# eda-in-business-analytics

February 7, 2024

**Data Collection and Preprocessing:**

[1]:
```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

[2]:
```
data = pd.read_csv("/content/vgsales.csv")
data.head()
```

[2]:
```
   Rank                      Name Platform    Year         Genre Publisher  \
0     1                Wii Sports      Wii  2006.0        Sports  Nintendo
1     2         Super Mario Bros.      NES  1985.0      Platform  Nintendo
2     3            Mario Kart Wii      Wii  2008.0        Racing  Nintendo
3     4         Wii Sports Resort      Wii  2009.0        Sports  Nintendo
4     5  Pokemon Red/Pokemon Blue       GB  1996.0  Role-Playing  Nintendo

   NA_Sales  EU_Sales  JP_Sales  Other_Sales  Global_Sales
0     41.49     29.02      3.77         8.46         82.74
1     29.08      3.58      6.81         0.77         40.24
2     15.85     12.88      3.79         3.31         35.82
3     15.75     11.01      3.28         2.96         33.00
4     11.27      8.89     10.22         1.00         31.37
```

**Data Understanding:**

[3]:
```
print(data.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 16598 entries, 0 to 16597
Data columns (total 11 columns):
 #   Column        Non-Null Count  Dtype
---  ------        --------------  -----
 0   Rank          16598 non-null  int64
 1   Name          16598 non-null  object
 2   Platform      16598 non-null  object
 3   Year          16327 non-null  float64
 4   Genre         16598 non-null  object
 5   Publisher     16540 non-null  object
 6   NA_Sales      16598 non-null  float64
 7   EU_Sales      16598 non-null  float64
 8   JP_Sales      16598 non-null  float64
 9   Other_Sales   16598 non-null  float64
 10  Global_Sales  16598 non-null  float64
dtypes: float64(6), int64(1), object(4)
memory usage: 1.4+ MB
None
```

**Data Cleaning and Preparation:**

[4]:
```
# Handling missing values
data.dropna(inplace=True)
```

[5]:
```
# Removing outliers
# Define a function to detect outliers using z-score
def detect_outliers(df, col):
    z_scores = (df[col] - df[col].mean()) / df[col].std()
    return df[abs(z_scores) < 3]
```

[6]:
```
#Removing outliers in Global_Sales column
data = data[data['Global_Sales'] < data['Global_Sales'].quantile(0.99)]
```

[7]:
```
#Convert Year to datetime format
data['Year'] = pd.to_datetime(data['Year'], format='%Y')
```

[8]:
```
print(data['Year'].info())
```

