Group No : 52

Group Member Names:

1. CHAGANTI S S V S LAKSHMINARAYANA(2023ad05097@wilp.bits-pilani.ac.in)- contribution -100%
2. PRADYOTHANA D P(2023ad05059@wilp.bits-pilani.ac.in) - contribution -100%
3. BRAJESH KUMAR SETHI(2023ac05335@wilp.bits-pilani.ac.in)- contribution -100%
4. BHOGE ANKIT ARUN MADHURI(2023dc04324@wilp.bits-pilani.ac.in)- contribution -100%

Feature Store and Data Versioning Documentation

Feature Store Implementation

Overview

We implemented a Feature Store using a custom CSV-based approach to manage machine learning features efficiently. This allows us to store, retrieve, and serve features for training and inference without using Feast's built-in materialization methods.

Implementation Steps

1. Initialize the Feature Store

Method Explanation

This method creates a synthetic dataset for a feature store using Python's pandas, numpy, and random libraries. The steps include:

Data Generation : A dataset with 100 customer records is created, including customer_id, total_spend, tenure, churn, and last_purchase.

Data Processing : Integer-based features are converted to floats for ML compatibility.

Logging and Storage : The processed dataset is stored as feature_store.csv, ensuring structured feature storage for ML workflows.

Feature Retrieval : The stored feature set is loaded from CSV for verification.

This approach ensures structured feature storage, making it easy to track and use for ML model training and inference.

2, Feature Retrieval

Method Explanation

This method loads the previously stored feature dataset from the CSV file and displays the first few rows to verify data integrity. The steps include:

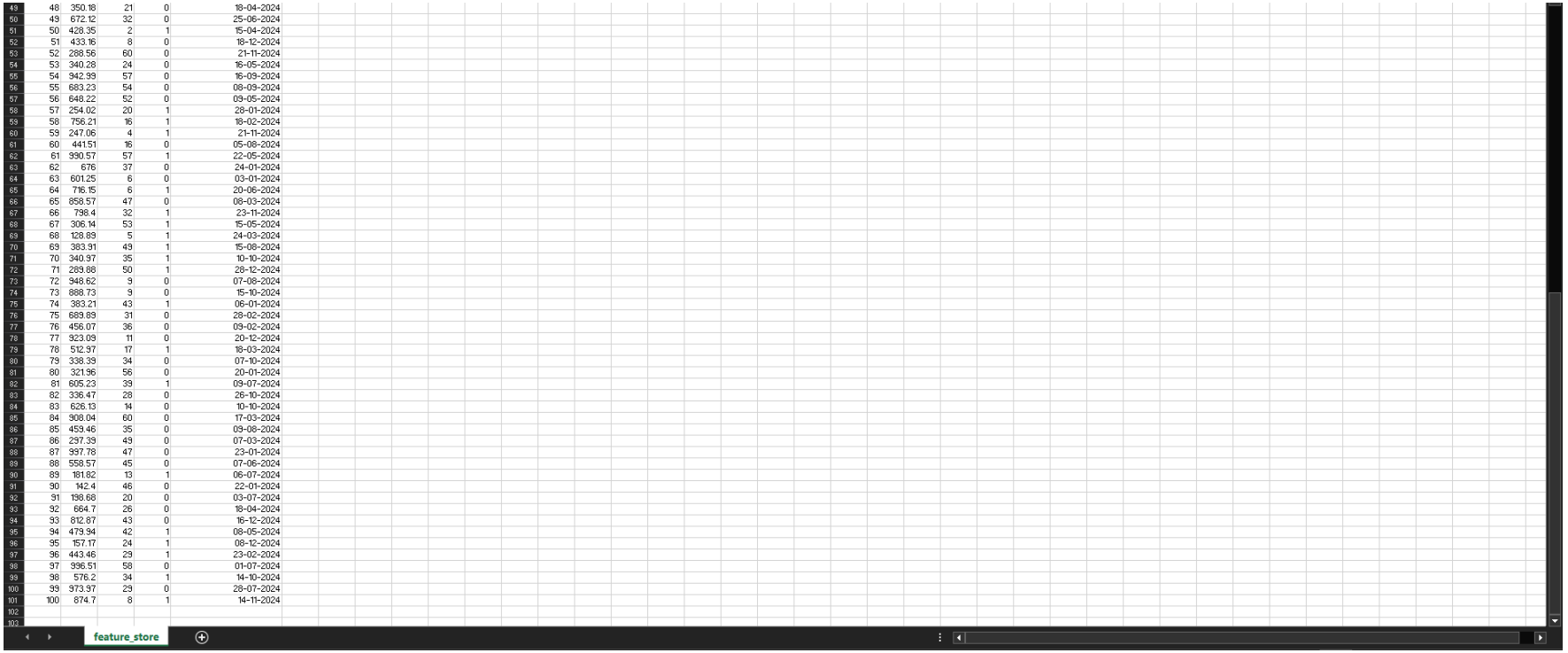
Reading Data from CSV: The dataset stored as feature_store.csv is read using pandas.read_csv().

