



Data Scientist Intern R2

-Agam Pandey
IIT Roorkee

Prompt Engineering Documentation:

Get the EDA documentation at [Link](#). It contains detailed documentation

1. **Problem Statement**
2. **Approach**
3. **Insights**

Problem Statement

In this task, I had to analyze a dataset of 10 candidates, which included emotion scores, transcript scores, and transcripts from their introduction videos. The objective was to leverage prompt engineering with ChatGPT and conduct EDA to derive actionable insights, such as:

- **Candidate Recruitment Decision:** Assess whether to recruit candidates, supported by data-driven reasons.
- **Communication Skills Analysis:** Evaluate communication abilities and identify areas of expertise.
- **Additional Insights:** Any other findings that aid decision-making regarding candidates.

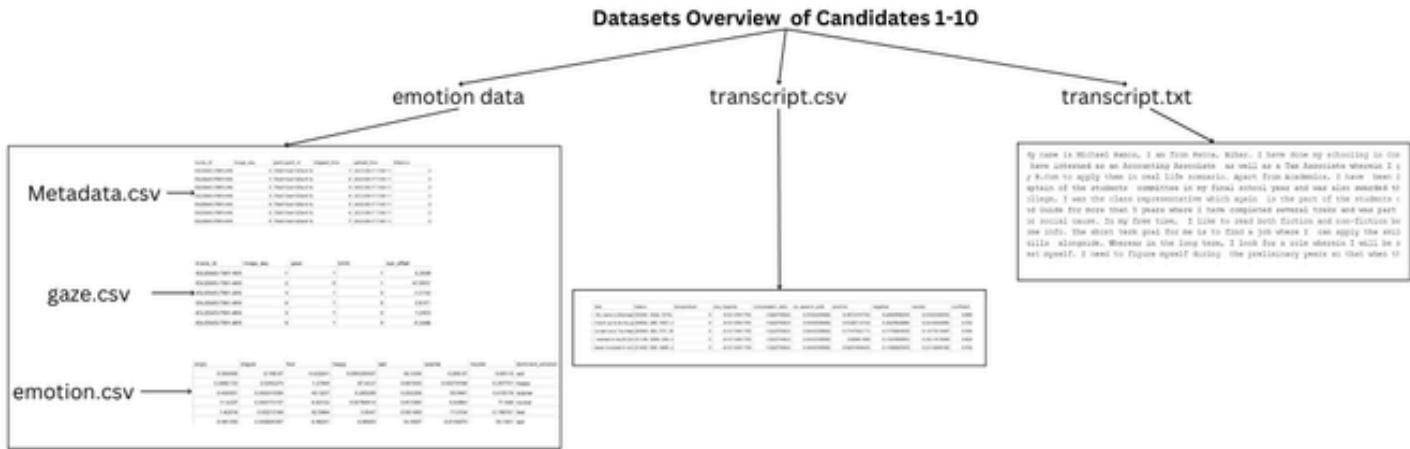
Dataset Components:

- **Emotion Scores:** Captures emotional expressions throughout the video.
- **Transcript Scores:** Scores derived from the transcripts.
- **Transcript Text:** Actual text of the video transcripts.

Deliverables:

1. **Prompt Engineering Documentation:** Outline the prompt engineering process, including prompts used and rationale.
2. **Responses from ChatGPT:** Document the outputs generated.
3. **Exploratory Data Analysis Report:** Summarize findings with supporting visualizations and any scripts used.

Dataset Overview and Approach:



How did I use Prompt Engineering?

To remove the more manual task of directly uploading and asking questions in another website i.e.

1. [Chatgpt.com](#)
2. [Claude.ai](#)
3. [Gemini](#)

I used [PandasAI](#), which has both features of Pandas to process the dataframes and AI (LLMs) for Natural Language Query on the datasets.



Prompt Engineering with LLMs website:

My first prompts were just to understand the problem statement and compared each Chat model's response.

ChatGPT

Visualizations (e.g., plots, charts) to support your insights.
Any code or scripts used for EDA.

Evaluation Criteria:
Your performance in this task will be evaluated based on the following criteria:
Effectiveness of prompts in extracting insights from the data.
Clarity and depth of EDA findings.
Creativity and innovation in approaching the task.
Documentation quality and organization.

< 2/2 >

To tackle this project, we need to break it down step-by-step. Here's a comprehensive roadmap for analyzing the dataset, performing Exploratory Data Analysis (EDA), applying prompt engineering with ChatGPT, and scaling the analysis across multiple candidates.

Key Topics to Focus On:

1. Candidate Evaluation Metrics:
 - Emotion Analysis: Evaluate how candidates manage their emotions throughout the introduction video.

ClaudeAI



AP this is the problem statement I have to work with
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:
Can we recruit the candidate or not? With suitable reasons from the data.
Analysis of communication skills and finding areas of expertise based on data.
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.

Prompt Given

Please analyze the attached dataset and provide a comprehensive list of all deliverables for the project based on the problem statement. I need an in-depth understanding of the topics I should cover from the dataset in order to build the most effective project. Specifically:

1. Identify the key topics I should focus on from the dataset.
2. Provide a clear and detailed structure or roadmap for approaching this problem.
3. Suggest the exact metrics I should track or use for evaluation.
4. Deliver actionable insights derived from the data that will guide decision-making.
5. Since I need to compare this data among 10 different candidates (the attached dataset represents only 1 candidate), explain how I should scale this analysis for multiple candidates, ensuring consistency and accuracy across comparisons.

