gbgoeild2

June 20, 2024

[1]:
```
# This Python 3 environment comes with many helpful analytics libraries␣
 ↪installed
# It is defined by the kaggle/python Docker image: https://github.com/kaggle/
 ↪docker-python
# For example, here's several helpful packages to load

import numpy as np # linear algebra
import pandas as pd # data processing, CSV file I/O (e.g. pd.read_csv)

# Input data files are available in the read-only "../input/" directory
# For example, running this (by clicking run or pressing Shift+Enter) will list␣
 ↪all files under the input directory

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

# You can write up to 20GB to the current directory (/kaggle/working/) that␣
 ↪gets preserved as output when you create a version using "Save & Run All"
# You can also write temporary files to /kaggle/temp/, but they won't be saved␣
 ↪outside of the current session
```

```
/kaggle/input/lottery-uk-euromillions-all-data/eum_df.csv
/kaggle/input/lottery-uk-euromillions-all-data/lottery-uk-euromillions-all-data-
eda.ipynb
```

[18]:
```
import base64
import io
import os

from IPython.display import display, HTML
import matplotlib.pyplot as plt
import seaborn as sns
```

[19]:
```
df=pd.read_csv("/kaggle/input/lottery-uk-euromillions-all-data/eum_df.csv")
```

[20]:
```
df.head()
```

[20]:
```
   DrawNumber    DrawDate  Ball1  Ball2  Ball3  Ball4  Ball5  LuckyStar1  \
0        1747  18-06-2024     34     11     36     33      3          12
1        1746  14-06-2024     13      2     16     24     32           7
2        1745  11-06-2024     48     15      7     34     45           7
3        1744  07-06-2024     30     26     15     37     16           8
4        1743  04-06-2024     43      7      6      9     14           4

   LuckyStar2 UKMillionaireMaker  … Match_2_and_1_Star_TotalWinners  \
0           1      ['TKJZ31572']  …                          553256
1           1      ['TJHX51504']  …                          848501
2           9      ['MHHQ42012']  …                          616187
3           5      ['HGGP69328']  …                          885563
4           3      ['XFGJ15568']  …                          473384

  Match_2_UKWinners  Match_2_PrizePerWinner  Match_2_UKPrizeFund  \
0          441863.0                     2.7            1193030.1
1          520586.0                     2.6            1353523.6
2          353357.0                     3.0            1060071.0
3          514213.0                     2.9            1491217.7
4          246089.0                     2.6             639831.4

   Match_2_TotalWinners  Totals_UKWinners  Totals_UKPrizeFund  \
0             1534807.0            697016           3129209.8
1             1847351.0            875922           3211204.5
2             1209516.0            619887           2324229.6
3             1876170.0            893735           3479475.8
4              948932.0            430410           1673301.4

   Totals_TotalWinners  JackpotWinner_country  TicketPrice
0              2401278                    NaN          2.5
1              3168432                    NaN          2.5
2              2176357                    NaN          2.5
3              3309518                    NaN          2.5
4              1648077                    NaN          2.5

[5 rows x 70 columns]
```

[21]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1747 entries, 0 to 1746
Data columns (total 70 columns):
 #   Column                              Non-Null Count  Dtype
---  ------                              --------------  -----
 0   DrawNumber                          1747 non-null   int64
```

