Spotify Data EDA and Hypothesis

Jonathan Kristianto (23229555), Clarita Nathania Worrow (23229533) , Nathanael Iskandar Wibowo (23229579)

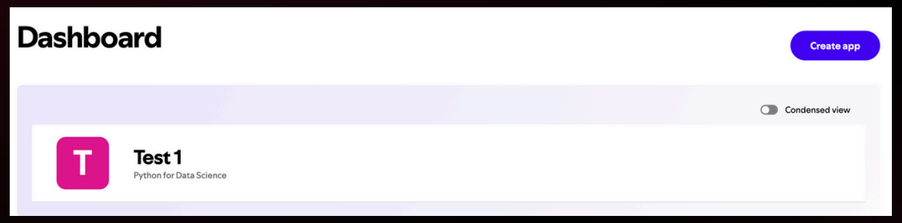
ABOUT SPOTIFY WEB API

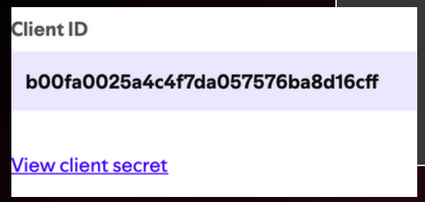
The Spotify Web API is a restful API with different endpoints which return JSON metadata about music artists, albums, and tracks, directly from the Spotify Data Catalogue

EDA (Explanatory Data Analysis)

In this analysis, we explore data collected from the Spotify Web API, focusing on a selection of globally recognized artists. The goal of this Exploratory Data Analysis (EDA) is to understand key characteristics of artists and their music performance metrics, such as followers, popularity, and genres.

Developer Spotify






```python
import requests
import base64
import json
import pandas as pd
from datetime import datetime

CLIENT_ID = 'b00fa0025a4c4f7da057576ba8d16cff'
CLIENT_SECRET = '9498a8e214a6461bb8abb439a3ef9ee0'

def get_access_token(client_id, client_secret):
    auth_url = 'https://accounts.spotify.com/api/token'
    auth_header = base64.b64encode(f"{client_id}:{client_secret}".encode('utf-8')).decode('utf-8
    headers = {'Authorization': f'Basic {auth_header}'}
    data = {'grant_type': 'client_credentials'}
    response = requests.post(auth_url, headers=headers, data=data)
    response.raise_for_status()
    token_info = response.json()
    return token_info['access_token']

def get_artist_details(artist_id, headers):
    artist_url = f"https://api.spotify.com/v1/artists/{artist_id}"
    response = requests.get(artist_url, headers=headers)
    response.raise_for_status()
    artist_data = response.json()

    followers = artist_data['followers']['total']
    genres = artist_data.get('genres', [])
    return followers, genres

def get_decade(release_date):
    try:
        year = int(release_date.split('-')[0])
        decade = (year // 10) * 10
        return f"{decade}s"
    except (ValueError, IndexError):
        return None

def fetch_spotify_data(access_token, num_tracks=700):
    search_url = 'https://api.spotify.com/v1/search'
    headers = {'Authorization': f'Bearer {access_token}'}

    all_tracks_data = []
    limit = 50
    offset = 0

    while len(all_tracks_data) < num_tracks:
        if offset > 1000:
            offset = 0

        current_year = datetime.now().year
        search_year = current_year - (len(all_tracks_data) % (current_year - 1960))

        params = {
            'q': f'year:{search_year}',
```

EDA RESULTS

	artist
0	Taylor Swift
1	Taylor Swift
2	Taylor Swift
3	Taylor Swift
4	Taylor Swift
5	Taylor Swift
6	Tate McRae
7	Sabrina Carpenter
8	Taylor Swift
9	Rumi
10	HUNTR/X
11	Taylor Swift
12	Olivia Dean
13	Taylor Swift
14	HUNTR/X
15	Saja Boys
16	Sabrina Carpenter
17	Saja Boys
18	Dream Supplier
19	Taylor Swift
20	HUNTR/X
21	Tyler, The Creator
22	Taylor Swift
23	HUNTR/X
24	Taylor Swift
25	Eminem
26	Linkin Park
27	Furacão 2000
28	Sade
29	Coldplay
30	Nelly
31	Aaron Y Su Grupo Ilusion
32	Fuel
33	Mindless Self Indulgence
34	The Offspring
35	U2
36	Deftones
37	Limp Bizkit
38	Air
39	Sade
40	Ludacris
41	A Perfect Circle
42	Linkin Park
43	Radiohead
44	Alice Deejay