```
<class 'pandas.core.series.Series'>
Int64Index: 16128 entries, 163 to 16597
Series name: Year
Non-Null Count  Dtype
--------------  -----
16128 non-null  datetime64[ns]
dtypes: datetime64[ns](1)
memory usage: 252.0 KB
None
```

[9]:
```
# Apply outlier detection for sales columns
for col in ['NA_Sales', 'EU_Sales', 'JP_Sales', 'Other_Sales', 'Global_Sales']:
    data = detect_outliers(data, col)
```

**Descriptive Analysis:**

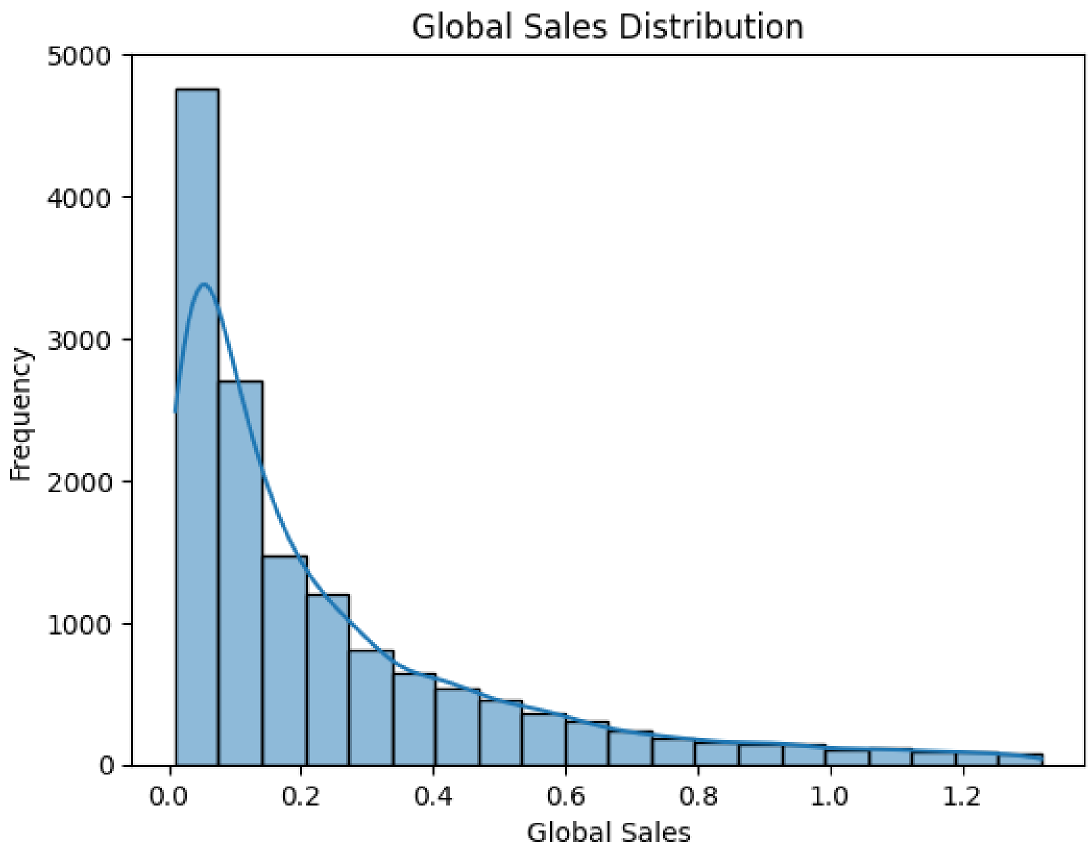
[10]:
```
# Summary statistics
print(data.describe())
```

```
               Rank      NA_Sales      EU_Sales      JP_Sales   Other_Sales  \
count  14598.000000  14598.000000  14598.000000  14598.000000  14598.000000
mean    9141.011851      0.128156      0.061558      0.037305      0.019860
```

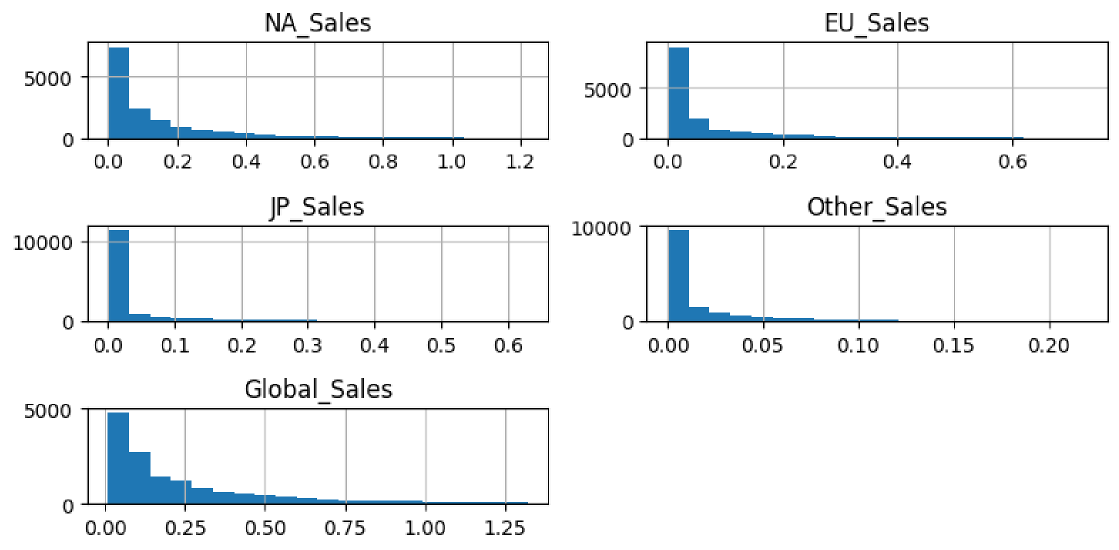
```
std     4313.324812      0.175197      0.102637      0.088822      0.032281
min     1484.000000      0.000000      0.000000      0.000000      0.000000
25%     5426.250000      0.000000      0.000000      0.000000      0.000000
50%     9157.500000      0.060000      0.020000      0.000000      0.010000
75%    12871.750000      0.180000      0.070000      0.030000      0.020000
max    16600.000000      1.220000      0.730000      0.630000      0.220000

       Global_Sales
count  14598.000000
mean       0.247189
std        0.273519
min        0.010000
25%        0.050000
50%        0.140000
75%        0.340000
max        1.320000
```

[11]:
```python
# Visualize data distributions
sns.histplot(data['Global_Sales'], bins=20, kde=True)
plt.title('Global Sales Distribution')
plt.xlabel('Global Sales')
plt.ylabel('Frequency')
plt.show()
```



[12]:
```
# Visualize data distributions
data[['NA_Sales', 'EU_Sales', 'JP_Sales', 'Other_Sales', 'Global_Sales']].
 ↪hist(bins=20, figsize=(8, 4))
plt.tight_layout()
plt.show()
```



**Segmentation and Profiling:**

[13]:
```
#segmenting by Genre
genre_groups = data.groupby('Genre')
for genre, group_data in genre_groups:
    print(f"Genre: {genre}")
    print(group_data.describe())
```

```
Genre: Action
               Rank     NA_Sales     EU_Sales     JP_Sales  Other_Sales  \
count   2924.000000  2924.000000  2924.000000  2924.000000  2924.000000
mean    8774.075581     0.139733     0.073399     0.028782     0.023697
std     4295.772132     0.170342     0.108401     0.073346     0.035925
min     1484.000000     0.000000     0.000000     0.000000     0.000000
25%     5058.500000     0.010000     0.000000     0.000000     0.000000
50%     8508.500000     0.080000     0.030000     0.000000     0.010000
75%    12448.500000     0.200000     0.100000     0.020000     0.030000
max    16592.000000     1.120000     0.640000     0.620000     0.220000

       Global_Sales
count   2924.000000
mean       0.265944
std        0.281355
min        0.010000
25%        0.060000
50%        0.160000
75%        0.380000
max        1.320000
Genre: Adventure
```

```
               Rank     NA_Sales     EU_Sales     JP_Sales  Other_Sales  \
count   1242.000000  1242.000000  1242.000000  1242.000000  1242.000000
mean   11812.929952     0.053430     0.029428     0.033768     0.008559
std     3955.662886     0.105938     0.071125     0.067431     0.019881
min     1508.000000     0.000000     0.000000     0.000000     0.000000
25%     9003.750000     0.000000     0.000000     0.000000     0.000000
50%    13018.000000     0.000000     0.000000     0.010000     0.000000
75%    15179.500000     0.070000     0.020000     0.030000     0.010000
max    16594.000000     0.760000     0.670000     0.620000     0.220000

       Global_Sales
count   1242.000000
mean       0.125386
std        0.185107
min        0.010000
25%        0.020000
50%        0.050000
75%        0.140000
max        1.310000
Genre: Fighting
               Rank    NA_Sales    EU_Sales    JP_Sales  Other_Sales  \
count    736.000000  736.000000  736.000000  736.000000   736.000000
mean    8525.069293    0.126128    0.059280    0.066943     0.020897
std     4125.455638    0.167847    0.095814    0.112706     0.034266
min     1494.000000    0.000000    0.000000    0.000000     0.000000
25%     5084.000000    0.000000    0.000000    0.000000     0.000000
50%     8569.500000    0.060000    0.020000    0.010000     0.010000
75%    11875.250000    0.190000    0.080000    0.082500     0.030000
max    16566.000000    0.880000    0.520000    0.630000     0.200000

