

BITS Pilani

API-driven Cloud Native Solutions (CCZG506)

Assignment - 1

Group -28

Submitted By

Student Name - ID

M KAMAL KUMAR

2023mt03153@wilp.bits-pilani.ac.in

SHANMUKHI LAKSHMI SIDWINI

2023mt03194@wilp.bits-pilani.ac.in

KAKANI VARSHITHA

2023mt03002@wilp.bits-pilani.ac.in

JALAMANCHILI RAMA SURYAM

2023mt03101@wilp.bits-pilani.ac.in

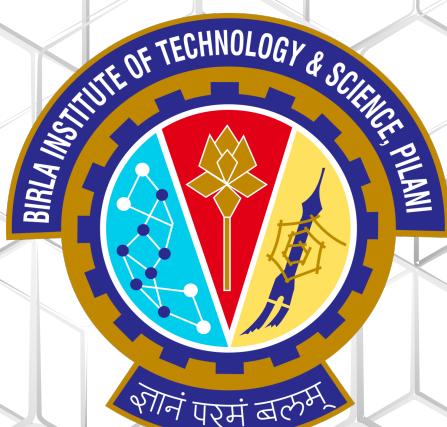
KAPADIYA KRUNALKUMAR GHANSHYAMBHAI

2023mt03176@wilp.bits-pilani.ac.in

Submitted To

Prof. Shreyas Suresh Rao

Oct 31, 2024



<u>SL.N O</u>	<u>BITS ID</u>	<u>NAME</u>	<u>CONTRIBUTION OF TEAM MEMBERS (Qualitative)</u>	<u>PERCENTA GE CONTRIBU TION (Out of 100) QUANTITA TIVE</u>
1	2023mt03 153	M KAMAL KUMAR	Datapipline,prefectexecution, Sagemaker Execution, Video making	30%
2	2023mt03 002	KAKANI VARSHITHA	MLflow execution, Documentation	20%
3	2023mt03 194	SHANMUKHI LAKSHMI SIDWINI	Dataset identification, Documentation	20%
4	2023mt03 176	KAPADIYA KRUNALKUM AR GHANSHYAM Bhai	Assignment flow planning, ML pipeline, ML models	15%
5	2023mt03 101	JALAMANCHI LI RAMA SURYAM	MLops execution, Documentation	15%

Sub-Objective 1: Design and Development of a Data Pipeline

Prerequisites: -

- 1) Python
- 2) Prefect Cloud
- 3) MLflows
- 4) SageMaker

1.1 Business Understanding: Identify a business problem where we analysed the Factors affecting the Student Performance in the area of data science.

Using cloud-based tools for data science and machine learning enables efficient, scalable analysis of datasets, such as examining factors impacting student performance. Here's how each tool can play a role in this process:

1. **Prefect Cloud:** Prefect allows for orchestrating and scheduling complex workflows. For this project, it can manage data ingestion, transformation, and preprocessing pipelines, automating steps like data cleaning, feature engineering, and model retraining. Prefect Cloud ensures workflows run reliably and are resilient to errors, with all runs and logs easily monitored.
2. **MLflow:** MLflow facilitates the tracking and management of machine learning experiments. You can log metrics, parameters, and models across various experiments and versions, making it easier to track model performance and hyperparameter tuning as you test different predictive approaches for student outcomes.
3. **Amazon SageMaker:** SageMaker enables the building, training, and deployment of machine learning models at scale. It provides pre-built algorithms and manages resources for training models quickly. For student performance prediction, SageMaker could deploy your model as a web service, allowing for real-time predictions, such as identifying at-risk students.
4. **Python:** Python is central to this process for data manipulation (using libraries like Pandas), exploratory data analysis (Matplotlib, Seaborn), feature engineering, and model building (Scikit-learn, TensorFlow, or PyTorch). Python scripts can be integrated into Prefect workflows and MLflow for full pipeline management.

Together, these tools streamline the end-to-end process—from data preprocessing to model deployment—offering a scalable, reproducible solution for analyzing factors that influence student academic success.

1.2 Data Ingestion: For the identified problem, we found an appropriate dataset from a public repository which is Kaggle.

The screenshot shows the Kaggle interface. On the left, there's a sidebar with navigation links: Create, Home, Competitions, Datasets (which is selected), Models, Code, Discussions, Learn, More, Your Work, and View Active Events. The main content area has a search bar at the top. Below it, there's a thumbnail for a notebook titled "PRACTICE DATA ANALYSIS WITH ME - UPDATED 2 MONTHS AGO" with 750 views, followed by a "New Notebook" and "Download" button. The main title "Student Performance Factors" is displayed in large bold letters, with a subtitle "Insights into Student Performance and Contributing Factors". Below the title is a photograph of three students in a classroom. A "Data Card" section includes tabs for Data Card, Code (127), Discussion (1), and Suggestions (3). To the right, there are sections for "Usability" (10.00), "License" (CC0: Public Domain), and "Expected update frequency" (Never).

ImputedStudentPerformanceFactors.csv: -

The screenshot shows a Microsoft Excel spreadsheet titled "ImputedStudentPerformanceFactors". The first few rows of data are visible, including columns such as Hours_Std, Attendance, Parental_Involvement, Access_to_Resources, Extracurricular_Activities, Sleep_Hours, Previous_Scores, Motivation_Level, Internet_Access, Tutoring_Sessions, Family_Income, Teacher_Quality, School_Type, Peer_Influence, Physical_Activity, Learning_Disabilities, Parental_Education_Level, Distance_from_Home, Gender, and Exam_Score. The data appears to be a tabular representation of the Kaggle dataset.

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T
1	Hours_Std	Attendance	Parental_Involvement	Access_to_Resources	Extracurricular_Activities	Sleep_Hours	Previous_Scores	Motivation_Level	Internet_Access	Tutoring_Sessions	Family_Income	Teacher_Quality	School_Type	Peer_Influence	Physical_Activity	Learning_Disabilities	Parental_Education_Level	Distance_from_Home	Gender	Exam_Score
2	23	84	1	0	0	7	73	1	1	0	1	2	1	2	3	0	1	2	1	67
3	19	64	1	2	0	8	59	1	1	2	2	2	1	0	4	0	0	1	0	61
4	24	98	2	2	1	7	91	2	1	2	2	2	1	1	4	0	2	2	1	74
5	29	89	1	2	1	8	98	2	1	1	2	2	1	0	4	0	1	1	1	71
6	19	92	2	2	1	6	65	2	1	3	2	0	1	1	4	0	0	2	0	70
7	19	88	2	2	1	8	89	2	1	3	2	2	1	2	3	0	2	2	1	71
8	29	84	2	1	1	7	68	1	1	1	1	2	0	1	2	0	1	1	1	67
9	25	78	1	0	1	6	50	2	1	1	0	0	1	0	2	0	1	0	1	66
10	17	94	2	0	0	6	80	0	1	0	2	1	0	1	1	0	0	2	1	69
11	23	98	2	2	1	8	71	2	1	0	0	0	1	2	5	0	1	1	1	72
12	17	80	1	0	0	8	88	2	0	4	2	0	0	1	4	0	0	1	1	68
13	17	97	2	0	1	6	87	1	1	2	1	0	0	1	2	0	1	2	1	71
14	21	83	2	2	1	8	97	1	1	2	2	2	1	2	4	0	1	2	1	70
15	9	82	2	2	1	8	72	2	1	2	2	2	0	2	3	0	2	2	1	66
16	10	78	2	0	1	8	74	2	1	1	2	0	1	4	0	2	2	1	65	
17	17	68	2	2	0	8	70	2	1	2	2	2	0	2	4	0	1	2	0	64
18	14	60	2	1	1	10	65	1	1	0	0	2	0	2	3	0	0	2	1	60
19	22	70	1	2	1	6	82	2	1	1	0	1	1	3	0	1	2	0	65	
20	15	80	2	2	1	9	91	1	1	3	1	2	1	2	2	0	0	1	0	67
21	12	75	2	0	1	7	58	2	1	3	2	2	0	2	4	0	0	2	1	66
22	29	78	2	2	0	5	99	0	1	0	0	2	1	0	1	0	1	1	0	69
23	19	99	2	0	0	6	84	2	1	1	2	0	1	1	3	0	1	2	1	72
24	20	74	2	0	0	8	89	1	1	1	2	2	1	0	2	0	0	1	0	66
25	11	78	0	2	1	8	100	0	1	1	1	2	1	1	3	0	1	1	1	66
26	17	65	1	0	1	5	75	2	1	2	1	2	1	2	3	0	1	2	0	63
27	21	62	0	1	1	6	54	0	1	0	0	0	1	2	3	0	2	0	1	64

StudentPerformanceFactors.csv

	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	Hours_Studied	Attendance	Parental_Involvement	Access_to_Resources	Extracurricular_Activities	Sleep_Hours	Previous_Scores	Motivation_Level	Internet_Access	Tutoring_Sessions	Family_Income	Teacher_Quality	School_Type	Peer_Influence	Physical_Activity	Learning_Disabilities	Parental_Education_Level	Distance_from_Home	Gender	Exam_Score	
2	23	84	Low	High	No	7	73	Low	Yes	0 Low	Medium	Public	Positive	3 No	High Scho	Near	Male	67			
3	19	64	Low	Medium	No	8	59	Low	Yes	2 Medium	Medium	Public	Negative	4 No	College	Moderate	Female	61			
4	24	98	Medium	Medium	Yes	7	91	Medium	Yes	2 Medium	Medium	Public	Neutral	4 No	Postgradu	Near	Male	74			
5	29	89	Low	Medium	Yes	8	98	Medium	Yes	1 Medium	Medium	Public	Negative	4 No	High Scho	Moderate	Male	71			
6	19	92	Medium	Medium	Yes	6	65	Medium	Yes	3 Medium	High	Public	Neutral	4 No	College	Near	Female	70			
7	19	88	Medium	Medium	Yes	8	89	Medium	Yes	3 Medium	Medium	Public	Positive	3 No	Postgradu	Near	Male	71			
8	29	84	Medium	Low	Yes	7	68	Low	Yes	1 Low	Medium	Private	Neutral	2 No	High Scho	Moderate	Male	67			
9	25	78	Low	High	Yes	6	50	Medium	Yes	1 High	High	Public	Negative	2 No	High Scho	Far	Male	66			
10	17	94	Medium	High	No	6	80	High	Yes	0 Medium	Low	Private	Neutral	1 No	College	Near	Male	69			
11	23	98	Medium	Medium	Yes	8	71	Medium	Yes	0 High	High	Public	Positive	5 No	High Scho	Moderate	Male	72			
12	17	80	Low	High	No	8	88	Medium	No	4 Medium	High	Private	Neutral	4 No	College	Moderate	Male	68			
13	17	97	Medium	High	Yes	6	87	Low	Yes	2 Low	High	Private	Neutral	2 No	High Scho	Near	Male	71			
14	21	83	Medium	Medium	Yes	8	97	Low	Yes	2 Medium	Medium	Public	Positive	4 No	High Scho	Near	Male	70			
15	9	82	Medium	Medium	Yes	8	72	Medium	Yes	2 Medium	Medium	Private	Positive	3 No	Postgradu	Near	Male	66			
16	10	78	Medium	High	Yes	8	74	Medium	Yes	1 Low	Medium	Private	Neutral	4 No	Postgradu	Near	Male	65			
17	17	68	Medium	Medium	No	8	70	Medium	Yes	2 Medium	Medium	Private	Positive	4 No	High Scho	Near	Female	64			
18	14	60	Medium	Low	Yes	10	65	Low	Yes	0 High	Medium	Private	Positive	3 No	College	Near	Male	60			
19	22	70	Low	Medium	Yes	6	82	Medium	Yes	1 Low	High	Public	Neutral	3 No	High Scho	Near	Female	65			
20	15	80	Medium	Medium	Yes	9	91	Low	Yes	3 Low	Medium	Public	Positive	2 No	College	Moderate	Female	67			
21	12	75	Medium	High	Yes	7	58	Medium	Yes	3 Medium	Medium	Private	Positive	4 No	College	Near	Male	66			
22	29	78	Medium	Medium	No	5	99	High	Yes	0 High	Medium	Public	Negative	1 No	High Scho	Moderate	Female	69			
23	19	99	Medium	High	No	6	84	Medium	Yes	1 Medium	High	Public	Neutral	3 No	High Scho	Near	Male	72			
24	20	74	Medium	High	No	8	89	Low	Yes	1 Medium	Medium	Public	Negative	2 No	College	Moderate	Female	66			
25	11	78	High	Medium	Yes	8	100	High	Yes	1 Low	Medium	Public	Neutral	3 No	High Scho	Moderate	Male	66			
26	17	65	Low	High	Yes	5	75	Medium	Yes	2 Low	Medium	Public	Positive	3 No	High Scho	Near	Female	63			
27	21	62	High	Low	Yes	6	54	High	Yes	0 High	High	Public	Positive	3 No	Postgradu	Far	Male	64			

This dataset captures a broad range of academic, social, and personal factors, enabling an in-depth analysis of what impacts student success in exams like study habits, attendance, parental involvement in their study, Access to Resources, Extracurricular Activities, Sleep Hours, Previous Scores, Motivation Level, Internet Access, Tutoring Sessions, Family Income, Teacher Quality, School Type, Peer Influence, Physical Activity, Learning Disabilities, Parental Education Level, Distance from Home, Gender, Exam Score .

Data Set Column Description: -

Column Descriptions

Attribute	Description
Hours_Studied	Number of hours spent studying per week.
Attendance	Percentage of classes attended.
Parental_Involvement	Level of parental involvement in the student's education (Low, Medium, High).
Access_to_Resources	Availability of educational resources (Low, Medium, High).

Extracurricular_Activities	Participation in extracurricular activities (Yes, No).
Sleep_Hours	Average number of hours of sleep per night.
Previous_Scores	Scores from previous exams.
Motivation_Level	Student's level of motivation (Low, Medium, High).
Internet_Access	Availability of internet access (Yes, No).
Tutoring_Sessions	Number of tutoring sessions attended per month.

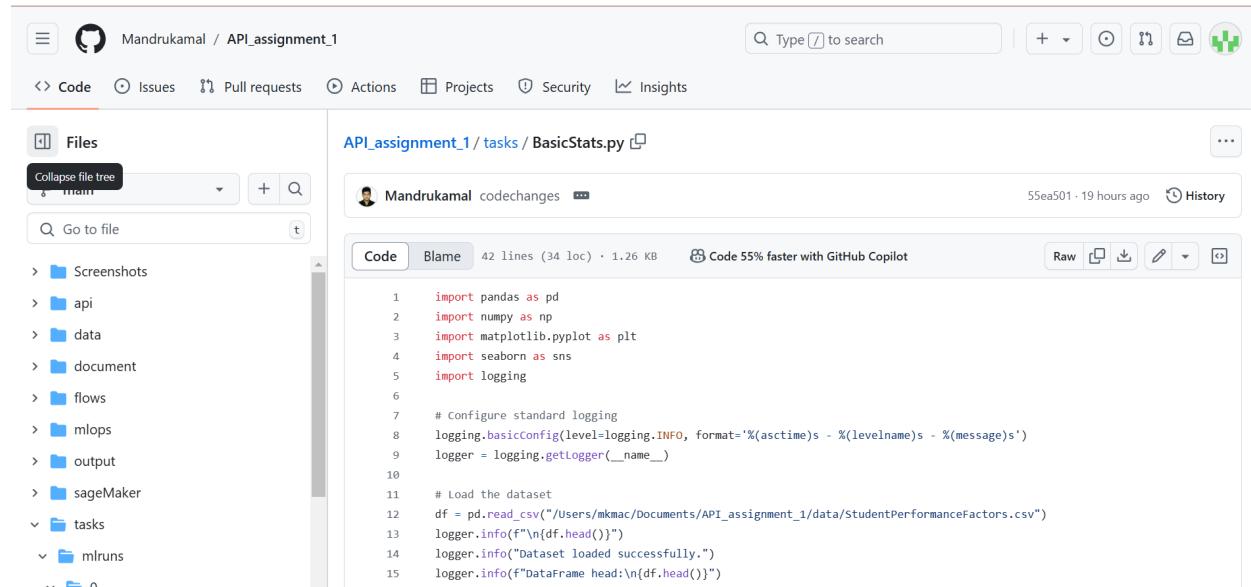
Family_Income	Family income level (Low, Medium, High).
Teacher_Quality	Quality of the teachers (Low, Medium, High).
School_Type	Type of school attended (Public, Private).
Peer_Influence	Influence of peers on academic performance (Positive, Neutral, Negative).
Physical_Activity	Average number of hours of physical activity per week.
Learning_Disabilities	Presence of learning disabilities (Yes, No).

Parental_Education_Level	Highest education level of parents (High School, College, Postgraduate).
Distance_from_Home	Distance from home to school (Near, Moderate, Far).
Gender	Gender of the student (Male, Female).
Exam_Score	Final exam score.

1.3 Data Pre-processing: Perform activities such as displaying summary statistics, checking for missing values, imputing missing data for numeric columns, displaying data types, and normalizing data.

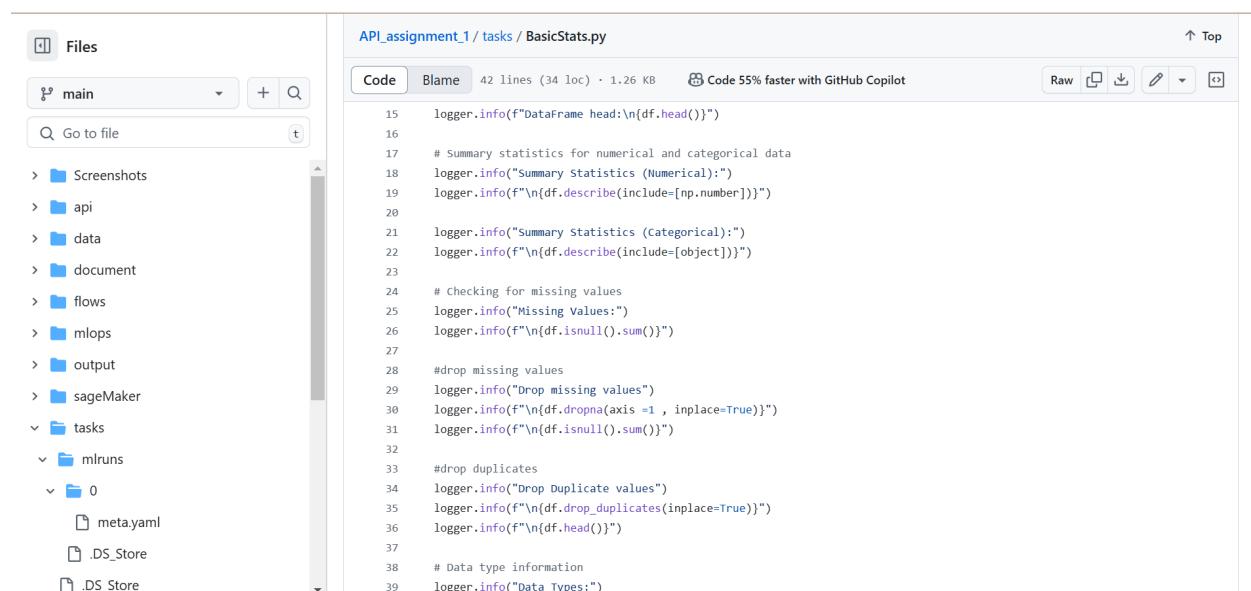
Below process is used for capturing the summary statistics, imputing missing data for numeric columns, displaying datatypes, checking for missing values and normalizing data;

BasicStats.py: -



The screenshot shows the GitHub interface for the repository 'API_assignment_1'. The left sidebar displays a file tree with several folders like 'Screenshots', 'api', 'data', etc. The main area shows the 'BasicStats.py' file. The code imports pandas, numpy, matplotlib.pyplot, seaborn, and logging. It configures standard logging, loads a dataset from 'StudentPerformanceFactors.csv', and prints the head of the DataFrame.

```
1 import pandas as pd
2 import numpy as np
3 import matplotlib.pyplot as plt
4 import seaborn as sns
5 import logging
6
7 # Configure standard logging
8 logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')
9 logger = logging.getLogger(__name__)
10
11 # Load the dataset
12 df = pd.read_csv("/Users/mkmac/Documents/API_assignment_1/data/StudentPerformanceFactors.csv")
13 logger.info(f"\n{df.head()}")
14 logger.info("Dataset loaded successfully.")
15 logger.info(f"DataFrame head:\n{df.head()}")
```



The screenshot shows the same GitHub interface for the 'BasicStats.py' file. The code has been expanded to include sections for summary statistics, checking for missing values, dropping missing values, dropping duplicates, and displaying data types. The logger is used to print these details to the console.

```
15 logger.info(f"DataFrame head:\n{df.head()}")
16
17 # Summary statistics for numerical and categorical data
18 logger.info("Summary Statistics (Numerical):")
19 logger.info(f"\n{df.describe(include=[np.number])}")
20
21 logger.info("Summary Statistics (Categorical):")
22 logger.info(f"\n{df.describe(include=[object])}")
23
24 # Checking for missing values
25 logger.info("Missing Values:")
26 logger.info(f"\n{df.isnull().sum()}")
27
28 #drop missing values
29 logger.info("Drop missing values")
30 logger.info(f"\n{df.dropna(axis=1, inplace=True)}")
31 logger.info(f"\n{df.isnull().sum()}")
32
33 #drop duplicates
34 logger.info("Drop Duplicate values")
35 logger.info(f"\n{df.drop_duplicates(inplace=True)}")
36 logger.info(f"\n{df.head()}")
37
38 # Data type information
39 logger.info("Data Types:")
```

The screenshot shows a GitHub Copilot interface. On the left, there's a sidebar titled 'Files' with a tree view of files. The file 'BasicStats.py' is selected and highlighted with a blue bar at the bottom. The main area is titled 'API_assignment_1 / tasks / BasicStats.py'. It contains the following code:

```
20
21     logger.info("Summary Statistics (Categorical):")
22     logger.info(f"\n{df.describe(include=[object])}")
23
24     # Checking for missing values
25     logger.info("Missing Values:")
26     logger.info(f"\n{df.isnull().sum()}")
27
28     #drop missing values
29     logger.info("Drop missing values")
30     logger.info(f"\n{df.dropna(axis=1, inplace=True)}")
31     logger.info(f"\n{df.isnull().sum()}")
32
33     #drop duplicates
34     logger.info("Drop Duplicate values")
35     logger.info(f"\n{df.drop_duplicates(inplace=True)}")
36     logger.info(f"\n{df.head()}")
37
38     # Data type information
39     logger.info("Data Types:")
40     logger.info(f"\n{df.dtypes}")
41     logger.info(f"\n{df.info()}")
42     logger.info(f"\n{df.describe()}")
```

```
Terminal Shell Edit View Window Help
```

Last login: Thu Oct 31 15:19:34 on ttys007
mkmac@MKMacs-MacBook-Air ~ % cd /Users/mkmac/Documents/API_assignment_1
mkmac@MKMacs-MacBook-Air API_assignment_1 % source /Users/mkmac/Documents/API_assignment_1/venv/bin/activate
(venv) mkmac@MKMacs-MacBook-Air API_assignment_1 %

```
(venv) ahsan@iMacs-MacBook-Air tasks % python BasicStats.py
2024-10-31 15:58:24,592 - INFO -
    Hours_Studied Attendance Parental_Involvement Access_to_Resources Extracurricular_Activities ... Learning_Disabilities Parental_Education_Level Distance_from_Home Gender Exam_Score
0           23          84            Low        High      No ...          No        High School        Near     Male       67
1           19          64            Low        Medium     No ...          No        College        Moderate   Female      61
2           28          98            Medium      Medium     Yes ...          No        Postgraduate        Near     Male       74
3           29          89            Low        Medium     Yes ...          No        High School        Moderate   Male       73
4           19          92            Medium      Medium     Yes ...          No        College        Near     Female      70

[5 rows x 20 columns]

2024-10-31 15:58:24,592 - INFO - Dataset loaded successfully.
2024-10-31 15:58:24,596 - INFO - DataFrame head:
    Hours_Studied Attendance Parental_Involvement Access_to_Resources Extracurricular_Activities ... Learning_Disabilities Parental_Education_Level Distance_from_Home Gender Exam_Score
0           23          84            Low        High      No ...          No        High School        Near     Male       67
1           19          64            Low        Medium     No ...          No        College        Moderate   Female      61
2           24          98            Medium      Medium     Yes ...          No        Postgraduate        Near     Male       74
3           29          89            Low        Medium     Yes ...          No        High School        Moderate   Male       73
4           19          92            Medium      Medium     Yes ...          No        College        Near     Female      70

[5 rows x 20 columns]

2024-10-31 15:58:24,596 - INFO - Summary Statistics (Numerical):
2024-10-31 15:58:24,602 - INFO -
    Hours_Studied  Attendance  Sleep_Hours  Previous_Scores  Tutoring_Sessions  Physical_Activity  Exam_Score
count      6607.000000  6607.000000  6607.000000  6607.000000  6607.000000  6607.000000
mean      19.75329  79.97748  7.02996  66.670000  1.493719  2.967610  67.235659
std       5.998594  11.547475  1.46812  14.399784  1.230570  1.831231  3.898466
min       0.000000  0.000000  0.000000  0.000000  0.000000  0.000000  0.000000
25%      15.000000  70.000000  6.00000  60.000000  1.000000  2.000000  65.000000
50%      20.000000  80.000000  7.00000  75.000000  1.000000  3.000000  67.000000
75%      24.000000  90.000000  8.00000  85.000000  2.000000  4.000000  69.000000
max      44.000000  100.000000  10.00000  100.000000  8.000000  6.000000  101.000000

2024-10-31 15:58:24,602 - INFO - Summary Statistics (Categorical):
2024-10-31 15:58:24,613 - INFO -
    Parental_Involvement Access_to_Resources Extracurricular_Activities Motivation_Level Internet_Access ... Peer_Influence Learning_Disabilities Parental_Education_Level Distance_from_Home Gender
count      6607          6607          6607          6607          6607 ...          6607          6607          6607          6517          6540          6607
unique         2             3             3             2             3 ...             2             2             2             3             2             2
top          Medium          Medium          Medium          Yes          Medium ...          Yes          Positive          No          High School          Near     Male
freq       3362          33319          3938          3351          6108 ...          2638          5912          3223          3884          3814

[4 rows x 13 columns]

2024-10-31 15:58:24,613 - INFO - Missing Values:
2024-10-31 15:58:24,615 - INFO -
    Hours_Studied  Attendance  Parental_Involvement  Access_to_Resources  Extracurricular_Activities  Motivation_Level  Internet_Access  Peer_Influence  Learning_Disabilities  Parental_Education_Level  Distance_from_Home  Gender
count      6607          6607          6607          6607          6607 ...          6607          6607          6607          6607          6607          6607
unique         0             0             0             0             0 ...             0             0             0             0             0             0             0
top          None          None          None          None          None ...          None          None          None          None          None          None
freq       3362          33319          3938          3351          6108 ...          2638          5912          3223          3884          3814

[4 rows x 13 columns]
```

```

Terminal Shell Edit View Window Help
tasks -- zsh -- 204x60
Thu 31 Oct 3:50:47PM

[5 rows x 17 columns]
2024-10-31 15:58:24,626 - INFO - Data Types:
2024-10-31 15:58:24,626 - INFO -
Hours_Studied           int64
Attendance               int64
Parental_Involvement    object
Access_to_Resources      object
Extracurricular_Activities object
Sleep_Hours              int64
Previous_Scores          int64
Motivation_Level         object
Internet_Access          object
Tutoring_Sessions        int64
Family_Income             object
School_Type               object
Peer_Influence            object
Physical_Activity         int64
Learning_Disabilities    object
Gender                   object
Exam_Score                int64
dtypes: object
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6647 entries, 0 to 6646
Data columns (total 17 columns):
 #   Column          Non-Null Count  Dtype  
0   Hours_Studied   6647 non-null    int64  
1   Attendance      6647 non-null    int64  
2   Parental_Involvement 6647 non-null    object 
3   Access_to_Resources 6647 non-null    object 
4   Extracurricular_Activities 6647 non-null    object 
5   Sleep_Hours     6647 non-null    int64  
6   Previous_Scores 6647 non-null    int64  
7   Motivation_Level 6647 non-null    object 
8   Internet_Access 6647 non-null    object 
9   Tutoring_Sessions 6647 non-null    int64  
10  Family_Income   6647 non-null    object 
11  School_Type     6647 non-null    object 
12  Peer_Influence  6647 non-null    object 
13  Physical_Activity 6647 non-null    int64  
14  Learning_Disabilities 6647 non-null    object 
15  Gender          6647 non-null    object 
16  Exam_Score      6647 non-null    int64  
dtypes: int64(7), object(10)
memory usage: 877.6+ KB
2024-10-31 15:58:24,631 - INFO -
None
2024-10-31 15:58:24,636 - INFO -
   Hours_Studied  Attendance  Sleep_Hours  Previous_Scores  Tutoring_Sessions  Physical_Activity  Exam_Score
count  6647.000000  6647.000000  6647.000000  6647.000000  6647.000000  6647.000000  6647.000000
mean   19.975329  11.977448  76.470031  1.492719  1.947610  67.235649
std    5.998594  11.547475  14.68812  4.099784  1.239570  1.031231  3.094456
min    1.000000  60.000000  4.000000  50.000000  0.000000  0.000000  55.000000
25%   16.000000  70.000000  6.000000  63.000000  1.000000  2.000000  65.000000
50%   20.000000  73.000000  7.000000  70.000000  1.000000  3.000000  77.000000
75%   24.000000  79.000000  8.000000  86.000000  2.000000  4.000000  69.000000
max   44.000000  100.000000  10.000000  100.000000  8.000000  6.000000  101.000000
(venv) mkmac@MKMac-MacBook-Air tasks %
```

1.4 Exploratory Data Analysis (EDA): Conduct EDA to include calculating correlation coefficients, identifying correlations between numeric and/or categorical features, binning, encoding, assessing feature importance, and visualizing data (using charts and graphs for univariate and bivariate analyses).

Binning.py:

Mandrukamal / API_assignment_1

Type to search

Code Issues Pull requests Actions Projects Security Insights

Files

main + Go to file

document flows mlops output sageMaker tasks mlruns .DS_Store BasicStats.py Binning.py CorrelationCoffecient.py

API_assignment_1 / tasks / Binning.py

Mandrukamal Added extra codes in tasks and updated workflow.py 3891b49 · 3 hours ago History

Code Blame 58 lines (48 loc) · 2.12 KB Code 55% faster with GitHub Copilot

```

1 import pandas as pd
2 import numpy as np
3 import logging
4 import matplotlib.pyplot as plt
5 import seaborn as sns
6 import os
7
8 # Configure standard logging
9 logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')
10 logger = logging.getLogger(__name__)
11
12 # Load the dataset
13 df = pd.read_csv("./Users/mkmac/Documents/API_assignment_1/data/ImputedStudentPerformanceFactors.csv")
14 logger.info(f"\n{df.head()}")
15 logger.info("Dataset loaded for binning.")

```

API_assignment_1 / tasks / Binning.py

```

15     logger.info("Dataset loaded for binning.")
16
17     # Create output directory for figures
18     output_dir = "figures"
19     os.makedirs(output_dir, exist_ok=True)
20
21     # Binning Hours_Studied into custom bins
22     min_value = df['Hours_Studied'].min()
23     max_value = df['Hours_Studied'].max()
24     logger.info(f"Min Hours_Studied: {min_value}, Max Hours_Studied: {max_value}")
25
26     # Bin size is custom, with 4 bins (e.g., Low, Medium, High, Very High)
27     bins = np.linspace(min_value, max_value, 5)
28     labels = ['Low', 'Medium', 'High', 'Very High']
29
30     # Distance binning
31     df['Hours_Studied_Binned'] = pd.cut(df['Hours_Studied'], bins=bins, labels=labels, include_lowest=True)
32     logger.info(f"Hours_Studied Distance Binning Results:\n{df['Hours_Studied_Binned']}")
```

Frequency binning for Exam_Score into quartiles

```

34     df['Exam_Score_Binned'] = pd.qcut(df['Exam_Score'], q=4, precision=1, labels=['Low', 'Medium', 'High', 'Very High'])
35     logger.info(f"Exam_Score Frequency Binning Results:\n{df['Exam_Score_Binned']}")
```

Plotting the binned results for Hours_Studied

```

38     plt.figure(figsize=(8, 6))
39     sns.countplot(data=df, x='Hours_Studied_Binned')
40     plt.title("Hours_Studied binning plot")
41     plt.xlabel("Hours_Studied")
42     plt.ylabel("Count")
43     plt.savefig(f"/Users/mkmac/Documents/API_assignment_1/output/binning/Binned_Hours_Studied.png")
44     plt.close()
45     logger.info("Binned Hours_Studied plot saved.")
```

Plotting the binned results for Exam_Score

```

48     plt.figure(figsize=(8, 6))
49     sns.countplot(data=df, x='Exam_Score_Binned')
50     plt.title("Exam_Score binning plot")
51     plt.xlabel("Exam_Score")
52     plt.ylabel("Count")
53     plt.savefig(f"/Users/mkmac/Documents/API_assignment_1/output/binning/Binned_Exam_Score.png")
54     plt.close()
55     logger.info("Binned Exam_Score plot saved.")
```

logger.info("Binning process completed successfully.")

API_assignment_1 / tasks / Binning.py

```

36     logger.info(f"Exam_Score Frequency Binning Results:\n{df['Exam_Score_Binned']}")
```

Plotting the binned results for Hours_Studied

```

39     plt.figure(figsize=(8, 6))
40     sns.countplot(data=df, x='Hours_Studied_Binned')
41     plt.title("Hours_Studied binning plot")
42     plt.xlabel("Hours_Studied")
43     plt.ylabel("Count")
44     plt.savefig(f"/Users/mkmac/Documents/API_assignment_1/output/binning/Binned_Hours_Studied.png")
45     plt.close()
46     logger.info("Binned Hours_Studied plot saved.")
```

Plotting the binned results for Exam_Score

```

49     plt.figure(figsize=(8, 6))
50     sns.countplot(data=df, x='Exam_Score_Binned')
51     plt.title("Exam_Score binning plot")
52     plt.xlabel("Exam_Score")
53     plt.ylabel("Count")
54     plt.savefig(f"/Users/mkmac/Documents/API_assignment_1/output/binning/Binned_Exam_Score.png")
55     plt.close()
56     logger.info("Binned Exam_Score plot saved.")
```

logger.info("Binning process completed successfully.")

API_assignment_1 / tasks / Binning.py

```

36     logger.info(f"Exam_Score Frequency Binning Results:\n{df['Exam_Score_Binned']}")
```

Plotting the binned results for Hours_Studied

```

39     plt.figure(figsize=(8, 6))
40     sns.countplot(data=df, x='Hours_Studied_Binned')
41     plt.title("Hours_Studied binning plot")
42     plt.xlabel("Hours_Studied")
43     plt.ylabel("Count")
44     plt.savefig(f"/Users/mkmac/Documents/API_assignment_1/output/binning/Binned_Hours_Studied.png")
45     plt.close()
46     logger.info("Binned Hours_Studied plot saved.")
```

Plotting the binned results for Exam_Score

```

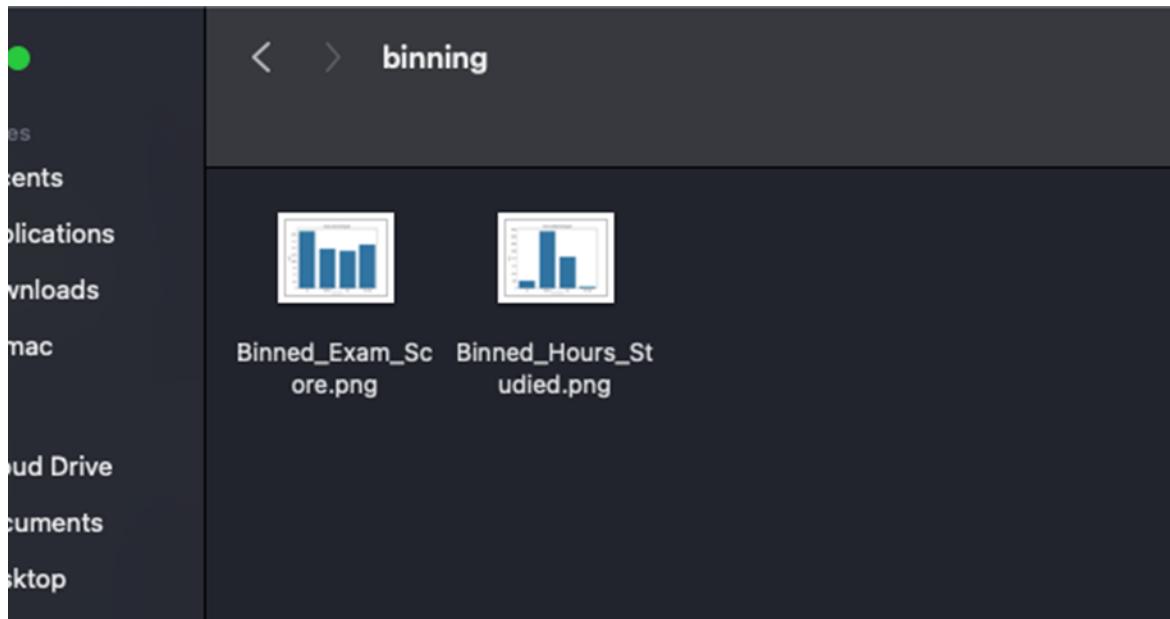
49     plt.figure(figsize=(8, 6))
50     sns.countplot(data=df, x='Exam_Score_Binned')
51     plt.title("Exam_Score binning plot")
52     plt.xlabel("Exam_Score")
53     plt.ylabel("Count")
54     plt.savefig(f"/Users/mkmac/Documents/API_assignment_1/output/binning/Binned_Exam_Score.png")
55     plt.close()
56     logger.info("Binned Exam_Score plot saved.")
```

logger.info("Binning process completed successfully.")

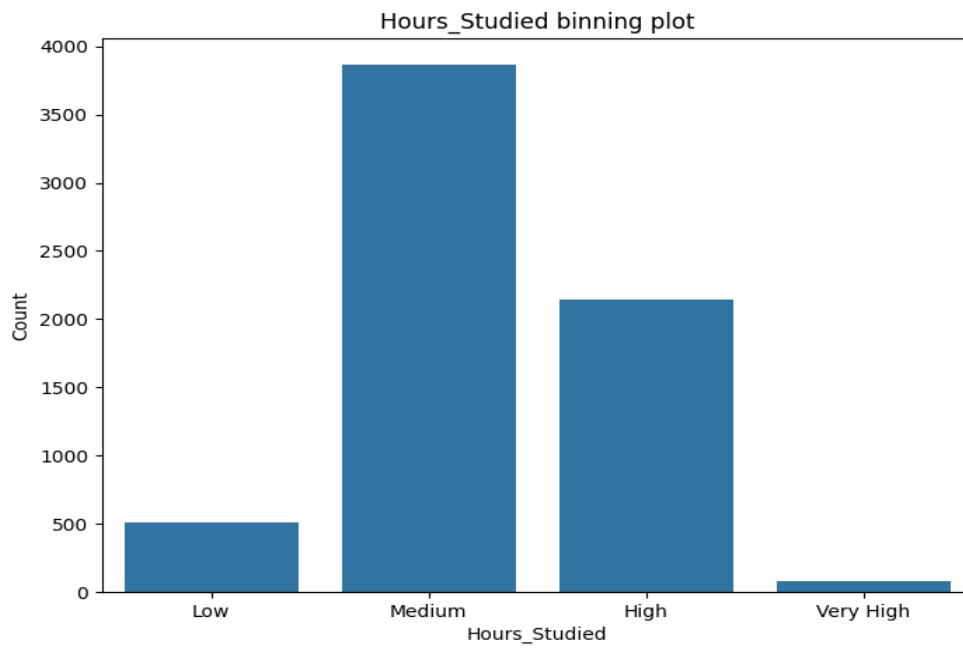
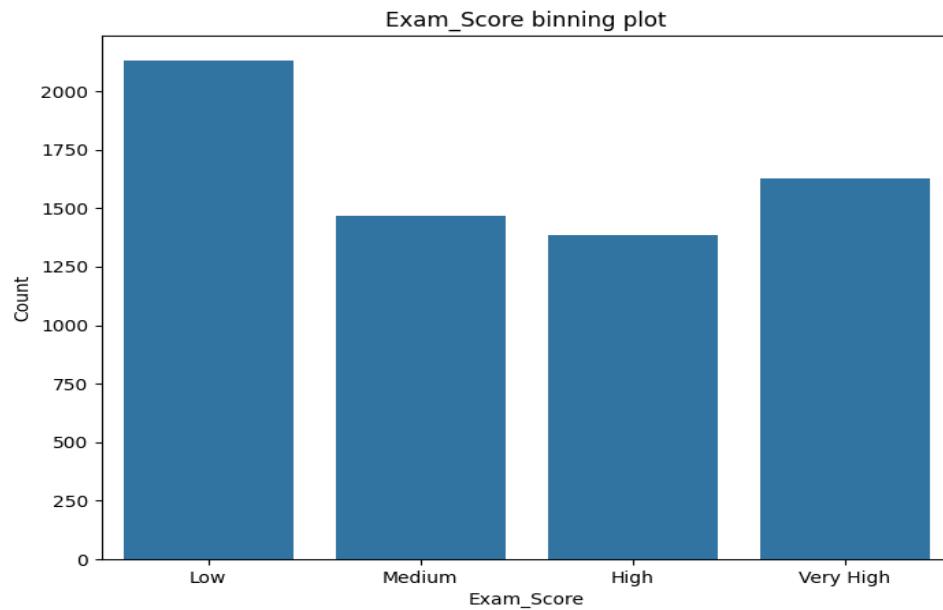
```

Terminal Shell Edit View Window Help tasks -- zsh -- 204x60
(menv) mkmac@MKMacBook-Air tasks % python Binning.py
2024-10-31 16:59:59,838 - INFO - Dataset loaded for binning.
2024-10-31 16:59:59,838 - INFO - Min Hours_Studied: 1.0, Max Hours_Studied: 44.6
2024-10-31 16:59:59,838 - INFO - Hours_Studied Distance Binning Results:
0      High
1      Medium
2      High
3      High
4      Medium
6692   High
6693   High
6694   Medium
6695   Low
6696   Medium
Name: Hours_Studied_Binned, Length: 6687, dtype: category
Categories: (4, object): ['Low' < 'Medium' < 'High' < 'Very High']
2024-10-31 16:59:59,838 - INFO - Exam_Score Frequency Binning Results:
0      Medium
1      Low
2      Very High
3      Very High
4      ...
6692   High
6693   High
6694   High
6695   High
Name: Exam_Score_Binned, Length: 6687, dtype: category
Categories: (4, object): ['Low' < 'Medium' < 'High' < 'Very High']
2024-10-31 16:59:59,838 - INFO - Binned_Hours_Studied plot saved.
2024-10-31 16:59:59,174 - INFO - Exam_Score plot saved.
2024-10-31 16:59:59,174 - INFO - Binning process completed successfully.
(venv) mkmac@MKMacBook-Air tasks % 