```
1   DrawDate                            1747 non-null   object
2   Ball1                               1747 non-null   int64
3   Ball2                               1747 non-null   int64
4   Ball3                               1747 non-null   int64
5   Ball4                               1747 non-null   int64
6   Ball5                               1747 non-null   int64
7   LuckyStar1                          1747 non-null   int64
8   LuckyStar2                          1747 non-null   int64
9   UKMillionaireMaker                  1747 non-null   object
10  BallSet                             1741 non-null   object
11  DrawMachine                         1741 non-null   object
12  Match_5_and_2_Stars_roll            1747 non-null   int64
13  Match_5_and_2_Stars_UKWinners       1747 non-null   int64
14  Match_5_and_2_Stars_PrizePerWinner  1747 non-null   float64
15  Match_5_and_2_Stars_UKPrizeFund     1747 non-null   float64
16  Match_5_and_2_Stars_TotalWinners    1747 non-null   int64
17  Match_5_and_1_Star_UKWinners        1747 non-null   int64
18  Match_5_and_1_Star_PrizePerWinner   1747 non-null   float64
19  Match_5_and_1_Star_UKPrizeFund      1747 non-null   float64
20  Match_5_and_1_Star_TotalWinners     1747 non-null   int64
21  Match_5_UKWinners                   1747 non-null   int64
22  Match_5_PrizePerWinner              1747 non-null   float64
23  Match_5_UKPrizeFund                 1747 non-null   float64
24  Match_5_TotalWinners                1747 non-null   int64
25  Match_4_and_2_Stars_UKWinners       1747 non-null   int64
26  Match_4_and_2_Stars_PrizePerWinner  1747 non-null   float64
27  Match_4_and_2_Stars_UKPrizeFund     1747 non-null   float64
28  Match_4_and_2_Stars_TotalWinners    1747 non-null   int64
29  Match_4_and_1_Star_UKWinners        1747 non-null   int64
30  Match_4_and_1_Star_PrizePerWinner   1747 non-null   float64
31  Match_4_and_1_Star_UKPrizeFund      1747 non-null   float64
32  Match_4_and_1_Star_TotalWinners     1747 non-null   int64
33  Match_3_and_2_Stars_UKWinners       1747 non-null   int64
34  Match_3_and_2_Stars_PrizePerWinner  1747 non-null   float64
35  Match_3_and_2_Stars_UKPrizeFund     1747 non-null   float64
36  Match_3_and_2_Stars_TotalWinners    1747 non-null   int64
37  Match_4_UKWinners                   1747 non-null   int64
38  Match_4_PrizePerWinner              1747 non-null   float64
39  Match_4_UKPrizeFund                 1747 non-null   float64
40  Match_4_TotalWinners                1747 non-null   int64
41  Match_2_and_2_Stars_UKWinners       1747 non-null   int64
42  Match_2_and_2_Stars_PrizePerWinner  1747 non-null   float64
43  Match_2_and_2_Stars_UKPrizeFund     1747 non-null   float64
44  Match_2_and_2_Stars_TotalWinners    1747 non-null   int64
45  Match_3_and_1_Star_UKWinners        1747 non-null   int64
46  Match_3_and_1_Star_PrizePerWinner   1747 non-null   float64
47  Match_3_and_1_Star_UKPrizeFund      1747 non-null   float64
48  Match_3_and_1_Star_TotalWinners     1747 non-null   int64
```

