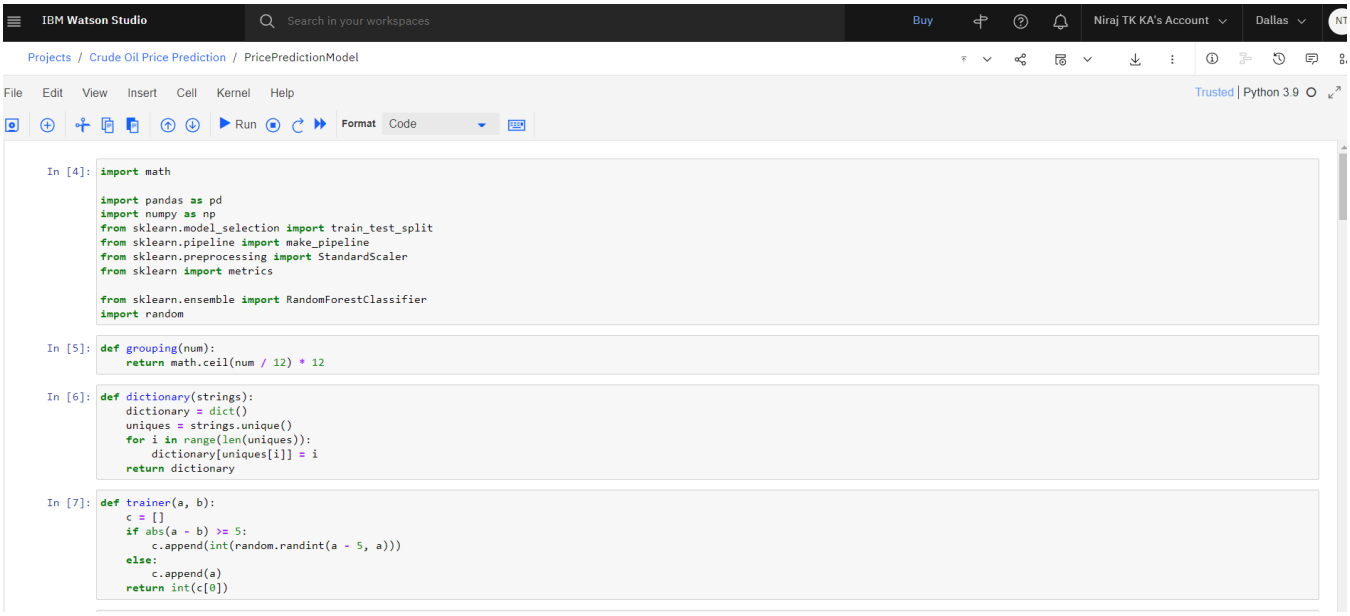
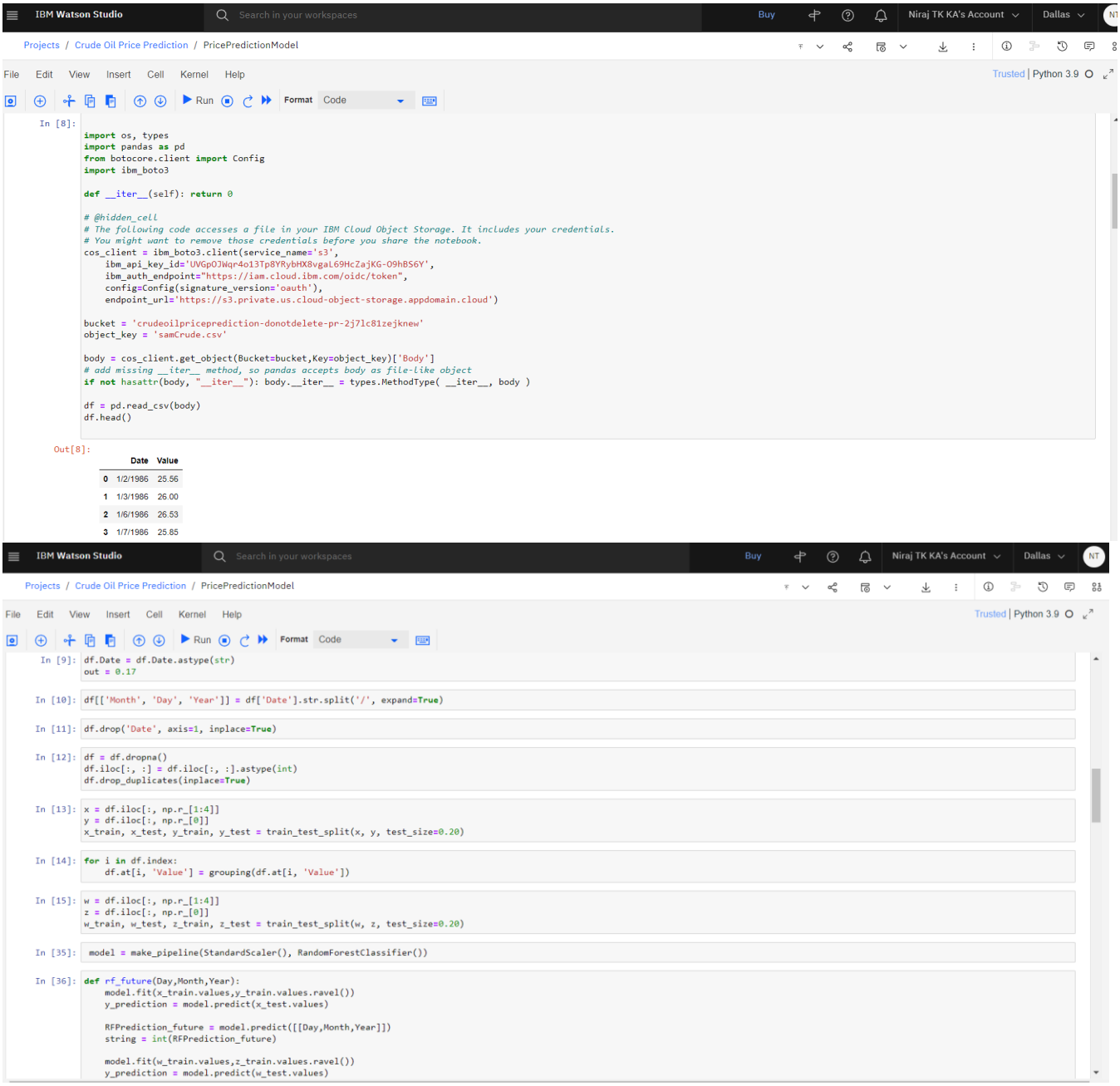