Verification: The print(retrieved_features.head()) function ensures that the data is loaded correctly and is ready for further processing in machine learning workflows.

This ensures that stored features can be retrieved efficiently and used in various ML tasks such as training, inference, and model evaluation.

```
>=1.4.3->feast) (1.16.0)
Requirement already satisfied: anyio<5,>=3.6.2 in c:\users\laksh\anaconda3\lib\site-packages (from starlette<0.47.0,>=0.40.0-
>fastapi>=0.68.0->feast) (4.8.0)
Requirement already satisfied: sniffio>=1.1 in c:\users\laksh\anaconda3\lib\site-packages (from anyio<5,>=3.6.2->starlette<0.
47.0,>=0.40.0->fastapi>=0.68.0->feast) (1.3.0)

Feature store updated successfully with 100 samples!
   customer_id  total_spend  tenure  churn last_purchase
0            1   675.484119     1.0    0.0    2024-07-22
1            2   122.509680    44.0    0.0    2024-03-09
2            3   347.526387    47.0    0.0    2024-12-10
3            4   300.889664     8.0    1.0    2024-11-27
4            5   762.824093    44.0    0.0    2024-06-03
Retrieved Feature Data:
   customer_id  total_spend  tenure  churn last_purchase
0            1   675.484119     1.0    0.0    2024-07-22
1            2   122.509680    44.0    0.0    2024-03-09
2            3   347.526387    47.0    0.0    2024-12-10
3            4   300.889664     8.0    1.0    2024-11-27
4            5   762.824093    44.0    0.0    2024-06-03
```

customer	total_spe	tenure	churn	last_purchase
1	675.48	1	0	22-07-2024
2	122.51	44	0	09-03-2024
3	347.53	47	0	10-12-2024
4	300.89	8	1	27-11-2024
5	762.82	44	0	03-06-2024
6	709.03	57	0	23-08-2024
7	902.96	35	0	11-06-2024
8	178.24	49	0	08-02-2024
9	479.73	18	0	06-01-2024
10	126.82	50	0	23-08-2024
11	296.77	42	0	15-11-2024
12	554.82	22	0	16-10-2024
13	123.88	8	0	22-02-2024
14	278.95	19	1	08-02-2024
15	684.9	28	0	03-10-2024
16	590.45	11	0	20-04-2024
17	298.4	30	1	17-09-2024
18	630.34	1	1	16-05-2024
19	828.49	47	0	09-03-2024
20	105.85	57	0	28-06-2024
21	825.24	47	1	06-02-2024
22	728.33	17	0	06-05-2024
23	406.23	33	1	09-07-2024
24	239.93	49	1	26-05-2024
25	961.49	12	0	22-03-2024
26	402.94	33	0	13-08-2024
27	183.47	59	0	06-10-2024
28	187.04	7	1	27-12-2024
29	862.74	56	1	04-06-2024
30	643.35	41	1	10-11-2024
31	826.42	20	1	01-12-2024
32	756.76	54	1	28-09-2024
33	582.61	41	0	06-01-2024
34	975.8	33	0	08-12-2024
35	440.68	39	0	11-10-2024
36	596.84	13	1	03-06-2024
37	846.46	10	1	06-12-2024
38	656.67	24	0	24-02-2024
39	875.54	49	0	10-03-2024
40	619.62	11	0	16-05-2024
41	734.11	35	0	01-03-2024
42	141.24	50	1	25-02-2024
43	305.11	60	0	11-10-2024
44	360.45	34	1	21-03-2024
45	171.81	59	0	20-05-2024
46	309.51	1	1	25-05-2024
47	190.9	39	1	06-11-2024
48	350.18	21	0	18-04-2024
49	672.12	32	0	25-06-2024
50	428.35	2	1	15-04-2024
51	433.16	8	0	18-12-2024
52	288.56	60	0	21-11-2024
53	340.28	24	0	16-05-2024