	track	genre
0	The Fate of Ophelia	N/A
1	Honey	N/A
2	Opalite	N/A
3	Actually Romantic	N/A
4	Wi$h Li$t	N/A
5	Wood	N/A
6	TIT FOR TAT	N/A
7	Tears	pop
8	CANCELLED!	N/A
9	Free	N/A
10	How It’s Done	k-pop
11	Eldest Daughter	N/A
12	Man I Need	pop soul
13	Father Figure	N/A
14	Takedown	k-pop
15	Soda Pop	N/A
16	When Did You Get Hot?	pop
17	Your Idol	N/A
18	Clean Baby Sleep White Noise (Loopable no fade)	N/A
19	Elizabeth Taylor	N/A
20	What It Sounds Like	k-pop
21	Sugar On My Tongue	N/A
22	Ruin The Friendship	N/A
23	Golden	k-pop
24	The Life of a Showgirl (feat. Sabrina Carpenter)	N/A
25	Kill You	rap
26	By Myself	nu metal
27	Mimosa 2000	funk melody
28	Lovers Rock	N/A
29	Trouble	N/A
30	E.I.	N/A
31	Todo Me Gusta De Ti	cumbia
32	Hemorrhage (In My Hands)	post-grunge
33	Bitches	N/A
34	Want You Bad	punk
35	Beautiful Day	rock
36	Passenger	nu metal
37	My Generation	nu metal
38	Playground Love (with Gordon Tracks)	trip hop
39	By Your Side	N/A
40	What's Your Fantasy (Featuring Shawna)	southern hip hop
41	Judith	alternative metal
42	Points of Authority	nu metal
43	How to Disappear Completely	art rock
44	Better Off Alone	eurodance

	popularity	followers	decade
0	98	144684817	2020s
1	92	144684817	2020s
2	96	144684817	2020s
3	94	144684817	2020s
4	94	144684817	2020s
5	94	144684817	2020s
6	91	9111296	2020s
7	93	26182086	2020s
8	94	144684817	2020s
9	92	195501	2020s
10	94	3261865	2020s
11	93	144684817	2020s
12	94	1312383	2020s
13	95	144684817	2020s
14	92	3261865	2020s
15	95	2094993	2020s
16	92	26182086	2020s
17	95	2094993	2020s
18	91	12311	2020s
19	96	144684817	2020s
20	93	3261865	2020s
21	91	24270773	2020s
22	93	144684817	2020s
23	100	3261865	2020s
24	94	144684817	2020s
25	69	104182027	2000s
26	69	31615606	2000s
27	78	272990	2000s
28	69	5066720	2000s
29	75	60596972	2000s
30	65	3713874	2000s
31	71	861289	2000s
32	68	928275	2000s
33	69	1596906	2000s
34	73	6613487	2000s
35	64	13564879	2000s
36	71	7648991	2000s
37	75	8075309	2000s
38	71	1154185	2000s
39	74	5066720	2000s
40	66	3417995	2000s
41	66	2065897	2000s
42	71	31615606	2000s
43	73	13692686	2000s
44	79	209684	2000s
45	74	104182027	2000s

SPOTIFY EDA

Hypothesis 1

H_0 (Null Hypothesis): There is no significant difference in the average Spotify popularity scores of pop songs released in the 2010s and those released in the 2020s.

H_1 (Alternative Hypothesis): Pop songs released in the 2020s have significantly higher average Spotify popularity scores than those released in the 2010s.

Why this Hypothesis?

Viral Platforms

Platforms like TikTok create viral hits overnight, giving 2020s songs a rapid popularity boost that didn't exist in the same way in the 2010s.

Larger Audience

The global streaming audience is much bigger now, so new songs are released to more listeners from day one.

Algorithmic Bias

Spotify's algorithms and major playlists (like "Today's Top Hits") are designed to promote new music, giving recent songs an inherent advantage.