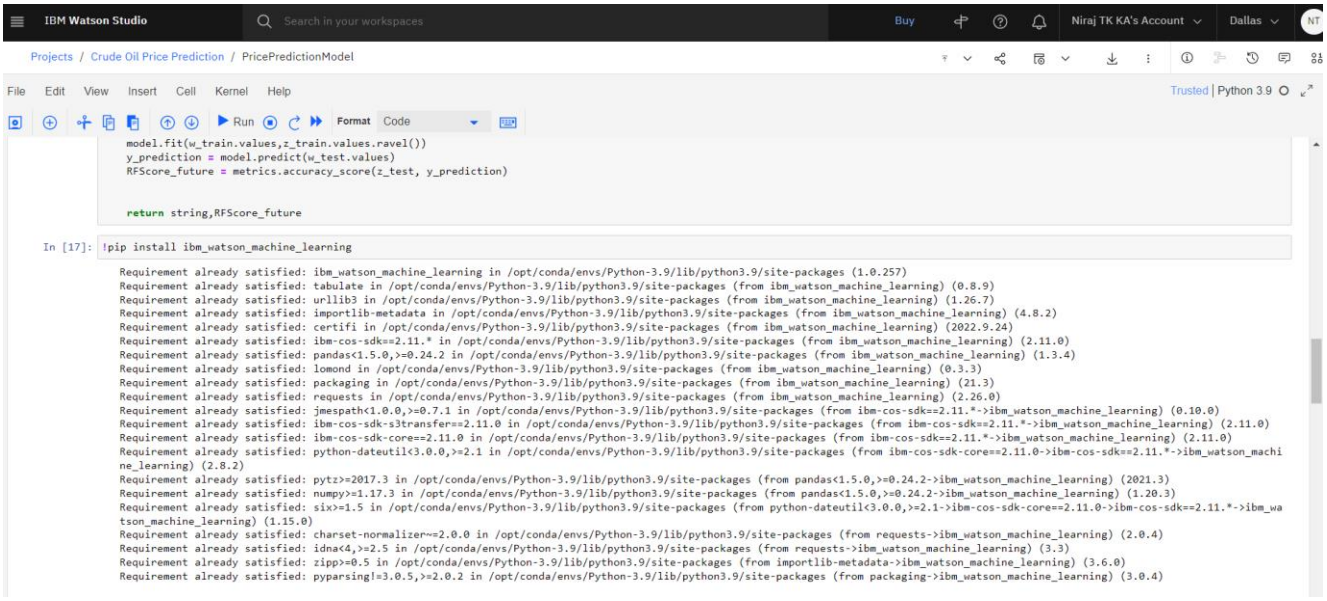

