1. A report, in PDF format, documenting the following:

    1. The data set, features, tasks, and algorithms chosen
    2. The procedures used to analyze the data and build models
    3. The performance of each of the models on training and test sets

Data Set

1. Title: Wine recognition data

2. Description: These data are the results of a chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The analysis determined the quantities of 13 constituents found in each of the three types of wines.

    The wine data contains 178 object. The first column is the class identifier (1-3).

3. Data Features

    The 13 attributes include: alcohol, malic acid, ash, alkalinity of ash, magnesium, total phenols, proanthocyanins, color intensity, hue, OD280/OD315 of diluted wines, proline

4. Classes: 3

    Class 1: 59

    Class 2: 71

    Class 3: 48

Tasks: Classification

This project aims to classify the new wine data correctly to the category (1, 2 or 3)

Algorithms:

Two classifiers are used: **Multinomial logistic regression** and **K-nearest neighbor**

Procedures

Alg1: Multinomial Logistic Regression

1. Data preparation

    a. Set the a new attribute called “CAT” as a factor representing category

    ```
    > wine$CAT=factor(wine$V1)
    ```

    b. Split data: use split.sample() to separate the data into training and test data

    ```
    > set.seed(100) # make sure you get the results the same each time and other people can replicate it

    > split<-sample.split(wine$V1,SplitRatio = 0.8)

    > dresstrain<-wine[split, 1, 1]
    ```

```
> dresstest<-wine[split, 1, 1]
```

2. Build multinomial logistic regression model: mlmodel
   a. Include “nnet” library (install if necessary: install.packages(“nnet”)
      ```
      > library(nnet)
      ```
   b. Call “multinom” function to build a logistic regression model
      ```
      > mlmodel<-multinom(V1~.-CAT, data=dresstrain, family=binomial)
      ```
   c. (Optional) Check the details of the model by calling “summary”
3. Predict

   Call “predict” functions and use the logistic model built from last step to classify
4. Misclassification error

   Show the confusion matrix to see how much you correctly classify the test data
   ```
   > table(predict(mlmodel, dresstest), dresstest$CAT)
   ```

Alg2: K-Nearest Neighbor

1. Data Preparation – same as Alg1
2. Build multinomial logistic regression model
   a. Include “class” library (install if necessary: install.packages(“class”)
      ```
      > library(class)
      ```
   b. Call “knn” function to build a model
      ```
      > kmodel<-knn(train=dresstrain[,2:14], test=dresstest[,2:14], cl=dresstrain$CAT, k=3)
      ```
   c. (Optional) Check the details of the model by calling “summary”
3. Predict

   Call “predict” functions and use the logistic model built from last step to classify
4. Misclassification error

   Show the confusion matrix to see how much you correctly classify the test data
   ```
   > cm<-table(kmodel, dresstest$V1) #(also for dresstrain)
   > print(cm)
   ```

Performance on Training and Test Sets

1. **MLMODEL (Multinomial Logistic Model)**

-Training set: 142 objects

- **The confusion matrix:**

	1	2	3
1	47	0	0
2	0	57	0
3	0	0	38

- **MSE = 0**

-Test set: 36 objects

- **The confusion matrix:**

	1	2	3
1	10	0	1
2	2	14	0
3	0	0	9

- **MSE = ((3-1)*(3-1)+(2-1)*(2-1))/36 = 0.14**

2. KMODEL (KNN Model)

-Training set: 142 objects

- **The confusion matrix**

	1	2	3
1	9	1	1
2	1	10	3
3	2	3	6

- **MSE = (2*(2-1)*(2-1)+3*(3-1)*(3-1)+6*(3-2)*(3-2))/142 = 0.14**

-Test set: 36 objects

- **The confusion matrix**

	1	2	3
1	9	1	1
2	1	10	4
3	2	3	5

- **MSE = (2*(2-1)*(2-1)+3*(3-1)*(3-1)+7*(3-2)*(3-2))/36 = 0.58**

From the comparison of the above 2 models, it is obvious the multinomial logistic model has much better performance since it is more sophisticated.