DataSet Kaggle con i Modelli

April 23, 2024

1 DataSet Kaggle

1.1 1.0 Caricamento e Visualizzazione dei Dati dei Titoli Netflix

[2]:
```
# Importa le librerie, carica il file csv nel dataframe e lo stampa
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
percorso_file_csv = "C:\\Users\\zetam\\Desktop\\2␣
 ↪Superiore\\Robotica\\netflix_titles.csv"
df = pd.read_csv(percorso_file_csv)
print(df)
```

```
     show_id     type                  title         director  \
0         s1    Movie   Dick Johnson Is Dead  Kirsten Johnson
1         s2  TV Show           Blood & Water              NaN
2         s3  TV Show               Ganglands  Julien Leclercq
3         s4  TV Show  Jailbirds New Orleans              NaN
4         s5  TV Show           Kota Factory              NaN
…       …       …                     …              …
8802   s8803    Movie                 Zodiac    David Fincher
8803   s8804  TV Show            Zombie Dumb              NaN
8804   s8805    Movie             Zombieland  Ruben Fleischer
8805   s8806    Movie                   Zoom     Peter Hewitt
8806   s8807    Movie                 Zubaan      Mozez Singh

                                                   cast        country  \
0                                                   NaN  United States
1     Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban…   South Africa
2     Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi…            NaN
3                                                   NaN            NaN
4     Mayur More, Jitendra Kumar, Ranjan Raj, Alam K…          India
…                                                   …            …
8802  Mark Ruffalo, Jake Gyllenhaal, Robert Downey J…  United States
8803                                                NaN            NaN
8804  Jesse Eisenberg, Woody Harrelson, Emma Stone, …  United States
8805  Tim Allen, Courteney Cox, Chevy Chase, Kate Ma…  United States
```

```
8806  Vicky Kaushal, Sarah-Jane Dias, Raaghav Chanan…          India

              date_added  release_year rating   duration  \
0     September 25, 2021          2020  PG-13     90 min
1     September 24, 2021          2021  TV-MA  2 Seasons
2     September 24, 2021          2021  TV-MA   1 Season
3     September 24, 2021          2021  TV-MA   1 Season
4     September 24, 2021          2021  TV-MA  2 Seasons
…                    …           …    …        …
8802   November 20, 2019          2007      R    158 min
8803        July 1, 2019          2018  TV-Y7  2 Seasons
8804    November 1, 2019          2009      R     88 min
8805    January 11, 2020          2006     PG     88 min
8806       March 2, 2019          2015  TV-14    111 min

                                              listed_in  \
0                                         Documentaries
1       International TV Shows, TV Dramas, TV Mysteries
2     Crime TV Shows, International TV Shows, TV Act…
3                               Docuseries, Reality TV
4     International TV Shows, Romantic TV Shows, TV …
…                                                    …
8802                    Cult Movies, Dramas, Thrillers
8803            Kids' TV, Korean TV Shows, TV Comedies
8804                           Comedies, Horror Movies
8805                Children & Family Movies, Comedies
8806    Dramas, International Movies, Music & Musicals

                                            description
0     As her father nears the end of his life, filmm…
1     After crossing paths at a party, a Cape Town t…
2     To protect his family from a powerful drug lor…
3     Feuds, flirtations and toilet talk go down amo…
4     In a city of coaching centers known to train I…
…                                                   …
8802  A political cartoonist, a crime reporter and a…
8803  While living alone in a spooky town, a young g…
8804  Looking to survive in a world taken over by zo…
8805  Dragged from civilian life, a former superhero…
8806  A scrappy but poor boy worms his way into a ty…

[8807 rows x 12 columns]
```

1.2 1.1 Identificazione del Tipo di Programma più Frequente nei Titoli Netflix

[12]:
```
# Conta quante volte compare ogni tipo di programma e stampa quello con il␣
 ↪numero maggiore
# utilizzando il metodo idmax
import pandas as pd
percorso_file_csv = "C:\\Users\\zetam\\Desktop\\2␣
 ↪Superiore\\Robotica\\netflix_titles.csv"
df = pd.read_csv(percorso_file_csv)
tipo_programma = df['type'].value_counts().idxmax()
print(tipo_programma)
```

```
Movie
```

1.3 1.2 Conteggio dei Programmi Netflix per Anno di Rilascio

[13]:
```
# Conta quanti programmi ci sono per ogni anno e stampa i numeri
import pandas as pd

anno_programma = df['release_year'].value_counts()
print(anno_programma)
```

```
release_year
2018    1147
2017    1032
2019    1030
2020     953
2016     902
        …
1959       1
1925       1
1961       1
1947       1
1966       1
Name: count, Length: 74, dtype: int64
```

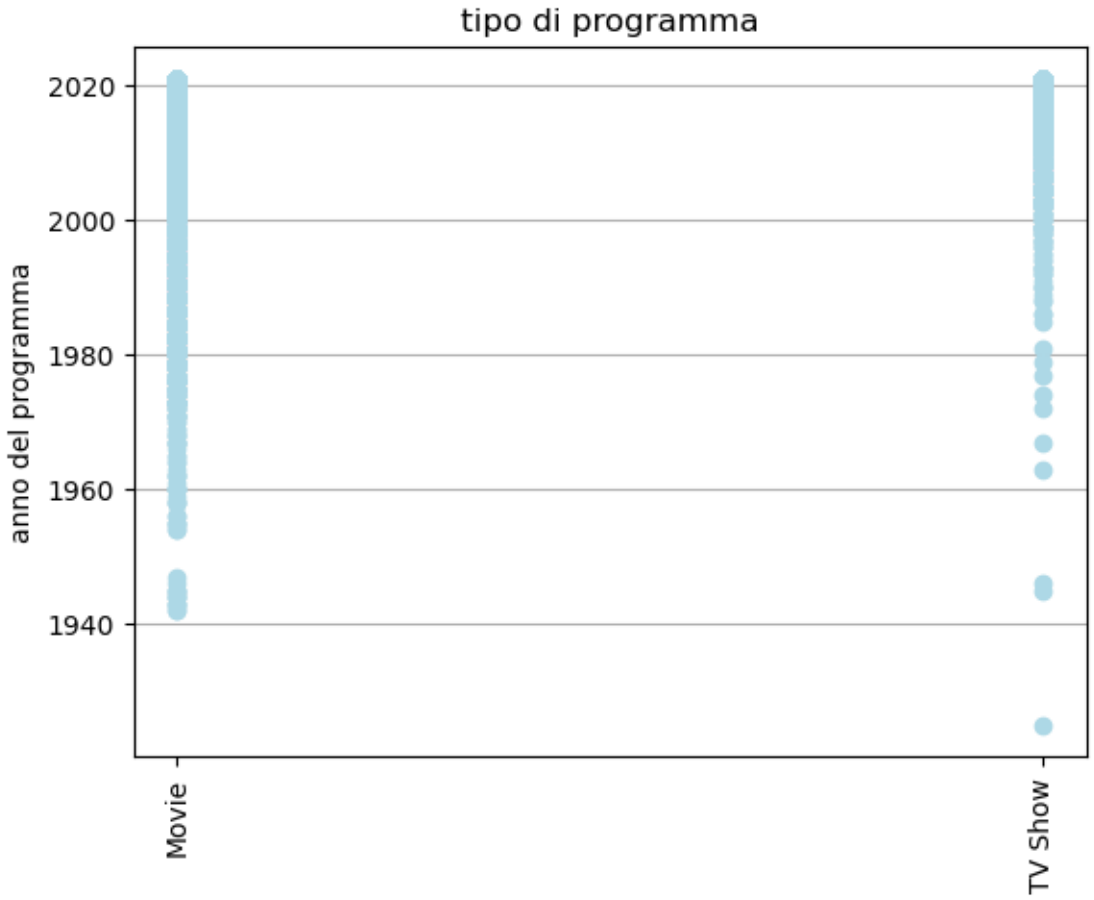
1.4 1.3 Identificazione dell'Anno con il Maggior Numero di Programmi Netflix

[14]:
```
# Conta quanti programmi ci sono per ogni anno e stampa l'anno che ne ha di più
import pandas as pd
anno_prog = df['release_year'].value_counts().idxmax()
print(anno_prog)
```

```
2018
```

1.5 1.4 Visualizzazione della Distribuzione dei Tipi di Programmi Netflix nel Corso degli Anni

[15]:
```
# Grafico a dispersione che mostra la distribuzione dei tipi di programmi negli␣
 ↪anni
import matplotlib.pyplot as plt
plt.plot(df['type'],df['release_year'], marker='o', linestyle='',␣
 ↪color='lightblue')
plt.title('tipo di programma')
plt.ylabel('anno del programma')
plt.xticks(rotation=90)
plt.grid(True, axis="y")
plt.show()
```



1.6 1.5 Identificazione e Stampa delle Righe con Valori Mancanti nel DataFrame

[16]:
```
# identifica le righe con valori mancanti e lo stampa
righe_con_dati_mancanti = df[df.isnull().any(axis=1)]
righe_con_dati_mancanti
```

[16]:

	show_id	type	title	director	cast	country	date_added	release_year	rating	duration
0	s1	Movie	Dick Johnson Is Dead	Kirsten Johnson	NaN	United States	September 25, 2021	2020	PG-13	90 min
1	s2	TV Show	Blood & Water	NaN	Ama Qamata, Khosi Ngema, Gail Mabalane, Thaban…	South Africa	September 24, 2021			
2	s3	TV Show	Ganglands	Julien Leclercq	Sami Bouajila, Tracy Gotoas, Samuel Jouy, Nabi…	NaN	September 24, 2021			
3	s4	TV Show	Jailbirds New Orleans	NaN	NaN	NaN	September 24, 2021			
4	s5	TV Show	Kota Factory	NaN	Mayur More, Jitendra Kumar, Ranjan Raj, Alam K…	India	September 24, 2021			
…	…	…	…	…	…	…	…			
8795	s8796	TV Show	Yu-Gi-Oh! Arc-V	NaN	Mike Liscio, Emily Bauer, Billy Bob Thompson, …	Japan, Canada	May 1, 2018			
8796	s8797	TV Show	Yunus Emre	NaN	Gökhan Atalay, Payidar Tüfekçioglu, Baran Akbu…	Turkey	January 17, 2017			
8797	s8798	TV Show	Zak Storm	NaN	Michael Johnston, Jessica Gee-George, Christin…	United States, France, South Korea, Indonesia	September 13, 2018			
8800	s8801	TV Show	Zindagi Gulzar Hai	NaN	Sanam Saeed, Fawad Khan, Ayesha Omer, Mehreen …	Pakistan	December 15, 2016			
8803	s8804	TV Show	Zombie Dumb	NaN	NaN	NaN	July 1, 2019			

```
1             2021  TV-MA  2 Seasons
2             2021  TV-MA   1 Season
3             2021  TV-MA   1 Season
4             2021  TV-MA  2 Seasons
…              …     …         …
8795          2015  TV-Y7  2 Seasons
8796          2016  TV-PG  2 Seasons
8797          2016  TV-Y7  3 Seasons
8800          2012  TV-PG   1 Season
8803          2018  TV-Y7  2 Seasons

                                                listed_in  \
0                                           Documentaries
1       International TV Shows, TV Dramas, TV Mysteries
2     Crime TV Shows, International TV Shows, TV Act…
3                                 Docuseries, Reality TV
4     International TV Shows, Romantic TV Shows, TV …
…                                                     …
8795                            Anime Series, Kids' TV
8796                 International TV Shows, TV Dramas
8797                                          Kids' TV
8800  International TV Shows, Romantic TV Shows, TV …
8803            Kids' TV, Korean TV Shows, TV Comedies

                                              description
0     As her father nears the end of his life, filmm…
1     After crossing paths at a party, a Cape Town t…
2     To protect his family from a powerful drug lor…
3     Feuds, flirtations and toilet talk go down amo…
4     In a city of coaching centers known to train I…
…                                                   …
8795  Now that he's discovered the Pendulum Summonin…
8796  During the Mongol invasions, Yunus Emre leaves…
8797  Teen surfer Zak Storm is mysteriously transpor…
8800  Strong-willed, middle-class Kashaf and carefre…
8803  While living alone in a spooky town, a young g…

[3475 rows x 12 columns]
```

