به نام خدا

دانشگاه تهران

دانشکده مهندسی مکانیک

درس هوش مصنوعی

تمرین سوم

نام و نام خانوادگی	محمد اخلاقی
شماره دانشجویی	۸۱۰۶۰۱۰۲۷
تاریخ ارسال گزارش	۱۴۰۲.۰۳.۰۴

فهرست

شکل‌ها

پاسخ ۱. دسته بندی گیاهان بر پایه‌ی ویژگی های ظاهری آن‌ها

۱-۱. بررسی داده ها و پیش‌پردازش آنها

در مرحله اول داده‌های Iris خوانده شده و با توجه به این که در ستون کلاس لیبل ها به صورت کلمه نوشته شده است، لیبل ها با شماره جایگزین میشوند. ۳ کلاس ۰-۱-۲ برای سه دسته انتخاب شد.

سپس با توجه به اینکه پراکندگی متفاوت اعداد ویژگی‌ها نیاز به نرمال سازی این اعداد می‌باشد، پس به کمک تابع MinMaxScaler() داده ها از بین ۰-۱ تنظیم می‌شوند. این مراحل در تابعی به نام step1 نوشته شده است. سوال نتایج برای حالتی که داده ها نرمال نشدن رو هم خواسته پس در یک حالت نتایج بدون نرمال سازی داده هم به دست می‌آوریم.

```
11  def step1_ver1(dataset):
12      # Load the dataset
13      data = pd.read_csv(dataset)
14
15      # Encode class names with numeric labels
16      label_encoder = LabelEncoder()
17      data['Class'] = label_encoder.fit_transform(data['Class'])
18
19      # Normalize the features using Min-Max scaling
20      scaler = MinMaxScaler()
21      data[['Sepal_Length', 'Sepal_Width', 'Petal_Length', 'Petal_Width']] = scaler.fit_transform(data[['Sepal_Length',
22
23      return data
24
25  # Preprocess the Iris dataset using step1_ver1
26  preprocessed_data = step1_ver1('Iris.csv')
```

شکل ۱: پیش پردازش داده ها

نمونه داده های پیش پردازش داده شده در شکل زیر مشاهده می‌شود.

	Sepal_Length	Sepal_Width	Petal_Length	Petal_Width	Class
0	0.222222	0.625000	0.067797	0.041667	0
1	0.166667	0.416667	0.067797	0.041667	0
2	0.111111	0.500000	0.050847	0.041667	0
3	0.083333	0.458333	0.084746	0.041667	0
4	0.194444	0.666667	0.067797	0.041667	0

در ادامه داده‌ها طبق گفته سوال به دو قسمت train و test تقسیم شد. به شکلی که از هر کلاس به ۸۰ درصد در قسمت train و ۲۰ درصد test باشد. این مرحله در تابع step2 تعریف شد.

```python
def step2_ver1(preprocessed_data):
    # Separate features and target variable
    X = preprocessed_data[['Sepal_Length', 'Sepal_Width', 'Petal_Length', 'Petal_Width']]
    y = preprocessed_data['Class']

    # Split the data into training and test sets, stratified by the target variable
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, stratify=y, random_state=42)

    # Concatenate the features and target variable for each set
    train_data = pd.concat([X_train, y_train], axis=1)
    test_data = pd.concat([X_test, y_test], axis=1)

    return train_data, test_data,X_train, X_test, y_train, y_test

# Split the preprocessed data using step2_ver1
train_data, test_data,X_train, X_test, y_train, y_test = step2_ver1(preprocessed_data)
```

شکل ۲: تقسیم دادگان به تست و ترین

۲-۱. ساخت مدل KNN و تربیت آن:

در ادامه مدل KNN با N=5 تعریف شد و با داده‌های ترین آموزش داده شد.(train.fit) در تابع step3 که به این منظور توسعه یافته، خروجی مدل بر روی داده های ترین و تست نیز به دست آمد.

```python
def step3_ver2(train_data, test_data):
    # Separate features and target variable from training data
    X_train = train_data[['Sepal_Length', 'Sepal_Width', 'Petal_Length', 'Petal_Width']]
    y_train = train_data['Class']

    # Separate features and target variable from test data
    X_test = test_data[['Sepal_Length', 'Sepal_Width', 'Petal_Length', 'Petal_Width']]
    y_test = test_data['Class']

    # Create a KNN classifier with K=5
    knn = KNeighborsClassifier(n_neighbors=5)

    # Train the KNN model
    knn.fit(X_train, y_train)

    # Predict on training and test data
    train_predictions = knn.predict(X_train)
    test_predictions = knn.predict(X_test)

    return knn, train_predictions, test_predictions, y_test
```

شکل ۳: تعریف و تربیت مدل

۳-۱. نتایج

نتیجه آموزش مدل با به دست آوردن ماتریس سردرگمی و پارامترهای گفته شده در صورت سوال بده دست آمد. همچنین برای سه پارامتر گفته شده در صورت سوال میانگیری به روش **macro** انجام شد. نتایج در دو قسمت الف و ب قرار گرفته است که تفاوت آنها در نرمالسازی داده هاست. در قسمت الف نرمالسازی انجام نشده است.

```
# Predict on the test data
y_pred = knn_model.predict(X_test)

# Compute the confusion matrix
cm = confusion_matrix(y_test, y_pred)

# Calculate precision, recall, accuracy, and Jaccard similarity score
precision = precision_score(y_test, y_pred, average='macro')
recall = recall_score(y_test, y_pred, average='macro')
accuracy = accuracy_score(y_test, y_pred)
jaccard = jaccard_score(y_test, y_pred, average='macro')
```

شکل ۴: به دست آوردن نتایج

الف)در این قسمت نرمال سازی داده کامنت شد و داده های اورجینال برای ترین شدن استفاده شد.

پارامترهای خواسته شده به دست آمد. در تصویر زیر نتایج مشاهده میشود.

```
Precision: 1.0
Recall: 1.0
Accuracy: 1.0
Jaccard Similarity Score: 1.0
```

شکل ۵: نتایج دقت مدل

در ادامه ماتریس سردرگمی نیز رسم شد.

	واقعی	تشخیص		
	مجموع	Iris-setosa	Iris-versicolor	Iris-virginica
Iris-setosa	۱۰	۱۰	۰	۰
Iris-versicolor	۱۰	۰	۱۰	۰
Iris-virginica	۱۰	۰	۰	۱۰

ب) در این بخش نرمال سازی داده ها انجام شد. نتایج به شکل زیر به دست آمد.

```
Precision: 0.9696969696969697
Recall: 0.9666666666666667
Accuracy: 0.9666666666666667
Jaccard Similarity Score: 0.9363636363636364
```

شکل ۶:نتایج دقت مدل

ماتریس سردرگمی هم رسم شد.

واقعی		تشخیص		
	مجموع	Iris-setosa	Iris-versicolor	Iris-virginica
Iris-setosa	۱۰	۱۰	۰	۰
Iris-versicolor	۱۰	۰	۱۰	۰
Iris-virginica	۱۰	۰	۱	۹

برای نتیجه گیری از تاثیر نرمال سازی بر روی دقت مدل **random_state** را در تقسیم داده ها تغییر دادم و نتایج ثابت نبود. گاهی دقت برابر بود، گاهی دقت مدل با فیچرهای نرمال شده بهتر بود و گاهی دقت بدون نرمال سازی. بنابراین میتوان نتیجه گرفت نرمال کردن داده بر مدل **KNN** تاثیر خاصی ندارد.

پاسخ ۲ – پیش بینیِ گونه و میزان محبوبیت آهنگ های جدید

الف و ب) پیش پردازش داده:

پس از خواندن فایل ستون **popularity** به ۵ قسمت تقسیم شد و در ستون دیگر با نام **popularity_class** ثبت شد. سپس ۳ ستون اضافه + ستون **popularity** قبلی حذف شد.

سپس ستونهای کیفی باقی مانده به صورت عددی در آمد. **true-false** به صورت ۰-۱ و ژانر به صورت یک عدد باینری یا بردار گفته شده در صورت سوال در آمد. ستون ژانر قبلی هم حذف شد. نمونه دیتا پس از پیش‌پردازش در زیر آمده است.

	duration_ms	explicit	danceability	energy	key	loudness	mode	\
0	211160	0	0.751	0.834	1	-5.444	0	
1	167066	0	0.434	0.897	0	-4.918	1	
2	250546	0	0.529	0.496	7	-9.007	1	
3	224493	0	0.551	0.913	0	-4.063	0	
4	200560	0	0.614	0.928	8	-4.806	0	

	speechiness	acousticness	instrumentalness	liveness	valence	tempo	\
0	0.0437	0.3000	0.000018	0.3550	0.894	95.053	
1	0.0488	0.0103	0.000000	0.6120	0.684	148.726	
2	0.0290	0.1730	0.000000	0.2510	0.278	136.859	
3	0.0466	0.0263	0.000013	0.3470	0.544	119.992	
4	0.0516	0.0408	0.001040	0.0845	0.879	172.656	

	popularity_class	genre_vector
0	60-80	000100000000000
1	60-80	000100000001000
2	60-80	000100000100000
3	60-80	000000000011000
4	60-80	000100000000000

شکل ۷: دیتا پس از پیش پردازش

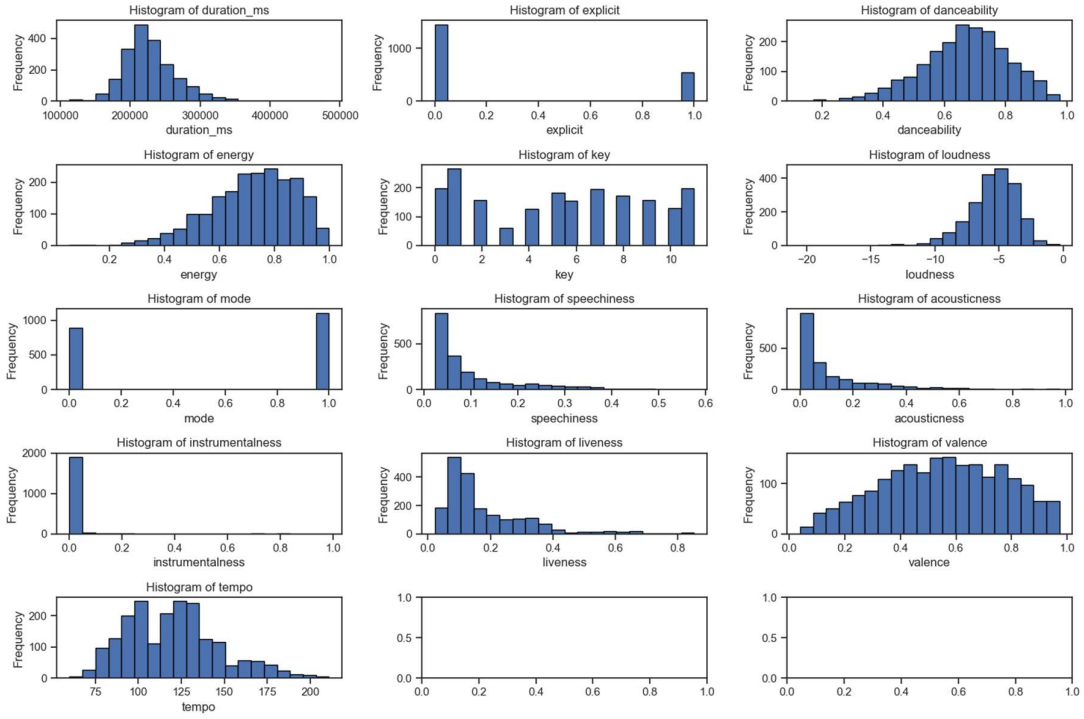
همچینین در این مرحله یک دیکشنری ساخته شد و جایگاه هر ژانر در بردار ژانر ثبت شد تا بعدا بتوانیم از بردار ژانر، ژانر را تشخیص دهیم.

```
{'jazz': 0, 'Dance/Electronic': 1, 'blues': 2, 'pop': 3, 'classical': 4, 'latin': 5, 'R&B': 6,
```

شکل ۸: قسمتی از دیکشنری بردار ژانر

ج)

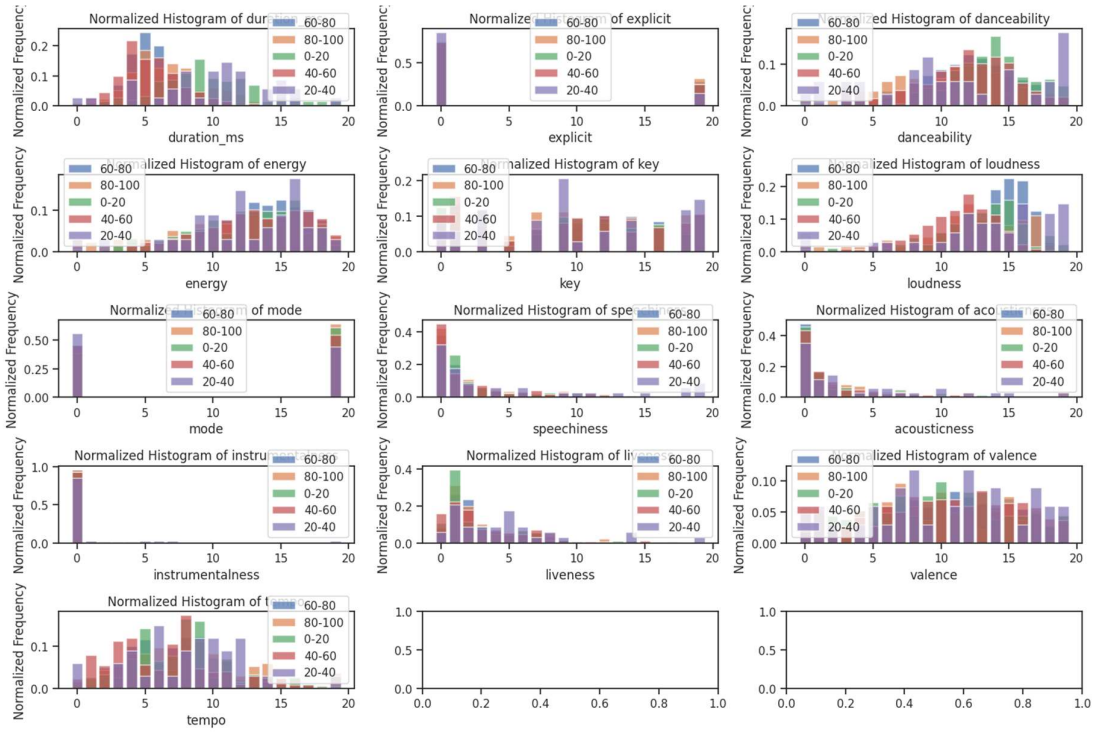
نمودار پراکندگی داده در همه‌ی ویژگی ها ویژگی در شکل زیر آمده است.



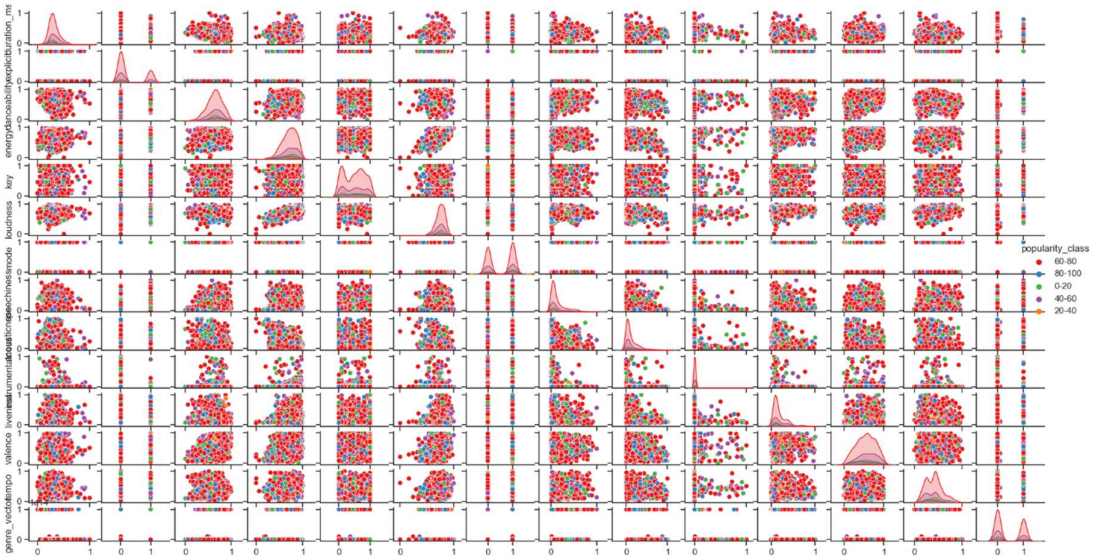
شکل ۹: هیستوگرام کل داده ها برای هر فیچر

در شکل زیر نمودار پراکندگی داده‌ها با محبوبیت متفاوت در رنگهای متفاوت رسم شده اند. قبل از رسم این نمودار تعداد داده ها نسبت به میزان داده‌ی هر دسته نرمال شد.در غیر این صورت تعداد داده های محبوبیت ۶۰-۸۰ آنقدر زیاد بود که داده های دیگر دیده نمی‌شدند.

در شکل ۱۱ میتوان pairplot را مشاهده کرد که با توجه به عدم یکسان بودن تعداد داده های هر دسته چیزی دیده نمیشود و نتیجه خاصی نمیتوان از آن گرفت.



شکل ۱۰: هیستوگرام با نرمال سازی تعداد داده هر دسته



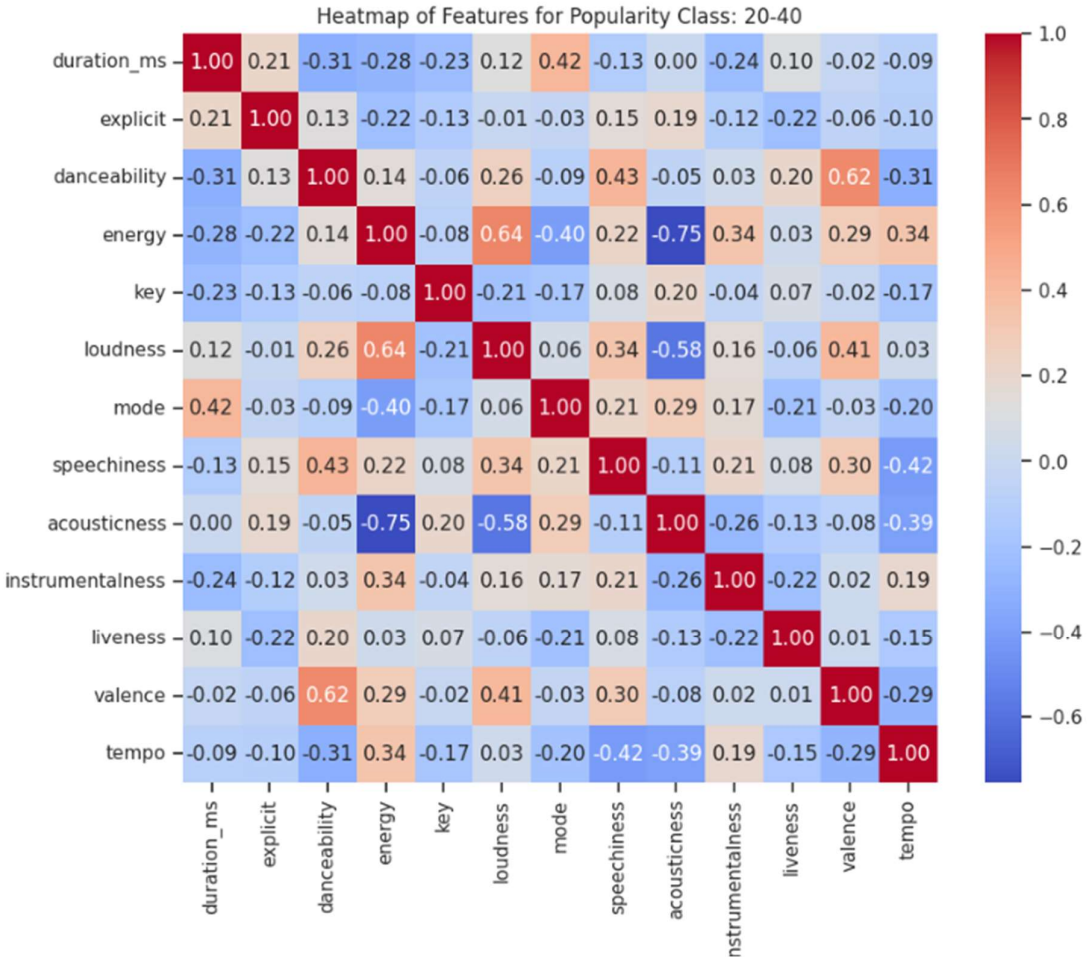
شکل ۱۱ **pairplot**

د)

در این قسمت نمودار گرمایی کورولیشن ماتریس برای هر دسته ی محبوبیت رسم شد.

خانه هایی که در آن عدد بیشتر است نشان میدهد در صورتی که داده در این دسته محبوبیت باشد احتمال خوبی وجود دارد هر دو فیچر را با هم داشته باشد. در گزارش تنها هیت مپ کورولیشن ماتریس کلاس محبوبیت ۴۰-۲۰ برای مثال آمده است اما در فایل ها بقیه نمودارها نیز وجود دارند.

برای مثال در هیت مپ زیر مشاهده میشود که loudness و انرژی بالا به طور همزمان در بخش زیادی از داده های با محبوبیت ۲۰-۴۰ وجود دارند و یا آکوستیکنس و قابلیت رقص که با یک دیگر در تضادند احتمالا اتفاق نمی‌افتند، مقدار کورولیشن -۰,۷۵ دارند.



۵)

همانطور که سوال خواسته داده ها به سه بخش تقسیم شد. با توجه به این که از svm استفاده خواهیم کرد داده ها نرمال سازی نیز شد.

```
# Normalize the features
scaler = MinMaxScaler()
X_normalized = scaler.fit_transform(X)

# Split the normalized data into train and test sets (80% for training, 20% for testing)
X_train, X_test, y_train, y_test = train_test_split(X_normalized, y, test_size=0.2, random_state=42)

# Split the training set further into training and validation sets (80% for training, 20% for validation)
X_test, X_val, y_test, y_val = train_test_split(X_test, y_test, test_size=0.5, random_state=42)

# Print the shapes of the data subsets
print("Training Set Shape:", X_train.shape, y_train.shape)
print("Validation Set Shape:", X_val.shape, y_val.shape)
print("Test Set Shape:", X_test.shape, y_test.shape)
```

```
Training Set Shape: (1600, 13) (1600,)
Validation Set Shape: (200, 13) (200,)
Test Set Set Shape: (200, 13) (200,)
```

شکل ۱۲: تشسیم داده ها به **train,test, validation**

و)

هر سه مدل تربیت شد و نتایج آن به شرح زیر به دست آمد. ماتریس سردرگمی به صورت مجزا رسم نشده و ستون ها به ترتیب محبوبیت کم به زیاد از چپ به راست و سطرها از بالا به پایین محبوبیت افزایش می یابد.

تمامی مدلها خطای زیادی دارند و مهم ترین دلیل آن را میتوان تعداد نا متوازن تعداد داده های هر کلاس محبوبیت حدس زد. در شکل زیر تعداد داده هی هر کلاس محبوبیت قابل مشاهده است. میتواند دید تعداد داده های کلاس محبوبیت ۶۰-۸۰ تقریبا ۴۰ برابر داده های کلاس ۲۰-۴۰ است، لذا مدل نمیتواند به درستی فیت شود.

```
Popularity Class: 60-80, Count: 1205
Popularity Class: 40-60, Count: 442
Popularity Class: 0-20, Count: 184
Popularity Class: 80-100, Count: 135
Popularity Class: 20-40, Count: 34
```

شکل ۱۳: تعداد داده های هر کلاس محبوبیت

KNN: Kهای ۲ تا ۱۰ چک شد که بهترین دقت در K=9 بود. دقت دادگان تست: ۶۰,۵ درصد

```
Training Accuracy: 0.625625
Test Accuracy: 0.605
Confusion Matrix:
[[  0   0   1  20   0]
 [  0   0   0   1   0]
 [  0   0   9  35   0]
 [  0   0   8 112   0]
 [  0   0   0  14   0]]
```

شکل ۱۴ نتایج دقت و ماتریس سردرگمی الگوریتم **KNN**

درخت تصمیم گیری:

نتایج نشان میدهد این مدل اورفیت شده و تنها برای داده های ترین خوب عمل کرده. ۴۶,۵٪

```
Training Accuracy: 0.99875
Test Accuracy: 0.465
Confusion Matrix:
[[ 3  1  1 16  0]
 [ 0  0  1  0  0]
 [ 1  0 16 24  3]
 [11  2 23 73 11]
 [ 2  1  2  8  1]]
```

شکل ۱۵: نتایج دقت و ماتریس سردرگمی الگوریتم درخت تصمیم گیری

SVM: همه‌ی کرنل ها تست شد و rbf بهترین نتیجه رو داشت. دقتی نزدیک به الگوریتم KNN به دست اومد.میشه دید تنها کاری که کرده اینه که تمام داده ها رو در دسته ۶۰-۸۰ فرض کرده و عملا به درد نخورده. % ۶۰

```
Training Accuracy: 0.6025
Test Accuracy: 0.6
Confusion Matrix:
[[  0   0   0  21   0]
 [  0   0   0   1   0]
 [  0   0   0  44   0]
 [  0   0   0 120   0]
 [  0   0   0  14   0]]
```

شکل ۱۶: نتایج دقت و ماتریس سردرگمی الگوریتم **SVM**

نتیجه و و ز)

بهترین مدل تربیت شده (KNN) دقت ۶۰,۵ درصدی داشت که دقت خوبی نیست. پیشنهاد من چند برابر کردن تعداد داده های دسته هاییست که دارای محبوبیت کم هستند. برای مثال با ۱۰ برابر کردن تعداد داده های دسته‌ی ۸۰-۱۰۰ تاثیر ۱۳۰ این داده ها حدودا به اندازه‌ی ۱۲۰۰ داده‌ی دسته‌ی ۶۰-۸۰ میشود و مدل بهتر میتواند فرق دو دسته را تشخیص دهد.

ح)

موارد خواسته شدهبه دست آمد و درشکل های زیر مشاهده میشود. در این بخش از دادگان اصلی سوال استفاده نشد و برای تبدیل بردار ژانر به نام ژانر از دیکشنری که در پیش پردازش تنظیم شده بود استفاده شد.

```
Most Frequent Genre: pop
Count: 1633
Least Frequent Genre: classical
Count: 1
```

شکل ۱۷: بیشترین ژانر و کمترین ژانر

```
Maximum Genre Count: 4
```

شکل ۱۸: بیشترین تعداد ژانر در یک آهنگ

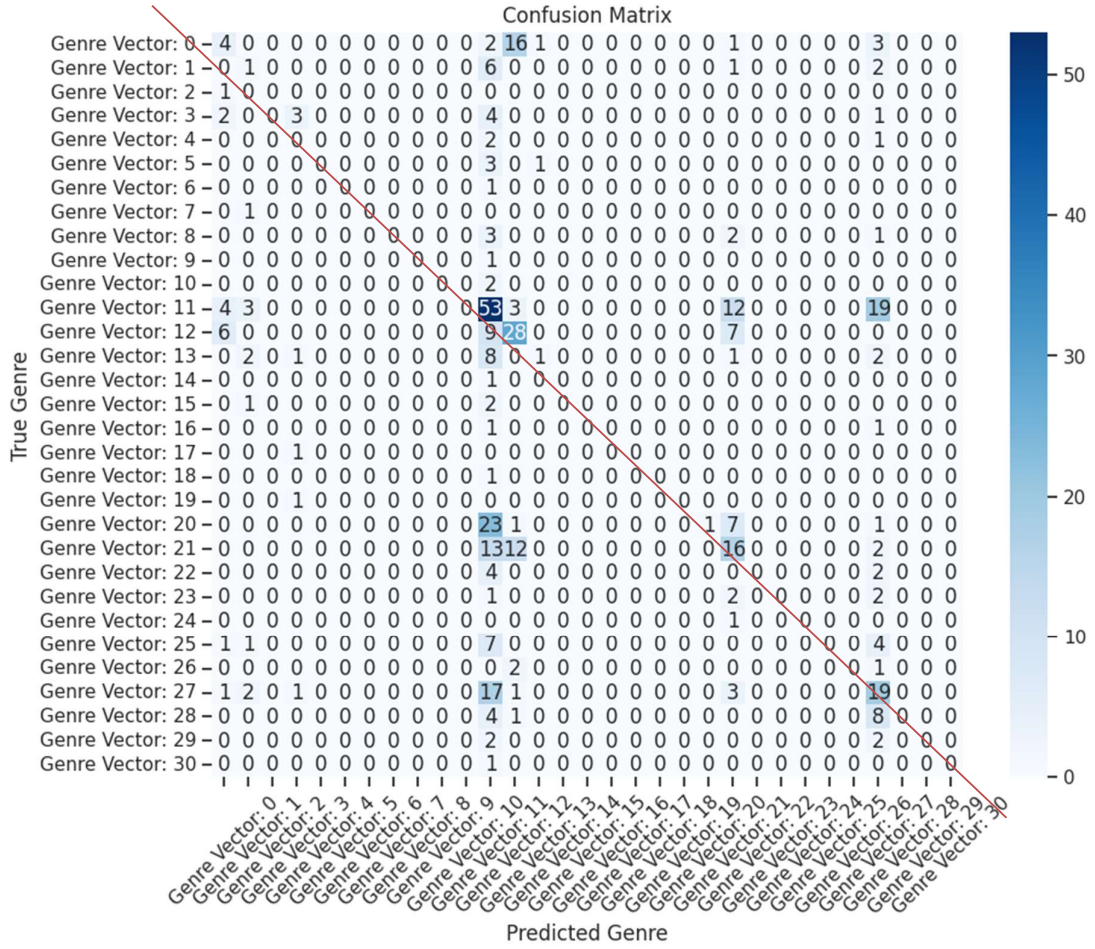
```
Chance of having R&B given both 'hip hop' and 'pop' genres: 0.39619651347068147
```

شکل ۱۹: احتمال وجود ژانر **R&B** در آهنگاهای ژانر پاپ و هیپ هاپ

ت)

داده ها به سه قسمت تنظیم شد و با استفاده از الگوریتم درخت تصمیم گیری مدلی تربیت شد که ژانر را بر اساس فیچرها پیدا کند.

دقت این مدل هم ۳۱,۵ درصد بود. البته باید توجه کرد که در این ارزیابی صورتی که مدل نتواند تمام ژانرهای یک آهنگ را اعلام کند اشتباه محسوب شده و ۳۱,۵٪ حدس درست مدل مربوط به آهنگهاییست که مدل توانسته کاملا درست حدس بزند. خط قرمز رسم شده قطر ماتریس و آهنگهایی که درست تخمین زده شده اند را نشان میدهد.



شکل ۲۰ماتریس سردرگمی داده های تست برای مدل تعیین ژانر