Northeastern University, Khoury College of Computer Science

CS 6220 Data Mining | Assignment 1

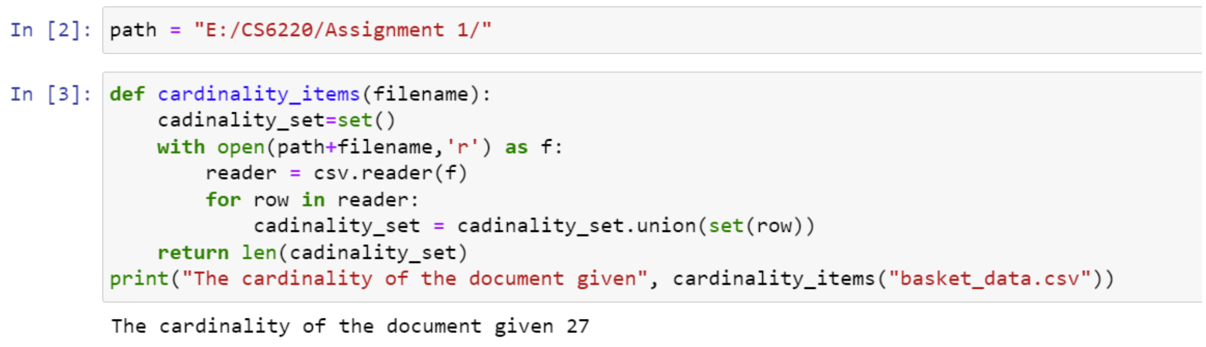
Due: January 18, 2024(100 points)

Dharun Suryaa Nagarajan
https://github.com/dharun4772/CS6220/tree/main/Assignment%201

Coding Review

1. The cardinality of a set or collection of items is the number of unique items in that collection. Write a function called cardinality_items that takes a .csv text string file as input, where the format is as the above, and calculates the cardinality of the set of all the grocery items in any given dataset.

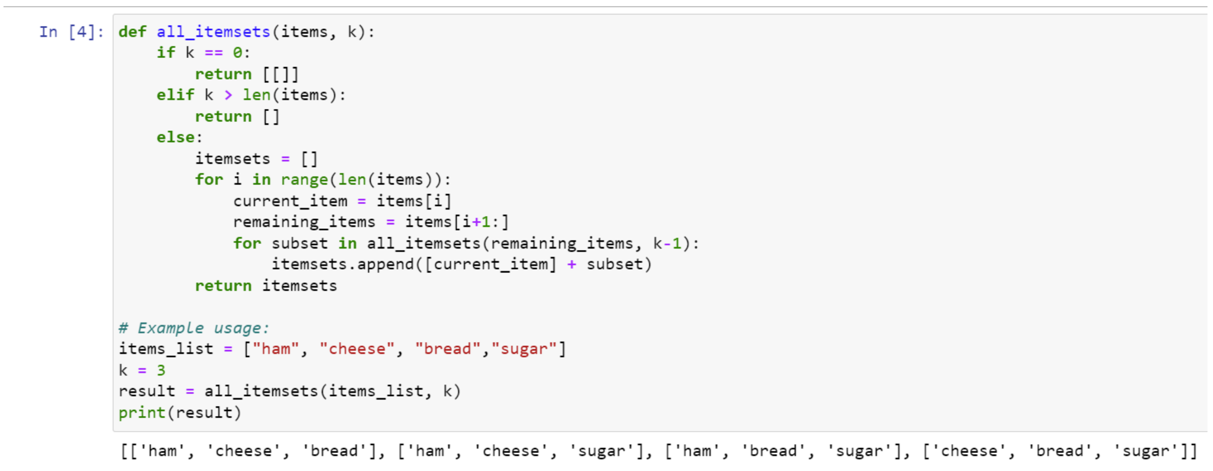
   Solution:
   The provided code aims to determine the cardinality of items within a document, specifically a CSV file named "basket_data.csv." Cardinality, in this context, refers to the number of unique items present in the document.
   a. The function takes a filename as a parameter, representing the CSV file to be analyzed.
   b. A set named cadinality_set is initialized to store unique items found in the document.
   c. Read the CSV file using opencv and iterate through the rows
   d. For each row, the unique items are added to the cadinality_set using the union operation.
   e. The function returns the length of the final cadinality_set, representing the cardinality of unique items in the document.



```
In [2]: path = "E:/CS6220/Assignment 1/"

In [3]: def cardinality_items(filename):
            cadinality_set=set()
            with open(path+filename,'r') as f:
                reader = csv.reader(f)
                for row in reader:
                    cadinality_set = cadinality_set.union(set(row))
            return len(cadinality_set)
        print("The cardinality of the document given", cardinality_items("basket_data.csv"))

        The cardinality of the document given 27
```

