3-praveenk-chintatejdeepreddy-code

May 21, 2023

Name : Praveen K Roll No: 211AI028 Name : Chinta Tejdeep Reddy Roll No: 211AI013 ASSIGNMENT - 3

[93]:
```
import pandas as pd
import numpy as np;
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.decomposition import PCA
from sklearn.preprocessing import StandardScaler
from sklearn.preprocessing import LabelEncoder
from sklearn.preprocessing import MinMaxScaler
```

[61]:
```
#Loading the Dataframe
df = pd.read_csv(r"C:
 ↪\Users\Praveen\Desktop\ASSIGNMENTSEM-III\DS\3\Mobile_Recommendation_System-main\mainDataset
 ↪csv",encoding='ISO-8859-1')
```

[62]:
```
#Printing the dataframe
print(df)
```

```
                                                 Name  Rating  RAM Gb  ROM Gb  \
0    Samsung Galaxy Note 10 Plus (Aura Glow, 256 GB)     4.6      12     256
1                           Lenovo A7 (Blue, 32 GB)     4.0       2      32
2       Redmi Note 9 Pro Max (Champagne Gold, 64 GB)     4.3       6      64
3                 Realme X50 Pro (Moss Green, 256 GB)     4.1      12     256
4                             Lava Z61 (Gold, 16 GB)     4.2       2      16
..                                                 …     …      …       …
956               Lava Z61 Pro (Lavender Blue, 16 GB)     4.0       2      16
957                          Lenovo A7 (Black, 32 GB)     4.0       2      32
958            Itel VISION 1 (GRADATION BLUE, 32 GB)     2.5       2      32
959            Itel Vision1 (Gradation Green, 32 GB)     4.1       3      32
960             Itel Vision1 (Gradation Blue, 32 GB)     4.1       3      32

     Expandable GB  Size Cm  Size Inch  R1 Cam MP  R2 Cam MP  R3 Cam MP  …  \
0              1.0    17.27      6.800         12        0.0          0  …
1            128.0    15.47      6.090         13        0.0          0  …
2              NaN    16.94      6.670         64        0.0          0  …
3              NaN    16.36      6.440         64       12.0          8  …
```

```
4             64.0    13.84      5.450          8        0.0          0  …
..             …       …          …            …         …          …  …
956           128.0    13.84      5.450          8        0.0          0  …
957           128.0    15.47      6.090         13        0.0          0  …
958           128.0    15.46      6.088          8        0.0          0  …
959           128.0    15.46      6.088          8        0.3          0  …
960           128.0    15.46      6.088          8        0.3          0  …

     Brand name  Unnamed: 17  Unnamed: 18 Unnamed: 19 Unnamed: 20 Unnamed: 21  \
0       Samsung          NaN          NaN         NaN          NaN          NaN
1        Lenovo          NaN          NaN         NaN          NaN          NaN
2         Redmi          NaN          NaN         NaN          NaN          NaN
3        Realme          NaN          NaN         NaN          NaN          NaN
4          Lava          NaN          NaN         NaN          NaN          NaN
..          …            …            …           …            …            …
956        Lava          NaN          NaN         NaN          NaN          NaN
957      Lenovo          NaN          NaN         NaN          NaN          NaN
958        Itel          NaN          NaN         NaN          NaN          NaN
959        Itel          NaN          NaN         NaN          NaN          NaN
960        Itel          NaN          NaN         NaN          NaN          NaN

    Unnamed: 22  Unnamed: 23  Unnamed: 24  Pro
0           NaN          NaN          NaN  NaN
1           NaN          NaN          NaN  NaN
2           NaN          NaN          NaN  NaN
3           NaN          NaN          NaN  NaN
4           NaN          NaN          NaN  NaN
..          …            …            …    …
956         NaN          NaN          NaN  NaN
957         NaN          NaN          NaN  NaN
958         NaN          NaN          NaN  NaN
959         NaN          NaN          NaN  NaN
960         NaN          NaN          NaN  NaN

[961 rows x 26 columns]
```

[63]:
```
#Printing the first 5 elements of the dataframe
df.head()
```

[63]:
```
                                                Name  Rating  RAM Gb  ROM Gb  \
0  Samsung Galaxy Note 10 Plus (Aura Glow, 256 GB)     4.6      12     256
1                        Lenovo A7 (Blue, 32 GB)     4.0       2      32
2    Redmi Note 9 Pro Max (Champagne Gold, 64 GB)     4.3       6      64
3             Realme X50 Pro (Moss Green, 256 GB)     4.1      12     256
4                          Lava Z61 (Gold, 16 GB)     4.2       2      16

   Expandable GB  Size Cm  Size Inch  R1 Cam MP  R2 Cam MP  R3 Cam MP  …  \
```

```
0           1.0    17.27       6.80         12        0.0          0  …
1         128.0    15.47       6.09         13        0.0          0  …
2           NaN    16.94       6.67         64        0.0          0  …
3           NaN    16.36       6.44         64       12.0          8  …
4          64.0    13.84       5.45          8        0.0          0  …

   Brand name  Unnamed: 17  Unnamed: 18 Unnamed: 19 Unnamed: 20 Unnamed: 21  \
0     Samsung          NaN          NaN         NaN         NaN         NaN
1      Lenovo          NaN          NaN         NaN         NaN         NaN
2       Redmi          NaN          NaN         NaN         NaN         NaN
3      Realme          NaN          NaN         NaN         NaN         NaN
4        Lava          NaN          NaN         NaN         NaN         NaN

  Unnamed: 22  Unnamed: 23  Unnamed: 24  Pro
0         NaN          NaN          NaN  NaN
1         NaN          NaN          NaN  NaN
2         NaN          NaN          NaN  NaN
3         NaN          NaN          NaN  NaN
4         NaN          NaN          NaN  NaN

[5 rows x 26 columns]
```

