

Airflow DAGs page:

The screenshot shows the Airflow web interface with the following details:

- Header:** The top navigation bar includes tabs for "JL Rent GPU servers", "JL JarvisLabs.ai", "MLflow", "DAGs - Airflow", "constants.py (16)", "upGrad | Learning", "Google Docs: On", "Shivani_Agrawal", and a "+" button.
- Breadcrumbs:** The current path is "Not Secure | notebooksc.jarvislabs.ai:10980/home".
- Page Title:** "Airflow" (with a blue icon).
- Page Submenu:** "DAGs", "Security", "Browse", "Admin", and "Docs".
- Time:** "15:44 UTC" and a blue "UU" badge.
- Content Area:**
 - A yellow box at the top states: "Do not use SQLite as metadata DB in production – it should only be used for dev/testing. We recommend using Postgres or MySQL. [Click here](#) for more information."
 - A yellow box below it states: "Do not use SequentialExecutor in production. [Click here](#) for more information."
- DAGs Section:** A large table titled "DAGs" with the following columns:
 - Buttons:** "All" (36), "Active" (3), "Paused" (33).
 - Search:** "Filter DAGs by tag" and "Search DAGs" fields.
 - Header:** "DAG", "Owner", "Runs", "Schedule", "Last Run", "Next Run", "Recent Tasks".
 - Data:** Three rows for "Lead_Scoring_Data_Engineering_Pipeline", "Lead_scoring_inference_pipeline", and "Lead_scoring_training_pipeline". Each row shows the DAG name, owner ("airflow"), run counts (1 active, 2 pending, 0 failed), schedule type (daily/hourly/monthly), last run date, next run date, and recent task counts (7, 4, 2).

Lead Scoring Data Engineering Pipeline DAG:

The screenshot shows the Airflow web interface for the 'Lead Scoring Data Engineering Pipeline'. The top navigation bar includes links for Rent GPU servers, JarvisLabs.ai, MLflow, Lead_Scoring_D, constants.py (16), upGrad | Learning, Google Docs: On, and Shivani_Agrawal. The main header displays the DAG name 'Lead_Scoring_Data_Engineering_Pipeline' and its purpose 'DAG to run data pipeline for lead scoring'. It shows the schedule '@daily' and the next run at '2022-09-18, 00:00:00'. Below the header are various navigation tabs: Grid (selected), Graph, Calendar, Task Duration, Task Tries, Landing Times, Gantt, Details, Code, and Audit Log. A toolbar below the tabs includes filters for date (18/09/2022, 03:49:42 PM), run count (25), run types (All Run Types), run states (All Run States), and a 'Clear Filters' button. A status bar at the bottom shows deferred, failed, queued, running, scheduled, skipped, success, up_for_reschedule, up_for_retry, upstream_failed, and no_status metrics. On the left, a sidebar lists tasks: building_db, checking_raw_data_schema, loading_data, mapping_city_tier, mapping_categorical_vars, mapping_interactions, and checking_model_inputs_schema. The main area displays a timeline chart for the DAG runs, showing task execution times and failure points. The 'DAG Details' section provides summary statistics for the three runs displayed.

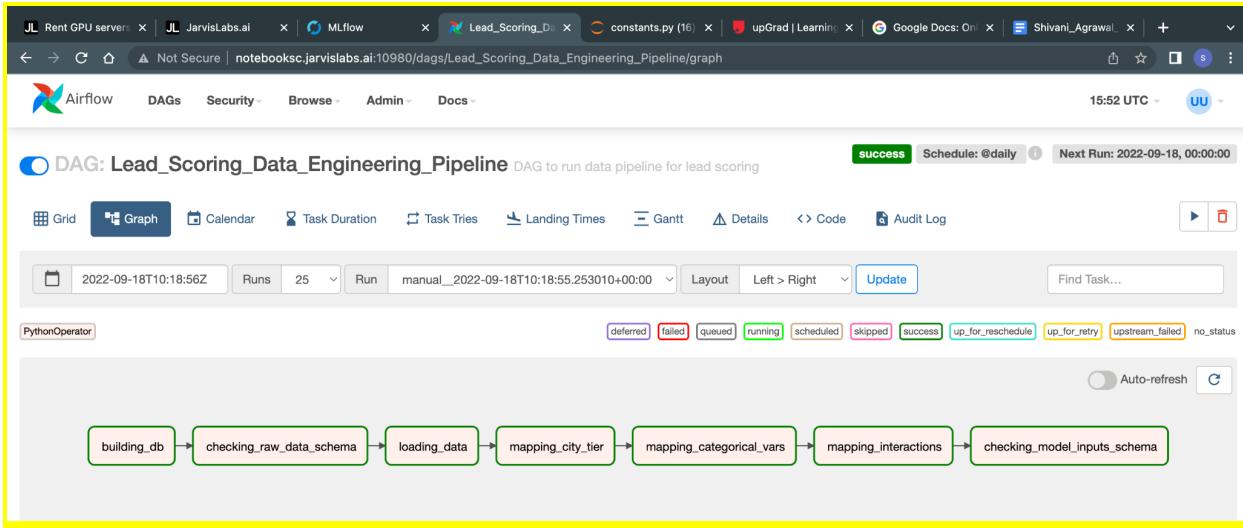
Run Type	Count
Total success	1
Total failed	2

DAG Runs Summary

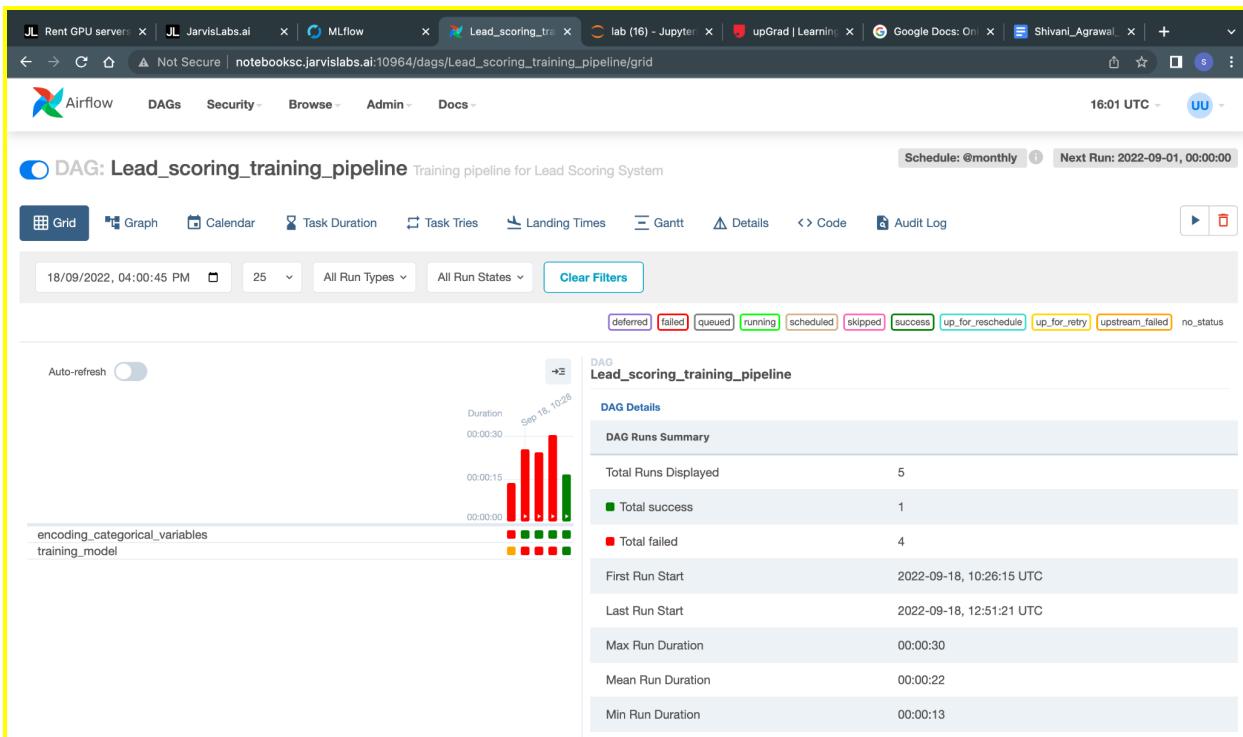
Run Type	Count
Total success	1
Total failed	2

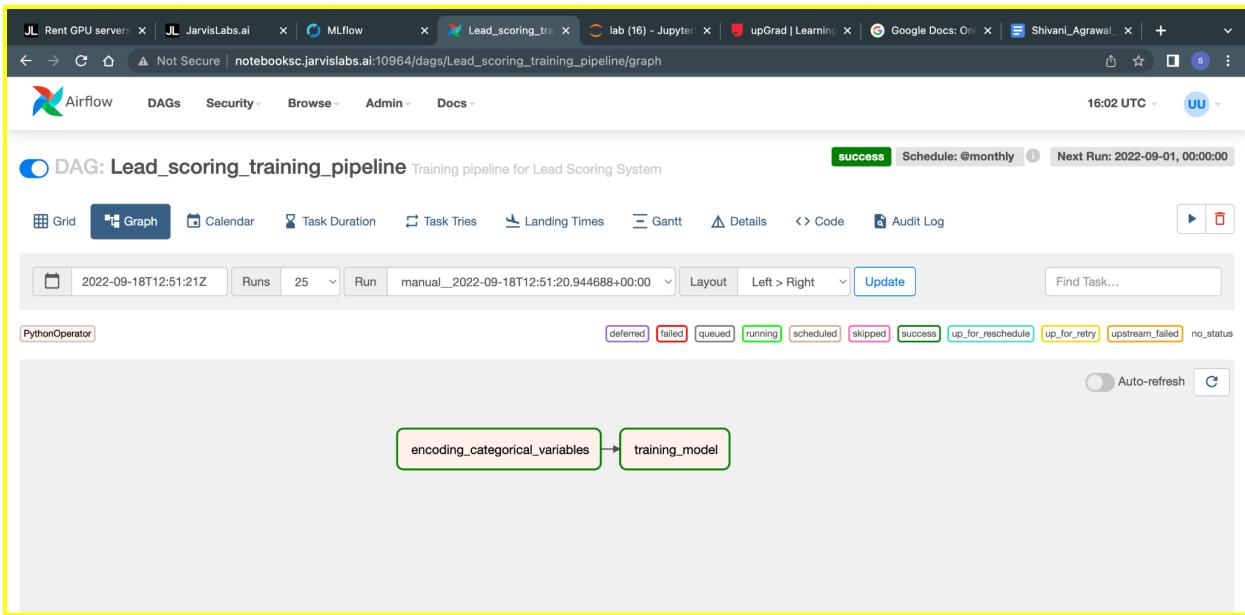
DAG Details

Metric	Value
Total Runs Displayed	3
Total success	1
Total failed	2
First Run Start	2022-09-18, 10:03:00 UTC
Last Run Start	2022-09-18, 10:18:56 UTC
Max Run Duration	00:01:11
Mean Run Duration	00:00:33
Min Run Duration	00:00:14

