CMPS451: Data Mining, Big Data and Data Analytics

Project Report

Team: 16

Team Members

Mai Abdelhameed	1190365	maiabdelhameed16@gmail.com
Mark Milad	1190518	
Mohamed Mohsen	1190277	mohamedmohsen96661@gmail.com
Malak Hamed	1190357	

I. Brief Problem Description

Our project aims to explore patterns and trends in mental health based on a comprehensive dataset. Mental health is a critical aspect of overall societal well-being, yet it remains a complex and sensitive issue. By analyzing this dataset, we aim to uncover insights that can inform interventions, policies, and support systems for individuals struggling with mental health issues.

The dataset comprises data on various aspects, such as stress levels, depression, anxiety, subjective well- being, and the utilization of mental health services. Respondents from different demographic backgrounds, including gender, employment status, and geographic region, participated in the survey. The dataset's purpose is to provide insights into global mental health trends during the specified period.

Hence, we are planning to implement a model that predicts whether the user should seek treatment or not, based on the factors surrounding the individual.

II. Pipeline

Our pipeline consists of the following modules:

1. **Data Collection:** The dataset was obtained from Kaggle containing information about individuals' demographics, mental health history, and their treatment-seeking behavior.

2. **Data Preprocessing:** The dataset underwent cleaning procedures to handle missing values, duplicates, and irrelevant columns. Categorical variables were encoded for modeling.

3. **Exploratory Data Analysis (EDA):** Various visualizations and statistical analyses were performed to gain insights into the relationships between different features and the target variable.

4. **Feature Selection:** To enhance the effectiveness of our analysis, we will identify relevant features that capture meaningful aspects of mental health. Furthermore, we will perform appropriate encoding technique to present data in a familiar format to the model.

5. **Model Training:** Several machine learning models were trained on the preprocessed data to predict treatment-seeking behavior based on individual characteristics.

6. **Model Evaluation:** The trained models were evaluated on both training and test datasets to assess their performance in terms of accuracy and other relevant metrics.

III. Analysis and Solution of the Problem

1) Data Preprocessing

Initial exploratory data analysis (EDA) is conducted to gain insights into the dataset's structure, including checking for null values, duplicates, and understanding the distribution of categorical variables. Hence, Missing values and duplicates were removed from the dataset.

```
before removing nulls:
Timestamp                      0
Gender                         0
Country                        0
Occupation                     0
self_employed               5202
family_history                 0
treatment                      0
Days_Indoors                   0
Growing_Stress                 0
Changes_Habits                 0
Mental_Health_History          0
Mood_Swings                    0
Coping_Struggles               0
Work_Interest                  0
Social_Weakness                0
mental_health_interview        0
care_options                   0
dtype: int64
```

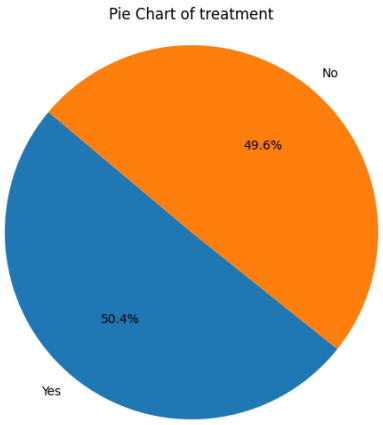
```
after removing nulls:
Timestamp                   0
Gender                      0
Country                     0
Occupation                  0
self_employed               0
family_history              0
treatment                   0
Days_Indoors                0
Growing_Stress              0
Changes_Habits              0
Mental_Health_History       0
Mood_Swings                 0
Coping_Struggles            0
Work_Interest               0
Social_Weakness             0
mental_health_interview     0
care_options                0
dtype: int64
```

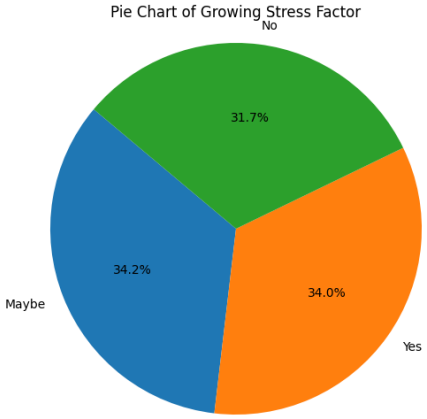
```
before removing duplicates :  (287162, 17)
after removing duplicates :  (286808, 17)
```

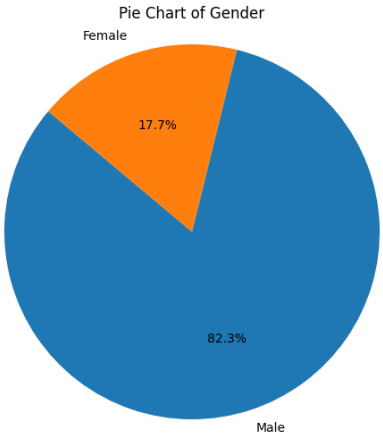
```
distribution of data
                  Timestamp  Gender        Country Occupation self_employed  \
count                286808  286808         286808     286808        286808
unique                  721       2             35          5             2
top     2014-08-27 12:54:11    Male  United States  Housewife            No
freq                    780  235950         167819      65173        257661

       family_history treatment Days_Indoors Growing_Stress Changes_Habits  \
count          286808    286808       286808         286808          286808
unique              2         2            5              3               3
top                No       Yes     1-14 days          Maybe             Yes
freq           173527    144501        62429          98225          107579

       Mental_Health_History Mood_Swings Coping_Struggles Work_Interest  \
count                 286808      286808            286808        286808
unique                     3           3                 2             3
top                       No      Medium                No            No
freq                  102179       99272            151373        103846

       Social_Weakness mental_health_interview care_options
count           286808                 286808       286808
unique               3                      3            3
top              Maybe                     No           No
freq            101441                 228986       116403
```

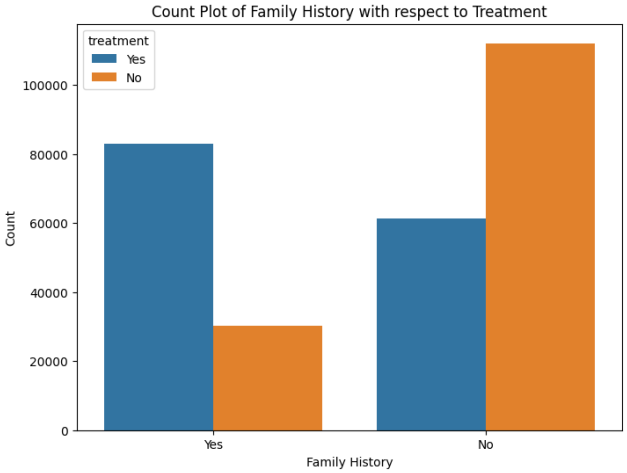
2) Visualization

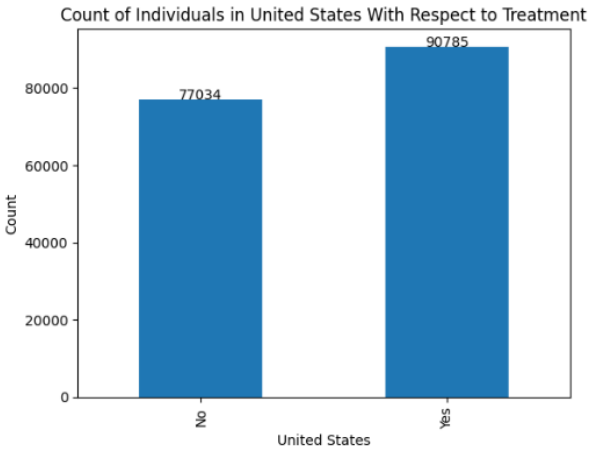
Visualizations such as pie charts, count plots, and bar plots were used to explore the distributions of various features and their relationships with the target variable.

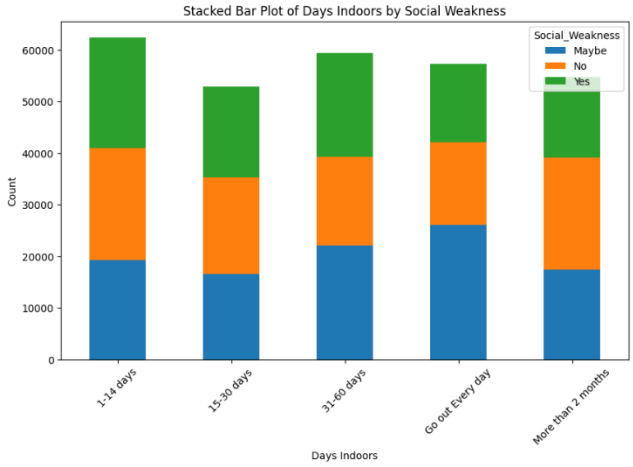












3) Extracting Insights from Data

Insights regarding the distribution of treatment-seeking behavior across different demographic and psychographic categories were extracted.

➢ *Insights drawn from dataset*

- USA is the most country that data was collected from followed by the United Kingdom (might be due to population or mental health awareness) (59%)
- In the mental_health_interview column 79% of the individuals answered no
- The individuals that are not self-employed are far more than those who are (88%)
- Most of the individuals in the dataset are Males (82%).
- Females tend to seek treatment more than Males.
- The rest of the columns have fairly distributed categories
- Treatment's values are balanced, which means that we don't need to perform any dataset balancing methods to handle unbalancing classes.
- Housewives are the most category in Occupation to receive treatment.
- Students are the second most category in Occupation to receive treatment
- Mood swings are somewhat correlated with Changes Habits, but the High column of Mood Swings the Maybe category of Changes Habits is the highest. Moreover, this can show that Mood swings and changing habits are somewhat correlated but changing habits is not the decisive factor in this
- Count of people who have mental health history are less likely to have family history, which was logically unexpected from the dataset.

4) Feature Selection

Certain columns deemed unnecessary for analysis and hence, they were dropped from the dataset, such as Timestamp, Country, Self Employed, Work Interest, Mental Health Interview, Days Indoors, Occupation, Social Weakness, and Changes Habits.

Categorical features are encoded using either binary encoding for 2 values columns or one-hot encoding for columns with more than 2 values; for compatibility with machine learning algorithms.

5) Model/Classifier Training

- Training Phase

Feature-Specific Mapper and Reducer (mapper, reducer)

A mapper function (mapper) is defined to map each row of the dataset to key-value pairs. Each key-value pair consists of a tuple containing the index, feature value, and the target label (0 or 1), along with a count initialized to 1.
A reducer function (reducer) is defined to aggregate the mapped values. It flattens the list of key-value pairs and updates a global dictionary (counts) with the counts of each unique key-value pair.

MapReduce Execution (map_worker, reduce_worker)

The MapReduce process is executed using threads. map_worker and reduce_worker functions are defined to handle mapping and reducing tasks respectively. These functions are executed concurrently using multiple threads to speed up the process.
In the mapping phase, the dataset is divided into slices, and each slice is processed by a separate thread. Each thread calls the map_worker function to map the data slice and update the result list.

In the reducing phase, the mapped results are aggregated. Similar to the mapping phase, multiple threads are used to concurrently execute the reduce_worker function.

Label-specific Mapper and Reducer (mapper_label, reducer_label)

Additional mapper (mapper_label) and reducer (reducer_label) functions are defined to calculate counts specific to each label (0 or 1). This is done separately to facilitate conditional probability calculation during testing.

- Testing Phase:

Mapper and Reducer Function (mapper_test,reducer_test)

A mapper function (mapper_test) is defined to calculate conditional probabilities for each feature given the target label during testing. It computes the probability of each feature value given both label 0 and label 1.
A reducer function (reducer_test) is defined to calculate the overall probability for each label given the feature values. It computes the probability for each label based on the conditional probabilities calculated in the mapping phase.
MapReduce Execution for Testing:

Similar to the training phase, the testing phase involves executing the MapReduce process using threads. Functions map_worker_test and reduce_worker_test handle mapping and reducing tasks respectively.
The process is similar to the training phase, with separate threads for mapping and reducing, but using the testing data and the conditional probabilities calculated during training.

- Inference Phase:

Mapper Function (mapper_infer):

This function takes a row of data as input and calculates the conditional probabilities for each feature given the target variable.
It iterates over each element in the row, calculates the conditional probabilities for each class of the target variable (0 and 1), and appends them to a list.
The conditional probabilities are calculated based on precomputed counts stored in a dictionary (counts) and the total counts of each class of the target variable (counts_label).

Reducer Function (reducer_infer):

This function takes the mapped values (conditional probabilities) from multiple rows and combines them to make predictions.
It iterates over the mapped values, multiplying the probabilities for each feature given the target variable class (0 and 1).
After multiplying the probabilities, it adjusts them based on the prior probabilities of each target variable class and divides by the total counts of each class.
Finally, it compares the probabilities for each class and assigns the class with the higher probability as the prediction.

Overall Naive Bayes Implementation:

The Naive_Bayes function serves as the entry point for both training and testing. It controls the flow of execution based on the Train parameter. During training, it performs MapReduce to calculate counts and conditional probabilities. During testing, it uses the trained model to make predictions and evaluates the performance.

This implementation of Naive Bayes classifier using MapReduce is tailored for large-scale datasets where parallel processing can significantly speed up the computation. It's a scalable approach that can handle big data efficiently. Random Forest and Logistic Regression were also trained on the preprocessed data.

IV. Results and Evaluation

After obtaining predictions, the code calculates evaluation metrics using accuracy_score, precision_score, recall_score, f1_score, and confusion_matrix functions from sklearn.metrics.

Accuracy scores for each model on the test dataset were as follows:

- Naive Bayes:

```
Naive Bayes Results:
accuracy =  0.5102332554652906
precision = 0.5073483137748491
recall =  0.9488368872888396
f1 score = 0.6611669923052801
confusion matrix =  [[  929 13308]
 [  739 13705]]
```

- Random Forest:

```
Result for Random Forest:
Accuracy: 0.7110630731146055
precision = 0.6900425952219272
recall =  0.7738853503184714
f1 score = 0.7295630323401756
confusion matrix =  [[ 9216  5021]
 [ 3266 11178]]
```

- Logistic Regression:

```
Result for Logistic Regression:
Accuracy: 0.7113420034168962
precision = 0.6903835464146748
recall =  0.7738853503184714
f1 score = 0.7297535498612698
confusion matrix =  [[ 9224  5013]
 [ 3266 11178]]
```

There are many metrics to consider, but in our case, we have balanced consideration: both false positives and false negatives are of similar concern, we have to opt for a metric that balances both types of errors. F1 score is a commonly used metric that combines precision and recall. It provides a single score that balances the trade-off between false positives and false negatives. Additionally, accuracy can be considered since the classes (target) in the dataset are well-balanced.

V. Unsuccessful Trials That Were Not Included in the Final Solution

One of the attempts that failed was training a KNN model on this data.

When dealing with a large dataset with multiple categorical columns, the performance of K-Nearest Neighbors (KNN) algorithm can be challenging due to the categorical nature of the features.

As the number of categorical columns increases, the dimensionality of the feature space grows, leading to the curse of dimensionality.

High-dimensional spaces can result in increased computational complexity and can negatively impact the performance of KNN, as distances become less meaningful in higher dimensions. This was very clear in the training process when it took more than 5 mins to train KNN on our dataset.

Also, categorical data often results in sparse feature representations, especially when using one-hot encoding.

In high-dimensional spaces, the majority of data points may be far apart from each other, making it difficult to find meaningful neighbors and increasing the likelihood of misclassification.

VI. Enhancements and Future Work

- ✓ Feature engineering: Experimenting with different feature combinations or transformations to capture more nuanced patterns in the data.
- ✓ Ensemble methods: Exploring ensemble methods to combine predictions from multiple models for improved accuracy.
- ✓ Deployment: Implementing the trained model in a real-world application for automated decision-making or support system.