In [1]:

```
from mpl_toolkits.mplot3d import Axes3D
from sklearn.preprocessing import StandardScaler
import matplotlib.pyplot as plt # plotting
import numpy as np # linear algebra
import os # accessing directory structure
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)
```

In [2]:

```
for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

```
/kaggle/input/movies_metadata.csv
/kaggle/input/ratings.csv
/kaggle/input/links_small.csv
/kaggle/input/links.csv
/kaggle/input/credits.csv
/kaggle/input/ratings_small.csv
/kaggle/input/keywords.csv
```

In [ ]:

```
# Distribution graphs (histogram/bar graph) of column data
def plotPerColumnDistribution(df, nGraphShown, nGraphPerRow):
    nunique = df.nunique()
    df = df[[col for col in df if nunique[col] > 1 and nunique[col] < 50]] # For displaying purposes, pick columns that have betw
    nRow, nCol = df.shape
    columnNames = list(df)
    nGraphRow = (nCol + nGraphPerRow - 1) / nGraphPerRow
    plt.figure(num = None, figsize = (6 * nGraphPerRow, 8 * nGraphRow), dpi = 80, facecolor = 'w', edgecolor = 'k')
    for i in range(min(nCol, nGraphShown)):
        plt.subplot(nGraphRow, nGraphPerRow, i + 1)
        columnDf = df.iloc[:, i]
        if (not np.issubdtype(type(columnDf.iloc[0]), np.number)):
            valueCounts = columnDf.value_counts()
            valueCounts.plot.bar()
        else:
            columnDf.hist()
        plt.ylabel('counts')
        plt.xticks(rotation = 90)
        plt.title(f'{columnNames[i]} (column {i})')
    plt.tight_layout(pad = 1.0, w_pad = 1.0, h_pad = 1.0)
    plt.show()
```

In [4]:

```
# Correlation matrix
def plotCorrelationMatrix(df, graphWidth):
    filename = df.dataframeName
    df = df.dropna('columns') # drop columns with NaN
    df = df[[col for col in df if df[col].nunique() > 1]] # keep columns
where there are more than 1 unique values
    if df.shape[1] < 2:
        print(f'No correlation plots shown: The number of non-NaN or con
stant columns ({df.shape[1]}) is less than 2')
        return
    corr = df.corr()
    plt.figure(num=None, figsize=(graphWidth, graphWidth), dpi=80, facec
olor='w', edgecolor='k')
    corrMat = plt.matshow(corr, fignum = 1)
```

```
plt.xticks(range(len(corr.columns)), corr.columns, rotation=90)
plt.yticks(range(len(corr.columns)), corr.columns)
plt.gca().xaxis.tick_bottom()
plt.colorbar(corrMat)
plt.title(f'Correlation Matrix for {filename}', fontsize=15)
plt.show()
```

In [ ]:

```
# Scatter and density plots
def plotScatterMatrix(df, plotSize, textSize):
    df = df.select_dtypes(include =[np.number]) # keep only numerical columns
    # Remove rows and columns that would lead to df being singular
    df = df.dropna('columns')
    df = df[[col for col in df if df[col].nunique() > 1]] # keep columns where there are more than 1 unique values
    columnNames = list(df)
    if len(columnNames) > 10: # reduce the number of columns for matrix inversion of kernel density plots
        columnNames = columnNames[:10]
    df = df[columnNames]
    ax = pd.plotting.scatter_matrix(df, alpha=0.75, figsize=[plotSize, plotSize], diagonal='kde')
    corrs = df.corr().values
    for i, j in zip(*plt.np.triu_indices_from(ax, k = 1)):
        ax[i, j].annotate('Corr. coef = %.3f' % corrs[i, j], (0.8, 0.2), xycoords='axes fraction', ha='center', va='center', size
    plt.suptitle('Scatter and Density Plot')
    plt.show()
```

In [6]:

```
nRowsRead = 1000 # specify 'None' if want to read whole file
# credits.csv may have more rows in reality, but we are only loading/previ
ewing the first 1000 rows
df1 = pd.read_csv('/kaggle/input/credits.csv', delimiter=',', nrows = nR
owsRead)
df1.dataframeName = 'credits.csv'
nRow, nCol = df1.shape
print(f'There are {nRow} rows and {nCol} columns')
```

```
There are 1000 rows and 3 columns
```

In [7]:

```
df1.head(5)
```

Out[7]:

	cast	crew	id
0	[{'cast_id': 14, 'character': 'Woody (voice)',...	[{'credit_id': '52fe4284c3a36847f8024f49', 'de...	862
1	[{'cast_id': 1, 'character': 'Alan Parrish', '...	[{'credit_id': '52fe44bfc3a36847f80a7cd1', 'de...	8844
2	[{'cast_id': 2, 'character': 'Max Goldman', 'c...	[{'credit_id': '52fe466a9251416c75077a89', 'de...	15602
3	[{'cast_id': 1, 'character': "Savannah 'Vannah...	[{'credit_id': '52fe44779251416c91011acb', 'de...	31357
4	[{'cast_id': 1, 'character': 'George Banks', '...	[{'credit_id': '52fe44959251416c75039ed7', 'de...	11862

In [8]:
```
plotPerColumnDistribution(df1, 10, 5)
```

```
<Figure size 2400x512 with 0 Axes>
```

In [9]:

```
nRowsRead = 1000 # specify 'None' if want to read whole file
# keywords.csv may have more rows in reality, but we are only loading/prev
iewing the first 1000 rows
df2 = pd.read_csv('/kaggle/input/keywords.csv', delimiter=',', nrows = n
RowsRead)
df2.dataframeName = 'keywords.csv'
nRow, nCol = df2.shape
print(f'There are {nRow} rows and {nCol} columns')
```

```
There are 1000 rows and 2 columns
```

In [10]:

```
df2.head(5)
```

Out[10]:

	id	keywords
0	862	[{'id': 931, 'name': 'jealousy'}, {'id': 4290,...
1	8844	[{'id': 10090, 'name': 'board game'}, {'id': 1...
2	15602	[{'id': 1495, 'name': 'fishing'}, {'id': 12392...
3	31357	[{'id': 818, 'name': 'based on novel'}, {'id':...
4	11862	[{'id': 1009, 'name': 'baby'}, {'id': 1599, 'n...

In [11]:

```
plotPerColumnDistribution(df2, 10, 5)
```

```
<Figure size 2400x512 with 0 Axes>
```

In [12]:

```
nRowsRead = 1000 # specify 'None' if want to read whole file
# links.csv may have more rows in reality, but we are only loading/preview
ing the first 1000 rows
df3 = pd.read_csv('/kaggle/input/links.csv', delimiter=',', nrows = nRow
sRead)
df3.dataframeName = 'links.csv'
nRow, nCol = df3.shape
print(f'There are {nRow} rows and {nCol} columns')
```

