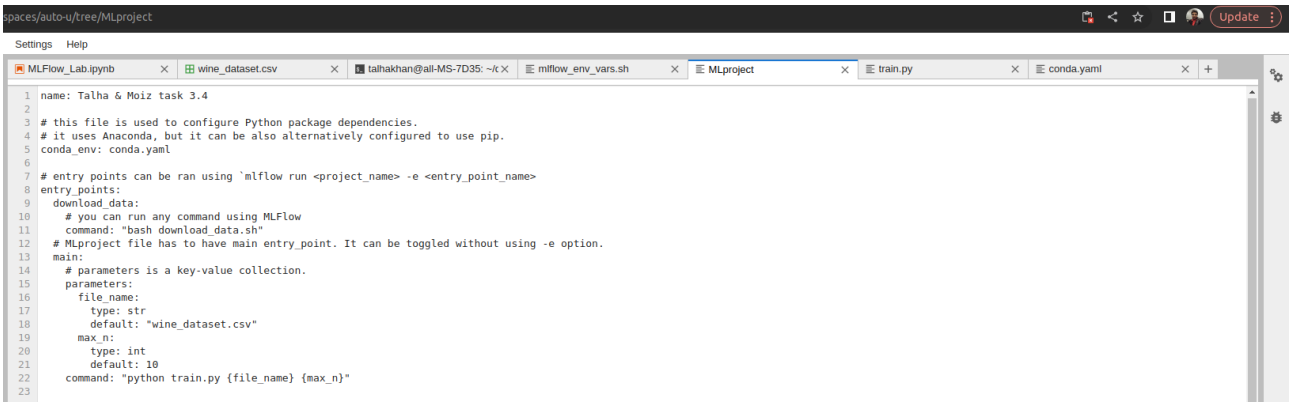
Talha Khan (2303.009.KHI.DEG)

Muhammad Moiz Khan (2303.022.KHI.DEG)