```
 49  Match_3_UKWinners                    1747 non-null   int64
 50  Match_3_PrizePerWinner               1747 non-null   float64
 51  Match_3_UKPrizeFund                  1747 non-null   float64
 52  Match_3_TotalWinners                 1747 non-null   int64
 53  Match_1_and_2_Stars_UKWinners        1747 non-null   int64
 54  Match_1_and_2_Stars_PrizePerWinner   1747 non-null   float64
 55  Match_1_and_2_Stars_UKPrizeFund      1747 non-null   float64
 56  Match_1_and_2_Stars_TotalWinners     1747 non-null   int64
 57  Match_2_and_1_Star_UKWinners         1747 non-null   int64
 58  Match_2_and_1_Star_PrizePerWinner    1747 non-null   float64
 59  Match_2_and_1_Star_UKPrizeFund       1747 non-null   float64
 60  Match_2_and_1_Star_TotalWinners      1747 non-null   int64
 61  Match_2_UKWinners                    1369 non-null   float64
 62  Match_2_PrizePerWinner               1369 non-null   float64
 63  Match_2_UKPrizeFund                  1369 non-null   float64
 64  Match_2_TotalWinners                 1369 non-null   float64
 65  Totals_UKWinners                     1747 non-null   int64
 66  Totals_UKPrizeFund                   1747 non-null   float64
 67  Totals_TotalWinners                  1747 non-null   int64
 68  JackpotWinner_country                417 non-null    object
 69  TicketPrice                          1747 non-null   float64
dtypes: float64(30), int64(35), object(5)
memory usage: 955.5+ KB
```

[22]:
```
df.isnull().sum()
```

[22]:
```
DrawNumber                0
DrawDate                  0
Ball1                     0
Ball2                     0
Ball3                     0
                         ..
Totals_UKWinners          0
Totals_UKPrizeFund        0
Totals_TotalWinners       0
JackpotWinner_country    1330
TicketPrice               0
Length: 70, dtype: int64
```

[23]:
```
df.describe()
```

[23]:

	DrawNumber	Ball1	Ball2	Ball3	Ball4	\
count	1747.000000	1747.000000	1747.000000	1747.000000	1747.000000	
mean	874.000000	25.625072	24.960504	25.895249	25.367487	
std	504.459777	14.377767	14.361881	14.210249	14.462181	
min	1.000000	1.000000	1.000000	1.000000	1.000000	
25%	437.500000	13.000000	13.000000	13.000000	13.000000	

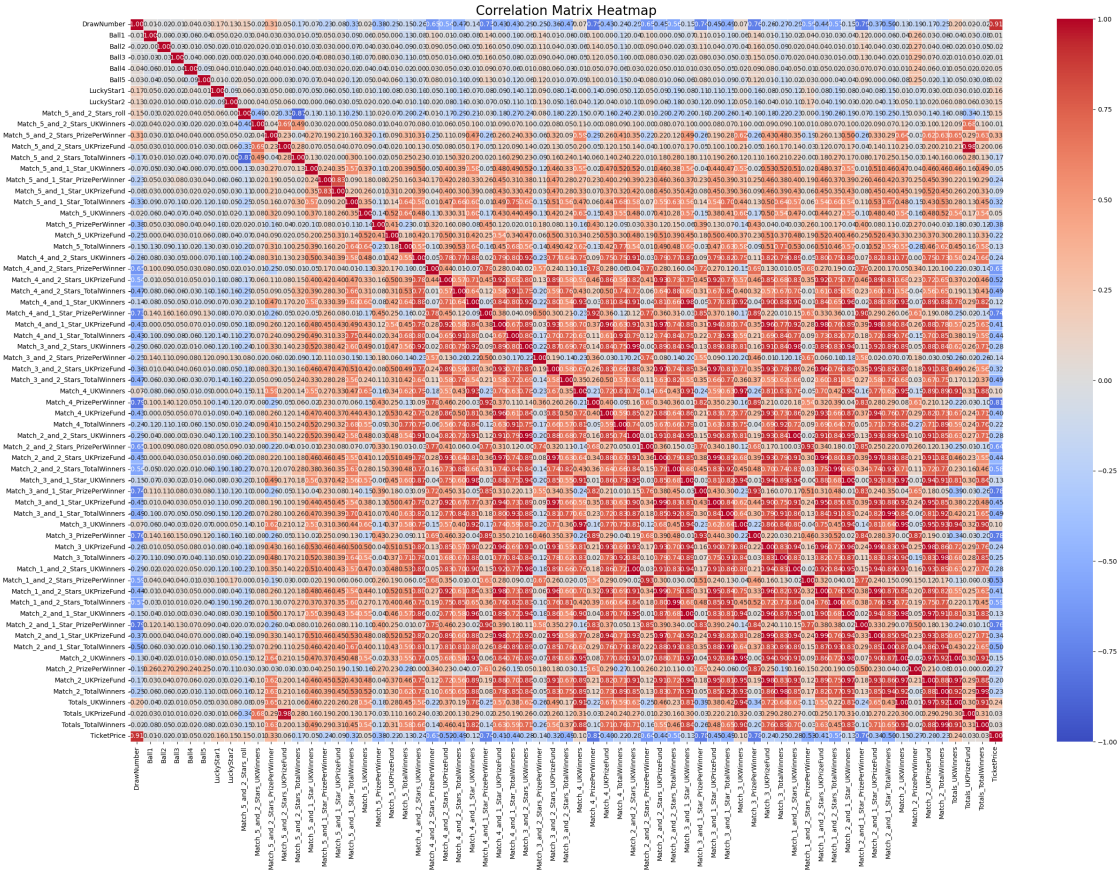
```
50%     874.000000    26.000000    25.000000    26.000000    25.000000
75%    1310.500000    38.000000    37.500000    38.000000    38.000000
max    1747.000000    50.000000    50.000000    50.000000    50.000000

             Ball5   LuckyStar1   LuckyStar2  Match_5_and_2_Stars_roll  \
count  1747.000000  1747.000000  1747.000000               1747.000000
mean     25.400114     6.081282     5.954207                  0.759015
std      14.371071     3.257950     3.250019                  0.427803
min       1.000000     1.000000     1.000000                  0.000000
25%      13.000000     3.000000     3.000000                  1.000000
50%      25.000000     6.000000     6.000000                  1.000000
75%      38.000000     9.000000     9.000000                  1.000000
max      50.000000    12.000000    12.000000                  1.000000

       Match_5_and_2_Stars_UKWinners  …  Match_2_and_1_Star_UKPrizeFund  \
count                    1747.000000  …                    1.747000e+03
mean                        0.059531  …                    8.322926e+05
std                         0.261952  …                    5.902284e+05
min                         0.000000  …                    1.721475e+05
25%                         0.000000  …                    4.889375e+05
50%                         0.000000  …                    6.725268e+05
75%                         0.000000  …                    1.012473e+06
max                         3.000000  …                    6.798180e+06

       Match_2_and_1_Star_TotalWinners  Match_2_UKWinners  \
count                     1.747000e+03       1.369000e+03
mean                      6.928464e+05       3.216798e+05
std                       3.671390e+05       1.528710e+05
min                       1.811980e+05       1.469070e+05
25%                       4.428090e+05       2.217600e+05
50%                       5.747620e+05       2.811580e+05
75%                       8.680585e+05       3.694700e+05
max                       3.747671e+06       2.368907e+06

