**GOVERNMENT POLYTECHNIC** **NAGAMANGALA**

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING**

**“V”th Semester Diploma**

**Artificial Intelligence and Machine Learning(20CS51)**

**Assignment:03**

**NAME:NUTHAN.H.K**

**ROLL NO:158CS22032**

AIML (20CS51)

ASSIGNMENT – WEEK 02

1. Download any two datasets from the internet and perform the following operations.

a) Analyze the univariate dataset Ex- Mean, Mode, Median, Range, Std, and Variance and perform Univariate tests for the dataset.

b) Analyze the multivariate of the dataset Ex- co-variance, co-relation.

c) Visualize the univariate and multivariate with various plots.

d) Push the code to your GitHub Repository.

1. Download any two datasets from the internet and perform the following operations.

a) Analyze the univariate dataset Ex- Mean, Mode, Median, Range, Std, and Variance and perform Univariate tests for the dataset.

\*MEAN:

import pandas as pd

path = ('/content/ecommerce\_sales\_analysis.csv')

df = pd.read\_csv(path)

df.mean(numeric\_only=True)

OUTPUT:

| **0** |
| --- |
| **product\_id** | 500.50000 |
| **price** | 247.67713 |
| **review\_score** | 3.02760 |
| **review\_count** | 526.50600 |
| **sales\_month\_1** | 498.30600 |
| **sales\_month\_2** | 507.66100 |
| **sales\_month\_3** | 506.73900 |
| **sales\_month\_4** | 503.82300 |
| **sales\_month\_5** | 487.19400 |
| **sales\_month\_6** | 491.65300 |
| **sales\_month\_7** | 507.01100 |
| **sales\_month\_8** | 504.56900 |
| **sales\_month\_9** | 491.93400 |
| **sales\_month\_10** | 514.79800 |
| **sales\_month\_11** | 505.83800 |
| **sales\_month\_12** | 500.38600 |

**dtype:** float64

**MODE:**

df.mode()

OUTPUT:

| **product\_id** | **product\_name** | **category** | **price** | **review\_score** | **review\_count** | **sales\_month\_1** | **sales\_month\_2** | **sales\_month\_3** | **sales\_month\_4** | **sales\_month\_5** | **sales\_month\_6** | **sales\_month\_7** | **sales\_month\_8** | **sales\_month\_9** | **sales\_month\_10** | **sales\_month\_11** | **sales\_month\_12** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 1 | Product\_1 | Books | 47.25 | 4.5 | 881.0 | 725.0 | 206.0 | 236.0 | 438.0 | 129.0 | 56.0 | 25.0 | 395.0 | 105.0 | 416.0 | 835.0 | 645.0 |
| **1** | 2 | Product\_10 | NaN | 97.35 | NaN | NaN | NaN | NaN | 719.0 | 657.0 | NaN | NaN | 28.0 | NaN | 252.0 | NaN | 844.0 | NaN |
| **2** | 3 | Product\_100 | NaN | 159.30 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 57.0 | NaN | 616.0 | NaN | NaN | NaN |
| **3** | 4 | Product\_1000 | NaN | 248.85 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 68.0 | NaN | 715.0 | NaN | NaN | NaN |
| **4** | 5 | Product\_101 | NaN | 318.53 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | 72.0 | NaN | 782.0 | NaN | NaN | NaN |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **995** | 996 | Product\_995 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **996** | 997 | Product\_996 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **997** | 998 | Product\_997 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **998** | 999 | Product\_998 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **999** | 1000 | Product\_999 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |

1000 rows × 18 columns

MEDIAN:

df.median(numeric\_only=True)

OUTPUT:

| **0** |
| --- |
| **product\_id** | 500.50 |
| **price** | 250.92 |
| **review\_score** | 3.10 |
| **review\_count** | 543.00 |
| **sales\_month\_1** | 507.50 |
| **sales\_month\_2** | 508.00 |
| **sales\_month\_3** | 493.00 |
| **sales\_month\_4** | 501.50 |
| **sales\_month\_5** | 497.00 |
| **sales\_month\_6** | 479.50 |
| **sales\_month\_7** | 522.50 |
| **sales\_month\_8** | 499.50 |
| **sales\_month\_9** | 495.50 |
| **sales\_month\_10** | 532.00 |
| **sales\_month\_11** | 502.00 |
| **sales\_month\_12** | 500.50 |

**dtype:** float64

**STANDARD DEVIATION:**

df.std(numeric\_only=True)

OUTPUT:

| **0** |
| --- |
| **product\_id** | 288.819436 |
| **price** | 144.607983 |
| **review\_score** | 1.171243 |
| **review\_count** | 282.269932 |
| **sales\_month\_1** | 289.941478 |
| **sales\_month\_2** | 285.992689 |
| **sales\_month\_3** | 294.010873 |
| **sales\_month\_4** | 286.645567 |
| **sales\_month\_5** | 287.844324 |
| **sales\_month\_6** | 289.234018 |
| **sales\_month\_7** | 291.047287 |
| **sales\_month\_8** | 289.945691 |
| **sales\_month\_9** | 287.514731 |
| **sales\_month\_10** | 288.710119 |
| **sales\_month\_11** | 288.824510 |
| **sales\_month\_12** | 278.509459 |

