Semestrální projekt NI-VMM

Vojtěch Sillik, Ondřej Wrzecionko

Obsah

Úvod

Cílem projektu je vytvořit **webovou** aplikaci, do které uživatel nahraje libovolný audio soubor. Aplikace soubor **porovná** s databází písní v jednotlivých žánrech a zobrazí uživateli **míru shody** nahraného souboru s jednotlivými žánry v databázi.

Aplikace umožní také vybudování **databáze** žánrů na základě oanotování (manuálním určení žánru) jednotlivých "trénovacích" písní.

Aplikace umožní při porovnávání volit **parametry porovnání** *(podle kolika nejbližších skladeb se určuje příslušnost k žánru, jak přesné má porovnání být, s jakou váhou mají být použity jednotlivé deskriptory)*.

Analýza

Tato kapitola obsahuje popis analytické části projektu od výběru skladeb, deskriptorů, přes volbu metody porovnání deskriptorů a klastrování až po návrh samotné webové aplikace.

Skladby a žánry

Při porovnávání jednotlivých žánrů je nutné vybrat vhodně z každého žánru skladby, které tento žánr reprezentují. Zároveň je cílem, aby byly jednotlivé skladby v rámci žánru co "nejblíže" k sobě *(podobné)* a jednotlivé žánry co "nejdále" od sebe *(nejméně podobné)*.

Při výběru žánrů jsme zvolili následující žánry s jejich typickými charakteristikami, které budeme následně extrahovat pomocí zvolených deskriptorů:

- klasická hudba (**classic**), která je typická zastoupením **velkého počtu** hudebních nástrojů pokrývajících celé **spektrum** frekvencí a zároveň tato hudba **ne**obsahuje zpěv
- jazzová hudba (**jazz**), která má specifickou skladbu nástrojů, saxofonová **sóla** na vyšších frekvencích spektra a taktéž **ne**obsahuje zpěv
- **metal**, který se vyznačuje velkými skoky v rámci spektra (nízko položený zpěv → vysoko položený zpěv), zvláštní skladbou nástrojů
- **opera**, založená na **zpěvu** na vysokých nebo nízkých frekvencích, s minimem hudebního doprovodu
- **rap**, obsahující především **mluvené slovo** (ani ne příliš zpěv) na jedné podobné frekvenci po celou dobu skladby s minimem hudebního doprovodu
- **reggae**, se specifickým rytmem a varhanovými nástroji

Pro každý žánr jsme ručně *(ze skladeb, které poslouchámе a z knihovny https://freemusicarchive.org/)* vybrali skladby, které obsahují právě vlastnosti **specifické** pro daný žánr.

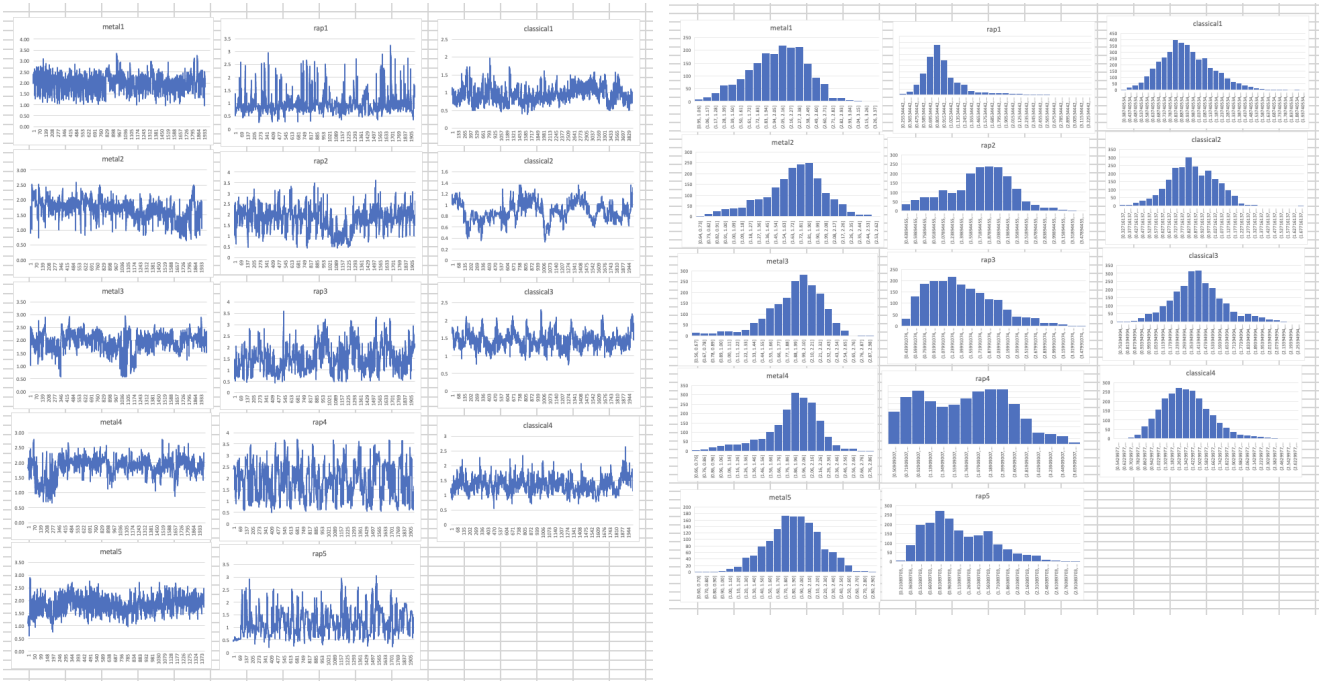
Deskriptory

K porovnání skladeb je nutné z nich **extrahovat** deskriptory popisující specifické charakteristiky. Provedli jsme tedy analýzu deskriptorů z knihoven MPEG7 a jAudio.

MPEG7

MPEG7 knihovna umožňuje extrahovat ze zadaného .wav souboru sadu některých MPEG7 **low-level** deskriptorů. Těchto deskriptorů je celkem **17**, knihovna ale pro předané audio soubory extrahovala pouze následující: SpectrumSpreadType, SpectrumCentroidType, SpectralEnvelope a SpectrumMinMax.

Pro sadu 3 žánrů (klasická hudba, metal, rap) jsme tedy spustili program MPEG7, zanesli získané hodnoty deskriptorů do grafu a histogramu a také jsme si spočítali průměr, medián, standardní odchylku těchto hodnot.



Graf a histogram časové řady deskriptoru `SpectrumSpreadType`

	metal1	metal2	metal3	metal4	metal5	rap1	rap2	rap3	rap4	rap5	classical1	classical2	classical3	classical4
average	2.04163044	1.67775558	1.92434377	1.86435342	1.86283884	0.97629352	1.79434111	1.48191683	1.86239076	1.1871216	0.92132347	0.87792805	1.44369878	1.39211891
min	0.95	0.64	0.56	0.66	0.60	0.26	0.44	0.44	0.51	0.21	0.39	0.33	0.75	0.54
max	3.37	2.58	2.96	2.78	2.90	3.23	3.61	3.58	3.69	3.06	1.96	1.70	2.31	2.65
median	2.08	1.72	1.98	1.92	1.87	0.88	1.85	1.40	1.90	1.07	0.89	0.87	1.43	1.37
dev	0.38602362	0.33310222	0.37678023	0.33560883	0.3341656	0.39524907	0.56599916	0.57816361	0.75777713	0.52491338	0.23811348	0.17817837	0.22242677	0.26882152

Tabulka s průměrem, mediánem a standardní odchylkou hodnot deskriptoru `SpectrumSpreadType`

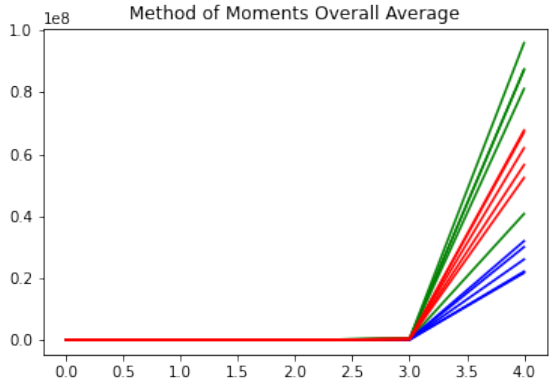
Například u parametru `SpectrumSpreadType` lze z výše uvedené tabulky vyčíst, že pro jednotlivé žánry nabývá průměr a standardní odchylka **podobné** hodnoty, zatímco mezi různými žánry jsou hodnoty víceméně **rozdílné**.

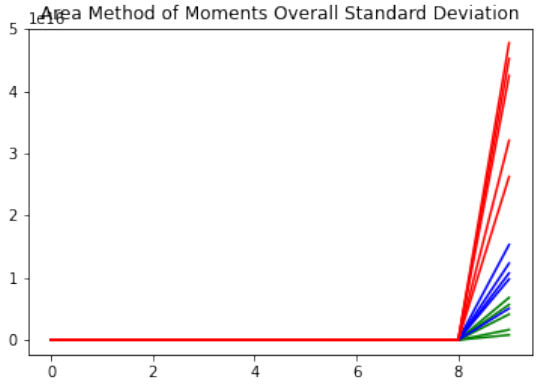
Z analýzy nám vyšlo, že z knihovny MPEG7 budou nejlépe použitelné `SpectrumSpreadType` a `SpectrumCentroidType` *(konkrétně jejich průměr)*, které popisují **rozsah** a **rozložení** frekvenčního spektra, což souhlasí s hypotézou o rozdílnosti rozsahu a rozložení frekvencí v jednotlivých žánrech.

jAudio

Knihovna jAudio umožňuje podobně jako MPEG7 knihovna extrakci jednotlivých MPEG7 deskriptorů, kromě nich umožňuje ale získání **mnoha dalších** deskriptorů.

Pro práci s knihovnou jAudio jsme použili sadu dalších 3 žánrů (opera, jazz a reggae), spustili jsme program pro získání všech deskriptorů a pomocí Jupiter notebooku, který naleznete ve složce `analysis/sound.ipynb` jsme vygenerovali grafy porovnávající jednotlivé žánry.





Graf průměru `Method of Moments` a standardních odchylek `Area Method of Moments`, červená je `reggae`, modrá `opera` a zelená `jazz`.

Z analýzy nám vyšlo, že z knihovny jAudio bude nejlépe použitelný průměr `Method of Moments` a standardní odchylka `Area Method of Moments`, což je víceúrovňový deskriptor, který sdružuje všechny MPEG7 deskriptory a počítá jejich hodnoty pomocí **momentové metody**.

MFCC

Kromě výše zmíněných deskriptorů jsme se rozhodli vyzkoušet také **MFCC** — Mel Frequency Cepstral koeficienty. Tato metoda umožňuje pomocí lineární kosinové transformace převést audio signál na sadu koeficientů, které lze následně použít pro **porovnání**.

MFCC dobře slouží při **rozpoznávání** mluvené řeči a také k rozpoznání hudebních signálů, což bude **užitečné** právě proto, že námi zvolené žánry se vždy skládají z různých hudebních nástrojů a v některých zpěv je, v některých není obsažen.

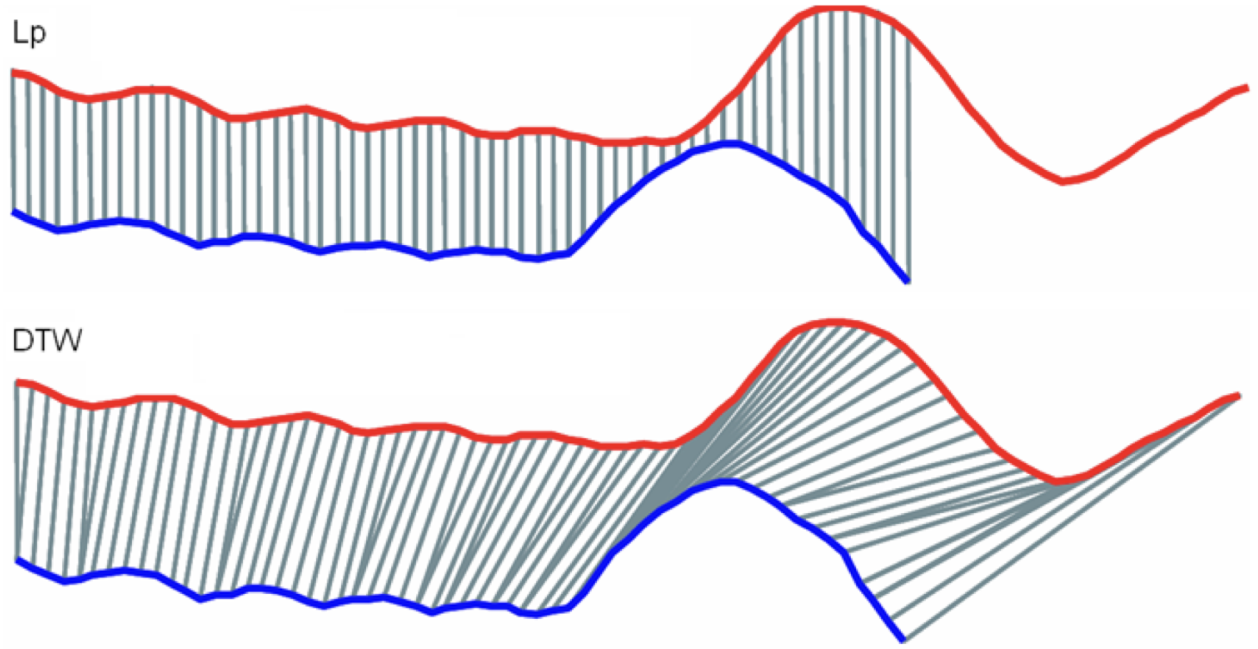
Z MFCC budeme používat 1. seznam **13 koeficientů** zprůměrovaných přes celou skladbu a za 2. **časovou řadu** jednoho vybraného koeficientu, a to implementaci v Pythonu.