The goal is to obtain precise guidance on how to best approach the project, ensure robust evaluation criteria, and maximize insights from the dataset.

Then the text content copied from the Problem Statement Google Slide was added below

Evaluate Introduction videos

In this task, you will work with a dataset of 10 candidates containing emotion

Response

1. [ChatGPT Response Link](#)
2. [ClaudeAI Response Link](#)

Prompt engineering with PandasAI Agents (python notebook interactive cell)

Used PandasAI to convert the loaded pandas dataframe into SmartDataframe. and configured ChatGPT “gpt-4” LLM for inference.

```
gaze_agent = SmartDataframe(gaze_df[candidate_id], config={"llm": llmodel})
emotion_agent = SmartDataframe(emotion_df[candidate_id], config={"llm": llmodel})
transcript_agent = SmartDataframe(transcript_df[candidate_id], config={"llm": llmodel})
```

1. General EDA of all the dataset with PandasAI ChatGPT Agent with tailored and concise prompts so that model does not hallucinate.

“ emotion_response = emotion_agent.chat("Perform an EDA of the dataset and generate key statistics and sentiment analysis. Include plots.") ”

The EDA based on the response of ChatGPT was:

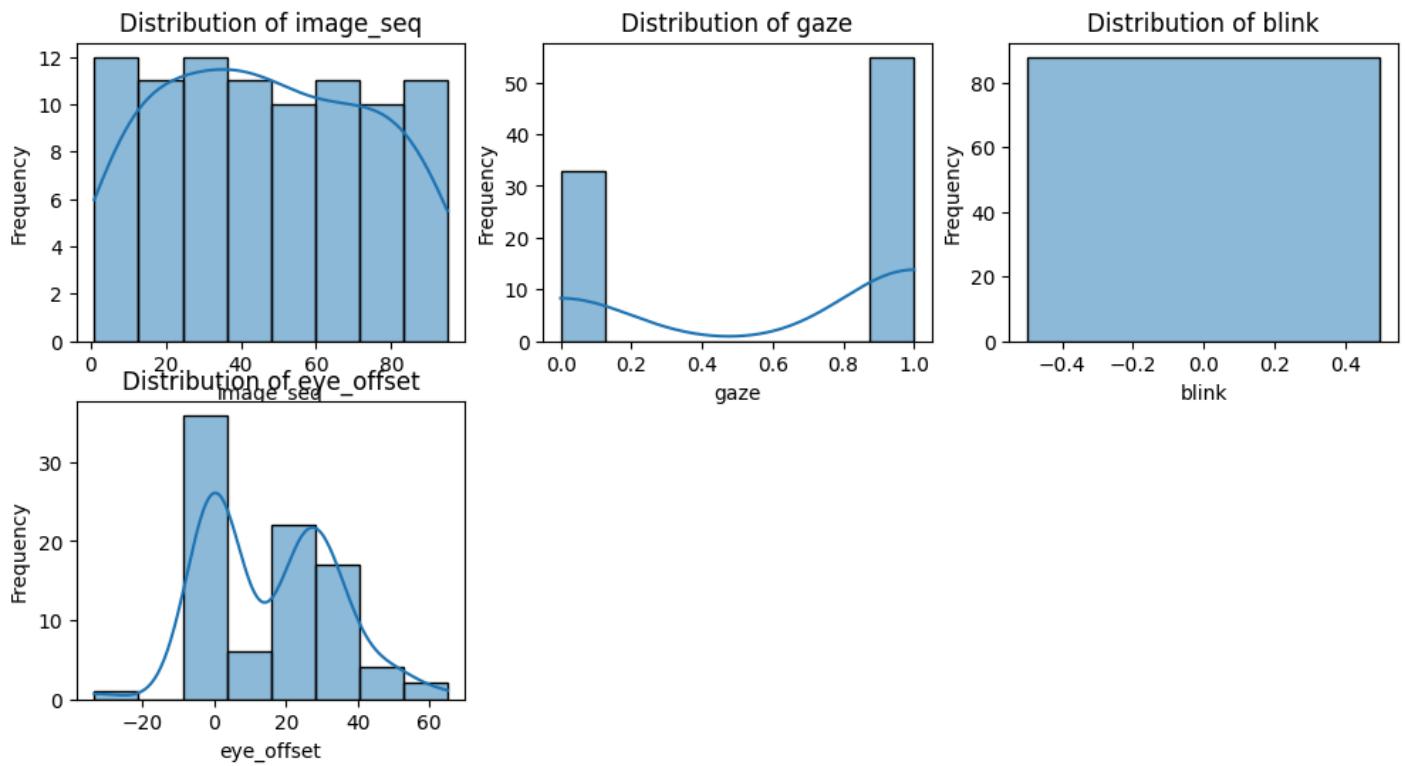
```
Id      seek      start      end      temperature      avg_logprob \
count  18.000000  18.000000  18.000000  18.000000  18.000000 \
mean   8.500000  3009.333333  41.002222  45.931111   0.0   -0.306962
std    5.338539  2598.406891  26.117089  26.294913   0.0   0.197591
min    0.000000  0.000000  5.500000   0.0   0.197591
25%   4.250000  0.000000  19.680000  24.400000   0.0   -0.447179
50%   8.500000  2776.000000  40.560000  46.640000   0.0   -0.278873
75%  12.750000  5336.000000  62.420000  66.660000   0.0   -0.196253
max   17.000000  8272.000000  62.720000  68.770000   0.0   -0.196253

compression_ratio      no_speech_prob      positive      negative      neutral \
count   18.000000  18.000000  18.000000  18.000000  18.000000 \
mean   1.666544  0.748891  0.799199  0.141214  0.149586
std    0.148591  0.213693  0.267232  0.100333  0.089974
min    1.120200  0.635093  0.685554  0.000777  0.000453
25%   1.649819  0.635989  0.687941  0.043334  0.082926
50%   1.651235  0.831773  0.739705  0.080390  0.155741
75%   1.785235  0.910383  0.870056  0.160178  0.224622
max   1.785235  0.910383  0.980389  0.532010  0.267454

confident      hesitant      concise      enthusiastic      speech_speed \
count   18.000000  18.000000  18.000000  18.000000  18.000000 \
mean   0.723003  0.485374  0.294118  0.466497  3.113771
std    0.288339  0.260785  0.272635  0.286292  0.599958
min    0.286049  0.080425  0.012767  0.088580  2.034884
25%   0.576869  0.342871  0.280775  0.211429  2.665702
50%   0.789854  0.487792  0.441477  0.418900  3.134286
75%   0.898626  0.710799  0.612890  0.686992  3.589744
max   0.989931  0.845698  0.919735  0.998310  4.166667

mean
positive  0.76049
negative  0.141214
neutral   0.149586
confident  0.733828
hesitant   0.485172
concise    0.429418
enthusiastic  0.466497
```

The statistical values like mean, std dev, count,etc given was store in DataFrames for each candidates to do plots and understand the dataset. Below is a snippet of plots generated from Gaze dataset of Candidate 1 using Prompts to agents.



Although the current GPT models do not generate unlimited plots in Free APIs, they are a good way to begin with the EDA.

EDA and other processing was done and can be referred at link [Here](#). [EDA - Google Drive](#)



The ChatGPT conversation Prompt engineering have been explained in the main EDA documentation in detail at each step. This documentation follows a appendix for it.



Prompts used in each sections listed below:

Section-1 About the dataset

10.txt Document

Gaze

	movie_id	image_seq	gaze	blink
	f8			
6	83c20b83-7881-499 d-a40d-cc06b65869 f8	6	1	0
7	83c20b83-7881-499 d-a40d-cc06b65869 f8	7	1	0
8	83c20b83-7881-499 d-a40d-cc06b65869 f8	8	1	0
	83c20b83-7881-499			

Emotion

	movie_id	image_seq	angry	disgust
1	83c20b83-7881-499 d-a40d-cc06b65869 f8	0	0.393006	2.1918e-07
2	83c20b83-7881-499 d-a40d-cc06b65869 f8	1	0.0992172	0.0292275

Section-2 Basic EDA on all datasets

Prompt for EDA

- i have a transcript of interview of candidate and from that first i have to know what major things they are talking about so that would be a word cloud i need that then i have to use a analysis to understand technical knowledge score of the candidate is there any such thing like sentiment analysis specific to technical knowledge that i can run on the transcript?

<https://chatgpt.com/c/66f1d2ed-9390-8009-b1ee-638660d9ed33>

<https://chatgpt.com/c/66f1d2ed-9390-8009-b1ee-638660d9ed33>

To understand the structure of the dataset prompt:

movie_id	image_seq	participant_id	elapsed_time	upload_time	distance
83c20b83-7881-499	0	70a013ed-120a-41fe	1	2023-09-17 7:49:11	0
83c20b83-7881-499	1	70a013ed-120a-41fe	2	2023-09-17 7:49:11	0
83c20b83-7881-499	2	70a013ed-120a-41fe	3	2023-09-17 7:49:11	0
83c20b83-7881-499	3	70a013ed-120a-41fe	4	2023-09-17 7:49:11	0
83c20b83-7881-499	4	70a013ed-120a-41fe	5	2023-09-17 7:49:11	0
83c20b83-7881-499	5	70a013ed-120a-41fe	6	2023-09-17 7:49:11	0
83c20b83-7881-499	6	70a013ed-120a-41fe	7	2023-09-17 7:49:11	0
83c20b83-7881-499	7	70a013ed-120a-41fe	8	2023-09-17 7:49:11	0
83c20b83-7881-499	8	70a013ed-120a-41fe	9	2023-09-17 7:49:11	0
83c20b83-7881-499	9	70a013ed-120a-41fe	10	2023-09-17 7:49:11	0
83c20b83-7881-499	10	70a013ed-120a-41fe	11	2023-09-17 7:49:11	0

give me a short description of this dataset and keep it concise
also return me the columns here

The dataset in the image appears to track participant interaction with a movie, represented by `movie_id`. Each row represents a distinct frame or image sequence (`image_seq`) viewed by the participant (`participant_id`). Key metrics include the time elapsed (`elapsed_time`) for each image, the `upload_time` (date and time of upload), and the `distance`, which is currently set to 0 for all rows.

The columns in this dataset are:

- `movie_id`
- `image_seq`
- `participant_id`
- `elapsed_time`
- `upload_time`
- `distance`

Prompt to combine all the data basic describe() eda into one dataframe

The screenshot shows a Jupyter Notebook cell with the following content:

```
1 df=gaze_df[1].describe()
2 df
```

The output of the cell is a DataFrame showing statistical summary statistics for the columns `image_seq`, `gaze`, `blink`, and `eye_offset`:

	image_seq	gaze	blink	eye_offset
count	88.000000	88.000000	88.0	88.000000
mean	46.795455	0.625000	0.0	15.801362
std	27.690058	0.486897	0.0	17.858517
min	1.000000	0.000000	0.0	-33.465500
25%	23.750000	0.000000	0.0	0.293050
50%	45.500000	1.000000	0.0	19.694100
75%	70.250000	1.000000	0.0	28.332000
max	95.000000	1.000000	0.0	65.027600

Below the table, there are buttons for "Generate code with df" and "View recommended plots".

```
1 gaze_all=pd.DataFrame()
2
3 for candidate_id in range(1,candidates+1):
4     gaze_df_idx=gaze_df[candidate_id]
5     df=gaze_df_idx.describe()
6     df.drop(columns=['image_seq'].inplace=True)
7     gaze_all=pd.concat([gaze_all,df],axis=0)
8
9 gaze_all
10
11
```

The code defines a variable `gaze_all` as an empty DataFrame, then loops through `candidate_id` from 1 to `candidates+1`. For each candidate, it selects the corresponding row from `gaze_df` and performs a `describe()` operation. The resulting DataFrame is then concatenated with `gaze_all` along axis 0. Finally, the variable `gaze_all` is printed.

this is the dataset shown in the image, there are 10 such different dataframe i can access using graze_df which is a dictionary now i have used .describe to know about the statistical

numbers here waht i want at the end is that there should be a combined dataframe with candidate id (1 to 10) and their corresponding value given in individual .describe() function
You are an experienced and best data analyst, give me code for this

ChatGPT said:

ChatGPT

To achieve the result where you combine the .describe() output from each DataFrame into a single DataFrame with the corresponding candidate ID (from 1 to 10), you can follow these steps:

Code Explanation:

1. Loop through each candidate ID (from 1 to 10).
2. For each candidate, compute the .describe() output.
3. Add the candidate ID as a new column in each .describe() DataFrame.
4. Concatenate all the individual .describe() outputs into a single DataFrame.

Here's the Python code to accomplish this:

The screenshot shows a code editor window with the following content:

```
4. Concatenate all the individual .describe() outputs into a single DataFrame.

Here's the Python code to accomplish this:

pythontab
Copy code

import pandas as pd

# Initialize an empty DataFrame to store all describe outputs
gaze_all = pd.DataFrame()

# Loop through each candidate's DataFrame and compute the .describe() for each
for candidate_id in range(1, 11): # Assuming candidate IDs range from 1 to 10
    # Access each DataFrame from the dictionary
    gaze_df_idx = gaze_df[candidate_id]

    # Get the describe() output
    df_describe = gaze_df_idx.describe()

    # Add a column for candidate ID
    df_describe['candidate_id'] = candidate_id

    # Drop unnecessary columns if needed (optional)
    df_describe = df_describe.drop(columns=['image_seq'], inplace=False)

    # Append to the cumulative DataFrame
    gaze_all = pd.concat([gaze_all, df_describe])

# Reset index if needed, to make the DataFrame cleaner
gaze_all = gaze_all.reset_index()

# Display the final combined DataFrame
print(gaze_all)
```

Key Details:

1. `[gaze_df[candidate_id]]`: This accesses each DataFrame in your dictionary by the candidate ID.

Key Details:

1. `gaze_df[candidate_id]`: This accesses each DataFrame in your dictionary by the candidate ID.
2. `.describe()`: Summarizes the statistics for each DataFrame.
3. `concat`: Combines all the `.describe()` results into one large DataFrame.
4. The column `candidate_id` is added to help identify which statistics correspond to which candidate.

Let me know if you need further adjustments!

q. adding horizontal mean line

```
gaze_all, x='candidate_id', y='gaze',
nbins=10, color='candidate_id',
title='Gaze Distribution Across Candidates',
labels={'candidate_id':'Candidate ID', 'gaze':'Gaze'},
color_discrete_sequence=palette # Apply Pastel color palette
)
```

i also want to add an horizontal line showing mean value in this plot

To add a horizontal line showing the mean value of `gaze` in the `px.histogram` plot, you can use the `add_shape` method in Plotly to overlay the line on the existing plot. Here's how you can modify the histogram plot to include a horizontal line representing the mean value:

Updated Code for Histogram with Mean Line:

```
python
import plotly.graph_objs as go
import plotly.express as px

# Define the color palette
palette = px.colors.qualitative.Pastel
# Calculate the mean value for 'gaze'
```

 Copy code

 Message ChatGPT

is this insight correct? give more insights from this plot data and rephrase it and make it more concise and correct The plot between sum of gaze i.e. the total number of times candidate is looking at the camera in an interview. As can be seen Candidate 5 and 6 sum_of_gaze value is among the lowest. This indicates these candidates might be not looking at the camera and might either be reading their introduction from a device/paper or ChatGPT models Might be visually impaired (can be a possibility)

The insight from the plot is generally correct but can be refined and made more concise. Here's a rephrased version of the insight:

Rephrased Insight:

The plot shows the total `sum of gaze`, indicating how often each candidate looked at the camera during an interview. Candidates 5 and 6 have notably lower gaze values compared to others. This suggests that these candidates might not have been focused on the camera, potentially reading from a device or paper. Another possibility could be visual impairment, though further investigation would be needed to confirm this.

Additional Insights:

1. **Candidate 4**: This candidate has the highest gaze sum, which could indicate strong engagement and focus on the camera during the interview.
2. **Even Distribution**: Most candidates (except 5 and 6) have gaze sums around or above the mean, suggesting consistent engagement across the majority of candidates.
3. **Possible External Factors**: Candidates with lower gaze values might have experienced distractions, technical difficulties, or external factors affecting their eye contact, which could be worth investigating.

These insights provide a clearer understanding of the candidate's behavior based on the sum of gaze, helping to infer possible reasons behind the varying engagement levels.

this is the plot in gaze dataframe for eye offset i need insight from this plot and explain concisely how does it impact candidate hiring and what does it show?

Insight from the Eye Offset Distribution Plot:

The plot shows the distribution of **eye offset** for each candidate, which represents the deviation of their gaze from a central point (likely the camera) during an interview.

1. **High Variability in Eye Offset:** Candidates 1, 5, 7, and 10 exhibit a wide range of eye offset values, indicating inconsistent focus on the camera. This might suggest distractions or discomfort in maintaining eye contact, which could be interpreted as a lack of confidence or engagement during the interview.
2. **Candidates 6 and 9:** These candidates have narrower distributions of eye offset, meaning their gaze was more consistently focused. Consistent eye contact could be perceived positively, indicating attentiveness and engagement, which may enhance their chances in the hiring process.
3. **Outliers:** Several candidates show significant outliers (e.g., Candidates 1, 2, and 4). These extreme deviations from central eye contact could be due to distractions, nervousness, or technical issues.

Impact on Hiring:

- **Eye Contact and Engagement:** Candidates with more stable and centered eye contact (e.g., Candidates 6 and 9) may be viewed as more focused and engaged, potentially leading to a more favorable impression in the hiring process.
- **Distractions or Nervousness:** Candidates with higher variability (e.g., Candidates 1 and 7) may come across as distracted or nervous, which could negatively impact their performance evaluation.

Conclusion:

Eye offset patterns can influence how candidates are perceived in terms of attentiveness, confidence, and engagement. Consistent eye contact is often associated with confidence and professionalism, while large deviations or frequent shifts may raise concerns about a candidate's focus during the interview.

similarly for this i need plot like the one you previosuly made for the gaze dataframe.
Remember all the conversation, i asked you to make changes in the color pallete and other

things as well

here the plot was wrong in mean line so i changed it to get mean of sum not overall mean.

CHANGE MADE BY ME

```
happy_mean_sum=0

# Calculate the mean value for 'gaze'

for candidate in range(1,candidates+1):

    happy_mean = emotion_all[emotion_all['candidate_id'] == candidate]['happy'].mean()

    happy_mean_sum+=happy_mean

emotion_hist.add_shape(
    type="line",
    x0=0, x1=emotion_all['candidate_id'].max(), # Span the entire x-axis
    y0=happy_mean_sum, y1=happy_mean_sum, # y0 and y1 at mean value
    line=dict(color="red", width=2, dash="dash"),
)
```

To create similar visualizations for the **emotion** DataFrame, as we did for the **gaze** DataFrame, while incorporating the **Pastel** color palette and adding features such as a mean line, here's the Python code for plotting:

Updated Code for Plotting Emotion Data:

```
```python
```

```
import plotly.express as px

import plotly.graph_objs as go
```

```
Define the Pastel color palette
palette = px.colors.qualitative.Pastel

Example 1: Histogram for 'happy' emotion across different candidates
emotion_hist = px.histogram(
 emotion_all, x='candidate_id', y='happy',
 nbins=10, color='candidate_id',
 title='Happiness Distribution Across Candidates',
 labels={'candidate_id':'Candidate ID', 'happy':'Happiness'},
 color_discrete_sequence=palette # Apply Pastel color palette
)

Calculate mean happiness value and add a horizontal line for it
happy_mean = emotion_all['happy'].mean()

emotion_hist.add_shape(
 type="line",
 x0=0, x1=emotion_all['candidate_id'].max(), # Span the entire x-axis
 y0=happy_mean, y1=happy_mean, # y0 and y1 at mean value
 line=dict(color="red", width=2, dash="dash"),
)

Add annotation for the mean line
emotion_hist.add_annotation(
 x=emotion_all['candidate_id'].max() / 2, # Position annotation in the middle of the x-axis
```

```
y=happy_mean,
text=f"Mean Happiness: {happy_mean:.2f}",
showarrow=False,
yshift=10,
font=dict(color="red", size=12)
)
```

```
Example 2: Box plot for 'sad' emotion distribution by candidate
```

```
sad_box = px.box(
 emotion_all, x='candidate_id', y='sad',
 points='all', title='Sadness Distribution Across Candidates',
 labels={'candidate_id':'Candidate ID', 'sad':'Sadness'},
 color_discrete_sequence=palette # Apply Pastel color palette
)
```

```
Example 3: Scatter plot showing 'neutral' emotion vs. 'happy' emotion
```

```
emotion_scatter = px.scatter(
 emotion_all, x='neutral', y='happy',
 color='candidate_id',
 title='Neutral vs. Happiness Emotions per Candidate',
 labels={'neutral':'Neutral', 'happy':'Happiness'},
 color_discrete_sequence=palette # Apply Pastel color palette
)
```

```
Show the plots
```

```
emotion_hist.show()
```

```
sad_box.show()
```

```
emotion_scatter.show()
```

三

### ### Explanation:

1. `emotion_hist`: A histogram for the `happiness` distribution, including a red dashed line for the mean happiness value.
  2. `sad_box`: A box plot showing the distribution of `sadness` across different candidates.
  3. `emotion_scatter`: A scatter plot displaying the relationship between `neutral` and `happiness` emotions, colored by candidate ID.

This code applies the **Pastel** color palette to each plot and ensures a consistent visualization style as requested.

Let me know if you need any further adjustments!

I have a dataset of transcripts.txt of interview video of 10 candidates now i want to do things lto get insights into the candidates and hiring What should i do with the trannscript to ensure i udnerstand each candidate's potential, capability, role fit what kind of role can each be allotted It can be NLP techniques also

To gain insights into each candidate's potential, capability, and role fit from the interview transcripts, you can apply several **NLP techniques** and other methods to analyze the text. Here's a step-by-step guide to help you:

## 1. Text Preprocessing:

- **Tokenization:** Break down the transcript into sentences or words to analyze individual parts of speech or phrases.

- **Stopword Removal:** Remove common words like “the,” “is,” etc., which do not contribute to understanding the candidate’s traits.
- **Lemmatization/Stemming:** Convert words to their root form (e.g., “running” becomes “run”) to reduce redundancy and improve analysis accuracy.

## 2. Sentiment Analysis:

- **Purpose:** Understand the emotional tone of the interview responses, such as whether the candidate is confident, enthusiastic, or doubtful.
- **Tool:** Use libraries like VADER or TextBlob to perform sentiment analysis on the transcripts.
- **Application:** Higher positivity in sentiment could suggest enthusiasm, while a mix of neutral and negative sentiment could indicate hesitation or uncertainty.

## 3. Topic Modeling:

- **Purpose:** Discover the key topics or themes each candidate discussed during the interview.
- **Technique:** Use **Latent Dirichlet Allocation (LDA)** to identify themes in the text.
- **Application:** This will help you understand which candidates discussed more relevant topics related to their role or showcase expertise in particular areas.

## 4. Named Entity Recognition (NER):

- **Purpose:** Identify specific entities such as skills, job titles, organizations, etc., mentioned in the transcript.
- **Tool:** Use models like **SpaCy** for NER to detect important terms.
- **Application:** You can map out the candidate's experience, capabilities, and role fit based on the entities they mention (e.g., "Python" for a developer role, "team management" for a leadership role).

## 5. Skills Extraction:

- **Purpose:** Automatically extract mentions of technical or soft skills.
- **Technique:** Use **keyword extraction** methods (TF-IDF or RAKE) or specific pre-built models like **SkillNER** to detect competencies.
- **Application:** This will help match each candidate's skill set to the requirements of different roles.

## 6. Text Summarization:

- **Purpose:** Summarize the transcript for a high-level overview of the candidate's strengths and responses.
- **Technique:** Use extractive or abstractive summarization algorithms (e.g., Hugging Face transformers) to generate summaries of each interview.
- **Application:** These summaries can give you quick insights into each candidate's performance without going through the full transcript.

## 7. Personality Insights:

- **Purpose:** Derive personality traits of candidates based on their responses.
- **Tool:** Use a tool like **IBM Watson Personality Insights** or **LIWC (Linguistic Inquiry and Word Count)** to analyze traits such as openness, agreeableness, conscientiousness, etc.
- **Application:** This can help you match candidates to roles where certain personality traits are essential (e.g., leadership roles for highly extroverted or assertive candidates).

## 8. Role Fit Analysis:

- **Purpose:** Automatically determine which role suits each candidate based on their responses.
- **Approach:**
  - Build a classification model or rule-based engine to map responses to predefined job roles.
  - **BERT-based models** can be used for contextual analysis, where the candidate's answers are classified into different job-related categories (e.g., technical, management, creative).
- **Application:** This can aid in automating the process of matching candidates with job roles based on how well their answers align with the role's requirements.

## 9. Behavioral Analysis (Question Answering Performance):

- **Purpose:** Evaluate the candidate's ability to respond clearly and concisely to interview questions.
- **Tool:** Use **BERT** or **RoBERTa** based models to evaluate the clarity and relevance of the answers.
- **Application:** This analysis helps assess communication skills and whether candidates respond directly to key questions or tend to evade important topics.

## 10. Key Metrics/Word Clouds:

- **Purpose:** Generate key insights like most frequently mentioned words or phrases.
- **Tool:** Create **word clouds** or use **n-gram analysis** to find recurring themes and important terms.
- **Application:** You can visualize what each candidate emphasized in their interview to match with job requirements.

## Conclusion:

By applying these NLP techniques, you can derive meaningful insights from the transcripts, helping you evaluate the potential and role fit of each candidate effectively. You can automate some of these tasks to assist in making data-driven hiring decisions by identifying key strengths, competencies, and emotional intelligence.

Let me know if you need help with specific implementations of these techniques!

I had error in transcript sentiment analysis.

The approach you're using to create the DataFrame looks reasonable, but there are a few issues to address:

- DataFrame Creation:** The `pd.DataFrame` constructor expects `data` to be a 2D list-like object (e.g., a list of rows or columns) where each row contains data corresponding to each column. However, in your current code, `data` is a list where each element contains a different type of data (processed text, sentiment, etc.), which will lead to errors.
- Reshaping the Data:** You need to ensure that each row of the DataFrame corresponds to a candidate, and the data for each feature (processed text, sentiment, etc.) is placed correctly in columns.
- Handling Complex Data:** The `lda_topics`, `ner_entities`, and other data might be complex structures (lists, dictionaries, etc.) that need to be flattened or converted to string format for inclusion in a CSV file.

Here's the corrected version:

**Corrected Code:**

```
python
import pandas as pd

def analyze_candidate(candidate_id, transcript):
 print(f"\n--- Analyzing Candidate {candidate_id} ---\n")
```

Message ChatGPT

## Section-3 Communication Analysis Score

For graph explanations, ChatGPT response generated.

### Graph Overview:

The graph represents the **Transcript Analysis Scores** for 10 candidates, divided into the following major categories:

1. **Overall Score** (line plot)
2. **Hiring Decision Score** (orange)
3. **Emotional and Personality Insights Score** (purple)
4. **Communication Analysis Score** (pink)
5. **Technical Knowledge Score** (yellow)
6. **Role Fit Score** (cyan)

Each segment indicates how well the candidate performed in that specific area, and the **overall score** provides a cumulative score for each candidate.

### Overall Insights:

- **Top Performers:** Candidates 2, 5, and 9 stand out with high overall scores (above 3.8), indicating well-rounded performances across all metrics.

- **Low Performers:** Candidates 1 and 3 show relatively lower overall scores, particularly in the **communication** and **hiring decision** categories.
- **Balance:** Candidates 4, 6, 7, and 10 have consistent performances, with moderate to high scores in most categories, but a few weaknesses in either technical knowledge or role fit.

## Candidate-by-Candidate Insights:

- Candidate 1:**
  - **Overall Score:** 0.8225 (one of the lowest).
  - **Weakness:** Low in **Role Fit** and **Technical Knowledge** (0.34), affecting the overall score.
  - **Verdict:** **Not recommended** for hiring due to poor technical knowledge and a low role fit score.
- Candidate 2:**
  - **Overall Score:** 0.9705 (one of the highest).
  - **Strengths:** Strong performance in all areas, especially **Communication** and **Role Fit**.
  - **Verdict:** **Highly recommended** for hiring due to strong performance across the board.
- Candidate 3:**
  - **Overall Score:** 0.7964.
  - **Weakness:** Lower **Role Fit** and **Communication** scores, but good emotional insights.
  - **Verdict:** **Not recommended** due to poor role fit and communication, despite strong emotional insights.
- Candidate 4:**
  - **Overall Score:** 0.9118.
  - **Strengths:** Well-rounded, with good **Technical Knowledge** and **Hiring Decision** scores.
  - **Verdict:** **Recommended** for hiring, with balanced scores across most categories.
- Candidate 5:**
  - **Overall Score:** 0.9371 (high performer).
  - **Strengths:** Strong in **Technical Knowledge** and **Role Fit**, consistent across all metrics.
  - **Verdict:** **Highly recommended** due to strong technical knowledge and role fit.
- Candidate 6:**
  - **Overall Score:** 0.930.
  - **Strengths:** Strong in **Communication** and **Hiring Decision** scores.
  - **Verdict:** **Recommended** for hiring, with strengths in communication and technical abilities.
- Candidate 7:**
  - **Overall Score:** 0.9571.
  - **Strengths:** Strong in all areas except **Role Fit**.
  - **Verdict:** **Recommended** with caution; needs a better role fit but excels in technical and communication areas.
- Candidate 8:**
  - **Overall Score:** 0.9501.
  - **Strengths:** Balanced across all areas, with particular strengths in **Hiring Decision** and **Technical Knowledge**.
  - **Verdict:** **Highly recommended** due to strong technical skills and role fit.
- Candidate 9:**
  - **Overall Score:** 0.9571 (high performer).
  - **Strengths:** Balanced performance across all areas.

- **Verdict: Highly recommended** for hiring, showing strength in technical and emotional insights.

#### 10. Candidate 10:

- **Overall Score:** 0.8824.
- **Strengths:** Strong in **Technical Knowledge** and **Emotional Insights**, with a lower score in **Hiring Decision**.
- **Verdict: Recommended** but may need further evaluation in fit and final hiring decision.

### Final Hiring Recommendations:

- **Highly Recommended for Hiring:** Candidates 2, 5, 8, and 9 – all exhibit high scores in **Role Fit, Technical Knowledge, and Hiring Decision**.
- **Recommended for Hiring:** Candidates 4, 6, 7, and 10 – while not perfect, these candidates have strong overall scores with specific areas of strength.
- **Not Recommended:** Candidates 1 and 3 – these candidates performed poorly in key areas like **Role Fit** and **Technical Knowledge**, making them less suitable.

The candidates highlighted for **high performance** demonstrate strong overall capabilities, making them suitable for a wide range of roles, especially those demanding technical expertise and good role fit.

### Section-4 Emotional Intelligence Score

### Section-5 Transcript Analysis

Prompt given to do analysis on the Transcripts using NLP techniques.

Combined Candidate Transcript AnalysisNLP				
	candidate_id	processed_text	sentiment	lda_topics
1	1	hello jeffrey shepherd currently pursue postgraduate management iim .....	{'neg': 0.025, 'neu': 0.809, 'pos': 0.167, 'compound': 0.9914}	(0, '0.018**add + 0.018**year' + 0.018**experi + 0.018**work 0.018**help' + 0.018**iim')
2	2	hello beside cameron barajas thnll apply opportunity today recently complete ..... name michael	{'neg': 0.0, 'neu': 0.849, 'pos': 0.151, 'compound': 0.9923}	(0, '0.009**star + 0.009**internl + 0.009**basic 0.009**provide')
		.....	.....	(0, '0.007**smz')

This is NLP analysis that involves topics like

- 'processed\_text':
- 'sentiment':
- 'lda\_topics':
- 'ner\_entities':
- 'skills':
- 'summary'

for sentiment analysis i used vader sentiment analyser and other NLP techniques suitable for different columns.

Analyze the output stored in this csv file and give me insights on each candidate, keep it concise and also add verdict of which role is the candidate suitable for, should we hire them or not and give reasons

Analyzed ^

```

python

import pandas as pd
Load the CSV file
file_path = '/mnt/data/combined_candidate_transcript_analysisNLP.csv'
candidate_data = pd.read_csv(file_path)

Display the first few rows of the dataset to understand its structure
candidate_data.head()

```

This is NLP analysis that involves topics like 'processed\_text': 'sentiment': 'lda\_topics': 'ner\_entities': 'skills': 'summary' for sentiment analysis i used vader sentiment analyser and other NLP techniques suitable for different columns. Analyze the output stored in this csv file and give me insights on each canidate, keep it concise and also add verdict of which role is the candidate suitable for, should we hire them or not and give reasons

# Prompt

## Transcripyt analyse

This is what I did. The ChatGPT agent was working as an experienced interviewer who read through the transcript of candidate divided it into 4 major segments and made a final decision of Hiring. The response generated by ChatGPT was then passed through VaderSentiment Analyser to get scores corresponding to: Role Fit Technical Knowledge Communication Skills Emotional and Personality Insights Hiring Decision Higher score indicates higher metrics in each objective. Now based on this explain me the graph in short and give me insights overall and for each candidate 1 to 10 and who should we hire and why give reasons

## **Section-6. Cumulative Scores**

## Score generation

When chatgpt was hallucinating i used this

i just need scores i told you , i need a table where i get candidate then some parameters where number scores 1 to 5 based on analysis given is written and at the end overall score for that section

Overall EI Score			Recommendation
Candidate	El Score (1-5)	Emotional Stability (1-5)	
Candidate 9	15	Low	Recommended (Positive El, emotionally stable)
Candidate 10	-55	Moderate	Not Recommended (Negative emotional tendencies, poor El)

### Final Verdict:

- Highly Recommended:** Candidates 4, 5, 9 (Strong or positive El with low to moderate emotional volatility).
- Recommended:** Candidate 2 (Moderate El with good emotional stability).
- Not Recommended:** Candidates 1, 3, 6, 7, 8, 10 (Either low El or high emotional volatility, indicating challenges in emotional regulation).

Candidates with **high emotional intelligence** and **low volatility** are well-suited for roles requiring strong interpersonal interactions, stress management, and emotional regulation.



i just need scores i told you , i need a table where i get candidate then some parameters where number scores 1 to 5 based on analysis given is written and at the end overall score for that section



Here is the table with scores from **1 to 5** for each candidate based on the analysis given, including **El Score**, **Emotional Stability**, and an **Overall El Section Score**.

Candidate	El Score (1-5)	Emotional Stability (1-5)	Overall El Score (1-5)
Candidate 1	1	3 ↓	2

To get overall score combined from all the individual metric scores, the prompt was:



now here you have all the images which has tables with each candidates score for different metrics, i need you to give me a overall average score of each candidate from these data  
You can extract the scores into a table then can find average per row so you will get for each candidate the overall score for hiring



Analyzing ^

python

```
import pandas as pd

Extracting the data from the tables into dictionaries for each candidate's scores

candidate_scores = {
 "Candidate 1": [3.5, 3, 3.25, 3.5, 2],
 "Candidate 2": [4, 3.5, 4.25, 4.5, 4.9],
```

Always show details

0 Message ChatGPT 

## Appendix ChatGPT:

∅ ERROR:pandasai.helpers.logger:Failed with error: Traceback (most recent call last):  
File "/usr/local/lib/python3.10/dist-packages/pandasai/pipelines/chat/code\_execution.py", line 85, in execute  
    result = self.execute\_code(code\_to\_run, code\_context)  
File "/usr/local/lib/python3.10/dist-packages/pandasai/pipelines/chat/code\_execution.py", line 175, in execute\_code  
    raise NoResultNotFoundError("No result returned")  
pandasai.exceptions.NoResultNotFoundError: No result returned



The error you're encountering (`NoResultNotFoundError: No result returned`) is likely caused by an issue with the `pandasai` library's interaction with the `chat` method. This typically happens when the code executed in response to the prompt doesn't produce any output or when there is a problem with how the result is processed.

#### Possible Reasons for the Error:

1. **Prompt Doesn't Return an Output:** The prompts you are providing to the `SmartDataframe` objects might not result in any executable code or return a meaningful result. The LLM (language model) might be interpreting the prompt in a way that doesn't yield any output.
2. **Execution Context:** The code execution pipeline in `pandasai` might not be able to generate or capture the result properly, especially if the prompt leads to an operation that doesn't directly produce a returnable value (e.g., generating a plot but not returning it).

1. <https://chatgpt.com/c/66f11129-4188-8009-898d-8a943692a50c>