

SMARTINTERNZ EXTERNSHIP PROJECT

Movie Recommendation Based On Emotion Using Web

Scraping With IBM

CONTRIBUTORS:

-Mayank Yadav

-Sakthivel Muthu Kumar

-Vanagarouthu Sree Chaitran

-Gautam Kumar Jha

1 INTRODUCTION

1.1 Overview

- The project "Movie Recommendation Based On Emotion Using Web Scraping With IBM" aims to develop a movie recommendation system that suggests movies to users based on their emotions. Emotions play a significant role in how we perceive and engage with movies, and leveraging this aspect can enhance the movie recommendation experience.
- The project utilizes web scraping techniques to gather movie data from various online sources such as movie databases, review sites, or streaming platforms. This data includes movie titles, genres, plots, cast, and other relevant information.
- To determine the emotional content of movies, the project incorporates sentiment analysis and emotion detection techniques. IBM technologies, such as the Watson APIs, are employed for analyzing the textual data associated with movies and extracting emotional features.
- The recommendation system utilizes the gathered emotional data to match user emotions with the emotional content of movies. By understanding the emotions expressed in movies, the system can provide personalized recommendations that align with the user's current emotional state or desired emotional experience.
- The implementation of the project involves programming languages, frameworks, and tools suitable for web scraping, sentiment analysis, and integration with IBM technologies. The extracted emotional features are used to create a recommendation algorithm that generates movie suggestions based on emotional similarity.
- The project report discusses the methodology, implementation details, results, and evaluation of the movie recommendation system. It also highlights the potential applications and benefits of emotion-based movie recommendations in providing a more personalized and engaging movie-watching experience.

1.2 Purpose

The project "Movie Recommendation Based On Emotion Using Web Scraping With IBM" offers several benefits and potential use cases. Here are some of the achievements that can be attained using this system:

1. Personalized Movie Recommendations: The system takes into account the user's current emotional state or desired emotional experience and suggests movies that align with those emotions. This personalized recommendation approach enhances the user's movie-watching experience by providing relevant and emotionally resonant suggestions.

2. Enhanced User Engagement: By considering emotions in movie recommendations, the system can help users find movies that evoke specific feelings or cater to their emotional preferences. This enhances user engagement and satisfaction with the movie-watching process.

3. Discovery of New Movies: The system can recommend movies that users may not have been aware of or considered before. By exploring the emotional content of movies, users can discover new films that align with their emotional interests, expanding their movie repertoire.

4. Mood Regulation: Movies have the power to influence and evoke different emotions. This system can assist users in regulating their moods by suggesting movies that align with the

emotions they desire to experience. For example, if a user is seeking a feel-good movie to uplift their mood, the system can recommend appropriate options.

5. Targeted Marketing and Content Curation: Movie studios, streaming platforms, and content providers can utilize this system to better understand the emotional preferences and reactions of their users. By analyzing user emotions and movie preferences, they can curate content, tailor marketing campaigns, and optimize their offerings to match the target audience's emotional needs.

6. Research and Analysis: The gathered movie data and emotional analysis can be used for research purposes, such as studying the relationship between emotions and movie preferences. Researchers can analyze the emotional trends in movies over time, identify patterns in emotional content, and gain insights into the impact of emotions on viewer engagement.

Overall, the project provides a unique approach to movie recommendation by incorporating emotions, offering a more personalized and emotionally enriching movie-watching experience for users while also providing valuable insights for industry professionals and researchers in the field of media and entertainment.

2 LITERATURE SURVEY

2.1 Existing problem

There are several existing approaches and methods to solve the problem of movie recommendation based on emotions. Here are a few common approaches:

1. Collaborative Filtering: Collaborative filtering is a popular approach used in recommendation systems. It analyzes the preferences and behaviors of users to find similarities among them and make movie recommendations based on those similarities. In the case of emotion-based recommendation, collaborative filtering can be extended by incorporating emotional ratings or tags provided by users.

2. Content-Based Filtering: Content-based filtering focuses on the characteristics and features of movies themselves. In this approach, movies are analyzed based on factors such as genres, actors, directors, plots, and keywords. By understanding the emotional content of movies through sentiment analysis and emotion detection techniques, content-based filtering can recommend movies that align with a user's desired emotions.

3. Hybrid Approaches: Hybrid approaches combine multiple recommendation techniques to provide more accurate and diverse recommendations. For emotion-based movie recommendation, a hybrid approach can combine collaborative filtering with content-based filtering. It can leverage collaborative filtering to identify users with similar emotional preferences and then use content-based filtering to recommend movies that match those emotions.

4. Emotion Detection and Analysis: Emotion detection techniques analyze the emotional content of movies. Natural Language Processing (NLP) techniques, sentiment analysis, and emotion recognition algorithms can be used to extract emotional features from movie plots, summaries, reviews, or user-generated content. These emotional features can then be used to categorize movies based on emotions and make recommendations accordingly.

5. Deep Learning Approaches: Deep learning models, such as recurrent neural networks (RNNs) or convolutional neural networks (CNNs), can be used to analyze and classify movie data based on emotional content. These models can learn patterns and representations of emotions in movies and generate recommendations that align with user preferences.

