

# Data Analysis Project

*Ayush Kumar Mishra*

September 18, 2024

# Contents

# 1 Introduction

This document outlines the process of using ChatGPT to generate prompts for results and analysis of the project, extracted from the data.

## 1.1 Outline of the Document

- Prompts used for data analysis
- Prompts used for Streamlit Dashboard
- Prompts used for project results
- Prompts used for LaTeX writing

# 2 Data Overview

**Primary Prompt:** "What types of data were analyzed (emotion, gaze, and transcript data)? Can you explain the key features like emotion categories, gaze measurements, and transcript sentiments?"

**Follow-up Prompts:**

- "Can you explain the structure of the gaze data, such as blink and eye offset values?"
- "What are the basic statistics for the transcript data?"

# 3 Data Preprocessing

**Prompt:** "What preprocessing steps were applied to the data, including handling missing values, encoding categorical features, and scaling or normalization?"

## 4 Data Preparation and Integration

In this section, I explain how the data was prepared and important features were extracted from the three provided CSV files.

### 4.1 Extracting Key Features from Emotion Data from each CSV file and making single DATAFRAME

**Initial Prompt:**

```
movie_id image_seq  angry disgust  fear  happy sad  surprise neutral
dominant_emotion
0  93663f94-bf0a-4ce8-a29a-a5236cc7fe6a  0  4.31735  0.000594  2.879790
1.65035  2.779980  0.600814  87.77110 neutral
1  93663f94-bf0a-4ce8-a29a-a5236cc7fe6a  1  53.22530  2.981640  12.736800
1.52347  1.051320  27.216800  1.26462  angry

this is data of csv file of every emotion data and this is student 1.. there
are 10
such students, so 10 such df_emotion, so i want to make
combined df where two most frequent emotion data throughout
the video of the student reflects All df_emotion csv files are emotion_data
folder.
```

---

Chatgpt gave wrong response to the initial prompt. The response was giving wrong results.  
So i further explained what i intended to do

**Refined Prompt:**

You are doing it incorrectly. I want to first calculate the most frequent of every single emotion CSV and then make one column out of it.  
Then, finally make 10 columns out of 10 CSV files with column names dominant\\_top1, dominant\\_top2 emotions. From every emotion CSV, find 2 max frequency dominant features and then add that.

---

**For this prompt, response from the chatgpt was:**

```

import pandas as pd
import os

# Path to the folder containing emotion data
folder_path = 'emotion_data'

# Initialize an empty dictionary to store top 2 dominant emotions for each
student
dominant_emotions = {'student_top1': [], 'student_top2': []}

# Iterate over all files in the folder (assuming 10 files: emotion-df1.csv to
emotion-df10.csv)
for i in range(1, 11):
    file_name = f'{i}/emotion.csv'
    file_path = os.path.join(folder_path, file_name)

    # Read the CSV file into a DataFrame
    df = pd.read_csv(file_path)

    # Find the 2 most frequent dominant emotions
    top_emotions =
df['dominant_emotion'].value_counts().nlargest(2).index.tolist()

    # Append the top 2 emotions for this student (add NaN if there's less than
2 emotions)
    dominant_emotions['student_top1'].append(top_emotions[0] if
len(top_emotions) > 0 else None)
    dominant_emotions['student_top2'].append(top_emotions[1] if
len(top_emotions) > 1 else None)

# Create a DataFrame for top emotions for every student
df_dominantEmotion_for_everystudent = pd.DataFrame(dominant_emotions)

# Rename columns to reflect student numbers correctly (top1 and top2)
df_dominantEmotion_for_everystudent.columns = [
    f'student_{i}_dominant_emotion_top1' if 'top1' in col else
    f'student_{i}_dominant_emotion_top2'
    for i, col in enumerate(df_dominantEmotion_for_everystudent.columns,
start=1)
]

# Set index to range from 1 to 10
df_dominantEmotion_for_everystudent.index = range(1, 11)

# Print the final DataFrame
print(df_dominantEmotion_for_everystudent)

```

With the help of this code, two most freq dominant emotions were extracted from each emotion CSV file and added to the final dataframe.

Now the final dataframe has 10 columns with features top\_1\_dominant, top\_2\_dominant emotion.

	dominant_emotion_top1	dominant_emotion_top2
1	neutral	fear
2	happy	neutral
3	neutral	fear
4	neutral	fear
5	neutral	neutral
6	neutral	neutral
7	fear	sad
8	neutral	fear
9	neutral	happy
10	sad	fear

## 4.2 Integrating transcript data with emotion data

Prompt:

```
id      "neutral  confident  hesitant  concise  enthusiastic  speech_speed
0      0.267454   0.846701   0.845698  0.635805    0.647783     2.517986
1      0.260410   0.679283   0.733701  0.544145    0.417390     3.217822

this is data of csv file of every transcript data and this is student 1.. there
are 10
such students, so 10 such df_transcript , so i want to make
combined df where two most frequent text data throughout
the video of the student reflects All df_transcript files are transcript_data
folder.
```

The response from the chatgpt awesome this time.

```

# Path to the folder containing transcript data
folder_path = 'transcript_data'

# Initialize a list to store average DataFrames for each student
average_dfs = []

for f in os.listdir(folder_path):
    id = f.split('.')[0]
    file_path = os.path.join(folder_path, f)
    features = ['positive', 'negative', 'neutral', 'confident', 'hesitant',
                'concise', 'enthusiastic', 'speech_speed']
    df = pd.read_csv(file_path, usecols=features)

    # Calculate averages
    new_features = ['avg_positive', 'avg_negative', 'avg_neutral',
                    'avg_confident', 'avg_hesitant', 'avg_concise', 'avg_enthusiastic',
                    'avg_speech_speed']
    avg_values = df[features].mean().values

    # Create a new DataFrame with averages and id
    new_df = pd.DataFrame([avg_values], columns=new_features)
    new_df['id'] = id

    # Append to the list of average DataFrames
    average_dfs.append(new_df)

# Combine all average DataFrames into a single DataFrame
final_df = pd.concat(average_dfs, ignore_index=True)

print(final_df)

```

Using above code, It extracted the above features of transcript data of each student and added to the final dataframe.

	avg_positive	avg_negative	avg_neutral	avg_confident	avg_hesitant	avg_concise	avg_enthusiastic	avg_speech_speed
1	0.709199	0.141214	0.149586	0.733828	0.485172	0.429418	0.466497	3.113771
2	0.722006	0.107541	0.170453	0.684879	0.436158	0.484221	0.516685	3.269092
3	0.567257	0.264337	0.168406	0.573566	0.604004	0.394715	0.448050	3.385636
4	0.655748	0.169142	0.175110	0.621740	0.570452	0.403479	0.440626	2.775454
5	0.630573	0.187013	0.182414	0.590094	0.461488	0.413644	0.378110	2.817341
6	0.711182	0.138992	0.149826	0.679755	0.490252	0.367792	0.481433	2.583163
7	0.717354	0.140232	0.142414	0.703714	0.457070	0.398571	0.463940	2.284897
8	0.605402	0.192292	0.202306	0.555011	0.507622	0.352011	0.437399	2.902953
9	0.617353	0.223949	0.158699	0.591842	0.538732	0.381809	0.505152	3.329938
10	0.589267	0.220948	0.189785	0.619852	0.520637	0.385655	0.325507	3.248518

### Note

Now we have to merge the transcript data with the emotion data.  
For that we have to merge the dataframes on the basis of student id.

**Prompt for merging the emotion\_df and transcript\_df:**

```

df_dominantEmotion_for_everystudent
and final_df add these df on index

```

response from the chatgpt:

```
final_df_with_emotions = pd.concat([final_df,
    df_dominantEmotion_for_everystudent], axis=1)

# Print the final merged DataFrame
print(final_df_with_emotions.head(10))
```

Using the above code, the emotion data and transcript data were merged on the basis of student id. The final dataframe after merging the emotion and transcript data is shown below:

### Final Dataframe

id	avg_positive	avg_negative	avg_neutral	avg_confident
1	0.709199	0.141214	0.149586	0.733828
2	0.722006	0.107541	0.170453	0.684879
avg_hesitant		avg_concise	avg_enthusiastic	avg_speech_speed
0.485172		0.429418	0.466497	3.113771
0.436158		0.484221	0.516685	3.269092
dominant_emotion_top1		dominant_emotion_top2		
neutral		fear		
happy		neutral		

## 5 Prompts for Analysis on the DataFrame made above

### 5.1 Basic Statistics

Prompt:

```
How to know whether there are any missing values in the final DataFrame?
or just give me description of the dataframe.
```

Response:

```
print(final_df_with_emotions.describe())
```

The basic statistics of the final DataFrame are as follows:



	avg_positive	avg_negative	avg_neutral	avg_confident	avg_hesitant	avg_concise	avg_enthusiastic	avg_speech_speed
count	10.000000	10.000000	10.000000	10.000000	10.000000	10.000000	10.000000	10.000000
mean	0.652534	0.178566	0.168900	0.635428	0.507159	0.401131	0.446340	2.971076
std	0.058600	0.048346	0.019202	0.060950	0.052672	0.036592	0.057471	0.361271
min	0.567257	0.107541	0.142414	0.555011	0.436158	0.352011	0.325507	2.284897
25%	0.608390	0.140478	0.152044	0.590531	0.467409	0.382770	0.438206	2.785926
50%	0.643161	0.178077	0.169430	0.620796	0.498937	0.396643	0.455995	3.008362
75%	0.710686	0.213784	0.180588	0.683598	0.534208	0.411103	0.477699	3.263949
max	0.722006	0.264337	0.202306	0.733828	0.604004	0.484221	0.516685	3.385636

## 5.2 Correlation Analysis

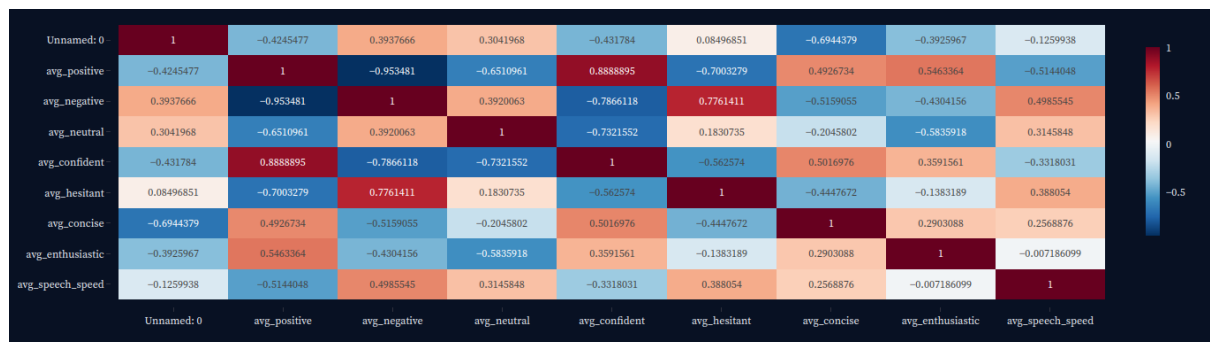
Now i did correlation analysis on the final DataFrame to understand the relationship between different features. **Prompt:**

Now, calculate the correlation matrix of the final DataFrame final\\_df.

**Response:**

```
correlation_matrix = final_df_with_emotions.corr()
print(correlation_matrix)
```

Writng and running the code, there was a error in chatgpt response, but I corrected it and ran the code. The correlation matrix is shown below:



## 5.3 Based on the final\_df, I asked for analysis on the data.

**Prompt:**

now problem statement is

In this task, you will work with a dataset of 10 candidates containing emotion scores, transcript scores, and the corresponding transcripts extracted from their introduction videos. Your goal is to use the ChatGPT with your prompt engineering skills along with performing Exploratory Data Analysis (EDA) on the data provided, to generate valuable and actionable insights from the data.

Actionable insights include:

1. Can we recruit the candidate or not? With suitable reasons from the data.
2. Analysis of communication skills and finding areas of expertise based on data.
3. Any other insights which help us make decision about the candidate are also welcomed.

This task is designed to assess your ability to preprocess data, create effective prompts, and perform EDA to extract meaningful and actionable information.

Main flow should be like this:

- give plots idea to perform some exploratory data analysis (EDA) on the datasets.
- how to analysis to generate insights about each candidate.
- recommendations based on those insights.
- Finally give a scoring mechanism to rank the students
- Any other insights is welcomed.

## Response:

Let's start with the EDA. I'll create a Python script to load and analyze the data. We'll use pandas for data manipulation and matplotlib for visualization.

Candidate Video Analysis Script [Click to open code](#)

This script performs the following tasks:

Loads the emotion and transcript data

Analyzes the emotion data to calculate average emotion scores, gaze, and blink information

Analyzes the transcript data to calculate average scores and speaking time

Combines the summaries from both datasets

Creates a function to plot emotion distribution over time for a candidate

Generates insights for each candidate based on their data

Prints the insights for all candidates

Plots the emotion distribution for the first candidate as an example

## 6 Conclusion

This section summarizes the key findings and insights from the data analysis project.