The Ultimate Film Statistics

WUM Projekt 2 - Klasteryzcja

June 2024

Członkowie zespołu

Zofia Kamińska, Krzysztof Adamczyk, Mikołaj Guzik

Walidatorzy

Karolina Dunal, Miłosz Chodań

Spis treści

1 Temat projektu

Dane: **The Ultimate Film Statistics Dataset - for ML**.

Zbiór danych zawiera informacje na temat różnego rodzaju filmów. W 14 kolumnach znajdują się cechy takie jak: tytuły, oceny filmów, wydatki, zarobki, daty produkcji, a także dokładne informacje o reżyserach takie jak data urodzenia, śmierci oraz role które pełnił podczas produkcji filmowych (dokładna lista kolumn znajduje się w kolejnej sekcji).

2 Model Biznesowy

Projekt ten ma na celu wykorzystanie uczenia maszynowego do klasteryzacji filmów na podstawie ich różnorakich cech i statystyk.

Wybrano dwa potencjalne zastosowania biznesowe:

1. Stworzenie modelu, który dla dowolnego filmu dostępnego w ramce danych, przyporządkuje nam kilka rekomendacji na obejrzenie następnego filmu.
2. Stworzenie modelu, dla producentów filmowych, który na podstawie dostępnych cech tj. ranking filmów i dochód filmu, przedstawi najlepsze cechy do tworzenia przyszłych produkcji.

3 Eksploracyjna analiza danych

3.1 Cechy

Opis kolumn:

- Movie_title
- Production_date
- Genres
- Runtime_minutes
- Director_name
- Director_professions
- Director_birthYear
- Director_deathYear
- Movie_averageRating (podane przez użytkowników online dla danego filmu)
- Movie_numberOfVotes (podanych przez użytkowników online dla danego filmu)
- Approval_Index (znormalizowany wskaźnik (w skali 0-10) obliczany przez pomnożenie logarytmu liczby głosów przez średnią ocenę użytkowników)
- Production_budget ($)
- Domestic_gross ($)
- Worldwide_gross ($)

3.2 Podział danych

W ramce danych w sumie znajduje się 4380 rekordów. Dane zostały podzielone na zbiór modelarzy i walidatorów w proporcji 70-30.

```
1 Nasze dane:            (3504, 14)
2 Dane walidatorow:      (876,14)
```

3.3 Opis ramki danych - brakujące wartości

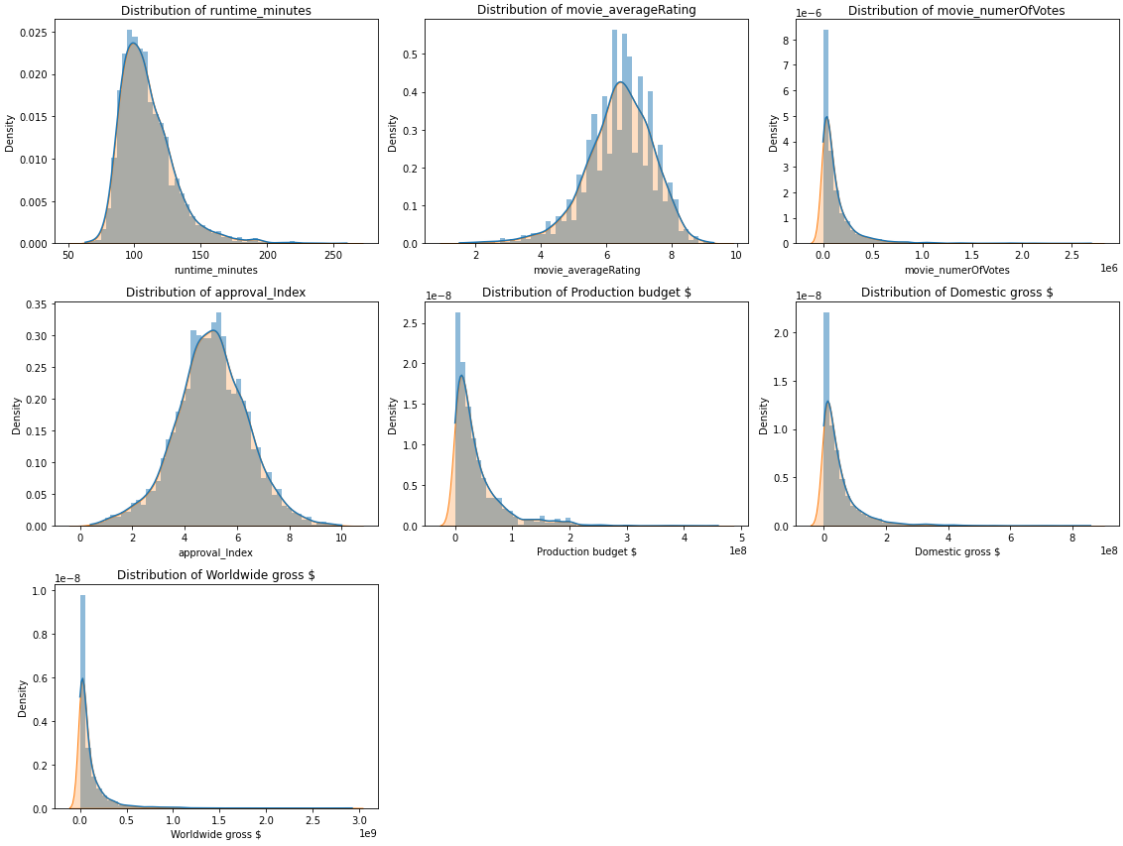
W naszych danych nie znaleźliśmy żadnych wartości NULL, ani definitywnych outlierów do usunięcia. Jednak w niektórych kolumnach znajdowały się brakujące wartości w innej postaci, głównie w danych dotyczących reżyserów. Zaopiekujemy się nimi odpowiednio podczas feature engineeringu.

```
wartosci brakujace w postaci "-":
director_name 265
director_professions 270
director_birthYear 265
director_deathYear 265

wartosci brakujace w postaci "\N":
director_birthYear 468
```

3.4 Rozkłady zmiennych

Przeanalizowaliśmy rozkłady poszczególnych zmiennych numerycznych w naszym zbiorze danych, aby zrozumieć ich charakterystykę.

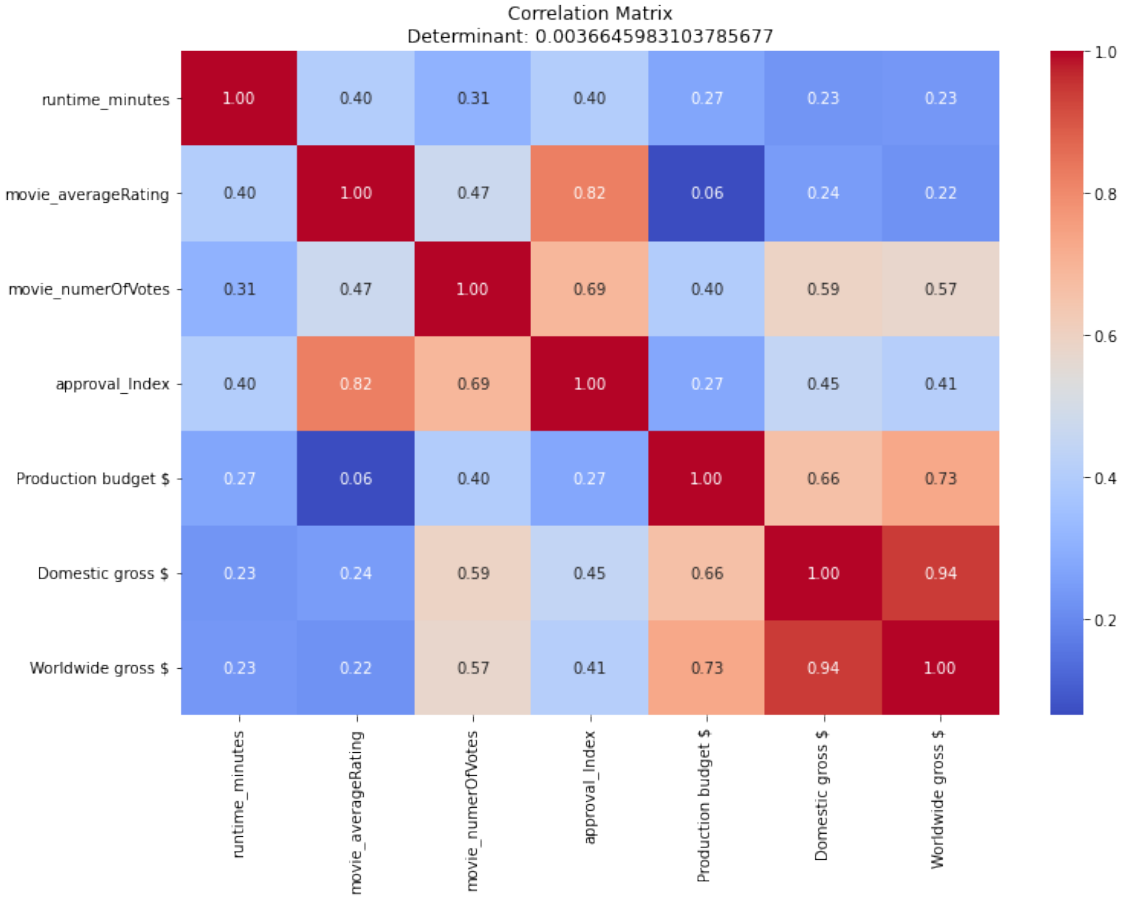


Rysunek 1: Rozkłady zmiennych numerycznych

Jak widać na powyższym obrazku rozkłady cech takich jak runtime_minutes, movie_averageRating oraz approval_index są rozkładami normalnymi. W niektórych cechach obserwuje się skośność w rozkładzie danych, co oznacza, że wartości są mocno przesunięte w jedną stronę. Szczególnie widoczne jest to w przypadku danych budżetowych, gdzie wartości często koncentrują się w niższym zakresie, co prowadzi do asymetrycznego rozkładu.

3.5 Ocena korelacji

Przeprowadziliśmy analizę korelacji między poszczególnymi atrybutami, co pozwoliło nam zidentyfikować, jak zmienne są ze sobą powiązane.

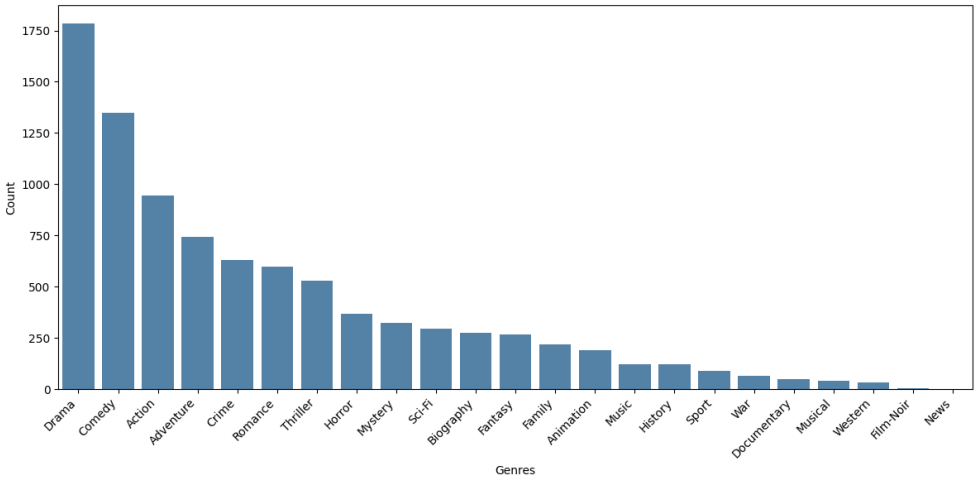


Rysunek 2: Macierz korelacji zmiennych, bardziej intesywne kolory oznaczają większą korelację między zmiennymi.

Jak widać, budżety są między sobą dość mocno pozytywnie skorelowane, co nie dziwi. Ciekawe jednak jest to że movie_average_rating oraz approval_indeks mają ujemną korelację z zarobkami filmu, gdyż zdawałoby się że filmy bardziej popularne wśród internautów, oraz z wyższą średnią oceną zarabiałyby więcej.

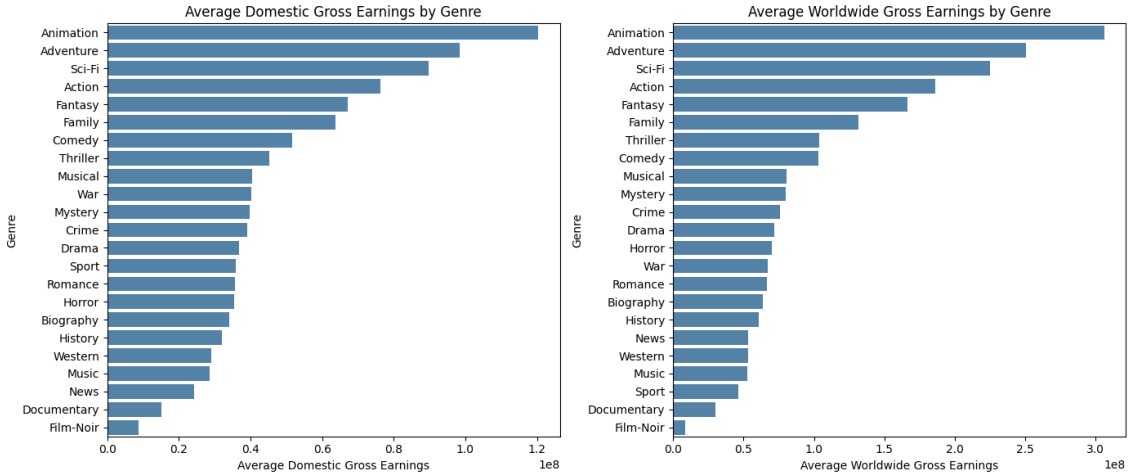
3.6 Wizualizacje i Interpretacja

Stworzyliśmy wykresy i grafiki, aby lepiej zobrazować zależności między cechami i zwizualizować pewne cechy.



Rysunek 3: Występowanie gatunków.

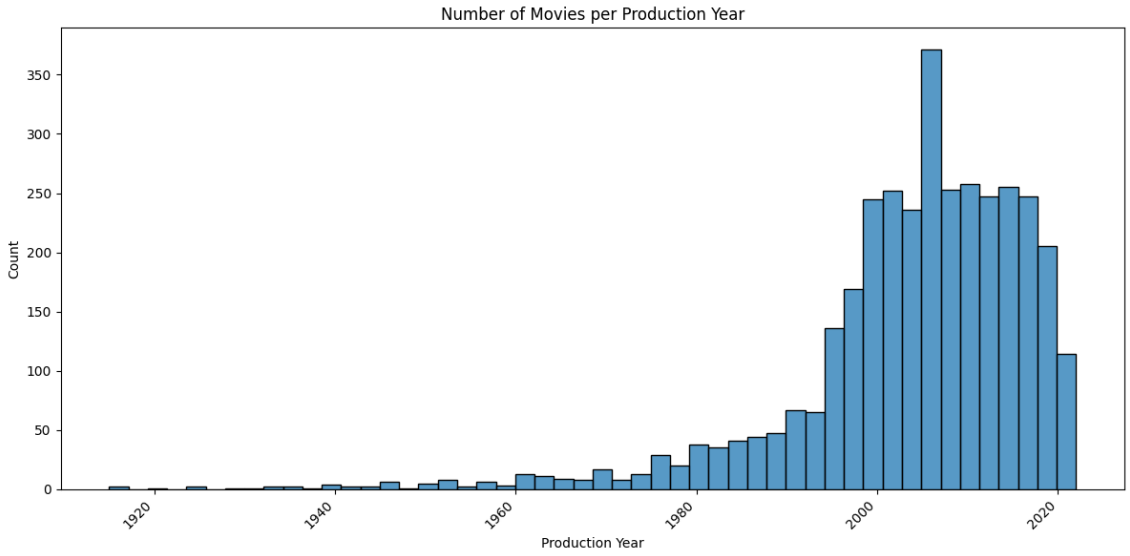
Powyższy wykres przedstawia zliczenie i występowanie gatunków filmowych. Jeden film może mieć więcej niż jeden gatunek.



Rysunek 4: Dochód filmów w zależności od gatunku.

Powyższy wykres przedstawia dochód krajowy i światowy w zależności od gatunku filmowego.
Jak widać najlepiej zarabiające filmy to filmy animowane i przygodowe.

Analiza rozkładu lat produkcji filmów pozwoliła nam zauważyć, że większość filmów w naszej ramce danych pochodzi z XXI wieku, podczas gdy starsze filmy, datowane przed tą dekadą, stanowią znacznie mniejszy odsetek. Poniżej przedstawiono wykres ilustrujący zakres lat z którego pochodzą filmy.



Rysunek 5: Liczba filmów produkowana w danym przedziale czasowym.

Zgłębialiśmy również inne cechy, w tym te związane z reżyserami filmów. Niemniej jednak, nie udało nam się wyłonić z tych danych konkretnych informacji, które mogłyby mieć zasadnicze znaczenie dla dalszych etapów analizy.

4 Preprocessing i Feature Engineering

Kolejnym istotnym etapem w naszym projekcie było wstępne przetwarzanie danych (preprocessing), które miało na celu przygotowanie zbioru danych do późniejszego modelowania. Podczas tego etapu przeprowadziliśmy szereg operacji opisanych poniżej.

4.1 Brakujące dane

4.1.1 Uzupełnienie Brakujących Wartości w Danych Reżyserów

Kolumna director_birthYear, zawierająca lata urodzenia reżyserów, miała brakujące wartości oznaczone jako '-'. Te wartości zostały zamienione na NaN, a kolumna została przekonwertowana na typ liczbowy. Na podstawie lat urodzenia oraz lat produkcji, obliczono wiek reżyserów w momencie produkcji filmu, gdyż uznano, że ta informacja może być bardziej istotna niż sam rok urodzenia reżysera. Wiek został podzielony na trzy przedziały: 'young' (do 40 lat), 'middle-aged' (od 40 do 55 lat) oraz 'old' (powyżej 55 lat). Przedziały wiekowe wybrano w ten sposób, aby po podziale do przedziałów wpadała porównywalna liczba wartości. Następnie zakodowano te przedziały jako wartości całkowite. Brakujące wartości w kolumnie age zostały uzupełnione losowymi wartościami z przedziału 0-2.

4.1.2 Imputacja Brakujących Wartości Finansowych

W kolejnym kroku skupiono się na imputacji brakujących wartości finansowych (budżet produkcji, przychody krajowe oraz przychody globalne). Było to konieczne z uwagi na to, że w ramce występowały filmy o tym samym tytule, pochodzące od różnych reżyserów. Dane budżetowe dla każdego z tych filmów były identyczne, zdecydowaliśmy więc stworzyć dodatkowy model, który pomógłby nam przewidzieć wartości bliższe prawdziwym. Zidentyfikowano zatem filmy, które miały więcej niż jedną wersję (różne daty produkcji i imiona rezyserów dla tego samego tytułu), a następnie podzielono dane na zestawy treningowy i testowy. Model RandomForestRegressor został użyty do przewidywania brakujących wartości na podstawie dostępnych cech takich jak średnia ocena filmu, liczba głosów oraz wskaźnik aprobaty. Po przeprowadzeniu predykcji, wartości te zostały scalone z oryginalnym DataFrame, zastępując brakujące wartości.

4.2 Kodowanie zmiennych kategorycznych

4.2.1 Rozdzielenie Dat

Kolumna production_date została rozdzielona na dwie osobne kolumny: production_year oraz production_month. Rok produkcji był wyodrębniany jako wartość całkowita, a miesiąc został zakodowany cyklicznie przy użyciu funkcji trygonometrycznych sinus i cosinus, co pozwoliło na bardziej efektywne przetwarzanie miesięcy w modelach uczenia maszynowego.

4.2.2 Kodowanie Zawodów Reżysera

Kolumna director_professions, zawierająca listę zawodów reżysera, została użyta do utworzenia nowej kolumny number_of_professions, która określała liczbę zawodów dla każdego reżysera. Następnie kolumna director_professions została usunięta jako zbędna.

4.2.3 Kodowanie Nazwisk Reżyserów

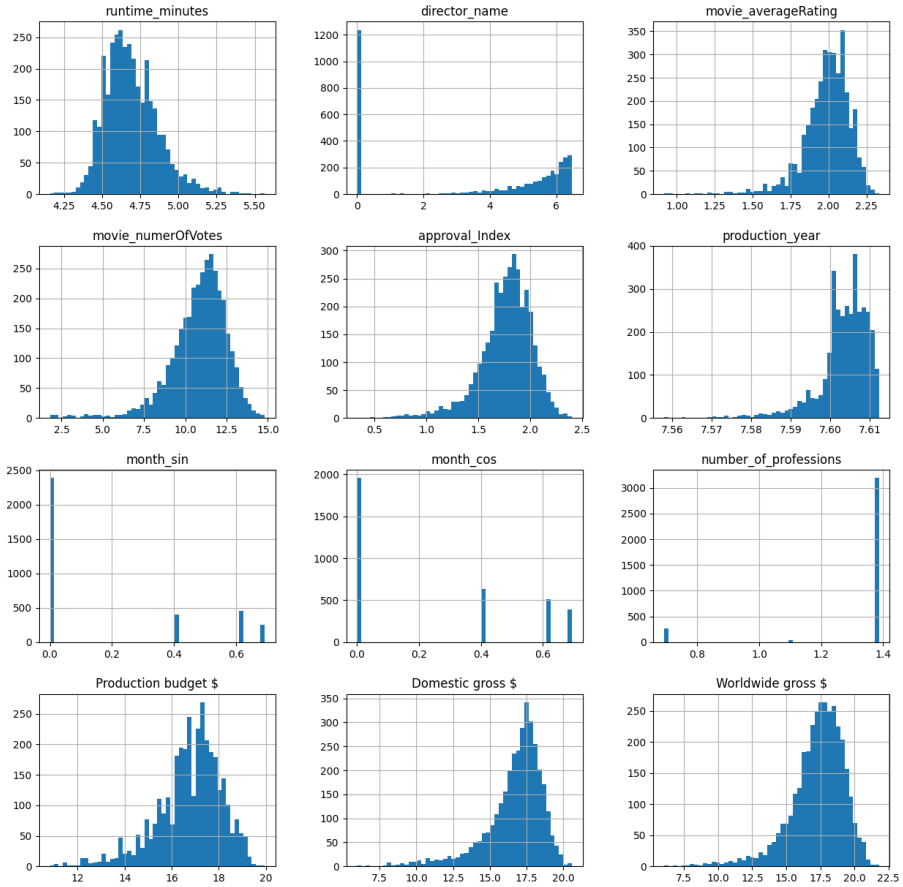
Reżyserzy, którzy pojawiali się tylko raz w zbiorze danych, zostali zastąpieni przez znak '-', a następnie kolumna director_name została zakodowana liczbowo przy użyciu LabelEncoder.

4.2.4 Kodowanie gatunków

Gatunki filmów zostały odpowiednio rozdzielone i zakodowane przy użyciu one hot encoding.

4.3 Transformacja Danych i Wartości Odstające

Z uwagi na mocno skośne rozkłady niektórych cech, przeskalowaliśmy zmienne numeryczne logarytmicznie.



Rysunek 6: Rozkłady zmiennych po zlogarytmizowaniu.

Z powyższego wykresu widać, że po zlogarytmizowaniu, więcej zmiennych numerycznych przyjmuje rozkład normalny. Po zlogarytmowaniu daynch również nie widać definitywnych outlierów do usunięcia, także obsługa ich jest zbędna.

4.4 Standaryzacja Zmiennych

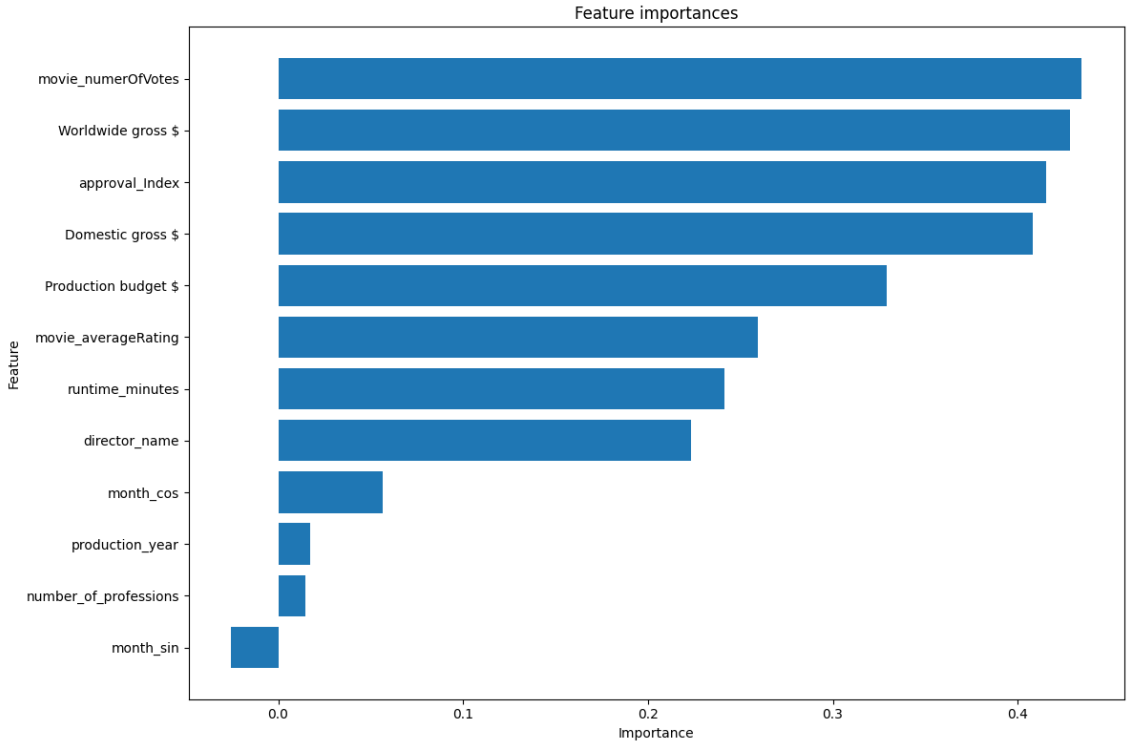
Wszystkie cechy potraktowaliśmy MinMaxScalerem oraz StandardScalerem.
Stworzyliśmy również kilka ramek danych, w których używaliśmy na przykład tylko jednego skalowania.

```
1 from sklearn.preprocessing import StandardScaler
2 scaler = StandardScaler()
3
4 from sklearn.preprocessing import MinMaxScaler
5 scaler = MinMaxScaler()
```

5 Feature Importance

W przypadku klasteryzacji określenie najważniejszych cech jest bardziej skomplikowane niż w przypadku klasyfikacji, bo nic nie przewidujemy, ale postanowiliśmy przeprowadzić feature importance za pomocą kilku metoda, aby poglądowo zrozumieć wpływ cech na przewidywania klastrów

5.1 Feature importance za pomocą PCA (analizy głównych składowych).



Rysunek 7: Feature importance za pomocą PCA

Ważność cech w PCA jest określana na podstawie obciążeń (loadings) oryginalnych cech na poszczególne składowe główne. Obciążenie to wartość, która wskazuje, jak duży wpływ ma dana cecha na daną składową główną. Im większe obciążenie, tym większy wpływ ma cecha na składową główną.
Movie_numerOfVotes ma największe obciążenie, co oznacza, że ocena filmu przez użytkowników online jest najważniejszą cechą w wyjaśnianiu zmienności w danych. Może to sugerować, że filmy z większą oceną mają silniejszy wpływ na zmienność w zbiorze danych.

5.2 Feature importance za pomocą analizy skupień na podstawie cech (klasteryzujemy cechy).

Cechy są transponowane i traktowane jako punkty danych, które są następnie grupowane w klastry. Ważność cech w klasteryzacji KMeans nie jest bezpośrednio wyznaczana, jak w przypadku PCA, ale możemy analizować, które cechy zostały przypisane do tych samych klastrów, aby zrozumieć, jakie cechy mają podobne wzorce w danych.

```
                    Feature  Cluster
0          runtime_minutes        0
1            director_name        2
2      movie_averageRating        0
3       movie_numerOfVotes        0
4           approval_Index        0
5          production_year        1
6                month_sin        1
7                month_cos        0
8    number_of_professions        0
9      Production budget $        1
10        Domestic gross $        1
11       Worldwide gross $        1
```

Analiza wyników:

1. Klaster 0 - Cechy w klastrze 0 wydają się być związane z charakterystykami filmów i ich popularnością. Podejrzewamy, że cechy znajdujące się w tym klastrze będą miały największy wpływ na wyniki klasteryzacji.
2. Klaster 1 - Cechy w klastrze 1 są związane z finansowymi i czasowymi aspektami produkcji filmu. production_year, month_sin, Production budget $, Domestic gross $ i Worldwide gross $ sugerują, że te cechy są skorelowane z finansowymi wynikami filmów i ich czasem produkcji.
3. Klaster 2 - Director_name jest jedyną cechą w klastrze 2, co może sugerować, że wpływ nazwiska reżysera na dane filmowe jest inny od pozostałych cech.

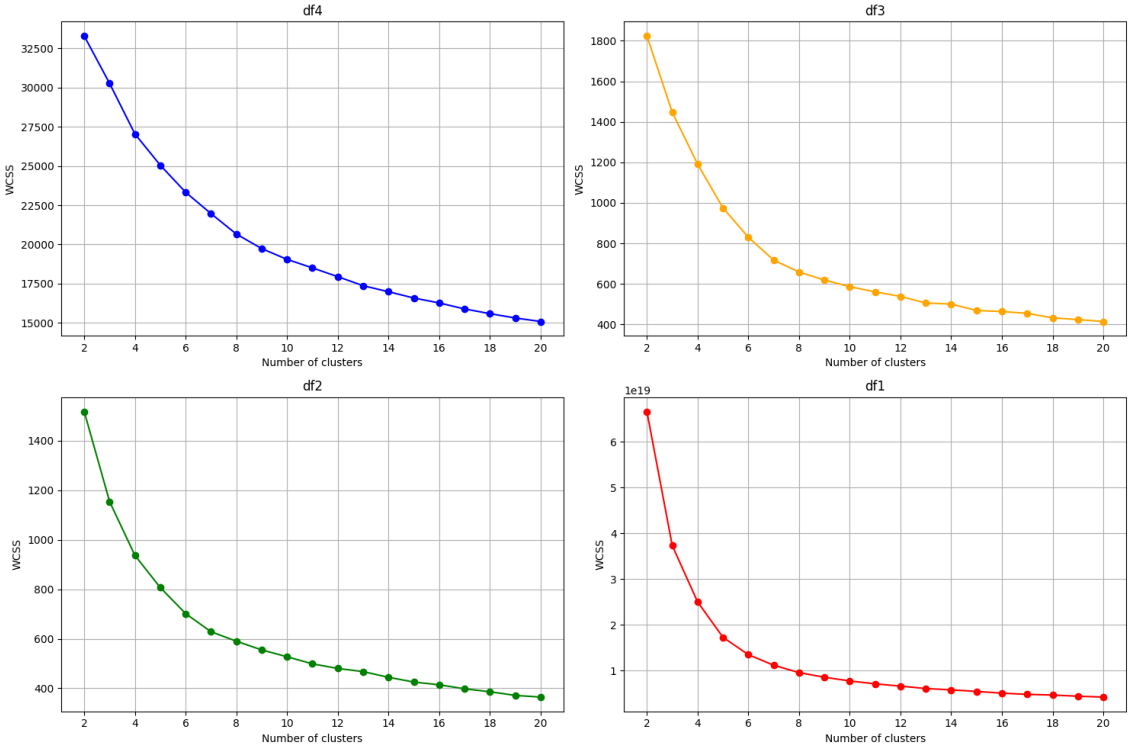
6 Wstępne modelowanie

W naszym projekcie testujemy cztery wariatny danych:

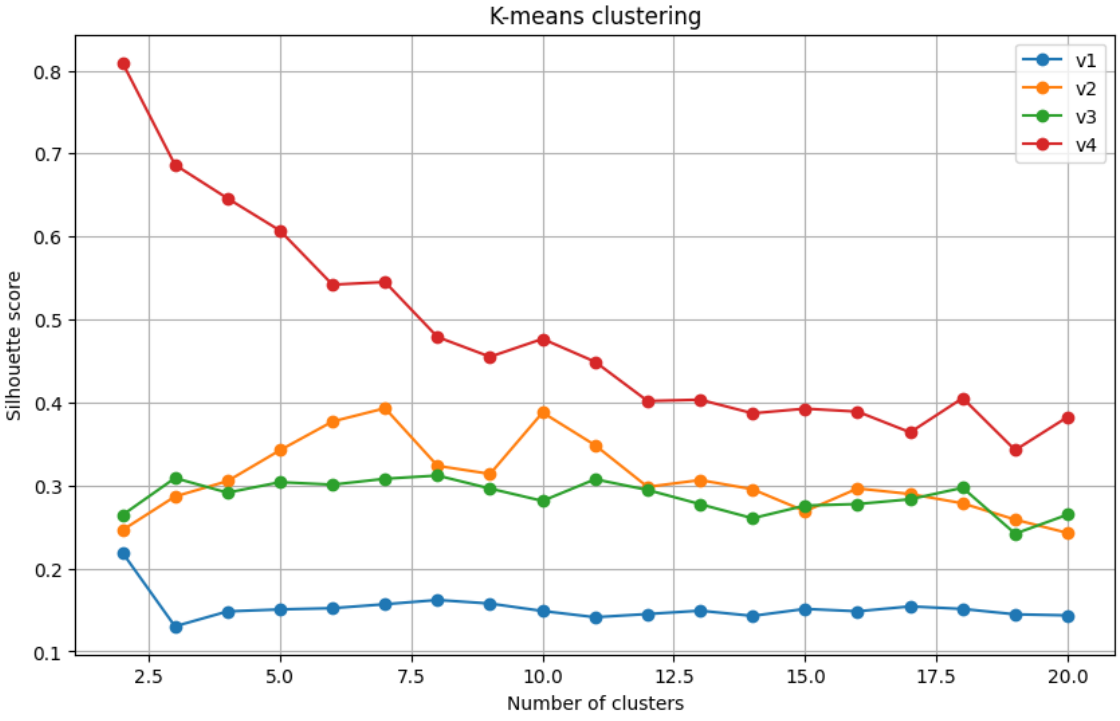
- df1 - przetworzone dane bez standaryzacji
- df2 - dane znormalizowane
- df3 - dane zlogarytmowane i znormalizowane
- df4 - dane zlogarytmowane i przeskalowane

6.1 Metoda łokci

W celu określenia jaka liczba klastrów będzie optymalna w naszym projekcie, wykorzystaliśmy metodę łokci.



Rysunek 8: Metoda łokci za pomocą wcss z KMeans

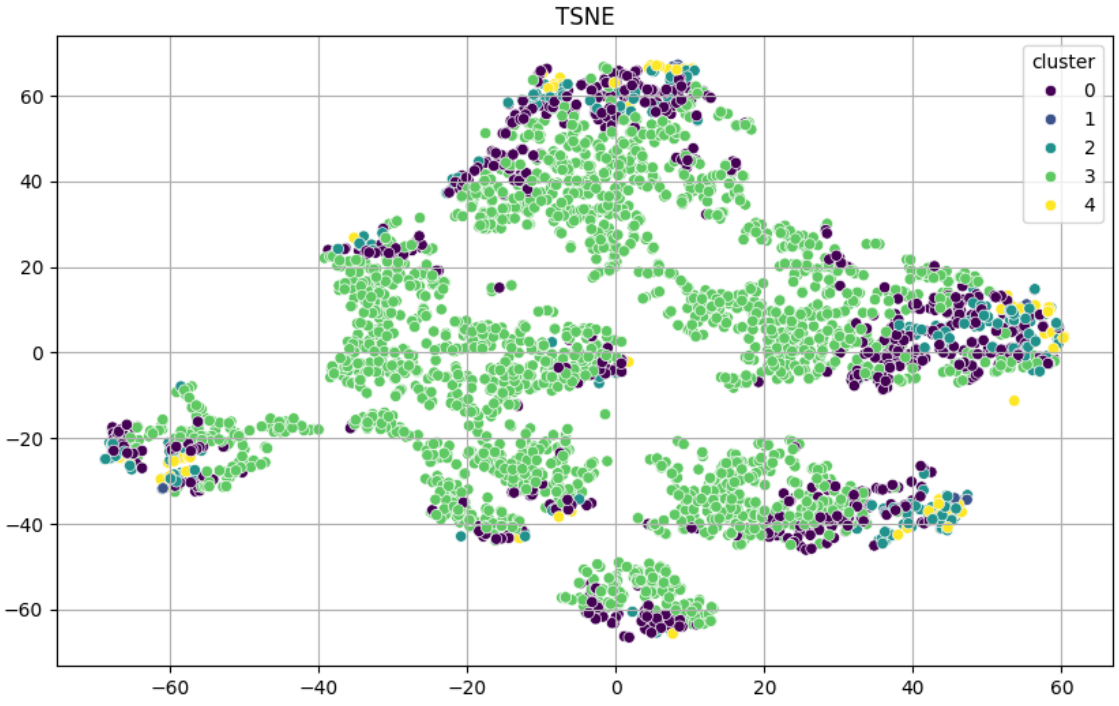


Rysunek 9: Metoda łokci za pomocą silhouette_score

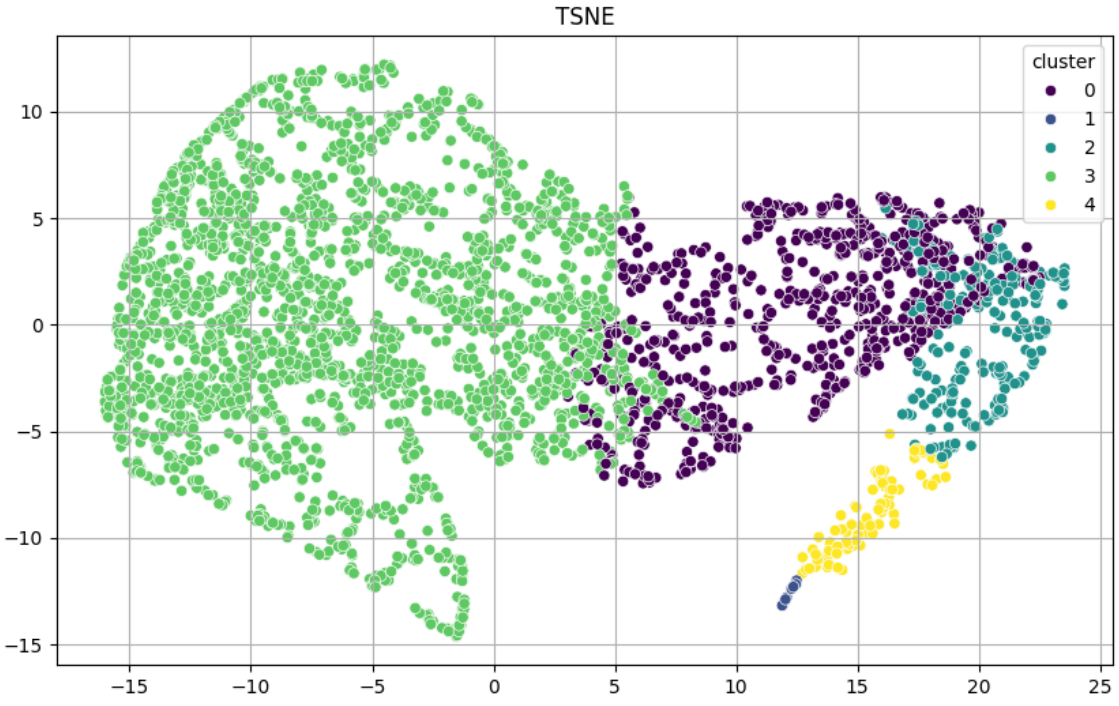
Na podstawie obu wykresów stwierdziliśmy, iż optymalną liczbą we wstępnym modelowaniu, będzie liczba pięciu klastrów.

6.2 Kmeans i tSNE

Do wstępnego modelowania użyliśmy metody Kmeans dla dwóch zestawów danych df1 i df4, oraz tSNE poznanego na laboratoriach, żeby zwizualizować i zobaczyć jak formują się wstępne klastry.



Rysunek 10: Wizualiazcja klastrów za pomocą tSNE dla zestawu danych df1



Rysunek 11: Wizualiazcja klastrów za pomocą tSNE dla zestawu danych df4

7 Zaawansowane modelowanie

7.1 Wprowadzenie kolejnych algorytmów klasteryzacji

W ostatnim etapie naszego projektu wprowadziliśmy więcej algorytmów klasteryzacji i porównaliśmy ze sobą wyniki poszczególnych algorytmów. Na tym etapie, bazując ponownie na wykresach łokciowych, postanowiliśmy przetestować modele dla 7 klastrów.

Algorytmy użyte do klasteryzacji, wymagające specyfikacji liczby klastrów (wyniki dla siedmiu klastrów):

- kmeans
- agglomerative
- gaussian_mixture
- birch
- mini_batch_kmeans
- spectral_clustering

Algorytmy niewymagające specyfikacji liczby klastrów:

- affinity_propagation
- optics
- dbscan
- mean_shift

Otrzymane wyniki dla ramki danych df1:

Model	silhouette_score	calinski_harabasz_score	davies_bouldin_score
kmeans	0.559681	7411.287517	0.682597
agglomerative	0.421027	6941.199597	0.752019
dbscan	0.758833	1940.795344	1.189924
birch	0.421027	6941.199597	0.752019
mini_batch_kmeans	0.417965	5102.728189	0.841951
mean_shift	0.688682	888.209209	0.486664
spectral_clustering	-0.0959	0.803325	75.449422
affinity_propagation	0.158397	574.736087	0.402879
optics	-0.414591	9.052247	1.801458
gaussian_mixture	0.004944	216.47386	3.419647

Tabela 1: Wyniki różnych algorytmów klasteryzacji dla ramki danych df1.

Otrzymane wyniki dla ramki danych df2:

Model	silhouette_score	calinski_harabasz_score	davies_bouldin_score
kmeans	0.308087	1308.782422	1.150094
agglomerative	0.263196	1097.148241	1.243363
dbscan	0.22593	408.997154	1.671115
birch	0.279069	361.004255	1.019096
mini_batch_kmeans	0.29994	1157.835852	1.178877
mean_shift	0.324197	570.878707	1.288852
spectral_clustering	0.322403	1104.531446	1.187204
affinity_propagation	0.18475	393.420575	1.360635
optics	-0.516143	11.429369	1.277648
gaussian_mixture	0.098104	275.622296	3.037599

Tabela 2: Wyniki różnych algorytmów klasteryzacji dla ramki danych df2.

Otrzymane wyniki dla ramki danych df3:

Model	silhouette_score	calinski_harabasz_score	davies_bouldin_score
kmeans	0.393054	1438.382724	0.984198
agglomerative	0.371785	1321.856699	1.045844
dbscan	0.09418	316.198504	1.521899
birch	0.316599	741.336045	1.252742
mini_batch_kmeans	0.393054	1438.382724	0.984198
mean_shift	0.312671	559.444098	1.296444
spectral_clustering	0.376709	1229.405885	1.13261
affinity_propagation	0.16094	391.18903	1.508474
optics	-0.564188	10.779907	1.242193
gaussian_mixture	0.251506	808.656867	1.631183

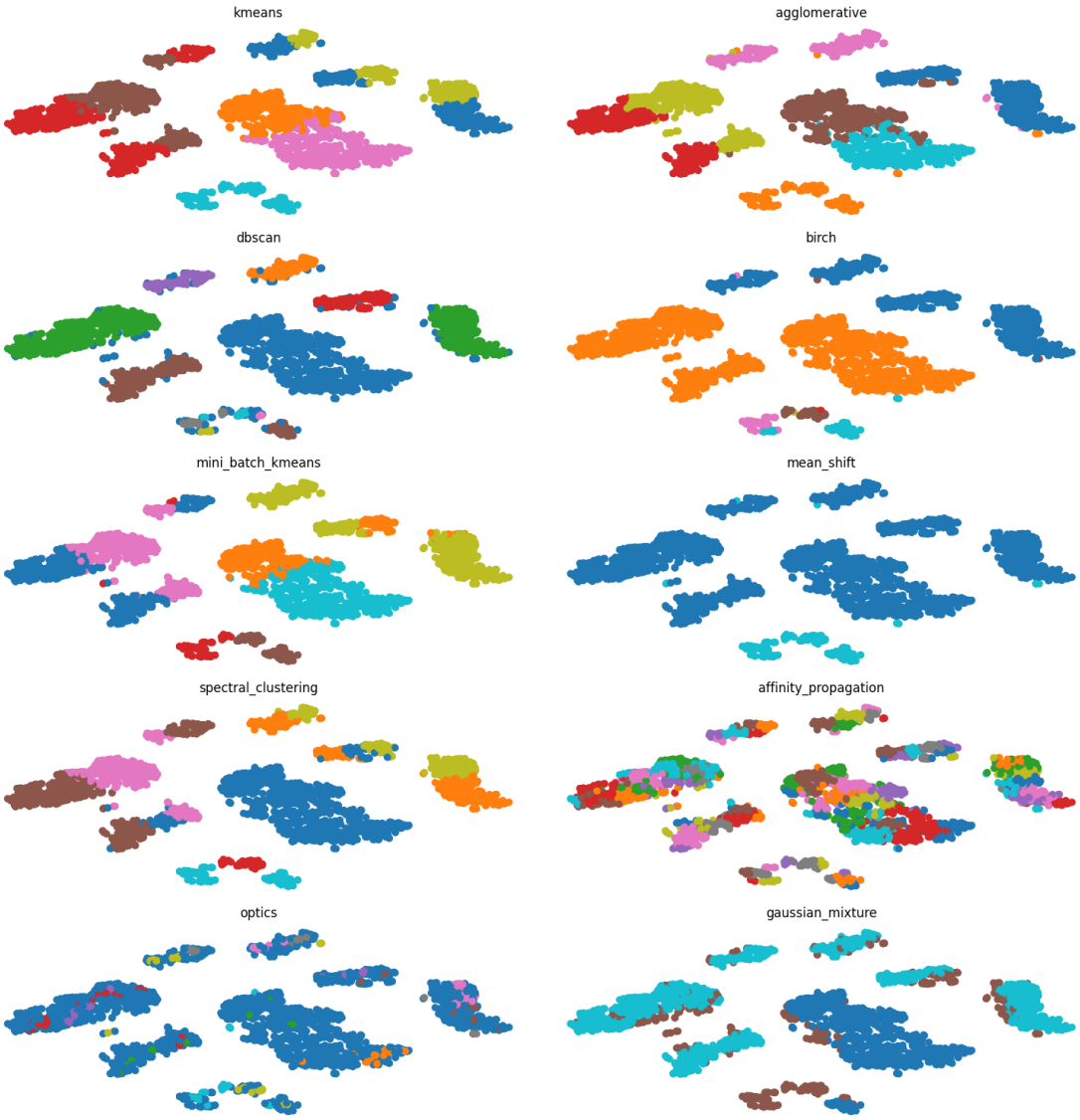
Tabela 3: Wyniki różnych algorytmów klasteryzacjidla ramki danych df3.

Otrzymane wyniki dla ramki danych df4:

Model	silhouette_score	calinski_harabasz_score	davies_bouldin_score
kmeans	0.156005	532.28245	1.672573
agglomerative	0.119197	439.370465	1.741751
dbscan	0.227346	218.00668	1.752422
birch	0.140588	422.930777	1.70906
mini_batch_kmeans	0.141176	519.317135	1.743314
mean_shift	0.371101	54.2612	1.199665
spectral_clustering	0.292428	31.672814	0.969533
affinity_propagation	0.127554	101.217509	1.493348
optics	-0.462526	6.227234	1.461413
gaussian_mixture	0.265906	415.495209	1.642593

Tabela 4: Wyniki różnych algorytmów klasteryzacji dla ramki danych df4.

Poniżej znajdują się przykładowe wizualizacje klasteryzacji przy użyciu tSNE. Wizualizacje dla pozostałych wariantów są przedstawione w pliku KM3_final.html.



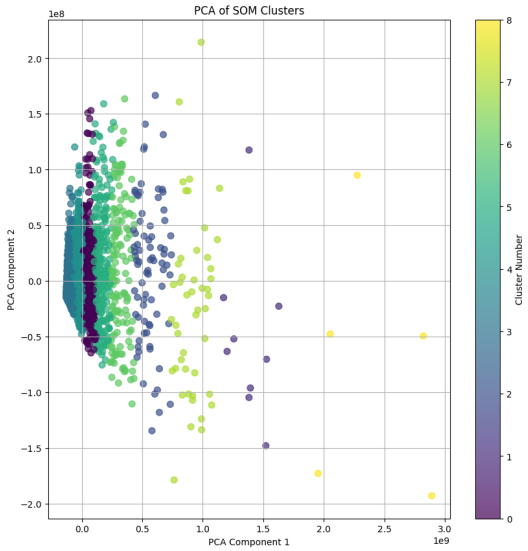
Rysunek 12: Wizualiazcja klastrów przy użyciu różnych metod za pomocą tSNE dla zestawu danych df3

7.2 Sieci neuronowe

Aby wprowadzić nasz projekt na wyższy poziom zaawansowania i aby podnieść skuteczność, użyliśmy dwóch modeli sieci neuronowych. Szczegóły implementacji można zobaczyć w pliku KM3_final.html.

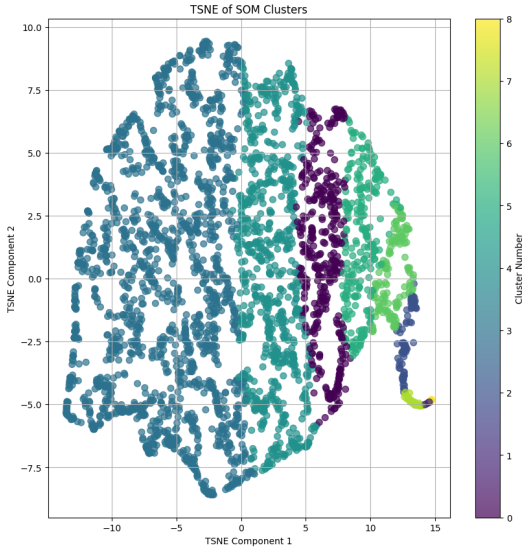
7.2.1 Sieci neuronowe przy użyciu pakietu MiniSom

Wizualizacja wyników przy użyciu PCA:



Rysunek 13: Wizualiazcja klastrów przy użyciu sieci neurnowej Som za pomocą PCA dla zestawu danych df1

Wizualizacja przy użyciu tSNE:



Rysunek 14: Wizualiazcja klastrów przy użyciu sieci neurnowej Som za pomocą tSNE dla zestawu danych df1

Wyniki dla sieci neuronowej z pakietu MiniSom:

Metric	Score
Silhouette Score	0.4649
Calinski-Harabasz Score	7605.4071
Davies-Bouldin Score	0.8320

Tabela 5: Wyniki metryk dla klasteryzacji dla Self-Organizing Map (SOM).

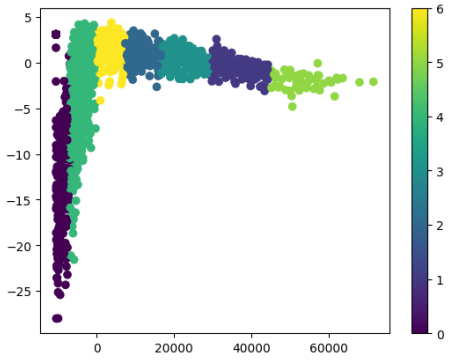
Otrzymane wyniki są znacznie lepsze od wyników otrzymanych przy użyciu wcześniejszych metod. Dodatkowo, po dostrojeniu modelu za pomocą najlepszych hiperparametrów wynik silhouette score przy użyciu tej sieci neuronowej polepszył się.

```
Best Hyperparameters:
{'width': 2, 'height': 2, 'sigma': 0.1, 'learning_rate': 1.0}
Best Silhouette Score: 0.7081854042726141
```

Dla ramki danych df1 jest to zdecydowanie najlepszy wynik.

7.2.2 Sieci neuronowe przy użyciu Keras z pakietu TensofFlow

Wizualizacja przy użyciu PCA:



Rysunek 15: Wizualiazcja klastrów przy użyciu sieci neurnowej Keras za pomocą PCA dla zestawu danych df4

Wyniki dla sieci neuronowej z pakietu TensorFlow:

Metric	Score
Silhouette Score	0.7313
Calinski-Harabasz Score	27404.0075
Davies-Bouldin Index	0.4660

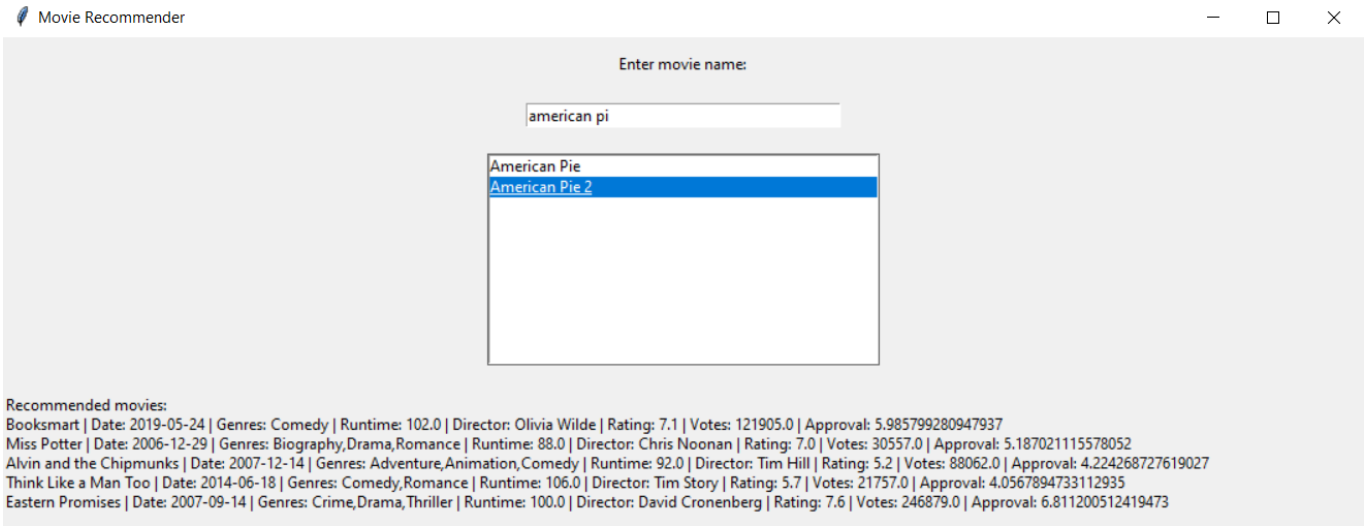
Tabela 6: Wyniki metryk klasteryzacji.

8 Zastosowania biznesowe

Jak już wspominaliśmy nasz projekt miał dwa podstawowe założenia biznesowe: - rekomendacje filmów podobnych do obejrzenia dla użytownika - rekomendacje jakie filmy najlepiej się sprzedają dla producentów filmowych.

8.1 rekomendacje

W rekomendacjach dla usera, użyliśmy ramki danych df1 z użyciem algorytmu MiniSom. W tym celu napisana została funkcja, która dla danego filmu znajduje daną ilość filmów znajdujących najbliżej wyszukiwanego w danym klastrze. Dzięki temu z dużą skutecznościa da się znaleźć film, który może zainteresować użytkownika. Aby zwizualizować działanie tej funkcji utworzyliśmy podstawowe GUI, które pozwala użytkownikowi na wpisanie filmu z bazy danch i pokazane zostają mu rekomendacje. Przykładowe wyszukiwanie znajduje się poniżej:



Rysunek 16: Przykładowe wyszukiwanie w GUI

Filmy są przewidywane z nawet dużą skutecznością, gdyż zazwyczaj 4/5 rekomendacji są naprawdę sensowne.

8.2 dla producentów filmowych

Aby zadowolić również producentów filmowych utworzono funkcję, która podzieliła filmy na klastry ze względu na maksymalizowanie cech, zarówno dotyczących ocen jak i dochodów. Do tego użyliśmy ramki danych df4 z modelem sieci neuronowych keras z Tensorflow, gdyż osiągały one również dobre wyniki. Następnie funkcją analizującą filmy w najlepszym klastrze wydobyliśmy informację jakie filmy się najlepiej sprzedają. Skalsyfikowaliśmy je po: gatunku, długości trwania i miesiącu wypuszczenia. Otrzymaliśmy wynik:

```
Most common genre: Drama
Average runtime: 115.15412186379929
Most common month: 12
```

Z tego wynika, iż najlepiej tworzyć filmy z gatunku Drama, o średnim czasie trwania seansu około 115 minut oraz najlepszym miesiącem na wydanie takiego filmu jest grudzień.

9 Podsumowanie

W naszym projekcie dokonaliśmy wszechstronnej analizy danych filmowych, uwzględniając różnorodne cechy i dostępne dane. W procesie przetwarzania danych zastosowaliśmy zarówno konwencjonalne, jak i niekonwencjonalne metody, co pozwoliło nam na uzyskanie interesujących i skutecznych rezultatów. Przeanalizowaliśmy szereg modeli na różnych zestawach danych, aby ostatecznie wybrać najlepsze kombinacje do naszych celów biznesowych.

Wykorzystanie ramki danych df1 z sieciami neuronowymi z pakietu minisom pozwoliło nam stworzyć aplikację rekomendującą filmy użytkownikowi w sposób innowacyjny. Natomiast wykorzystanie df1 z sieciami neuronowymi keras z Tensorflow + kmeans pozwoliło nam zidentyfikować kluczowe cechy, które decydują o sukcesie filmu. Nasz projekt ma zastosowanie zarówno dla kinomanów, którzy poszukują nowych inspiracji, jak i dla producentów, którzy chcą lepiej zrozumieć preferencje widzów, co czyni naszą pracę niezwykle wszechstronną i wartościową.

Oczywiście istnieje wiele możliwości rozwoju projektu. Możemy kontynuować eksplorację różnych algorytmów i poszukiwać jeszcze lepszych modeli. Ponadto, możemy rozbudować aplikację kliencką, dodając funkcje takie jak wyszukiwanie rekomendacji dla filmów spoza bazy danych poprzez zgłaszanie przez użytkowników odpowiednich danych, lub automatyczne uzupełnianie brakujących informacji za pomocą modeli predykcyjnych, na przykład w przypadku duplikatów. Niestety, ograniczony czas uniemożliwił nam dalszy rozwój projektu, jednak dostarczyliśmy dobrze przeszkolone modele wraz z przykładami ich zastosowania.