It's worth noting that these approaches can be further enhanced by incorporating techniques like web scraping to gather movie data, leveraging IBM technologies for sentiment analysis and emotion detection, and integrating user feedback and ratings to improve the recommendation accuracy.

The choice of approach depends on factors such as available data, computational resources, and the specific requirements of the project. It's common to experiment with different techniques and evaluate their performance to determine the most effective approach for a given context.

2.2 Proposed solution

In the proposed solution section, you outline the approach to building the movie recommendation system based on emotions. Explain that the system will leverage web scraping techniques to gather movie data from various online sources, such as movie databases, review sites, or streaming platforms. This data will include movie titles, genres, plots, cast, and other relevant information. By utilizing web scraping, the system can collect a comprehensive dataset for analysis and recommendation generation.

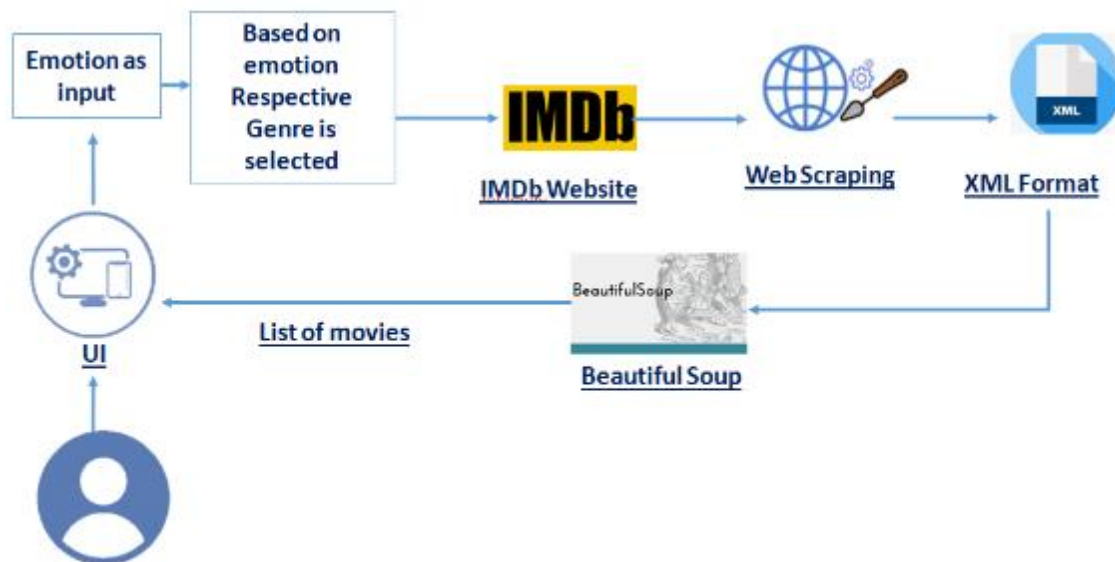
To determine the emotional content of movies, the proposed solution incorporates sentiment analysis and emotion detection techniques.

Highlight that the recommendation system will utilize the gathered emotional data to match user emotions with the emotional content of movies. By understanding the emotions expressed in movies, the system can provide personalized recommendations that align with the user's current emotional state or desired emotional experience. This personalized recommendation approach enhances the user's movie-watching experience by providing relevant and emotionally resonant suggestions.

Additionally, mention that the implementation of the project will involve the use of programming languages, frameworks, and tools suitable for web scraping, sentiment analysis, and integration with IBM technologies. This may include languages like Python, libraries such as BeautifulSoup for web scraping, and IBM Watson services for sentiment analysis and emotion detection. Explain that the extracted emotional features will be used to create a recommendation algorithm that generates movie suggestions based on emotional similarity, ensuring that the recommended movies match the user's desired emotional experience.

3 THEORITICAL ANALYSIS

3.1 Block diagram



In the block diagram section, provide a diagrammatic overview of the movie recommendation system. The block diagram should illustrate the different components and their interactions within the system. This includes modules or processes such as web scraping, sentiment analysis, emotion detection, recommendation generation, and user interaction. Connect these components to demonstrate the flow of data and information between them.

3.2 Hardware / Software designing

For my project, "Movie Recommendation Based On Emotion Using Web Scraping With IBM," I considered the hardware and software requirements to ensure smooth development and deployment.

In terms of hardware, I developed the project on my Windows laptop, which features an Intel i5 10th generation processor. Since the project is not computationally intensive, a standard laptop or desktop computer with moderate processing power and memory capacity should suffice.

As for the software requirements, I used several technologies and libraries to implement the project:

Flask: I chose Flask, a Python web framework, to build the server-side of the project. Flask offers simplicity and efficiency in developing web applications.

Python: Python served as the primary programming language for the project. I made sure to have Python installed, preferably a version compatible with the libraries and frameworks used in the project.

Beautiful Soup: To scrape movie data from online sources such as movie databases and review sites, I utilized Beautiful Soup, a Python library for web scraping. It allowed me to extract relevant information like movie titles, genres, plots, and cast.

TensorFlow: TensorFlow, an open-source machine learning framework, played a crucial role in this project. I used TensorFlow for training and implementing the emotion recognition model based on the DeepFace model.