IBM Watson Studio
Search in your workspaces
Buy
Niraj TK KA's Account
Dallas
Projects / Crude Oil Price Prediction / PricePredictionModel
File Edit View Insert Cell Kernel Help
Trusted | Python 3.9
Run
Format Code
In [4]:
import math
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import StandardScaler
from sklearn import metrics
from sklearn.ensemble import RandomForestClassifier
import random
In [5]:
def grouping(num):
return math.ceil(num / 12) * 12
In [6]:
def dictionary(strings):
dictionary = dict()
uniques = strings.unique()
for i in range(len(uniques)):
dictionary[uniques[i]] = i
return dictionary
In [7]:
def trainer(a, b):
c = []
if abs(a - b) >= 5:
c.append(int(random.randint(a - 5, a)))
else:
c.append(a)
return int(c[0])


IBM Watson Studio
Search in your workspaces
Buy
Niraj TK KA's Account
Dallas
Projects / Crude Oil Price Prediction / PricePredictionModel
File Edit View Insert Cell Kernel Help
Trusted | Python 3.9
Run
Format Code
In [8]:
import os, types
import pandas as pd
from botocore.client import Config
import ibm_boto3
def __iter__(self): return 0
# @hidden_cell
# The following code accesses a file in your IBM Cloud Object Storage. It includes your credentials.
# You might want to remove those credentials before you share the notebook.
cos_client = ibm_boto3.client(service_name='s3',
ibm_api_key_id='UVGpOJWqr4o13Tp8YRybHX8vgaL69HcZajKG-O9hBS6Y',
ibm_auth_endpoint="https://iam.cloud.ibm.com/oidc/token",
config=Config(signature_version='oauth'),
endpoint_url='https://s3.private.us.cloud-object-storage.appdomain.cloud')
bucket = 'crudeoilpriceprediction-donotdelete-pr-2j7lc81zejknew'
object_key = 'samCrude.csv'
body = cos_client.get_object(Bucket=bucket,Key=object_key)['Body']
# add missing __iter__ method, so pandas accepts body as file-like object
if not hasattr(body, "__iter__"): body.__iter__ = types.MethodType(__iter__, body)
df = pd.read_csv(body)
df.head()
Out[8]:
Date Value
0 1/2/1986 25.56
1 1/3/1986 26.00
2 1/6/1986 26.53
3 1/7/1986 25.85