[64]:
```
#Describe the dataset and all the measures
df.describe()
```

[64]:
```
            Rating      RAM Gb      ROM Gb  Expandable GB     Size Cm  \
count  936.000000  961.000000  961.000000     715.000000  961.000000
mean     4.227137    4.500520   78.817898     298.641958   15.682352
std      0.368420    2.094201   65.408979     151.091947    1.162916
min      2.300000    2.000000   16.000000       1.000000   12.700000
25%      4.200000    3.000000   32.000000     256.000000   15.210000
50%      4.300000    4.000000   64.000000     256.000000   16.000000
75%      4.400000    6.000000  128.000000     512.000000   16.510000
max      5.000000   12.000000  512.000000     512.000000   17.780000

        Size Inch   R1 Cam MP   R2 Cam MP   R3 Cam MP   R4 Cam MP  …  \
count  961.000000  961.000000  961.000000  961.000000  961.000000  …
mean     6.173996   26.070760    4.323205    1.637877    0.514048  …
std      0.457630   20.893111    5.162666    2.862469    1.113394  …
min      5.000000    5.000000    0.000000    0.000000    0.000000  …
25%      5.990000   13.000000    0.000000    0.000000    0.000000  …
50%      6.300000   13.000000    2.000000    0.000000    0.000000  …
75%      6.500000   48.000000    8.000000    2.000000    0.000000  …
max      7.000000  108.000000   48.000000   16.000000    8.000000  …

           Price Rs  Unnamed: 17  Unnamed: 18  Unnamed: 19  Unnamed: 20  \
count     959.000000          0.0          0.0          0.0          0.0
```

```
mean    17537.970803          NaN          NaN          NaN          NaN
std     14675.059110          NaN          NaN          NaN          NaN
min      3899.000000          NaN          NaN          NaN          NaN
25%      9560.000000          NaN          NaN          NaN          NaN
50%     13999.000000          NaN          NaN          NaN          NaN
75%     18734.000000          NaN          NaN          NaN          NaN
max    124999.000000          NaN          NaN          NaN          NaN

       Unnamed: 21  Unnamed: 22  Unnamed: 23  Unnamed: 24  Pro
count          0.0          0.0          0.0          0.0  0.0
mean           NaN          NaN          NaN          NaN  NaN
std            NaN          NaN          NaN          NaN  NaN
min            NaN          NaN          NaN          NaN  NaN
25%            NaN          NaN          NaN          NaN  NaN
50%            NaN          NaN          NaN          NaN  NaN
75%            NaN          NaN          NaN          NaN  NaN
max            NaN          NaN          NaN          NaN  NaN

[8 rows x 21 columns]
```

[117]:
```
#Shape of the dataset
df.shape
```

[117]:
```
(961, 29)
```

[66]:
```
#Checking the number of null values in each feature
df.isnull().sum()
```

[66]:
```
Name               0
Rating            25
RAM Gb             0
ROM Gb             0
Expandable GB    246
Size Cm            0
Size Inch          0
R1 Cam MP          0
R2 Cam MP          0
R3 Cam MP          0
R4 Cam MP          0
Battery Mah        0
Price Rs           2
Processor          0
Image              0
Processor name     0
Brand name         0
Unnamed: 17      961
Unnamed: 18      961
```

```
Unnamed: 19       961
Unnamed: 20       961
Unnamed: 21       961
Unnamed: 22       961
Unnamed: 23       961
Unnamed: 24       961
Pro               961
dtype: int64
```

Finding the Correlation between the various features to see how they vary with each other.

[67]:
```
correlation = df['Price Rs'].corr(df['Rating'])
print("Correlation:")
print(correlation)
```

```
Correlation:
0.30696556781567247
```

[68]:
```
correlation = df['Battery Mah'].corr(df['Rating'])
print("Correlation:")
print(correlation)
```

```
Correlation:
0.451432302331625
```

[69]:
```
correlation = df['RAM Gb'].corr(df['Rating'])
print("Correlation:")
print(correlation)
```

```
Correlation:
0.38343858940106756
```

[70]:
```
correlation = df['RAM Gb'].corr(df['Price Rs'])
print("Correlation:")
print(correlation)
```

```
Correlation:
0.6668573282484104
```

[71]:
```
correlation = df['ROM Gb'].corr(df['Rating'])
print("Correlation:")
print(correlation)
```

```
Correlation:
0.33083043521783173
```

