Section 2
Supervised learning: hyperparam. tuning & perf. assessment

Universitat de les Illes Balears
Departament de Ciències Matemàtiques i Informàtica

11752 Aprendizaje Automático
Máster Universitario
en Sistemas Inteligentes

Alberto ORTIZ RODRÍGUEZ

Hyperparam. tuning & perform. assessment

- Hyperparameters tuning & performance assessment with **scikit-learn**:

```
from sklearn.preprocessing import MinMaxScaler
from sklearn.model_selection import GridSearchCV
from sklearn.model_selection import cross_validate
from sklearn.pipeline import make_pipeline
from sklearn.metrics import classification_report
from sklearn.metrics import accuracy_score
from sklearn.metrics import precision_score
from sklearn.metrics import f1_score
from sklearn.metrics import recall_score
```

Hyperparam. tuning & perform. assessment

- Hyperparameter tuning by **grid search** (only main parameters considered):

```
# model is the classifier, whose parameters have to be tuned
gsclf = GridSearchCV(estimator=model,cv=n_folds,param_grid=parameters,scoring=score)
cv: cross-validation splitting strategy, e.g. 3
param_grid: dictionary or lsit of dictionaries
scoring: string or tuple of strings referring to implemented metrics
         'accuracy', 'f1', 'precision', 'recall', …
```

```
model = RandomForestClassifier(n_estimators=10)
parameters = {
    'max_depth': [3, None],
    'max_features': randint(1, 11),
    'min_samples_split': randint(2, 11),
    'bootstrap': [True, False],
    'criterion': ['gini', 'entropy'],
}
gsclf = GridSearchCV(estimator=model, cv=4,
                     param_grid=parameters, scoring='recall')
gsclf.fit(X_train, y_train)
# gsclf.best_estimator_, gsclf.best_params_, gsclf.best_score_, gsclf.predict, etc.
```

https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.GridSearchCV.html
https://scikit-learn.org/stable/modules/model_evaluation.html

Data preprocessing

- Data **preprocessing**:

```
 scaler = MinMaxScaler()
 X_train_ = scaler.fit_transform(X_train)
 X_test_ = scaler.transform(X_test)

# other data transformations: StandardScaler
```

https://scikit-learn.org/stable/modules/classes.html#module-sklearn.preprocessing

Performance assessment

- Performance assessment by **cross-validation**:

```
y_pred = clf_.predict(X_test_)
classification_report(y_test,  y_pred)
test_accuracy = accuracy_score(y_test, y_pred)
test_precision = precision_score(y_test, y_pred)
test_recall = recall_score(y_test, y_pred)
test_f1 = f1_score(y_test, y_pred)

clf = clf_.best_estimator_
clfp = make_pipeline(StandardScaler(), clf)
scoring = ['accuracy']
scores = cross_validate(clfp, X, y, cv=5, scoring=scoring)
print('accuracy: ', scores['test_accuracy'])
print('avg acc.: ', np.mean(scores['test_accuracy']))
```

https://scikit-learn.org/stable/modules/cross_validation.html#cross-validation