1.7 1.6 Calcolo e Stampa del Numero Totale di Righe con Dati Mancanti

[18]:
```
# calcola il numero totale di righe con dati mancanti e lo assegna alla␣
 ↪variabile tot_dati_mancanti e poi la stampa
tot_dati_mancanti = righe_con_dati_mancanti.shape[0]
tot_dati_mancanti
```

[18]:
```
3475
```

1.8 1.7 Identificazione e Rimozione delle Righe con Valori Mancanti dal DataFrame

[ ]:
```
# identifica le righe con valori mancanti e le rimuove dal dataframe df1, poi lo stampa
df1=df.dropna(inplace=False)
df1
```

1.9 1.8 Creazione di una Matrice Booleana per Indicare Valori Mancanti nel DataFrame

[ ]:
```
# Utilizza il metodo isnull() sul DataFrame df  per creare una matrice booleana (vslori True o False)
# missing_matrix che indica se c'è un valore mancante (NaN) in ciascuna posizione del DataFrame.
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt
import numpy as np
missing_matrix = df.isnull()
missing_matrix
```

1.10 1.9 Selezione e Stampa dei Nomi delle Colonne Numeriche del DataFrame

[ ]:
```
# selezioa le colonne del Df che contengono dati numerici e le mette nella variabile numeric_cols, e poi stampa il nome delle colonne
numeric_cols = df.select_dtypes(include=['number'])
numeric_cols.columns
```

1.11 2.0 Calcolo del Numero di Valori Mancanti per Ogni Colonna in un DataFrame

[ ]:
```
# calcola il numero di valori mancanti per ogni colonna
df.isnull().sum()
```

1.12 2.1 Calcolo della Percentuale di Valori Mancanti per Ogni Colonna in un DataFrame

[ ]:
```
# calcola per ogni colonna la percentuale di valori mancanti su tutte le righe del dataframe
missing_percent = df.isnull().sum() / len(df) * 100
missing_percent
```

1.13 2.2 Calcolo della Percentuale di Valori Mancanti per Ogni Colonna in un DataFrame e Creazione del Grafico a Barre Corrispondente

[ ]:
```
# calcola per ogni colonna la percentuale di valori mancanti su tutte le righe␣
 ↪del dataframe e poi crea il grafico a barre
missing_percent= (df.isnull().sum()) / len(df) * 100
plt.figure(figsize=(10,6))
missing_percent.plot(kind='bar', color='orange', alpha=0.8)
plt.xlabel('Variabile')
plt.ylabel('% di missing values')
plt.title('missing values per colonna')
plt.xticks(rotation=90)
plt.show()
```

1.14 2.3 Visualizzazione dell'Andamento dei Paesi Produttori nel Tempo tramite un Grafico Lineare e un Box Plot

[ ]:
```
# Visualizza un grafico dei paesi produttori nel tempo
plt.figure(figsize=(2^16, 2^16))
sns.lineplot(x='country', y='release_year', data=df)
plt.title('Andamento dei paesi produttori nel tempo')
plt.xlabel('country')
plt.ylabel('anni')
plt.xticks(rotation=90)
plt.show()
# Visualizza una box plot dei paesi produttori nel tempo
plt.figure(figsize=(2^16, 2^16))
sns.boxplot(x='country', y='release_year', data=df)
plt.title('Box Plot dei paesi produttori negli anni')
plt.xlabel('paesi')
plt.ylabel('anni')
plt.xticks(rotation=90)
plt.show()
```

1.15 2.4 Suddivisione del Dataset in Training e Test Set, Creazione di un Grafico a Dispersione e Stampa delle Dimensioni dei Set

[ ]:
```
import numpy as np
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
# Suddivisione del dataset in training set (70%) e test set (30%)
X_train, X_test, y_train, y_test =␣
 ↪train_test_split(df['release_year'],df['type'],test_size=0.3,␣
 ↪random_state=42)
# Creazione di un grafico a dispersione
plt.figure(figsize=(2^16, 2^16))
plt.scatter(X_train, y_train, label='Training Set', color='blue', alpha=0.7)
```

```
plt.scatter(X_test, y_test, label='Test Set', color='orange', alpha=0.7)
plt.xlabel('anno di produzione')
plt.ylabel('paese')
plt.title('Relazione tra tipo programma e anno di produzione')
plt.legend()
plt.grid(True)
plt.show()
# Stampare le dimensioni dei training set e test set
print("Dimensioni del Training Set (tipo programma e anno di produzione):
 ↪",X_train.shape, y_train.shape)
print("Dimensioni del Test Set (tipo programma e anno di produzione):",X_test.
 ↪shape, y_test.shape)
```

