**I. Pen-and-paper**







0.527289

Silhouette level is high (S(C) very close to 1), thus there is good evidence for clusters to be cohesive and

well-separated.

The VC dimension for a:

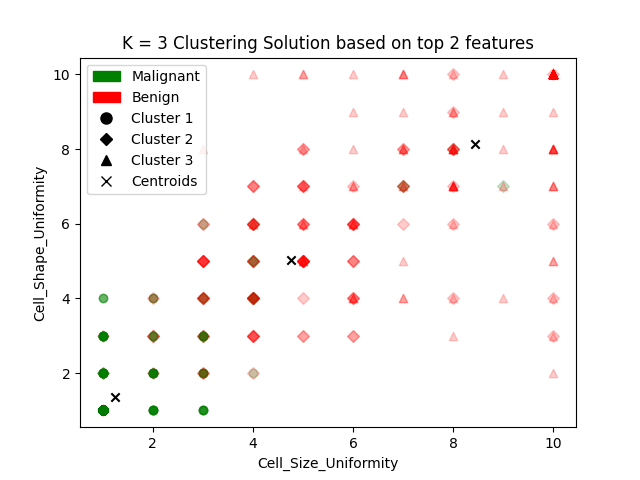
1. MLP with three hidden layers with as much nodes as the number of input variables is
2. Decision tree (DT) assuming input variables are discretized using three bins is
3. Bayesian (B) classifier with a multivariate Gaussian likelihood is

Given the decision tree’s lower VC dimension for data dimensionalities lower than 5 its more likely to have higher in sample error and lower overfitting susceptibility, for higher values it is the most likely to fit the training data (lower in-sample-error) and has increased overfitting susceptibility, followed by the MLP and the Bayesian Classifier.

The MLP, regardless of the data dimensionality always has higher VC dimension that the Bayesian Classifier, meaning that it is overall more likely to fit the training data (having lower in-sample-error) although having increased overfitting susceptibility.

**II. Programming and critical analysis**

* 1. Based on the external measure of each solution we conclude that in the solution with k=2 the clusters, overall, resemble less of their class than in the k=3 solution, meaning, each cluster in the k=3 solution is more adapted to a specific class.
  2. Based on the silhouette values of each solution, we conclude that in the solution with k=2 the cluster are overall further apart from each other than in the k=3 solution.



1. Analyzing the graph obtained in the previous question based on the two top features we can see that: in terms of classification, they classify reasonably well the data as benign/malignant; in terms of clustering, they don't describe a very good solution, with two of the cluster basically blended, working almost as a single cluster and the last one, even if clear, slightly overlapping the other group. Indicating that probably a better k (number of clusters) would be k=2.

**III. APPENDIX**

import pandas as pd

import matplotlib.pyplot as plt

import matplotlib.patches as mpatches

import matplotlib.lines as mlines

import numpy as np

from scipy.io import arff

from sklearn.cluster import KMeans

from sklearn.metrics import silhouette\_score

from sklearn.feature\_selection import mutual\_info\_classif

from sklearn.feature\_selection import SelectKBest

SEED = 39

# Extract Data

D\_breast = pd.DataFrame( arff.loadarff( "breast.w.arff" )[0] )

# Elements array

X = D\_breast.drop(columns=D\_breast.columns[-1]).to\_numpy().astype(int)

# Results array binarized

Y = D\_breast[D\_breast.columns[-1]].replace(b'benign', 0).replace(b'malignant', 1)

def quest4():

    def Cluster(k, Y\_pred):

        cluster = np.where(Y\_pred == k) [0]

        return cluster

    def ECR(k, Y, Y\_pred):

        soma = 0

        for i in range(k):

            cluster = Cluster(i, Y\_pred)

            total = [0]\*k

            for ind in cluster: total[Y[ind]] += 1

            phi = max(total)

            soma += (len(cluster) - phi)

        return (1/k) \* soma

    for k in [2, 3]:

        # Train model

        kmeans = KMeans(n\_clusters=k, random\_state=SEED).fit(X)

        Y\_pred = kmeans.predict(X)

        print(f"\n------------ K = {k} ------------")

        # Error classification rate (external measure)

        ecr = ECR(k, Y, Y\_pred)

        print(f"ECR = {ecr}")

        # Silhouette coefficient (internal measure)

        sil\_score = silhouette\_score(X, Y\_pred, random\_state=SEED)

        print(f"Silhouette = {sil\_score}")

def quest5():

    kmeans = KMeans(n\_clusters=3, random\_state=SEED)

    Y\_pred = kmeans.fit\_predict(X)

    Kbest = SelectKBest(mutual\_info\_classif, k=2)

    Kbest = Kbest.fit\_transform(X, Y)

    def marker(i):

        if Y\_pred[i] == 0: return "o"

        elif Y\_pred[i] == 1: return "^"

        elif Y\_pred[i] == 2: return "D"

    def color(i):

        if Y[i] == 0: return "green"

        elif Y[i] == 1: return "red"

    for i, point in enumerate(Kbest):

        plt.scatter(point[0], point[1], alpha=0.2, marker=marker(i), color=color(i))

    cluster\_centers = kmeans.cluster\_centers\_

    for c in cluster\_centers: plt.scatter(c[1], c[2], marker="x", color="black")

    plt.title("K = 3 Clustering Solution based on top 2 features")

    plt.xlabel("Cell\_Size\_Uniformity")

    plt.ylabel("Cell\_Shape\_Uniformity")

    green = mpatches.Patch(color='green', label='Malignant')

    red = mpatches.Patch(color='red', label='Benign')

    cl\_1 = mlines.Line2D([], [], marker='o', color="black", linestyle='None', markersize=8, label='Cluster 1')

    cl\_2 = mlines.Line2D([], [], marker='D', color="black", linestyle='None', markersize=6, label='Cluster 2')

    cl\_3 = mlines.Line2D([], [], marker='^', color="black", linestyle='None', markersize=7, label='Cluster 3')

    cl\_c = mlines.Line2D([], [], marker='x', color="black", linestyle='None', markersize=7, label='Centroids')

    plt.legend(handles=[green, red, cl\_1, cl\_2, cl\_3, cl\_c])

    plt.savefig("graph5")

quest4()

quest5()

**END**