[72]:
```
correlation = df['ROM Gb'].corr(df['Price Rs'])
print("Correlation:")
```

```
print(correlation)
```

```
Correlation:
0.6427890816574197
```

[73]:
```
correlation = df['Battery Mah'].corr(df['Price Rs'])
print("Correlation:")
print(correlation)
```

```
Correlation:
0.07362529314706924
```

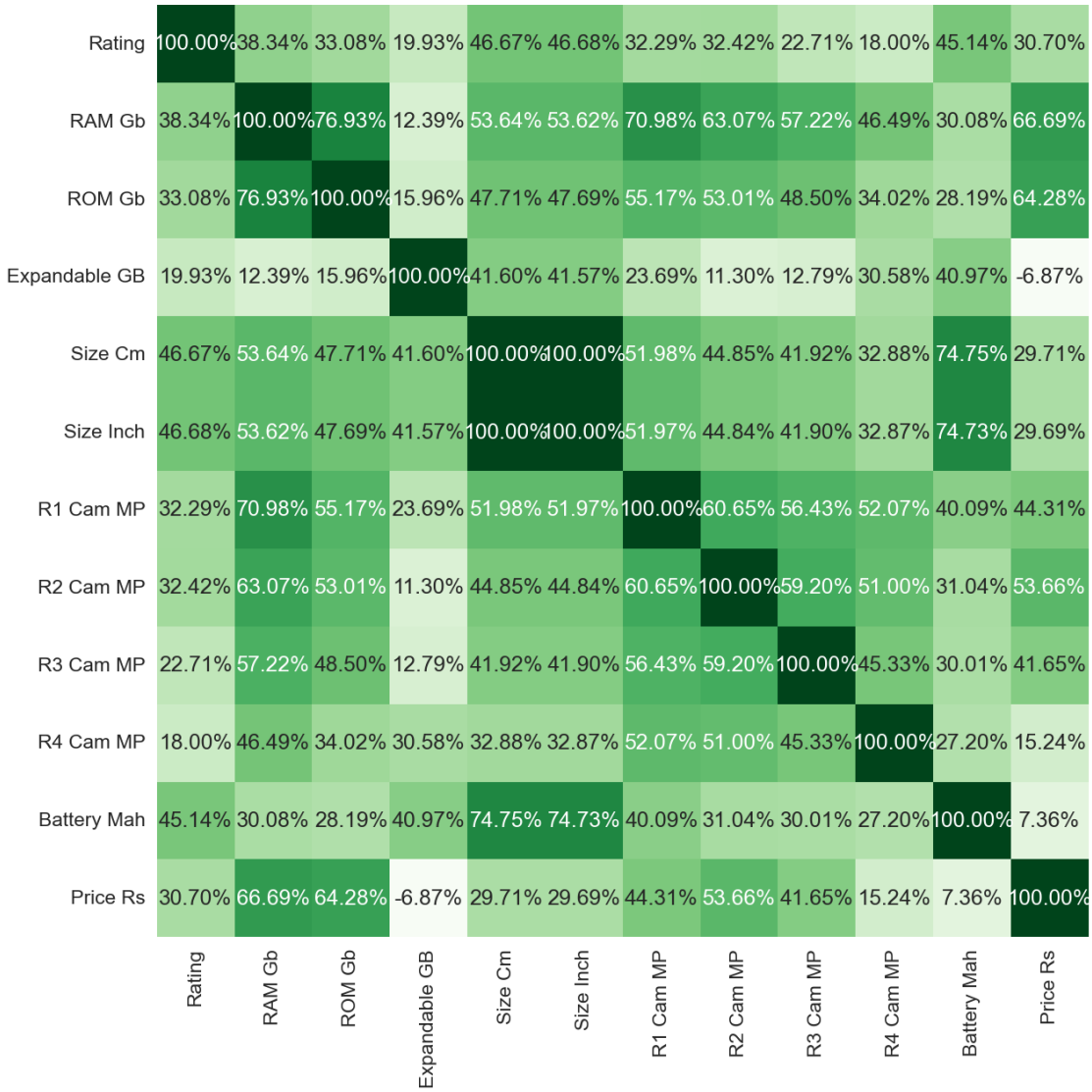
[74]:
```
pd = df
label_encoder = LabelEncoder()
pd['Processor_enc'] = label_encoder.fit_transform(df['Processor'])
correlation = pd['Processor_enc'].corr(df['Price Rs'])
print("Correlation:")
print(correlation)
```

```
Correlation:
-0.016568678246270813
```

Plotting the correlation heat map between the various features.