       Global_Sales
count    736.000000
mean       0.273220
std        0.279208
min        0.010000
25%        0.070000
50%        0.160000
75%        0.372500
max        1.320000
Genre: Misc
               Rank     NA_Sales     EU_Sales     JP_Sales  Other_Sales  \
count   1531.000000  1531.000000  1531.000000  1531.000000  1531.000000
mean    9321.558459     0.123494     0.052012     0.030261     0.017975
std     4099.055909     0.170070     0.096898     0.078750     0.029545
min     1503.000000     0.000000     0.000000     0.000000     0.000000
25%     5949.500000     0.000000     0.000000     0.000000     0.000000
50%     9381.000000     0.070000     0.010000     0.000000     0.010000
75%    12745.000000     0.170000     0.060000     0.020000     0.020000
```

```
max    16545.000000     1.220000     0.690000     0.630000     0.220000

       Global_Sales
count   1531.000000
mean       0.224180
std        0.245787
min        0.010000
25%        0.060000
50%        0.130000
75%        0.295000
max        1.310000
Genre: Platform
               Rank    NA_Sales    EU_Sales    JP_Sales  Other_Sales  \
count    721.000000  721.000000  721.000000  721.000000   721.000000
mean    8176.313454    0.177712    0.078738    0.031276     0.021096
std     4236.817262    0.199132    0.109028    0.089804     0.032068
min     1507.000000    0.000000    0.000000    0.000000     0.000000
25%     4407.000000    0.040000    0.010000    0.000000     0.000000
50%     8107.000000    0.100000    0.040000    0.000000     0.010000
75%    11665.000000    0.250000    0.100000    0.000000     0.030000
max    16600.000000    1.220000    0.640000    0.620000     0.220000

       Global_Sales
count    721.000000
mean       0.309293
std        0.315986
min        0.010000
25%        0.080000
50%        0.180000
75%        0.450000
max        1.310000
Genre: Puzzle
               Rank    NA_Sales    EU_Sales    JP_Sales  Other_Sales  \
count    519.000000  519.000000  519.000000  519.000000   519.000000
mean   10479.221580    0.092736    0.035645    0.037823     0.010039
std     4106.489981    0.135881    0.075221    0.100443     0.017531
min     1492.000000    0.000000    0.000000    0.000000     0.000000
25%     7250.000000    0.010000    0.000000    0.000000     0.000000
50%    11192.000000    0.040000    0.000000    0.000000     0.000000
75%    13861.000000    0.110000    0.030000    0.000000     0.010000
max    16599.000000    0.770000    0.490000    0.630000     0.140000

       Global_Sales
count    519.000000
mean       0.177148
std        0.229573
min        0.010000
25%        0.040000
```

```
50%         0.090000
75%         0.220000
max         1.320000
Genre: Racing
                Rank     NA_Sales     EU_Sales     JP_Sales  Other_Sales  \
count   1094.000000  1094.000000  1094.000000  1094.000000  1094.000000
mean    8753.384826     0.154150     0.085878     0.010347     0.024698
std     4323.855481     0.192325     0.120666     0.045877     0.037629
min     1498.000000     0.000000     0.000000     0.000000     0.000000
25%     5018.250000     0.030000     0.010000     0.000000     0.000000
50%     8722.000000     0.080000     0.040000     0.000000     0.010000
75%    12399.250000     0.210000     0.110000     0.000000     0.030000
max    16598.000000     1.220000     0.710000     0.550000     0.220000

       Global_Sales
count   1094.000000
mean       0.275402
std        0.301203
min        0.010000
25%        0.060000
50%        0.150000
75%        0.380000
max        1.320000
Genre: Role-Playing
                Rank     NA_Sales     EU_Sales     JP_Sales  Other_Sales  \
count   1277.000000  1277.000000  1277.000000  1277.000000  1277.000000
mean    9121.296006     0.088121     0.042913     0.092592     0.015936
std     4163.412555     0.141553     0.080761     0.126676     0.027125
min     1487.000000     0.000000     0.000000     0.000000     0.000000
25%     5780.000000     0.000000     0.000000     0.000000     0.000000
50%     9089.000000     0.030000     0.000000     0.040000     0.010000
75%    12706.000000     0.120000     0.050000     0.130000     0.020000
max    16593.000000     1.030000     0.630000     0.630000     0.220000