```



Outputs: -



Pearson Correlation: -

Mandrukamal / API_assignment_1

Code Issues Pull requests Actions Projects Security Insights

Files

main Go to file

output sageMaker tasks mlruns .DS_Store BasicStats.py Binning.py CorrelationCoffecient.py Encoding.py FeatureImportanceMLAlgorithm... PearsonCorrelation.py Visualization.py .DS_Store .gitattributes

API_assignment_1 / tasks / PearsonCorrelation.py

Mandrukamal Added extra codes in tasks and updated workflow.py 3891b49 · 3 hours ago History

Code Blame 67 lines (55 loc) · 2.54 KB Code 55% faster with GitHub Copilot

```
1 import numpy as np
2 import pandas as pd
3 from scipy.stats import pearsonr
4 import logging
5 import matplotlib.pyplot as plt
6 import os
7
8 # Configure standard Python logging
9 logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')
10 logger = logging.getLogger(__name__)
11
12 # Loading the dataset
13 df = pd.read_csv("/Users/mkmac/Documents/API_assignment_1/data/ImputedStudentPerformanceFactors.csv")
14 logger.info(f"\n{df.head()}")
15 logger.info("dataset loaded for Pearson correlation.")
```

API_assignment_1 / tasks / PearsonCorrelation.py ↑ Top

Code Blame 67 lines (55 loc) · 2.54 KB Code 55% faster with GitHub Copilot

```
15 logger.info("Dataset loaded for Pearson correlation.")
16
17 # Set a seed for reproducibility
18 np.random.seed(42)
19
20 # Reduce noise in Exam_Score by creating a stronger linear relationship for testing purposes
21 df['Hours_Studied'] = np.random.uniform(0, 10, size=len(df)) # Random hours studied between 0 and 10
22 df['Previous_Scores'] = 0.5 * df['Hours_Studied'] + np.random.normal(0, 1, len(df)) # Dependent on Hours_Studied
23
24 # Create Exam_Score based on both Hours_Studied and Previous_Scores
25 df['Exam_Score'] = 0.85 * df['Hours_Studied'] + 0.85 * df['Previous_Scores'] + np.random.normal(0, 2, len(df))
26
27 # Create output directory for figures
28 output_dir = "figures"
29 os.makedirs(output_dir, exist_ok=True)
30
31 # Variables for Pearson correlation
32 list1 = df['Hours_Studied']
33 list2 = df['Exam_Score']
34
35 # Compute Pearson correlation
36 corr1, _ = pearsonr(list1, list2)
37 logger.info(f'Pearson correlation between Hours_Studied and Exam_Score: {corr1:.2f}')
38
```

Files

main Go to file

- > output
- > sageMaker
- > tasks
- > mlruns
- .DS_Store
- BasicStats.py
- Binning.py
- CorrelationCoffecient.py
- Encoding.py
- FeatureImportanceMLAlgorithm...
- PearsonCorrelation.py
- Visualization.py
- .DS_Store
- .gitattributes

API_assignment_1 / tasks / PearsonCorrelation.py

Code Blame 67 lines (55 loc) · 2.54 KB Code 55% faster with GitHub Copilot

```
38     # Scatter plot for Hours_Studied vs Exam_Score
39     plt.figure(figsize=(8, 6))
40     plt.scatter(list1, list2, alpha=0.7)
41     plt.xlabel('Hours_Studied')
42     plt.ylabel('Exam_Score')
43     plt.title(f'Scatter plot of Hours_Studied vs Exam_Score (Pearson r = {corr1:.2f})')
44     scatter_plot_path1 = "/Users/mkmac/Documents/API_assignment_1/output/pearson_correlation/scatter_HoursStudied_ExamScore.png"
45     plt.savefig(scatter_plot_path1)
46     logger.info(f"Scatter plot saved as '{scatter_plot_path1}'")
47     plt.close()
48
49
50     # Variables for Pearson correlation
51     list3 = df['Previous_Scores']
52     list4 = df['Exam_Score']
53
54     # Compute Pearson correlation
55     corr2, _ = pearsonr(list3, list4)
56     logger.info(f'Pearson correlation between Previous_Scores and Exam_Score: {corr2:.2f}')
57
58     # Scatter plot for Previous_Scores vs Exam_Score
59     plt.figure(figsize=(8, 6))
60     plt.scatter(list3, list4, alpha=0.7)
61     plt.xlabel('Previous_Scores')
62     plt.ylabel('Exam_Score')
63     plt.title(f'Scatter plot of Previous_Scores vs Exam_Score (Pearson r = {corr2:.2f})')
64     scatter_plot_path2 = "/Users/mkmac/Documents/API_assignment_1/output/pearson_correlation/scatter_Previou...  
65     plt.savefig(scatter_plot_path2)
66     logger.info(f"Scatter plot saved as '{scatter_plot_path2}'")
67     plt.close()
```

Files

main Go to file

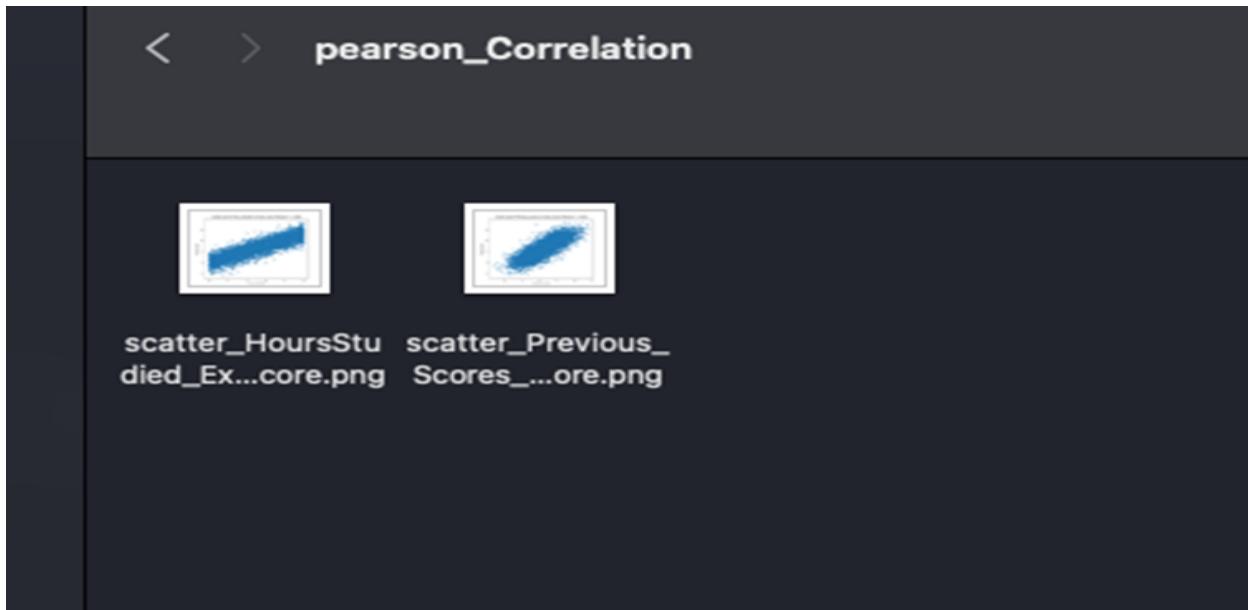
- > output
- > sageMaker
- > tasks
- > mlruns
- .DS_Store
- BasicStats.py
- Binning.py
- CorrelationCoffecient.py
- Encoding.py
- FeatureImportanceMLAlgorithm...
- PearsonCorrelation.py
- Visualization.py
- .DS_Store
- .gitattributes

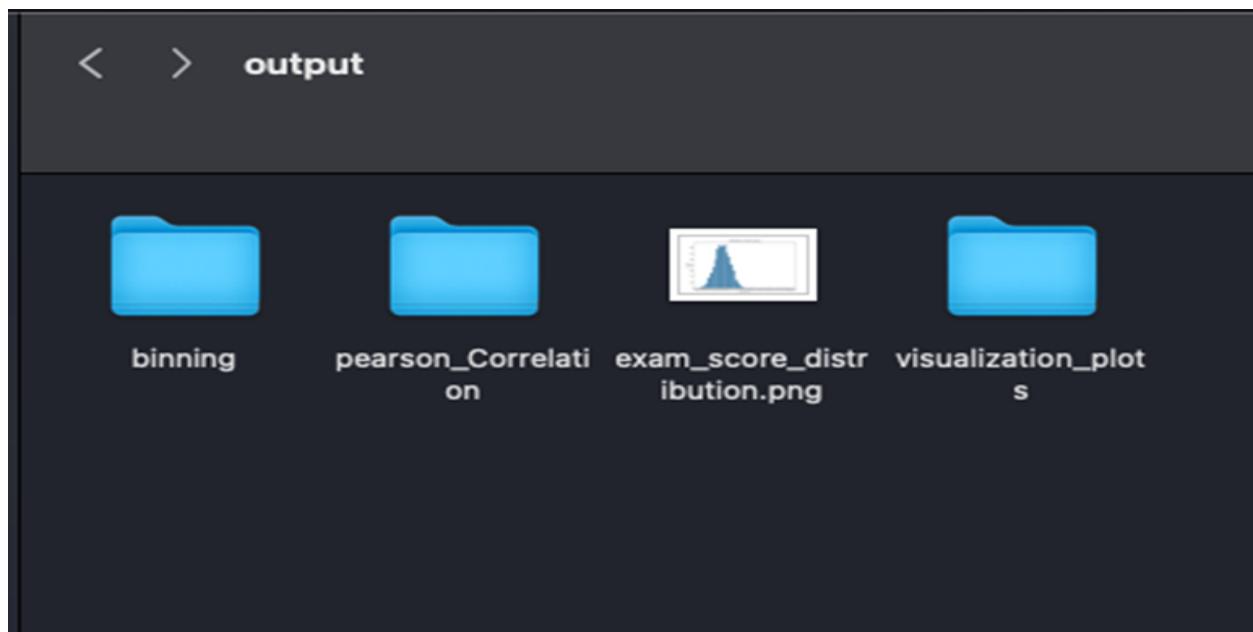
API_assignment_1 / tasks / PearsonCorrelation.py

Code Blame 67 lines (55 loc) · 2.54 KB Code 55% faster with GitHub Copilot

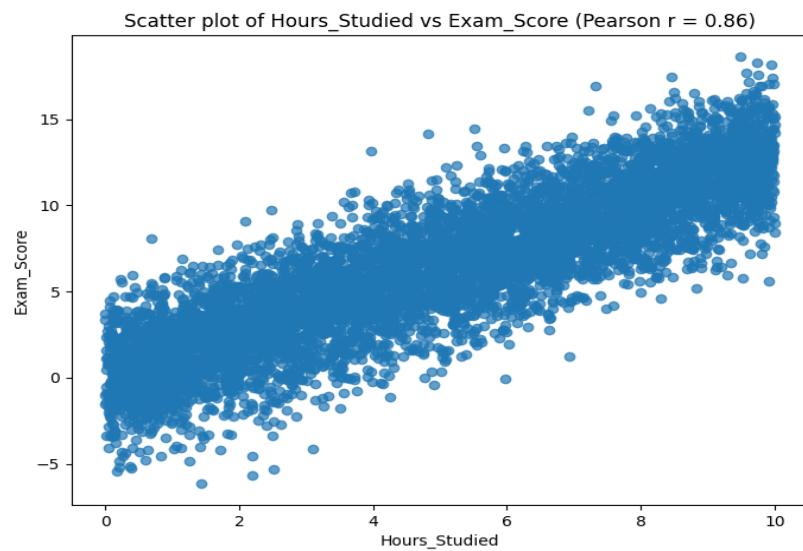
```
46     plt.savefig(scatter_plot_path1)
47     logger.info(f"Scatter plot saved as '{scatter_plot_path1}'")
48     plt.close()
49
50     # Variables for Pearson correlation
51     list3 = df['Previous_Scores']
52     list4 = df['Exam_Score']
53
54     # Compute Pearson correlation
55     corr2, _ = pearsonr(list3, list4)
56     logger.info(f'Pearson correlation between Previous_Scores and Exam_Score: {corr2:.2f}')
57
58     # Scatter plot for Previous_Scores vs Exam_Score
59     plt.figure(figsize=(8, 6))
60     plt.scatter(list3, list4, alpha=0.7)
61     plt.xlabel('Previous_Scores')
62     plt.ylabel('Exam_Score')
63     plt.title(f'Scatter plot of Previous_Scores vs Exam_Score (Pearson r = {corr2:.2f})')
64     scatter_plot_path2 = "/Users/mkmac/Documents/API_assignment_1/output/pearson_correlation/scatter_Previou...  
65     plt.savefig(scatter_plot_path2)
66     logger.info(f"Scatter plot saved as '{scatter_plot_path2}'")
67     plt.close()
```

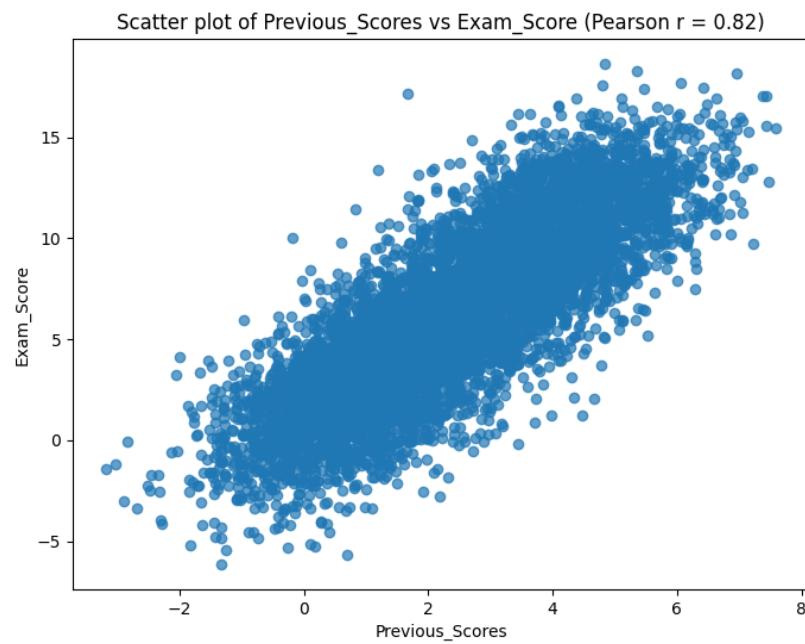
```
Terminal Shell Edit View Window Help tasks --zsh -- 204x60
(mkmac) mkmac@MKMacBook-Air tasks % ls
BasicStat.py          CorrelationCoefficient.py      FeatureImportanceMLalgorithms.py    Visualization.py
Binning.py            Encoding.py                  PearsonCorrelation.py             figures
(mkmac) mkmac@MKMacBook-Air tasks % python PearsonCorrelation.py
2024-10-31 15:51:31,866 - INFO -
  Hours_Studied Attendance Parental_Involvement Access_to_Resources Extracurricular_Activities ... Learning_Disabilities Parental_Education_Level Distance_from_Home Gender Exam_Score
0           23.0        84.0            1.0                 0.0               0.0 ...             0.0                   1.0                2.0       1.0        67.0
1           19.0        66.0            1.0                 2.0               0.0 ...             0.0                   0.0                2.0       1.0        61.0
2           24.0        98.0            2.0                 2.0               1.0 ...             0.0                   0.0                2.0       1.0        75.0
3           29.0        89.0            1.0                 2.0               1.0 ...             0.0                   1.0       1.0       1.0        71.0
4           19.0        92.0            2.0                 2.0               0.0 ...             0.0                   0.0                2.0       0.0        70.0
[5 rows x 20 columns]
2024-10-31 15:51:31,866 - INFO - Dataset loaded for Pearson correlation.
2024-10-31 15:51:31,867 - INFO - Pearson correlation between Hours_Studied and Exam_Score: 0.86
2024-10-31 15:51:31,865 - INFO - Scatter plot saved as '/Users/mkmac/Documents/API_assignment_1/output/pearson_Correlation/scatter_HoursStudied_ExamScore.png'
2024-10-31 15:51:31,865 - INFO - Pearson correlation between Previous_Scores and Exam_Score: 0.82
2024-10-31 15:51:31,839 - INFO - Scatter plot saved as '/Users/mkmac/Documents/API_assignment_1/output/pearson_Correlation/scatter_Previous_Scores_ExamScore.png'
(mkmac) mkmac@MKMacBook-Air tasks %
```





Outputs: -





Correlation Coefficient: -

```

from tkinter import TRUE
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt

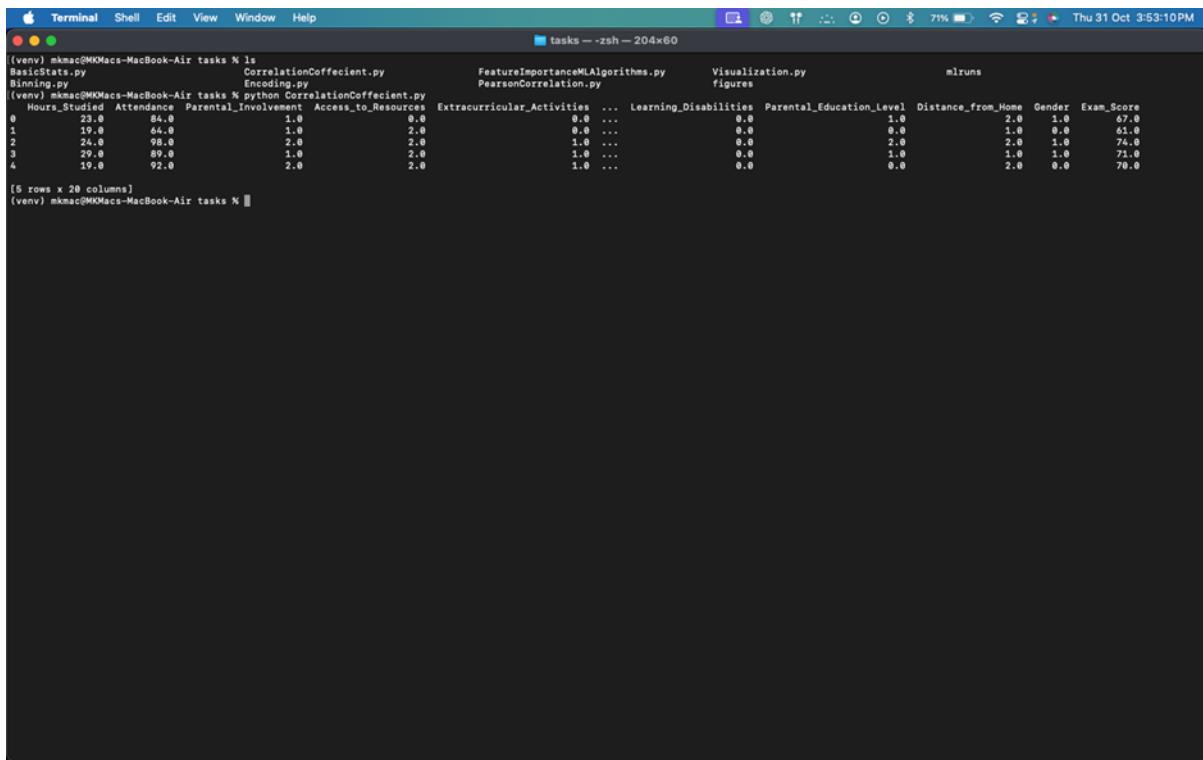
# Load the dataset
df = pd.read_csv("/Users/mkmac/Documents/API_assignment_1/data/ImputedStudentPerformanceFactors.csv")
print(df.head()) # Print the first few rows of the dataset for a quick preview

# Correlation Matrix using Pearson Correlation
corr_matrix = df.corr() # Compute the correlation matrix

# Plotting Heatmap
plt.figure(figsize=(10, 6))
sns.heatmap(corr_matrix, annot=True, cmap='coolwarm', fmt=".2f", linewidths=0.5)
plt.title("Correlation Heatmap of Student Performance Factors")

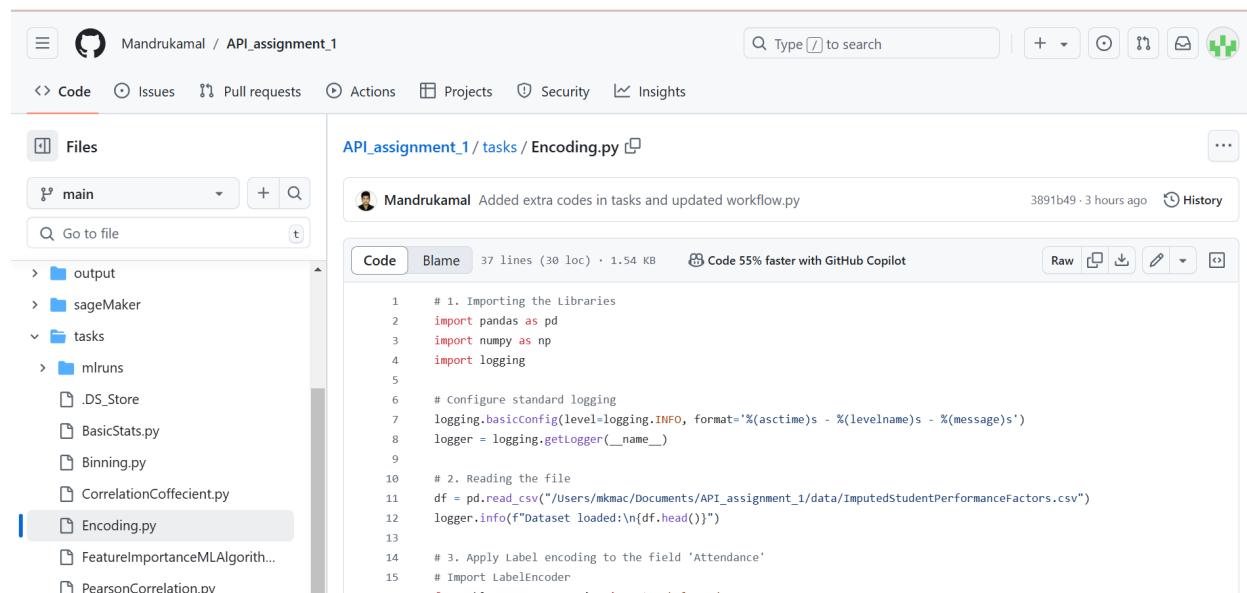
```

Output: -



```
(venv) mkmac@MKMac-MacBook-Air tasks % ls
BasicStats.py          CorrelationCoefficient.py      FeatureImportanceMLAlgorithms.py    Visualization.py
Binning.py              Encoding.py                  PearsonCorrelation.py            mlruns
(venv) mkmac@MKMac-MacBook-Air tasks % python CorrelationCoefficient.py
   Hours_Studied  Attendance  Parental_Involvement  Access_to_Resources  Extracurricular_Activities ...  Learning_Disabilities  Parental_Education_Level  Distance_from_Home  Gender  Exam_Score
0           23.0        84.0             1.0                   0.0                 0.0 ...               0.0                   1.0                2.0       1.0      67.0
1           19.0        68.0             1.0                   2.0                 0.0 ...               0.0                   0.0                0.0       1.0      61.0
2           24.0        98.0             2.0                   2.0                 1.0 ...               0.0                   0.0                2.0       1.0      75.0
3           29.0        89.0             1.0                   2.0                 1.0 ...               0.0                   1.0                1.0       1.0      71.0
4           19.0        92.0             2.0                   2.0                 1.0 ...               0.0                   0.0                2.0       2.0      70.0
(5 rows x 20 columns)
(venv) mkmac@MKMac-MacBook-Air tasks %
```

Encoding.py: -



Mandrukamal / API_assignment_1

Type ⌂ to search

Code Issues Pull requests Actions Projects Security Insights

Files

main

Go to file

output

sageMaker

tasks

mlruns

.DS_Store

BasicStats.py

Binning.py

CorrelationCoefficient.py

Encoding.py

FeatureImportanceMLAlgorithm...

PearsonCorrelation.py

API_assignment_1 / tasks / Encoding.py

Mandrukamal Added extra codes in tasks and updated workflow.py 3891b49 · 3 hours ago History

Code Blame 37 lines (30 loc) · 1.54 KB Code 55% faster with GitHub Copilot

```
1 # 1. Importing the Libraries
2 import pandas as pd
3 import numpy as np
4 import logging
5
6 # Configure standard logging
7 logging.basicConfig(level=logging.INFO, format='%(asctime)s - %(levelname)s - %(message)s')
8 logger = logging.getLogger(__name__)
9
10 # 2. Reading the file
11 df = pd.read_csv("/Users/mkmac/Documents/API_assignment_1/data/ImputedStudentPerformanceFactors.csv")
12 logger.info(f"Dataset loaded:\n{df.head()}")
13
14 # 3. Apply Label encoding to the field 'Attendance'
15 # Import LabelEncoder
```

The screenshot shows a Jupyter Notebook interface. On the left, there's a sidebar with a 'Files' tab and a tree view of files and directories. The 'tasks' directory is expanded, showing files like 'main', 'output', 'sageMaker', 'mlruns', '.DS_Store', 'BasicStats.py', 'Binning.py', 'CorrelationCoefficient.py', 'Encoding.py' (which is highlighted), 'FeatureImportanceMLAlgorithm...', 'PearsonCorrelation.py', 'Visualization.py', '.DS_Store', and '.gitattributes'. The main pane displays the code for 'Encoding.py'. The code uses LabelEncoder from sklearn.preprocessing to encode categorical columns ('Attendance' and 'Teacher_Quality') and pd.get_dummies to create one-hot encoding for these columns. It also includes a comment about potentially encoding multiple columns in the future.

```
15     # Import LabelEncoder
16     from sklearn.preprocessing import LabelEncoder
17     le = LabelEncoder()
18     df['Attendance'] = le.fit_transform(df['Attendance'])
19     logger.info(f"Label encoded 'Attendance' column:\n{df['Attendance']}")
20
21     # For Teacher_Quality field
22     df['Teacher_Quality'] = le.fit_transform(df['Teacher_Quality'])
23     logger.info(f"Label encoded 'Teacher_Quality' column:\n{df['Teacher_Quality']}")
24
25     # Represent Attendance using One-Hot Encoding
26     logger.info(f"Attendance value counts:\n{df['Attendance'].value_counts()}")
27     df = pd.get_dummies(df, columns=['Attendance'])
28     logger.info(f"One-hot encoded 'Attendance' column:\n{df.head()}")
29
30     # One hot encoding for Teacher_Quality field
31     logger.info(f"Teacher_Quality value counts:\n{df['Teacher_Quality'].value_counts()}")
32     df = pd.get_dummies(df, columns=['Teacher_Quality'])
33     logger.info(f"One-hot encoded 'Teacher_Quality' column:\n{df.head()}")
34
35     # For multiple columns (if needed in the future, uncomment below)
36     # df = pd.get_dummies(df, columns=['Teacher_Quality', 'Attendance'])
37     # logger.info(f"One-hot encoded columns 'Teacher_Quality' and 'Attendance':\n{df.head()}"
```

Output: -

```
[venv] mmac@MMacBook-Air tasks % ls
BasicStats.py           CorrelationCoefficient.py      FeatureImportanceMLAlgorithms.py    Visualization.py          mruns
Binning.py              Encoding.py                  PearsonCorrelation.py            figures
[venv] mmac@MMacBook-Air tasks % python Encoding.py
2024-10-31 15:53:46.375 - INFO - Dataset loaded:
   Hours_Studied  Attendance  Parental_Involvement  Access_to_Resources  Extracurricular_Activities ...  Learning_Disabilities  Parental_Education_Level  Distance_from_Home  Gender  Exam_Score
0            23.0        84.0                 1.0                0.0                   0.0 ...                   0.0                   1.0                2.0       1.0        67.0
1            19.0        64.0                 1.0                2.0                   0.0 ...                   0.0                   0.0                1.0       0.0        61.0
2            24.0        93.0                 2.0                2.0                   1.0 ...                   0.0                   0.0                2.0       2.0        70.0
3            39.0        89.0                 1.0                2.0                   1.0 ...                   0.0                   0.0                2.0       1.0        71.0
4            19.0        92.0                 2.0                2.0                   1.0 ...                   0.0                   0.0                2.0       0.0        70.0

[5 rows x 20 columns]
2024-10-31 15:53:46.939 - INFO - Label encoded 'Attendance' column:
   0
   1
   2
   3
   4
   ..
6682  9
6683  16
6684  30
6685  26
6686  7
Name: Attendance, Length: 6687, dtype: int64
2024-10-31 15:53:46.940 - INFO - Label encoded 'Teacher_Quality' column:
   0
   1
   2
   3
   4
   ..
6682  2
6683  0
6684  2
6685  2
6686  2
Name: Teacher_Quality, Length: 6687, dtype: int64
2024-10-31 15:53:46.941 - INFO - Attendance value counts:
Attendance
7     198
38    187
16    105
37    184
4     182
34    180
24    175
19    175
35    175
32    173
8     170
9     170
28    169
21    168
15    168
33    167
12    167
14    165
```

```

1  175
2  173
3  170
9  170
20 169
21 168
36 168
15 165
33 167
12 167
14 165
18 165
1  164
35 163
29 162
11 162
37 161
10 161
6  160
23 157
10 166
28 155
3  155
39 154
32 154
2  152
16 151
27 151
15 149
25 146
6  145
8  87
40 81
Name: count, dtype: int64
2024-10-31 15:53:46,954 - INFO - One-hot encoded 'Attendance' column:
   Hours_Studied Parental_Involvement Access_to_Resources Extracurricular_Activities Sleep_Hours ... Attendance_36 Attendance_37 Attendance_38 Attendance_39 Attendance_40
0      23.0          1.0            0.0             0.0       7.0 ...    False     False    False    False    False
1      19.0          1.0            2.0             0.0       8.0 ...    False     False    False    False    False
2      24.0          2.0            2.0             1.0       7.0 ...    False     False    True     False    False
3      29.0          1.0            2.0             1.0       8.0 ...    False     False    False    False    False
4      19.0          2.0            2.0             1.0       6.0 ...    False     False    False    False    False
[5 rows x 60 columns]
2024-10-31 15:53:46,955 - INFO - Teacher_Quality value counts:
Teacher_Quality
2  3925
0  1947
1  657
3  78
Name: count, dtype: int64
2024-10-31 15:53:46,967 - INFO - One-hot encoded 'Teacher_Quality' column:
   Hours_Studied Parental_Involvement Access_to_Resources Extracurricular_Activities Sleep_Hours ... Attendance_40 Teacher_Quality_0 Teacher_Quality_1 Teacher_Quality_2 Teacher_Quality_3
0      23.0          1.0            0.0             0.0       7.0 ...    False     False    False    True     False
1      19.0          1.0            2.0             0.0       8.0 ...    False     False    True     False    False
2      24.0          2.0            2.0             1.0       7.0 ...    False     False    True     False    False
3      29.0          1.0            2.0             1.0       8.0 ...    False     False    True     False    False
4      19.0          2.0            2.0             1.0       6.0 ...    False     True     False    False    False
[5 rows x 63 columns]
(venv) mkmac@MKMacBook-Air tasks %

```

Feature Importance ML Algorithm: -

The screenshot shows a GitHub repository interface for a project named "API_assignment_1". The repository has 4836fa7 commits, 3 hours ago, and a history link.

The left sidebar shows the file structure:

- Files
- main
- output
- sageMaker
- tasks
- mldrns
- .DS_Store
- BasicStats.py
- Binning.py
- CorrelationCoffecient.py
- Encoding.py
- FeatureImportanceMLAlgorithms.py
- PearsonCorrelation.py

The right pane displays the content of the "FeatureImportanceMLAlgorithms.py" file:

```

import pandas as pd
import numpy as np

df = pd.read_csv("/Users/mkmac/Documents/API_assignment_1/data/ImputedStudentPerformanceFactors.csv")
print(df)

from sklearn import preprocessing
import matplotlib.pyplot as plt

# Encode the Categorical data of Gender - using Dummy Column - Creates a new column called Gender_M
# Use Dummy Variables
df = pd.get_dummies(df, columns=['Gender'], drop_first=True)

print(df.columns)
print(df.columns[0])

```

The screenshot shows a Jupyter Notebook interface. On the left, there's a sidebar with a 'Files' section containing a tree view of files: 'main', 'output', 'sageMaker', 'tasks', 'mlruns', 'DS_Store', 'BasicStats.py', 'Binning.py', 'CorrelationCofficient.py', 'Encoding.py', 'FeatureImportanceMLAlgorith...', 'PearsonCorrelation.py', 'Visualization.py', '.DS_Store', and '.gitattributes'. The file 'FeatureImportanceMLAlgorith...' is currently selected and highlighted in grey. At the bottom of the sidebar, there are buttons for 'Run', 'Kernel', 'Cell', 'File', 'Edit', 'Help', and 'About'. The main area is titled 'API_assignment_1 / tasks / FeatureImportanceMLAlgorithms.py'. It contains the following Python code:

```
14 print(df.columns)
15 print(df.columns[0])
16 x = df.iloc[:,[0,1,4,5,7,10]]
17 y = df.iloc[:,[6]]
18
19 # 1. Decision Tree
20 # Decision tree algorithms like classification and regression trees (CART) offer importance scores based on the reduction in
21 # After being fit, the model provides a feature_importances_ property that can be accessed to retrieve the relative importan
22 from sklearn.tree import DecisionTreeClassifier
23 # define the model
24 model = DecisionTreeClassifier()
25 # fit the model
26 model.fit(X,Y)
27 # get importance
28 importance = model.feature_importances_
29 # summarize feature importance
30 for i,v in enumerate(importance):
31     print('Feature: %d, score: %.5f' % (i,v))
32 # plot feature importance
33 plt.bar([x for x in range(len(importance))], importance)
34
35 # 2. Random Forest Feature Importance
36 # After being fit, the model provides a feature_importances_ property that can be accessed to retrieve the relative importan
37 from sklearn.ensemble import RandomForestRegressor
38 # define the model
```

The screenshot shows a Jupyter Notebook interface with the following details:

- Left Sidebar:** A file tree showing the directory structure. The current file is `FeatureImportanceMLAlgorithms.py`.
- Top Bar:** Shows the path `API_assignment_1 / tasks / FeatureImportanceMLAlgorithms.py` and a link to `Top`.
- Code Editor:** The code itself, which includes imports for `RandomForestRegressor` and `KNeighborsClassifier`, defines models, fits them to data, and calculates feature importance using permutation importance for both models.
- Code Completion:** A tooltip indicates that the code is 55% faster with GitHub Copilot.
- Toolbar:** Standard Jupyter Notebook toolbar buttons for raw code, copy, paste, etc.

```
from sklearn.ensemble import RandomForestRegressor
model = RandomForestRegressor()
model.fit(X,Y)
importance = model.feature_importances_
for i,v in enumerate(importance):
    print('Feature: %d, Score: %.5f' % (i,v))
plt.bar([x for x in range(len(importance))], importance)

# 3. Permutation Feature Importance - using KNN
from sklearn.neighbors import KNeighborsClassifier
from sklearn.inspection import permutation_importance

# define the model
model = KNeighborsClassifier()
model.fit(X,Y)
results = permutation_importance(model, X, Y, scoring='accuracy')
# get importance
```

The screenshot shows a Jupyter Notebook interface. On the left, there's a sidebar with a 'Files' section containing a tree view of files: main, output, sageMaker, tasks, mlruns, .DS_Store, BasicStats.py, Binning.py, CorrelationCoffecent.py, Encoding.py, FeatureImportanceMLAlgorithm..., PearsonCorrelation.py, and Visualization.py. Below these are two empty slots for .DS_Store and .gitattributes.

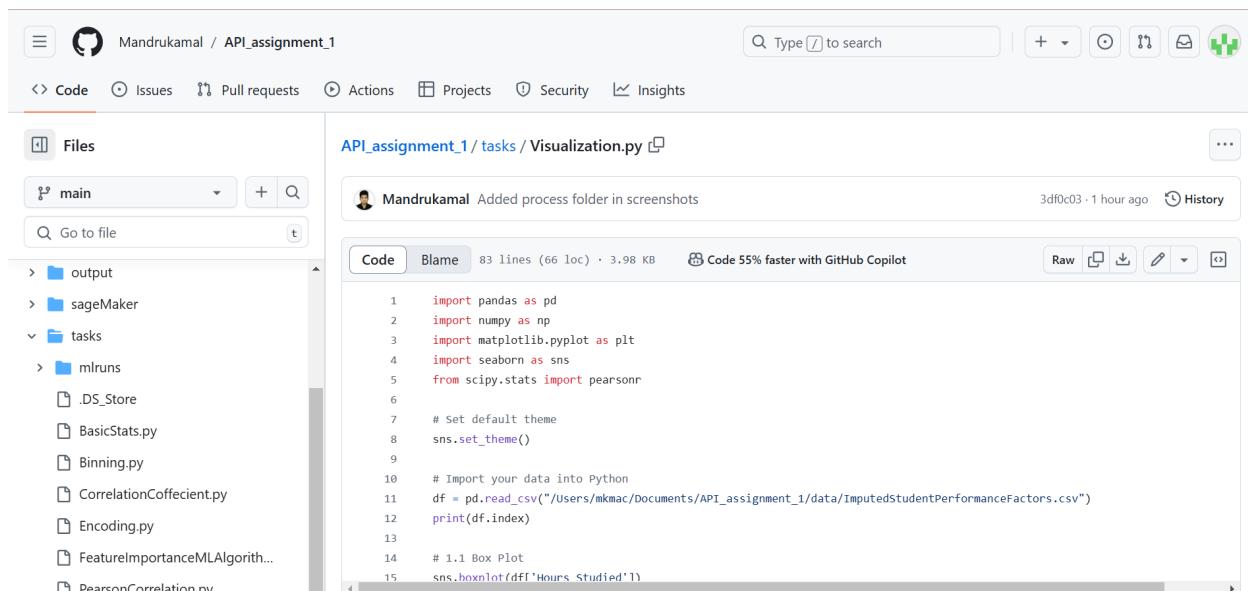
The main area has a title bar 'API_assignment_1 / tasks / FeatureImportanceMLAlgorithms.py' and a top right corner with a 'Top' link. The main content is a code editor with the following Python script:

```
45     for i,v in enumerate(importance):
46         print('Feature: %d, Score: %.5f' % (i,v))
47     # plot feature importance
48     plt.bar([x for x in range(len(importance))], importance)
49
50     # 3. Permutation Feature Importance - using KNN
51     from sklearn.neighbors import KNeighborsClassifier
52     from sklearn.inspection import permutation_importance
53
54     # define the model
55     model = KNeighborsClassifier()
56     # fit the model
57     model.fit(X,Y)
58
59     # perform permutation importance
60     results = permutation_importance(model, X, Y, scoring='accuracy')
61
62     # get importance
63     importance = results.importances_mean
64     # summarize feature importance
65     for i,v in enumerate(importance):
66         print('Feature: %d, Score: %.5f' % (i,v))
67     # plot feature importance
68     plt.bar([x for x in range(len(importance))], importance)
```

Output: -

```
(venv) mkmac@MKMacs-MacBook-Air tasks % ls
BasicStats.py          CorrelationCoffeient.py      FeatureImportanceMLalgorithms.py    Visualization.py      mlruns
Binning.py             Encoding.py                  PearsonCorrelation.py            figures
(venv) mkmac@MKMacs-MacBook-Air tasks % python FeatureImportanceMLalgorithms.py
   Hours_Studied Attendance Parental_Involvement Access_to_Resources Extracurricular_Activities ... Learning_Disabilities Parental_Education_Level Distance_from_Home Gender Exam_Score
0       23.0        84.0           1.0            0.0              0.0 ...             0.0           1.0            2.0       1.0       67.0
1       14.0        64.0           1.0            2.0              2.0 ...             0.0           0.0            0.0       1.0       61.0
2       24.0        92.0           2.0            2.0              2.0 ...             0.0           0.0            2.0       1.0       76.0
3       29.0        89.0           1.0            2.0              2.0 ...             1.0           0.0            1.0       1.0       71.0
4       19.0        92.0           2.0            2.0              2.0 ...             1.0           0.0            0.0       2.0       70.0
...
...
6692      25.0        69.0           0.0            2.0              0.0 ...             0.0           1.0            2.0       0.0       68.0
6693      23.0        76.0           0.0            2.0              0.0 ...             0.0           1.0            2.0       0.0       69.0
6694      20.0        98.0           2.0            1.0              1.0 ...             0.0           0.0            2.0       0.0       68.0
6695      10.0        86.0           0.0            0.0              0.0 ...             1.0           0.0            1.0       0.0       68.0
6696      15.0        67.0           2.0            1.0              1.0 ...             0.0           0.0            2.0       1.0       64.0
[6697 rows x 20 columns]
Index(['Hours_Studied', 'Attendance', 'Parental_Involvement',
       'Access_to_Resources', 'Extracurricular_Activities', 'Sleep_Hours',
       'Previous_Scores', 'Motivation_Level', 'Internet_Access',
       'Tutoring_Sessions', 'Family_Income', 'Teacher_Quality', 'School_Type',
       'Peer_Influence', 'Physical_Activity', 'Learning_Disabilities',
       'Parental_Education_Level', 'Distance_from_Home', 'Exam_Score',
       'Gender_1vs0'],
      dtype='object')
Hours_Studied
Feature: 0, Score: 0.31483
Feature: 1, Score: 0.27622
Feature: 2, Score: 0.08558
Feature: 3, Score: 0.15458
Feature: 4, Score: 0.15458
Feature: 5, Score: 0.05752
/Users/mkmac/Documents/API_assignment_1/venv/lib/python3.12/site-packages/sklearn/base.py:1473: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
    return fit_method(self, *args, **kwargs)
Feature: 0, Score: 0.27221
Feature: 1, Score: 0.32013
Feature: 2, Score: 0.06491
Feature: 3, Score: 0.05648
Feature: 4, Score: 0.00998
Feature: 5, Score: 0.02146
/Users/mkmac/Documents/API_assignment_1/venv/lib/python3.12/site-packages/sklearn/neighbors/_classification.py:238: DataConversionWarning: A column-vector y was passed when a 1d array was expected. Please change the shape of y to (n_samples,), for example using ravel().
    return self._fit(X, y)
Feature: 0, Score: 0.18919
Feature: 1, Score: 0.20567
Feature: 2, Score: -0.00115
Feature: 3, Score: 0.10095
Feature: 4, Score: 0.04817
Feature: 5, Score: 0.02785
(venv) mkmac@MKMacs-MacBook-Air tasks %
```

Visualization.py: -



```
API_assignment_1 / tasks / Visualization.py
Code Blame 83 lines (66 loc) · 3.98 KB Code 55% faster with GitHub Copilot
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from scipy.stats import pearsonr
# Set default theme
sns.set_theme()
# Import your data into Python
df = pd.read_csv("/Users/mkmac/Documents/API_assignment_1/data/ImputedStudentPerformanceFactors.csv")
print(df.index)
# 1.1 Box Plot
sns.boxplot(df['Hours_Studied'])
```

Files

- main
- Go to file
- output
- sageMaker
- tasks
- mlruns
- .DS_Store
- BasicStats.py
- Binning.py
- CorrelationCofficient.py
- Encoding.py
- FeatureImportanceMLAlgorithm.py
- PearsonCorrelation.py
- Visualization.py
- .DS_Store
- .gitattributes

API_assignment_1 / tasks / Visualization.py

Code Blame 83 lines (66 loc) · 3.98 KB Code 55% faster with GitHub Copilot

```

15     sns.boxplot(df['Hours_Studied'])
16     plt.title('1. Box Plot of Hours_Studied')
17     plt.savefig("/users/mkmac/Documents/API_assignment_1/output/visualization_plots/boxplot_hours_studied.png")
18
19     # 1.2 Strip Plot
20     sns.stripplot(y=df['Hours_Studied'])
21     plt.title('2. Strip Plot of Hours_Studied')
22     plt.savefig("/users/mkmac/Documents/API_assignment_1/output/visualization_plots/striplplot_hours_studied.png")
23
24     # 1.3 Swarm Plot
25     sns.swarmplot(x=df['Hours_Studied'])
26     plt.title('3. Swarm Plot of Hours_Studied')
27     plt.savefig("/users/mkmac/Documents/API_assignment_1/output/visualization_plots/swarmplot_hours_studied.png")
28
29     #sns.swarmplot(x=df['Attendance'])
30     plt.title('4. Swarm Plot of Attendance')
31     plt.savefig("/users/mkmac/Documents/API_assignment_1/output/visualization_plots/swarmplot_attendance.png")
32
33     # 1.4 Histogram
34     plt.hist(df['Hours_Studied'])
35     plt.title('5. Histogram of Hours_Studied')
36     plt.savefig("/users/mkmac/Documents/API_assignment_1/output/visualization_plots/histogram_hours_studied.png")
37
38     # 1.5 Distplot

```

main API_assignment_1 / tasks / Visualization.py

Code Blame 83 lines (66 loc) · 3.98 KB Code 55% faster with GitHub Copilot

```

38     # 1.5 Distplot
39     sns.distplot(df['Hours_Studied'], kde=False, color='blue', bins=5)
40     plt.title('6. Dist Plot of Hours_Studied with 5 bins')
41     plt.savefig("/users/mkmac/Documents/API_assignment_1/output/visualization_plots/distplot_hours_studied.png")
42
43     # 1.6 Countplot
44     sns.countplot(df['Gender'])
45     plt.title('7. Count Plot of Gender (Categorical)')
46     plt.savefig("/users/mkmac/Documents/API_assignment_1/output/visualization_plots/countplot_gender.png")
47
48     # 2.1 Boxplot for Bivariate Analysis
49     sns.boxplot(x=df['Exam_Score'], y=df['Attendance'], data=df)
50     plt.title('8. Box Plot of Exam_Score vs Attendance')
51     plt.savefig("/users/mkmac/Documents/API_assignment_1/output/visualization_plots/boxplot_exam_score_attendance.png")
52
53     # 2.2 Scatter Plot
54     sns.scatterplot(x=df['Attendance'], y=df['Hours_Studied'])
55     plt.title('9. Scatter Plot of Hours_Studied vs Attendance')
56     plt.savefig("/users/mkmac/Documents/API_assignment_1/output/visualization_plots/scatterplot_hours_studied_attendance.png")
57
58     sns.scatterplot(x=df['Attendance'], y=df['Hours_Studied'], hue=df['Exam_Score'])
59     plt.title('10. Scatter Plot of Hours_Studied vs Attendance vs Exam_Score')
60     plt.savefig("/users/mkmac/Documents/API_assignment_1/output/visualization_plots/scatterplot_hours_studied_attendance_exam_score.png")
61
62     # 2.3 FacetGrid for Gender vs Extracurricular_Activities

```

main API_assignment_1 / tasks / Visualization.py

Code Blame 83 lines (66 loc) · 3.98 KB Code 55% faster with GitHub Copilot

```

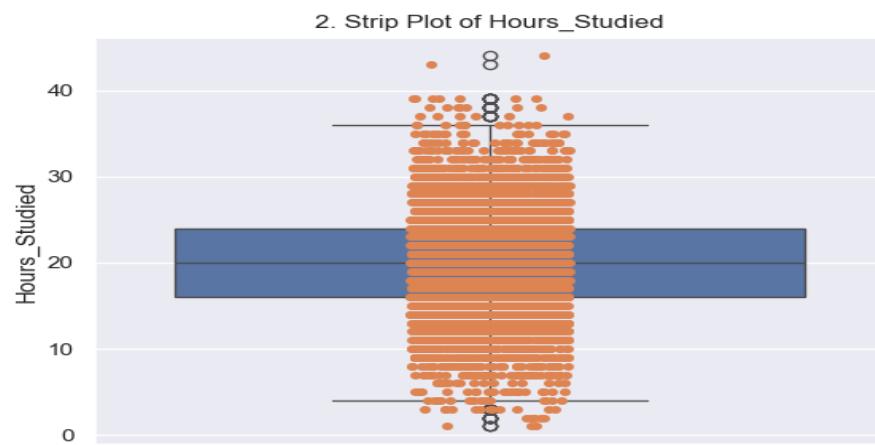
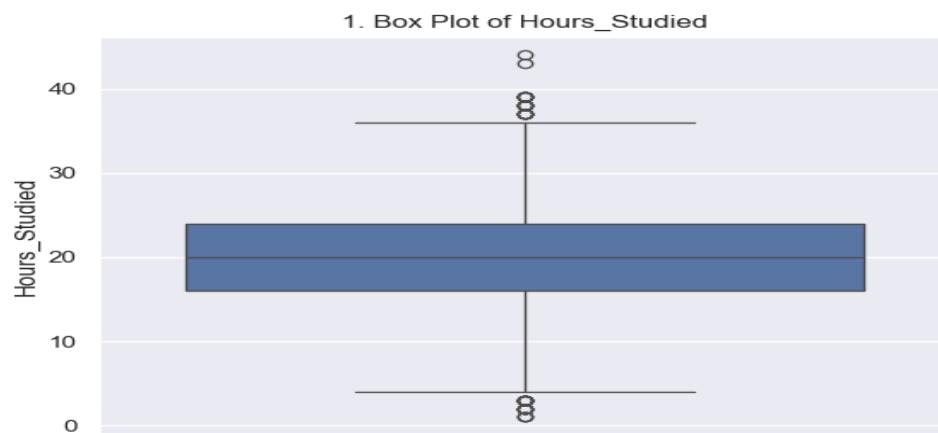
61
62     # 2.3 FacetGrid for Gender vs Extracurricular_Activities
63     g = sns.FacetGrid(df, col="Gender", height=6.5, aspect=.85)
64     g.map(sns.histplot, "Extracurricular_Activities")
65     plt.savefig("/users/mkmac/Documents/API_assignment_1/output/visualization_plots/facetgrid_gender_extracurricular.png")
66
67     # Multivariate Analysis - FacetGrid
68     g = sns.FacetGrid(df, col="Extracurricular_Activities", hue="Gender", margin_titles=True, height=6.5, aspect=.85)
69     g.map(sns.histplot, "Hours_Studied")
70     plt.savefig("/users/mkmac/Documents/API_assignment_1/output/visualization_plots/facetgrid_hours_studied_extracurricular.png")
71
72     # 2.4 lmplot
73     sns.lmplot(data=df, x="Hours_Studied", y="Extracurricular_Activities", hue="Gender")
74     plt.title('13. lmplot of Hours_Studied vs Extracurricular_Activities vs Gender')
75     plt.savefig("/users/mkmac/Documents/API_assignment_1/output/visualization_plots/lmplot_hours_studied_extracurricular.png")
76
77     sns.lmplot(data=df, x="Hours_Studied", y="Attendance", hue="Gender")
78     plt.title('14. lmplot of Hours_Studied vs Attendance vs Gender')
79     plt.savefig("/users/mkmac/Documents/API_assignment_1/output/visualization_plots/lmplot_hours_studied_attendance_gender.png")
80
81     sns.lmplot(data=df, x="Attendance", y="Hours_Studied", hue="Extracurricular_Activities")
82     plt.title('15. lmplot of Hours_Studied vs Attendance vs Extracurricular Activities')
83     plt.savefig("/users/mkmac/Documents/API_assignment_1/output/visualization_plots/lmplot_attendance_hours_studied_extracurricular.png")

```

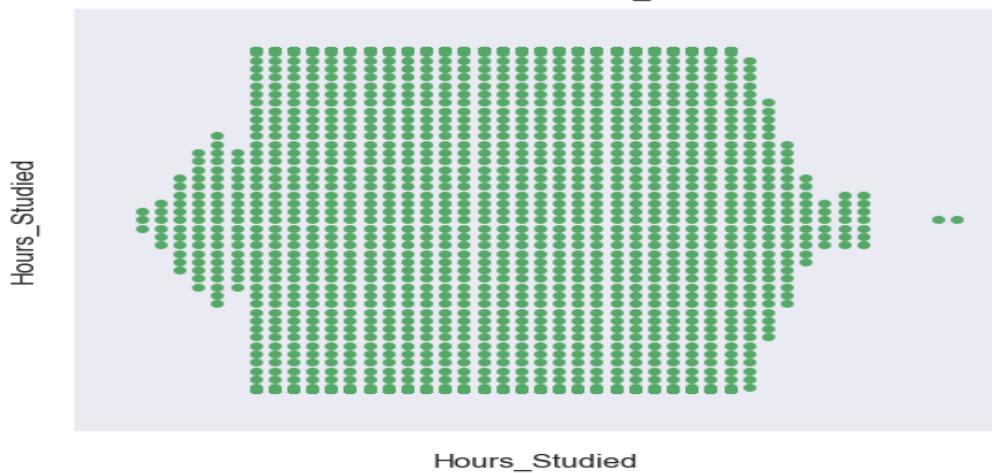
```
(venv) mkmac@MKMacs-MacBook-Air tasks % ls
[BasicStats.py                               PearsonCorrelation.py
Binning.py                                  Visualization.py
CorrelationCoefficient.py                  figures
Encoding.py                                 mlruns
FeatureImportanceMLAlgorithms.py
(venv) mkmac@MKMacs-MacBook-Air tasks % python Visualization.py
[RangeIndex(start=0, stop=6607, step=1)
(venv) mkmac@MKMacs-MacBook-Air tasks %
```

Outputs:

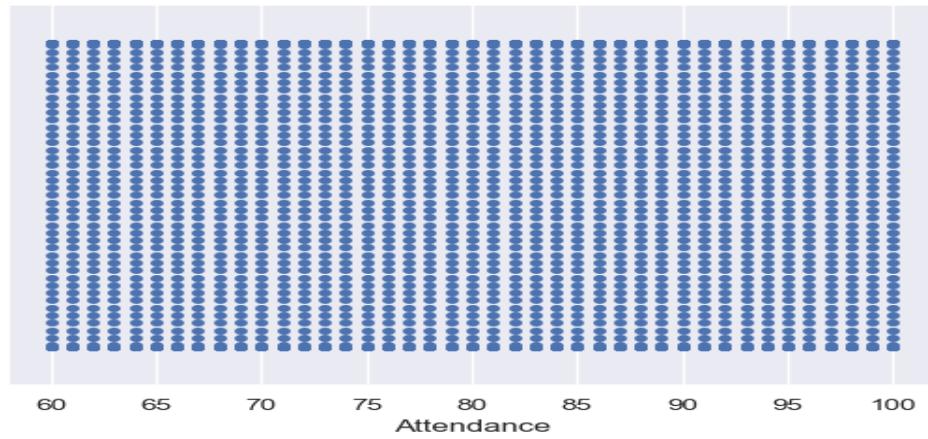
Univariate



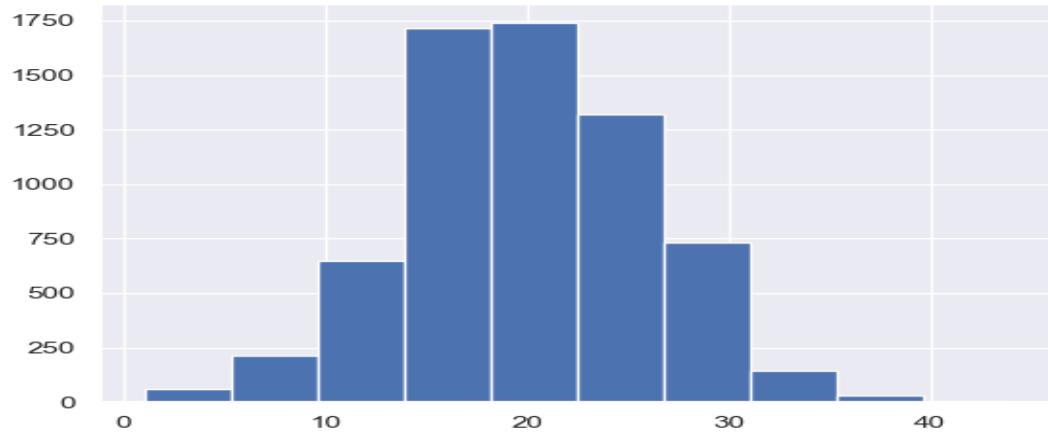
3. Swarm Plot of Hours_Studied

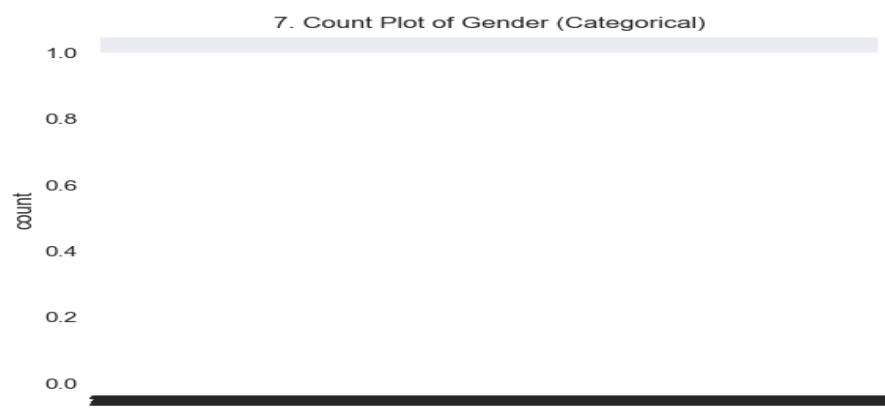
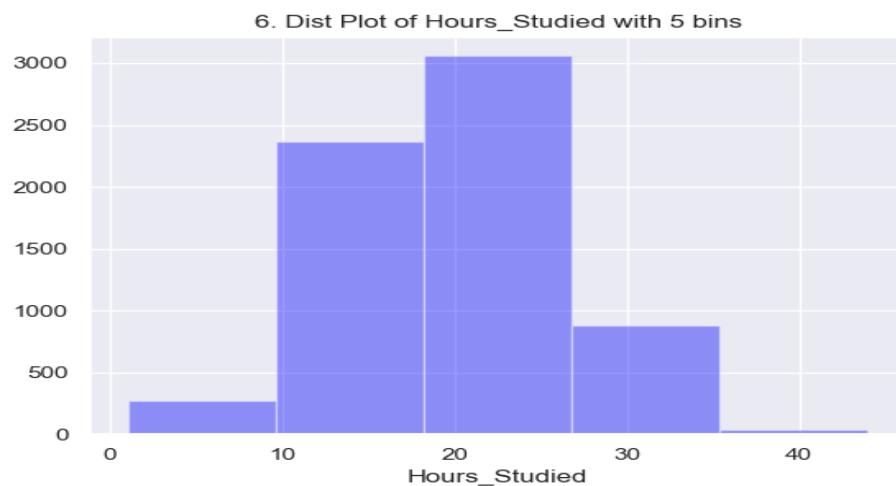


4. Swarm Plot of Attendance

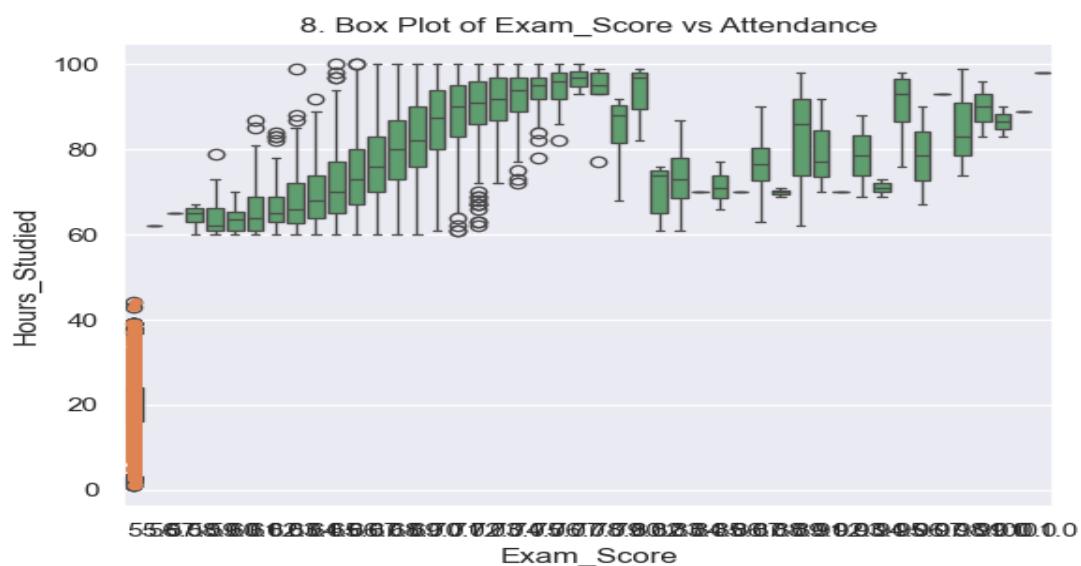


5. Histogram of Hours_Studied

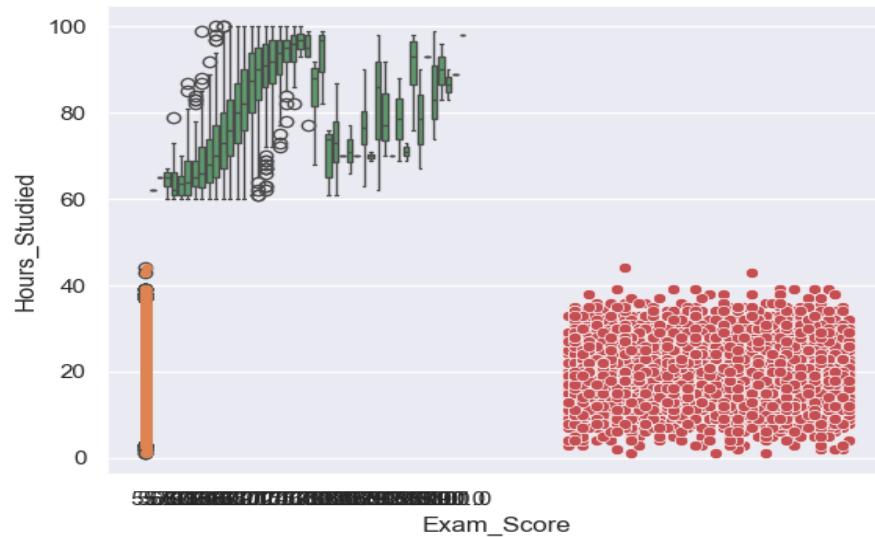




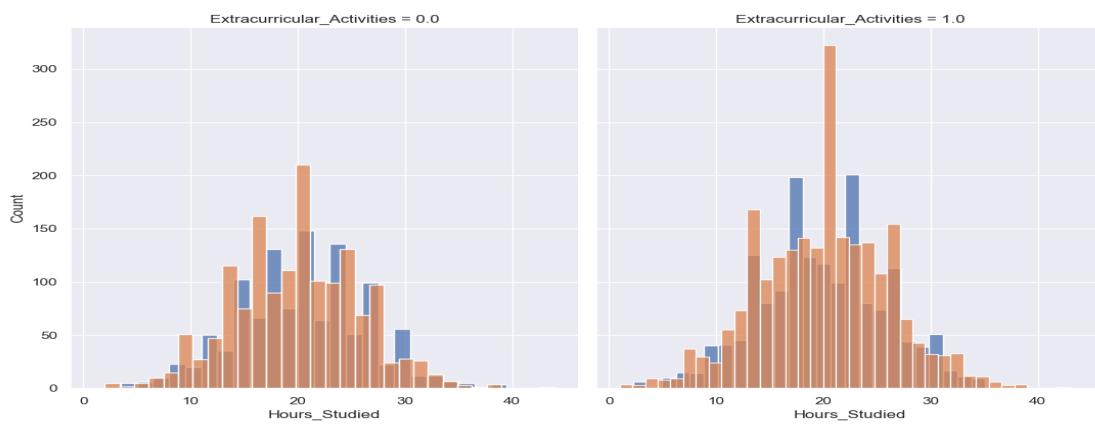
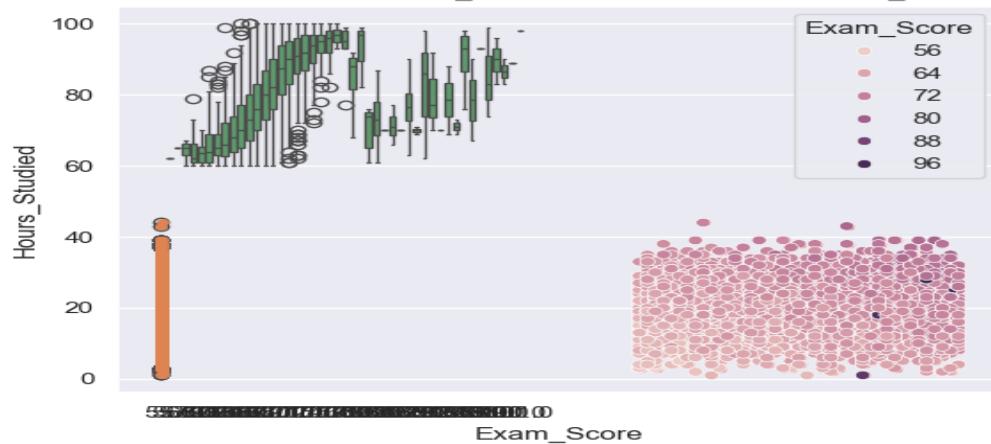
Bivariate

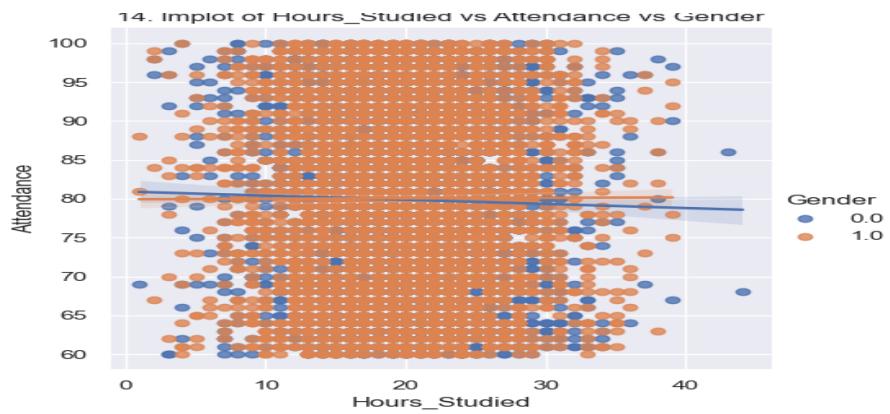
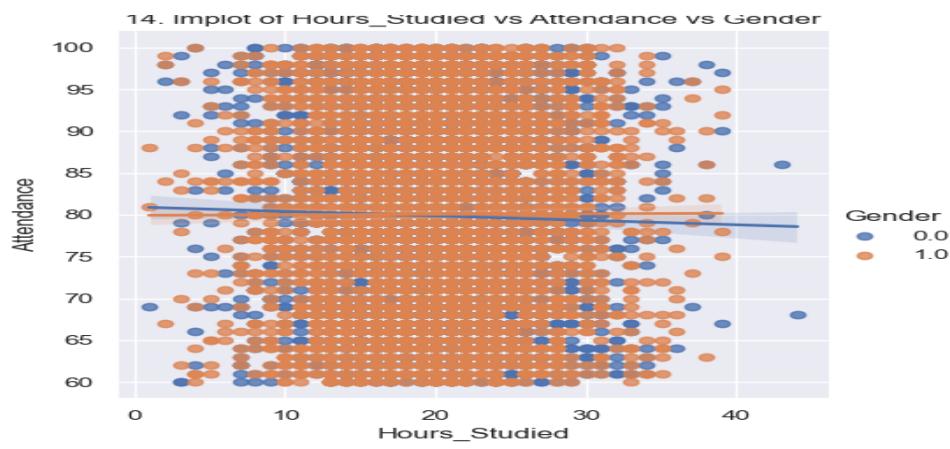
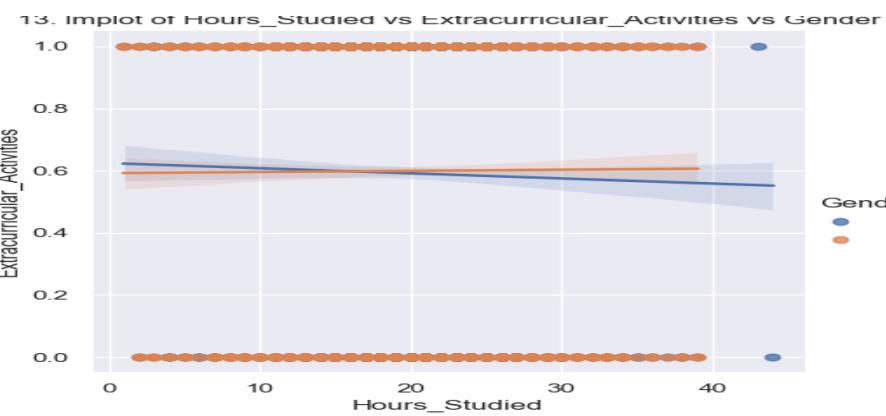
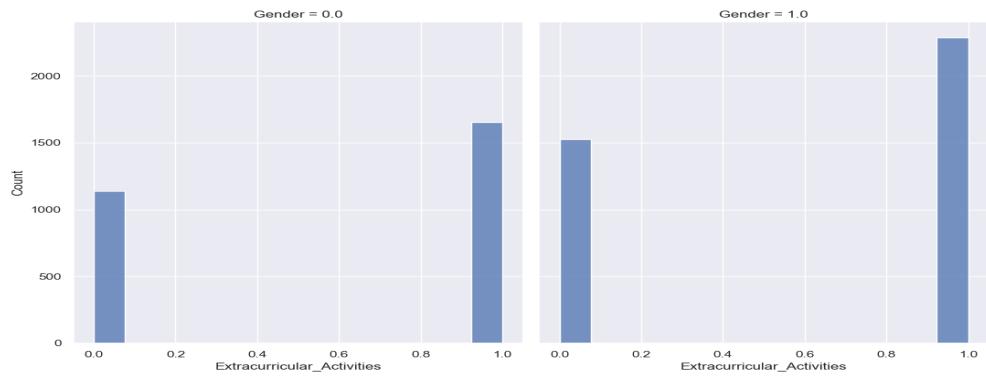


9. Scatter Plot of Hours_Studied vs Attendance

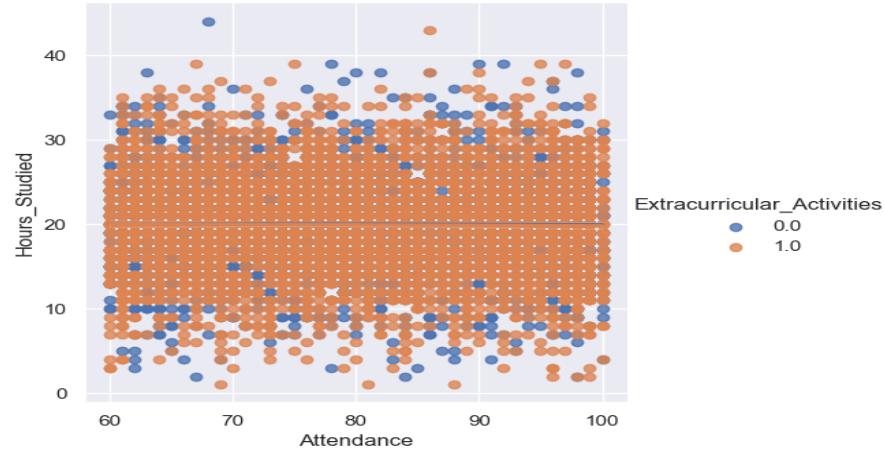


10. Scatter Plot of Hours_Studied vs Attendance vs Exam_Score





15. Implot of Hours_Studied vs Attendance vs Extracurricular Activities



1.5 DataOps: Implement workflows to automate activities from steps 1.3 and 1.4 within a data pipeline. Schedule these workflows to run every 2 minutes, logging all activity details and displaying them on a Cloud dashboard.

Workflow.py: -

The screenshot shows a GitHub repository named 'API_assignment_1'. The 'Code' tab is selected. The 'workflow.py' file is open, showing the following Python code:

```
from prefect import flow, task, get_run_logger
import subprocess
import os

@task(log_prints=True) # Ensures all prints are captured in logs
def run_task(script_name):
    logger = get_run_logger()

    # Define the full path to the script based on the project structure
    script_path = os.path.join(os.path.dirname(__file__), '/Users/mkmac/Documents/API_assignment_1/tasks', script_name)

    try:
        # Run the external Python script using its full path
        result = subprocess.run(['python', script_path], capture_output=True, text=True)
    except Exception as e:
        logger.error(f"An error occurred while running the task: {e}")

    return result
```

API_assignment_1 / flows / workflow.py

```

Code Blame 47 lines (38 loc) · 1.83 KB Code 55% faster with GitHub Copilot
6 def run_task(script_name):
15     # Log both stdout and stderr
16     if result.returncode == 0:
17         logger.info(f"Successfully executed {script_name}: \n{result.stdout}") # Log standard output
18     else:
19         logger.error(f"Error in {script_name}: {result.stderr}") # Log standard error
20
21     # Always print both stdout and stderr, regardless of success or failure
22     print(result.stdout)
23     print(result.stderr)
24
25
26 except Exception as e:
27     logger.error(f"Failed to execute {script_name}: {str(e)}")
28
29 return 0
30
31 @flow
32 def student_performance_factors():
33     # Run tasks sequentially and capture the results
34     data1 = run_task("Basicstats.py")
35     data2 = run_task("Binning.py", wait_for=[data1])
36     data3 = run_task("PearsonCorrelation.py", wait_for=[data2])
37     data4 = run_task("Encoding.py", wait_for=[data3])
38     data5 = run_task("CorrelationCoefficient.py", wait_for=[data4])

```

API_assignment_1 / flows / workflow.py

```

Code Blame 47 lines (38 loc) · 1.83 KB Code 55% faster with GitHub Copilot
6 def run_task(script_name):
7     except Exception as e:
8         logger.error(f"Failed to execute {script_name}: {str(e)}")
9
10    return 0
11
12 @flow
13 def student_performance_factors():
14     # Run tasks sequentially and capture the results
15     data1 = run_task("BasicStats.py")
16     data2 = run_task("Binning.py", wait_for=[data1])
17     data3 = run_task("PearsonCorrelation.py", wait_for=[data2])
18     data4 = run_task("Encoding.py", wait_for=[data3])
19     data5 = run_task("CorrelationCoefficient.py", wait_for=[data4])
20     data6 = run_task("FeatureImportanceMLAlgorithms.py", wait_for=[data5])
21     data7 = run_task("Visualization.py", wait_for=[data6])
22
23 # To run locally
24 if __name__ == "__main__":
25     student_performance_factors.serve(name="student_performance_factors",
26                                         tags=["studentperformancefactors datascience project workflow"],
27                                         parameters={},
28                                         interval=60)

```

Terminal

```

Terminal Shell Edit View Window Help
flows - Python workflow.py - 204x60
Last login: Thu Oct 31 16:05:04 on ttys008
mkmac@MKMac-MacBook-Air: ~ % cd /Users/mkmac/Documents/API_assignment_1
mkmac@MKMac-MacBook-Air API_assignment_1 % source /Users/mkmac/Documents/API_assignment_1/venv/bin/activate
(venv) mkmac@MKMac-MacBook-Air API_assignment_1 % cd flows
(venv) mkmac@MKMac-MacBook-Air flows % ls
prefect Cloud -- API Key.txt  figures
__pycache__  workflow.py
(venv) mkmac@MKMac-MacBook-Air flows % prefect cloud login
It looks like you're already authenticated on this profile.
? Would you like to reauthenticate? [y/n]: n
Using configuration from the current profile.
Authenticated with Prefect Cloud using workspace 'kamal-kumar-mandru/default'.
(venv) mkmac@MKMac-MacBook-Air flows % python workflow.py
Your flow 'student-performance-factors' is being served and polling for scheduled runs!
To trigger a run for this flow, use the following command:
$ prefect deployment run 'student-performance-factors/student_performance_factors'
You can also run your flow via the Prefect UI: https://app.prefect.cloud/account/b31b-8d1-2034-4493-8b4c-bcfca6e6c985/workspace/479354b6-1ff7-41a6-a2fa-d03eb91e6375/deployments/deployment/c61dc499-6e2e-44
8c-8abb-7fc55d77@e44

```

Scheduling workflows to run every 2 minutes

The screenshot shows the Prefect Cloud interface. On the left, a sidebar navigation includes Home, Runs, Flows, Deployments (which is selected and highlighted in blue), Work pools, Automations, Events, Incidents, Configuration, Blocks, Variables, Webhooks, and Concurrency. The main content area is titled 'Deployments' and shows three entries:

Deployment	Status	Activity	Tags	Schedules
student-performance-d...	Not Ready	first workflow	Every 2 minutes
student_performance_fa...	Not Ready	+1	Every minute
student_performance_fa...	Ready	+1	Every minute

At the bottom of the main content area, there are buttons for 'Items per page' (set to 10) and 'Page 1 of 1'.

logging activity details and displaying them on Cloud dashboard

The screenshot shows the Prefect Cloud interface. On the left, a sidebar navigation includes Home, Runs, Flows, Deployments (selected and highlighted in blue), Work pools, Automations, Events, Incidents, Configuration, Blocks, Variables, Webhooks, and Concurrency. The main content area is titled 'Deployments / student_performance_factors' and shows the following details for the 'student-performance-factors' flow:

Flow: student-performance-factors

- Schedules:** Every minute (toggle switch is green)
- Triggers:** + Add
- Status:** Ready
- Created:** 2024/10/31 12:03:54 PM
- Created By:** kappa82-togashi-v
- Last Updated:** 2024/10/31 04:11:08 PM
- Updated By:** kappa82-togashi-v

Metrics:

- SUCCESS RATE:** 100%
- AVERAGE LATENESS:** 0s
- AVERAGE DURATION:** 3m 45s

Deployment Details:

- Deployment:** student_performance_factors
- Flow:** student-performance-factors
- Run Type:** Scheduled
- Scheduled:** Scheduled for 2024/10/31 04:12:07 PM
- Parameters:** 0 Parameters

Terminal Shell Edit View Window Help flows - Python workflow.py - 204x60

```
? Would you like to reauthenticate? [y/N]: n
Using the existing authentication on this profile.
Authenticated with Prefect Cloud! Using workspace 'kamal-kumar-mandru/default'.
(venv) mkmac0MKMacs-MacBook-Air flows % python workflow.py
Your flow 'student-performance-factors' is being served and polling for scheduled runs!

To trigger a run for this flow, use the following command:

$ prefect deployment run 'student-performance-factors/student_performance_factors'

You can also run your flow via the Prefect UI: https://app.prefect.cloud/account/b31bc8d1-2034-4493-8b4c-bfcf65e5c985/workspace/479354b6-1ff7-41a6-a2fa-d03eb91e6375/deployments/deployment/c51dc499-6e2e-44
bc-6abb-77c5d778e4e

16:12:06.379 | INFO | prefect.flow_runs.runner - Runner 'student_performance_factors' submitting flow run 'c4861626-c863-4d77-a9ce-5f435086e94'
16:12:06.396 | INFO | prefect.flow_runs.runner - Opening process...
16:12:06.913 | INFO | prefect.flow_runs.runner - Completed submission of flow run 'c4861626-c863-4d77-a9ce-5f435086e94'
16:12:07.997 | INFO | Flow run 'lyrical-mule' - Downloading flow code from storage at https://app.prefect.cloud/api/accounts/b31bc8d1-2034-4493-8b4c-bfcf65e5c985工作空间/479354b6-1ff7-41a6-a2fa-d03eb91e6375/events/in
16:12:08.008 | INFO | Flow run 'lyrical-mule' - Flow run 'lyrical-mule' triggered by event https://app.prefect.cloud/api/accounts/b31bc8d1-2034-4493-8b4c-bfcf65e5c985工作空间/479354b6-1ff7-41a6-a2fa-d03eb91e6375/events/in
Please check your network settings to ensure websocket connections to the API are allowed. Otherwise event data (including task run data) may be lost. Reason: [SSL: CERTIFICATE_VERIFY_FAILED] certificate e verify failed: unable to get local issuer certificate (.ssl:::1000). Set PREFECT_DEBUG_MODE=1 to see the full error.
16:12:09.234 | ERROR | GlobalEventLoopThread | prefect._internal.concurrency - Service 'EventsWorker' failed with 2 pending items.
16:12:10.466 | INFO | Task run 'run_task-eff' - Successfully executed BasicStats.py:
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6607 entries, 0 to 6606
Data columns (total 17 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Hours_Studied  6607 non-null    int64  
 1   Attendance      6607 non-null    int64  
 2   Parental_Involvement 6607 non-null    object  
 3   Access_to_Resources 6607 non-null    object  
 4   Extracurricular_Activities 6607 non-null    object  
 5   Sleep_Hours     6607 non-null    int64  
 6   Previous_Scores 6607 non-null    int64  
 7   Motivation_Level 6607 non-null    object  
 8   Internet_Access 6607 non-null    object  
 9   Tutoring_Sessions 6607 non-null    int64  
 10  Family_Income   6607 non-null    object  
 11  School_Type     6607 non-null    object  
 12  Extracurricular 6607 non-null    object  
 13  Physical_Activity 6607 non-null    int64  
 14  Learning_Disabilities 6607 non-null    object  
 15  Gender          6607 non-null    object  
 16  Exam_Score      6607 non-null    int64  
dtypes: int64(7), object(10)
memory usage: 877.6+ KB
16:12:10.467 | INFO | Task run 'run_task-eff' - <class 'pandas.core.frame.DataFrame'>
RangeIndex: 6607 entries, 0 to 6606
Data columns (total 17 columns):
 #   Column           Non-Null Count  Dtype  
--- 
 0   Hours_Studied  6607 non-null    int64  
 1   Attendance      6607 non-null    int64  
 2   Parental_Involvement 6607 non-null    object  
 3   Access_to_Resources 6607 non-null    object  
 4   Extracurricular_Activities 6607 non-null    object  
 5   Sleep_Hours     6607 non-null    int64  
 6   Previous_Scores 6607 non-null    int64  
 7   Motivation_Level 6607 non-null    object  
 8   Internet_Access 6607 non-null    object  
 9   Tutoring_Sessions 6607 non-null    int64  
 10  Family_Income   6607 non-null    object  
 11  School_Type     6607 non-null    object  
 12  Extracurricular 6607 non-null    object  
 13  Physical_Activity 6607 non-null    int64  
 14  Learning_Disabilities 6607 non-null    object  
 15  Gender          6607 non-null    object  
 16  Exam_Score      6607 non-null    int64  
dtypes: int64(7), object(10)

Chrome File Edit View History Bookmarks Profiles Tab Window Help Thu 31 Oct 4:12:57 PM
kruns | Deploy | study | sager | mflfo | API-L | Mlflo | ssl- | how | Instal | M | APLx | +
app.prefect.cloud/account/b31bc8d1-2034-4493-8b4c-bfcf65e5c985/workspace/479354b6-1ff7-41a6-a2fa-d03eb91e6375/runs/flow-run/c4861626-c863-4... Dismiss Retry Opt out All Bookmarks

Gmail


Runs / student-performance-factors / student_performance_factors / lyrical-mule / Completed



Beta This is a beta preview of the new run page; make this the default by enabling it in feature previews Dismiss Opt out



16:11:30 16:11:40 16:11:50 16:12:00 16:12:10 16:12:20



This flow run did not generate any task or subflow runs



Filter Search Display



lyrical-mule +9



- prefect.flow-run.Scheduled 04:11:08 PM
- Runner 'student_performance_factor...' 04:12:05 PM
- Opening process... 04:12:05 PM
- Completed submission of flow run '... 04:12:05 PM
- Downloading flow code from storage... 04:12:07 PM
- prefect.flow-run.Running 04:12:08 PM
- prefect.flow-run.Completed 04:12:21 PM
- Finished in state Completed() 04:12:21 PM
- Process for flow run 'lyrical-mule...' 04:12:22 PM



Properties



|                 |                             |
|-----------------|-----------------------------|
| State           | Completed                   |
| Scheduled start | 2024/10/31 04:12:07 PM      |
| Start           | 2024/10/31 04:12:08 PM      |
| End             | 2024/10/31 04:12:21 PM      |
| Duration        | 14s                         |
| Run count       | 1/1                         |
| Creator         | IntervalSchedule            |
| Flow            | student-performance-factors |
| Run             | lyrical-mule                |
| Deployment      | student_performance_factors |
| Work pool       | None                        |
| Work queue      | None                        |
| Automation      | None                        |


```

Flows / student-performance-factors

The screenshot shows the Prefect Cloud interface for the 'student-performance-factors' workspace. On the left, a sidebar navigation bar includes Home, Runs, Flows (selected), Deployments, Work pools, Automations, Events, Incidents, Configuration (Blocks, Variables, Webhooks, Concurrency), Help, and a search bar for the workspace.

Flow Runs

7 total

Runs Deployments Details

Search by run name: All run states: Newest to oldest

- student-performance-factors > weightless-tarsier**: Scheduled for 2024/10/31 05:52:07 PM, 0 Parameters, Deployment: student_performance_factors
- student-performance-factors > peculiar-stallion**: Scheduled for 2024/10/31 05:51:07 PM, 0 Parameters, Deployment: student_performance_factors
- student-performance-factors > eager-nyala**: Scheduled for 2024/10/31 05:50:07 PM, 0 Parameters, Deployment: student_performance_factors
- student-performance-factors > conscious-mule**: auto-scheduled, studentperfomacefactors datasience project workflow

Task Runs

0 completed

Deployments / student_performance_factors

Run ▶

SUCCESS RATE
100%

AVERAGE LATENESS
0s

AVERAGE DURATION
3m 27s

Flow `student-performance-factors`

Schedules: Every minute (toggle switch)

Triggers: + Add

Status: Ready

Created: 2024/10/31 12:03:54 PM

Created By: kappa82-togashi-v

Last Updated: 2024/10/31 04:11:08 PM

Updated By: kappa82-togashi-v

Entrypoint: workflow.py:student_performance_factors

Path: .

Concurrency Limit: None

Next Run

- student-performance-factors > natural-kakapo**: Scheduled for 2024/10/31 04:14:07 PM, 0 Parameters, Deployment: student_performance_factors
- student-performance-factors > bold-penguin**: Running, 2024/10/31 04:13:07 PM, 0 Parameters, 7s, 0 Task runs, Deployment: student_performance_factors

8 Flow runs Search by run name: All except scheduled: Newest to oldest

The screenshot shows the Prefect Cloud interface with three main sections:

- Deployments / student_performance_factors**: A list of 8 flow runs for the workflow `student_performacefactors`. The runs are ordered from newest to oldest. Each run details its status (Running, Completed), date, parameters, duration, and task count.
- API Keys**: A table listing 6 API keys. Each key has columns for Name, Created, and Expiration. The keys listed are:

Name	Created	Expiration
cli-6b39ed58-9325-458f-be3a-9f4672e31a10	Oct 27th, 2024	2024/11/26 12:00:00 AM
cli-9b37d2ec-8dc0-48d7-85fd-149515989e76	Oct 27th, 2024	2024/11/26 12:00:00 AM
cli-f506f69e-8cf8-470f-a460-857994bb5f66	Oct 27th, 2024	2024/11/26 12:00:00 AM
cli-3e3d4e74-7471-4d16-b71c-e07f53b62beb	Oct 31st, 2024	2024/11/30 12:00:00 AM
cli-f47172b5-7dac-488d-ad8c-7cff4e618a0	Oct 31st, 2024	2024/11/30 12:00:00 AM
PREFECT_API_KEY	Oct 31st, 2024	Never

The screenshot shows the Account settings page in the Prefect Cloud interface. The left sidebar lists various settings categories:

- Account (selected)
- Workspaces
- Members
- Billing
- Controls
- PRO Features
- Audit Log
- SSO
- Service Accounts
- Roles
- Incidents
- ENTERPRISE Features

The main content area displays the following account information:

Account settings	
Name	Kamal Kumar Mandru
Handle	kamal-kumar-mandru
Billing Email	mandrukamalkumar@gmail.com
Location	None
External Link	None
Account ID	b31bc8d1-2034-4493-8b4c-bcfc65e5c985

Sub-Objective 2: Design and Development of a Machine Learning Pipeline

2.1 Model Preparation: Identify suitable machine learning algorithms for solving the business problem based on the dataset. Select any two algorithms.

For the dataset we used, we have opted for linear regression and random forest regression machine learning models for solving the business problems.

Linear Regression: -

A linear regression prediction model is a statistical tool used to predict the value of a dependent variable (response variable) based on one or more independent variables (predictor variables). This model assumes a linear relationship between the dependent and independent variables, meaning the change in the dependent variable is proportional to the change in the independent variables. Linear regression models are valued for their simplicity and interpretability, especially when the relationship between variables is approximately linear.

Random Forest Regression: -

A Random Forest Regression model is a predictive model that uses an ensemble (collection) of multiple decision trees to perform regression tasks. In this context, regression refers to predicting continuous numeric values, rather than discrete class labels. Random Forest Regression is well-suited for tasks like predicting housing prices, stock market values, or any scenario where continuous value prediction with reduced model variance is desired.

2.2 Model Training: Split the dataset into training (70%) and testing (30%) sets and train the models.

Below screenshot captured the sample split into 70-30 for training the model.

Where the total data samples are 6607, it has been divided as

4624 train samples and 1983 test samples as highlighted in yellow.

```
(venv) nkmacdMMac-MacBook-Air:mllops % ls
ml_flow.py mlartifacts mlruns
(venv) nkmacdMMac-MacBook-Air:mllops % python ml_flow.py
16:35:12.439 | INFO  | prefect.engine - Created flow run 'stoic-fox' for flow 'student_performance_factors_ml_pipeline'
16:35:12.442 | INFO  | prefect.engine - View at https://app.prefect.cloud/account/b31bc8d1-2034-4493-8b4c-bcf65e5c985/workspace/479354b6-1ff7-41a6-a2fa-d03eb91e6375/runs/flow-run/116afe47-11dc-4bcd-b01e-f96a1278e783
16:35:13.441 | INFO  | Task run 'model_preparation-347' - Preparing models: Linear Regression and Random Forest Regressor
16:35:13.458 | INFO  | Task run 'model_preparation-347' - Finished in state Completed()
16:35:13.474 | INFO  | Task run 'split_data-9a' - Data split completed: 4624 train samples, 1983 test samples
16:35:13.495 | INFO  | Task run 'split_data-9a' - Finished in state Completed()
16:35:13.571 | WARNING | prefect.events.clients - Unable to connect to 'https://api.prefect.cloud/api/accounts/b31bc8d1-2034-4493-8b4c-bcf65e5c985/workspaces/479354b6-1ff7-41a6-a2fa-d03eb91e6375/events/in'. Please check your network settings to ensure websocket connections to the API are allowed. Otherwise event data (including task run data) may be lost. Reason: [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: unable to get local issuer certificate (.ssl::1000). Set PREFECT_DEBUG_MODE=1 to see the full error.
16:35:13.571 | ERROR  | GlobalEventLoopThread | prefect._internal.concurrency - Service 'EventsWorker' failed with 8 pending items.
16:35:14.427 | INFO  | Task run 'train_models-9b' - Finished in state Completed()
16:35:14.703 | WARNING | prefect.events.clients - Unable to connect to 'https://api.prefect.cloud/api/accounts/b31bc8d1-2034-4493-8b4c-bcf65e5c985/workspaces/479354b6-1ff7-41a6-a2fa-d03eb91e6375/events/in'. Please check your network settings to ensure websocket connections to the API are allowed. Otherwise event data (including task run data) may be lost. Reason: [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: unable to get local issuer certificate (.ssl::1000). Set PREFECT_DEBUG_MODE=1 to see the full error.
16:35:14.704 | ERROR  | GlobalEventLoopThread | prefect._internal.concurrency - Service 'EventsWorker' failed with 3 pending items.
16:35:14.713 | INFO  | Task run 'evaluate_models-014' - Linear Regression - MSE: 4.27798524619834, MAE: 1.020321384998475, R2: 0.6886467116230378
16:35:14.816 | INFO  | Task run 'evaluate_models-014' - Random Forest Regressor - MSE: 4.70076817952597, MAE: 1.1697772566817955, R2: 0.6578764193041962
16:35:14.824 | INFO  | Task run 'evaluate_models-014' - Model evaluation completed
16:35:14.826 | INFO  | Task run 'evaluate_models-014' - Finished in state Completed()
16:35:14.911 | WARNING | prefect.events.clients - Unable to connect to 'https://api.prefect.cloud/api/accounts/b31bc8d1-2034-4493-8b4c-bcf65e5c985/workspaces/479354b6-1ff7-41a6-a2fa-d03eb91e6375/events/in'. Please check your network settings to ensure websocket connections to the API are allowed. Otherwise event data (including task run data) may be lost. Reason: [SSL: CERTIFICATE_VERIFY_FAILED] certificate verify failed: unable to get local issuer certificate (.ssl::1000). Set PREFECT_DEBUG_MODE=1 to see the full error.
16:35:14.912 | ERROR  | GlobalEventLoopThread | prefect._internal.concurrency - Service 'EventsWorker' failed with 1 pending items.
16:35:15.196 | INFO  | Flow run 'stoic-fox' - Finished in state Completed()
(venv) nkmacdMMac-MacBook-Air:mllops %
```

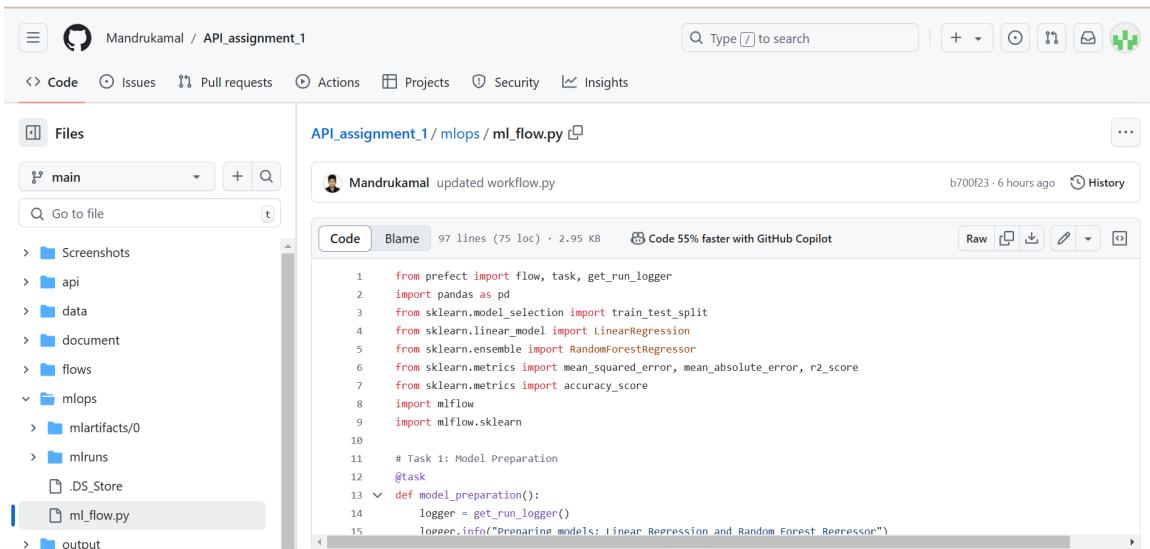
2.3 Model Evaluation: Evaluate the models using at least one metric (e.g., accuracy for classification models).

and

2.4 MLOps: Monitor the model and log relevant metrics (at least four, such as accuracy, precision, recall, F1 score, etc.)

Model evaluation

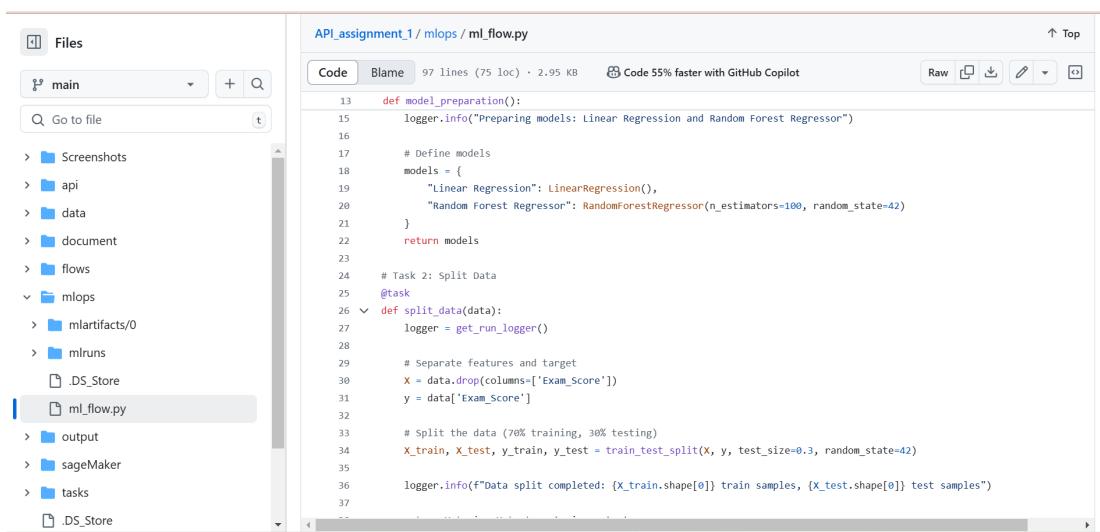
ML flow.py:-



The screenshot shows a GitHub repository interface for 'Mandrakamal / API_assignment_1'. The left sidebar shows a file tree with 'main' selected. The main pane displays the 'ml_flow.py' file content. The code imports various libraries from sklearn and prefect, defines a logger, and a task named 'model_preparation' which logs the preparation of Linear Regression and Random Forest Regressor models.

```
from prefect import flow, task, get_run_logger
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LinearRegression
from sklearn.ensemble import RandomForestRegressor
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score
from sklearn.metrics import accuracy_score
import mlflow
import mlflow.sklearn

# Task 1: Model Preparation
@task
def model_preparation():
    logger = get_run_logger()
    logger.info("Preparing models: Linear Regression and Random Forest Regressor")
```



This screenshot shows the same GitHub repository and file structure as the previous one, but the code content is more extensive. It includes the 'model_preparation' function and adds a 'split_data' function that splits the data into training and testing sets using 'train_test_split' with a test size of 0.3 and random state of 42. It also logs the completed data split.

```
def model_preparation():
    logger.info("Preparing models: Linear Regression and Random Forest Regressor")

    # Define models
    models = {
        "Linear Regression": LinearRegression(),
        "Random Forest Regressor": RandomForestRegressor(n_estimators=100, random_state=42)
    }
    return models

# Task 2: Split Data
@task
def split_data(data):
    logger = get_run_logger()

    # Separate features and target
    X = data.drop(columns=['Exam_Score'])
    y = data['Exam_Score']

    # Split the data (70% training, 30% testing)
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

    logger.info(f"Data split completed: {X_train.shape[0]} train samples, {X_test.shape[0]} test samples")
```

Files

- Screenshots
- api
- data
- document
- flows
- mlops
 - mlartifacts/0
 - mlruns
- .DS_Store
- ml_flow.py
- output
- sageMaker
- tasks
- .DS_Store

API_assignment_1 / mlops / ml_flow.py

```
Code Blame 97 lines (75 loc) · 2.95 KB Code 55% faster with GitHub Copilot
26     def split_data(data):
27         logger.info(f"Data split completed: {X_train.shape[0]} train samples, {X_test.shape[0]} test samples")
28
29         return X_train, X_test, y_train, y_test
30
31     # Task 3: Train Models
32     @task
33     def train_models(X_train, y_train, models):
34         trained_models = {}
35
36         for model_name, model in models.items():
37             model.fit(X_train, y_train)
38             trained_models[model_name] = model
39
40         return trained_models
41
42     # Task 4: Evaluate Models
43     @task
44     def evaluate_models(X_test, y_test, trained_models):
45         logger = get_ru_logger()
46
47         for model_name, model in trained_models.items():
48             # Predictions
49             y_pred = model.predict(X_test)
```

Files

- Screenshots
- api
- data
- document
- flows
- mlops
 - mlartifacts/0
 - mlruns
- .DS_Store
- ml_flow.py
- output
- sageMaker
- tasks
- .DS_Store

API_assignment_1 / mlops / ml_flow.py

```
Code Blame 97 lines (75 loc) · 2.95 KB Code 55% faster with GitHub Copilot
53     def evaluate_models(X_test, y_test, trained_models):
54         y_pred = model.predict(X_test)
55
56         # Metrics
57         mse = mean_squared_error(y_test, y_pred)
58         mae = mean_absolute_error(y_test, y_pred)
59         r2 = r2_score(y_test, y_pred)
60
61         logger.info(f"{model_name} - MSE: {mse}, MAE: {mae}, R2: {r2}")
62
63         # Log metrics using MLflow
64         with mlflow.start_run(run_name=model_name):
65             mlflow.log_param("Model", model_name)
66             mlflow.log_metric("MSE", mse)
67             mlflow.log_metric("MAE", mae)
68             mlflow.log_metric("R2", r2)
69
70         logger.info("Model evaluation completed")
71
72     # Main Flow
73     @flow(name="student_performance_factors_ml_pipeline")
74     def student_performance_factors(file_path):
75         # Load data (assumes data preprocessing has been done)
76         data = pd.read_csv(file_path)
```

Files

- Screenshots
- api
- data
- document
- flows
- mlops
 - mlartifacts/0
 - mlruns
- .DS_Store
- ml_flow.py
- output
- sageMaker
- tasks
- .DS_Store

API_assignment_1 / mlops / ml_flow.py

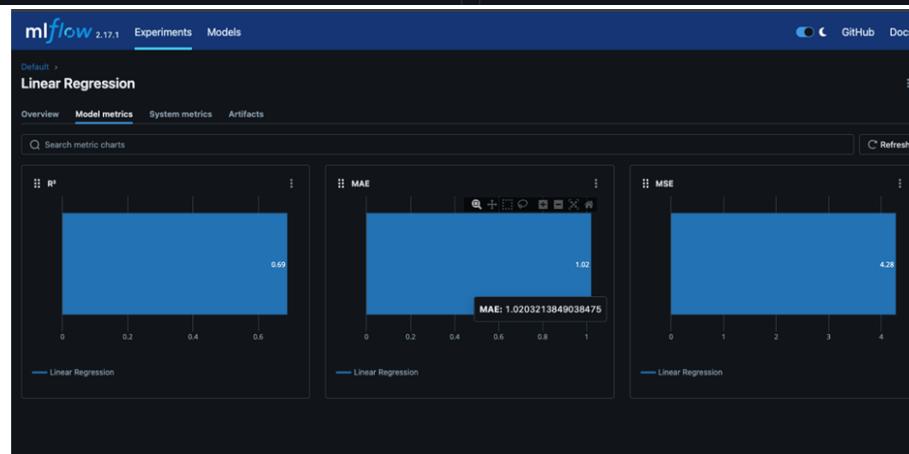
```
Code Blame 97 lines (75 loc) · 2.95 KB Code 55% faster with GitHub Copilot
76     # Main Flow
77     @flow(name="student_performance_factors_ml_pipeline")
78     def student_performance_factors(file_path):
79         # Load data (assumes data preprocessing has been done)
80         data = pd.read_csv(file_path)
81
82         # Model Preparation
83         models = model_preparation()
84
85         # Split Data
86         X_train, X_test, y_train, y_test = split_data(data)
87
88         # Train Models
89         trained_models = train_models(X_train, y_train, models)
90
91         # Evaluate Models
92         evaluate_models(X_test, y_test, trained_models)
93
94         # Run the pipeline
95         if __name__ == "__main__":
96             file_path = "/Users/mkmac/Documents/API_assignment_1/data/ImputedStudentPerformanceFactors.csv" # Update the path to your dataset
97             student_performance_factors(file_path)
```

Linear Regression: -

The screenshot shows the mlflow UI for an experiment named "Linear Regression". The "Overview" tab is selected. The experiment was created at 2024-10-31 16:35:14 by "mkmac". It has an Experiment ID of 0, a status of "Finished", and a Run ID of 68e8a8bcc4004c48b0d1e0a490484603. The duration is 7ms, and no datasets or tags were used. The source is "ml_flow.py", and there are no logged or registered models.

Metrics captured below;

The screenshot shows the mlflow UI for the same experiment. The "Overview" tab is selected. The experiment details remain the same. In the "Metrics (3)" section, three metrics are listed: R² with a value of 0.6886467116230378, MAE with a value of 1.0203213849038475, and MSE with a value of 4.277085246198344.



Random Forest Regression: -

The image shows two screenshots of the mlflow UI. The top screenshot displays the 'Experiments' page with a table of runs. The bottom screenshot shows a detailed view of a specific experiment named 'Random Forest Regressor'.

Experiments Page (Top Screenshot):

Run Name	Created	Dataset	Duration	Source	Models
Random Forest Regressor	21 seconds ago	-	5ms	ml_flow...	-
Linear Regression	21 seconds ago	-	7ms	ml_flow...	-

Random Forest Regressor Experiment View (Bottom Screenshot):

Details
Created at: 2024-10-31 16:35:14
Created by: mkmac
Experiment ID: 0
Status: Finished
Run ID: 1dad74453c1942559375548334beade4
Duration: 5ms
Datasets used: -
Tags: Add
Source: ml_flow.py
Logged models: -
Registered models: -

Metrics captured below:

The screenshot shows the detailed view of the 'Random Forest Regressor' experiment, focusing on the 'Metrics' section.

Metric	Value
R ²	0.6578764193041962
MAE	1.1696772566817955
MSE	4.70076817952597



MLOps: Monitoring the model and logging relevant metrics

ML Models deployment in SageMaker: -

The screenshot shows a GitHub code editor for a file named 'sagemaker_api.py' under the 'sageMaker' directory. The code is as follows:

```

import boto3
# Create a low-level client representing Amazon SageMaker Runtime
sagemaker_runtime = boto3.client("sagemaker-runtime", region_name='ap-south-1')
# The endpoint name
endpoint_name = 'canvas-studentPredictedDeployment'
# Specify the content type of the input data
content_type = 'application/json'
# Gets inference from the model hosted at the specified endpoint:
response = sagemaker_runtime.invoke_endpoint(
    EndpointName=endpoint_name,
    ContentType=content_type,
    Body=b64bytes(json.dumps({"features": [23.0, 84.0, 1.0, 0.0, 0.0, 7.0, 73.0, 1.0, 1.0, 0.0, 1.0, 2.0, 1.0, 2.0, 3.0, 0.0, 1.0, 2.0, 1.0]}), 'utf-8')
)
# Decodes and prints the response body:
print(response['Body'].read().decode('utf-8'))

```

The screenshot shows the Amazon SageMaker Canvas interface. On the left, there's a sidebar with icons for Data Wrangler, Datasets, My Models, ML Ops, Ready-to-use, and Gen AI. The main area has sections like 'Get started', 'Learn to use SageMaker Canvas', 'End to end ML', 'GenAI in Canvas', 'Free walkthrough', 'Recent work', and 'Canvas news'. The 'Canvas news' section includes a blog post titled 'Unlock the power of data governance and no-code machine learning with Amazon SageMaker Canvas and Amazon DataZone' posted on Aug 21, 2024.

The screenshot displays three sequential steps in a web-based data import interface:

Step 1: Upload files to import

A modal window titled "ImputedStudentPerf... : Create Tabular dataset". It shows a file named "imputedStudentPerformanceFactors.csv" ready to import. The interface includes fields for selecting a data source (Local upload) and options to drag a CSV or Parquet file or select files from the computer.

Step 2: Import preview

The modal shows the first 100 rows of the dataset. A preview table displays columns: Hours_Studied, Attendance, Parental_Involve..., Access_to_Re..., Extracurricula..., and Sleep_Hours. Row details are as follows:

Hours_Studied	Attendance	Parental_Involve...	Access_to_Re...	Extracurricula...	Sleep_Hours
23.0	84.0	1.0	0.0	0.0	7.0
19.0	64.0	1.0	2.0	0.0	8.0
24.0	96.0	2.0	2.0	1.0	7.0
29.0	89.0	1.0	2.0	1.0	8.0
19.0	92.0	2.0	2.0	1.0	6.0
19.0	88.0	2.0	2.0	1.0	8.0

Step 3: Datasets page

The main interface shows the "Datasets" section. A sidebar on the left includes icons for Home, Data Wrangler (selected), My Models, ML Ops, Ready-to-use, Gen AI, Help, and Log out. The datasets table lists various datasets, including "imputedStudentPerformanceFactors" and several sample datasets like "canvas-sample-data" and "canvas-sample-maintenance".

Screenshot of the SageMaker Canvas interface showing the "Select" tab for a studentPerformancePredictor model version 1.

Select Tab Content:

- Message:** "There's a new way to join data. You can now join datasets by creating a data flow in Data Wrangler. [Learn how to join data](#)
- Import and prepare:** [Import and prepare](#)
- Select dataset:** [+ Create dataset](#)
- Search bar:** Search datasets in Canvas
- Dataset List:**

All	Joined
ImputedStudentPerformanceFactors	V1 20 6,607 1,32,140 10/31/2024 2:50 PM Ready
canvas-sample-databricks-dolly-15k.csv	V1 2 10,879 21,758 10/31/2024 2:48 PM Ready
canvas-sample-maintenance.csv	V1 9 1,000 9,000 10/31/2024 2:48 PM Ready
canvas-sample-loans-part-1.csv	V1 19 1,000 19,000 10/31/2024 2:48 PM Ready
canvas-sample-retail-electronics-forecasting.csv	V1 6 40,500 2,43,000 10/31/2024 2:48 PM Ready
- Select dataset button:** [Select dataset](#)

Build Tab Content:

- Select a column to predict:** Choose the target column. The model that you build predicts values for the column that you select. **Target column:** Exam_Score
- Value distribution:** A histogram showing the distribution of Exam_Score values between 55.00 and 82.55.
- Model type:** SageMaker Canvas automatically recommends the appropriate model type for your analysis. **3+ category prediction:** Your model classifies Exam_Score into 3 or more categories. **Configure model:** [Configure model](#)
- Build Options:** [Quick build](#) [Preview model](#)

Data Wrangler Tab Content:

- Dataset Overview:** ImputedStudentPerformanceFactors, Full dataset: 6.6k rows
- Column Details:**

Column name	Data type	Feature type	Missing	Mismatched	Unique	Mode
Tutoring_Sessions	123 Numeric	-	0.00% (0)	0.00% (0)	9	1
Teacher_Quality	123 Numeric	-	0.00% (0)	0.00% (0)	4	2
Sleep_Hours	123 Numeric	-	0.00% (0)	0.00% (0)	7	7
School_Type	123 Numeric	Binary	0.00% (0)	0.00% (0)	2	1
Previous_Scores	123 Numeric	-	0.00% (0)	0.00% (0)	51	66
- Total statistics:** Total columns: 20, Total rows: 6,607, Total cells: 1,32,140

My models > studentPerformancePredictor > Version 1

Model overview

Your model is being created. Quick build usually takes 2–15 minutes. You can now leave this view.

Expected build time: 2–15 minutes

Build type: Quick build

Detailed progress: Generating column impact

Metrics captured below

My models > studentPerformancePredictor > Version 1

Model status: Quick build

Accuracy: 32.186%

The model predicts the correct Exam_Score 32.186% of the time.

Column impact:

Column	Impact (%)
Attendance	21.993%
Hours_Studied	12.886%
Access_to_Resources	9.398%
Parental_Involvement	9.02%

Impact of Attendance on prediction of Exam_Score

Impact on prediction: 100.0

Attendance: All other classes

Screenshot of the AWS SageMaker Canvas interface showing the Predict and Deploy tabs.

Predict Tab

Predict target values:

- Single prediction** (selected)
- Batch prediction**

Modify values to predict Exam_Score in real time.

Column	Value
Hours_Studied	20
Attendance	67
Parental_Involvement	2
Access_to_Resources	2
Extracurricular_Activities	1

Exam_Score Prediction: **65.0**

New prediction Last prediction

100.0 0% 0 101.0 0% [Download prediction](#)

Deploy Tab

No result found

make predictions from outside of the Canvas application, test and monitor your model to proactively detect issues such as model drift.

Selected model version: studentPerformancePredictor

v1 Ready Created: 10-31-2024-2:58 PM

Deployment type: Real-time

Deployment name: Deployment name: studentPredictedDeployment

Instance type: ml.t2.medium

Instance count: 1

Buttons: Cancel, Deploy

The screenshot shows two consecutive screenshots of the SageMaker Canvas interface, both taken at Thu 31 Oct 2:59:04 PM.

Screenshot 1: The user is in the "Deployments" section of the "studentPerformancePredictor" model's version 1. The "Deploy" tab is selected. A deployment named "studentPredictedDeployment" is listed with status "Creating". The deployment URL is https://runtime.sagemaker.ap-south-1.amazonaws.com/... . The deployment was created on 10/31/24 02:59 PM.

Deployment name	Status	Deployment URL	Created
studentPredictedDeployment	Creating	https://runtime.sagemaker.ap-south-1.amazonaws.com/...	10/31/24 02:59 PM

Screenshot 2: The user has completed the deployment process. A success message "studentPerformancePredictor was deployed" is displayed. The "Create new version" button is visible in the top right corner.

Screenshot of the AWS SageMaker Canvas interface showing a deployment named "canvas-studentPredictedDeployment".

Deployment Details:

Deployment name	Status	Deployment type	Model
canvas-studentPredictedDeployment	In service	Real-time	studentPerformancePredictor v1

Created: 10/31/24 02:59 PM

Instance type: ml.t2.medium

Average predictions per day: --

Instance count: 1

Last prediction: --

Inference response content: predicted_label, probability, probabilities, labels

Input format: text/csv

Deployment URL: <https://runtime.sagemaker.ap-south-1.amazonaws.com/endpoints/canvas-studentPredictedDeployment/invocations>

View sample code

Help | **Log out**

Access key created

This is the only time that the secret access key can be viewed or downloaded. You cannot recover it later. However, you can create a new access key any time.

Step 1
Access key best practices & alternatives

Step 2 - optional
Set description tag

Step 3
Retrieve access keys

Retrieve access keys

Access key
If you lose or forget your secret access key, you cannot retrieve it. Instead, create a new access key and make the old key inactive.

Access key	Secret access key
AKIARJU2L6KQBU4TKTHF	KWsCzygOeGqQIByGlcZeejPSdZDC9FuX/EjbPr4R Hide

Access key best practices

- Never store your access key in plain text, in a code repository, or in code.
- Disable or delete access key when no longer needed.
- Enable least-privilege permissions.
- Rotate access keys regularly.

For more details about managing access keys, see the [best practices for managing AWS access keys](#).

[Download .csv file](#) | [Done](#)

CloudShell | Feedback | © 2024, Amazon Web Services, Inc. or its affiliates. | Privacy | Terms | Cookie preferences

Sub-Objective 3: API Access

3.1 Retrieve Key Application Details: Use Built-in APIs to access important application information (e.g., flow, deployment etc.)

The screenshot shows a GitHub repository interface. On the left, there's a sidebar with a tree view of files and folders. The main area displays a file named 'Prefect Cloud -- API Key.txt'. The file content is a series of numbered steps for logging into a Prefect account. Step 1 shows environment variable setup. Step 2 shows the command to save the key to the active profile. Step 3 shows the command to log in with the account ID. Step 4 shows the command to get the workspace ID. Step 5 shows the command to set the workspace ID. A GitHub Copilot badge indicates the code is 55% faster with Copilot.

```
1. PREFECT_API_KEY
2.
3. set PREFECT_API_KEY=pnu_M6BV8jNMS1drgrro0jwdit0hhcyNup2dzRLW
4. prefect cloud login --key pnu_ev1mrDrRHZhK36CsR5rAlYOnN1KqKO3smWqG
5.
6. 2. run this command to save the key to your active profile
7. prefect cloud login -k pnu_ev1mrDrRHZhK36CsR5rAlYOnN1KqKO3smWqG
8. prefect cloud login -k pnu_ev1mrDrRHZhK36CsR5rAlYOnN1KqKO3smWqG --workspace "kamal-kumar-mandru/default"
9.
10. prefect cloud login -k cli-6b39ed58-9325-458f-be3a-9f4672e31a10
11.
12. 3. Prefect Account Id -> b31bc8d1-2034-4493-8b4c-bcfcc65e5c985
13.
14. 4. Workspace Id -> 479354b6-1ff7-41a6-a2fa-d03eb91e6375
15.
16. 5. Command to set workspace Id -> prefect cloud workspace set --workspace "kamal-kumar-mandru/default"
```

API.py: -

Mandrukamal / API_assignment_1

Type ⌂ to search

Code Issues Pull requests Actions Projects Security Insights

Files

main Go to file

Screenshots

api

- api.py
- deploymentAPI.py
- flowAPI.py

data document flows mlops output sageMaker

API_assignment_1 / api / api.py

Mandrukamal Added process folder in screenshots 3df0c03 · 1 hour ago History

Code Blame 39 lines (31 loc) · 1.3 KB Code 55% faster with GitHub Copilot

```
1 # URL - https://app.prefect.cloud/account/8ff8f613-92c4-44ce-b811-f9956023e78d/workspace/04d8fc9a9-df2e-40c8-ae4f-a3733114c47
2
3 # Ref - https://app.prefect.cloud/api/docs
4
5 import requests
6
7 # Replace these variables with your actual Prefect Cloud credentials
8 PREFECT_API_KEY = "pnu_eVlmPrrHzHk36CSr5rA1Y0nN1KqO3smMqQ" # Your Prefect Cloud API key
9 ACCOUNT_ID = "b1b1c8d1-2034-4493-8b4c-bfcf65e5e985" # Your Prefect Cloud Account ID
10 WORKSPACE_ID = "479354b6-1ff7-41a6-a2fa-d03eb91e6375" # Your Prefect Cloud Workspace ID
11
12 # Correct API URL to list flow runs
13 PREFECT_API_URL = f"https://api.prefect.cloud/api/accounts/{ACCOUNT_ID}/workspaces/{WORKSPACE_ID}/artifacts/filter"
14
15 # Data to filter artifacts
```

Mandrukamal / API_assignment_1

Type ⌂ to search

Code Issues Pull requests Actions Projects Security Insights

Files

main Go to file

Screenshots

api

- api.py
- deploymentAPI.py
- flowAPI.py

data document flows mlops output sageMaker tasks .DS_Store .gitattributes

API_assignment_1 / api / api.py

Code Blame 39 lines (31 loc) · 1.3 KB Code 55% faster with GitHub Copilot

```
15 # Data to filter artifacts
16 data = {
17     "sort": "CREATED_DESC",
18     "limit": 5,
19     "artifacts": {
20         "key": {
21             "exists_": True
22         }
23     }
24 }
25
26 # Set up headers with Authorization
27 headers = {"Authorization": f"Bearer {PREFECT_API_KEY}"}
28
29 # Make the request
30 response = requests.post(PREFECT_API_URL, headers=headers, json=data)
31 print(response)
32
33 # Check the response status
34 if response.status_code != 200:
35     print(f"Error: Received status code {response.status_code}")
36     print(f"connection is bot succesful {response.status_code}")
37     print(f"Response content: {response.text}")
38 else:
```

The screenshot shows the GitHub interface for the `API_assignment_1` repository. On the left, the file tree shows a `main` folder containing `Screenshots`, `api`, and `.DS_Store`. The `api` folder contains `deploymentAPI.py`, `flowAPI.py`, and `api.py`, with `api.py` being the active file. The code in `api.py` is as follows:

```

16     data = {
17         "artifacts": {
18             "key": {
19                 "exists": True
20             }
21         }
22     }
23
24
25
26     # Set up headers with Authorization
27     headers = {"Authorization": f"Bearer {PREFECT_API_KEY}"}
28
29     # Make the request
30     response = requests.post(PREFECT_API_URL, headers=headers, json=data)
31     print(response)
32
33     # Check the response status
34     if response.status_code != 200:
35         print(f"Error: Received status code {response.status_code}")
36         print(f"connection is bot succesful {response.status_code}")
37         print(f"Response content: {response.text}")
38     else:
39         print("connection is succesful")

```

DeploymentAPI.py: -

The screenshot shows the GitHub interface for the `API_assignment_1` repository. On the left, the file tree shows a `main` folder containing `Screenshots`, `api`, and `.DS_Store`. The `api` folder contains `deploymentAPI.py` and `flowAPI.py`, with `deploymentAPI.py` being the active file. The code in `deploymentAPI.py` is as follows:

```

1     # URL - https://app.prefect.cloud/account/8ff8f613-92c4-44ce-b811-f9956023e78d/workspace/0ad8fc9a-df2e-40c8-ae4f-a3733114c475/dashboard
2
3     # URL - https://app.prefect.cloud/api/docs
4
5     import requests
6
7     # Replace these variables with your actual Prefect Cloud credentials
8     PREFECT_API_KEY = "pnu_eV1mrDrriZhK36CsR5rAlY0mNIkq03smig" # Your Prefect Cloud API key
9     ACCOUNT_ID = "b31bc8d1-2034-4493-8b4c-bfc6565e985" # Your Prefect Cloud Account ID
10    WORKSPACE_ID = "479354b6-1ff7-41a6-a2fa-d03eb91e6375" # Your Prefect Cloud Workspace ID
11    DEPLOYMENT_ID = "c51dc499-6e2e-440c-8abb-7fc55d770e4e" # Your Deployment ID
12
13    # Correct API URL to get deployment details
14    PREFECT_API_URL = f"https://api.prefect.cloud/api/accounts/{ACCOUNT_ID}/workspaces/{WORKSPACE_ID}/deployments/{DEPLOYMENT_ID}"
15
16    # Set up headers with Authorization
17    headers = {"Authorization": f"Bearer {PREFECT_API_KEY}"}
18
19    # Make the request using GET
20    response = requests.get(PREFECT_API_URL, headers=headers)
21

```

The screenshot shows the GitHub interface for the `API_assignment_1` repository. On the left, the file tree shows a `main` folder containing `Screenshots`, `api`, and `.DS_Store`. The `api` folder contains `deploymentAPI.py`, `flowAPI.py`, and `api.py`, with `deploymentAPI.py` being the active file. The code in `deploymentAPI.py` is identical to the one shown in the previous screenshot.

FlowAPI.py:-

The screenshot shows the GitHub interface for the file `flowAPI.py`. The code is a Python script for interacting with Prefect Cloud. It includes comments for URLs, API keys, account ID, workspace ID, and flow ID. It sets up headers for Authorization and makes a GET request to the Prefect API to retrieve flow details. The code is 28 lines long and was last updated 6 hours ago.

```
1 # URL - https://app.prefect.cloud/account/bff8f613-92c4-44ce-b811-f9956023e78d/workspace/04d8fc9-df2e-40c8-ae4f-a3733114c475/dashboard
2
3 # URL - https://app.prefect.cloud/api/docs
4
5 import requests
6
7 # Replace these variables with your actual Prefect Cloud credentials
8 PREFECT_API_KEY = "pnu_eV1mrDrrZhK36CsR5rAlYOnN1KqKO3smWqG" # Your Prefect Cloud API key
9 ACCOUNT_ID = "b31bc8d1-2034-4493-8b4c-bcf65e5c985" # Your Prefect Cloud Account ID
10 WORKSPACE_ID = "479354b6-1ff7-41a6-a2fa-d03eb91e6375" # Your Prefect Cloud Workspace ID
11 FLOW_ID = "fee0b8ea-4107-411b-93d7-d024be07c700" # Your Flow ID
12
13 # Correct API URL to get flow details
14 PREFECT_API_URL = f"https://api.prefect.cloud/api/accounts/{ACCOUNT_ID}/workspaces/{WORKSPACE_ID}/flows/{FLOW_ID}"
15
16 # Set up headers with Authorization
17 headers = {"Authorization": f"Bearer {PREFECT_API_KEY}"}
18
19 # Make the request using GET
20 response = requests.get(PREFECT_API_URL, headers=headers)
```

This screenshot shows the same `flowAPI.py` file on GitHub, but with additional code at the bottom to handle the response status. It checks if the status code is 200, prints the flow info if it is, and prints an error message if it's not. The code is now 28 lines long.

```
7 # Replace these variables with your actual Prefect Cloud credentials
8 PREFECT_API_KEY = "pnu_eV1mrDrrZhK36CsR5rAlYOnN1KqKO3smWqG" # Your Prefect Cloud API key
9 ACCOUNT_ID = "b31bc8d1-2034-4493-8b4c-bcf65e5c985" # Your Prefect Cloud Account ID
10 WORKSPACE_ID = "479354b6-1ff7-41a6-a2fa-d03eb91e6375" # Your Prefect Cloud Workspace ID
11 FLOW_ID = "fee0b8ea-4107-411b-93d7-d024be07c700" # Your Flow ID
12
13 # Correct API URL to get flow details
14 PREFECT_API_URL = f"https://api.prefect.cloud/api/accounts/{ACCOUNT_ID}/workspaces/{WORKSPACE_ID}/flows/{FLOW_ID}"
15
16 # Set up headers with Authorization
17 headers = {"Authorization": f"Bearer {PREFECT_API_KEY}"}
18
19 # Make the request using GET
20 response = requests.get(PREFECT_API_URL, headers=headers)
21
22 # Check the response status
23 if response.status_code == 200:
24     flow_info = response.json()
25     print(flow_info)
26 else:
27     print(f"Error: Received status code {response.status_code}")
28     print(f"Response content: {response.text}")
```

3.2 Display Application Details: Present at least four application details retrieved via APIs.

The screenshot shows a terminal window on a Mac OS X system. The user is in a virtual environment named `(venv)` and runs the command `python api.py`. The output shows a successful connection to the API, indicating that the application details were retrieved correctly.

```
[(venv) mkmac@MKMacs-MacBook-Air api % python api.py
<Response [200]>
connection is successful
(venv) mkmac@MKMacs-MacBook-Air api % ]
```

```
[(venv) mkmac@MKMacs-MacBook-Air api % python deploymentAPI.py
{'id': 'c51dc499-6e2e-440c-8abb-7fc55d770e4e', 'created': '2024-10-31T06:33:54.495843+00:00', 'updated': '2024-10-31T10:41:08.732686+00:00', 'infra_overrides': {}, 'name': 'student_performance_factors', 'version': '88a0d8e41becaa7ed13650a64677a6b9', 'description': 'None', 'flow_id': 'fee0b8ea-4107-411b-93d7-d024be07c700', 'schedule': {'interval': 60.0, 'anchor_date': '2024-10-31T16:11:07.832464+05:30', 'timezone': 'Asia/Kolkata'}, 'is_schedule_active': True, 'paused': False, 'disabled': False, 'schedules': [{'id': 'c7b01e30-f0e8-44ae-b201-8d1fd43c9766', 'created': '2024-10-31T10:41:08.750504+00:00', 'updated': '2024-10-31T10:45:33.953915+00:00', 'deployment_id': 'c51dc499-6e2e-440c-8abb-7fc55d770e4e', 'schedule': {'interval': 60.0, 'anchor_date': '2024-10-31T16:11:07.832464+05:30', 'timezone': 'Asia/Kolkata'}, 'active': False, 'last_scheduled_at': None, 'soonest_scheduled_run': None, 'latest_scheduled_run': None, 'max_active_runs': None, 'max_scheduled_runs': None, 'catchup': False}, {'id': 'c7b01e30-f0e8-44ae-b201-8d1fd43c9766', 'created': '2024-10-31T10:41:08.750504+00:00', 'updated': '2024-10-31T10:45:33.953915+00:00', 'deployment_id': 'c51dc499-6e2e-440c-8abb-7fc55d770e4e', 'schedule': {'interval': 60.0, 'anchor_date': '2024-10-31T16:11:07.832464+05:30', 'timezone': 'Asia/Kolkata'}, 'active': False, 'last_scheduled_at': None, 'soonest_scheduled_run': None, 'latest_scheduled_run': None, 'max_active_runs': None, 'max_scheduled_runs': None, 'catchup': False}], 'global_concurrency_limit': None, 'concurrency_options': None, 'job_variables': {}, 'parameters': {}, 'tags': ['studentperformancefactors datascience project workflow'], 'labels': {'prefect.flow.id': 'fee0b8ea-4107-411b-93d7-d024be07c700'}, 'work_queue_name': None, 'last_polled': '2024-10-31T0:46:54.772173+00:00', 'parameter_openapi_schema': {'type': 'object', 'title': 'Parameters', 'properties': {}}, 'path': '.', 'pull_steps': [], 'entrypoint': 'workflow.py:student_performance_factors', 'manifest_path': None, 'storage_document_id': None, 'infrastructure_document_id': None, 'created_by': {'id': 'fc70383c-a362-43a4-935e-933ba919e26b', 'type': 'USER', 'display_value': 'kappa82-togashi-v'}, 'updated_by': {'id': 'fc70383c-a362-43a4-935e-933ba919e26b', 'type': 'USER', 'display_value': 'kappa82-togashi-v'}, 'work_pool_name': None, 'status': 'NOT_READY', 'enforce_parameter_schema': True}
(venv) mkmac@MKMacs-MacBook-Air api %
```

```
[(venv) mkmac@MKMacs-MacBook-Air api % ls
api.py           deploymentAPI.py      flowAPI.py
(venv) mkmac@MKMacs-MacBook-Air api % python flowAPI.py
{'id': 'fee0b8ea-4107-411b-93d7-d024be07c700', 'created': '2024-10-31T06:33:54.130514+00:00', 'updated': '2024-10-31T06:33:54.130522+00:00', 'name': 'student-performance-factors', 'tags': [], 'labels': []}
(venv) mkmac@MKMacs-MacBook-Air api % []
```