CS 506 FALL 2019 - HW0

Introduction to Python

Due date: October 2, 2019

1 Parse and Preprocess Data

[2 pts.]
For this part you will have to write Python code to parse data. Specifically, you will use the **arrhythmia dataset** "arrhythmia.data" (you can find a detailed description about the dataset here).
Each line in this dataset corresponds to a patient and contains 280 comma-separated values. **The first 279 are the attributes**, whereas **the last element corresponds to the class** of this patient (an integer ranging from 1-16).
Your goal is to write a **function** that reads the dataset and returns two arrays (X and y), where X contains the attributes for every patient and y the corresponding class.
Be careful! The dataset also contains **missing values** denoted with a question mark '?'. You will need to take care of them and store them as NaN entries in your X array.

```
def import_data(filename):
    """
        Write your code here
    """

    return X, y
```

2 Impute or delete missing entries

(a) [1pt.] As described above, the matrix X will contain missing entries, denoted as NaN. Write a **function** that imputes these missing entries with the **median** of the corresponding feature - column in X.

(b) [1pt.] Explain why sometimes it is **better to use the median instead of the mean** of an attribute for missing values.

(c) [1pt.] Another way to deal with missing entries, is to **discard** completely the samples that do not have an entry for every attribute. Write a Python

function that discards those samples from the dataset.

```
def impute_missing( X ):
    """
    Write your code here
    """

    return

def discard_missing( X, y ):
    """
    Write your code here
    """

    return
```

3 Train-test split

[1pt.]
Usually in Machine Learning tasks, in order to test the effectiveness of an algorithm in a labeled dataset, we use a part of the dataset for training, and test the efficiency of the learned algorithm on the remaining one.
Write a **function** that gets as input the fraction of the dataset that will be used as the test set (*t_f*) and **randomly** splits the dataset into train and test sets.

```
def train_test_split( X, y, t_f ):
    """
    Write your code here
    """

    return X_train , y_train , X_test , y_test
```