       Match_2_PrizePerWinner  Match_2_UKPrizeFund  Match_2_TotalWinners  \
count             1369.000000         1.369000e+03          1.369000e+03
mean                 2.777429         8.924514e+05          1.256562e+06
std                  0.312124         4.412292e+05          4.844003e+05
min                  1.800000         3.386980e+05          4.882450e+05
25%                  2.600000         6.163102e+05          9.159120e+05
50%                  2.800000         7.727885e+05          1.132120e+06
75%                  3.000000         1.027521e+06          1.446799e+06
max                  3.800000         6.632940e+06          4.598171e+06

       Totals_UKWinners  Totals_UKPrizeFund  Totals_TotalWinners  TicketPrice
count      1.747000e+03        1.747000e+03         1.747000e+03  1747.000000
mean       4.967537e+05        5.889056e+06         2.079965e+06     2.145106
```

```
std     2.817519e+05    1.693941e+07    8.538112e+05    0.370725
min     4.078800e+04    4.148861e+05    4.099370e+05    1.500000
25%     3.386550e+05    1.747638e+06    1.541699e+06    2.000000
50%     4.371740e+05    2.338868e+06    1.876274e+06    2.000000
75%     6.003570e+05    3.495378e+06    2.401997e+06    2.500000
max     4.125489e+06    1.998566e+08    7.746365e+06    2.500000

[8 rows x 65 columns]
```

[24]:
```
df = df.drop(['DrawDate'], axis=1)
```

[25]:
```
df2 = df.select_dtypes(exclude=['object'])
```

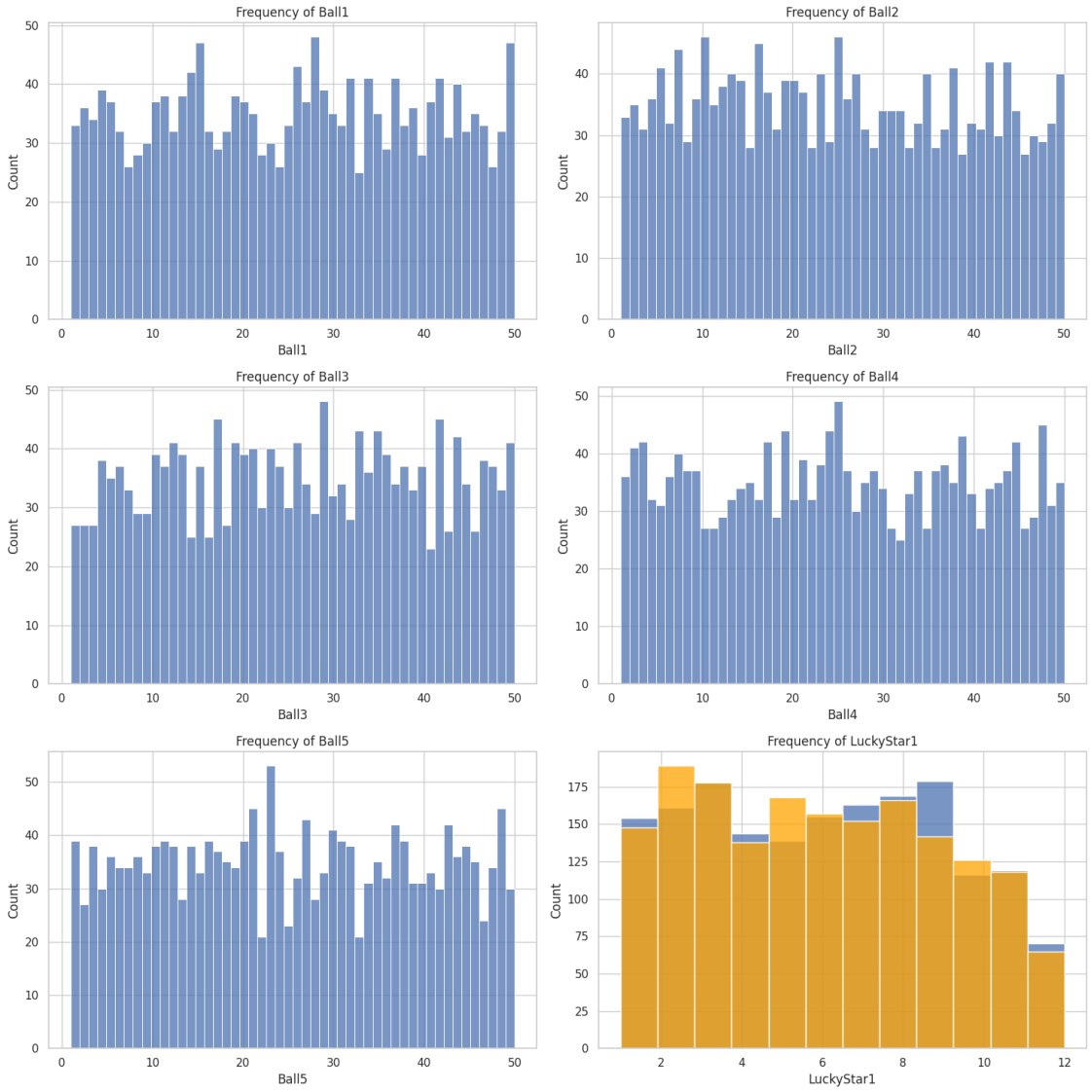
[28]:
```
correlation_matrix = df2.corr()

