实验一

姓名：李昀哲　　学号：20123101

1 题目和数据：

题目：数据准备及绘制散点图

数据描述

海伦收集约会数据已经有了一段时间，她把这些数据存放在文本文件 datingTestSet.txt 中，每个样本数据占据一行，总共有 1000 行。

海伦收集的样本数据主要包含以下 3 种特征：

每年获得的飞行常客里程数（Number of frequent flyer miles earned per year）；

玩视频游戏所消耗时间百分比（Percentage of time spent playing video games）；

每周消费的冰淇淋公升数（Liters of ice cream consumed per week）；

数据标签包含以下 3 类：

不喜欢的人（didntLike）；

魅力一般的人（smallDoses）；

极具魅力的人（largeDoses）；

2 算法：

本题以数据预处理和绘制散点图为主，不涉及算法。简单介绍预处理使用的函数

strip 函数：

- 用于移除字符串头尾指定的字符（默认为空格或换行符）或字符序列；
- 注意：该方法只能删除开头或是结尾的字符，不能删除中间部分的字符；
- 使用方法：***str.strip([chars])***

split 函数：

- 通过指定分隔符对字符串进行切片，如果参数 num 有指定值，则分隔 num+1 个子字符串；
- 使用方法：***str.split(str="", num=string.count(str))***

3 代码及结果

数据读入及预处理

```
import numpy as np
def file2matrix(filename):
    """
    函数说明：加载数据集
    parameters:
        fileName - 文件名
    return:
        featureMat - 特征矩阵
        classLabelVector - 类别标签向量(didntLike - 0, smallDoses - 1, largeDoses - 2)
    """
    # basic way to open txt
    data_file    = open(filename)
    data_raw     = data_file.readlines()
    data_amount = len(data_raw)

    # Prepare containers for pre-processed data
    featureMat = np.ones((data_amount, 3))
    classLabelVector = []

    # Pre-process
    line = 0
    for instance in data_raw:
        instance = instance.strip()
        instance = instance.split("\t")
        featureMat[line, :] = instance[0:3]

        if instance[3] == 'largeDoses':
            classLabelVector.append(2)
        if instance[3] == 'smallDoses':
            classLabelVector.append(1)
        if instance[3] == 'didntLike':
            classLabelVector.append(0)
        line += 1
    return featureMat, classLabelVector
```

绘制散点图展示数据

```
def showdatas(datingDataMat, datingLabels):
    """
    函数说明：绘制散点图
    parameters:
        datingDataMat - 特征矩阵
```

```
        datingLabels -  类别标签向量(didntLike - 0, smallDoses - 1, largeDoses - 2)
    """
    # 设置子图
    fig, axs = plt.subplots(nrows=3, ncols=1, figsize=(18, 12))
    # 设置颜色
    colors = []
    for i in datingLabels:
        if i == 0:
            colors.append('black')
        if i == 1:
            colors.append('orange')
        if i == 2:
            colors.append('red')
    axs[0].scatter(x = datingDataMat[:,0], color=colors, y = datingDataMat[:,1], s=15)
    axs[0].set_xlabel('Number of frequent flight miles earned per year')
    axs[0].set_ylabel('Percentage of time spent playing video games')

    axs[1].scatter(x = datingDataMat[:,0], color=colors, y = datingDataMat[:,2], s=15)
    axs[1].set_xlabel('Number of frequent flight miles earned per year')
    axs[1].set_ylabel('Liters of ice cream consumed per week')

    axs[2].scatter(x = datingDataMat[:,2], color=colors, y = datingDataMat[:,1], s=15)
    axs[2].set_xlabel('Liters of ice cream consumed per week')
    axs[2].set_ylabel('Percentage of time spent playing video games')


    didntLike = mlines.Line2D([],[],color='black', marker='.', markersize=6, label='didntLike')
    smallDoses = mlines.Line2D([],[],color='orange', marker='.', markersize=6,
label='smallDoses')
    largeDoses = mlines.Line2D([],[],color='red', marker='.', markersize=6, label='largeDoses')

    axs[0].legend(handles=[didntLike, smallDoses, largeDoses])
    axs[1].legend(handles=[didntLike, smallDoses, largeDoses])
    axs[2].legend(handles=[didntLike, smallDoses, largeDoses])
```

主函数

```
%matplotlib inline
if __name__ == '__main__':
    filename = "kNN_Dating/datingTestSet.txt"
    datingDataMat, datingLabels = file2matrix(filename)
    showdatas(datingDataMat, datingLabels)
```

数据结果：根据要求基本实现预期可视化的效果（图 3 横纵坐标不同于示例，进行了交换）