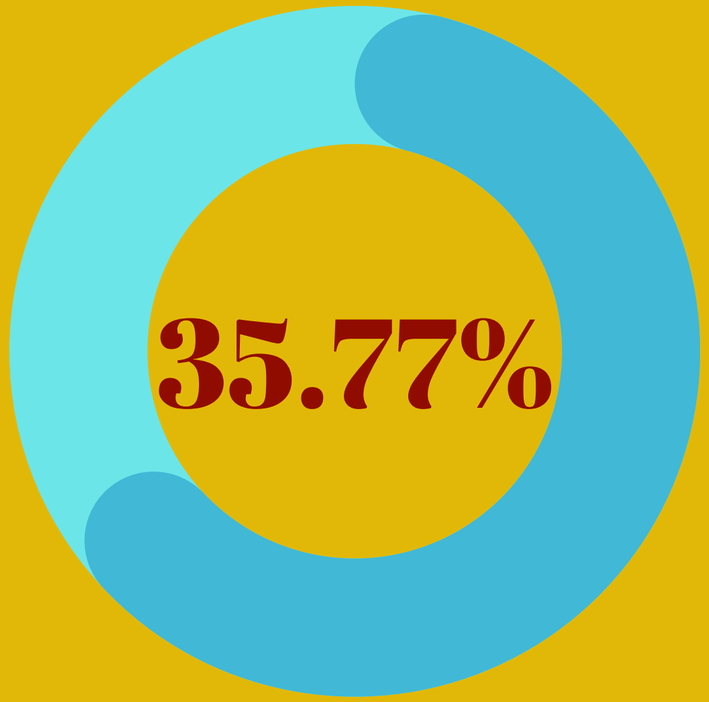
ANALISIS PREDICTIVO

TP-2 COMPETENCIA KAGGLE

Matias Jose Capurro

Score obtenido:



21	—	**Matias Capurro**		0.35773	9	6d

ANÁLISIS EXPLORATORIO DE DATOS

1

Observaciones generales

2

Estudio de valores nulo

3

Outliers

4

Correlaciones y encoding

```
RangeIndex: 977541 entries, 0 to 977540
Data columns (total 29 columns):
 #   Column                Non-Null Count   Dtype
---  ------                --------------   -----
 0   Unnamed: 0            977541 non-null  int64
 1   averageRating         977541 non-null  float64
 2   numVotes              977541 non-null  int64
 3   titleType             977541 non-null  object
 4   isAdult               977541 non-null  float64
 5   startYear             977541 non-null  int64
 6   endYear               977541 non-null  int64
 7   runtimeMinutes        977541 non-null  int64
 8   genres_x              977539 non-null  object
 9   directors             977541 non-null  object
 10  writers               977541 non-null  object
 11  seasonNumber          438243 non-null  float64
 12  episodeNumber         438243 non-null  float64
 13  ordering              370623 non-null  float64
 14  language              370623 non-null  object
 15  attributes            370623 non-null  object
 16  isOriginalTitle       370623 non-null  float64
 17  adult                 47370 non-null   object
 18  budget                47370 non-null   float64
 19  genres_y              47370 non-null   object
 20  original_language     47358 non-null   object
 21  popularity            47369 non-null   float64
 22  production_companies  47369 non-null   object
 23  production_countries  47369 non-null   object
 24  revenue               47369 non-null   float64
 25  runtime               47158 non-null   float64
```

Observaciones generales

Cantidad de filas
Cantidad de columnas
Valores no nulos
Tipos de dato

VALORES NULOS

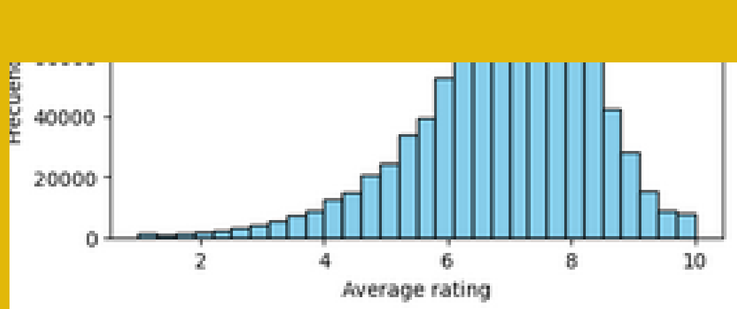
Se decide borrar todas las variables con porcentaje de nulos superior a 95%

