12/8/2020

DATA CLEANING & TRANSFORMATION RESULTS

Team 6
Oguzhan Akan
Darius Hooks
Jaymish Raju Patel
Jason Sabal
Bibinur Zhursinbek

California State University, Northridge
COMP 541 – Data Mining – F2020
Assignment 4

Contents

2. Data Cleaning
2.1. Identifying Missing Values

By observing the data, we see that only the following columns need initial transformation:

- overview
- tagline
- spoken_languages_edited
- production_countries_edited
- keywords_edited

And the remaining features do not have to change since they do not contain zero values.

```
df.isnull().sum()
```

budget	0
company	0
country	0
director	0
genre	0
gross	0
name	0
rating	0
released	0
runtime	0
score	0
star	0
votes	0
writer	0
year	0
isprofit	0
profitability_ratio	0
profitability_ratio_bucket	0
adult	0
id	0
imdb_id	0
original_title	0
overview	1
popularity	0
tagline	371
title	0
genres_edited	0
spoken_languages_edited	7
production_countries_edited	31
keywords_edited	167
year_released	0

dtype: int64

2. Data Cleaning
2.2. Dealing with Missing Values

- The previously mentioned featured had to change due to containing missing or zero values. Therefore we apply a replacement function to replace all zero values with the value None.

```
df['overview'] = df['overview'].map(lambda x:x if x != 0 else None)
df['tagline'] = df['tagline'].map(lambda x:x if x != 0 else None)
df['spoken_languages_edited'] = df['spoken_languages_edited'].map(lambda x:x if x != 0 else None)
df['production_countries_edited'] = df['production_countries_edited'].map(lambda x:x if x != 0 else None)
df['keywords_edited'] = df['keywords_edited'].map(lambda x:x if x != 0 else None)
```

2. Data Cleaning
2.3. Removing Rows

- The figure on the left is used to remove rows with missing values and will be left with a complete dataset
- With the figure on the right we can se that after removing about 14% of the rows from our dataset we see an increase in the amount of profitable films with1 meaning profit and 0 meaning no profit

```
data_dropped = df.dropna()
```

```
data_dropped = data_dropped.reset_index()
del data_dropped['index']
```

```
data_dropped.shape
```

```
(3031, 31)
```

```
df.shape
```

```
(3524, 31)
```

```
df['isprofit'].value_counts(normalize=True)   #  split of trues and falses before rows dropped
```

```
1    0.510499
0    0.489501
Name: isprofit, dtype: float64
```

```
data_dropped['isprofit'].value_counts(normalize=True)   #  split of trues and falses after rows dropped
```

```
1    0.533157
0    0.466843
Name: isprofit, dtype: float64
```

2. Data Cleaning
2.4. & 2.4.1 Finding Outliers - IQR Score method

- Using the code below we find outliers in our dataset using the IQR Score method. Please refer to the HTML report for detailed result table with true or false values to tell whether a value is an outlier or not.

```
Q1 = data_dropped.quantile(0.25)
Q3 = data_dropped.quantile(0.75)
IQR = Q3 - Q1
print(IQR)
```

```
print((data_dropped < (Q1 - 1.5 * IQR)) | (data_dropped > (Q3 + 1.5 * IQR)))
```

```
data_dropped.shape
```

```
(3031, 31)
```

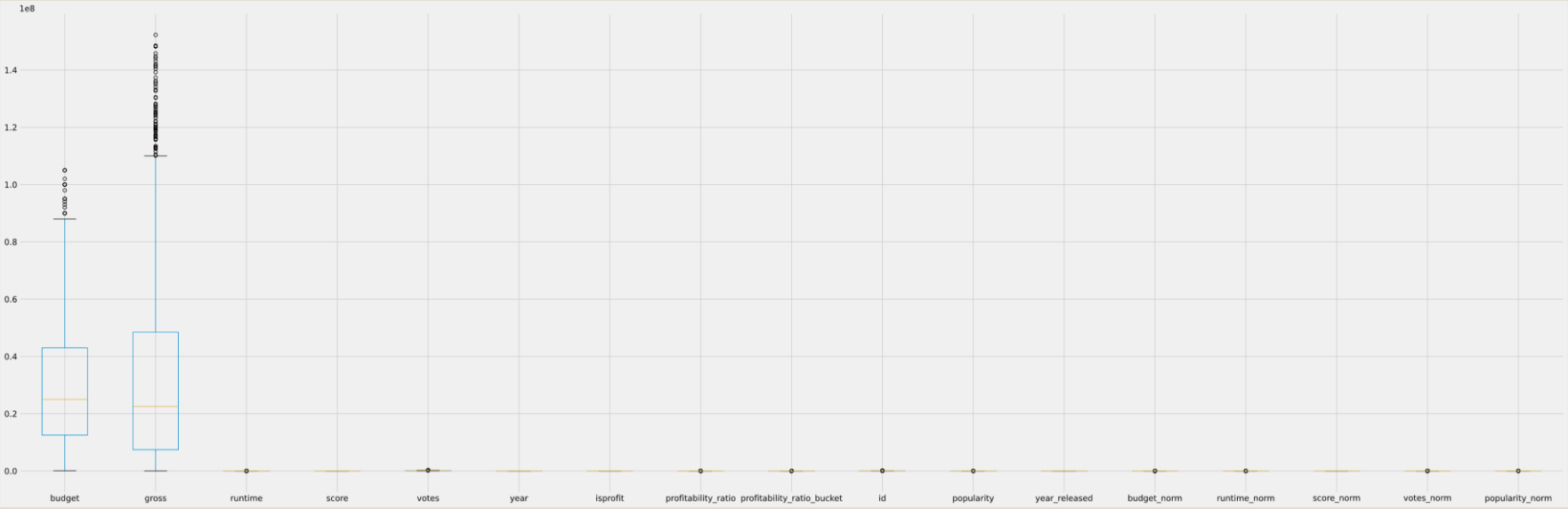
```
data_dropped_outlier_IQR = data_dropped[~((data_dropped < (Q1 - 1.5 * IQR)) | (data_dropped > (Q3 + 1.5 * IQR))).any(axis=1)]
```

```
data_dropped_outlier_IQR.shape
```

```
(1993, 31)
```

2. Data Cleaning
2.5. Box Plot

- This figure shows the distribution of data after dropping outliers. The graph represents five number summary: minimum, first quartile, median, third quartile and maximum. The ends of the box are the quartiles, Q1 and Q3, while the box length is interquartile range (IQR). Two lines (called whiskers) outside the box extend to the smallest (Minimum) and largest (Maximum) observations. The circles are the outliers. As we can see in our figure budget and gross are the only two features that have outliers passed the maximum point being the top most black bar.



3. Data Transformation
3.1. Binning

- Our data has the following columns that we can perform transformation on:

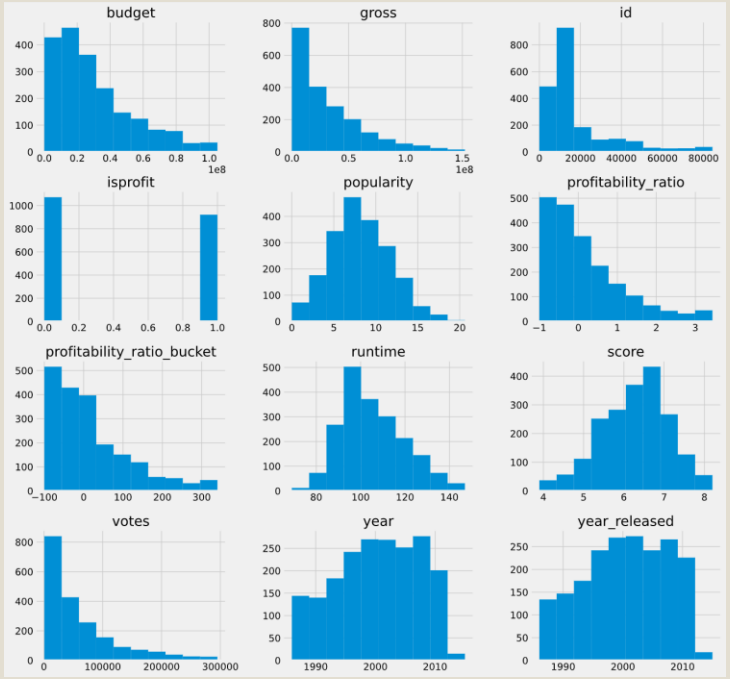
  budget, runtime, score, votes and popularity

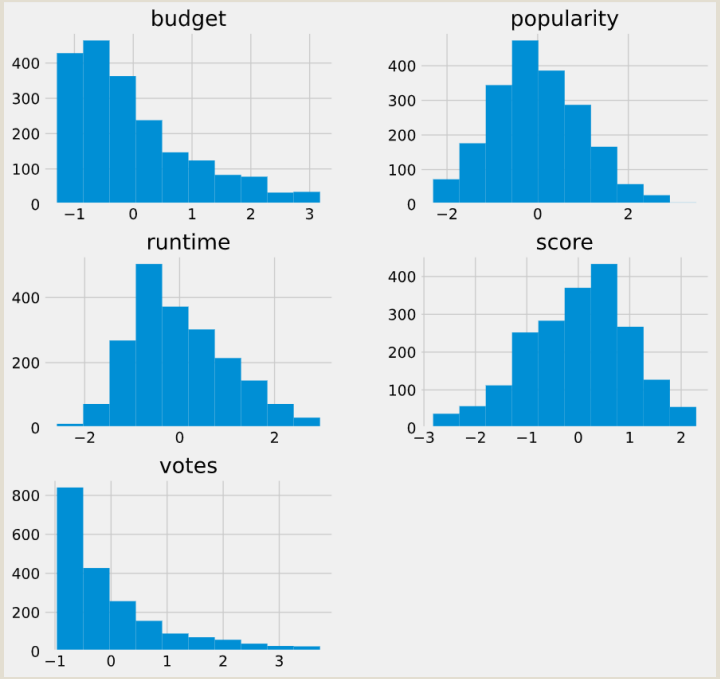
- Using the code below we discretize the data before moving on to normalizing. Please refer to the HTML report to see the distributions that the discretized data is partitioned into.

```
for i in continuous_features:
    display(data_dropped_outlier_IQR[i].describe())

    display(pd.qcut(data_dropped_outlier_IQR[i], q=4))

    display(pd.qcut(data_dropped_outlier_IQR[i], q=4).value_counts())
```

3. Data Transformation
3.2. & 3.2.1. Normalization – Z-score Method

The figure on the left is a histogram of all the features with numerical values while the figure on the right is a histogram of the specific five values we found important to normalize for use in our model.





4. Results
4.1 Data Cleaning

As we analyze all the features in Section 2.1. we can see that the only features that need cleaning include:

- overview
- tagline
- spoken_languages_edited
- production_countries_edited
- keywords_edited
- The rest of the features produced an output of having zero missing values which means that they didn't require either dropping or replacement of the values they contained. After cleaning the data we were able to see that our "isprofit" feature grew slightly in percentage.

4. Results
4.2 Data Transformation

- After analyzing and cleaning the five categorical features found in Section 2.1. we are able to focus on transforming our data.
- Our data includes categorical and continuous types but for transformation we want to focus on five that need to be discretized and normalized in order to be ready for modeling. These five features include budget, runtime, score, votes and popularity. However, after discretizing and normalizing we are left with a little over 50% of our dataset. This may hinder our modeling process since we are working with less data. In, conclusion after cleaning and transforming our data we hope it will be accurate enough to conduct our models.