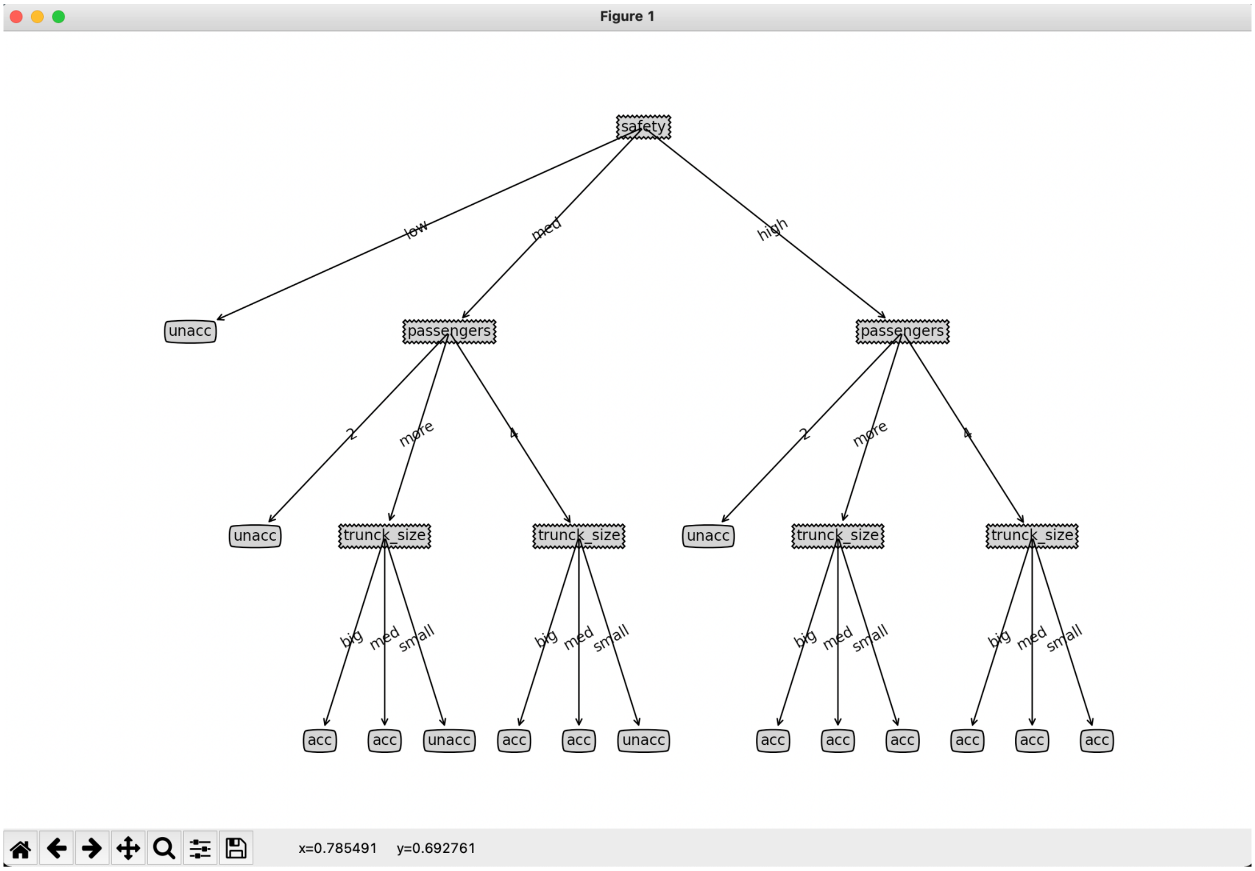
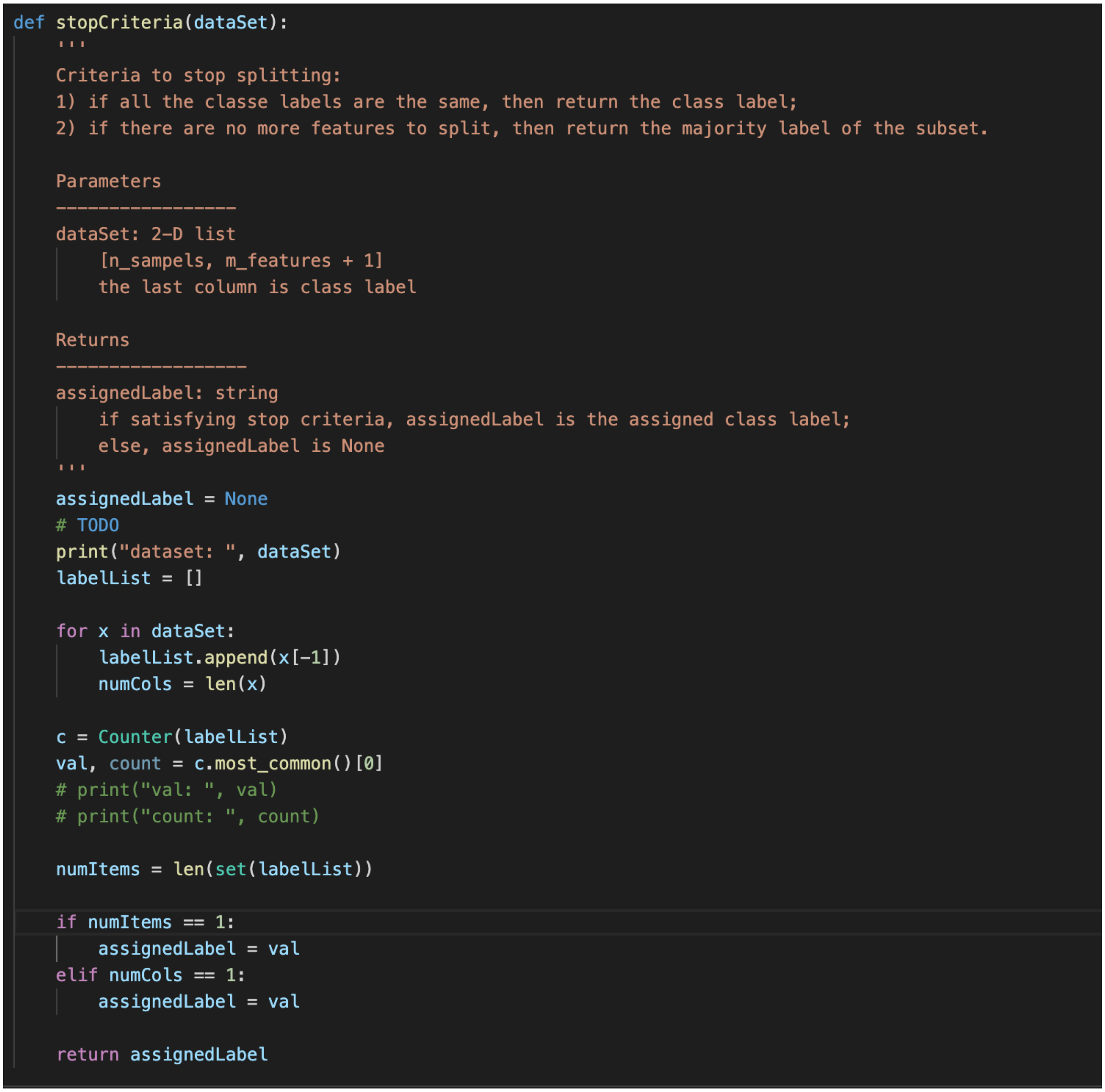
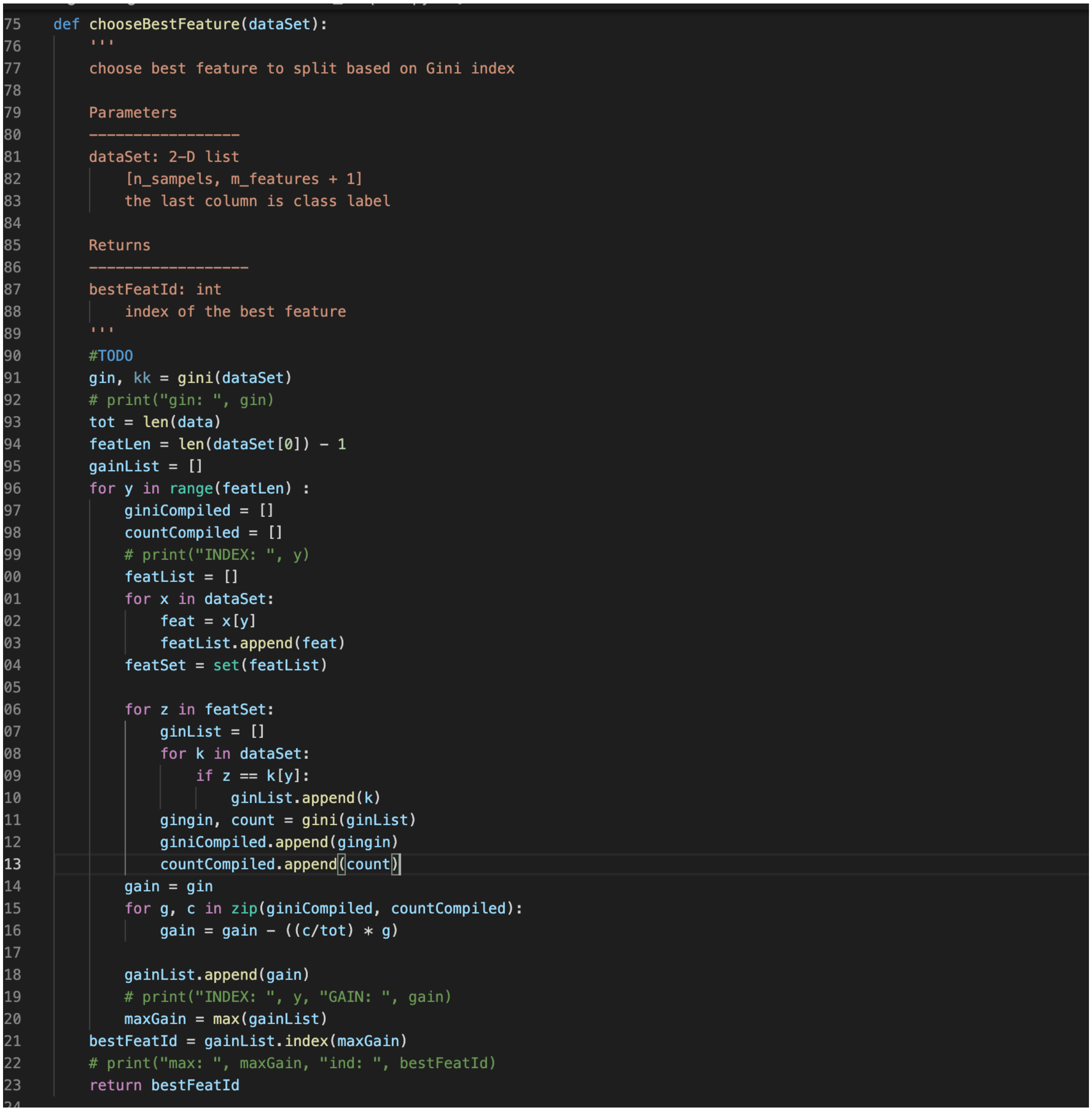
Vivek Khanolkar

ECE49595 - Assignment 4





```python
def stopCriteria(dataSet):
    '''
    Criteria to stop splitting:
    1) if all the classe labels are the same, then return the class label;
    2) if there are no more features to split, then return the majority label of the subset.

    Parameters
    -----------------
    dataSet: 2-D list
        [n_sampels, m_features + 1]
        the last column is class label

    Returns
    ------------------
    assignedLabel: string
        if satisfying stop criteria, assignedLabel is the assigned class label;
        else, assignedLabel is None
    '''
    assignedLabel = None
    # TODO
    print("dataset: ", dataSet)
    labelList = []

    for x in dataSet:
        labelList.append(x[-1])
        numCols = len(x)

    c = Counter(labelList)
    val, count = c.most_common()[0]
    # print("val: ", val)
    # print("count: ", count)

    numItems = len(set(labelList))

    if numItems == 1:
        assignedLabel = val
    elif numCols == 1:
        assignedLabel = val

    return assignedLabel
```



```
def chooseBestFeature(dataSet):
    '''
    choose best feature to split based on Gini index

    Parameters
    -----------------
    dataSet: 2-D list
        [n_sampels, m_features + 1]
        the last column is class label

    Returns
    ------------------
    bestFeatId: int
        index of the best feature
    '''
    #TODO
    gin, kk = gini(dataSet)
    # print("gin: ", gin)
    tot = len(data)
    featLen = len(dataSet[0]) - 1
    gainList = []
    for y in range(featLen) :
        giniCompiled = []
        countCompiled = []
        # print("INDEX: ", y)
        featList = []
        for x in dataSet:
            feat = x[y]
            featList.append(feat)
        featSet = set(featList)

        for z in featSet:
            ginList = []
            for k in dataSet:
                if z == k[y]:
                    ginList.append(k)
            gingin, count = gini(ginList)
            giniCompiled.append(gingin)
            countCompiled.append(count)
        gain = gin
        for g, c in zip(giniCompiled, countCompiled):
            gain = gain - ((c/tot) * g)

        gainList.append(gain)
        # print("INDEX: ", y, "GAIN: ", gain)
        maxGain = max(gainList)
    bestFeatId = gainList.index(maxGain)
    # print("max: ", maxGain, "ind: ", bestFeatId)
    return bestFeatId
```