11752 Machine Learning
Master in Intelligent Systems
Universitat de les Illes Balears

Handout #1: **Data Analysis**

T1. Download the dataset diabetes from `https://www.openml.org/search?type=data&status=active&id=55`. As you will see, this dataset is stored as a file ARFF (Attribute-Relation File Format).

T2. Load the dataset in Python using the ARFF importer from the *scipy* library. You can use the following function:

```
import pandas as pd
from scipy.io import arff
def read_arff_as_df_scipy(filename):
  # input : name of the file where the dataset is stored
  # output: a dataframe
  dataset = arff.loadarff(filename)
  df = pd.DataFrame(dataset[0],columns=dataset[1])
  return df
```

Listing 1: Function to load an ARFF file using the *scipy* library.

T3. Have a look at the dataset by means of methods `info()` and `head()`: find the column which stores class data (ground truth), determine the number of samples and features of this dataset, detect columns with missing values, etc.

T4. Adopt the following cleaning and filling strategy for dealing with missing values:

- Drop features with more than 40% of missing values.
- As strategy #1, drop samples with missing values and store the resulting dataframe as **df1**. Check the number of available samples and features.
- As strategy #2, fill features with missing values using the *median* for real-valued data and the *mode* for categorical data and store the resulting dataframe as **df2**. Check the number of available samples and features.

T5. Transform categorical features into integer-valued features, using progressive integer labels. You have to do it for both **df1** and **df2**.

T6. Normalize the features using *min-max* normalization. Again, do it for both **df1** and **df2**. (Do not normalize the respective *ground truths*.)

T7. Using *feature sequential selection*, choose the best 5 features, for both **df1** and **df2**, and using *forward sequential selection* (FSS) and *backward sequential selection* (BSS). This would provide you with four more datasets, let us say **df3** - **df4** (FSS) and **df5** - **df6** (BSS). [You can keep them as numpy arrays, it is not necessary to transform them to dataframes]

To use the implementation of *sequential selection* available in *scikit-learn*, you will need an evaluation model. To this end, you can use the following code:

```
  from sklearn.svm import SVC
  svm = SVC(C = 1e6, kernel = 'rbf', gamma = 'auto')
```

Listing 2: Machine learning model to be used for FSS and BSS.

and use the **svm** (Support Vector Machine) model when defining the *sequential selection* objects.

T8. Using PCA, reduce the dimensionality of the original datasets **df1** and **df2** up to 5 components. You would obtain reduced versions of **df1** and **df2** that you have to use in task T9.

T9. Compare the performance of the *six data preparation strategies* designed along tasks T4 - T8 by calculating the *accuracy* resulting from using **df1** - **df6** to train six SVM models. The closer the accuracy is to 1, the better is the performance of that model. Given the matrix of features $\mathbf{X}$ and the vector of labels $\mathbf{y}$, the following fragment of code fits a SVM and displays the resulting accuracy:

```
from sklearn.metrics import accuracy_score
from sklearn.svm import SVC
svm = SVC(C = 1e6, kernel = 'rbf', gamma = 'auto')
svm.fit(X, y)
yp = svm.predict(X)
print(accuracy_score(y, yp))
```

Listing 3: Performance evaluation for the 6 data preparation strategies.