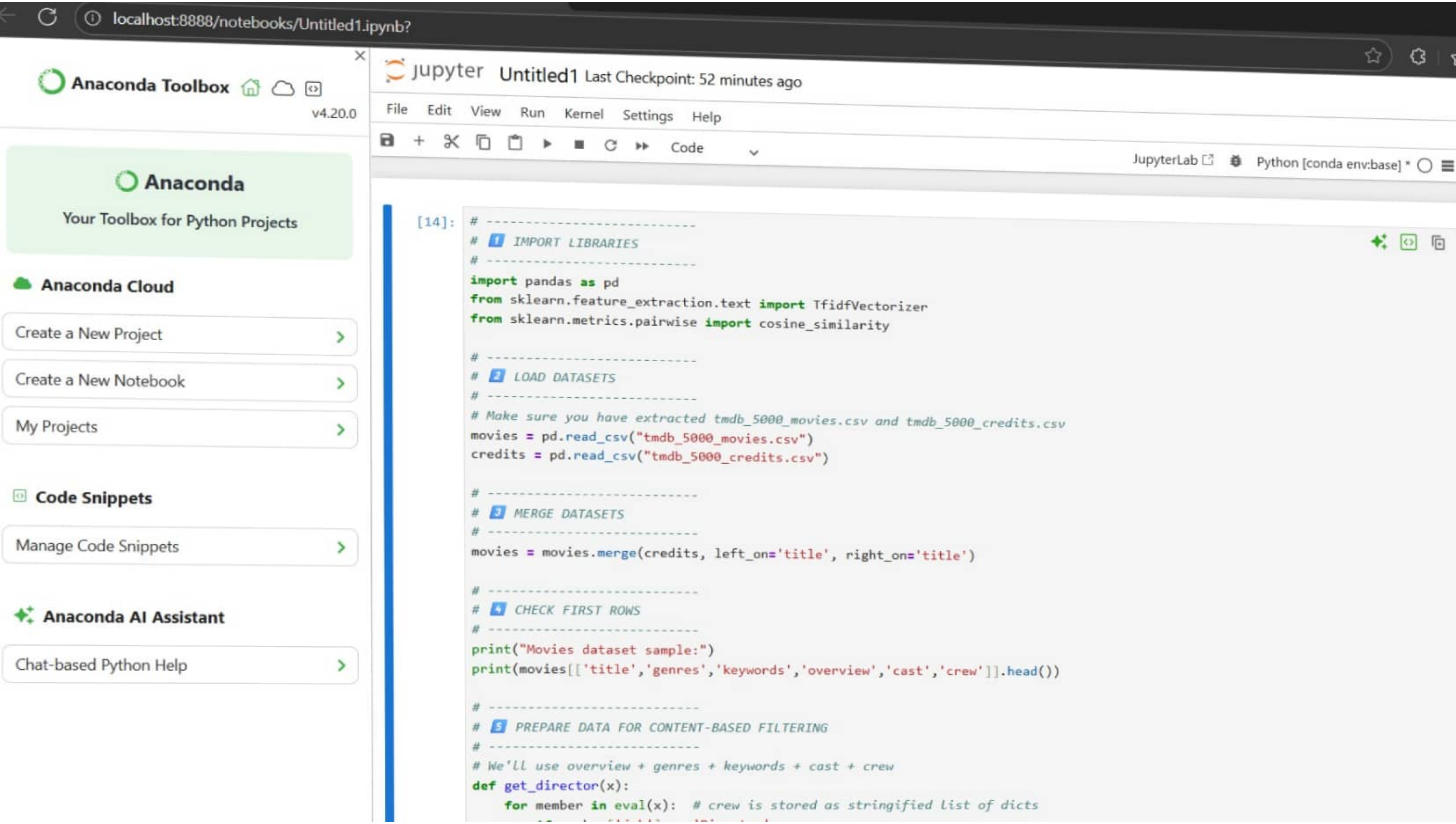
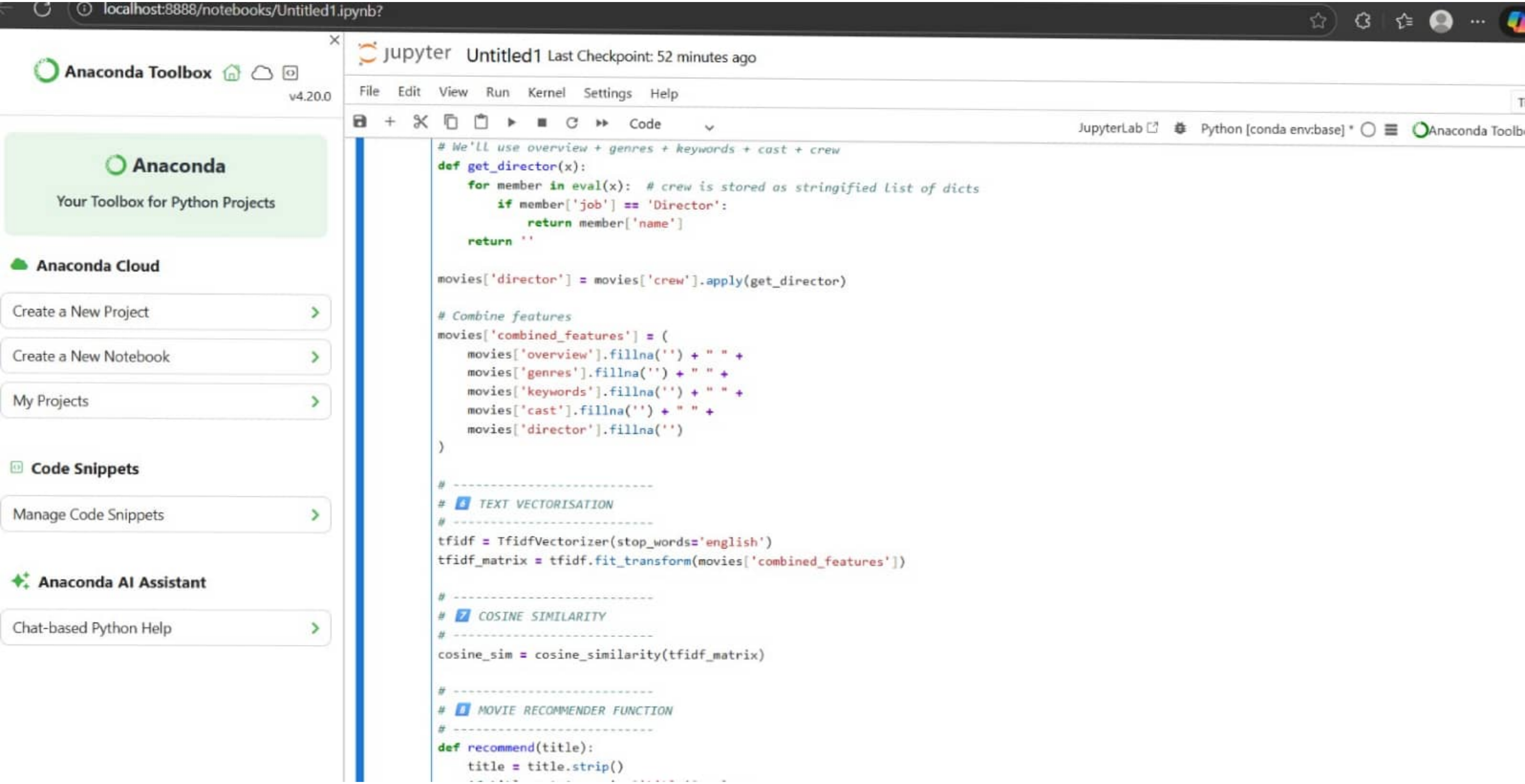
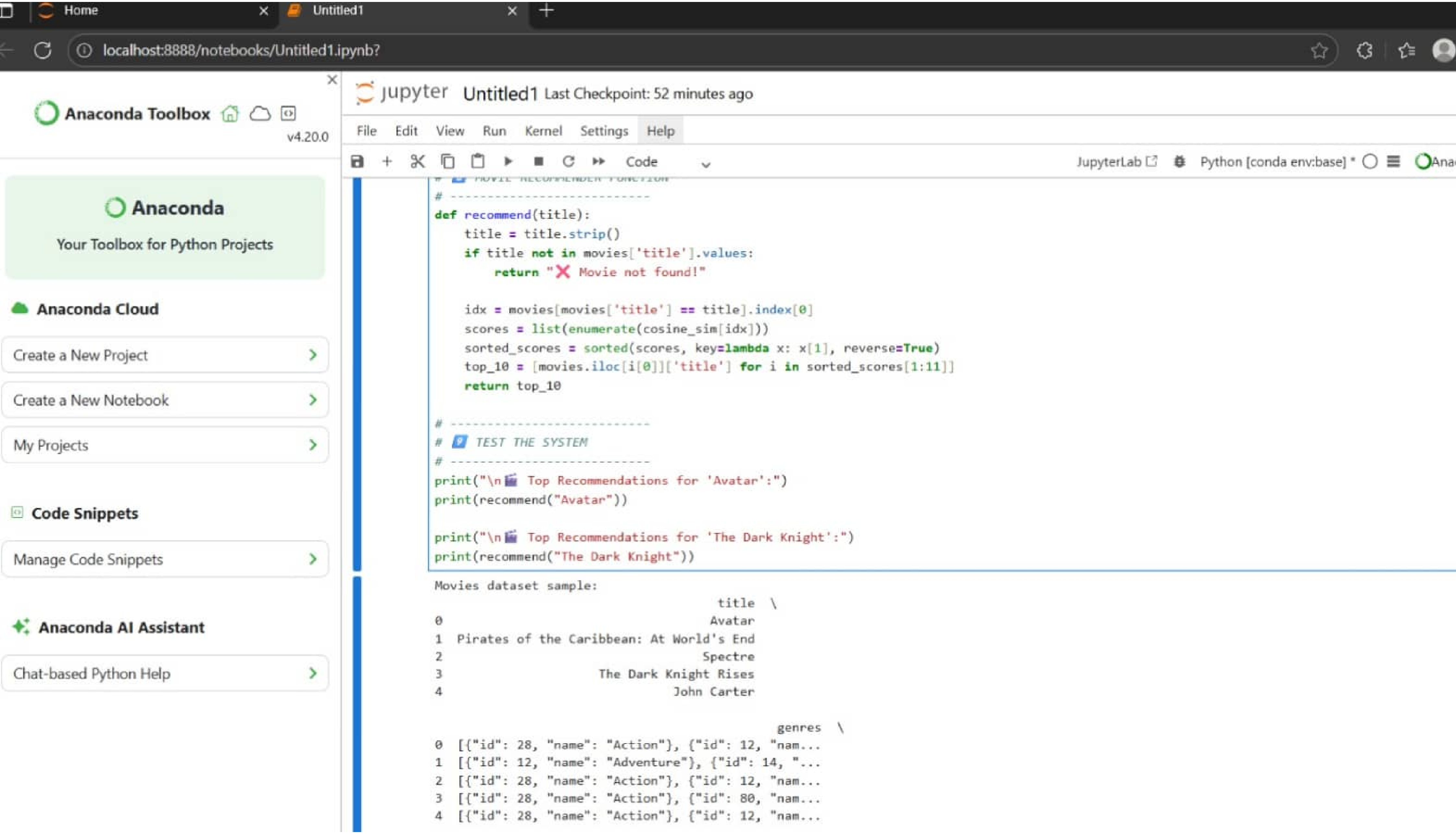


```
# ---------------------------
# 1️⃣ IMPORT LIBRARIES
# ---------------------------
import pandas as pd
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.metrics.pairwise import cosine_similarity

# ---------------------------
# 2️⃣ LOAD DATASETS
# ---------------------------
# Make sure you have extracted tmdb_5000_movies.csv and tmdb_5000_credits.csv
movies = pd.read_csv("tmdb_5000_movies.csv")
credits = pd.read_csv("tmdb_5000_credits.csv")

# ---------------------------
# 3️⃣ MERGE DATASETS
# ---------------------------
movies = movies.merge(credits, left_on='title', right_on='title')

# ---------------------------
# 4️⃣ CHECK FIRST ROWS
# ---------------------------
print("Movies dataset sample:")
print(movies[['title','genres','keywords','overview','cast','crew']].head())

# ---------------------------
# 5️⃣ PREPARE DATA FOR CONTENT-BASED FILTERING
# ---------------------------
# We'll use overview + genres + keywords + cast + crew
def get_director(x):
    for member in eval(x):  # crew is stored as stringified list of dicts
```

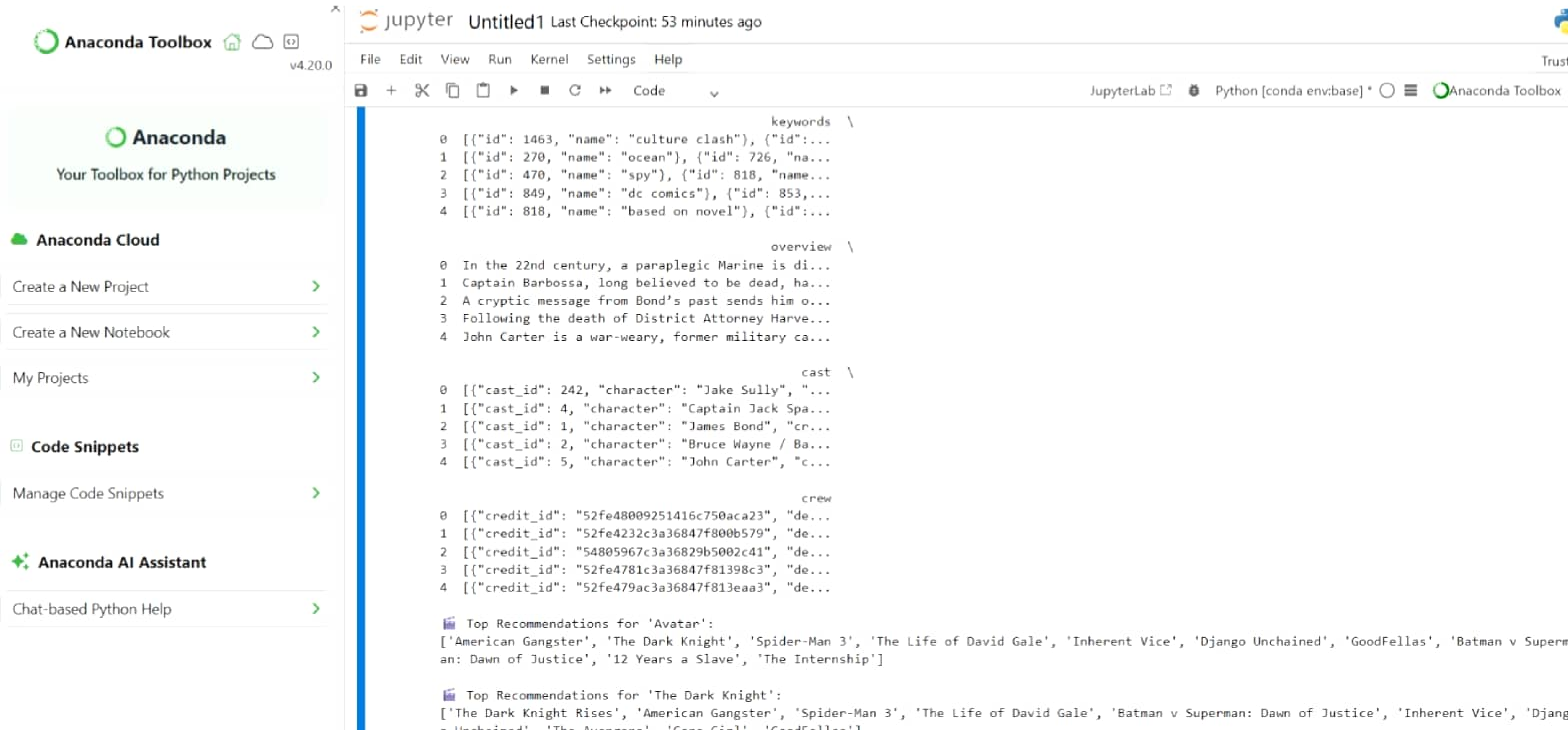


```python
# We'll use overview + genres + keywords + cast + crew
def get_director(x):
    for member in eval(x):  # crew is stored as stringified list of dicts
        if member['job'] == 'Director':
            return member['name']
    return ''

movies['director'] = movies['crew'].apply(get_director)

# Combine features
movies['combined_features'] = (
    movies['overview'].fillna('') + " " +
    movies['genres'].fillna('') + " " +
    movies['keywords'].fillna('') + " " +
    movies['cast'].fillna('') + " " +
    movies['director'].fillna('')
)

# --------------------------
# 6️⃣ TEXT VECTORISATION
# --------------------------
tfidf = TfidfVectorizer(stop_words='english')
tfidf_matrix = tfidf.fit_transform(movies['combined_features'])

# --------------------------
# 7️⃣ COSINE SIMILARITY
# --------------------------
cosine_sim = cosine_similarity(tfidf_matrix)

# --------------------------
# 8️⃣ MOVIE RECOMMENDER FUNCTION
# --------------------------
def recommend(title):
    title = title.strip()
```



```
# MOVIE RECOMMENDER FUNCTION
# ---------------------------
def recommend(title):
    title = title.strip()
    if title not in movies['title'].values:
        return "❌ Movie not found!"

    idx = movies[movies['title'] == title].index[0]
    scores = list(enumerate(cosine_sim[idx]))
    sorted_scores = sorted(scores, key=lambda x: x[1], reverse=True)
    top_10 = [movies.iloc[i[0]]['title'] for i in sorted_scores[1:11]]
    return top_10

# ---------------------------
# 9️⃣ TEST THE SYSTEM
# ---------------------------
print("\n🎬 Top Recommendations for 'Avatar':")
print(recommend("Avatar"))

print("\n🎬 Top Recommendations for 'The Dark Knight':")
print(recommend("The Dark Knight"))
```

```
Movies dataset sample:
                                      title  \
0                                    Avatar
1  Pirates of the Caribbean: At World's End
2                                   Spectre
3                     The Dark Knight Rises
4                               John Carter

                                              genres  \
0  [{"id": 28, "name": "Action"}, {"id": 12, "nam...
1  [{"id": 12, "name": "Adventure"}, {"id": 14, "...
2  [{"id": 28, "name": "Action"}, {"id": 12, "nam...
3  [{"id": 28, "name": "Action"}, {"id": 80, "nam...
4  [{"id": 28, "name": "Action"}, {"id": 12, "nam...
```



```
                                            keywords  \
0  [{"id": 1463, "name": "culture clash"}, {"id":...
1  [{"id": 270, "name": "ocean"}, {"id": 726, "na...
2  [{"id": 470, "name": "spy"}, {"id": 818, "name...
3  [{"id": 849, "name": "dc comics"}, {"id": 853,...
4  [{"id": 818, "name": "based on novel"}, {"id":...

                                            overview  \
0  In the 22nd century, a paraplegic Marine is di...
1  Captain Barbossa, long believed to be dead, ha...
2  A cryptic message from Bond's past sends him o...
3  Following the death of District Attorney Harve...
4  John Carter is a war-weary, former military ca...

                                                cast  \
0  [{"cast_id": 242, "character": "Jake Sully", "...
1  [{"cast_id": 4, "character": "Captain Jack Spa...
2  [{"cast_id": 1, "character": "James Bond", "cr...
3  [{"cast_id": 2, "character": "Bruce Wayne / Ba...
4  [{"cast_id": 5, "character": "John Carter", "c...

                                                crew
0  [{"credit_id": "52fe48009251416c750aca23", "de...
1  [{"credit_id": "52fe4232c3a36847f800b579", "de...
2  [{"credit_id": "54805967c3a36829b5002c41", "de...
3  [{"credit_id": "52fe4781c3a36847f81398c3", "de...
4  [{"credit_id": "52fe479ac3a36847f813eaa3", "de...

🎬 Top Recommendations for 'Avatar':
['American Gangster', 'The Dark Knight', 'Spider-Man 3', 'The Life of David Gale', 'Inherent Vice', 'Django Unchained', 'GoodFellas', 'Batman v Superm
an: Dawn of Justice', '12 Years a Slave', 'The Internship']

🎬 Top Recommendations for 'The Dark Knight':
['The Dark Knight Rises', 'American Gangster', 'Spider-Man 3', 'The Life of David Gale', 'Batman v Superman: Dawn of Justice', 'Inherent Vice', 'Djang
o Unchained', 'The Avengers', 'Gone Girl', 'GoodFellas']
```

Scanned with OKEN Scanner