

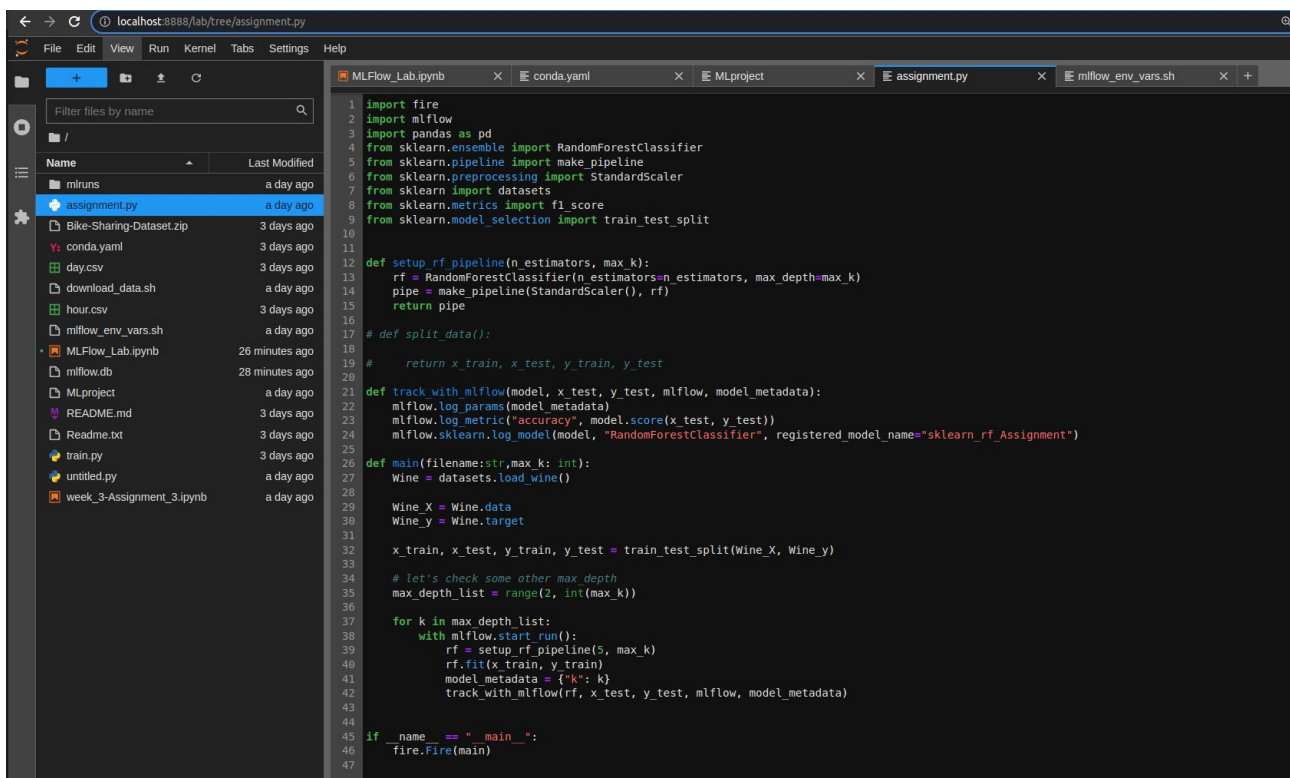
# Group Member's Name:-

**Muhammad Arsalan (2303.KHI.DEG.025)**

**Abdul Rehman (2303.KHI.DEG.035)**

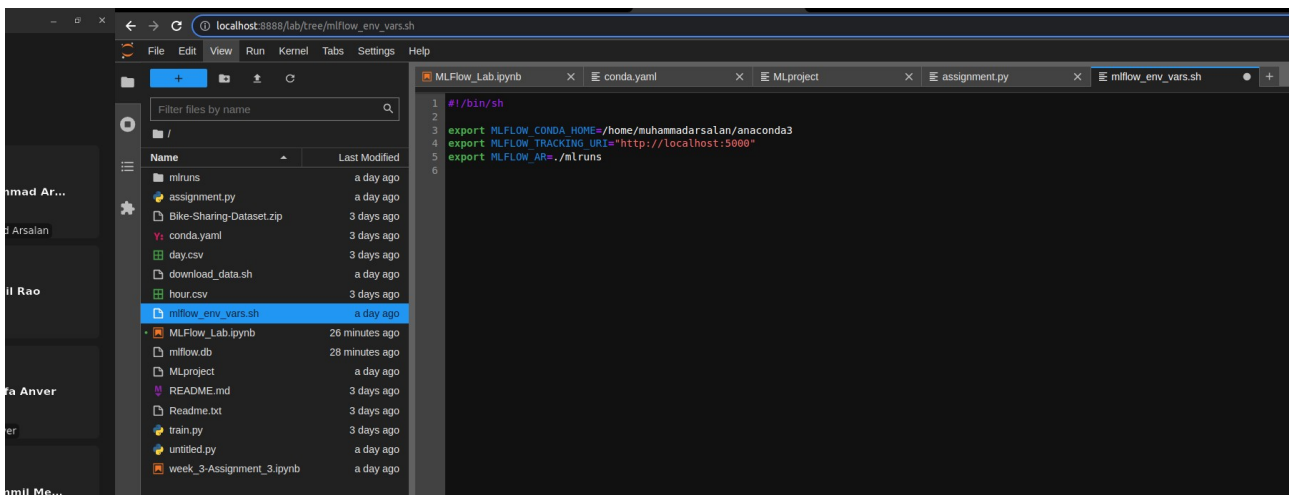
**Arshad Shiwani (2303.KHI.DEG.026)**

This is the Assignment.py file in which we used the RandomForestClassifier.



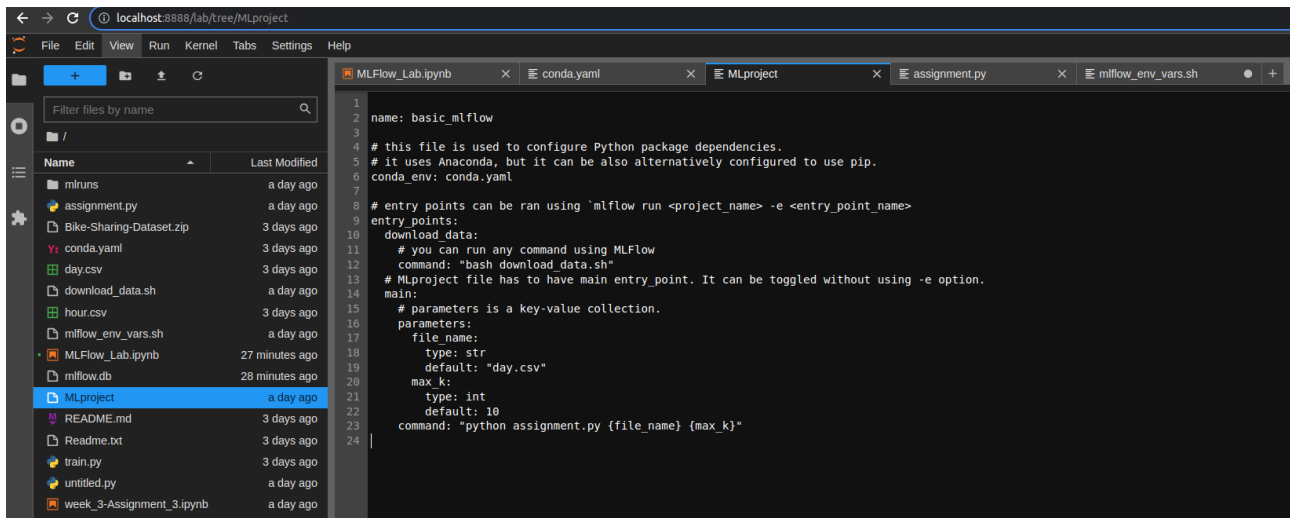
```
1 import fire
2 import mlflow
3 import pandas as pd
4 from sklearn.ensemble import RandomForestClassifier
5 from sklearn.pipeline import make_pipeline
6 from sklearn.preprocessing import StandardScaler
7 from sklearn import datasets
8 from sklearn.metrics import f1_score
9 from sklearn.model_selection import train_test_split
10
11
12 def setup_rf_pipeline(n_estimators, max_k):
13     rf = RandomForestClassifier(n_estimators=n_estimators, max_depth=max_k)
14     pipe = make_pipeline(StandardScaler(), rf)
15     return pipe
16
17 # def split_data():
18 #     return x_train, x_test, y_train, y_test
19
20
21 def track_with_mlflow(model, x_test, y_test, mlflow, model_metadata):
22     mlflow.log_params(model_metadata)
23     mlflow.log_metric("accuracy", model.score(x_test, y_test))
24     mlflow.sklearn.log_model(model, "RandomForestClassifier", registered_model_name="sklearn_rf_Assignment")
25
26 def main(filename:str,max_k: int):
27     Wine = datasets.load_wine()
28
29     Wine_X = Wine.data
30     Wine_y = Wine.target
31
32     x_train, x_test, y_train, y_test = train_test_split(Wine_X, Wine_y)
33
34     # let's check some other max depth
35     max_depth_list = range(2, int(max_k))
36
37     for k in max_depth_list:
38         with mlflow.start_run():
39             rf = setup_rf_pipeline(5, max_k)
40             rf.fit(x_train, y_train)
41             model_metadata = {"k": k}
42             track_with_mlflow(rf, x_test, y_test, mlflow, model_metadata)
43
44
45 if __name__ == "__main__":
46     fire.Fire(main)
47
```

This is the mlflow\_env\_vars.sh file in which we set the environment and localhost port and ./mlruns folder location where versions are saved.



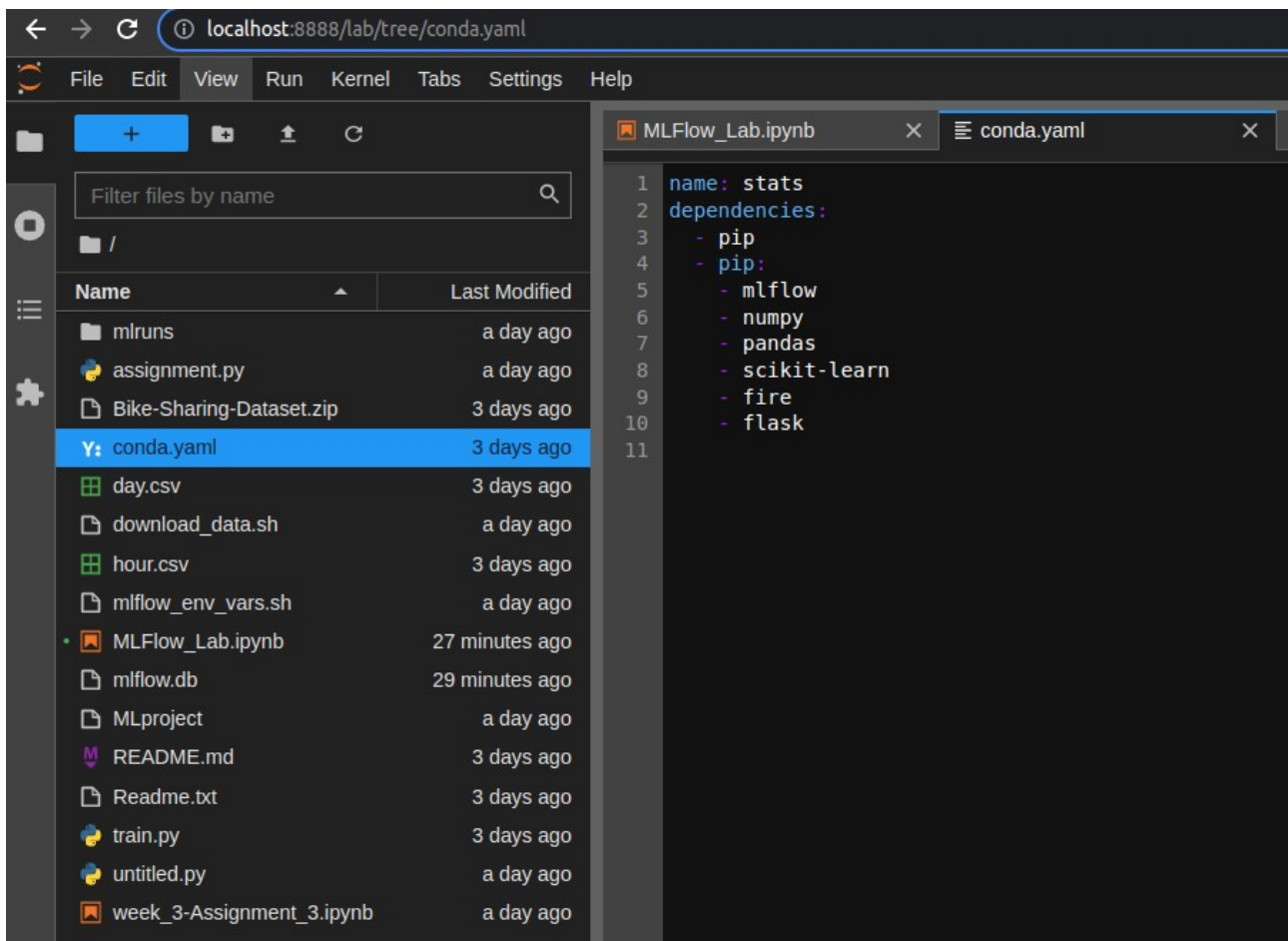
```
1 #!/bin/sh
2
3 export MLFLOW_CONDA_HOME=/home/muhammadsalan/anaconda3
4 export MLFLOW_TRACKING_URI="http://localhost:5000"
5 export MLFLOW_AR=./mlruns
6
```

This is the MLproject file in which we set the parameters and assign the assignment.py file and we use the RandomForestClassifier so we give the Depth so we gave the Max\_k which default 10 as we set and also used the file\_name.



