Project Report

The goal of this project is to identify and match similar products from the Flipkart dataset with those in the Amazon dataset. Upon receiving user input, the model will retrieve the prices of the specified product from both Flipkart and Amazon and present them for comparison.

Sample output:

Product name in Flipkart	Retail Price in Flipkart	Discounted Price in Flipkart	Product name in Amazon	Retail Price in Amazon	Discounted Price in Amazon
FDT Women's Leggings	699	309	FDT WOMEN'S Leggings Pants	698	362

https://colab.research.google.com/drive/1_Olh5GjvklWxasFGRJKQ6lMDkezOpfl0?usp=sharing

Approach:

For this project, we are using a dataset that includes 15 features for each product:

'Uniq_id', 'crawl_timestamp', 'product_url', 'product_name', 'product_category_tree', 'pid', 'retail_price', 'discounted_price', 'image', 'is_FK_Advantage_product', 'description', 'product_rating', 'overall_rating', 'brand', 'product_specifications' .

Each product has a **unique ID and a Product Information Document (PID) number**, which remain constant even if the product's name changes. Our approach to comparing prices is based on the PID number of the product.
When a user inputs the **PID number**, we search for it in both the Flipkart and Amazon datasets. If the PID is found in both datasets, we display a table comparing the prices and key features of the product on both platforms. If the PID is not found in both datasets, we may show an error message or provide alternative options for the user.

Major Task is how we differentiate the Multiple results:

To address the issue of multiple results when a user searches for a product, we can use the **"product specification"** feature to differentiate between the different products and avoid confusing the user. However, this feature may contain a lot of noise that needs to be cleaned and preprocessed before it can be used effectively. We can extract relevant features by removing the noise and replacing null values with the status "no info" to avoid errors later on.

When displaying the results to the user, it will be important to include the key differences between the multiple products in the comparison table. This can help the user understand the variations between the products and make an informed decision.

In this example, if a user searches for a specific type of t-shirt, the search results may include multiple t-shirts from the same company that come in different pack sizes (e.g. a pack of 2 t-shirts, a pack of 3 t-shirts, etc.). Each of these t-shirts may have a different price due to the different pack sizes.

Multiple result example output:

In this example, the user searches for the product "FabHomeDecor Fabric Double Sofa Bed" and multiple results are returned with the same name. However, upon examining the "product specification" feature, it is revealed that the color of the sofa varies between the results.
This color difference is a key feature that affects the price of the sofa.

	Product name in Filpkart	Retail Price in Filpkart	Retail Price in Flipkart	Product name in Amazon	Retail Price in Amazon	Discounted Price in Amazon	Key Features
0	FabHomeDecor Fabric Double Sofa Bed	32157.0	22646.0	FabHomeDecor Fabric Double Sofa Bed	32143	29121	{ Black, FHD112, Leatherette Black}
1	FabHomeDecor Fabric Double Sofa Bed	32157.0	22646.0	FabHomeDecor Fabric Double Sofa Bed	32137	28664	{ Brown, FHD107}
2	FabHomeDecor Fabric Double Sofa Bed	32157.0	22646.0	FabHomeDecor Fabric Double Sofa Bed	32150	28650	{ Purple, FHD132}
3	FabHomeDecor Fabric Double Sofa Bed	32157.0	22646.0	FabHomeDecor Fabric Double Sofa Bed	32144	26423	{ keyInstallation & Demo Details, product_spec...

Explanation:

The code was divided into four sections for proper understanding

- **Importing libraries and Datasets:**

  In this section of the code, the necessary datasets are loaded into the environment and the pandas and numpy libraries are imported. It is important to verify that the datasets are properly uploaded before proceeding with any further processing.

- **Observations / EDA:**

  Over this section, the strength and characteristics of the dataset are assessed. This includes checking the column names, data types of each feature, and the presence of null values. It is important to ensure that the dataset is clean and accurate before proceeding with any analysis or comparison.
  If necessary, the data can be cleaned and preprocessed to correct any errors or issues.

  For example, if the prices of some products are negative, they can be replaced with a value of "0" to avoid any issues later on.
  Similarly, null values in the product description can be replaced with the status "no info" to ensure that the data is complete and consistent. These steps will help ensure that the dataset is reliable and ready for further analysis.

```
amazon["product_specifications"].fillna("No info", inplace = True)
filpkart["product_specifications"].fillna("No info", inplace = True)
```

  Later some features were dropped which includes
  {"product_url","product_category_tree","image","is_FK_Advantage_product","description", "product_rating","overall_rating","brand". **[using user defined pre_processing func]**

  Final features were :
  uniq_id, pid, retail and discount price, product_specification and timestamp of crawl.

- **Approach:**

  This section of the code is responsible for handling the user input and retrieving the details of the product.

  >> This section of the code involves receiving input from the user and using it to retrieve the PID number(s) of the product(s) from the Flipkart dataset. If multiple products are available with the same name, all of the PID numbers will be recorded.

  >> In this step of the process, two lists are generated: one called **"amazon_pid_list"** and one called **"flipkart_pid_list"** These lists will consist of all of the PID numbers for the products in the Amazon and Flipkart datasets, respectively.

  >> **Searching_filpkart**

The func searching_filpkart will look generate a dictionary and return to the variable called filpkart_results, The dictionary consist of {index: [name, id, PID, retail_price, discount, product_sepecifications] With index as key and features in a list.	

```
def searching_filpkart(product):
  index = 0
  result_index={}
  for i in filpkart.product_name==product:
    if(i==True):
      result_index[index] = [filpkart.product_name[index],
    index+=1

  return result_index
```

  Output:
  Filpkart_results a dictionary data type with features listed in values.

  >> **Searching_amazon** and concating the both the results

```
def searching_amazon(details):
  if(details[2] in amazon_pid_list):
    filpkart_results = [details[0],details[3],details[4]]
    variations.append(details[-1])
    ref = amazon[amazon["pid"]==details[2]]
    ref = ref.values.tolist()[0]
    amazon_results = [ref[2],ref[4],ref[5]]

    ## concating the both filpkart and amazon result into the final_results
    final_result.append(filpkart_results + amazon_results)
```

  In this step, the PID numbers from the Flipkart results are passed into a function called "searching_amazon." This function searches the "amazon_pid_list" for the matching PID numbers. If a match is found, the code retrieves the product details (such as the name, retail price, and discounted price) from both the Flipkart and Amazon datasets and stores them in a variable called "final_result."

  If there are **multiple products with the same name**, the code is generating a "Variations list" in which the "product_specifications" for each product are appended. The "product_specifications" feature is likely to contain a lot of noise and may need to be cleaned and preprocessed to extract the key features.

Example: For a product such as a pack of shirts, the "product_specifications" feature may contain a lot of noise and extraneous information.
Here is an example of what the raw "product_specifications" data might look like:

{"product_specification"=>[{"key"=>"Number of Contents in Sales Package", "value"=>"Pack of 3"}, {"key"=>"Fabric", "value"=>"Cotton Lycra"}, {"key"=>"Type", "value"=>"Cycling Shorts"}, {"key"=>"Pattern", "value"=>"Solid"}, {"key"=>"Ideal For", "value"=>"Women's"}, {"value"=>"Gentle Machine Wash in Lukewarm Water, Do Not Bleach"}, {"key"=>"Style Code", "value"=>"ALTHT_3P_21"}, {"value"=>"3 shorts"}]}

Once noise was reduced our final result will be
{ Pack of 3, 3 shorts, ALTHT_3P_21}
The key features of this product include: 3 shorts in a pack and the model number.

>>**Final Result**
In the "final result" section, the code will convert the output data (including the price comparison and the varying parameters) into a data frame format using the Pandas library. The data frame can then be used to display the results to the user in a structured and organized way.

- **SAMPLE OUTPUTS:**

Product: FDT Women's Leggings from Flipkart

	Product name in Filpkart	Retail Price in Filpkart	Retail Price in Flipkart	Product name in Amazon	Retail Price in Amazon	Discounted Price in Amazon	Key Features
0	FDT Women's Leggings	699.0	309.0	FDT WOMEN'S Leggings Pants	698	362	{Same key features found}

Product: FabHomeDecor Fabric Double Sofa Bed

	Product name in Filpkart	Retail Price in Filpkart	Retail Price in Flipkart	Product name in Amazon	Retail Price in Amazon	Discounted Price in Amazon	Key Features
0	FabHomeDecor Fabric Double Sofa Bed	32157.0	22646.0	FabHomeDecor Fabric Double Sofa Bed	32143	29121	{ Leatherette Black, FHD112, Black}
1	FabHomeDecor Fabric Double Sofa Bed	32157.0	22646.0	FabHomeDecor Fabric Double Sofa Bed	32137	28664	{ FHD107, Brown}
2	FabHomeDecor Fabric Double Sofa Bed	32157.0	22646.0	FabHomeDecor Fabric Double Sofa Bed	32150	28650	{ FHD132, Purple}
3	FabHomeDecor Fabric Double Sofa Bed	32157.0	22646.0	FabHomeDecor Fabric Double Sofa Bed	32144	26423	{ 939.8 mm, Brown, 838.2 mm, FHD115, keyln...

Product: Shopmania Music Band A5 Notebook Spiral Bound

	Product name in Filpkart	Retail Price in Filpkart	Retail Price in Flipkart	Product name in Amazon	Retail Price in Amazon	Discounted Price in Amazon	Key Features
0	Shopmania Music Band A5 Notebook Spiral Bound	499.0	275.0	SHOPMANIA MUSIC BAND A5 NOTEBOOK SPIRAL BOUND	483	347	{ NB00664}
1	Shopmania Music Band A5 Notebook Spiral Bound	499.0	275.0	SHOPMANIA MUSIC BAND A5 NOTEBOOK SPIRAL BOUND	487	330	{ NB00678}

In the "final results" a table with the price comparison for the products will be displayed.
If there are multiple products with the same name, the table will include additional columns to show the key features such as quantity, color variation, and model number. These features will help the user differentiate between the products and make an informed decision.