DATA APPROACH

```python
def get_popularity(track):
    return track["popularity"]

def get_pop_songs(decade_query):
    all_tracks = []
    for offset in range(0, 1000, 50):  # get up to 1000 results
        params = {
            "q": decade_query,
            "type": "track",
            "limit": 50,
            "offset": offset
        }
        r = requests.get("https://api.spotify.com/v1/search", headers=headers,
        params=params)
        data = r.json()
        if not data.get("tracks"):
            break
        all_tracks.extend(data["tracks"]["items"])

    all_tracks.sort(key=get_popularity, reverse=True)

    return all_tracks[:350]


pop_2010s = get_pop_songs("genre:pop year:2010-2019")
pop_2020s = get_pop_songs("genre:pop year:2020-2025")

min_len = min(len(pop_2010s), len(pop_2020s))
pop_2010s = pop_2010s[:min_len]
pop_2020s = pop_2020s[:min_len]
```

Function to fetch the data

```python
def create_dataframe(tracks, decade):
    return pd.DataFrame({
        "Track": [t["name"] for t in tracks],
        "Artist": [t["artists"][0]["name"] if t["artists"] else "Unknown
        Artist" for t in tracks],
        "Popularity": [t["popularity"] for t in tracks],
        "Decade": decade
    })

df_2010s = create_dataframe(pop_2010s, "2010s")
df_2020s = create_dataframe(pop_2020s, "2020s")