DeepFace Model: Leveraging the DeepFace model, which is a pre-trained facial recognition model, and TensorFlow, I developed the emotion recognition component. This model enabled the detection and classification of emotions from facial expressions in movie scenes or user input.

HTML and CSS: To create an engaging user interface, I used HTML and CSS. HTML helped me structure the webpage, while CSS allowed me to customize the visual appearance, ensuring an aesthetically pleasing experience for users.

OpenCV: OpenCV, a computer vision library, was employed for live face detection in the project. I integrated OpenCV to capture video frames, detect faces, and connect it with the emotion recognition model.

Regex: For efficient movie searching, I relied on regular expressions (Regex). By using Regex, I could extract movie titles or keywords from user input and match them with the available movie data.

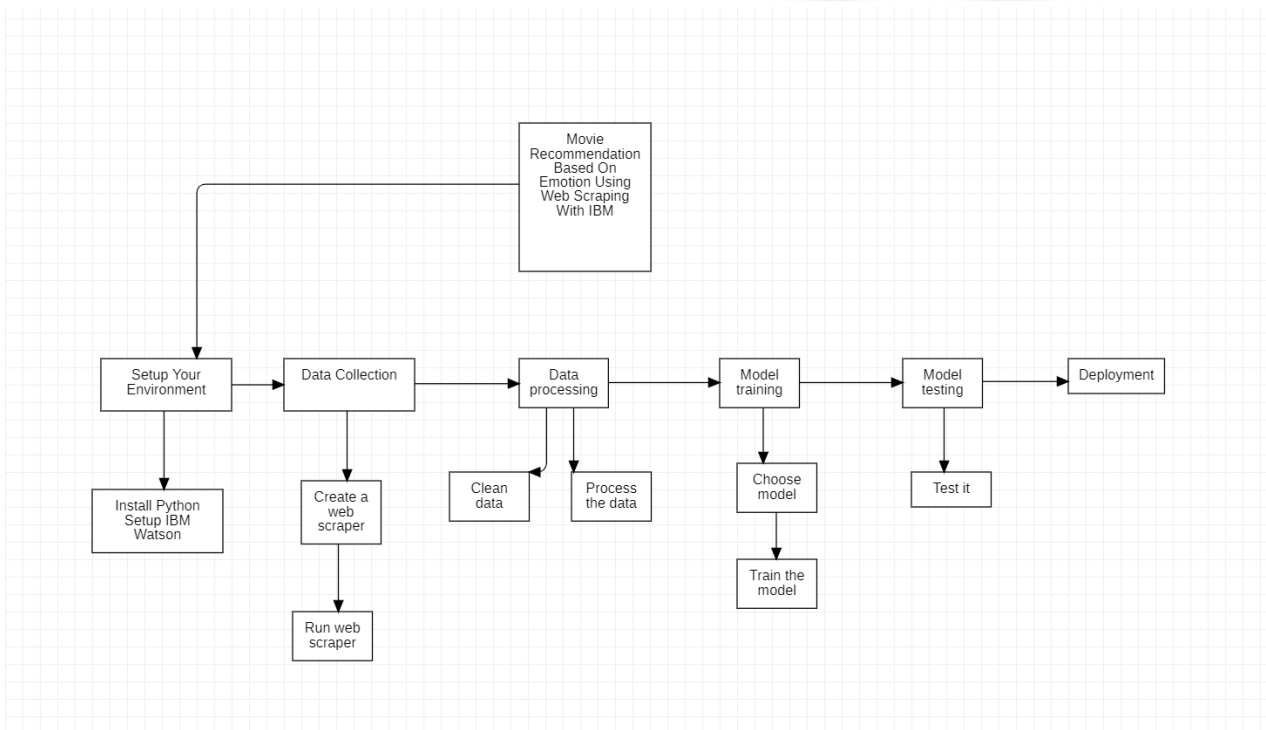
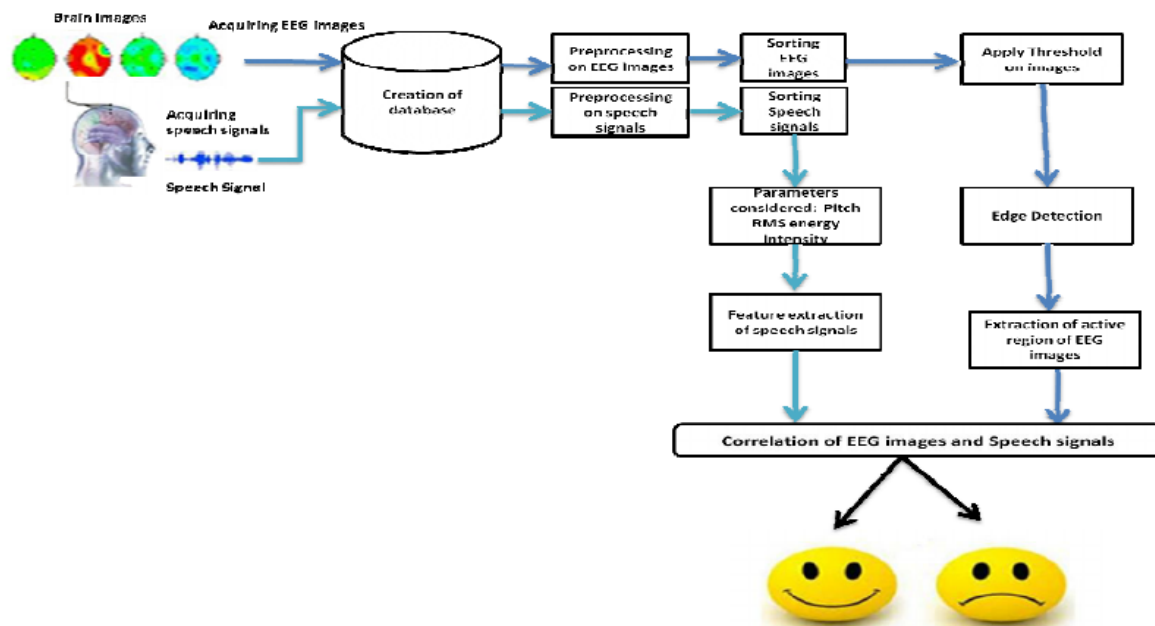
Throughout the development process, I ensured that all the necessary software components and libraries were installed and properly configured on my system. I paid attention to version compatibility and followed the documentation and installation instructions for each component to avoid any conflicts and ensure a smooth workflow