IBM Watson Studio
Search in your workspaces
Buy
Niraj TK KA's Account
Dallas
Projects / Crude Oil Price Prediction / PricePredictionModel
File Edit View Insert Cell Kernel Help
Trusted | Python 3.9
Run
Format Code
In [9]:
df.Date = df.Date.astype(str)
out = 0.17
In [10]:
df[['Month', 'Day', 'Year']] = df['Date'].str.split('/', expand=True)
In [11]:
df.drop('Date', axis=1, inplace=True)
In [12]:
df = df.dropna()
df.iloc[:, :] = df.iloc[:, :].astype(int)
df.drop_duplicates(inplace=True)
In [13]:
x = df.iloc[:, np.r_[1:4]]
y = df.iloc[:, np.r_[0]]
x_train, x_test, y_train, y_test = train_test_split(x, y, test_size=0.20)
In [14]:
for i in df.index:
df.at[i, 'Value'] = grouping(df.at[i, 'Value'])
In [15]:
w = df.iloc[:, np.r_[1:4]]
z = df.iloc[:, np.r_[0]]
w_train, w_test, z_train, z_test = train_test_split(w, z, test_size=0.20)
In [35]:
model = make_pipeline(StandardScaler(), RandomForestClassifier())
In [36]:
def rf_future(Day,Month,Year):
model.fit(x_train.values,y_train.values.ravel())
y_prediction = model.predict(x_test.values)
RFPrediction_future = model.predict([[Day,Month,Year]])
string = int(RFPrediction_future)
model.fit(w_train.values,z_train.values.ravel())
y_prediction = model.predict(w_test.values)




```
    model.fit(w_train.values,z_train.values.ravel())
    y_prediction = model.predict(w_test.values)
    RFScore_future = metrics.accuracy_score(z_test, y_prediction)


    return string,RFScore_future
```

In [17]:
```
!pip install ibm_watson_machine_learning
```

```
Requirement already satisfied: ibm_watson_machine_learning in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (1.0.257)
Requirement already satisfied: tabulate in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (0.8.9)
Requirement already satisfied: urllib3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (1.26.7)
Requirement already satisfied: importlib-metadata in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (4.8.2)
Requirement already satisfied: certifi in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2022.9.24)
Requirement already satisfied: ibm-cos-sdk==2.11.* in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2.11.0)
Requirement already satisfied: pandas<1.5.0,>=0.24.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (1.3.4)
Requirement already satisfied: lomond in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (0.3.3)
Requirement already satisfied: packaging in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (21.3)
Requirement already satisfied: requests in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm_watson_machine_learning) (2.26.0)
Requirement already satisfied: jmespath<1.0.0,>=0.7.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm_watson_machine_learning) (0.10.0)
Requirement already satisfied: ibm-cos-sdk-s3transfer==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm_watson_machine_learning) (2.11.0)
Requirement already satisfied: ibm-cos-sdk-core==2.11.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk==2.11.*->ibm_watson_machine_learning) (2.11.0)
Requirement already satisfied: python-dateutil<3.0.0,>=2.1 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from ibm-cos-sdk-core==2.11.0->ibm-cos-sdk==2.11.*->ibm_watson_machi
ne_learning) (2.8.2)
Requirement already satisfied: pytz>=2017.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas<1.5.0,>=0.24.2->ibm_watson_machine_learning) (2021.3)
Requirement already satisfied: numpy>=1.17.3 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from pandas<1.5.0,>=0.24.2->ibm_watson_machine_learning) (1.20.3)
Requirement already satisfied: six>=1.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from python-dateutil<3.0.0,>=2.1->ibm-cos-sdk-core==2.11.0->ibm-cos-sdk==2.11.*->ibm_wa
tson_machine_learning) (1.15.0)
Requirement already satisfied: charset-normalizer~=2.0.0 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests->ibm_watson_machine_learning) (2.0.4)
Requirement already satisfied: idna<4,>=2.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from requests->ibm_watson_machine_learning) (3.3)
Requirement already satisfied: zipp>=0.5 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from importlib-metadata->ibm_watson_machine_learning) (3.6.0)
Requirement already satisfied: pyparsing!=3.0.5,>=2.0.2 in /opt/conda/envs/Python-3.9/lib/python3.9/site-packages (from packaging->ibm_watson_machine_learning) (3.0.4)
```

