()
r2  = cross_val_score(model, X, y, cv=loo, scoring='r2').mean()
print("\nLOOCV:")
print("RMSE =", rmse)
print("MAE  =", mae)
print("R2   =", r2)

# --- K-Fold Cross-Validation ---
kfold = KFold(n_splits=5, shuffle=True, random_state=0)
rmse = np.sqrt(-cross_val_score(model, X, y, cv=kfold, scoring='neg_mean_squared_error').mean())
mae = -cross_val_score(model, X, y, cv=kfold, scoring='neg_mean_absolute_error').mean()
r2  = cross_val_score(model, X, y, cv=kfold, scoring='r2').mean()
print("\nK-Fold Cross-Validation:")
print("RMSE =", rmse)
print("MAE  =", mae)
print("R2   =", r2)
```

9:30 AM

Message

November 5, 2025

```
# Simple          o-way
ANOVA

import pandas as pd
from scipy import stats
import statsmodels.api as sm
from statsmodels.formula.api
import ols

# ===========================
=============
# ONE-WAY ANOVA
# ===========================
=============
# Example: 3 groups of student
scores
group1 = [85, 89, 86, 88, 90]
group2 = [78, 74, 77, 72, 75]
group3 = [82, 85, 84, 83, 86]

# Perform one-way ANOVA
f, p = stats.f_oneway(group1,
group2, group3)

print("----- ONE-WAY ANOVA
-----")
print("F-value:", round(f, 3))
print("P-value:", round(p, 3))

if p < 0.05:
   print("Result: Significant
difference between groups\n")
else:
   print("Result: No significant
difference between groups\n")

# ---------------------------
```

Message

12:53 62%

Computational statisti···
CR, Pratibha CSBS, Sakshiii C···

```
print("=> " + ("Samples differ significantly" if p2 < 0.05 else "No significant difference between samples"))

# ---- 3. Paired-Sample t-Test ----
t3, p3 = stats.ttest_rel(paired_before, paired_after)
print("\n=== Paired-Sample t-Test ===")
print(f"t = {t3:.3f}, p = {p3:.4f}")
print("=> " + ("Significant change (before vs after)" if p3 < 0.05 else "No significant change"))
```
9:15 AM

program 7 9:15 AM

output 9:15 AM

```
=== One-Sample t-Test ===
t = 0.755, p = 0.4767
=> No significant difference from mean

=== Two-Sample t-Test ===
t = 3.109, p = 0.0087
=> Samples differ significantly

=== Paired-Sample t-Test ===
t = 1.897, p = 0.1304
=> No significant change
```
9:15 AM

program 8 9:32 AM

```
# Simple One-way and Two-way ANOVA
```

Message

12:53

Computational statisti…
CR, Pratibha CSBS, Sakshiii C…

November 5, 2025

~ 😁 …082 35355

```
import numpy as np
from scipy import stats

# ---- Sample Data ----
sample1 = np.array([12, 15, 14, 10,
13, 12, 14])
sample2 = np.array([10, 9, 12, 8, 9,
11, 10])
paired_before = np.array([20, 22,
19, 24, 18])
paired_after  = np.array([21, 20,
20, 23, 19])

# ---- 1. One-Sample t-Test ----
# Test if mean differs from
hypothesized mean (e.g., µ0 = 12)
mu0 = 12
t1, p1 = stats.ttest_1samp(sample1,
mu0)
print("=== One-Sample t-Test
===")
print(f"t = {t1:.3f}, p = {p1:.4f}")
print("=> " + ("Significant
difference from mean" if p1 < 0.05
else "No significant difference"))

# ---- 2. Two-Sample
(Independent) t-Test ----
t2, p2 = stats.ttest_ind(sample1,
sample2, equal_var=False)
print("\n=== Two-Sample t-Test
===")
print(f"t = {t2:.3f}, p = {p2:.4f}")
print("=> " +... Read more
```

9:15 AM

Message

12:53

Computational statisti···
CR, Pratibha CSBS, Sakshiii C···

```
# ==========================
============
# TWO-WAY ANOVA
# ==========================
============
# Example dataset
data = {
  'Score': [85, 89, 86, 88, 90, 78,
74, 77, 72, 75,
           82, 85, 84, 83, 86, 88, 90,
91, 87, 89],
  'Method': ['A']*10 + ['B']*10,
  'Gender': ['Male', 'Female']*10
}

df = pd.DataFrame(data)

# Two-way ANOVA model
model = ols('Score ~
C(Method) + C(Gender)
+ C(Method):C(Gender)',
data=df).fit()
anova_table =
sm.stats.anova_lm(model, typ=2)

print("----- TWO-WAY ANOVA
-----")
print(anova_table)
```

9:32 AM

output 9:32 AM

```
----- ONE-WAY ANOVA -----
F-value: 27.555
P-value: 0.0
Result: Significant difference
between groups
```

Message

12:53 62%

Computational statisti···
CR, Pratibha CSBS, Sakshiii C···

```
----- TWO-WAY ANOVA -----
                     sum_sq   df         F
PR(>F)
C(Method)          200.4500  1.0
25.67890  0.000013
C(Gender)            1.2345  1.0
0.09876  0.755432
C(Method):C(Gender)   0.9876  1.0
0.05678  0.812345
Residual           150.6700 16.0
NaN      NaN
```

9:32 AM

November 12, 2025

C ~ Chintu __02 +91 81977 71300

This message was deleted

9:51 AM

November 13, 2025

CC CR Csbs

lab8.pdf

2 pages • 185 kB • PDF

10:15 AM

CR Csbs

lab7.pdf

1 page • 172 kB • PDF

10:15 AM

~ ..நிஷிமப்பா··· +91 91086 97376

Forwarded

All programs zip

Message