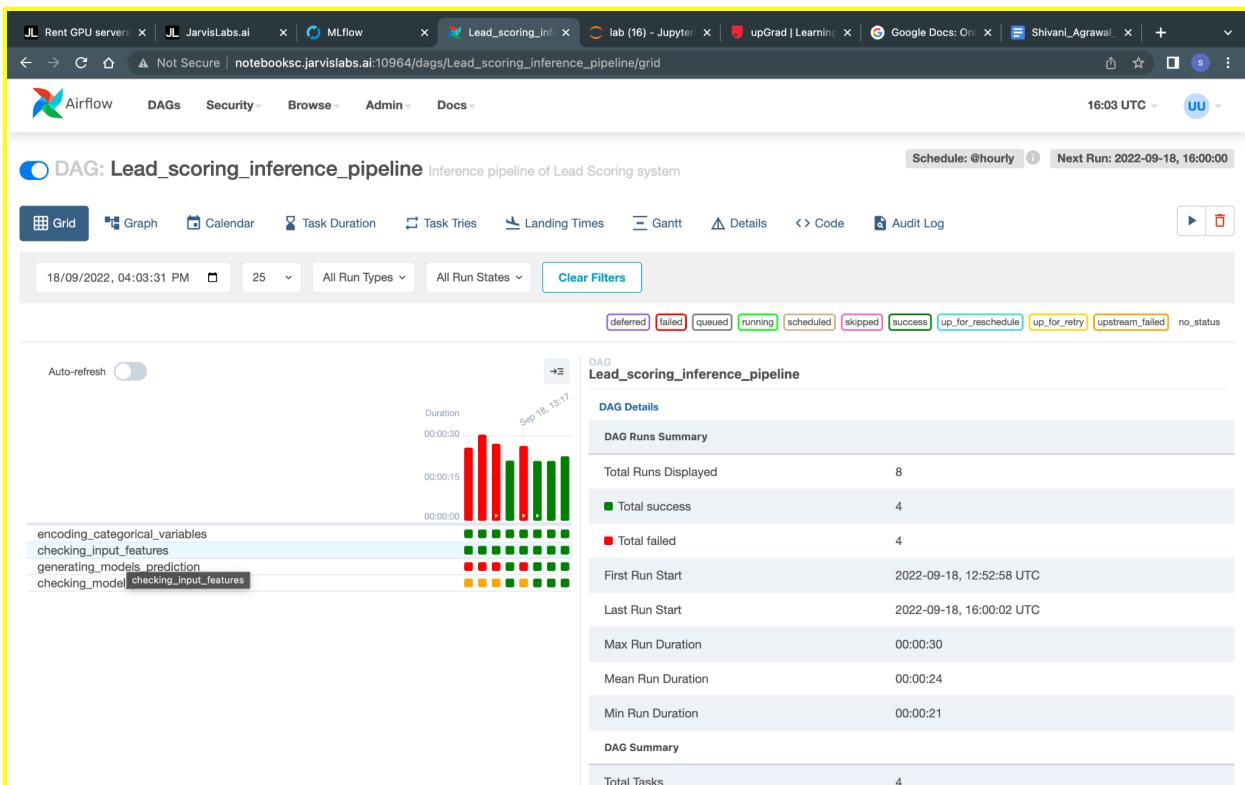


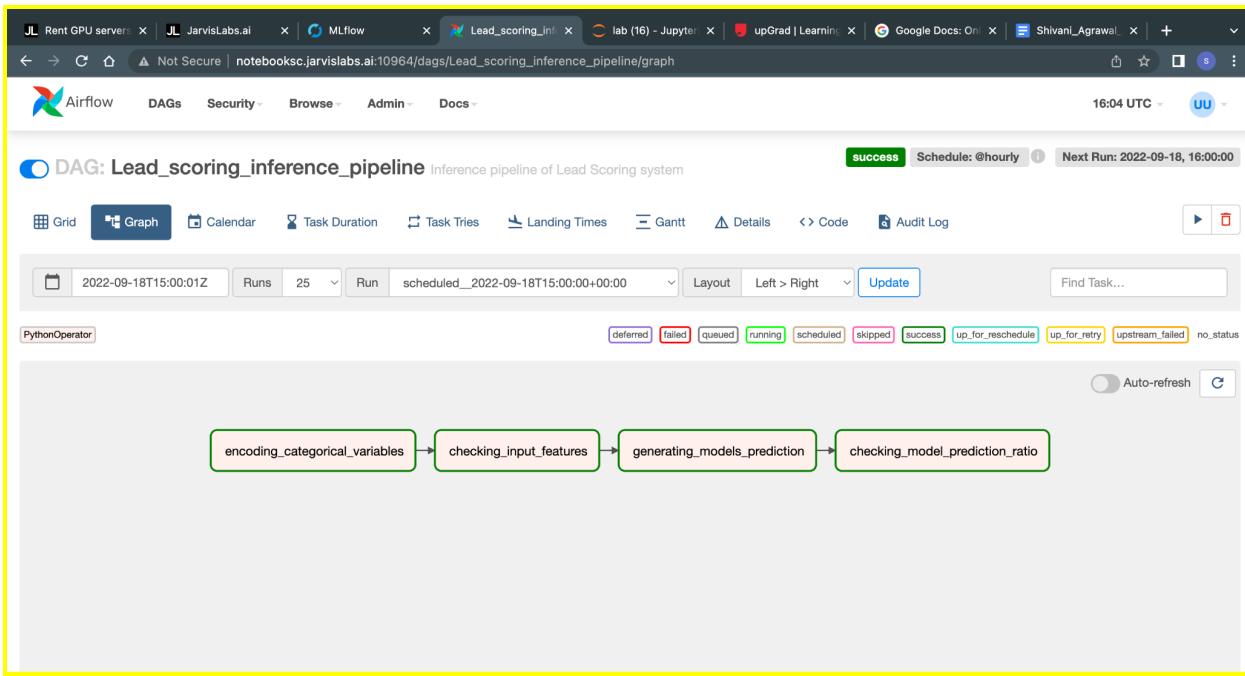
Lead_scoring_training_pipeline:





