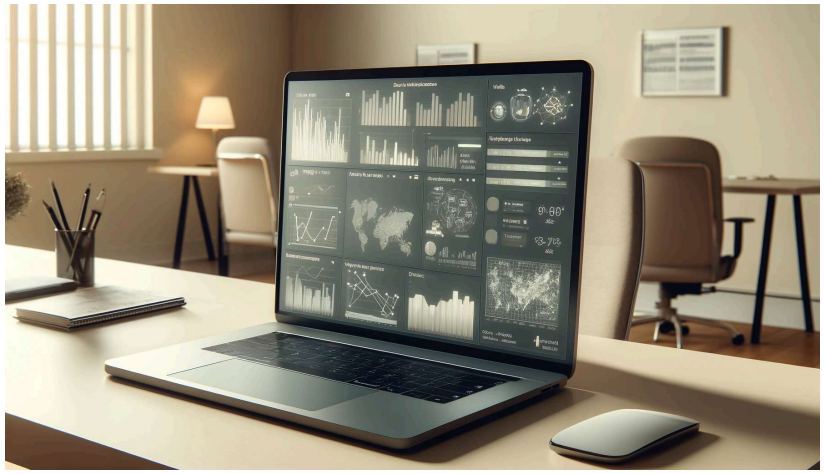
Use natural language to create charts and graphs

Estimated time needed: **30** minutes

Overview

Imagine you are a data analyst or a data scientist of a marketing team at an e-commerce company. The company needs to understand customer purchasing behaviors over the last year to tailor their upcoming holiday campaigns. Traditionally, this would involve complex SQL queries, data wrangling in Python, and perhaps building visual dashboards to interpret the results including analyzing spreadsheets, creating charts, and maybe even some statistical analysis—tasks that require considerable time and expertise.

With the integration of Langchain and LLMs, you can simply ask, "Show me a visualization of monthly sales trends by product category," or "Generate a heatmap of customer activity by region." The system would use the `create_pandas_dataframe_agent` to process the CSV data, and then dynamically generate visualizations such as line graphs, bar charts, or heatmaps in response to these queries. This not only speeds up the data analysis process but also allows team members who may not be tech-savvy to engage directly with the data and make informed decisions quickly. This approach fosters a more collaborative environment and ensures that strategic decisions are backed by real-time data insights, visually represented for easy comprehension.



In this lab, you will learn how to seamlessly integrate data visualization into your conversational data analysis using Langchain and LLMs. Starting with CSV file data, you will use the `create_pandas_dataframe_agent` to build an interactive agent that not only understands and responds to your queries but also translates data responses into visual formats. You will explore how to dynamically generate charts, graphs, and heatmaps directly in response to natural language questions. This capability will enable you to visualize trends, compare figures, and spot patterns immediately, making your data analysis workflow both efficient and visually engaging. By the end of this project, you will have the skills to create a data conversational agent that acts as both analyst and visualizer, bringing data to life through dialogue.

In this lab we are going to use Llama 3.3 LLM hosted on the IBM watsonx.ai platform.

Table of contents

Objectives

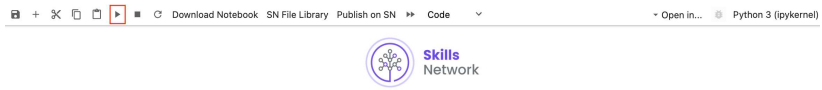
After completing the project, you should be able to:

- **Use Langchain with large language models**: Understand and apply the Langchain framework in conjunction with LLMs to interact with and analyze data stored in CSV files through natural language processing.
- **Create conversational data agents**: Build a conversational agent that can understand and respond to natural language queries about data, enabling users to ask questions directly and receive immediate answers.
- **Implement data visualization through dialogue**: Integrate data visualization tools within your conversational agent, allowing you to request and generate visual data representations such as graphs, charts, and heatmaps dynamically based on your queries.
- **Enhance decision-making process**: Develop the capability to derive actionable insights from data via interactive dialogues and visual outputs, thereby improving the decision-making process and making data analysis accessible to non-technical stakeholders.

Setup

This project is based on Jupyter Notebook. If you're not familiar with it, here's a quick guide on how to run code within it:

A Jupyter Notebook consists of cells. To execute a code cell, click on the cell that you want to run and click the 'Run' button, as shown in the picture.



For this lab, we will be using the following libraries:

- `ibm-watson-ai` for using LLMs from IBM's watsonx.ai.
- `LangChain`, `langchain-ibm`, `langchain-experimental` for using its agent function to interact with data.
- `matplotlib` for additional plotting tools.
- `seaborn` for visualizing the data.

Installing required libraries

The following required libraries are **not** preinstalled in the Skills Network Labs environment. **You must run the following cell** to install them:

Note: We are pinning the version here to specify the version. It's recommended that you do this as well. Even if the library is updated in the future, the installed library could still support this lab work.

This might take approximately 1-2 minutes.

As we use `%%capture` to capture the installation, you won't see the output process. But after the installation completes, you will see a number beside the cell.

In [6]:
```
%%capture
!pip install ibm-watsonx-ai=="0.2.6"
!pip install --user langchain=="0.1.16"
!pip install --user langchain-ibm=="0.1.4"
!pip install --user langchain-experimental=="0.0.57"
!pip install matplotlib=="3.8.4"
!pip install seaborn=="0.13.2"
```

After you installat the libraries, restart your kernel. You can do that by clicking the **Restart the kernel** icon.

Importing required libraries

We recommend you import all required libraries in one place (here):

In [1]:
```
# We use this section to suppress warnings generated by your code:
def warn(*args, **kwargs):
    pass
import warnings
warnings.warn = warn
warnings.filterwarnings('ignore')

from ibm_watsonx_ai.foundation_models import Model
from ibm_watsonx_ai.metanames import GenTextParamsMetaNames as GenParams
from ibm_watson_machine_learning.foundation_models.extensions.langchain import WatsonxLLM

from langchain_experimental.agents.agent_toolkits import create_pandas_dataframe_agent

import matplotlib.pyplot as plt
import pandas as pd
```

Data set

In this lab, you will work on the Student Alcohol Consumption data set `student-mat.csv` by UCI Machine Learning as an example. For more information, see Kaggle. It is based on data collected from two secondary schools in Portugal. The students included in the survey were in mathematics and Portuguese courses.

The data set we are using is for the mathematics course. The number of mathematics students involved in the collection was 395. The data collected in locations such as Gabriel Pereira and Mousinho da Silveira includes several pertinence values. Examples of such data are records of demographic information, grades, and alcohol consumption.

Field	Description
school	GP/MS for the student's school
sex	M/F for gender
age	15-22 for the student's age
address	U/R for urban or rural, respectively
famsize	LE3/GT3 for less than or greater than three family members
Pstatus	T/A for living together or apart from parents, respectively
Medu	0 (none) / 1 (primary-4th grade) / 2 (5th - 9th grade) / 3 (secondary) / 4 (higher) for mother's education
Fedu	0 (none) / 1 (primary-4th grade) / 2 (5th - 9th grade) / 3 (secondary) / 4 (higher) for father's education
Mjob	'teacher,' 'health' care related, civil 'services,' 'at_home' or 'other' for the student's mother's job
Fjob	'teacher,' 'health' care related, civil 'services,' 'at_home' or 'other' for the student's father's job
reason	reason to choose this school (nominal: close to 'home', school 'reputation', 'course' preference or 'other')
guardian	mother/father/other as the student's guardian
traveltime	1 (<15mins) / 2 (15 - 30 mins) / 3 (30 mins - 1 hr) / 4 (>1hr) for a time from home to school
studytime	1 (<2hrs) / 2 (2 - 5hrs) / 3 (5 - 10hrs) / 4 (>10hrs) for weekly study time
failures	1-3/4 for the number of class failures (if more than three, then record 4)
schoolsup	yes/no for extra educational support
famsup	yes/no for family educational support
paid	yes/no for extra paid classes for Math or Portuguese
activities	yes/no for extra-curricular activities

Field	Description
nursery	yes/no for whether attended nursery school
higher	yes/no for the desire to continue studies
internet	yes/no for internet access at home
romantic	yes/no for relationship status
famrel	1-5 scale on quality of family relationships
freetime	1-5 scale on how much free time after school
goout	1-5 scale on how much student goes out with friends
Dalc	1-5 scale on how much alcohol consumed on weekdays
Walc	1-5 scale on how much alcohol consumed on the weekend
health	1-5 scale on health condition
absences	0-93 number of absences from school
G1	0-20 for the first-period grade
G2	0-20 for the second-period grade
G3	0-20 for the final grade

Load the data set

Execute the code in the following cell to load your dataset. This code reads the CSV file into a pandas DataFrame, making the data accessible for processing in Python.

In [2]:
```
df = pd.read_csv(
    "https://cf-courses-data.s3.us.cloud-object-storage.appdomain.cloud/ZNoKMJ9rssJn-QbJ49kOzA/student-mat.csv"
)
```

Let's examine the first five rows of the dataset to get a glimpse of the data structure and its contents.

In [3]:
```
df.head(5)
```

Out[3]:

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob	...	famrel	freetime	goout	Dalc	Walc	healt
0	GP	F	18	U	GT3	A	4	4	at_home	teacher	...	4	3	4	1	1	
1	GP	F	17	U	GT3	T	1	1	at_home	other	...	5	3	3	1	1	
2	GP	F	15	U	LE3	T	1	1	at_home	other	...	4	3	2	2	3	
3	GP	F	15	U	GT3	T	4	2	health	services	...	3	2	2	1	1	
4	GP	F	16	U	GT3	T	3	3	other	other	...	4	3	2	1	2	

5 rows × 33 columns

We can also review the detailed information for each column in the dataset, focusing on the presence of null values and the specific data types of each column.

In [4]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 395 entries, 0 to 394
Data columns (total 33 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   school      395 non-null    object
 1   sex         395 non-null    object
 2   age         395 non-null    int64 
 3   address     395 non-null    object
 4   famsize     395 non-null    object
 5   Pstatus     395 non-null    object
 6   Medu        395 non-null    int64 
 7   Fedu        395 non-null    int64 
 8   Mjob        395 non-null    object
 9   Fjob        395 non-null    object
 10  reason      395 non-null    object
 11  guardian    395 non-null    object
 12  traveltime  395 non-null    int64 
 13  studytime   395 non-null    int64 
 14  failures    395 non-null    int64 
 15  schoolsup   395 non-null    object
 16  famsup      395 non-null    object
 17  paid        395 non-null    object
 18  activities  395 non-null    object
 19  nursery     395 non-null    object
 20  higher      395 non-null    object
 21  internet    395 non-null    object
 22  romantic    395 non-null    object
 23  famrel      395 non-null    int64 
 24  freetime    395 non-null    int64 
 25  goout       395 non-null    int64 
 26  Dalc        395 non-null    int64 
 27  Walc        395 non-null    int64 
 28  health      395 non-null    int64 
 29  absences    395 non-null    int64 
 30  G1          395 non-null    int64 
 31  G2          395 non-null    int64 
 32  G3          395 non-null    int64 
dtypes: int64(16), object(17)
memory usage: 102.0+ KB
```

Load LLM

Execute the code in the cell below to load the llama-3-3-70b LLM model from watsonx.ai.

Additionally, we will configure the LLM to interact with data by integrating it with Langchain's `create_pandas_dataframe_agent`.

In [5]:
```
# Create a dictionary to store credential information
credentials = {
    "url"    : "https://us-south.ml.cloud.ibm.com"
}

# Indicate the model we would like to initialize. In this case, Llama 3 70B.
model_id    = 'meta-llama/llama-3-3-70b-instruct'

# Initialize some watsonx.ai model parameters
params = {
        GenParams.MAX_NEW_TOKENS: 256, # The maximum number of tokens that the model can generate in a single run.
        GenParams.TEMPERATURE: 0,   # A parameter that controls the randomness of the token generation. A lower value
    }
project_id  = "skills-network" # <--- NOTE: specify "skills-network" as your project_id
space_id    = None
verify      = False

# Launch a watsonx.ai model
model = Model(
    model_id=model_id,
    credentials=credentials,
    params=params,
    project_id=project_id,
    space_id=space_id,
    verify=verify
)