49	48	350.18	21	0	18-04-2024
50	49	672.12	32	0	25-06-2024
51	50	428.35	2	1	15-04-2024
52	51	433.16	8	0	18-12-2024
53	52	288.56	60	0	21-11-2024
54	53	340.28	24	0	16-05-2024
55	54	942.99	57	0	16-09-2024
56	55	683.23	54	0	08-09-2024
57	56	648.22	52	0	09-05-2024
58	57	254.02	20	1	28-01-2024
59	58	756.21	16	1	18-02-2024
60	59	247.06	4	1	21-11-2024
61	60	441.51	16	0	05-08-2024
62	61	990.57	57	1	22-05-2024
63	62	676	37	0	24-01-2024
64	63	601.25	6	0	03-01-2024
65	64	716.15	6	1	20-06-2024
66	65	858.57	47	0	08-03-2024
67	66	798.4	32	1	23-11-2024
68	67	306.14	53	1	15-05-2024
69	68	128.89	5	1	24-03-2024
70	69	383.91	49	1	15-08-2024
71	70	340.97	35	1	10-10-2024
72	71	289.88	50	1	28-12-2024
73	72	948.62	9	0	07-08-2024
74	73	888.73	9	0	15-10-2024
75	74	383.21	43	1	06-01-2024
76	75	689.89	31	0	28-02-2024
77	76	456.07	36	0	09-02-2024
78	77	923.09	11	0	20-12-2024
79	78	512.97	17	1	18-03-2024
80	79	338.39	34	0	07-10-2024
81	80	321.96	56	0	20-01-2024
82	81	605.23	39	1	09-07-2024
83	82	336.47	28	0	26-10-2024
84	83	626.13	14	0	10-10-2024
85	84	908.04	60	0	17-03-2024
86	85	459.46	35	0	09-08-2024
87	86	237.39	49	0	07-03-2024
88	87	997.78	47	0	23-01-2024
89	88	558.57	45	0	07-06-2024
90	89	181.82	13	1	06-07-2024
91	90	142.4	46	0	22-01-2024
92	91	198.68	20	0	03-07-2024
93	92	664.7	26	0	18-04-2024
94	93	812.87	43	0	16-12-2024
95	94	479.94	42	1	08-05-2024
96	95	157.17	24	1	08-12-2024
97	96	443.46	29	1	23-02-2024
98	97	996.51	58	0	01-07-2024
99	98	576.2	34	1	14-10-2024
100	99	973.37	29	0	28-07-2024
101	100	674.7	8	1	14-11-2024
102					
103					

feature_store

Data Versioning Implementation

Overview

We implemented Data Versioning using Git with DVC (Data Version Control) to track changes in datasets and ensure reproducibility.