In [18]:
```
from ibm_watson_machine_learning import APIClient
wml_credentials = {
    "url" : "https://us-south.ml.cloud.ibm.com",
    "apikey" : "H9q7MlXblpOAVJTZabR5SSGbfqKisLIRhz3h4HK_LGxW"
}

client = APIClient(wml_credentials)
```

In [24]:
```
def guid_from_space_name(client, space_name):
    space = client.spaces.get_details()
    return(next(item for item in space['resources'] if item['entity']["name"] == space_name)['metadata']['id'])
```

In [26]:
```
space_uid = guid_from_space_name(client, 'Model')
print("Space UID = " + space_uid)
```

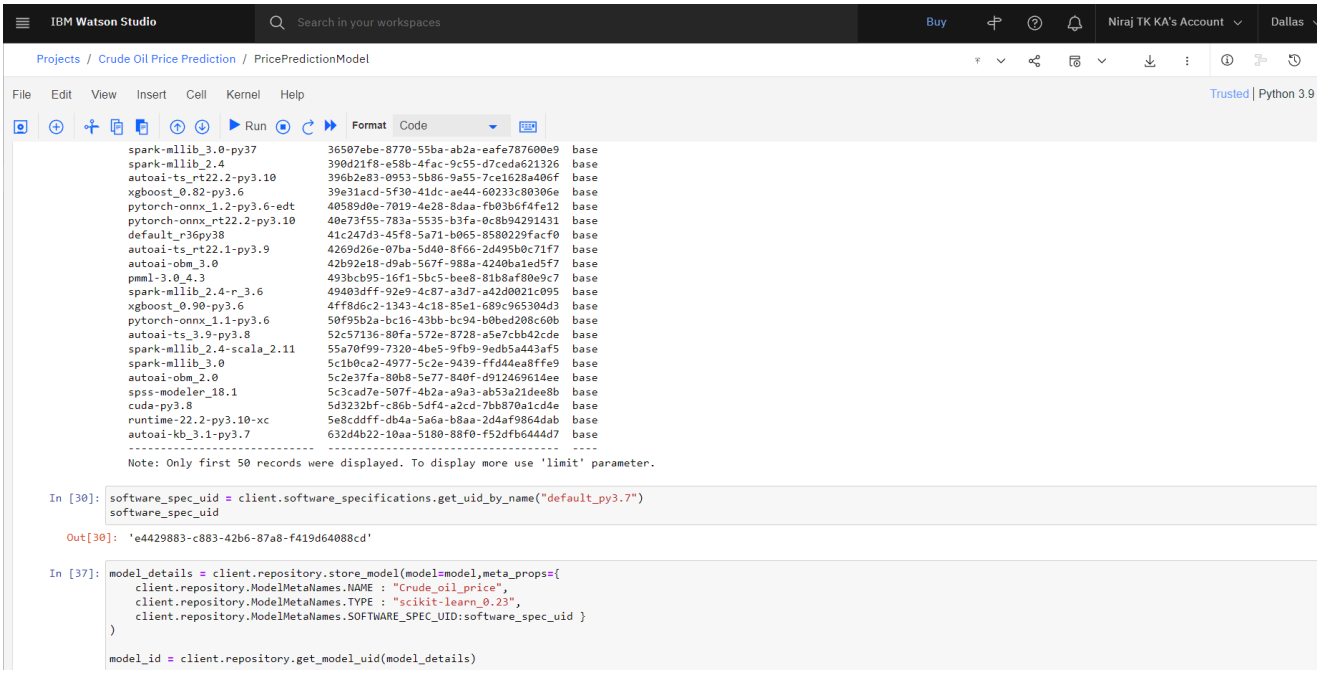
```
Space UID = cd01dfda-870a-4246-8480-3313a662f2d6
```

In [27]:
```
client.set.default_space(space_uid)
```

Out[27]: 'SUCCESS'

