Project- Report

Product price analysis and comparison from various website

Department of Computer Science

ITCS 6190/8190: Cloud Computing for Data Analysis

Submitted to

Dr. Srinivas Akella

Submitted By

Adithi Amuru	Ayappa Krishnappa	Ravi Kumar Mittapalli
801099858	801101748	801103210
aamuru@uncc.edu	akrish19@uncc.edu	rmittapa@uncc.edu

PROBLEM FORMULATION

1.1 Introduction

The problem we have chosen to address in this project is "Product price analysis and comparison from various website". Here we are analyzing the dataset from websites like Amazon and Walmart. A method called web scrapping was used to get the dataset. The project concentrates on recommending products based on rating, purchases and other such factors.

1.2 Formulation

- Goal: To create a search engine, where using IR methods to recommend products based on purchase history
- Software used: pyspark, Flask API for data analysis, Scrapy (web scrapping), Android App (UI)
- Information Retrieval algorithm: Indexing of single and bi-word keywords.

DATA AND PRE-PROCESSING

2.1 Acquiring the Dataset

The dataset used for this project is acquired from online shopping websites Walmart and Amazon. The data comprises of the mobile phone data available from these websites. A process call web scrapping was used to collect this real-time data. The data mainly contains details like Title of the product, product image, cost, rating given by the users. This dataset can be extended for further extended for acquiring other details about the product.

A framework called "Scrapy" was used to collect the data. This framework was originally designed for web scraping, it can also be used to extract data using APIs or as a general-purpose web crawler. This framework was written in python which can be used as an extension to "pyspark".

How Scrapy works? The architecture of Scrapy contains five main components: Scrapy Engine, Scheduler, Downloader, Spiders and Item Pipelines.

The Scrapy engine is the main component of Scrapy which is aimed at controlling the data flow between all other components. The engine generates requests and manages events against an action. The scheduler receives the requests sent by the engine and queues them.

The objective of the downloader is to fetch all the web pages and send them to the engine. The engine then sends the web pages to the spider. Spiders are the codes you write to parse websites and extract data. The item pipeline processes the items side by side after the spiders extract them. The collected data is raw and requires pre-processing. The pre-processing methods used was text processing, removal of incomplete data and missing data. The scrapping also results in some unwanted data, this data needs to be removed from the necessary data.

ALGORITHMS AND PROCESSES

3.1 Information Retrieval Algorithms

Information retrieval (IR) is the activity of obtaining information system resources that are relevant to an information need from a collection of those resources. Searches can be based on full-text or other content-based indexing. Information retrieval is the science of searching for information in a document, searching for documents themselves, and searching for the metadata that describes data, and for databases of texts, images or sounds.

An information retrieval process begins when a user enters a query into the system. Queries are formal statements of information needs, for example search strings in web search engines. In information retrieval a query does not uniquely identify a single object in the collection. Instead, several objects may match the query, perhaps with different degrees of relevancy.

A basic method of information retrieval used in this project is where the user enters the keyword and the UI and the spark work in the background to retrieve the relevant data.

3.2 Spark and Python

Spark provides an interface for programming entire clusters with implicit data parallelism and fault tolerance. Apache Spark has as its architectural foundation the Resilient Distributed Dataset (RDD), a read-only multiset of data items distributed over a cluster of machines, that is maintained in a fault-tolerant way.

Python is a multi-paradigm programming language. Object-oriented programming and structured programming are fully supported, and many of its features support functional programming and aspect-oriented programming (including by metaprogramming and metaobjects (magic methods)). Many other paradigms are supported via extensions, including design by contract and logic programming.

Python uses dynamic typing and a combination of reference counting and a cycle-detecting garbage collector for memory management. It also features dynamic name resolution (late binding), which binds method and variable names during program execution. Pyspark is a collaboration of python a spark, where we could implement the necessary Information retrieval algorithms.

3.3 Flask API

Flask-RESTful is an extension for Flask that adds support for quickly building REST APIs. It is a lightweight abstraction that works with your existing ORM/libraries. Flask-RESTful encourages best practices with minimal setup. If you are familiar with Flask, Flask-RESTful should be easy to pick up.

This data could be represented in a database, saved in a file, or be accessible through some network protocol, but for us an in-memory data structure works fine. One of the purposes of an API is to decouple the data from the application that uses it, hiding the data implementation details. Here we have converted this data obtained into JSON format, displayed on a website.

3.4 Android UI

The user interface (UI) for an Android app is built as a hierarchy of layouts and widgets. The layouts are ViewGroup objects, containers that control how their

child views are positioned on the screen. Widgets are View objects, UI components such as buttons and text boxes.

Android provides an XML vocabulary for ViewGroup and View classes, so most of your UI is defined in XML files. However, rather than teach you to write XML, this lesson shows you how to create a layout using Android Studio's Layout Editor. The Layout Editor writes the XML for you as you drag and drop views to build your layout.

METHODOLOGY IMPLEMENTED

4.1 Web Scraping

A python framework Scarpy was used for web scrapping data from online shopping websites Amazon and Walmart. The dataset is related to mobile and their details. Titles, rating, cost, product image of the products is scraped. This web crawled data is processed and displayed on the Android UI to search the product required. The data thus acquired was cleaned and processed. First the text is processed, split by text, single word and bi-word indices are retrieved. The data scraped are in JSON format displayed on a hosted website. The search algorithm uses the indices to retrieve the data on the UI. The data scraping can be extended to get all kinds of data from different websites.

4.2 IR Algorithms and Recommendation systems

We have implemented the single word and bi-word indexing for forming the tokens. These indices are used as query for information retrieval. When a keyword is inserted in the search engine, the query is made based on this index compared with indices and the corresponding product data is retrieved. This information algorithm can be extended to search for sentences.

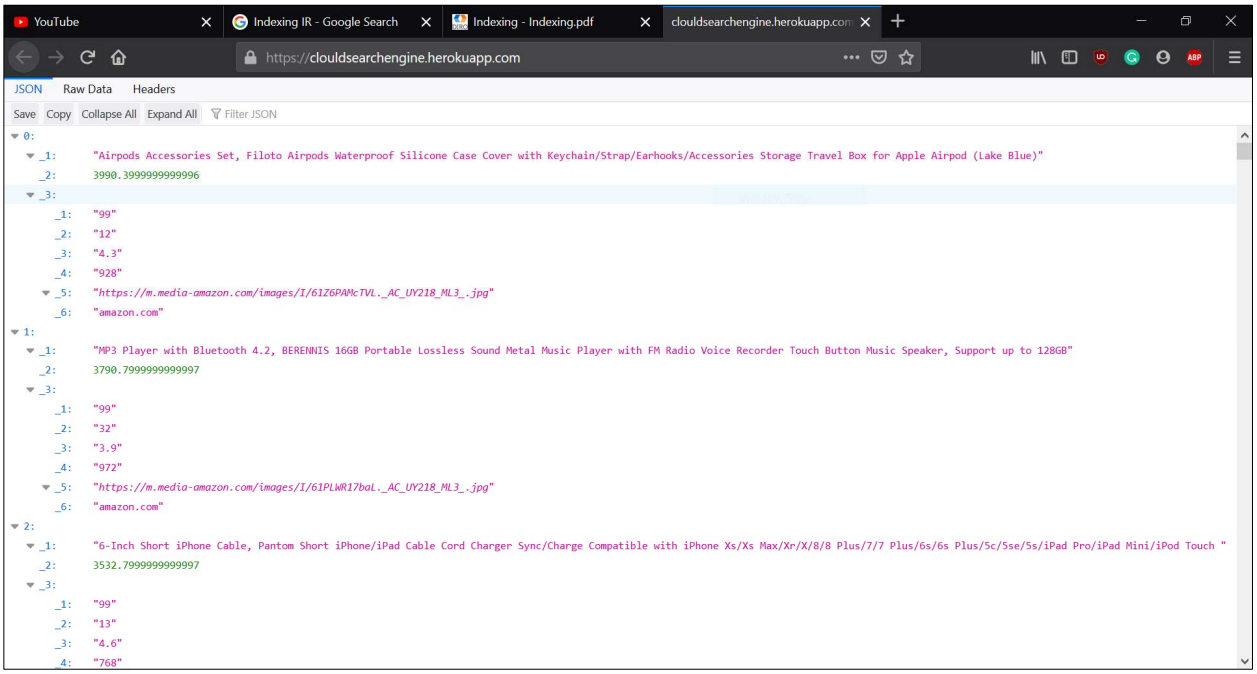
The map and reduce implementation are done using spark using pyspark where the SparkContext and SparkSession are used. The data scraped are in JSON format, which is processed and split to create the indices. The recommendation of the products is implemented in the python program. The ratings by the user retrieved is compared between the products and then recommended. The results

are reflected on the Android UI created to capture the results. The top 100 recommended results are displayed on the UI with the image and other details. The recommended data is also stored in the a separate JSON file which is used in the UI.

4.3 User Interface

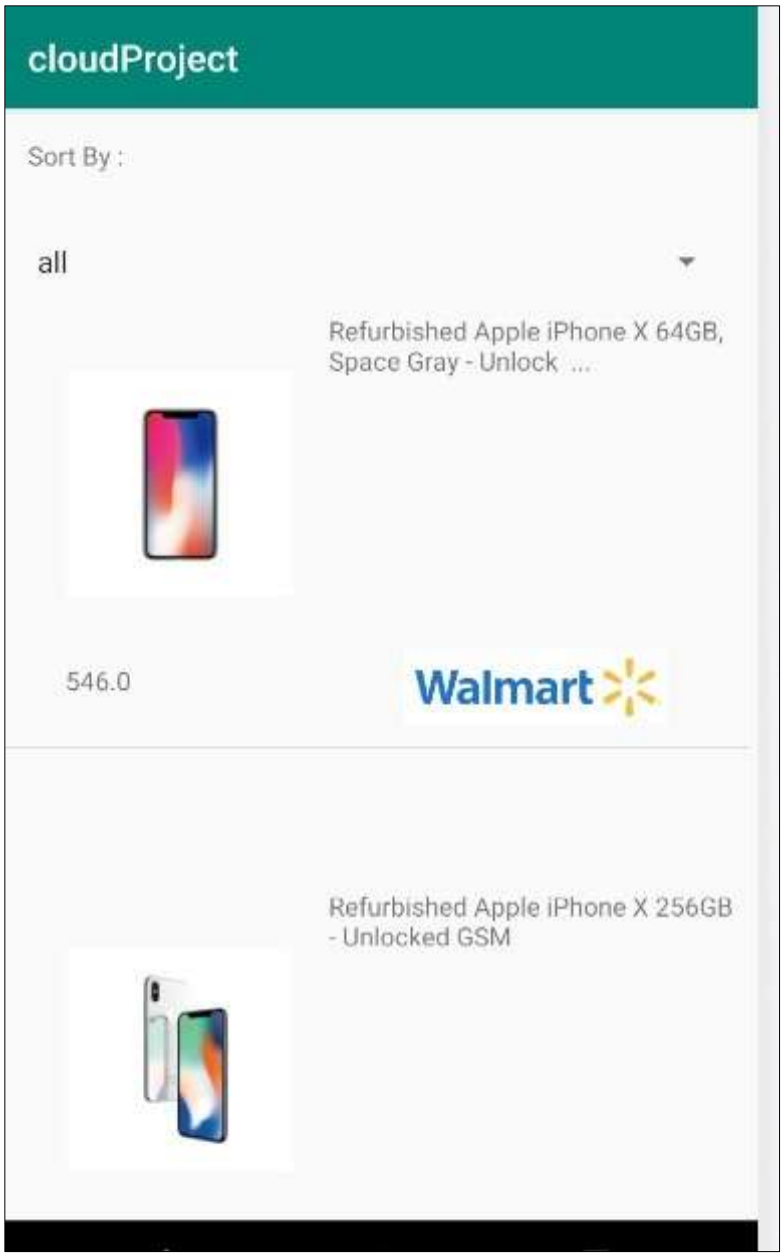
We have used the flask API to display the recommended products and to display the Search Engine. The user can insert the keyword in the Android App provided, the backend system compares with the results from JSON files and displayed the details of the product containing the keyword entered.

Snapshots of the UI:

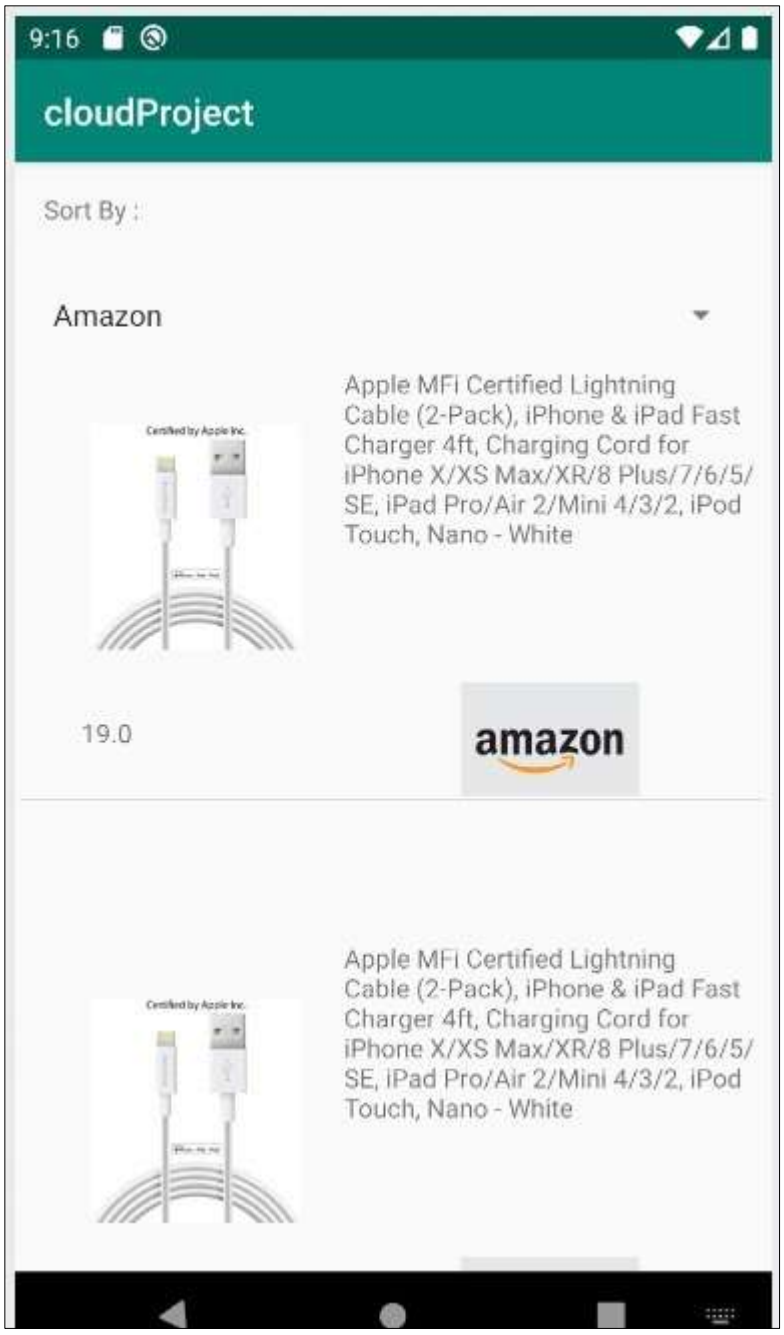


Website showing the retrieved JSON data

Home Page of the Android UI



Showing products from Walmart



Showing results from Amazon

Motivation and Accomplished tasks

5.1 Context of the project

We were motivated to implement a recommendation system for the users will to purchase mobiles and its products. This data is specifically chosen from websites like Amazon and Walmart since it contains a huge number of users and most users give reviews and ratings which we are using to build our recommendation system.

We have surely accomplished the web scraping, and implementation of the single word and bi-word indexing. The querying systems can be further be extended to search using sentence processing. Also, this web scrapping can be extended to get data from various other websites. Since the user can sort based on the website it provided the user to choose and better website to place the order.

5.2 Division of tasks

We have divided the task in three parts, Web crawling, indexing and UI. The web crawling was implemented by Adithi Amuru, the indexing was achieved by Ayappa Krishnappa and finally the UI was created and connected by Ravi Mittappali.

5.3 Challenges faced

There were challenges faced at all stages of the program. The web crawled data needed to be given a structure, so as the UI and the python program can process. The indexing included the splitting the data into single indices and bi-word indices. The bi-word indices were trickier compared the single word indices. The Android UI is the next stage where this JSON stored needs to be displayed as an APP. The construction the UI was new for us and was an interesting subject to learn.

References

1. Introduction to information retrieval, Textbook by Christopher D. Manning, Hinrich Schütze, and Prabhakar Raghavan
2. https://scrapy.org/
3. https://towardsdatascience.com/a-brief-introduction-to-pyspark-ff4284701873