**dtype:** float64

RANGE:

df.max(numeric\_only=True) - df.min(numeric\_only=True)

OUTPUT:

| **0** |
| --- |
| **product\_id** | 999.00 |
| **price** | 492.57 |
| **review\_score** | 4.00 |
| **review\_count** | 998.00 |
| **sales\_month\_1** | 1000.00 |
| **sales\_month\_2** | 998.00 |
| **sales\_month\_3** | 999.00 |
| **sales\_month\_4** | 1000.00 |
| **sales\_month\_5** | 1000.00 |
| **sales\_month\_6** | 1000.00 |
| **sales\_month\_7** | 1000.00 |
| **sales\_month\_8** | 995.00 |
| **sales\_month\_9** | 1000.00 |
| **sales\_month\_10** | 999.00 |
| **sales\_month\_11** | 1000.00 |
| **sales\_month\_12** | 996.00 |

**dtype:** float64

**VARIANCE:**

df.var(numeric\_only=True)

OUTPUT:

| **0** |
| --- |
| **product\_id** | 83416.666667 |
| **price** | 20911.468807 |
| **review\_score** | 1.371810 |
| **review\_count** | 79676.314278 |
| **sales\_month\_1** | 84066.060424 |
| **sales\_month\_2** | 81791.817897 |
| **sales\_month\_3** | 86442.393272 |
| **sales\_month\_4** | 82165.681352 |
| **sales\_month\_5** | 82854.354719 |
| **sales\_month\_6** | 83656.316908 |
| **sales\_month\_7** | 84708.523402 |
| **sales\_month\_8** | 84068.503743 |
| **sales\_month\_9** | 82664.720364 |
| **sales\_month\_10** | 83353.532729 |
| **sales\_month\_11** | 83419.597353 |
| **sales\_month\_12** | 77567.518523 |

**dtype:** float64

**TEST:**

T-TEST:

import pandas as pd

import scipy.stats as stats

df=pd.read\_csv('/content/ecommerce\_sales\_analysis.csv')

price\_df=df['price'].values

t\_stat, p\_val=stats.ttest\_1samp(price\_df, popmean=0)

print(f"One-sample t-test: t\_stat={t\_stat}, p\_val={p\_val}")

OUTPUT:

One-sample t-test: t\_stat=54.161868367327685, p\_val=1.6256677281864542e-299

CHI-SQUARE TEST:

import pandas as pd

from scipy.stats import chi2\_contingency

df=pd.read\_csv('/content/ecommerce\_sales\_analysis.csv')

contingency\_table=pd.crosstab(df['price'],df['category'])

chi2, p, dof, ex = chi2\_contingency(contingency\_table)

print(f"Chi-Square Test of Independent: chi2={chi2}, p={p}, dog={dof}, expected={ex}")

OUTPUT:

Chi-Square Test of Independent: chi2=5952.821923996988, p=0.47263552554058164, dog=5946, expected=[[0.154 0.14 0.138 ... 0.125 0.153 0.151]

[0.154 0.14 0.138 ... 0.125 0.153 0.151]

[0.154 0.14 0.138 ... 0.125 0.153 0.151]

...

[0.154 0.14 0.138 ... 0.125 0.153 0.151]

[0.154 0.14 0.138 ... 0.125 0.153 0.151]

[0.154 0.14 0.138 ... 0.125 0.153 0.151]]

KRUSKAL-WALLIS:

import numpy as np

from scipy.stats import kruskal

data=pd.read\_csv('/content/ecommerce\_sales\_analysis.csv')

h\_stat, p\_val = kruskal('price', 'category', 'product\_id')

print(f"Kruskal-Wallis Test: H\_stat={h\_stat}, p\_val={p\_val}")

OUTPUT:

Kruskal-Wallis Test: H\_stat=2.0, p\_val=0.36787944117144245

CO-VARIANCE:

df.cov(numeric\_only=True)

OUTPUT:

product\_idpricereview\_scorereview\_countsales\_month\_1sales\_month\_2sales\_month\_3sales\_month\_4sales\_month\_5sales\_month\_6sales\_month\_7sales\_month\_8sales\_month\_9sales\_month\_10sales\_month\_11sales\_month\_12product\_id83416.666667-1023.055511-13.597598-719.6276281574.6746756432.8033032768.777277-2411.340841-435.428428-2011.4199203913.314815-23.530030-4069.7327331998.4814813736.096096-2317.355355price-1023.05551120911.4688074.9049501722.108230-1419.686028-123.165008-986.5008301847.8404521249.754772-2151.105241-1178.285894-1889.512229-341.9188681598.500591332.979895768.461760review\_score-13.5975984.9049501.3718109.0424778.132987-25.694939-4.276973-5.665580-13.1247795.934912-1.9803846.9518476.5847065.3112860.921593-4.230584review\_count-719.6276281722.1082309.04247779676.314278-1578.375211-2940.981447-2438.353287-1757.297736-1121.117281-537.1235422451.149584-42.450364-1650.081686-5172.480268-2549.288316-2099.797113sales\_month\_11574.674675-1419.6860288.132987-1578.37521184066.0604244302.665399528.8387054047.8630251358.3780143041.5337162925.725359794.7846713573.937133179.599411512.822394-2955.137253sales\_month\_26432.803303-123.165008-25.694939-2940.9814474302.66539981791.817897-1072.830309-8276.1231264619.3431092906.904271-3372.749020-1888.816926405.402028-1526.553031-1457.3782961831.735590sales\_month\_32768.777277-986.500830-4.276973-2438.353287528.838705-1072.83030986442.393272-960.224421-766.0193851599.186620-2938.107236-4818.440932-2878.0302564883.411690-928.4927753120.691437sales\_month\_4-2411.3408411847.840452-5.665580-1757.2977364047.863025-8276.123126-960.22442182165.681352-1340.2148773823.5261071590.490437509.565278-1440.6853673232.0442903687.132458-2632.580258sales\_month\_5-435.4284281249.754772-13.124779-1121.1172811358.3780144619.343109-766.019385-1340.21487782854.3547193079.770088-1112.500635-1003.413800-1725.2884841305.7839722686.902330-3345.988873sales\_month\_6-2011.419920-2151.1052415.934912-537.1235423041.5337162906.9042711599.1866203823.5261073079.77008883656.316908-625.038221-4252.419977237.193291-24.021115-1517.962176-4308.812871sales\_month\_73913.314815-1178.285894-1.9803842451.1495842925.725359-3372.749020-2938.1072361590.490437-1112.500635-625.03822184708.523402-693.3586187427.3330593710.320543-5184.996214-2308.861107sales\_month\_8-23.530030-1889.5122296.951847-42.450364794.784671-1888.816926-4818.440932509.565278-1003.413800-4252.419977-693.35861884068.5037432957.616170-3542.293355-1912.282104384.543910sales\_month\_9-4069.732733-341.9188686.584706-1650.0816863573.937133405.402028-2878.030256-1440.685367-1725.288484237.1932917427.3330592957.61617082664.720364-1117.153485-4750.340032-1392.160685sales\_month\_101998.4814811598.5005915.311286-5172.480268179.599411-1526.5530314883.4116903232.0442901305.783972-24.0211153710.320543-3542.293355-1117.15348583353.5327292346.177453-4758.141169sales\_month\_113736.096096332.9798950.921593-2549.288316512.822394-1457.378296-928.4927753687.1324582686.902330-1517.962176-5184.996214-1912.282104-4750.3400322346.17745383419.597353-859.682150sales\_month\_12-2317.355355768.461760-4.230584-2099.797113-2955.1372531831.7355903120.691437-2632.580258-3345.988873-4308.812871-2308.861107384.543910-1392.160685-4758.141169-859.68215077567.518523

COR-RELATION:

df.corr(numeric\_only=True)

OUTPUT:

| **product\_id** | **price** | **review\_score** | **review\_count** | **sales\_month\_1** | **sales\_month\_2** | **sales\_month\_3** | **sales\_month\_4** | **sales\_month\_5** | **sales\_month\_6** | **sales\_month\_7** | **sales\_month\_8** | **sales\_month\_9** | **sales\_month\_10** | **sales\_month\_11** | **sales\_month\_12** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **product\_id** | 1.000000 | -0.024495 | -0.040197 | -0.008827 | 0.018804 | 0.077879 | 0.032606 | -0.029126 | -0.005238 | -0.024078 | 0.046554 | -0.000281 | -0.049009 | 0.023967 | 0.044788 | -0.028809 |
| **price** | -0.024495 | 1.000000 | 0.028960 | 0.042189 | -0.033860 | -0.002978 | -0.023203 | 0.044579 | 0.030024 | -0.051430 | -0.027996 | -0.045065 | -0.008224 | 0.038288 | 0.007972 | 0.019081 |
| **review\_score** | -0.040197 | 0.028960 | 1.000000 | 0.027351 | 0.023949 | -0.076709 | -0.012420 | -0.016875 | -0.038930 | 0.017519 | -0.005810 | 0.020471 | 0.019554 | 0.015707 | 0.002724 | -0.012969 |
| **review\_count** | -0.008827 | 0.042189 | 0.027351 | 1.000000 | -0.019286 | -0.036431 | -0.029381 | -0.021719 | -0.013798 | -0.006579 | 0.029836 | -0.000519 | -0.020332 | -0.063471 | -0.031269 | -0.026710 |
| **sales\_month\_1** | 0.018804 | -0.033860 | 0.023949 | -0.019286 | 1.000000 | 0.051889 | 0.006204 | 0.048705 | 0.016276 | 0.036269 | 0.034670 | 0.009454 | 0.042872 | 0.002146 | 0.006124 | -0.036595 |
| **sales\_month\_2** | 0.077879 | -0.002978 | -0.076709 | -0.036431 | 0.051889 | 1.000000 | -0.012759 | -0.100955 | 0.056114 | 0.035142 | -0.040520 | -0.022778 | 0.004930 | -0.018488 | -0.017643 | 0.022997 |
| **sales\_month\_3** | 0.032606 | -0.023203 | -0.012420 | -0.029381 | 0.006204 | -0.012759 | 1.000000 | -0.011394 | -0.009051 | 0.018806 | -0.034335 | -0.056523 | -0.034046 | 0.057530 | -0.010934 | 0.038111 |
| **sales\_month\_4** | -0.029126 | 0.044579 | -0.016875 | -0.021719 | 0.048705 | -0.100955 | -0.011394 | 1.000000 | -0.016243 | 0.046118 | 0.019064 | 0.006131 | -0.017481 | 0.039054 | 0.044536 | -0.032976 |
| **sales\_month\_5** | -0.005238 | 0.030024 | -0.038930 | -0.013798 | 0.016276 | 0.056114 | -0.009051 | -0.016243 | 1.000000 | 0.036992 | -0.013279 | -0.012023 | -0.020847 | 0.015713 | 0.032319 | -0.041738 |
| **sales\_month\_6** | -0.024078 | -0.051430 | 0.017519 | -0.006579 | 0.036269 | 0.035142 | 0.018806 | 0.046118 | 0.036992 | 1.000000 | -0.007425 | -0.050707 | 0.002852 | -0.000288 | -0.018171 | -0.053489 |
| **sales\_month\_7** | 0.046554 | -0.027996 | -0.005810 | 0.029836 | 0.034670 | -0.040520 | -0.034335 | 0.019064 | -0.013279 | -0.007425 | 1.000000 | -0.008216 | 0.088758 | 0.044156 | -0.061681 | -0.028484 |
| **sales\_month\_8** | -0.000281 | -0.045065 | 0.020471 | -0.000519 | 0.009454 | -0.022778 | -0.056523 | 0.006131 | -0.012023 | -0.050707 | -0.008216 | 1.000000 | 0.035478 | -0.042316 | -0.022835 | 0.004762 |
| **sales\_month\_9** | -0.049009 | -0.008224 | 0.019554 | -0.020332 | 0.042872 | 0.004930 | -0.034046 | -0.017481 | -0.020847 | 0.002852 | 0.088758 | 0.035478 | 1.000000 | -0.013458 | -0.057205 | -0.017386 |
| **sales\_month\_10** | 0.023967 | 0.038288 | 0.015707 | -0.063471 | 0.002146 | -0.018488 | 0.057530 | 0.039054 | 0.015713 | -0.000288 | 0.044156 | -0.042316 | -0.013458 | 1.000000 | 0.028136 | -0.059175 |
| **sales\_month\_11** | 0.044788 | 0.007972 | 0.002724 | -0.031269 | 0.006124 | -0.017643 | -0.010934 | 0.044536 | 0.032319 | -0.018171 | -0.061681 | -0.022835 | -0.057205 | 0.028136 | 1.000000 | -0.010687 |
| **sales\_month\_12** | -0.028809 | 0.019081 | -0.012969 | -0.026710 | -0.036595 | 0.022997 | 0.038111 | -0.032976 | -0.041738 | -0.053489 | -0.028484 | 0.004762 | -0.017386 | -0.059175 | -0.010687 | 1.000000 |

c) Visualize the univariate and multivariate with various plots.

import matplotlib.pyplot as plt

import seaborn as sns

import pandas as pd

df = pd.read\_csv('/content/ecommerce\_sales\_analysis.csv')

plt.figure(figsize=(16, 4))

plt.subplot(1, 4, 1)

plt.hist(df['price'], bins=10, color='skyblue', edgecolor='black')

plt.xlabel('price')

plt.ylabel('category')

plt.title('Histogram ')

plt.subplot(1, 4, 2)

sns.boxplot(data=df, x='price', color='salmon')

plt.xlabel('price')

plt.title('Box Plot')

plt.subplot(1, 4, 3)

sns.violinplot(data=df, x='price', color='orange')

plt.xlabel('price')

plt.title('Violin Plot ')

plt.subplot(1, 4, 4)

sns.displot(df['price'], kde=True, color='purple')

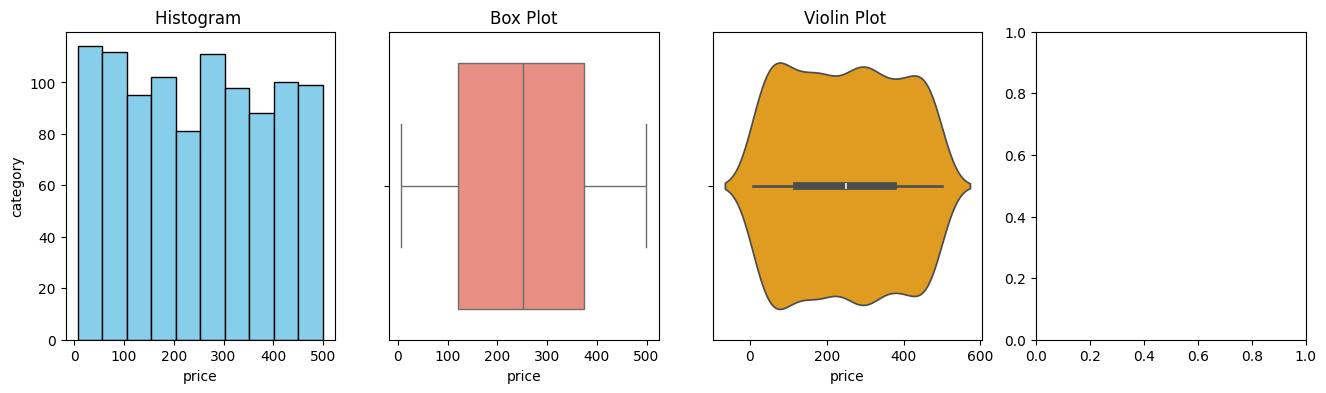
plt.xlabel('price')

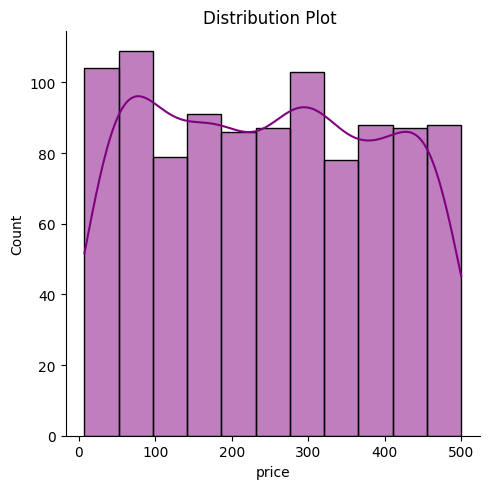
plt.title('Distribution Plot ')

plt.tight\_layout()

plt.show()

OUTPUT:





import matplotlib.pyplot as plt

import seaborn as sns

import pandas as pd

df=pd.read\_csv('/content/ecommerce\_sales\_analysis.csv')

plt.figure(figsize=(16, 4))

plt.subplot(1, 4, 1)

plt.scatter(df['price'], df['category'], color='green')

plt.xlabel('price')

plt.ylabel('category')

plt.title('Scatter Plot')

plt.subplot(1, 4, 2)

sns.pairplot(df, vars=['price', 'category', 'product\_id'])

plt.subplot(1, 4, 3)

sns.heatmap(df.corr(), annot=True, cmap='coolwarm')

plt.title('Heatmap')

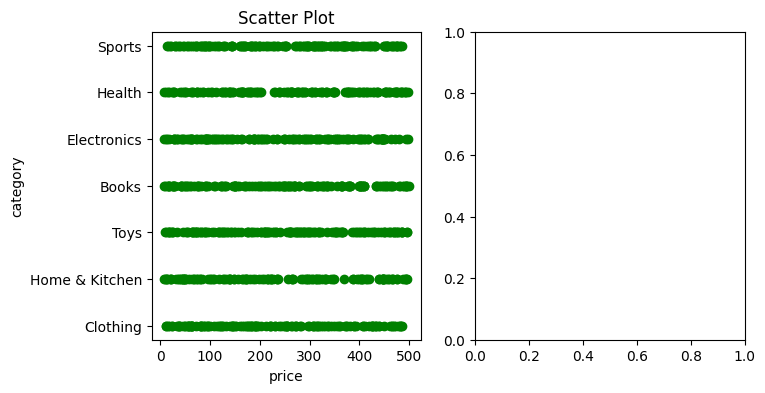
plt.subplot(1, 4, 4)

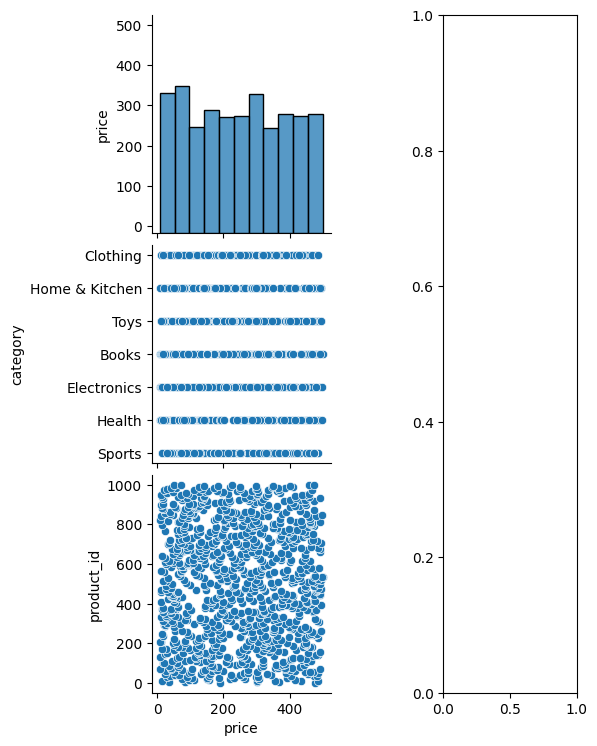
sns.jointplot(x='price', y='category', data=df, kind='hex', color='purple')

plt.tight\_layout()

plt.show()

OUTPUT:





**DATASET – 2**

**MEAN:**

import pandas as pd

df=pd.read\_csv('/content/student-dataset.csv')

df.mean(numeric\_only=True)