1.16 2.5 Creazione di Tre Subset Casuali da un DataFrame

[21]:
```
# Creare tre subset di dimensioni simili
# primo subset: campione causale di 1/3 delle righe del df di partenza
subset1 = df.sample(frac=1/3)
# stampa il numero di righe del subset1
l1=len(subset1)
print(l1)
df = df.drop(subset1.index)
# secondo subset: campione casuale con metà delle righe rimanenti (la metà dei␣
 ↪2/3 rimanenti)
subset2 = df.sample(frac=1/2)
# stampa il numero di righe del subset2
l2=len(subset2)
print(l2)
df = df.drop(subset2.index)
# terzo subset: le righe restanti
subset3 = df
# stampa il numero di righe del subset3
l3=len(subset3)
print(l3)
```

```
2936
2936
2935
```

1.17 2.6 Calcolo delle Percentuali dei Valori Unici per il Paese nel Subset1

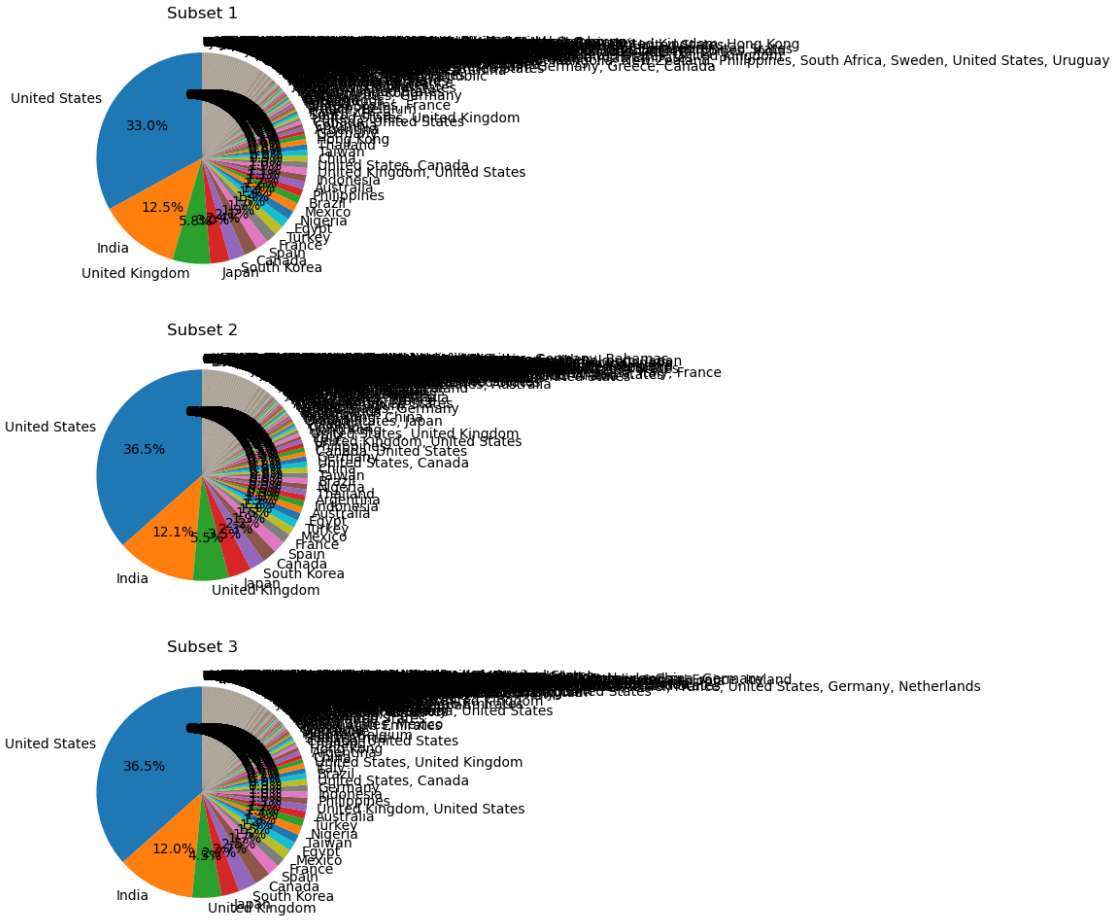
[22]:
```
percentuali_subset1 = subset1['country'].value_counts(normalize=True)
percentuali_subset1
```

[22]:
```
country
United States                 0.329981
India                         0.124767
```

```
United Kingdom                                0.057728
Japan                                         0.029795
South Korea                                   0.024209
                                                 …
United Kingdom, Hungary, Australia            0.000372
United States, China, Hong Kong               0.000372
United States, Morocco                        0.000372
India, Germany                                0.000372
United States, Hungary, Ireland, Canada       0.000372
Name: proportion, Length: 347, dtype: float64
```

1.18 2.7 Visualizzazione delle Distribuzioni dei Valori 'Country' nei Tre Subset con Grafici a Torta

[23]:
```
percentuali_subset1 = subset1['country'].value_counts(normalize=True)
percentuali_subset2 = subset2['country'].value_counts(normalize=True)
percentuali_subset3 = subset3['country'].value_counts(normalize=True)
# Creare i grafici a torta
fig, axs = plt.subplots(3, 1, figsize=(6, 12))
# Subset 1
axs[0].pie(percentuali_subset1, labels=percentuali_subset1.index, autopct='%1.
 ↪1f%%', startangle=90)
axs[0].set_title('Subset 1')
# Subset 2
axs[1].pie(percentuali_subset2, labels=percentuali_subset2.index, autopct='%1.
 ↪1f%%', startangle=90)
axs[1].set_title('Subset 2')
# Subset 3
axs[2].pie(percentuali_subset3, labels=percentuali_subset3.index, autopct='%1.
 ↪1f%%', startangle=90)
axs[2].set_title('Subset 3')
# Mostrare il grafico
plt.show()
```