Lead_scoring_inference_pipeline:





Some Jarvis jupyter code snaps:

```

1 # You can create more variables according to your project. The following are the basic variables that have been provided to you
2 DB_PATH = '/home/dags/Lead_scoring_data_pipeline/'
3 DB_FILE_NAME = 'lead_scoring_data_cleaning.db'
4 DATA_DIRECTORY = '/home/dags/Lead_scoring_data_pipeline/data/'
5 INTERACTION_MAPPING = '/home/dags/Lead_scoring_data_pipeline/mapping/'
6 INDEX_COLUMNS = ['created_date', 'first_platform_c', 'first_utm_medium_c', 'total_leads_dropped',
7 'city_tier', 'referred_lead', 'app_complete_flag']
8 LEAD_SCORING_FILE='leadscore'

```

JL Rent GPU servers | JL JarvisLabs.ai | MLflow | Lead_scoring_inf... | utils.py (16) - Jupyter Notebook | upGrad | Learning | Google Docs: Online | Shivani_Agrawal | +

File Edit View Run Kernel Git Tabs Settings Help

Filter files by name

/ dags / Lead_scoring_data_pipeline /

Name	Last Modified
assignment_folder_files	a day ago
data	a day ago
mapping	an hour ago
constants.py	a day ago
data_validation_checks.py	a day ago
lead_scoring_data_clean...	10 minutes ago
lead_scoring_data_pipeline...	a day ago
schema.py	a day ago
utils.py	2 minutes ago

```
1 """
2 filename: utils.py
3 functions: load_data, __read_input_data, __save_data_to_db, encode_features, get_models_prediction, prediction_ratio_check,
4 creator: shivani.agrawal
5 version: 1
6 """
7
8
9
10 #####
11 ##### Import necessary modules and files
12 # Import necessary modules and files
13 # #####
14
15 import pandas as pd
16 import os
17 import sqlite3
18 from sqlite3 import Error
19
20 def load_data(file_path_list):
21     data = []
22     for eachfile in file_path_list:
23         data.append(pd.read_csv(eachfile, index_col=0))
24
25     return data
26
27 #####
28 # Define the function to build database
29 # #####
30
31 def build_dbs(db_path, db_file_name):
32     """
33     This function checks if the db file with specified name is present
34     in the /Assignment/01_data_pipeline/scripts folder. If it is not present it creates
35     the db file with the given name at the given path.
36
37
38 INPUTS
39     db_file_name : Name of the database file 'utils_output.db'
40     db_path : path where the db file should be
41
```

JL Rent GPU servers | JL JarvisLabs.ai | MLflow | Lead_scoring_inf... | lead_scoring... (1) | upGrad | Learning | Google Docs: Online | Shivani_Agrawal | +

File Edit View Run Kernel Git Tabs Settings Help

Filter files by name

/ dags / Lead_scoring_data_pipeline /

Name	Last Modified
assignment_folder_files	a day ago
data	a day ago
mapping	an hour ago
constants.py	a day ago
data_validation_checks.py	a day ago
lead_scoring_data_clean...	12 minutes ago
lead_scoring_data_pipeline...	a day ago
schema.py	a day ago
utils.py	3 minutes ago

```
1 #####
2 # Import necessary modules
3 # #####
4
5
6 from airflow import DAG
7 from airflow.operators.python import PythonOperator
8 import sys
9 from datetime import datetime, timedelta
10 import importlib.util
11
12 def module_from_file(module_name, file_path):
13     spec = importlib.util.spec_from_file_location(module_name, file_path)
14     module = importlib.util.module_from_spec(spec)
15     spec.loader.exec_module(module)
16     return module
17
18 #Importing variables for ex: from.. import * does not work, loading modules as taught in live session by instructor
19 utils = module_from_file("utils", "/home/dags/Lead_scoring_data_pipeline/utils.py")
20 data_validation_checks = module_from_file("data_validation_checks",
21                                         "/home/dags/Lead_scoring_data_pipeline/data_validation_checks.py")
22 constants = module_from_file("utils", "/home/dags/Lead_scoring_data_pipeline/constants.py")
23 schema = module_from_file("data_validation_checks", "/home/dags/Lead_scoring_data_pipeline/schema.py")
24 city_tier_mapping = module_from_file("utils", "/home/dags/Lead_scoring_data_pipeline/mapping/city_tier_mapping.py")
25 significant_categorical_level = module_from_file("data_validation_checks",
26                                                 "/home/dags/Lead_scoring_data_pipeline/mapping/significant_categorical_level.py")
27
28 #Load all variables to pass it to utils function
29 db_path=constants.DB_PATH
30 db_file_name=constants.DB_FILE_NAME
31 data_directory=constants.DATA_DIRECTORY
32 interaction_mapping=constants.INTERACTION_MAPPING
33 index_columns=constants.INDEX_COLUMNS
34 lead_scoring_file=constants.LEAD_SCORING_FILE
35
36 raw_data_schema=schema.raw_data_schema
37 model_input_schema=schema.model_input_schema
38 city_tier_mapping_dict=city_tier_mapping.city_tier_mapping_dict
39 list_platform=significant_categorical_level.list_platform
40 list_medium=significant_categorical_level.list_medium
41 list_source=significant_categorical_level.list_source
42
```

Simple 4 0 Python

Ln 1, Col 1 Spaces: 4 lead_scoring_data_pipeline.py

JL Rent GPU servers | JL JarvisLabs.ai | MLflow | Lead_scoring_inf | constants.py (16) | upGrad | Learning | Google Docs: On | Shivani_Agrawal | +

File Edit View Run Kernel Git Tabs Settings Help

Filter files by name

/ dags / Lead_scoring_training_pipeline /

Name	Last Modified
assignment_folder_files	a day ago
notebooks_folder_files	a day ago
constants.py	3 hours ago
lead_scoring_training_pi...	2 days ago
utils.py	a minute ago

```
1 #DB_PATH = '/home/dags/Lead_scoring_data_pipeline/'
2 DB_PATH = '/home/dags/Lead_scoring_data_pipeline/'
3 DB_FILE_NAME = 'lead_scoring_data_cleaning.db'
4
5 DB_FILE_MLFLOW = 'lead_scoring_mlflow_production.db'
6
7 TRACKING_URI = "http://0.0.0.0:6007"
8 EXPERIMENT = "Lead_scoring_mlflow_production"
9
10 # model config imported from pycaret experimentation
11 MODEL_CONFIG = {
12     'boosting_type': 'gbdt',
13     'class_weight': 'None',
14     'colsample_bytree': 1.0,
15     'importance_type': 'split',
16     'learning_rate': 0.1,
17     'max_depth': -1,
18     'min_child_samples': 20,
19     'min_child_weight': 0.001,
20     'min_split_gain': 0.0,
21     'n_estimators': 100,
22     'n_jobs': -1,
23     'num_leaves': 31,
24     'objective': 'None',
25     'random_state': 42,
26     'reg_alpha': 0.0,
27     'reg_lambda': 0.0,
28     'silent': 'warn',
29     'subsample': 1.0,
30     'subsample_for_bin': 200000 ,
31     'subsample_freq': 0
32 }
33
34 # list of the features that needs to be there in the final encoded dataframe
35 ONE_HOT_ENCODED_FEATURES = ['total_leads_dropped', 'city_tier', 'first_platform_c_Level8', 'first_platform_c_Level2',
36 'first_platform_c_others', 'first_platform_c_Level0',
37 'first_utm_source_c_Level6', 'app_complete_flag']
38 # list of features that need to be one-hot encoded
39 FEATURES_TO_ENCODE = ['first_platform_c', 'first_utm_medium_c', 'first_utm_source_c']
```

Simple 4 0 Python

Ln 1, Col 1 Spaces: 4 constants.py

JL Rent GPU servers | JL JarvisLabs.ai | MLflow | Lead_scoring_inf | utils.py (16) | upGrad | Learning | Google Docs: On | Shivani_Agrawal | +

File Edit View Run Kernel Git Tabs Settings Help

Filter files by name

/ dags / Lead_scoring_training_pipeline /

Name	Last Modified
assignment_folder_files	a day ago
notebooks_folder_files	a day ago
constants.py	3 hours ago
lead_scoring_training_pi...	2 days ago
utils.py	5 minutes ago

```
1 """
2 filename: utils.py
3 functions: __read_input_data,__save_data_to_db, encode_features, get_trained_model
4 creator: shivani.agrawal
5 version: 1
6 """
7 #####
8 # Import necessary modules
9 # #####
10 # #####
11 import pandas as pd
12 import numpy as np
13 from sklearn import preprocessing
14 import sqlite3
15 from sqlite3 import Error
16
17 import mlflow
18 import mlflow.sklearn
19
20 from sklearn.model_selection import train_test_split
21 from sklearn.metrics import roc_auc_score
22 import lightgbm as lgb
23 from sklearn.model_selection import train_test_split
24 from sklearn.metrics import accuracy_score
25 from sklearn.model_selection import train_test_split
26 from sklearn.metrics import classification_report
27 from sklearn.metrics import confusion_matrix
28
29 import mlflow.sklearn
30 from sklearn.metrics import accuracy_score
31 from sklearn.metrics import precision_score, recall_score
32 from sklearn.metrics import precision_recall_fscore_support
33
34 #####
35 # Define the function to encode features
36 # #####
37 # When imported in airflow from.. import * does not work
38 # #####
39
40 #when imported in airflow from.. import * does not work
41 def __read_input_data(db_path, db_file_name):
42     cnx = sqlite3.connect(db_path + db_file_name)
43
44     return cnx
```

Simple 4 0 Python

Ln 6, Col 1 Spaces: 4 utils.py

JL Rent GPU servers | JL JarvisLabs.ai | MLflow | Lead_scoring_inference_pipeline | lead_scoring_training_pipeline.py | upGrad | Learning | Google Docs: On | Shivani_Agrawal | + |

File Edit View Run Kernel Git Tabs Settings Help

Filter files by name

Name Last Modified

- assignment_folder_files a day ago
- notebooks_folder_files a day ago
- constants.py 3 hours ago
- lead_scoring_training_pipeline.py** 2 days ago
- utils.py 5 minutes ago

```
1 #####  
2 # Import necessary modules  
3 #####  
4 from airflow import DAG  
5 from airflow.operators.python import PythonOperator  
6 from airflow.operators.bash import BashOperator  
7  
8 from datetime import datetime, timedelta  
9 import sys  
10 import importlib.util  
11  
12 def module_from_file(module_name, file_path):  
13     spec = importlib.util.spec_from_file_location(module_name, file_path)  
14     module = importlib.util.module_from_spec(spec)  
15     spec.loader.exec_module(module)  
16     return module  
17  
18 #Importing variables for ex: from.. import * does not work, loading modules as taught in live session  
19 utils = module_from_file("utils", "/home/dags/Lead_scoring_training_pipeline/utils.py")  
20 constants = module_from_file("utils", "/home/dags/Lead_scoring_training_pipeline/constants.py")  
21  
22 db_path=constants.DB_PATH  
23 db_file_name=constants.DB_FILE_NAME  
24 one_hot_encoded_features=constants.ONE_HOT_ENCODED_FEATURES  
25 features_to_encode=constants.FEATURES_TO_ENCODE  
26 model_config=constants.MODEL_CONFIG  
27 experiment=constants.EXPERIMENT  
28 tracking_uri=constants.TRACKING_URI  
29  
30 #####  
31 # Define default arguments and DAG  
32 #####  
33 default_args = {  
34     'owner': 'airflow',  
35     'start_date': datetime(2022,7,30),  
36     'retries' : 1,  
37     'retry_delay': timedelta(seconds=5)  
38 }  
39  
40  
41 ML_training_dag = DAG(  
42     dag_id = 'Lead_scoring_training_pipeline',  
43     default_args=default_args,  
44     schedule_interval='@monthly'  
45 )  
46  
47 with ML_training_dag:  
48     training_task = BashOperator(  
49         task_id = 'train_ml_model',  
50         bash_command = "python /home/dags/Lead_scoring_training_pipeline/training.py",  
51         dag = ML_training_dag  
52     )  
53  
54     training_task
```

Ln 1, Col 1 Spaces: 4 lead_scoring_training_pipeline.py

JL Rent GPU servers | JL JarvisLabs.ai | MLflow | Lead_scoring_inference_pipeline | constants.py | upGrad | Learning | Google Docs: On | Shivani_Agrawal | + |

File Edit View Run Kernel Git Tabs Settings Help

Filter files by name

Name Last Modified

- assignment_folder_files 4 hours ago
- constants.py** 3 hours ago
- lead_scoring_inference_pipeline.py 2 days ago
- prediction_distribution_2... 2 hours ago
- prediction_distribution_2... 2 hours ago
- prediction_distribution_2... an hour ago
- prediction_distribution_2... 15 minutes ago
- utils.py 5 minutes ago

```
1 DB_PATH = '/home/dags/Lead_scoring_data_pipeline/'  
2 DB_FILE_NAME = 'lead_scoring_data_cleaning.db'  
3  
4 SCRIPTS_OUTPUT='/home/dags/Lead_scoring_inference_pipeline/'  
5 DB_FILE_MLFLOW = '/home/database/Lead_scoring_mlflow_production.db'  
6 TRACKING_URI = "http://0.0.0.0:6007"  
7  
8 # experiment, model name and stage to load the model from mlflow model registry  
9 MODEL_NAME = 'LightGBM'  
10 STAGE = 'production'  
11  
12 # list of the features that needs to be there in the final encoded dataframe  
13 ONE_HOT_ENCODED_FEATURES = ['total_leads_dropped', 'city_tier', 'first_platform_c_Level8', 'first_platform_c_Level2',  
14 'first_platform_c_others', 'first_platform_c_Level0', 'first_platform_c_Level7', 'first_utm_medium_c_others',  
15 'first_utm_medium_c_Level13', 'first_utm_source_c_Level6']  
16 FEATURES_TO_ENCODE = ['first_platform_c', 'first_utm_medium_c', 'first_utm_source_c']
```

Ln 1, Col 1 Spaces: 4 constants.py

The screenshot shows a Jupyter Notebook interface with multiple tabs at the top. The active tab is 'utils.py (16) - Jun'. The left sidebar displays a file tree for 'dags / Lead_scoring_inference_pipeline /'. The main area contains the code for 'utils.py'.

```
1 """
2 filename: utils.py
3 functions: load_data, __read_input_data, __save_data_to_db, encode_features, get_models_prediction, prediction_ratio_check,
4 creator: shivani.agrawal
5 version: 1
6 """
7 #####
8 # Import necessary modules
9 # #####
10 import mlflow
11 import mlflow.sklearn
12 import pandas as pd
13 import sqlite3
14 import os
15 import logging
16 from datetime import datetime
17 from sklearn.model_selection import train_test_split
18 from sklearn.metrics import roc_auc_score
19 import lightgbm as lgb
20 from sklearn.model_selection import train_test_split
21 from sklearn.metrics import accuracy_score
22 from sklearn.model_selection import train_test_split
23 from sklearn.metrics import classification_report
24 from sklearn.metrics import confusion_matrix
25 import collections
26 #####
27 Define the function to train the model
28 # #####
29
30 #when imported in airflow from.. import * does not work
31
```

At the bottom, there are status indicators: Simple, 4, 0, Python, and Ln 6, Col 4, Spaces: 4, utils.py.

MLflow:

mlflow 1.26.1 Experiments Models GitHub Docs

Experiments Default

Search Experiments

Default lead_scoring_model_ex... lead_scoring_model_ex...

Experiment ID: 0

Description Edit

Refresh Compare Delete Download CSV Start Time All time

Columns Only show differences Search Filter Clear

Showing 1 matching run

	Start Time	Duration	Run Name	User	Source	Version	Models	False Negative	Precision	Recall
<input type="checkbox"/>	3 hours ago	4.7s	Lead_scoring...	root	airflow	-	LightGBM/1	4790	0.724	0.812

Load more

mlflow 1.26.1 Experiments Models GitHub Docs

Registered Models

Share and manage machine learning models. Learn more

Create Model

Search by model name

Filter Clear

Name	Latest Version	Staging	Production	Last Modified	Tags
LightGBM	Version 1	-	Version 1	2022-09-18 18:32:30	-

< 1 > 10 / page

The screenshot shows the mlflow UI interface. At the top, there's a navigation bar with tabs for 'Experiments' and 'Models'. The 'Models' tab is selected, showing a list of registered models. One model, 'LightGBM', is highlighted. Below the list, there's a detailed view for 'LightGBM'. It shows the creation time (2022-09-18 18:21:36) and last modified time (2022-09-18 18:32:30). There are sections for 'Description' and 'Tags', both currently empty. A table for 'Versions' lists one active version (Version 1) registered at 2022-09-18 18:21:36 and assigned to the 'Production' stage.

Parameters:

This screenshot shows the detailed parameters for the 'LightGBM' model. A table lists 20 parameters with their corresponding values. The parameters include boosting_type (gbdt), class_weight (None), colsample_bytree (1.0), importance_type (split), learning_rate (0.01), max_depth (-1), min_child_samples (10), min_child_weight (0.005), min_split_gain (0.1), n_estimators (75), n_jobs (-1), num_leaves (20), objective (None), random_state (45), and reg_alpha (0.0).

Name	Value
boosting_type	gbdt
class_weight	None
colsample_bytree	1.0
importance_type	split
learning_rate	0.01
max_depth	-1
min_child_samples	10
min_child_weight	0.005
min_split_gain	0.1
n_estimators	75
n_jobs	-1
num_leaves	20
objective	None
random_state	45
reg_alpha	0.0

reg_lambda	0.0
silent	warn
subsample	1.0
subsample_for_bin	2000
subsample_freq	0

Metrics :

Not Secure | notebooksc.jarvislabs.ai:10965/#/experiments/0/runs/d849edce31ee4ad886d2892d8c4f9a77

- ▶ Description [Edit](#)
- ▶ Parameters (20)
- ▼ Metrics (11)

Name	Value
False Negative ↗	3511
Precision ↗	0.721
Precision_0 ↗	0.846
Precision_1 ↗	0.663
Recall ↗	0.754
Recall_0 ↗	0.539
Recall_1 ↗	0.902
True Negative ↗	19244
f1_0 ↗	0.659
f1_1 ↗	0.764
test_accuracy ↗	0.721

- ▶ Tags
- ▼ Artifacts

Screenshot of a web browser showing the MLflow Model page for a registered model named "LightGBM, v1".

Artifacts

- models
 - MLmodel
 - conda.yaml
 - model.pkl
 - python_env.yaml
 - requirements.txt

Full Path:/home/mlruns/0/bf0c16f6020408b843624eaf020f565/artifacts/models

Registered on 2022/09/18

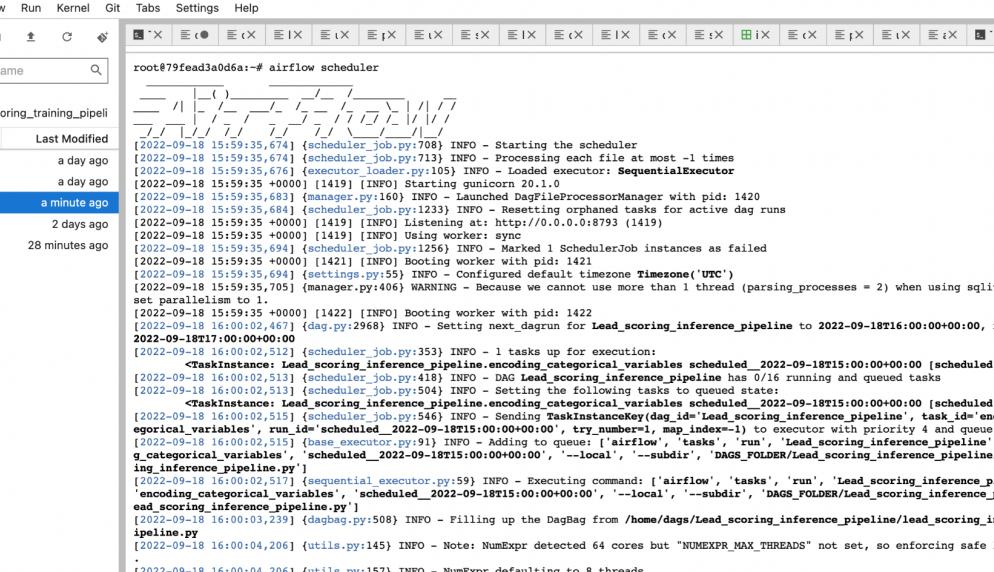
MLflow Model

The code snippets below demonstrate how to make predictions using the logged model. This model is also registered to the [model registry](#).

Model schema	Make Predictions				
Input and output schema for your model. Learn more <table border="1"> <thead> <tr> <th>Name</th> <th>Type</th> </tr> </thead> <tbody> <tr> <td colspan="2">No schema. See MLflow docs for how to include input and output schema with your model.</td> </tr> </tbody> </table>	Name	Type	No schema. See MLflow docs for how to include input and output schema with your model.		Predict on a Spark DataFrame: <pre>import mlflow logged_model = 'runs:/bf0c16f6020408b843624eaf020f565/models' # Load model as a Spark UDF. Override result_type if the model does not # return double values. loaded_model = mlflow.pyfunc.spark_udf(spark, model_uri=logged_model, result_type='double') # Predict on a Spark DataFrame. columns = list(df.columns) df.withColumn('predictions', loaded_model(*columns)).collect()</pre> Predict on a Pandas DataFrame: <pre>import mlflow logged_model = 'runs:/bf0c16f6020408b843624eaf020f565/models' # Load model as a PyFuncModel. loaded_model = mlflow.pyfunc.load_model(logged_model) # Predict on a Pandas DataFrame. import pandas as pd loaded_model.predict(pd.DataFrame(data))</pre>
Name	Type				
No schema. See MLflow docs for how to include input and output schema with your model.					