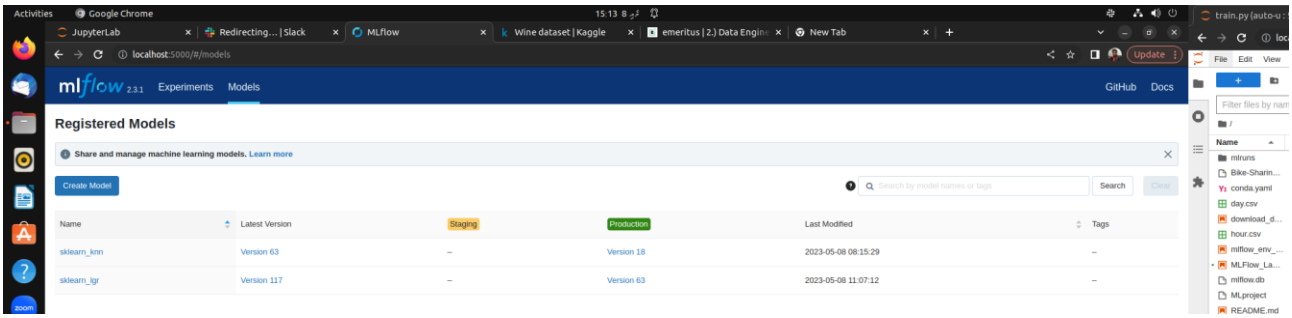
ASSIGNMNET 3.4

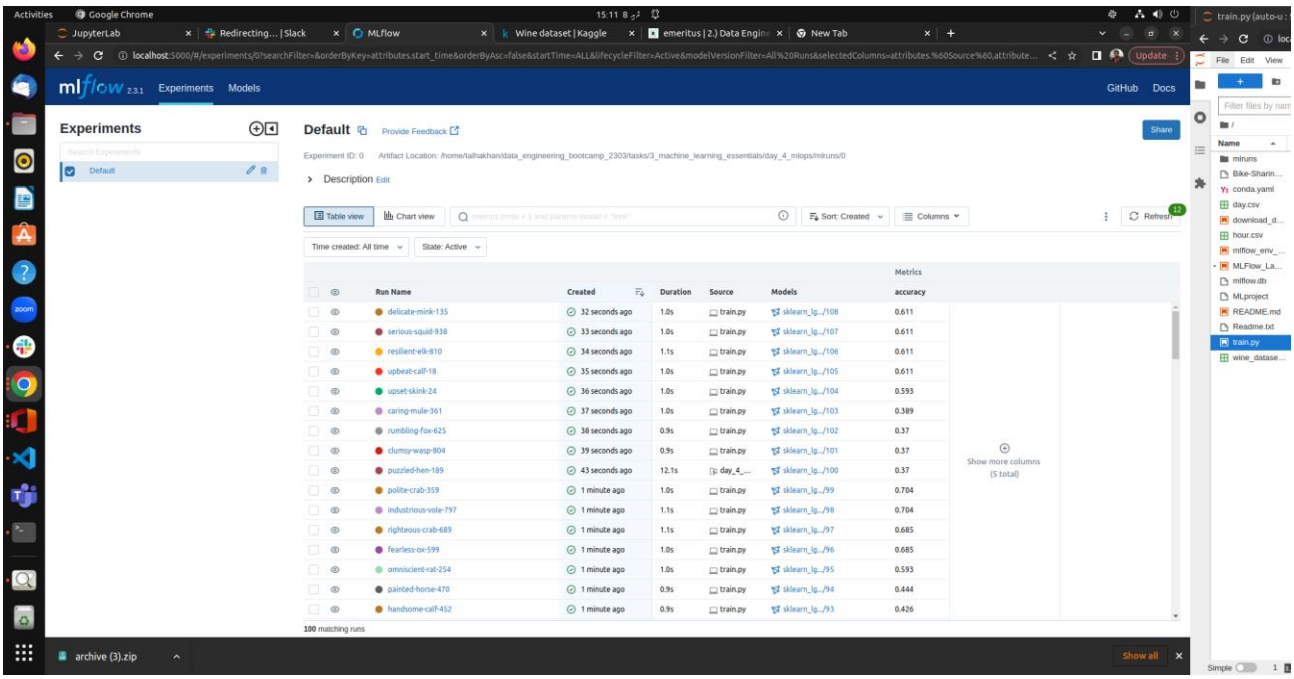


```
name: Talha & Moiz task 3.4

# 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: "wine_dataset.csv"
      max_n:
        type: int
        default: 10
    command: "python train.py {file_name} {max_n}"
```





Settings Help

MLFlow_Lab.ipynb | wine_dataset.csv | talhakhan@all-MS-7D35: ~/c | mlflow_env_vars.sh | MLproject | train.py | conda.yaml

```python
import fire
import mlflow
import pandas as pd
from sklearn.neighbors import KNeighborsClassifier
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn.datasets import load_wine
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import StandardScaler

def train_classification_model(n):
    # Initialize model and fit to training data
    lgr = LogisticRegression(max_iter=n)
    return lgr

def split_data(df):
    x = df.data
    y = df.target

    # Split dataset into training and testing
    x_train, x_test, y_train, y_test = train_test_split(x,y,test_size=0.3)
    return x_train, x_test, y_train, y_test


def track_with_mlflow(model, x_test, y_test, mlflow, model_metadata):
    mlflow.log_params(model_metadata)
    mlflow.log_metric("accuracy", model.score(x_test, y_test))
    mlflow.sklearn.log_model(model, "lgr", registered_model_name="sklearn_lgr")


def main(file_name: str, max_n: int):
    wine= load_wine()
    df = pd.read_csv(file_name)

    x_train, x_test, y_train, y_test = split_data(wine)
    # let's check some other n
    n_list = range(1, max_n)

    for n in n_list:
        with mlflow.start_run():
            lgr_pipe = train_classification_model(n)
            lgr_pipe.fit(x_train, y_train)
            model_metadata = {"n": n}
            track_with_mlflow(lgr_pipe, x_test, y_test, mlflow, model_metadata)


if __name__ == "__main__":
    fire.Fire(main)

```

Python Ln 50, Col 1 Spaces: 4 train.py

Settings Help

MLFlow_Lab.ipynb | wine_dataset.csv | talhakhan@all-MS-7D35: ~/c | mlflow_env_vars.sh | MLproject | train.py | conda.yaml

Code Python 3 (ipykernel)

[42]:
```
!python -c "import sys; print(sys.executable)"
```
```
/home/talhakhan/anaconda3/bin/python
```

MLFlow lab ¶

[43]:
```
import pandas as pd
```

[44]:
```
pd.__version__
```

[44]:
```
'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`

[45]:
```
%%bash --bg

mlflow server --host 0.0.0.0 \
    --port 5000 \
    --backend-store-uri sqlite:///mlflow.db \
    --default-artifact-root ./mlruns
```

MLProject file

This file is used to configure MLFlow steps.

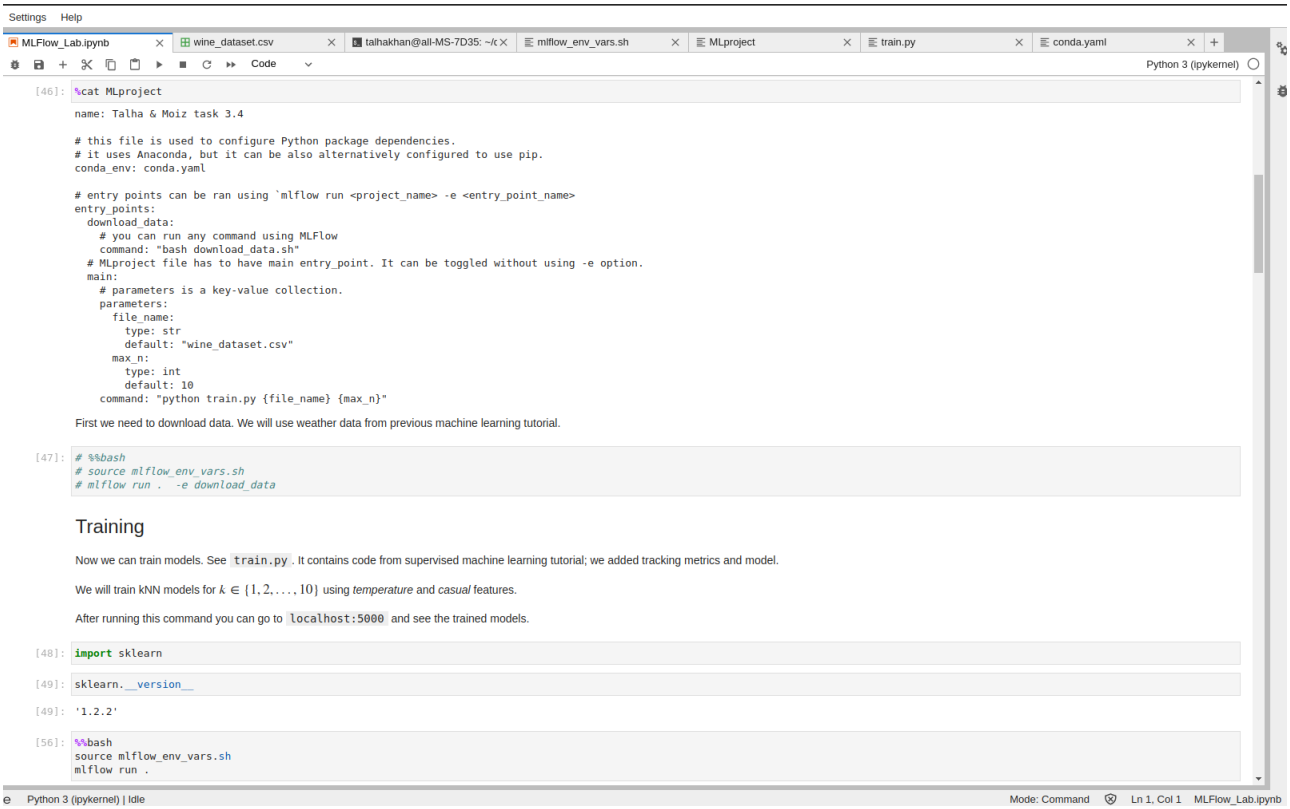
Using `MLproject` we can define our project's pipeline steps, called *entry points*.

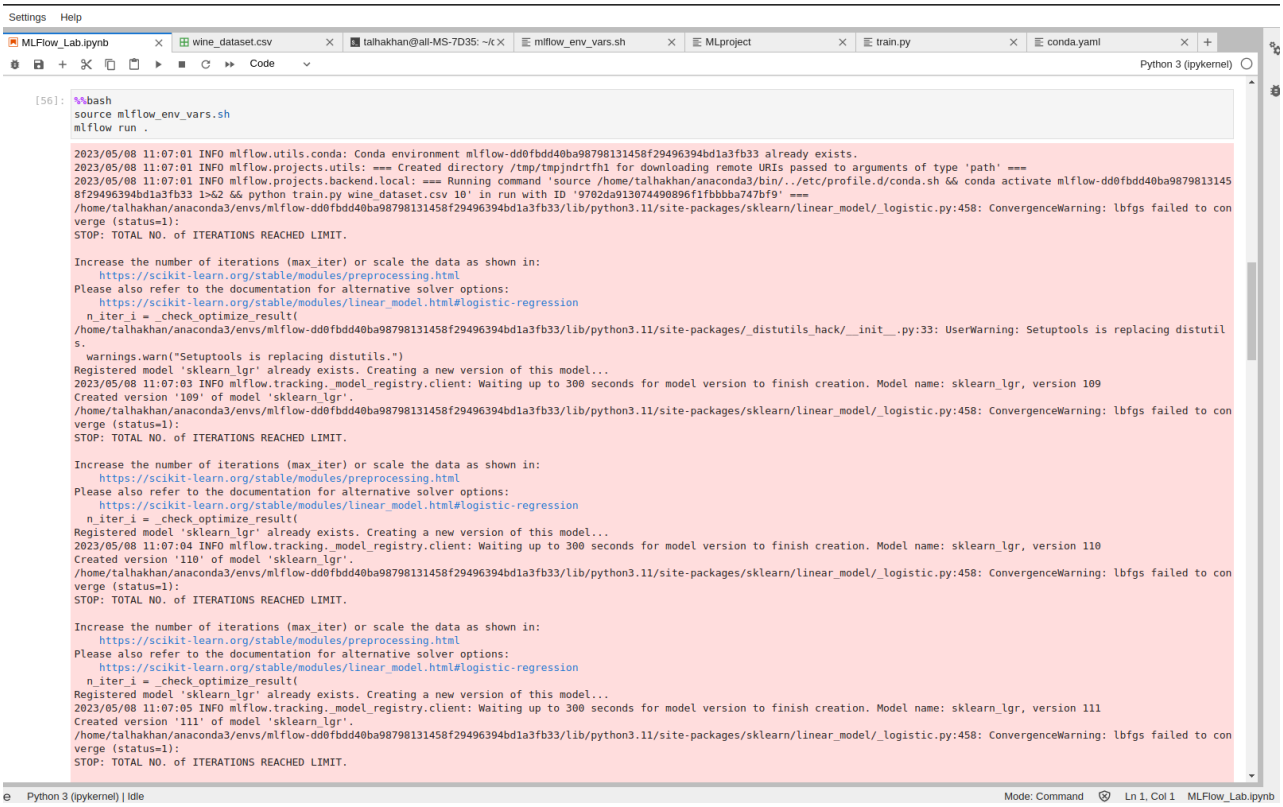
Each entry point in this file corresponds to a shell command.

Entry points can be ran using

