Cross-Platform Data Harmony: Machine Learning for Movie and TV ID Validation, Mapping, and Price Prediction

Rohan Shah
202218002@daiict.ac.in
MSc DS, DA-IICT
Ahmedabad, India

Internship Details:
Biztic Technologies
Ahmedabad, Gujarat
Biztic

Duration: 15 weeks
Start Date: 01st Jan 2024.
End Date: 13th April 2024

Abstract

This project describes a research that aims to improve cross-platform data harmonization using machine learning for movie and TV ID validation, mapping, and pricing prediction. The project used proprietary code for verification, validation, and mapping, as well as machine learning methods, to simplify operations and enhance accuracy. Key components include code checking and mapping, as well as machine learning approaches for price prediction. The project's accomplishments are highlighted as efficiency benefits and accuracy improvements in data processing and prediction activities.

Keywords: Machine Learning, Verification, Validation, Mapping, Delta(Days) Prediction, SQL(Structured Query Language)

1 Introduction (problem statement)

The issue statement expresses the requirement to smoothly integrate and reconcile data from numerous media channels, assure proper identification and mapping of movies and TV series across these platforms, and forecast market pricing using machine learning techniques. This includes dealing with the difficulties of divergent data formats, various identification systems, and dynamic content values in the digital media world. The goal of employing machine learning algorithms is to automate these procedures, enhance accuracy, and give actionable insights for content producers, distributors, and platform owners to optimize content management, distribution strategies, and income generation.

2 Block Diagram (Project Structure and Flow)

Step 1 - Identify Data Sources: Choose the websites or online platforms from which you wish to gather information. This might include e-commerce websites, news sites, social media platforms, or any other sources that are pertinent to your research.

Step 2 - Choose a web scraping tool: Select an appropriate web scraping tool or library. Popular choices include BeautifulSoup (for Python), Scrapy, Selenium, and Octoparse. These tools allow you to extract data from HTML or JSON formats on web sites.

Step 3 - Understand Website Structure: Examine the structure of the website(s) you plan to scrape. Determine which HTML elements, classes, and attributes contain the data you want. This will help you develop the best scraping approach.

Step 4 - Create Scraping Scripts: Write scripts to extract desired data from the recognized web pages. Navigate through the website using the web scraping tool or library of your choice, discover important items, and extract the required information. Ensure that the website's terms of service and legal requirements are followed.

Step 5 - Data Preprocessing: Scraped data should be cleaned and preprocessed to remove noise, handle missing values, and formatted as a structured dataset, providing good data quality for further analysis. This might involve text normalization, encoding conversion, unusual character handling, and outlier identification and removal procedures.

Step 6 - Data Analysis: Use exploratory data analysis (EDA) on the preprocessed dataset to get insights and analyze trends. This might include statistical summaries, data visualization with tools such as Matplotlib or Seaborn, and correlation analysis to determine variables' correlations.

Step 7 - ML Models: A variety of machine learning models are used to acquire insights and better comprehend the information. These models assist in identifying patterns, correlations, and trends in the data. The purpose of using techniques including as classification, clustering, and dimensionality reduction is to extract useful information that may be used to inform decision-making.

Step 8 - Regression: Regression analysis is used to predict numerical values from input variables. In your example, a regression model is used to anticipate movie ticket prices and the number of days between a film's theatrical premiere and availability on OTT platforms. Regression algorithms allow these crucial parameters to be estimated by examining past data and taking into account aspects such as release date, genre, and market trends.

Step 9 - Recommendation model: Users are recommended TV series depending on their viewing history and preferences. The system uses techniques such as collaborative filtering or content-based filtering to identify similarities between persons or products (in this example, TV series) and create tailored suggestions. The cosine similarity metric is frequently used to assess the similarity of feature vectors, which aids in the selection of relevant information for each user across several platforms.

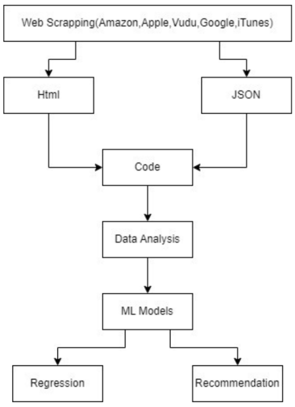


Figure 1: Flow Diagram Of Project

3 Tools and Technologies

A. Visual Studio Code (VS Code)

The major tool for web scraping is Visual Studio Code (VS Code), which employs Python code written using the BeautifulSoup module. This supports data extraction from a variety of platforms, including Amazon Prime Video, Vudu, Google Play Store, and the Apple TV app.

B. HeidiSQL

HeidiSQL is also used to connect to databases, which makes data administration and storage more efficient. Given Warner Bros. as a prominent customer, these solutions facilitate the extraction and structuring of pertinent data.

C. Microsoft Excel 365

Microsoft Excel 365 is used to convert code outputs to.xlsx format, which ensures compatibility and simplicity of analysis. Overall, this integrated set of tools and technologies enables seamless data retrieval, management, and analysis, resulting in effective decision-making and insight development.

4 Use Cases

Step 1 - Verification:

The major use case for this project is the full extraction and validation of TV show data from multiple platforms via web scraping techniques performed with BeautifulSoup. Once the data has been recovered, the following step is to thoroughly verify it for correctness. This verification step involves cross-referencing the retrieved data with information held in databases and Excel files to discover inconsistencies.

Step 2 - Validation:

To ease this validation process, validation code is created, which allows other team members to easily identify and correct inconsistencies using a deployed internet interface. In addition, functions such as TV program merging and unmerging are provided, with checkboxes and filters used to facilitate the mapping process.

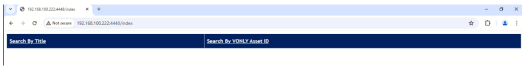
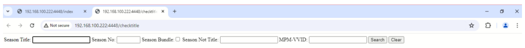
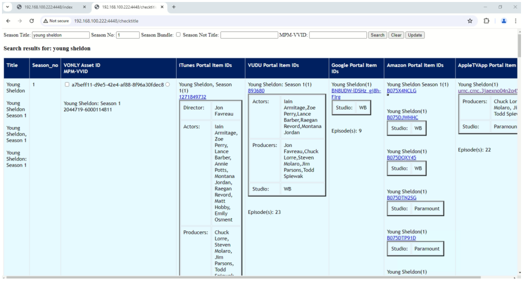


Figure 2: Validation While Opening Through API


192.168.100.222:4448/index
192.168.100.222:4448/checktitl
Not secure
192.168.100.222:4448/checktitle
Search
Clear

Season Title: Season No: Season Bundle: □ Season Not Title: MPM-VVID:

Figure 3: Searching By title Representation

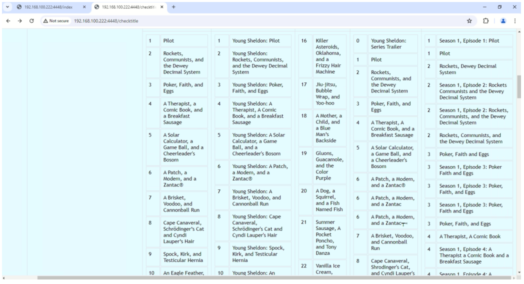

192.168.100.222:4448/index
192.168.100.222:4448/checktitl
Not secure
192.168.100.222:4448/checktitle

Season Title: young sheldon Season No: 1 Season Bundle: ☐ Season Not Title: MPM-VVID: Search Clear Update

Search results for: young sheldon

Title	Season_no	VONLY Asset ID MPM-VVID	iTunes Portal Item IDs	VUDU Portal Item IDs	Google Portal Item IDs	Amazon Portal Item IDs	AppleTVApp Portal Item
Young Sheldon Young Sheldon Season 1 Young Sheldon, Season 1 Young Sheldon: Season 1	1	☐ a7beff11-d9e5-42e4-af88-8f96a30fdec8 ○ Young Sheldon: Season 1 2044719-6000114811	Young Sheldon, Season 1(1) 1271849732 Director: Jon Favreau Actors: Iain Armitage, Zoe Perry, Lance Barber, Annie Potts, Montana Jordan, Raegan Revord, Matt Hobby, Emily Osment Producers: Chuck Lorre, Steven Molaro, Jim Parsons, Todd	Young Sheldon: Season 1(1) 893680 Actors: Iain Armitage,Zoe Perry,Lance Barber,Raegan Revord,Montana Jordan Producers: Jon Favreau,Chuck Lorre,Steven Molaro,Jim Parsons,Todd Spiewak Studio: WB Episode(s): 23	Young Sheldon(1) BN8UDW-lDSHz_gj8h-FIrg Studio: WB Episode(s): 9	Young Sheldon Season 1(1) B075X4NCLG Young Sheldon(1) B075DJWHHC Studio: WB Young Sheldon(1) B075DQXY45 Studio: WB Young Sheldon(1) B075DTN2SG Studio: Paramount Young Sheldon(1) B075DTP91D Studio: Paramount Young Sheldon(1)	Young Sheldon(1) umc.cmc.3jaexno04n2q4 Producers: Chuck Lor Molaro, Ji Todd Spie Studio: Paramoun Episode(s): 22