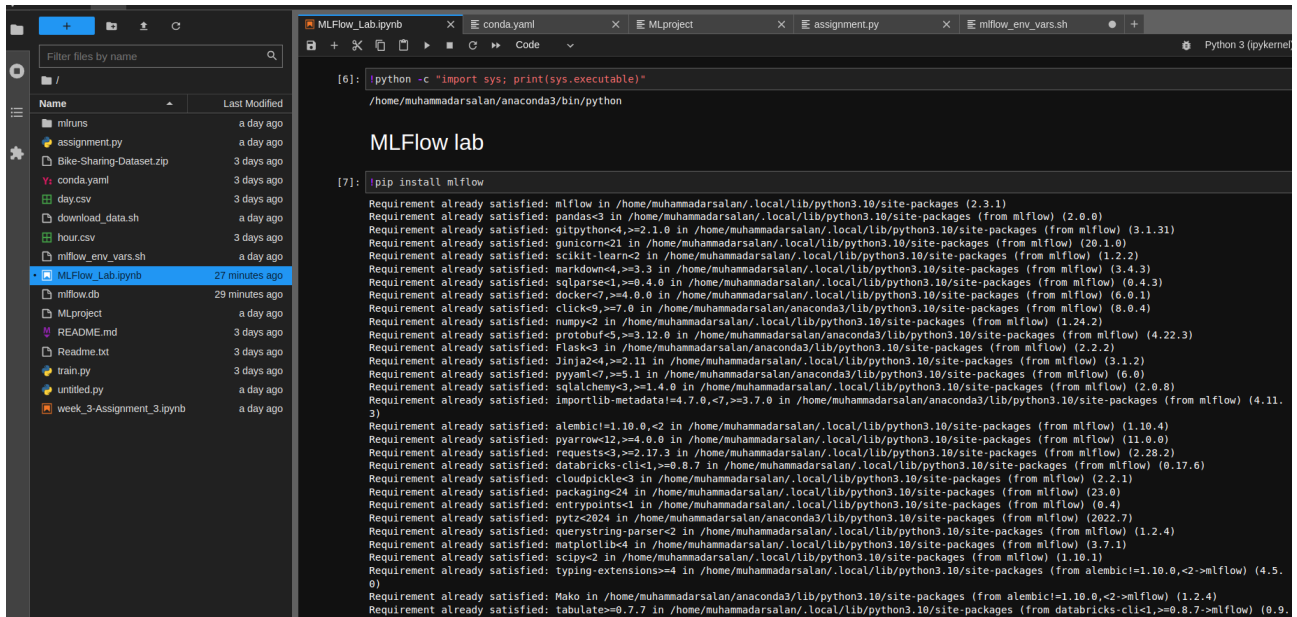
```
1 name: basic_mlflow
2
3 # this file is used to configure Python package dependencies.
4 # it uses Anaconda, but it can be also alternatively configured to use pip.
5 conda_env: conda.yaml
6
7 # entry points can be ran using `mlflow run <project_name> -e <entry_point_name>`
8 entry_points:
9   download_data:
10     # you can run any command using MLflow
11     command: "bash download_data.sh"
12 # MLproject file has to have main entry_point. It can be toggled without using -e option.
13 main:
14   # parameters is a key-value collection.
15   parameters:
16     file_name:
17       type: str
18       default: "day.csv"
19     max_k:
20       type: int
21       default: 10
22   command: "python assignment.py {file_name} {max_k}"
```

This is the conda.yaml file in which we set the dependencies which we used in your Project.



```
1 name: stats
2 dependencies:
3   - pip
4   - pip:
5     - mlflow
6     - numpy
7     - pandas
8     - scikit-learn
9     - fire
10    - flask
```

This is the ML\_Flow\_lab.ipynb file in which we Firstly install mlflow which we used in your Project.



The screenshot shows a Jupyter Notebook interface with a file explorer on the left and a code editor on the right. The file explorer shows a list of files including 'mlruns', 'assignment.py', 'Bike-Sharing-Dataset.zip', 'conda.yaml', 'download\_data.sh', 'hour.csv', 'mlflow\_env\_vars.sh', 'MLFlow\_Lab.ipynb' (selected), 'mlflow.db', 'MLproject', 'README.md', 'Readme.txt', 'train.py', 'untitled.py', and 'week\_3-Assignment\_3.ipynb'. The code editor shows the following code:

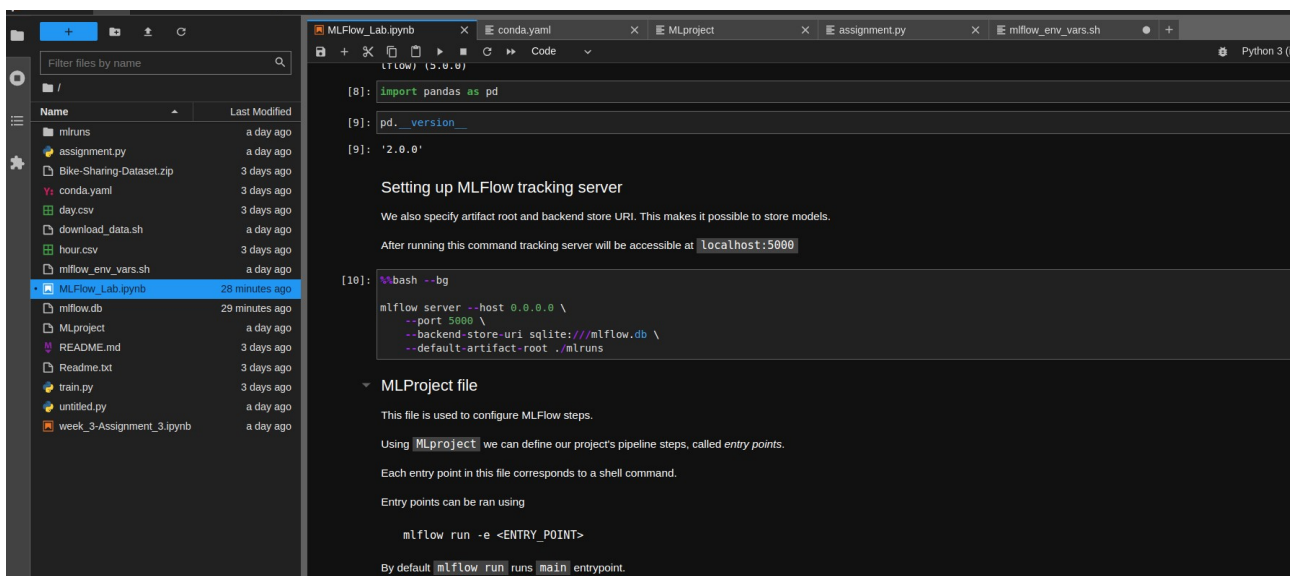
```
[6]: !python -c "import sys; print(sys.executable)"
/home/muhammadsalan/anaconda3/bin/python

MLFlow lab

[7]: !pip install mlflow
```

