Omdena Liverpool Chapter

Detecting Pediatric Acute Lymphoblastic Leukemia using Computer Vision

Damir Zunic - Contribution & Experience

May 14, 2023

Contribution

Traditional Machine Learning classification using features extracted by Transfer Learning (batch_4 dataset)

- XGBoost + Optuna
- Stacking Classifier
  - *Final estimator* - XGBoost
  - *Base estimators* - SVM + RF + LR + MLP + Extra Trees

- Feature Selection
  - ANOVA + XGBoost + Optuna
- Neural Network
  - SciKeras + PCA + Optuna

Skills Learned #1

- Optuna for Hyperparameter Tuning
  - Importance of parameter spaces
  - Calling *optimize()* function again so that it tries more trials to improve results further
- Feature Selection Using ANOVA
  - Scikit-learn SelectKBest(score_func=f_classif)
  - *k* - number of top features to select as hyperparameter

Skills Learned #2

- Stacking Classifier
  - slightly improved model performance - not enough
  - longer to train and slower predictions
  - computationally expensive
- SciKeras
  - Scikit-Learn API wrapper for Keras
  - last year introduced to SciKeras
  - Keras model optimization with Optuna

Challenges / Obstacles #1

SciKeras Hidden Layers for Optuna - [a]

Keras Classifier:

```
clf = KerasClassifier(
    model=km.get_clf,
    loss="binary_crossentropy",
    model__hidden_layer_sizes=(10, 10),
    model__dropout=0.5,
    batch_size=64,
        ...
    random_state=random_state)
```

Challenges / Obstacles #1

SciKeras Hidden Layers for Optuna - [b]

Optuna Objective Function:

```
n_layers = trial.suggest_int('n_layers', 1, 2)          # no. of hidden layers
layers = [300]        # max nodes in layers
for i in range(n_layers):
    layers.append(trial.suggest_int(f"n_units_{i+1}", 20, layers[i], 20))   # no. of hidden
units
   ...

 params['clf__model__hidden_layer_sizes'] = tuple(layers[1:])
```

Challenges / Obstacles #1

SciKeras Hidden Layers for Optuna - [c]

```
params_t = {'n_layers': 2,
            'n_units_1': 260, 'n_units_2': 120,
            'clf__epochs': 120, ...}
```

```
params = {k:v for k,v in params_t.items() if '__' in k}

params['clf__model__hidden_layer_sizes'] = tuple([v for k,v in params_t.items() if 'n_units' in k])
```

⬇

```
params = {'clf__epochs': 120,
          ... ,
          'clf__model__hidden_layer_sizes': (260, 120)}
```

Challenges / Obstacles #2

SciKeras Save & Load Pipelined Model - [a]

```
my_model = Pipeline(
    steps = (
        ('scaler', MinMaxScaler()),
        ('pca', PCA(num_pca_components)),
        ('clf', clf)
    ))
```

❖ Requires saving separately the Keras model and the pipeline

Challenges / Obstacles #2

SciKeras Save & Load Pipelined Model - [b]

SAVE

```
my_model.named_steps[‘clf’].model_.save(path_keras)           # Save the Keras model first
my_model.named_steps[‘clf’].model_ = None          # This hack allows to save the pipeline
joblib.dump(my_model, path_pipe)                                       # Save the pipeline
```

LOAD

```
model_l = joblib.load(path_pipe)                                 # Load the pipeline first
model_l.named_steps[‘clf’].model_ = load_model(path_keras)     # Then, load the Keras model
```

Overall Experience

- 1st Omdena project
- Liked the teamwork
- Enjoyed working with the team!!