# TP2: Implementac¸ao do Algoritmo de Boosting

Tasso Augusto Tomaz Pimenta 2021072198

## Table of contents

## List of Tables

```
import pandas as pd
from math import log, e

def splits(k, df):
    splits = []
    for i in range(k):
        splits.append(df[(len(df)//k)*(i):(len(df)//k)*(i+1)])
    return splits
```

```
def tree(w:list, df):#w: list,
    impury = {}
    gini_  = {}
    for x in df.columns[:len(df.columns) - 1]:#Para cada coluna dos inputs
        positive = {'x': 0, 'o': 0, 'b': 0}
        negative = {'x': 0, 'o': 0, 'b': 0}
        gini     = {'x': 0, 'o': 0, 'b': 0}
        impurity = 0.0
        for i, value in enumerate(df[x]):# Para cada valor de cada input
            if value in ('x', 'o', 'b'):
                if df.loc[i, 'x-win'] == 'positive':
                    positive[value] += w[i]
                else:
                    negative[value] += w[i]
```

```
            else: print(f"Erro na posição {i}")
        for value in ('x','o','b'):
            gini[value] = 1 - pow((positive[value]/(positive[value]+negative[value])),2) -
        for value in ('x','o','b'):
            impurity += ((positive[value] + negative[value])*gini[value])/1#soma dos pesos
        impury[x] = impurity
        gini_[x] = gini
    col = min(impury, key=impury.get)
    return [col, min(gini_[col],key=gini_[col].get)]
```

```
class H():
    def __init__(self, result: list):
        self.col = result[0]
        self.val = result[1]
    def classifier(self, x:list):
        if x[self.col] == self.val:
            return 1
        else:
            return -1
```

```
def y(x:list):
        if x['x-win']   == 'positive': return 1
        elif x['x-win'] == 'negative': return -1
        else:   print(f"Erro na posição {i}")
```

df = pd.read_csv('tic+tac+toe+endgame/tic-tac-toe.data', sep=',') cross = splits(5, df) train = pd.concat(cross[0:4]) test = pd.concat(cross[4:5]) w = [1/len(train)]*len(train) stump = tree(w,train) #print(stump) h = H(stump) sucesso,error = 0.0,0.0 for index, row in train.iterrows(): if h.classifier(row) == 1: sucesso +=1 else: error += 1 error = error/sucesso alpha = (1/2)*log((1-error)/error) for index, row in train.iterrows(): w[index] = w[index]*pow(e,-alpha*y(row)*h.classifier(row))