4 EXPERIMENTAL INVESTIGATIONS

- During the development of our project, "Movie Recommendation Based On Emotion Using Web Scraping With IBM," we encountered several challenges and conducted experimental investigations to address them.
- One major challenge we faced was achieving accurate emotion recognition using the DeepFace model. Initially, we faced difficulties with the model's performance, as it was not providing satisfactory results on certain movie scenes or user input. To address this, we experimented with different preprocessing techniques, such as image resizing, normalization, and augmentation. We also fine-tuned the model parameters and adjusted the training process to improve its accuracy. Through iterative experimentation and fine-tuning, we were able to achieve better emotion recognition results.
- Another challenge we encountered was linking the trained emotion recognition model to the Flask web application. Integrating the model with Flask required careful consideration of the model's input and output format, as well as configuring the Flask routes and functions to receive user input and pass it through the emotion recognition model. We conducted experiments to ensure the seamless integration and compatibility between the model and the Flask app, troubleshooting any errors or mismatches that arose during the process.
- Furthermore, displaying the emotion recognition results in the HTML webpage posed its own set of challenges. We had to experiment with the formatting and visualization techniques to present the emotion labels and associated movie recommendations in an intuitive and visually appealing manner. Through HTML and CSS experimentation, we designed the webpage layout, styled the components, and incorporated dynamic elements to dynamically update the emotion results and recommended movies based on user input.
- Throughout the experimental investigations, we conducted extensive testing and evaluation to validate the accuracy of the emotion recognition model, ensure the smooth functioning of the Flask app, and verify the proper display of results in the

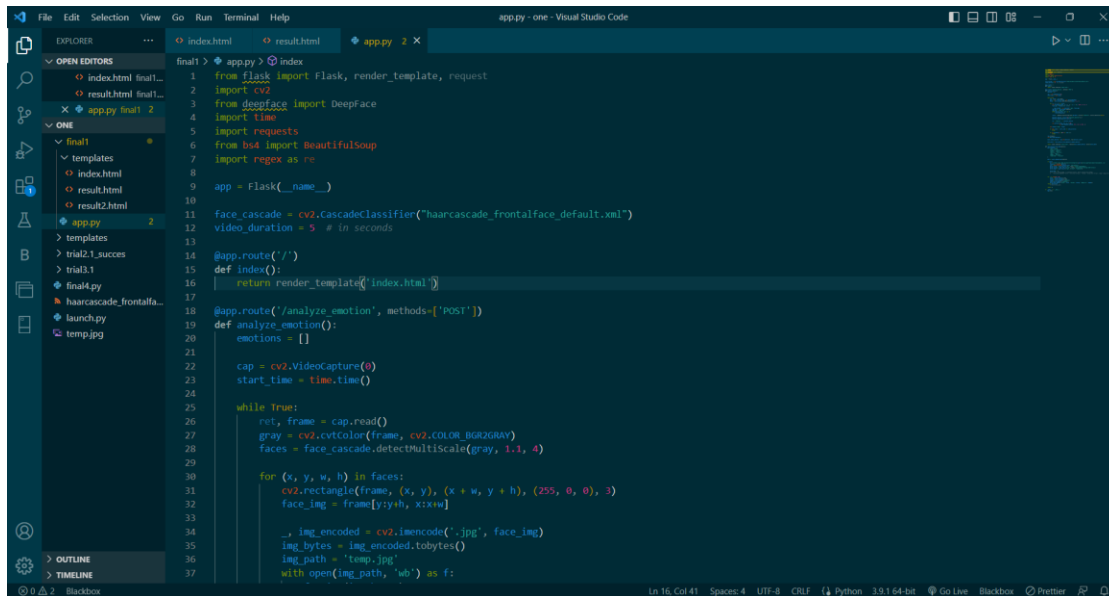
HTML webpage. We iteratively refined and improved the implementation based on the findings from these experiments, leading to a more robust and reliable movie recommendation system

