

Pactica_Muestreo

July 5, 2023

Reporte del Proceso de Muestreo

Marco Antonio Obregón Flores

Paso 1: Identificar y definir el grupo objetivo

El conjunto de datos está compuesto por diversas reseñas de anime, con información variada como el título, el género, la popularidad y las calificaciones dadas por los usuarios. Nuestro grupo objetivo es el género del anime, ya que buscamos realizar un análisis basado en estos distintos géneros.

```
[ ]: import pandas as pd
import matplotlib.pyplot as plt
import numpy as np
```

```
[2]: df_anime = pd.read_csv("/Users/marcgrayson/Documents/Big Data/Practica 4_
↳Muestreo/archive (3)/final_animatedataset.csv")
```

```
[3]: df_anime.head()
```

```
[3]:  username  anime_id  my_score  user_id  gender  title type source \
0  karthiga      21         9  2255153  Female  One Piece  TV  Manga
1  karthiga      59         7  2255153  Female  Chobits   TV  Manga
2  karthiga      74         7  2255153  Female  Gakuen Alice TV  Manga
3  karthiga     120         7  2255153  Female  Fruits Basket TV  Manga
4  karthiga     178         7  2255153  Female  Ultra Maniac  TV  Manga
```

```
      score  scored_by  rank  popularity \
0    8.54    423868    91.0         35
1    7.53    175388   1546.0        188
2    7.77    33244    941.0       1291
3    7.77   167968    939.0        222
4    7.26     9663   2594.0       2490
```

```
                                genre
0  Action, Adventure, Comedy, Super Power, Drama,...
1    Sci-Fi, Comedy, Drama, Romance, Ecchi, Seinen
2                Comedy, School, Shoujo, Super Power
3  Slice of Life, Comedy, Drama, Romance, Fantasy...
4                Magic, Comedy, Romance, School, Shoujo
```

```
[54]: # Rows
df_anime.shape[0]
```

```
[54]: 35305695
```

Paso 2: Crear un rango específico de muestreo

Se decide utilizar una fracción del conjunto de datos original debido a su gran tamaño. Se seleccionó aleatoriamente un 10% del conjunto de datos original para facilitar el manejo de los datos.

```
[28]: df_frac = df_anime.sample(frac=0.1)
```

```
[29]: df_sample.shape[0]
```

```
[29]: 3530570
```

Paso 3: Seleccionar el método adecuado para los datos

Se decide utilizar el muestreo estratificado para garantizar que cada género estuviera representado en nuestra muestra. Primero, separamos los géneros, que estaban en una única columna separada por comas, en varias filas para facilitar el análisis. Luego, realizamos un muestreo estratificado basado en la popularidad de cada género.

```
[30]: df_popularity_sample = df_frac.sample(n=1000, weights='popularity',
↳ random_state=1)
```

```
[32]: # Crear una nueva columna que es una lista de los géneros
df_popularity_sample2 = df_popularity_sample
df_popularity_sample2['genre_list'] = df_popularity_sample2['genre'].str.
↳ split(',', '')
```

```
# Explode la lista de géneros en filas separadas
df_genre_exploded = df_popularity_sample2.explode('genre_list')
```

```
[33]: df_genre_exploded.head()
```

```
[33]:
```

	username	anime_id	my_score	user_id	gender	\
7278571	1st_King	3466	7	4455785	Male	
7278571	1st_King	3466	7	4455785	Male	
7278571	1st_King	3466	7	4455785	Male	
22776852	tomu	7375	0	239407	Male	
22776852	tomu	7375	0	239407	Male	

		title	type	\
7278571	Kino no Tabi: The Beautiful World - Tou no Kuni	Special		
7278571	Kino no Tabi: The Beautiful World - Tou no Kuni	Special		
7278571	Kino no Tabi: The Beautiful World - Tou no Kuni	Special		
22776852	Shakugan no Shana S Specials	Special		
22776852	Shakugan no Shana S Specials	Special		

	source	score	scored_by	rank	popularity \
7278571	Light novel	7.67	13521	1170.0	2320
7278571	Light novel	7.67	13521	1170.0	2320
7278571	Light novel	7.67	13521	1170.0	2320
22776852	Light novel	6.99	9162	3616.0	2671
22776852	Light novel	6.99	9162	3616.0	2671

	genre	genre_list
7278571	Adventure, Fantasy, Psychological	Adventure
7278571	Adventure, Fantasy, Psychological	Fantasy
7278571	Adventure, Fantasy, Psychological	Psychological
22776852	Comedy, Parody	Comedy
22776852	Comedy, Parody	Parody

Paso 4: Especificar el tamaño de la muestra

Se decide que el tamaño de la muestra sería de 1000 registros. Los tamaños de las muestras para cada estrato (género) se calcularon basándose en las proporciones de cada género en el conjunto de datos.

```
[38]: #
sample_size = 1000

# Calcular los tamaños de las muestras para cada estrato (género)
stratum_sizes = df_genre_exploded['genre_list'].value_counts(normalize=True) * \
    ↪sample_size

# Redondear los tamaños de las muestras a enteros y convertir a diccionario
stratum_sizes = np.round(stratum_sizes).astype(int).to_dict()
```

```
[39]: print(stratum_sizes)
```

```
{'Comedy': 120, 'Action': 79, 'Sci-Fi': 62, 'Fantasy': 62, 'Adventure': 60,
'Drama': 59, 'Romance': 58, 'Shounen': 51, 'School': 48, 'Supernatural': 39,
'Slice of Life': 37, 'Ecchi': 27, 'Magic': 26, 'Mecha': 23, 'Seinen': 22,
'Shoujo': 19, 'Super Power': 17, 'Historical': 16, 'Hentai': 16, 'Mystery': 15,
'Military': 15, 'Harem': 13, 'Sports': 12, 'Music': 11, 'Psychological': 10,
'Parody': 10, 'Space': 9, 'Martial Arts': 9, 'Demons': 9, 'Horror': 8, 'Game':
6, 'Kids': 6, 'Samurai': 5, 'Police': 4, 'Shoujo Ai': 3, 'Thriller': 3,
'Vampire': 3, 'Josei': 2, 'Dementia': 2, 'Yaoi': 2, 'Shounen Ai': 1}
```

Paso 5: Recolectar los datos muestreados

Aplicamos la función de muestreo estratificado a nuestro conjunto de datos y obtuvimos una muestra estratificada. En esta muestra, cada género estaba representado en proporción a su prevalencia en el conjunto de datos original.