[75]:
```
plt.figure(figsize=(14,14))
cols = ['Rating','RAM Gb','ROM Gb','Expandable GB','Size Cm','Size Inch','R1␣
 ↪Cam MP','R2 Cam MP','R3 Cam MP','R4 Cam MP','Battery Mah','Price Rs']
pd = df[cols]
corr = pd.corr()
sns.heatmap(corr, cbar=False, square= True, fmt='.2%', annot=True,␣
 ↪cmap='Greens')
```

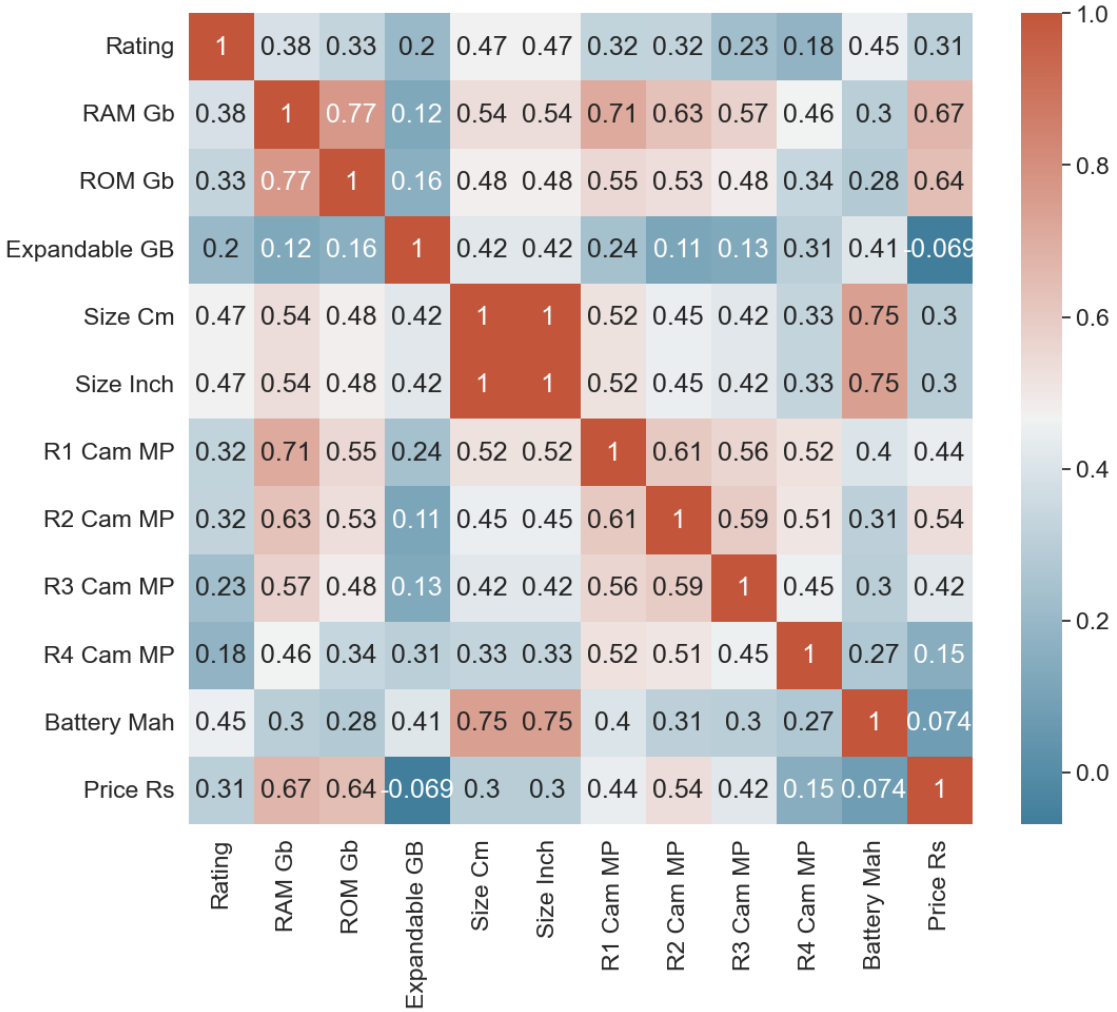
[75]: <AxesSubplot: >



[76]:
```
cols = ['Rating','RAM Gb','ROM Gb','Expandable GB','Size Cm','Size Inch','R1␣
↪Cam MP','R2 Cam MP','R3 Cam MP','R4 Cam MP','Battery Mah','Price Rs']
pd = df[cols]
corr = pd.corr()
#
# Set up the matplotlib plot configuration
#
f, ax = plt.subplots(figsize=(12, 10))
#
# Generate a mask for upper traingle
#
#mask = np.triu(np.ones_like(corr, dtype=bool))
#
```

```
# Configure a custom diverging colormap
#
cmap = sns.diverging_palette(230, 20, as_cmap=True)
#
# Draw the heatmap
#
sns.heatmap(corr, annot=True, cmap=cmap)
```

[76]: <AxesSubplot: >



Finding the Covariance between the different features.