The output of the code shows a list of requirements that are already satisfied, including mlflow, pandas, gitpython, gunicorn, scikit-learn, markdown, sqlalchemy, click, numpy, protobuf, flask, Jinja2, pyyaml, SQLAlchemy, importlib-metadata, alembic, pyarrow, requests, databricks-cli, cloudpickle, packaging, entrypoints, querystring-parser, matplotlib, scipy, typing-extensions, and Mako.

This is the ML\_Flow\_lab.ipynb file in which we run our Project and check the result weather it is running or not in <http://localhost:5000/#/models>.



The screenshot shows a Jupyter Notebook interface with a file explorer on the left and a code editor on the right. The file explorer shows a list of files including 'mlruns', 'assignment.py', 'Bike-Sharing-Dataset.zip', 'conda.yaml', 'download\_data.sh', 'hour.csv', 'mlflow\_env\_vars.sh', 'MLFlow\_Lab.ipynb' (selected), 'mlflow.db', 'MLproject', 'README.md', 'Readme.txt', 'train.py', 'untitled.py', and 'week\_3-Assignment\_3.ipynb'. The code editor shows the following code:

```
[8]: import pandas as pd
[9]: pd.__version__
[9]: '2.0.0'

Setting up MLFlow tracking server

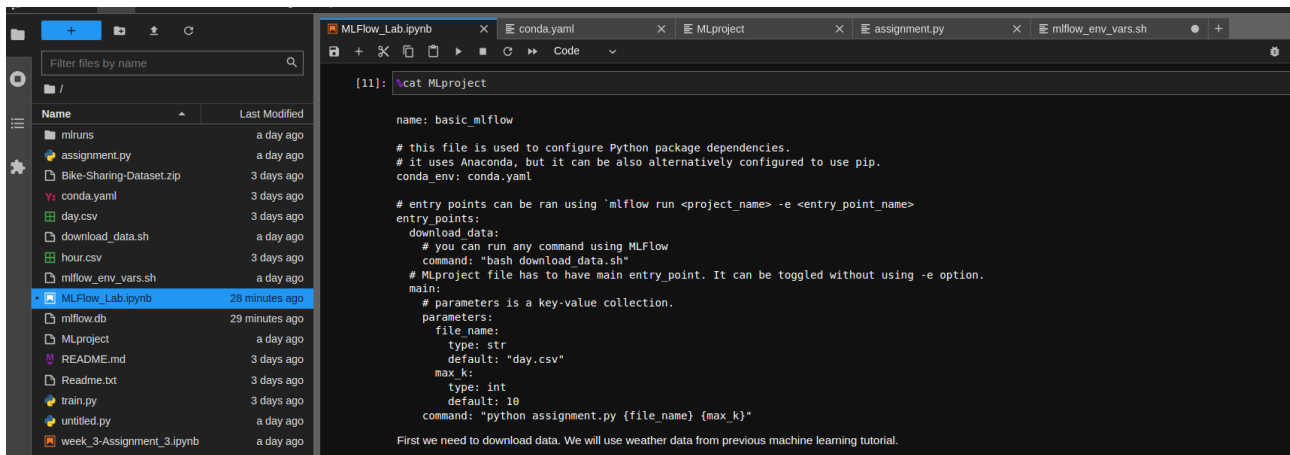
We also specify artifact root and backend store URI. This makes it possible to store models.

After running this command tracking server will be accessible at localhost:5000

[10]: %bash -bg
mlflow server --host 0.0.0.0 \
--port 5000 \
--backend-store-uri sqlite:///mlflow.db \
--default-artifact-root ./mlruns
```

The code also includes a section titled 'MLProject file' which explains that this file is used to configure MLFlow steps, called entry points. It states that each entry point in this file corresponds to a shell command and that entry points can be run using the command `mlflow run -e <ENTRY_POINT>`. It also mentions that by default `mlflow run` runs the `main` entrypoint.