```
mlflow run -e <ENTRY_POINT>
```

By default `mlflow run` runs `main` entrypoint.


Settings Help
MLFlow_Lab.ipynb
wine_dataset.csv
talhakhan@all-MS-7D35: ~/c
mlflow_env_vars.sh
MLproject
train.py
conda.yaml
Code
Python 3 (ipykernel)
[46]: %cat MLproject
name: Talha & Moiz task 3.4
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: "wine_dataset.csv"
max_n:
type: int
default: 10
command: "python train.py {file_name} {max_n}"
First we need to download data. We will use weather data from previous machine learning tutorial.
[47]: # %%bash
source mlflow_env_vars.sh
mlflow run . -e download_data
Training
Now we can train models. See train.py. It contains code from supervised machine learning tutorial; we added tracking metrics and model.
We will train kNN models for $k \in \{1, 2, \ldots, 10\}$ using temperature and casual features.
After running this command you can go to localhost:5000 and see the trained models.
[48]: import sklearn
[49]: sklearn.__version__
[49]: '1.2.2'
[56]: %%bash
source mlflow_env_vars.sh
mlflow run .
Python 3 (ipykernel) | Idle
Mode: Command
Ln 1, Col 1
MLFlow_Lab.ipynb


Settings Help
MLFlow_Lab.ipynb
wine_dataset.csv
talhakhan@all-MS-7D35: ~/c
mlflow_env_vars.sh
MLproject
train.py
conda.yaml
Code
Python 3 (ipykernel)
[56]: %%bash
source mlflow_env_vars.sh
mlflow run .
2023/05/08 11:07:01 INFO mlflow.utils.conda: Conda environment mlflow-dd0fbdd40ba98798131458f29496394bd1a3fb33 already exists.
2023/05/08 11:07:01 INFO mlflow.projects.utils: === Created directory /tmp/tmpjndrtfh1 for downloading remote URIs passed to arguments of type 'path' ===
2023/05/08 11:07:01 INFO mlflow.projects.backend.local: === Running command 'source /home/talhakhan/anaconda3/bin/../etc/profile.d/conda.sh && conda activate mlflow-dd0fbdd40ba98798131458f29496394bd1a3fb33 1>&2 && python train.py wine_dataset.csv 10' in run with ID '9702da913074490896f1fbbbba747bf9' ===
/home/talhakhan/anaconda3/envs/mlflow-dd0fbdd40ba98798131458f29496394bd1a3fb33/lib/python3.11/site-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
Increase the number of iterations (max_iter) or scale the data as shown in:
https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
n_iter_i = _check_optimize_result(
/home/talhakhan/anaconda3/envs/mlflow-dd0fbdd40ba98798131458f29496394bd1a3fb33/lib/python3.11/site-packages/_distutils_hack/__init__.py:33: UserWarning: Setuptools is replacing distutils.
warnings.warn("Setuptools is replacing distutils.")
Registered model 'sklearn_lgr' already exists. Creating a new version of this model...
2023/05/08 11:07:03 INFO mlflow.tracking._model_registry.client: Waiting up to 300 seconds for model version to finish creation. Model name: sklearn_lgr, version 109
Created version '109' of model 'sklearn_lgr'.
/home/talhakhan/anaconda3/envs/mlflow-dd0fbdd40ba98798131458f29496394bd1a3fb33/lib/python3.11/site-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
Increase the number of iterations (max_iter) or scale the data as shown in:
https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
n_iter_i = _check_optimize_result(
Registered model 'sklearn_lgr' already exists. Creating a new version of this model...
2023/05/08 11:07:04 INFO mlflow.tracking._model_registry.client: Waiting up to 300 seconds for model version to finish creation. Model name: sklearn_lgr, version 110
Created version '110' of model 'sklearn_lgr'.
/home/talhakhan/anaconda3/envs/mlflow-dd0fbdd40ba98798131458f29496394bd1a3fb33/lib/python3.11/site-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
Increase the number of iterations (max_iter) or scale the data as shown in:
https://scikit-learn.org/stable/modules/preprocessing.html
Please also refer to the documentation for alternative solver options:
https://scikit-learn.org/stable/modules/linear_model.html#logistic-regression
n_iter_i = _check_optimize_result(
Registered model 'sklearn_lgr' already exists. Creating a new version of this model...
2023/05/08 11:07:05 INFO mlflow.tracking._model_registry.client: Waiting up to 300 seconds for model version to finish creation. Model name: sklearn_lgr, version 111
Created version '111' of model 'sklearn_lgr'.
/home/talhakhan/anaconda3/envs/mlflow-dd0fbdd40ba98798131458f29496394bd1a3fb33/lib/python3.11/site-packages/sklearn/linear_model/_logistic.py:458: ConvergenceWarning: lbfgs failed to converge (status=1):
STOP: TOTAL NO. of ITERATIONS REACHED LIMIT.
Python 3 (ipykernel) | Idle
Mode: Command
Ln 1, Col 1
MLFlow_Lab.ipynb

Settings Help

MLFlow_Lab.ipynb | wine_dataset.csv | talhakhan@all-MS-7D35: ~/c | mlflow_env_vars.sh | MLproject | train.py | conda.yaml

Code — Python 3 (ipykernel)

Inspecting stored models

The trained models are stored in `mlruns/0` .

These directories contain artifacts and config that is needed to serve them.

[57]:
```
%%bash
last_model_path=$(ls -tr mlruns/0/ | tail -1)
cat mlruns/0/$last_model_path/artifacts/lgr/MLmodel
```

```
artifact_path: lgr
flavors:
  python_function:
    env:
      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: eaac5502aff242ada80b7d04523a4150
run_id: b2478ea4a9114fe0921530e2a898e98e
utc_time_created: '2023-05-08 06:07:11.098846'
```

[58]:
```
import mlflow
```

[59]:
```
mlflow.__version__
```

[59]:
```
'2.3.1'
```

Settings Help

MLFlow_Lab.ipynb | wine_dataset.csv | talhakhan@all-MS-7D35: ~/c | mlflow_env_vars.sh | MLproject | train.py | conda.yaml

Code — Python 3 (ipykernel)

Serving model

Now that we trained our models we can go to *Models* page on MLFLow UI (http://localhost:5000/#/models).

Click *sklearn_knn* on this page, choose a model and move it to *Production* stage.

The following cell will serve the model at localhost on port 5001.

[60]:
```
%%bash --bg
source mlflow_env_vars.sh
mlflow --version
mlflow models serve -m models:/sklearn_lgr/Production -p 5001 --env-manager=conda
```

Prediction

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

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

[64]:
```
%%bash
#data='[[0.344,331], [0.43, 401]]'
data='[[0.344,1.99,11.01,34.09,0.9978,3.51,0.56,331,9.4,5.00,15.6,28.2,5.64]]'
echo $data

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

```
[[0.344,1.99,11.01,34.09,0.9978,3.51,0.56,331,9.4,5.00,15.6,28.2,5.64]]
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   103  100    20  100    83   3373  14001 --:--:-- --:--:-- --:--:-- 20600
{"predictions": [1]}
```

[68]:
```
%%bash
data='[[0.344,1.99,11.01,34.09,0.9978,3.51,0.56,331,9.4,5.00,15.6,28.2,5.64]]'
echo $data

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

```
[[0.344,1.99,11.01,34.09,0.9978,3.51,0.56,331,9.4,5.00,15.6,28.2,5.64]]
  % Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                                 Dload  Upload   Total   Spent    Left  Speed
100   106  100    20  100    86  11730  50439 --:--:-- --:--:-- --:--:--  103k
{"predictions": [1]}
```

Voila! We see that the model outputs correct predictions.

Python 3 (ipykernel) | Idle — Mode: Command — Ln 1, Col 1 — MLFlow_Lab.ipynb