       Global_Sales
count   1277.000000
mean       0.239632
std        0.261328
min        0.010000
25%        0.060000
50%        0.140000
75%        0.310000
max        1.320000
Genre: Shooter
                Rank     NA_Sales     EU_Sales     JP_Sales  Other_Sales  \
count   1088.000000  1088.000000  1088.000000  1088.000000  1088.000000
mean    8508.290441     0.164210     0.088640     0.016664     0.027923
std     4456.644484     0.199064     0.116082     0.058354     0.040260
```

```
min     1496.000000     0.000000     0.000000     0.000000     0.000000
25%     4340.250000     0.030000     0.010000     0.000000     0.000000
50%     8433.000000     0.090000     0.040000     0.000000     0.010000
75%    12356.250000     0.240000     0.130000     0.000000     0.040000
max    16597.000000     1.180000     0.690000     0.620000     0.220000

       Global_Sales
count   1088.000000
mean       0.297886
std        0.311112
min        0.010000
25%        0.060000
50%        0.170000
75%        0.450000
max        1.320000
Genre: Simulation
               Rank    NA_Sales    EU_Sales    JP_Sales  Other_Sales  \
count    769.000000  769.000000  769.000000  769.000000   769.000000
mean    9340.036411    0.131118    0.049844    0.035644     0.018244
std     4343.632544    0.175531    0.093689    0.090609     0.027912
min     1542.000000    0.000000    0.000000    0.000000     0.000000
25%     5478.000000    0.000000    0.000000    0.000000     0.000000
50%     9522.000000    0.060000    0.010000    0.000000     0.010000
75%    13100.000000    0.200000    0.050000    0.020000     0.020000
max    16595.000000    1.220000    0.640000    0.620000     0.220000

       Global_Sales
count    769.000000
mean       0.235150
std        0.260909
min        0.010000
25%        0.050000
50%        0.130000
75%        0.330000
max        1.280000
Genre: Sports
               Rank     NA_Sales     EU_Sales     JP_Sales  Other_Sales  \
count   2058.000000  2058.000000  2058.000000  2058.000000  2058.000000
mean    8192.985909     0.164806     0.067804     0.032843     0.023022
std     4062.290062     0.202271     0.113203     0.087302     0.033264
min     1493.000000     0.000000     0.000000     0.000000     0.000000
25%     4733.500000     0.010000     0.000000     0.000000     0.000000
50%     7917.500000     0.090000     0.020000     0.000000     0.010000
75%    11387.000000     0.230000     0.080000     0.000000     0.030000
max    16590.000000     1.130000     0.730000     0.590000     0.220000

       Global_Sales
count   2058.000000
```

```
mean       0.288664
std        0.279430
min        0.010000
25%        0.080000
50%        0.190000
75%        0.410000
max        1.320000
Genre: Strategy
               Rank    NA_Sales    EU_Sales    JP_Sales  Other_Sales  \
count    639.000000  639.000000  639.000000  639.000000   639.000000
mean   10499.325509    0.059296    0.040469    0.059484     0.011768
std     4135.705678    0.122313    0.077164    0.118624     0.021250
min     1509.000000    0.000000    0.000000    0.000000     0.000000
25%     7069.000000    0.000000    0.000000    0.000000     0.000000
50%    11097.000000    0.000000    0.010000    0.000000     0.000000
75%    14106.500000    0.070000    0.040000    0.060000     0.010000
max    16569.000000    1.190000    0.640000    0.600000     0.170000

       Global_Sales
count    639.000000
mean       0.171471
std        0.208228
min        0.010000
25%        0.030000
50%        0.090000
75%        0.230000
max        1.310000
```

**Correlation and Trends:**