Screenshot of a terminal window showing the execution of an MLflow server command.

```
root@79fead3a0d6a:~# mlflow server --backend-store-uri='sqlite:///database/Lead_scoring_mlflow_production.db' --default-artifact-root=/home/mlruns/ --port=6007 --host=0.0.0.0
[2022-09-18 15:58:38 +0000] [1295] [INFO] Starting unicorn 20.1.0
[2022-09-18 15:58:38 +0000] [1295] [INFO] Listening at http://0.0.0.0:6007 (1295)
[2022-09-18 15:58:38 +0000] [1295] [INFO] Using worker: sync
[2022-09-18 15:58:38 +0000] [1297] [INFO] Booting worker with pid: 1297
[2022-09-18 15:58:38 +0000] [1298] [INFO] Booting worker with pid: 1298
[2022-09-18 15:58:38 +0000] [1299] [INFO] Booting worker with pid: 1299
[2022-09-18 15:58:38 +0000] [1300] [INFO] Booting worker with pid: 1300
[2022-09-18 16:36:07 +0000] [1295] [INFO] Handling signal: winch
[2022-09-18 16:36:14 +0000] [1295] [INFO] Handling signal: winch
[2022-09-18 16:36:14 +0000] [1295] [INFO] Handling signal: winch
[2022-09-18 16:36:14 +0000] [1295] [INFO] Handling signal: winch
```



The screenshot shows a terminal window with multiple tabs open, all related to airflow. The current tab displays the output of the command `airflow scheduler`. The log output is as follows:

```
root@79feed3a0d6a:~# airflow scheduler
[2022-09-18 15:59:35,674] {scheduler_job.py:708} INFO - Starting the scheduler
[2022-09-18 15:59:35,674] {scheduler_job.py:713} INFO - Processing each file at most -1 times
[2022-09-18 15:59:35,676] {executor_loader.py:105} INFO - Loaded executor: SequentialExecutor
[2022-09-18 15:59:35 +0000] [1419] [INFO] Starting gunicorn 20.1.0
[2022-09-18 15:59:35,683] {manager.py:160} INFO - Launched DagFileProcessorManager with pid: 1420
[2022-09-18 15:59:35,684] {scheduler_job.py:1233} INFO - Resetting orphaned tasks for active dag runs
[2022-09-18 15:59:35 +0000] [1419] [INFO] Listening at: http://0.0.0.0:8793 (1419)
[2022-09-18 15:59:35 +0000] [1419] [INFO] Using worker: sync
[2022-09-18 15:59:35,694] {scheduler_job.py:1256} INFO - Marked 1 SchedulerJob instances as failed
[2022-09-18 15:59:35 +0000] [1421] [INFO] Booting worker with pid: 1421
[2022-09-18 15:59:35,694] {settings.py:55} INFO - Configured default timezone Timezone('UTC')
[2022-09-18 15:59:35,705] {manager.py:406} WARNING - Because we cannot use more than 1 thread (parsing_processes = 2) when using sqlite. So we set parallelism to 1.
[2022-09-18 15:59:35 +0000] [1422] [INFO] Booting worker with pid: 1422
[2022-09-18 16:00:02,467] {dagbag.py:2968} INFO - Setting next_dagrun for Lead_scoring_inference_pipeline to 2022-09-18T16:00:00+00:00, run_after_start=2022-09-18T17:00:00+00:00
[2022-09-18 16:00:02,512] {scheduler_job.py:353} INFO - 1 tasks up for execution:
    <TaskInstance Lead_scoring_inference_pipeline.encoding_categorical_variables scheduled_2022-09-18T15:00+00:00> [scheduled]
[2022-09-18 16:00:02,513] {scheduler_job.py:418} INFO - DAG Lead_scoring_inference_pipeline has 0/16 running and queued tasks
[2022-09-18 16:00:02,513] {scheduler_job.py:504} INFO - Setting the following tasks to queued state:
    <TaskInstance Lead_scoring_inference_pipeline.encoding_categorical_variables scheduled_2022-09-18T15:00+00:00> [scheduled]
[2022-09-18 16:00:02,515] {scheduler_job.py:546} INFO - Sending TaskInstanceStateKey(dag_id='Lead_scoring_inference_pipeline', task_id='encoding_categorical_variables', run_id='scheduled_2022-09-18T15:00+00:00', try_number=1, map_index=1) to executor with priority 4 and queue default
[2022-09-18 16:00:02,515] {base_executor.py:91} INFO - Adding to queue: ['airflow', 'tasks', 'run', 'Lead_scoring_inference_pipeline', 'encoding_categorical_variables', 'scheduled_2022-09-18T15:00+00:00', '-local', '-subdir', 'DAGS_FOLDER/Lead_scoring_inference_pipeline/lead_scoring_inference_pipeline.py']
[2022-09-18 16:00:02,517] {sequential_executor.py:59} INFO - Executing command: ['airflow', 'tasks', 'run', 'Lead_scoring_inference_pipeline', 'encoding_categorical_variables', 'scheduled_2022-09-18T15:00+00:00', '-local', '-subdir', 'DAGS_FOLDER/Lead_scoring_inference_pipeline/lead_scoring_inference_pipeline.py']
[2022-09-18 16:00:03,239] {dagbag.py:508} INFO - Filling up the DagBag from /home/dags/Lead_scoring_inference_pipeline/lead_scoring_inference_pipeline.py
[2022-09-18 16:00:04,206] {utils.py:145} INFO - Note: NumExpr detected 64 cores but "NUMEXPR_MAX_THREADS" not set, so enforcing safe limit of 8 .
[2022-09-18 16:00:04,206] {utils.py:157} INFO - NumExpr defaulting to 8 threads.
[2022-09-18 16:00:05,443] {example_kubernetes_executor.py:39} WARNING - The example_kubernetes_executor example DAG requires the kubernetes provider. Please install it with pip install apache-airflow[cnfc.kubernetes]
[2022-09-18 16:00:05,446] {example_local_kubernetes_executor.py:37} WARNING - Could not import DAGs in example_local_kubernetes_executor.py
```