hhyperparamater-tuning

May 28, 2020

[1]:
```
import azureml.core
from azureml.core import Workspace
```

[2]:
```
ws = Workspace.from_config()
```

```
Performing interactive authentication. Please follow the instructions on the
terminal.
To sign in, use a web browser to open the page https://microsoft.com/devicelogin
and enter the code FYS5KSSH4 to authenticate.
Interactive authentication successfully completed.
```

[3]:
```
ws.name
```

[3]:
```
'aml-ws'
```

[4]:
```
tab_data_set = ws.datasets.get('heart dataset')
```

[5]:
```
tab_data_set.to_pandas_dataframe().head(2)
```

[5]:

	age	sex	cp	trestbps	chol	fbs	restecg	thalach	exang	oldpeak	slope	ca	thal	target
0	63	1	3	145	233	1	0	150	0	2.3	0	0	1	1
1	37	1	2	130	250	0	1	187	0	3.5	0	0	2	1

[6]:
```
import os
experiment_folder = "heart_hyper_drive"
os.makedirs(experiment_folder, exist_ok=True)
print('Folder ready.')
```

```
Folder ready.
```

[7]:
```
%%writefile $experiment_folder/heart_training.py

# Import libraries
import argparse
import joblib
```

```
from azureml.core import Run
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import roc_auc_score
from sklearn.metrics import roc_curve

# Set regularization parameter
parser = argparse.ArgumentParser()
parser.add_argument('--regularization', type=float, dest='reg_rate', default=0.
 ↪01, help='regularization rate')
args = parser.parse_args()
reg = args.reg_rate

# Get the experiment run context
run = Run.get_context()

# load the diabetes dataset
print("Loading Data...")
data = run.input_datasets['heartdata'].to_pandas_dataframe() # Get the training␣
 ↪data from the estimator input

X = data.drop(['target'], axis=1)
y = data['target']

# Split data into training set and test set
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.30,␣
 ↪random_state=0)

# Train a logistic regression model
print('Training a logistic regression model with regularization rate of', reg)
run.log('Regularization Rate',  np.float(reg))
model = LogisticRegression(C=1/reg, solver="liblinear").fit(X_train, y_train)

# calculate accuracy
y_hat = model.predict(X_test)
acc = np.average(y_hat == y_test)
print('Accuracy:', acc)
run.log('Accuracy', np.float(acc))

# calculate AUC
y_scores = model.predict_proba(X_test)
auc = roc_auc_score(y_test,y_scores[:,1])
print('AUC: ' + str(auc))
run.log('AUC', np.float(auc))
```

```
os.makedirs('outputs', exist_ok=True)
# note file saved in the outputs folder is automatically uploaded into␣
 ↪experiment record
joblib.dump(value=model, filename='outputs/heart_model.pkl')

run.complete()
```

```
Overwriting heart_hyper_drive/heart_training.py
```

[8]:
```
compute_name = ws.compute_targets.get('aml-compute')
```

[10]:
```
#compute_name
```

[11]:
```
from azureml.core import Experiment
from azureml.train.sklearn import SKLearn
from azureml.train.hyperdrive import GridParameterSampling, BanditPolicy,␣
 ↪HyperDriveConfig, PrimaryMetricGoal, choice
from azureml.widgets import RunDetails


# Sample a range of parameter values
params = GridParameterSampling(
    {
        # There's only one parameter, so grid sampling will try each value -␣
 ↪with multiple parameters it would try every combination
        '--regularization': choice(0.001, 0.005, 0.01, 0.05, 0.1, 1.0)
    }
)


# Get the training dataset
tab_data_set = tab_data_set

# Create an estimator that uses the remote compute
hyper_estimator = SKLearn(source_directory=experiment_folder,
                          inputs=[tab_data_set.as_named_input('heartdata')],
                          pip_packages=['azureml-sdk'],
                          entry_script='heart_training.py',
                          compute_target = compute_name,)

# Configure hyperdrive settings
hyperdrive = HyperDriveConfig(estimator=hyper_estimator,
                          hyperparameter_sampling=params,
                          policy=None,
                          primary_metric_name='AUC',
                          primary_metric_goal=PrimaryMetricGoal.MAXIMIZE,
                          max_total_runs=6,
                          max_concurrent_runs=4)
```

```
# Run the experiment
experiment = Experiment(workspace = ws, name = 'heart_training_hyperdrive')
run = experiment.submit(config=hyperdrive)

# Show the status in the notebook as the experiment runs
RunDetails(run).show()
run.wait_for_completion()
```

```
_HyperDriveWidget(widget_settings={'childWidgetDisplay': 'popup', 'send_telemetry': False, 'log_
```

[11]:
```
{'runId': 'HD_114a6592-8312-4c2b-bd09-f2914613799e',
 'target': 'aml-compute',
 'status': 'Completed',
 'startTimeUtc': '2020-05-25T11:49:58.617334Z',
 'endTimeUtc': '2020-05-25T12:02:39.392711Z',
 'properties': {'primary_metric_config': '{"name": "AUC", "goal": "maximize"}',
  'resume_from': 'null',
  'runTemplate': 'HyperDrive',
  'azureml.runsource': 'hyperdrive',
  'platform': 'AML',
  'ContentSnapshotId': '286709c2-b803-4929-8b88-de2e6070cb14',
  'score': '0.8820116054158608',
  'best_child_run_id': 'HD_114a6592-8312-4c2b-bd09-f2914613799e_0',
  'best_metric_status': 'Succeeded'},
 'inputDatasets': [],
 'logFiles': {'azureml-logs/hyperdrive.txt': 'https://amlws3499330668.blob.core.
windows.net/azureml/ExperimentRun/dcid.HD_114a6592-8312-4c2b-bd09-f2914613799e
/azureml-logs/hyperdrive.txt?sv=2019-02-02&sr=b&sig=M6viiZN7%2FVxVGgHtdPUvQn%2BU
ButTN7U%2F6p1J2KWdaHg%3D&st=2020-05-25T11%3A52%3A40Z&se=2020-05-25T20%3A02%3A40Z
&sp=r'}}
```

[12]:
```
for child_run in run.get_children_sorted_by_primary_metric():
    print(child_run)

best_run = run.get_best_run_by_primary_metric()
best_run_metrics = best_run.get_metrics()
parameter_values = best_run.get_details() ['runDefinition']['arguments']

print('Best Run Id: ', best_run.id)
print(' -AUC:', best_run_metrics['AUC'])
print(' -Accuracy:', best_run_metrics['Accuracy'])
print(' -Regularization Rate:',parameter_values)
```

```
{'run_id': 'HD_114a6592-8312-4c2b-bd09-f2914613799e_0', 'hyperparameters': '{"--
regularization": 0.001}', 'best_primary_metric': 0.8820116054158608, 'status':
'Completed'}
```

```
{'run_id': 'HD_114a6592-8312-4c2b-bd09-f2914613799e_5', 'hyperparameters': '{"--
regularization": 1.0}', 'best_primary_metric': 0.8805609284332689, 'status':
'Completed'}
{'run_id': 'HD_114a6592-8312-4c2b-bd09-f2914613799e_4', 'hyperparameters': '{"--
regularization": 0.1}', 'best_primary_metric': 0.8786266924564797, 'status':
'Completed'}
{'run_id': 'HD_114a6592-8312-4c2b-bd09-f2914613799e_3', 'hyperparameters': '{"--
regularization": 0.05}', 'best_primary_metric': 0.8786266924564797, 'status':
'Completed'}
{'run_id': 'HD_114a6592-8312-4c2b-bd09-f2914613799e_2', 'hyperparameters': '{"--
regularization": 0.01}', 'best_primary_metric': 0.8786266924564797, 'status':
'Completed'}
{'run_id': 'HD_114a6592-8312-4c2b-bd09-f2914613799e_1', 'hyperparameters': '{"--
regularization": 0.005}', 'best_primary_metric': 0.8781431334622823, 'status':
'Completed'}
{'run_id': 'HD_114a6592-8312-4c2b-bd09-f2914613799e_preparation',
'hyperparameters': None, 'best_primary_metric': None, 'status': 'Completed'}
Best Run Id:  HD_114a6592-8312-4c2b-bd09-f2914613799e_0
 -AUC: 0.8820116054158608
 -Accuracy: 0.8131868131868132
 -Regularization Rate: ['--regularization', '0.001']
```

[14]:
```
from azureml.core import Model

# Register model
best_run.register_model(model_path='outputs/heart_model.pkl',␣
 ↪model_name='heart_model',
                        tags={'Training context':'Hyperdrive'},
                        properties={'AUC': best_run_metrics['AUC'], 'Accuracy':␣
 ↪best_run_metrics['Accuracy']})

# List registered models
for model in Model.list(ws):
    print(model.name, 'version:', model.version)
    for tag_name in model.tags:
        tag = model.tags[tag_name]
        print ('\t',tag_name, ':', tag)
    for prop_name in model.properties:
        prop = model.properties[prop_name]
        print ('\t',prop_name, ':', prop)
    print('\n')
```

```
heart_model version: 5
         Training context : Hyperdrive
         AUC : 0.8820116054158608
         Accuracy : 0.8131868131868132
```

```
heart_model version: 4
         training context : pipeline


heart_model version: 3
         Training context : SKLearn Estimator (tabular dataset)
         AUC : 0.8771760154738878
         Accuracy : 0.8131868131868132


heart_model version: 2
         Training context : SKLearn Estimator (tabular dataset)
         AUC : 0.8786266924564797
         Accuracy : 0.8131868131868132


heart_model version: 1
         Training context : SKLearn Estimator (tabular dataset)
         AUC : 0.8786266924564797
         Accuracy : 0.8131868131868132


heart-model version: 1
```

[ ]:

[ ]: