MAKERERE UNIVERSITY

SEMESTER ONE 2024/2025 ACADEMIC YEAR

COLLEGE OF COMPUTING AND INFORMATION SCIENCES

SCHOOL OF COMPUTING AND INFORMATICS TECHNOLOGY

DEPARTMENT OF COMPUTER SCIENCE

MASTER OF SCIENCE IN COMPUTER SCIENCE

COURSE NAME: STRUCTURE AND INTERPRETATION OF COMPUTER PROGRAMS

COURSE CODE: MCN 7105

LECTURER: DR MARRIETTE KATARAHWEIRE

DATE: Monday, 8th Dec 2024

SEMESTER PROJECT 2

STUDENT DETAILS

NAME	STUDENT NUMBER	REG NUMBER
ABILA Raphael	2400721912	2024/HD05/21912U

Question one

A Detailed Overview of Data Science Abstractions

Introduction

In data science, we often deal with complex processes that require breaking down tasks into simpler, more manageable steps. These steps are organized into different "layers of abstraction," which make it easier to work with and understand the data. In this report, we'll dive into how abstraction works in five key components of a typical data science workflow:

1. **Linear Model**
2. **List → Sentiment**
3. **Read CSV**
4. **QQ-Plot**
5. **Histogram**

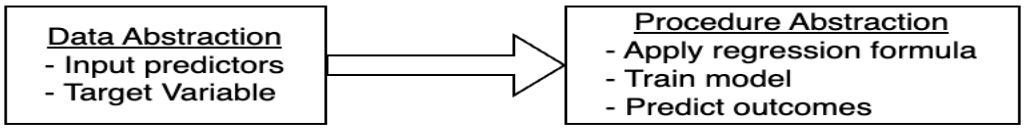
Each of these components has different layers of abstraction that help streamline data handling and analysis. We will look at each one closely and include diagrams to visually explain these abstractions.

1. Linear Model Abstraction

A linear model is a method used in statistics to predict outcomes based on input data. Here's how the abstraction works in different layers:

- **Data Layer**: The variables involved are broken down into two groups – independent variables (X) and dependent variables (Y).
- **Algorithm Layer**: This involves the mathematical process of fitting a line to the data, estimating coefficients, and understanding relationships between variables.
- **Output Layer**: After the model is built, it provides useful results like the coefficients (which show how much each variable contributes to the prediction), the intercept, and evaluation metrics that tell you how well the model is performing.

Diagram: Linear Model Abstraction



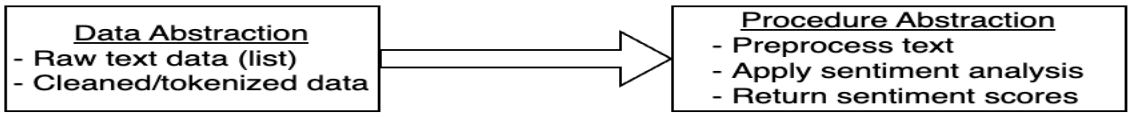
2. List → Sentiment Abstraction

This part takes text data (like a list of words or phrases) and determines the sentiment behind it. Here's the abstraction breakdown:

- **Data Layer**: The input consists of raw text or tokenized words.

- **Processing Layer**: This is where the magic happens. Sentiment analysis models or lexicons analyze the text to determine if the sentiment is positive, negative, or neutral.
- **Output Layer**: The final output is usually a numeric sentiment score or a label (such as "positive," "neutral," or "negative").

Diagram: List → Sentiment Abstraction

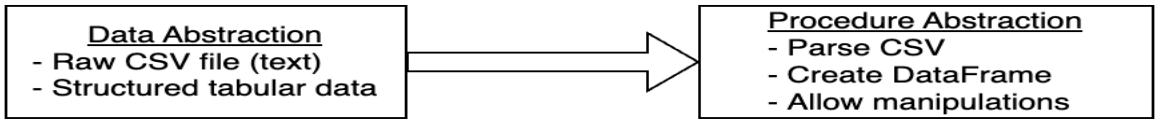


3. Read CSV Abstraction

Reading data from a CSV file is a common operation. The abstraction here includes:

- **Data Layer**: The file you're working with, including its structure and metadata.
- **Procedure Layer**: This involves reading the rows and columns from the file and handling any issues, like missing values, to make sure the data is clean.
- **Output Layer**: The result is a DataFrame (a table of data) that's ready to be analyzed further.

Diagram: Read CSV Abstraction

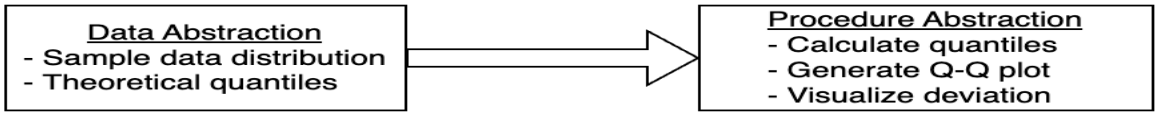


4. QQ-Plot Abstraction

A QQ-Plot is used to compare the distribution of your data to a theoretical distribution, often to check if your data follows a normal distribution. The abstraction process looks like this:

- **Data Layer**: The observed data and the theoretical distribution you're comparing it to.
- **Procedure Layer**: In this layer, the data is sorted, quantiles are calculated, and the plot is generated.
- **Output Layer**: The result is a QQ-plot, which helps you visually assess whether the data fits the expected distribution.

Diagram: QQ-Plot Abstraction

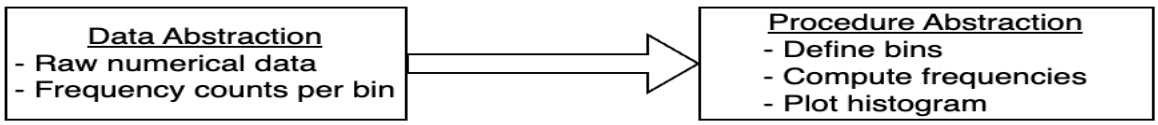


5. Histogram Abstraction

Histograms are used to show the distribution of data by grouping values into bins. Here's how the abstraction works:

- **Data Layer**: The raw values that will be grouped into bins.
- **Procedure Layer**: The data is divided into bins, and the number of values in each bin is counted. Then, the histogram is generated.
- **Output Layer**: The final output is a graphical representation of the data's distribution in the form of bars, showing the frequency of values in each bin.

Diagram: Histogram Abstraction



Question two

New Abstractions Developed

1. Load and Clean Dataset

Motivation: To facilitate smooth data processing, we abstracted the loading and cleaning phases into dedicated functions. This layer ensures data consistency and readability by preprocessing tweet text.

Design Decisions:

- Introduced modular cleaning (`clean-tweet`) to handle noise, including URLs, punctuation, and extra spaces.
- Built-in flexibility for future extensions to accommodate additional cleaning requirements (e.g., language detection).

2. Analyze Sentiment

Motivation: Sentiment analysis is the core of the project. We utilized the NRC lexicon for mapping tokens to emotions and sentiments.

Design Decisions:

- Modularized the process using the `analyze-sentiment` function.
- Designed for extensibility with support for integrating other lexicons or machine learning-based sentiment analyzers.

3. Summarize and Visualize Sentiment

Motivation: Clear visualizations enhance understanding of the mood trends. Aggregating sentiments and plotting histograms provide immediate insight into the mood distribution.

Design Decisions:

- `summarize-sentiment` aggregates sentiment into frequency counts for quick analysis.
- `visualize-sentiment` uses intuitive histograms to showcase data trends in a clean and interpretable manner.

4. Main Analysis Function

Motivation: A single entry point (`analyze-twitter-moods`) ties together all abstractions for streamlined execution and ensures that developers can integrate this workflow easily into larger systems.

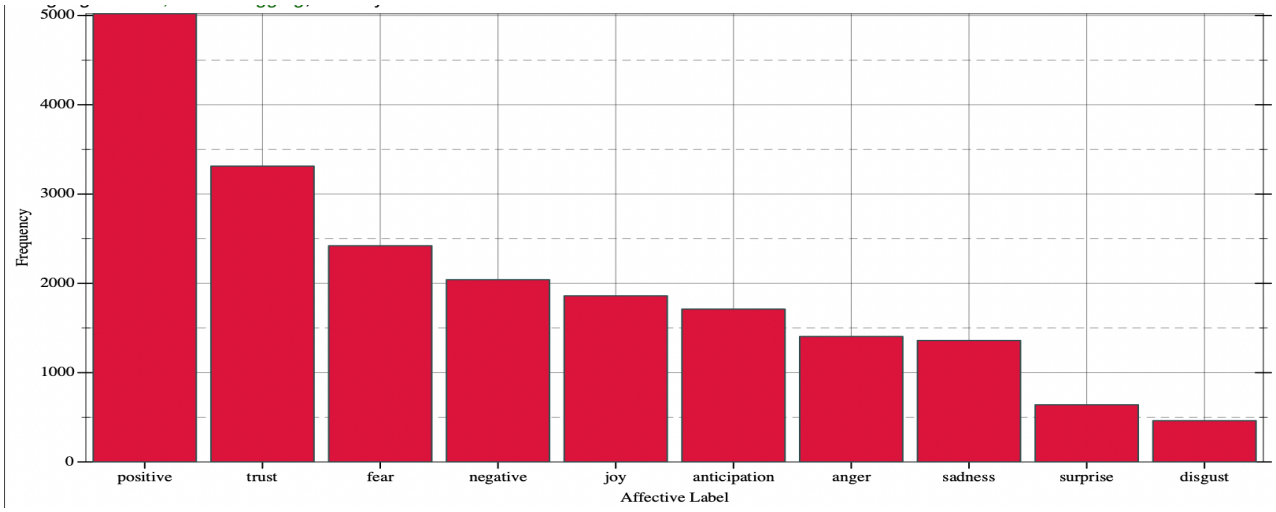
Design Decisions:

- Parameterized for flexibility, allowing different datasets to be analyzed.
- Modularized to facilitate debugging and testing individual components.

Levels of Abstraction

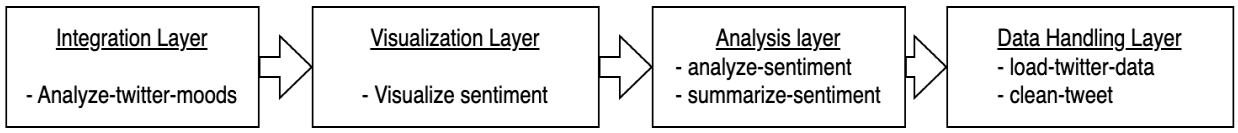
Level	Components	Purpose
Integration Layer	`analyze-twitter-moods`	Combines all layers to execute the end-to-end workflow for mood analysis.
Visualization Layer	`visualize-sentiment`	Generates clear visual representations (e.g., histograms) of sentiment trends.
Analysis Layer	`analyze-sentiment`, `summarize-sentiment`	Maps tweets to sentiments using a lexicon and aggregates results.
Data Handling Layer	`load-twitter-data`, `clean-tweet`	Handles data ingestion and cleaning to ensure high-quality inputs.

Sample Output (Histogram)



Visual Representation

The diagram below illustrates the hierarchy and data flow:



GitHub repository

https://github.com/RaphaelAbila/SICP-Project2