[77]:
```
covariance = df['Price Rs'].cov(df['Rating'])

# Print the covariance
print("Covariance:")
```

```
print(covariance)
```

```
Covariance:
1673.9839854857914
```

[78]:
```
covariance = df['Size Cm'].cov(df['Battery Mah'])

# Print the covariance
print("Covariance:")
print(covariance)
```

```
Covariance:
751.345585598769
```

[79]:
```
covariance = df['RAM Gb'].cov(df['Battery Mah'])

# Print the covariance
print("Covariance:")
print(covariance)
```

```
Covariance:
544.4242976066597
```

[80]:
```
covariance = df['R1 Cam MP'].cov(df['Rating'])

# Print the covariance
print("Covariance:")
print(covariance)
```

```
Covariance:
2.492478632478632
```

[81]:
```
covariance = df['Expandable GB'].cov(df['Battery Mah'])

# Print the covariance
print("Covariance:")
print(covariance)
```

```
Covariance:
51086.74128812366
```

Covariance Heat Map

[82]:
```
cols = ['Rating','RAM Gb','ROM Gb','Expandable GB','Size Cm','Size Inch','R1␣
↪Cam MP','R2 Cam MP','R3 Cam MP','R4 Cam MP','Battery Mah','Price Rs']
pd = df[cols]

plt.figure(figsize=(24,24))
```

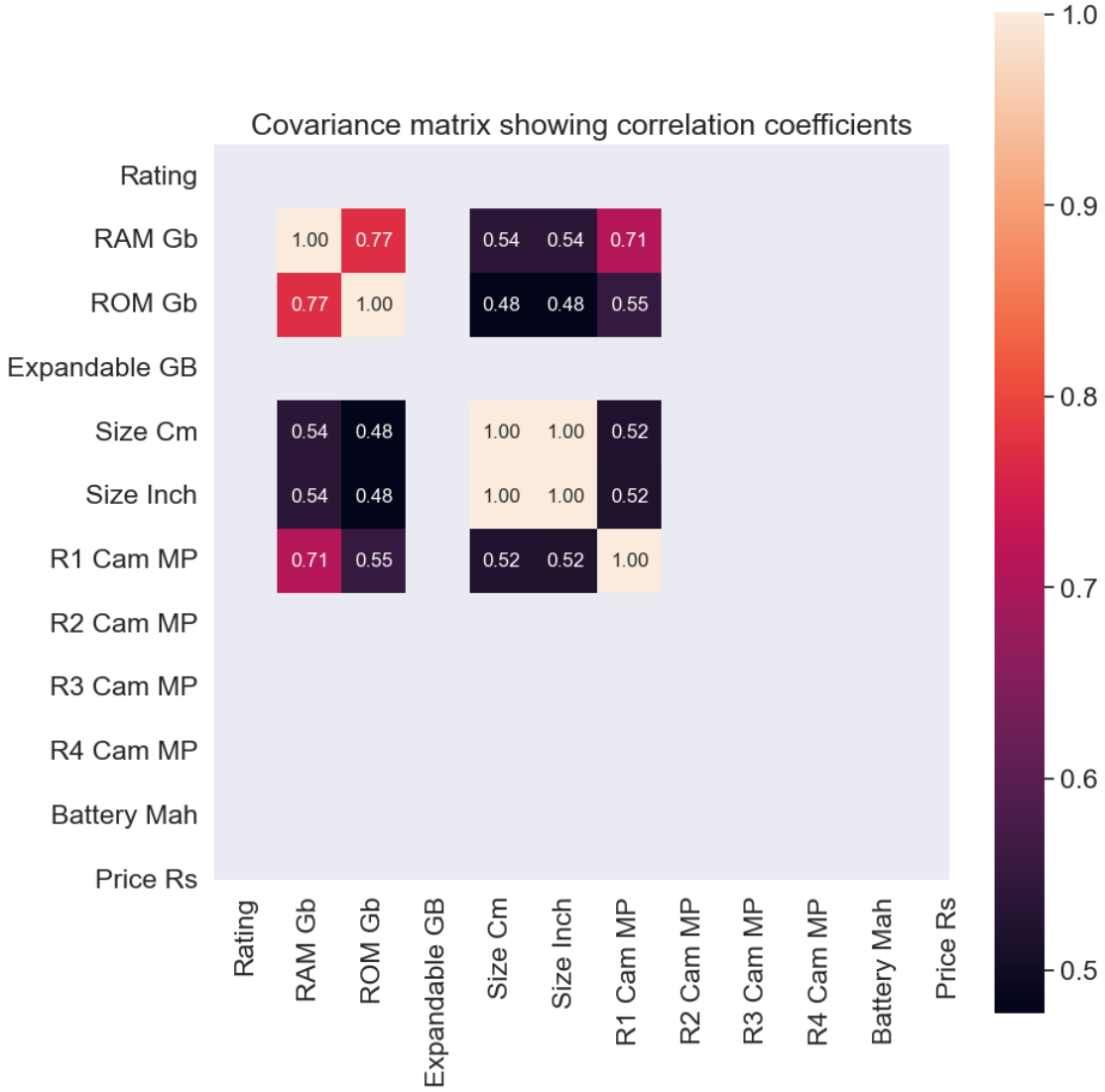
```
corr = pd.cov()
sns.heatmap(corr, cbar=False, square= True, fmt='.2%', annot=True,␣
 ↪cmap='Greens')
```

[82]: <AxesSubplot: >

Plotting the covariance matrix showing correlations coefficients

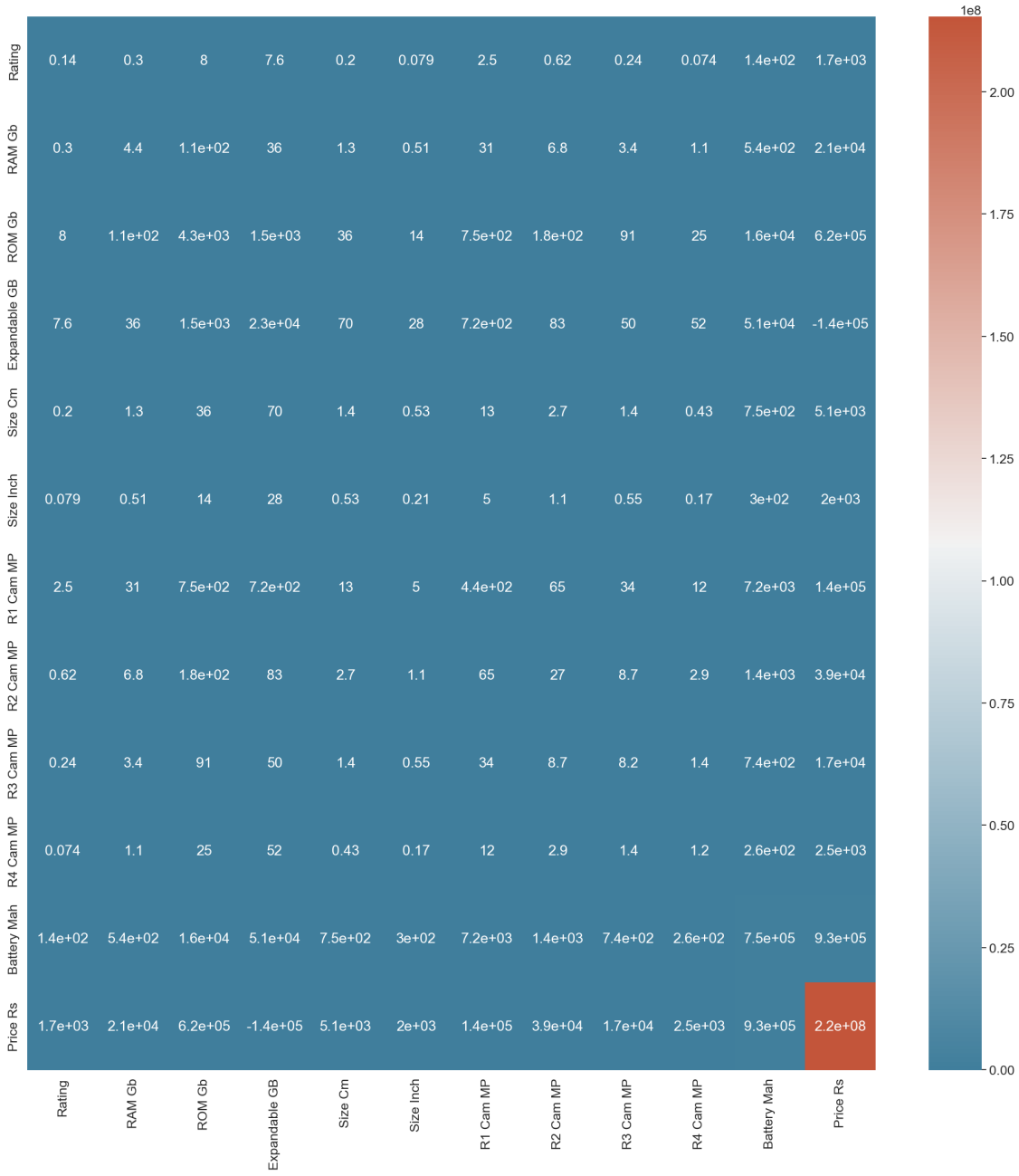
[83]:
```
cols = ['Rating','RAM Gb','ROM Gb','Expandable GB','Size Cm','Size Inch','R1␣
 ↪Cam MP','R2 Cam MP','R3 Cam MP','R4 Cam MP','Battery Mah','Price Rs']
from sklearn.preprocessing import StandardScaler
stdsc = StandardScaler()
X_std = stdsc.fit_transform(df[cols].iloc[:,range(0,7)].values)
```

```
cov_mat =np.cov(X_std.T)
plt.figure(figsize=(10,10))
sns.set(font_scale=1.5)
hm = sns.heatmap(cov_mat,
                 cbar=True,
                 annot=True,
                 square=True,
                 fmt='.2f',
                 annot_kws={'size': 12},
                 yticklabels=cols,
                 xticklabels=cols)
plt.title('Covariance matrix showing correlation coefficients')
plt.tight_layout()
plt.show()
```



[84]:
```
cols = ['Rating','RAM Gb','ROM Gb','Expandable GB','Size Cm','Size Inch','R1␣
 ↪Cam MP','R2 Cam MP','R3 Cam MP','R4 Cam MP','Battery Mah','Price Rs']
pd = df[cols]
corr = pd.cov()
#
# Set up the matplotlib plot configuration
#
f, ax = plt.subplots(figsize=(24,24))
#
# Generate a mask for upper traingle
#
#mask = np.triu(np.ones_like(corr, dtype=bool))
#
# Configure a custom diverging colormap
#
cmap = sns.diverging_palette(230, 20, as_cmap=True)
#
# Draw the heatmap
#
sns.heatmap(corr, annot=True, cmap=cmap)
```

[84]:
```
<AxesSubplot: >
```



Applying PCA on the camera features and reducing it to a single feature.

