Project Report

Session: 2024-25 AY

Course Code: BCA-PR 501

Course Name: **MINI PROJECT**

Submitted by:

Aman Singh (41222121)

Vidit (41222175)

Yash Gaula (41222178)

Under Supervision of:

Mr.Inderjeet

DSEU
Crafting Excellence

Delhi Skill & Entrepreneurship University

Dwarka Sec-9, New Delhi

Dwarka sec-9,New Delhi

(Bachelor of Computer Application)

Certificate

This project is for the department of Bachelor of computer Application, under the supervision of **Mr.Inderjeet**, faculty of computer science and engineering at **Delhi skill and Entrepreneurship University**, the project entitled **Astro** in satisfactory manner as a partial fulfillment of required degree of bachelor of computer application for the academic year 2024-25, of DSEU Dwarka Campus

This is to certify that **Aman Singh (41222121)**, **Vidit (41222175)**and **Yash Gaula (41222178)** has contributed in successfully completing the project.

Acknowledgement

We would like to thank **Mr.Inderjeet**, our Professor-in-charge for their support and guidance in completing our project on the topic **Astro.** It was a great learning experience.

We are also thankful to our parents and friend for their constant encouragement and cooperation throught out this project. Without the contribution of group members the project have not been completed so, thankful to each of group members who fairly contributed to the project.

Aman Singh (4122121)

Vidit (41222175)

Yash Gaula (41222178)

Abstract of the Project

The Zodiac Sign Predictor project utilizes the Random Forest algorithm to predict an individual's zodiac sign and mood of person based on their date of birth. Random Forest, a robust ensemble learning method, ensures high accuracy by combining multiple decision trees to map input dates to the corresponding zodiac sign. The project involves preprocessing date data and training the model on zodiac date ranges. With a user-friendly interface, the application delivers precise and efficient predictions. This project demonstrates the practical application of machine learning techniques in creating an engaging astrology-based solution. This project also contain a Power BI dashboard for show essential astrology information related to the each zodiac sign.

I N D E X

INTRODUCTION

The "Zodiac Sign Predictor" project is an innovative application that combines programming, data processing, and data visualization to predict an individual's Zodiac sign based on their date of birth. This project demonstrates the fusion of technology and astrology, catering to the widespread interest in Zodiac signs for entertainment, education, and personal curiosity. The project leverages Python for the predictive logic and Power BI for an interactive and user-friendly representation of insights.

1. Project Background:

Astrology has been a part of human culture for centuries, with Zodiac signs being one of its most popular elements. Each Zodiac sign is associated with specific personality traits and is determined by the position of celestial bodies at the time of a person's birth. The fascination with Zodiac signs persists across cultures and demographics.

This project aims to explore this intriguing concept by creating a tool that can instantly determine the Zodiac sign from a given birth date, making the process both accessible and engaging.

2. Objective of the Project

The primary objective of the "Zodiac Sign Predictor" is to develop an automated system that accurately predicts a person’s Zodiac sign. This project also seeks to present the data in an interactive and visually appealing way using Power BI. The ultimate aim is to blend the power of computational logic with effective data visualization to create a seamless user experience.

3. Scope of the Project

This project encompasses the following key aspects:

Input: The user inputs their date of birth.

Processing: The system calculates the corresponding Zodiac sign based on predefined date ranges for each sign.

Output: The predicted Zodiac sign is displayed, accompanied by visual insights created in Power BI.

While the project focuses on Western Zodiac signs, it does not account for variations like Chinese Zodiac or other astrological systems.

4. Tools and Technologies

The implementation of this project involves:

Python: Utilized for writing the algorithm to calculate the Zodiac sign from the date of birth. Libraries like pandas and numpy streamline data handling.

Jupyter Notebook: Provides an interactive environment to test and refine the code.

Power BI: A powerful visualization tool used to present data insights and trends in an engaging manner.

5. Significance of the Project

This project demonstrates the practical application of programming skills in Python and data visualization techniques in Power BI. By focusing on a widely appealing concept like Zodiac signs, the project bridges the gap between technical implementation and user engagement. It serves as a testament to the versatility of modern tools in solving creative and educational problems.

OBJECTIVE OF THE ANALYSIS

The primary objective of the Zodiac Sign Predictor project is to develop a system that accurately predicts an individual's zodiac sign based on their date of birth. Zodiac signs have been historically associated with personality traits, preferences, and other characteristics, making this analysis engaging and insightful for users interested in astrology.

This project focuses on achieving the following goals:

1. **Prediction of Zodiac Signs:** Implement a Python-based algorithm to correctly determine the zodiac sign of individuals using their birth dates. This involves accurately mapping date ranges to the corresponding astrological signs.

2. **Exploration of Patterns and Trends:** Analyze the data to uncover trends in zodiac sign distribution. For example, understanding the frequency of different zodiac signs among a given population or spotting any seasonal patterns in birth dates.

3. **Data Visualization:** Utilize Power BI to create visually appealing and interactive dashboards. These visualizations will help in presenting insights clearly, such as the proportion of each zodiac sign, trends over time, and other related patterns.

4. **User Accessibility and Insight Delivery:** Ensure that the insights derived from the analysis are easy to interpret and accessible to both casual users and individuals interested in astrology or data analysis.

By combining data analysis, prediction, and visualization, this project aims to bridge the gap between astrology and modern data science, providing an engaging experience for users.

REQUIREMENT OF THE PROJECT

The Zodiac Sign Predictor project involves several technical and functional requirements to ensure successful implementation and analysis.

These requirements are divided into the following categories:

1. Software Requirements:

Python: For implementing the zodiac sign prediction logic and data analysis.

Jupyter Notebook: To develop, test, and document Python code in an interactive environment.

Power BI: For creating dashboards and visualizations to represent the data and insights effectively.

Libraries and Tools:

Pandas and NumPy: For data handling and manipulation.

Date time: For working with date-related operations.

Matplotlib and plotly: For initial data visualization in Python.

Sklearn: used for model building.

Gradio: used for user interface

2. Hardware Requirements:

Processor: Minimum Intel i3 or equivalent for smooth execution.

RAM: At least 8 GB for handling data processing and visualization tasks.

Storage: 2–5 GB of free space to store datasets, scripts, and Power BI files.

3. Data Requirements:

Birth Date Data: A dataset containing individuals' birth dates to predict their zodiac signs.

Additional Data: Additional demographic data (e.g., lucky color, lucky number, mood, description) for extended analysis.

4. Functional Requirements:

A robust Python program to determine zodiac signs based on the birth date.

Accurate mapping of birth dates to zodiac ranges.

Meaningful visualizations showcasing trends, distributions, and insights about zodiac signs.

5. User Requirements:

An intuitive interface or explanation of outputs for easy interpretation by users.

Clear and engaging visual representation of results in Power BI dashboards.

Meeting these requirements ensures the project delivers accurate predictions, insightful analysis, and a user-friendly experience.

METHODOLOGY

The methodology of the Zodiac Sign Predictor project is structured in the following stages:

1. Data Collection and Preparation

The first step involves collecting relevant data, which primarily consists of individuals' birth dates. For this project, publicly available datasets or synthetic data can be used to create a sample dataset.

The collected data needs to be preprocessed, ensuring that:

- Birth dates are in a consistent format (e.g., YYYY-MM-DD).
- Missing or erroneous data is handled appropriately.

2. Zodiac Sign Prediction Logic

The core of the project lies in implementing the logic to predict zodiac signs based on birth dates.

