



Electric Essentials Sales using Demographics

A PROJECT REPORT

ON

BIG DATA LAB

24CAP-684

Submitted by

Vishnu Pratap Singh Jaswal (24MCC20010)

in partial fulfillment for the award of the degree of

**MASTER OF COMPUTER APPLICATION
(CLOUD COMPUTING AND DEVOPS)**

Chandigarh University

APRIL 2025



BONAFIDE CERTIFICATE

Certified that this project report “**Electric Essentials Sales using Demographics**” is the Bonafide work of “**Vishnu Pratap Singh Jaswal**” who carried out the project work under my/our supervision.

SIGNATURE
Dr. Krishan Tuli

SIGNATURE
Mr. Rishabh Tomar

HEAD OF THE DEPARTMENT
UIC

SUPERVISOR
Assistant Professor

Submitted for the project viva voce examination held on

INTERNAL EXAMINER

EXTERNAL EXAMINER

Declaration

I at this moment declare that the project report entitled “**Electric Essentials Sales using Demographics**” Submitted by me to the **University Institute of Computing, Chandigarh University, Gharuan**, in partial fulfillment of the requirement for the award of the degree “**Master of Computer Application- Cloud Computing & DevOps**” is a Bonafide project work carried out by me under the guidance of “**Mr. Rishabh Tomar**,” I further declare that the work reported in this project has not been submitted and will not be submitted, either in part or in full, for the award of any other degree or diploma in this institute or any other institute or university.

Project Guide: Mr. Rishabh Tomar

HOD: Dr. Krishan Tuli

Date: April 2025

Submitted by:

Vishnu Pratap Singh Jaswal (24MCC20010)

Certificate Of Originality

This is to certify that the project report entitled “**Electric Essentials Sales using Demographics**” submitted by me in partial fulfillment of the requirements for the award of the Degree Master of Computer Application- Cloud Computing & DevOps (MCA CC & DevOps) is a bonafide record of the work carried out under my guidance and supervision at the University Institute of Computing of the Chandigarh University.

Submitted by:

Vishnu Pratap Singh Jaswal (24MCC20010)

ACKNOWLEDGMENT

I would like to express my sincere gratitude to everyone who contributed to the successful completion of this project. First and foremost, I would like to thank Mr. Rishabh Tomar Sir, my project mentor, for their invaluable guidance, support, and encouragement throughout the entire process. Their insights and expertise have been instrumental in shaping this project, and I am deeply grateful for their time and effort.

I would also like to thank University Institute of Computing for providing the resources and platform necessary for conducting this project. The tools and facilities made available were crucial in bringing this project to fruition. A special mention to my family and friends for their unwavering support, patience, and encouragement, without which this project would not have been possible.

Lastly, I would like to express my heartfelt thanks to my peers and colleagues for their collaborative spirit and helpful feedback during various stages of the project. Your input helped improve the quality and scope of the work.

Thank you all for your contributions to making this project a success.

Vishnu Pratap Singh Jaswal (24MCC20010)

TABLE OF CONTENTS

TITLE	PAGE NO.
Bonafide Certificate	2
Declaration	3
Certificate of Originality	4
Acknowledgement	5
Project Description	6
Introduction	7
Implementation	9-12
Results	13-18
Key Learnings and Observations	18
Application In Real-World	19
Conclusion	20
Plagiarism Report	21

Project Description

"Electric Essentials Sales using Demographics" aims to analyze retail sales data to analyze customer behavior and revenue patterns across demographics for items used for electric purposes. Apache Pig is used for data preprocessing and transformation on Hadoop in a Linux environment.

INTRODUCTION

Retail sales analysis is a critical aspect of understanding customer behavior, optimizing inventory, and enhancing marketing strategies. With the exponential growth of big data, traditional tools struggle to handle large-scale datasets efficiently. This project leverages the Hadoop ecosystem, specifically Apache Pig, to preprocess and analyze retail sales data stored in HDFS.

Apache Pig simplifies complex ETL (Extract, Transform, Load) tasks by providing a high-level scripting language called Pig Latin. It abstracts the complexities of MapReduce programming, enabling efficient filtering, grouping, and aggregation of data. By analyzing sales across demographic attributes such as age groups, regions, and purchasing patterns, businesses can gain actionable insights into customer preferences and revenue drivers.

This project demonstrates an end-to-end workflow for sales analysis using retail demographics. It involves loading raw sales data into HDFS, cleansing it using Apache Pig scripts, aggregating metrics like revenue by region or customer segment, and generating meaningful insights to support decision-making processes in retail management.

TOOLS AND TECHNOLOGY USED

1. **Apache Pig:** For data preprocessing and transformation.
2. **Apache Hadoop:** For distributed data storage and processing.
3. **Linux OS (Ubuntu 16.04 or higher):** Operating system for running the Hadoop and Pig ecosystem.
4. **Java (JDK 1.7 or higher):** Required for Hadoop and Pig execution.

SYSTEM REQUIREMENTS

1. **Operating System:** Ubuntu 16.04 or higher.
2. **Processor:** x64 architecture with 2 GHz or higher.
3. **RAM:** Minimum 8 GB (16 GB recommended for large datasets).
4. **Storage:** At least 100 GB of disk space for Hadoop and dataset storage

IMPLEMENTATION

- For implementation of the project, we are using the Oracle VirtualBox to run the ubuntu operating system inside that.
- In order to run the Pig Latin Scripts, we will first setup the Hadoop in the Ubuntu Operating System. For that we will go to official website of Apache Hadoop www.hadoop.apache.org/ and download the hadoop-3.3.6 version.
- After successful installation we will extract the files of the downloaded tar files.
- Then we will download the java development kit through command line interpreter of Ubuntu from the official website of Java that is www.oracle.com/java/technologies/downloads/#java8.

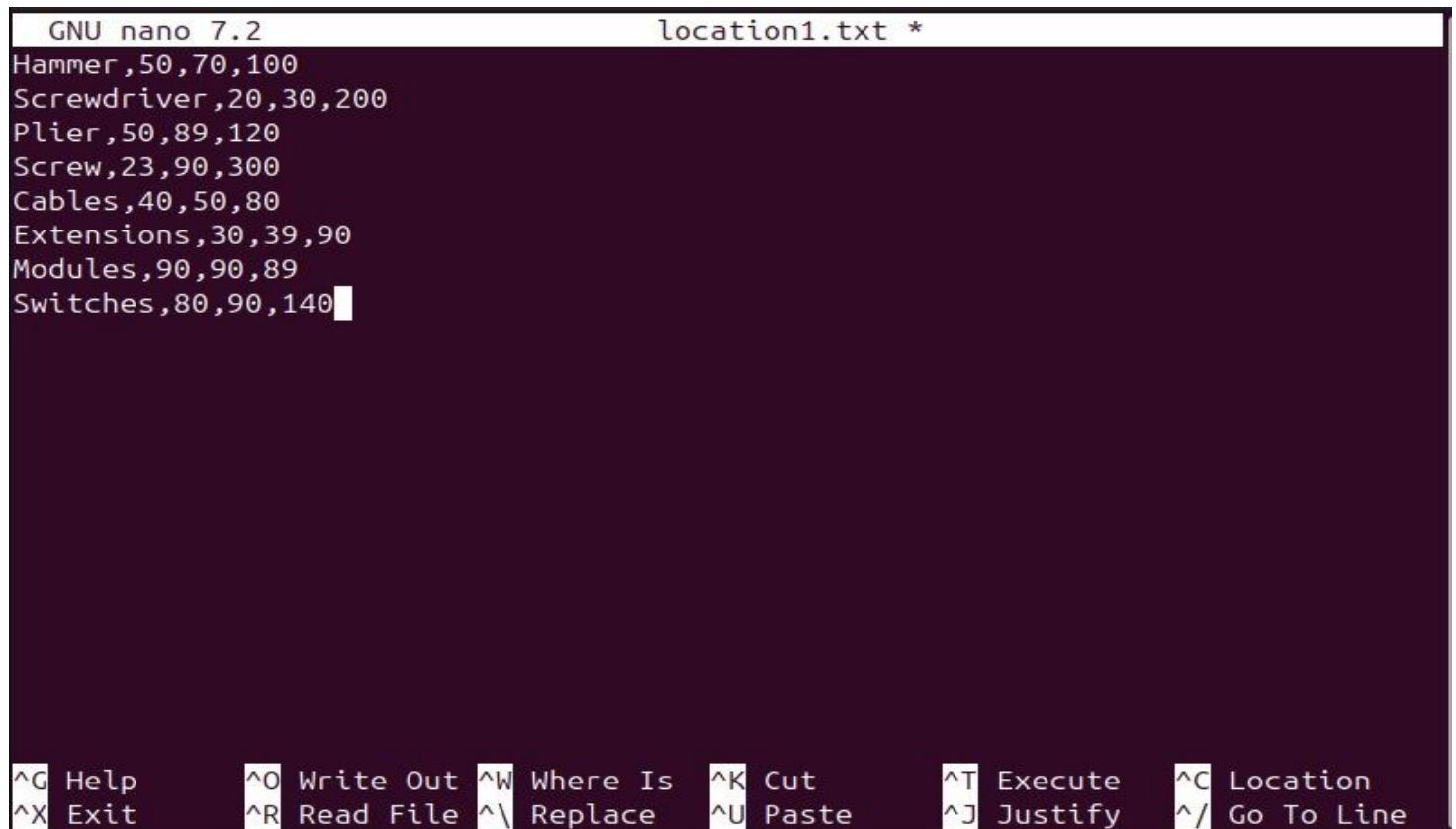
Data Modelling:

In data modelling we are deciding which type and nature of data we want to insert in the pig to filter the desired output of the retail analysis. We will look forward to model the data in forms of tables that needed to be integrated to get the desired output.

After successful completion the modelling, the output should show the most sold product in all the locations , most sold product on different location, most liked at all and different locations and the end the product which is least sold at end is also shown.

To implement the whole project we can work with the following steps as well:

- 1. Loading data:** This steps consist of typing the required datasets through hands in nano based editor inside ubuntu terminal. The commands and the files are as follows:



```
GNU nano 7.2 location1.txt *
Hammer,50,70,100
Screwdriver,20,30,200
Plier,50,89,120
Screw,23,90,300
Cables,40,50,80
Extensions,30,39,90
Modules,90,90,89
Switches,80,90,140
^G Help      ^O Write Out ^W Where Is  ^K Cut      ^T Execute  ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste    ^J Justify  ^/ Go To Line
```

```
GNU nano 7.2 location2.txt
Hammer,60,89,90
Screwdriver,30,78,10
Plier,78,90,170
Screw,45,90,200
Cables,67,89,90
Extensions,28,89,400
Modules,23,80,45
Switches,78,90,100

[ Read 8 lines ]
^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^/ Go To Line
```

```
GNU nano 7.2 location3.txt
Hammer,40,89,160
Screwdriver,23,89,60
Plier,23,90,300
Screw,60,89,180
Cables,60,89,140
Extensions,39,80,70
Modules,40,89,90
Switches,40,69,80

[ Read 8 lines ]
^G Help      ^O Write Out ^W Where Is  ^K Cut       ^T Execute   ^C Location
^X Exit      ^R Read File ^\ Replace   ^U Paste     ^J Justify   ^/ Go To Line
```

At the end after saving each of the files the terminal will look like this:

```
vboxuser@Ubuntu:~$ nano location1.txt
vboxuser@Ubuntu:~$ nano location2.txt
vboxuser@Ubuntu:~$ nano location3.txt
vboxuser@Ubuntu:~$
```

- 2. Finding most sold product in overall locations:** This consists of finding the most sold product of the company by combining the different locations like location1, location2 and location3. Here are the commands for that:

```
grunt> sales_group= GROUP sales BY product;
grunt> total_sales= FOREACH sales_group GENERATE group AS product, SUM(sales.offline_sales) + SUM(sales.online_sales) AS total_sold;
grunt> most_sold_product= ORDER total_sales BY total_sold DESC;
grunt> dump most_sold_product;
```

- 3. Most sold product at each location:** Here the most sold product at different locations is displayed by running the commands:

```
grunt> all_sales= UNION location1, location2, location3;
grunt> grouped_by_location= GROUP all_sales BY (location,product);
grunt> sales_per_location= FOREACH grouped_by_location GENERATE group.location AS location, group.product AS product, SUM(all_sales.offline_sales) + SUM(all_sales.online_sales) AS total_sold;
grunt> most_sold_by_location= ORDER sales_per_location BY location, total_sold DESC;
grunt> DUMP most_sold_by_location;
```

- 4. Finding the most liked product in all locations:** In this the most liked product is displayed in overall locations. By running the commands you will get the most liked product on all the locations.

```
grunt> liked_group= GROUP sales BY product;
grunt> total_liked= FOREACH liked_group GENERATE group AS product, SUM(sales.wishlisted) AS total_wishlisted;
grunt> most_liked_product= ORDER total_liked BY total_wishlisted DESC;
grunt> DUMP most_liked_product;
```

5. Finding the most liked product at each location: Here the most liked product at different locations that is location 1, location 2 and location 3. The following commands is used to find out this:

```
grunt> liked_by_location= GROUP all_sales BY (location,product);
grunt> total_liked_location= FOREACH liked_by_location GENERATE group.location AS location, group.product AS product, SUM(all_sales.wishlisted) AS total_wishlisted;
grunt> most_liked_by_location= ORDER total_liked_location BY location,total_wishlisted DESC;
grunt> DUMP most_liked_by_location;
2025-03-31 10:16:09,069 [main] INFO org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: GROUP_BY,ORDER_BY,UNION
```

6. Finding products with decremementing sales: The product whose sales are going down is shown through this commands. The commands are as following :

```
grunt> lowest_sales_by_location= ORDER sales_per_location BY location,total_sold ASC;
grunt> DUMP lowest_sales_by_location;
2025-03-31 10:21:03,732 [main] INFO org.apache.pig.tools.pigstats.ScriptState - Pig features used in the script: GROUP_BY,ORDER_BY,UNION
```


RESULTS

The outputs of the commands are as following that make the overall project successful. The commands result are as following:

1. **Data loading in Pig:** The data is loaded to the Pig through this command successfully. Here is the dump command that is used to load the data to the pig storage is as shown:

```
2025-04-01 07:30:10,122 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Total input paths
o process : 1
(Hammer,50,70,100)
(Screwdriver,20,30,200)
(Plier,50,89,120)
(Screw,23,90,300)
(Cables,40,50,80)
(Extensions,30,39,90)
(Modules,90,90,89)
(Switches,80,90,140)
(Hammer,60,89,90)
(Screwdriver,30,78,10)
(Plier,78,90,170)
(Screw,45,90,200)
(Cables,67,89,90)
(Extensions,28,89,400)
(Modules,23,80,45)
(Switches,78,90,100)
(Hammer,40,89,160)
(Screwdriver,23,89,60)
(Plier,23,90,300)
(Screw,60,89,180)
(Cables,60,89,140)
(Extensions,39,80,70)
(Modules,40,89,90)
(Switches,40,69,80)
grunt>
```

2. **Finding most sold product overall:** This consists of finding the most sold product of the company by combining the different locations like location1, location2 and location3. Here are the commands for that:

```
vboxuser@Ubuntu: ~
Total bags proactively spilled: 0
Total records proactively spilled: 0

Job DAG:
job_local2042448301_0002

2025-04-01 07:51:17,030 [main] WARN org.apache.hadoop.metrics2.impl.MetricsSystemImpl - JobTracker metrics system already initialized!
2025-04-01 07:51:17,033 [main] WARN org.apache.hadoop.metrics2.impl.MetricsSystemImpl - JobTracker metrics system already initialized!
2025-04-01 07:51:17,034 [main] WARN org.apache.hadoop.metrics2.impl.MetricsSystemImpl - JobTracker metrics system already initialized!
2025-04-01 07:51:17,041 [main] INFO org.apache.pig.backend.hadoop.executionengine.mapReduceLayer.MapReduceLauncher - Success!
2025-04-01 07:51:17,057 [main] INFO org.apache.hadoop.conf.Configuration.deprecation - io.bytes.per.checksum is deprecated. Instead, use dfs.bytes-per-checksum
2025-04-01 07:51:17,058 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2025-04-01 07:51:17,062 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input files to process : 1
2025-04-01 07:51:17,063 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(Plier,590)
(Screw,680)
(Cables,310)
(Hammer,350)
(Modules,224)
(Switches,320)
(Extensions,560)
(Screwdriver,270)
grunt>
```

3. **Most sold product at each location:** Here the most sold product at different locations is displayed by running the commands:

```
(location1,Hammer,50,70,100)
(location1,Screwdriver,20,30,200)
(location1,Plier,50,89,120)
(location1,Screw,23,90,300)
(location1,Cables,40,50,80)
(location1,Extensions,30,39,90)
(location1,Modules,90,90,89)
(location1,Switches,80,90,140)
grunt>
```

```
(location2,Hammer,60,89,90,,,)
(location2,Screwdriver,30,78,10,,,)
(location2,Plier,78,90,170,,,)
(location2,Screw,45,90,200,,,)
(location2,Cables,67,89,90,,,)
(location2,Extensions,28,89,400,,,)
(location2,Modules,23,80,45,,,)
(location2,Switches,78,90,100,,,)
grunt>
```

```
(location3,Hammer,40,89,160)
(location3,Screwdriver,23,89,60)
(location3,Plier,23,90,300)
(location3,Screw,60,89,180)
(location3,Cables,60,89,140)
(location3,Extensions,39,80,70)
(location3,Modules,40,89,90)
(location3,Switches,40,69,80)
```

4. **Finding the most liked product at each location:** In this the most liked product is displayed in overall locations. By running the commands you will get the most liked product at each location.

```
(Plier,590)
(Screw,680)
(Cables,310)
(Hammer,350)
(Modules,224)
(Switches,320)
(Extensions,560)
(Screwdriver,270)
grunt> █
```


5. FINDING THE MOST LIKED PRODUCT AT DIFFERENT LOCATIONS:

Here the most liked product at different locations that is location 1, location 2 and location 3.

```
ted. Instead, use dfs.bytes-per-checksum
2025-04-01 07:54:08,486 [main] WARN  org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2025-04-01 07:54:08,490 [main] INFO  org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input files to process : 3
2025-04-01 07:54:08,490 [main] INFO  org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 3
(location2,Hammer,60,89,90,,,)
(location2,Screwdriver,30,78,10,,,)
(location2,Plier,78,90,170,,,)
(location2,Screw,45,90,200,,,)
(location2,Cables,67,89,90,,,)
(location2,Extensions,28,89,400,,,)
(location2,Modules,23,80,45,,,)
(location2,Switches,78,90,100,,,)
(location3,Hammer,40,89,160)
(location3,Screwdriver,23,89,60)
(location3,Plier,23,90,300)
(location3,Screw,60,89,180)
(location3,Cables,60,89,140)
(location3,Extensions,39,80,70)
(location3,Modules,40,89,90)
(location3,Switches,40,69,80)
(location1,Hammer,50,70,100)
(location1,Screwdriver,20,30,200)
(location1,Plier,50,89,120)
(location1,Screw,23,90,300)
(location1,Cables,40,50,80)
(location1,Extensions,30,39,90)
(location1,Modules,90,90,89)
(location1,Switches,80,90,140)
grunt> █
```


6. Finding products with decrementing sales: The product whose sales are going down is shown through this commands. The commands are as following :

```
ted. Instead, use dfs.bytes-per-checksum
2025-04-01 07:58:15,647 [main] WARN org.apache.pig.data.SchemaTupleBackend - SchemaTupleBackend has already been initialized
2025-04-01 07:58:15,651 [main] INFO org.apache.hadoop.mapreduce.lib.input.FileInputFormat - Total input files to process : 1
2025-04-01 07:58:15,651 [main] INFO org.apache.pig.backend.hadoop.executionengine.util.MapRedUtil - Total input paths to process : 1
(location1,Modules,180)
(location1,Switches,170)
(location1,Plier,139)
(location1,Hammer,120)
(location1,Screw,113)
(location1,Cables,90)
(location1,Extensions,69)
(location1,Screwdriver,50)
(location2,Screw,45,90,200,)
(location2,Screwdriver,30,78,10,)
(location2,Extensions,28,89,400,)
(location2,Switches,78,90,100,)
(location2,Modules,23,80,45,)
(location2,Plier,78,90,170,)
(location2,Hammer,60,89,90,)
(location2,Cables,67,89,90,)
(location3,Screw,149)
(location3,Cables,149)
(location3,Hammer,129)
(location3,Modules,129)
(location3,Extensions,119)
(location3,Plier,113)
(location3,Screwdriver,112)
(location3,Switches,109)
grunt> █
```

KEY LEARNINGS AND OBSERVATIONS

The following key learnings and observations occurred throughout the project.

1. **Efficient Data Processing with Apache Pig:** Implementing Apache Pig for sales analysis provided a deep understanding of how to handle and process large datasets efficiently using a high-level scripting language over Hadoop. The ability to work with structured and semi-structured data without writing complex MapReduce code made data transformation and analysis more streamlined.
2. **Importance of Data Preprocessing:** The project highlighted the importance of data preprocessing, as raw sales data from different locations needed to be cleaned and structured properly. Handling missing values, ensuring data consistency, and dealing with different formats were crucial steps before performing meaningful analysis.
3. **Effective Data Manipulation with Pig Latin:** The use of Pig Latin commands for data manipulation, such as filtering, grouping, and joining, allowed efficient extraction of key business insights. The ability to process large-scale retail data across multiple files helped in identifying important sales trends and customer preferences at different locations.
4. **Understanding Regional Sales Patterns:** Analyzing sales data across multiple locations demonstrated how offline and online sales patterns differ regionally. Some products showed high online demand but lower offline sales, whereas others performed better in physical stores, highlighting the impact of local customer behavior and preferences.
5. **Identification of Popular Products and Trends:** Identifying the most sold products and the most liked products helped in understanding market demand and consumer interests. The analysis also revealed seasonal trends, where certain products had fluctuating sales and wishlist counts, emphasizing the need for dynamic inventory management.
6. **Comparing Pig with Other Big Data Tools:** The project showcased the advantages of Apache Pig in handling large datasets with ease, but also highlighted some performance limitations when compared to more optimized frameworks like Apache Spark. While Pig simplifies data transformation, it may not be the best choice for real-time processing.
7. **Detecting Declining Sales and Optimization Needs:** Detecting declining sales across different locations helped in identifying underperforming products and areas where marketing strategies or pricing adjustments could be improved. This insight is crucial for businesses to optimize their sales strategies and improve overall revenue.
8. **Impact of Demographics on Sales:** The project reinforced the importance of demographic factors in sales analysis. Different regions exhibited varied preferences, showing that customer behavior is highly influenced by location, economic conditions, and digital adoption trends.

APPLICATION IN REAL-WORLD

This project has direct applications in various real-world situations, e.g.

1. **Retail Sales Optimization:** Businesses can use sales analysis to understand which products perform well in different locations, allowing them to optimize stock levels, pricing strategies, and promotional campaigns to maximize sales and customer satisfaction.
2. **E-commerce Personalization:** Online platforms can leverage the insights from wish list and sales data to recommend products based on customer preferences, improving personalized marketing efforts and increasing conversion rates.
3. **Supply Chain and Inventory Management:** By identifying high-demand and low-demand products across different regions, companies can adjust inventory distribution to ensure optimal stock availability, reducing overstocking and stockouts.
4. **Marketing Strategy Enhancement:** Retailers can analyze offline and online sales trends to design location-specific marketing campaigns, promotional discounts, and targeted advertisements to boost engagement and sales.
5. **Consumer Behavior Analysis:** Understanding which products are most liked or frequently wish listed helps businesses gain insights into consumer interests, allowing them to develop new products or modify existing ones to align with market demand.

CONCLUSION

The "Sales Analysis Through Retail Demographics" project using Apache Pig provided valuable insights into how big data analytics can enhance decision-making in the retail industry. By leveraging the capabilities of Pig on Hadoop, we efficiently processed large-scale sales data from multiple locations, identifying key patterns and trends in offline and online sales. The project highlighted the significance of data preprocessing, structured querying, and analytical operations in deriving meaningful business insights, such as the most sold products, most liked items, and declining sales across different regions.

One of the major takeaways from this project was the understanding of regional sales variations and consumer preferences. The analysis revealed that sales performance is influenced by multiple factors, including location, customer demographics, and shopping preferences. By identifying top-performing and underperforming products, businesses can make data-driven decisions regarding inventory management, marketing strategies, and sales optimization. Furthermore, the ability to track Wishlist data helped in forecasting demand and improving product recommendations, which is crucial for enhancing customer experience in e-commerce platforms.

Overall, this project demonstrated the power of big data technologies like Apache Pig in handling and analyzing vast amounts of retail data efficiently. The insights gained from sales analysis can be applied in real-world scenarios such as personalized marketing, supply chain optimization, and business expansion strategies. While Apache Pig proved to be a useful tool for batch processing, further improvements could be made by integrating advanced analytics tools like Apache Spark for real-time processing. This project reinforced the importance of data-driven decision-making in modern retail, showcasing how businesses can gain a competitive edge through effective sales analysis and customer-centric strategies.

PLAGIARISM REPORT

The screenshot shows the SmallSEOTools Plagiarism Checker interface. The browser address bar displays `smallseotools.com/plagiarism-checker/`. The page features a navigation bar with links to Pricing, Plagiarism Checker, Free Grammar Checker, Reverse Image Search, Paraphrasing Tool, Login, and a language selector (EN). The main heading is "Plagiarism Checker".

The interface is divided into two main sections. The left section, titled "Original Text", contains the text being checked: "across multiple files helped in identifying important sales trends and customer preferences at different locations." Below this text, it shows "Words 827" and "Characters 6136". A "Download Report" button is visible. The right section displays the results: "0% Plagiarized Content" and "100% Unique Content". It also shows "0% Exact Plagiarized" and "0% Partial Plagiarized". A "Give Feedback" button is present. Below the results, there is an illustration of a person at a computer and the text "Congratulation! No Plagiarism Found".

A green banner at the bottom states: "Your Personal Information is 100% secured with us. We do not save/share your data with any third party."

The screenshot shows the SmallSEOTools AI Content Detector interface. The browser address bar displays `smallseotools.com/ai-content-detector/`. The page features a navigation bar with links to Other Tools, Plagiarism Checker, Grammar Checker, Paraphrasing Tool, and ChatGPT Detector. The main heading is "AI Content Detector".

The interface is divided into two main sections. The left section, titled "Other Tools", contains the text being checked: "KEY LEARNINGS AND OBSERVATIONS The following key learnings and observations occurred throughout the project." Below this text, it shows "Up to 2000 Words will be used." and "Words Limit/Search: 827/2000". A "Upload" button is visible. The right section displays the results: "Likely To Be Human Written Text". It shows "0.02% AI Written Content" and "99.98% Human Written Content". A "Feedback" button is present. Below the results, there is an illustration of a person at a computer and the text "Feel free to drop us your feedback."