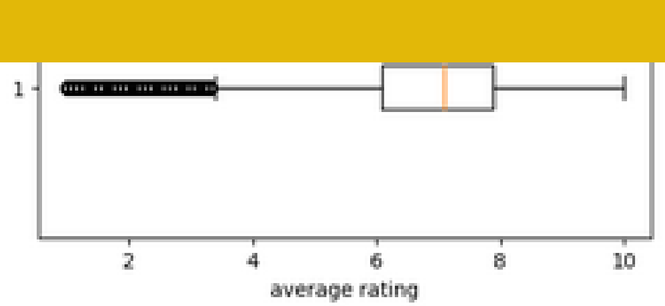
Tambien, la columna unnamed: o ya que no aporta informacion (id)

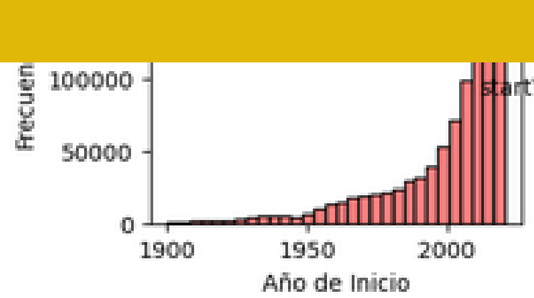
```
Porcentaje de valores nulos por columna:
Unnamed: 0              0.000000
averageRating           0.000000
numVotes                0.000000
titleType               0.000000
isAdult                 0.000000
startYear               0.000000
endYear                 0.000000
runtimeMinutes          0.000000
genres_x                0.000002
directors               0.000000
writers                 0.000000
seasonNumber            0.551688
episodeNumber           0.551688
ordering                0.620862
language                0.620862
attributes              0.620862
isOriginalTitle         0.620862
adult                   0.951542
budget                  0.951542
genres_y                0.951542
original_language       0.951554
popularity              0.951543
production_companies    0.951543
production_countries    0.951543
```

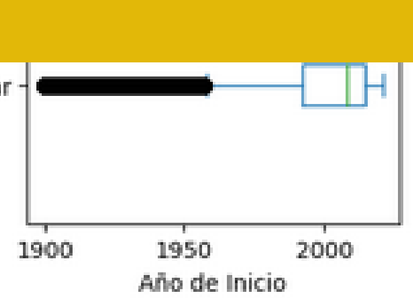
A los objetos de formato json se les aplico una funcion para convertirlos en string

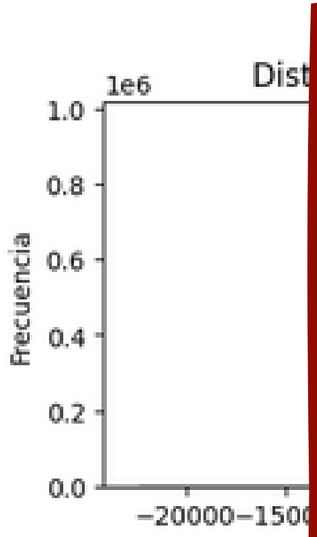
```python
# Función para convertir valores JSON en una cadena de nombres separados por comas
def convertir_a_cadena(valor):
    try:
        # Si el valor es "nan" o una lista vacía, devolver una cadena vacía
        if valor == "nan" or valor == "[]" or pd.isna(valor):
            return ""

        # Intenta cargar el JSON y obtener los nombres de los génerosr)
        valor = valor.replace("'", "\"")
        json_data = json.loads(valor)
        if isinstance(json_data, list):
            nombres = [item["name"] for item in json_data]
            return ", ".join(nombres)
        else:
            return ""
    except (json.JSONDecodeError, TypeError):
        return ""

# Aplicar la función a la columna "genres_y" y crear una nueva columna "genres_str"
df['genres_str'] = df['genres_y'].apply(convertir_a_cadena)
df['companies_str'] = df['production_companies'].apply(convertir_a_cadena)
df['countries_str'] = df['production_countries'].apply(convertir_a_cadena)
```