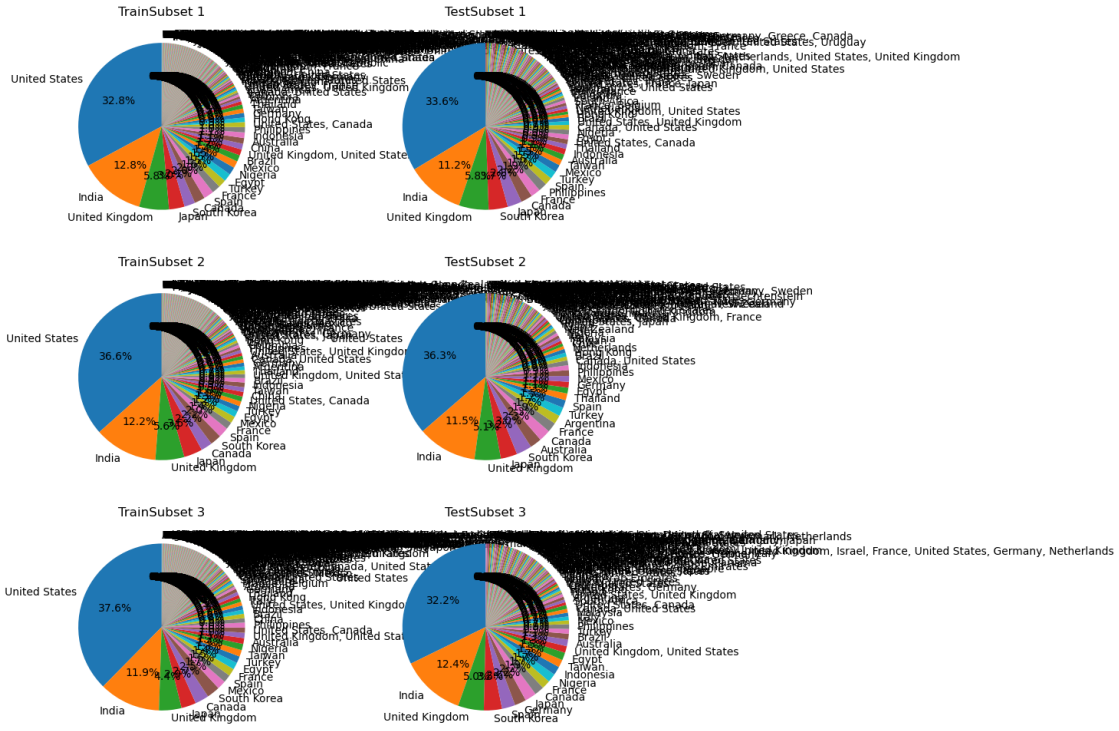
1.19 2.8 Divisione dei Subset in Training e Test Set e Visualizzazione delle Distribuzioni dei Valori 'Country' con Grafici a Torta

[24]:
```
# Dividere ciascun subset in training set e test set
train_subset1, test_subset1 = train_test_split(subset1, test_size=0.
 ↪2,random_state=42)
train_subset2, test_subset2 = train_test_split(subset2, test_size=0.
 ↪2,random_state=42)
train_subset3, test_subset3 = train_test_split(subset3, test_size=0.
 ↪2,random_state=42)
# Creare il grafico con 6 torte
fig, axs = plt.subplots(3, 2, figsize=(10, 12))
# Funzione per disegnare una torta con etichette
def draw_pie(ax, data, title):
    ax.pie(data, labels=data.index, autopct='%1.1f%%', startangle=90)
    ax.set_title(title)
# Prima riga di torte (Subset 1)
```

```
draw_pie(axs[0, 0], train_subset1['country'].value_counts(normalize=True),␣
 ↪'TrainSubset 1')

draw_pie(axs[0, 1], test_subset1['country'].value_counts(normalize=True),␣
 ↪'TestSubset 1')
# Seconda riga di torte (Subset 2)
draw_pie(axs[1, 0], train_subset2['country'].value_counts(normalize=True),␣
 ↪'TrainSubset 2')
draw_pie(axs[1, 1], test_subset2['country'].value_counts(normalize=True),␣
 ↪'TestSubset 2')
# Terza riga di torte (Subset 3)
draw_pie(axs[2, 0], train_subset3['country'].value_counts(normalize=True),␣
 ↪'TrainSubset 3')
draw_pie(axs[2, 1], test_subset3['country'].value_counts(normalize=True),␣
 ↪'TestSubset 3')
# Regolare lo spaziamento tra i subplots
plt.tight_layout()
# Mostrare il grafico
plt.show()
```

```
C:\Users\zetam\AppData\Local\Temp\ipykernel_11536\3629292171.py:22: UserWarning:
Tight layout not applied. tight_layout cannot make axes width small enough to
accommodate all axes decorations
  plt.tight_layout()
```



1.20 2.9 Identificazione degli Outliers nell'Anno di Rilascio

[10]:
```
import pandas as pd
import matplotlib.pyplot as plt
# Lista con outliers da entrambi i lati
# Calcola la media e la deviazione standard
mean_value = df['release_year'].mean()
print('media anno:')
print(mean_value)
std_dev = df['release_year'].std()
print('deviazione standard:')
print(std_dev)
# Identifica gli outliers considerando ±3 * dev_std dalla media
outliers = df[(df['release_year'] > mean_value + 3 * std_dev) |␣
 ↪(df['release_year'] < mean_value - 3 * std_dev)]
outliers
```

```
media anno:
2014.1801975701146
deviazione standard:
8.819312130834057
```