**OUTPUT:**

| **0** |
| --- |
| **id** | 153.000000 |
| **latitude** | 32.863388 |
| **longitude** | -64.539121 |
| **ethnic.group** | NaN |
| **age** | 21.964169 |
| **english.grade** | 3.369707 |
| **math.grade** | 3.414332 |
| **sciences.grade** | 3.446580 |
| **language.grade** | 4.396417 |
| **portfolio.rating** | 3.986971 |
| **coverletter.rating** | 4.110749 |
| **refletter.rating** | 4.188925 |

**dtype:** float64

**MODE:**

df.mode(numeric\_only=True)

**OUTPUT:**

| **id** | **latitude** | **longitude** | **ethnic.group** | **age** | **english.grade** | **math.grade** | **sciences.grade** | **language.grade** | **portfolio.rating** | **coverletter.rating** | **refletter.rating** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | 0 | 34.05 | -118.24 | NaN | 22.0 | 3.9 | 3.9 | 3.9 | 5.0 | 4.0 | 4.0 | 4.0 |
| **1** | 1 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **2** | 2 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **3** | 3 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **4** | 4 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **...** | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... | ... |
| **302** | 302 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **303** | 303 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **304** | 304 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **305** | 305 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |
| **306** | 306 | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN | NaN |

307 rows × 12 columns

**MEDIAN:**

df.median(numeric\_only=True)

**OUTPUT:**

| **0** |
| --- |
| **id** | 153.00 |
| **latitude** | 34.39 |
| **longitude** | -99.14 |
| **ethnic.group** | NaN |
| **age** | 22.00 |
| **english.grade** | 3.50 |
| **math.grade** | 3.50 |
| **sciences.grade** | 3.60 |
| **language.grade** | 5.00 |
| **portfolio.rating** | 4.00 |
| **coverletter.rating** | 4.00 |
| **refletter.rating** | 4.00 |

**dtype:** float64

RANGE:

df.max(numeric\_only=True)-df.min(numeric\_only=True)

OUTPUT:

| **0** |
| --- |
| **id** | 306.00 |
| **latitude** | 93.34 |
| **longitude** | 262.88 |
| **ethnic.group** | NaN |
| **age** | 7.00 |
| **english.grade** | 2.50 |
| **math.grade** | 1.90 |
| **sciences.grade** | 2.60 |
| **language.grade** | 4.00 |
| **portfolio.rating** | 4.00 |
| **coverletter.rating** | 4.00 |
| **refletter.rating** | 4.00 |

**dtype:** float64

VARIANCE:

df.var(numeric\_only=True)

OUTPUT:

| **0** |
| --- |
| **id** | 7879.666667 |
| **latitude** | 182.211709 |
| **longitude** | 6601.423744 |
| **ethnic.group** | NaN |
| **age** | 1.557536 |
| **english.grade** | 0.290223 |
| **math.grade** | 0.227376 |
| **sciences.grade** | 0.259163 |
| **language.grade** | 0.992961 |
| **portfolio.rating** | 0.862575 |
| **coverletter.rating** | 0.678871 |
| **refletter.rating** | 0.709289 |

**Dtype:** float64

STANDARD DEVIATION:

df.std(numeric\_only=True)

OUTPUT:

| **0** |
| --- |
| **id** | 88.767487 |
| **latitude** | 13.498582 |
| **longitude** | 81.249146 |
| **ethnic.group** | NaN |
| **age** | 1.248013 |
| **english.grade** | 0.538724 |
| **math.grade** | 0.476839 |
| **sciences.grade** | 0.509081 |
| **language.grade** | 0.996474 |
| **portfolio.rating** | 0.928749 |
| **coverletter.rating** | 0.823936 |
| **refletter.rating** | 0.842193 |

**dtype:** float64

TEST:

name1=df[df['name']=='name']['longitude']

id2=df[df['id']=='id']['longitude']

t\_stat,p\_value=stats.ttest(name1,id2)

print("T-test Results:")

print("T-statistic:", t\_stat)

OUTPUT:

T-test Results:

T-statistic: nan

observed\_values=pd.crosstab(df['id'],df['age'])

chi2,p,expected=stats.chi2\_contingency(observed\_values)

print("Chi-square Test Results:")

print("Chi-square statistic:",chi2)

OUTPUT:

Chi-square Test Results:

Chi-square statistic: 2149.0

addCode

addText

ages=[df[df['age']==group]['language.grade']for group in df['age'].unique()]

f\_stat,p\_value=stats\_oneway(\*ages)

print("ANOVA Test Results:")

print("F-statistic:",f\_stat)

OUTPUT:

Chi-square Test Results:

Chi-square statistic: 2149.0

ages=[df[df['age']==group]['language.grade']for group in df['age'].unique()]

f\_stat,p\_value=stats\_oneway(\*ages)

print("ANOVA Test Results:")

print("F-statistic:",f\_stat)

OUTPUT:

ANOVA Test Results:

F-statistic: 0.7042213380228663

CO-VARIANCE:

df.cov(numeric\_only=True)