Implementation Steps

Initialize DVC Repository

git init

dvc init

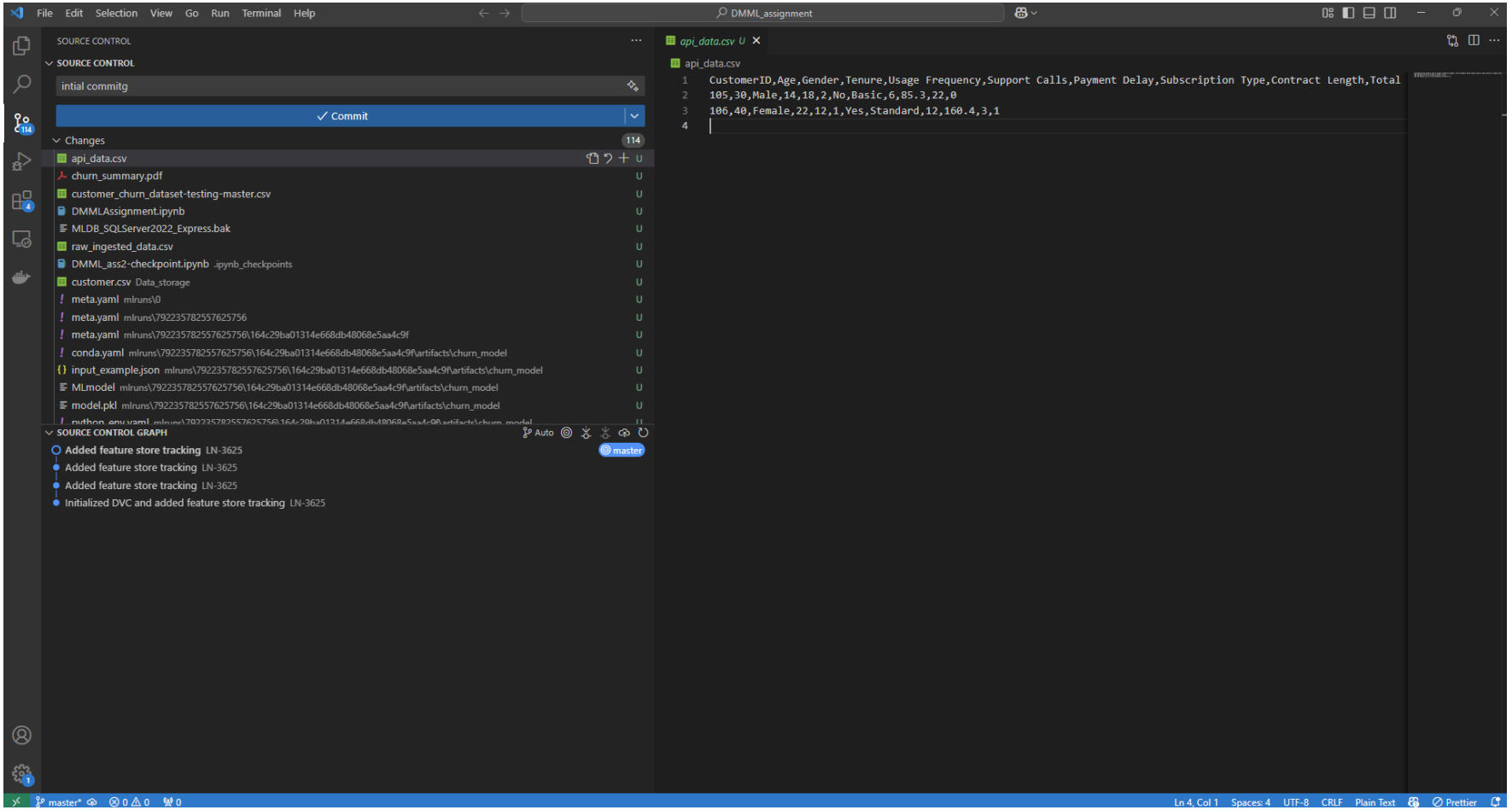
```
git add .dvc .gitignore

git commit -m "Initialize DVC tracking"
```

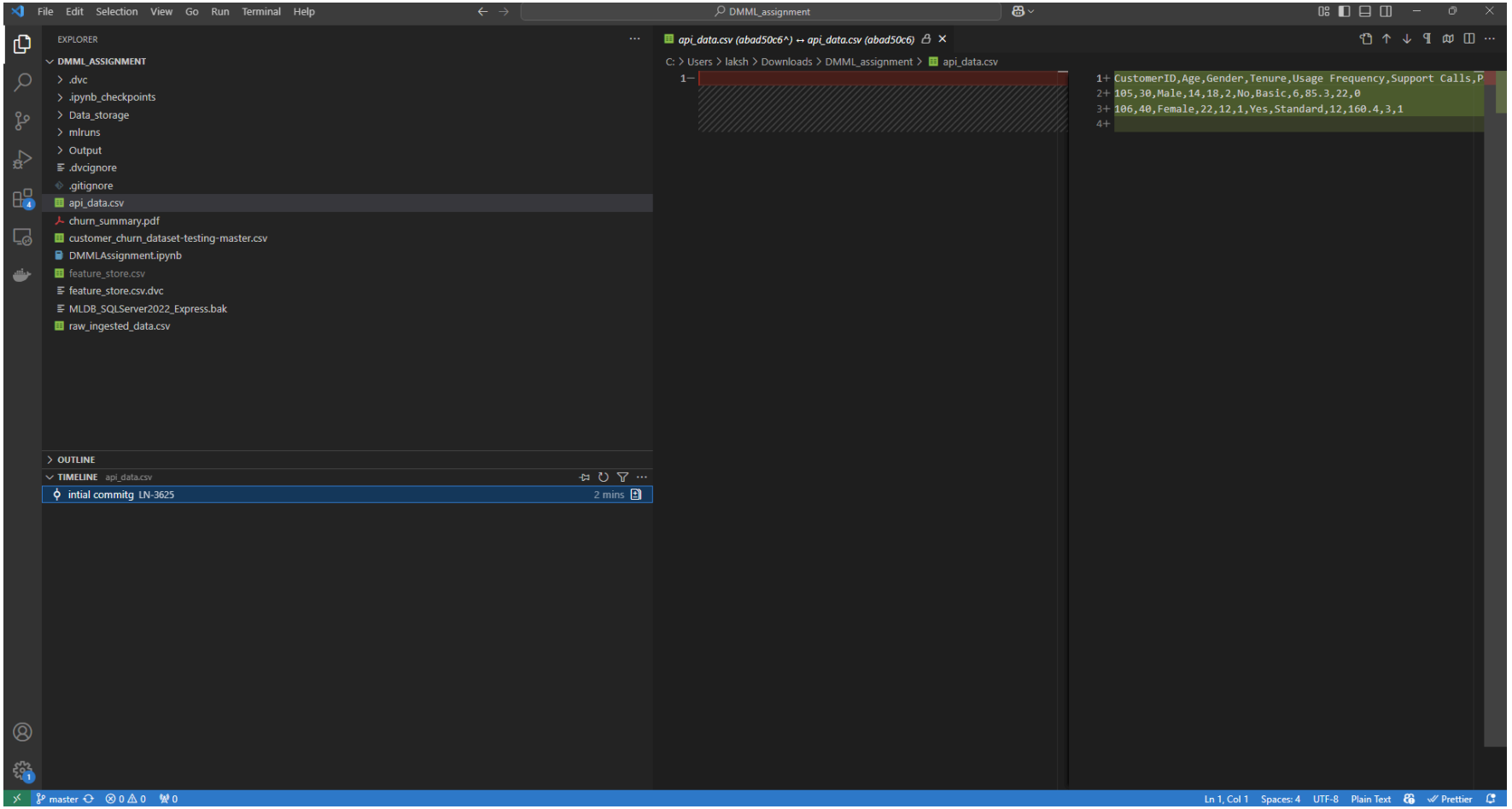
Track Datasets

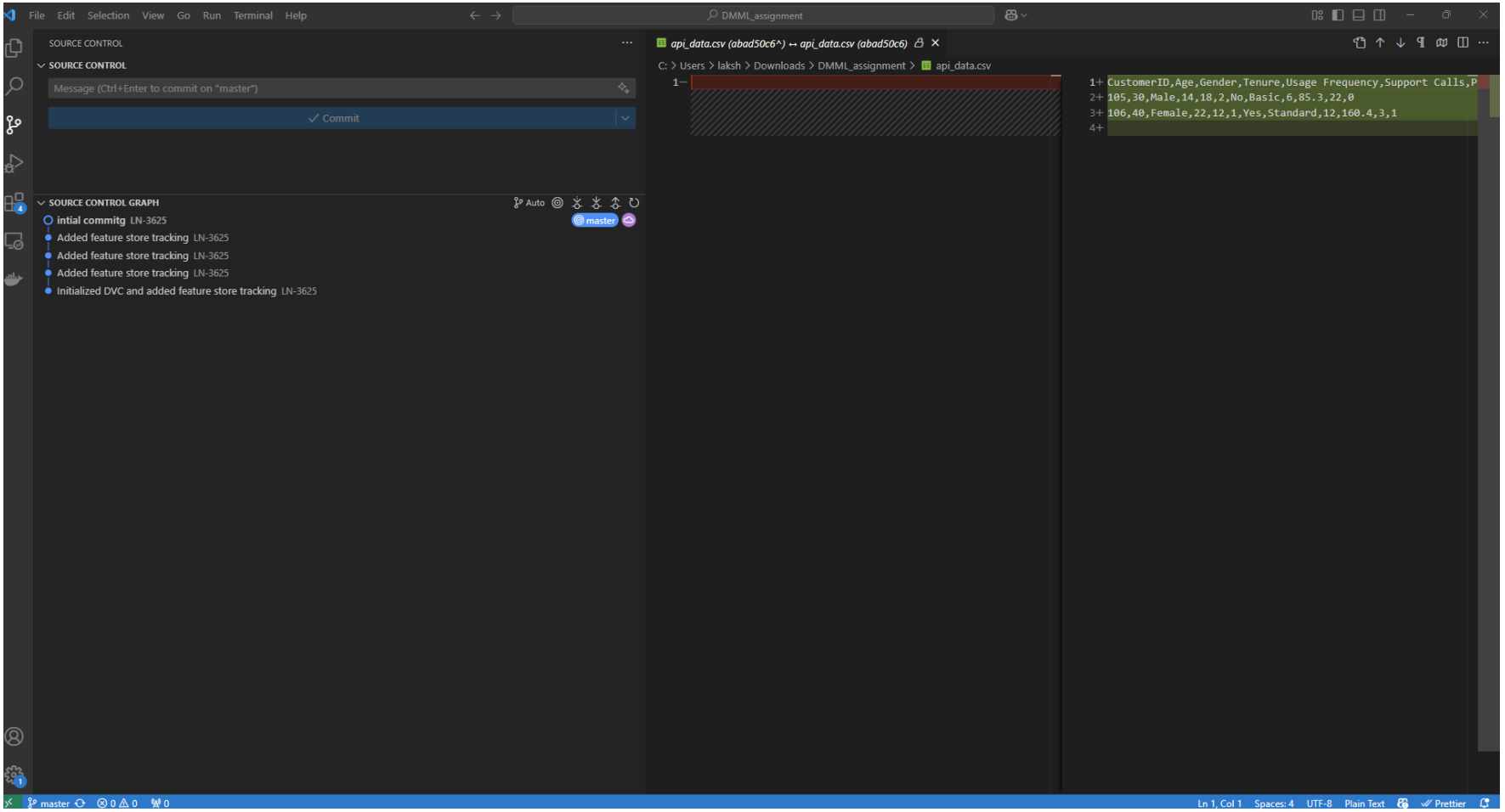
```
dvc add feature_store.csv

git add feature_store.csv.dvc

git commit -m "Track feature store dataset"
```

Track Dataset Versions

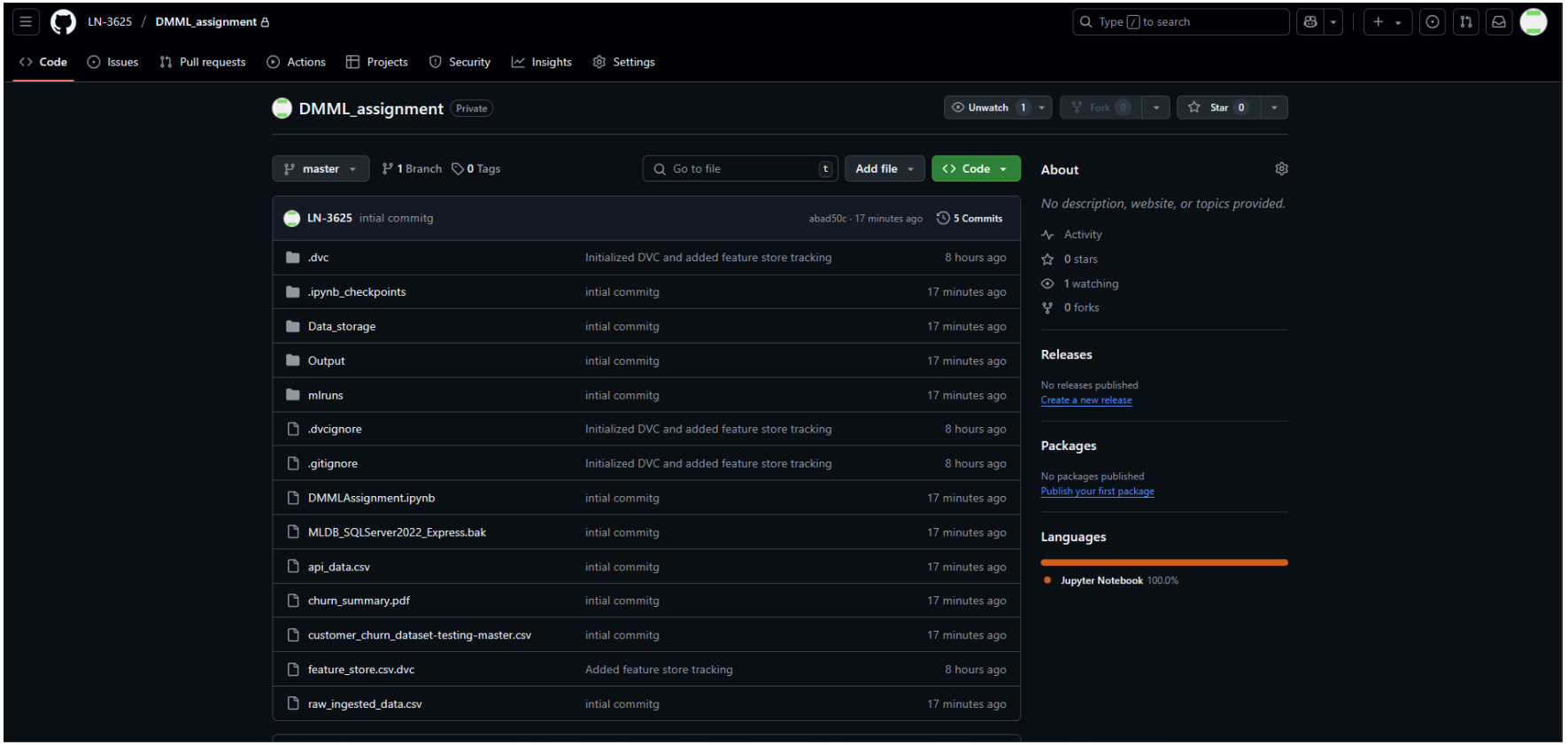
```
dvc metrics show

git tag -a v1.0 -m "First version of feature store"

git push origin v1.0
```


File Edit Selection View Go Run Terminal Help
DMML_assignment
SOURCE CONTROL
SOURCE CONTROL
intial commitg
Commit
Changes 114
api_data.csv U
churn_summary.pdf U
customer_churn_dataset-testing-master.csv U
DMMLAssignment.ipynb U
MLDB_SQLServer2022_Express.bak U
raw_ingested_data.csv U
DMML_ass2-checkpoint.ipynb .ipynb_checkpoints U
customer.csv Data_storage U
meta.yaml mlruns\0 U
meta.yaml mlruns\792235782557625756 U
meta.yaml mlruns\792235782557625756\164c29ba01314e668db48068e5aa4c9f U
conda.yaml mlruns\792235782557625756\164c29ba01314e668db48068e5aa4c9f\artifacts\churn_model U
input_example.json mlruns\792235782557625756\164c29ba01314e668db48068e5aa4c9f\artifacts\churn_model U
MLmodel mlruns\792235782557625756\164c29ba01314e668db48068e5aa4c9f\artifacts\churn_model U
model.pkl mlruns\792235782557625756\164c29ba01314e668db48068e5aa4c9f\artifacts\churn_model U
SOURCE CONTROL GRAPH
Auto
Added feature store tracking LN-3625
master
Added feature store tracking LN-3625
Added feature store tracking LN-3625
Initialized DVC and added feature store tracking LN-3625
api_data.csv U
api_data.csv
1 CustomerID,Age,Gender,Tenure,Usage Frequency,Support Calls,Payment Delay,Subscription Type,Contract Length,Total
2 105,30,Male,14,18,2,No,Basic,6,85.3,22,0
3 106,40,Female,22,12,1,Yes,Standard,12,160.4,3,1
4
master*
Ln 4, Col 1 Spaces: 4 UTF-8 CRLF Plain Text
Prettier


File Edit Selection View Go Run Terminal Help
DMML_assignment
EXPLORER
DMML_ASSIGNMENT
.dvc
.ipynb_checkpoints
Data_storage
mlruns
Output
.dvcignore
.gitignore
api_data.csv
churn_summary.pdf
customer_churn_dataset-testing-master.csv
DMMLAssignment.ipynb
feature_store.csv
feature_store.csv.dvc
MLDB_SQLServer2022_Express.bak
raw_ingested_data.csv
OUTLINE
TIMELINE api_data.csv
intial commitg LN-3625 2 mins
api_data.csv (abad50c6^) ↔ api_data.csv (abad50c6)
C: > Users > laksh > Downloads > DMML_assignment > api_data.csv
1- 1+ CustomerID,Age,Gender,Tenure,Usage Frequency,Support Calls,P
2+ 105,30,Male,14,18,2,No,Basic,6,85.3,22,0
3+ 106,40,Female,22,12,1,Yes,Standard,12,160.4,3,1
4+
master 0 0 0
Ln 1, Col 1 Spaces: 4 UTF-8 Plain Text Prettier



Git RePo: https://github.com/LN-3625/DMML_assignment



Pipeline Design Explanation

Overview

The data pipeline is designed to ensure efficient data ingestion, transformation, storage, versioning, and accessibility for machine learning models. The pipeline consists of:

Pipeline Stages

Data Ingestion: Collecting raw data and storing it in a structured CSV format.

Feature Engineering: Applying transformations and feature extraction techniques to create meaningful features.

Feature Store Management: Storing processed features in a central CSV file for easy retrieval.

Data Versioning: Using DVC and Git to track dataset changes over time.

Model Training Readiness: Ensuring datasets are versioned and reproducible for ML model training.

Benefits of this Pipeline

Ensures data consistency with structured storage.

Enables efficient tracking of dataset changes.

Provides reproducibility for ML experiments.

Supports scalability for larger datasets in the future.

Conclusion

The Feature Store helps streamline ML feature management.

DVC ensures dataset reproducibility and tracking.

Git integration allows proper dataset versioning and history tracking.

These components enhance the end-to-end machine learning pipeline.