[10]:

	show_id	type	title	\
41	s42	Movie	Jaws	
42	s43	Movie	Jaws 2	
43	s44	Movie	Jaws 3	
44	s45	Movie	Jaws: The Revenge	
131	s132	Movie	Blade Runner: The Final Cut	
…	…	…	…	
8739	s8740	Movie	Why We Fight: The Battle of Russia	
8745	s8746	Movie	Willy Wonka & the Chocolate Factory	
8748	s8749	Movie	Winter of Our Dreams	
8763	s8764	Movie	WWII: Report from the Aleutians	
8792	s8793	Movie	Young Tiger	

	director	\
41	Steven Spielberg	
42	Jeannot Szwarc	
43	Joe Alves	
44	Joseph Sargent	
131	Ridley Scott	
…	…	
8739	Frank Capra, Anatole Litvak	
8745	Mel Stuart	
8748	John Duigan	
8763	John Huston	

```
8792                      Mu Chu

                                                    cast  \
41    Roy Scheider, Robert Shaw, Richard Dreyfuss, L…
42    Roy Scheider, Lorraine Gary, Murray Hamilton, …
43    Dennis Quaid, Bess Armstrong, Simon MacCorkind…
44    Lorraine Gary, Lance Guest, Mario Van Peebles,…
131   Harrison Ford, Rutger Hauer, Sean Young, Edwar…
…                                                   …
8739                                              NaN
8745  Gene Wilder, Jack Albertson, Peter Ostrum, Roy…
8748  Judy Davis, Bryan Brown, Cathy Downes, Baz Luh…
8763                                              NaN
8792  Qiu Yuen, Charlie Chin, Jackie Chan, Hu Chin, …

                                        country          date_added  \
41                                United States  September 16, 2021
42                                United States  September 16, 2021
43                                United States  September 16, 2021
44                                United States  September 16, 2021
131                               United States   September 1, 2021
…                                           …                   …
8739                              United States      March 31, 2017
8745  United States, East Germany, West Germany     January 1, 2020
8748                                  Australia    November 1, 2016
8763                              United States      March 31, 2017
8792                                  Hong Kong    November 1, 2016

      release_year rating duration  \
41            1975     PG  124 min
42            1978     PG  116 min
43            1983     PG   98 min
44            1987  PG-13   91 min
131           1982      R  117 min
…              …    …       …
8739          1943  TV-PG   82 min
8745          1971      G  100 min
8748          1981     NR   86 min
8763          1943  TV-PG   45 min
8792          1973     NR   81 min

                                              listed_in  \
41          Action & Adventure, Classic Movies, Dramas
42                   Dramas, Horror Movies, Thrillers
43        Action & Adventure, Horror Movies, Thrillers
44        Action & Adventure, Horror Movies, Thrillers
131    Action & Adventure, Classic Movies, Cult Movies
```

```
…                                                    …
8739                                        Documentaries
8745  Children & Family Movies, Classic Movies, Come…
8748                              Classic Movies, Dramas
8763                                        Documentaries
8792           Action & Adventure, International Movies

                                                description
41    When an insatiable great white shark terrorize…
42    Four years after the last deadly shark attacks…
43    After the staff of a marine theme park try to …
44    After another deadly shark attack, Ellen Brody…
131   In a smog-choked dystopian Los Angeles, blade …
…                                                    …
8739  This installment of Frank Capra's acclaimed do…
8745  Zany Willy Wonka causes a stir when he announc…
8748  After the death of a long-ago lover, married p…
8763  Filmmaker John Huston narrates this Oscar-nomi…
8792  Aided only by a tough female police officer, a…

[217 rows x 12 columns]
```