In [29]:
```
client.software_specifications.list()
```

```
------------------------------  ------------------------------------  ----
NAME                            ASSET_ID                              TYPE
default_py3.6                   0062b8c9-8b7d-44a0-a9b9-46c416adcbd9  base
kernel-spark3.2-scala2.12       020d69ce-7ac1-5e68-ac1a-31189867356a  base
pytorch-onnx_1.3-py3.7-edt      069ea134-3346-5748-b513-49120e15d288  base
scikit-learn_0.20-py3.6         09c5a1d0-9c1e-4473-a344-eb7b665ff687  base
spark-mllib_3.0-scala_2.12      09f4cff0-90a7-5899-b9ed-1ef348aebdee  base
pytorch-onnx_rt22.1-py3.9       0b848dd4-e681-5599-be41-b5f6fccc6471  base
ai-function_0.1-py3.6           0cdb0f1e-5376-4f4d-92dd-da3b69aa9bda  base
shiny-r3.6                      0e6e79df-875e-4f24-8ae9-62dcc2148306  base
tensorflow_2.4-py3.7-horovod    1092590a-307d-563d-9b62-4eb7d64b3f22  base
```


IBM Watson Studio
Search in your workspaces
Buy
Niraj TK KA's Account
Dallas
Projects / Crude Oil Price Prediction / PricePredictionModel
File Edit View Insert Cell Kernel Help
Trusted | Python 3.9
Run
Format
Code
spark-mllib_3.0-py37 36507ebe-8770-55ba-ab2a-eafe787600e9 base
spark-mllib_2.4 390d21f8-e58b-4fac-9c55-d7ceda621326 base
autoai-ts_rt22.2-py3.10 396b2e83-0953-5b86-9a55-7ce1628a406f base
xgboost_0.82-py3.6 39e31acd-5f30-41dc-ae44-60233c80306e base
pytorch-onnx_1.2-py3.6-edt 40589d0e-7019-4e28-8daa-fb03b6f4fe12 base
pytorch-onnx_rt22.2-py3.10 40e73f55-783a-5535-b3fa-0c8b94291431 base
default_r36py38 41c247d3-45f8-5a71-b065-8580229facf0 base
autoai-ts_rt22.1-py3.9 4269d26e-07ba-5d40-8f66-2d495b0c71f7 base
autoai-obm_3.0 42b92e18-d9ab-567f-988a-4240ba1ed5f7 base
pmml-3.0_4.3 493bcb95-16f1-5bc5-bee8-81b8af80e9c7 base
spark-mllib_2.4-r_3.6 49403dff-92e9-4c87-a3d7-a42d0021c095 base
xgboost_0.90-py3.6 4ff8d6c2-1343-4c18-85e1-689c965304d3 base
pytorch-onnx_1.1-py3.6 50f95b2a-bc16-43bb-bc94-b0bed208c60b base
autoai-ts_3.9-py3.8 52c57136-80fa-572e-8728-a5e7cbb42cde base
spark-mllib_2.4-scala_2.11 55a70f99-7320-4be5-9fb9-9edb5a443af5 base
spark-mllib_3.0 5c1b0ca2-4977-5c2e-9439-ffd44ea8ffe9 base
autoai-obm_2.0 5c2e37fa-80b8-5e77-840f-d912469614ee base
spss-modeler_18.1 5c3cad7e-507f-4b2a-a9a3-ab53a21dee8b base
cuda-py3.8 5d3232bf-c86b-5df4-a2cd-7bb870a1cd4e base
runtime-22.2-py3.10-xc 5e8cddff-db4a-5a6a-b8aa-2d4af9864dab base
autoai-kb_3.1-py3.7 632d4b22-10aa-5180-88f0-f52dfb6444d7 base
Note: Only first 50 records were displayed. To display more use 'limit' parameter.
In [30]:
software_spec_uid = client.software_specifications.get_uid_by_name("default_py3.7")
software_spec_uid
Out[30]: 'e4429883-c883-42b6-87a8-f419d64088cd'
In [37]:
model_details = client.repository.store_model(model=model,meta_props={
client.repository.ModelMetaNames.NAME : "Crude_oil_price",
client.repository.ModelMetaNames.TYPE : "scikit-learn_0.23",
client.repository.ModelMetaNames.SOFTWARE_SPEC_UID:software_spec_uid }
)
model_id = client.repository.get_model_uid(model_details)