The methodology includes:

- Mapping each zodiac sign to a specific date range (e.g., Aries: March 21–April 19).
- Writing a Python function to check the birth date against these date ranges and return the corresponding zodiac sign.
- Handling edge cases, such as users born on the border dates of zodiac signs.

3. Data Analysis and Exploration

Once the prediction algorithm is in place, the next step is to analyze the data.

This includes:

- **Distribution Analysis:** Identifying how different zodiac signs are distributed across the dataset.
- **Seasonal Trends:** Investigating if there are any patterns related to birth months or seasons.
- **Correlations:** Checking for any correlations between zodiac signs and other demographic data, if available.

4. Visualization and Reporting

After completing the data analysis, visualizing the results is crucial. The methodology for this step involves:

- Using Python libraries like matplotlib and seaborn for initial visualizations (e.g., bar charts, pie charts) to show the distribution of zodiac signs.
- Creating interactive Power BI dashboards to represent the analysis and predictions, making the data accessible to a broader audience.

These dashboards will include visualizations such as:

- Zodiac sign distribution charts.
- Trends over time or by demographics.
- Any insights or patterns identified from the data analysis.

5. Testing and Validation:

To ensure accuracy and reliability, the prediction model and analysis are tested:

- **Testing the Prediction Algorithm:** Validating the zodiac sign prediction by checking against known birth dates.
- **Cross-validation:** If demographic data is included, cross-validation techniques can be used to ensure that any observed trends are statistically significant.

6. User Feedback and Refinement

The final stage involves sharing the project with users or stakeholders to gather feedback. The methodology includes:

- Collecting user feedback on the prediction accuracy and visualization clarity.
- Refining the user interface, visualizations, or prediction logic based on the feedback received.

This methodology ensures that the Zodiac Sign Predictor project is systematically developed, analyzed, and presented with clear insights.

WORKFLOW

The workflow of the Zodiac Sign Predictor project follows a structured series of steps to ensure smooth execution and deliver accurate results.

Step 1: Data Collection

Gather a dataset containing birth dates (and optional demographic details).

Ensure the data is clean and properly formatted for analysis

Astro

About Dataset

The dataset contains 503 rows of Indian names, actual dates of birth, zodiac signs, and additional details like lucky color, lucky number, mood, description, birthplace (Indian cities), occupation, hobby, and favorite cuisine. It is designed for astrology-related or personality-based analysis.

[1]:
```
import pandas as pd
import numpy as np
import plotly.express as px
data = pd.read_excel(r"C:\Users\yash\OneDrive\College\Zodiac Dataset.xlsx")
print("Shape of dataset :- ",data.shape)
print("size of dataset :- ",data.size)
```

```
Shape of dataset :-  (503, 11)
size of dataset :-  5533
```

[2]:
```
data.columns
```

[2]:
```
Index(['Name', 'Date of Birth', 'Zodiac Sign', 'Lucky Color', 'Lucky Number',
       'Mood', 'Description', 'Birth Place', 'Occupation', 'Hobby',
       'Favorite Cuisine'],
      dtype='object')
```

Step 2: Data Preprocessing

Handle missing, duplicate, or invalid data entries.

Organize the data into a tabular format for easy manipulation.

[3]:
```
data.head()
```

[3]:

	Name	Date of Birth	Zodiac Sign	Lucky Color	Lucky Number	Mood	Description	Birth Place	Occupation	Hobby	Favorite Cuisine
0	Onkar Srinivas	2001-05-02	Taurus	DarkSeaGreen	87.0	Calm	Occaecati iusto itaque dolorum incidunt invent...	Udaipur	Psychiatric nurse	Painting	Indian
1	Neysa Gokhale	1944-10-17	Libra	GoldenRod	11.0	Calm	Doloribus voluptatibus aperiam voluptatum esse...	Navi Mumbai	Psychologist, sport and exercise	Sports	Mexican
2	Vaibhav Bansal	2004-07-02	Cancer	LightPink	85.0	Calm	Eos neque reiciendis maxime magnam esse iusto ...	Rewa	Horticulturist, commercial	Sports	Japanese
3	Kaira Guha	1984-03-12	Pisces	SeaGreen	91.0	Relaxed	Voluptatibus consequuntur repudiandae similiqu...	Raurkela Industrial Township	Insurance broker	Painting	Mexican
4	Zain Sundaram	1968-02-06	Aquarius	LightGray	7.0	Energetic	Odio fugiat cupiditate architecto ab odio dolo...	Rohtak	Hydrologist	Traveling	Mexican

[4]:
```
data.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 503 entries, 0 to 502
Data columns (total 11 columns):
 #   Column            Non-Null Count  Dtype
---  ------            --------------  -----
 0   Name              503 non-null    object
 1   Date of Birth     503 non-null    datetime64[ns]
 2   Zodiac Sign       503 non-null    object
 3   Lucky Color       501 non-null    object
 4   Lucky Number      500 non-null    float64
 5   Mood              502 non-null    object
 6   Description       454 non-null    object
 7   Birth Place       503 non-null    object
 8   Occupation        503 non-null    object
 9   Hobby             503 non-null    object
 10  Favorite Cuisine  503 non-null    object
dtypes: datetime64[ns](1), float64(1), object(9)
memory usage: 43.4+ KB
```

[5]:
```
data.count()
```

[5]:
```
Name               503
Date of Birth      503
Zodiac Sign        503
Lucky Color        501
Lucky Number       500
Mood               502
Description        454
Birth Place        503
Occupation         503
Hobby              503
Favorite Cuisine   503
dtype: int64
```

[6]:
```
data.describe()
```

[6]:

	Date of Birth	Lucky Number
count	503	500.000000
mean	1976-07-13 16:56:18.131212736	49.300000
min	1943-11-21 00:00:00	1.000000
25%	1959-11-16 12:00:00	24.750000
50%	1975-06-26 00:00:00	50.000000
75%	1994-05-29 12:00:00	73.000000
max	2006-11-06 00:00:00	99.000000
std	NaN	28.222811

Data Cleaning

[7]:
```
data.isnull().sum()
```

[7]:
```
Name                0
Date of Birth       0
Zodiac Sign         0
Lucky Color         2
Lucky Number        3
Mood                1
Description        49
Birth Place         0
Occupation          0
Hobby               0
Favorite Cuisine    0
dtype: int64
```

Fill Numerical Values with median and Text Values with forward filling

[8]:
```
x = data['Lucky Number'].median()
print("median of Lucky Number column - ",x)
```

```
median of Lucky Number column -  50.0
```

[9]:
```
data["Lucky Number"].fillna(x, inplace = True)
```

[10]:
```
data['Lucky Color'].fillna(method='ffill',inplace=True)
```

[11]:
```
data['Mood'].fillna(method='ffill',inplace=True)
```

Drop entire column due to large amount of missing values

[12]:
```
data.drop(columns=['Description'],inplace=True)
```

Remove Duplicates values

```
[13]: data.isnull().sum()
```

```
[13]: Name               0
      Date of Birth      0
      Zodiac Sign        0
      Lucky Color        0
      Lucky Number       0
      Mood               0
      Birth Place        0
      Occupation         0
      Hobby              0
      Favorite Cuisine   0
      dtype: int64
```

```
[14]: data.duplicated()
```

```
[14]: 0      False
      1      False
      2      False
      3      False
      4      False
             ...
      498    False
      499    False
      500     True
      501     True
      502     True
      Length: 503, dtype: bool
```

```
[15]: data.drop_duplicates(inplace = True)
```

```
[16]: data.duplicated()
```

```
[16]: 0      False
      1      False
      2      False
      3      False
      4      False
             ...
      495    False
      496    False
      497    False
      498    False
      499    False
      Length: 500, dtype: bool
```

Dataset Status After Cleaning

```
[17]: print("Shape of dataset after cleaning :- ",data.shape)
      print("size of dataset after cleaning :- ",data.size)
```

```
Shape of dataset after cleaning :-  (500, 10)
size of dataset after cleaning :-  5000
```

```
[18]: data.head()
```

[18]:

	Name	Date of Birth	Zodiac Sign	Lucky Color	Lucky Number	Mood	Birth Place	Occupation	Hobby	Favorite Cuisine
0	Onkar Srinivas	2001-05-02	Taurus	DarkSeaGreen	87.0	Calm	Udaipur	Psychiatric nurse	Painting	Indian
1	Neysa Gokhale	1944-10-17	Libra	GoldenRod	11.0	Calm	Navi Mumbai	Psychologist, sport and exercise	Sports	Mexican
2	Vaibhav Bansal	2004-07-02	Cancer	LightPink	85.0	Calm	Rewa	Horticulturist, commercial	Sports	Japanese
3	Kaira Guha	1984-03-12	Pisces	SeaGreen	91.0	Relaxed	Raurkela Industrial Township	Insurance broker	Painting	Mexican
4	Zain Sundaram	1968-02-06	Aquarius	LightGray	7.0	Energetic	Rohtak	Hydrologist	Traveling	Mexican

Step 3: Data Analysis

- **Feature Selection:** Select relevant features like day, month, and demographic attributes using statistical methods.Train a machine learning model (e.g., Random forest classifier, logistic regression) to predict zodiac signs.

- **Insights:** Analyze feature importance and document the model's predictive accuracy.

```
[26]: world = data.groupby('Favorite Cuisine')['Name'].count()
      print("Count of Person in Dataset With Same Favorite Cuisine",world)

      Count of Pers
      Chinese     7
      Indian      6
      Italian     9
      Japanese    7
      Mexican     9
      Thai        8
      Name: Name, d
```

Model Bu

```
[27]: import pandas
      from sklearn.
      from sklearn.
      from sklearn.
      from sklearn.
      from sklearn.
      import gradio
      import calend
```

Extract Day and Month from Date of Birth Column

```
[28]: data['Date of Birth'] = pd.to_datetime(data['Date of Birth'])
      data['Month'] = data['Date of Birth'].dt.month
      data['Day'] = data['Date of Birth'].dt.day
```

```
[29]: data['Month']
```

```
[29]: 0       5
      1      10
      2       7
      3       3
      4       2
             ..
      495    12
      496     4
      497    11
      498     6
      499     9
      Name: Month, Length: 500, dtype: int32
```

```
[30]: data['Day']
```

```
[30]: 0       2
      1      17
      2       2
      3      12
      4       6
             ..
      495    28
      496     5
      497    26
      498    11
      499    10
      Name: Day, Length: 500, dtype: int32
```

Data Encoding for Model

```
[31]: label_encoder = LabelEncoder()
      data['Zodiac Label'] = label_encoder.fit_transform(data['Zodiac Sign'])
```

Feature Selection

```
[32]: X = data[['Month', 'Day']]
      y = data['Zodiac Label']
```

```
[33]: X.head()
```

	Month	Day
0	5	2
1	10	17
2	7	2
3	3	12
4	2	6

```
[34]: y.head()
```

```
[34]: 0    10
      1     6
      2     2
      3     7
      4     0
      Name: Zodiac Label, dtype: int32
```

```
[35]: scaler = StandardScaler()
      X_scaled = scaler.fit_transform(X)
```

```
[36]: X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
```

```
[37]: model = RandomForestClassifier(random_state=42)
      model.fit(X_train, y_train)
```

```
[37]: RandomForestClassifier
      RandomForestClassifier(random_state=42)
```

```
[38]: y_pred = model.predict(X_test)
      accuracy = accuracy_score(y_test, y_pred)
      print("Model Accuracy:", accuracy * 100, "%")

      Model Accuracy: 97.0 %
```

```
[39]: class_report = classification_report(y_test, y_pred)
```

Step 4: Visualization Creation

Create initial visualizations in Python using libraries like matplotlib and seaborn to understand the data.

Develop interactive dashboards in Power BI to present key insights:

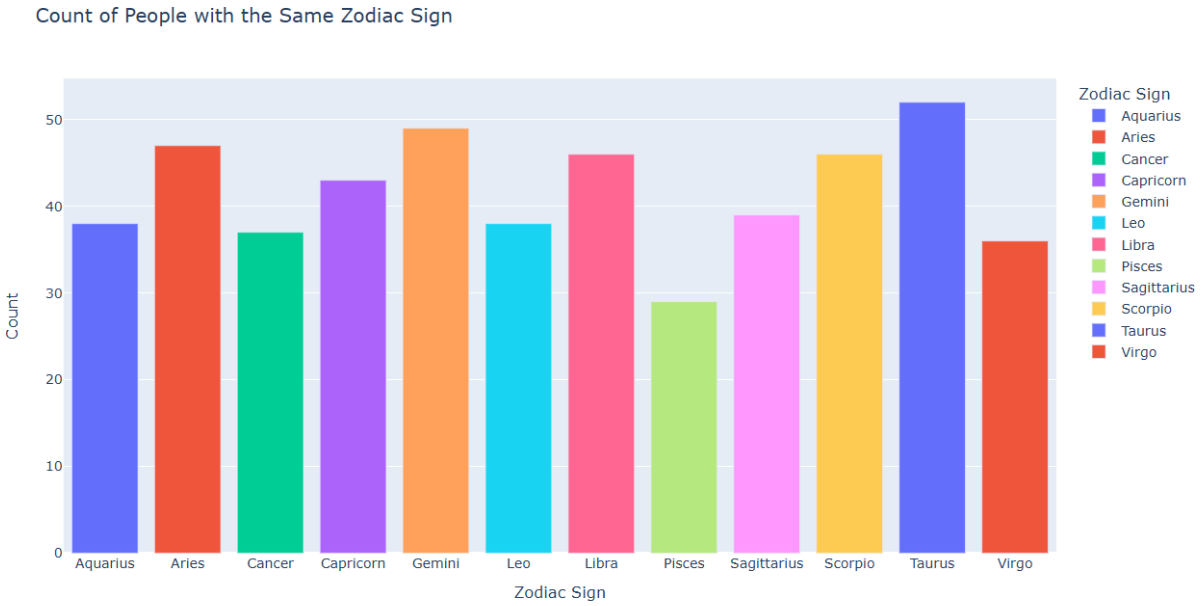
- Proportional distribution of zodiac signs.
- Time-based trends (e.g., monthly or seasonal patterns).
- Any interesting or unusual findings.

Insights using Visualization

[19]:
```
group = data.groupby('Zodiac Sign')['Name'].count()
print("Count of Person in Dataset With Same Zodic Signs",group)
```

```
Count of Person in Dataset With Same Zodic Signs Zodiac Sign
Aquarius       38
Aries          47
Cancer         37
Capricorn      43
Gemini         49
Leo            38
Libra          46
Pisces         29
Sagittarius    39
Scorpio        46
Taurus         52
Virgo          36
Name: Name, dtype: int64
```

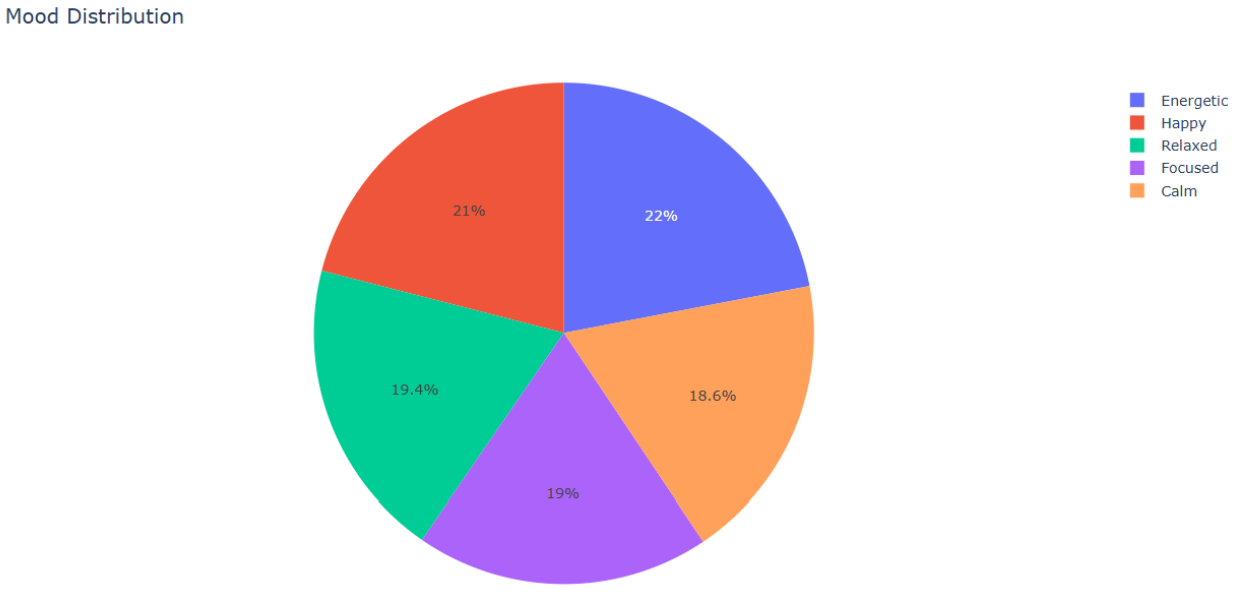
[20]:
```
group = group.reset_index()
group.columns = ['Zodiac Sign', 'Count']

fig = px.bar(
    group,
    x='Zodiac Sign',
    y='Count',
    title='Count of People with the Same Zodiac Sign',
    color='Zodiac Sign'
)

fig.update_layout(height=600, xaxis_title="Zodiac Sign", yaxis_title="Count")
fig.show()
```



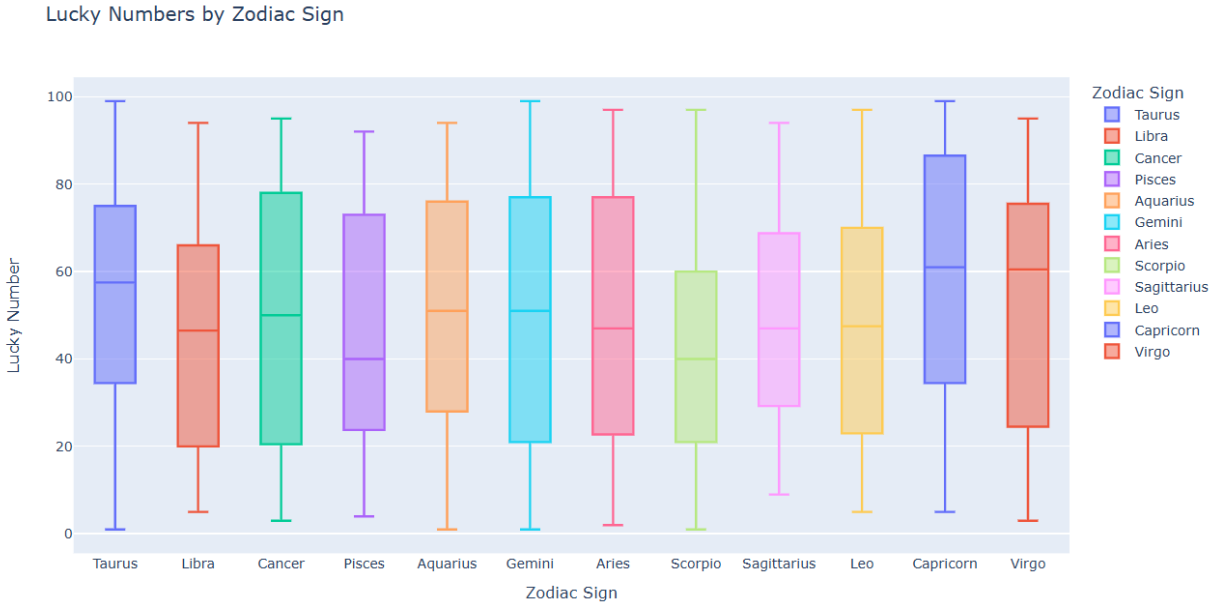
[21]:
```
group = data.groupby('Hobby')['Name'].count()
print("Count of Person in Dataset With Same Hobby",group)
```

```
Count of Person in Dataset With Same Hobby Hobby
Cooking      87
Gardening    89
Painting     82
Reading      83
Sports       83
Traveling    76
Name: Name, dtype: int64
```

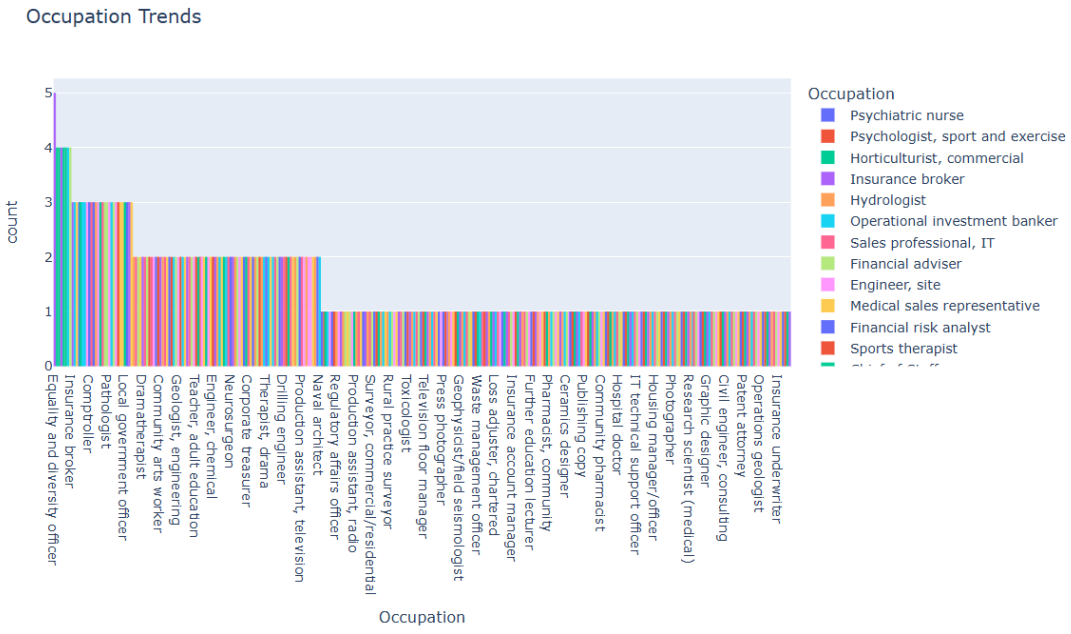
[22]:
```
fig = px.pie(
    data,
    names='Mood',
    title="Mood Distribution"
)
fig.update_layout(height=600)
fig.show()
```



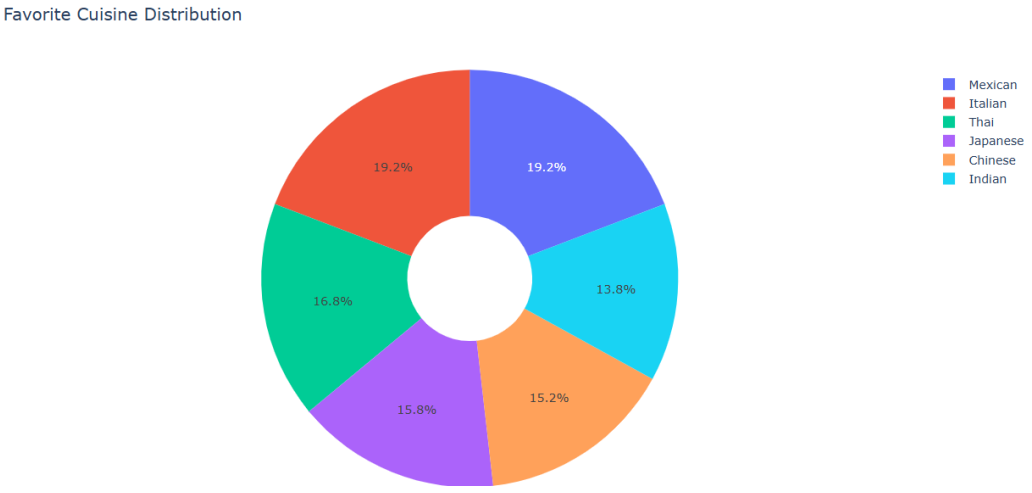
[23]:
```
fig = px.box(
    data,
    x='Zodiac Sign',
    y='Lucky Number',
    title="Lucky Numbers by Zodiac Sign",
    color='Zodiac Sign'
)
fig.update_layout(height=600)
fig.show()
```



[24]:
```
fig = px.histogram(
    data,
    x='Occupation',
    title="Occupation Trends",
    color='Occupation',
    labels={'Occupation': 'Occupation', 'count': 'Frequency'}
)
fig.update_xaxes(categoryorder='total descending')
fig.update_layout(height=600)
fig.show()
```



[25]:
```
fig = px.pie(
    data,
    names='Favorite Cuisine',
    title="Favorite Cuisine Distribution",
    hole=0.3
)
fig.update_layout(height=600)
fig.show()
```



Step 5: Model Building

Model building contains Feature selectin and scaling then split dataset for Training and Testing.

Model Building

[36]:
```
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score,classification_report
import gradio as gr
import calendar
```

Extract Day and Month from Date of Birth Column

[38]:
```
data['Date of Birth'] = pd.to_datetime(data['Date of Birth'])
data['Month'] = data['Date of Birth'].dt.month
data['Day'] = data['Date of Birth'].dt.day
```

[39]:
```
data['Month']
```

[39]:
```
0       5
1      10
2       7
3       3
4       2
       ..
495    12
496     4
497    11
498     6
499     9
Name: Month, Length: 500, dtype: int32
```

[40]:
```
data['Day']
```

[40]:
```
0      2
1     17
2      2
3     12
4      6
      ..
495   28
496    5
497   26
498   11
499   10
Name: Day, Length: 500, dtype: int32
```

Data Encoding for Model

[42]:
```
label_encoder = LabelEncoder()
data['Zodiac Label'] = label_encoder.fit_transform(data['Zodiac Sign'])
```

Feature Selection

[44]:
```
X = data[['Month', 'Day']]
y = data['Zodiac Label']
```

[45]:
```
X.head()
```

[45]:

	Month	Day
0	5	2
1	10	17
2	7	2
3	3	12
4	2	6

[47]:
```
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
```

[48]:
```
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
```

[49]:
```
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
```

[49]:
```
RandomForestClassifier(random_state=42)
```

[50]:
```
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Model Accuracy:", accuracy * 100, "%")
```

```
Model Accuracy: 97.0 %
```

[51]:
```
classt_report = classification_report(y_test, y_pred)
print(classt_report)
```

```
              precision    recall  f1-score   support

           0       1.00      1.00      1.00         7
           1       1.00      0.80      0.89        10
           2       1.00      0.89      0.94         9
           3       1.00      1.00      1.00        12
           4       1.00      1.00      1.00        11
           5       0.92      1.00      0.96        12
           6       1.00      1.00      1.00        11
           7       0.67      1.00      0.80         4
           8       1.00      1.00      1.00         4
           9       1.00      1.00      1.00        10
          10       1.00      1.00      1.00         4
          11       1.00      1.00      1.00         6

    accuracy                           0.97       100
   macro avg       0.97      0.97      0.97       100
weighted avg       0.98      0.97      0.97       100
```

Step 6: Testing and Validation

Test the zodiac sign prediction function against a set of known birth dates.

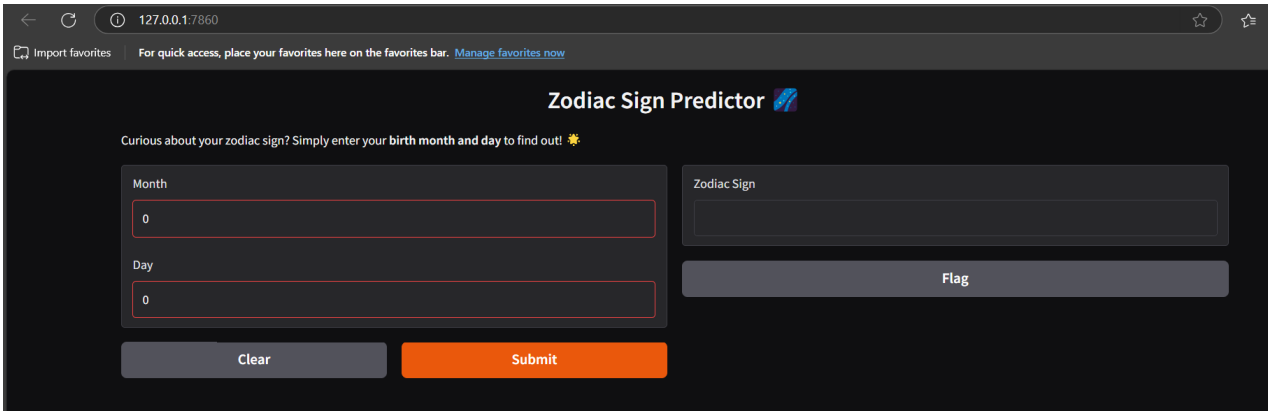
Cross-verify visualizations with the raw data to ensure accuracy.

Refine the analysis or dashboards based on testing feedback.

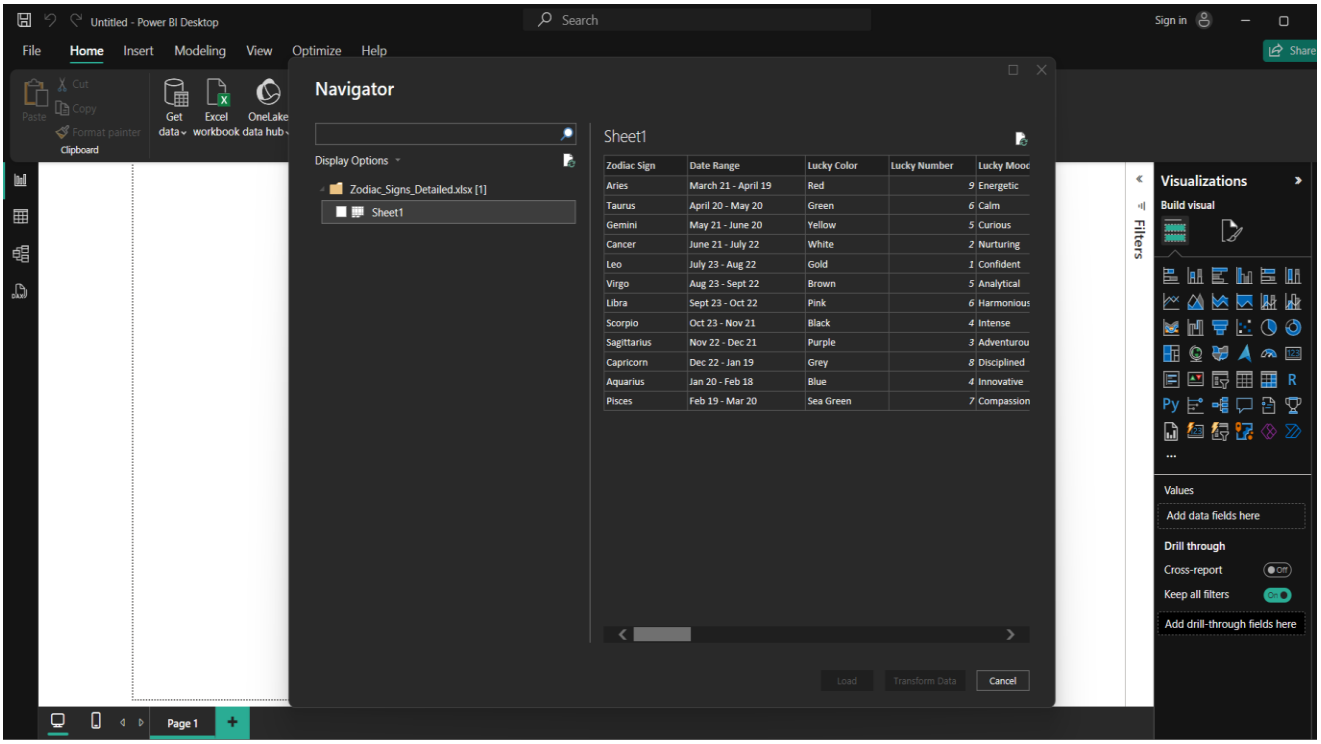
Model Testing

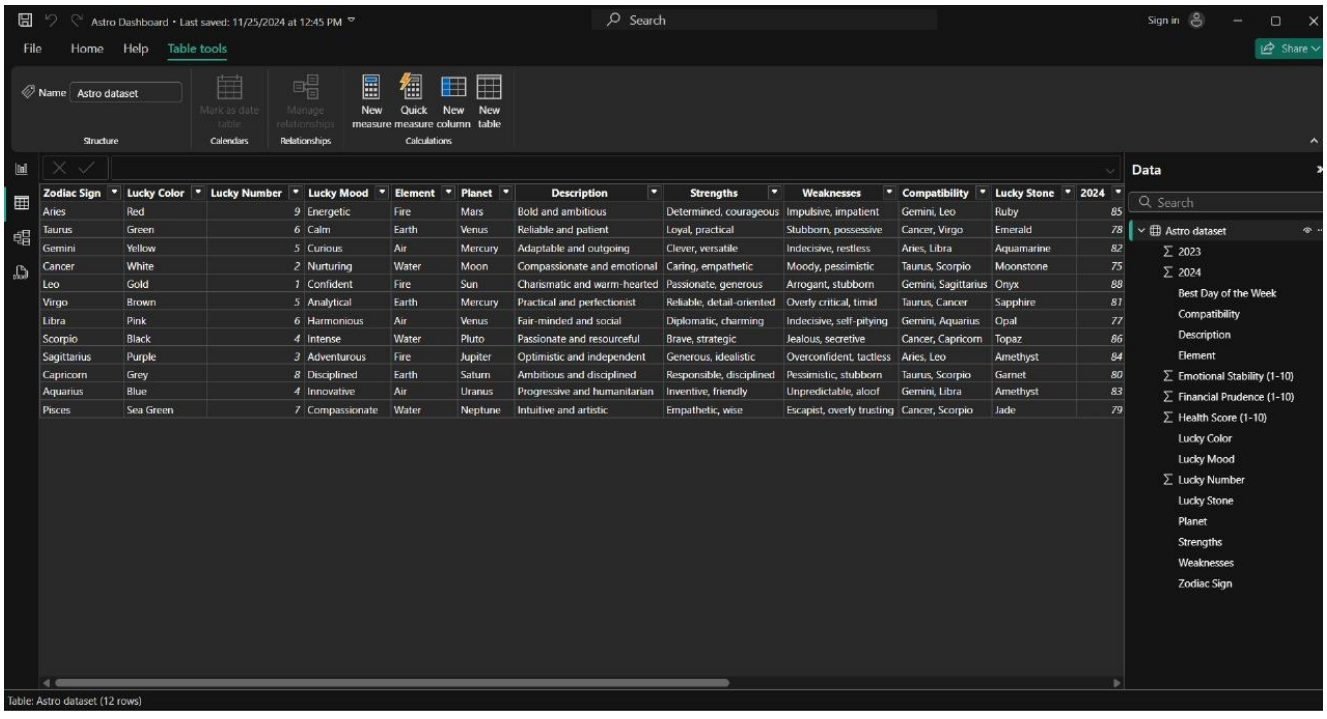
[40]:
```
def predict_zodiac(month, day):
    # Validate the month
    if not (1 <= month <= 12):
        return "Error: Month must be between 1 and 12."

    # Validate the day based on the month
    days_in_month = calendar.monthrange(2024, month)[1]  # 2024 is a leap year
    if not (1 <= day <= days_in_month):
        return f"Error: Day must be between 1 and {days_in_month} for the selected month."


    input_data = pd.DataFrame({"Month": [month], "Day": [day]})
    input_scaled = scaler.transform(input_data)
    prediction = model.predict(input_scaled)
    zodiac_sign = label_encoder.inverse_transform(prediction)
    return f"✨ Your Zodiac Sign is: **{zodiac_sign[0]}** ✨"


# Create the Gradio interface with customized layout
interface = gr.Interface(
    fn=predict_zodiac,
    inputs=[gr.Number(label="Month", minimum=1, maximum=12),
            gr.Number(label="Day", minimum=1, maximum=31)],
    outputs=gr.Textbox(label="Zodiac Sign"),
    title="Zodiac Sign Predictor 🌠",
    description="""Curious about your zodiac sign?
                   Simply enter your **birth month and day** to find out! 🌟""",

)

interface.launch()
```

```
* Running on local URL:  http://127.0.0.1:7860
```

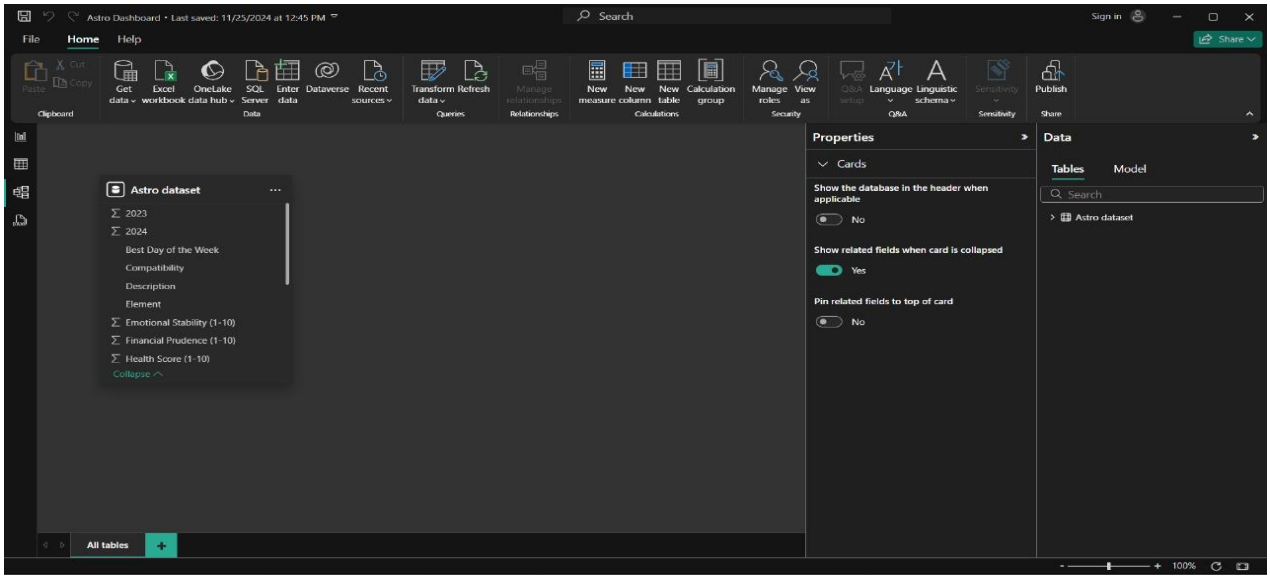


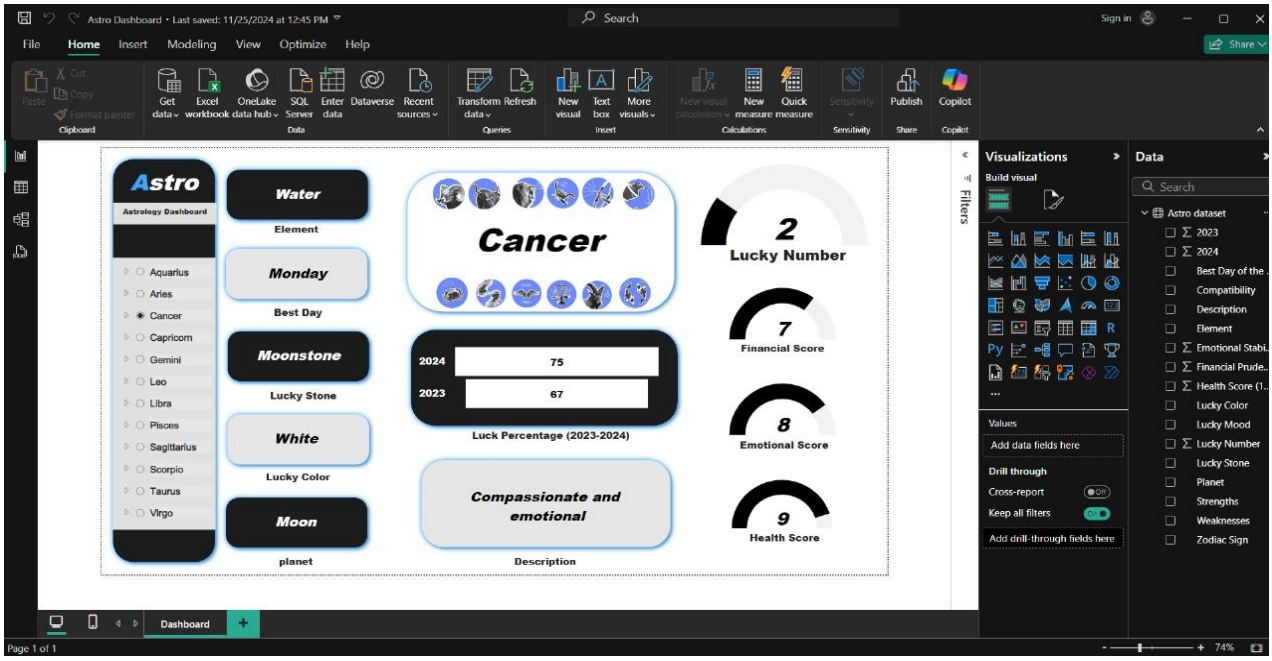
VISUALIZATION (POWER BI):





Zodiac Sign	Lucky Color	Lucky Number	Lucky Mood	Element	Planet	Description	Strengths	Weaknesses	Compatibility	Lucky Stone	2024
Aries	Red	9	Energetic	Fire	Mars	Bold and ambitious	Determined, courageous	Impulsive, impatient	Gemini, Leo	Ruby	85
Taurus	Green	6	Calm	Earth	Venus	Reliable and patient	Loyal, practical	Stubborn, possessive	Cancer, Virgo	Emerald	78
Gemini	Yellow	5	Curious	Air	Mercury	Adaptable and outgoing	Clever, versatile	Indecisive, restless	Aries, Libra	Aquamarine	82
Cancer	White	2	Nurturing	Water	Moon	Compassionate and emotional	Caring, empathetic	Moody, pessimistic	Taurus, Scorpio	Moonstone	75
Leo	Gold	1	Confident	Fire	Sun	Charismatic and warm-hearted	Passionate, generous	Arrogant, stubborn	Gemini, Sagittarius	Onyx	88
Virgo	Brown	5	Analytical	Earth	Mercury	Practical and perfectionist	Reliable, detail-oriented	Overly critical, timid	Taurus, Cancer	Sapphire	81
Libra	Pink	6	Harmonious	Air	Venus	Fair-minded and social	Diplomatic, charming	Indecisive, self-pitying	Gemini, Aquarius	Opal	77
Scorpio	Black	4	Intense	Water	Pluto	Passionate and resourceful	Brave, strategic	Jealous, secretive	Cancer, Capricorn	Topaz	86
Sagittarius	Purple	3	Adventurous	Fire	Jupiter	Optimistic and independent	Generous, idealistic	Overconfident, tactless	Aries, Leo	Amethyst	84
Capricorn	Grey	8	Disciplined	Earth	Saturn	Ambitious and disciplined	Responsible, disciplined	Pessimistic, stubborn	Taurus, Scorpio	Garnet	80
Aquarius	Blue	4	Innovative	Air	Uranus	Progressive and humanitarian	Inventive, friendly	Unpredictable, aloof	Gemini, Libra	Amethyst	83
Pisces	Sea Green	7	Compassionate	Water	Neptune	Intuitive and artistic	Empathetic, wise	Escapist, overly trusting	Cancer, Scorpio	Jade	79





Step7: Report Preparation and Presentation

Document the methodology, findings, and key insights in the project report.

Use Power BI dashboards and Python-generated charts to support the narrative.

Prepare the report for submission or presentation, ensuring clarity and coherence.

This workflow ensures an organized approach to the project, leading from data collection to a comprehensive analysis and visualization.

Project Code

About Dataset

```
import pandas as pd
import numpy as np
import plotly.express as px
data = pd.read_excel("C:/Users/AMAN/Desktop/Astro/Zodiac Dataset.xlsx")
print("Shape of dataset :- ",data.shape)
print("size of dataset :- ",data.size)
data.columns
data.head()
data.info()
data.count()
data.describe()
```

Data Cleaning

```
data.isnull().sum()
x = data['Lucky Number'].median()
print("median of Lucky Number column - ",x)
data["Lucky Number"].fillna(x, inplace = True)
data['Lucky Color'].fillna(method='ffill',inplace=True)
data['Mood'].fillna(method='ffill',inplace=True)
data.drop(columns=['Description'],inplace=True)
data.isnull().sum()
data.duplicated()
data.drop_duplicates(inplace = True)
data.duplicated()
```

Insights using Visualization

```
group = data.groupby('Zodiac Sign')['Name'].count()
print("Count of Person in Dataset With Same ZodicSigns",group)
group = group.reset_index()
group.columns = ['Zodiac Sign', 'Count']
fig = px.bar(
    group,
    x='Zodiac Sign',
    y='Count',
    title='Count of People with the Same Zodiac Sign',
color='Zodiac Sign'
)
fig.update_layout(height=600, xaxis_title="Zodiac Sign", yaxis_title="Count")
fig.show()
group = data.groupby('Hobby')['Name'].count()
print("Count of Person in Dataset With Same Hobby",group)
fig = px.pie(
    data,
    names='Mood',
    title="Mood Distribution"
)
fig.update_layout(height=600)
fig.show()
fig = px.box(
    data,
    x='Zodiac Sign',
    y='Lucky Number',
    title="Lucky Numbers by Zodiac Sign",
```

```
color='Zodiac Sign'
)
fig.update_layout(height=600)
fig.show()
fig = px.histogram(
    data,
    x='Occupation',
    title="Occupation Trends",
color='Occupation',
    labels={'Occupation': 'Occupation', 'count': 'Frequency'}
)
fig.update_xaxes(categoryorder='total descending')
fig.update_layout(height=600)
fig.show()
fig = px.pie(
    data,
    names='Favorite Cuisine',
    title="Favorite Cuisine Distribution",
    hole=0.3
)
fig.update_layout(height=600)
fig.show()
world = data.groupby('Favorite Cuisine')['Name'].count()
print("Count of Person in Dataset With Same FavoriteCuisine",world)
```

Model Building

```
import pandas as pd
from sklearn.ensemble import RandomForestClassifier
from sklearn.preprocessing import LabelEncoder
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler
from sklearn.metrics import accuracy_score,classification_report
import gradio as gr
import calendar
data['Date of Birth'] = pd.to_datetime(data['Date of Birth'])
data['Month'] = data['Date of Birth'].dt.month
data['Day'] = data['Date of Birth'].dt.day
data['Month']
data['Day']
label_encoder = LabelEncoder()
data['Zodiac Label'] = label_encoder.fit_transform(data['Zodiac Sign'])
X = data[['Month', 'Day']]
y = data['Zodiac Label']
X.head()
y.head()
scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)
X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)
model = RandomForestClassifier(random_state=42)
model.fit(X_train, y_train)
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)
print("Model Accuracy:", accuracy * 100, "%")
```

Model User Interface

```
def predict_zodiac(month, day):
    if not (1 <= month <= 12):
        return "Error: Month must be between 1 and 12.", None
days_in_month = calendar.monthrange(2024, month)[1]
    if not (1 <= day <= days_in_month):
        return f"Error: Day must be between 1 and {days_in_month} for the selected month.", None
input_data = pd.DataFrame({"Month": [month], "Day": [day]})
input_scaled = scaler.transform(input_data)
    prediction = model.predict(input_scaled)
zodiac_sign = label_encoder.inverse_transform(prediction)[0]
    attributes = zodiac_attributes.get(zodiac_sign, {})
    mood = attributes.get('Mood', "Unknown mood")
    return f"✨ Your Zodiac Sign is: **{zodiac_sign}** ✨", mood
interface = gr.Interface(
fn=predict_zodiac,
    inputs=[gr.Number(label="Month", minimum=1, maximum=12),
gr.Number(label="Day", minimum=1, maximum=31)],
    outputs=[
gr.Textbox(label="Zodiac Sign"),
gr.Textbox(label="Mood"),
    ],
    title="Zodiac Sign Predictor 🔮",
    description="""Curious about your zodiac sign?
                Simply enter your **birth month and day** to find out! 🌟
                Ensure the date is valid based on the month.""",
)
interface.launch()
```

FUTURE SCOPE OF THE PROJECT

1. **Enhanced Data Features**: The model can be extended to consider additional features, such as the time of day, geographical location, or planetary positions, to provide more personalized astrology predictions beyond just the date of birth.
2. **Mood Prediction Improvements**: Further research can be conducted to improve mood prediction by incorporating psychological and emotional factors based on the user's date of birth, incorporating astrological elements like moon signs or rising signs.
3. **Deep Learning Integration**: Future versions of the project could explore integrating deep learning techniques, such as neural networks, to improve accuracy and handle more complex relationships in the data, like incorporating multiple aspects of astrology.
4. **Mobile App Development**: Expanding the project to mobile platforms could increase accessibility, allowing users to interact with the system on-the-go. The integration of push notifications for daily predictions or insights could also enhance user engagement.
5. **Data Expansion**: The dataset could be expanded by including a broader set of data points, such as user feedback or self-reported moods, to refine the model's predictions and make them more dynamic and relevant over time.
6. **Personalized Recommendations**: The application could provide personalized astrological advice or daily horoscopes based on the user's zodiac sign and mood prediction, increasing the overall value of the application.

CONCLUSION

The Zodiac Sign Predictor project successfully demonstrates how data science and technology can be utilized to analyze and predict zodiac signs based on birth dates. Through the combination of Python programming, data analysis, and Power BI visualizations, this project not only provides accurate predictions but also delivers insightful patterns and trends related to zodiac signs.

Key achievements of the project include:

- A robust algorithm for predicting zodiac signs based on date ranges.
- Clear and interactive visualizations showcasing the distribution and trends of zodiac signs.
- Seamless integration of Python and Power BI to make the analysis both technical and user-friendly.

This project highlights the potential of integrating astrology with modern data analytics, offering both practical and engaging insights. With further enhancements, such as advanced machine learning models, interactive applications, or enriched datasets, this project can evolve into a comprehensive and widely applicable tool.

Overall, the Zodiac Sign Predictor serves as an excellent example of blending traditional concepts with contemporary technology to deliver meaningful results.

BIBLIOGRAPHY

1. **Tools and Technologies:**
   - **Python Software Foundation(2024):** Python Programming Language. Retrieved from https://www.python.org
   - **Jupyter. (2024):** Project Jupyter. Retrieved from https://jupyter.org
   - **Microsoft. (2024):** Power BI. Retrieved from https://powerbi.microsoft.com
2. **Online References:**
   - **Zodiac Sign Date Ranges (2024):** Retrieved from https://www.astrology.com
   - **Data Visualization Tutorials (2024):** Retrieved from https://towardsdatascience.com
3. **Datasets:**
   - **People's Dataset from Kaggle:**https:/www.kaggle.com
4. **Additional References:**
   - **Stack Overflow. (2024):** Various solutions for Python programming challenges. Retrieved from https://stackoverflow.com