AN ALTERNATE MEDICINE RECOMMENDATION SYSTEM USING NLP FOR OPTIMIZING PHARMACEUTICAL CARE

Submitted By

Group 11

Ambily Treesa Varghese – W0826454
Dileep Sathyan – W0826453
Joel Soloman Raj Addala - W0823570
Ram Sundar Thanumalaya Perumal - W0804423
Ranisha Rajagopalan Girija - W0826459

DAB304-HEALTHCARE ANALYTICS Winter 2024 - 003

Table of Contents

Abstract

The Indian pharmaceutical market is characterized by a vast array of medications, presenting challenges in selecting the most appropriate alternative drugs for patients and healthcare providers. To address this, a Medicine Recommendation System aimed at streamlining the decision-making process was developed. Utilizing a comprehensive dataset from Kaggle featuring over 250,000 records, the system employs advanced natural language processing (NLP) techniques to analyze medication descriptions and compositions. The core of the recommendation engine is based on the Cosine Similarity algorithm, which assesses similarities between medicines to suggest viable alternatives effectively. This system not only enhances the accessibility and efficiency of medicine selection but also supports end users by providing a user-friendly interface for easy navigation. This is a significant step towards optimizing pharmaceutical care and improving health outcomes by leveraging technology to bridge the gap in medicine information accessibility.

Introduction

Navigating the Indian pharmaceutical landscape can be daunting due to its extensive variety of medications, each with different compositions and dosages. This complexity not only hampers the efficiency of healthcare delivery but also impacts patient care. Recognizing the need for a solution that could simplify the process of alternative medicine selection, the project introduces the Medicine Recommendation System, a sophisticated tool designed to assist in the accurate and efficient selection of alternate medications.

The system is engineered to address the critical gaps in knowledge and accessibility that both healthcare providers and patients face when choosing medicines. By integrating data science and machine learning techniques, specifically natural language processing (NLP), the system analyzes extensive pharmaceutical data to identify and recommend medication alternatives that are most relevant to the users' needs. The project harnesses a robust dataset that reflects the diverse range of medications available in the Indian market, ensuring the recommendations are well-founded and comprehensive.

The Medicine Recommendation System is not just a technological solution; it is a step towards transforming how healthcare providers and patients interact with the pharmaceutical industry. The reliable and easy-to-use platform is aimed to empower users, enhance the decision-making process, and ultimately improve health outcomes through better-informed medicine choices.

Related Work

The field of pharmaceutical informatics has seen significant advancements over the years, primarily driven by the need to improve drug discovery, patient care, and medication management. Several studies and systems have laid the groundwork for the development of decision-support systems in medicine. Below, are some key related works that have influenced the project or tackled similar challenges:

Clinical Decision Support Systems (CDSS):
These systems provide evidence-based recommendations to healthcare providers during patient care. Kawamoto et al. (2005) demonstrated the effectiveness of CDSS in improving clinical practice and patient outcomes.

Pharmaceutical Data Mining:
Han et al. (2012) used association rule mining to identify frequent drug combinations that may lead to adverse effects, enhancing safer prescription practices.

Medication Therapy Management (MTM) Programs:
Bluml (2005) emphasized the role of informatics in optimizing drug therapy and improving outcomes for patients through personalized interventions and counseling.

Drug Recommendation System:
Zhang et al. (2017) developed machine learning-based systems to predict patient responses to different drugs, enabling personalized medicine approaches.

NLP in Medicine Labeling and Information Retrieval:
Xu et al. (2018) demonstrated how NLP can automate the extraction of drug-related information from unstructured medical texts, such as drug labels and clinical notes.

Methods

The dataset used in this project, sourced from Kaggle, is one of the largest collections of pharmaceutical data specific to the Indian market.

It features 8 attributes including medicine name, manufacturer, price, and composition.