```python
# Correlation matrix
correlation_matrix = data.corr()
print(correlation_matrix)
```

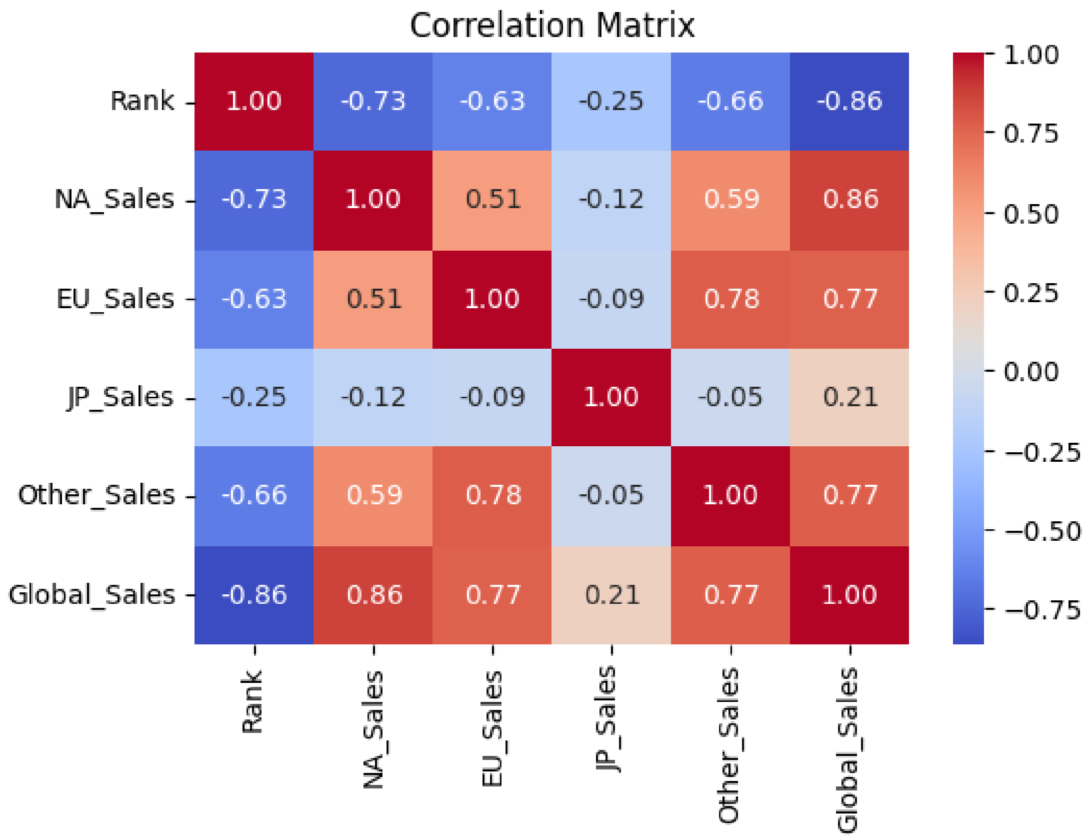
```
                  Rank  NA_Sales  EU_Sales  JP_Sales  Other_Sales  \
Rank          1.000000 -0.732888 -0.629139 -0.248371    -0.656816
NA_Sales     -0.732888  1.000000  0.513408 -0.118101     0.593643
EU_Sales     -0.629139  0.513408  1.000000 -0.087906     0.780590
JP_Sales     -0.248371 -0.118101 -0.087906  1.000000    -0.050698
Other_Sales  -0.656816  0.593643  0.780590 -0.050698     1.000000
Global_Sales -0.862347  0.864086  0.767183  0.209550     0.773436

              Global_Sales
Rank             -0.862347
NA_Sales          0.864086
EU_Sales          0.767183
JP_Sales          0.209550
Other_Sales       0.773436
```

```
Global_Sales      1.000000

<ipython-input-14-dd3106a641cc>:2: FutureWarning: The default value of
numeric_only in DataFrame.corr is deprecated. In a future version, it will
default to False. Select only valid columns or specify the value of numeric_only
to silence this warning.
  correlation_matrix = data.corr()
```

[15]:
```
# Visualize correlations
plt.figure(figsize=(6, 4))
sns.heatmap(correlation_matrix, annot=True, cmap='coolwarm', fmt=".2f")
plt.title('Correlation Matrix')
plt.show()
```


Correlation Matrix
Rank
NA_Sales
EU_Sales
JP_Sales
Other_Sales
Global_Sales