jupyter TF-IDF Last Checkpoint: 1 minute ago

File Edit View Run Kernel Settings Help

Code JupyterLab Python

[1]:
```
import numpy as np
import pandas as pd
import math

# Step 1: Define Movie Descriptions (Manually Preprocessed)
movies = {
    "The Matrix": ["hacker", "discover", "truth", "real", "fight", "AI", "machine"],
    "Inception": ["thief", "steal", "corporate", "secret", "dream", "undertake", "final", "mission"]
}

# Step 2: Create a Unique Vocabulary
unique_terms = list(set([word for words in movies.values() for word in words]))

# Step 3: Compute Term Frequency (TF)
tf_matrix = {movie: [words.count(term) for term in unique_terms] for movie, words in movies.items()}

# Step 4: Compute Inverse Document Frequency (IDF)
N = len(movies)  # Total number of movies
idf_values = []
for term in unique_terms:
    df = sum([1 for words in movies.values() if term in words])  # Count documents containing the term
    idf = math.log(N / df) if df > 0 else 0  # Avoid division by zero
    idf_values.append(idf)

# Step 5: Compute TF-IDF Matrix Manually
tfidf_matrix = {movie: [tf * idf for tf, idf in zip(tf_values, idf_values)] for movie, tf_values in tf_matrix.items()}

# Convert TF-IDF results into a DataFrame for better readability
df_tfidf = pd.DataFrame(tfidf_matrix, index=unique_terms)

# Step 6: Display the TF-IDF scores
print("Manual TF-IDF Scores:")
print(df_tfidf)
```

```
Manual TF-IDF Scores:
           The Matrix  Inception
real         0.693147   0.000000
AI           0.693147   0.000000
hacker       0.693147   0.000000
final        0.000000   0.693147
corporate    0.000000   0.693147
machine      0.693147   0.000000
mission      0.000000   0.693147
truth        0.693147   0.000000
```