Figure 4: Young Sheldon Season-1.1


192.168.100.222:4448/index
192.168.100.222:4448/checktitl
Not secure
192.168.100.222:4448/checktitle

1	Pilot
2	Rockets, Communists, and the Dewey Decimal System
3	Poker, Faith, and Eggs
4	A Therapist, a Comic Book, and a Breakfast Sausage
5	A Solar Calculator, a Game Ball, and a Cheerleader's Bosom
6	A Patch, a Modem, and a Zantac®
7	A Brisket, Voodoo, and Cannonball Run
8	Cape Canaveral, Schrödinger's Cat and Cyndi Lauper's Hair
9	Spock, Kirk, and Testicular Hernia
10	An Eagle Feather,

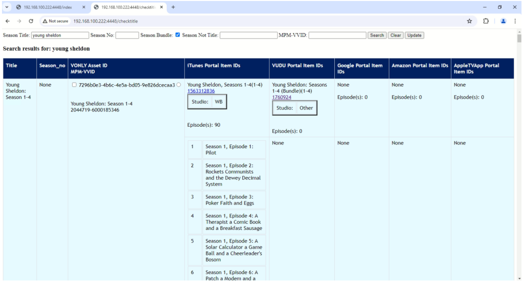
1	Young Sheldon: Pilot
2	Young Sheldon: Rockets, Communists, and the Dewey Decimal System
3	Young Sheldon: Poker, Faith, and Eggs
4	Young Sheldon: A Therapist, A Comic Book, and a Breakfast Sausage
5	Young Sheldon: A Solar Calculator, a Game Ball, and a Cheerleader's Bosom
6	Young Sheldon: A Patch, a Modem, and a Zantac®
7	Young Sheldon: A Brisket, Voodoo, and Cannonball Run
8	Young Sheldon: Cape Canaveral, Schrödinger's Cat and Cyndi Lauper's Hair
9	Young Sheldon: Spock, Kirk, and Testicular Hernia
10	Young Sheldon: An

16	Killer Asteroids, Oklahoma, and a Frizzy Hair Machine
17	Jiu-jitsu, Bubble Wrap, and Yoo-hoo
18	A Mother, a Child, and a Blue Man's Backside
19	Gluons, Guacamole, and the Color Purple
20	A Dog, a Squirrel, and a Fish Named Fish
21	Summer Sausage, A Pocket Poncho, and Tony Danza
22	Vanilla Ice Cream,

0	Young Sheldon: Series Trailer
1	Pilot
2	Rockets, Communists, and the Dewey Decimal System
3	Poker, Faith, and Eggs
4	A Therapist, A Comic Book, and a Breakfast Sausage
5	A Solar Calculator, a Game Ball, and a Cheerleader's Bosom
6	A Patch, a Modem, and a Zantac®
6	A Patch, a Modem, and a Zantac
6	A Patch, a Modem, and a Zantac┬«
7	A Brisket, Voodoo, and Cannonball Run
8	Cape Canaveral, Shrodinger's Cat, and Cyndi Lauper's

1	Season 1, Episode 1: Pilot
1	Pilot
2	Rockets, Dewey Decimal System
2	Season 1, Episode 2: Rockets Communists and the Dewey Decimal System
2	Season 1, Episode 2: Rockets, Communists, and the Dewey Decimal System
2	Rockets, Communists, and the Dewey Decimal System
3	Poker, Faith and Eggs
3	Season 1, Episode 3: Poker Faith and Eggs
3	Season 1, Episode 3: Poker, Faith, and Eggs
3	Season 1, Episode 3: Poker, Faith and Eggs
3	Poker, Faith, and Eggs
4	A Therapist, A Comic Book
4	Season 1, Episode 4: A Therapist a Comic Book and a Breakfast Sausage
4	Season 1, Episode 4: A

Figure 5: Young Sheldon Season-1.2


192.168.100.222:4448/index
192.168.100.222:4448/checktitl
Not secure
192.168.100.222:4448/checktitle

Season Title: young sheldon Season No: Season Bundle: ☑ Season Not Title: MPM-VVID: Search Clear Update

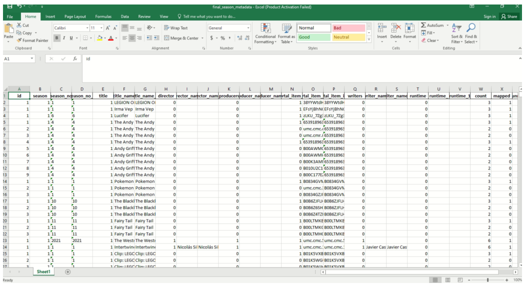
Search results for: young sheldon

Title	Season_no	VONLY Asset ID MPM-VVID	iTunes Portal Item IDs	VUDU Portal Item IDs	Google Portal Item IDs	Amazon Portal Item IDs	AppleTVApp Portal Item IDs
Young Sheldon: Season 1-4	None	☐ 7296b0e3-4b6c-4e5a-bd05-9e826dcecaa3 ○ Young Sheldon: Season 1-4 2044719-6000185346	Young Sheldon, Seasons 1-4(1-4) 1563312836 Studio: WB Episode(s): 90	Young Sheldon: Seasons 1-4 (Bundle)(1-4) 1760924 Studio: Other Episode(s): 0	None Episode(s): 0	None Episode(s): 0	None Episode(s): 0
			1 Season 1, Episode 1: Pilot 2 Season 1, Episode 2: Rockets Communists and the Dewey Decimal System 3 Season 1, Episode 3: Poker Faith and Eggs 4 Season 1, Episode 4: A Therapist a Comic Book and a Breakfast Sausage 5 Season 1, Episode 5: A Solar Calculator a Game Ball and a Cheerleader's Bosom 6 Season 1, Episode 6: A Patch a Modem and a	None	None	None	None

Figure 6: Young Sheldon Season-Bundle

Step 3 - Mapping:

Following that, mapping code is created to match TV program data given by Warner Bros. with platform-specific seasons, using a point system to track mapping progress and provide detailed results in Excel format. Throughout these operations, SQL databases aid data administration, guaranteeing effective processing of the project's large volume of data.

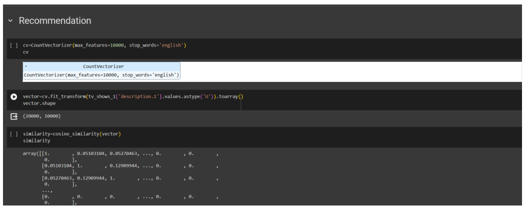


	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U	V	W	X
1	id	season	season_n	eason_no_	title	title_name	tle_name_	director	rector_nar	ector_nam	producers	ducer_na	ducer_nam	rtal_item	rtal_item_	tal_item_i	writers	riter_nam	iter_name	runtime	runtime_	runtime_1	count	mapped
2	1	1	1	1	1	LEGION OI	LEGION OI	0			0			1	38YYWtdH	38YYWtdH	0			0			3	1
3	1	1	1	1	1	Irma Vep	Irma Vep	0			0			1	EFsYjBhN(	EFsYjBhN(	0			0			3	1
4	1	1	6	6	1	Lucifer	Lucifer	0			0			1	zLKU_7Zg3	zLKU_7Zg3	0			0			3	1
5	1	1	4	4	1	The Andy	The Andy	0			0			1	653918963	653918963	0			0			3	1
6	2	1	4	4	1	The Andy	The Andy	0			0			0	umc.cmc.(	653918963	0			0			2	0
7	3	1	4	4	1	The Andy	The Andy	0			0			0	umc.cmr.l	653918963	0			0			2	0
8	4	1	4	4	1	The Andy	The Andy	0			0			1	653918963	653918963	0			0			3	1
9	5	1	4	4	1	Andy Grif	The Andy	0			0			0	B00AWMC	653918963	0			0			2	0
10	6	1	4	4	1	Andy Grif	The Andy	0			0			0	B00AWMI	653918963	0			0			2	0
11	7	1	4	4	1	Andy Grif	The Andy	0			0			0	B00K3AMI	653918963	0			0			2	0
12	8	1	4	4	1	Andy Grif	The Andy	0			0			0	B010U2C1	653918963	0			0			2	0
13	9	1	4	4	1	Andy Grif	The Andy	0			0			0	B00C177E	653918963	0			0			2	0
14	1	1	1	1	1	Pokemon	Pokemon	0			0			1	B0834GVN	B0834GVN	0			0			3	1
15	2	1	1	1	1	Pokemon	Pokemon	0			0			0	umc.cmc.:	B0834GVN	0			0			2	0
16	3	1	1	1	1	Pokemon	Pokemon	0			0			0	B0834GZJ	B0834GVN	0			0			2	0
17	1	1	10	10	1	The Black	The Black	0			0			1	B0B6ZJFLH	B0B6ZJFLH	0			0			3	1
18	2	1	10	10	1	The Black	The Black	0			0			0	B0B6Z6SH	B0B6ZJFLH	0			0			2	0
19	3	1	10	10	1	The Black	The Black	0			0			0	B0B6Z4TZ	B0B6ZJFLH	0			0			2	0
20	1	1	11	11	1	Fairy Tail	Fairy Tail	0			0			1	B00LTMKE	B00LTMKE	0			0			3	1
21	2	1	11	11	1	Fairy Tail	Fairy Tail	0			0			0	B00LTMKC	B00LTMKE	0			0			2	0
22	3	1	11	11	1	Fairy Tail	Fairy Tail	0			0			0	B00LTMHC	B00LTMKE	0			0			2	0
23	1	1	2021	2021	1	The West	The West	1			1			1	umc.cmc.!	umc.cmc.!	1			0			6	1
24	1	1	1	1	1	Intertwin	Intertwin	1	Nicolás Sil	Nicolás Sil	1			1	umc.cmc.:	umc.cmc.:	1	Javier Cas	Javier Cas	0			6	1
25	1	1	1	1	1	Clip: LEGC	Clip: LEGC	0			0			1	B01K5VXB	B01K5VXB	0			0			3	1
26	2	1	1	1	1	Clip: LEGC	Clip: LEGC	0			0			0	B01K5WG	B01K5VXB	0			0			2	0
27	3	1	1	1	1	Clip: LEGC	Clip: LEGC	0			0			0	B01K5WH	B01K5VXB	0			0			2	0

Figure 7: Season Mapping Results

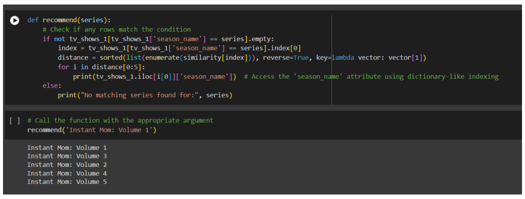
5 Results

After methodically gathering and categorizing the dataset over three months, regression models forecast the delta days before TV series are released on various OTT platforms such as Amazon Prime Video, Vudu, iTunes, and the Apple TV app. The study seeks to predict whether TV series will be distributed one or three months following their debut. In addition, a recommendation model is created utilizing season descriptions to offer related TV series based on user feedback. Due to project challenges, experts at DAI-ICT were consulted, and it was found that clustering methods were unsuitable for the data collection. Despite the limits of some methods, the internship gave great insights into data analysis approaches and the complexity of dealing with enormous datasets. Further conversations and investigations may be necessary to improve model accuracy and efficacy in future projects.



```
[ ] cv=CountVectorizer(max_features=10000, stop_words='english')
    cv

    CountVectorizer(max_features=10000, stop_words='english')

    vector=cv.fit_transform(tv_shows_1['description.1'].values.astype('U')).toarray()
    vector.shape

    (20000, 10000)

[ ] similarity=cosine_similarity(vector)
    similarity

    array([[1.        , 0.05103104, 0.05270463, ..., 0.        , 0.        ,
             0.        ],
           [0.05103104, 1.        , 0.12909944, ..., 0.        , 0.        ,
             0.        ],
           [0.05270463, 0.12909944, 1.        , ..., 0.        , 0.        ,
             0.        ],
           ...,
           [0.        , 0.        , 0.        , ..., 0.        , 0.        ,
             0.        ],
```

Figure 8: Recommendation code



```
def recommend(series):
    # Check if any rows match the condition
    if not tv_shows_1[tv_shows_1['season_name'] == series].empty:
        index = tv_shows_1[tv_shows_1['season_name'] == series].index[0]
        distance = sorted(list(enumerate(similarity[index])), reverse=True, key=lambda vector: vector[1])
        for i in distance[0:5]:
            print(tv_shows_1.iloc[i[0]]['season_name'])  # Access the 'season_name' attribute using dictionary-like indexing
    else:
        print("No matching series found for:", series)

[ ] # Call the function with the appropriate argument
    recommend('Instant Mom: Volume 1')

    Instant Mom: Volume 1
    Instant Mom: Volume 3
    Instant Mom: Volume 2
    Instant Mom: Volume 4
    Instant Mom: Volume 5
```

Figure 9: Recommendation Output

6 References

1. VS Code Retrived from
https://code.visualstudio.com/ Visual Studio

2. Web-Scrapping Retrieved from
https://beautiful-soup-4.readthedocs.io/en/latest/
Beautiful Soup

3. HeidiSQL. Retrieved from
https://www.heidisql.com SQL

4. Microsoft Excel. Retrieved from
https://www.microsoft.com/en-us/
microsoft-365/excel

5. SQL. Retrieved from
https://www.w3schools.com/sq1/

7 Github Link

Internship Project https://github.com/Rohan0412/Biztic
Biztic