Udemy Courses Dataset Analysis - Complete Documentation

Overview

This Jupyter Notebook analyzes a comprehensive dataset of Udemy courses across all subjects. The analysis uses the Pandas library to explore course characteristics, pricing, popularity, and temporal patterns. This documentation provides detailed explanations of each analysis step, code functionality, and expected outputs.

Dataset Description

The dataset (`Udemy_Dataset.csv`) contains information about Udemy courses including:

- Course titles and subjects
- Pricing information (free vs paid courses)
- Subscriber counts
- Course levels (Beginner, Intermediate, Advanced)
- Publication timestamps
- Course ratings and reviews

Setup and Data Loading

Import Required Libraries

python

```python
import pandas as pd
```

Purpose: Imports the Pandas library for data manipulation and analysis.

Load the Dataset

python

```python
data = pd.read_csv(r"Udemy_Dataset.csv")
```

Purpose: Loads the CSV file into a Pandas DataFrame for analysis. **Expected Output**: Creates a DataFrame object containing all course data.

Initial Data Exploration

python

```python
data.head()
```

Purpose: Displays the first 5 rows to understand the dataset structure. **Expected Output**: A table showing columns like course_title, subject, price, is_paid, num_subscribers, level, published_timestamp, etc.

Analysis Questions and Solutions

1. Course Subject Diversity

Question: What are all different subjects for which Udemy is offering courses?

```python
data.subject.unique()
```

Explanation:

- Uses the `.unique()` method on the 'subject' column
- Returns an array of all distinct subject categories
- Helps understand the breadth of Udemy's course offerings

Expected Output:

```
array(['Web Development', 'Business Finance', 'Musical Instruments',
       'Graphic Design', 'Python', 'JavaScript', 'Photography', etc.])
```

2. Most Popular Subject by Course Count

Question: Which subject has the maximum number of courses?

```python
data.subject.value_counts()
```

Explanation:

- `.value_counts()` counts occurrences of each unique value in the 'subject' column
- Results are sorted in descending order by default
- The first entry represents the subject with the most courses

Expected Output:

```
Web Development        1000
Business Finance        800
Graphic Design          600
Musical Instruments     400
...
```

3. Free Courses Analysis

Question: Show all the courses which are Free of Cost.

```python
data[data.is_paid == False]
```

Explanation:

- Creates a boolean mask where `is_paid` column equals `False`
- Filters the DataFrame to show only free courses
- Useful for identifying educational opportunities at no cost

Expected Output: A filtered DataFrame containing only courses where `is_paid = False`.

4. Paid Courses Analysis

Question: Show all the courses which are Paid.

```python
data[data.is_paid == True]
```

Explanation:

- Creates a boolean mask where `is_paid` column equals `True`
- Filters the DataFrame to show only paid courses
- Helps analyze the premium course market

Expected Output: A filtered DataFrame containing only courses where `is_paid = True`.

5. Top Selling Courses Identification

Question: Which are Top Selling Courses?

```python
data.sort_values('num_subscribers', ascending=False)
```

Explanation:

- Sorts the entire DataFrame by the 'num_subscribers' column
- `ascending=False` ensures highest subscriber counts appear first
- Identifies the most popular courses by enrollment

Expected Output: DataFrame sorted with courses having the highest subscriber counts at the top.

6. Least Popular Courses Identification

Question: Which are Least Selling Courses?

```python
data.sort_values('num_subscribers')
```

Explanation:

- Sorts by 'num_subscribers' in ascending order (default)
- Shows courses with the lowest enrollment numbers
- Useful for identifying underperforming courses

Expected Output: DataFrame sorted with courses having the lowest subscriber counts at the top.

7. Subject-Specific Price Filtering

Question: Show all courses of Graphic Design where the price is below 100?

```python
data[(data.subject == 'Graphic Design') & (data.price < '100')]
```

Explanation:

- Uses compound boolean filtering with `&` operator
- First condition: `data.subject == 'Graphic Design'` filters by subject
- Second condition: `data.price < '100'` filters by price threshold
- **Note**: Price comparison should ideally be numeric, but string comparison is used here

Expected Output: DataFrame containing only Graphic Design courses priced under 100.

Validation Check:

```python
data[(data.subject == 'Graphic Design') & (data.price == '100')]
```

This checks for courses exactly at the 100 price point.

8. Python Course Search

Question: List out all courses that are related with 'Python'.

```python
data[data.course_title.str.contains('Python')]
```

Explanation:

- Uses `.str.contains()` method for substring search
- Searches for 'Python' anywhere in the course title
- Case-sensitive search by default

Count of Python Courses:

```python
len(data[data.course_title.str.contains('Python')])
```

Expected Output:

- Filtered DataFrame showing all courses with 'Python' in the title
- Count showing the total number of Python-related courses

9. Temporal Analysis - Courses by Year

Question: What are courses that published in year 2015?

Step-by-step Implementation:

```python
# Check current data types
data.dtypes
```

```python
# Convert timestamp to datetime
data['published_timestamp'] = pd.to_datetime(data.published_timestamp)
```

```python
# Verify conversion
data.dtypes
```

```python
# Extract year from datetime
data['Year'] = data['published_timestamp'].dt.year
```

```python
# Filter for 2015 courses
data[data.Year == 2015]
```

Explanation:

- **Step 1**: Check data types to understand current format
- **Step 2**: Convert string timestamps to datetime objects using `pd.to_datetime()`
- **Step 3**: Verify successful conversion
- **Step 4**: Extract year component using `.dt.year` accessor
- **Step 5**: Filter DataFrame for courses published in 2015

Expected Output: DataFrame containing only courses published in 2015.

10. Advanced Grouping Analysis

Question: What are the Max. Number of Subscribers for Each Level of courses?

Check Available Levels:

```python
data.level.unique()
```

Analysis Methods:

Method 1 - Specific Column Aggregation:

```python
data.groupby('level')['num_subscribers'].max()
```

Method 2 - Complete Group Statistics:

```python
data.groupby('level').max()
```

Explanation:

- **Method 1**: Groups data by course level and finds maximum subscribers for each level
- **Method 2**: Groups by level and finds maximum values for all numeric columns
- Reveals which difficulty levels attract the most subscribers

Expected Output:

```
level
Beginner        500000
Intermediate    300000
Advanced        150000
Name: num_subscribers, dtype: int64
```

Key Insights and Patterns

Data Quality Considerations

1. **Price Data Type**: Price appears to be stored as string, which may affect numerical comparisons
2. **Date Handling**: Timestamp conversion is necessary for temporal analysis
3. **Text Search**: String contains operations are case-sensitive

Analytical Insights

1. **Course Distribution**: Identifies which subjects dominate the platform
2. **Pricing Strategy**: Separates free vs. paid course analysis
3. **Popularity Metrics**: Uses subscriber count as success indicator
4. **Temporal Trends**: Analyzes course publication patterns over time
5. **Skill Level Analysis**: Examines subscriber engagement across difficulty levels

Best Practices Demonstrated

1. **Data Exploration**: Using `.head()`, `.dtypes`, and `.unique()` for initial understanding
2. **Boolean Filtering**: Multiple condition filtering with logical operators
3. **String Operations**: Text search capabilities with `.str.contains()`
4. **Date Manipulation**: Proper datetime conversion and extraction
5. **Grouping Operations**: Advanced aggregation with `.groupby()`

Recommendations for Further Analysis

Additional Questions to Explore

1. **Revenue Analysis**: Calculate total revenue by subject or level
2. **Rating Correlation**: Analyze relationship between ratings and subscriber count
3. **Seasonal Trends**: Identify publication patterns by month/quarter
4. **Price Optimization**: Find optimal pricing for different subjects
5. **Course Length Analysis**: Examine relationship between duration and popularity

Code Improvements

1. **Data Type Conversion**: Convert price to numeric for proper comparisons
2. **Error Handling**: Add try-catch blocks for robust data processing
3. **Visualization**: Add charts and graphs for better insights
4. **Statistical Analysis**: Include correlation and statistical tests
5. **Data Validation**: Check for missing values and data quality issues

Conclusion

This analysis provides a comprehensive exploration of the Udemy courses dataset, demonstrating essential pandas operations for data analysis. The combination of filtering, sorting, grouping, and temporal analysis offers valuable insights into course popularity, pricing strategies, and platform trends. The documented approach serves as a foundation for more advanced analytics and business intelligence applications.