df = pd.concat([df_2010s, df_2020s], ignore_index=True)
print("\n")
df.info()
df.isnull().sum()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 0 entries
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype
---  ------      --------------  -----
 0   Track       0 non-null      float64
 1   Artist      0 non-null      float64
 2   Popularity  0 non-null      float64
 3   Decade      0 non-null      object
dtypes: float64(3), object(1)
memory usage: 132.0+ bytes
```

	0
Track	0
Artist	0
Popularity	0
Decade	0

dtype: int64

Create Data Frame

DATA APPROACH

```
df_2010s = df_2010s.drop_duplicates(subset=["Track", "Artist"])
df_2020s = df_2020s.drop_duplicates(subset=["Track", "Artist"])
```

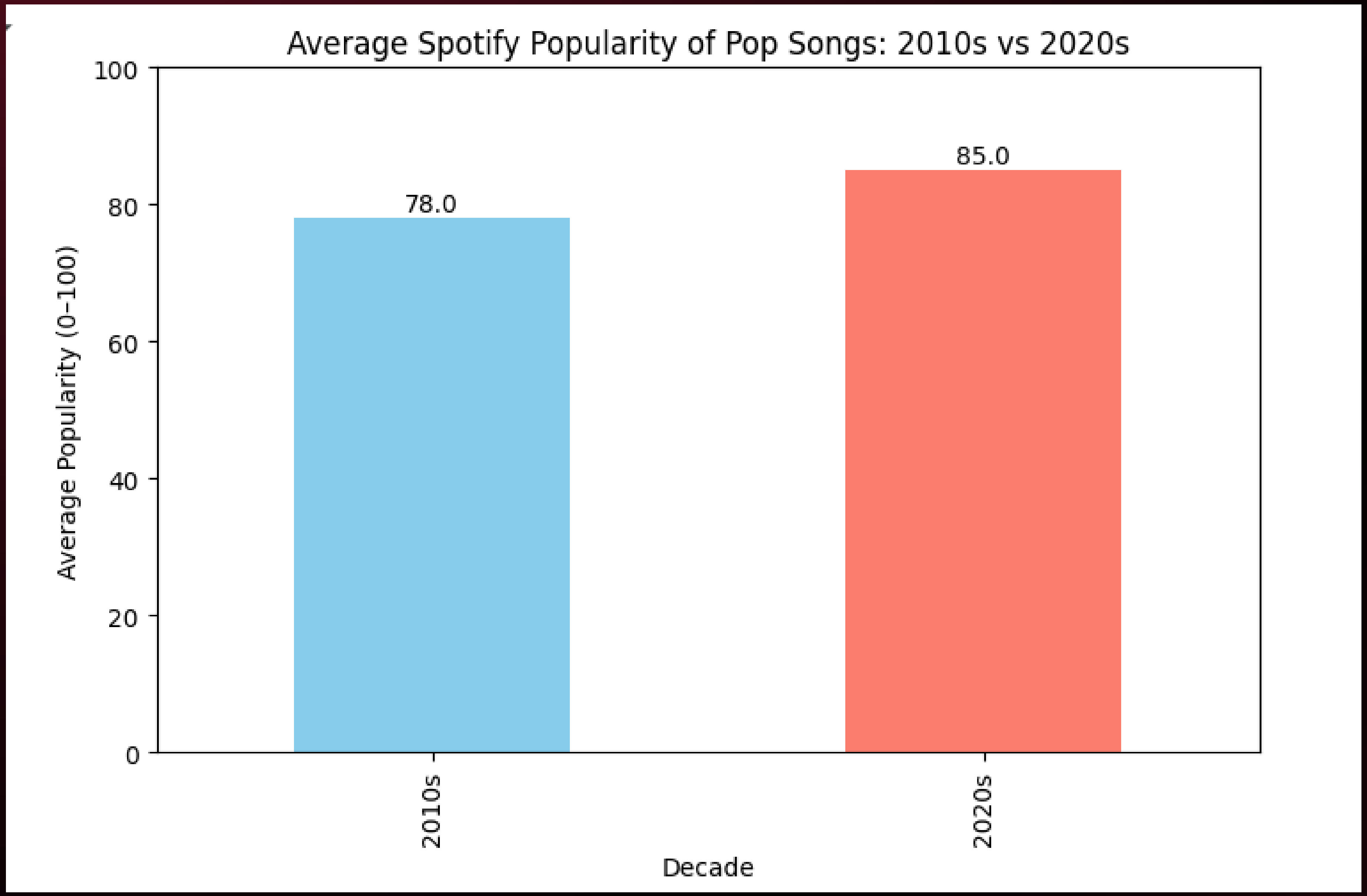
```
t_stat, p_value = stats.ttest_ind(df_2020s["Popularity"], df_2010s
["Popularity"], equal_var=False)
alpha = 0.05

print("\nHypothesis Test Result")
print(f"T-statistic: {t_stat:.4f}")
print(f"P-value: {p_value/2:.100f}  (one-tailed test)")

if (t_stat > 0) and (p_value/2 < alpha):
    print("Reject H₀ -> 2020s pop songs are significantly more popular.")
else:
    print("Fail to reject H₀ → No significant difference in popularity.")
print("Conclusion: There is a strong evidence that 2020s pop songs are
significantly more popular than 2010s pop songs.")
```