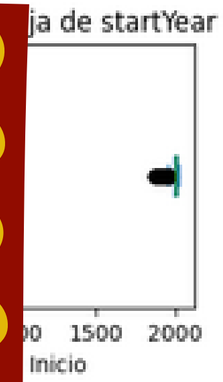






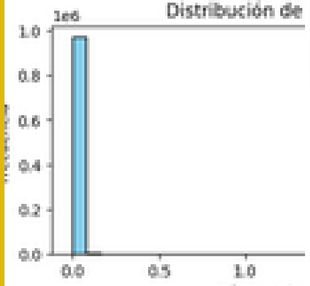


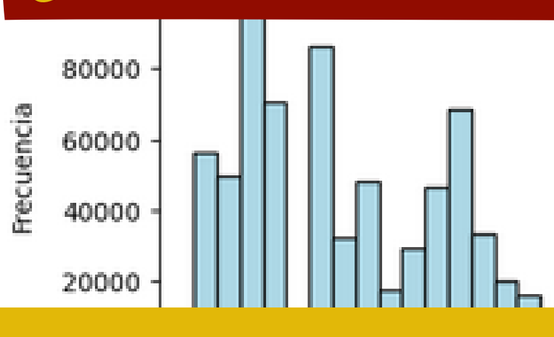




OUTLIERS

Calculando y graficando métricas de todas las variables, se descartaron outliers con diferentes criterios, null o eliminacion





DESCARTE DE NUEVAS COLUMNAS

POR EXCESO DE VALORES SIN INFORMACION

```
count    974894.000000
mean         58.354727
std         336.897800
min           0.000000
25%           0.000000
50%           0.000000
75%           0.000000
max        2022.000000
Name: endYear, dtype: float64
0.9708696535213059
```

```
Porcentaje de valores nulos por columna:
averageRating      0.000000
numVotes           0.000000
titleType          0.000000
isAdult            0.000000
startYear          0.000000
runtimeMinutes     0.259868
genres_x           0.000000
directors          0.000000
writers            0.000000
seasonNumber       0.546673
episodeNumber      0.574986
ordering           0.622822
language           0.613754
attributes         0.613754
```

ME QUEDAN 11 VARIABLES EXPLICATICAS

ENCODING

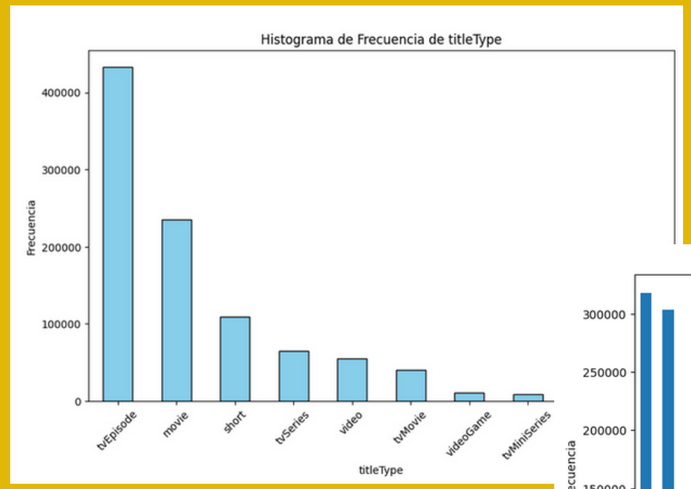
```python
all_writers = []
for writers in df['writers']:
    if isinstance(writers, str):  # Solo procesar las cadenas
        writers_list = writers.split(',')
        all_writers.extend(writers_list)

# Contar la frecuencia de cada género
df['writers'] = df['writers'].replace('0', np.nan)
writers_counts = pd.Series(all_writers).value_counts()
print(writers_counts)
writers_count = df['writers'].nunique()

print("Cantidad de escritores únicos:", directors_count)
```

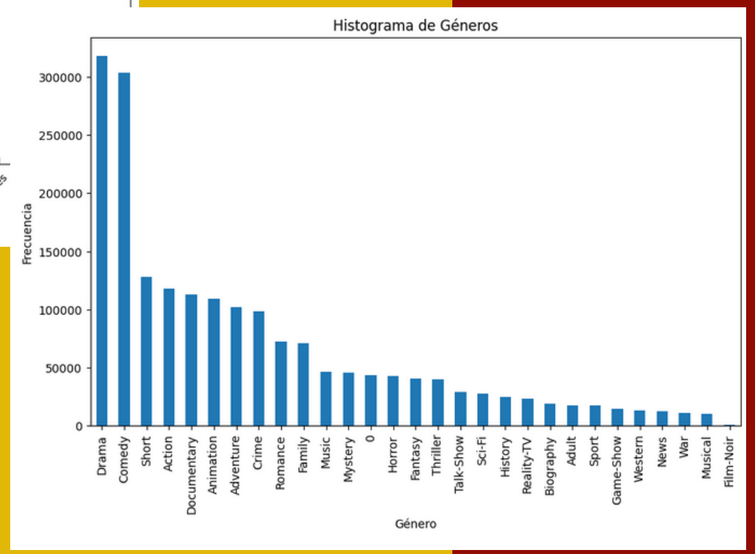
```
0            218592
nm0829537      3306
nm0935540      3123
nm0810431      3116
nm0170306      2350
              ...
nm0476777         1
nm0701155         1
nm1691051         1
nm2748904         1
nm3948180         1
Name: count, Length: 342302, dtype: int64
Cantidad de escritores únicos: 239397
```

ESTUDIO DE VARIABLES CATEGORICAS

- Procesamiento de cadenas
- Análisis de dimensiones
- Metodo de encoding

Estudio de las variables categoricas

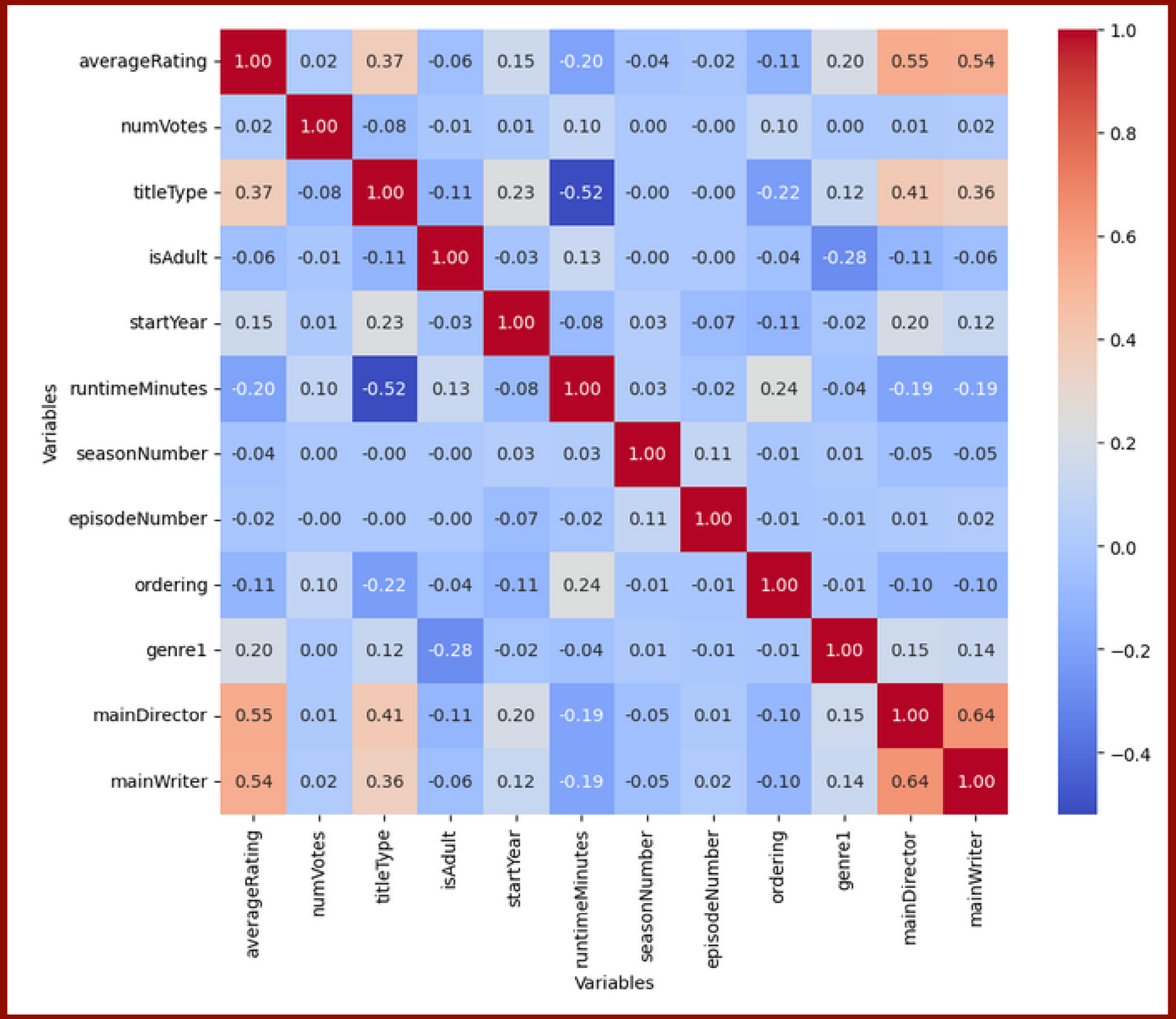




VARIABLES CATEGORICAS CON MUCHAS CATEGORIAS

Como un metodo one hot encoding transformando las variables en binarias aumentaria demasiado la dimensionalidad, se busca hacer un encoding por target

Matriz de correlacion



ENODING BASADO EN TARGET

Se agrega una columna con la mediana del rating de cada director y escritor

```
mean_runtimeMinutes = df['runtimeMinutes'].median()
mean_seasonNumber = df['seasonNumber'].median()
mean_episodeNumber = df['episodeNumber'].median()
mean_ordering = df['ordering'].median()


# Llena los valores nulos con las medias calculadas
df['runtimeMinutes'].fillna(mean_runtimeMinutes, inplace=True)
df['seasonNumber'].fillna(mean_seasonNumber, inplace=True)
df['episodeNumber'].fillna(mean_episodeNumber, inplace=True)
df['ordering'].fillna(mean_ordering, inplace=True)

testeoKaggle['runtimeMinutes'].fillna(mean_runtimeMinutes, inplace=True)
testeoKaggle['seasonNumber'].fillna(mean_seasonNumber, inplace=True)
testeoKaggle['episodeNumber'].fillna(mean_episodeNumber, inplace=True)
testeoKaggle['ordering'].fillna(mean_ordering, inplace=True)
```

seasonNumber	episodeNumber	ordering	mainWriter
23.0	12.0	1.0	6.909469
2.0	2.0	2.0	8.282087
2.0	7.0	1.0	6.559465
2.0	7.0	2.0	7.168698
1.0	44.0	2.0	6.709306

MODELOS PRINCIPALES TESTEADOS

REGRESION LINEAL	MODELO NO PARAMETRICO RIDGE	XGBOOST	RANDOM FOREST
MODELO DE INICIACION Y PRUEBA	MODELO PARA INTENTAR REDUCIR DIMENSIONALIDAD PARA MEJORAR EL MODELO	MODELO	MODELO CON LOGICA DE ARBOLES DE DECISION, FINALMENTE UTILIZADO PARA EL ANALISIS

HIPERPARAMETROS

```python
param_grid = {
    'n_estimators': [100, 200, 300],
    'max_depth': [10, 20, 30, 40, 50],
    # Agrega más hiperparámetros aquí
}
from sklearn.model_selection import GridSearchCV

grid_search = GridSearchCV(estimator=random_forest_model, param_grid=param_grid, cv=5, scoring='r2', n_jobs=-1)
grid_search.fit(X_train, y_train)
best_params = grid_search.best_params_
best_model = grid_search.best_estimator_
y_pred = best_model.predict(X_test)
mse = mean_squared_error(y_test, y_pred)
r2 = r2_score(y_test, y_pred)

print("Mejores hiperparámetros:", best_params)
print(f"Error Cuadrático Medio (MSE): {mse}")
print(f"Coeficiente de Determinación (R^2): {r2}")
```

REGRESION LINEAL

SCORE:

30,31%

```
Error Cuadrático Medio (MSE): 1.3850071667855899
Coeficiente de Determinación (R^2): 0.30307284412416047
```

RIDGE

SCORE:

30,31%

```
Ridge Regression:
Error Cuadrático Medio (MSE): 1.3849766870145799
Coeficiente de Determinación (R^2): 0.303088181358964
```

XGB

SCORE:

45,31%

```
Error Cuadrático Medio (MSE): 1.0868396958368134
Coeficiente de Determinación (R^2): 0.45310889627347906
```

```
Coeficiente de Determinación (R^2) promedio en validación cruzada: 0.675512169096428
```

RANDOM FOREST

SCORE:

34.70%

```
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, r2_score
```

```
random_forest_model = RandomForestRegressor(n_estimators=600, random_state=42, max_depth=20)
```

```
MSE: 1.2976941642484117
R^2: 0.3470082142713359
```

Aspectos a mejorar

CODIFICACION

Al haber tenido mas tiempo se podria haber probado diferentes tipos de encoding y de filtrado de datos

PROCESAMIENTO

Al haber tenido mas recursos se podria mejorar la eficiencia de la computadora

METADATA

Al haber tenido mas recursos y tiempo, se podria estudiar la metadata detras de la db