This is the cat command which is used to reads each File parameter in sequence and writes it to standard output.



```
[11]: %cat MLproject

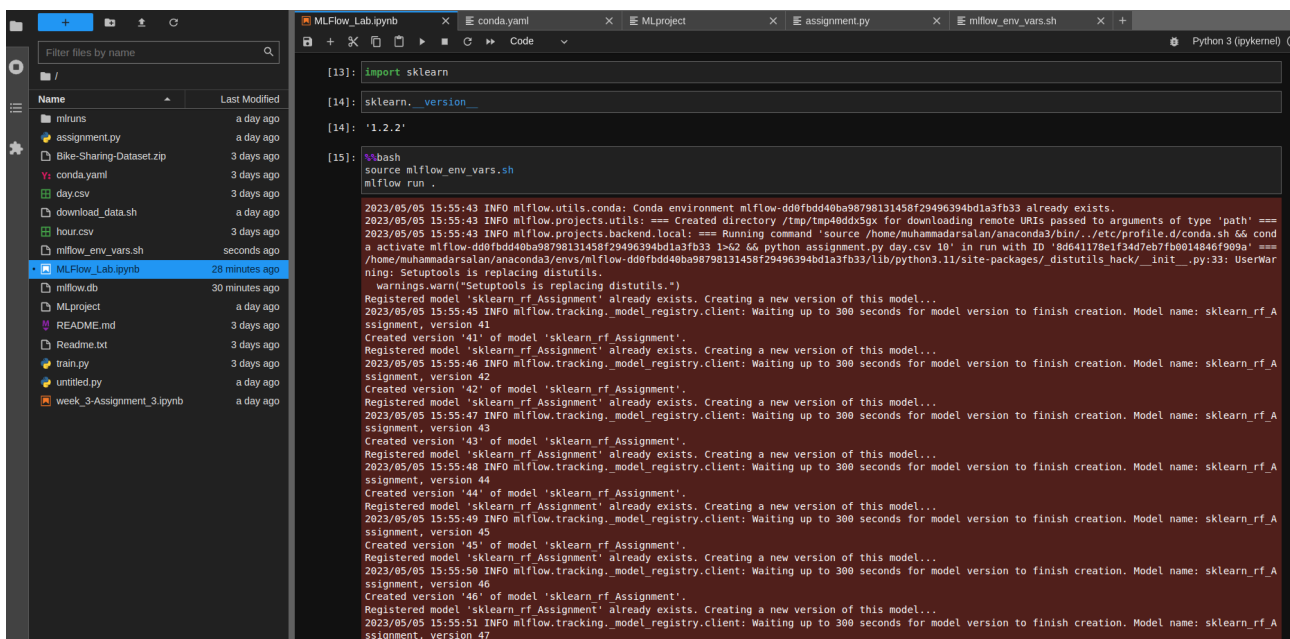
name: basic_mlflow

# this file is used to configure Python package dependencies.
# it uses Anaconda, but it can be also alternatively configured to use pip.
conda_env: conda.yaml

# entry points can be ran using `mlflow run <project_name> -e <entry_point_name>`
entry_points:
  download_data:
    # you can run any command using MLflow
    command: "bash download_data.sh"
  # MLproject file has to have main entry_point. It can be toggled without using -e option.
  main:
    # parameters is a key-value collection.
    parameters:
      file_name:
        type: str
        default: "day.csv"
      max_k:
        type: int
        default: 10
    command: "python assignment.py {file_name} {max_k}"

First we need to download data. We will use weather data from previous machine learning tutorial.
```

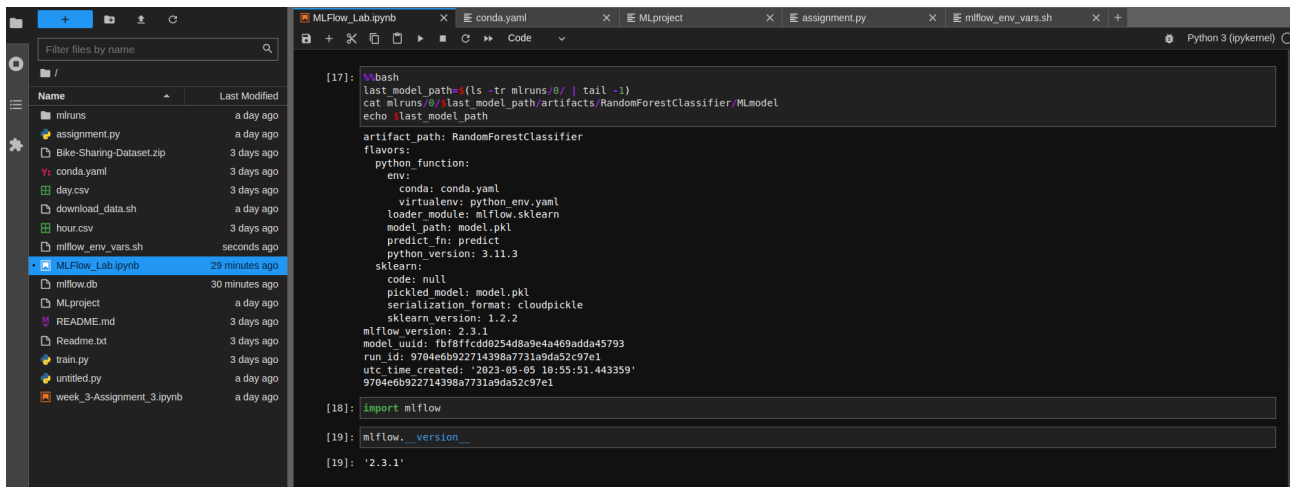
Now we import sklearn and source mlflow\_env\_vars.sh; mlflow run . runs an MLflow project in the current directory with the necessary environment variables sourced from mlflow\_env\_vars.sh. .



```
[13]: import sklearn
[14]: sklearn.__version__
[14]: '1.2.2'
[15]: %bash
      source mlflow_env_vars.sh
      mlflow run .

2023/05/05 15:55:43 INFO mlflow.utils.conda: Conda environment mlflow-dd0fbd40ba98798131458f29496394bd1a3fb33 already exists.
2023/05/05 15:55:43 INFO mlflow.projects.utils: == Created directory /tmp/tmp40ddx0gx for downloading remote URIs passed to arguments of type 'path' ==
2023/05/05 15:55:43 INFO mlflow.projects.backend.local: == Running command 'source /home/muhammadarsalan/anaconda3/bin/./etc/profile.d/conda.sh && conda
a activate mlflow-dd0fbd40ba98798131458f29496394bd1a3fb33 1-62 && python assignment.py day.csv 10' in run with ID '8d641178e1f34d7eb7fb9014846f999a' ==
/home/muhammadarsalan/anaconda3/envs/mlflow-dd0fbd40ba98798131458f29496394bd1a3fb33/lib/python3.11/site-packages/distutils/hack/_init_.py:33: UserWar
ning: Setuptools is replacing distutils.
warnings.warn("Setuptools is replacing distutils.")
Registered model 'sklearn_rf Assignment' already exists. Creating a new version of this model...
2023/05/05 15:55:45 INFO mlflow.tracking._model_registry.client: Waiting up to 300 seconds for model version to finish creation. Model name: sklearn_rf_A
ssignment, version 41
Created version '41' of model 'sklearn_rf Assignment'.
Registered model 'sklearn_rf Assignment' already exists. Creating a new version of this model...
2023/05/05 15:55:46 INFO mlflow.tracking._model_registry.client: Waiting up to 300 seconds for model version to finish creation. Model name: sklearn_rf_A
ssignment, version 42
Created version '42' of model 'sklearn_rf Assignment'.
Registered model 'sklearn_rf Assignment' already exists. Creating a new version of this model...
2023/05/05 15:55:47 INFO mlflow.tracking._model_registry.client: Waiting up to 300 seconds for model version to finish creation. Model name: sklearn_rf_A
ssignment, version 43
Created version '43' of model 'sklearn_rf Assignment'.
Registered model 'sklearn_rf Assignment' already exists. Creating a new version of this model...
2023/05/05 15:55:48 INFO mlflow.tracking._model_registry.client: Waiting up to 300 seconds for model version to finish creation. Model name: sklearn_rf_A
ssignment, version 44
Created version '44' of model 'sklearn_rf Assignment'.
Registered model 'sklearn_rf Assignment' already exists. Creating a new version of this model...
2023/05/05 15:55:49 INFO mlflow.tracking._model_registry.client: Waiting up to 300 seconds for model version to finish creation. Model name: sklearn_rf_A
ssignment, version 45
Created version '45' of model 'sklearn_rf Assignment'.
Registered model 'sklearn_rf Assignment' already exists. Creating a new version of this model...
2023/05/05 15:55:50 INFO mlflow.tracking._model_registry.client: Waiting up to 300 seconds for model version to finish creation. Model name: sklearn_rf_A
ssignment, version 46
Created version '46' of model 'sklearn_rf Assignment'.
Registered model 'sklearn_rf Assignment' already exists. Creating a new version of this model...
2023/05/05 15:55:51 INFO mlflow.tracking._model_registry.client: Waiting up to 300 seconds for model version to finish creation. Model name: sklearn_rf_A
ssignment, version 47
```