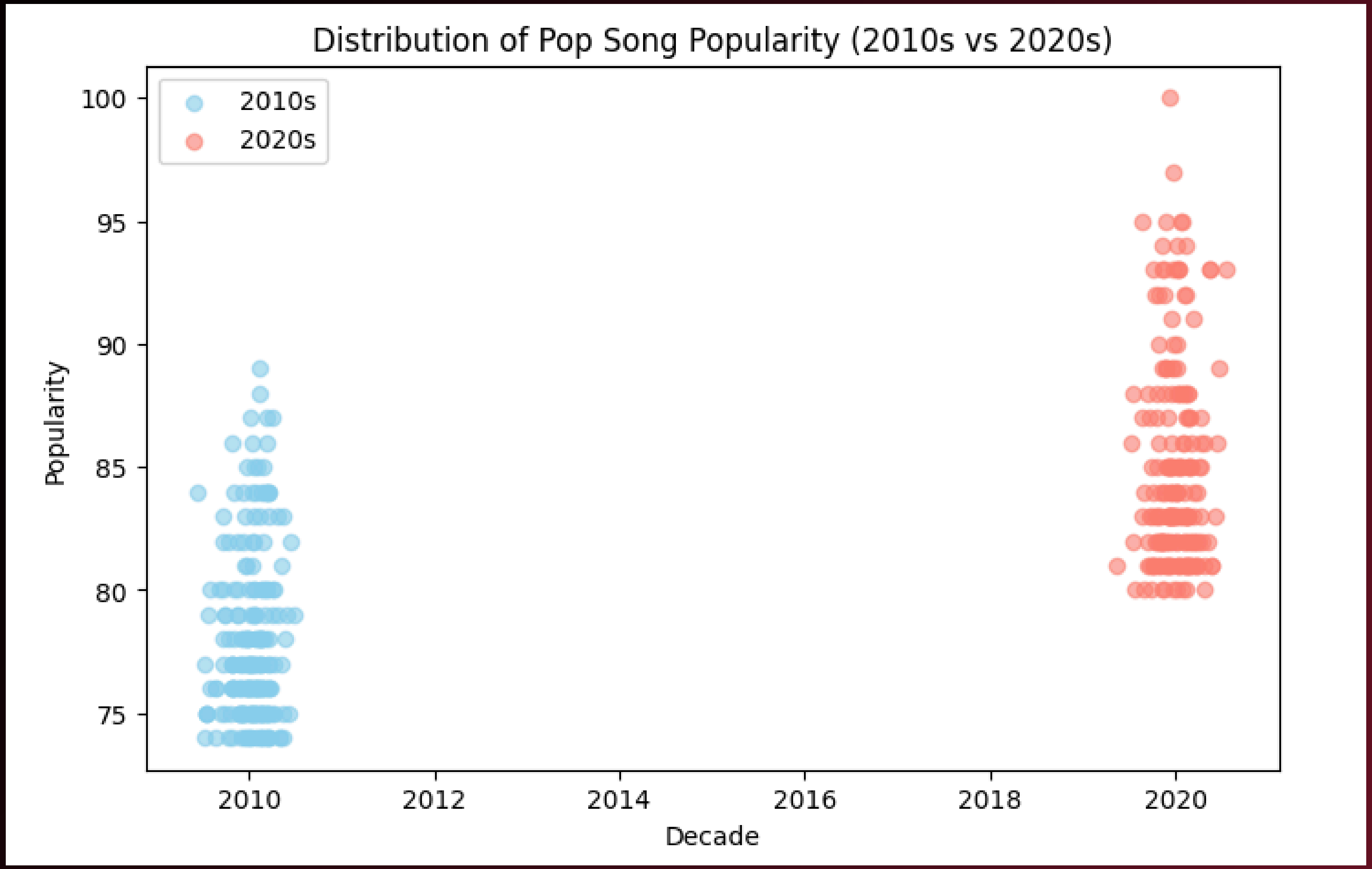
VISUALIZATION

```
avg_2010s = df_2010s["Popularity"].mean()
avg_2020s = df_2020s["Popularity"].mean()

print(f"Average popularity of Pop songs (2010s): {avg_2010s:.2f}")
print(f"Average popularity of Pop songs (2020s): {avg_2020s:.2f}")
plt.figure(figsize=(8,5))
df.groupby("Decade")["Popularity"].mean().plot(kind="bar", color=["skyblue",
"salmon"])
plt.title("Average Spotify Popularity of Pop Songs: 2010s vs 2020s")
plt.ylabel("Average Popularity (0-100)")
plt.ylim(0, 100)
for i, v in enumerate([avg_2010s, avg_2020s]):
    plt.text(i, v + 1, f"{v:.1f}", ha='center')
plt.show()

plt.figure(figsize=(8,5))
plt.scatter(np.random.normal(2010, 0.2, len(df_2010s)), df_2010s["Popularity"],
alpha=0.6, label="2010s", color="skyblue")
plt.scatter(np.random.normal(2020, 0.2, len(df_2020s)), df_2020s["Popularity"],
alpha=0.6, label="2020s", color="salmon")
plt.title("Distribution of Pop Song Popularity (2010s vs 2020s)")
plt.xlabel("Decade")
plt.ylabel("Popularity")
plt.legend()
plt.show()
```