[86]:
```
cam_features = df[['R1 Cam MP', 'R2 Cam MP', 'R3 Cam MP', 'R4 Cam MP']]
scaler = StandardScaler()
cam_features_std = scaler.fit_transform(cam_features)
pca = PCA(n_components=1)
cam_features_pca = pca.fit_transform(cam_features_std)
df['Cam Reduced'] = cam_features_pca
```

```
print("Explained Variance Ratio:", pca.explained_variance_ratio_)
```

```
Explained Variance Ratio: [0.65677317]
```

Applying PCA on the Size features and reducing it to a single feature.

[87]:
```
siz_features = df[['Size Cm','Size Inch']]


scaler = StandardScaler()
siz_features_std = scaler.fit_transform(siz_features)


pca = PCA(n_components=1)
siz_features_pca = pca.fit_transform(siz_features_std)


df['Size Reduced'] = siz_features_pca


print("Explained Variance Ratio:", pca.explained_variance_ratio_)
```

```
Explained Variance Ratio: [0.99999894]
```

[88]:
```
print(df.columns)
```

```
Index(['Name', 'Rating', 'RAM Gb', 'ROM Gb', 'Expandable GB', 'Size Cm',
       'Size Inch', 'R1 Cam MP', 'R2 Cam MP', 'R3 Cam MP', 'R4 Cam MP',
       'Battery Mah', 'Price Rs', 'Processor', 'Image', 'Processor name',
       'Brand name', 'Unnamed: 17', 'Unnamed: 18', 'Unnamed: 19',
       'Unnamed: 20', 'Unnamed: 21', 'Unnamed: 22', 'Unnamed: 23',
       'Unnamed: 24', 'Pro', 'Processor_enc', 'Cam Reduced', 'Size Reduced'],
      dtype='object')
```

Creating the new dataframe with reduced features.

[109]:
```
new_pdf = df[['Name', 'Rating', 'RAM Gb', 'ROM Gb', 'Expandable GB','Size␣
 ↪Reduced','Cam Reduced',
       'Battery Mah', 'Price Rs', 'Processor','Processor name',
       'Brand name']]
```

[110]:
```
new_pdf.head()
```

[110]:

	Name	Rating	RAM Gb	ROM Gb	\
0	Samsung Galaxy Note 10 Plus (Aura Glow, 256 GB)	4.6	12	256	
1	Lenovo A7 (Blue, 32 GB)	4.0	2	32	
2	Redmi Note 9 Pro Max (Champagne Gold, 64 GB)	4.3	6	64	
3	Realme X50 Pro (Moss Green, 256 GB)	4.1	12	256	
4	Lava Z61 (Gold, 16 GB)	4.2	2	16	

```
   Expandable GB  Size Reduced  Cam Reduced  Battery Mah  Price Rs  \
0            1.0     -1.933641     0.292194         4300   85000.0
1          128.0      0.259041     0.333936         4000    8490.0
2            0.0     -1.531907     2.462778         5020   17767.0
3            0.0     -0.823487     8.276223         4200   47999.0
4           64.0      2.240084     0.125226         3000    5999.0

                                  Processor Processor name Brand name
0                          Exynos 9820 Processor         Exynos    Samsung
1              Octa-core Unisoc SC9863 Processor      Octa-core     Lenovo
2   Qualcomm Snapdragon 720G octa-core processor       Qualcomm      Redmi
3              Qualcomm Snapdragon 865 Processor       Qualcomm     Realme
4          1.5 GHz Quad Core Processor Processor            1.5       Lava
```

Standardizing the Cam Reduced features to a range of values between 1 to 16

[111]:
```
scaler = MinMaxScaler(feature_range=(0, 16))
# Select the column you want to scale
column_name = 'Cam Reduced'

# Extract the values of the selected column
column_values = new_pdf[column_name].values

# Reshape the column values to a 2D array
reshaped_values = column_values.reshape(-1, 1)

# Apply Min-Max scaling to the column values
scaled_values = scaler.fit_transform(reshaped_values)

# Assign the scaled values back to the dataframe
new_pdf[column_name] = scaled_values.flatten()
```

```
C:\Users\Praveen\AppData\Local\Temp\ipykernel_21376\1600270228.py:15:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  new_pdf[column_name] = scaled_values.flatten()
```

