Session 3

In this third session we will apply logistic regression to some classification tasks. This notebook describes how to do it in Iris with the sklearn implementation of logistic regression. The main objective of this session is to extend the example given to other tasks, trying to minimize test error.

Logistic regression applied to Iris

Reading and partitioning the dataset:

In [1]:
```
import numpy as np; from sklearn.datasets import load_iris
from sklearn.model_selection import train_test_split
iris = load_iris(); X = iris.data.astype(np.float16); y = iris.target.astype(np.uint)
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, shuffle=True, random_state=23)
```

LogisticRegression: implementation of logistic regression in sklearn

In [2]:
```
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
clf = LogisticRegression(random_state=23).fit(X_train, y_train)
y_test_pred = clf.predict(X_test)
err_test = 1 - accuracy_score(y_test, clf.predict(X_test))
print(f"Test error: {err_test:.1%}")
```

```
Test error: 0.0%
```

Warnings: sklearn is a bit "picky" with warnings; we will ignore the warnings about convergence

In [3]:
```
import warnings; from sklearn.exceptions import ConvergenceWarning
warnings.filterwarnings("ignore", category=ConvergenceWarning, module="sklearn")
```

Solvers: the `solver` parameter of LogisticRegression allows you to choose between different solvers (optimization algorithms)

In [4]:
```
for solver in ['lbfgs', 'liblinear', 'newton-cg', 'newton-cholesky', 'sag', 'saga']:
    clf = LogisticRegression(random_state=23, solver=solver, max_iter=10000).fit(X_train, y_train)
    err_test = 1 - accuracy_score(y_test, clf.predict(X_test))
    print(f"Test error after training with the solver {solver!s}: {err_test:.1%}")
```

```
Test error after training with the solver lbfgs: 0.0%
Test error after training with the solver liblinear: 3.3%
Test error after training with the solver newton-cg: 0.0%
Test error after training with the solver newton-cholesky: 3.3%
Test error after training with the solver sag: 0.0%
Test error after training with the solver saga: 0.0%
```

Tolerance: the `tol` parameter sets a tolerance threshold to end training (1e4 by default)

In [5]:
```
for tol in (1e-4, 1e-2, 1, 1e2, 1e4):
    clf = LogisticRegression(tol=tol, random_state=23, max_iter=10000).fit(X_train, y_train)
    err_test = 1 - accuracy_score(y_test, clf.predict(X_test))
    print(f"Test error with tolerance {tol}: {err_test:.1%}")
```

```
Test error with tolerance 0.0001: 0.0%
Test error with tolerance 0.01: 3.3%
Test error with tolerance 1: 60.0%
Test error with tolerance 100.0: 60.0%
Test error with tolerance 10000.0: 60.0%
```

Regularization: parameter `C` (positive, 1.0 by default) de-regularizes the training criterion

- **Possibility of under-adjustment:** with a value close to zero (maximum regularization)
- **Possibility of overfitting:** with a very high positive value (minimum regularization)

In [6]:
```
for C in (1e-2, 1e-1, 1, 1e1, 1e2):
    clf = LogisticRegression(C=C, random_state=23, max_iter=10000).fit(X_train, y_train)
    err_test = 1 - accuracy_score(y_test, clf.predict(X_test))
    print(f"Test error with C {C:g}: {err_test:.1%}")
```

```
Test error with C 0.01: 6.7%
Test error with C 0.1: 3.3%
Test error with C 1: 0.0%
Test error with C 10: 3.3%
Test error with C 100: 3.3%
```

Early stopping: saving computation and avoiding over-training ("regularize") by finishing earlier (in a few iterations)

In [7]:
```
for max_iter in (10, 20, 50, 100):
    clf = LogisticRegression(random_state=23, max_iter=max_iter).fit(X_train, y_train)
    err_test = 1 - accuracy_score(y_test, clf.predict(X_test))
    print(f"Test error with max_iter {max_iter}: {err_test:.1%}")
```

```
Test error with max_iter 10: 0.0%
Test error with max_iter 20: 3.3%
Test error with max_iter 50: 0.0%
Test error with max_iter 100: 0.0%
```

Logistic regression applied to openml

In [1]:
```
import warnings; warnings.filterwarnings("ignore"); import numpy as np
from sklearn.datasets import fetch_openml
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.metrics import accuracy_score
```

In [2]:
```
def err_eval(data_id):
    X, y = fetch_openml(data_id=data_id, return_X_y=True, as_frame=False, parser="liac-arff")
    mask = ~np.isnan(X).any(axis=1); X = X[mask, :]; y = y[mask]
    if X.shape[0] < 10: return(1.0)
    X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, shuffle=True, random_state=23)
    clf = LogisticRegression(random_state=23).fit(X_train, y_train)
    return(1 - accuracy_score(y_test, clf.predict(X_test)))
```

In [3]:
```python
import openml
for sid in (99, 334):
    benchmark_suite = openml.study.get_suite(suite_id=sid)
    df = openml.datasets.list_datasets(data_id=benchmark_suite.data, output_format='dataframe')
    for did, name, C in zip(df['did'], df['name'], df['NumberOfClasses']):
        err = err_eval(did)
        print(f"sid: {sid:5d}  did: {did:5d}  C: {C:5.0f}  err: {err:7.1%}  name: {name:s}")
```

```
sid:    99  did:     3  C:     2  err:    5.5%  name: kr-vs-kp
sid:    99  did:     6  C:    26  err:   23.7%  name: letter
sid:    99  did:    11  C:     3  err:    8.0%  name: balance-scale
sid:    99  did:    12  C:    10  err:    4.2%  name: mfeat-factors
sid:    99  did:    14  C:    10  err:   21.0%  name: mfeat-fourier
sid:    99  did:    15  C:     2  err:    2.2%  name: breast-w
sid:    99  did:    16  C:    10  err:    4.5%  name: mfeat-karhunen
sid:    99  did:    18  C:    10  err:   46.8%  name: mfeat-morphological
sid:    99  did:    22  C:    10  err:   18.2%  name: mfeat-zernike
sid:    99  did:    23  C:     3  err:   50.8%  name: cmc
sid:    99  did:    28  C:    10  err:    3.3%  name: optdigits
sid:    99  did:    29  C:     2  err:   13.7%  name: credit-approval
sid:    99  did:    31  C:     2  err:   25.5%  name: credit-g
sid:    99  did:    32  C:    10  err:    6.0%  name: pendigits
sid:    99  did:    37  C:     2  err:   16.9%  name: diabetes
sid:    99  did:    38  C:     2  err:  100.0%  name: sick
sid:    99  did:    44  C:     2  err:    7.4%  name: spambase
sid:    99  did:    46  C:     3  err:    6.7%  name: splice
sid:    99  did:    50  C:     2  err:   29.2%  name: tic-tac-toe
sid:    99  did:    54  C:     4  err:   29.4%  name: vehicle
sid:    99  did:   151  C:     2  err:   24.9%  name: electricity
sid:    99  did:   182  C:     6  err:   14.2%  name: satimage
sid:    99  did:   188  C:     5  err:   60.5%  name: eucalyptus
sid:    99  did:   300  C:    26  err:    4.5%  name: isolet
sid:    99  did:   307  C:    11  err:   39.4%  name: vowel
sid:    99  did:   458  C:     4  err:    1.2%  name: analcatdata_authorship
sid:    99  did:   469  C:     6  err:   81.2%  name: analcatdata_dmft
sid:    99  did:   554  C:    10  err:    8.0%  name: mnist_784
sid:    99  did:  1049  C:     2  err:   11.6%  name: pc4
sid:    99  did:  1050  C:     2  err:    8.9%  name: pc3
sid:    99  did:  1053  C:     2  err:   19.5%  name: jm1
sid:    99  did:  1063  C:     2  err:   13.3%  name: kc2
sid:    99  did:  1067  C:     2  err:   16.8%  name: kc1
sid:    99  did:  1068  C:     2  err:    5.9%  name: pc1
sid:    99  did:  1461  C:     2  err:   11.5%  name: bank-marketing
sid:    99  did:  1462  C:     2  err:    1.5%  name: banknote-authentication
sid:    99  did:  1464  C:     2  err:   27.3%  name: blood-transfusion-service-center
sid:    99  did:  1468  C:     9  err:    6.0%  name: cnae-9
sid:    99  did:  1475  C:     6  err:   53.7%  name: first-order-theorem-proving
sid:    99  did:  1478  C:     6  err:    1.8%  name: har
sid:    99  did:  1480  C:     2  err:   29.9%  name: ilpd
sid:    99  did:  1485  C:     2  err:   43.7%  name: madelon
sid:    99  did:  1486  C:     2  err:    5.2%  name: nomao
sid:    99  did:  1487  C:     2  err:    5.3%  name: ozone-level-8hr
sid:    99  did:  1489  C:     2  err:   25.3%  name: phoneme
```

```
sid:    99  did:  1494  C:     2  err:   16.6%  name: qsar-biodeg
sid:    99  did:  1497  C:     4  err:   29.6%  name: wall-robot-navigation
sid:    99  did:  1501  C:    10  err:    9.4%  name: semeion
sid:    99  did:  1510  C:     2  err:    3.5%  name: wdbc
sid:    99  did:  1590  C:     2  err:   20.8%  name: adult
sid:    99  did:  4134  C:     2  err:   24.0%  name: Bioresponse
sid:    99  did:  4534  C:     2  err:   10.2%  name: PhishingWebsites
sid:    99  did:  4538  C:     5  err:   54.6%  name: GesturePhaseSegmentationProcessed
sid:    99  did:  6332  C:     2  err:   28.6%  name: cylinder-bands
sid:    99  did: 23381  C:     2  err:   50.0%  name: dresses-sales
sid:    99  did: 23517  C:     2  err:   47.4%  name: numerai28.6
sid:    99  did: 40499  C:    11  err:    2.1%  name: texture
sid:    99  did: 40668  C:     3  err:   33.4%  name: connect-4
sid:    99  did: 40670  C:     3  err:    5.2%  name: dna
sid:    99  did: 40701  C:     2  err:   13.4%  name: churn
sid:    99  did: 40923  C:    46  err:   27.7%  name: Devnagari-Script
sid:    99  did: 40927  C:    10  err:   58.9%  name: CIFAR_10
sid:    99  did: 40966  C:     8  err:    6.3%  name: MiceProtein
sid:    99  did: 40975  C:     4  err:   15.6%  name: car
sid:    99  did: 40978  C:     2  err:    3.2%  name: Internet-Advertisements
sid:    99  did: 40979  C:    10  err:    3.2%  name: mfeat-pixel
sid:    99  did: 40982  C:     7  err:   58.4%  name: steel-plates-fault
sid:    99  did: 40983  C:     2  err:    3.5%  name: wilt
sid:    99  did: 40984  C:     7  err:    7.8%  name: segment
sid:    99  did: 40994  C:     2  err:    5.6%  name: climate-model-simulation-crashes
sid:    99  did: 40996  C:    10  err:   14.7%  name: Fashion-MNIST
sid:    99  did: 41027  C:     3  err:   31.8%  name: jungle_chess_2pcs_raw_endgame_complete
sid:   334  did: 44156  C:     2  err:   26.6%  name: electricity
sid:   334  did: 44157  C:     2  err:   47.9%  name: eye_movements
sid:   334  did: 44159  C:     2  err:   24.5%  name: covertype
sid:   334  did: 45035  C:     2  err:   36.7%  name: albert
sid:   334  did: 45036  C:     2  err:   37.7%  name: default-of-credit-card-clients
sid:   334  did: 45038  C:     2  err:   42.0%  name: road-safety
sid:   334  did: 45039  C:     2  err:   30.2%  name: compas-two-years
```

```
In [4]: import openml
        for sid in (271, ):
            benchmark_suite = openml.study.get_suite(suite_id=sid)
            df = openml.datasets.list_datasets(data_id=benchmark_suite.data, output_format='dataframe')
            for did, name, C in zip(df['did'], df['name'], df['NumberOfClasses']):
                if did == 41147: continue;
                err = err_eval(did)
                print(f"sid: {sid:5d}  did: {did:5d}  C: {C:5.0f}  err: {err:7.1%}  name: {name:s}")
```

```
sid:   271  did:     3  C:     2  err:    5.5%  name: kr-vs-kp
sid:   271  did:    12  C:    10  err:    4.2%  name: mfeat-factors
sid:   271  did:    23  C:     3  err:   50.8%  name: cmc
sid:   271  did:    31  C:     2  err:   25.5%  name: credit-g
sid:   271  did:    54  C:     4  err:   29.4%  name: vehicle
sid:   271  did:   181  C:    10  err:   45.1%  name: yeast
sid:   271  did:   188  C:     5  err:   60.5%  name: eucalyptus
sid:   271  did:  1049  C:     2  err:   11.6%  name: pc4
sid:   271  did:  1067  C:     2  err:   16.8%  name: kc1
sid:   271  did:  1111  C:     2  err:  100.0%  name: KDDCup09_appetency
sid:   271  did:  1169  C:     2  err:   42.1%  name: airlines
sid:   271  did:  1457  C:    50  err:   37.3%  name: amazon-commerce-reviews
sid:   271  did:  1461  C:     2  err:   11.5%  name: bank-marketing
sid:   271  did:  1464  C:     2  err:   27.3%  name: blood-transfusion-service-center
sid:   271  did:  1468  C:     9  err:    6.0%  name: cnae-9
sid:   271  did:  1475  C:     6  err:   53.7%  name: first-order-theorem-proving
sid:   271  did:  1486  C:     2  err:    5.2%  name: nomao
sid:   271  did:  1487  C:     2  err:    5.3%  name: ozone-level-8hr
sid:   271  did:  1489  C:     2  err:   25.3%  name: phoneme
sid:   271  did:  1494  C:     2  err:   16.6%  name: qsar-biodeg
sid:   271  did:  1515  C:    20  err:   13.9%  name: micro-mass
sid:   271  did:  1590  C:     2  err:   20.8%  name: adult
sid:   271  did:  1596  C:     7  err:   38.3%  name: covertype
sid:   271  did:  4134  C:     2  err:   24.0%  name: Bioresponse
sid:   271  did:  4135  C:     2  err:    5.6%  name: Amazon_employee_access
sid:   271  did:  4534  C:     2  err:   10.2%  name: PhishingWebsites
sid:   271  did:  4538  C:     5  err:   54.6%  name: GesturePhaseSegmentationProcessed
sid:   271  did:  4541  C:     3  err:   46.0%  name: Diabetes130US
sid:   271  did: 23517  C:     2  err:   47.4%  name: numerai28.6
sid:   271  did: 40498  C:     7  err:   54.4%  name: wine-quality-white
sid:   271  did: 40668  C:     3  err:   33.4%  name: connect-4
sid:   271  did: 40670  C:     3  err:    5.2%  name: dna
sid:   271  did: 40685  C:     7  err:    7.7%  name: shuttle
sid:   271  did: 40701  C:     2  err:   13.4%  name: churn
sid:   271  did: 40900  C:     2  err:    0.8%  name: Satellite
sid:   271  did: 40975  C:     4  err:   15.6%  name: car
sid:   271  did: 40978  C:     2  err:    3.2%  name: Internet-Advertisements
sid:   271  did: 40981  C:     2  err:   16.7%  name: Australian
sid:   271  did: 40982  C:     7  err:   58.4%  name: steel-plates-fault
sid:   271  did: 40983  C:     2  err:    3.5%  name: wilt
sid:   271  did: 40984  C:     7  err:    7.8%  name: segment
sid:   271  did: 40996  C:    10  err:   14.7%  name: Fashion-MNIST
sid:   271  did: 41027  C:     3  err:   31.8%  name: jungle_chess_2pcs_raw_endgame_complete
sid:   271  did: 41138  C:     2  err:    2.6%  name: APSFailure
sid:   271  did: 41142  C:     2  err:   33.2%  name: christine
```

```
sid:   271  did: 41143  C:     2  err:   19.8%  name: jasmine
sid:   271  did: 41144  C:     2  err:   45.5%  name: madeline
sid:   271  did: 41145  C:     2  err:   31.2%  name: philippine
sid:   271  did: 41146  C:     2  err:   14.0%  name: sylvine
sid:   271  did: 41150  C:     2  err:   14.3%  name: MiniBooNE
sid:   271  did: 41156  C:     2  err:   17.1%  name: ada
sid:   271  did: 41157  C:     2  err:    5.0%  name: arcene
sid:   271  did: 41158  C:     2  err:   17.4%  name: gina
sid:   271  did: 41159  C:     2  err:   33.3%  name: guillermo
sid:   271  did: 41161  C:     2  err:    1.1%  name: riccardo
sid:   271  did: 41162  C:     2  err:    5.0%  name: kick
sid:   271  did: 41163  C:     5  err:    8.3%  name: dilbert
sid:   271  did: 41164  C:     7  err:   31.6%  name: fabert
sid:   271  did: 41165  C:    10  err:   63.2%  name: robert
sid:   271  did: 41166  C:    10  err:   46.9%  name: volkert
sid:   271  did: 41167  C:   355  err:   98.7%  name: dionis
sid:   271  did: 41168  C:     4  err:   39.6%  name: jannis
sid:   271  did: 41169  C:   100  err:   76.3%  name: helena
sid:   271  did: 42732  C:     2  err:   12.2%  name: sf-police-incidents
sid:   271  did: 42733  C:     2  err:   16.5%  name: Click_prediction_small
sid:   271  did: 42734  C:     3  err:   26.5%  name: okcupid-stem
sid:   271  did: 42742  C:     2  err:    4.6%  name: porto-seguro
sid:   271  did: 42746  C:    23  err:    2.9%  name: KDDCup99
sid:   271  did: 42769  C:     2  err:   35.9%  name: Higgs
sid:   271  did: 43072  C:     2  err:  100.0%  name: KDDCup09-Upselling
```