13. Source code

```
# Movie Recommendation: Simple Content-Based Filtering


from sklearn.feature_extraction.text import TfidfVectorizer

from sklearn.metrics.pairwise import cosine_similarity


# Sample movie data

movies = {

  'The Matrix': 'A hacker discovers reality is a simulation and joins a rebellion.',

  'Inception': 'A thief enters people's dreams to steal secrets and plant ideas.',

  'Interstellar': 'Astronauts travel through a wormhole to find a new home for humanity.',

  'The Dark Knight': 'Batman battles the Joker to save Gotham City.',

  'Shutter Island': 'A detective investigates a psychiatric facility on a mysterious island.'

}


# Prepare data

titles = list(movies.keys())

descriptions = list(movies.values())


# Convert text to TF-IDF vectors
```

```
tfidf = TfidfVectorizer(stop_words='english')
tfidf_matrix = tfidf.fit_transform(descriptions)

# Compute cosine similarity matrix
similarity = cosine_similarity(tfidf_matrix)

# Recommend similar movies
def recommend(movie_title):
    if movie_title not in titles:
        return "Movie not found."
    idx = titles.index(movie_title)
    scores = list(enumerate(similarity[idx]))
    scores = sorted(scores, key=lambda x: x[1], reverse=True)
    recommendations = [titles[i] for i, _ in scores[1:4]]
    return recommendations

# Example usage
movie = 'Inception'
print(f"Movies similar to '{movie}':")
print(recommend(movie))
```

Output ;

Movies similar to 'Inception':