OUTPUT:

idlatitudelongitudeethnic.groupageenglish.grademath.gradesciences.gradelanguage.gradeportfolio.ratingcoverletter.ratingrefletter.ratingid7879.666667-4.623366-1035.030621NaN-4.5947711.988889-0.160784-3.72222218.4866015.5326802.887255-0.271242latitude-4.623366182.211709-24.577710NaN-0.227133-0.4052990.1220400.3399925.687163-0.308877-0.514184-0.563158longitude-1035.030621-24.5777106601.423744NaN-0.192354-3.3800622.682605-0.233361-49.9908503.847201-2.3848852.902382ethnic.groupNaNNaNNaNNaNNaNNaNNaNNaNNaNNaNNaNNaNage-4.594771-0.227133-0.192354NaN1.557536-0.0010890.010973-0.0617240.033531-0.059292-0.051574-0.035692english.grade1.988889-0.405299-3.380062NaN-0.0010890.290223-0.0029630.0013180.0170150.012349-0.042059-0.007984math.grade-0.1607840.1220402.682605NaN0.010973-0.0029630.2273760.014134-0.0093600.0456120.034028-0.010887sciences.grade-3.7222220.339992-0.233361NaN-0.0617240.0013180.0141340.2591630.0061480.0277330.0441710.002936language.grade18.4866015.687163-49.990850NaN0.0335310.017015-0.0093600.0061480.992961-0.031746-0.038491-0.035922portfolio.rating5.532680-0.3088773.847201NaN-0.0592920.0123490.0456120.027733-0.0317460.862575-0.0148920.015542coverletter.rating2.887255-0.514184-2.384885NaN-0.051574-0.0420590.0340280.044171-0.038491-0.0148920.6788710.028028refletter.rating-0.271242-0.5631582.902382NaN-0.035692-0.007984-0.0108870.002936-0.0359220.0155420.0280280.709289

COR-RELATION:

df.corr(numeric\_only=True)

OUTPUT:

idlatitudelongitudeethnic.groupageenglish.grademath.gradesciences.gradelanguage.gradeportfolio.ratingcoverletter.ratingrefletter.ratingid1.000000-0.003858-0.143509NaN-0.0414750.041590-0.003799-0.0823690.2089960.0671090.039476-0.003628latitude-0.0038581.000000-0.022410NaN-0.013483-0.0557340.0189600.0494760.422806-0.024638-0.046231-0.049537longitude-0.143509-0.0224101.000000NaN-0.001897-0.0772220.069241-0.005642-0.6174550.050983-0.0356250.042415ethnic.groupNaNNaNNaNNaNNaNNaNNaNNaNNaNNaNNaNNaNage-0.041475-0.013483-0.001897NaN1.000000-0.0016200.018438-0.0971520.026963-0.051154-0.050156-0.033958english.grade0.041590-0.055734-0.077222NaN-0.0016201.000000-0.0115350.0048040.0316960.024681-0.094754-0.017596math.grade-0.0037990.0189600.069241NaN0.018438-0.0115351.0000000.058225-0.0196990.1029940.086612-0.027108sciences.grade-0.0823690.049476-0.005642NaN-0.0971520.0048040.0582251.0000000.0121190.0586560.1053070.006848language.grade0.2089960.422806-0.617455NaN0.0269630.031696-0.0196990.0121191.000000-0.034303-0.046881-0.042804portfolio.rating0.067109-0.0246380.050983NaN-0.0511540.0246810.1029940.058656-0.0343031.000000-0.0194610.019869coverletter.rating0.039476-0.046231-0.035625NaN-0.050156-0.0947540.0866120.105307-0.046881-0.0194611.0000000.040391refletter.rating-0.003628-0.0495370.042415NaN-0.033958-0.017596-0.0271080.006848-0.0428040.0198690.0403911.000000

import pandas as pd

df=pd.read\_csv('/content/student-dataset.csv')

import matplotlib.pyplot as plt

import seaborn as sns

plt.figure(figsize=(10,15))

plt.subplot(5,6,7)

plt.bar(df['latitude'],df['age'])

plt.title("bar plot")

plt.xlabel('latitude')

plt.ylabel('age')

plt.show()

plt.subplot(3,4,5)

plt.boxplot(df['math.grade'])

plt.title("box plot")

plt.xlabel('sciences.grade')

plt.ylabel('math.grade')

plt.show()

plt.subplot(4,5,6)

plt.hist(df['age'],color='green')

plt.title("histogram")

plt.xlabel("math.grade")

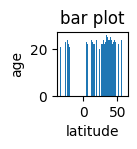
plt.ylabel("age")

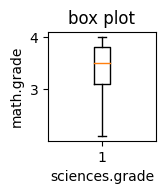
plt.show()

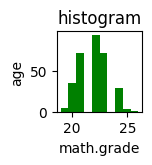
sns.violinplot(y=df['age'])