- 253,973 Records
- 249,398 Unique name medications

- 7,648 Unique Manufacturers
- Over a quarter-million records, providing a broad base for analysis and model training.
- Unique Identifiers for each medicine, ensuring accurate recommendation outputs.

The project's methodology is structured into several critical components:

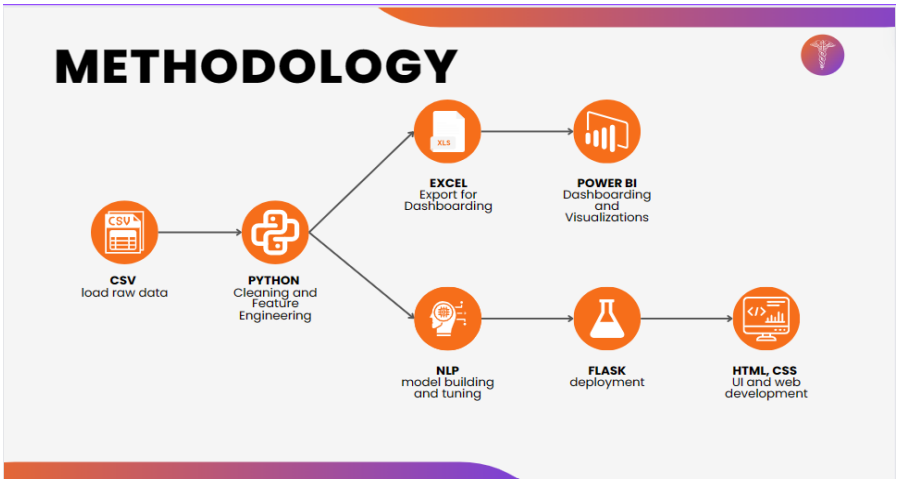

Figure 1: Methodology.

Data Preprocessing: Using Python, and Excel for data cleaning, handling missing values, and preparing the dataset for analysis.

Model Building: Leveraging natural language processing (NLP) techniques to interpret medicine descriptions and compositions.

Model Deployment: Utilizing Flask to create a web-based application that allows real-time medicine recommendations.

Data Visualization: Developing interactive dashboards with Power BI to visualize data trends and insights.

Exploratory Data Analysis (EDA)

Detailed EDA was conducted to uncover patterns and insights such as:

Price Trends: Analysis of medicine prices across different categories and manufacturers.

Manufacturer Analysis: Evaluation of the market share and reputation of pharmaceutical companies based on the dataset.

Top Medicines by Cost: Identified the most expensive drugs, such as 'Tecentriq 1200mg Injection', spotlighting the premium segment in the market.

Pricing Overview: Presented a range of medicine prices with an average price point of 265.00 INR, highlighting the economic variability in drug costs.

Medicine Type Proportion: Demonstrated that tablets are the most common medication type, followed by injections and capsules, reflecting prescribing and consumption trends.

Price and Dosage Distribution: Revealed that certain forms of medication, such as 'pen' systems, are higher in price, and common dosages like 100mg are more prevalent, guiding potential areas for cost optimization and inventory focus.

The Power BI dashboard effectively distills complex data into clear, actionable insights, driving informed decision-making in the pharmaceutical space.

Medicine Recommendation Model

The recommendation engine was designed with a focus on accuracy and relevance:

Feature Extraction: Identification and extraction of key features from the dataset based on the therapeutic use and chemical properties of medications.

Similarity Scoring: Implementation of the Cosine Similarity algorithm to evaluate the closeness between medicines based on their features.

Weight Assignment: Assigning appropriate weights to each feature is a critical aspect of the Weighted Similarity method. The weights reflect the significance of each attribute in determining the similarity between medicines. For example, the composition carries more weight than other attributes.

Recommendation Generation: Using the assigned weights, we calculate the similarity scores between the target medicine and the other medicines in the dataset. The medicines with the highest similarity scores are then recommended as suitable alternatives. The recommendation code is detailed in Appendix B.

Results

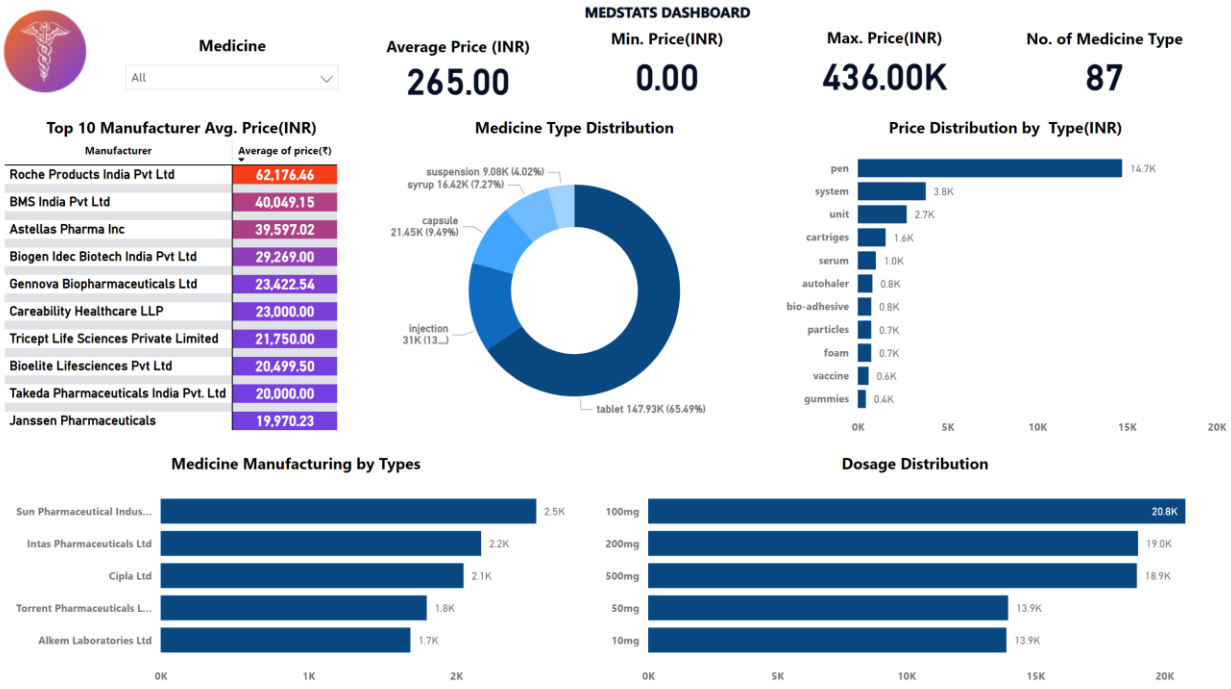

Figure 2: Medicine Distributions Dashboard

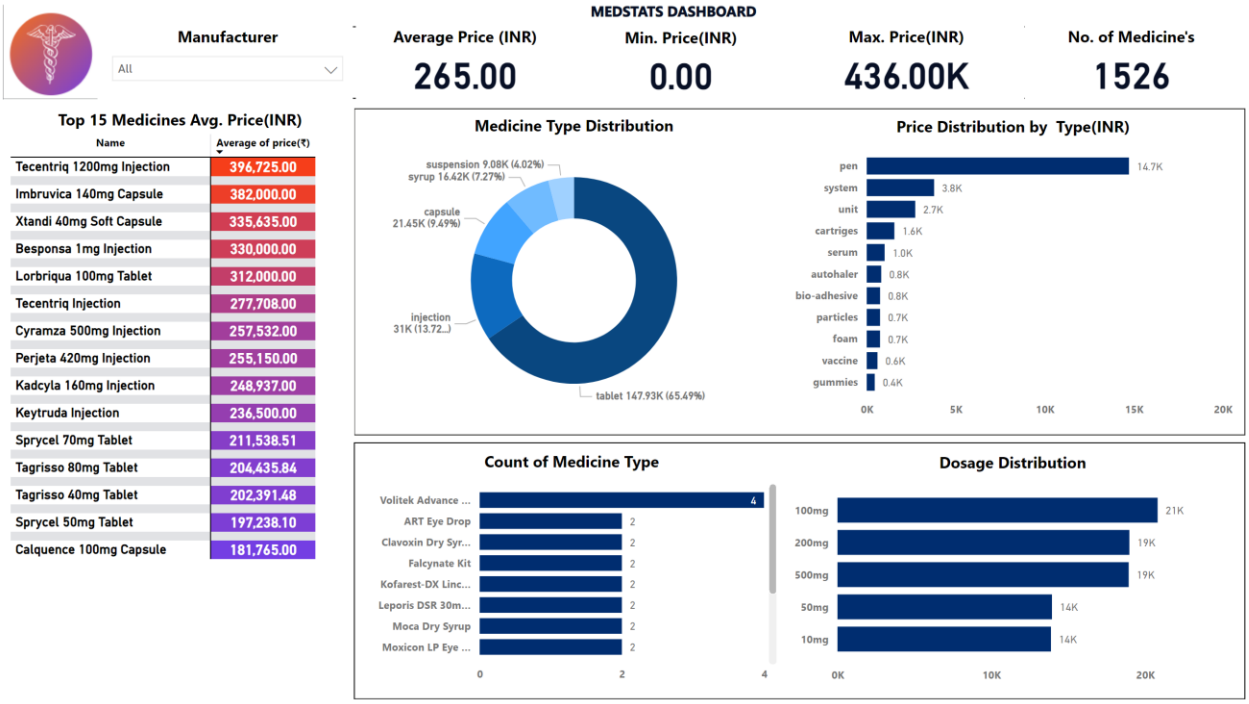

Figure 3: Manufacturer Distributions Dashboard

Figure 4: Medicine Recommendation System User Interface.

Discussion

The Medicine Recommendation System successfully streamlined medication selection within the complex Indian pharmaceutical market using advanced NLP techniques and the Cosine Similarity algorithm. This achievement aligns with existing literature on decision-support systems for medicine selection.

Challenges arose in data preprocessing and system scalability due to the diverse nature of pharmaceutical data. However, strategic data cleaning and algorithmic optimizations enabled effective handling of these challenges. To further enhance the system's capabilities, several improvements are proposed.

Database Expansion: Incorporate international pharmaceutical data to broaden the recommendation scope.

Algorithm Optimization: Refine algorithms to improve speed and accuracy using advanced techniques like deep learning.

Interactive Features: Enable the users to personalize recommendations based on preferences or medical conditions.

User Experience Improvements: Continuously update the user interface for enhanced usability and visual appeal.

These enhancements aim to evolve the system into a more robust platform, providing accurate and personalized medication alternatives to healthcare providers and patients.

Conclusion

The Medicine Recommendation System has significantly advanced the management of pharmaceutical complexities in India, blending advanced natural language processing with machine learning to streamline medication selection. This system not only enhances decision-making for healthcare providers and patients but also promises great potential for expansion and refinement. Plans include incorporating broader datasets, updating algorithms, and potentially expanding to global markets. As the project evolves, it aims to transform medication prescription and management globally, optimizing therapeutic outcomes and elevating patient care.

Contributions

The success of the Medicine Recommendation System project is a direct result of the dedicated efforts and specific contributions of each team member:

Ambily Treesa Varghese:

Front-End Programming: Handled the front-end development, coding in HTML, and CSS, to bring the design to life responsively and interactively.

Data Visualization Integration: Integrated dynamic data visualizations within the UI, enabling real-time interaction with the system’s analytics for users.

Dileep Sathyan:

Engineered the backend logic for the recommendation algorithm, applying natural language processing techniques to analyze drug compositions.

Implemented the algorithm within the system, optimizing it for performance and accuracy.

Joel Soloman Raj Addala:

Spearheaded the development of the project's web application using Flask, integrating the recommendation engine with the front end.

Managed the deployment of the web application, ensuring robustness and scalability.

Ram Sundar Thanumalaya Perumal:

Assisted in the data curation process, and Model development.

Played a pivotal role in conducting the exploratory data analysis using Power BI, extracting meaningful insights on medicine distribution and pricing.

Ranisha Rajagopalan Girija:

Led the data curation process, ensuring the integrity and quality of the dataset used for the recommendation engine.

Oversaw the creation of interactive visualizations, including dashboard design and data presentation.

References

1. Kawamoto, K., Houlihan, C. A., Balas, E. A., & Lobach, D. F. (2005). Improving clinical practice using clinical decision support systems: a systematic review of trials to identify features critical to success. *BMJ (Clinical research ed.)*, *330*(7494), 765. https://doi.org/10.1136/bmj.38398.500764.8F
2. Han, J., Pei, J., & Kamber, M. (2012). Data mining: Concepts and techniques. Elsevier.
3. Bluml B. M. (2005). Definition of medication therapy management: development of professionwide consensus. *Journal of the American Pharmacists Association : JAPhA*, *45*(5), 566–572. https://doi.org/10.1331/1544345055001274
4. Zhang, P., Wang, F., Hu, J., & Sorrentino, R. (2014). Towards personalized medicine: leveraging patient similarity and drug similarity analytics. *AMIA Joint Summits on Translational Science proceedings. AMIA Joint Summits on Translational Science*, *2014*, 132–136. http://www.ncbi.nlm.nih.gov/pmc/articles/pmc4333693/
5. Xu, H., Stenner, S. P., Doan, S., Johnson, K. B., Waitman, L. R., & Denny, J. C. (2010). MedEx: a medication information extraction system for clinical narratives. *Journal of the American Medical Informatics Association : JAMIA*, *17*(1), 19–24. https://doi.org/10.1197/jamia.M3378
6. Singh, S. (2022) *A-Z medicine dataset of India*, *Kaggle*. Available at: https://www.kaggle.com/datasets/shudhanshusingh/az-medicine-dataset-of-india
7. Varghese, A., Sathyan, D., Addala, J., Perumal, R., & Girija, R. (2024). An Alternate Medicine Recommendation System Using NLP for Optimizing Pharmaceutical Care (Appendix B).

Appendices

Appendix A: Table of Figures

Appendix B: Source Code

model.py – NLP Recommendation code in Python

```
import pandas as pd
import numpy as np
from gensim.models import Word2Vec
from sklearn.metrics.pairwise import cosine_similarity
from tqdm import tqdm

weightage = {
    'price(₹)': 0.3,
    'manufacturer_name_embeddings': 0.1,
    'type_embeddings': 1.0,
    'primary_comp_embeddings': 0.8,
    'entire_comp_embeddings':1.0,
    'value_embeddings': 0.8,
    'entire_value_embeddings': 1.0
}

### Functions to Use
def getType(row):
    for pack_size in pack_sizes:
        if pack_size in row['pack_size_label'].lower():
            return pack_size

def getComp(row):
    return row['short_composition1'].split('(')[0].lower().rstrip()

def getValue(row):
    return row['short_composition1'].split('(')[1].lower().rstrip().rstrip(')')

def generate_word_embeddings(column, dataset):
    unique_values = dataset[column].unique()
    sentences = [[str(value)] for value in unique_values]
    model = Word2Vec(sentences, min_count=1, vector_size=100)
```

```
    return model

# Calculate similarity scores for 'price' column using cosine similarity
def calculate_price_similarity(price1, price2):
    price1 = np.array(price1).reshape(1, -1)
    price2 = np.array(price2).reshape(1, -1)
    similarity = cosine_similarity(price1, price2)
    return similarity[0][0]

def calculate_text_similarity(value1, value2):
    similarity = cosine_similarity(value1, value2)
    return similarity[0][0]

# Calculate weighted similarity between two medicines
def calculate_weighted_similarity(medicine1, medicine2):
    similarity_scores = []
    for column, weight in weightage.items():
        if column == 'price(₹)':
            similarity = calculate_price_similarity(medicine1[column].values[0],
medicine2[column])
        else:
            similarity = calculate_text_similarity(medicine1[column].values[0],
medicine2[column])
        similarity_scores.append(similarity * weight)
    weighted_similarity = sum(similarity_scores)
    return weighted_similarity

try:
    ### READ Data
    df = pd.read_csv(r'medicines_dataset.csv')

    # 'type' column has only 1 value and will not have an effect on the final model
    df=df.drop(columns=['type'])
    # 'Is_discontinued','id' can also be dropped
    df = df[df.Is_discontinued==False]
    df=df.drop(columns=['Is_discontinued','id'])

    # Exploring the 'pack_size_label' column
    form=[]
    count={}
    for i in df.pack_size_label:
        words=i.split()
        x=words[-1].lower()
        if len(x)<=2:
            x=words[-2].lower()
        if x in form:
            count[x]+=1
        else:
```

```
            count[x]=1
            form.append(x)

    sorted_count = dict(sorted(count.items(), key=lambda x:x[1], reverse=True))

    # Fix: removing the plural issue (eg: tablet | tablets)
    final_count=sorted_count.copy()
    for key,value in sorted_count.items():
        test=(key+'s') in final_count
        if test:
            final_count[key]+=final_count[key+'s']
            del final_count[key+'s']
            form.remove(key+'s')

    final_count = dict(sorted(final_count.items(), key=lambda x:x[1], reverse=True))

    # Extract unique pack sizes from 'pack_size_label' column
    pack_sizes = df['pack_size_label'].str.extract(r'(\b\w+\b)')[0].unique()


    df['type'] = df.apply(lambda row: getType(row), axis=1)
    df['primary_comp'] = df.apply(lambda row: getComp(row), axis=1)
    df['entire_comp'] = df['primary_comp'] + ' ' +
df['short_composition2'].astype(str).apply(lambda x: x.split('(')[0].lower().rstrip() if
pd.notna(x) else '')
    df['value'] = df.apply(lambda row: getValue(row), axis=1)
    df['entire_value'] = df['value'] + ' ' + df['short_composition2'].astype(str).apply(lambda x:
x.split('(')[1].lower().rstrip().rstrip(')') if pd.notna(x) and len(x.split('(')) > 1 else '')

    # Drop redundant columns
    dataset = df.drop(columns=['pack_size_label','short_composition1','short_composition2'])

    dataset['Disp'] = df.apply(lambda row: row['short_composition1'] if
pd.isnull(row['short_composition2']) else row['short_composition1'] + ' / ' +
row['short_composition2'], axis=1)

    # Define the text columns for which word embeddings need to be generated
    text_columns = ['manufacturer_name', 'type', 'primary_comp','entire_comp',
'value','entire_value']

    # Dictionary to store the models
    word_embedding_models = {}

    # Generate and store word embeddings for each text column
    for column in text_columns:
        model = generate_word_embeddings(column, dataset)
        word_embedding_models[column] = model
```

```
    # Apply word embeddings and create new columns for embeddings
    for column, model in tqdm(word_embedding_models.items(), desc="embeddings"):
        new_column_name = column + '_embeddings'
        dataset[new_column_name] = dataset[column].apply(lambda x: model.wv[str(x)].reshape(1, -
1) if str(x) in model.wv else [])

    print("<<<-- SUCCESS -->>>")

except:
    print(" \(>.<)/ FAILURE - Error Occured")
```