It will get the Model from previously train Model and Predict the newest train model.



```
[17]: %bash
last_model_path=$(ls -tr mlruns/0/ | tail -1)
cat mlruns/0/$last_model_path/artifacts/RandomForestClassifier/MLmodel
echo $last_model_path

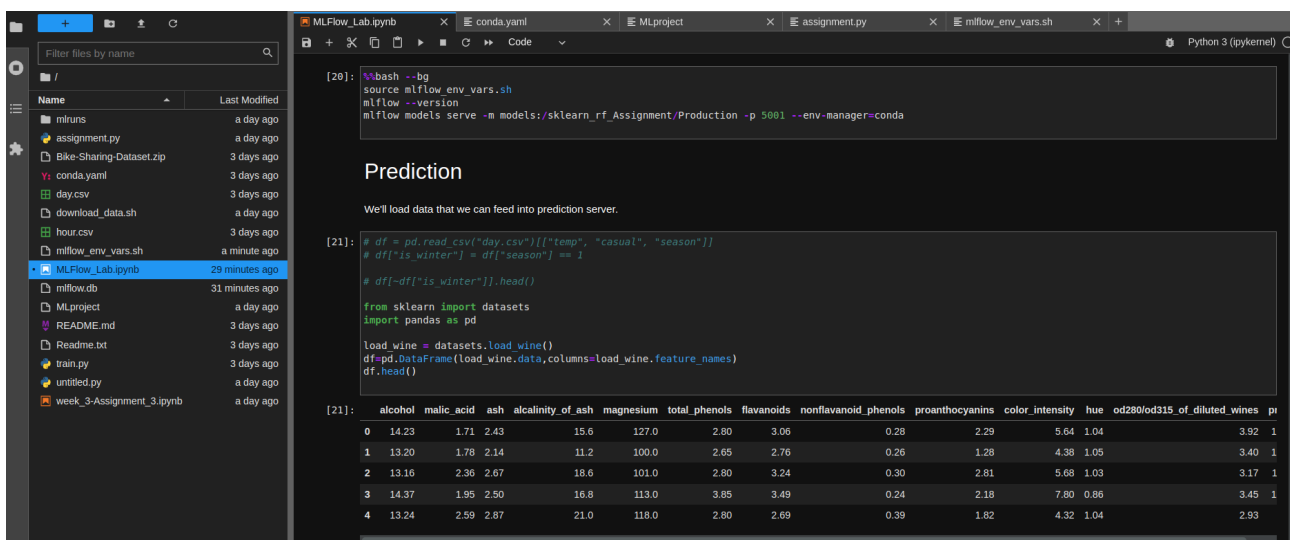
artifact_path: RandomForestClassifier
flavors:
  python_function:
    envs:
      conda: conda.yaml
      virtualenv: python_env.yaml
    loader_module: mlflow.sklearn
    model_path: model.pkl
    predict_fn: predict
    python_version: 3.11.3
  sklearn:
    code: null
    pickled_model: model.pkl
    serialization_format: cloudpickle
    sklearn_version: 1.2.2
mlflow version: 2.3.1
model_uuid: fb8ffcd0254d8a9e4a469adda45793
run_id: 9704e6b922714398a7731a9da52c97e1
utc_time_created: '2023-05-05 10:55:51.443359'
9704e6b922714398a7731a9da52c97e1

[18]: import mlflow

[19]: mlflow.__version__

[19]: '2.3.1'
```

Now we load the Data of wine data set and show the first five column through using head()).



```
[20]: %bash --bg
source mlflow_env_vars.sh
mlflow --version
mlflow models serve -m models/sklearn_rf_Assignment/Production -p 5001 --env-manager=conda

Prediction

We'll load data that we can feed into prediction server.

[21]: # df = pd.read_csv("day.csv")[["temp", "casual", "season"]]
# df["is_winter"] = df["season"] == 1
# df[~df["is_winter"]].head()

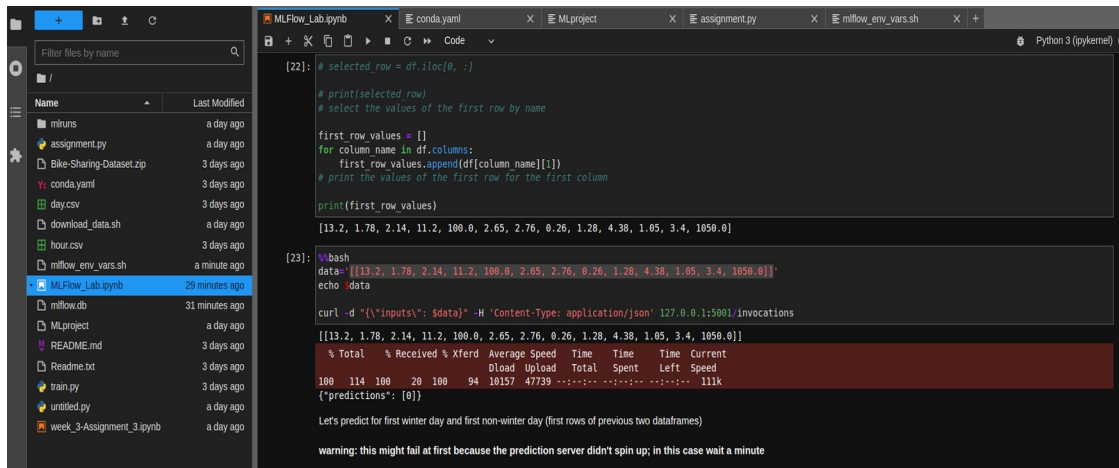
from sklearn import datasets
import pandas as pd

load_wine = datasets.load_wine()
df=pd.DataFrame(load_wine.data,columns=load_wine.feature_names)
df.head()

[21]:  alcohol  malic_acid  ash  alkalinity_of_ash  magnesium  total_phenols  flavanoids  nonflavanoid_phenols  proanthocyanins  color_intensity  hue  od280/od315_of_diluted_wines  pr
```

	alcohol	malic_acid	ash	alkalinity_of_ash	magnesium	total_phenols	flavanoids	nonflavanoid_phenols	proanthocyanins	color_intensity	hue	od280/od315_of_diluted_wines	pr
0	14.23	1.71	2.43	15.6	127.0	2.80	3.06	0.28	2.29	5.64	1.04	3.92	1
1	13.20	1.78	2.14	11.2	100.0	2.65	2.76	0.26	1.28	4.38	1.05	3.40	1
2	13.16	2.36	2.67	18.6	101.0	2.80	3.24	0.30	2.81	5.68	1.03	3.17	1
3	14.37	1.95	2.50	16.8	113.0	3.85	3.49	0.24	2.18	7.80	0.86	3.45	1
4	13.24	2.59	2.87	21.0	118.0	2.80	2.69	0.39	1.82	4.32	1.04	2.93	

Now we put the values to predict it.



```
[22]: # selected_row = df.iloc[0, :]
# print(selected_row)
# select the values of the first row by name

first_row_values = []
for column_name in df.columns:
    first_row_values.append(df[column_name][0])
# print the values of the first row for the first column

print(first_row_values)

[13.2, 1.78, 2.14, 11.2, 100.0, 2.65, 2.76, 0.26, 1.28, 4.38, 1.05, 3.4, 1050.0]

[23]: %bash
data='[[13.2, 1.78, 2.14, 11.2, 100.0, 2.65, 2.76, 0.26, 1.28, 4.38, 1.05, 3.4, 1050.0]]'
echo $data

curl -d '{"inputs": $data}' -H 'Content-Type: application/json' 127.0.0.1:5001/invocations

[[13.2, 1.78, 2.14, 11.2, 100.0, 2.65, 2.76, 0.26, 1.28, 4.38, 1.05, 3.4, 1050.0]]

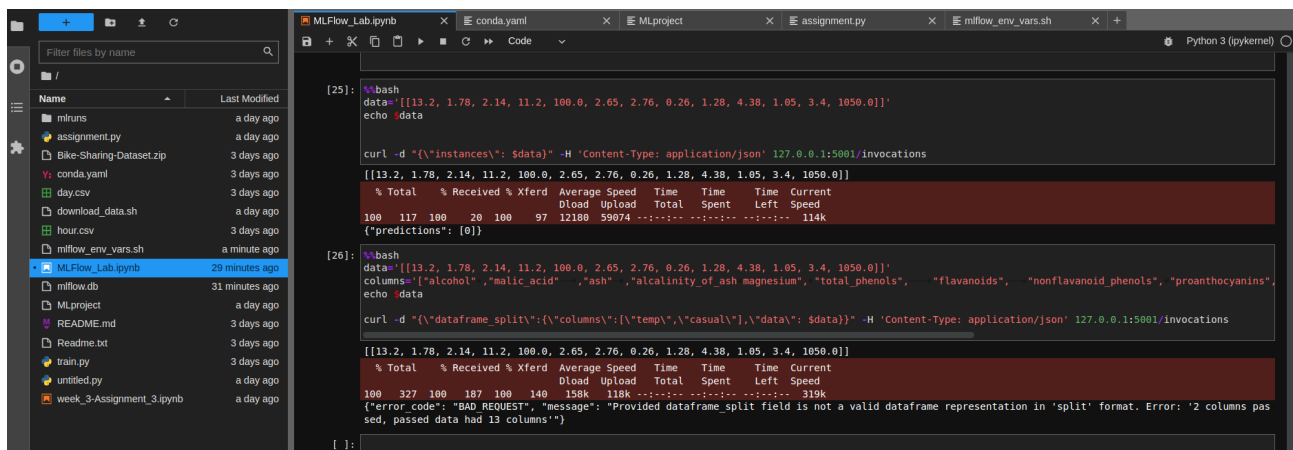
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left  Speed

100  114  100   20  100   94 10157  47739  --:--:-- --:--:-- --:--:--  111k

{"predictions": [0]}

Let's predict for first winter day and first non-winter day (first rows of previous two dataframes)

warning: this might fail at first because the prediction server didn't spin up; in this case wait a minute
```



```
[25]: %bash
data='[[13.2, 1.78, 2.14, 11.2, 100.0, 2.65, 2.76, 0.26, 1.28, 4.38, 1.05, 3.4, 1050.0]]'
echo $data

curl -d '{"instances": $data}' -H 'Content-Type: application/json' 127.0.0.1:5001/invocations

[[13.2, 1.78, 2.14, 11.2, 100.0, 2.65, 2.76, 0.26, 1.28, 4.38, 1.05, 3.4, 1050.0]]

% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left  Speed

100  117  100   20  100   97 12100  59074  --:--:-- --:--:-- --:--:--  114k

{"predictions": [0]}

[26]: %bash
data='[[13.2, 1.78, 2.14, 11.2, 100.0, 2.65, 2.76, 0.26, 1.28, 4.38, 1.05, 3.4, 1050.0]]'
columns='["alcohol", "malic_acid", "ash", "alcalinity_of_ash", "magnesium", "total_phenols", "flavanoids", "nonflavanoid_phenols", "proanthocyanins",
echo $data

curl -d '{"dataframe_split":{"columns":["temp","casual"],"data": $data}}' -H 'Content-Type: application/json' 127.0.0.1:5001/invocations

[[13.2, 1.78, 2.14, 11.2, 100.0, 2.65, 2.76, 0.26, 1.28, 4.38, 1.05, 3.4, 1050.0]]

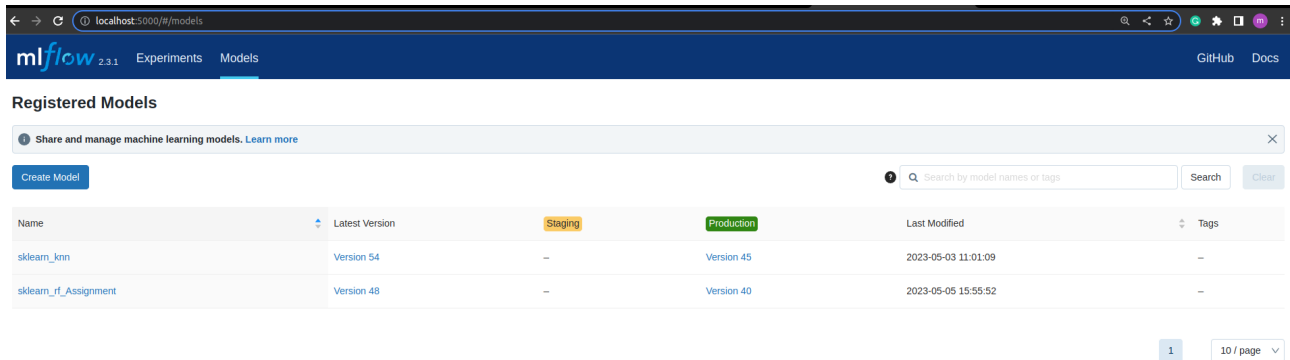
% Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
           Dload  Upload   Total   Spent    Left  Speed

100  327  100  187  100  140 158k  118k  --:--:-- --:--:-- --:--:--  319k

{"error code": "BAD REQUEST", "message": "Provided dataframe_split field is not a valid dataframe representation in 'split' format. Error: '2 columns pas
sed, passed data had 13 columns'"}

[ ]:
```

