Logistic regression

We study a dataset coming from the UCI Machine Learning repository, related to direct marketing campaigns (phone calls) of a Portuguese banking institution. The classification goal is to predict whether the client will subscribe (1/0) to a term deposit (variable y).

0.1 Importation of data and preprocessing

1. Import the following librairies
   `import pandas as pd`
   `import numpy as np`

2. Import the data
   `data=pd.read_csv('your_own_path/banking.csv')`
   `data=data.dropna()`

3. Look at the dataset using `data.head()`. What are the input variables? What is the output variable?

4. Which variables are categorical? Remove them.

0.2 Logistic Regression

Import the package `sklearn.linear_model` which contains the class `LogisticRegression` `from sklearn import linear_model`.
We test logistic regression on the dataset `banking.csv` preprocessed in the last subsection.

1. Create a model `LogisticRegression` : `my log = linear_model.LogisticRegression()`

2. To learn the model from the data `dataX` and their corresponding labels `dataY`, one can use `fit` : `my_log.fit(data_X,dataY)`
   Fit the model on the dataset `banking.csv` defining `data_X` and `data_Y` as follows
   `data_X=data.iloc[:,0:10]`
   `data_Y=data['y']`

3. What is the variable `coef_` of the model ? `intercep_` ?

4. What is the output of the function `predict` ? of the function `score` ?

0.3 Evaluation of logistic regression with `scikit-learn`

0.3.1 Hold out evaluation

In this method, the dataset is randomly divided into two subsets:

- Training set is a subset of the dataset used to build the two predictive models.
- Test set, or unseen data, is a subset of the dataset used to assess the likely future performance of a model. If a model fits to the training set much better than it fits the test set, overfitting is probably the cause.

1. Import the function `train_test_split`
   `from sklearn.model_selection import train_test_split`
2. Divide the dataset into train and test. Some indications on the function `train_test_split` are given in the website
   `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.train_test_split.html`
3. Learn the two models on the train set.
4. Compare then the performance of Logistic regression and Perceptron on the test set using as metric : the accuracy, and then the $F1$ score :
   `from sklearn.metrics import accuracy_score`
   `from sklearn.metrics import f1_score`
   See `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html` for more information.

0.3.2 Cross-Validation

Here we use k-fold cross-validation, where the original dataset is partitioned into k equal size subsamples, called folds. The k is a user-specified number, usually 5 or 10.

1. Import the function `KFold`
   `from sklearn.model_selection import KFold`
2. Divide the dataset into k folds. Some indications on the function `KFold` are given in the website
   `https://scikit-learn.org/stable/modules/generated/sklearn.model_selection.KFold.html`
3. For each fold
   (a) Learn the two models on the train set.
   (b) Evaluate the model on the test set.
4. Average the evaluation. Compare then the two models using the $F1$ score as metric.
   `from sklearn.metrics import f1_score`
   See `https://scikit-learn.org/stable/modules/generated/sklearn.metrics.f1_score.html` for more information.