Results

```
Hypothesis Test Result
T-statistic: 19.6115
P-value: 0.0000000000000000000000000000000000000000000000000000000000000000000235636696333687947612466964744428  (one-tailed test)
Reject Hₒ -> 2020s pop songs are significantly more popular.
Conclusion: There is a strong evidence that 2020s pop songs are significantly more popular than 2010s pop songs.
```





Hypothesis 2

H1 : Multi-genre artists have more followers than single-genre artists.

H0 : There is a no correlation between the number of genres an artist has and their number of followers.

Why this Hypothesis?

Wider Viral Appeal

Multi-genre songs often have broader appeal on platforms like TikTok, allowing them to fit into more trends and reach more diverse communities.

Broader Audience Reach

By blending genres, artists can tap into multiple, distinct fanbases at the same time, creating a larger combined audience.

Cross-Playlist Promotion

Spotify's algorithms can place multi-genre artists on a wider variety of popular playlists, potentially increasing their visibility and introducing them to different listener groups.

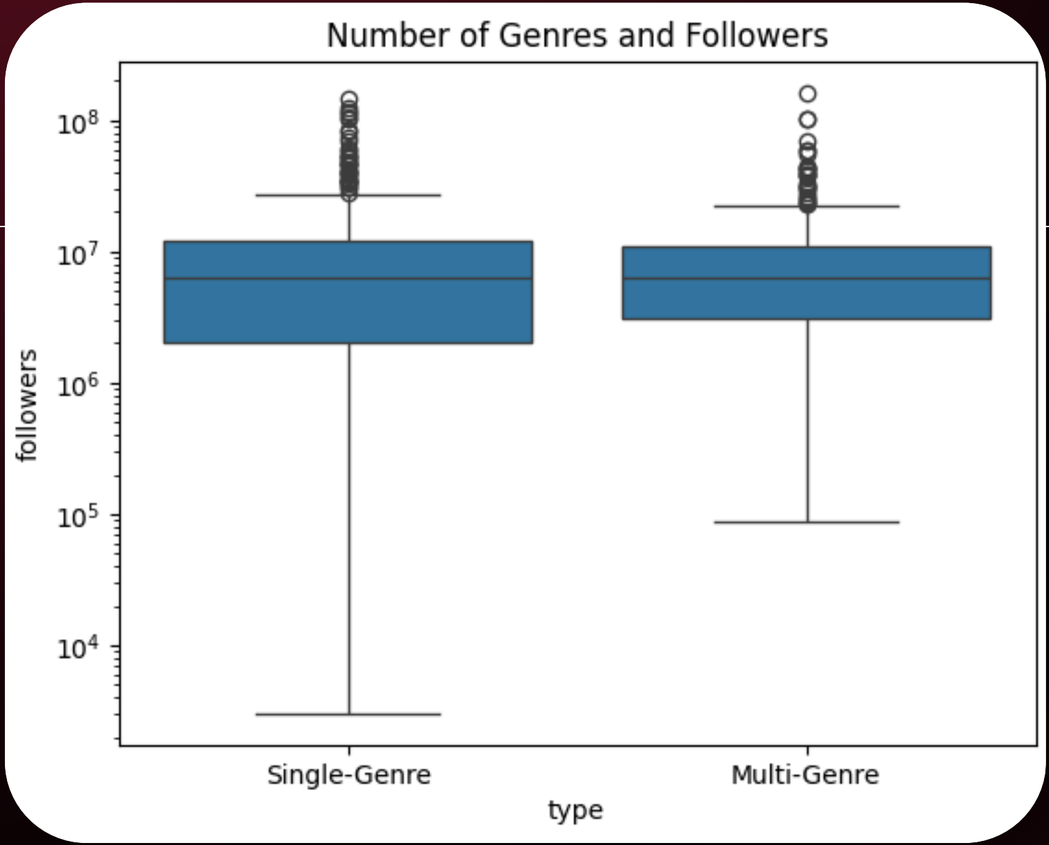
Data Approach

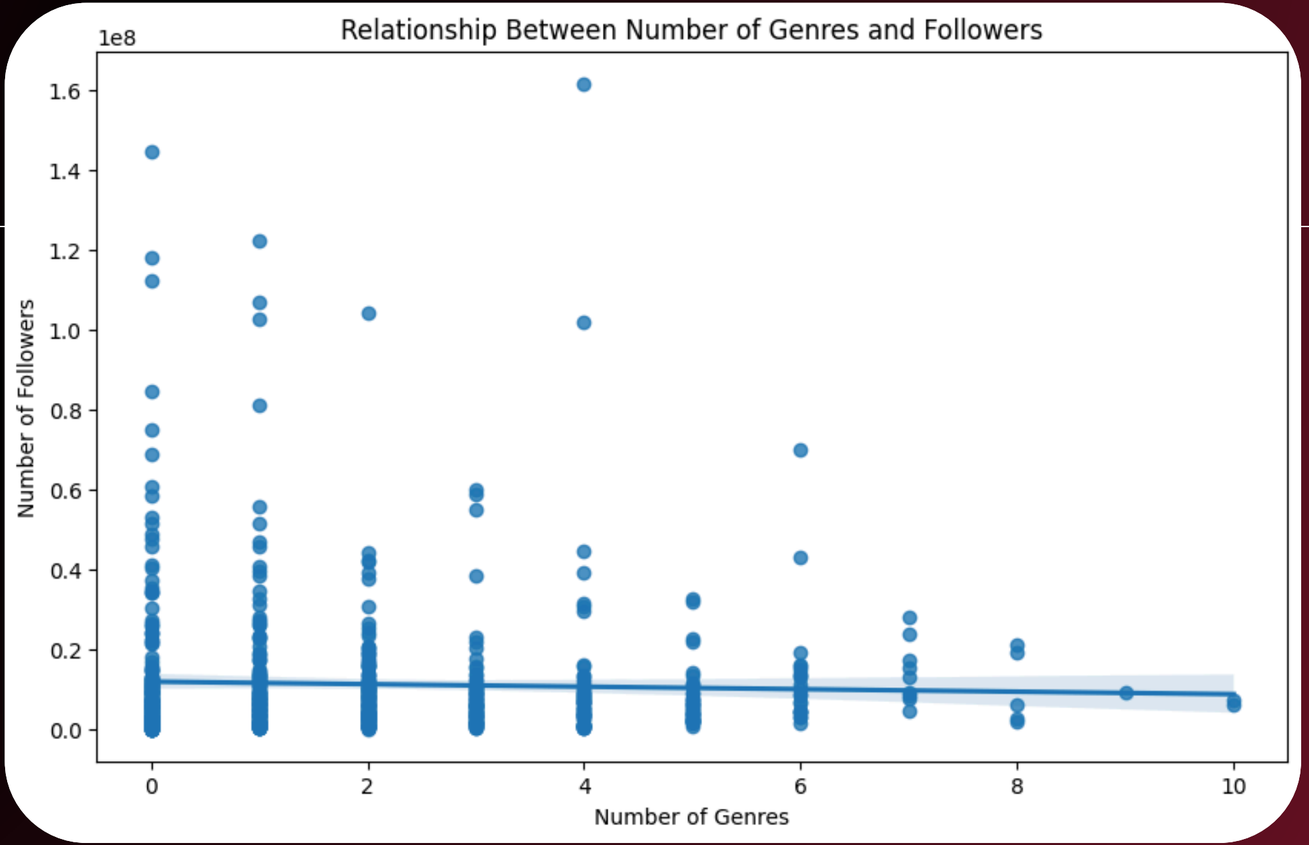
```
single_genre = []
multi_genre = []
offset = 0
target = 350

while len(single_genre) < target or len(multi_genre) < target:
    search_params = {'q': 'year:2024', 'type': 'artist', 'limit': 50, 'offset': offset}
    search_results = requests.get(
        'https://api.spotify.com/v1/search',
        headers=api_headers,
        params=search_params
    ).json()

    artist_batch = search_results.get('artists', {}).get('items', [])
    if not artist_batch:
        print("No more artists found.")