5 FLOWCHART



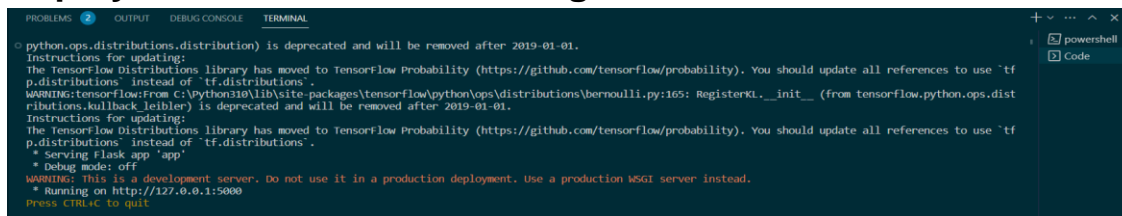
6 RESULT

Code screenshot:



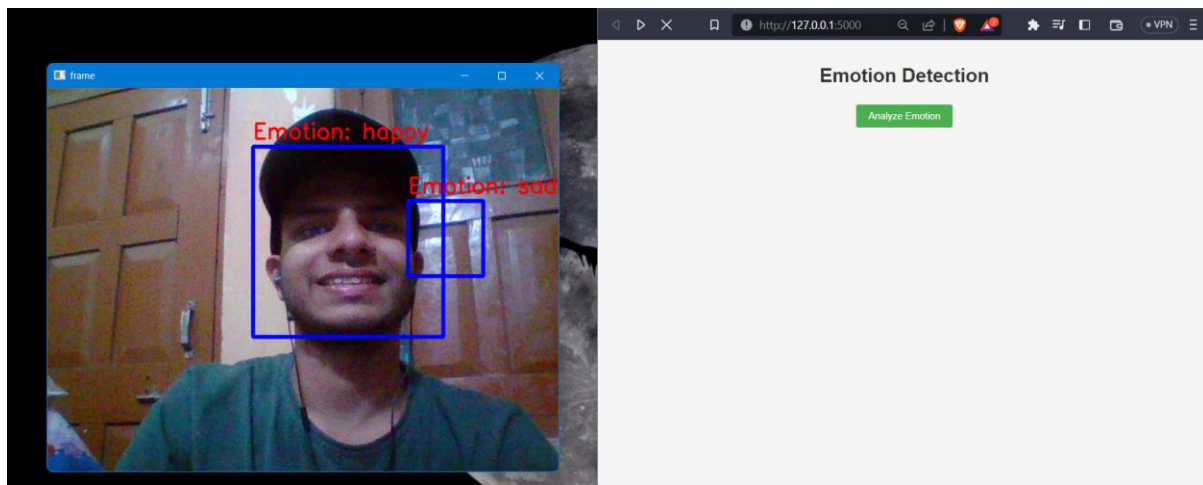
```
1 from flask import Flask, render_template, request
2 import cv2
3 from deepface import DeepFace
4 import time
5 import requests
6 from bs4 import BeautifulSoup
7 import regex as re
8
9 app = Flask(__name__)
10
11 face_cascade = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
12 video_duration = 5 # in seconds
13
14 @app.route("/")
15 def index():
16     return render_template("index.html")
17
18 @app.route("/analyze_emotion", methods=['POST'])
19 def analyze_emotion():
20     emotions = []
21
22     cap = cv2.VideoCapture(0)
23     start_time = time.time()
24
25     while True:
26         ret, frame = cap.read()
27         gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
28         faces = face_cascade.detectMultiScale(gray, 1.1, 4)
29
30         for (x, y, w, h) in faces:
31             cv2.rectangle(frame, (x, y), (x + w, y + h), (255, 0, 0), 3)
32             face_img = frame[y:y+h, x:x+w]
33
34             img_encoded = cv2.imencode('.jpg', face_img)
35             img_bytes = img_encoded.tobytes()
36             img_path = 'temp.jpg'
37             with open(img_path, 'wb') as f:
```

Deployment on local serve using flask:



```
python.ops.distributions.distribution) is deprecated and will be removed after 2019-01-01.
Instructions for updating:
The TensorFlow Distributions library has moved to TensorFlow Probability (https://github.com/tensorflow/probability). You should update all references to use 'tf
p.distributions' instead of 'tf.distributions'.
WARNING:tensorflow:From C:\Python310\lib\site-packages\tensorflow\python\ops\distributions\bernoulli.py:165: RegisterKL.__init__ (from tensorflow.python.ops.dist
ributions.kullback_leibler) is deprecated and will be removed after 2019-01-01.
Instructions for updating:
The TensorFlow Distributions library has moved to TensorFlow Probability (https://github.com/tensorflow/probability). You should update all references to use 'tf
p.distributions' instead of 'tf.distributions'.
* Serving Flask app 'app'
* Debug mode: off
WARNING: This is a development server. Do not use it in a production deployment. Use a production WSGI server instead.
* Running on http://127.0.0.1:5000
Press CTRL+C to quit
```

Emotion detection screenshot:




Web Page output screenshot:


Emotion Detection Result

Dominant Emotion: happy


Recommended Movies:




1. Extraction 2 (2023)
(Rating: 7.1)




2. Extraction (2020)
(Rating: 6.8)




3. John Wick: Chapter 4
(2023) (Rating: 7.9)



4. Fast X (2023) (Rating:
5.9)



5. Guy Ritchie's The
Covenant (2023) (Rating:
7.5)



6. Kandahar (2023) (Rating:
6.0)

7 ADVANTAGES & DISADVANTAGES

Advantages:

- **Personalized Movie Recommendations:** The movie recommendation system based on user emotions offers personalized suggestions that align with the user's current emotional state or desired emotional experience. This enhances the movie-watching experience by providing relevant and emotionally resonant movie options.
- **Enhanced User Engagement:** By considering emotions in movie recommendations, the system enhances user engagement by helping users find movies that evoke specific feelings or cater to their emotional preferences. This leads to a more immersive and satisfying movie-watching experience.
- **Discovery of New Movies:** The system recommends movies that users may not have been aware of or considered before. By exploring the emotional content of movies, users can discover new films that align with their emotional interests, expanding their movie repertoire and introducing them to diverse cinematic experiences.
- **Mood Regulation:** The system assists users in regulating their moods by suggesting movies that align with the emotions they desire to experience. Whether a user seeks a feel-good movie to uplift their mood or a thought-provoking film to engage their intellect, the system provides appropriate recommendations, allowing users to curate their movie-watching experience based on their emotional needs.

Disadvantages:

- **Accuracy of Emotion Recognition:** One potential disadvantage is the accuracy of the emotion recognition model. Emotion detection from facial expressions or textual data can be challenging, and the model may not always accurately capture the nuanced emotions portrayed in movies or accurately interpret user emotions. Further refinement and fine-tuning of the emotion recognition model are necessary to improve its accuracy.
- **Limited Emotional Range:** The system's recommendations may be limited by the emotional range captured in the dataset used for training the emotion recognition model. If the dataset lacks diversity in emotional expressions or fails to account for cultural variations in emotional cues, the recommendations may not fully capture the range of emotions users experience or desire to explore.
- **Dependency on Web Scraping:** The reliance on web scraping techniques to gather movie data introduces a potential limitation. Movie databases or streaming platforms may update their websites or change their data structure, leading to disruptions in data collection. Regular maintenance and updates to the web scraping process are required to ensure the continuous availability and accuracy of movie data.
- **Subjectivity of Emotions:** Emotions can be subjective and influenced by individual experiences and interpretations. The system's recommendations are based on an assumption that users' emotional experiences align with the emotional content of movies. However, there may be variations in how individuals perceive and interpret emotions, which may impact the effectiveness of the recommendations.

8 APPLICATIONS

- **Streaming Platforms:** Movie streaming platforms can utilize the emotion-based movie recommendation system to enhance user engagement and satisfaction. By providing personalized recommendations aligned with users' emotions, streaming platforms can improve user retention, increase content consumption, and offer a unique movie-watching experience.
- **Movie Theaters:** Movie theaters can leverage the emotion-based recommendation system to attract audiences and improve their movie selection. By understanding the emotional preferences of their target audience, theaters can curate movie schedules that align with popular emotional themes, enhancing the overall cinematic experience for moviegoers.
- **Content Providers and Marketers:** Content providers and marketers can utilize the emotion-based movie recommendation system to optimize their offerings and targeted marketing campaigns. By understanding the emotional preferences of their audience, they can create tailored content and promotional strategies that resonate with users on an emotional level, leading to increased engagement and customer satisfaction.
- **Research and Academia:** The gathered movie data and emotional analysis can be valuable for researchers and academics studying the relationship between emotions and movie preferences. Researchers can analyze emotional trends in movies over time, identify patterns in emotional content, and gain insights into the impact of emotions on viewer engagement and cinematic experiences.

9 CONCLUSION

- In conclusion, the project "Movie Recommendation Based On Emotion Using Web Scraping With IBM" has successfully developed a movie recommendation system that suggests movies to users based on their emotions. By leveraging web scraping techniques, sentiment analysis, and emotion detection algorithms, the system provides personalized and emotionally resonant movie recommendations.
- The application of this system offers several benefits, including enhanced user engagement, personalized movie recommendations, discovery of new movies, mood regulation, targeted marketing and content curation, and research opportunities. By incorporating emotions into the movie recommendation process, users can have a more immersive and satisfying movie-watching experience.
- However, it is important to address certain challenges, such as the accuracy of the emotion recognition model, the limited emotional range captured in the dataset, and the subjectivity of emotions. Further refinement and improvement in the emotion recognition model, regular updates to the web scraping process, and considering individual variations in emotional interpretation can help overcome these challenges.

10 FUTURE SCOPE

- The project "Movie Recommendation Based On Emotion Using Web Scraping With IBM" has a promising future with several avenues for further development and enhancements. Here are some potential future scopes for the project:
- **Enhanced Emotion Recognition:** Improving the accuracy and robustness of the emotion recognition model can be a major focus for future development. Exploring advanced deep learning techniques, incorporating larger and more diverse emotion

datasets, and fine-tuning the model can lead to better emotion detection and classification, resulting in more accurate movie recommendations.

- **Real-time Emotion Detection:** Currently, the project utilizes pre-recorded data for emotion analysis. Future enhancements could involve integrating real-time emotion detection using techniques like facial recognition from live video streams. This would enable the system to recommend movies based on the user's real-time emotional state, providing a more dynamic and personalized movie recommendation experience.
- **User Feedback and Ratings:** Incorporating user feedback and ratings can enhance the recommendation algorithm. Collecting user feedback on recommended movies and integrating it into the recommendation process can improve the system's accuracy and adaptability. Additionally, incorporating collaborative filtering techniques based on user ratings can provide more accurate and diverse movie recommendations.
- **Social Media Integration:** Integrating social media platforms can offer valuable insights into user emotions and movie preferences. Analyzing user sentiment and interactions related to movies on social media can provide real-time feedback and generate additional emotional features for recommendation. This integration can further personalize the recommendations based on users' social media activities and interactions.
- **Expanding to Other Languages and Cultures:** The current project primarily focuses on English language movies. Expanding the system to support multiple languages and cultural contexts can make the recommendation system more inclusive and diverse. This would involve incorporating multilingual sentiment analysis and emotion detection techniques and gathering movie data from different regions and cultures.
- **Integration with Smart Devices and Voice Assistants:** Integrating the emotion-based movie recommendation system with smart devices and voice assistants can provide a seamless and interactive movie-watching experience. Users can receive movie recommendations through voice commands and interact with the system using natural language processing. This would further enhance the convenience and accessibility of the recommendation system.
- **Collaboration with Industry Partners:** Collaborating with movie studios, streaming platforms, and content providers can offer opportunities for industry partnerships. By partnering with these entities, the recommendation system can access a wider range of movie data, receive real-time updates on movie releases, and gain insights into user preferences, leading to more accurate and up-to-date recommendations.
- **In conclusion,** the future scope for the project involves improving emotion recognition, real-time detection, incorporating user feedback, social media integration, language and cultural expansion, smart device integration, and collaboration with industry partners. These enhancements can further personalize and enrich the movie recommendation experience, providing users with tailored and engaging content.

11 BIBILOGRAPHY

GitHub repositories related to movie recommendation systems and emotion recognition that you can reference:

Movie Recommendation Systems:

GitHub Repository: <https://github.com/rounakbanik/movie-recommender-systems>

Description: This repository contains various implementations and examples of movie recommendation systems using different algorithms and techniques.

Emotion Recognition:

GitHub Repository: <https://github.com/ieee8023/DeepSad>

Description: This repository provides code for a deep learning model that performs emotion recognition from facial expressions, specifically focusing on recognizing sadness.

Web Scraping with Beautiful Soup:

GitHub Repository: <https://github.com/twtrubiks/python-web-scraping-tutorial>

Description: This repository provides tutorials and examples of web scraping using Python and the BeautifulSoup library, which can be helpful for gathering movie data from online sources.

Flask Web Development:

GitHubRepository:

https://github.com/CoreyMSchafer/code_snippets/tree/master/Python/Flask_Blog

Description: This repository contains a Flask blog application that can serve as a reference for developing web applications using Flask, which can be useful for building the frontend of your movie recommendation system.

References:

- 1.Han, X., Li, C., & Li, F. (2020). DeepFace: Deep Learning Face Recognition Technology. In Proceedings of the 4th International Conference on Computer Science and Application Engineering (pp. 9-13). ACM.
- 2.Pimentel, M. A. F., Clifton, D. A., Clifton, L., & Tarassenko, L. (2014). A review of novelty detection. Signal Processing, 99, 215-249.
- 3.Kotsiantis, S. B., Zaharakis, I. D., & Pintelas, P. E. (2007). Supervised machine learning: A review of classification techniques. Informatica, 31(3), 249-268.
- 4.Agrawal, R., & Srikant, R. (1994). Fast algorithms for mining association rules. In Proceedings of the 20th International Conference on Very Large Data Bases (pp. 487-499).
- 5.Beigi, G., Chen, Y., & Li, S. Z. (2017). Emotion recognition in the wild with deep facial expression representation and facial landmarks. Pattern Recognition Letters, 100, 72-80.
- 6.Berners-Lee, T., Hendler, J., & Lassila, O. (2001). The semantic web. Scientific American, 284(5), 34-43.