2 Utilizzo della Random Forest Classifier

[11]:
```
import category_encoders as ce
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import classification_report

# 'release_year' è la variabile target
X = df.drop('release_year', axis=1)
y = df['release_year']

# splitto i dati in train e test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)

# creo un codificatore di tutte le variabili categoriali
encoder = ce.OrdinalEncoder(cols=['show_id', 'type', 'title', 'director',␣
 ↪'cast', 'country', 'date_added', 'rating', 'duration', 'listed_in',␣
 ↪'description'])

# adatto e trasformo con il codificatore i train data
X_train = encoder.fit_transform(X_train)

# adatto e trasformo con il codificatore i test data
```

```
X_test = encoder.transform(X_test)

# inizializzo il classificatore RandomForest con 100 stime
rfc = RandomForestClassifier(random_state=42)

# addestrqa il modello al train set
rfc.fit(X_train, y_train)

# effettua le predizioni sul test set
y_pred = rfc.predict(X_test)

accuracy = accuracy_score(y_test, y_pred)
cl_rep = classification_report(y_test, y_pred)

# visualizza l'accuratezza e il report di classificazione
print("Accuratezza del modello", accuracy)
print("\nReport di classificazione:\n", cl_rep)
```

```
Accuratezza del modello 0.1424517593643587

Report di classificazione:
              precision    recall  f1-score   support

        1925       0.00      0.00      0.00         1
        1942       0.00      0.00      0.00         1
        1943       0.00      0.00      0.00         2
        1954       0.00      0.00      0.00         2
        1960       0.00      0.00      0.00         2
        1961       0.00      0.00      0.00         1
        1962       0.00      0.00      0.00         1
        1963       0.00      0.00      0.00         2
        1967       0.00      0.00      0.00         3
        1969       0.00      0.00      0.00         1
        1971       0.00      0.00      0.00         1
        1973       0.00      0.00      0.00         2
        1974       0.00      0.00      0.00         2
        1975       0.00      0.00      0.00         4
        1976       0.00      0.00      0.00         3
        1977       0.00      0.00      0.00         2
        1978       0.00      0.00      0.00         1
        1979       0.00      0.00      0.00         2
        1980       0.00      0.00      0.00         1
        1981       0.00      0.00      0.00         2
        1982       0.00      0.00      0.00         6
        1983       0.00      0.00      0.00         1
        1984       0.00      0.00      0.00         3
        1985       0.00      0.00      0.00         3
        1986       0.00      0.00      0.00         3
```

```
        1987       0.00      0.00      0.00         1
        1988       0.00      0.00      0.00         2
        1989       0.00      0.00      0.00         5
        1990       0.00      0.00      0.00         3
        1991       0.00      0.00      0.00         3
        1992       0.00      0.00      0.00         6
        1993       0.00      0.00      0.00         4
        1994       0.00      0.00      0.00         3
        1995       0.00      0.00      0.00         5
        1996       0.00      0.00      0.00         9
        1997       0.00      0.00      0.00         9
        1998       0.00      0.00      0.00         7
        1999       0.00      0.00      0.00         4
        2000       0.17      0.17      0.17         6
        2001       0.00      0.00      0.00         5
        2002       0.00      0.00      0.00         5
        2003       0.00      0.00      0.00        16
        2004       0.00      0.00      0.00        17
        2005       0.00      0.00      0.00        14
        2006       0.00      0.00      0.00        26
        2007       0.00      0.00      0.00        17
        2008       0.02      0.08      0.03        24
        2009       0.00      0.00      0.00        33
        2010       0.20      0.02      0.04        42
        2011       0.00      0.00      0.00        39
        2012       0.14      0.02      0.03        51
        2013       0.00      0.00      0.00        59
        2014       0.00      0.00      0.00        69
        2015       0.03      0.01      0.01       101
        2016       0.16      0.09      0.11       194
        2017       0.13      0.28      0.18       202
        2018       0.16      0.30      0.21       211
        2019       0.14      0.17      0.15       189
        2020       0.15      0.26      0.19       193
        2021       0.30      0.18      0.23       136

    accuracy                           0.14      1762
   macro avg       0.03      0.03      0.02      1762
weighted avg       0.12      0.14      0.12      1762


C:\Users\zetam\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1469: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\zetam\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1469: UndefinedMetricWarning:
```

```
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\zetam\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1469: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

3 USO DEI MODELLI

3.1 Uso della Logistic Regression

[4]:
```
import category_encoders as ce
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score, classification_report

# 'release_year' è la variabile target
X = df.drop('release_year', axis=1)
y = df['release_year']

# splitto i dati in train e test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
# creo un codificatore di tutte le variabili categoriali
encoder = ce.OrdinalEncoder(cols=['show_id', 'type', 'title', 'director',␣
 ↪'cast', 'country', 'date_added', 'rating', 'duration', 'listed_in',␣
 ↪'description'])

# adatto e trasformo con il codificatore i train data
X_train = encoder.fit_transform(X_train)

# adatto e trasformo con il codificatore i test data
X_test = encoder.transform(X_test)

# inizializzazione del classificatore di regressioni logistica
log_reg_classifier = LogisticRegression(random_state=42)

# addestra il modello sul train set
log_reg_classifier.fit(X_train, y_train)

# effettua le predizioni sul test set
y_pred = log_reg_classifier.predict(X_test)

# valutazione prestazioni modello
accuracy = accuracy_score(y_test, y_pred)
```

```
cl_rep = classification_report(y_test, y_pred)

# visualizza l'accuratezza e il report di classificazione
print("Accuratezza del modello", accuracy)
print("\nReport di classificazione:\n", cl_rep)
```

```
Accuratezza del modello 0.13904653802497163

Report di classificazione:
              precision    recall  f1-score   support

        1925       0.00      0.00      0.00         1
        1942       0.00      0.00      0.00         1
        1943       0.00      0.00      0.00         2
        1954       0.00      0.00      0.00         2
        1960       0.00      0.00      0.00         2
        1961       0.00      0.00      0.00         1
        1962       0.00      0.00      0.00         1
        1963       0.00      0.00      0.00         2
        1967       0.00      0.00      0.00         3
        1969       0.00      0.00      0.00         1
        1971       0.00      0.00      0.00         1
        1973       0.00      0.00      0.00         2
        1974       0.00      0.00      0.00         2
        1975       0.00      0.00      0.00         4
        1976       0.00      0.00      0.00         3
        1977       0.00      0.00      0.00         2
        1978       0.00      0.00      0.00         1
        1979       0.00      0.00      0.00         2
        1980       0.00      0.00      0.00         1
        1981       0.00      0.00      0.00         2
        1982       0.00      0.00      0.00         6
        1983       0.00      0.00      0.00         1
        1984       0.00      0.00      0.00         3
        1985       0.00      0.00      0.00         3
        1986       0.00      0.00      0.00         3
        1987       0.00      0.00      0.00         1
        1988       0.00      0.00      0.00         2
        1989       0.00      0.00      0.00         5
        1990       0.00      0.00      0.00         3
        1991       0.00      0.00      0.00         3
        1992       0.00      0.00      0.00         6
        1993       0.00      0.00      0.00         4
        1994       0.00      0.00      0.00         3
        1995       0.00      0.00      0.00         5
        1996       0.00      0.00      0.00         9
        1997       0.00      0.00      0.00         9
        1998       0.00      0.00      0.00         7
```

```
        1999       0.00      0.00      0.00         4
        2000       0.00      0.00      0.00         6
        2001       0.00      0.00      0.00         5
        2002       0.00      0.00      0.00         5
        2003       0.00      0.00      0.00        16
        2004       0.00      0.00      0.00        17
        2005       0.00      0.00      0.00        14
        2006       0.00      0.00      0.00        26
        2007       0.00      0.00      0.00        17
        2008       0.00      0.00      0.00        24
        2009       0.00      0.00      0.00        33
        2010       0.00      0.00      0.00        42
        2011       0.00      0.00      0.00        39
        2012       0.00      0.00      0.00        51
        2013       0.00      0.00      0.00        59
        2014       0.00      0.00      0.00        69
        2015       0.00      0.00      0.00       101
        2016       0.12      0.29      0.18       194
        2017       0.16      0.11      0.13       202
        2018       0.10      0.09      0.10       211
        2019       0.00      0.00      0.00       189
        2020       0.13      0.47      0.20       193
        2021       0.21      0.40      0.28       136

    accuracy                           0.14      1762
   macro avg       0.01      0.02      0.01      1762
weighted avg       0.07      0.14      0.09      1762


C:\Users\zetam\anaconda3\Lib\site-
packages\sklearn\linear_model\_logistic.py:460: ConvergenceWarning: lbfgs failed
to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.

Increase the number of iterations (max_iter) or scale the data as shown in:
    https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
    https://scikit-learn.org/stable/modules/linear_model.html#logistic-
regression
  n_iter_i = _check_optimize_result(
C:\Users\zetam\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1469: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\zetam\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1469: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
```

```
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\zetam\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1469: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```

3.2 Uso della Support Vector Classifier

[25]:
```
import category_encoders as ce
from sklearn.model_selection import train_test_split
from sklearn.svm import SVC
from sklearn.metrics import accuracy_score, classification_report

# 'release_year' è la variabile target
X = df.drop('release_year', axis=1)
y = df['release_year']

# splitto i dati in train e test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,␣
 ↪random_state=42)
# creo un codificatore di tutte le variabili categoriali
encoder = ce.OrdinalEncoder(cols=['show_id', 'type', 'title', 'director',␣
 ↪'cast', 'country', 'date_added', 'rating', 'duration', 'listed_in',␣
 ↪'description'])

# adatto e trasformo con il codificatore i train data
X_train = encoder.fit_transform(X_train)

# adatto e trasformo con il codificatore i test data
X_test = encoder.transform(X_test)
# inizializzazione del classificatore SVC
svm_cl = SVC(random_state=42)

# addestra il modello sul train set
svm_cl.fit(X_train, y_train)

# effettua le predizioni sul test set
y_pred = svm_cl.predict(X_test)

# valutazione prestazioni modello
accuracy = accuracy_score(y_test, y_pred)
cl_rep = classification_report(y_test, y_pred)

# visualizza l'accuratezza e il report di classificazione
print("Accuratezza del modello", accuracy)
```

```
print("\nReport di classificazione:\n", cl_rep)
```

```
Accuratezza del modello 0.12776831345826234

Report di classificazione:
               precision    recall  f1-score   support

        1944       0.00      0.00      0.00         1
        1963       0.00      0.00      0.00         1
        1967       0.00      0.00      0.00         1
        1968       0.00      0.00      0.00         1
        1971       0.00      0.00      0.00         1
        1974       0.00      0.00      0.00         2
        1978       0.00      0.00      0.00         1
        1980       0.00      0.00      0.00         1
        1981       0.00      0.00      0.00         1
        1982       0.00      0.00      0.00         5
        1983       0.00      0.00      0.00         1
        1986       0.00      0.00      0.00         1
        1989       0.00      0.00      0.00         1
        1990       0.00      0.00      0.00         1
        1992       0.00      0.00      0.00         1
        1993       0.00      0.00      0.00         3
        1994       0.00      0.00      0.00         2
        1995       0.00      0.00      0.00         2
        1996       0.00      0.00      0.00         1
        1997       0.00      0.00      0.00         4
        1998       0.00      0.00      0.00         2
        1999       0.00      0.00      0.00         5
        2000       0.00      0.00      0.00         5
        2001       0.00      0.00      0.00         5
        2002       0.00      0.00      0.00         4
        2003       0.00      0.00      0.00         4
        2004       0.00      0.00      0.00         7
        2005       0.00      0.00      0.00         4
        2006       0.00      0.00      0.00         6
        2007       0.00      0.00      0.00         6
        2008       0.00      0.00      0.00        10
        2009       0.00      0.00      0.00         9
        2010       0.00      0.00      0.00         8
        2011       0.00      0.00      0.00        16
        2012       0.00      0.00      0.00        17
        2013       0.00      0.00      0.00        25
        2014       0.00      0.00      0.00        22
        2015       0.00      0.00      0.00        36
        2016       0.00      0.00      0.00        46
        2017       0.26      0.13      0.17        78
        2018       0.12      0.87      0.21        70
```

```
        2019       0.00      0.00      0.00        66
        2020       0.17      0.07      0.10        61
        2021       0.00      0.00      0.00        43

    accuracy                           0.13       587
   macro avg       0.01      0.02      0.01       587
weighted avg       0.07      0.13      0.06       587

C:\Users\zetam\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1469: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\zetam\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1469: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
C:\Users\zetam\anaconda3\Lib\site-
packages\sklearn\metrics\_classification.py:1469: UndefinedMetricWarning:
Precision and F-score are ill-defined and being set to 0.0 in labels with no
predicted samples. Use `zero_division` parameter to control this behavior.
  _warn_prf(average, modifier, msg_start, len(result))
```