# Integrate the watsonx.ai model with the langchain framework
llm = WatsonxLLM(model = model)

agent = create_pandas_dataframe_agent(
    llm,
```

```
    df,
    verbose=False,
    return_intermediate_steps=True  # set return_intermediate_steps=True so that model could return code that it come
)
```

Interact with your data

Let's start with a simple interaction.

Ask LLM how many rows of data are in the CSV file.

In [6]:
```
response = agent.invoke("how many rows of data are in this file?")
```

In [7]:
```
response['output']
```

Out[7]:
```
'There are 395 rows of data in this file.'
```

From the output above, the model reports that there are 395 rows of data in the file.

Let's verify this count using Python code to ensure accuracy.

In [8]:
```
len(df)
```

Out[8]:
```
395
```

The row count matches and is correct!

Curious about the code the LLM generated and used to create this result?

Run the code in the cell below to reveal the underlying commands.

In [9]:
```
response['intermediate_steps'][-1][0].tool_input.replace('; ', '\n')
```

Out[9]:
```
'len(df)'
```

Surprisingly, the LLM uses the same code as we do.

Also, we could let LLM return some data that we are looking for based on the CSV file.

In [10]:
```
response = agent.invoke("Give me all the data where student's age is over 18 years old.")
```

In [11]:
```
print(response)
```

{'input': "Give me all the data where student's age is over 18 years old.", 'output': "The data where the student's a
ge is over 18 years old is as follows:\n   school sex age address famsize Pstatus Medu Fedu   Mjob   Fjob
\\\n127   GP   F   19   U   GT3   T   0   1   at_home   other   \n153   GP   M   19   U   G
T3   T   3   2 services   at_home   \n210   GP   F   19   U   GT3   T   3   3   other
other   \n247   GP   M   22   U   GT3   T   3   1 services services   \n257   GP   M   19   U
LE3   A   4   3 services   at_home   \n270   GP   F   19   U   GT3   T   3   3   other se
rvices   \n296   GP   F   19   U   GT3   T   4   4   health   other   \n304   GP   M   19
U   GT3   T   3", 'intermediate_steps': [(AgentAction(tool='python_repl_ast', tool_input="df[df['age'] > 1
8]", log="Thought: To get the data where the student's age is over 18 years old, I need to filter the dataframe based
on the 'age' column. I can use the pandas dataframe's filtering functionality to achieve this.\nAction: python_repl_a
st\nAction Input: df[df['age'] > 18]"),   school sex age address famsize Pstatus Medu Fedu   Mjob   Fjob
\

	school	sex	age	address	famsize	Pstatus	Medu	Fedu	Mjob	Fjob
127	GP	F	19	U	GT3	T	0	1	at_home	other
153	GP	M	19	U	GT3	T	3	2	services	at_home
210	GP	F	19	U	GT3	T	3	3	other	other
247	GP	M	22	U	GT3	T	3	1	services	services
257	GP	M	19	U	LE3	A	4	3	services	at_home
270	GP	F	19	U	GT3	T	3	3	other	services
296	GP	F	19	U	GT3	T	4	4	health	other
304	GP	M	19	U	GT3	T	3	3	other	other
306	GP	M	20	U	GT3	A	3	2	services	other
307	GP	M	19	U	GT3	T	4	4	teacher	services
308	GP	M	19	R	GT3	T	3	3	other	services
309	GP	F	19	U	LE3	T	1	1	at_home	other
310	GP	F	19	U	LE3	T	1	2	services	services
311	GP	F	19	U	GT3	T	2	1	at_home	other
312	GP	M	19	U	GT3	T	1	2	other	services
313	GP	F	19	U	LE3	T	3	2	services	other
314	GP	F	19	U	GT3	T	1	1	at_home	health
315	GP	F	19	R	GT3	T	2	3	other	other
336	GP	F	19	R	GT3	A	3	1	services	at_home
340	GP	F	19	U	GT3	T	2	1	services	services
350	MS	M	19	R	GT3	T	1	1	other	services
353	MS	M	19	R	GT3	T	1	1	other	other
370	MS	F	19	U	LE3	T	3	2	services	services
376	MS	F	20	U	GT3	T	4	2	health	other
383	MS	M	19	R	GT3	T	1	1	other	services
387	MS	F	19	R	GT3	T	2	3	services	other
390	MS	M	20	U	LE3	A	2	2	services	services
392	MS	M	21	R	GT3	T	1	1	other	other
394	MS	M	19	U	LE3	T	1	1	other	at_home

	reason	guardian	traveltime	studytime	failures	schoolsup	famsup
127	course	other	1	2	3	no	yes
153	home	mother	1	1	3	no	yes
210	reputation	other	1	4	0	no	yes
247	other	mother	1	1	3	no	no
257	reputation	mother	1	2	0	no	yes
270	home	other	1	2	2	no	yes
296	reputation	other	2	2	0	no	yes
304	home	other	1	2	1	no	yes
306	course	other	1	1	0	no	no
307	reputation	other	2	1	1	no	yes
308	reputation	father	1	2	1	no	no
309	reputation	other	1	2	1	yes	yes
310	home	other	1	2	1	no	no
311	other	other	3	2	0	no	yes
312	course	other	1	2	1	no	no
313	reputation	other	2	2	1	no	yes
314	home	other	1	3	2	no	no
315	reputation	other	1	3	1	no	no
336	home	other	1	3	1	no	no
340	home	other	1	3	1	no	no
350	home	other	3	2	3	no	no
353	home	other	3	1	1	no	yes
370	home	other	2	2	2	no	no
376	course	other	2	3	2	no	yes
383	other	mother	2	1	1	no	no
387	course	mother	1	3	1	no	no
390	course	other	1	2	2	no	yes
392	course	other	1	1	3	no	no
394	course	father	1	1	0	no	no

	paid	activities	nursery	higher	internet	romantic	famrel	freetime	goout
127	no	no	no	no	no	no	3	4	2
153	no	no	yes	no	yes	yes	4	5	4
210	yes	yes	yes	yes	yes	no	4	3	3
247	no	no	no	no	yes	yes	5	4	5
257	no	no	yes	yes	yes	no	4	3	1
270	yes	yes	yes	yes	yes	no	4	3	5
296	yes	yes	yes	yes	yes	no	2	3	4
304	no	yes	yes	yes	yes	yes	4	4	4
306	no	yes	yes	yes	no	no	5	5	3
307	yes	no	yes	yes	yes	yes	4	3	4

308	no	yes	yes	yes	no	yes	4	5	3
309	no	yes	no	yes	yes	no	4	4	3
310	no	yes	no	yes	no	yes	4	2	4
311	no	no	yes	no	yes	yes	3	4	1
312	no	no	no	yes	yes	no	4	5	2
313	yes	no	no	yes	yes	yes	4	2	2
314	no	no	no	yes	yes	yes	4	1	2
315	no	no	yes	yes	yes	yes	4	1	2
336	yes	no	yes	yes	no	no	5	4	3
340	yes	yes	yes	yes	yes	yes	4	3	4
350	no	no	yes	yes	yes	no	5	4	4
353	no	no	yes	yes	yes	no	4	4	4
370	no	yes	yes	yes	no	yes	3	2	2
376	yes	no	no	yes	yes	yes	5	4	3
383	no	no	yes	yes	no	no	4	3	2
387	no	yes	no	yes	yes	no	5	4	2
390	yes	no	yes	yes	no	no	5	5	4
392	no	no	no	yes	no	no	5	5	3
394	no	no	yes	yes	yes	no	3	2	3

	Dalc	Walc	health	absences	G1	G2	G3
127	1	1	5	2	7	8	9
153	1	1	4	0	5	0	0
210	1	2	3	10	8	8	8
247	5	5	1	16	6	8	8
257	1	1	1	12	11	11	11
270	3	3	5	15	9	9	9
296	2	3	2	0	10	9	0
304	1	1	3	20	15	14	13
306	1	1	5	0	17	18	18
307	1	1	4	38	8	9	8
308	1	2	5	0	15	12	12
309	1	3	3	18	12	10	10
310	2	2	3	0	9	9	0
311	1	1	2	20	14	12	13
312	2	2	4	3	13	11	11
313	1	2	1	22	13	10	11
314	1	1	3	14	15	13	13
315	1	1	3	40	13	11	11
336	1	2	5	12	14	13	13
340	1	3	3	4	11	12	11
350	3	3	2	8	8	7	8

```
353     3     3       5         4   8   8   8
370     1     1       3         4   7   7   9
376     1     1       3         4  15  14  15
383     1     3       5         0   6   5   0
387     1     2       5         0   7   5   0
390     4     5       4        11   9   9   9
392     3     3       3         3  10   8   7
394     3     3       5         5   8   9   9  )]}
```

Let's get the code LLM used for charting this plot.

In [12]:
```
response['intermediate_steps'][-1][0].tool_input.replace('; ', '\n')
```

Out[12]:
```
"df[df['age'] > 18]"
```

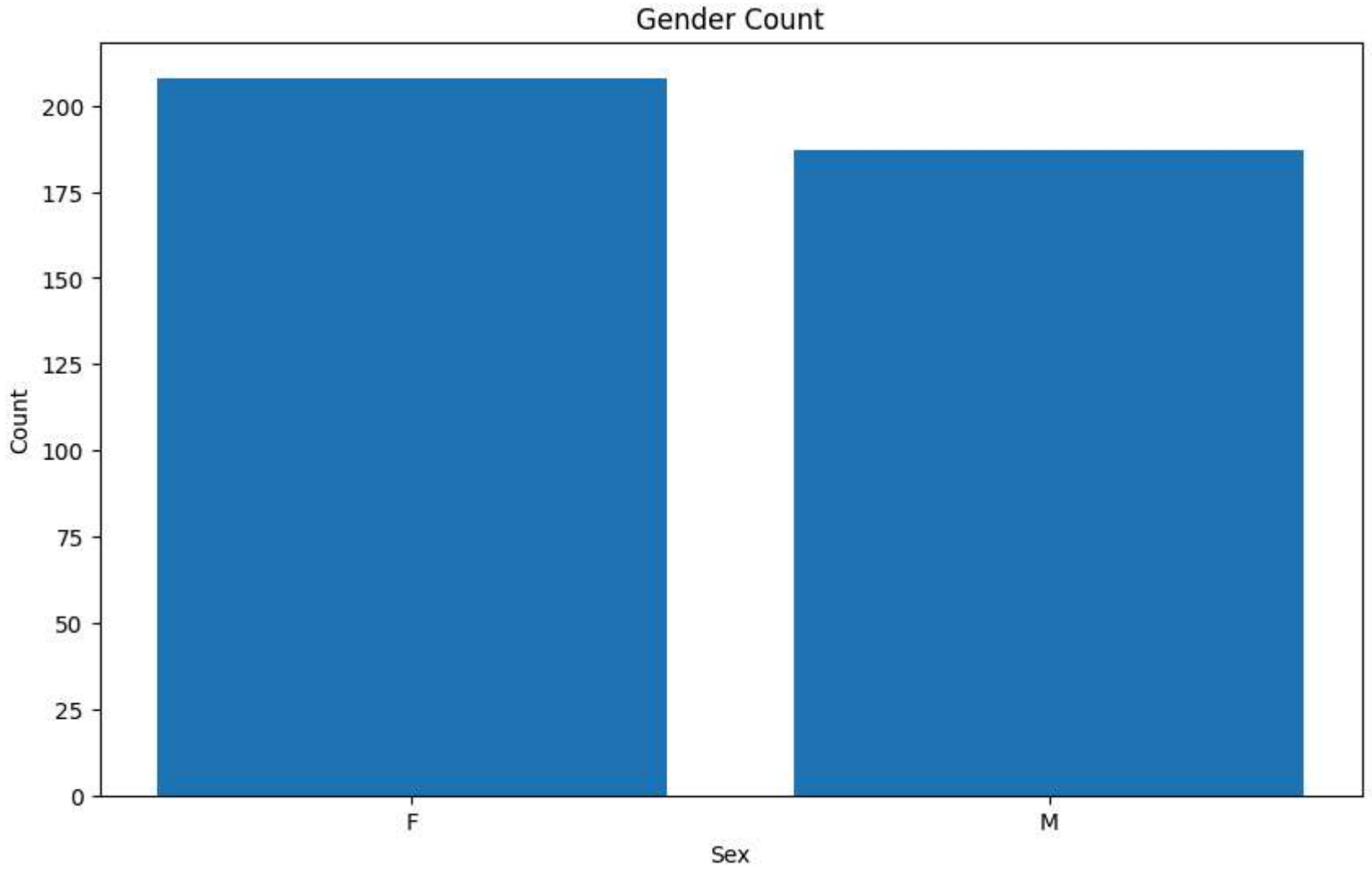
Plot your data with natural language

Task 1

Generating a first visual on the data set to know the total number of male and female students in the data set.

We just tell the agent that "Plot the gender count with bars."

In [13]:
```
response = agent.invoke("Plot the gender count with bars.")
```



Let's see what code the LLM generated for ploting this chart.

In [14]:
```
print(response['intermediate_steps'][-1][0].tool_input.replace('; ', '\n'))
```

```python
import matplotlib.pyplot as plt
gender_counts = df['sex'].value_counts()
plt.figure(figsize=(10,6))
plt.bar(gender_counts.index, gender_counts.values)
plt.xlabel('Sex')
plt.ylabel('Count')
plt.title('Gender Count')
plt.show()
```

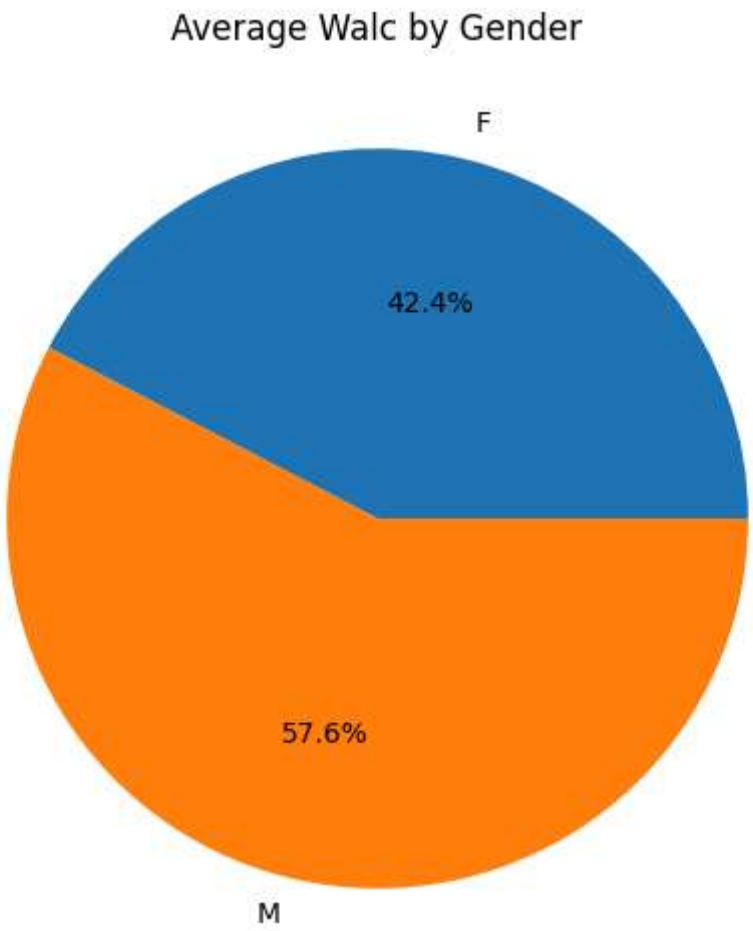
Task 2

Generating a pie chart for displaying the average value of weekend alcohol for each gender in the data set.

We will use the prompt "Generate a pie chart to display average value of Walc for each gender."

You may notice that the model generates two charts. The charts actually indicate the progressive improvement of the agent's code as it searches for the best way to answer to our prompt. It presents an improvement in the response to our query.

In [15]:
```python
response = agent.invoke("Generate a pie chart to display average value of Walc for each Gender.")
```



Let's get the code LLM used for charting this plot.

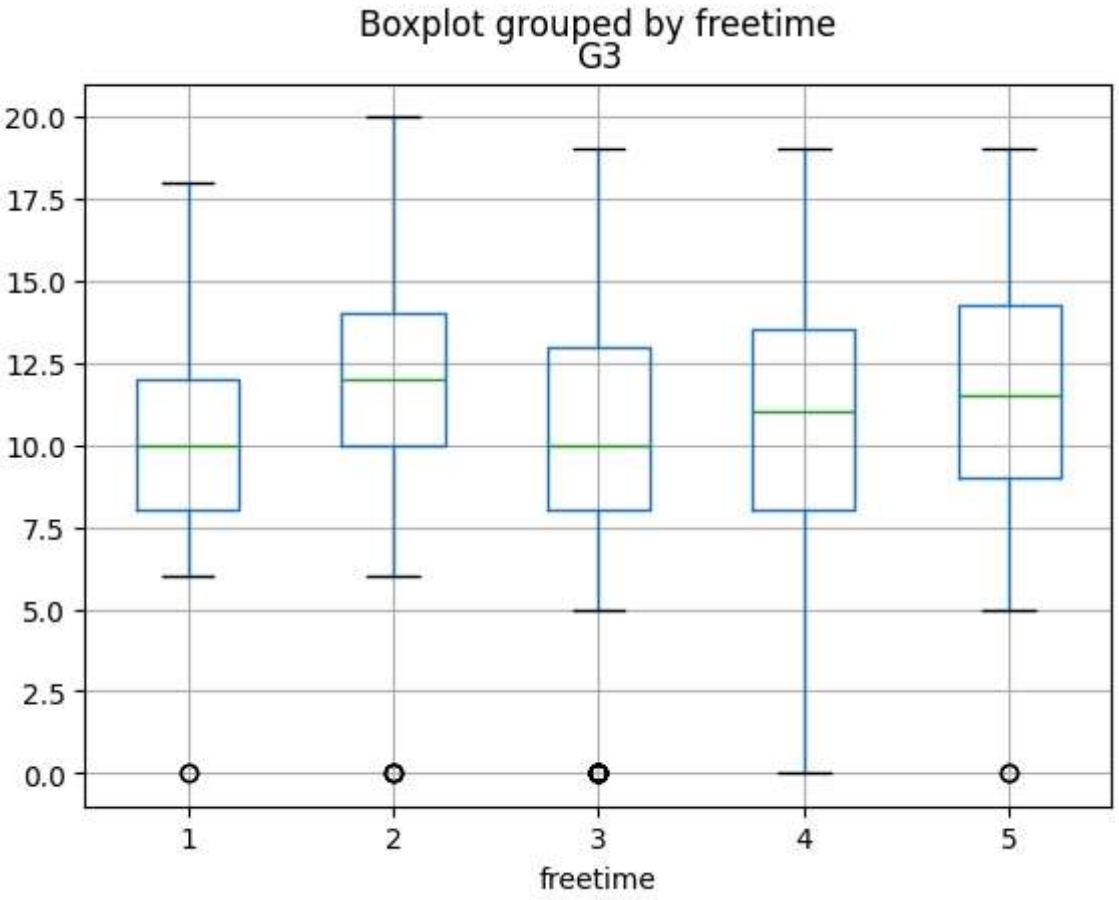
In [16]:
```
print(response['intermediate_steps'][-1][0].tool_input.replace('; ', '\n'))
```

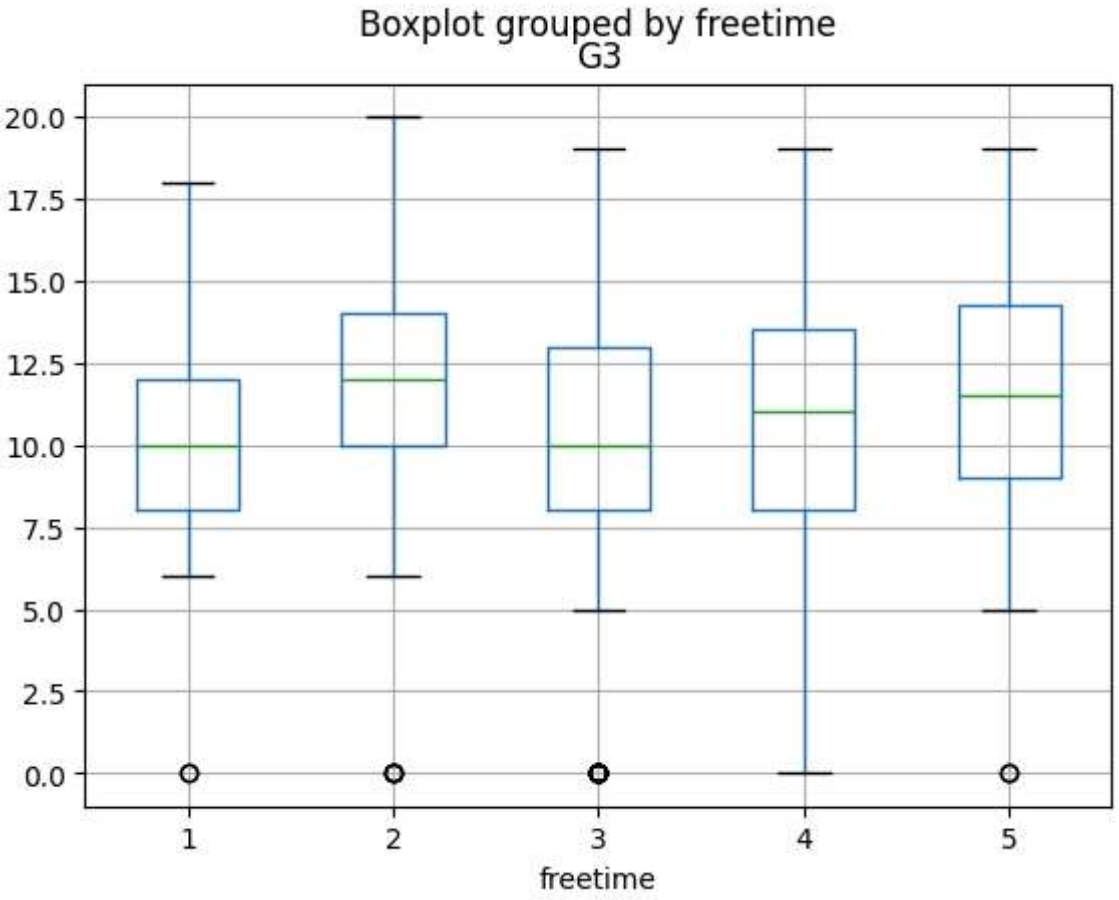
```python
import matplotlib.pyplot as plt

# Group by 'sex' and calculate the mean of 'Walc'
average_walc = df.groupby('sex')['Walc'].mean()

# Create a pie chart
plt.figure(figsize=(10,6))
plt.pie(average_walc, labels = average_walc.index, autopct='%1.1f%%')
plt.title('Average Walc by Gender')
plt.show()
```

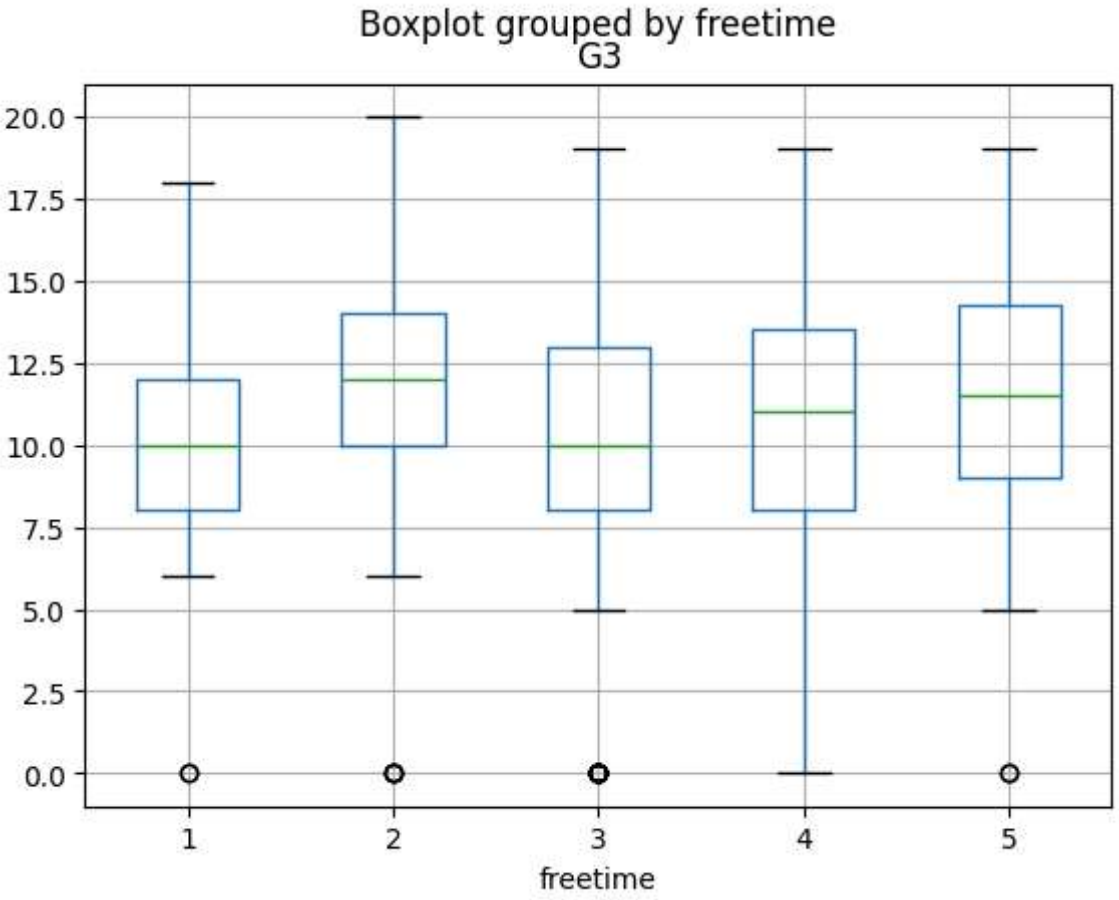
Task 3

We can explore the impact of free time on grades based on the data.

In [17]:
```python
response = agent.invoke("Create box plots to analyze the relationship between 'freetime' (amount of free time) and 'G
```


Boxplot grouped by freetime
G3
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
1
2
3
4
5
freetime


Boxplot grouped by freetime
G3
20.0
17.5
15.0
12.5
10.0
7.5
5.0
2.5
0.0
1
2
3
4
5
freetime



Execute the code below to retrieve the Python script the LLM used for plotting.

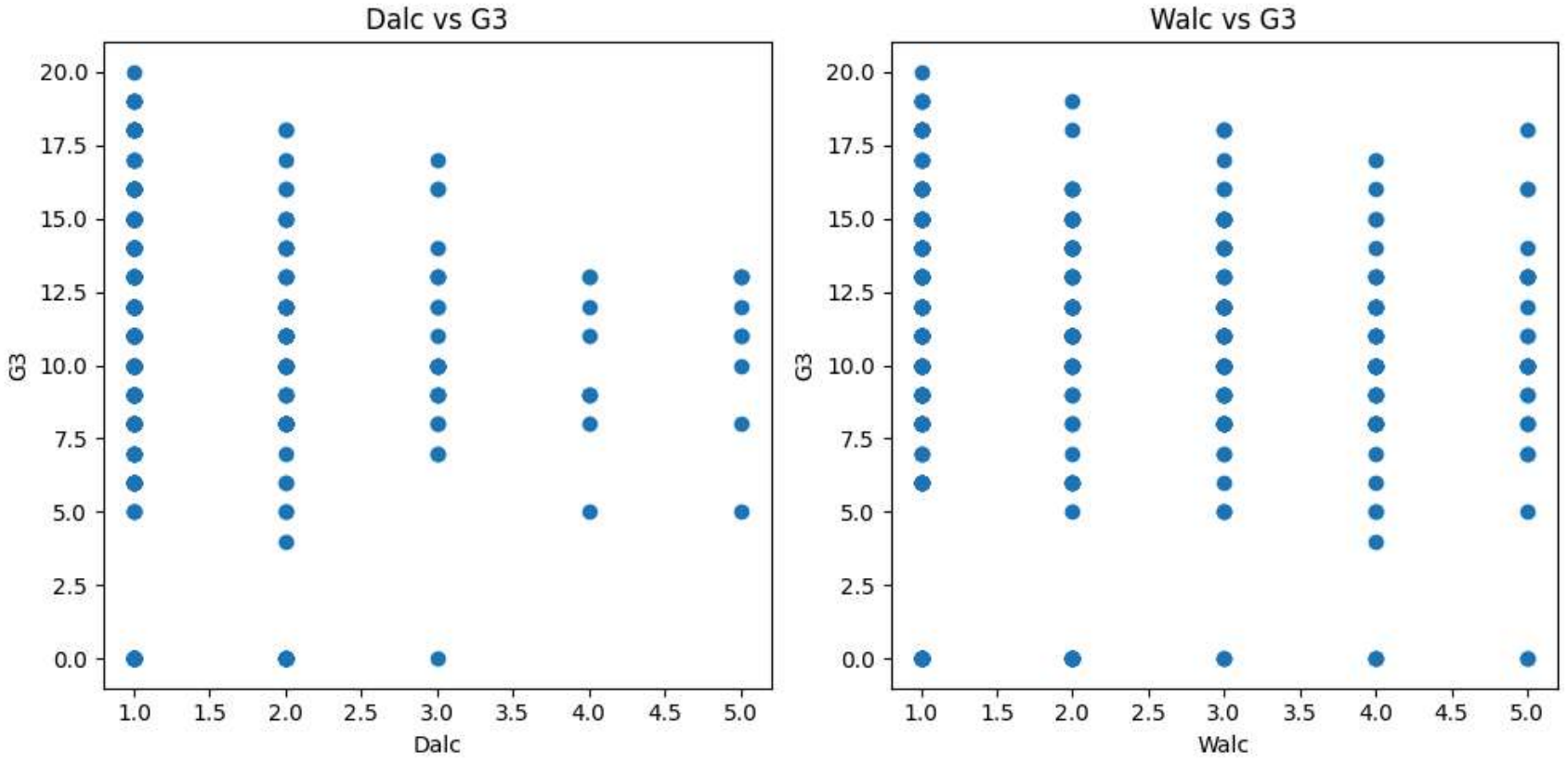
In [18]:
```
print(response['intermediate_steps'][-1][0].tool_input.replace('; ', '\n'))
```

```python
import pandas as pd
import matplotlib.pyplot as plt

# Assuming df is the DataFrame
df['freetime'] = pd.Categorical(df['freetime'])
correlation_coefficient = df['G3'].corr(df['freetime'].cat.codes)
print(correlation_coefficient)
```

Task 4

We can explore the effect of alcohol consumption on academic performance.

In [19]:
```
response = agent.invoke("Generate scatter plots to examine the correlation between 'Dalc' (daily alcohol consumption)
```



Execute the code below to retrieve the Python script the LLM used for plotting.

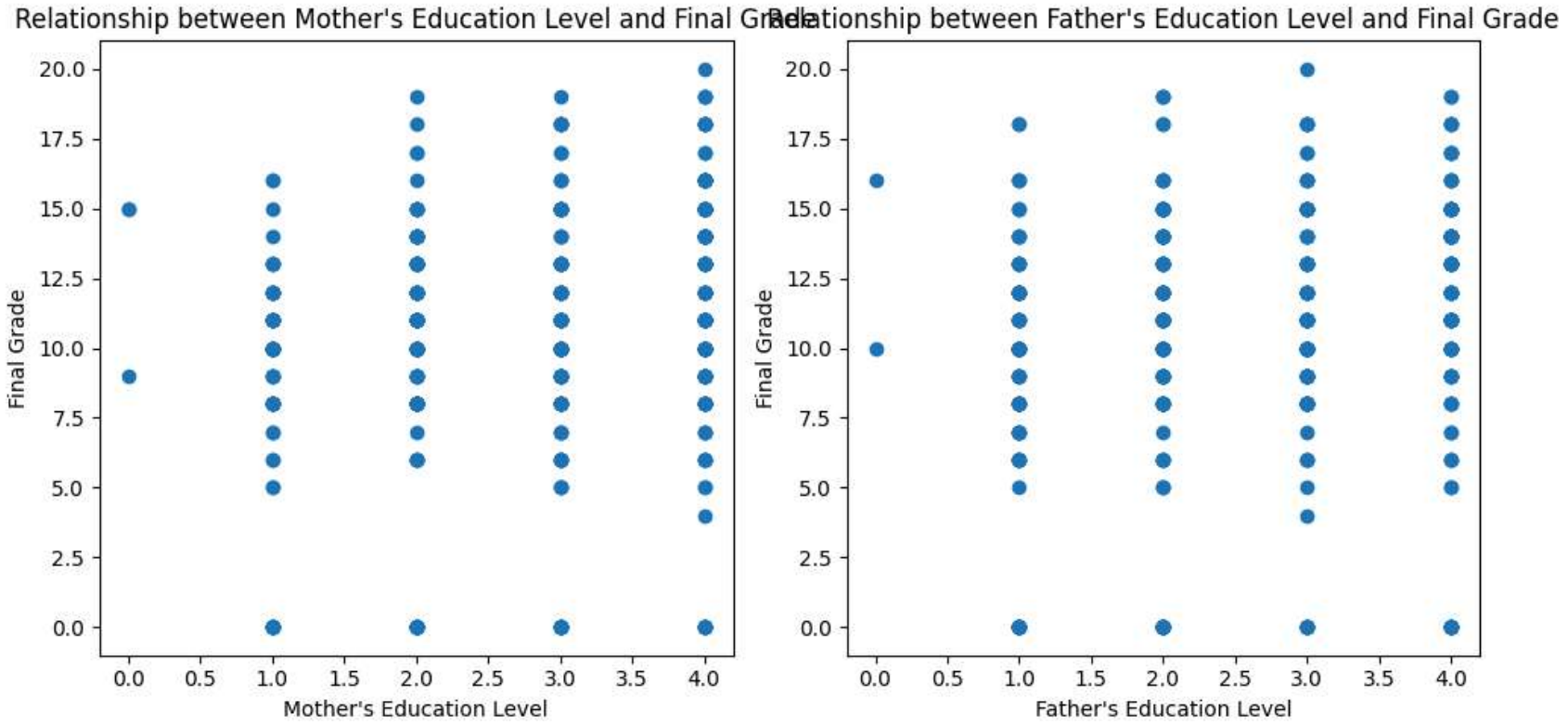
In [20]:
```
print(response['intermediate_steps'][-1][0].tool_input.replace('; ', '\n'))
```

```python
import pandas as pd
import matplotlib.pyplot as plt

# Assuming df is the DataFrame
plt.figure(figsize=(10,5))

plt.subplot(1, 2, 1)
plt.scatter(df['Dalc'], df['G3'])
plt.title('Dalc vs G3')
plt.xlabel('Dalc')
plt.ylabel('G3')

plt.subplot(1, 2, 2)
plt.scatter(df['Walc'], df['G3'])
plt.title('Walc vs G3')
plt.xlabel('Walc')
plt.ylabel('G3')

plt.tight_layout()
plt.show()
```

Exercises

Exercise 1 - Relationship between parental education level and student grades

In [21]:
```
# your code here
response = agent.invoke("Plot scatter plots to show the relationship between 'Medu' (mother's education level) and 'G
```



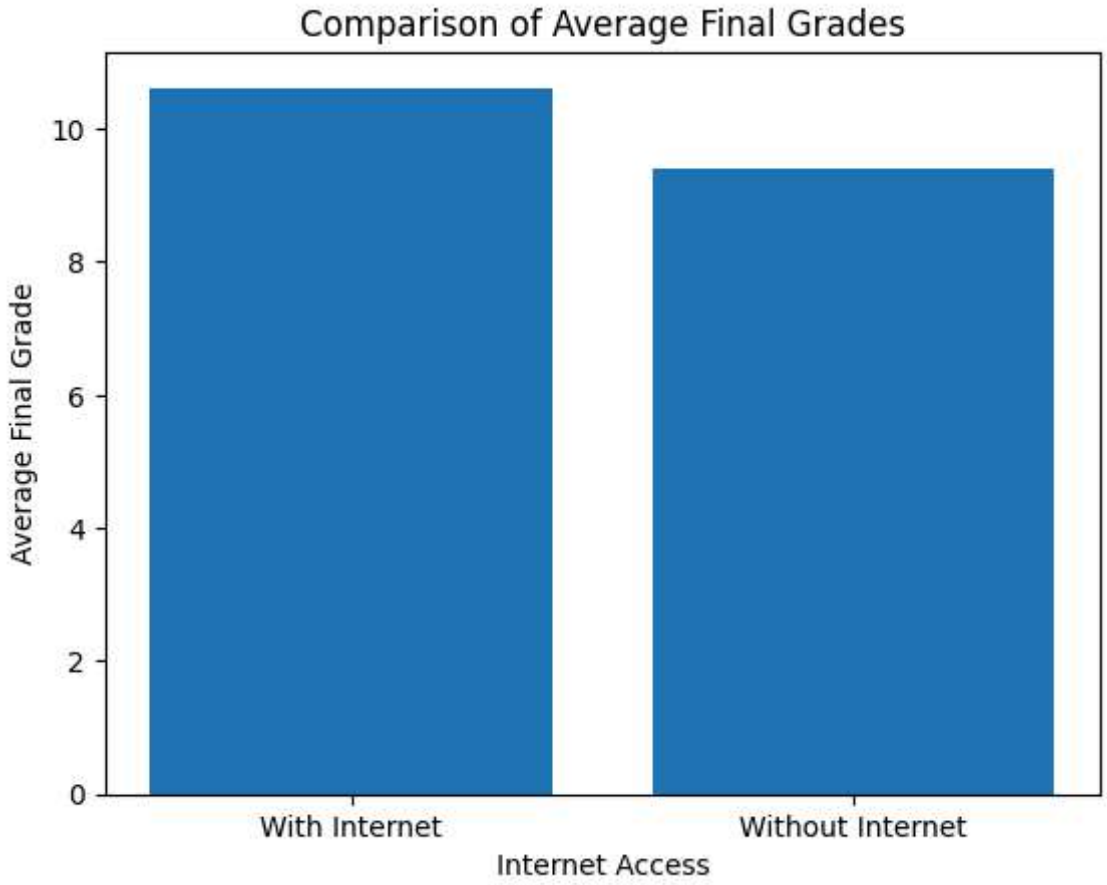
In [23]:
```
print(response['intermediate_steps'][-1][0].tool_input.replace('; ', '\n'))
```

