Zadanie č.1

Neurónové siete

Autor: Adrián Somor

Úloha č.1: Predspracovanie, trénovanie neurónovej siete a vyhodnotenie 2
Predspracovanie dát 2
Odstránenie outlierov 2
Riešenie chýbajúcich hodnôt 2
Kódovanie nečíselných stĺpcov 3
Vytvorenie vstupných a výstupných dátových sád 3
Rozdelenie dát na trénovacie, validačné a testovacie množiny 3
Normalizácia a škálovanie dát 3
Trénovanie jednoduchej neurónovej siete 4
Vyhodnotenie 5
Úloha č.2: Analýza dát prostredníctvom EDA 6
Úvod 6
Korelačná matica 6
Scatter Plot 7
Barchart #1 8
Barchart #2 8
Úloha č.3: Priebeh trénovania a optimalizácia neurónovej siete 9
Parametre pretrénovania 9
Vyhodnotenie a demonštrácia priebehu pretrénovania 10
Zavedenie EarlyStopping-u 11
Vyhodnotenie a demonštrácia priebehu trénovania 11
Experimentálne ladenie hyperparametrov 13
Demonštrácia najhoršieho trénovania 14
Demonštrácia najlepšieho trénovania 15

Úloha č.1: Predspracovanie, trénovanie neurónovej siete a vyhodnotenie

Cieľom úlohy bolo spracovať a vyčistiť daný dátový súbor odstránením odchýliek, riešením chýbajúcich hodnôt a kódovaním nečíselných stĺpcov. Po príprave dát bolo potrebné vytvoriť vstupné a výstupné dátové sady, rozdeliť ich na trénovacie, validačné a testovacie množiny a následne dáta normalizovať alebo škálovať. Potom som pomocou týchto dát natrénoval jednoduchú neurónovú sieť. Nakoniec som vyhodnotil natrénovanú sieť na trénovacej aj testovacej množine.

Predspracovanie dát

Aby som súbor údajov zjednodušil, niektoré stĺpce, ktoré nie sú nevyhnutné pre ďalšie spracovanie alebo modelovanie boli vynechané.

URL aj názov sú jedinečné identifikátory, preto neprispievajú k rozoznateľným modelom. Stĺpce týkajúce sa žánrov, konkrétne *genres*, *filtered_genres* a *top_genre*, zavádzajú kategoriálnu zložitosť. Pokiaľ nie je klasifikácia žánrov naším explicitným cieľom, je praktickejšie ich vynechať.

Odstránenie outlierov

Outlieri môžu skresliť výsledky našej analýzy a viesť k nesprávnym záverom.

danceability: Očakávaný rozsah je medzi 0 a 1, ale existujú hodnoty až do 8.375.
loudness: Očakávaný rozsah je medzi -60 dB a 0 dB, ale sú hodnoty až do 1.519 dB.
duration: Očakávaný rozsah sú pozitívne čísla.

Riešenie chýbajúcich hodnôt

V dátových množinách môžu byť prázdne hodnoty (null), ktoré môžu ovplyvniť výsledky analýzy.

Najprv som si zobrazil počet prázdnych hodnôt v každom stĺpci. Na základe výstupu som pozoroval , že stĺpec *popularity* obsahuje 116 prázdnych hodnôt. Rozhodol som sa odstrániť všetky riadky s prázdnotami v tomto stĺpci, keďže dataset mal pôvodne 11825 riadkov. To isté so stĺpcom *number_of_artists*, ktorý obsahoval 120 null hodnôt. Týmto krokom som zabezpečil, že dataset neobsahuje žiadne prázdne hodnoty v stĺpcoch *popularity* a *number_of_artists*, čo robí dáta konzistentnejšími.

Kódovanie nečíselných stĺpcov

V analýze dát je často potrebné previesť kategorické (nečíselné) hodnoty na číselné formáty, aby boli kompatibilné s mnohými algoritmami strojového učenia. Najprv som identifikoval nečíselné stĺpce v datasete.

Stĺpec *emotion* je kategorický a nemá prirodzené usporiadanie. Preto som sa rozhodol použiť metódu "one hot encoding" na tento stĺpec. Okrem toho som stĺpec *explicit*, ktorý je kategorický, ale môže byť reprezentovaný číselne (pravda/hodnota nepravdy), premenil na int formát.

Vytvorenie vstupných a výstupných dátových sád

Najprv som oddelil stĺpce reprezentujúce emócie (výstupné hodnoty) od zvyšku datasetu.

```
X = df.drop(columns=['emotion_calm', 'emotion_energetic', 'emotion_happy',
'emotion_sad'])
y = df[['emotion_calm', 'emotion_energetic', 'emotion_happy', 'emotion_sad']]
```

Rozdelenie dát na trénovacie, validačné a testovacie množiny

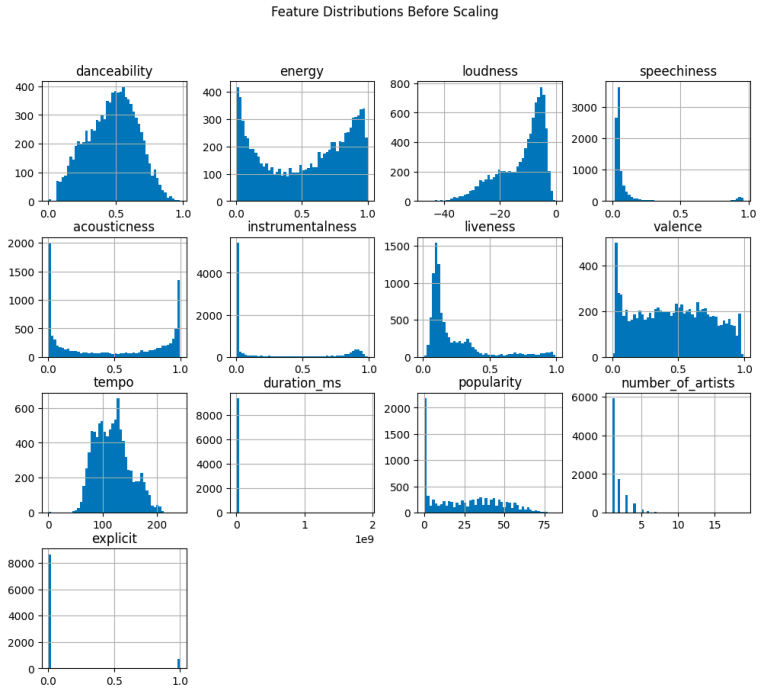
Potom som rozdelil dáta na trénovacie, validačné a testovacie množiny v pomere 8:1:1

```
X_train, X_test, y_train, y_test = train_test_split(X, y, shuffle=True, test_size=0.2,
random_state=69)
X_valid, X_test, y_valid, y_test = train_test_split(X_test, y_test, shuffle=True, test_size=0.5,
random_state=69)
```

Normalizácia a škálovanie dát

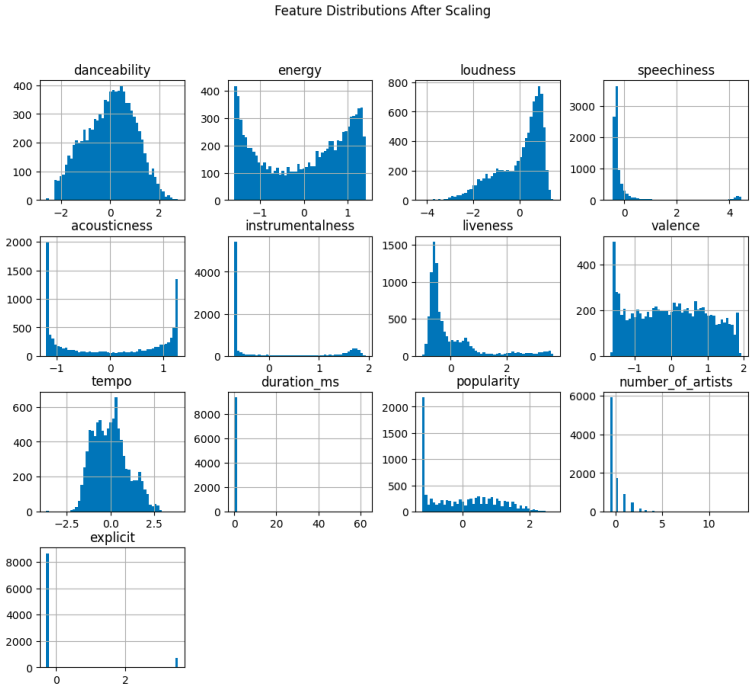
Pred trénovaním modelu je dôležité, aby mali všetky features rovnaký vplyv na model. Jedným zo spôsobov, ako to dosiahnuť, je škálovanie alebo normalizácia.

Najprv som sa pozrel na distribúciu features pred škálovaním



Potom som škáloval dáta pomocou metódy StandardScaler, ktorá transformuje dáta tak, aby mali priemernú hodnotu 0 a štandardnú odchýlku 1. Následne som premenil numpy polia na pandas DataFrames.

Na konci som sa znova pozrel na distribúciu features po škálovaní



Trénovanie jednoduchej neurónovej siete

Pre analýzu a klasifikáciu našich dát som sa rozhodol použiť jednoduchú neurónovú sieť. K tomu som použil triedu MLPClassifier z knižnice sklearn.

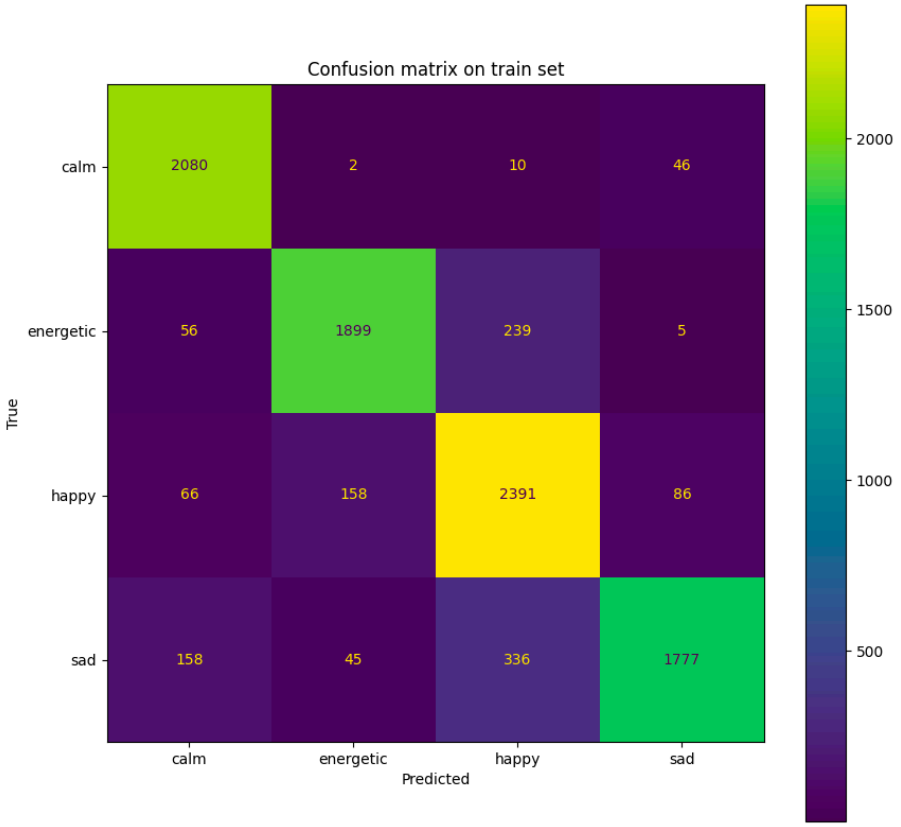
Pri definovaní modelu som nastavil nasledovné parametre:

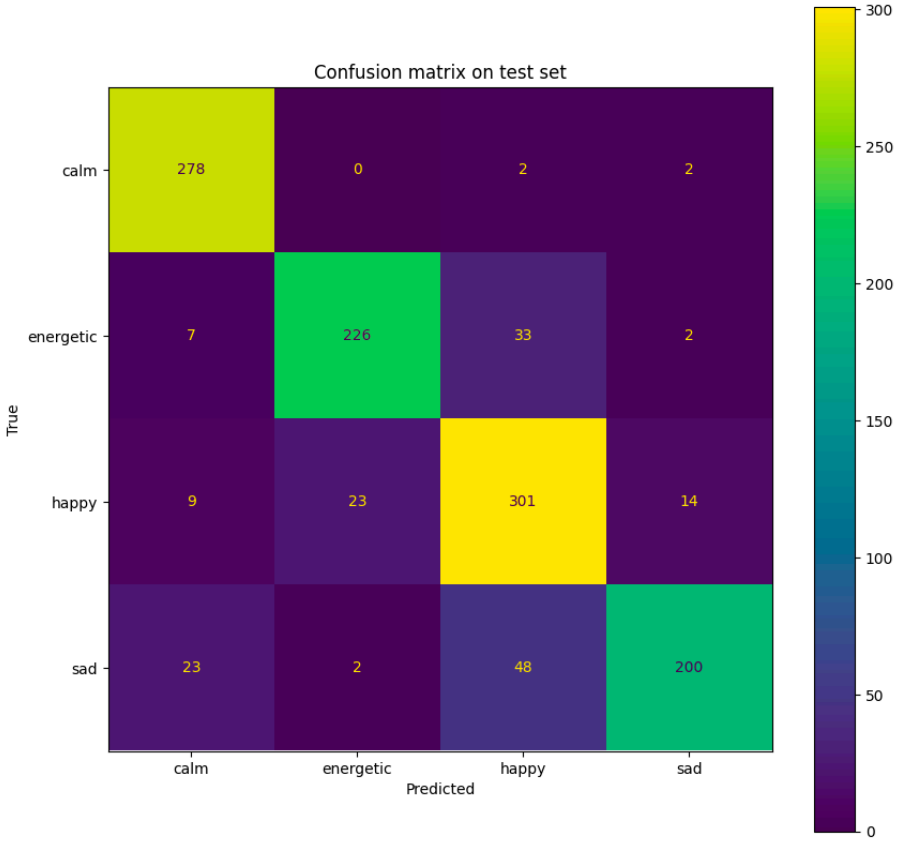
- **hidden_layer_sizes=(128, 128, 128):** Špecifikoval som tri skryté vrstvy, pričom každá vrstva má 128 neurónov.
- **random_state=1:** Zabezpečuje, že výsledky budú konzistentné pri opakovanom spustení. To je užitočné pre reprodukovateľnosť experimentu.
- **validation_fraction=0.2:** Určuje, koľko dát bude vyhradených ako validačná sada z trénovacích dát. Validačné dáta sú použité na monitorovanie výkonnosti modelu počas trénovania a na zastavenie trénovania, ak model prestane zlepšovať svoju výkonnosť.
- **early_stopping=True:** Ak model po určitom počte iterácií nezlepší svoju výkonnosť na validačnej sade, trénovanie sa zastaví. Toto zabráni pretrénovaniu.
- **learning_rate='adaptive':** Automaticky upravuje rýchlosť učenia sa počas trénovania. Ak výkonnosť modelu nezlepší, rýchlosť učenia sa zmenší.
- **learning_rate_init=0.0005:** Počiatočná rýchlosť učenia.

Následne som model natrénoval na trénovacích dátach pomocou metódy fit.

Vyhodnotenie

MLP accuracy on train set: 0.8709642933504383





MLP accuracy on test set: 0.8589743589743589

Úloha č.2: Analýza dát prostredníctvom EDA

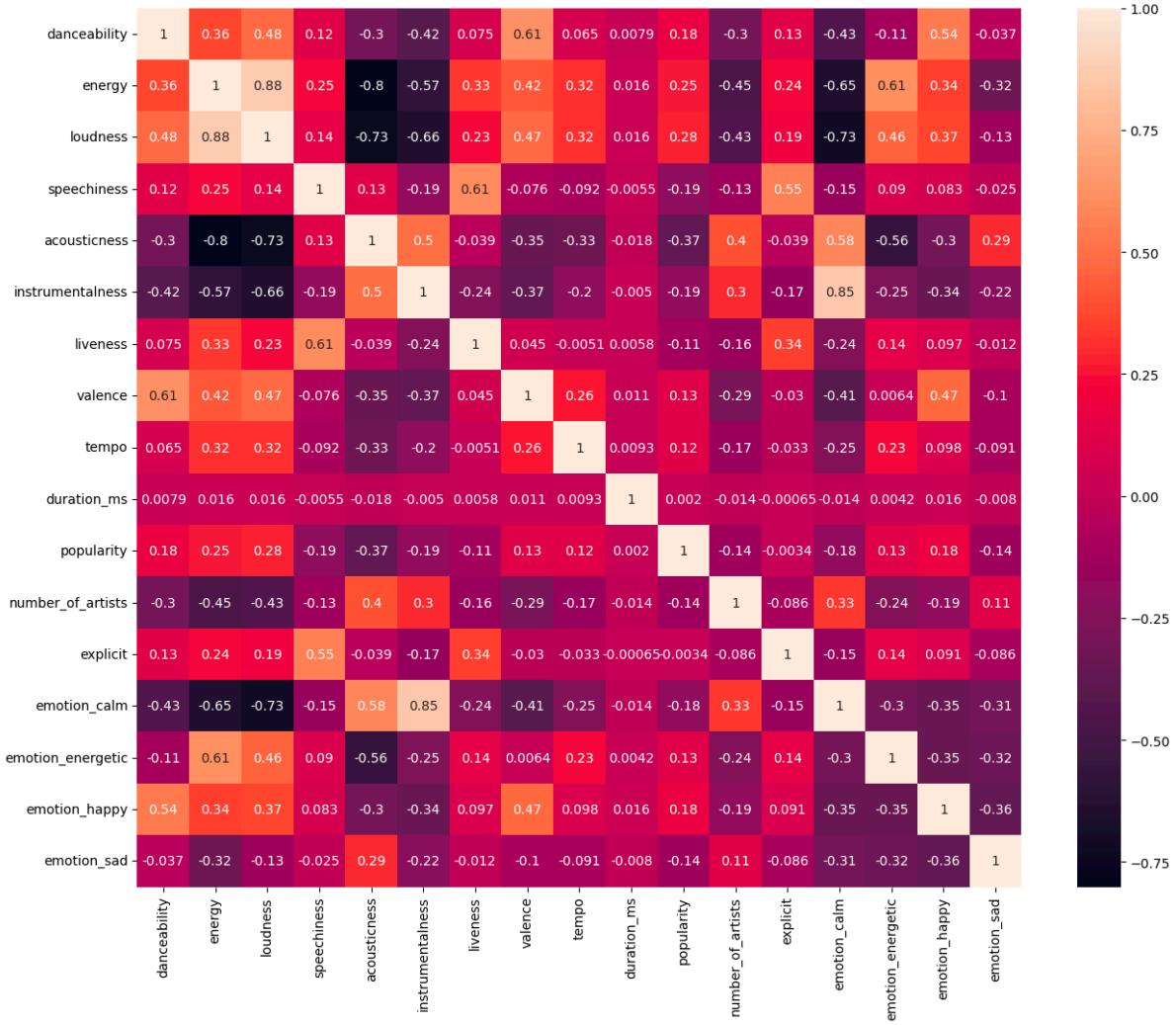
Cieľom tejto úlohy bola EDA datasetu. Sústredil som sa na analýzu oboch verzií dát - pôvodných a upravených. Identifikoval a vizualizoval som päť zaujímavých vzťahov v dátach. Každý graf zahŕňa slovný popis.

Úvod

Na lepšie pochopenie štruktúry a obsahu datasetu som zobrazil prvých niekoľko riadkov dát pomocou funkcie head(). Následne som sa pozrel na typy dát a počet nulových hodnôt v datasete. Použil som funkciu info(), ktorá poskytla prehľad o typoch dát v každom stĺpci a o tom, koľko nechýbajúcich hodnôt obsahuje každý stĺpec. Týmto spôsobom som bol schopný získať prehľad o potenciálnych problémoch s kvalitou dát ešte predtým, než som sa pustil do EDA.

Korelačná matica

Načítal som korelačnú maticu dát a následne som ju vizualizoval pomocou heatmap s využitím knižnice Seaborn, aby som lepšie pochopil vzťahy medzi rôznymi premennými v datasete.



- **Energia a Hlasitosť:** Existuje vysoká pozitívna korelácia (0,88) medzi energiou a hlasitosťou skladieb. Toto naznačuje, že skladby, ktoré sú hlasnejšie, sú vo všeobecnosti energetickejšie.

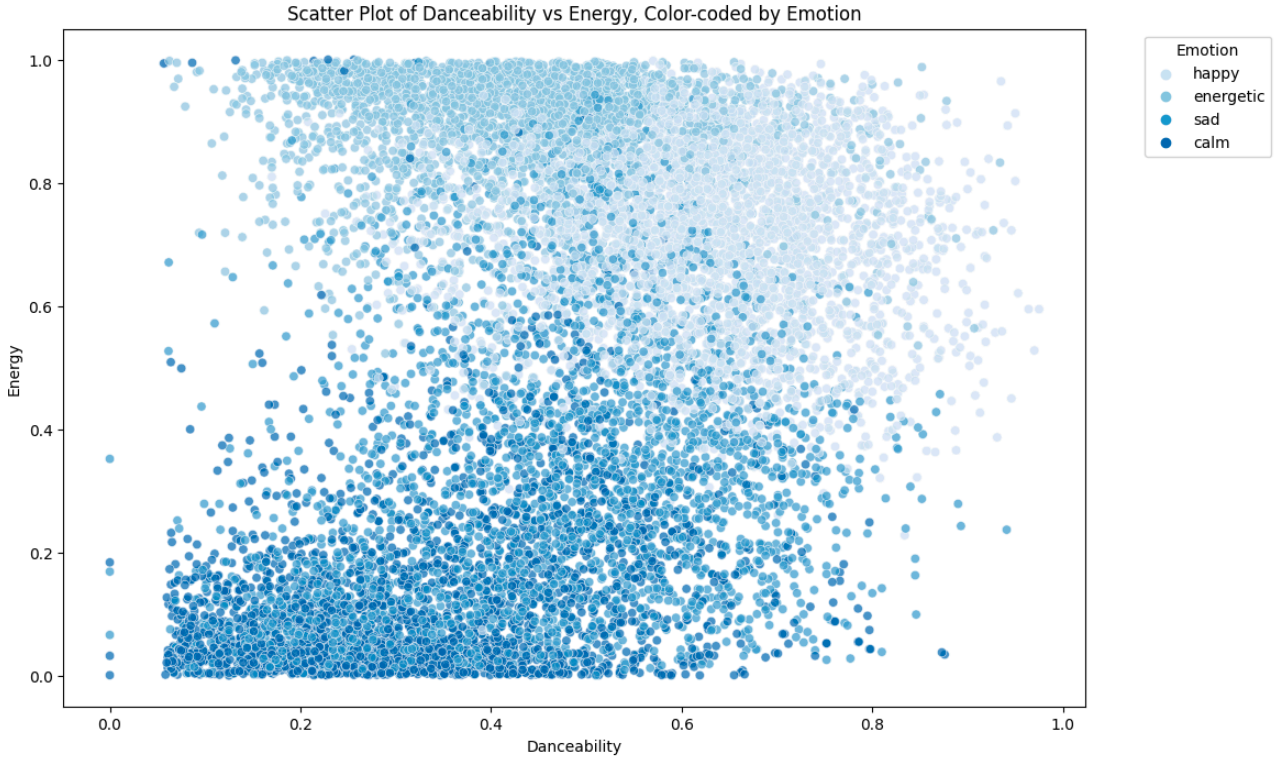
- **Akustika a Energia:** Medzi akustickosťou a energiou skladieb som identifikoval silnú negatívnu koreláciu (-0,80). To naznačuje, že skladby s vyššou akustickosťou zvyčajne majú nižšie energetické hodnoty.

- **Pokojné Emócie #1:** Pokojné emócie sú negatívne korelované s energiou a hlasitosťou skladieb. Skladby s vyššou energiou a hlasitosťou teda často nevyvolávajú pokojné emócie.

- **Pokojné Emócie #2:** Pokojné emócie sú pozitívne korelované s inštrumentálnosťou. To naznačuje, že piesne s vyššou mierou inštrumentálnych prvkov majú tendenciu vyvolávať v poslucháčoch pokojnejšie emócie.

Scatter Plot

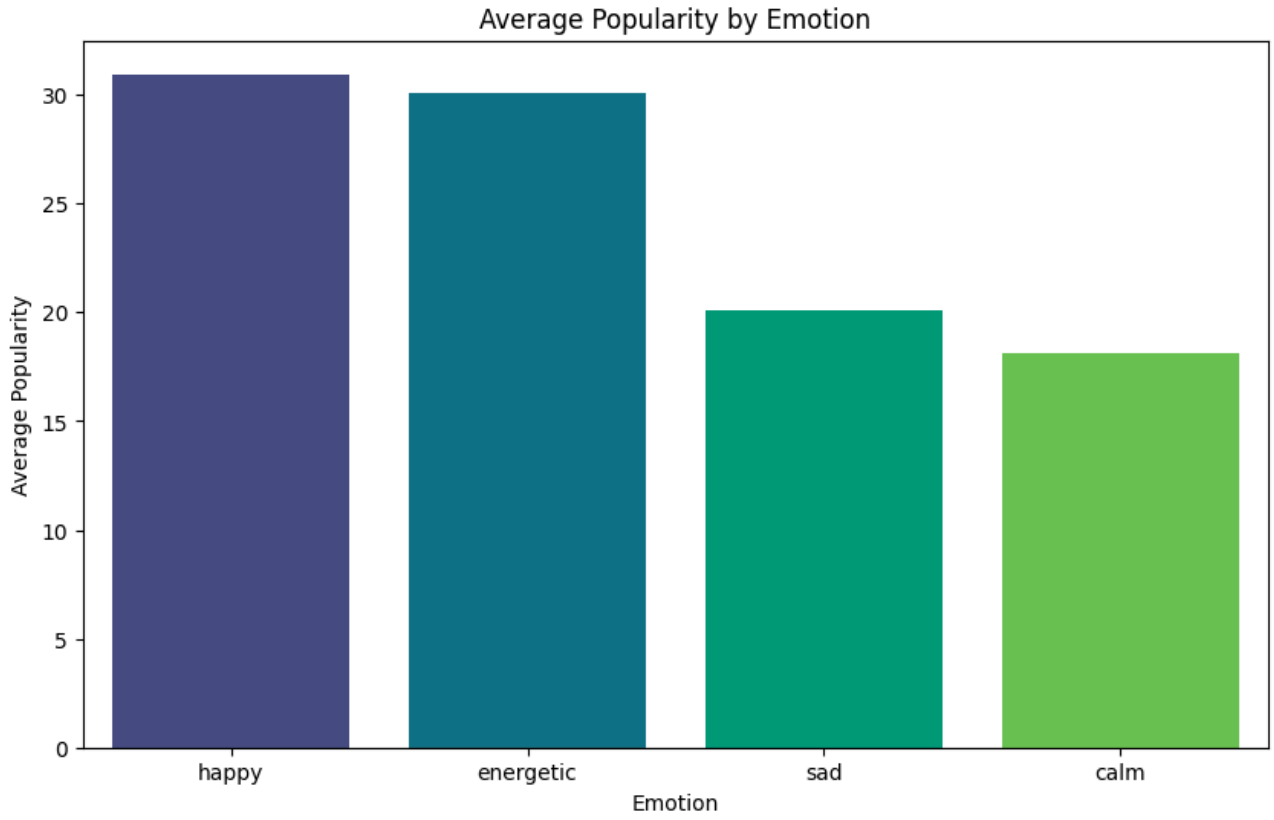
Skúmal som vzťah medzi tanečnosťou a energiou v datasete pomocou scatterplotu. Každý dátový bod bol farebne kódovaný na základe emócie skladby. Graf zobrazil rozloženie piesní naprieč rôznymi úrovňami tanečnosti a energie, čo odhalilo vzory spojené s každou emóciou.



Piesne charakterizované ako "šťastné" sa zvyčajne nachádzali v oblastiach ukazujúcich vyššiu tanečnosť a energiu. Naopak, "smutné" piesne boli prevažne zoskupené v oblastiach označujúcich nižšie hodnoty pre obe charakteristiky. Toto pozorovanie zdôrazňuje intuitívne pochopenie, že veselé melódie majú tendenciu byť živšie a vhodnejšie na tanec, zatiaľ čo melancholické skladby sa prikláňajú k viac stlmenej energii a tanečnému profilu.

Barchart #1

Na lepšie pochopenie vzťahu medzi emóciami skladieb a ich popularitou som vytvoril stĺpcový graf. Najprv som zoskupil dáta podľa emócie a vypočítal priemernú popularitu pre každú emóciu. Výsledné hodnoty som zoradil zostupne, aby som mohol vizualizovať emócie s najvyššou popularitou na začiatku.

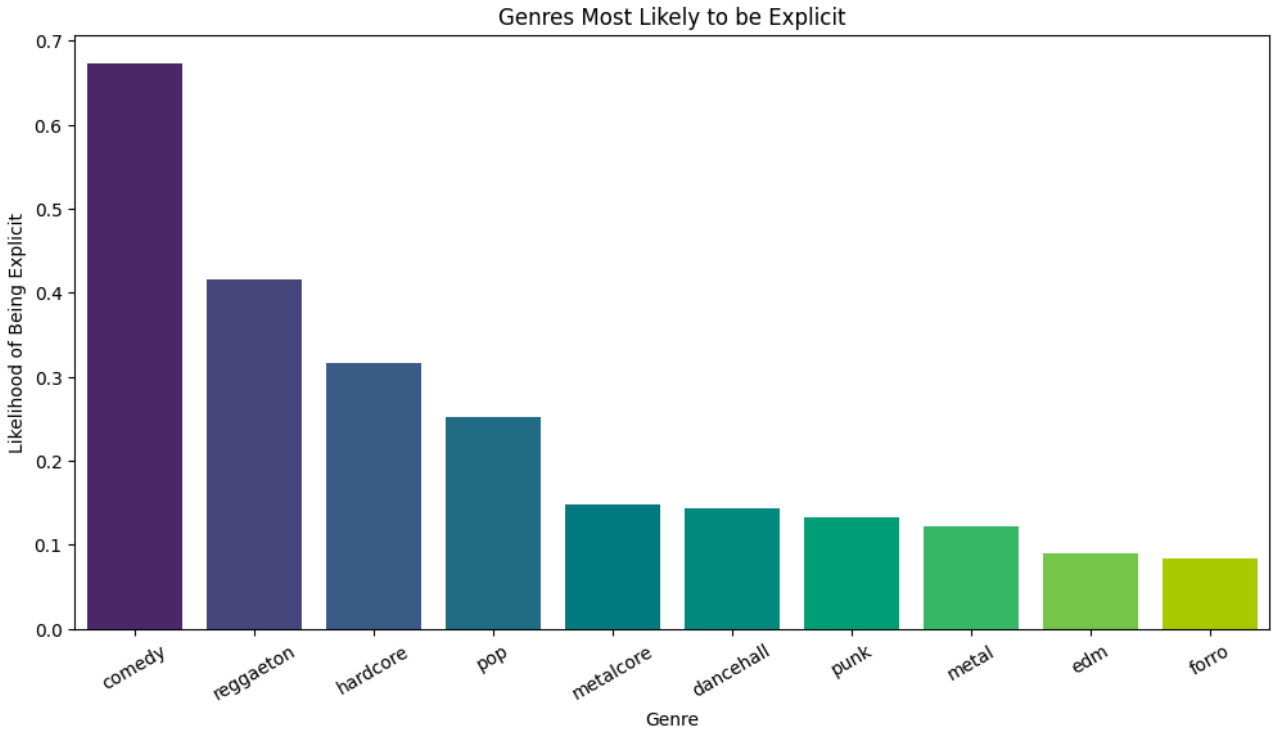


Skladby, ktoré boli klasifikované ako šťastné a energické, mali v priemere vyššiu popularitu. To naznačuje, že ľudia uprednostňujú skladby, ktoré vyvolávajú pozitívne a živé emócie.

Počty piesní v rámci jednotlivých emócií - veselých, energických, smutných a pokojných - sú v datasete pomerne vyrovnané. Napriek tejto vyváženosti vykazujú "veselé" a "energické" piesne vyššiu priemernú popularitu ako "smutné" a "pokojné" piesne. To naznačuje, že rozdiely v popularite skutočne odrážajú preferencie poslucháčov a nie sú skreslené množstvom piesní v jednotlivých kategóriách.

Barchart #2

Analyzoval som pravdepodobnosť, že rôzne hudobné žánre budú obsahovať explicitný obsah. Dáta som zoskupoval podľa stĺpca '*top_genre*' a vypočítal som priemer stĺpca '*explicit*' pre každý žáner. Tým som získal meranie toho, ako často boli skladby v každom žánri explicitné. Potom som žánre zoradil v zostupnom poradí podľa tejto hodnoty a vizualizoval 10 žánrov, ktoré najpravdepodobnejšie obsahovali explicitný obsah pomocou stĺpcového grafu. Výsledný graf poskytoval informácie o tom, ktoré žánre majú vyšší podiel skladieb s explicitným obsahom.



Najväčšiu pravdepodobnosť obsahu s explicitným obsahom mali žánre ako komédia, reggaeton, hardcore a pop. Tieto žánre majú značne vyšší podiel skladieb s explicitným textom v porovnaní s inými žánrami.

Úloha č.3: Priebeh trénovania a optimalizácia neurónovej siete

V tejto kapitole som sa zameriaval na proces trénovania neurónovej siete, s dôrazom na optimalizáciu jej architektúry a hyperparametrov. Začal som výberom architektúry siete a nastavením hyperparametrov, aby som dosiahol pretrénovanie. Potom som implementoval techniku EarlyStopping, ktorá mi pomohla predísť pretrénovaniu modelu tým, že skoršie zastavila trénovací proces v prípade, keď sa model prestal zlepšovať. Následne som experimentálne skúmal rôzne kombinácie hyperparametrov a porovnával ich výkonnosť. Výsledky som prezentoval v tabuľkovej forme, ktorá poskytovala prehľad o dosiahnutých úspešnostiach pre rôzne konfigurácie, a detailne som analyzoval priebeh trénovania pre najlepšie a najhoršie výkonné modely, vrátane vizualizácie konfúznych matíc.

Parametre pretrénovania

Pri trénovaní modelu, pri ktorom som dosiahnol pretrénovanie, som použil nasledujúce hyperparametre

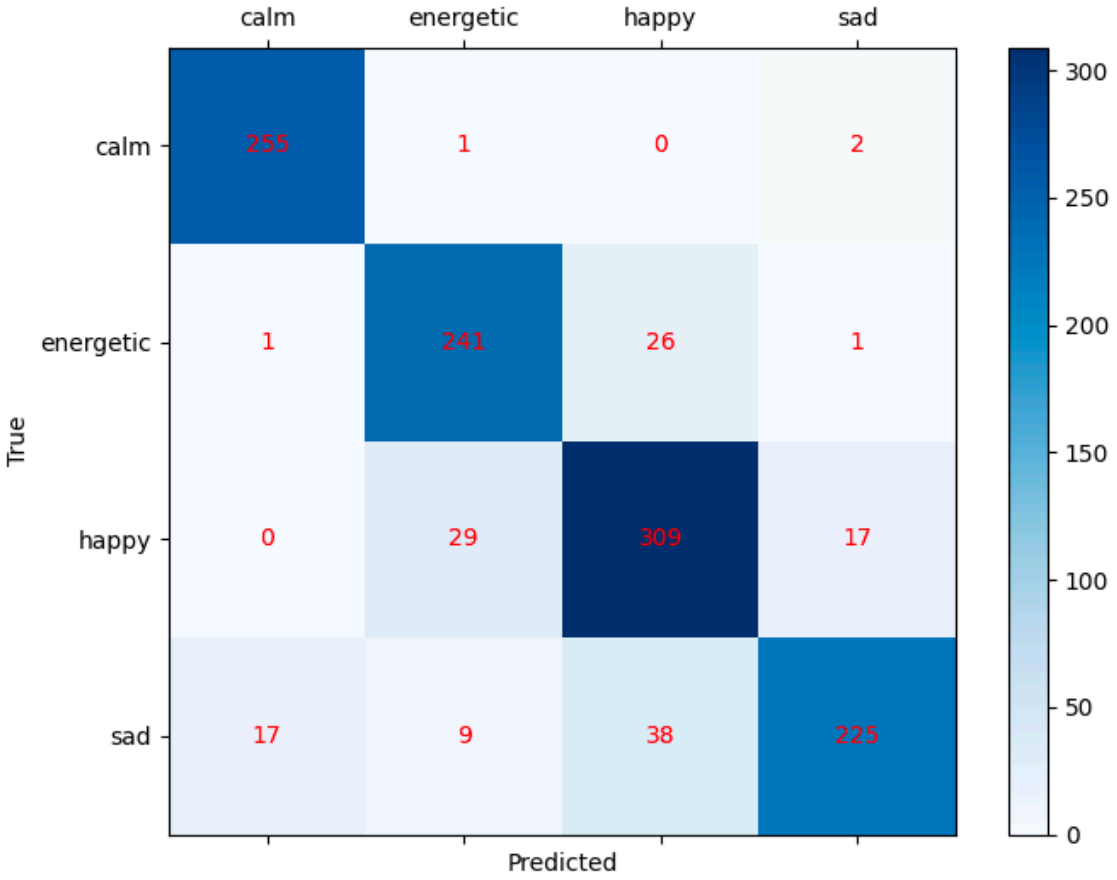
1. **Architektúra modelu**
   - Tri skryté vrstvy s 1024 neurónmi v každej vrstve
   - Aktivačná funkcia pre skryté vrstvy: relu
   - Výstupná vrstva s 4 neurónmi, čo zodpovedá štyrom kategóriám emocionálnych stavov
   - Aktivačná funkcia pre výstupnú vrstvu: softmax

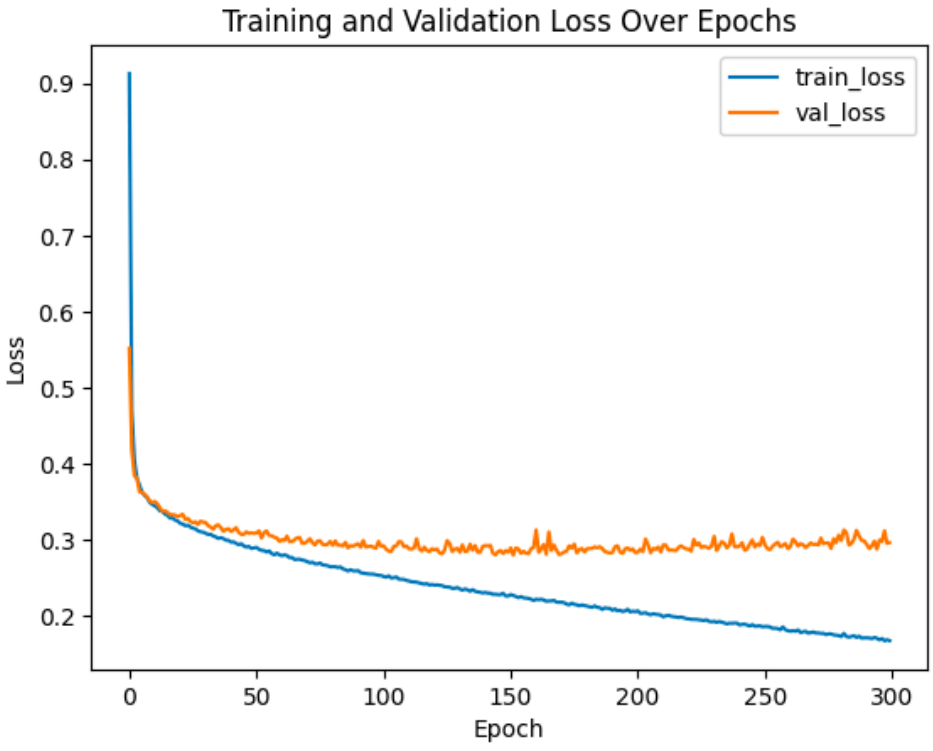
2. **Kompilačné nastavenia**
   - Loss: categorical_crossentropy
   - Optimizer: Adam
   - Learning rate: 0.00002

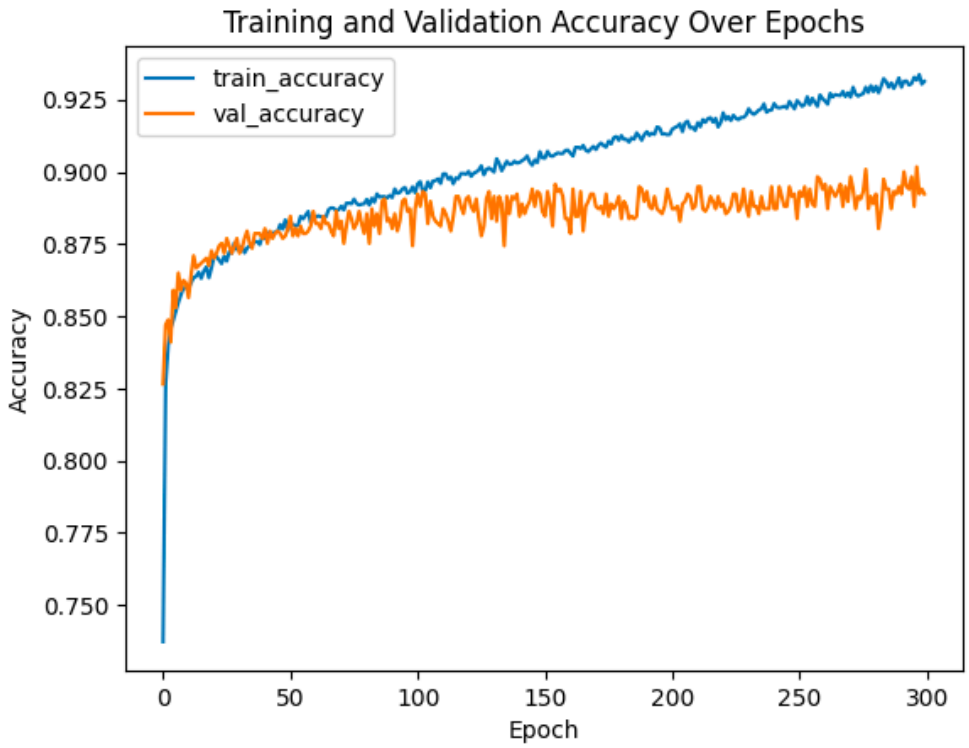
3. **Nastavenia trénovania**
   - Počet epoch: 300
   - Batch size: 32

Vyhodnotenie a demonštrácia priebehu pretrénovania

Train accuracy: 0.9325
Test accuracy: 0.8787







Zavedenie EarlyStopping-u

EarlyStopping je technika používaná na predchádzanie pretrénovaniu modelu tým, že zastaví proces trénovania, keď sa výkonnosť modelu na validačnej množine začne zhoršovať, a to ešte pred dokončením všetkých epoch.

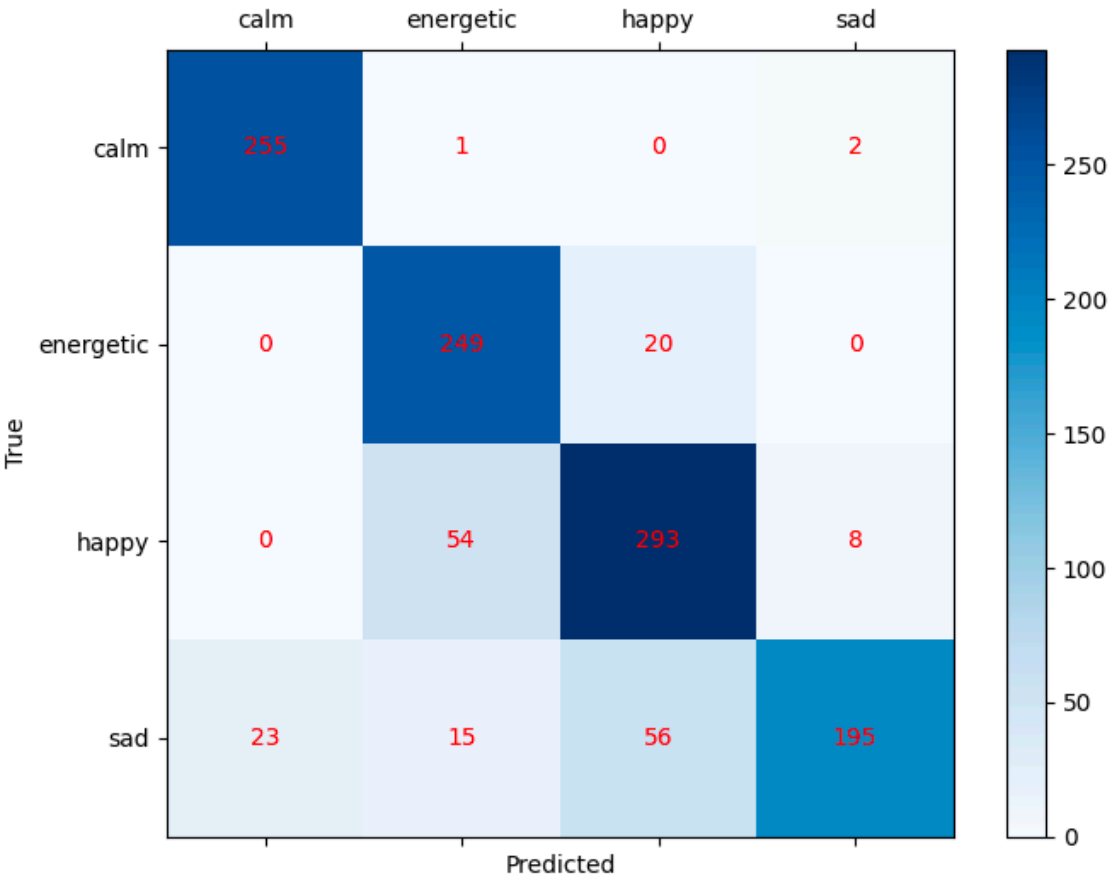
Inicializácia

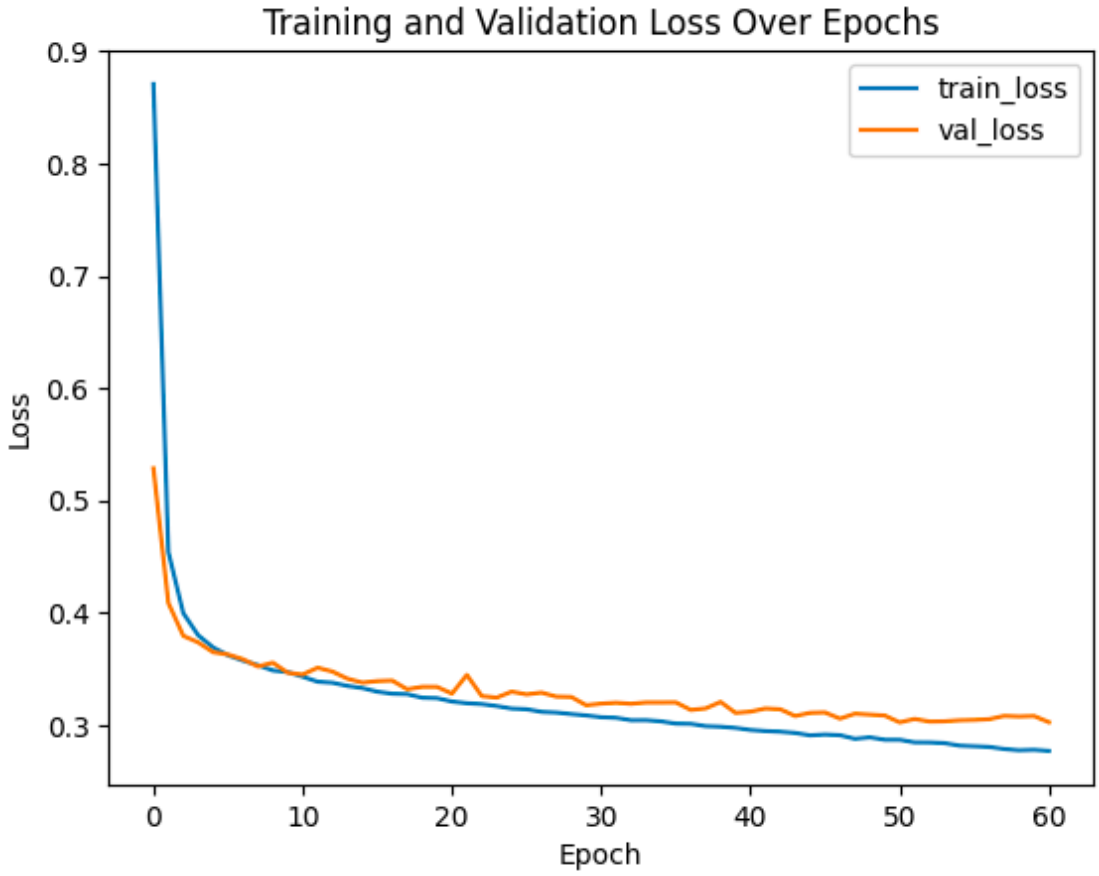
early_stopping = EarlyStopping(monitor='val_loss', patience=10, restore_best_weights=True)

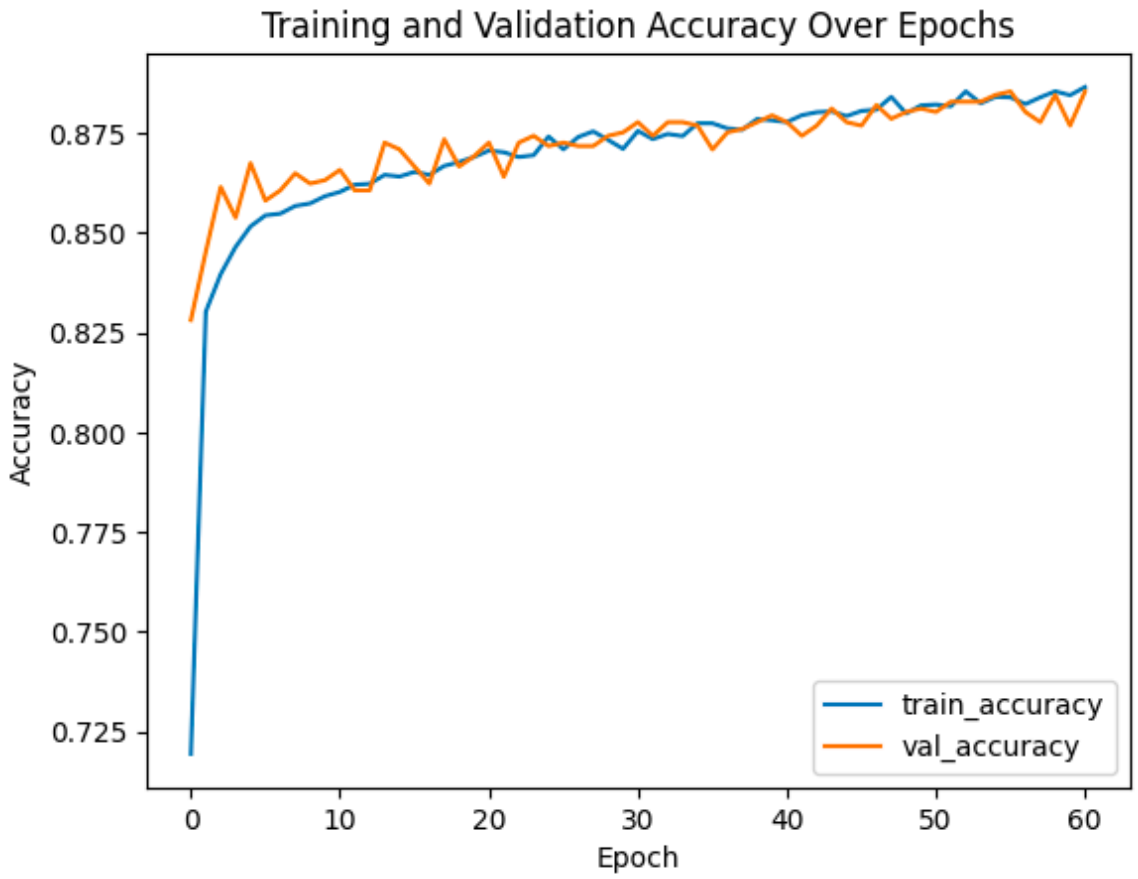
Následne bol early_stopping vložený ako callback do metody .fit

Vyhodnotenie a demonštrácia priebehu trénovania

Train accuracy: 0.8863
Test accuracy: 0.8745


calm
energetic
happy
sad
True
Predicted
255
1
0
2
0
249
20
0
0
54
293
8
23
15
56
195
250
200
150
100
50
0


Training and Validation Loss Over Epochs
train_loss
val_loss
Loss
Epoch
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0
10
20
30
40
50
60

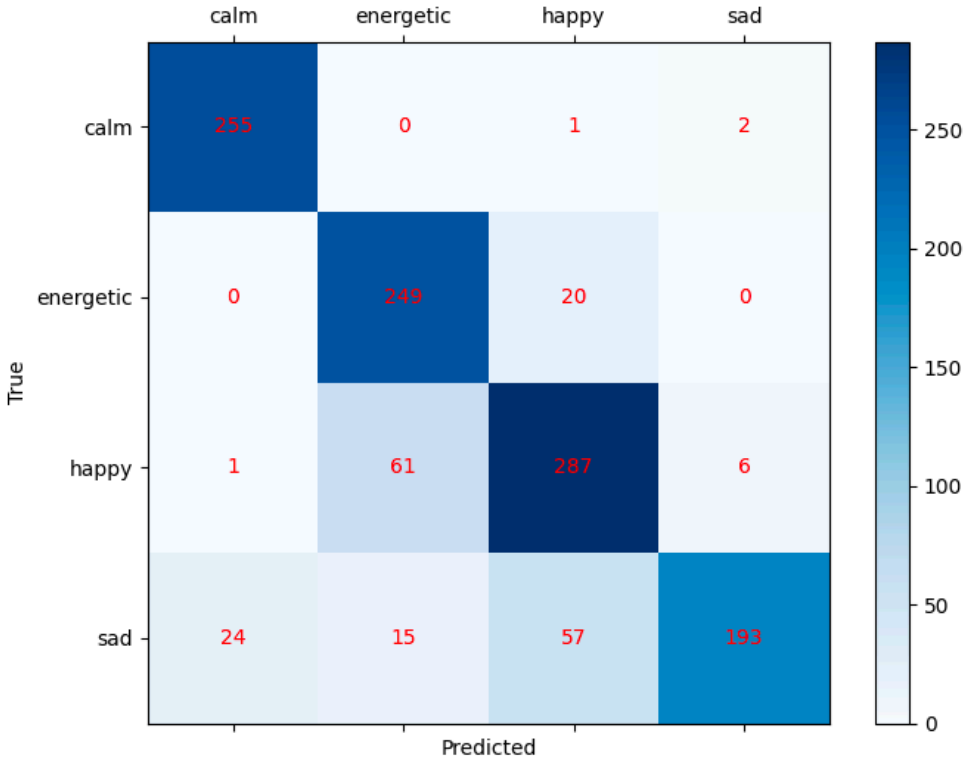


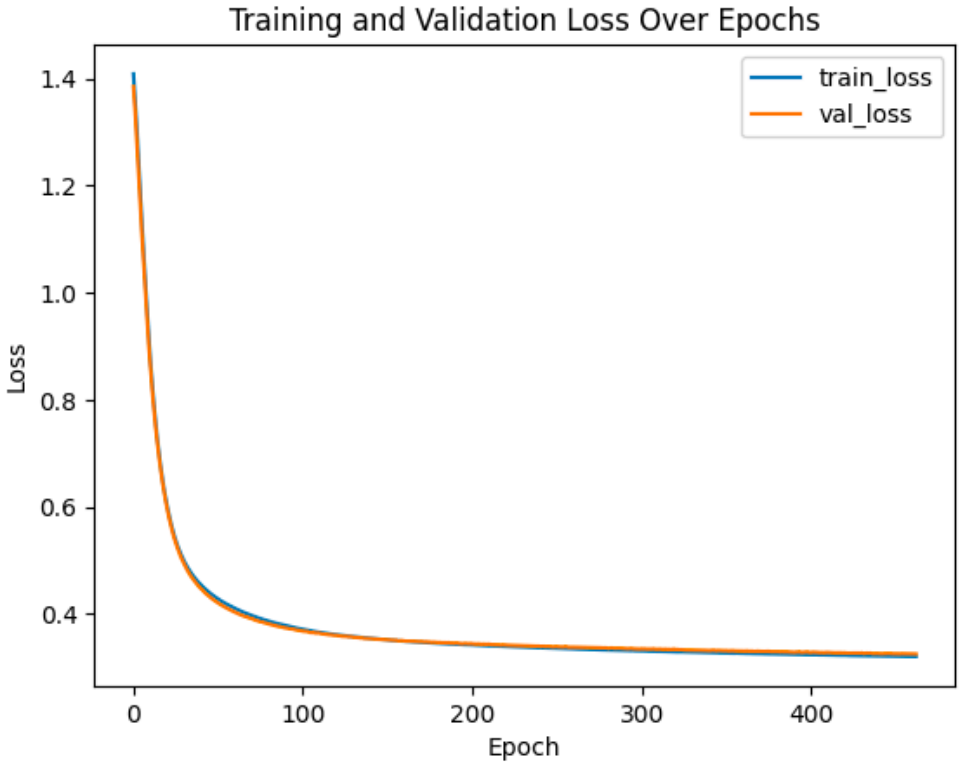
Experimentálne ladenie hyperparametrov

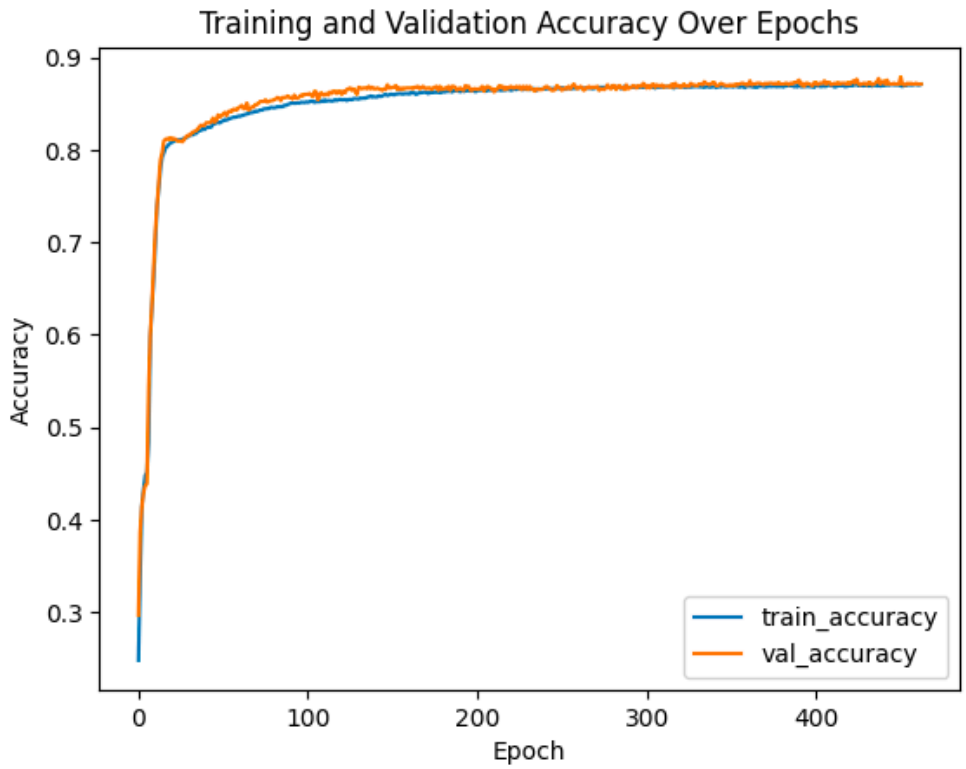
V tejto podkapitole som sa zameriaval na experimentálnu fázu vývoja neurónovej siete, kde som skúmali rôzne nastavenia hyperparametrov. Konkrétne som uskutočnil päť experimentov s cieľom optimalizovať výkon modelu. Zmeny zahrňovali úpravy rýchlosti učenia, počet neurónov v skrytých vrstvách a ďalšie. Výsledky týchto experimentov som systematicky zhodnotil a prezentoval vo forme tabuľky. Pre najlepšie a najhoršie výsledky som navyše zobrazil priebehy trénovania a konfúzne matice. Základné parametre boli spomenuté na strane 9. Všetky trénovania boli zastavené early stoppingom.

Parameter	Trénovacia Množina	Testovacia Množina
1024 neurónov v 3 sk. vrstvách	0.8863	0.8745
512 neurónov v 3 sk. vrstvách	0.8867	0.8745
32 neurónov v 3 sk. vrstvách	0.8670	0.8617
32 neurónov v 6 sk. vrstvách	0.8755	0.8668
512 neurónov v 4 sk. vrstvách	0.8982	0.8787
512 neurónov v 4 sk. vrstvách, learning_rate=0.0001	0.9088	0.8796

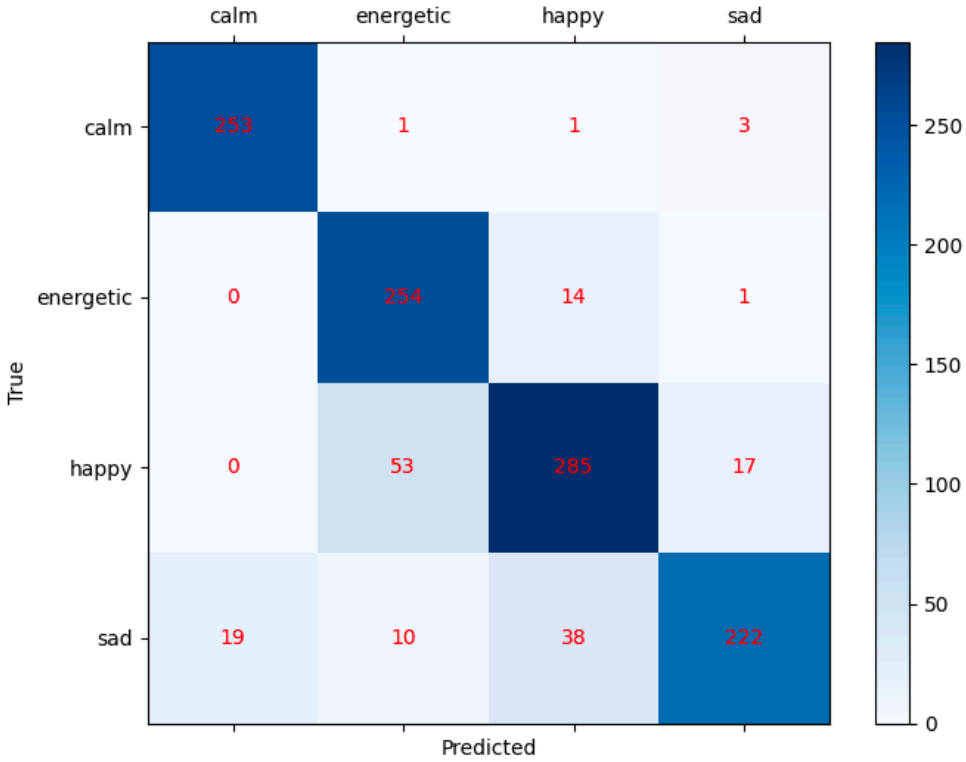
Demonštrácia najhoršieho trénovania

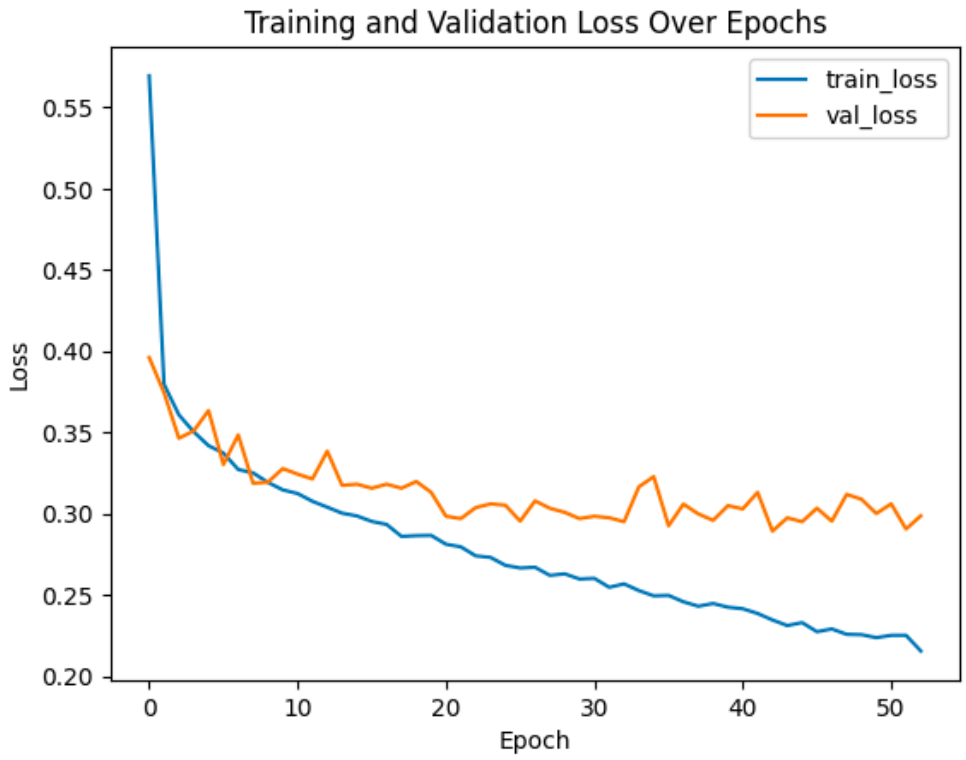


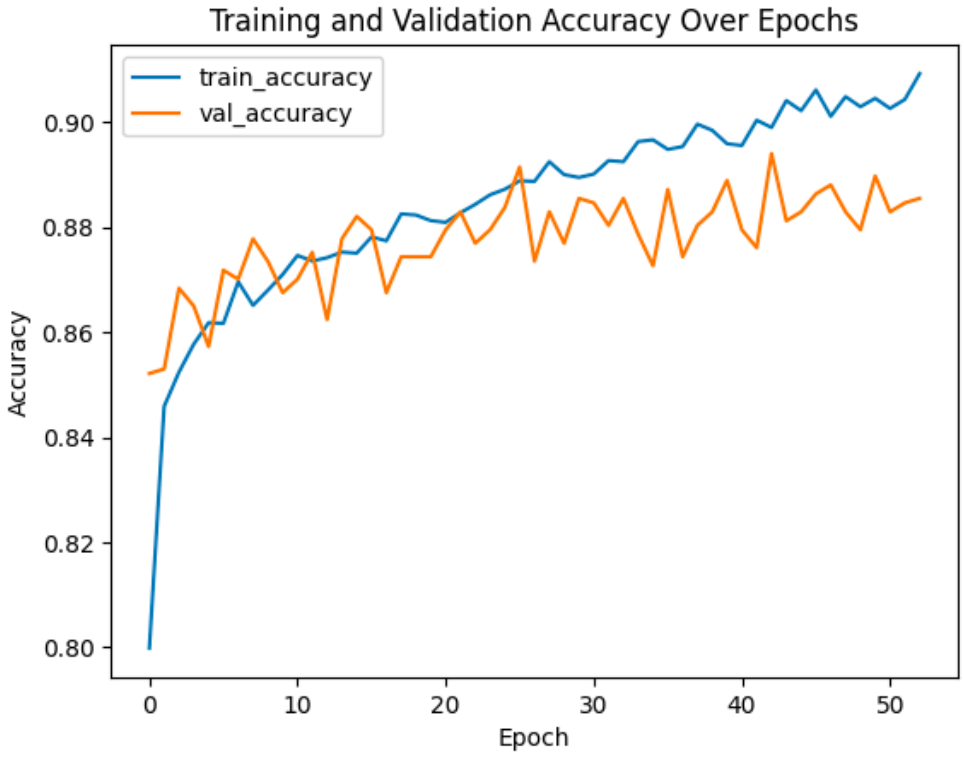




Demonštrácia najlepšieho trénovania




Training and Validation Loss Over Epochs
train_loss
val_loss
Loss
Epoch
0.55
0.50
0.45
0.40
0.35
0.30
0.25
0.20
0
10
20
30
40
50


Training and Validation Accuracy Over Epochs
train_accuracy
val_accuracy
Accuracy
Epoch
0.90
0.88
0.86
0.84
0.82
0.80
0
10
20
30
40
50