Porovnání deskriptorů

Některé získané deskriptory jsou v podobě **časových řad**, kdy se za každý **interval** spočítá hodnota daného kritéria *(rozsah spektra, hodnota MFCC koeficientu)*, některé pouze v podobě **jednotlivých hodnot** *(průměr / standardní odchylka přes celou skladbu)*.

Porovnávání jednotlivých hodnot jsme se rozhodli řešit jednoduchým vzorcem `d(a,b) = |a - b|`. U porovnání časových řad jsme volili dva způsoby: prvním z nich je vypočtení **průměru** hodnot napříč celou řadou, druhým je pak porovnávání pomocí algoritmu DTW *(dynamic time warping)*.

Časové řady nelze porovnávat vedle sebe hodnotu po hodnotě, jelikož můžou být vůči sobě posunuté a navíc může být jedna časová řada delší než druhá. Pro porovnání se tedy používá algoritmus **DTW**, který "natáhne" jednu řadu tak, aby bylo porovnání možné. *(viz. obrázek, zdroj: přednáška 5, NI-VMM, Podobnostní vyhledávání — populární podobnostní funkce*)



Z našich konkrétních deskriptorů jsme se na základě analýzy grafů jednotlivých deskriptorů rozhodli deskriptory `SpectrumSpreadType`, `SpectrumCentroidType`, `Area Method of Moments` a `Method of Moments` rozhodli porovnávat jakožto průměr časové řady *(ad obrázek u podsekce MPEG7 — jednotlivé časové řady nejsou v rámci žánrů příliš podobné)* a deskriptor u koeficientu/koeficientů `MFCC` jsme se rozhodli vyzkoušet porovnávat jak průměrem, tak jakožto časovou řadu.

Klastrování

Ještě před samotným rozpoznáváním je třeba vybudovat "databázi" žánrů, na základě které se následně budou testované soubory porovnávat.

V našem případě jsme se rozhodli databázi realizovat jako jednorozměrné pole **features**, kde jsou v databázovém souboru `db.dat` za sebou binárně uloženy features (pole deskriptorů) jednotlivých skladeb v žánrech. Jedna položka v databázi tedy vypadá takto: `(spectrum_spread_avg, spectral_centroid_avg, area_of_moments_avg, area_of_moments_std, method_of_moments_avg, mfcc_avg, mfcc_time_series, genre_name)`.

Když chceme pro zadanou skladbu rozpoznat, kterému patří žánru, vypočteme pro ni hodnoty všech deskriptorů a porovnáme je s jednotlivými deskriptory v rámci žánrů, čímž v podstatě vypočteme **vzdálenost** od jednotlivých skladeb v rámci žánru. Tuto vzdálenost následně seřadíme od nejmenší po největší a vezmeme **k** skladeb (žánrů), které jsou nejblíže.

Tímto postupem v podstatě realizujeme algoritmus **kNN** (hledání k nejbližších sousedů). Nejbližší sousedy pak zařadíme do jednotlivých žánrů a vydělíme k, čímž nám vznikne **pravděpodobnost**, že zadaná skladba **patří** do daného žánru.

Implementace

Popis použitých programových prostředků, tj. jaké byly použity programovací jazyky, stavba aplikace, použité knihovny třetích stran, požadavky na běh ...

Architektura aplikace

Aplikace je napsána v programovacím jazyce Python a skládá se z následujících modulů a skriptů *(souborů ve složce* `program/`*)*, které budou podrobně popsány v podkapitolách:

- `loader`, který obsahuje funkce pro načtení deskriptorů ze zadaného souboru
- `helper`, který obsahuje funkce pro spočítání vzdálenosti dvou časových řad pomocí DTW `distance_dtw`, spočítání vzdálenosti mezi dvěma položkami v databázi `distance`, nalezení nejbližších k sousedů `getNeighbours`, nalezení nejbližších žánrů `nearestClass` a načtení databáze `loadDataset`
- skript `script.py`, který vygeneruje databázi a následně otestuje přesnost porovnávání s parametry, které jsou zadány na začátku souboru
- skript `recognise.py`, který pro zadaný soubor a parametry vypíše nejpravděpodobnější žánry
- Streamlit soubor `main.py` a webové stránky `pages/`, které definují webový modul pro knihovnu Streamlit (https://streamlit.io)

Načítání (loader.py)

Modul loader obsahuje funkce pro načítání jednotlivých deskriptorů. Existují dva hlavní zdroje deskriptorů, ze kterých načítáme.

Prvním zdrojem jsou deskriptory z knihoven **MPEG7** a **jAudio**. Pro získání těchto deskriptorů musíme nejprve spustit v příkazové řádce danou knihovnu:

```
mpeg = subprocess.run(["java", "-jar", "mpeg.jar", file], stdout=subprocess.PIPE,
stderr=subprocess.PIPE)
```

a následně s pomocí knihovny `xml.etree.ElementTree` a `numpy` naparsovat hledané deskriptory:

```
centroid = root.find(f
'.//{xmlns}AudioDescriptor[@{xsi}type="AudioSpectrumCentroidType"]/{xmlns}SeriesOfScal
ar/{xmlns}Raw')
# ...
centroid_values = np.fromstring(centroid.text, sep=" ")
# ...
return np.average(centroid_values), # ...
```

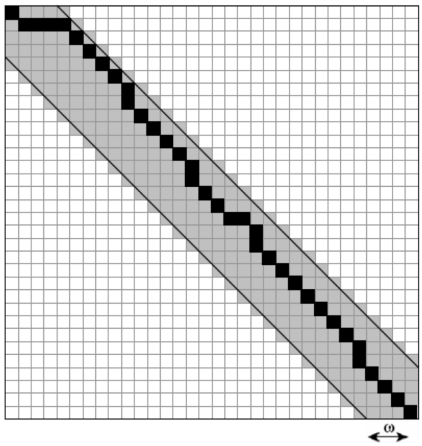
Druhým zdrojem jsou pak MFCC koeficienty, které získáváme přímo pomocí Python knihovny `python_speech_features`:

```
def loadMfccCoefficients(file):
    (rate, sig) = wav.read(file)
    mfcc_features = mfcc(sig, rate, winlen=0.020, appendEnergy=False, nfft=1024)
    return mfcc_features.mean(0).flatten(), mfcc_features[:, 0][:256]
```

Při načítání MFCC koeficientů spočítáme **průměr** přes všech 13 koeficientů a následně **prvních 256** hodnot časové řady **prvního** deskriptoru.

Pomocné funkce (helper.py)

Funkce `distance_dtw` počítá vzdálenost pomocí algoritmu Dynamic Time Warping. Jelikož by ale takový výpočet trval pro velké časové řady **příliš dlouho**, je zde implementována metoda **ořezu** prohledávaného prostoru pomocí parametru `dtw_width`.



Ukázka ořezu prohledávaného prostoru algoritmem DTW při použití parametru `dtw_width`.

Jaký **vliv** má konkrétní hodnota tohoto parametru na **rychlost** a **přesnost** výpočtu vzdálenosti bude diskutováno v Experimentální sekci.

Funkce `distance` slouží pro vypočítání vzdálenosti mezi **dvěma** sadami deskriptorů. Funkce pro každý deskriptor pomocí DTW spočítá vzdálenost a se zadanou **váhou** (parametrem `weights`) ji přičte k cílové vzdálenosti.

```
for descriptor_index in range(instance1_length - 1):
    if weights[descriptor_index] == 0:
        continue # skip (better performance)
    final_distance += weights[descriptor_index] * distance_dtw(
        instance1[descriptor_index],
        instance2[descriptor_index],
        dtw_width
    )
```

K povšimnutí jsou dvě věci: **první** je, že se při váze 0 výpočet přeskočí, aby se **ušetřil** čas. Druhou věcí je pak to, že se DTW používá k výpočtu vzdálenosti **všeho** (i jednotlivých hodnot — průměrů a standardních odchylek), nejen časových řad.

Tento výpočet ale bude fungovat správně, jelikož je DTW zavoláno na dvě pole velikosti jedna, nezávisle na parametru `dtw_width` se tedy vždy vrátí prosté porovnání a spočítání vzdáleností.

```
def getFeatures(file):
    # MPEG7 descriptors
    centroid_average, spread_average = ld.loadMpegDescriptors(file)
    spectrum_spread = [200 * spread_average]
    spectral_centroid = [100 * centroid_average]
    # ...
    return [spectrum_spread, spectral_centroid, ...]
```

Ve funkci `getFeatures` pro načtení jednotlivých deskriptorů do "databázového záznamu" je použito přeškálování — škálovací koeficienty jsme zvolili tak, že jsme spočítali **průměrnou** hodnotu pro **všechny** skladby v naší databázi a **přeškálovali** jsme je podle hodnot v MFCC.

Funkce `getNeighbours` spočítá všechny **vzdálenosti** k prvkům v zadaném datasetu, seřadí je a vrátí **prvních k** záznamů z databáze s **nejmenší** vzdáleností.

Funkce `nearestClass` pak pro zadané pole databázových **přepočítá** výskyty jednotlivých žánrů tak, aby bylo **výsledkem** pole pravděpodobností **příslušnosti** do daných žánrů (se součtem 1).

Sestavení databáze (script.py)

Pro sestavení databáze deskriptorů načteme hodnoty **všech** deskriptorů pro **všechny** skladby v naší databázi a uložíme je do databázového souboru `db.dat`:

```
DATABASE_FILE = "db.dat"
AUDIO_PATH = "../audio/"

for genre in os.listdir(AUDIO_PATH):
    for audioFile in os.listdir(AUDIO_PATH + genre):
        feature = hp.getFeatures(audioFilePath)
        feature.append(genre)
        pickle.dump(feature, file)
```

Testování přesnosti (script.py)

Součástí našeho projektu je také skript pro **testování přesnosti**. Při testování jsou skladby náhodně rozděleny s danou pravděpodobností do **testovací** sady a **trénovací** sady.

Skript následně pro každou z testovacích skladeb vypočítá její příslušnost k skladbám ze zadanému žánru, vypíše **očekávaný** žánr a **předpovězený** žánr a na konci vypíše také pravděpodobnost (v %), v kolika případech se předpověď **podařila**.

Skript umožňuje nastavit jednotlivé váhy deskriptorů `WEIGHTS`, šířku ořezu u DTW `DTW_WIDTH` a kolik nejbližších sousedů hledáme `TEST_K_NEAREST`.

```
loadDataset(training_set, test_set)
# ...
for test_idx in range(test_length):
    predictions.append(hp.nearestClass(hp.getNeighbours(training_set, test_set
[test_idx], WEIGHTS, DTW_WIDTH, TEST_K_NEAREST)))
# ...
return 100.0 * correct_tries / test_length
```

Rozpoznání žánru (recognise.py)

Poslední ze skriptů provede **rozpoznání** žánru zadané skladby se zadanými parametry *(váhy deskriptorů, šířka ořezu DTW, kolik nejbližších sousedů)*.

```
def recognise(file, k_nearest, dtw_width, descriptor_weights):
    feature = hp.getFeatures(file)
    feature.append('') # doesn't matter
    nearest_class = hp.nearestClass(hp.getNeighbours(
        dataset, feature, descriptor_weights, dtw_width, k_nearest
    ))
    return nearest_class
```

Webové rozhraní

Aplikace obsahuje kromě skriptů také **webové rozhraní** vytvořené pomocí knihovny **Streamlit** (https://streamlit.io). Webové rozhraní umožňuje na úvodní stránce nahrát skladbu, jejíž žánr chceme **rozpoznat** a na podstránkách **sestavit databázi** nebo interaktivně **testovat přesnost**.

Postranní menu

Součástí uživatelského rozhraní úvodní stránky a stránky pro testování přesnosti je **postranní menu**, které umožňuje nastavit hodnoty deskriptorů. To je definováno v souboru `st_sidebar.py`:

```
def sidebar(dataset):
    if 'spread_weight' not in st.session_state:
        st.session_state.spread_weight = 0.0
    # ...
    st.sidebar.header("Weights")
    st.sidebar.slider("Spectrum spread", key="spread_weight")
    # ...
```

Jednotlivé hodnoty jsou **nastavovány** s pomocí komponenty `st.slidebar.slider` a jsou uloženy v HTTP session.

Úvodní stránka

Při **prvním** načtení úvodní stránky se provede načtení databáze ze souboru `db.dat`, přičemž je použita komponenta `st.spinner`:

```
def loadDatabase():
    return hp.loadDataset('db.dat')

with st.spinner(text="Loading database..."):
    dataset = loadDatabase()
```

Následně se v sidebaru zobrazí s pomocí `st.sidebar.file_uploader` pro nahrání skladby, pro kterou chceme **rozpoznat** žánr:

```
uploaded_file = st.sidebar.file_uploader("Upload .wav file", type="wav", help="Upload
file you want to classify")
```

Po nahrání souboru se následně nahraný soubor **zkontroluje** (`checkUploadedFile`), získáme z něj **deskriptory** a spočítáme vzdálenost pomocí funkcí ze skriptu `helper`. Výsledek pak vykreslíme do koláčového grafu pomocí `matplotlib.pyplot` a zobrazíme přes `st.pyplot`:

```
with st.spinner(text="Processing uploaded file"):
    if 'features' in st.session_state:
        result = hp.nearestClass(hp.getNeighbours(...))
        # ...
        fig, ax = plt.subplots()
        ax.pie(values, explode=explode, labels=labels, autopct='%1.f%%')
        st.pyplot(fig)
    else:
        st.info('Upload a WAV file first', icon="")
```

Sestavení databáze

Stránka pro **sestavení databáze** pouze sestaví databázi, o čemž průběžně informuje uživatele pomocí `st.spinner` a `st.success`:

```
st.title("Create database")

with st.spinner("Building database..."):
    script.buildDatabase()

st.success("Database built")
```

Testování přesnosti

Stránka pro **testování přesnosti** si stejně jako `script.py` načte dataset, náhodně ho rozdělí na **trénovací** a **testovací** a vypočítá přesnost, kromě toho ale také zobrazí **tabulku** s tím, který žánr byl predikován s jakou pravděpodobností a **čas běhu** aplikace.

```
# ...
st.write(f"Accuracy {(100.0 * correct_tries / test_length):.2f}%")
st.write(f"Elapsed {(time.perf_counter() - started):.2f}s")

table = [
    {
        "Genre": st.session_state.test_set[i][-1],
        "Classified as": predictions[i][0][0],
        "Probability": f"{(predictions[i][0][1] * 100.0):.0f}%"
    } for i in range(len(st.session_state.test_set))
]
st.table(table)
```

Zobrazení výsledku testování s pomocí `st.write` a `st.table`.

Stránka umožňuje také, stejně jako úvodní stránka v postranním menu **nastavení parametrů** jako počet nejbližších sousedů a váhy jednotlivých deskriptorů. Aby nedocházelo k rozdělení datasetu při každé změně parametru, pamatujeme si prvotní náhodné rozdělení v `st.session`:

```
with st.spinner():
    if 'training_set' not in st.session_state:
        st.session_state.training_set = []

    if 'test_set' not in st.session_state:
        st.session_state.test_set = []

    if len(st.session_state.training_set) == 0:
        script.loadDataset(st.session_state.training_set, st.session_state.test_set)
```

Příklad výstupu

Tato kapitola obsahuje ukázku konkrétních vstupů a výstupů skriptů a snímky webového rozhraní aplikace.

Skripty

Jak jsme již zmínili v předchozích kapitolách, jednou ze součástí aplikace jsou skripty `script.py` pro sestavení databáze žánrů a otestování přesnosti a `recognise.py` pro rozpoznání žánru zadané skladby.

```
> python3 script.py
< Building database...
(...)
< Calculating precision...
< Test set:     ['metal', 'metal', 'classic', 'classic', 'reggae', 'reggae', 'reggae',
'opera', 'opera', 'rap', 'rap', 'rap', 'jazz', 'jazz']
< Predictions:  ['metal', 'metal', 'opera',   'opera', 'jazz', 'reggae', 'rap',
'opera', 'opera', 'reggae', 'rap', 'opera', 'jazz', 'jazz']
< Precision: 57.142857142857146 %
```

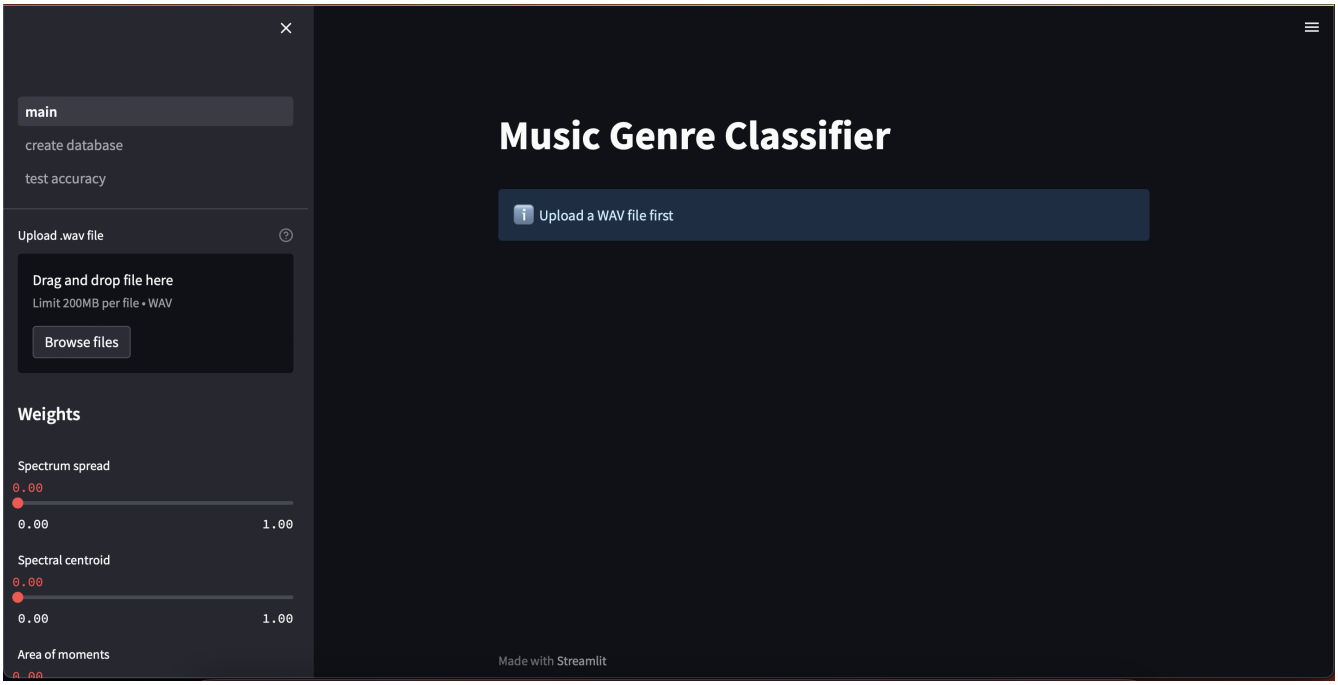
Ukázka spuštění skriptu `script.py` s parametry `TEST_K_NEAREST = 5`, `DTW_WIDTH = 20` a `WEIGHTS = [1, 1, 1, 1, 1, 1, 1]`. Výsledkem je seznam předpovědí a přesnost modelu.

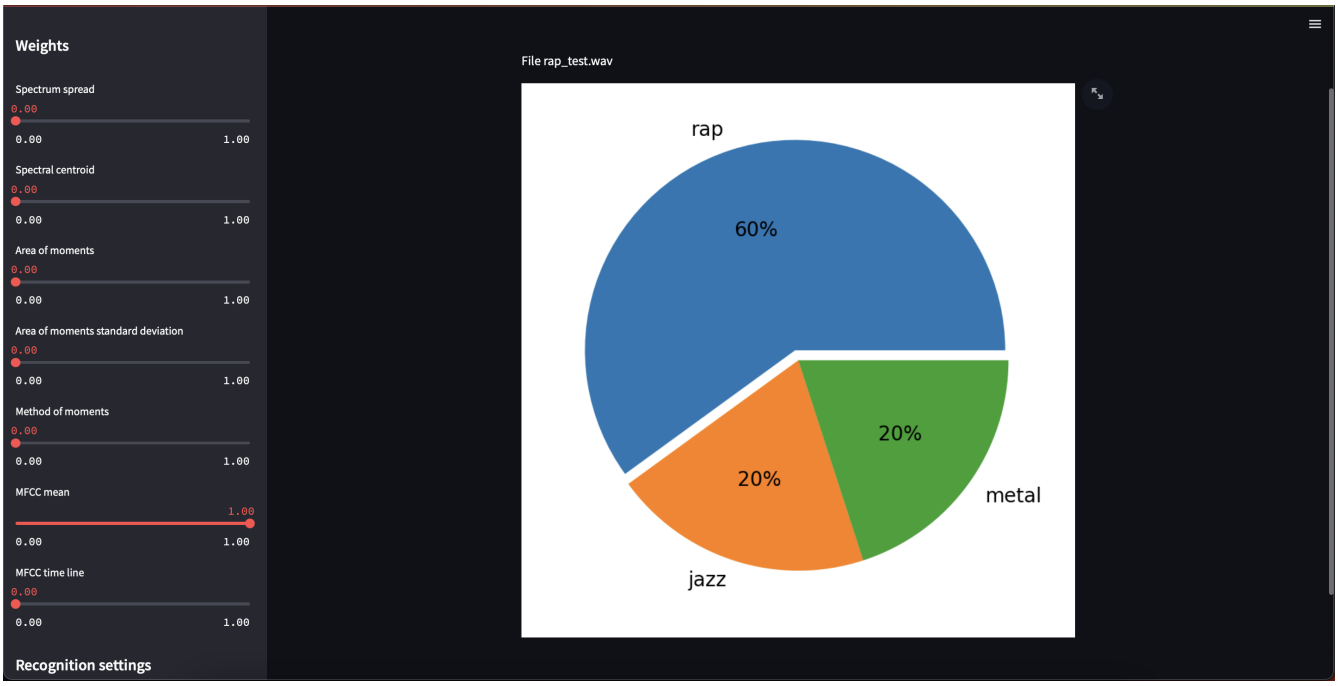
```
> time python3 recognise.py ../test/rap_test.wav 5 0 1 1 1 1 1 1 1
< Loading features for ../test/rap_test.wav
< [('jazz', 0.4), ('classic', 0.2), ('rap', 0.2), ('reggae', 0.2)]
< python3 recognise.py ../test/rap_test.wav 5 0 1 1 1 1 1 1 1  4.09s user 1.30s system
354% cpu 1.522 total

> time python3 recognise.py ../test/rap_test.wav 10 20 1 1 1 1 1 1 1
< Loading features for ../test/rap_test.wav
< [('jazz', 0.30000000000000004), ('rap', 0.2), ('classic', 0.2), ('reggae', 0.2),
('opera', 0.1)]
< python3 recognise.py ../test/rap_test.wav 10 20 1 1 1 1 1 1 1  5.41s user 1.20s
system 221% cpu 2.990 total
```

Ukázka spuštění skriptu `recognise.py` pro rozpoznání žánru souboru `rap_test.wav` s různými parametry (váhy koeficientů jsou voleny vždy jako 1, mění se `TEST_K_NEAREST` z 5 na 10 a `DTW_WIDTH` z 0 na 20). Výsledkem jsou pravděpodobnosti jednotlivých žánrů a čas běhu.

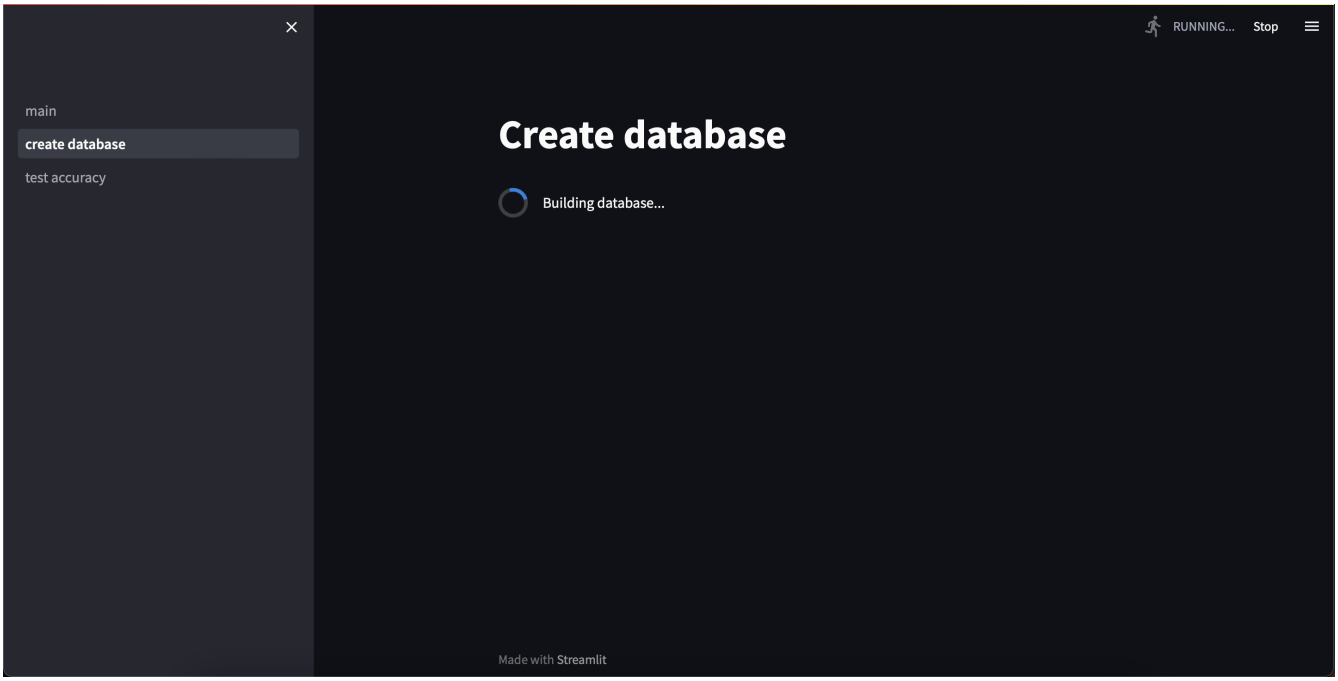
Webová aplikace

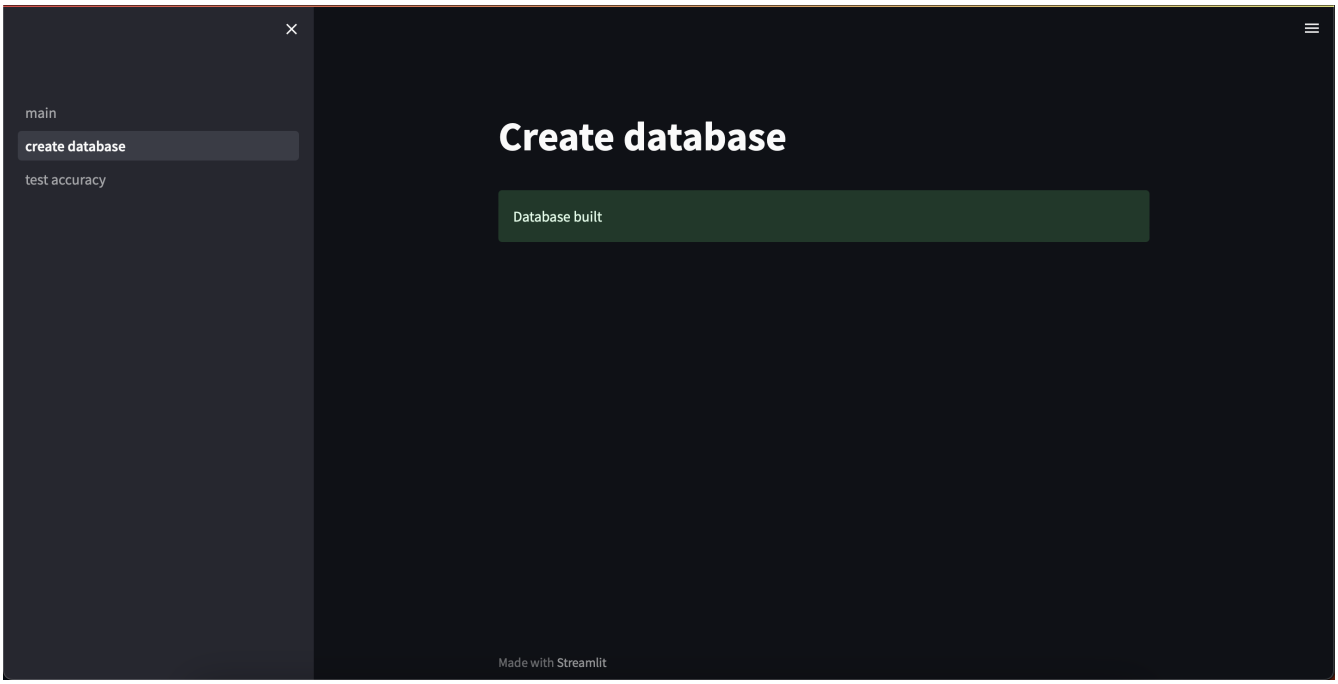




Úvodní stránka s **výzvou** k nahrání souboru *(vlevo)* a **po nahrání** souboru a změně parametrů *(vpravo)*.

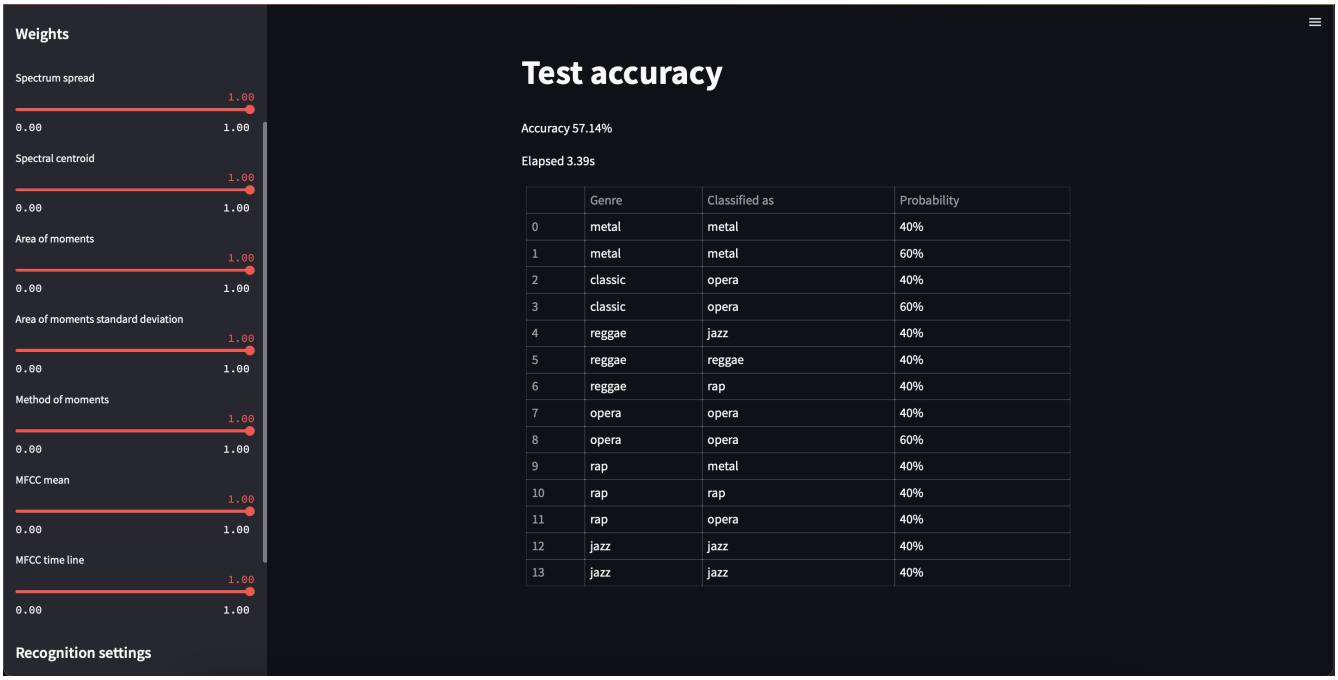
Sestavení databáze



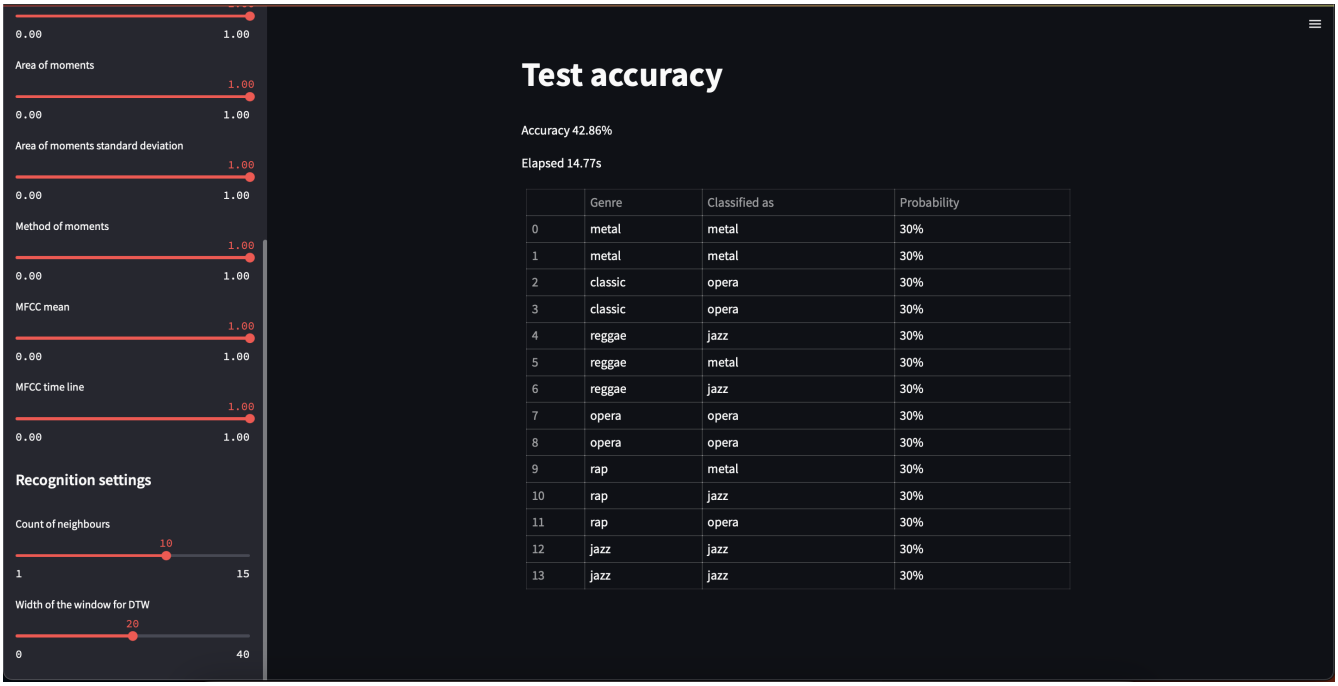


Stránka, která umožní **sestavit databázi** žánrů a jejich deskriptorů. Stav **při** sestavování *(vlevo)* a **po** sestavení *(vpravo)*.

Testování přesnosti



	Genre	Classified as	Probability
0	metal	metal	40%
1	metal	metal	60%
2	classic	opera	40%
3	classic	opera	60%
4	reggae	jazz	40%
5	reggae	reggae	40%
6	reggae	rap	40%
7	opera	opera	40%
8	opera	opera	60%
9	rap	metal	40%
10	rap	rap	40%
11	rap	opera	40%
12	jazz	jazz	40%
13	jazz	jazz	40%



	Genre	Classified as	Probability
0	metal	metal	30%
1	metal	metal	30%
2	classic	opera	30%
3	classic	opera	30%
4	reggae	jazz	30%
5	reggae	metal	30%
6	reggae	jazz	30%
7	opera	opera	30%
8	opera	opera	30%
9	rap	metal	30%
10	rap	jazz	30%
11	rap	opera	30%
12	jazz	jazz	30%
13	jazz	jazz	30%

Stránka umožňující **testování přesnosti** s různě nastavenými **parametry**.

Experimentální sekce

Obsahem této kapitoly je experimentální testování různých parametrů porovnávání deskriptorů v naší aplikaci a jejich vlivu na přesnost a/nebo rychlost.

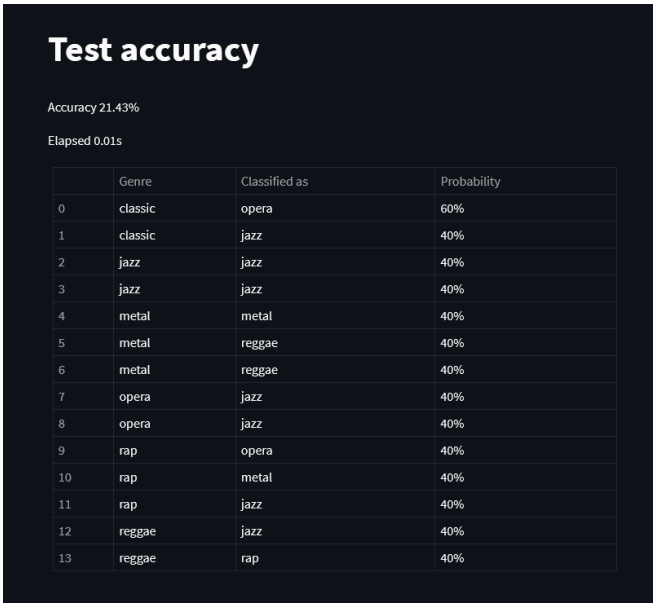
Deskriptory

Cílem experimentu bylo porovnání přesnosti jednotlivých deskriptorů, jak na testovací množině, tak i na testovacích souborech.

Testování přesnosti

Každý deskriptor byl vyhodnocován **zvlášť** *(měl nastavenou váhu na hodnotu 1, ostatní deskriptory měly nastavenou váhu na 0)*. Počet sousedů byl nastaven na hodnotu **5**. Níže je uvedena tabulka s **přesností** pro jednotlivé deskriptory. Dále jsou uvedené **snímky obrazovky** s podrobnými výsledky pro jednotlivé deskriptory.

Deskriptor	**Přesnost**
SpectrumSpreadType	21,43 %
SpectrumCentroidType	28,57 %
Area Method of Moments *(průměr)*	21,43 %
Area Method of Moments *(std. odchylka)*	21,43 %
Method of Moments *(průměr)*	50 %
MFCC *(průměr)*	71,43 %
MFCC *(časová řada)*	57,14 %

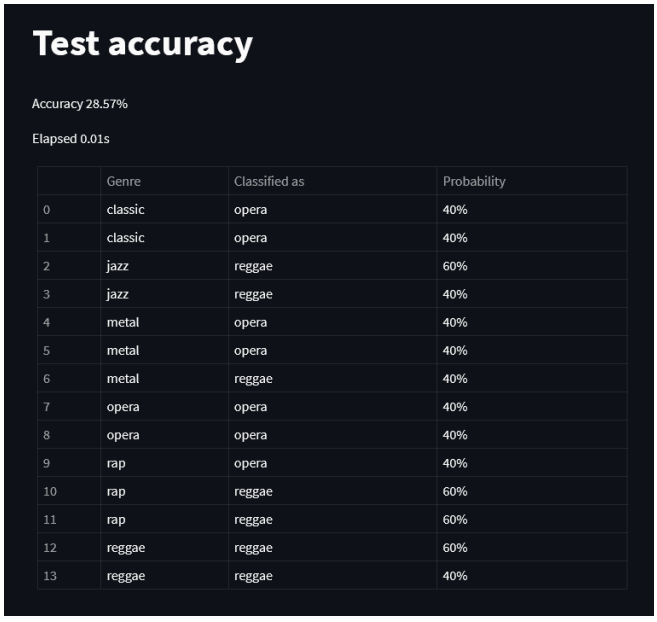


Test accuracy

Accuracy 21.43%

Elapsed 0.01s

	Genre	Classified as	Probability
0	classic	opera	60%
1	classic	jazz	40%
2	jazz	jazz	40%
3	jazz	jazz	40%
4	metal	metal	40%
5	metal	reggae	40%
6	metal	reggae	40%
7	opera	jazz	40%
8	opera	jazz	40%
9	rap	opera	40%
10	rap	metal	40%
11	rap	jazz	40%
12	reggae	jazz	40%
13	reggae	rap	40%

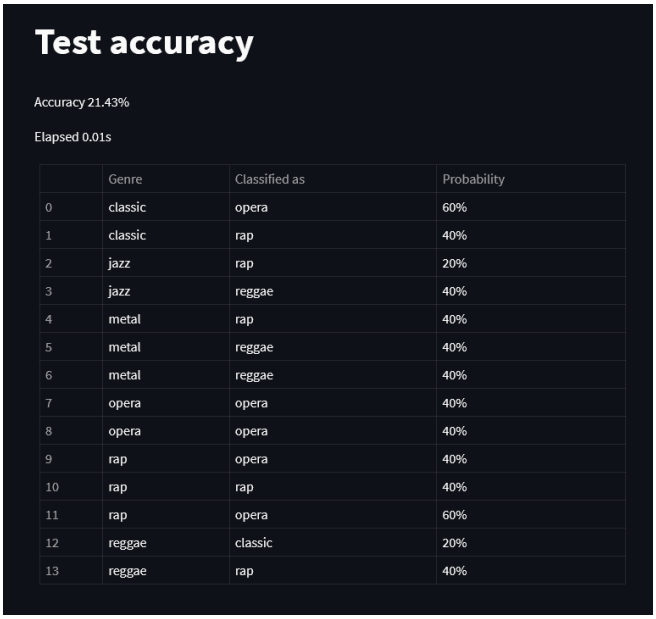


Test accuracy

Accuracy 28.57%

Elapsed 0.01s

	Genre	Classified as	Probability
0	classic	opera	40%
1	classic	opera	40%
2	jazz	reggae	60%
3	jazz	reggae	40%
4	metal	opera	40%
5	metal	opera	40%
6	metal	reggae	40%
7	opera	opera	40%
8	opera	opera	40%
9	rap	opera	40%
10	rap	reggae	60%
11	rap	reggae	60%
12	reggae	reggae	60%
13	reggae	reggae	40%

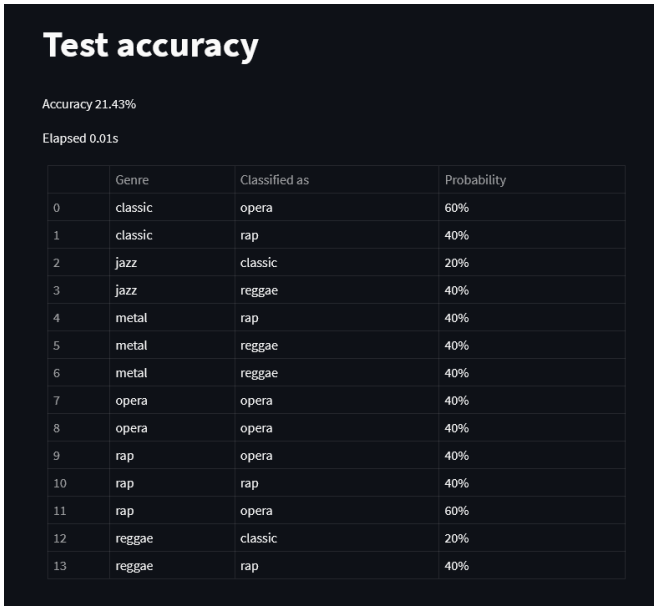


Test accuracy

Accuracy 21.43%

Elapsed 0.01s

	Genre	Classified as	Probability
0	classic	opera	60%
1	classic	rap	40%
2	jazz	rap	20%
3	jazz	reggae	40%
4	metal	rap	40%
5	metal	reggae	40%
6	metal	reggae	40%
7	opera	opera	40%
8	opera	opera	40%
9	rap	opera	40%
10	rap	rap	40%
11	rap	opera	60%
12	reggae	classic	20%
13	reggae	rap	40%

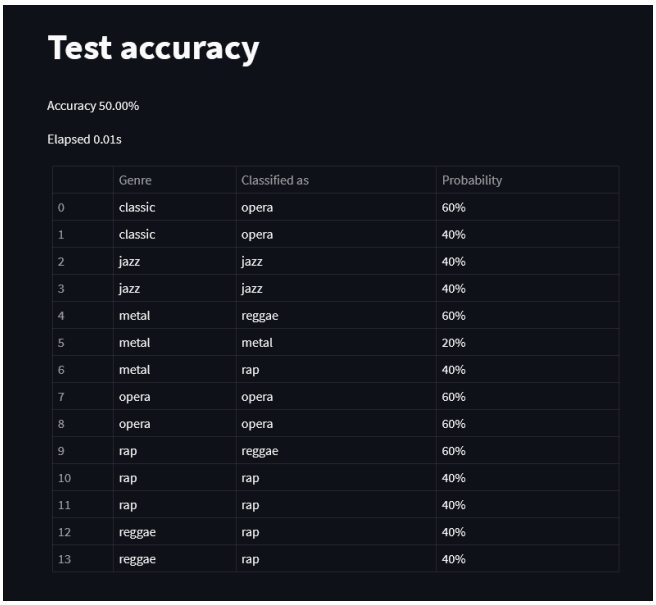


Test accuracy

Accuracy 21.43%

Elapsed 0.01s

	Genre	Classified as	Probability
0	classic	opera	60%
1	classic	rap	40%
2	jazz	classic	20%
3	jazz	reggae	40%
4	metal	rap	40%
5	metal	reggae	40%
6	metal	reggae	40%
7	opera	opera	40%
8	opera	opera	40%
9	rap	opera	40%
10	rap	rap	40%
11	rap	opera	60%
12	reggae	classic	20%
13	reggae	rap	40%

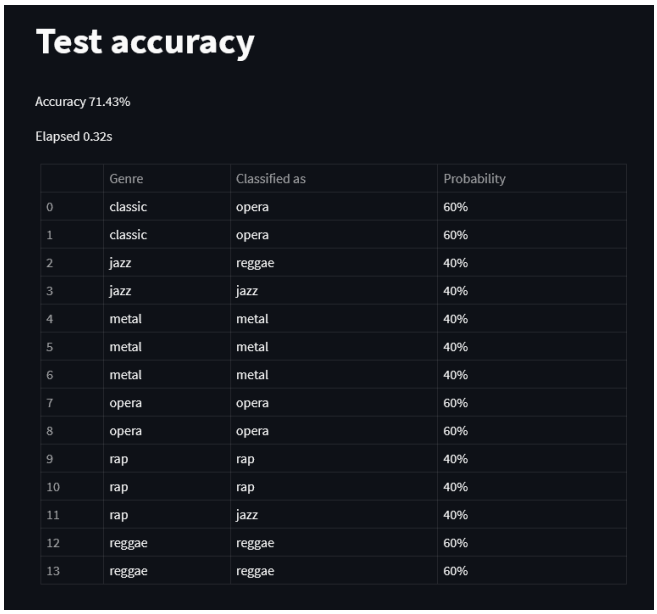


Test accuracy

Accuracy 50.00%

Elapsed 0.01s

	Genre	Classified as	Probability
0	classic	opera	60%
1	classic	opera	40%
2	jazz	jazz	40%
3	jazz	jazz	40%
4	metal	reggae	60%
5	metal	metal	20%
6	metal	rap	40%
7	opera	opera	60%
8	opera	opera	60%
9	rap	reggae	60%
10	rap	rap	40%
11	rap	rap	40%
12	reggae	rap	40%
13	reggae	rap	40%

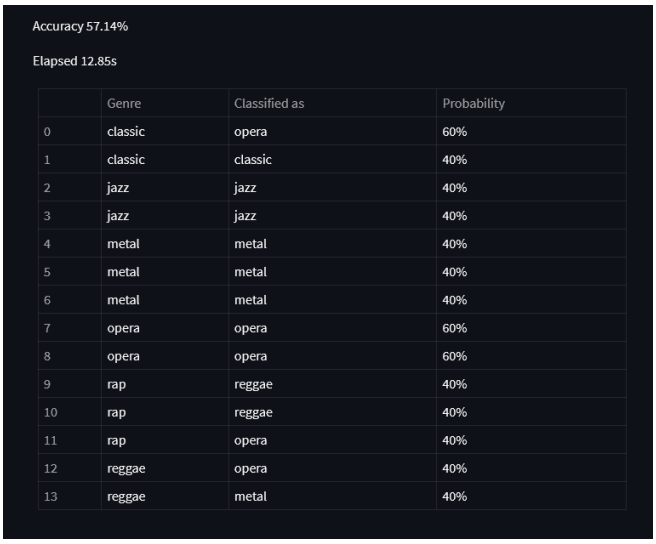


Test accuracy

Accuracy 71.43%

Elapsed 0.32s

	Genre	Classified as	Probability
0	classic	opera	60%
1	classic	opera	60%
2	jazz	reggae	40%
3	jazz	jazz	40%
4	metal	metal	40%
5	metal	metal	40%
6	metal	metal	40%
7	opera	opera	60%
8	opera	opera	60%
9	rap	rap	40%
10	rap	rap	40%
11	rap	jazz	40%
12	reggae	reggae	60%
13	reggae	reggae	60%



Accuracy 57.14%

Elapsed 12.85s

	Genre	Classified as	Probability
0	classic	opera	60%
1	classic	classic	40%
2	jazz	jazz	40%
3	jazz	jazz	40%
4	metal	metal	40%
5	metal	metal	40%
6	metal	metal	40%
7	opera	opera	60%
8	opera	opera	60%
9	rap	reggae	40%
10	rap	reggae	40%
11	rap	opera	40%
12	reggae	opera	40%
13	reggae	metal	40%

Výsledné tabulky testování přesnosti pro deskriptory *(zleva doprava, shora dolů)*: `SpectrumSpreadType`, `SpectrumCentroidType`, `Area Method of Moments` *(průměr)*, `Area Method of Moments` *(std. odchylka)*, `Method of Moments` *(průměr)*, `MFCC` *(průměr)*, `MFCC` *(časová řada)*.

Testování na jednotlivých souborech

Pro **každý** žánr jsme vybrali **jeden** soubor *(jiný od těch použitých pro sestavení databáze)*, na kterém jsme testovali **přesnost**. V tabulkách níže jsou v řádcích zapsány **skutečné** žánry testovacích souborů, ve sloupcích jsou vyobrazeny výsledné **pravděpodobnosti** daných žánrů na základě konkrétního **deskriptoru** a **vstupu**. Počet sousedů byl nastaven na hodnotu **5**.

Žánr	Pravděpodobnost klasické hudby	Pravděpodobnost jazzu	Pravděpodobnost metalu	Pravděpodobnost opery	Pravděpodobnost rapu	Pravděpodobnost reggae
Klasická hudba	**40 %**	20 %	0 %	40 %	0 %	0 %
Jazz	40 %	**20 %**	0 %	40 %	0 %	0 %
Metal	0 %	40 %	**0 %**	20 %	20 %	20 %
Opera	0 %	40 %	0 %	**20 %**	20 %	20 %
Rap	40 %	40 %	0 %	0 %	**0 %**	20 %
Reggae	40 %	40 %	0 %	0 %	0 %	**20 %**

Testování výstupů deskriptoru `SpectrumSpreadType`

Žánr	Pravděpodobnost klasické hudby	Pravděpodobnost jazzu	Pravděpodobnost metalu	Pravděpodobnost opery	Pravděpodobnost rapu	Pravděpodobnost reggae
Klasická hudba	**0 %**	40 %	40 %	0 %	20 %	0 %
Jazz	0 %	**0 %**	20 %	60 %	20 %	0 %
Metal	0 %	40 %	**40 %**	0 %	20 %	0 %
Opera	0 %	0 %	20 %	**40 %**	40 %	0 %
Rap	0 %	60 %	0 %	0 %	**20 %**	20 %
Reggae	0 %	20 %	40 %	0 %	0 %	**40 %**

Testování výstupů deskriptoru `SpectrumCentroidType`

Žánr	Pravděpodobnost klasické hudby	Pravděpodobnost jazzu	Pravděpodobnost metalu	Pravděpodobnost opery	Pravděpodobnost rapu	Pravděpodobnost reggae
Klasická hudba	**20 %**	20 %	20 %	20 %	20 %	0 %
Jazz	20 %	**20 %**	20 %	20 %	20 %	0 %
Metal	20 %	20 %	**20 %**	20 %	20 %	0 %
Opera	20 %	20 %	20 %	**20 %**	20 %	0 %
Rap	20 %	20 %	20 %	20 %	**20 %**	0 %
Reggae	20 %	0 %	20 %	0 %	40 %	**20 %**

Testování výstupů deskriptoru `Area Method of Moments` *(průměr)*

Žánr	Pravděpodobnost klasické hudby	Pravděpodobnost jazzu	Pravděpodobnost metalu	Pravděpodobnost opery	Pravděpodobnost rapu	Pravděpodobnost reggae
Klasická hudba	**20 %**	20 %	20 %	20 %	20 %	0 %
Jazz	20 %	**20 %**	20 %	20 %	20 %	0 %
Metal	20 %	20 %	**20 %**	20 %	20 %	0 %
Opera	20 %	20 %	20 %	**20 %**	20 %	0 %
Rap	20 %	20 %	20 %	20 %	**20 %**	0 %
Reggae	20 %	0 %	20 %	0 %	40 %	**20 %**

Testování výstupů deskriptoru `Area Method of Moments` *(std. odchylka)*

Žánr	Pravděpodobnost klasické hudby	Pravděpodobnost jazzu	Pravděpodobnost metalu	Pravděpodobnost opery	Pravděpodobnost rapu	Pravděpodobnost reggae
Klasická hudba	**40 %**	0 %	0 %	60 %	0 %	0 %
Jazz	40 %	**0 %**	0 %	60 %	0 %	0 %
Metal	0 %	0 %	**60 %**	0 %	20 %	20 %
Opera	0 %	0 %	0 %	**100 %**	0 %	0 %
Rap	0 %	80 %	20 %	0 %	**0 %**	0 %
Reggae	0 %	0 %	40 %	0 %	20 %	**40 %**

Testování výstupů deskriptoru `Method of Moments` *(průměr)*

Žánr	Pravděpodobnost klasické hudby	Pravděpodobnost jazzu	Pravděpodobnost metalu	Pravděpodobnost opery	Pravděpodobnost rapu	Pravděpodobnost reggae
Klasická hudba	**40 %**	0 %	0 %	40 %	0 %	20 %
Jazz	0 %	**20 %**	0 %	60 %	20 %	0 %
Metal	20 %	0 %	**20 %**	0 %	40 %	20 %
Opera	40 %	0 %	0 %	**40 %**	20 %	0 %
Rap	0 %	40 %	0 %	0 %	**20 %**	40 %
Reggae	0 %	0 %	40 %	0 %	40 %	**20 %**

Testování výstupů deskriptoru `MFCC` *(průměr)*

Žánr	Pravděpodobnost klasické hudby	Pravděpodobnost jazzu	Pravděpodobnost metalu	Pravděpodobnost opery	Pravděpodobnost rapu	Pravděpodobnost reggae
Klasická hudba	**40 %**	0 %	0 %	60 %	0 %	0 %
Jazz	40 %	**0 %**	0 %	60 %	0 %	0 %
Metal	0 %	20 %	**40 %**	0 %	20 %	20 %
Opera	40 %	0 %	0 %	**60 %**	0 %	0 %
Rap	0 %	20 %	40 %	0 %	**20 %**	20 %
Reggae	0 %	0 %	20 %	0 %	20 %	**60 %**

Testování výstupů deskriptoru `MFCC` *(časová řada)*

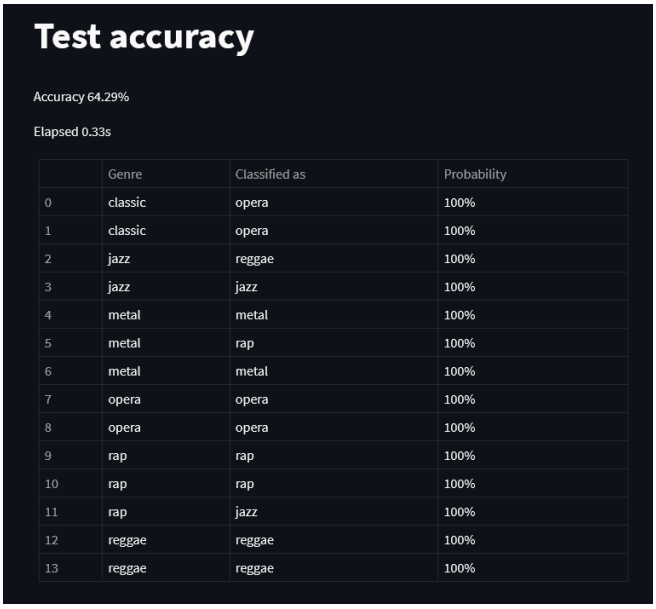
Porovnání parametrů

Cílem experimentu je porovnání parametrů počet sousedů a šířka DTW okna pro příslušné deskriptory.

Počet sousedů pro deskriptor MFCC průměr

Níže jsou výsledky testování **přesnosti** pro deskriptor `MFCC` *(průměr)* pro 1, 3, 5 a 10 **sousedů**.

Počet sousedů	Přesnost
1	64,29 %
3	71,43 %
5	71,43 %
10	35,71 %

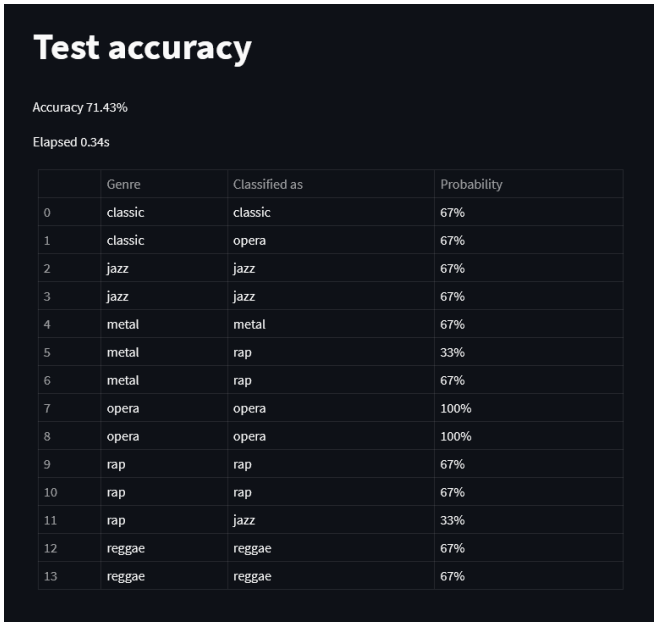


Test accuracy

Accuracy 64.29%

Elapsed 0.33s

	Genre	Classified as	Probability
0	classic	opera	100%
1	classic	opera	100%
2	jazz	reggae	100%
3	jazz	jazz	100%
4	metal	metal	100%
5	metal	rap	100%
6	metal	metal	100%
7	opera	opera	100%
8	opera	opera	100%
9	rap	rap	100%
10	rap	rap	100%
11	rap	jazz	100%
12	reggae	reggae	100%
13	reggae	reggae	100%

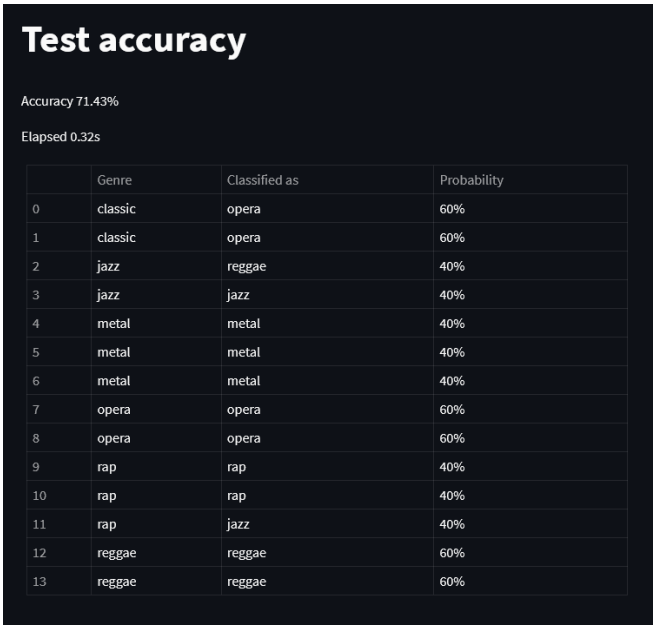


Test accuracy

Accuracy 71.43%

Elapsed 0.34s

	Genre	Classified as	Probability
0	classic	classic	67%
1	classic	opera	67%
2	jazz	jazz	67%
3	jazz	jazz	67%
4	metal	metal	67%
5	metal	rap	33%
6	metal	rap	67%
7	opera	opera	100%
8	opera	opera	100%
9	rap	rap	67%
10	rap	rap	67%
11	rap	jazz	33%
12	reggae	reggae	67%
13	reggae	reggae	67%

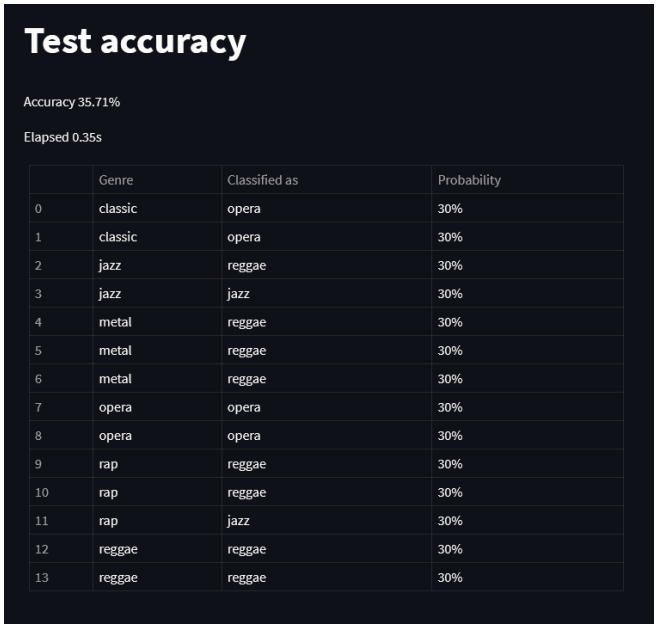


Test accuracy

Accuracy 71.43%

Elapsed 0.32s

	Genre	Classified as	Probability
0	classic	opera	60%
1	classic	opera	60%
2	jazz	reggae	40%
3	jazz	jazz	40%
4	metal	metal	40%
5	metal	metal	40%
6	metal	metal	40%
7	opera	opera	60%
8	opera	opera	60%
9	rap	rap	40%
10	rap	rap	40%
11	rap	jazz	40%
12	reggae	reggae	60%
13	reggae	reggae	60%



Test accuracy

Accuracy 35.71%

Elapsed 0.35s

	Genre	Classified as	Probability
0	classic	opera	30%
1	classic	opera	30%
2	jazz	reggae	30%
3	jazz	jazz	30%
4	metal	reggae	30%
5	metal	reggae	30%
6	metal	reggae	30%
7	opera	opera	30%
8	opera	opera	30%
9	rap	reggae	30%
10	rap	reggae	30%
11	rap	jazz	30%
12	reggae	reggae	30%
13	reggae	reggae	30%

V tabulkách níže jsou **výstupy** pro jednotlivé testovací **soubory**, opět postupně pro počet sousedů 1, 3, 5 a 10.

Žánr	**Pravděpodobnost klasické hudby**	**Pravděpodobnost jazzu**	**Pravděpodobnost metalu**	**Pravděpodobnost opery**	**Pravděpodobnost rapu**	**Pravděpodobnost reggae**
Klasická hudba	**100 %**	0 %	0 %	0 %	0 %	0 %
Jazz	0 %	**0 %**	0 %	100 %	0 %	0 %
Metal	0 %	0 %	**100 %**	0 %	0 %	0 %
Opera	0 %	0 %	0 %	**100 %**	0 %	0 %
Rap	0 %	0 %	0 %	0 %	**100 %**	0 %
Reggae	0 %	0 %	0 %	0 %	100 %	**0 %**

Žánr	**Pravděpodobnost klasické hudby**	**Pravděpodobnost jazzu**	**Pravděpodobnost metalu**	**Pravděpodobnost opery**	**Pravděpodobnost rapu**	**Pravděpodobnost reggae**
Klasická hudba	**67 %**	0 %	0 %	33 %	0 %	0 %
Jazz	0 %	**33 %**	0 %	67 %	0 %	0 %
Metal	33 %	0 %	**33 %**	0 %	33 %	0 %
Opera	33 %	0 %	0 %	**67 %**	0 %	0 %
Rap	0 %	33 %	33 %	0 %	**33 %**	0 %
Reggae	0 %	0 %	67 %	0 %	33 %	**0 %**

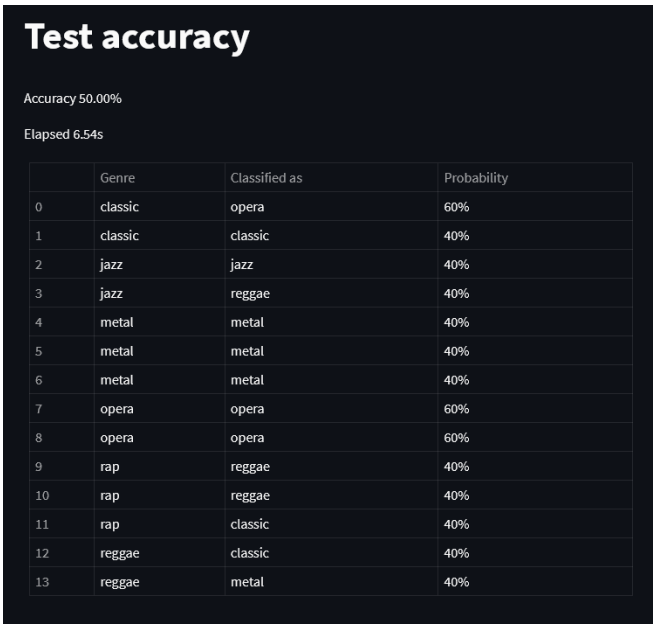
Žánr	**Pravděpodobnost klasické hudby**	**Pravděpodobnost jazzu**	**Pravděpodobnost metalu**	**Pravděpodobnost opery**	**Pravděpodobnost rapu**	**Pravděpodobnost reggae**
Klasická hudba	**40 %**	40 %	0 %	0 %	0 %	20 %
Jazz	0 %	**20 %**	0 %	60 %	20 %	0 %
Metal	20 %	0 %	**20 %**	0 %	40 %	20 %
Opera	40 %	0 %	0 %	**40 %**	20 %	0 %
Rap	0 %	20 %	20 %	0 %	**60 %**	0 %
Reggae	0 %	0 %	40 %	0 %	40 %	**20 %**

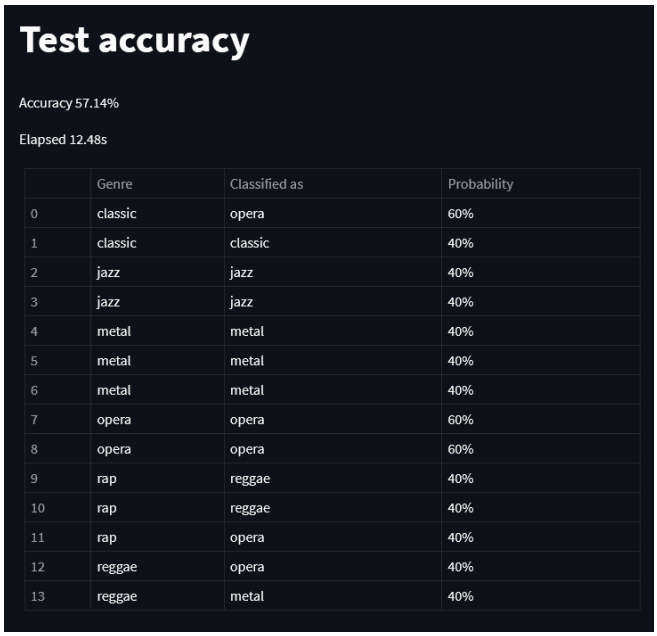
Žánr	Pravděpodobnost klasické hudby	Pravděpodobnost jazzu	Pravděpodobnost metalu	Pravděpodobnost opery	Pravděpodobnost rapu	Pravděpodobnost reggae
Klasická hudba	**30 %**	10 %	0 %	40 %	10 %	10 %
Jazz	20 %	**10 %**	0 %	50 %	10 %	10 %
Metal	10 %	0 %	**20 %**	30 %	30 %	10 %
Opera	30 %	10 %	10 %	**30 %**	20 %	0 %
Rap	0 %	20 %	20 %	0 %	**40 %**	20 %
Reggae	0 %	10 %	30 %	0 %	30 %	**30 %**

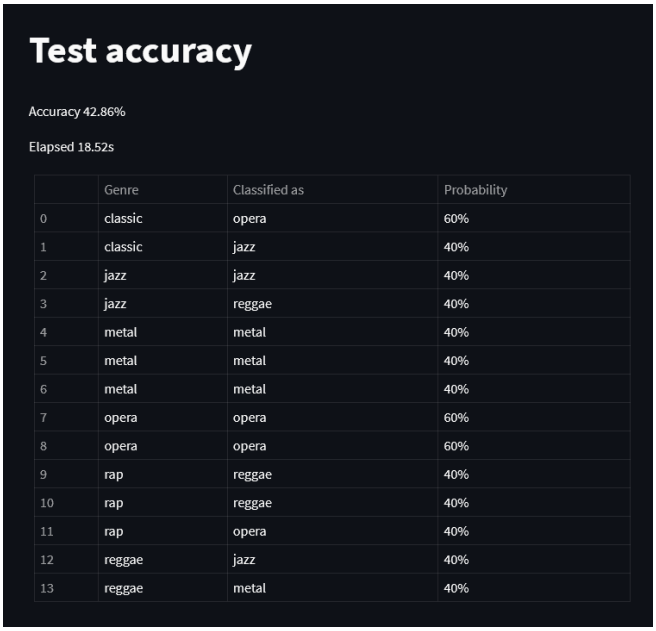
Šířka okna DTW pro deskriptor MFCC časová řada

Níže jsou uvedeny výsledky testování **přesnosti** a **doby běhu** pro deskriptor `MFCC` *(časová řada)* pro šířku okna **DTW** 0, 5, 10, 20 a 40.

Počet sousedů	Přesnost	Doba běhu
0	50 %	6,54 s
5	57,14 %	12,48 s
10	42,86 %	18,52 s
20	35,71 %	29,7 s
40	35,71 %	51,72 s





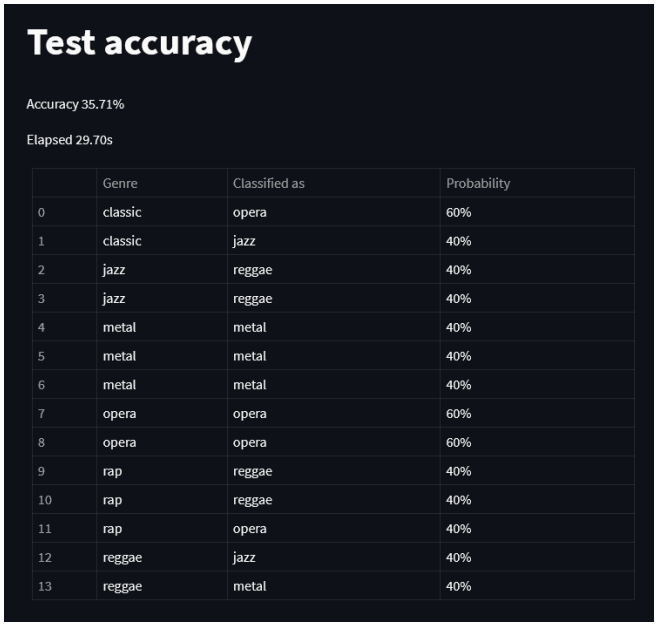


Test accuracy

Accuracy 42.86%

Elapsed 18.52s

	Genre	Classified as	Probability
0	classic	opera	60%
1	classic	jazz	40%
2	jazz	jazz	40%
3	jazz	reggae	40%
4	metal	metal	40%
5	metal	metal	40%
6	metal	metal	40%
7	opera	opera	60%
8	opera	opera	60%
9	rap	reggae	40%
10	rap	reggae	40%
11	rap	opera	40%
12	reggae	jazz	40%
13	reggae	metal	40%

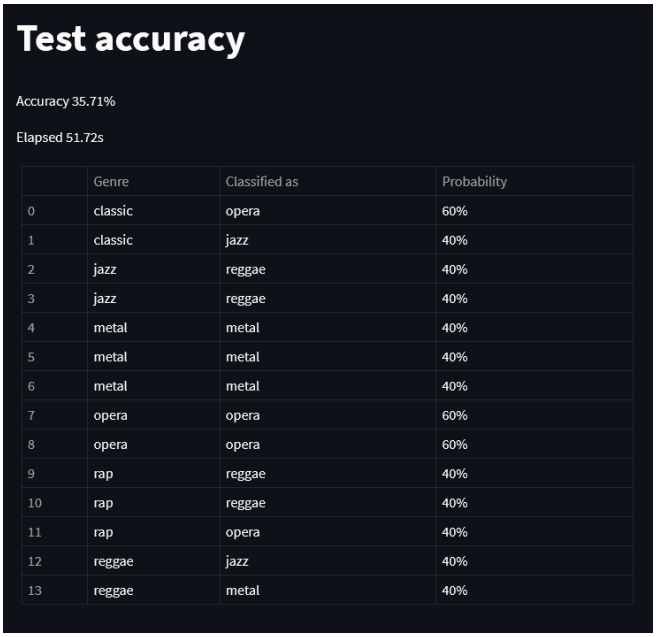


Test accuracy

Accuracy 35.71%

Elapsed 29.70s

	Genre	Classified as	Probability
0	classic	opera	60%
1	classic	jazz	40%
2	jazz	reggae	40%
3	jazz	reggae	40%
4	metal	metal	40%
5	metal	metal	40%
6	metal	metal	40%
7	opera	opera	60%
8	opera	opera	60%
9	rap	reggae	40%
10	rap	reggae	40%
11	rap	opera	40%
12	reggae	jazz	40%
13	reggae	metal	40%



Test accuracy

Accuracy 35.71%

Elapsed 51.72s

	Genre	Classified as	Probability
0	classic	opera	60%
1	classic	jazz	40%
2	jazz	reggae	40%
3	jazz	reggae	40%
4	metal	metal	40%
5	metal	metal	40%
6	metal	metal	40%
7	opera	opera	60%
8	opera	opera	60%
9	rap	reggae	40%
10	rap	reggae	40%
11	rap	opera	40%
12	reggae	jazz	40%
13	reggae	metal	40%

V tabulkách níže jsou výstupy pro **jednotlivé** testovací soubory, rozdělené dle velikosti DTW okna, opět popořadě 0, 5, 10, 20 a 40.

Žánr	Pravděpodobnost klasické hudby	Pravděpodobnost jazzu	Pravděpodobnost metalu	Pravděpodobnost opery	Pravděpodobnost rapu	Pravděpodobnost reggae
Klasická hudba	**40 %**	0 %	0 %	60 %	0 %	0 %
Jazz	60 %	**0 %**	0 %	40 %	0 %	0 %
Metal	0 %	20 %	**20 %**	0 %	20 %	40 %
Opera	40 %	0 %	0 %	**60 %**	0 %	0 %
Rap	40 %	40 %	0 %	0 %	**20 %**	0 %
Reggae	0 %	0 %	40 %	0 %	20 %	**40 %**

Žánr	Pravděpodobnost klasické hudby	Pravděpodobnost jazzu	Pravděpodobnost metalu	Pravděpodobnost opery	Pravděpodobnost rapu	Pravděpodobnost reggae
Klasická hudba	**40 %**	0 %	0 %	60 %	0 %	0 %
Jazz	40 %	**0 %**	0 %	60 %	0 %	0 %
Metal	0 %	20 %	**40 %**	0 %	20 %	20 %
Opera	40 %	0 %	0 %	**60 %**	0 %	0 %
Rap	20 %	40 %	0 %	0 %	**20 %**	20 %
Reggae	0 %	0 %	20 %	0 %	20 %	**60 %**

Žánr	Pravděpodobnost klasické hudby	Pravděpodobnost jazzu	Pravděpodobnost metalu	Pravděpodobnost opery	Pravděpodobnost rapu	Pravděpodobnost reggae
Klasická hudba	**40 %**	0 %	0 %	60 %	0 %	0 %
Jazz	40 %	**0 %**	0 %	60 %	0 %	0 %
Metal	0 %	20 %	**40 %**	0 %	20 %	20 %
Opera	40 %	0 %	0 %	**60 %**	0 %	0 %
Rap	20 %	20 %	0 %	0 %	**40 %**	20 %
Reggae	0 %	0 %	20 %	0 %	20 %	**60 %**

Žánr	Pravděpodobnost klasické hudby	Pravděpodobnost jazzu	Pravděpodobnost metalu	Pravděpodobnost opery	Pravděpodobnost rapu	Pravděpodobnost reggae
Klasická hudba	**40 %**	0 %	0 %	60 %	0 %	0 %
Jazz	40 %	**0 %**	0 %	60 %	0 %	0 %
Metal	0 %	20 %	**40 %**	0 %	20 %	20 %
Opera	40 %	20 %	0 %	**40 %**	0 %	0 %
Rap	20 %	40 %	0 %	0 %	**40 %**	0 %
Reggae	0 %	0 %	0 %	0 %	40 %	**60 %**

Žánr	Pravděpodobnost klasické hudby	Pravděpodobnost jazzu	Pravděpodobnost metalu	Pravděpodobnost opery	Pravděpodobnost rapu	Pravděpodobnost reggae
Klasická hudba	**40 %**	0 %	0 %	60 %	0 %	0 %
Jazz	40 %	**0 %**	0 %	60 %	0 %	0 %
Metal	0 %	20 %	**40 %**	0 %	20 %	20 %
Opera	20 %	40 %	0 %	**40 %**	0 %	0 %
Rap	20 %	40 %	0 %	0 %	**40 %**	0 %
Reggae	0 %	20 %	0 %	0 %	40 %	**40 %**

Diskuze

V této kapitole diskutujeme detaily našeho řešení, rozbor nedostatků — proč některé věci nefungují, a jak by to šlo potenciálně v budoucnu opravit.

Deskriptory

Z prvotní analýzy nám vyšlo jako **nejvhodnější** zkoumat deskriptory `SpectrumSpreadType`, `SpectrumCentroidType`, `Area Method of Moments`, `Method of Moments` a `MFCC`.

Na základě měření v experimentální sekci můžeme vidět, že pro deskriptory `SpectrumSpreadType` a `SpectrumCentroidType` počítající **rozsah** a **rozptyl** spektra nám vyšla přesnost jen necelých **22** %, respektive **28** %, což není o příliš více než **náhodný** výběr *(16%)*.

S téměř 100 % přesností rozpoznaly tyto deskriptory pouze žánr **opera** — to sedí s prvotní hypotézou, že budou tyto deskriptory vhodné pro rozpoznávání žánrů, kde se často objevují **vysoké frekvence**, mezi které opera patří. Přesto ale tyto deskriptory nerozpoznaly spolehlivě **metal**, kde se vysoké a nízké frekvence také často střídají. Zajímavostí je také to, že deskriptor `SpectrumCentroidType` správně rozpoznal **reggae**, zatímco deskriptor `SpectrumSpreadType` rozpoznal **jazz**.

Pro **celkové** použití tyto deskriptory příliš vhodné vzhledem ke své přesnosti **nebudou**, dalo by se je ale vhodně **použít** pro rozlišení **opery** a **jazz**u, respektive **opery** a **reggae**.

Deskriptor `Area Method of Moments` z knihovny jAudio vypadal při analýze jako velmi **užitečný**, při experimentálním testování se však ukázalo, že jeho přesnost *(u průměru i std. odchylky)* dosahuje jen pouhých **22** %, tedy téměř náhodný výběr. Spolehlivě rozpoznal pouze **operu**, tento deskriptor tedy **nemá** cenu používat pro **žádné** žánry.

Deskriptor `Method of Moments` *(průměr)* se ale již ukázal jako **použitelný**, jelikož jeho přesnost dosahuje **50** %, což je v případě audio retrieval **velká** úspěšnost.

Tento deskriptor spolehlivě rozpoznal **jazz**, **rap** a **operu**, šel by tedy využít pro jejich rozlišování.

Posledním zkoumaným deskriptorem bylo `MFCC`, a to jak **průměr** všech 13 koeficientů, tak **časová řada** prvních 250 hodnot prvního koeficientu.

Deskriptor průměr MFCC koeficientů dopadl ze všech deskriptorů **nejlépe** s přesností **71** % procent. Dokázal velmi dobře rozpoznat **všechny** žánry, s pár občasnými chybami mezi klasickou hudbou a operou, rapem, jazzem a reggae. To **sedí** s hypotézou, že MFCC slouží dobře k rozpoznávání mluvené řeči a typu nástrojů.

Deskriptor **časové řady** prvních 250 členů prvního koeficientu MFCC měl lehce **horší** přesnost než průměr všech koeficientů *(57 %)*. Stejně jako průměr všech koeficientů správně rozpoznal většinu žánrů, měl ale větší problém s **rapem**.

Po analýze **všech** deskriptorů a jejich experimentálním **testování** bychom tedy jako zlepšení zvolili použití deskriptoru **MFCC průměr** s tím, že skladby klasifikované jako jazz, rap a opera by se ještě **převážily** s pomocí deskriptoru **Method of Moments** pro jejich **přesnější** rozpoznání.

Parametry

Při testování parametru `TEST_K_NEAREST` neboli **počet nejbližších sousedů** můžeme vidět, že nejlepší přesnost a nejpřesnější výsledky vychází pro **k = 3**, respektive **k = 5**. To je velmi pravděpodobně tím, že z každého žánru je v databázi k dispozici právě 5 skladeb *(při testování přesnosti 2-3)*, u **nižšího** čísla k tedy může nějaký **false hit** jiného žánru zakrýt ten správný, u **vyššího** k zase není v databázi dostatek skladeb správného žánru, a **snižujeme** tedy přesnost.

Parametr `DTW_WIDTH` na většinu deskriptorů neměl vliv, jelikož se jednalo o jednu hodnotu. Mělo smysl ho zkoumat právě pro deskriptor `MFCC` *(časová řada)*, kde určil, jak **moc daleko** od diagonály se má hledat cesta mezi řadami.

Z experimentálního testování můžeme vidět, že čím **větší** šířku volíme, tím větší okolí prozkoumáváme a tím **déle** tedy běží porovnávání.

Co je ovšem zajímavé je ale to, že nejlepší přesnosti dosahujeme pro šířku **5**, i přesto, že se nejedná se o největší okolí a přesnost pro šířku **0** *(hledáme pouze na diagonále)* není o moc horší. Mezi časovými řadami je tedy nejspíše **nejčastější** posun právě do 5 hodnot, a proto vychází přesnost pro šířku 5 **nejlépe**.

Co není překvapivé je fakt, že čím **větší** šířku volíme, tím **déle** porovnávání běží. Při ostrém běhu bychom z časových důvodů volili spíše šířku 0 nebo právě deskriptor `MFCC` *(průměr)*, jelikož se jedná o časově rychlejší řešení s téměř stejnou, ne-li **lepší** přesností.

Závěr

Cílem projektu bylo vytvořit webovou aplikaci, která pro zadanou **skladbu** rozpozná, do kterého hudebního **žánru** patří.

Nejprve jsme pro každý z hudebních žánrů nalezli **skladby** a zvolili vhodné **deskriptory** tak, aby měly skladby v rámci stejného žánru deskriptory **blízko** a různé žánry byly co **nejdál** od sebe.

Následně jsme celou aplikaci **naimplementovali** — od načtení souboru, extrakce deskriptorů, spočítání vzdálenostní funkce a přiřazení jednotlivých skladeb do žánru. Aplikace je dostupná přes **konzolové** i **webové** rozhraní, kde lze zvolit jednotlivé **parametry** porovnávání.

Při vhodné volbě parametrů jsme došli u rozpoznávání k **přesnosti** až **72%**, což vzhledem k typické přesnosti modelů, času a prostředkům, které jsme měli považujeme za **úspěšné**.

V aplikaci jsme *(především z časových důvodů)* podrobně neanalyzovali a **nevyzkoušeli** úplně všechny možné *(například i vysokoúrovňové)* deskriptory, taktéž jsme neimplementovali další způsoby **porovnávání** jednotlivých deskriptorů (jiné než DTW) nebo jiné způsoby **klastrování** (například neuronové sítě).

Mezi případnými budoucími rozšířeními by mohlo být **podrobnější** rozlišení žánrů, jak bylo již avizováno v kapitole Diskuze *(po použití jednoho deskriptoru by se ještě skladby ohodnocené žánry jazz, rap a opera přeměřily dalším)*.

Dalším možným **vylepšením** by bylo zvolit v případě stejných pravděpodobností žánr **náhodně** *(aktuálně volíme vždy ten první v pořadí)*, čímž bychom mohli ještě **zvýšit** přesnost modelu.

Tento projekt rozhodně **pomohl** lépe pochopit problematiku **audio retrieval** — vyzkoušeli jsme si analýzu deskriptorů, napsat porovnávací funkci a navrhnout způsob klastrování. Naše řešení jsme následně experimentálně **odzkoušeli** a diskutovali jeho možná vylepšení.