[112]:
```
new_pdf.head()
```

[112]:
```
                                        Name  Rating  RAM Gb  ROM Gb  \
0  Samsung Galaxy Note 10 Plus (Aura Glow, 256 GB)     4.6      12     256
1                          Lenovo A7 (Blue, 32 GB)     4.0       2      32
```

```
2     Redmi Note 9 Pro Max (Champagne Gold, 64 GB)     4.3       6      64
3              Realme X50 Pro (Moss Green, 256 GB)     4.1      12     256
4                          Lava Z61 (Gold, 16 GB)     4.2       2      16

   Expandable GB  Size Reduced  Cam Reduced  Battery Mah  Price Rs  \
0            1.0     -1.933641     0.292194         4300   85000.0
1          128.0      0.259041     0.333936         4000    8490.0
2            0.0     -1.531907     2.462778         5020   17767.0
3            0.0     -0.823487     8.276223         4200   47999.0
4           64.0      2.240084     0.125226         3000    5999.0

                                    Processor Processor name Brand name
0                          Exynos 9820 Processor         Exynos    Samsung
1              Octa-core Unisoc SC9863 Processor      Octa-core     Lenovo
2   Qualcomm Snapdragon 720G octa-core processor       Qualcomm      Redmi
3              Qualcomm Snapdragon 865 Processor       Qualcomm     Realme
4          1.5 GHz Quad Core Processor Processor            1.5       Lava
```

Standardizing the Size Reduced features to a range of values between 1 to 16

[113]:
```
scaler = MinMaxScaler(feature_range=(0,6))
# Select the column you want to scale
column_name = 'Size Reduced'

# Extract the values of the selected column
column_values = new_pdf[column_name].values

# Reshape the column values to a 2D array
reshaped_values = column_values.reshape(-1, 1)

# Apply Min-Max scaling to the column values
scaled_values = scaler.fit_transform(reshaped_values)

# Assign the scaled values back to the dataframe
new_pdf[column_name] = scaled_values.flatten()
```

```
C:\Users\Praveen\AppData\Local\Temp\ipykernel_21376\2745112538.py:15:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame.
Try using .loc[row_indexer,col_indexer] = value instead

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  new_pdf[column_name] = scaled_values.flatten()
```

[114]:
```
new_pdf.head()
```

[114]:

	Name	Rating	RAM Gb	ROM Gb	\
0	Samsung Galaxy Note 10 Plus (Aura Glow, 256 GB)	4.6	12	256	
1	Lenovo A7 (Blue, 32 GB)	4.0	2	32	
2	Redmi Note 9 Pro Max (Champagne Gold, 64 GB)	4.3	6	64	
3	Realme X50 Pro (Moss Green, 256 GB)	4.1	12	256	
4	Lava Z61 (Gold, 16 GB)	4.2	2	16	

	Expandable GB	Size Reduced	Cam Reduced	Battery Mah	Price Rs	\
0	1.0	0.601181	0.292194	4300	85000.0	
1	128.0	2.729173	0.333936	4000	8490.0	
2	0.0	0.991063	2.462778	5020	17767.0	
3	0.0	1.678583	8.276223	4200	47999.0	
4	64.0	4.651771	0.125226	3000	5999.0	

	Processor	Processor name	Brand name
0	Exynos 9820 Processor	Exynos	Samsung
1	Octa-core Unisoc SC9863 Processor	Octa-core	Lenovo
2	Qualcomm Snapdragon 720G octa-core processor	Qualcomm	Redmi
3	Qualcomm Snapdragon 865 Processor	Qualcomm	Realme
4	1.5 GHz Quad Core Processor Processor	1.5	Lava

Imputing. Fill the null values woth 0 in the Expandable GB feature.

[115]:
```
new_pdf['Expandable GB'].fillna(0, inplace=True)
```

```
C:\Users\Praveen\AppData\Local\Temp\ipykernel_21376\2747860075.py:1:
SettingWithCopyWarning:
A value is trying to be set on a copy of a slice from a DataFrame

See the caveats in the documentation: https://pandas.pydata.org/pandas-
docs/stable/user_guide/indexing.html#returning-a-view-versus-a-copy
  new_pdf['Expandable GB'].fillna(0, inplace=True)
```

[116]:
```
new_pdf.head()
```

[116]:

	Name	Rating	RAM Gb	ROM Gb	\
0	Samsung Galaxy Note 10 Plus (Aura Glow, 256 GB)	4.6	12	256	
1	Lenovo A7 (Blue, 32 GB)	4.0	2	32	
2	Redmi Note 9 Pro Max (Champagne Gold, 64 GB)	4.3	6	64	
3	Realme X50 Pro (Moss Green, 256 GB)	4.1	12	256	
4	Lava Z61 (Gold, 16 GB)	4.2	2	16	

	Expandable GB	Size Reduced	Cam Reduced	Battery Mah	Price Rs	\
0	1.0	0.601181	0.292194	4300	85000.0	
1	128.0	2.729173	0.333936	4000	8490.0	
2	0.0	0.991063	2.462778	5020	17767.0	
3	0.0	1.678583	8.276223	4200	47999.0	
4	64.0	4.651771	0.125226	3000	5999.0	

		Processor	Processor name	Brand name
0	Exynos 9820	Processor	Exynos	Samsung
1	Octa-core Unisoc SC9863	Processor	Octa-core	Lenovo
2	Qualcomm Snapdragon 720G octa-core	processor	Qualcomm	Redmi
3	Qualcomm Snapdragon 865	Processor	Qualcomm	Realme
4	1.5 GHz Quad Core Processor	Processor	1.5	Lava

[ ]:

[ ]: