IPA 주관 인공지능센터 기본(fundamental) 과정

GitHub link: here

E-Mail: windkyle7@gmail.com

KerasClassifier

scikit-learn의 단점은 GPU를 지원하지 않는다는 것인데 keras에서 GPU를 지원함으로써 이와 같은 단점을 극복할 수 있다.

In [1]:

```
import tensorflow as tf
tf.__version__
```

Out[1]:

```
'2.0.0-beta1'
```

In [2]:

```
def createModel():
    model = tf.keras.Sequential([
        tf.keras.layers.Dense(16, activation='relu', input_shape=(4,)),
        tf.keras.layers.Dense(3, activation='softmax')
    ])
    model.compile(
        optimizer='adam',
        loss='categorical_crossentropy',
        metrics=['acc']
    )
    return model
```

In [3]:

```
import seaborn as sns
iris = sns.load_dataset('iris')
```

In [4]:

```
import pandas as pd
from tensorflow.keras.wrappers.scikit_learn import KerasClassifier

kc = KerasClassifier(createModel)
kc.fit(iris.iloc[:, :-1], pd.get_dummies(iris.iloc[:, -1]))
```

```
WARNING: Logging before flag parsing goes to stderr.
W0628 21:07:59.057663 140620569773888 deprecation.py:323] From
/home/user/workspace/.venv/lib/python3.6/site-
packages/tensorflow/python/ops/math_grad.py:1250: add_dispatch_support.<loc
als>.wrapper (from tensorflow.python.ops.array_ops) is deprecated and will
be removed in a future version.
Instructions for updating:
Use tf.where in 2.0, which has the same broadcast rule as np.where
Use tf.where in 2.0, which has the same broadcast rule as np.where
```

```
Train on 150 samples
150/150 [==============================] - 0s 726us/sample - loss: 1.7325 -
acc: 0.3333
```

Out[4]:

```
<tensorflow.python.keras.callbacks.History at 0x7fe4669e4cc0>
```

In [5]:

```
from sklearn.model_selection import train_test_split

iris = sns.load_dataset('iris')
iris.species = iris.species.map({'setosa': 0, 'versicolor': 1, 'virginica':
2})
X_train, X_test, y_train, y_test = train_test_split(iris.iloc[:, :-1],
                                                    iris.iloc[:, -1])
```

TPOT

Consider TPOT your Data Science Assistant. TPOT is a Python Automated Machine Learning tool that optimizes machine learning pipelines using genetic programming.

In [6]:

```
from tpot import TPOTClassifier

tp = TPOTClassifier(10, 10)
tp.fit(X_train, y_train)
```

```
/home/user/workspace/.venv/lib/python3.6/site-
packages/sklearn/externals/joblib/__init__.py:15: DeprecationWarning: sklea
rn.externals.joblib is deprecated in 0.21 and will be removed in 0.23. Plea
se import this functionality directly from joblib, which can be installed w
ith: pip install joblib. If this warning is raised when loading pickled mod
els, you may need to re-serialize those models with scikit-learn 0.21+.
  warnings.warn(msg, category=DeprecationWarning)
```

Out[6]:

```
TPOTClassifier(config_dict=None, crossover_rate=0.1, cv=5,
               disable_update_check=False, early_stop=None, generations=10,
               max_eval_time_mins=5, max_time_mins=None, memory=None,
               mutation_rate=0.9, n_jobs=1, offspring_size=None,
               periodic_checkpoint_folder=None, population_size=10,
               random_state=None, scoring=None, subsample=1.0,
               template='RandomTree', use_dask=False, verbosity=0,
               warm_start=False)
```

In [7]:

```
tp.score(X_test, y_test)
```

Out[7]:

```
0.9736842105263158
```

In [8]:

```
tp.export('model.py')
tp.export('model.py')
with open('model.py', 'r') as sourcefile:
    print(sourcefile.read())
```

```
import numpy as np
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.naive_bayes import MultinomialNB
from sklearn.pipeline import make_pipeline
from sklearn.preprocessing import PolynomialFeatures

# NOTE: Make sure that the class is labeled 'target' in the data file
tpot_data = pd.read_csv('PATH/TO/DATA/FILE', sep='COLUMN_SEPARATOR', dtype=
np.float64)
features = tpot_data.drop('target', axis=1).values
training_features, testing_features, training_target, testing_target = \
            train_test_split(features, tpot_data['target'].values,
random_state=None)

# Average CV score on the training set was:0.9730848861283643
exported_pipeline = make_pipeline(
    PolynomialFeatures(degree=2, include_bias=False,
interaction_only=False),
    MultinomialNB(alpha=0.001, fit_prior=False)
)

exported_pipeline.fit(training_features, training_target)
results = exported_pipeline.predict(testing_features)
```