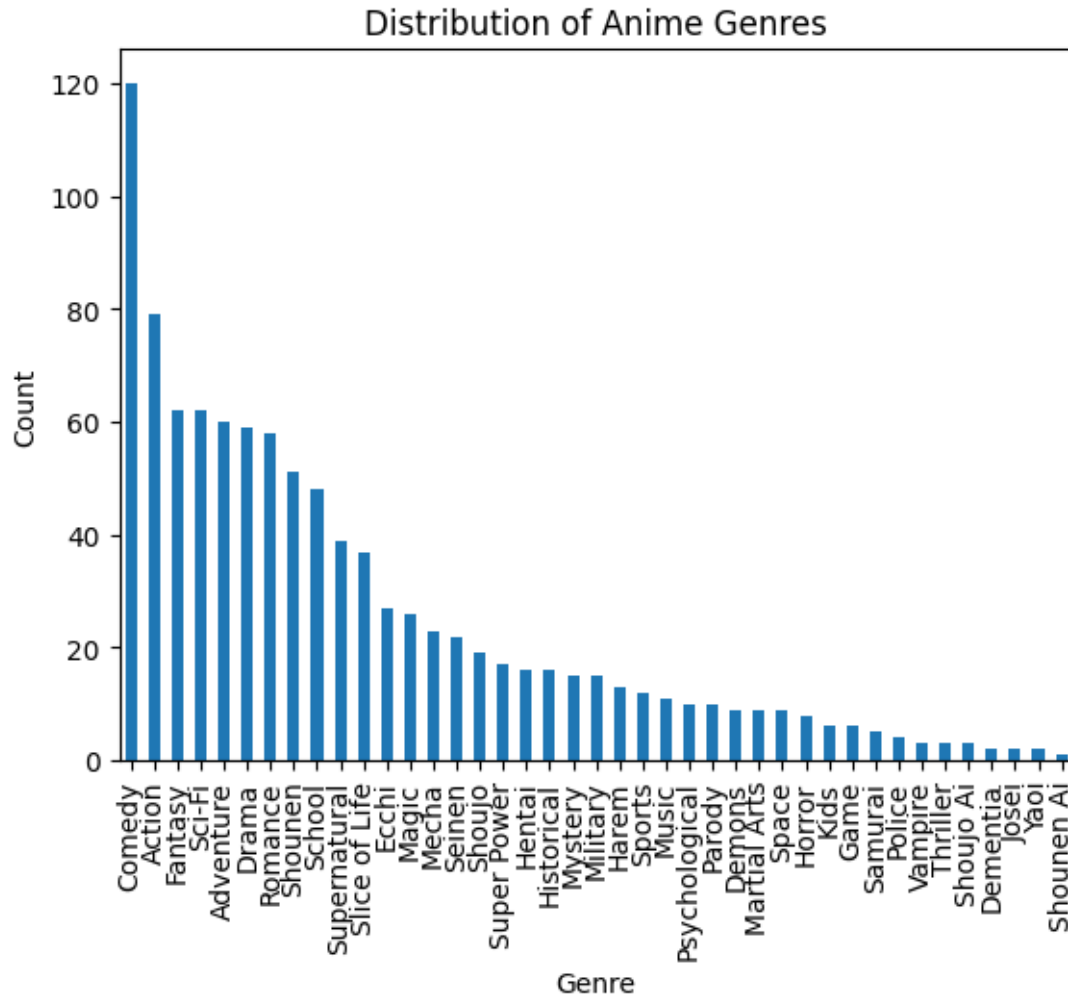
Finalmente, se realizó un análisis de las calificaciones medias por género y se descubrió que los géneros con las calificaciones promedio más altas eran Thriller, Dementia y Vampire. Esto propor-

ciona un insight valioso para los creadores de contenido de anime y los aficionados al anime, ya que muestra qué géneros tienden a ser mejor valorados por los espectadores. Además, la visualización de los datos a través de gráficos de barras y boxplots permitió entender mejor la distribución y el rango de las calificaciones para cada género.

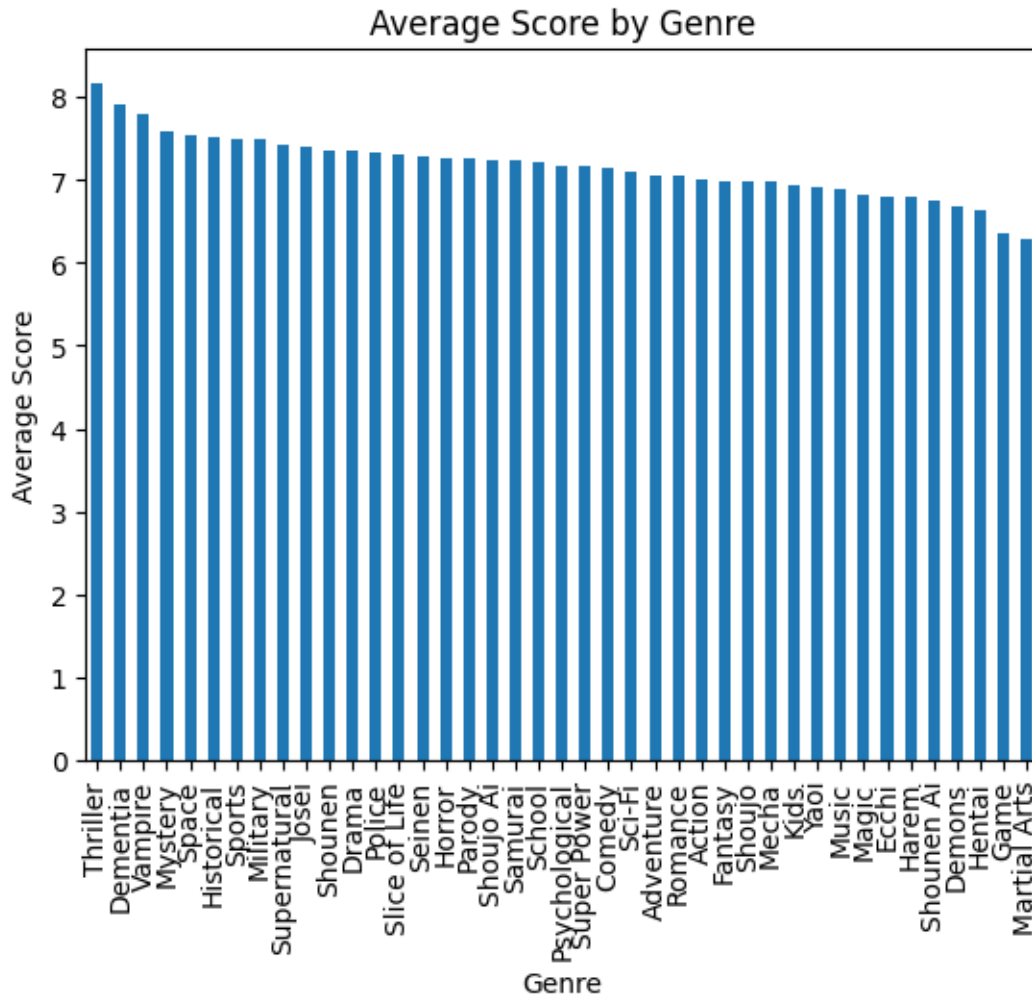
```
[40]: # Definir una función para hacer el muestreo estratificado
def stratified_sample(group):
    genre = group.name
    n = stratum_sizes.get(genre, 0)
    return group.sample(n=n)

# Aplicar el muestreo estratificado
df_stratified_sample = df_genre_exploded.groupby('genre_list',
↪group_keys=False).apply(stratified_sample)
```

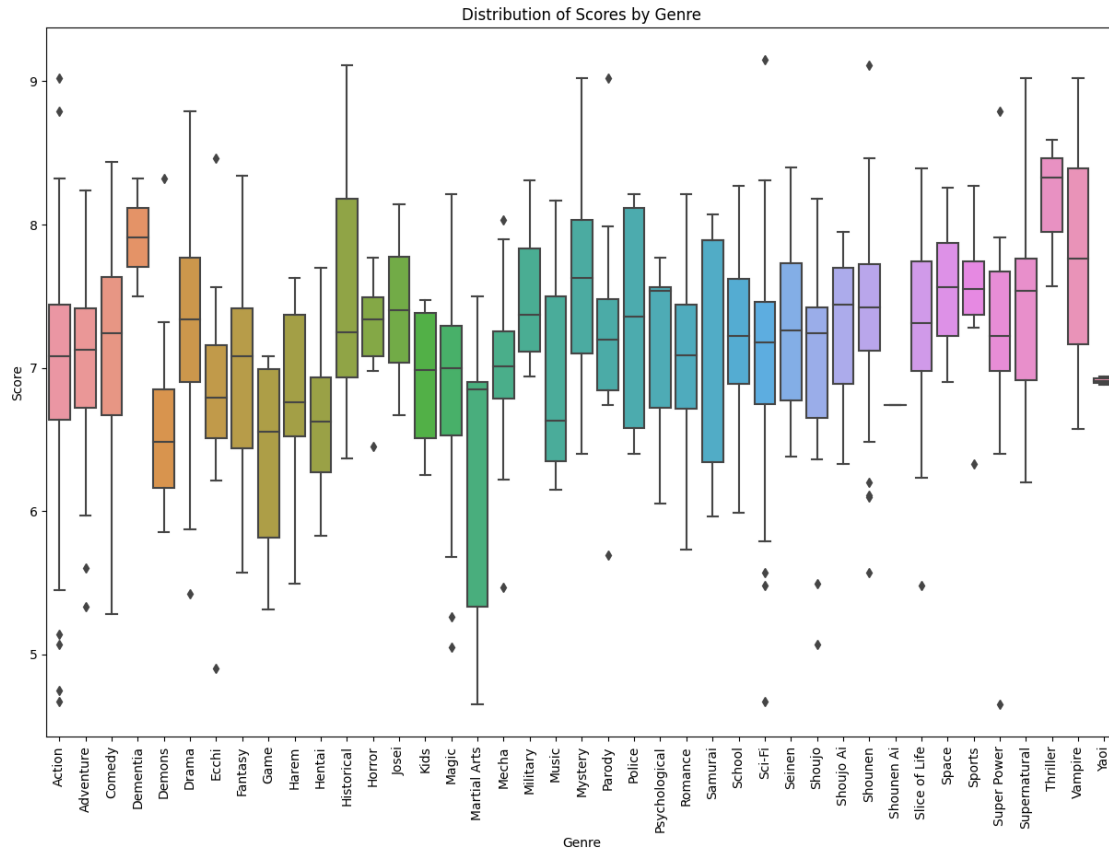
```
[44]: import matplotlib.pyplot as plt
df_stratified_sample['genre_list'].value_counts().plot(kind='bar')
plt.xlabel('Genre')
plt.ylabel('Count')
plt.title('Distribution of Anime Genres')
plt.show()
```



```
[48]: mean_scores = df_stratified_sample.groupby('genre_list')['score'].mean().
      ↪sort_values(ascending=False)
mean_scores.plot(kind='bar')
plt.xlabel('Genre')
plt.ylabel('Average Score')
plt.title('Average Score by Genre')
plt.show()
```



```
[49]: import seaborn as sns
plt.figure(figsize=(15, 10))
sns.boxplot(x='genre_list', y='score', data=df_stratified_sample)
plt.xlabel('Genre')
plt.ylabel('Score')
plt.title('Distribution of Scores by Genre')
plt.xticks(rotation=90)
plt.show()
```



```
[52]: # Calcular las calificaciones promedio por género
mean_scores = df_stratified_sample.groupby('genre_list')['score'].mean().
    ↪sort_values(ascending=False)

# Convertir el resultado en un DataFrame
mean_scores_df = mean_scores.reset_index()

# Cambiar el nombre de las columnas
mean_scores_df.columns = ['Genre', 'Average Score']

print(mean_scores_df)
```

	Genre	Average Score
0	Thriller	8.163333
1	Dementia	7.910000
2	Vampire	7.783333
3	Mystery	7.593333
4	Space	7.550000
5	Historical	7.509375
6	Sports	7.503333

7	Military	7.491333
8	Supernatural	7.417692
9	Josei	7.405000
10	Shounen	7.365294
11	Drama	7.342373
12	Police	7.332500
13	Slice of Life	7.301081
14	Seinen	7.285455
15	Horror	7.261250
16	Parody	7.253000
17	Shoujo Ai	7.240000
18	Samurai	7.230000
19	School	7.222500
20	Psychological	7.174000
21	Super Power	7.170588
22	Comedy	7.142167
23	Sci-Fi	7.088548
24	Adventure	7.060500
25	Romance	7.050172
26	Action	6.994430
27	Fantasy	6.991613
28	Shoujo	6.990526
29	Mecha	6.977391
30	Kids	6.926667
31	Yaoi	6.910000
32	Music	6.888182
33	Magic	6.828077
34	Ecchi	6.806667
35	Harem	6.803077
36	Shounen Ai	6.740000
37	Demons	6.673333
38	Hentai	6.642500
39	Game	6.366667
40	Martial Arts	6.296667

```
[53]: # Calcular estadísticas descriptivas por género
boxplot_stats = df_stratified_sample.groupby('genre_list')['score'].describe()

# Convertir el resultado en un DataFrame
boxplot_stats_df = boxplot_stats.reset_index()

print(boxplot_stats_df)
```

	genre_list	count	mean	std	min	25%	50%	75%	\
0	Action	79.0	6.994430	0.848839	4.67	6.6400	7.080	7.4400	
1	Adventure	60.0	7.060500	0.610775	5.33	6.7200	7.125	7.4125	
2	Comedy	120.0	7.142167	0.697848	5.28	6.6700	7.240	7.6350	
3	Dementia	2.0	7.910000	0.579828	7.50	7.7050	7.910	8.1150	

4	Demons	9.0	6.673333	0.766355	5.85	6.1600	6.480	6.8500
5	Drama	59.0	7.342373	0.649021	5.42	6.9000	7.340	7.7700
6	Ecchi	27.0	6.806667	0.619839	4.90	6.5100	6.790	7.1600
7	Fantasy	62.0	6.991613	0.692975	5.57	6.4375	7.080	7.4175
8	Game	6.0	6.366667	0.747922	5.31	5.8150	6.555	6.9875
9	Harem	13.0	6.803077	0.590768	5.49	6.5200	6.760	7.3700
10	Hentai	16.0	6.642500	0.501750	5.83	6.2700	6.625	6.9350
11	Historical	16.0	7.509375	0.807552	6.37	6.9325	7.245	8.1775
12	Horror	8.0	7.261250	0.421271	6.45	7.0775	7.340	7.4950
13	Josei	2.0	7.405000	1.039447	6.67	7.0375	7.405	7.7725
14	Kids	6.0	6.926667	0.533392	6.25	6.5100	6.985	7.3850
15	Magic	26.0	6.828077	0.735929	5.05	6.5275	6.995	7.2950
16	Martial Arts	9.0	6.296667	1.079977	4.65	5.3300	6.850	6.9000
17	Mecha	23.0	6.977391	0.554578	5.47	6.7850	7.010	7.2550
18	Military	15.0	7.491333	0.484869	6.94	7.1150	7.370	7.8350
19	Music	11.0	6.888182	0.703503	6.15	6.3500	6.630	7.5000
20	Mystery	15.0	7.593333	0.685072	6.40	7.1000	7.630	8.0300
21	Parody	10.0	7.253000	0.871449	5.69	6.8400	7.195	7.4800
22	Police	4.0	7.332500	0.944788	6.40	6.5800	7.360	8.1125
23	Psychological	10.0	7.174000	0.594590	6.05	6.7175	7.540	7.5650
24	Romance	58.0	7.050172	0.564715	5.73	6.7150	7.090	7.4400
25	Samurai	5.0	7.230000	0.997722	5.96	6.3400	7.890	7.8900
26	School	48.0	7.222500	0.542577	5.99	6.8875	7.220	7.6225
27	Sci-Fi	62.0	7.088548	0.700494	4.67	6.7450	7.180	7.4625
28	Seinen	22.0	7.285455	0.598910	6.38	6.7750	7.260	7.7300
29	Shoujo	19.0	6.990526	0.741826	5.07	6.6500	7.240	7.4200
30	Shoujo Ai	3.0	7.240000	0.828312	6.33	6.8850	7.440	7.6950
31	Shounen	51.0	7.365294	0.636350	5.57	7.1200	7.420	7.7250
32	Shounen Ai	1.0	6.740000	NaN	6.74	6.7400	6.740	6.7400
33	Slice of Life	37.0	7.301081	0.647837	5.48	6.9800	7.310	7.7400
34	Space	9.0	7.550000	0.449694	6.90	7.2200	7.560	7.8700
35	Sports	12.0	7.503333	0.460481	6.33	7.3700	7.550	7.7425
36	Super Power	17.0	7.170588	0.872951	4.65	6.9800	7.220	7.6700
37	Supernatural	39.0	7.417692	0.633937	6.20	6.9150	7.540	7.7650
38	Thriller	3.0	8.163333	0.530031	7.57	7.9500	8.330	8.4600
39	Vampire	3.0	7.783333	1.225167	6.57	7.1650	7.760	8.3900
40	Yaoi	2.0	6.910000	0.042426	6.88	6.8950	6.910	6.9250

	max
0	9.02
1	8.24
2	8.44
3	8.32
4	8.32
5	8.79
6	8.46
7	8.34
8	7.08

9	7.63
10	7.70
11	9.11
12	7.77
13	8.14
14	7.47
15	8.21
16	7.50
17	8.03
18	8.31
19	8.17
20	9.02
21	9.02
22	8.21
23	7.77
24	8.21
25	8.07
26	8.27
27	9.15
28	8.40
29	8.18
30	7.95
31	9.11
32	6.74
33	8.39
34	8.26
35	8.27
36	8.79
37	9.02
38	8.59
39	9.02
40	6.94