```python
import matplotlib.pyplot as plt

plt.figure(figsize=(10,5))

plt.subplot(1, 2, 1)
plt.scatter(df['Medu'], df['G3'])
plt.xlabel('Mother\'s Education Level')
plt.ylabel('Final Grade')
plt.title('Relationship between Mother\'s Education Level and Final Grade')

plt.subplot(1, 2, 2)
plt.scatter(df['Fedu'], df['G3'])
plt.xlabel('Father\'s Education Level')
plt.ylabel('Final Grade')
plt.title('Relationship between Father\'s Education Level and Final Grade')

plt.tight_layout()
plt.show()
```

▶ Click here for Solution

Exercise 2 - Impact of internet access at home on grades

In [24]:
```python
# your code here

response = agent.invoke("Use bar plots to compare the average final grades ('G3') of students with internet access at
```




```
Average final grade with internet: 10.617021276595745
Average final grade without internet: 9.409090909090908
```

In [25]:
```
print(response['intermediate_steps'][-1][0].tool_input.replace('; ', '\n'))
```

```python
import matplotlib.pyplot as plt

# Filter the dataframe
with_internet = df[df['internet'] == 'yes']
without_internet = df[df['internet'] == 'no']

# Calculate the average final grades
avg_with_internet = with_internet['G3'].mean()
avg_without_internet = without_internet['G3'].mean()

# Print the average final grades
print("Average final grade with internet:", avg_with_internet)
print("Average final grade without internet:", avg_without_internet)

# Plot the results
plt.bar(['With Internet', 'Without Internet'], [avg_with_internet, avg_without_internet])
plt.xlabel('Internet Access')
plt.ylabel('Average Final Grade')
plt.title('Comparison of Average Final Grades')
plt.show()
```

▶ Click here for a solution

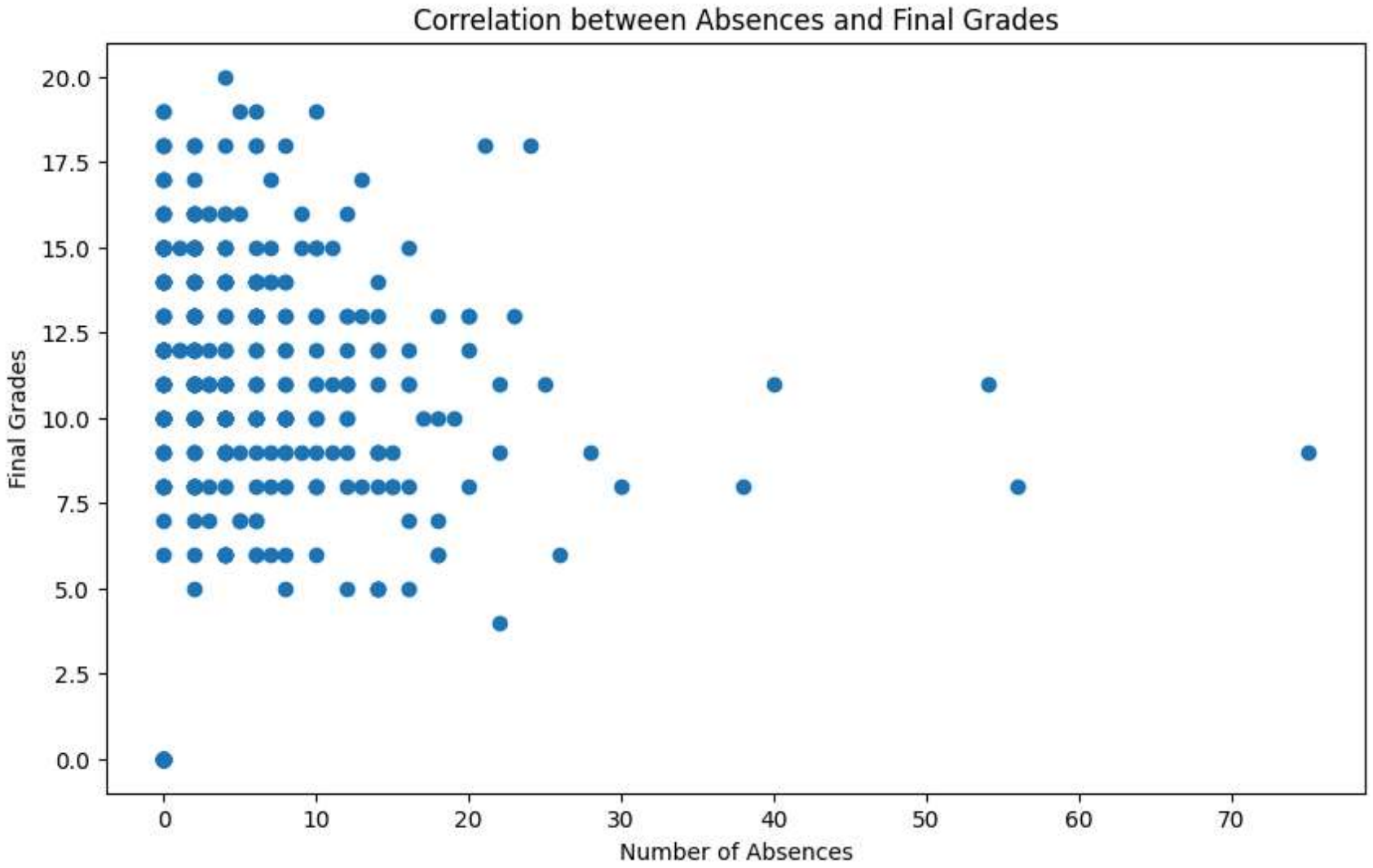
Exercise 3 - Explore LLM's code

Can you find what code the model used to generate the plot for exploring the relationship between absences and academic performance?

You could run the corresponding code and from the response chain, you could see the code used from charting.

In [26]:
```
# your code here
```

In [27]:
```
response = agent.invoke("Plot a scatter plot showing the correlation between the number of absences ('absences') and

for i in range(len(response['intermediate_steps'])):
    print(response['intermediate_steps'][i][0].tool_input.replace(';', '\n'))
```




```python
import matplotlib.pyplot as plt
plt.figure(figsize=(10,6))
plt.scatter(df['absences'], df['G3'])
plt.xlabel('Number of Absences')
plt.ylabel('Final Grades')
plt.title('Correlation between Absences and Final Grades')
plt.show()
```

▶ Click here for a solution

Authors

Kang Wang

Kang Wang is a Data Scientist in IBM. He is also a PhD Candidate in the University of Waterloo.

Wojciech Fulmyk
Wojciech "Victor" Fulmyk is a Data Scientist at IBM. He is also a PhD Candidate in Economics in the University of Calgary.

Other contributors

Ricky Shi
Ricky Shi is a data scientist at the Ecosystems Skills Network at IBM.

<!--## Change Log--!>

<!--|Date (YYYY-MM-DD)|Version|Changed By|Change Description| |-|-|-|-| |2024-05-10|0.2|Kang Wang & Wojciech Fulmyk|Initial version created| |2024-02-23|0.1|Elio Di Nino|Update library documentation|--!>