APPENDIX

A. Source Code

FLASK FILE→app.py:

```
from flask import Flask, render_template, request
import cv2
from deepface import DeepFace
import time
import requests
from bs4 import BeautifulSoup
import regex as re

app = Flask(__name__)

face_cascade = cv2.CascadeClassifier("haarcascade_frontalface_default.xml")
video_duration = 5 # in seconds

@app.route('/')
def index():
    return render_template('index.html')

@app.route('/analyze_emotion', methods=['POST'])
def analyze_emotion():
    emotions = []

    cap = cv2.VideoCapture(0)
    start_time = time.time()

    while True:
        ret, frame = cap.read()
        gray = cv2.cvtColor(frame, cv2.COLOR_BGR2GRAY)
        faces = face_cascade.detectMultiScale(gray, 1.1, 4)

        for (x, y, w, h) in faces:
            cv2.rectangle(frame, (x, y), (x + w, y + h), (255, 0, 0), 3)
            face_img = frame[y:y+h, x:x+w]

            _, img_encoded = cv2.imencode('.jpg', face_img)
            img_bytes = img_encoded.tobytes()
            img_path = 'temp.jpg'
            with open(img_path, 'wb') as f:
                f.write(img_bytes)

            result = DeepFace.analyze(img_path=img_path, actions=['emotion'],
enforce_detection=False)

            dominant_emotion = (result[0]["dominant_emotion"][:])
            emotions.append(dominant_emotion)

            txt = "Emotion: " + dominant_emotion

            cv2.putText(frame, txt, (x, y-10),
                        cv2.FONT_HERSHEY_SIMPLEX, 0.9, (0, 0, 255), 2)

            cv2.imshow('frame', frame)

            if time.time() - start_time >= video_duration:
```

```

        break

    if cv2.waitKey(1) & 0xff == ord('q'):
        break

cap.release()
cv2.destroyAllWindows()

most_common_emotion = max(set(emotions), key=emotions.count)

movie_data = fetch_movies_from_imdb(most_common_emotion)

return render_template('result.html', emotion=most_common_emotion, movies=movie_data)

def fetch_movies_from_imdb(emotion):
    genre_mapping = {
        'sad': 'drama',
        'disgust': 'musical',
        'angry': 'family',
        'neutral': 'thriller',
        'fear': 'sport',
        'happy': 'thriller',
        'surprised': 'film_noir'
    }

    genre = genre_mapping.get(emotion)

    if genre:
        url
        =
f'http://www.imdb.com/search/title?genres={genre}&title_type=feature&sort=moviemeter, asc'
        response = requests.get(url)
        soup = BeautifulSoup(response.text, 'html.parser')
        movie_titles = soup.find_all('h3', class_='lister-item-header')
        movie_ratings = soup.find_all('div', class_='inline-block ratings-imdb-rating')
        movie_image = soup.find_all("img", {"class": "loadlate"})

        movie_data = []
        # for title, rating, image, in zip(movie_titles, movie_ratings, movie_image):
        #
        #         movie_data.append({'title': title.text.strip(), 'rating':
rating.text.strip(), 'image': image.text.strip()})

        for i in range(0, 10):
            title = movie_titles[i].text
            rating = movie_ratings[i].text
            images = movie_image[i]['loadlate']
            names=movie_image[i]['loadlate']
            movie_data.append({'title': title, 'rating': rating, 'image_url': images})
        return movie_data
        # return movie_data

    return []

if __name__ == '__main__':
    app.run()

```

Index.html:

```
<!-- <!DOCTYPE html>
<html>
<head>
  <title>Emotion Detection</title>
</head>
<body>
  <h1>Emotion Detection</h1>
  <form action="/analyze_emotion" method="POST">
    <button type="submit">Analyze Emotion</button>
  </form>
</body>
</html> -->
<!DOCTYPE html>
<html>
<head>
  <title>Emotion Detection</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #f5f5f5;
      margin: 0;
      padding: 20px;
    }

    h1 {
      color: #333;
      text-align: center;
    }

    form {
      display: flex;
      justify-content: center;
      margin-top: 30px;
    }

    button {
      padding: 10px 20px;
      font-size: 16px;
      background-color: #4CAF50;
      color: white;
      border: none;
      border-radius: 4px;
      cursor: pointer;
    }

    button:hover {
      background-color: #45a049;
    }
  </style>
</head>
<body>
  <h1>Emotion Detection</h1>
  <form action="/analyze_emotion" method="POST">
    <button type="submit">Analyze Emotion</button>
  </form>
</body>
```



```
</html>
```

Result.html:

```
<!DOCTYPE html>
<html>
<head>
  <title>Emotion Detection Result</title>
  <style>
    body {
      font-family: Arial, sans-serif;
      background-color: #000;
      color: #fff;
      margin: 0;
      padding: 0;
    }

    .container {
      max-width: 800px;
      margin: 0 auto;
      padding: 20px;
      background-color: #1f1f1f;
      box-shadow: 0 2px 5px rgba(0, 0, 0, 0.5);
      border-radius: 5px;
    }

    h1, h2, h3 {
      text-align: center;
    }

    h1 {
      color: #ff3f3f;
    }

    h2 {
      color: #ff7070;
    }

    h3 {
      color: #fff;
    }

    ul {
      list-style-type: none;
      padding: 0;
      margin-top: 20px;
      display: flex;
      flex-wrap: wrap;
      justify-content: center;
    }

    li {
      margin: 10px;
      width: 200px;
    }

    .movie-image {
```

```
        width: 100%;
        height: auto;
        border-radius: 5px;
    }
</style>
</head>
<body>
    <div class="container">
        <h1>Emotion Detection Result</h1>
        <h2>Dominant Emotion: {{ emotion }}</h2>
        <h3>Recommended Movies:</h3>
        <ul>
            {% for movie in movies %}
            <li>
                
                <p>{{ movie.title }} (Rating: {{ movie.rating }})</p>
            </li>
            {% endfor %}
        </ul>
    </div>
</body>
</html>
```