# Set up the matplotlib figure
plt.figure(figsize=(30, 20))

# Draw the heatmap with the correlation matrix
sns.heatmap(correlation_matrix, annot=True, fmt='.2f', cmap='coolwarm',␣
 ↪vmin=-1, vmax=1, linewidths=0.5)

# Set title
plt.title('Correlation Matrix Heatmap', size=20)

# Show the plot
plt.show()
```



[31]:
```
sns.set(style="whitegrid")

# Plot the frequency of the main balls
fig, axs = plt.subplots(3, 2, figsize=(15, 15))

sns.histplot(df['Ball1'], bins=50, kde=False, ax=axs[0, 0]).
 ↪set(title='Frequency of Ball1')
sns.histplot(df['Ball2'], bins=50, kde=False, ax=axs[0, 1]).
 ↪set(title='Frequency of Ball2')
sns.histplot(df['Ball3'], bins=50, kde=False, ax=axs[1, 0]).
 ↪set(title='Frequency of Ball3')
sns.histplot(df['Ball4'], bins=50, kde=False, ax=axs[1, 1]).
 ↪set(title='Frequency of Ball4')
sns.histplot(df['Ball5'], bins=50, kde=False, ax=axs[2, 0]).
 ↪set(title='Frequency of Ball5')

# Plot the frequency of LuckyStars
sns.histplot(df['LuckyStar1'], bins=12, kde=False, ax=axs[2, 1]).
 ↪set(title='Frequency of LuckyStar1')
```

```python
sns.histplot(df['LuckyStar2'], bins=12, kde=False, ax=axs[2, 1], color="orange")

# Adjust layout
plt.tight_layout()
plt.show()
```

```
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119: FutureWarning:
use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119: FutureWarning:
use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119: FutureWarning:
use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119: FutureWarning:
use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119: FutureWarning:
use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119: FutureWarning:
use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
/opt/conda/lib/python3.10/site-packages/seaborn/_oldcore.py:1119: FutureWarning:
use_inf_as_na option is deprecated and will be removed in a future version.
Convert inf values to NaN before operating instead.
  with pd.option_context('mode.use_inf_as_na', True):
```



[34]:
```
# Handle missing values
df = df.fillna('0')  # Fill numerical missing values with the mean
df = df.fillna(df.mode().iloc[0])  # Fill categorical missing values with the␣
 ↪mode
```

[36]:
```
from sklearn.preprocessing import StandardScaler

# Select numerical columns
numerical_cols = df.select_dtypes(include=['float64', 'int64']).columns

# Standardize numerical features
scaler = StandardScaler()
df[numerical_cols] = scaler.fit_transform(df[numerical_cols])
```

```
print(df.head())
```

```
   DrawNumber     Ball1     Ball2     Ball3     Ball4     Ball5  LuckyStar1  \
0    1.731060  0.582658 -0.972331  0.711293  0.527908 -1.559141    1.817220
1    1.729077 -0.878348 -1.599169 -0.696545 -0.094583  0.459379    0.282073
2    1.727094  1.556663 -0.693736 -1.330072  0.597073  1.364234    0.282073
3    1.725111  0.304371  0.072400 -0.766937  0.804570 -0.654287    0.589103
4    1.723128  1.208804 -1.250926 -1.400464 -1.132068 -0.793495   -0.639015

   LuckyStar2 UKMillionaireMaker BallSet  … Match_2_and_1_Star_TotalWinners  \
0   -1.524799      ['TKJZ31572']      21  …                       -0.380320
1   -1.524799      ['TJHX51504']      21  …                        0.424088
2    0.937430      ['MHHQ42012']      21  …                       -0.208862
3   -0.293685      ['HGGP69328']      21  …                        0.525065
4   -0.909242      ['XFGJ15568']      21  …                       -0.597935

   Match_2_UKWinners  Match_2_PrizePerWinner  Match_2_UKPrizeFund  \
0           441863.0                     2.7            1193030.1
1           520586.0                     2.6            1353523.6
2           353357.0                     3.0            1060071.0
3           514213.0                     2.9            1491217.7
4           246089.0                     2.6             639831.4

   Match_2_TotalWinners  Totals_UKWinners  Totals_UKPrizeFund  \
0             1534807.0          0.710979           -0.162971
1             1847351.0          1.346138           -0.158129
2             1209516.0          0.437152           -0.210506
3             1876170.0          1.409378           -0.142288
4              948932.0         -0.235536           -0.248944

