

POLITECHNIKA WROCŁAWSKA
WYDZIAŁ INFORMATYKI I TELEKOMUNIKACJI



Preykcja popularności piosenek i system rekommendacji

Sprawozdanie z projektu

AUTOR

Aleksandra Walczybok

nr albumu: **272454**

kierunek: **Inżynieria Systemów**

14 czerwca 2024

1 Wprowadzenie – sformułowanie problemu, cel projektu.

W dzisiejszych czasach sektor muzyczny głównie rozwija się poprzez aplikacje streamingowe - takie jak Spotify. Wśród nowego pokolenia jest to najpopularniejszy środek do odsłuchu piosenek czy podcastów. Nieliczne jednostki kupują płyty lub winyle. Myśl ta skłania, więc do konkluzji, że dla artysty najistotniejszy będzie wynik popularności jego piosenki w tejże aplikacji. Wynik ten pozwoli wzbić się na najpopularniejsze playlisy tworzone przez aplikacje, dodatkowo wpłynie on na liczbę obserwujących a to wszyskto sprowadzi się do większej popularności samego artysty i wzrostu jego dochodów. Pytanie więc jak stworzyć utwór, który osiągnie sukces? Inaczej problem ten można zdefiniować biorąc pod uwagę stronę wytwórną - w jaką piosenkę warto zainwestować? Na te pytania można odpowiedzieć wytwarzając odpowiednie narzędzie do predykcji popularności piosenki, na podstawie jej cech, takich jak: energia, tempo, czas trwania, akustyczność, taneczność itp. Cechy te można zmierzyć w studiu, według zadanych instrukcji. Reagując na ten aktualny problem w sektorze muzycznym, pierwszym celem tego projektu jest wytworzenie specjalnej strony/aplikacji do predykcji popularności wytwarzanych, bądź dodawanych piosenek na Spotify.

Kolejną istotną kwestią w naszej codzienności stały się systemy rekomendacyjne. Są one nieodzowną częścią wszelkich aplikacji. Pozwalają one na stale poszerzanie horyzontów klientów. W ten sposób użytkownicy mogą poznawać nowe zasoby aplikacji i dzięki temu nie nudzą się i nie rezygnują z użytkowania. Problemem w tym podejściu staje się sposób w jaki należy rekomendować np. piosenki, aby były one zgodne z gustem użytkownika. W tym projekcie stworzony został taki system rekomendacji piosenek, który pod uwagę bierze ulubioną playlistę użytkownika.

Ostatnim celem zadania było sprawdzenie czy można przewidzieć, na podstawie cech muzycznych, jakiego gatunku jest dany utwór.

Spodziewanymi wynikami projektu są: aplikacja do predykcji popularności piosenki oraz aplikacja do systemu rekomendacyjnego. Podczas projektu porównane zostaną różne metody osiągnięcia tychże wyników. Szczególną uwagę należy skupić na wybraniu odpowiednich modeli do predykcji, klasyfikacji oraz sposobu tworzenia rekomendacji.

2 Opis badań.

2.1 Narzędzia.

Projekt został zrealizowany w języku Python. Kod dotyczący pobierania danych oraz tworzenia aplikacji zapisany został w środowisku PyCharm PRO. Analiza danych oraz modele predykcyjne, klasyfikacji, rekomendacji zostały zaimplementowane przy użyciu Jupyter Notebook, który pozwala na łatwiejszą obsługę kodów związanych z przetwarzaniem danych, rysowaniem wykresów i tworzeniem modeli.

- Użyte biblioteki:
 - spotipy = 2.23.0
 - pandas = 2.2.2
 - scikit-learn
 - dotenv
 - python-dotenv = 1.0.1
 - matplotlib = 3.8.4

- datetime
- seaborn
- scikeras
- keras
- pickle
- xgboost
- numpy = 1.26.4

2.2 Proces projektu.

2.2.1 Pobieranie danych.

Pierwszym wyzwaniem projektu był proces zbierania danych, w celu utworzenia bazy danych, na których modele predykcyjne oraz klasifikacji będą mogły się uczyć. Dodatkowo baza ta będzie użytkowana w systemie rekommendacji. Do pobierania piosenek użyto API aplikacji Spotify. Została również zaimplementowana odpowiednia biblioteka - Spotipy, pomagająca przy wydobywaniu informacji z tego API. Pierwszym procesem był wybór odpowiednich playlist z Spotify, które należało pobierać do bazy. Playlisty powinny zawierać różnorodne piosenki, pod względem gatunkowym, popularności czy też roku wydania. Ostatecznie wybrano 25 playlist, których linki można znaleźć w pliku: playlist.txt.

Pobierano następujące informacje o piosenkach:

- nazwa piosenki
- nazwa albumu
- wykonawca
- popularność piosenki
- popularność artysty
- gatunek piosenki
- długość piosenki
- data wydania
- cechy muzyczne
 - **danceability** (taneczność): Mierzy, jak odpowiednia jest piosenka do tańczenia. Opiera się na kombinacji elementów muzycznych, w tym tempa, stabilności rytmu, siły rytmu i ogólnej regularności. Wartości wynoszą od 0.0 do 1.0, gdzie 1.0 oznacza, że utwór jest najbardziej taneczny.
 - **energy** (energia): Mierzy intensywność i aktywność utworu. Typowe cechy energetyczne obejmują szybkie, głośne i hałaśliwe elementy. Wartości wynoszą od 0.0 do 1.0, gdzie 1.0 oznacza największą energię.

- **loudness** (głośność): Średni poziom głośności utworu w decybelach (dB). Głośność jest wartością uśrednioną na podstawie całego utworu. Typowe wartości wynoszą od -60 dB (bardzo cicho) do 0 dB (bardzo głośno).
- **speechiness** (mówność): Mierzy obecność słów w utworze. Wartości powyżej 0.66 wskazują, że utwór prawdopodobnie składa się głównie z mowy (np. audiobooki, podcasty). Wartości pomiędzy 0.33 a 0.66 mogą wskazywać utwory zawierające zarówno muzykę, jak i mowę (np. rap). Wartości poniżej 0.33 zazwyczaj oznaczają utwory muzyczne z małą ilością mowy.
- **acousticness** (akustyczność): Mierzy, jak akustyczny jest utwór. Wartość 1.0 oznacza wysokie prawdopodobieństwo, że utwór jest akustyczny.
- **instrumentalness** (instrumentalność): Mierzy prawdopodobieństwo, że utwór nie zawiera wokalu. Wartości powyżej 0.5 wskazują, że utwór jest prawdopodobnie instrumentalny. Im bliżej wartości 1.0, tym większe prawdopodobieństwo, że utwór jest instrumentalny.
- **liveness** (życiowość): Mierzy, jak prawdopodobne jest, że utwór był nagrany na żywo. Wartości powyżej 0.8 wskazują na dużą obecność publiczności w nagraniu.
- **valence** (pozytywność): Mierzy muzyczny pozytywizm utworu. Wartości wynoszą od 0.0 do 1.0, gdzie wartość 1.0 oznacza bardzo pozytywny utwór (szczęśliwy, wesoły), a 0.0 oznacza bardzo negatywny utwór (smutny, przygnębiający).
- **tempo** (tempo): Średnie tempo utworu w uderzeniach na minutę (BPM). Tempo to szybkość lub tempo utworu.

Każdą playlistę pobierano i zapisywano do pliku csv oddzielnie. Po zakończeniu pobierania danych, które zostało rozłożone na kilka dni, ze względu na ograniczenia Spotify, wszystkie pliki zostały połączone w jedną wspólną bazę. Wcześniej jednak należało rozdzielić na poszczególne kolumny wartości cech muzycznych, gdyż miały one formę słownika. Po udanym transformowaniu danych zapisano wszystkie rekordy do pliku *finaldata.csv*

Problemy:

- Po wybraniu playlist pojawił się pierwszy problem, gdyż okazało się, że Spotify ma spore ograniczenia na ilość pobieranych piosenek z jednej playlisty. Mimo, że każda playlista zawierała około 1000-2000 utworów, aplikacja pozwala pobierać zaledwie 100 z każdej. Ten problem mocno ograniczył wielkość bazy piosenek, gdyż wybierając więcej playlist znów aplikacja blokowała dostęp, ze względu na zbyt dużą liczbę zapytań.

2.2.2 Transformacja danych - tzw.preprocessing.

Celem tego etapu jest zapewnienie dobrej jakości danych. Początkowo usunięto zbędne kolumny odpowiadające za indeksowanie. Następnie ze względu na to, iż pobrano dane z wielu playlist, należało usunąć ewentualne duplikaty piosenek. Zmieniono również datę wydania na rok wydania, ponieważ jedynie on będzie użytyczny do dalszej analizy. Ważnym elementem jest także radzenie sobie z wartościami brakującymi. W tym przypadku takie wartości pojawiały się jedynie w kolumnie gatunku muzycznego. Takie "nulle" zamienione na wartość "Inne", gdyż gatunek jest niezanany.

2.2.3 EDA - Exploratory Data Analysis.

Kolejny etap dotyczył analizy danych i bliższego poznania bazy piosenek. Sprawdzono tutaj takie elementy:

1. Korelacje pomiędzy zmiennymi numerycznymi a wartością popularności piosenki.
2. Korelacje między wszystkimi zmiennymi numerycznymi.
3. Ilość piosenek względem roku wydania.
4. Zmiana cech piosenek (*acousticness, danceability, energy, instrumentalness, liveness, valence*) względem lat.
5. 10 najpopularniejszych wykonawców i utworów w zbiorze danych.
6. Histogramy zmiennych i ich charakterystyka statystyczna.
7. Rozkład popularności piosenek i wykonawców w zbiorze.

Przeprowadzono również analizy w zależności od gatunku muzycznego piosenek. Wpierw, należało jednak ujednolicić ich nazwy, gdyż spotify dysponuje wieloma gatunkami (łącznie w bazie było około 230 gatunków), które nie są kojarzone dla osób niezwiązanych z przemysłem muzycznym. Dlatego mając np. gatunek pod nazwą "*barbadian pop*" zamieniono go na '*pop*'. A więc jeśli gatunek zawierał część jednego z popularnych gatunków, zmieniano jego nazwę, jeśli nie - gatunek był zmieniany na wartość "*Inne*". Otrzymano w ten sposób następujące gatunki muzyczne: **pop, hip hop, rock, blues, indie, folk, metal, jazz, soul, dance, rap, classical, other**.

Aby wiedzieć, czy gatunek wpływa na popularność piosenki, przeprowadzono test statystyczny - ANOVA. Test sprawdzał czy średnia popularność piosenek względem gatunków jest różna.

2.2.4 Metody predykcji popularności.

Używana baza danych do tej części, zawiera już poprawki zadane w części preprocessingu, usunięto również kolumny: wykonawca, album. Dodatkowo użyto tutaj metody **One Hot Encoding**, gdyż kolumna z gatunkami muzycznymi jest typu object, a model predykcji może na wejściu otrzymać jedynie wartości numeryczne. Stosując One Hot Encoding każda wartość z kolumny gatunku zamieniała się w nową kolumnę, gdzie wartości są zero- jedynkowe.

Colour
Green
Red
Blue

→

	Green	Red	Blue
0	1	1	
1	1	1	
1	0	1	
0	0	0	
0	1	0	

Rysunek 1: Przykładowe użycie One Hot Encodingu.

Następnym etapem było sprawdzenie korelacji i usunięcie kolumn, które są ze sobą skorelowane powyżej wartości 0.6. Głównym celem tutaj jest redukcja wymiarów.

Dane zostały podzielone na zestawy treningowe i testowe przy użyciu metody '*train test split*' z biblioteki sklearn, w stosunku 4:1. Następnie dokonano standaryzacji według wzoru:

$$z = \frac{x - \mu}{\sigma}$$

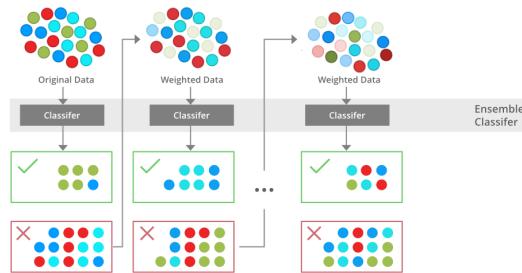
Użyte modele:

1. Regresja liniowa
2. Regresja wielomianowa st.2
3. Regresja wielomianowa st.3
4. Random Forest

- Dobór parametrów takich jak liczba drzew czy głębokość drzew zostało wybrany przy użyciu **Grid Searcha**. Metoda ta sprawdza funkcję kosztu (funkcję najmniejszych kwadratów) w zależności od podanych parametrów i wybiera te, których kombinacja dała najmniejszą funkcję kosztu.

5. XGBOOST

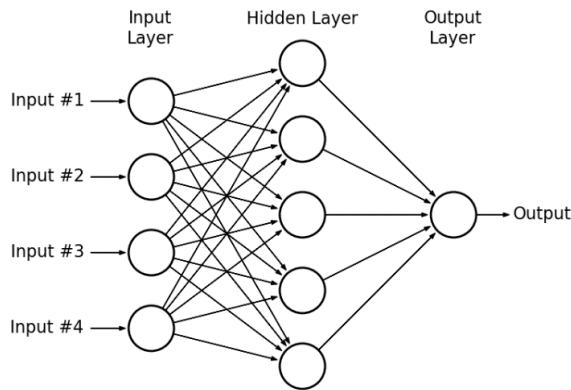
- XGBOOST działa poprzez iteracyjne dodawanie drzew decyzyjnych, z których każde koryguje błędy popełnione przez poprzednie drzewa, minimalizując funkcję straty. Każde drzewo jest dopasowywane do gradientów błędów, a ostateczna predykcja jest sumą wszystkich drzew, z uwzględnieniem mechanizmów regularyzacji, aby zapobiegać przetuzieniu.
- Dla tego modelu również przeprowadzono metodę Grid Search w celu uzyskania najlepszych parametrów drzewa.
- Implementacja tego modelu odbywa się za pomocą biblioteki xgboost.



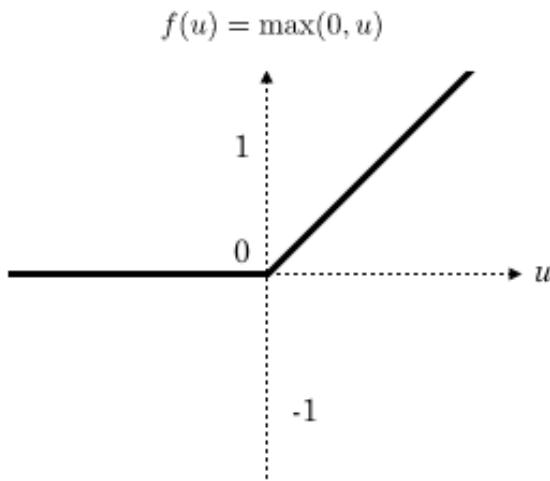
Rysunek 2: Działanie XGBOOST.

6. Multi Layer Perceptron

- MLP to mała sieć neuronowa, która na wejściu przyjmuje 22 cechy, następnie mamy 2 warstwy ukryte (jedna ma 64 węzły, druga 32), na wyjściu uzyskujemy predykowaną wartość popularności piosenki. Jako funkcję straty użyto funkcję najmniejszych kwadratów. Ilość epok uczenia to 100, a batch size 10. Batch size odnosi się do liczby przykładów danych, które są przetwarzane jednocześnie podczas jednej iteracji w procesie trenowania modelu uczenia maszynowego. Funkcje aktywacji na poszczególnych warstwach (oprócz ostatniej) to funkcje relu.
- Ten model został wykonany przy użyciu biblioteki keras.



Rysunek 3: Przykład sieci neuronowej.



Rysunek 4: Funkcja aktywacji ReLU

Walidacja Aby ocenić czy modele działają tak jak powinny i czy, któryś z nich jest zupełnie bezużyteczny przeprowadzono walidację. Do wykonania jej użyto krotną walidację krzyżową, a jako funkcję kosztu wybrano najmniejsze kwadraty. Podział danych treningowych, czyli liczba "foldów" to standardowe 5.



Rysunek 5: Proces walidacji.

Do oceny modeli użyto 3 metryk:

1. Mean Squared Error (MSE): MSE jest to średnia kwadratów różnic między prawdziwymi wartościami (y_i) a przewidywanymi wartościami (\hat{y}_i) przez model. Im niższa wartość MSE, tym lepiej dopasowany model.

$$MSE = \frac{1}{n} \sum_{i=1}^n (y_i - \hat{y}_i)^2$$

2. Mean Absolute Error (MAE): MAE to średnia wartość bezwzględnych różnic między prawdziwymi wartościami (y_i) a przewidywanymi wartościami (\hat{y}_i). Podobnie jak MSE, im niższa wartość MAE, tym lepiej dopasowany model.

$$MAE = \frac{1}{n} \sum_{i=1}^n |y_i - \hat{y}_i|$$

3. R^2 score: R^2 score, znany również jako współczynnik determinacji, mierzy, jak dobrze model regresyjny dopasowuje się do danych. Wartość R^2 znajduje się w zakresie od 0 do 1, gdzie 1 oznacza idealne dopasowanie modelu do danych.

$$R^2 = 1 - \frac{\sum_{i=1}^n (y_i - \hat{y}_i)^2}{\sum_{i=1}^n (y_i - \bar{y})^2}$$

2.2.5 Metody klasyfikacji gatunku muzycznego.

Dane zostały tutaj przetworzone w podoby sposób co na początku problemu predykcji. Nie encodowano jedynie gatunku. W tym procesie usunięto natomiast rekordy mające wartość "Inne" w kolumnie gatunku muzycznego. Dodatkowo ze względu na nieliczbowy typ tej kolumny, zastosowano **Label Encoding**. Metoda przypisuje każdemu gatunkowi określoną wartość liczbową np. "pop" = 1. Następnie dane podzielono na zestaw testowy i treningowy i dokonano standaryzacji.

Użyte modele:**1. Logistic Regresion**

- Aby model ten pasował do problemu wieloklasowego, w implemenacji modelu z biblioteki sklearn użyto parameterów:
multi class='multinomial'- Określa strategię klasyfikacji dla wielu klas. W przypadku 'multinomial', model będzie dostosowany do wieloklasowej regresji logistycznej.
solver='lbfgs' - Wskazuje, który algorytm optymalizacyjny zostanie użyty do dopasowania modelu. 'lbfgs' jest jednym z algorytmów optymalizacyjnych, który może być używany w regresji logistycznej dla problemów wieloklasowych.

2. KNN

- Algorytm klasyfikacji, który klasyfikuje nowy punkt danych na podstawie większościowej klasy jego k najbliższych sąsiadów w przestrzeni cech, gdzie k jest ustaloną liczbą sąsiadów.
- Aby wybrać odpowiednią liczbę sąsiadów, obliczana jest dokładność dla różnych wartości k (z pomocą k-krotnej walidacji krzyżowej, z podziałem na 3 foldy). Wybrana liczba sąsiadów zależy od wartości dokładności.

3. SVC

- Algorytm klasyfikacji, który znajduje hiperpłaszczyznę najlepiej oddzielającą klasy w przestrzeni wielowymiarowej, maksymalizując margines między najbliższymi punktami danych z różnych klas (zwany wektorami wspierającymi).

4. Random Forest**5. XGBOOST****6. MLP**

Tak jak w przypadku predykcji przeprowadzana jest analogiczna walidacja modeli.

Metryki używane w ocenie modeli klasyfikacyjnych:

1. Accuracy (dokładność) jest miarą ogólnej skuteczności modelu klasyfikacji. Mierzy ona stosunek poprawnie sklasyfikowanych przypadków do ogólnej liczby przypadków.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

2. Recall, znany również jako czułość, mierzy zdolność modelu do identyfikacji wszystkich rzeczywistych pozytywnych przypadków. Jest to stosunek prawdziwie pozytywnych przypadków (TP) do sumy prawdziwie pozytywnych przypadków i fałszywie pozytywnych przypadków (FN).

$$\text{Recall} = \frac{TP}{TP + FN}$$

3. Precision mierzy, jak wiele z pozytywnych przewidywań modelu jest prawdziwymi pozytywnymi przypadkami. Jest to stosunek prawdziwie pozytywnych przypadków (TP) do sumy prawdziwie pozytywnych przypadków i fałszywie pozytywnych przypadków (FP).

$$\text{Precision} = \frac{TP}{TP + FP}$$

4. F1 Score: F1 Score jest średnią harmoniczną precision i recall. Jest używany do wyważenia między precision a recall w przypadkach, gdy istnieje nierównowaga między klasami.

$$F1Score = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

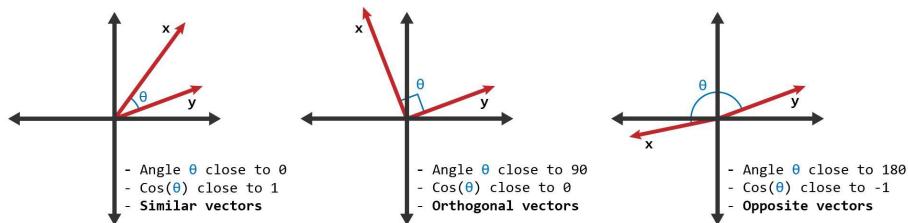
2.2.6 System rekomendacji.

W celu dokonania rekomendacji potrzebne są dwie bazy. Pierwsza baza czyli ta, której użyto do uczenia modeli. Druga to baza piosenek użytkownika, który chce otrzymać rekomendacje. Pobierając link od użytkownika, pobiera się z tej playlisty piosenki i tworzy dwie tabele. Jedną do analizy, drugą encodowaną (same wartości numeryczne z zastosowanym one hot encodingiem) do procesu klasteryzacji. Każdą z tabel osobno skaluje się do wartości w przedziale od 0 do 1, używając MinMaxScaler z biblioteki sklearn.

Następnie tworzony jest wektor średnich wartości ze wszystkich piosenek z playlisty dodanej przez użytkownika. Rekomendacja odbywa się za pomocą podobieństwa cosinusowego. Dla każdej piosenki w bazie tworzony jest wektor i sprawdzane jest podobieństwo tego wektora z uśrednionym wektorem z playlisty użytkownika. Im wartość większa, tym dwie piosenki są bardziej podobne/zbliżone do siebie. Aby rekomendacje nie zawierały piosenek znajdujących się już na playliście użytkownika, usuwane są duplikaty.

Wzór na cosinus kąta między wektorami \mathbf{u} i \mathbf{v} można zapisać w następujący sposób:

$$\cos(\theta) = \frac{\mathbf{u} \cdot \mathbf{v}}{\|\mathbf{u}\| \|\mathbf{v}\|}$$



Rysunek 6: Podobieństwo cosinusowe

W drugiej wersji zanim dokona się liczenia podobieństw cosinusowych, klastruje się dane za pomocą algorytmu KMeans. Uśredniony wektor, wyznaczony na tej samej zasadzie co wcześniej, klastrowany jest za pomocą modelu. Wyznaczane jest podobieństwo cosinusowe między uśrednionym wektorem, a piosenkami, które znajdują się w tym samym klastrze co uśredniony wektor.

Na samym początku należy znaleźć odpowiednią liczbę klastrów. Do tego zadania sprawdzana jest wartość silhouette dla liczby klastrów z przedziału od 2 do 10. Silhouette score mierzy, jak dobrze są oddzielone klastry oraz jak bardzo są spójne. Wartość wskaźnika sylwetki mieści się w przedziale

od -1 do 1, przy czym wyższe wartości wskazują na lepiej zdefiniowane klastry. Wzór na Silhouette Score można zapisać w następujący sposób:

$$\text{SilhouetteScore} = \frac{1}{N} \sum_{i=1}^N \frac{b(i) - a(i)}{\max\{a(i), b(i)\}}$$

$a(i)$ to średnia odległość między punktem i a innymi punktami w tym samym klastrze

$b(i)$ to najmniejsza średnia odległość między punktem i a punktami w innym klastrze, do którego nie należy.

Drugą metryką oceny jest sprawdzenie jak prezentuje się kwadratowa suma odległości punktów. SSD oblicza się poprzez zsumowanie kwadratowych odległości między wszystkimi parami punktów danych w obrębie klastra. Dla klastra C zawierającego n punktów danych, SSD jest dane wzorem:

$$SSD = \sum_{i,j} (x_i - x_j)^2$$

gdzie:

x_i i x_j są punktami danych w klastrze C

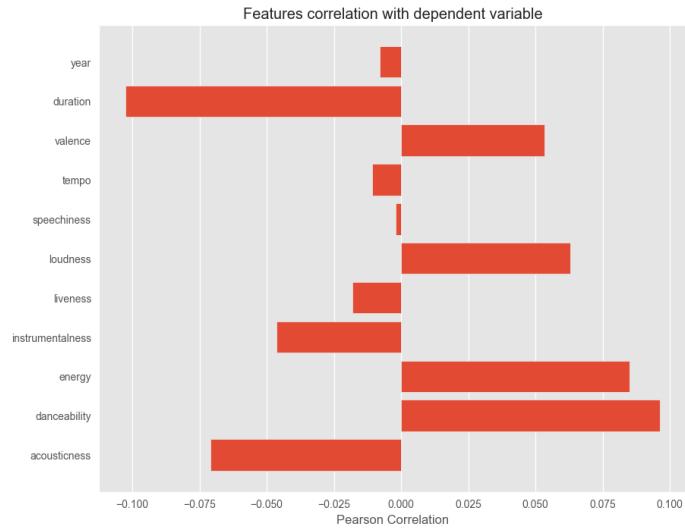
Znanym sposobem do znalezienia optymalnej liczby klastrów, przy użyciu obliczonych SSD, jest metoda tzw. łokcia, która pozwala patrząc na wykres znaleźć optymalną liczbę klastrów. Została ona również użyta w tym projekcie.

2.2.7 Aplikacja

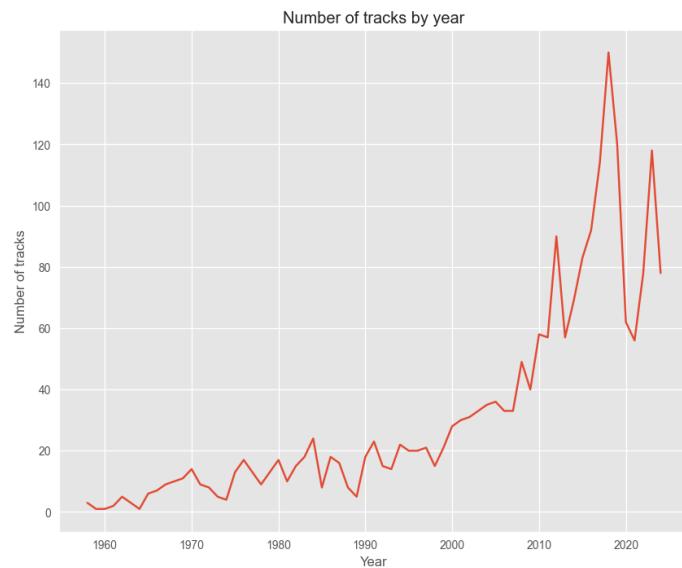
Aplikacja została stworzona przy użyciu biblioteki streamlit. Składa się z dwóch części: predyktora popularności oraz rekommendacji piosenek. W każdym z tych elementów oprócz głównej funkcjonalności, użytkownik może również przeprowadzić analizy swojej bazy lub bazy aplikacji, poznać zależności między zmiennymi, rozkłady, najpopularniejsze utwory i wiele więcej. Aby działała ona należycie zaimplementowano omawiane metody i narzędzia do predykcji i rekommendacji. Modele zapisano uwzględniając pliki pickle, a następnie użyto ich w kodzie aplikacji. Moduł pickle w Pythonie służy do serializacji obiektów Pythona do strumienia bajtów, a także deserializacji obiektów z takiego strumienia.

3 Wyniki

3.1 Wyniki EDA.

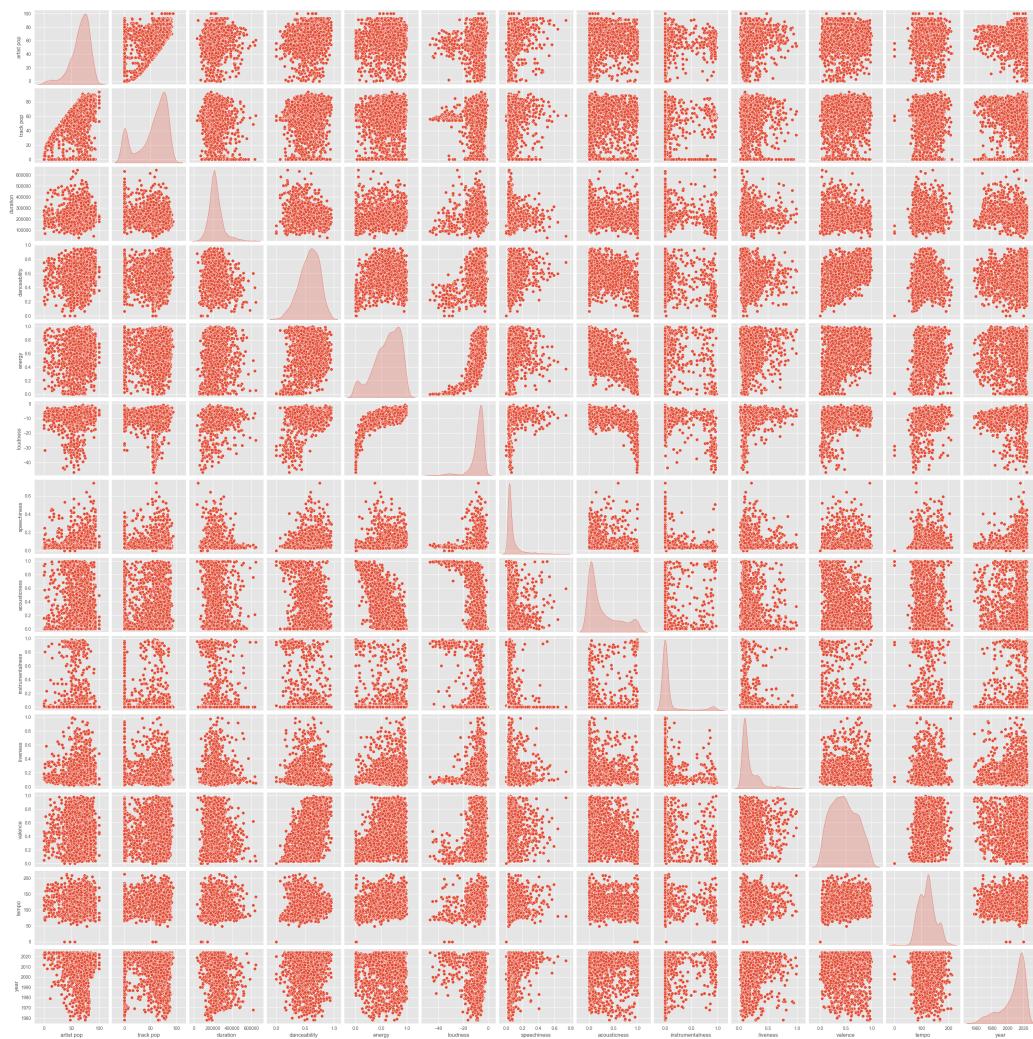


Rysunek 7: Korelacje pomiędzy zmiennymi numerycznymi a wartością popularności piosenki.

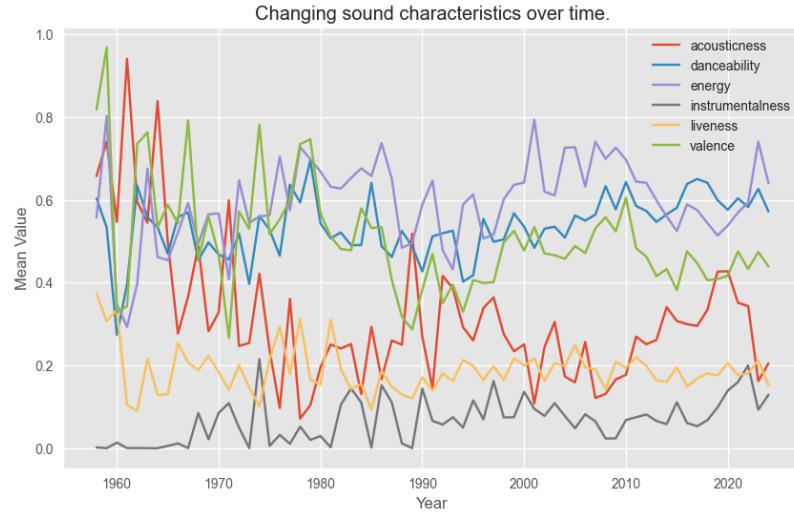


Rysunek 8: Ilość piosenek względem roku wydania.

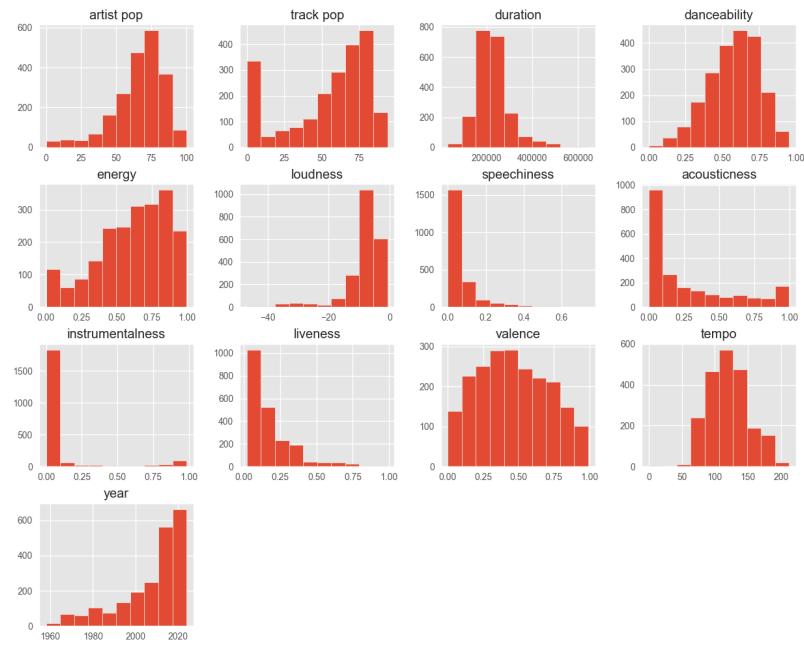
3.1 Wyniki EDA.



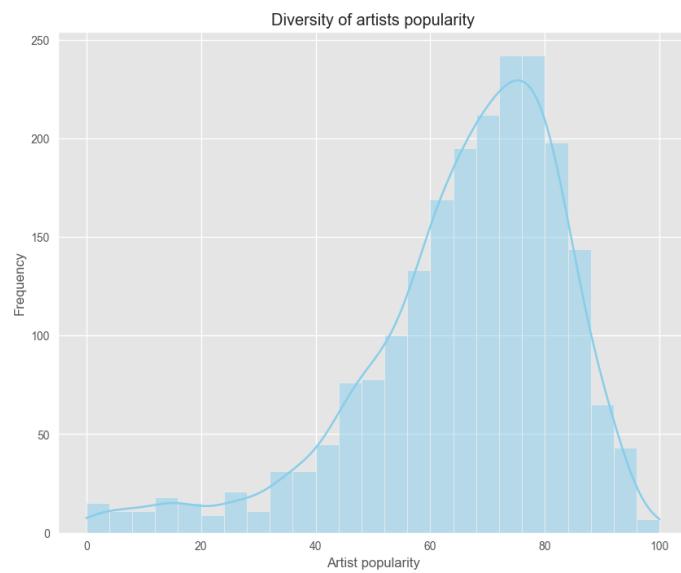
Rysunek 9: Korelacje między wszystkimi zmiennymi numerycznymi.



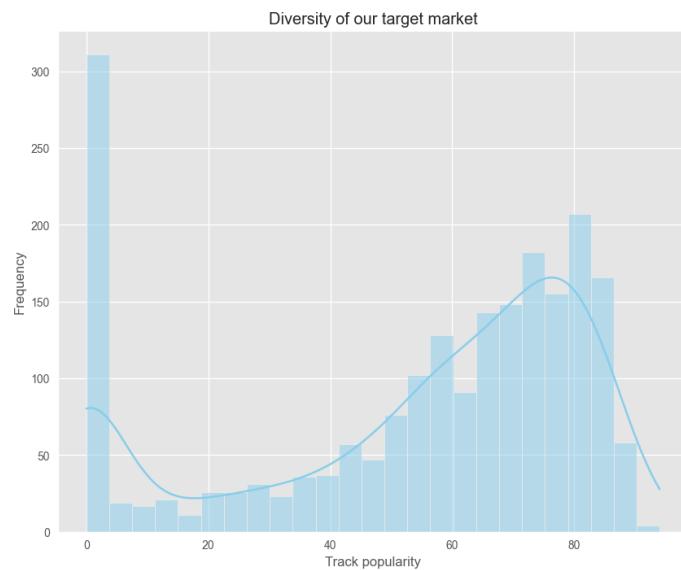
Rysunek 10: Zmiana cech piosenek (*acousticness, danceability, energy, instrumentalness, liveness, valence*) względem lat.



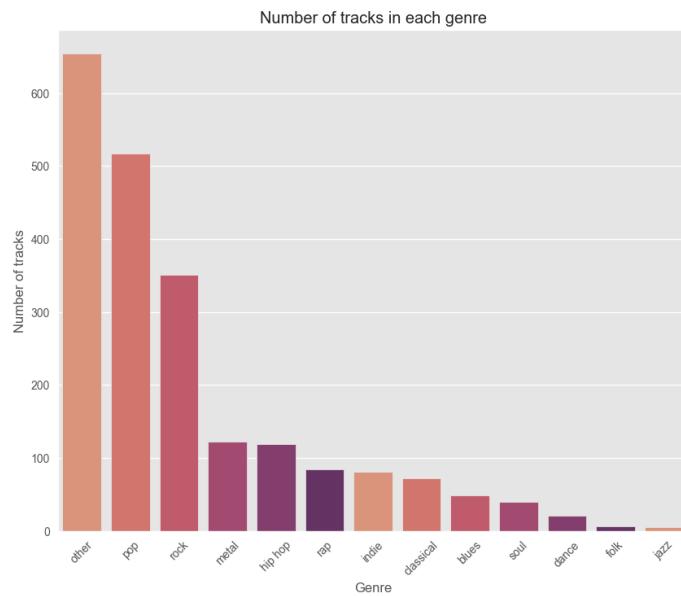
Rysunek 11: Histogramy zmiennych.



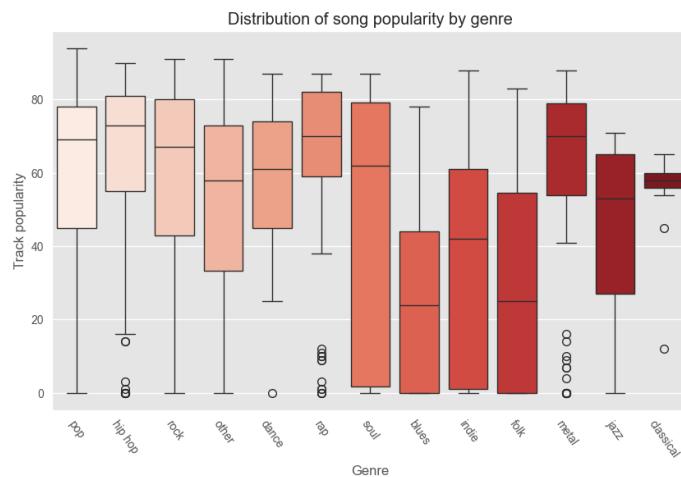
Rysunek 12: Rozkład popularności wykonawców.



Rysunek 13: Rozkład popularności piosenek.



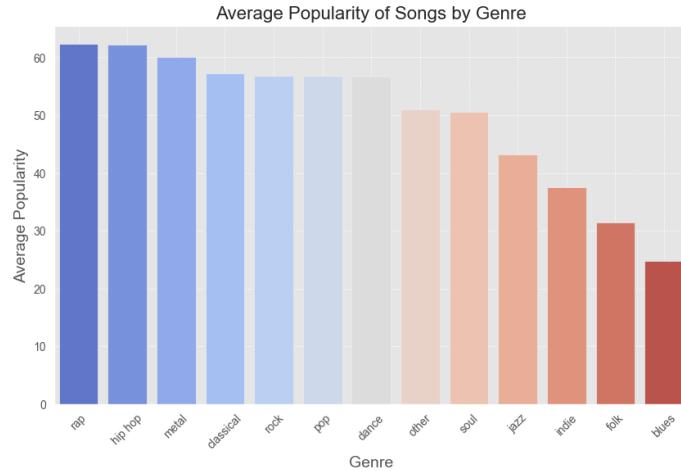
Rysunek 14: Liczba piosenek w każdym z gatunków.



Rysunek 15: Rozkład popularności w zależności od gatunku.

Wyniki testu statystycznego ANOVA, sprawdzającego czy średnia popularność piosenki zależy od jej gatunku.

$$F - \text{statistic} = 10.789, \quad p - \text{value} = 4.016 \times 10^{-21} \quad (1)$$

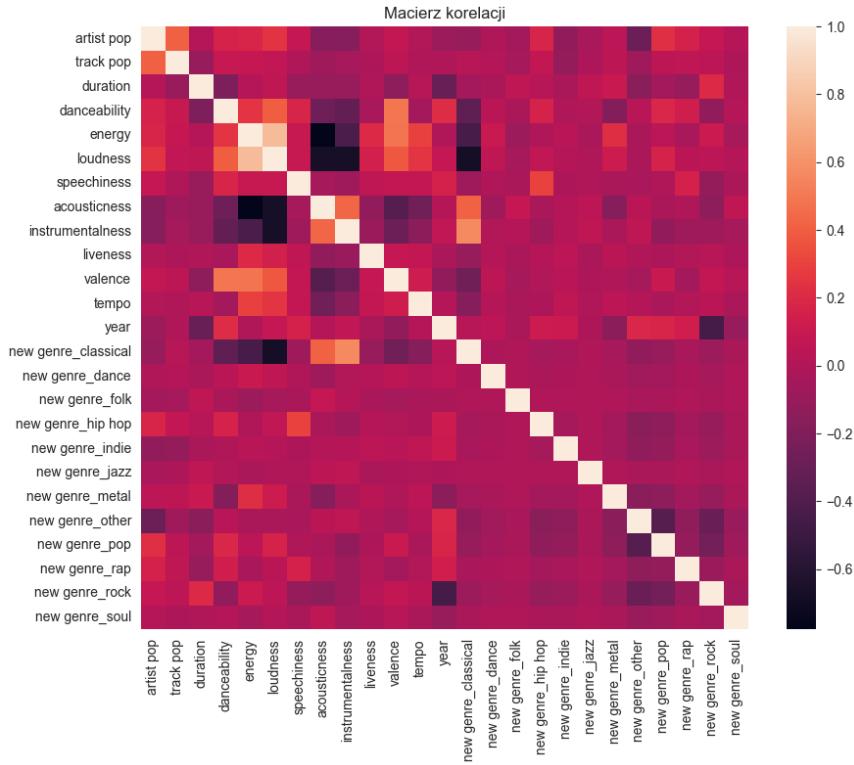


Rysunek 16: Średnia popularność piosenek w każdym gatunku.

3.2 Wyniki predykcji popularności.

- Pary skorelowanych cech:
 - ('acousticness', 'energy', 0.7778653706571921)
 - ('acousticness', 'loudness', 0.6702157700395519)
 - ('instrumentalness', 'loudness', 0.6734956210013407)
 - ('new genre_classical', 'loudness', 0.6819621263159126)
 - ('loudness', 'energy', 0.7799460045774012)

- Macierz korelacji



Rysunek 17: Korelacje między wszystkimi cechami (po one hot encoding).

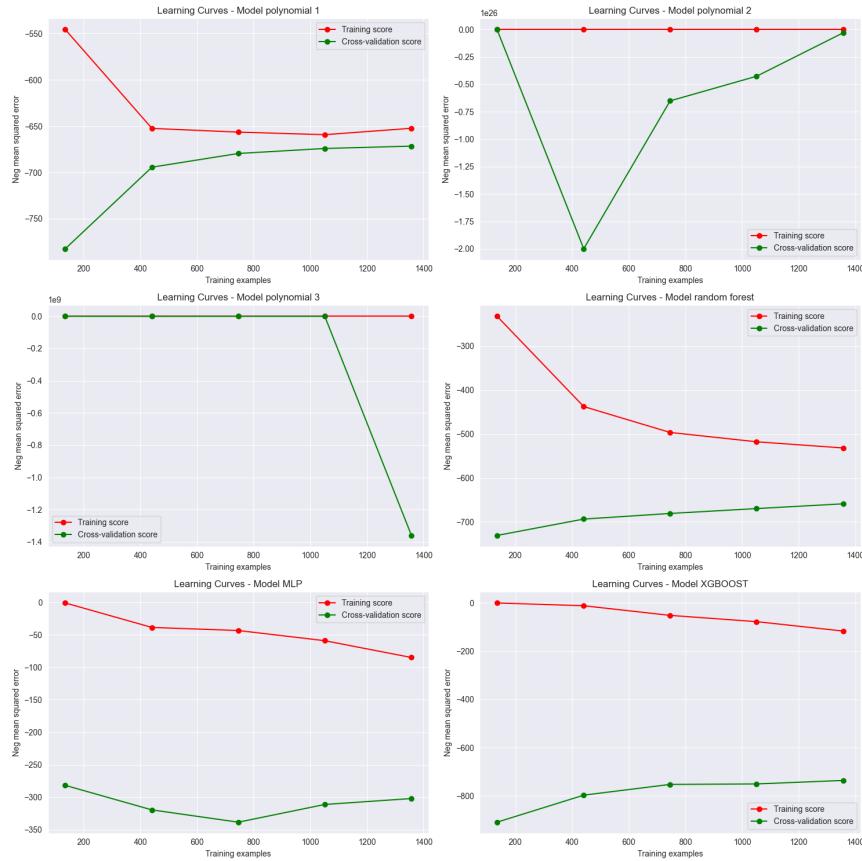
- Wyniki walidacji.

Model	wynik 1	wynik 2	wynik 3	wynik 4	wynik 5	średni wynik
regresja liniowa	670	713	691	594	683	670
regresja wielomianowa st.2	9.43e+24	7.56e+24	2.35e+24	9.90e+23	5.32e+20	4.06e+24
regresja wielomianowa st.3	8.80e+08	5.36e+09	7.13e+04	1.67e+04	1.50e+09	1.55e+09
random forest	633	708	663	576	679	652
XGBOOST	699	839	710	713	670	726
MLP	437	504	333	308	244	570

Tabela 1: Tabela wyników walidacji.

3.2 Wyniki predykcji popularności.

- "Learning curve" - pomoc przy sprawdzaniu overfittingu, czyli zjawiska przeuczenia danych. Wykres błędu dla zestawu treningowego i walidacyjnego.



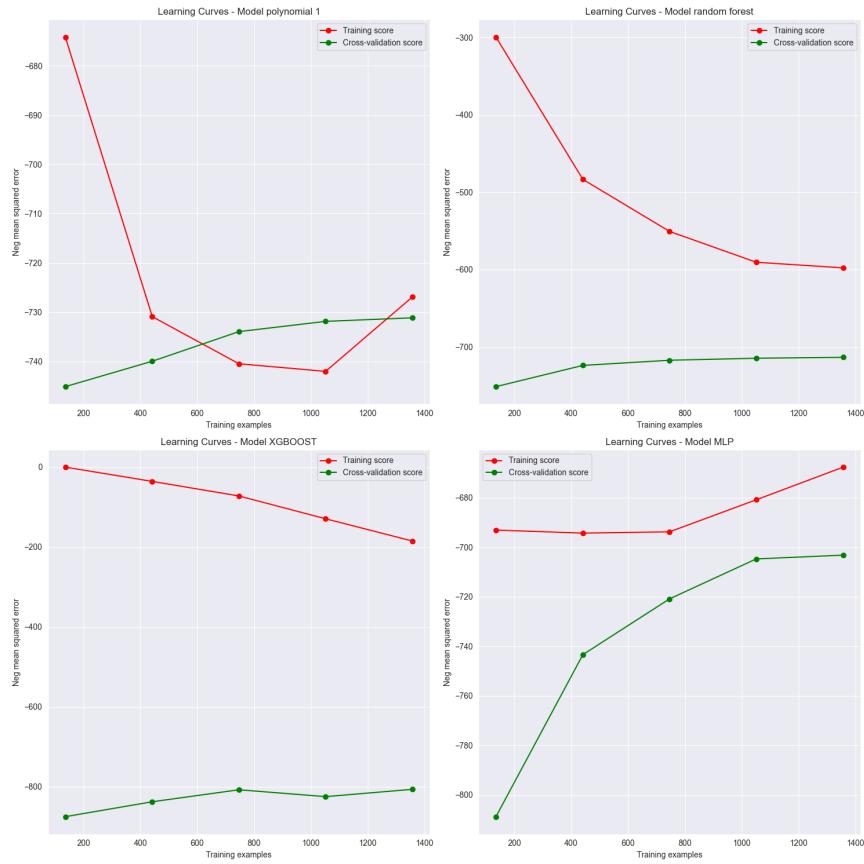
Rysunek 18: Wykres uczenia.

- Ocena modeli.

Model	MSE	MAE	R^2
linear	6.520649×10^2	1.967607×10^1	1.869639×10^{-1}
polynomial (2 degree)	6.646627×10^{22}	1.797378×10^{10}	-8.287439×10^{19}
polynomial (3 degree)	4.444156×10^{23}	1.057250×10^{11}	-5.541257×10^{20}
random forest	6.156997×10^2	1.885790×10^1	2.323064×10^{-1}
XGBOOST	6.629870×10^2	1.897182×10^1	1.733455×10^{-1}
MLP	1.381084×10^3	2.763827×10^1	-7.220239×10^{-1}

Tabela 2: Tabela metryk modeli.

- Learning curve po dodaniu PCA.



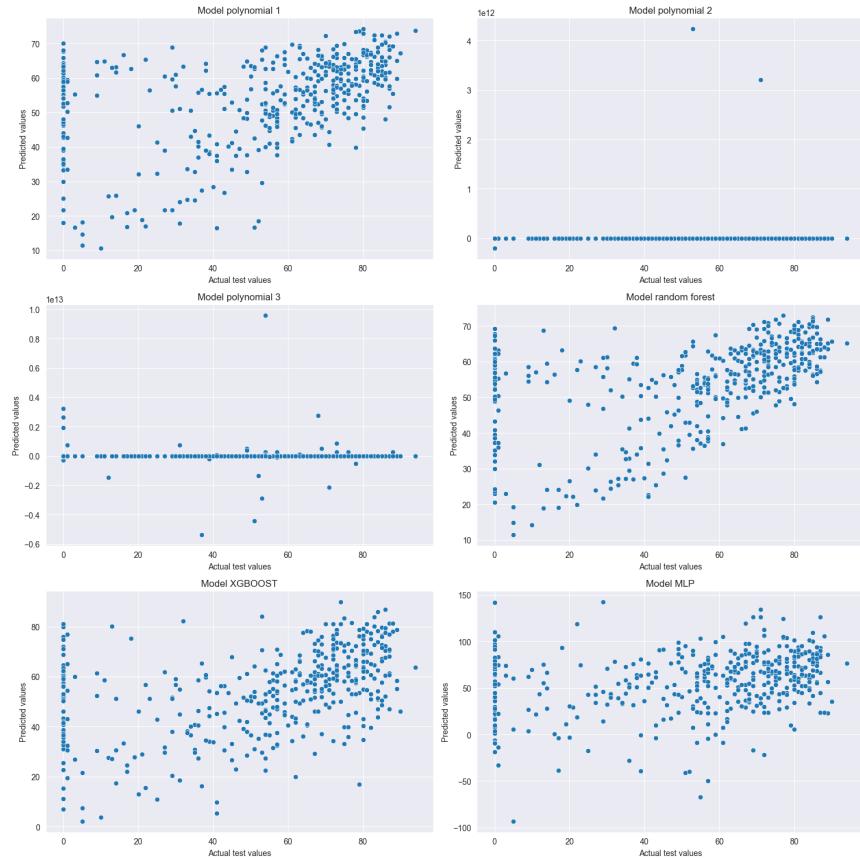
Rysunek 19: Wykres uczenia po dodaniu PCA.

- Ocena modeli po dodaniu PCA.

Model	MSE	MAE	R^2	MSE-PCA
linear	6.520649×10^2	1.967607×10^1	1.869639×10^{-1}	729.732303
polynomial (2 degree)	6.646627×10^{22}	1.797378×10^{10}	-8.287439×10^{19}	-
polynomial (3 degree)	4.444156×10^{23}	1.057250×10^{11}	-5.541257×10^{20}	-
random forest	6.156997×10^2	1.885790×10^1	2.323064×10^{-1}	708.183887
XGBOOST	6.629870×10^2	1.897182×10^1	1.733455×10^{-1}	824.288513
MLP	1.381084×10^3	2.763827×10^1	-7.220239×10^{-1}	723.808998

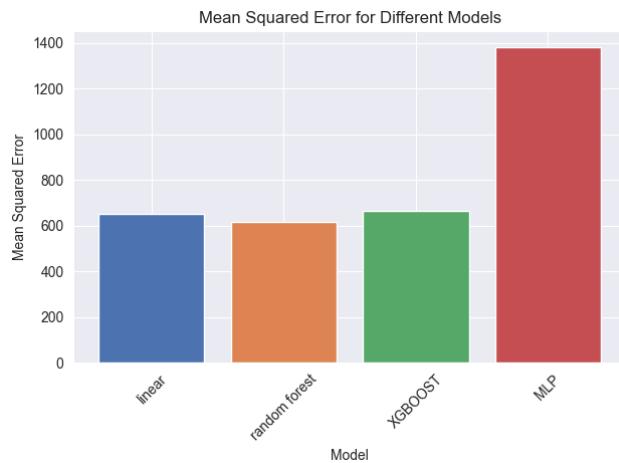
Tabela 3: Tabela wyników oceny modeli z użyciem PCA.

- Wykresy predykowanych wartości w stosunku do prawdziwych.

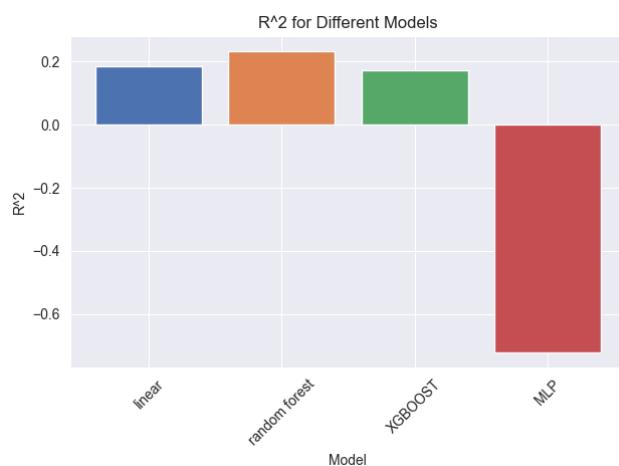


Rysunek 20: Wykres predykowanej wartości w stosunku do prawdziwej.

- Wykresy błędów modeli.



Rysunek 21: Wykres błędu najmniejszych kwadratów (MSE)



Rysunek 22: Wykres współczynnika determinacji.

3.3 Wyniki klasyfikacji.

- Sprawdzenie odpowiedniej liczby sąsiadów dla modelu KNN.



Rysunek 23: Wykres dokładności w zależności od liczby sąsiadów.

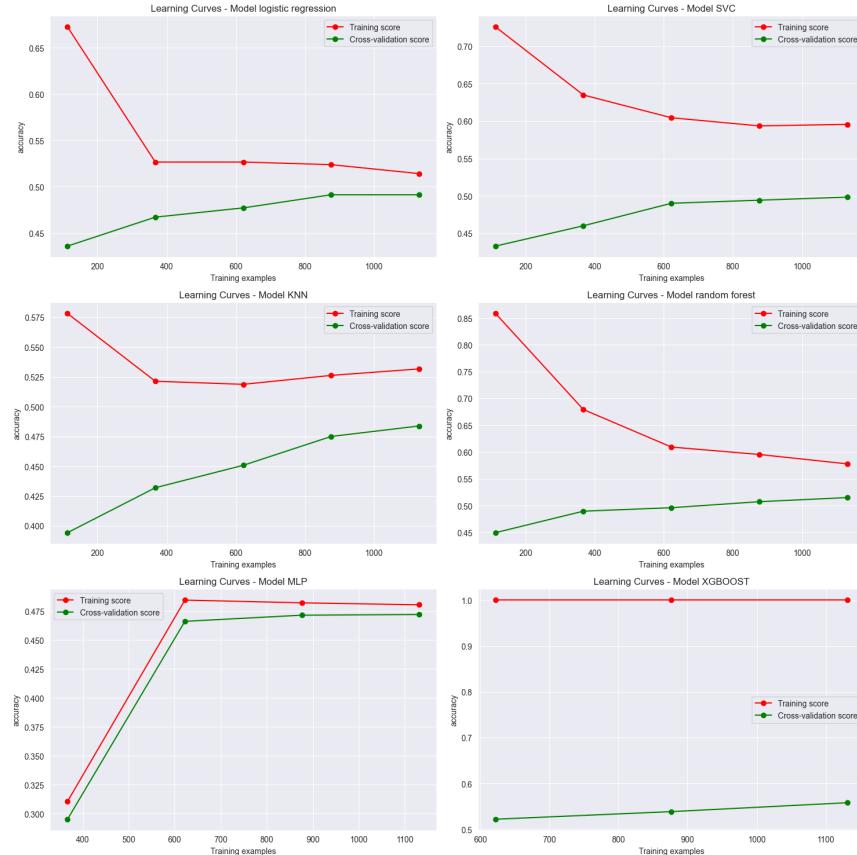
- Walidacja modeli używając metryki dokładności.

Model	Score 1	Score 2	Score 3	Mean Score
logistic regression	0.489399	0.498233	0.481416	0.489683
KNN	0.489399	0.466431	0.484956	0.480262
SVC	0.501767	0.489399	0.490265	0.493811
random forest	0.498233	0.508834	0.518584	0.508550
XGBOOST	0.537102	0.570671	0.541593	0.549789
MLP	0.379859	0.443463	0.470796	0.336514

Tabela 4: Wyniki walidacji modeli.

3.3 Wyniki klasyfikacji.

- Learning curve.



Rysunek 24: Wykres uczenia się dla modeli klasyfikacji.

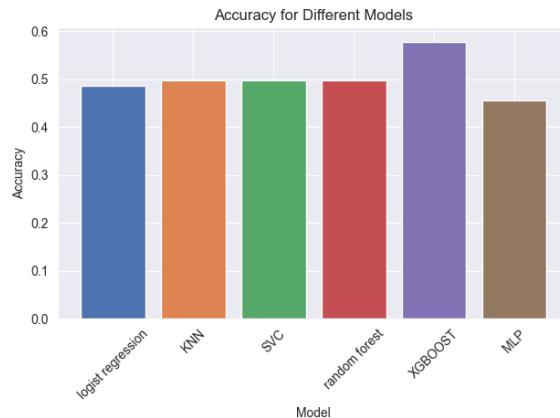
- Ocena modeli.

Model	Accuracy	Precision	Recall	F1
Logistic Regression	0.487059	0.455270	0.487059	0.453991
KNN	0.498824	0.476536	0.498824	0.453063
SVC	0.498824	0.428009	0.498824	0.442741
Random Forest	0.498824	0.413498	0.498824	0.437188
XGBOOST	0.578824	0.586585	0.578824	0.566319
MLP	0.456471	0.331538	0.456471	0.381567

Tabela 5: Metryki modeli klasyfikacji.

3.3 Wyniki klasyfikacji.

- Dokładność modeli.



Rysunek 25: Wykres słupkowy dokładności dla różnych modeli.

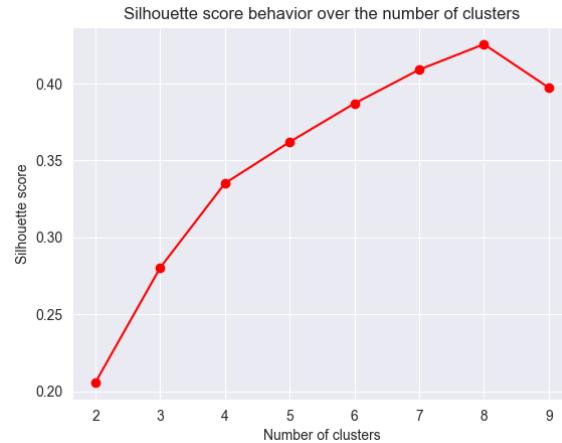
- Tablica pomyłek dla najlepszego modelu - XGBOOST.

Confusion Matrix												
True labels	0	1	2	3	4	5	6	7	8	9	10	11
	6	0	0	0	0	0	0	3	1	0	3	0
	0	12	0	0	0	0	0	2	0	0	0	0
	0	0	1	0	0	0	0	3	1	0	0	0
	0	0	0	4	0	0	0	5	12	0	0	0
	0	0	0	0	6	0	1	9	3	0	1	0
	0	0	0	0	0	0	0	0	1	0	0	0
	0	0	0	0	0	0	0	14	6	1	0	6
	1	4	0	0	3	0	3	91	18	2	9	0
	0	0	0	1	3	0	1	26	67	3	6	0
	0	0	0	2	0	0	0	3	6	6	0	0
	0	0	0	0	2	0	4	12	6	2	38	0
	0	1	2	3	4	5	6	7	8	9	10	11

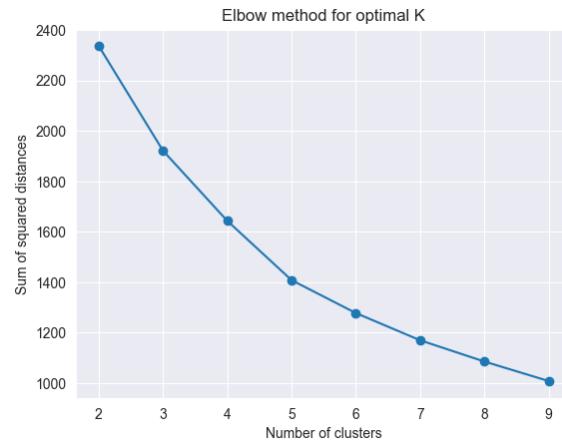
Rysunek 26: Tablica pomyłek modelu XGBOOST.

3.4 Wyniki klasteryzacji i rekomendacji.

- Wykresy pomagające w wyborze liczby klastrów.

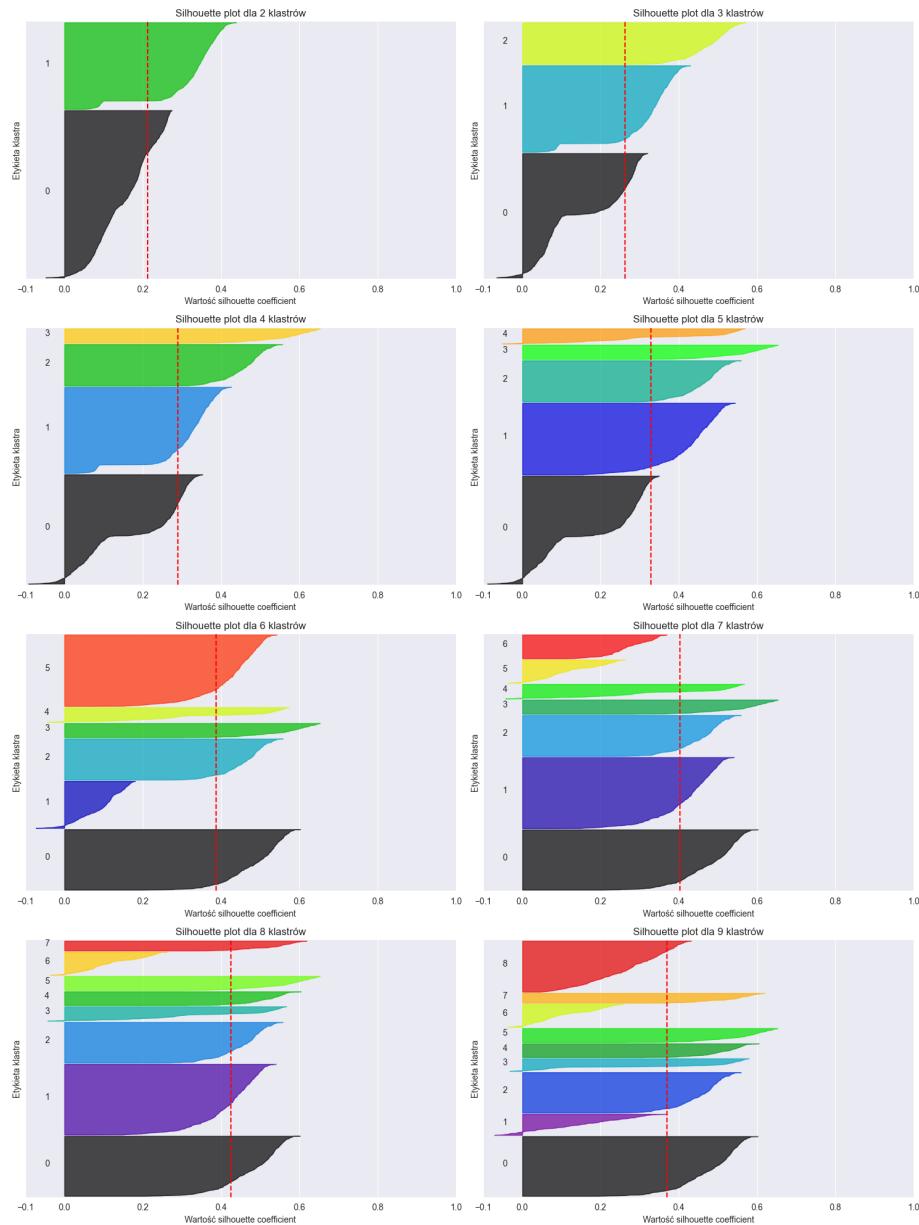


Rysunek 27: Silhouette score w zależności od liczby klastrów.



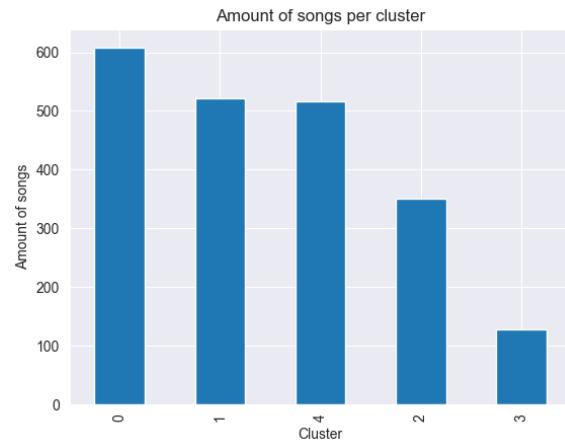
Rysunek 28: Metoda tzw. łokcia, czyli SSD w zależności od liczby klastrów.

3.4 Wyniki klasteryzacji i rekomendacji.



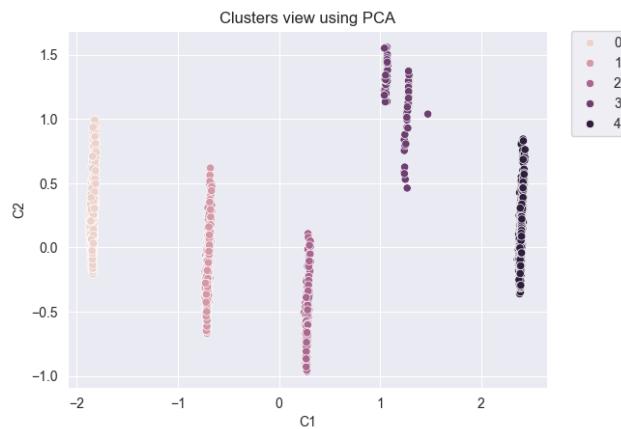
Rysunek 29: Wykresy wartości silhouete dla różnych klastrów.

- Liczba piosenek w każdym klastrze.



Rysunek 30: Ilość piosenek w poszczególnych klastrach.

- Wizualizacja klastrów za pomocą PCA.



Rysunek 31: Wizualizacja klastrów.

- Pierwsze 10 piosenek z każdego klastra.

	artist	track	album
8	The Weeknd	The Hills	Beauty Behind The Madness
9	One Direction	History	Made In The A.M. (Deluxe Edition)
16	Little Mix	Secret Love Song (feat. Jason Derulo)	Get Weird (Deluxe)
18	Marshmello	Happier	Happier
35	The Wanted	Walks Like Rihanna	Word Of Mouth (Deluxe)
42	Alan Walker	Faded	Faded
52	Macklemore & Ryan Lewis	Can't Hold Us (feat. Ray Dalton)	The Heist
69	One Direction	What Makes You Beautiful	Up All Night
70	One Direction	Story of My Life	Midnight Memories (Deluxe)
71	One Direction	Drag Me Down	Made In The A.M. (Deluxe Edition)

Rysunek 32: Klaster 0

	artist	track	album
1	JAY-Z	Empire State Of Mind	The Blueprint 3
2	Tyler, The Creator	ARE WE STILL FRIENDS?	IGOR
6	Eminem	Without Me	The Eminem Show
14	Sia	Cheap Thrills (feat. Sean Paul)	Cheap Thrills (feat. Sean Paul)
25	Sia	Chandelier	Chandelier
58	Roddy Ricch	The Box	Please Excuse Me for Being Antisocial
79	Alicia Keys	Girl on Fire	Girl on Fire (Remixes) - EP
88	CeeLo Green	Forget You	The Lady Killer (The Platinum Edition)
104	Tourist LeMC	En Route	En Route
105	Arno	Les yeux de ma mère (en concert)	En concert (À la française)

Rysunek 33: Klaster 1

	artist	track	album
4	Queen	Bohemian Rhapsody - Remastered 2011	A Night At The Opera (2011 Remaster)
5	Arctic Monkeys	Why'd You Only Call Me When You're High?	AM
44	Journey	Don't Stop Believin'	Flash Back International
49	The Script	Hall of Fame (feat. will.i.am)	#3 Deluxe Version
53	Mumford & Sons	I Will Wait	Babel
55	Panic! At The Disco	High Hopes	High Hopes on Saturday Night
73	OneRepublic	Counting Stars	Native
82	Queen	Another One Bites The Dust - Remastered 2011	The Game (2011 Remaster)
89	WALK THE MOON	Lost In The Wild	What If Nothing
100	Queen	Love Of My Life - Remastered 2011	A Night At The Opera (Deluxe Edition 2011 Rema...

Rysunek 34: Klaster 2

3.4 Wyniki klasteryzacji i rekommendacji.

3 WYNIKI

	artist	track	album
130	Lévon Minassian	Siretsi Yares Daran (They Have Taken the One L..)	The Doudouk - Beyond Borders
131	Ibrahim Maalouf	True Sorry	Illusions
132	Rabih Abou-Khalil	Sahara	Blue Camel
155	John Cage	Experiences No. 2	Different Every Time
156	Philip Glass	The Poet Acts	The Hours (Music from the Motion Picture Sound..)
183	David Gilmour	Red Sky at Night	On an Island
189	Kokoroko	Abusey Junction	We Out Here
193	Agnes Obel	September Song	Aventine (Deluxe)
686	Bobby "Blue" Bland	Ain't No Love In The Heart Of The City - Singl..	Dreamer
831	Jack Stauber's Micropop	Just Take My Wallet	Micropop

Rysunek 35: Klaster 3

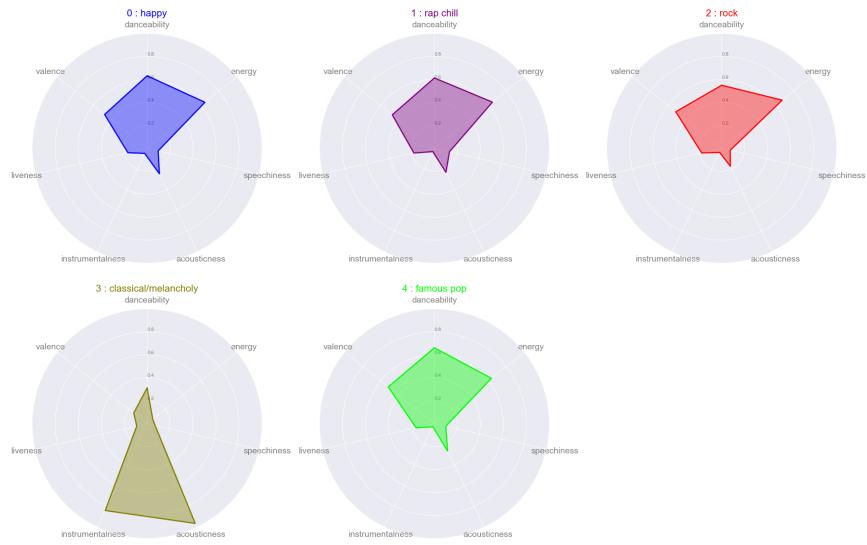
	artist	track	album
0	Rihanna	Desperado	ANTI (Deluxe)
3	Billie Eilish	i love you	WHEN WE ALL FALL ASLEEP, WHERE DO WE GO?
7	Ariana Grande	no tears left to cry	no tears left to cry
10	Bruno Mars	Grenade	Doo-Wops & Hooligans
11	Sam Smith	Stay With Me	In The Lonely Hour (Deluxe Edition)
12	Jonas Blue	Rise	Rise
13	Ava Max	Sweet but Psycho	Sweet but Psycho
15	P!nk	What About Us	Beautiful Trauma
17	Beyoncé	Halo	I AM...SASHA FIERCE
19	Jonas Blue	Fast Car	Fast Car

Rysunek 36: Klaster 4

- Cechy piosenek w poszczególnych klastrach.

Cluster	Danceability	Energy	Speechiness	Acousticness	Instrumentalness	Liveness	Valence
0	0.632475	0.643606	0.098380	0.246058	0.048864	0.171485	0.473363
1	0.613195	0.645905	0.134472	0.231886	0.031729	0.183331	0.469106
2	0.549949	0.673994	0.075914	0.173076	0.039967	0.178854	0.511417
3	0.312163	0.062864	0.071389	0.964409	0.840152	0.092398	0.148871
4	0.661409	0.634658	0.101189	0.264234	0.030362	0.163416	0.513985

Tabela 6: Tabela z klastrami i odpowiadającymi im cechami.



Rysunek 37: Cechy piosenek w zależności od klastra.

- 20 zarekomendowanych piosenek dla playlisty pod nazwą *100 Greatest Rock Songs*, którą można znaleźć pod linkiem playlista

Track	Artist	Similarity Score
You Shook Me All Night Long	AC/DC	0.981242
When I Come Around	Green Day	0.980648
Losing My Religion	R.E.M.	0.979764
Somebody Told Me	The Killers	0.978131
Buddy Holly	Weezer	0.975467
Last Nite	The Strokes	0.974845
Your Love	The Outfield	0.974211
Shut Up and Dance	WALK THE MOON	0.973110
Don't Go Breaking My Heart	Elton John	0.972789
Jump	Van Halen	0.971743
Carry on Wayward Son	Kansas	0.971130
Animal	Neon Trees	0.970950
Hellraiser	Ozzy Osbourne	0.970881
Everybody Talks	Neon Trees	0.970762
We Didn't Start the Fire	Billy Joel	0.970520
Sweet Emotion	Aerosmith	0.970504
Hall of Fame (feat. will.i.am)	The Script	0.970475
Believer	Imagine Dragons	0.970288
Are You Gonna Be My Girl	Jet	0.970115
Black Betty	Ram Jam	0.969786

Tabela 7: Tabela utworów, artystów i ich współczynników podobieństwa.

3.5 Aplikacja

Wyniki i działanie aplikacji można zobaczyć korzystając z linków:

- predyktor: app-predyktor
- system rekomendacji: app-rekomendacja

4 Analiza wyników

4.1 Analiza wyników EDA.

Pierwszy wykres 7 pokazuje, iż największy wpływ na wartość popularności piosenki mają następujące cechy: *duration, valence, loudness, instrumentalness, energy, danceability, acousticness*.

W zbiorze danych przeważająca liczba piosenek jest z okresu od 2013 do 2020 roku 8. Zbiór nie jest zrównoważony pod względem roku wydania. Można przypuszczać, iż powodem tego jest prosty fakt, iż z roku na rok sektor muzyczny się rozrasta. Dodatkowo, muzyka nagrywana przed 2000 rokiem, często nie jest dostępna i zdatna do udostępniania w aplikacjach streamingowych.

Wykres korelacji 9 pokazuje, że żadne z cech nie wykazują wykresu liniowego, tym samym nie ma czystej korelacji między nimi.

Na wykresie histogramów zmiennych 11 widać, iż zmienne mają różne skale, a więc istotne jest ustandaryzowanie ich. Ciekawą rzeczą można zobaczyć dla cechy instrumentalności, ponieważ zdecydowana większość piosenek ma ją ustawioną na 0, co oznacza, że w zbiorze mamy głównie śpiewane piosenki.

Wykresy rozkładu popularności, szczególnie wykres dotyczący popularności utworów 13, ukazuje, że zbiór nie jest zrównoważony pod tym kątem. Zdecydowana większość utworów jest zdefiniowana na popularność o wartości 0. Problem ten najprawdopodobniej wynika z faktu, iż skala zadana przez Spotify to 0-100. Może być tak, że nie jest ona wystarczająca i mając utwory popularne na całym świecie zbliżenie się do 100 dla utworów popularnych np. tylko w jednym kraju może być trudne i automatycznie ustawiane są na wartości niższe. Aby zrównoważyć bazę, w procesie predykcji próbowano dodać metodę z biblioteki SMOGN, która tworzy sztuczne punkty tak aby baza była zrównoważona. Niestety zabieg ten nie udał się, gdyż poprzez dużą skalę, biblioteka nie widziała potrzeby równoważenia bazy.

Wyniki testu ANOVA:

Wartość p jest mniejsza niż 0,05, co wskazuje, że istnieją statystycznie istotne różnice w średniej popularności utworów w różnych gatunkach. To oznacza, że niektóre gatunki są bardziej popularne od innych. 16

4.2 Analiza wyników predykcji popularności.

Z wyników korelacji 17 widać, iż szczególnie dwie cechy : 'instrumentalness', 'acousticness' mogą zostać usunięte, w celu zmniejszenia wymiarów, ponieważ są skorelowane z innymi cechami, a więc nie traci się informacji.

Walidacja modeli 1 pokazała, że szczególnie dwa modele - regresja wielomianowa st.2 i st.3 kompletnie nie nadają się do tego problemu. Ich wyniki są kilkanaście rzędów większe od pozostałych. Najlepsze wyniki osiąga model MLP. Dodatkowo, w każdym z modeli (oprócz dwóch wspomnianych na początku) wyniki w każdym fragmencie są podobne, co świadczy o dobrej jakości modeli.

Aby sprawdzić, czy nie zachodzi overfitting ukazano na wykresie 18 krzywe uczenia dla zestawu treningowego i walidacyjnego. Dla modelu regresji liniowej oraz random forest krzywe zbliżają się do siebie, co oznacza, że nie występuje zjawisko przeuczenia. Natomiast dla modeli MLP oraz XGBOOST różnice błędów między zestawem treningowym a testowym są dość spore, co może świadczyć o overfittingu.

Patrząc na tabele wyników modeli 2 widać, że najlepsze wyniki uzyskuje random forest. Modele regresji wielomianowej i tutaj okazały się bezużyteczne. Ciekawa kwestia wynika z rezultatów modelu MLP, gdyż w przypadku walidacji radził sobie najlepiej, natomiast w kwesti zbioru testowego jego MSE pogorszyło się. Jest to kolejny element mówiący za tym, iż dla tego modelu mogło wystąpić przeuczenie.

Aby sprawdzić, czy redukcja wymiarów pomoże w zmniejszeniu overfittingu, przeprowadzono PCA dla 5 komponentów. 19 Dla regresji liniowej zbiór walidacyjny osiąga teraz nawet mniejszy błąd niż treningowy. Poprawiła się również odległość między krzywymi dla modelu MLP. Co ciekawe w przypadku modelu XGBOOST, wykres nie poprawił się znacząco. Może to wynikać z charakterystyki tego modelu i jego bardzo dobrych wyników dla mniejszych zbiorów treningowych. Jeśli chodzi o wyniki metryk 3 to w przypadku modelu MLP znacząco poprawiło się MSE, dla innych modeli pogorszyło się. To świadczy o tym, że w przypadku MLP występował overfitting.

Z wykresów 20 i 21 wynika, że mimo nieznacznych różnic między 3 modelami, random forest radzi sobie najlepiej. Dla współczynnika determinacji 22 widać, że znów nieznacznie wygrywa random forest, mimo, że jego wartość nie jest najlepsza bo wynosi zaledwie około 0.23, co mówi o tym, że model ten jest nie dużo lepszy niż zwykłe "strzelanie" wyników.

4.3 Analiza wyników klasyfikacji.

Dla modelu KNN, problemem było wybranie odpowiedniej liczby sąsiadów. Posłużyono się tutaj wyborem na podstawie wykresu 23. Wybrano liczbę $k=22$.

Walidacja modeli 4 wykazała, że najgorzej radzi sobie model MLP, a najlepiej XGBOOST. Niestety wyniki dokładności dla XGBOOST i random forest były nieznacznie większe od 50%, gdy dla innych modeli były jeszcze mniejsze.

Wyniki wykresów uczenia nie wykazywały zdolności do overfittingu, jedynie dla XGBOOST dokładność na zbiorze testowym wynosiła prawie 100%, jednak wyniki walidacji z porównaniem wyników oceny tego modelu nie dają przesłanek o overfittingu.

Ostateczne wyniki 5 i 25 nie różnią się dużo od wyników walidacji. Najlepsze rezultaty osiąga XGBOOST, a najgorsze MLP.

4.4 Analiza wyników klasteryzacji i rekommendacji.

Początkowym zadaniem podczas klasteryzacji było wyznaczenie liczby klastrów. Analizując wykresy 27, 28, 29 wybrano liczbę 5. Najwięcej utworów trafiło do klastra 0, a najmniej do klastra 3 30. Wizualizację klastrów zrobiono za pomocą PCA, zmieniając wymiarowość do jedynie 2 31.

Analizując piosenki w poszczególnych klastrach, można przypuszczać iż do zerowego trafiło najwięcej piosenek i są to popularne, dość żywe popowe piosenki. Do klastra pierwszego trafiły piosenki z gatunku rap. Drugi natomiast to typowy rock, heavy-metal. Charakteryzuje się zwiększoną energią piosenek. Trzeci najbardziej różni się od pozostałych, szczególnie jeśli chodzi o instrumentalność. Są to piosenki jazzowe i klasyczne. Ostatni klaster jest najbardziej zbliżony do zerowego. Porównując pierwszymi dziesięcioma piosenkami ciężko przypuszczać jaki to dokładnie wzór stanowił rozróżnienie ich od pierwszego klastra.

Ostateczny wynik, czyli 20 zarekomendowanych piosenek 7, po dodaniu playlisty 100 hitów rockowych, wydaje się odpowiednia. Można zobaczyć, że rzeczywiście rekomendowane piosenki to popularne, znane wszystkim utwory rockowe. Wskaźniki podbieństwa cosinusowego świadczą o sporym podobieństwie utworów.

5 Wnioski

Modele predykcyjne okazały się dobrze spełniać swoją rolę. Trzy najlepsze z nich to: random forest, regresja liniowa oraz XGBOOST. W przypadku wielomianowych regresji wyniki zupełnie odbiegały od prawdziwych. Jeśli chodzi natomiast o MLP, to w przypadku tego modelu wykryto zjawisko przeuczenia. Poprawione zostało ono, używając PCA. To oznacza, że podejście zmniejszenia wymiarów w przypadku walki z overfittingiem, spełnia swoją rolę. Dodatkowo można byłoby zmieniać i zastanawiać się nad innymi licznosciami węzłów w warstwach ukrytych. Mimo całkiem dobrych predykcji, współczynnik determinacji nie jest najlepszy. Warto zauważać, że w modelach pozwolono na predykcję niecałkowitych liczb, jednak ocena popularności w Spotify zawiera liczby od 0 do 100, ale tylko całkowite. Dużym problem stanowiło to, iż baza była niezrównoważona pod względem popularności. Najwięcej wartości wynosiło 0. Na wykresie 20 dodatkowo można sprawdzić, że modele również wychwyciły tą nierównowagę w zbiorze i bardzo dużo punktów przewidziały na 0, nawet gdy prawdziwa wartość była znacznie większa. Tak jak wspomniano problem próbowało rozwiązać biblioteką SMOGN, jednak nie sprawdziła się ona dla tych danych. Warto więc poszerzyć wiedzę na temat równoważenia zbiorów lub lepiej dopasować utwory w bazie.

W przypadku klasyfikacji modele okazały się niezadawalające, gdyż ich średnia dokładność wynosiła około 50%, co jest za słabym wynikiem na udany klasyfikator. Od początku tego etapu, brak pozytywnych wyników był brany pod uwagę. Problem był skomplikowany, ponieważ pobierając dane, pobierano tylko jeden gatunek z kilku przypisanych do jednej piosenki. Dodatkowo łączna liczba różnych gatunków wynosiła około 230, a więc zmniejszono znacznie ich liczbę do najbardziej znanych (opis działania w sekcji Opis Badań). Te kroki spowodowały, że utracono dużą część informacji i modele nie były w stanie nauczyć się prawidłowych wzorców.

Klasteryzacja oraz system rekomendacji okazał się działać zadziwiająco dobrze. Oczywiście w tym przypadku ciężko o jakąś odpowiednią metrykę oceny wyników, gdyż jest to kwestia indywidualna, czy trafimy w gusta użytkownika. Jeśli chodzi natomiast o sprawdzanie wyników na osobie własnej oraz kilku znajomych okazało się, że predykcja jest dość dobra i zarekomendowane piosenki budziły zadowolenie wśród użytkowników. Problemem tutaj może być jedynie ograniczenie zbioru piosenek. Jeśli chodzi natomiast o algorytm klasteryzacji, działał on dobrze. Podzielił zbiór na 5 grup, z czego rzeczywiście dwie były dość podobne, natomiast reszta była rozróżnialna pod względem głównie gatunkowym. Podejście podobieństwa cosinusowego zarówno w wersji bez klasteryzacji oraz z nią, działało świetnie i nie było dużych różnic w zależności od wersji. Samo podobieństwo wychwyciło pewne klastry między piosenkami.

Stworzenie aplikacji w streamlit pozwoliło na interakcję z użytkownikami i szybsze przedstawianie wyników. Głównym elementem jaki można rozbudować i poprawić jest wielkość bazy danych. Poprzez spore ograniczenia ze strony aplikacji Spotify, zbieranie danych było utrudnione. Większa baza, pozwoli na lepsze wyniki modeli predykcyjnych oraz większą różnorodność w wyborze piosenek rekomendowanych.

Projekt ten przedstawia cały proces od stworzenia bazy danych do przeprowadzenia wszelkich badań wykorzystujących większość z poznanych podczas laboratoriów metod i praktyk. Poprzez rekomendacje, klasyfikacje, klasteryzacje zapoznano się zarówno z uczeniem nadzorowanym jak i

tym bez nadzoru. Dodatkowo wykorzystano także inne metody takie jak MLP czy XGBOOST, które dodały wartości analizom. Najważniejszym elementem i najbardziej pouczającym było ku zdziwieniu tworzenie własnej bazy danych. Pozwoliło to na lepsze poznanie problemu i naukę pracy z prawdziwymi danymi, które samemu trzeba przygotować do dalszych procesów projektu. Ciekawym elementem była również praca nad systemem rekommendacji, gdyż jest to temat dość "modny", a tym samym interesujący. Dodatkową wartością projektu jest fakt, że szczególnie taki system rekommendacji jest rzeczą niezbędną i wykorzystywana aktualnie przeze mnie oraz moich znajomych, więc ma to realne zastosowanie.

6 Dodatek

Kody źródłowe umieszczone zostały w repozytorium GitHub:

https://github.com/aleksandra0014/spotify_predictor_and_recommendation.