

I-SUNS: Zadanie č. 1

Richard Körösi (111313)

Deadline: 26.10.2023

1 Prvá časť

1.1 Hľadanie a odstránenie outlierov

Dáta, ktoré nám boli poskytnuté obsahovali aj takzvané outliery (neobvyklé hodnoty), ktoré sme následne odhalili a odstránili.

1.1.1 Hľadanie outlierov

Nechali sme si vypísať minimá a maximá všetkých číselných stĺpcov z databázy a následne sme ich porovnali s popismi a intervalmi stĺpcov, ktoré nám boli poskytnuté v zadanií. Stĺpce v ktorých sme odhalili prekročenie intervalov, alebo iné nejasnosti boli nasledovné:

	----- Min -----		----- Max -----
danceability	0.0	danceability	8.375
energy	0.000197	energy	1.0
loudness	-47.046	loudness	1.519
speechiness	0.0	speechiness	0.965
acousticness	0.0	acousticness	0.996
instrumentalness	0.0	instrumentalness	0.994
liveness	0.00967	liveness	0.997
valence	0.0	valence	0.995
tempo	0.0	tempo	241.423
duration_ms	-427346.0	duration_ms	1930821300.0
popularity	0.0	popularity	82.0
number_of_artists	1.0	number_of_artists	19.0
explicit	False	explicit	True
dtype: object		dtype: object	

Figure 1: Namerané MIN a MAX (pred úpravou)

Názov	Očakávané minimum	Nájdené minimum	Očakávané maximum	Nájdené maximum
Danceability	0.0	0.0	1.0	8.375
Energy	0.0	0.0	1.0	1.0
Loudness	-60.0	-47.046	0.0	1.519
Speechiness	0.0	0.0	1.0	0.965
Acousticness	0.0	0.0	1.0	0.996
Instrument...	0.0	0.0	1.0	0.994
Liveness	0.0	0.00967	1.0	0.997
Valence	0.0	0.0	1.0	0.995
Tempo	24.0	0.0	300.0	241.423
Duration_ms	20000.0	-427346.0	1967400.0	1930821300.0
Popularity	0.0	0.0	100.0	82.0
NO_Artists	1.0	1.0	25.0	19.0
Explicit	False	False	True	True

Table 1: Hľadanie outlierov (pred úpravou)

Prekročenia intervalov sme zaznamenali v 4 stĺpcach databázy:

- Danceability (MAX)
- Loudness (MAX)
- Tempo (MIN)
- Duration (MIN & MAX)

Vo všetkých týchto prípadoch okrem "Loudness" bol rozdiel medzi nameranými a očakávanými hodnotami pomerne veľký a tým pádom sme tie hodnoty považovali za outliery. V prípade "Loudness" namerané maximum bolo súčasťou vyššie ako očakávané, ale o pomerne malú časť a tým pádom ho nemusíme považovať za outlier.

1.1.2 Odstránenie outlierov

Po nájdení outlierov sme dané hodnoty vyfiltrovali, tak aby nám ostali len reprezentatívne dátá.

	----- Min -----	----- Max -----
danceability	0.0567	danceability
energy	0.000197	0.975
loudness	-47.046	energy
speechiness	0.0226	1.0
acousticness	0.0	loudness
instrumentalness	0.0	1.519
liveness	0.00967	speechiness
valence	0.00001	0.965
tempo	31.988	acousticness
duration_ms	21866.0	0.996
popularity	0.0	instrumentalness
number_of_artists	1.0	0.994
explicit	False	liveness
dtype: object		0.997
		valence
		tempo
		duration_ms
		1967400.0
		popularity
		82.0
		number_of_artists
		19.0
		explicit
		True
		dtype: object

Figure 2: Namerané MIN a MAX (po úprave)

Názov	Očakávané minimum	Nájdené minimum	Očakávané maximum	Nájdené maximum
Danceability	0.0	0.0	1.0	0.975
Energy	0.0	0.0	1.0	1.0
Loudness	-60.0	-47.046	0.0	1.519
Speechiness	0.0	0.0	1.0	0.965
Acousticness	0.0	0.0	1.0	0.996
Instrument...	0.0	0.0	1.0	0.994
Liveness	0.0	0.00967	1.0	0.997
Valence	0.0	0.0	1.0	0.995
Tempo	24.0	31.988	300.0	241.423
Duration_ms	20000.0	21866.0	1967400.0	1967400.0
Popularity	0.0	0.0	100.0	82.0
NO_Artists	1.0	1.0	25.0	19.0
Explicit	False	False	True	True

Table 2: Hľadanie outlierov (pred úpravou)

Tabuľka č.2 ukazuje nájdené minimá a maximá po vyfiltrovaní outlierov. Dáta, ktoré nám po filtrovaní ostali považujeme za reprezentatívne dátá a môžeme s nimi ďalej pracovať.

1.2 Odstraňovanie nedôležitých stĺpcov a ošetrenie NULL hodnôt

V databáze sa aj po odstránení outlierov nachádzali niektoré hodnoty, ktorých sme sa potrebovali zbaviť.

- Celé stĺpce, ktoré sme určili ako zbytočné
- Riadky, kde vlastnosti, ktoré sme si určili ako dôležité, mali hodnotu NULL

	Length of dataset: 11915	Length of dataset: 11512
danceability	0	0
energy	0	0
loudness	0	0
speechiness	0	0
acousticness	0	0
instrumentalness	0	0
liveness	0	0
valence	0	0
tempo	0	0
duration_ms	0	0
popularity	116	0
number_of_artists	120	0
explicit	0	0
name	0	0
url	0	0
genres	0	0
filtered_genres	0	0
top_genre	169	0
emotion	0	0
dtype:	int64	int64

Figure 3: Informácie o stĺpcoch (pred a po úprave)

Ako vidno na obrázku č.3 pred úpravou sme mali v 3 stĺpcoch (popularity, number_of_artists, top_genre) riadky, ktoré mali hodnoty NULL. Tieto riadky sme sa následne rozhodli odstrániť, keďže ich počet sme považovali za zanedbateľný. Za zbytočné stĺpce sme považovali:

- name
- url
- genres
- filtered_genres

Prvé dva spomenuté stĺpce boli tzv. identifikácie a tie sa do NS neposielajú, preto sme sa ich rozhodli vymazať. Druhé dva stĺpce sme sa rozhodli nepoužiť z dôvodu zachovania jednoduchosti implementácie, tieto dva stĺpce mali totiž alternatívu vo forme stĺpca top_genre, ktorý oproti dvom spomenutým obsahoval vždy len jednu stringovú hodnotu (vymazané stĺpce boli polia stringov).

1.3 Zakódovanie nečíselných stĺpcov

V ďalšej časti sme sa zamerali na zakódovanie nečíselných stĺpcov. V našom prípade išlo o stĺpce top_genre a emotion. Stĺpec top_genre sme zakódovali pomocou metódy Dummy Encoding. To znamená, že sa nám hodnoty, ktoré stĺpec obsahoval zmenili na samotné stĺpce a každému riadku sa následne priradila hodnota True/False, na základe toho, či danú hodnotu predtým stĺpec top_genre obsahoval. Následne sme ešte bool hodnoty zmenili na hodnoty typu float (True-1.0, False-0.0). Stĺpec emotion sme zakódovali pomocou metódy Label Encoding, to znamená že sa každému unikátnemu stringu pridelila int hodnota. Pre túto metódu sme sa rozhodli, aby sa nám lepšie pracovalo s knižnicou sklearn. V neskorších častiach dokumentácie, pri práci s Kerasom sme však využili Dummy Encoding aj na emócie.

danceability	float64	danceability	float64
energy	float64	energy	float64
loudness	float64	loudness	float64
speechiness	float64	speechiness	float64
acousticness	float64	acousticness	float64
instrumentalness	float64	instrumentalness	float64
liveness	float64	liveness	float64
valence	float64	valence	float64
tempo	float64	tempo	float64
duration_ms	float64	duration_ms	float64
popularity	float64	popularity	float64
number_of_artists	float64	number_of_artists	float64
explicit	bool	explicit	float64
top_genre	object	emotion	int32
emotion	object	ambient	float64
dtype: object		anime	float64
		...	
		rock	float64
		rockabilly	float64
		ska	float64
		sleep	float64
		soul	float64

Figure 4: Informácie o typoch premenných v stĺpcoch (pred a po úprave)

1.4 Vytvorenie vstupnej a výstupnej množiny a rozdeľenie dát do trénovacej, validačnej a testovacej množiny

Dáta sme rozdelili na dve množiny a to na vstupnú a výstupnú. Vstupná obsahovala všetky stĺpce, až na stĺpec, ktorý sme chceli aby NS na základe tréningu vedela určiť (emotion). Výstupná množina obsahovala práve tento jeden stĺpec. Následne sme hodnoty týchto množín rozdelili do 3 kategórií/množín podľa toho na čo sme dátu chceli použiť. Prvá a najväčšia množina bola trénovacia, táto množina obsahovala 80% dát. Následne si v pomere 1:1 dátu rozdelili validačná a testovacia množina. Výsledný pomer bol teda (8:1:1).

```
Full dataset: (11512, 46)
X_train: (9209, 45)
X_valid: (1151, 45)
X_test: (1152, 45)
y_train: (9209,)
y_valid: (1151,)
y_test: (1152,)
```

Figure 5: Informácie o počte hodnôt v daných množinách v pomere (8:1:1)

```
1 X = dframe.drop(columns=['emotion'])
2 y = dframe['emotion']
3
4 X_train, X_valid_test, y_train, y_valid_test = train_test_split(
5     X, y, shuffle=True, test_size=0.2, random_state=42)
6 X_valid, X_test, y_valid, y_test = train_test_split(X_valid_test,
7     y_valid_test, shuffle=True, test_size=0.5, random_state=42)
8 scaler = MinMaxScaler()
9 X_train = scaler.fit_transform(X_train)
10 X_valid = scaler.transform(X_valid)
11 X_test = scaler.transform(X_test)
```

Pri knižnici sklearn sa však validačné dáta nevyužívajú, dáta sme si rozdelili na pomer 8:1:1 aby sme si precvičili ako sa dajú na takýto pomer rozdeliť, pretože neskôr pri práci s Kerasom sme toto rozdelenie potrebovali.

1.5 Normalizácia a následné zobrazenie vstupných dát trénovacej množiny

V danej podúlohe sme mali za úlohu dátá správne normalizovať. Dátá sme normalizovali podľa postupu povedaného na cvičeniach/seminároch.

1.5.1 Dátá pred normalizáciou

	----- Min -----	----- Max -----
danceability	0.056700	0.975
energy	0.000197	1.000
loudness	-47.046000	1.519
speechiness	0.022600	0.965
acousticness	0.000000	0.996
instrumentalness	0.000000	0.988
liveness	0.009670	0.997
valence	0.000010	0.995
tempo	32.451000	213.990
duration_ms	21866.000000	1967400.000
popularity	0.000000	82.000
number_of_artists	1.000000	17.000
...		
sleep	0.000000	1.000
soul	0.000000	1.000
dtype: float64		

Figure 6: Tabuľky dát pred normalizáciou

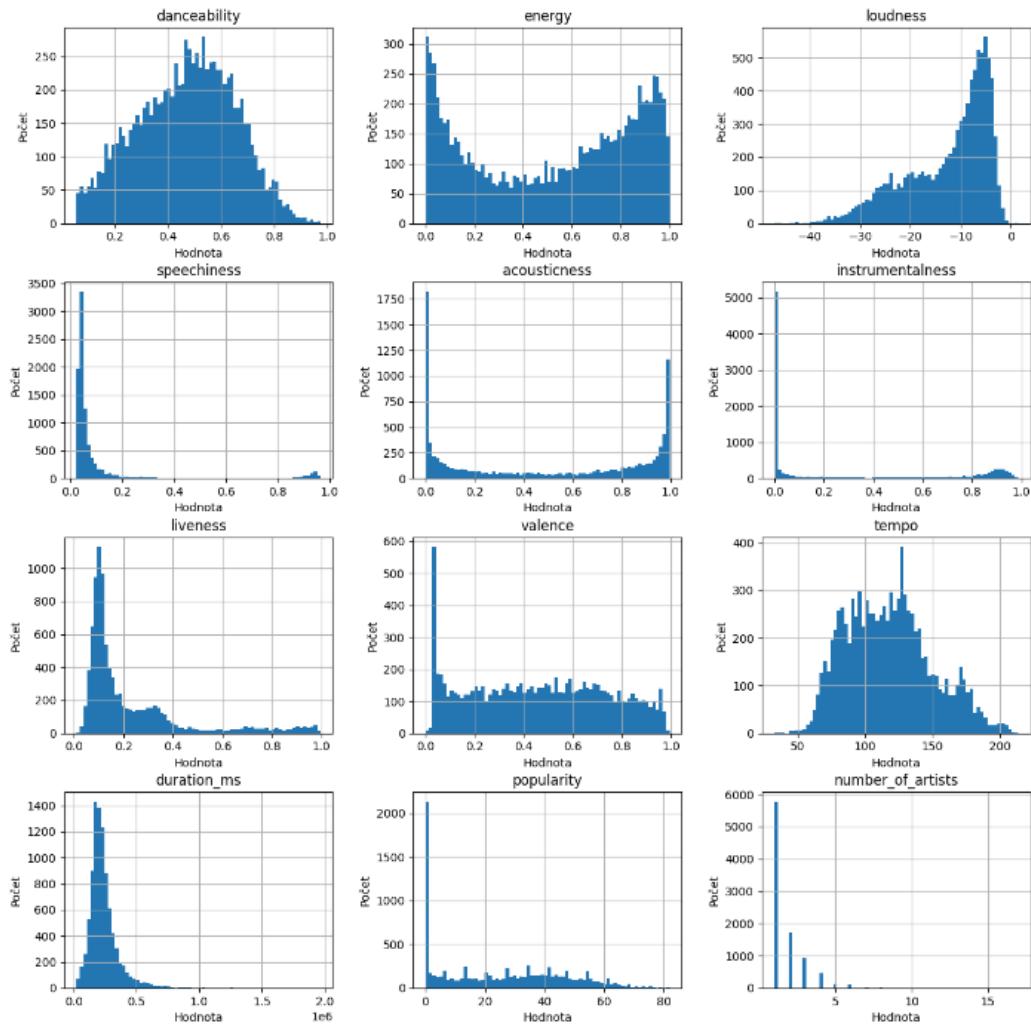


Figure 7: Histogramy dát pred normalizáciou

Os "Hodnota" predstavuje intervalové hodnoty pre dané vlastnosti pesničky (nad každým grafom je spomenuté o akú vlastnosť pesničky sa jedná) v danom stĺpci, os "Počet" predstavuje počet výskytov danej hodnoty v našich dátach.

1.5.2 Dáta po normalizácii

	----- Min -----	----- Max -----	
danceability	0.0	danceability	1.0
energy	0.0	energy	1.0
loudness	0.0	loudness	1.0
speechiness	0.0	speechiness	1.0
acousticness	0.0	acousticness	1.0
instrumentalness	0.0	instrumentalness	1.0
liveness	0.0	liveness	1.0
valence	0.0	valence	1.0
tempo	0.0	tempo	1.0
duration_ms	0.0	duration_ms	1.0
popularity	0.0	popularity	1.0
number_of_artists	0.0	number_of_artists	1.0
...		...	
sleep	0.0	sleep	1.0
soul	0.0	soul	1.0
dtype:	float64	dtype:	float64

Figure 8: Tabuľky dát po normalizácii

Normalizácia nám ”stlačila” všetky hodnoty do intervalu $<0,1>$. Zmeny sme si mohli všimnúť v stĺpcoch, kde pred normalizáciou tento interval neplatil, ako napríklad pri hlasitosti, tempe alebo dĺžky danej pesničky.

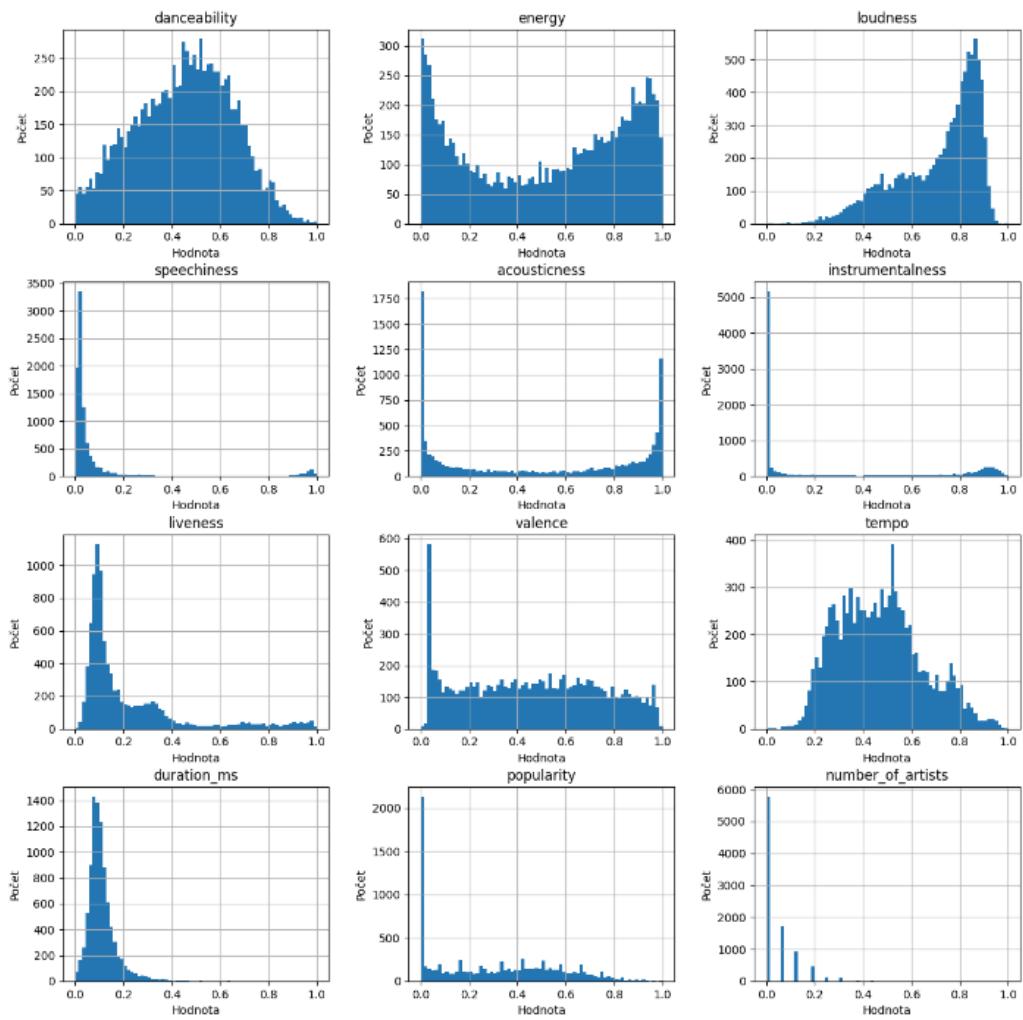


Figure 9: Histogramy dát po normalizácii

Os "Hodnota" predstavuje intervalové hodnoty pre dané vlastnosti pesničky (nad každým grafom je spomenuté o akú vlastnosť pesničky sa jedná) v danom stĺpci po normalizácii, os "Počet" predstavuje počet výskytov danej hodnoty v našich dátach.

1.5.3 Zobrazenie True/False dát

Dáta, ktoré sme si už síce zmenili tiež na hodnoty typu float, ale mali len dve hodnoty (1.0-True, 0.0-False) sme sa rozhodli zobraziť pomocou koláčových grafov.

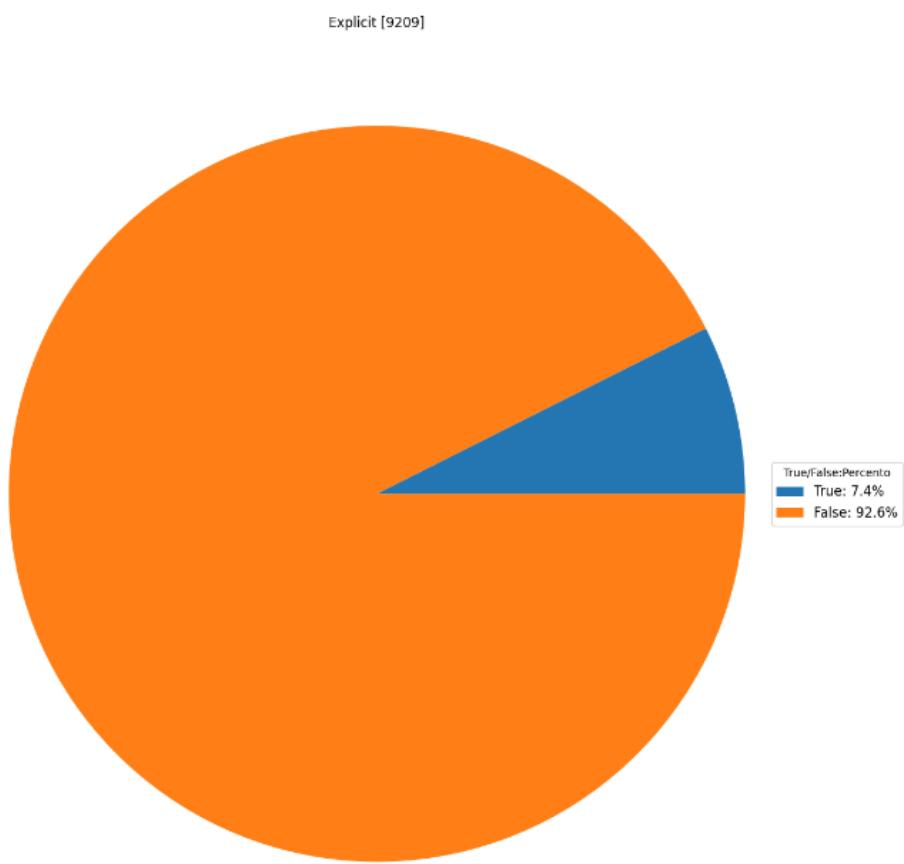


Figure 10: Graf zobrazujúci percentuálny pomer medzi hodnotami explicit v našich dátach

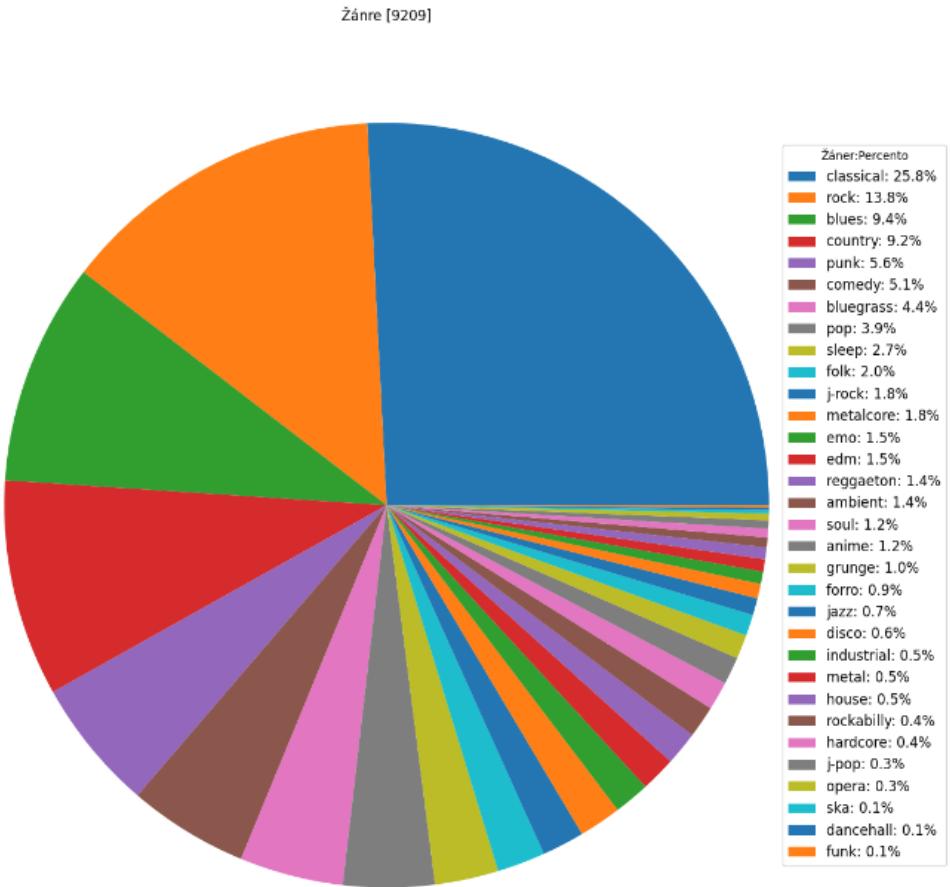


Figure 11: Graf zobrazujúci pomer medzi hodnotami top_genre

Oba koláčové grafy vyjadrujú percentuálny pomer medzi hodnotami, ktoré každá pesnička mohla nadobudnúť, jedná sa o stĺpce explicit a top_genre. Pričom platilo, že pesnička má práve jeden top žánr.

1.6 Trénovanie neurónovej siete

Pri trénovaní siete sme využili následovný kód:

```
1 clf = MLPClassifier(  
2     hidden_layer_sizes=(200, 150, 100),  
3     random_state=1,  
4     max_iter=250,  
5     early_stopping=True  
6 ).fit(X_train, y_train)
```

V kóde sme pri hľadaní optimálneho riešenia menili hodnotu `hidden_layer_sizes`. Nasledujúca tabuľka zobrazuje úspešnosť NS na základe zmien, ktoré sme vykonali v premennej `hidden_layer_sizes`.

Skryté vrstvy	Úspešnosť na trénovacích dátach	Úspešnosť na testovacích dátach
1	23.56%	22.74%
2	77.25%	79.08%
10	86.39%	85.07%
100	87.77%	86.55%
500	87.92%	86.63%
150,150	88.45%	86.20%
150,100,100	90.35%	86.81%
1000,300,200	90.45%	86.89%
200,150,100	88.98%	87.15%

Table 3: Úspešnosť NS vzhľadom na zmenu skrytých vrstiev

Za najlepšiu variantu sme považovali poslednú možnosť v tabuľke, keďže má jednu z najvyšších úspešností na trénovacích dátach a zároveň rozdiel medzi úspešnosťou na trénovacích a testovacích dátach mala zanedbateľný.

1.7 Vyhodnotenie natrénovanej siete

Po nájdení optimálneho nastavenia a natrénovaní neurónovej siete sme zanalýzovali jej úspešnosť pomocou percentuálnej úspešnosti a konfúznych matíc.

Random accuracy: 0.25

MLP accuracy on train set: 0.8897817352589857

MLP accuracy on test set: 0.8715277777777778

Figure 12: Úspešnosť neurónovej siete

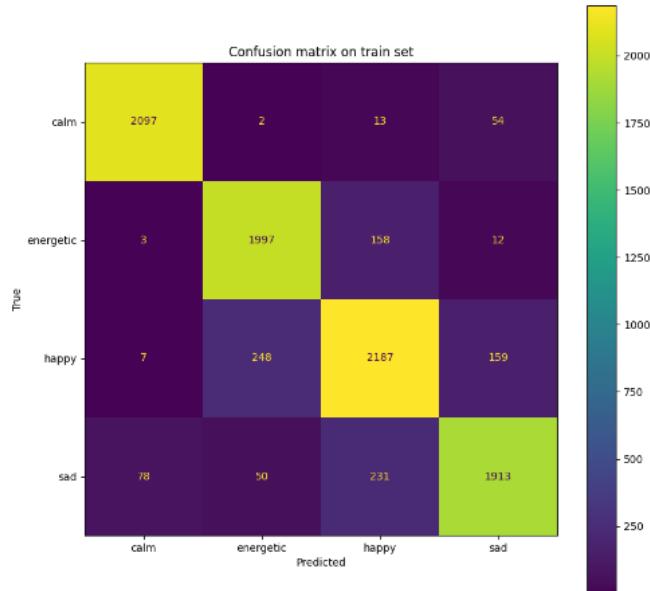


Figure 13: Konfúzna matica NS na trénovacích dátach

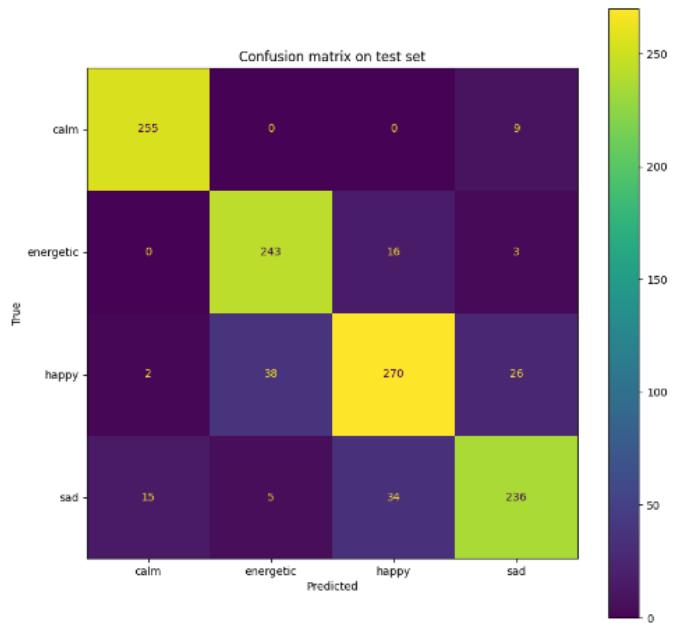


Figure 14: Konfúzna matica NS na testovacích dátach

Z konfúznych matíc na obrázkoch vyššie si môžeme všimnúť, že sa nám podarilo správne natrénovať neurónovú sieť (na konfúznych maticiach vidíme jasne diagonálu).

2 Druhá časť (EDA)

V druhej časti zadania sme sa venovali analýze dataframu cez EDA. Hľadali sme vzťahy medzi dátami a následne sme ich vizualizovali pomocou rôznych grafov.

2.1 Proces hľadania vzťahov

Pre jednoduchšie hľadanie vzťahov sme si vytvorili "heatmapy" (vizualizácie tzv. korelačných matíc, ktoré slúžia na zobrazovania hodnôt korelácií medzi rôznymi premennými v dátach).

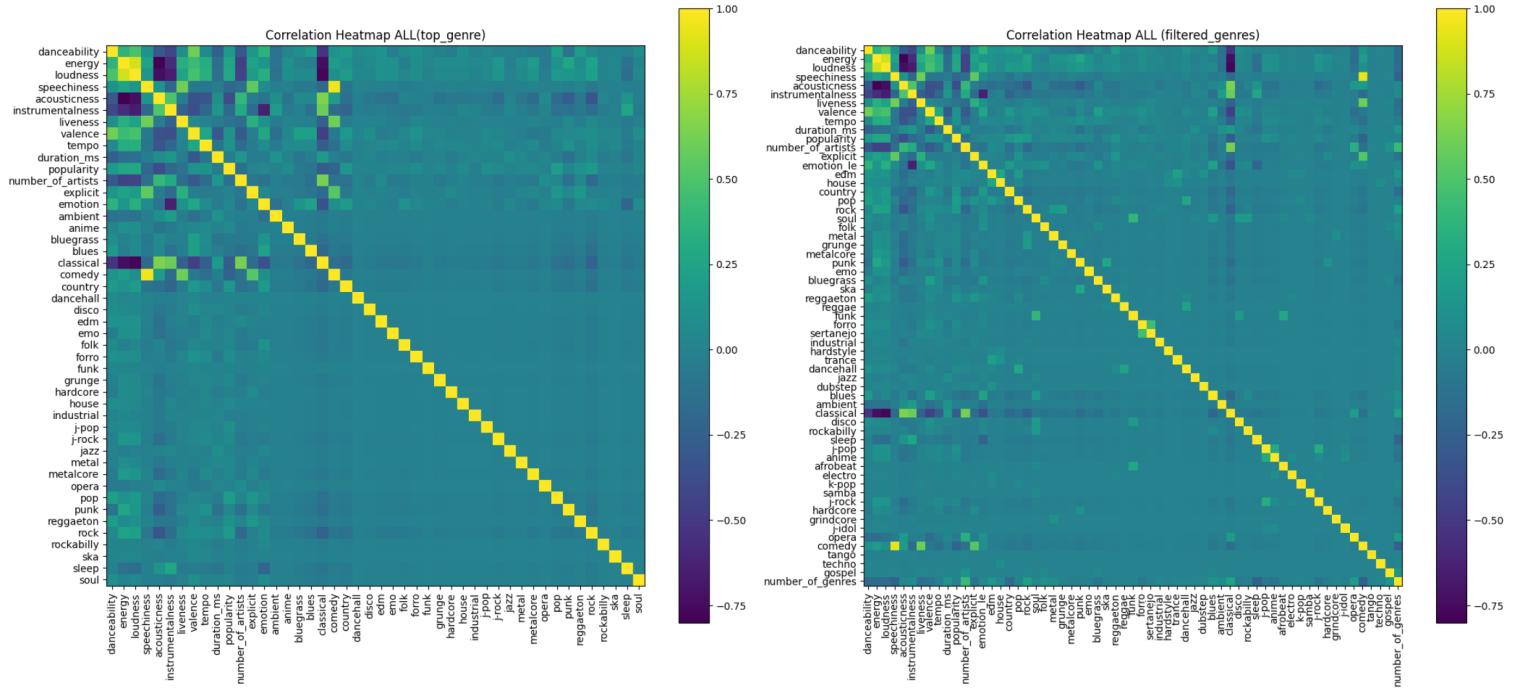


Figure 15: Heatmapy pre dataframe s top_genre a s filtered_genres

Heatmapy na obrázku č.15 znázorňujú vzťahy medzi všetkými relevantnými dátami o daných pesničkách.

2.2 Vzťah medzi hlasitostou a energiou pesničky

Podľa heatmapy, ktorú sme si vygenerovali by mal existovať vzťah medzi hlasitostou a energiou pesničky. Tento fakt sme si overili pomocou dvoch rôznych grafov.

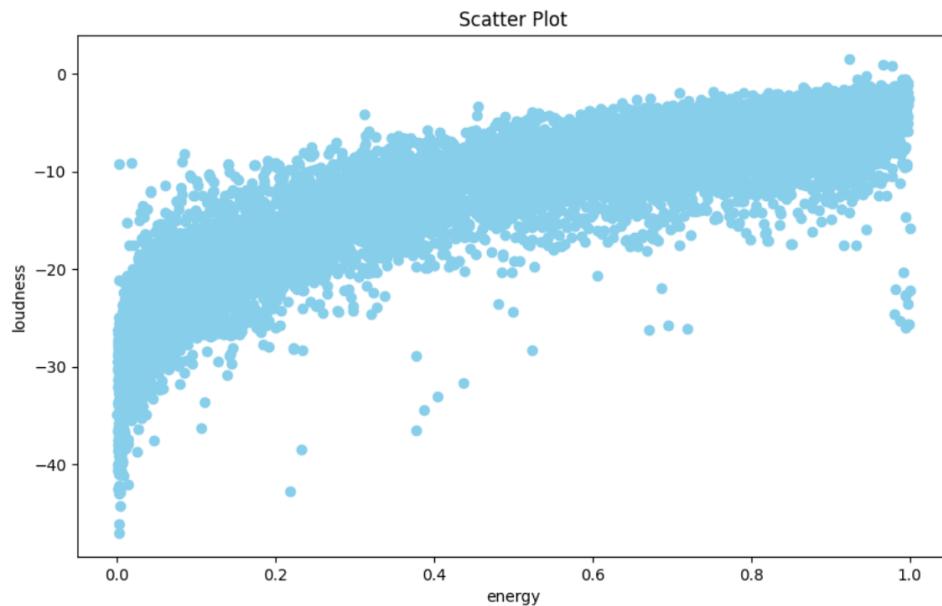


Figure 16: Korelačný graf Hlasitosť Energia

Obrázok č.16 reprezentuje korelačný graf, už priamo z neho sa dá vyčítať, že so stúpajúcou energiou rastie aj úroveň hlasitosti pesničky. Tento fakt sme si ešte následne overili aj pomocou druhého grafu, dátá sme si rozdelili na 12 rovnakých intervalov podľa energie a následne sme si na týchto intervaloch vypočítali priemerné hodnoty hlasitosti a tie sme následne zaznačili do grafu, ako je vidno na obrázku č.17.

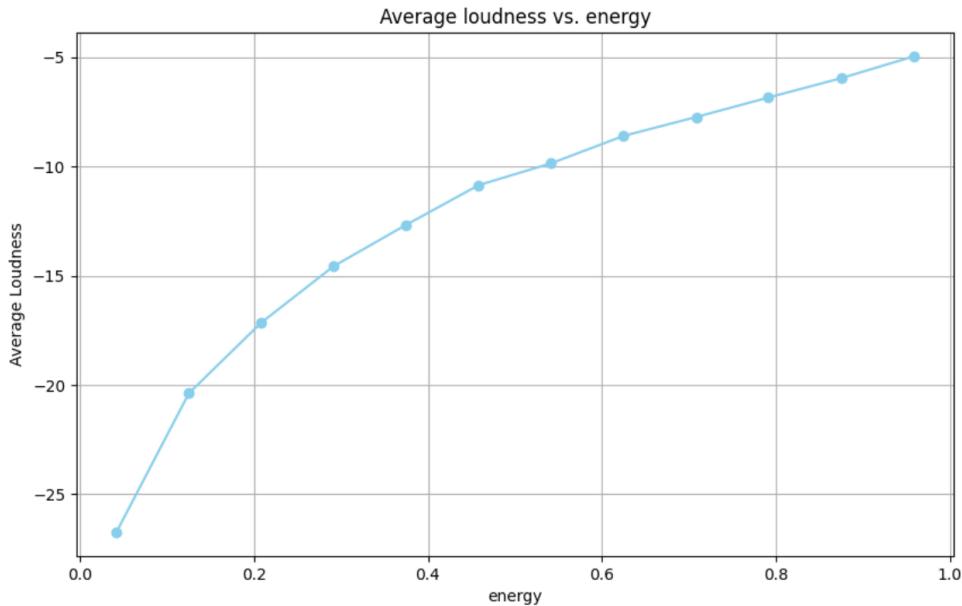


Figure 17: Priemerná Hlasitosť vs Energia

Na základe vizualizácie sme mohli skonštatovať, že medzi hlasitosťou a energiou pesničky existuje vzťah. Viac energetické pesničky sú priemerne aj hlasnejšie, ako menej energetické pesničky.

2.3 Vzťah medzi ”speechiness” a živostou

Ďalší vzťah, ktorý sme si chceli overiť bol medzi ”speechiness” (vo voľnom preklade hovoreným slovom v pesničkách) a živostou, ktorá reprezentuje pravdepodobnosť, že nahrávka bola zostrojená pred naživo. Postupovali sme podobne ako v predošлом príklade. Najprv sme si nechali vygenerovať korelačný graf a následne sme si dátá živosti rozdelili na 12 intervalov a v nich sme si prehľadne zobrazili hodnoty ”speechiness”.

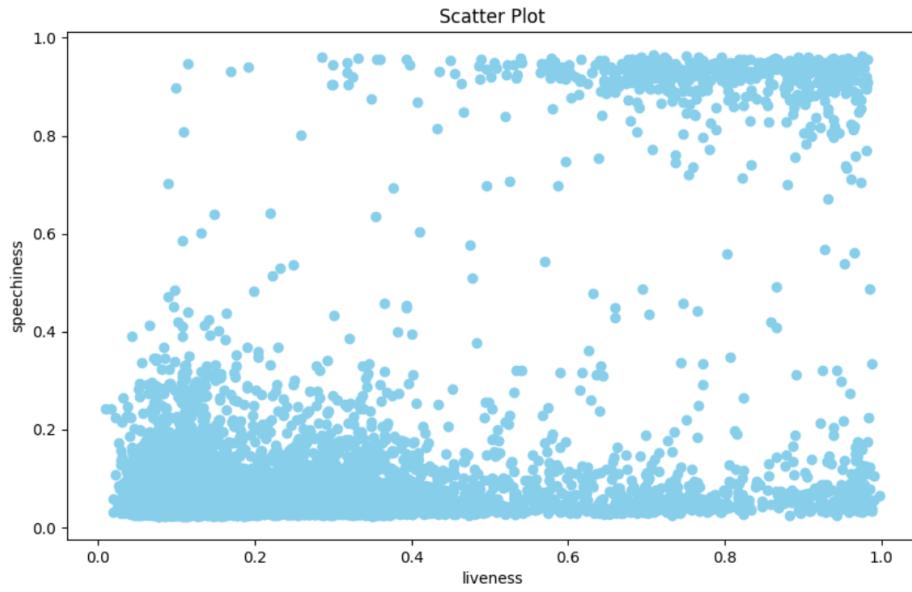


Figure 18: Korelačný graf Speechiness Živost

Na prvý pohľad je vidno, že vzťah medzi dátami existuje, keďže vidíme, že body na grafe nie sú po celej ploche náhodne, ale skôr sú zoskupené na dvoch miestach. Na grafe je presnejšie vidno, že po hodnote 0.5 živosti sa speechiness drží len v nízkych hodnotách a od 0.5 hore skáče už takmer na maximum. Pre jasnejšie zobrazenie vzťahu sme vygenerovali aj druhý graf, kde sme pracovali s hodnotami speechiness v intervaloch živosti.

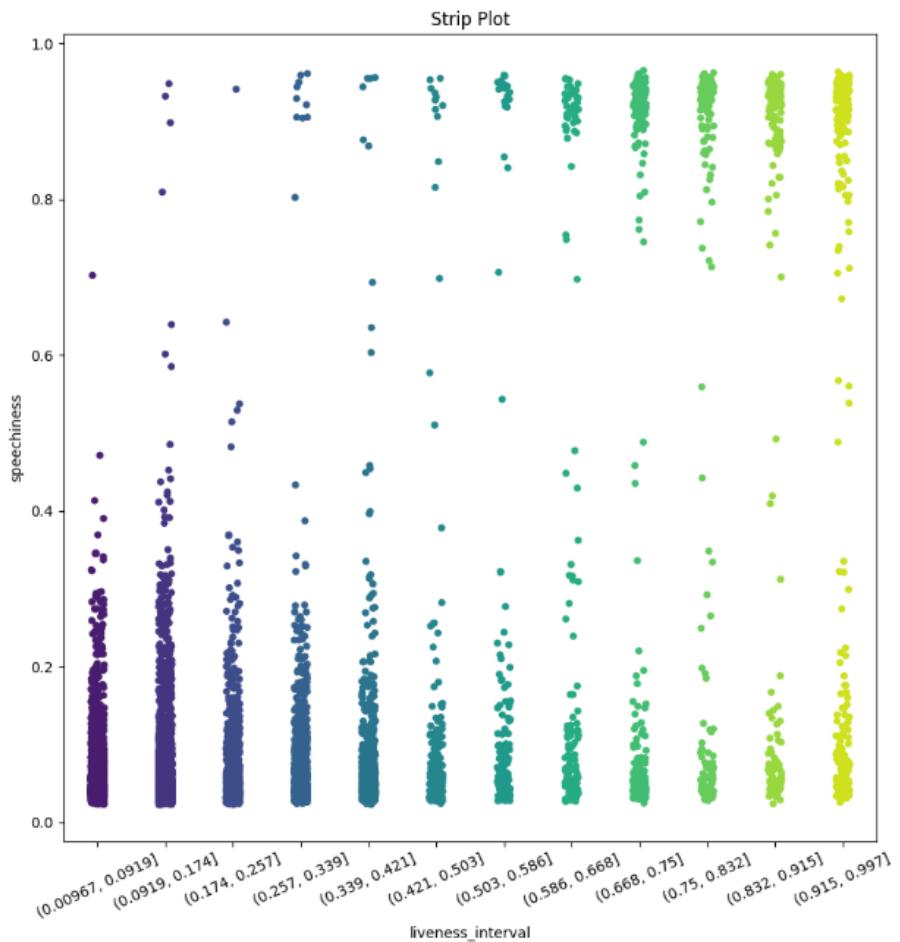


Figure 19: Speechiness vs Živost

Z obrázku č.19 vyplýva, že s rastúcou živostou rastie aj počet pesničiek s vysokou hodnotou speechiness. Môžeme teda konštatovať, že sa nám podarilo nájsť vzťah medzi týmito dvoma typmi dát.

2.4 Vzťah medzi top žánrom a danceability pesničky

Tretí vzťah, ktorý sme sa rozhodli zanalyzovať bol, vzťah medzi hlavným žánrom pesničky a jej vlastnosti, ako dobre sa na ňu tancuje. Pred samotnou analýzou sme očakávali, že žánre ako opera budú mať signifikatne nižšiu danceability ako napríklad disco.

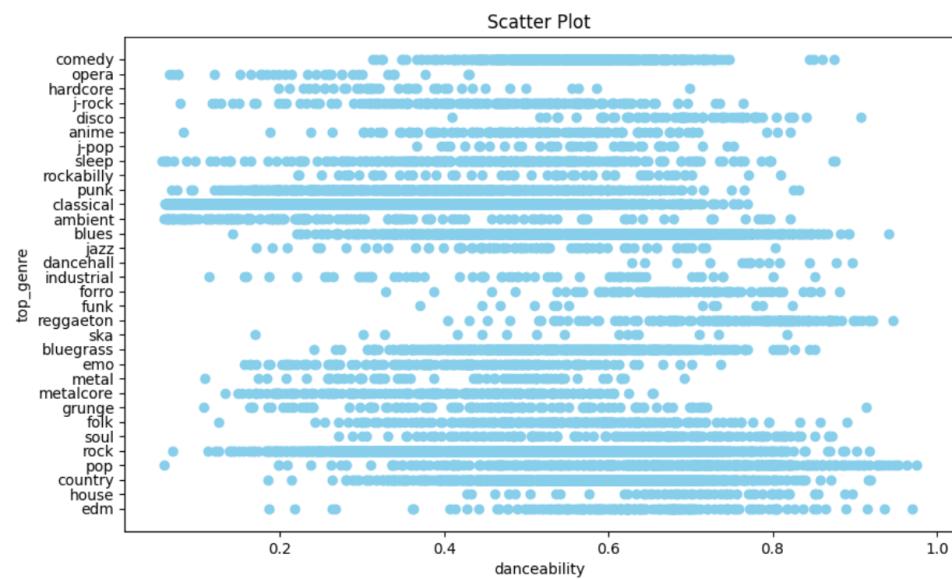


Figure 20: Korelačný graf TopŽáner Danceability

Obrázok č.20 zobrazuje korelačný graf medzi žánrami a danceability. Tento graf nám viac menej už potvrdil našu domnienku, keďže graf jednoznačne zobrazuje, že opera má vo väčšine danceability medzi intervalom $<0.2;0.4>$ a disco $<0.6;0.9>$. Okrem iného sme si taktiež všimli, že niektoré žánre sú rovnomerne rozprestreté po celej osi, zatiaľ čo niektoré sa zhlučujú len v specifických a malých intervaloch. Na analýzu týchto dát sme sa preto rozhodli využiť boxplot, v ktorom sme očakávali, že uvidíme oba vzťahy.

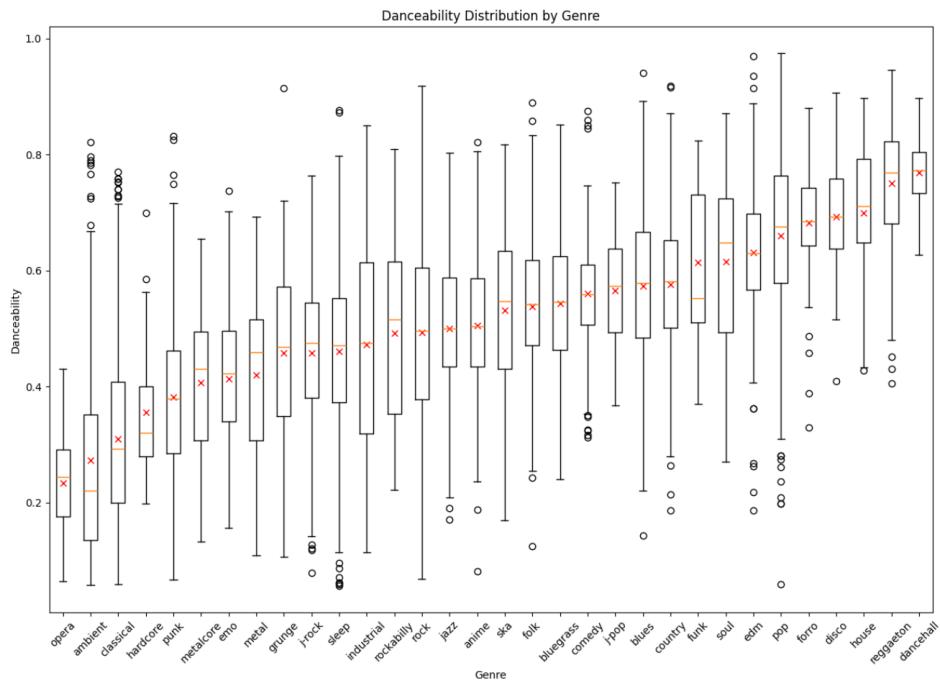


Figure 21: Boxplot Danceability vs TopŽáner

Boxplot graf na obrázku č.21 znázorňuje vzťahy medzi žánrom a danceability pesničky, medzi žánre s najlepšou danceability patrí napríklad dancehall, house, disco. Medzi najhoršie patria neprekvapivo opera alebo ambientná hudba. Dáta sú zoradené podľa priemeru od najslabšej danceability po najlepšiu, v grafe je priemer označený červeným krížikom. Okrem porovnania priemerných hodnôt danceability sme sa zamerali aj na samotné rozloženie danceability u rôznych žánrov. Tu taktiež boxplot potvrdil, to čo sme si všimli už pri korelačnom grafe, ako napríklad že žáner rock má 50% dát v intervale $<0.4;0.6>$ ale ďalších 25% z hornej aj dolnej časti pokrýva takmer celý interval danceability. Presný opak je napríklad žáner dancehall, u ktorého je 50% jeho všetkých dát len v malom intervale pri danceability 0.8 a aj jeho fúze sú oproti žánru rock veľmi malé. Môžeme teda skonštatovať, že sa nám podarilo nájsť dva zaujímavé vzťahy, prvý sa venoval zoradeniu žánrov podľa toho ako sa naň dá podľa priemeru tancovať a druhý sa zameriaval na vlastnosť žánrov, ktorá odhalovala pestrosť hodnôt.

2.5 BONUS Vzťah medzi žánrami navzájom

Pri pozorovaní heatmapy na obrázku č.15, ktorá bola vygenerovaná z dát filtered_genres sme si všimli, že existujú isté vzťahy aj medzi samotnými žánrami, preto sme si vytvorili ďalšie heatmapy, ktoré sa zamerali čisto len na žánre.

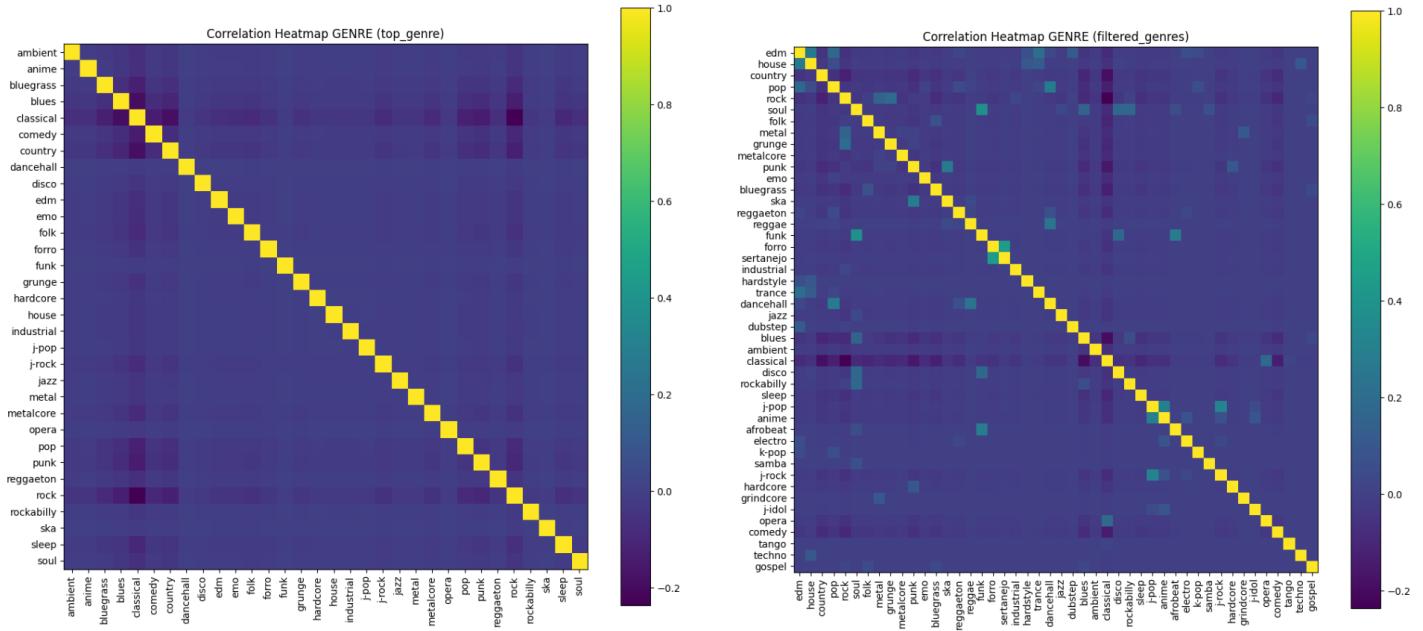


Figure 22: Heatmapy pre dataframe s top_genre a s filtered_genres

Na heatmape naľavo si môžeme všimnúť, že neexistujú žiadne korelácie medzi žánrami, to je zapríčinené tým, že dataframe pre danú heatmapu obsahoval len dátá zo stĺpca top_genre, to malo za príčinu, že každá pesnička mala len jeden žánier, zaitať čo pri druhom datarame s filtered_genres mohla mať pesnička viacero žánrov naraz. Na heatmape napravo sa nám podarilo nájsť pári žánrov, ktoré na prvý pohľad mohli spolu súvisieť. Následne sme si pre lepšiu vizualizáciu tieto žánre vyfiltrovali a zobrazili ich na novovygenerovanej mape.

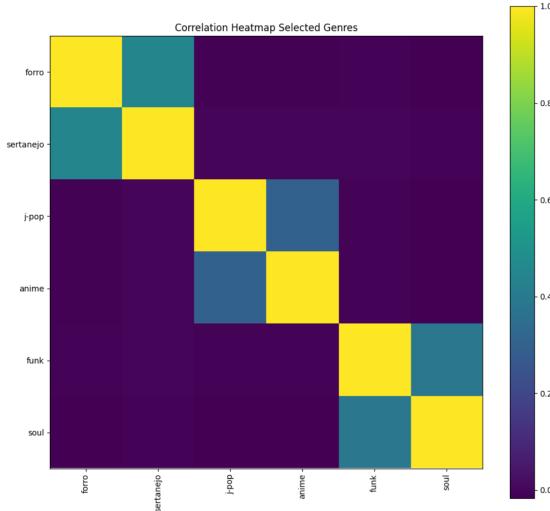


Figure 23: Heatmapa pre dataframe s filtered_genres

Z heatmapy vyplýva, že žánr funk je v korelácii so žánrom soul, ale nie je v korelácii so žánrom anime a tak podobne. Pre vizualizáciu samotnej korelácie dát sme sa rozhodli zobraziť percentuálne pomery pesničiek, kde sú dané žánre spolu, alebo zvlášť (buď sú úplne samé, alebo spojené s iným žánrom).

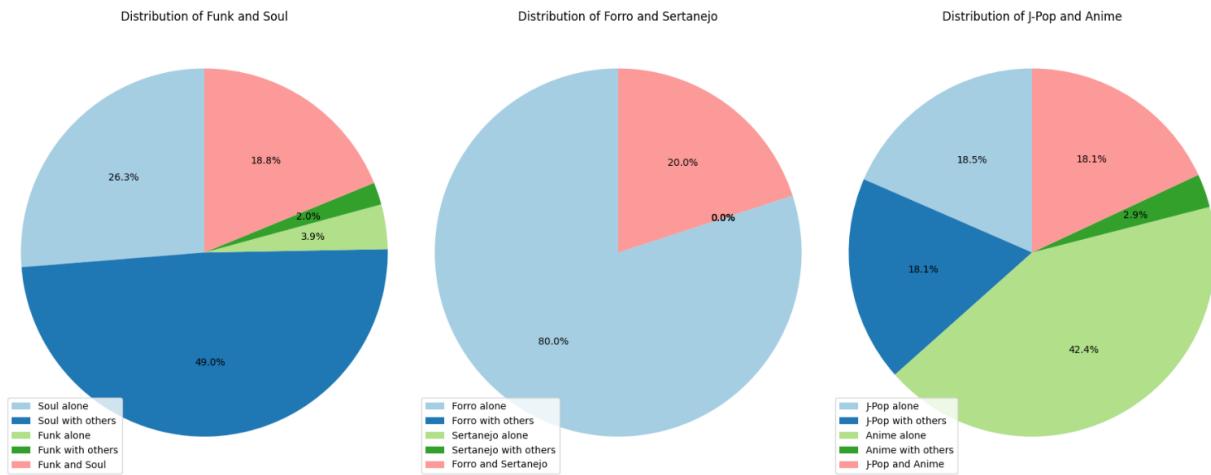


Figure 24: Koláčové grafy znázorňujúce vzťahy medzi žánrami

Koláčové grafy na obrázku č.24 znázorňujú vzťahy medzi žánrami. Fungovanie grafov si môžeme vysvetliť na konkrétnom príklade so žánrami Soul a Funk. V dataframe existuje 255 pesničiek, ktoré obsahujú aspoň jeden z týchto žánrov ($255=100\%$). 67 pesničiek ($67=26.3\%$) obsahuje čisto len Soul žáner. 155 pesničiek ($155=49.0\%$) obsahuje žáner Soul a iný žáner alebo žánre, ktoré nie sú Funk. Rovnakým spôsobom sú vypočítané aj informácie o žánre Funk bez žánru Soul. Žánre Soul a Funk sú spojované až v 48 prípadoch ($48=18.8\%$). Zaujímavejší vzťah je to však vtedy keď si uvedomíme, že Funk žáner takmer neexistuje bez žánru Soul. Z grafu sa dá jednoducho vyčítať, že ak je Funk spájaný s ľubovoľným žánrom alebo žánrami je obrovská pravdepodobnosť, že bude práve spájaný so žánrom Soul. Najväčší extrém je však zaznamenaný v druhom koláčovom grafe, kde si môžeme všimnúť že jeden žáner (Sertanejo) vôbec neexistuje bez druhého (Forro), jediný výskyt žánru Sertanejo je so žánrom Forro a zároveň Forro nie je taktiež v žiadnom inom vzťahu (buď je žáner sám, alebo spoločne so žánrom Sertanejo). Podobné vzťahy si vieme všimnúť aj na poslednom grafe medzi žánrami J-Pop a Anime.

2.6 BONUS Vzťah medzi žánrom comedy a speechiness

Posledný vzťah, na ktorý sme sa zamerali bol vzťah medzi žánrom comedy a speechiness.

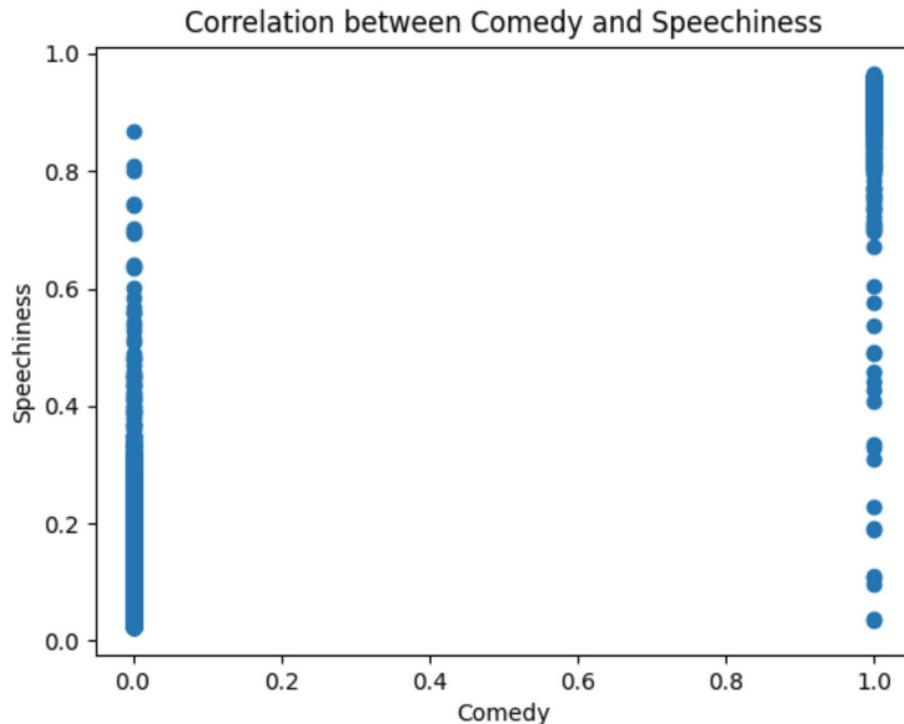


Figure 25: Korelačný graf Comedy Speechiness

Z grafu na obrázku č.25 vyplýva, že speechiness a comedy sú naozaj vo vzťahu. S rastúcou speechiness rastie aj počet pesničiek označených žánrom comedy. Pre lepšiu ilustráciu sme si nechali vygenerovať aj graf, ktorý podobne ako v prvom a druhom príklade pozostáva z intervalov a v daných intervaloch je vypočítaný priemerný výskyt žánru comedy.

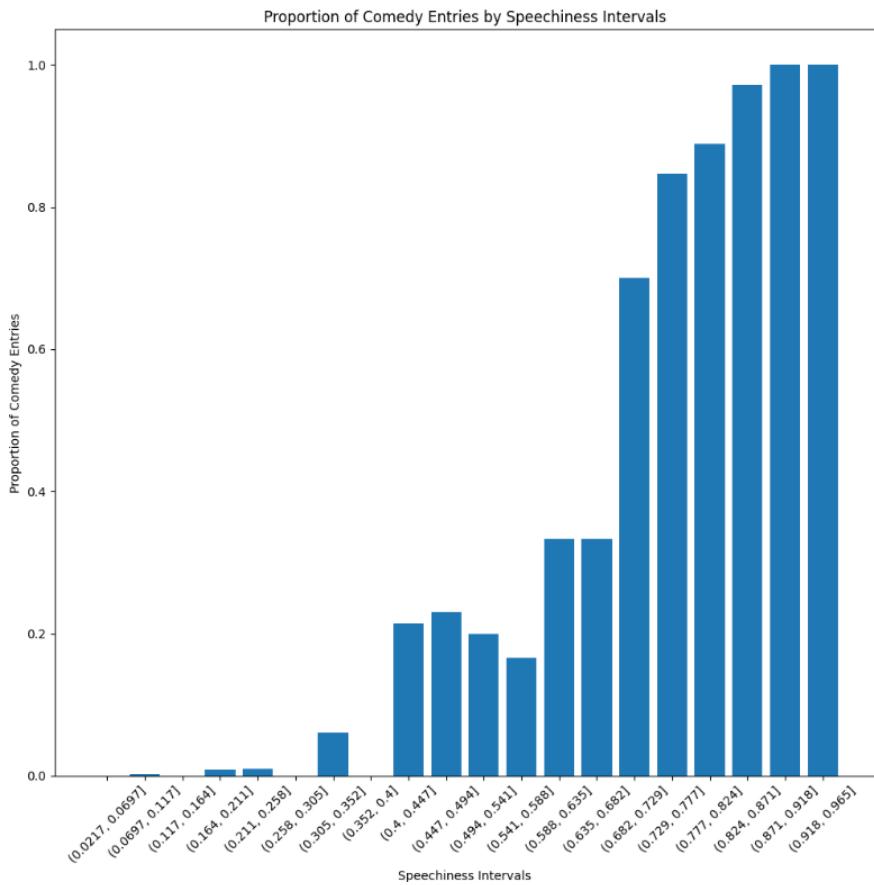


Figure 26: Pomer početu comedy žánru v intervaloch speechiness

Hodnoty na intervaloch potvrdili našu domnienku, z grafu je jasne vidieť že so stúpajcou speechiness rastie aj počet žánrov comedy. Za povšimnutie taktiež stojí fakt že od úrovne speechiness v hodnotách okolo 0.87 je takmer stopercentná pravdepodobnosť, že pesnička bude mať označenie comedy.

3 Tretia časť

3.1 Pretrénovanie siete

V tejto kapitole sme sa zamerali na nesprávne natrénovanú neurónovú sieť. Presnejšie išlo o jej pretrénovanie.

Hidden layers	Epochs	Early stopping patience	Solver	Learning rate
45,100,100,100,100,100	200	Early stopping nie je implementovaný	ADAM	0.0002

Table 4: Hyperparametre NS

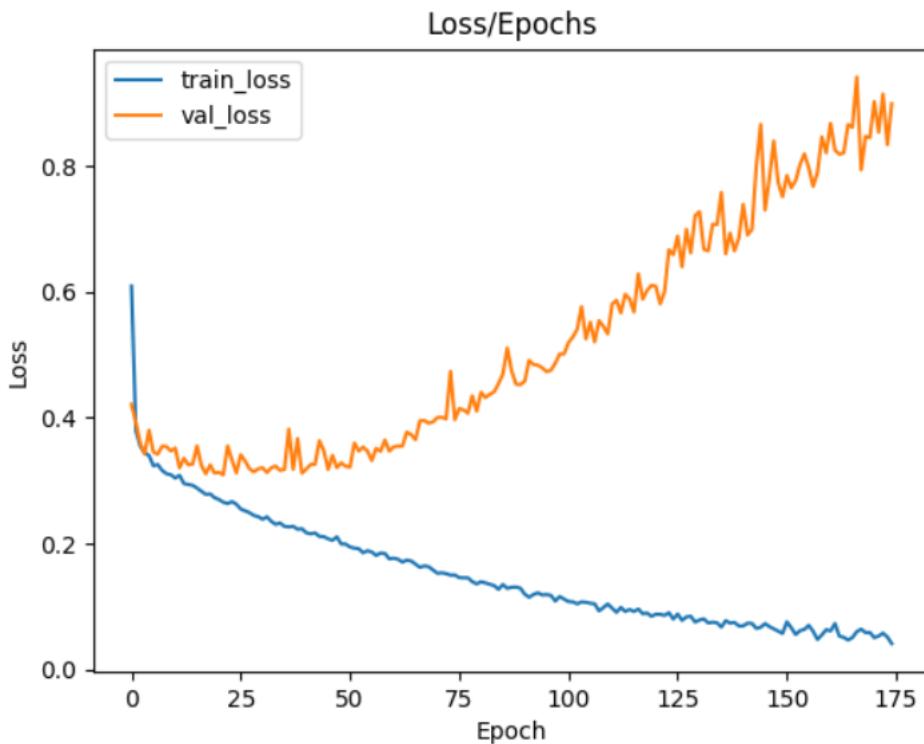


Figure 27: Zobrazenie pretrénovania na grafe

Na grafe môžeme vidieť chybovosť NS na základe epôch. Modré dátá reprezentujú trénovaciu časť a oranžové validačnú. Graf zobrazuje typický príklad pretrénovania, kde sa chyba na trénovacích dátach blíži k nule no na validačných dátach sa NS zhorsuje.

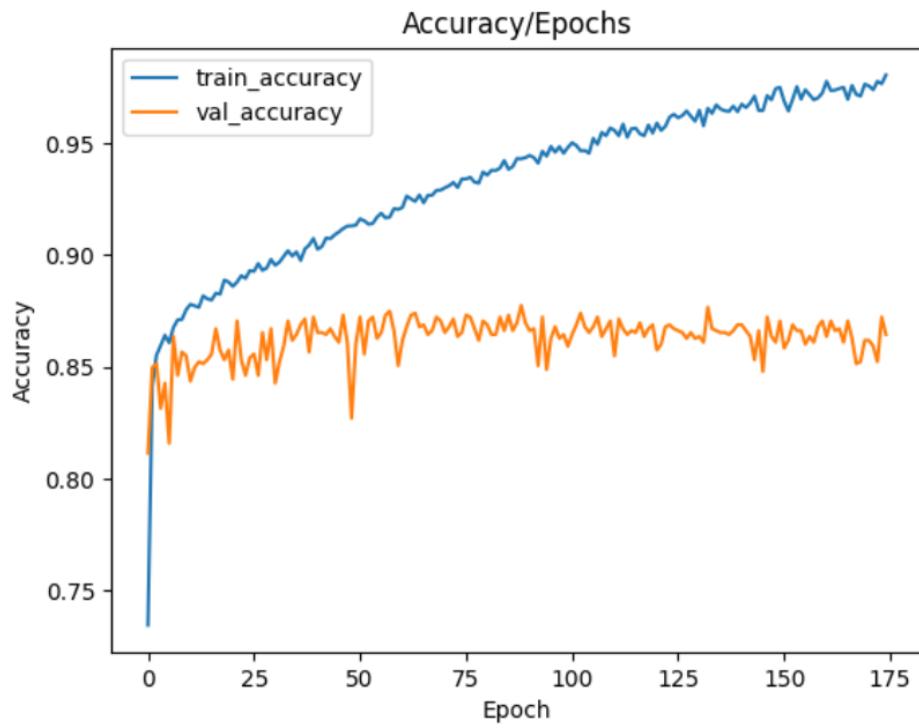


Figure 28: Zobrazenie pretrénovania na grafe

Obrázok č.28 taktiež dokazuje pretrénovanie siete, graf na obrázku predstavuje úspešnosť NS v danej epoche. Z grafu vyplýva, že NS dokáže na trénovacích takmer perfektne zaradiť pesničky do správnej kategórie emócií, ale na validačných dátach jej úspešnosť nestúpa. Obrázky č.28 a č.27 zobrazovali priebeh trénovania, preto sme pozorovali trénovacie a validačné dátá. V ďalšej časti sa pozrieme na hodnotenie NS pomocou trénovacích a testovacích dát.

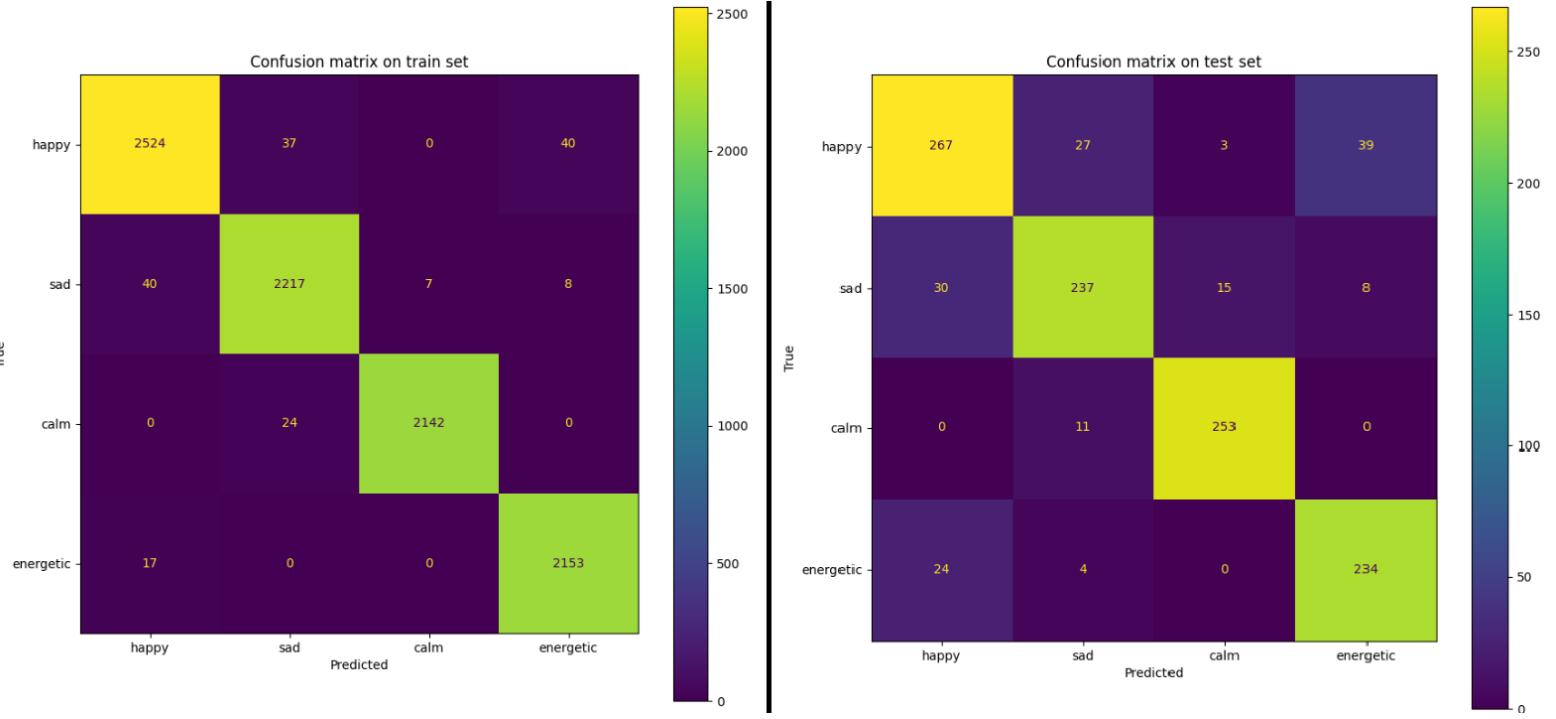


Figure 29: Konfúzne matice trénovacích/testovacích dát

Obrázky č.29 a č.30 taktiež naznačujú, že neurónová sieť mohla byť pretrénovaná, vidíme totiž veľký rozdiel úspešnosti medzi trénovacími a testovacími dátami a to dokonca až 12.1% (pri konfúznych maticiach vidíme vyšší pomer ne-správnych zatriedení medzi trénovacími a testovacími dátami).

Test accuracy: 0.8602

Train accuracy: 0.9812

Figure 30: Výsledky trénovania NS

3.2 Opravenie pretrénovania siete (early stopping)

V ďalšej časti sme sa následne venovali oprave pretrénovania. Pretrénovanie sme odstránili pomocou implemenzácie early-stoppingu. Metódu predčasného zastavenia sme nakonfigurovali tak, že ak po desiatich po sebe idúcich epochách

nedochádzalo k zlepšeniu výkonu na validačných dátach, trénovanie neurónovej siete bolo automaticky ukončené.

Hidden layers	Epochs	Early stopping patience	Solver	Learning rate
45,100,100,100,100,100	200	10	ADAM	0.0002

Table 5: Hyperparametre NS

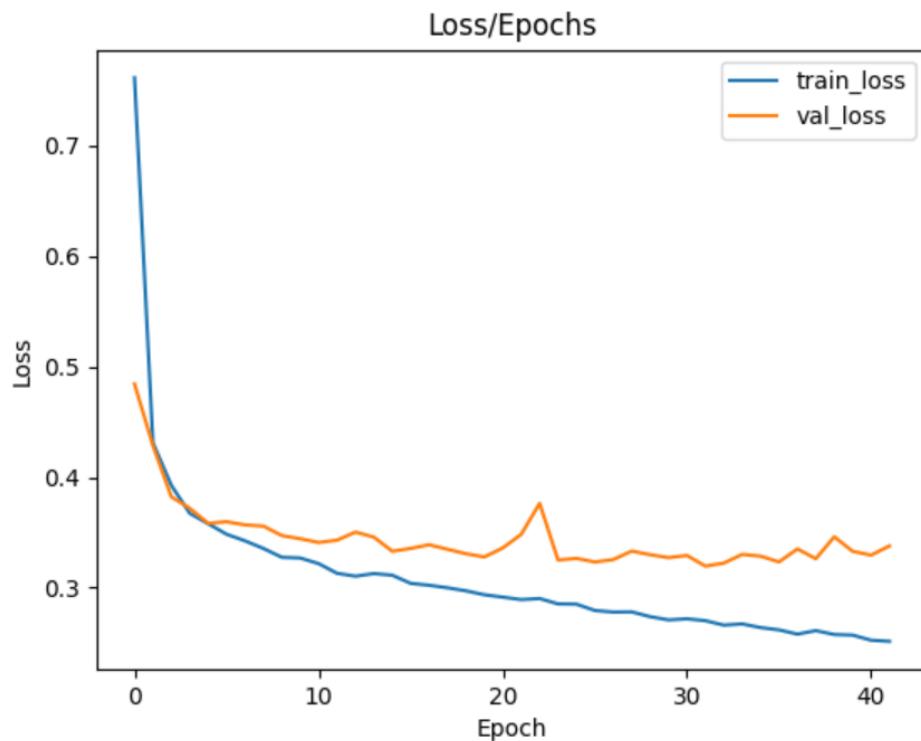


Figure 31: Zobrazenie early stoppingu na grafe

Na grafe na obrázku č.31 vidno, že sa trénovanie NS siete zastavilo približne po 40 epochách, to malo za následok, že sa nám NS nepretrénovala.

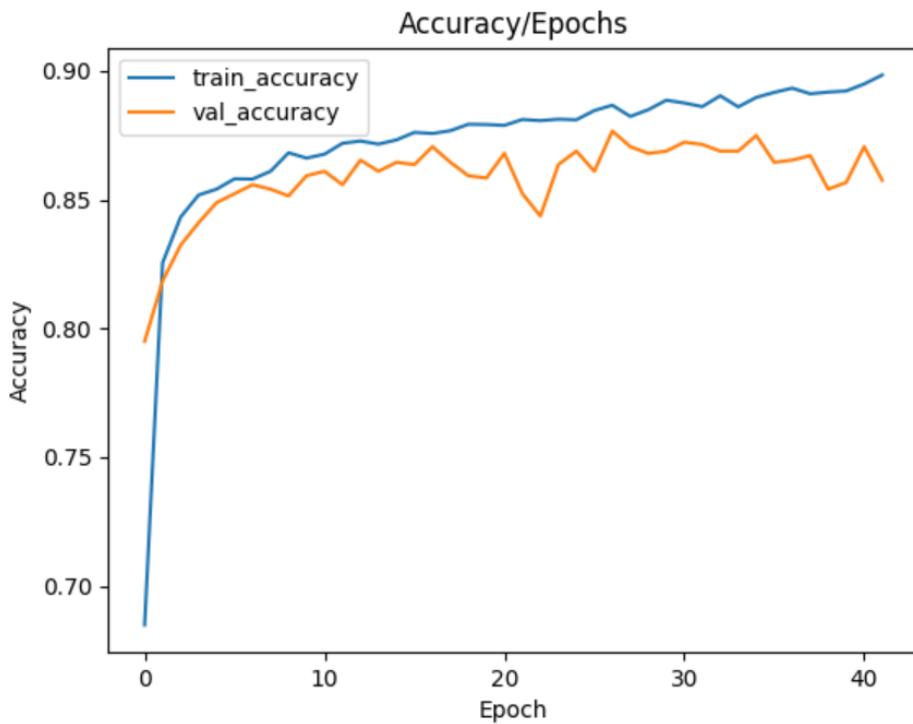


Figure 32: Zobrazenie early stoppingu na grafe

Na grafe na obrázku č.32 vidno, že sieť má súčasť menšiu úspešnosť na trénovacích dátach ako pretrénovaná sieť, ale už po 40 epochách má rovnakú na validačných. A z obrázku č.33 dokonca vyplýva, že nepretrénovaná sieť bola úspešnejšia na testovacích dátach ako pretrénovaná.

Test accuracy: 0.8759
Train accuracy: 0.8933

Figure 33: Výsledky trénovania NS

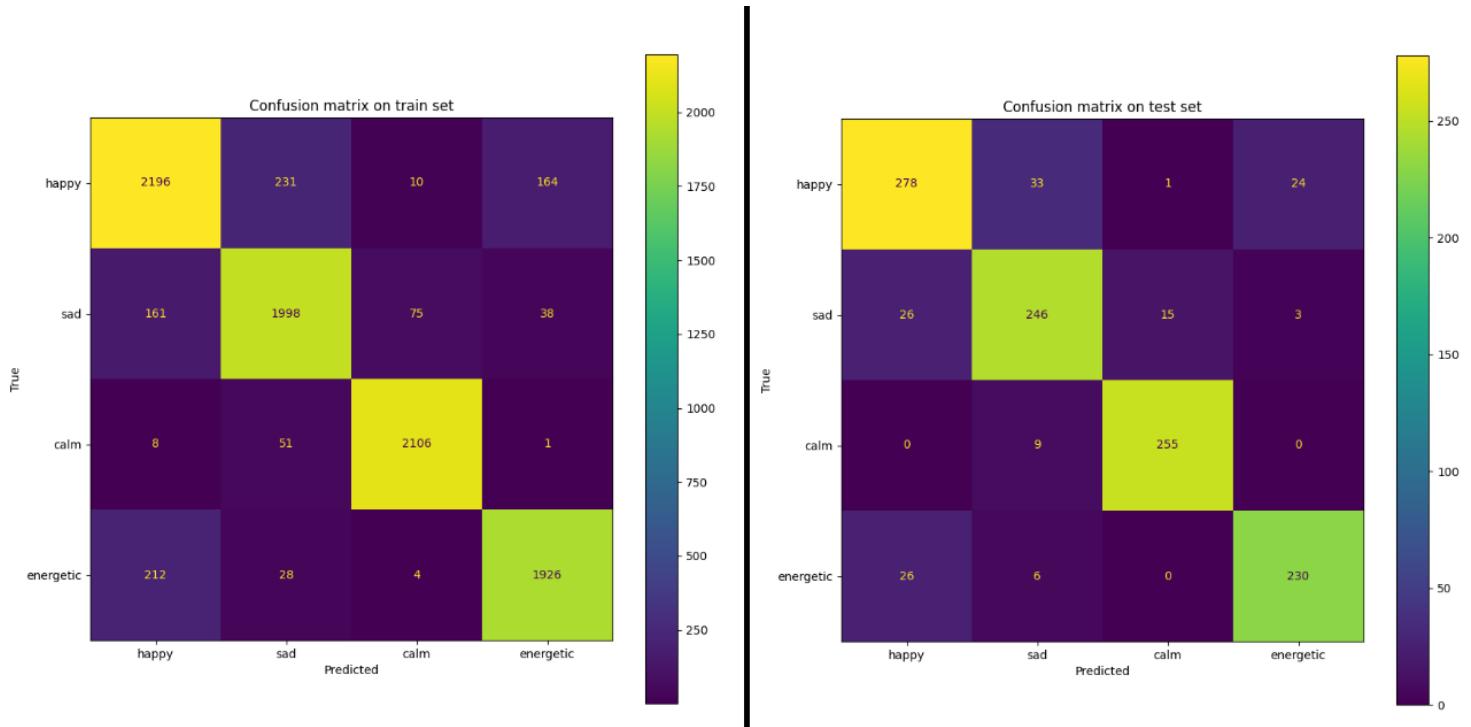


Figure 34: Konfúzne matice trénovacích/testovacích dát

Z konfúznych matíc na obrázku vyššie si môžeme všimnúť, že aj napriek tomu, že na trénovacích dátach nie je tak úspešná ako pretrénovaná sieť, tak na testovacích je na tom podobne (mierne lepšie).

3.3 Experimenty s trénovaním NS

V poslednej časti sme sa venovali experimentami s nastavovaním hyperparametrov. Následne sme si vytvorili tabuľku, kde sme dané experimenty zdokumentovali a vybrali najlepší a najhorší experiment z hľadiska úspešnosti. Pre vybrané experimenty sme následne vytvorili grafy priebehu trénovania a konfúzne matice.

Hidden layers	Epochs	Early stopping patience	Solver	Learning rate	Train data accuracy	Test data accuracy
45,100,100,100	200	5	ADAM	0.0002	89.05%	86.98%
45,100,100,100	200	15	ADAM	0.0002	90.84%	86.72%
45,100,100,100	200	15	ADAM	0.00005	89.26%	87.15%
45,25,25	200	10	ADAM	0.00005	89.99%	87.59%
45,25,25	200	10	SGD-M	0.00005	86.49%	86.37%
45,25,25	200	10	ADAM	0.000005	84.06%	83.33%

Table 6: Experimenty s hyperparametrami

3.3.1 Priebeh trénovania a výsledky najlepšieho experimentu

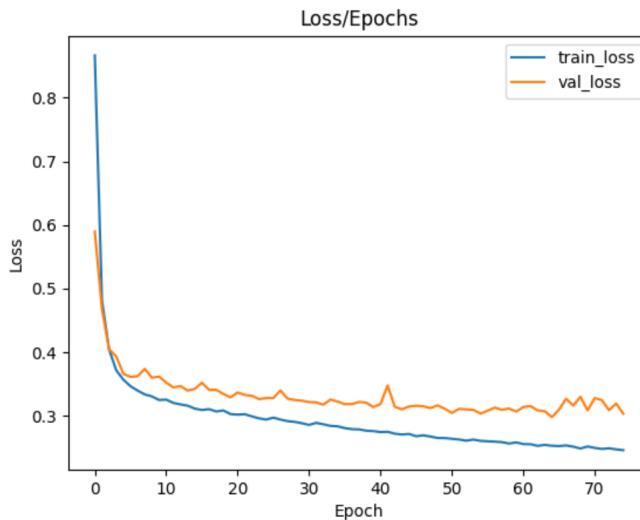


Figure 35: Priebeh trénovania NS

Na grafe na obrázku č.35 vidno, že sa trénovanie NS siete zastavilo približne po 75 epochách, to malo za následok, že sa nám NS nepretrénovala, ale zároveň mala dostatok času sa aj kvalitne natrénovala.

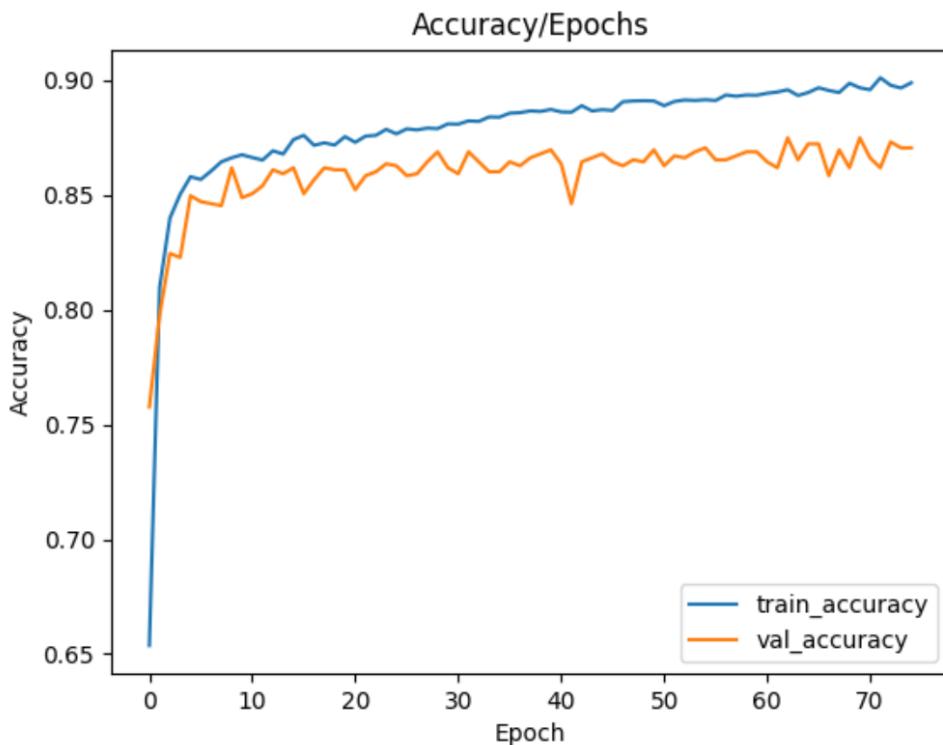


Figure 36: Priebeh trénovania NS

Na grafe na obrázku č.36 vidno úspešnosť NS na trénovacích a validačných dátach. Môžeme si všimnúť, že na rozdiel od grafu na obrázku č.28 rastú obe presnosti súbežne, zatiaľ čo na pretrénovanéj sieti už v neskorších epochách rástla úspešnosť len trénovacej množiny.

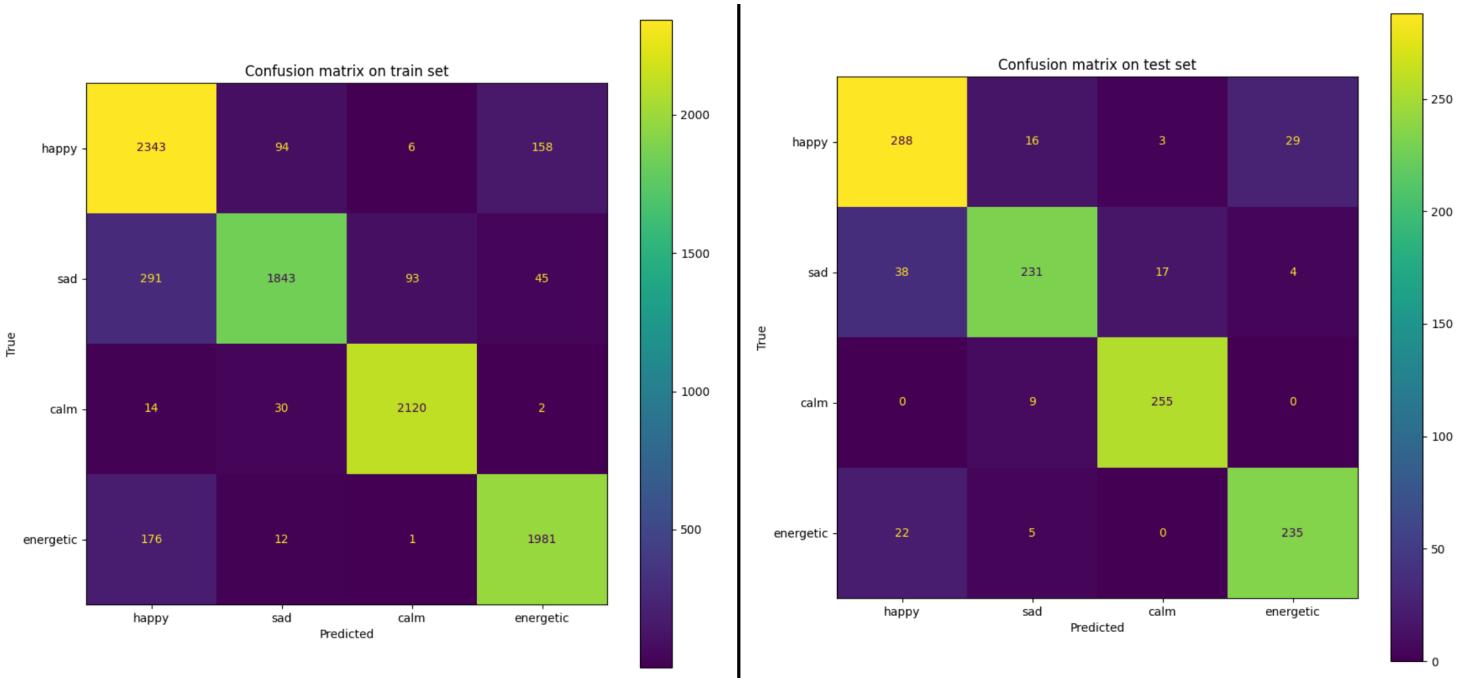


Figure 37: Konfúzne matice trénovacích/testovacích dát

Konfúzne matice na obrázku vyššie reprezentujú výsledky najlepšie natrénovanej NS z pomedzi 6tich experimentov.

```

1  early_stopping = EarlyStopping(monitor='val_loss',
2      patience=10, restore_best_weights=True)
3  model = Sequential()
4  model.add(Dense(45, input_dim=X_train.shape[1],
5      activation='relu'))
6  model.add(Dense(25, activation='relu'))
7  model.add(Dense(25, activation='relu'))
8  model.add(Dense(4, activation='softmax'))
9  model.compile(loss='categorical_crossentropy',
10     optimizer=Adam(learning_rate=0.0005),
11     metrics=['accuracy'])
12 history = model.fit(x=X_train, y=y_train, validation_data=(X_valid,
13     y_valid),
14     epochs=200, batch_size=30,
15     callbacks=[early_stopping])

```

3.3.2 Priebeh trénovania a výsledky najhoršieho experimentu

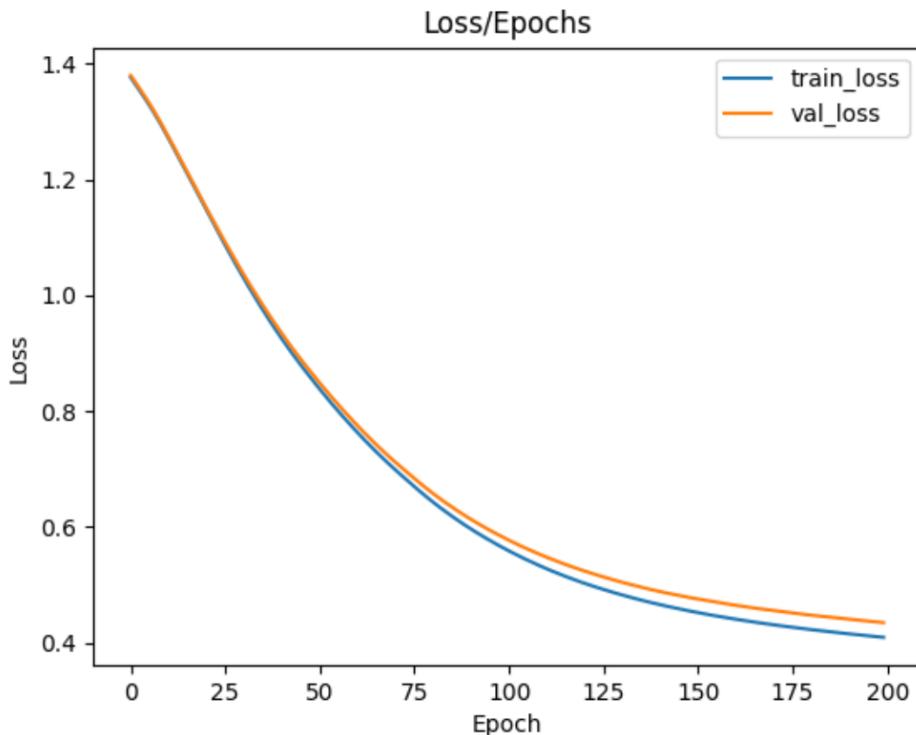


Figure 38: Priebeh trénovania NS

Na grafe na obrázku č.38 vidno, že sa trénovanie NS zastavilo až na 200 epoch, čo bolo maximálny počet povolených epoch pre NS na trénovanie. Taktiež si môžeme všimnúť aj pomerne veľkú chybovosť oproti ostatným NS. Môžeme teda skonštatovať, že spomedzi všetkých NS bola práve táto sieť podtrénovaná.

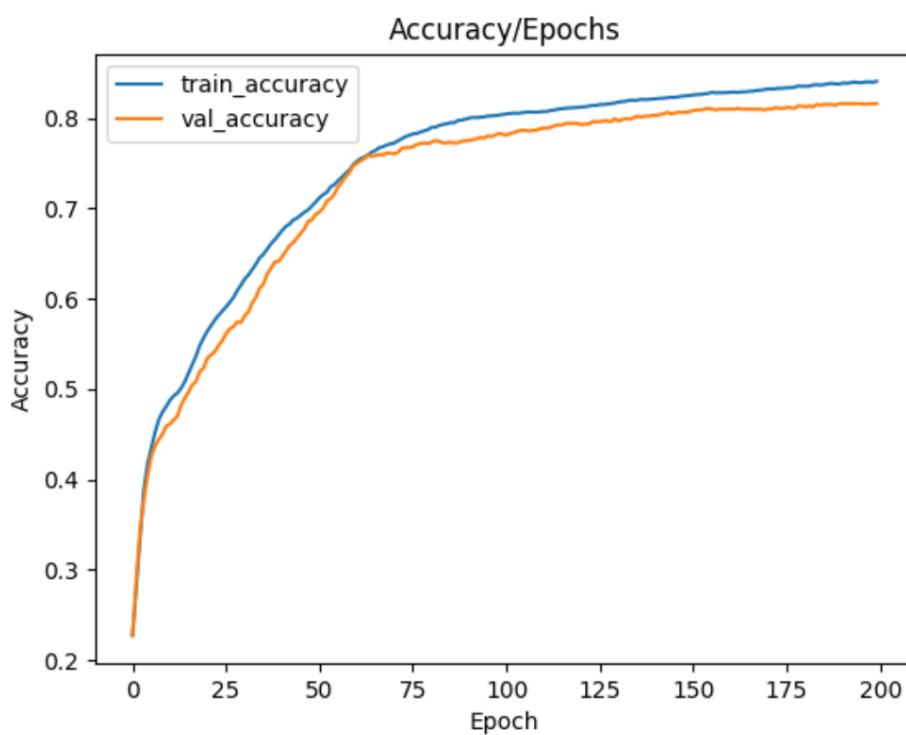


Figure 39: Priebeh trénovania NS

Na grafe na obrázku č.39 vidno, že sa trénovanie NS siete zastavilo až na 200 epoch, čo bolo maximálny počet povolených epoch pre NS na trénovanie. Taktiež si môžeme všimnúť aj pomerne nízku úspešnosť na trénovacích dátach oproti ostatným NS aj napriek tomu, že sieť využila všetkých 200 povolených epôch.

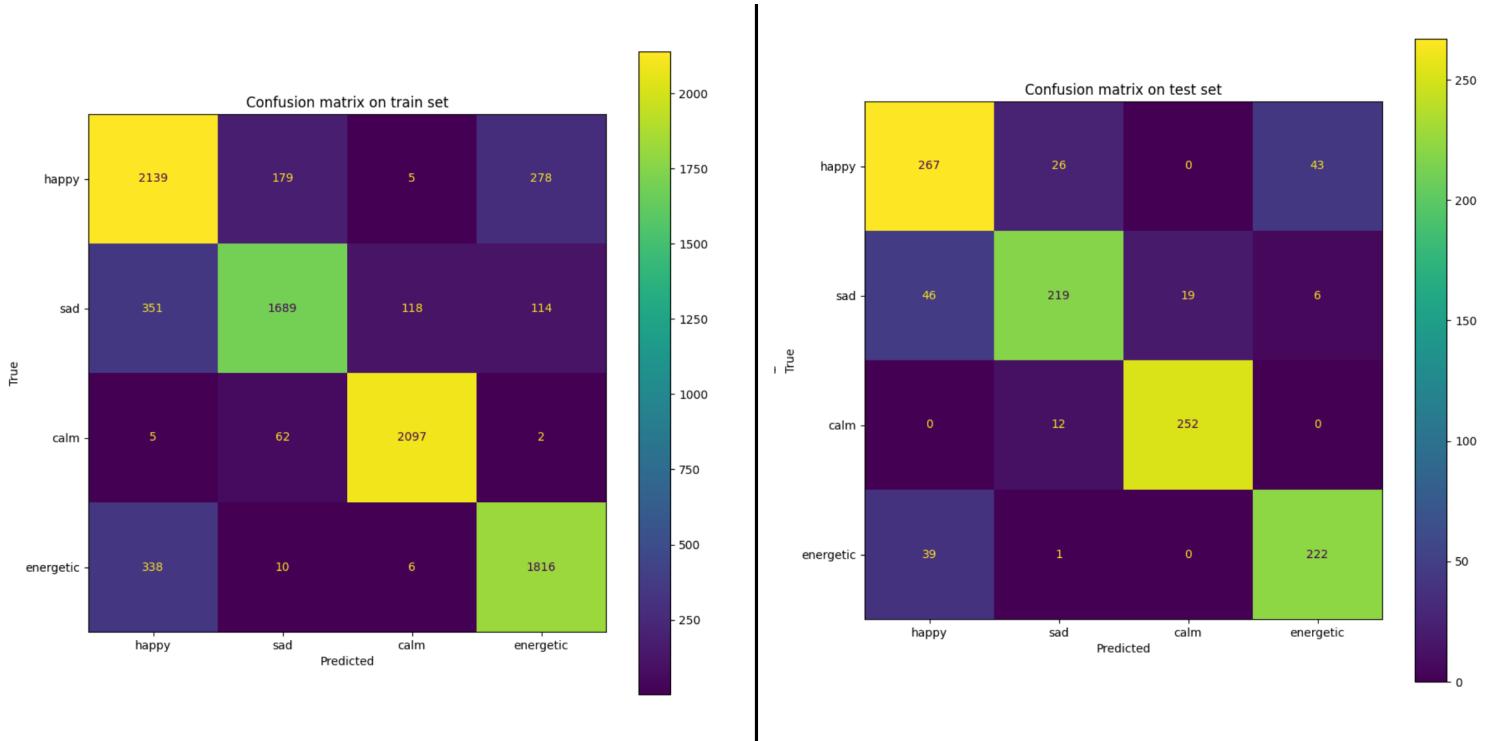


Figure 40: Konfúzne matice trénovacích/testovacích dát

Konfúzne matice na obrázku vyššie reprezentujú výsledky najhoršie natrénovanej NS z medzi 6tich experimentov.

```

1 early_stopping = EarlyStopping(monitor='val_loss',
2                               patience=10, restore_best_weights=True)
3 model = Sequential()
4 model.add(Dense(45, input_dim=X_train.shape[1],
5                 activation='relu'))
6 model.add(Dense(25, activation='relu'))
7 model.add(Dense(25, activation='relu'))
8 model.add(Dense(4, activation='softmax'))
9 model.compile(loss='categorical_crossentropy',
10               optimizer=Adam(learning_rate=learning_rate=0.000005),
11               metrics=['accuracy'])
12 history = model.fit(x=X_train, y=y_train
13                      ,validation_data=(X_valid, y_valid), epochs=200
14                      ,batch_size=30, callbacks=[early_stopping])

```

4 BONUS Gridsearch

V tejto bonusovej časti sme sa zaoberali Gridsearchom. Postupovali sme tak, že sme si najprv vytvorili kombinácie možností parametrov (hyperparametrov), ktoré sme následne nechali zanalyzovať a tak sme získali najoptimálnejšie riešenie. Následne sme si porovnali najlepšie a najhoršie parametre tým, že sme ich zadali NS a sledovali jej úspešnosť.

```
1 param_grid = {  
2     'hidden_layer_sizes': [  
3         (2,), (10,), (100,), # Single hidden layer  
4         (10, 10), (100, 100), # Two hidden layers  
5         (10, 10, 10), (100, 100, 100), # Three hidden layers  
6     ],  
7     'max_iter': [50, 100, 200], # Different max iterations  
8     'early_stopping': [True, False] # Early Stopping T/F  
9 }
```

```
Best Parameters: {'early_stopping': False, 'hidden_layer_sizes': (100, 100), 'max_iter': 200}  
Best score: 0.8776192452350905  
Worst Parameters: {'early_stopping': True, 'hidden_layer_sizes': (2,), 'max_iter': 50}  
Worst score: 0.6159860990443093
```

Figure 41: Vyhodnotenie parametrov a ich skóre

```
MLP accuracy on train set: 0.9312628949940276  
MLP accuracy on test set: 0.8793402777777778  
MLP accuracy on train set: 0.6215658594852861  
MLP accuracy on test set: 0.6345486111111112
```

Figure 42: Vyhodnotenie úspešnosti (najlepšie vs najhoršie parametre)

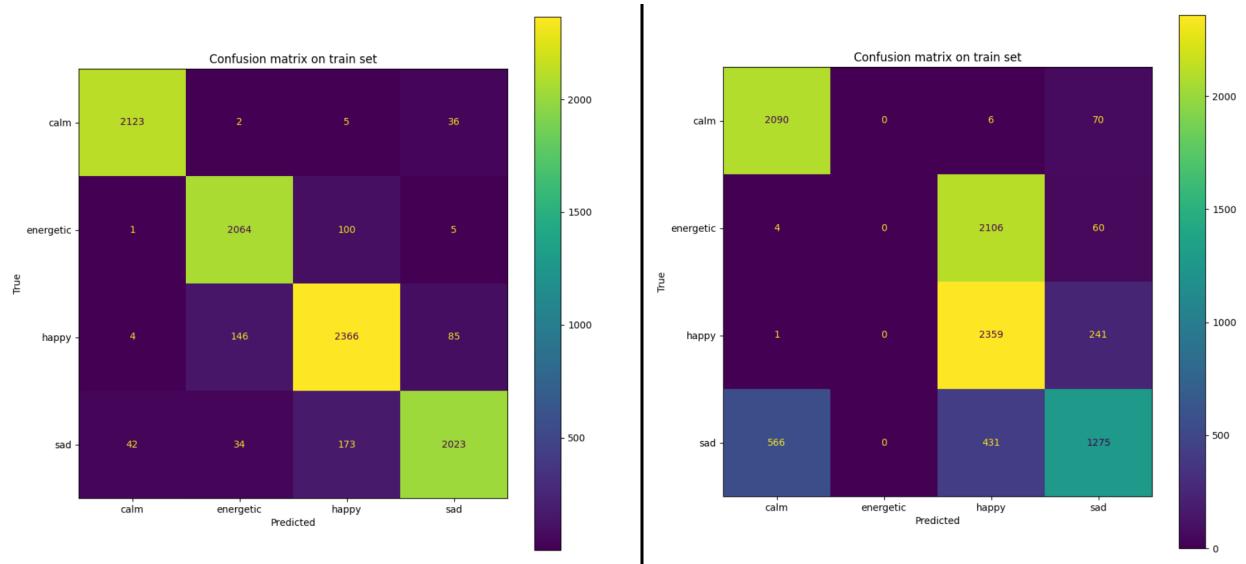


Figure 43: Konfúzne matice trénovacích dát(najlepšie vs najhoršie parametre)

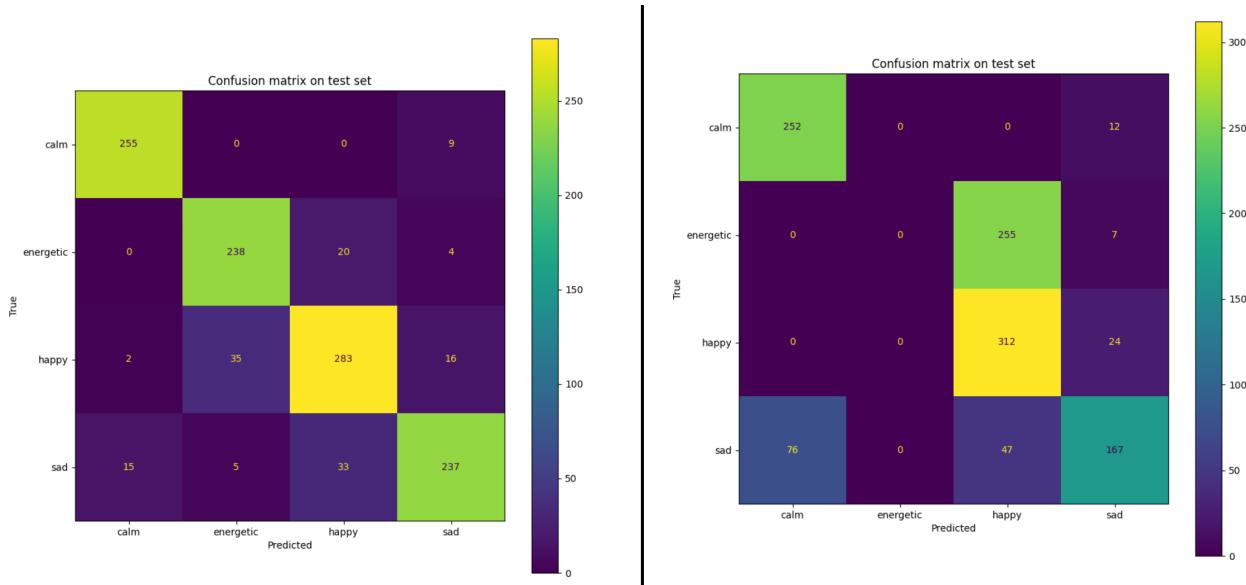


Figure 44: Konfúzne matice testovacích dát(najlepšie vs najhoršie parametre)

Z konfúznych matíc vyššie vieme pomerne jednoducho identifikovať, ktorá NS bola natrénovaná správne a ktorá nesprávne. Na záver môžeme teda konštatovať, že sa nám podarilo pomocou Gridsearchu nájsť najoptimálnejšie a najhoršie kombinácie parametrov (hyperparametrov).

5 BONUS Klasifikácia žánrov

V tejto časti sme sa venovali zaradeniu pesničiek do žánrov namiesto emócií. Pri riešení sme narazili na problém s nevybalancovaným dataframom. To malo za príčinu, že sa naša NS natrénovala len na správne rozpoznávanie párov najčastejších žánov a ostatné nevedela správne zaraďovať.

5.1 Nedokonalé experimenty

Prvé dva experimenty boli ponechanie originálneho dataframu a druhý jeho zmenšenie. V nasledujúcich porovnaniach si ukážeme výsledky pri trénovaní NS z dát bez úprav a následne NS natrénovala len na správne rozpoznávanie párov najčastejších žánov a ostatné nevedela správne zaraďovať.

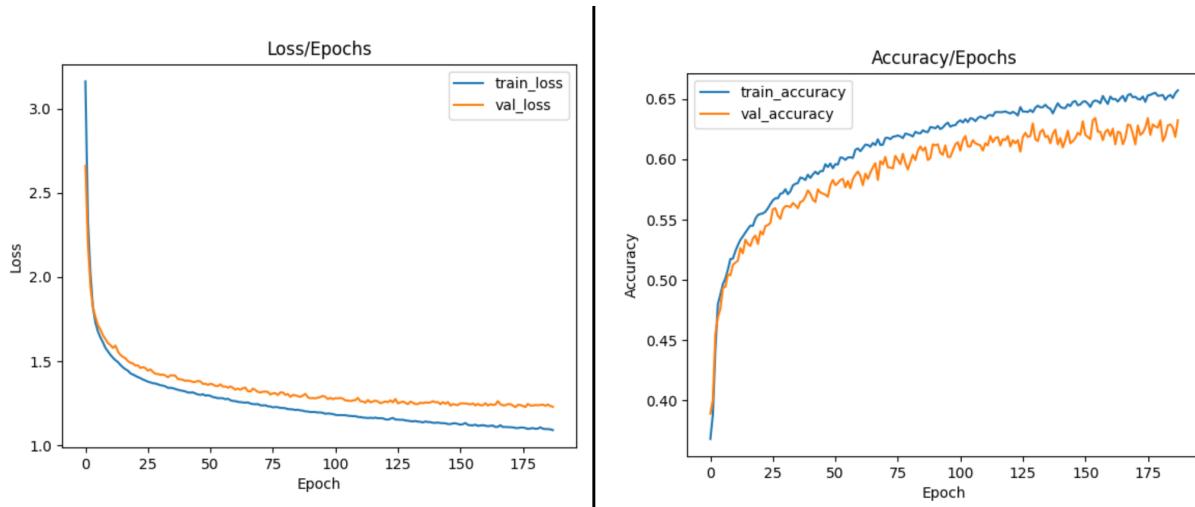


Figure 45: Priebeh trénovania (Originálny DF)

Z grafov, ktoré znázorňujú priebeh učenia vyplýva, že sieti sa nepodarilo správne naučiť. Vidíme totiž pomerne veľkú chybovosť a nízku presnosť.

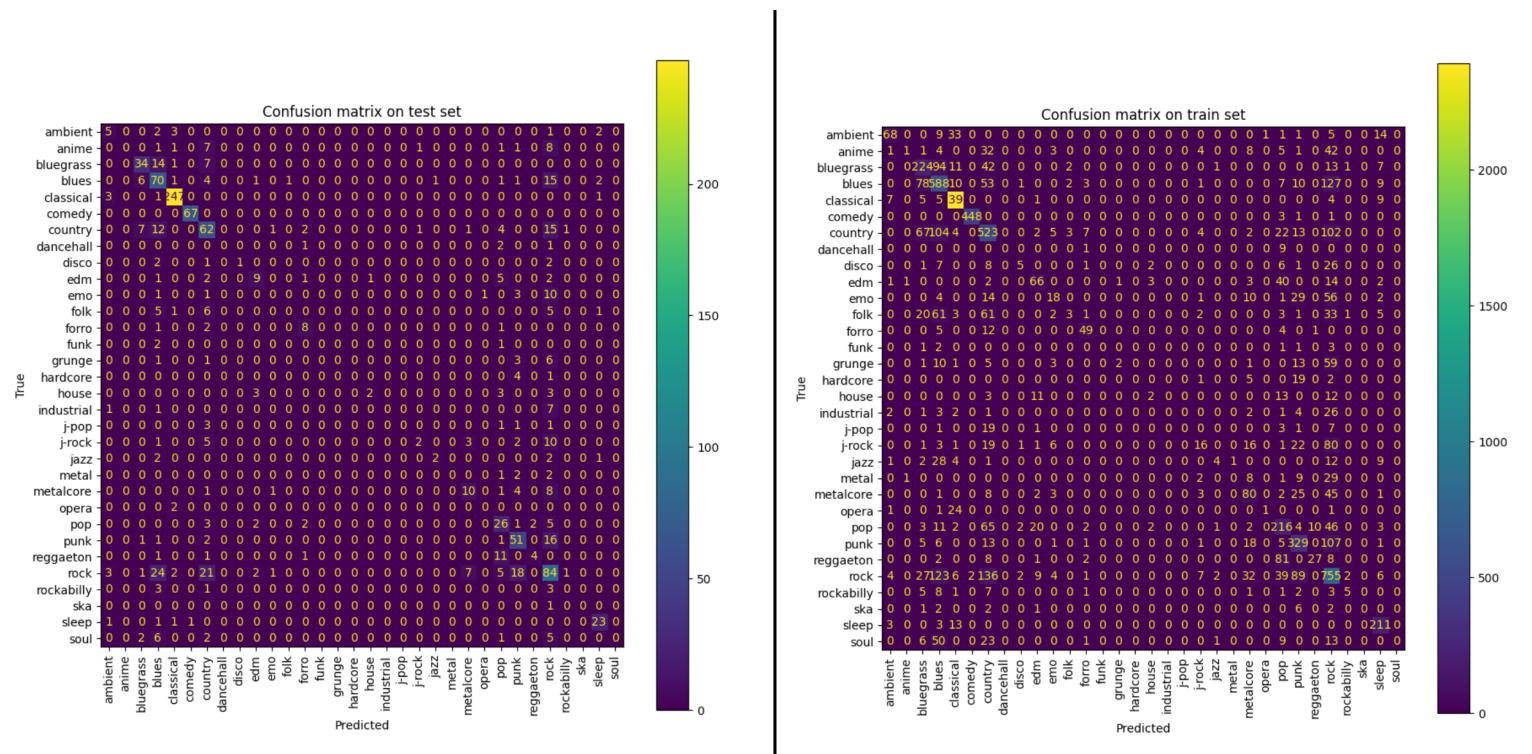


Figure 46: Konfúzne matice trénovacích/testovacích dát (Originálny DF)

Nesprávne natrénovanú sieť sme však vedeli odhaliť aj vďaka konfúznym maticiam, kde sme zistili, že sieť sa naučila rozpoznávať viac menej len jeden druh žánru.

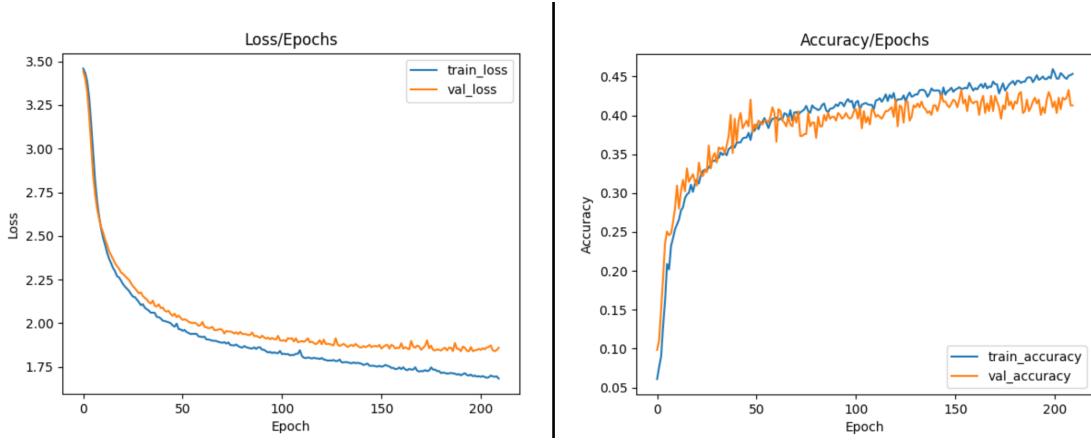


Figure 47: Priebeh trénovania (Zmenšený DF)

Z obrázkov č.47 a č.48 vyplýva, že ani tento experiment nebol úspešný, na grafoch trénovania vidno veľkú chybovosť a pomerne nízku presnosť.

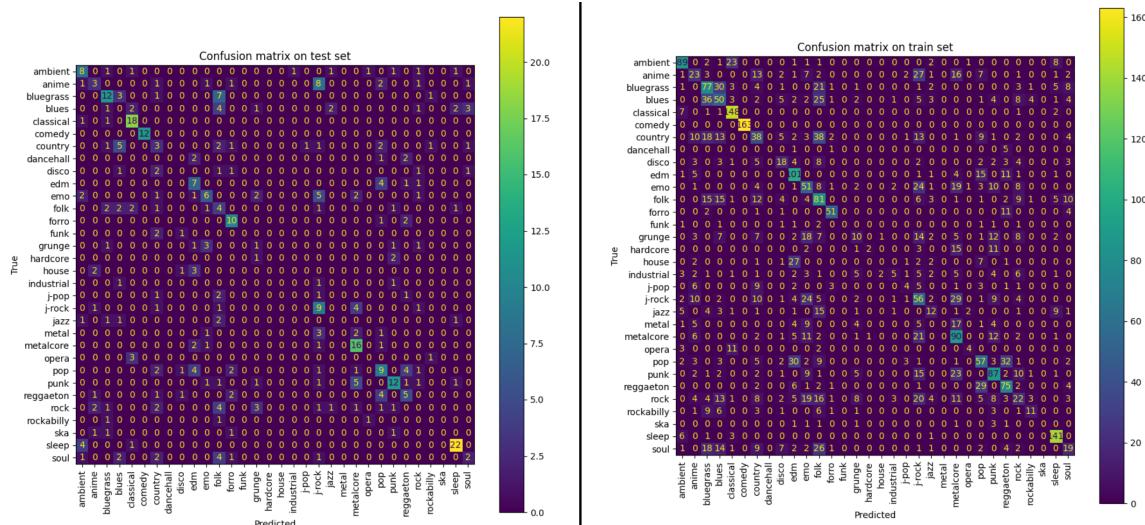


Figure 48: Konfúzne matice testovacích/trénovacích dát (Zmenšený DF)

Potvrdzujú to aj konfúzne matice, ktoré sú už majú "náznak" diagonály, ale aj tak považujeme odahdy NS za veľmi nepresné.

Experiment	Úspešnosť na trénovacích dátach	Úspešnosť na testovacích dátach
Originál	65.52%	61.37%
Zmenšený dataframe	45.61%	38.97%

Table 7: Úspešnosti NS

5.2 Najlepší experiment

Najlepší experiment sa nám podarilo vytvoriť za pomocí spojenia oversamplingu a zároveň undersemplingu. Vytvorili sme si upravený dataset, ktorý mal z každej kategórie žánrov 1000 riadkov (žánre, ktoré sa v datasete vyskytovali príliš často sme prefiltrovali a žánrom, ktoré sa v datasete vyskytovali pomenej, sme vytvorili ich vlastné kópie). Za spomenutie, taktiež stojí, že vo všetkých troch experimentoch bola NS trénovaná pomocou rovnakých hyperparametrov.

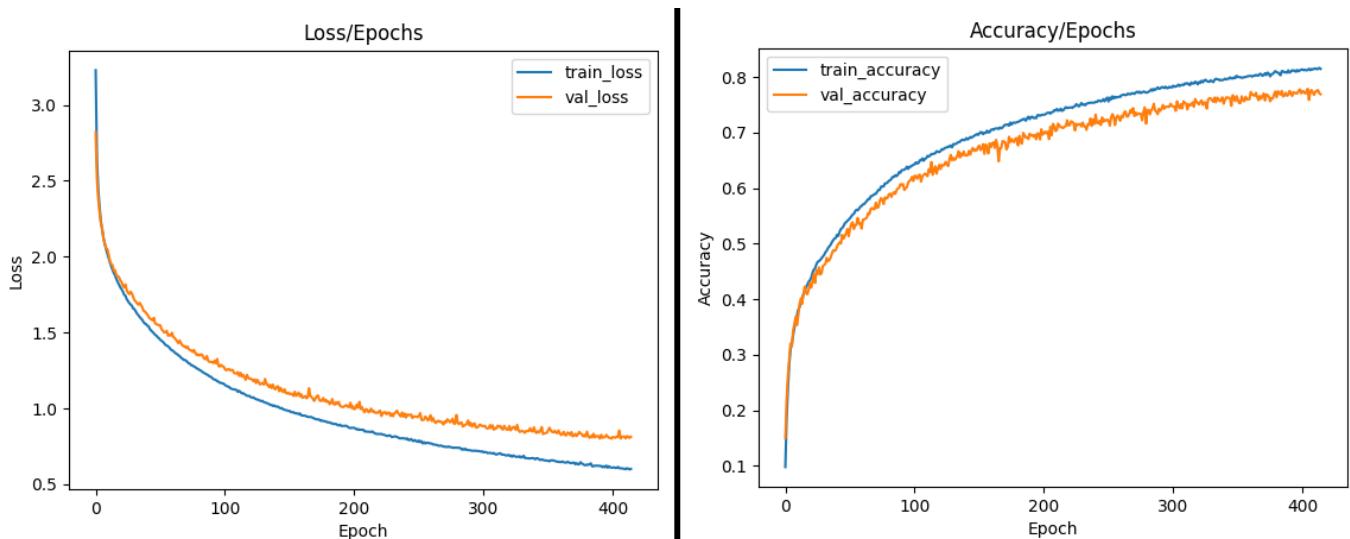


Figure 49: Priebeh trénovania NS

Na grafoch si môžeme všimnúť pomerne nízku chybovosť a aj vysokú presnosť na rozdiel od experimentu s filtrovaným datasetom (môžeme sa domnievať, že samotné filtrovanie nepomohlo, keďže aj tak bol dataframe nevybalanovaný).

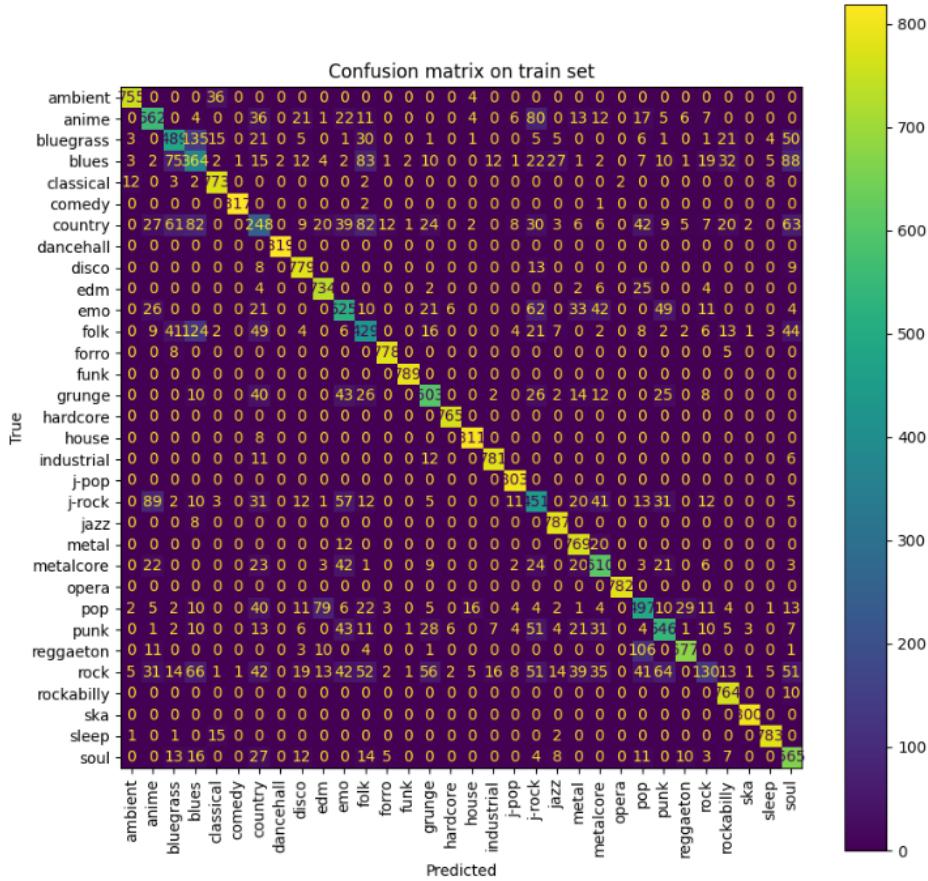


Figure 50: Konfúzna matica trénovacích dát

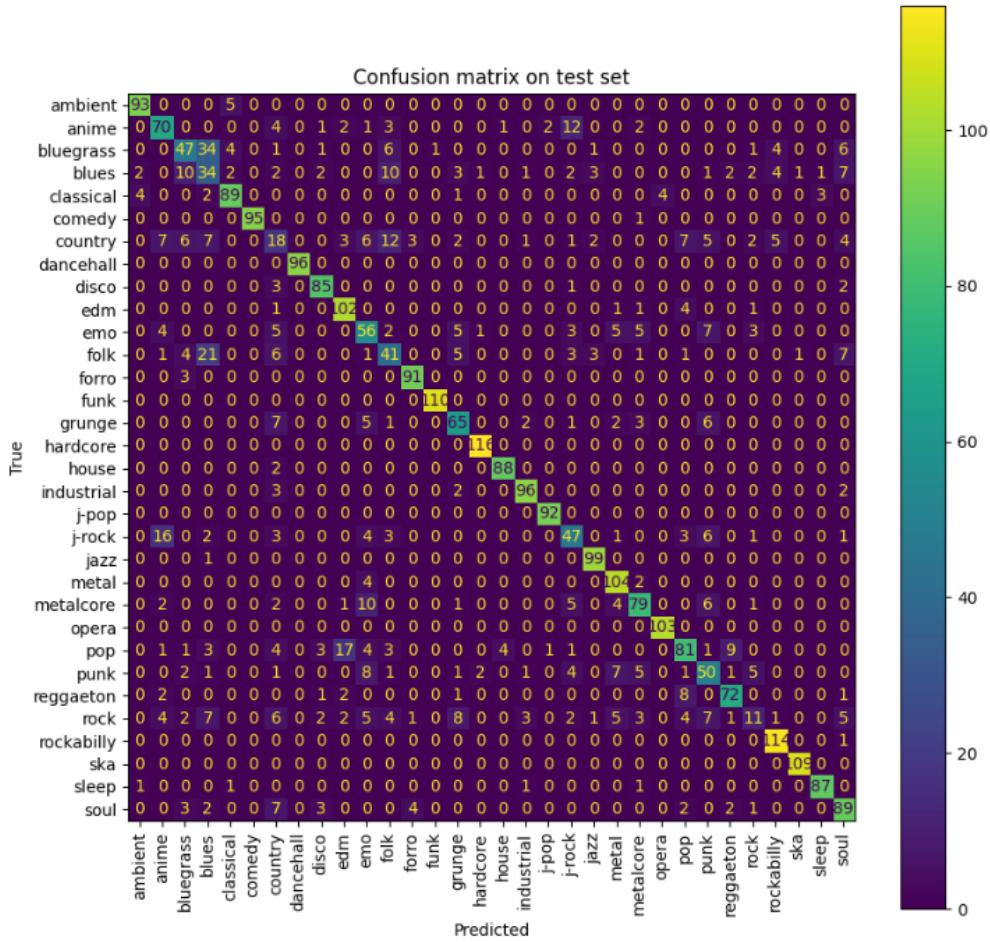


Figure 51: Konfúzna matica testovacích dát

Z obrázkov č.49, č.50 a č.51 vyplýva, že tento experiment bol úspešný, na grafoch trénovania vidno pomerne malú chybovosť a pomerne vysokú presnosť. Potvrdzujú to aj konfúzne matice, pri ktorých už môžeme jasne vidieť ich diagonály. Môžeme teda konštatovať, že sa nám podarilo správne natrénovať NS.

Experiment	Úspešnosť na trénovacích dátach	Úspešnosť na testovacích dátach
Originál	65.52%	61.37%
Zmenšený dataframe	45.61%	38.97%
Vybalancovaný dataframe	81.58%	79.03%

Table 8: Úspešnosti NS