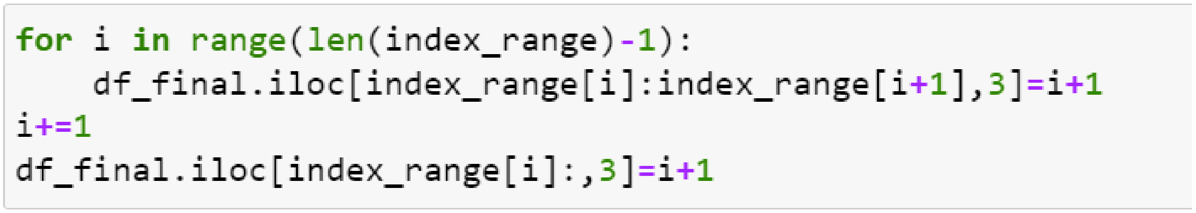
2. Write a function called all_itemsets that takes a list of unique items and an integer k as input, and the output is a list of all possible unique itemsets with non-repeating k items. That is, the output is L = [S1, S2, · · · SN], a list of all possible sets, where each Si has k items.

   Solution:
   The given Python code aims to generate all possible k-length itemsets from a list of items. The function all_itemsets takes a list of items and an integer k as input parameters and returns a list of all possible k-length itemsets.
   a. The function takes a list of items (items) and an integer (k) as parameters.
   b. The base cases are defined. If k is 0, an empty list (representing an empty itemset) is returned. If k is greater than the length of the items list, an empty list is returned, as it's not possible to form k-length itemsets.
   c. If k is greater than 0 and less than or equal to the length of items, the function proceeds to generate k-length itemsets recursively. For each item in the list, it considers the current item, removes it from the remaining items, and recursively generates (k-1)-length itemsets. The current item is then added to each subset, forming k-length itemsets.
   d. The function returns the list of all k-length itemsets generated.



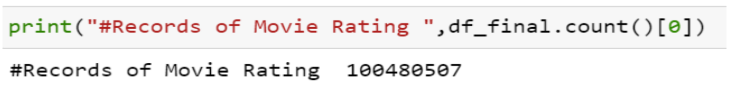
```
In [4]: def all_itemsets(items, k):
            if k == 0:
                return [[]]
            elif k > len(items):
                return []
            else:
                itemsets = []
                for i in range(len(items)):
                    current_item = items[i]
                    remaining_items = items[i+1:]
                    for subset in all_itemsets(remaining_items, k-1):
                        itemsets.append([current_item] + subset)
                return itemsets

        # Example usage:
        items_list = ["ham", "cheese", "bread","sugar"]
        k = 3
        result = all_itemsets(items_list, k)
        print(result)

        [['ham', 'cheese', 'bread'], ['ham', 'cheese', 'sugar'], ['ham', 'bread', 'sugar'], ['cheese', 'bread', 'sugar']]
```

3. Let’s review combined_data_*.txt:
   Data Cleaning: Since the data was not in a tabular format with movie as a separate row followed by the all customers rating for the movie. I used the pandas to first read the csv file using the three columns (Customer_id, rating, date) and then use transformation logic to remove the movie rows and add it as a column along each customer rating row under the name movie in the dataset using the indexing.



```
for i in range(len(index_range)-1):
    df_final.iloc[index_range[i]:index_range[i+1],3]=i+1
i+=1
df_final.iloc[index_range[i]:,3]=i+1
```

   a. How many total records of movie ratings are there in the entire dataset (over all of combined_data_*.txt)?



```
print("#Records of Movie Rating ",df_final.count()[0])

#Records of Movie Rating  100480507
```

   b. How many total unique users are there in the entire dataset (over all of combined_data_*.txt)?

```
print("#Unique users",len(df_final.customer_id.unique()))
```

```
#Unique users 480189
```

c. What is the range of years that this data is valid over?

3.c Range of years this data is valid for

```
print(f"The range of years this data is valid for is from {df_final.date.dt.year.min()} to {df_final.date.dt.year.max()}")
```

```
The range of years this data is valid for is from 1999 to 2005
```

4. Let's review movie_titles.csv:

Data Cleaning: Like hinted in the question the movie_titles file has commas in the movie title column that cannot be read in the typical way so no csv reader will work. Instead I used the inbuilt open function of python to read the file and then pull out information from each row separately using just two commas as separators and append the three variables into their respective columns. Finally used the columns to create a DataFrame structure in Python.

```
with open(path+"movie_titles.csv", 'r') as file:
    column1,column2,column3=[],[],[]
    for line in file:
        parts = line.strip().split(',')
        column1.append(parts[0])
        column2.append(parts[1])
        if(len(parts)>=2):
            column3.append(",".join(parts[2:]))

# Create a DataFrame using the extracted data
import pandas as pd

data = {'id': column1, 'year': column2, 'movie_title': column3}
df_movie = pd.DataFrame(data)
```

a. How many movies with unique names are there? That is to say, count the distinct names of the movies.

```
print("#Unique movie names",len(df_movie.movie_title.unique()))
```

```
#Unique movie names 17359
```

b. How many movie names refer to four different movies? 5 movies

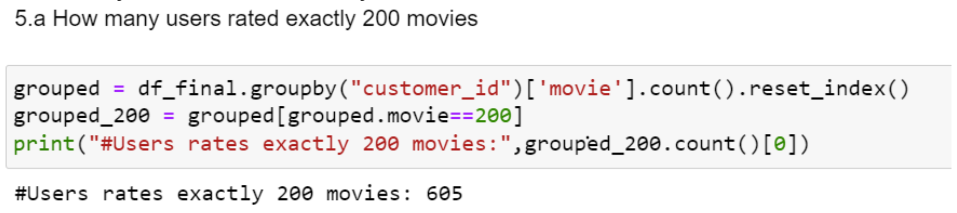
4.b How many movie names refer to four different movies?

```
grouped = df_movie.movie_title.value_counts().reset_index()
display(grouped[grouped.movie_title>=4])
print("#Movies with atleast four different ids", grouped[grouped.movie_title>=4].count()[0])
```

	index	movie_title
0	The Hunchback of Notre Dame	6
1	Oliver Twist	5
2	Anna Karenina	5
3	Hamlet	4
4	Treasure Island	4
5	Bad Company	4
6	Peter Pan	4
7	Pinocchio	4

```
#Movies with atleast four different ids 8
```

5. Let's review both:
   a. How many users rated exactly 200 movies?

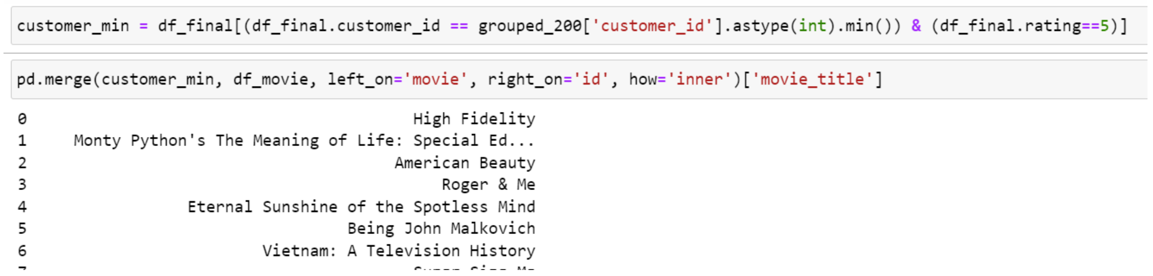


5.a How many users rated exactly 200 movies

```
grouped = df_final.groupby("customer_id")['movie'].count().reset_index()
grouped_200 = grouped[grouped.movie==200]
print("#Users rates exactly 200 movies:",grouped_200.count()[0])
```

```
#Users rates exactly 200 movies: 605
```

   b. Of these users, take the lowest user ID and print out the names of the movies that this person liked the most (all 5 star ratings).



```
customer_min = df_final[(df_final.customer_id == grouped_200['customer_id'].astype(int).min()) & (df_final.rating==5)]
```

```
pd.merge(customer_min, df_movie, left_on='movie', right_on='id', how='inner')['movie_title']
```

```
0                                        High Fidelity
1       Monty Python's The Meaning of Life: Special Ed...
2                                       American Beauty
3                                           Roger & Me
4                        Eternal Sunshine of the Spotless Mind
5                                  Being John Malkovich
6                           Vietnam: A Television History
```