        break

    for artist in artist_batch:
        genres_count = len(artist.get('genres', []))
        artist_info = {
            'name': artist.get('name'),
            'followers': artist.get('followers', {}).get('total', 0),
            'genres_count': genres_count
        }

        if genres_count < 2 and len(single_genre) < target:
            single_genre.append(artist_info)
        elif genres_count >= 2 and len(multi_genre) < target:
            multi_genre.append(artist_info)

    offset += 50
```

```
group1 = df[df['type'] == 'Multi-Genre']['followers']
group2 = df[df['type'] == 'Single-Genre']['followers']

t_stat, p_value = stats.ttest_ind(group1, group2, alternative='greater', equal_var=False)

print("--- T-Test Results ---")
print(f"P-value: {p_value:.4f}")

if p_value < 0.05:
    print("Conclusion:We reject the null hypothesis. The data supports that multi-genre artists have more followers. ")
else:
    print("Conclusion:We fail to reject the null hypothesis.")
    print("We don't have enough evidence to say multi-genre artists have more followers. ")
```

Result

```
--- T-Test Results ---
P-value: 0.9544
Conclusion:We fail to reject the null hypothesis.
We don't have enough evidence to say multi-genre artists have more followers.
```





CONCLUSION

- Based on Hypothesis 1, we have enough evidence to support that pop songs in 2020s are significantly more popular than pop songs in 2010s.
- Based on Hypothesis 2, we do not have enough evidence to support that multi-genre artists have more followers than single-genre artists.

REFERENCE

- AB, S. (2025). WEB API : SPOTIFY FOR DEVELOPERS. HTTPS://DEVELOPER.SPOTIFY.COM/DOCUMENTATION/WEB-API

Thankyou