```
There are 1000 rows and 3 columns
```

In [13]:
```
df3.head(5)
```

Out[13]:

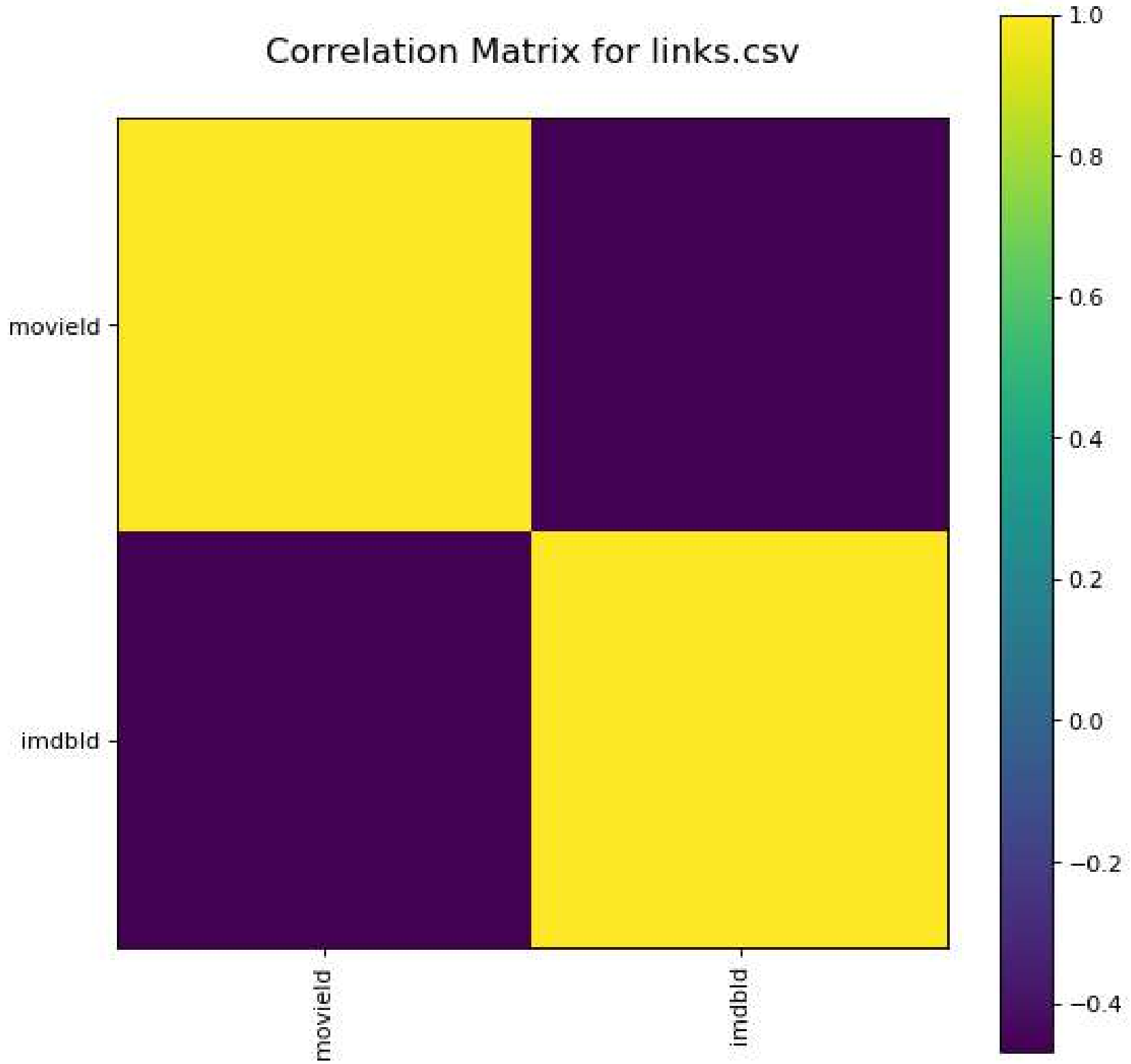
	movieId	imdbId	tmdbId
0	1	114709	862.0
1	2	113497	8844.0
2	3	113228	15602.0
3	4	114885	31357.0
4	5	113041	11862.0

In [14]:
```
plotPerColumnDistribution(df3, 10, 5)
```

```
<Figure size 2400x512 with 0 Axes>
```

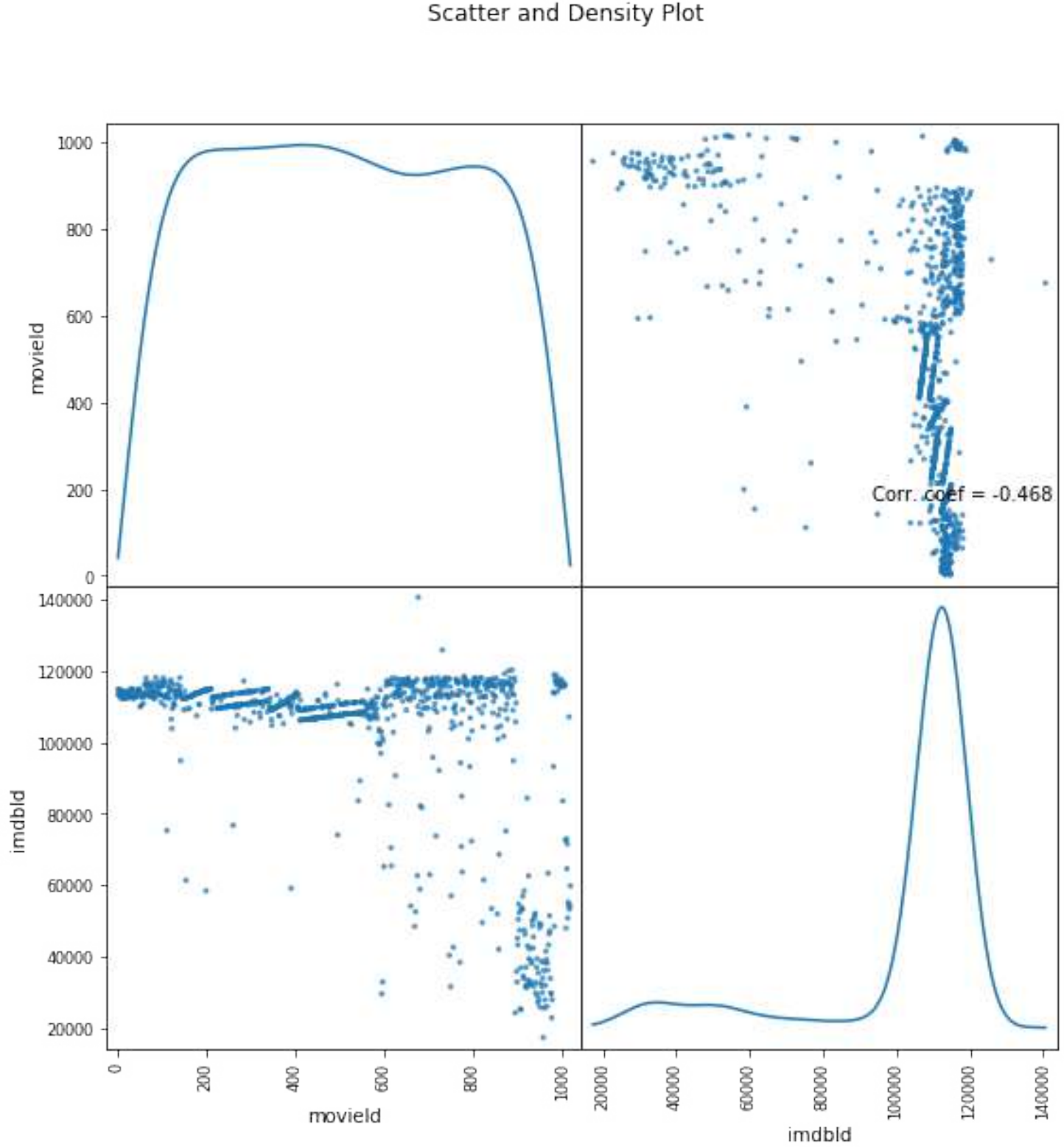
In [15]:

```
plotCorrelationMatrix(df3, 8)
```


Correlation Matrix for links.csv
movieId
imdbId
movieId
imdbId
1.0
0.8
0.6
0.4
0.2
0.0
−0.2
−0.4

In [16]:

```
plotScatterMatrix(df3, 9, 10)
```


Scatter and Density Plot
movieId
imdbId
Corr. coef = -0.468
movieId
imdbId