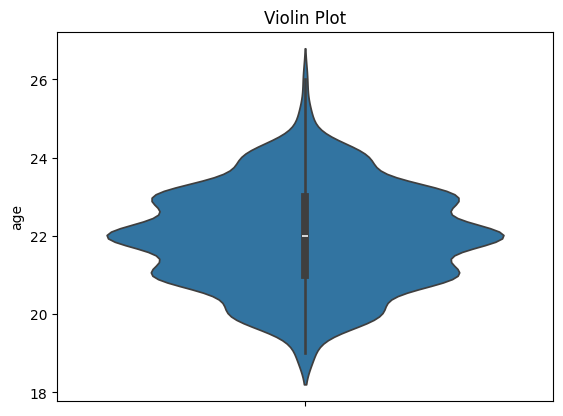
plt.title('Violin Plot')

plt.show()









import pandas as pd

df=pd.read\_csv('/content/student-dataset.csv')

import matplotlib.pyplot as plt

import seaborn as sns

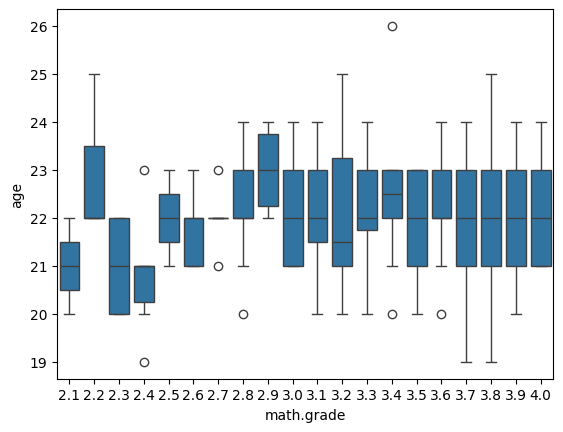
plt.figure(figsize=(10,15))

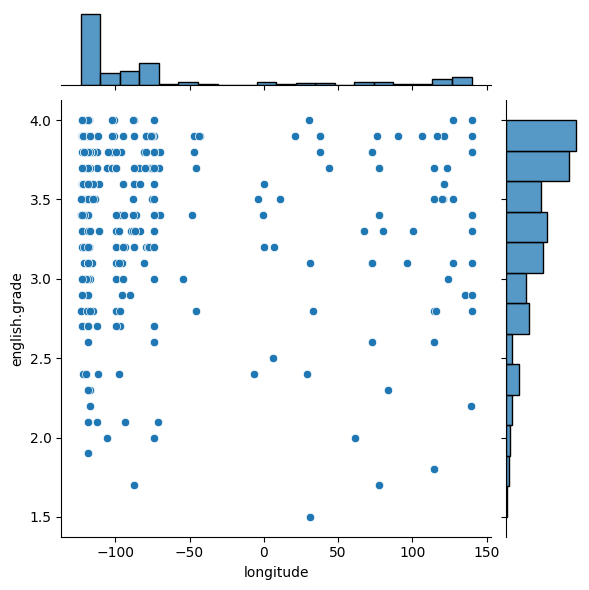
sns.boxplot(x="math.grade",y="age",data=df)

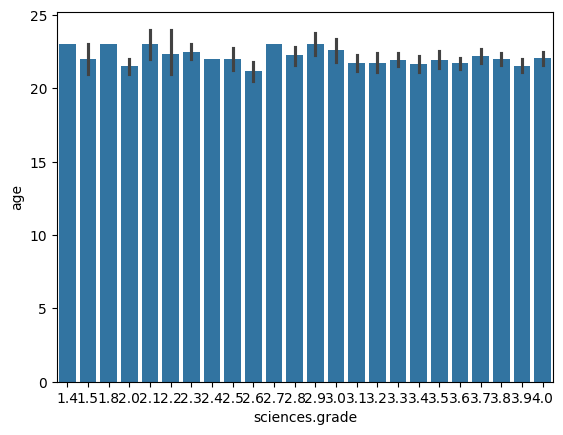
sns.jointplot(x="longitude",y="english.grade",data=df)

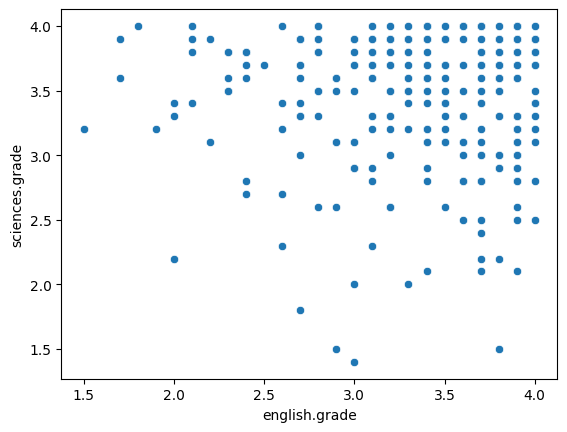
sns.barplot(x='sciences.grade',y='age',data= df)

sns.scatterplot(x='english.grade',y='sciences.grade',data=df)









Perform any probability calculation:

import pandas as pd

file\_path=('/content/student-dataset.csv')

if'age'in df.columns:

  student\_above\_40 = df[df['age']>40]

  probability\_above\_40=len(student\_above\_40)/len(df)

  print(f"probability of a person being older than 40:{probability\_above\_40:4f}")

else:

   print("The'age'column is not found in the dataset.")

output:

probability of a person being older than 40:0.000000