   Totals_TotalWinners  JackpotWinner_country  TicketPrice
0             0.376436                      0      0.95757
1             1.275198                      0      0.95757
2             0.112929                      0      0.95757
3             1.440488                      0      0.95757
4            -0.505980                      0      0.95757

[5 rows x 69 columns]
```

[39]:
```
# Identify categorical columns
categorical_cols = df.select_dtypes(include=['object']).columns

# Encode categorical features using One-Hot Encoding
df = pd.get_dummies(df, columns=categorical_cols, drop_first=True)

print(df.head())
```

```
   DrawNumber     Ball1     Ball2     Ball3     Ball4     Ball5  LuckyStar1  \
0    1.731060  0.582658 -0.972331  0.711293  0.527908 -1.559141    1.817220
1    1.729077 -0.878348 -1.599169 -0.696545 -0.094583  0.459379    0.282073
2    1.727094  1.556663 -0.693736 -1.330072  0.597073  1.364234    0.282073
3    1.725111  0.304371  0.072400 -0.766937  0.804570 -0.654287    0.589103
4    1.723128  1.208804 -1.250926 -1.400464 -1.132068 -0.793495   -0.639015

   LuckyStar2  Match_5_and_2_Stars_roll  Match_5_and_2_Stars_UKWinners  …  \
0   -1.524799                  0.563468                      -0.227323  …
1   -1.524799                  0.563468                      -0.227323  …
2    0.937430                  0.563468                      -0.227323  …
3   -0.293685                  0.563468                      -0.227323  …
4   -0.909242                  0.563468                      -0.227323  …

   JackpotWinner_country_['Portugal']  \
0                               False
1                               False
2                               False
3                               False
4                               False

   JackpotWinner_country_['Spain', 'France', 'Portugal']  \
0                                              False
1                                              False
2                                              False
3                                              False
4                                              False

   JackpotWinner_country_['Spain', 'Switzerland']  \
0                                           False
1                                           False
2                                           False
3                                           False
4                                           False

   JackpotWinner_country_['Spain', 'UK']  JackpotWinner_country_['Spain']  \
0                                  False                             False
1                                  False                             False
2                                  False                             False
3                                  False                             False
4                                  False                             False

   JackpotWinner_country_['Switzerland']  \
0                                  False
1                                  False
2                                  False
3                                  False
4                                  False
```

```
   JackpotWinner_country_['UK', 'France']  \
0                                   False
1                                   False
2                                   False
3                                   False
4                                   False

   JackpotWinner_country_['UK', 'Portugal', 'Spain']  \
0                                              False
1                                              False
2                                              False
3                                              False
4                                              False

   JackpotWinner_country_['UK', 'Portugal', 'Switzerland']  \
0                                              False
1                                              False
2                                              False
3                                              False
4                                              False

   JackpotWinner_country_['UK']
0                         False
1                         False
2                         False
3                         False
4                         False

[5 rows x 5687 columns]
```

[43]:
```
import tensorflow as tf
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
```

```
2024-06-20 10:28:44.942862: E
external/local_xla/xla/stream_executor/cuda/cuda_dnn.cc:9261] Unable to register
cuDNN factory: Attempting to register factory for plugin cuDNN when one has
already been registered
2024-06-20 10:28:44.943079: E
external/local_xla/xla/stream_executor/cuda/cuda_fft.cc:607] Unable to register
cuFFT factory: Attempting to register factory for plugin cuFFT when one has
already been registered
2024-06-20 10:28:45.118197: E
external/local_xla/xla/stream_executor/cuda/cuda_blas.cc:1515] Unable to
register cuBLAS factory: Attempting to register factory for plugin cuBLAS when
one has already been registered
```

[44]:
```
# Separate features and target variable
X = df.drop('Totals_TotalWinners', axis=1)
y = df['Totals_TotalWinners']
```

[45]:
```
# Normalize/Standardize numerical features
scaler = StandardScaler()
X = scaler.fit_transform(X)
```

[46]:
```
# Split the data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
```

[47]:
```
# Build a simple neural network model
model = tf.keras.Sequential([
    tf.keras.layers.Dense(64, activation='relu', input_shape=(X_train.
 ↪shape[1],)),
    tf.keras.layers.Dense(32, activation='relu'),
    tf.keras.layers.Dense(1)  # Assuming a regression problem. Use tf.keras.
 ↪layers.Dense(1, activation='sigmoid') for binary classification
])
```

```
/opt/conda/lib/python3.10/site-packages/keras/src/layers/core/dense.py:87:
UserWarning: Do not pass an `input_shape`/`input_dim` argument to a layer. When
using Sequential models, prefer using an `Input(shape)` object as the first
layer in the model instead.
  super().__init__(activity_regularizer=activity_regularizer, **kwargs)
```

[48]:
```
# Compile the model
model.compile(optimizer='adam', loss='mse', metrics=['mae'])
```

[49]:
```
# Train the model
history = model.fit(X_train, y_train, epochs=50, validation_split=0.2,␣
 ↪batch_size=32)

# Evaluate the model
loss, mae = model.evaluate(X_test, y_test)
print(f'Test MAE: {mae}')
```

```
Epoch 1/50
35/35             2s 12ms/step -
loss: 1.3440 - mae: 0.9090 - val_loss: 1.0495 - val_mae: 0.7763
Epoch 2/50
35/35             0s 7ms/step - loss:
1.1648 - mae: 0.8475 - val_loss: 0.9701 - val_mae: 0.7552
Epoch 3/50
35/35             0s 6ms/step - loss:
0.4199 - mae: 0.4957 - val_loss: 1.2032 - val_mae: 0.8379
```

```
Epoch 4/50
35/35            0s 6ms/step - loss:
0.2024 - mae: 0.3446 - val_loss: 1.9046 - val_mae: 1.0937
Epoch 5/50
35/35            0s 6ms/step - loss:
0.1298 - mae: 0.2632 - val_loss: 1.8925 - val_mae: 1.0887
Epoch 6/50
35/35            0s 6ms/step - loss:
0.0955 - mae: 0.2120 - val_loss: 2.3697 - val_mae: 1.2436
Epoch 7/50
35/35            0s 7ms/step - loss:
0.0729 - mae: 0.1742 - val_loss: 2.2589 - val_mae: 1.2114
Epoch 8/50
35/35            0s 6ms/step - loss:
0.0427 - mae: 0.1434 - val_loss: 2.6871 - val_mae: 1.3338
Epoch 9/50
35/35            0s 6ms/step - loss:
0.0348 - mae: 0.1292 - val_loss: 2.2586 - val_mae: 1.2119
Epoch 10/50
35/35            0s 6ms/step - loss:
0.0281 - mae: 0.1096 - val_loss: 2.8507 - val_mae: 1.3784
Epoch 11/50
35/35            0s 6ms/step - loss:
0.0292 - mae: 0.1094 - val_loss: 1.9584 - val_mae: 1.1209
Epoch 12/50
35/35            0s 6ms/step - loss:
0.0291 - mae: 0.1136 - val_loss: 2.5290 - val_mae: 1.2928
Epoch 13/50
35/35            0s 6ms/step - loss:
0.0304 - mae: 0.1139 - val_loss: 2.6645 - val_mae: 1.3297
Epoch 14/50
35/35            0s 6ms/step - loss:
0.0313 - mae: 0.1076 - val_loss: 2.7687 - val_mae: 1.3581
Epoch 15/50
35/35            0s 6ms/step - loss:
0.0361 - mae: 0.1140 - val_loss: 2.4525 - val_mae: 1.2735
Epoch 16/50
35/35            0s 6ms/step - loss:
0.0277 - mae: 0.1111 - val_loss: 2.6177 - val_mae: 1.3155
Epoch 17/50
35/35            0s 6ms/step - loss:
0.0307 - mae: 0.1149 - val_loss: 2.5466 - val_mae: 1.2996
Epoch 18/50
35/35            0s 6ms/step - loss:
0.0353 - mae: 0.1123 - val_loss: 2.7946 - val_mae: 1.3653
Epoch 19/50
35/35            0s 6ms/step - loss:
0.0296 - mae: 0.1053 - val_loss: 2.7158 - val_mae: 1.3489
```