This is the <http://localhost:5000/>.

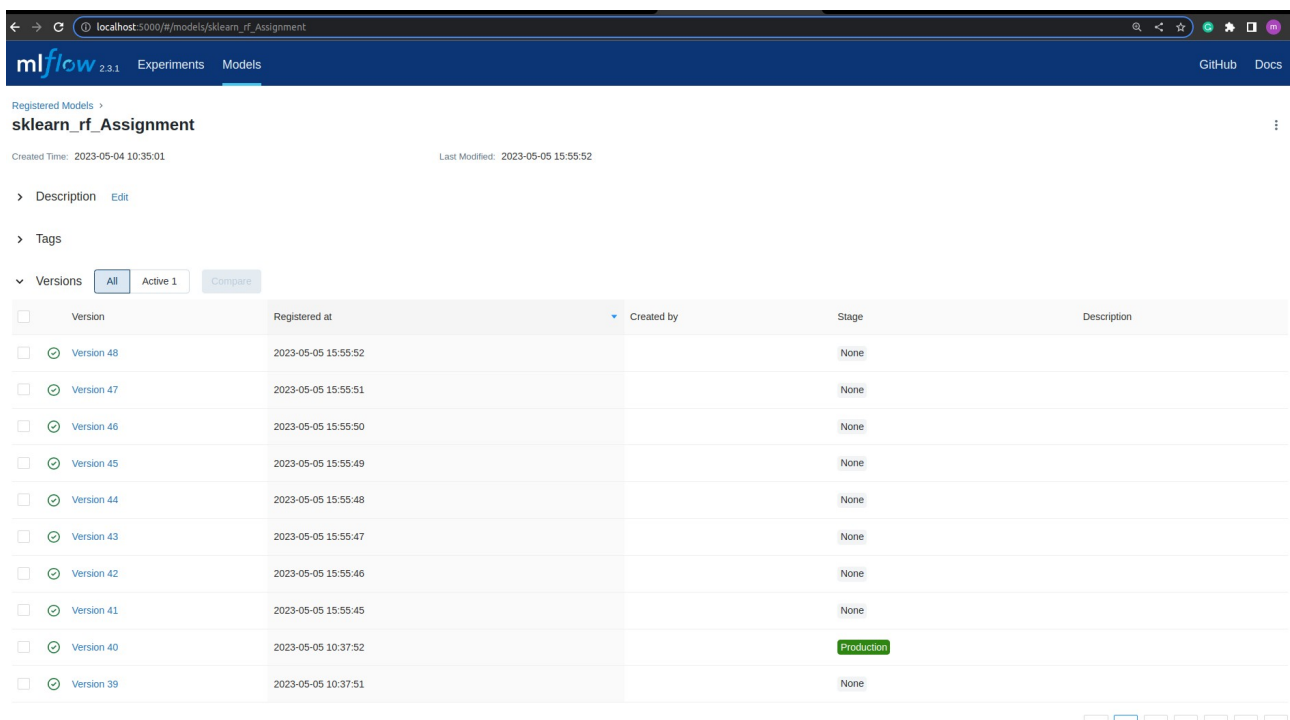


The screenshot shows the mlflow web interface at [localhost:5000/#/models](http://localhost:5000/#/models). The page title is "Registered Models". Below the title is a search bar with the placeholder text "Search by model names or tags". There are two tabs: "Create Model" and "Search". The main table lists the registered models:

Name	Latest Version	Staging	Production	Last Modified	Tags
sklearn_knn	Version 54	-	Version 45	2023-05-03 11:01:09	-
sklearn_rf_Assignment	Version 48	-	Version 40	2023-05-05 15:55:52	-

At the bottom right, there is a pagination control showing "1" and "10 / page".

When we click the Sklearn\_rf\_Assignment this screen Appears.



The screenshot shows the mlflow web interface at [localhost:5000/#/models/sklearn\\_rf\\_Assignment](http://localhost:5000/#/models/sklearn_rf_Assignment). The page title is "Registered Models > sklearn\_rf\_Assignment". Below the title, it shows the "Created Time: 2023-05-04 10:35:01" and "Last Modified: 2023-05-05 15:55:52". There are two tabs: "Description" and "Edit". Below the tabs, there is a "Tags" section. The "Versions" section is expanded, showing a list of versions:

Version	Registered at	Created by	Stage	Description
<input type="checkbox"/> Version 48	2023-05-05 15:55:52		None	
<input type="checkbox"/> Version 47	2023-05-05 15:55:51		None	
<input type="checkbox"/> Version 46	2023-05-05 15:55:50		None	
<input type="checkbox"/> Version 45	2023-05-05 15:55:49		None	
<input type="checkbox"/> Version 44	2023-05-05 15:55:48		None	
<input type="checkbox"/> Version 43	2023-05-05 15:55:47		None	
<input type="checkbox"/> Version 42	2023-05-05 15:55:46		None	
<input type="checkbox"/> Version 41	2023-05-05 15:55:45		None	
<input type="checkbox"/> Version 40	2023-05-05 10:37:52		Production	
<input type="checkbox"/> Version 39	2023-05-05 10:37:51		None	

At the bottom right, there is a pagination control showing "1" and "10 / page".



Now we put the version in production mode.

mlflow2.3.1ExperimentsModels

Registered Models > sklearn\_rf\_Assignment >

Version 40

Registered At: 2023-05-05 10:37:52Stage: ProductionLast Modified: 2023-05-05 11:10:19Source Run: loud-bug-71

> Description Edit

> Tags

> Schema

Name	Type
No schema. See <a href="#">MLflow docs</a> for how to include input and output schema with your model.	