```
Epoch 20/50
35/35            0s 6ms/step - loss:
0.0284 - mae: 0.1086 - val_loss: 2.8037 - val_mae: 1.3719
Epoch 21/50
35/35            0s 6ms/step - loss:
0.0272 - mae: 0.0999 - val_loss: 2.2755 - val_mae: 1.2228
Epoch 22/50
35/35            0s 6ms/step - loss:
0.0330 - mae: 0.1131 - val_loss: 2.4740 - val_mae: 1.2775
Epoch 23/50
35/35            0s 6ms/step - loss:
0.0321 - mae: 0.1186 - val_loss: 2.6094 - val_mae: 1.3145
Epoch 24/50
35/35            0s 6ms/step - loss:
0.0291 - mae: 0.1152 - val_loss: 2.5630 - val_mae: 1.3048
Epoch 25/50
35/35            0s 6ms/step - loss:
0.0273 - mae: 0.1098 - val_loss: 2.2959 - val_mae: 1.2262
Epoch 26/50
35/35            0s 6ms/step - loss:
0.0251 - mae: 0.1015 - val_loss: 2.4417 - val_mae: 1.2694
Epoch 27/50
35/35            0s 6ms/step - loss:
0.0259 - mae: 0.0982 - val_loss: 2.3588 - val_mae: 1.2479
Epoch 28/50
35/35            0s 6ms/step - loss:
0.0297 - mae: 0.1102 - val_loss: 2.0524 - val_mae: 1.1542
Epoch 29/50
35/35            0s 6ms/step - loss:
0.0267 - mae: 0.1072 - val_loss: 2.2732 - val_mae: 1.2194
Epoch 30/50
35/35            0s 6ms/step - loss:
0.0273 - mae: 0.1050 - val_loss: 2.1384 - val_mae: 1.1823
Epoch 31/50
35/35            0s 6ms/step - loss:
0.0287 - mae: 0.1121 - val_loss: 2.3571 - val_mae: 1.2472
Epoch 32/50
35/35            0s 7ms/step - loss:
0.0288 - mae: 0.1067 - val_loss: 2.5698 - val_mae: 1.3021
Epoch 33/50
35/35            0s 6ms/step - loss:
0.0275 - mae: 0.1007 - val_loss: 2.2408 - val_mae: 1.2074
Epoch 34/50
35/35            0s 6ms/step - loss:
0.0271 - mae: 0.0972 - val_loss: 2.1315 - val_mae: 1.1826
Epoch 35/50
35/35            0s 6ms/step - loss:
0.0249 - mae: 0.0970 - val_loss: 2.3186 - val_mae: 1.2296
```

```
Epoch 36/50
35/35            0s 6ms/step - loss:
0.0206 - mae: 0.0834 - val_loss: 2.3476 - val_mae: 1.2425
Epoch 37/50
35/35            0s 6ms/step - loss:
0.0157 - mae: 0.0772 - val_loss: 2.2806 - val_mae: 1.2184
Epoch 38/50
35/35            0s 6ms/step - loss:
0.0145 - mae: 0.0737 - val_loss: 1.9292 - val_mae: 1.1136
Epoch 39/50
35/35            0s 6ms/step - loss:
0.0099 - mae: 0.0639 - val_loss: 1.9154 - val_mae: 1.1062
Epoch 40/50
35/35            0s 6ms/step - loss:
0.0102 - mae: 0.0612 - val_loss: 2.0529 - val_mae: 1.1532
Epoch 41/50
35/35            0s 6ms/step - loss:
0.0098 - mae: 0.0588 - val_loss: 2.1145 - val_mae: 1.1684
Epoch 42/50
35/35            0s 6ms/step - loss:
0.0103 - mae: 0.0583 - val_loss: 1.7768 - val_mae: 1.0637
Epoch 43/50
35/35            0s 6ms/step - loss:
0.0114 - mae: 0.0622 - val_loss: 1.7965 - val_mae: 1.0681
Epoch 44/50
35/35            0s 6ms/step - loss:
0.0104 - mae: 0.0585 - val_loss: 1.5813 - val_mae: 0.9990
Epoch 45/50
35/35            0s 6ms/step - loss:
0.0100 - mae: 0.0606 - val_loss: 1.8061 - val_mae: 1.0715
Epoch 46/50
35/35            0s 6ms/step - loss:
0.0092 - mae: 0.0579 - val_loss: 1.5895 - val_mae: 1.0006
Epoch 47/50
35/35            0s 6ms/step - loss:
0.0076 - mae: 0.0563 - val_loss: 1.6694 - val_mae: 1.0257
Epoch 48/50
35/35            0s 6ms/step - loss:
0.0074 - mae: 0.0571 - val_loss: 1.7028 - val_mae: 1.0381
Epoch 49/50
35/35            0s 6ms/step - loss:
0.0090 - mae: 0.0587 - val_loss: 1.6189 - val_mae: 1.0050
Epoch 50/50
35/35            0s 6ms/step - loss:
0.0099 - mae: 0.0614 - val_loss: 1.7548 - val_mae: 1.0556
11/11            0s 2ms/step - loss:
1.6322 - mae: 1.0165
Test MAE: 1.0563933849334717
```

[50]:
```
from sklearn.linear_model import LinearRegression
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.metrics import mean_absolute_error
```

[51]:
```
models = {
    'Linear Regression': LinearRegression(),
    'Decision Tree': DecisionTreeRegressor(),
    'Random Forest': RandomForestRegressor(n_estimators=100),
    'Gradient Boosting': GradientBoostingRegressor(n_estimators=100)
}

# Train and evaluate models
for name, model in models.items():
    model.fit(X_train, y_train)
    y_pred = model.predict(X_test)
    mae = mean_absolute_error(y_test, y_pred)
    print(f'{name} Test MAE: {mae}')
```

```
Linear Regression Test MAE: 0.36084412644111546
Decision Tree Test MAE: 0.12066042636369813
Random Forest Test MAE: 0.09006758680037105
Gradient Boosting Test MAE: 0.10083079234361164
```

[ ]: