Python Tutorial 2 Exercises with Solutions

January 24, 2020

This is the exercise of "Python Tutorial 2" for Prof. Xin Tong's DSO 530 class at the University of Southern California in spring 2020.

1. Write a program to achieve the following things (try to do *parts 2) - 5)* without looking at the **Python Tutorial 2**):

1) read in the **Wine** dataset and add column names just like what we do in Section 3 of **Python Tutorial 2**.
2) use *train_test_split* from *sklearn.model_selection* to partition this dataset into separate test and training datasets to get *X_train1, X_test1, y_train1, y_test1*: set *test_size* to 0.4, set *random_state* to 1;
3) use *train_test_split* from *sklearn.model_selection* to partition this dataset into separate test and training datasets to get *X_train2, X_test2, y_train2, y_test2*: set *test_size* to 0.4, set *random_state* to 2;
4) use *train_test_split* from *sklearn.model_selection* to partition this dataset into separate test and training datasets to get *X_train3, X_test3, y_train3, y_test3*: set *test_size* to 0.4, set *random_state* to 1;
5) compare the column means of *X_train1*, *X_train2* and *X_train3*

Anwser:

This exercise aims to help you understand the *random_state* parameter when you are using *train_test_split* in Section 3 of **Python Tutorial 2**. The *random_state* parameter is very important for the reproducibility of the results.

[1]:
```
import pandas as pd
import numpy as np
df_wine = pd.read_csv('https://archive.ics.uci.edu/ml/
 ↪machine-learning-databases/wine/wine.data', header=None)
df_wine.columns = ['Class label', 'Alcohol', 'Malic acid', 'Ash', 'Alcalinity␣
 ↪of ash', 'Magnesium', 'Total phenols', 'Flavanoids', 'Nonflavanoid phenols',␣
 ↪'Proanthocyanins', 'Color intensity', 'Hue', 'OD280/OD315 of diluted wines',␣
 ↪'Proline']
df_wine.head()
```

[1]:
```
   Class label  Alcohol  Malic acid   Ash  Alcalinity of ash  Magnesium  \
0            1    14.23        1.71  2.43               15.6        127
1            1    13.20        1.78  2.14               11.2        100
```

```
2            1    13.16        2.36  2.67              18.6         101
3            1    14.37        1.95  2.50              16.8         113
4            1    13.24        2.59  2.87              21.0         118

   Total phenols  Flavanoids  Nonflavanoid phenols  Proanthocyanins  \
0           2.80        3.06                  0.28             2.29
1           2.65        2.76                  0.26             1.28
2           2.80        3.24                  0.30             2.81
3           3.85        3.49                  0.24             2.18
4           2.80        2.69                  0.39             1.82

   Color intensity   Hue  OD280/OD315 of diluted wines  Proline
0             5.64  1.04                          3.92     1065
1             4.38  1.05                          3.40     1050
2             5.68  1.03                          3.17     1185
3             7.80  0.86                          3.45     1480
4             4.32  1.04                          2.93      735
```

[2]:

```python
from sklearn.model_selection import train_test_split
X, y = df_wine.iloc[:, 1:].values, df_wine.iloc[:, 0].values
X_train1, X_test1, y_train1, y_test1 = \
    train_test_split(X, y,
                     test_size=0.3,
                     random_state=1,
                     stratify=y)

X_train2, X_test2, y_train2, y_test2 = \
    train_test_split(X, y,
                     test_size=0.3,
                     random_state=2,
                     stratify=y)

X_train3, X_test3, y_train3, y_test3 = \
    train_test_split(X, y,
                     test_size=0.3,
                     random_state=1,
                     stratify=y)
```

[3]:

```python
print("The mean of X_train1:", X_train1.mean())
print("The mean of X_train2:", X_train2.mean())
print("The mean of X_train3:", X_train3.mean())
```

```
The mean of X_train1: 68.62389330024814
The mean of X_train2: 68.99977419292803
The mean of X_train3: 68.62389330024814
```

You can see that the mean of *X_train1* and *X_train2* are different because their *random_state* are different and the mean of *X_train1* and *X_train3* are identical because their *random_state* are the same.

2. Write a program to achieve the following things (try to do the problems without looking at the **Python Tutorial 2**):

1) First, create some missing values out of this *Wine* dataset: replace the first 20 rows of the *Alcohol* feature by *np.NaN* in the whole **Wine** dataset and take the whole dataset with 20 missing values as the starting point.

2) Impute the miss values using the **median imputation** techniques.

3) Answer the following question:

   Is it a recommended practice to split the dataset which was imputed in *step 2)* into training and test sets? If not, what would you do if you knew that you would need to split the data into training and test sets?

Answer:

1)

[4]:
```
import pandas as pd
import numpy as np
df_wine = pd.read_csv('https://archive.ics.uci.edu/ml/
 ↪machine-learning-databases/wine/wine.data', header=None)
df_wine.columns = ['Class label', 'Alcohol', 'Malic acid', 'Ash', 'Alcalinity␣
 ↪of ash', 'Magnesium', 'Total phenols', 'Flavanoids', 'Nonflavanoid phenols',␣
 ↪'Proanthocyanins', 'Color intensity', 'Hue', 'OD280/OD315 of diluted wines',␣
 ↪'Proline']
df_wine.loc[0:19,'Alcohol'] = np.nan
print(df_wine.iloc[0:29,0:6])
```

```
    Class label  Alcohol  Malic acid   Ash  Alcalinity of ash  Magnesium
0             1      NaN        1.71  2.43               15.6        127
1             1      NaN        1.78  2.14               11.2        100
2             1      NaN        2.36  2.67               18.6        101
3             1      NaN        1.95  2.50               16.8        113
4             1      NaN        2.59  2.87               21.0        118
5             1      NaN        1.76  2.45               15.2        112
6             1      NaN        1.87  2.45               14.6         96
7             1      NaN        2.15  2.61               17.6        121
8             1      NaN        1.64  2.17               14.0         97
9             1      NaN        1.35  2.27               16.0         98
10            1      NaN        2.16  2.30               18.0        105
11            1      NaN        1.48  2.32               16.8         95
12            1      NaN        1.73  2.41               16.0         89
13            1      NaN        1.73  2.39               11.4         91
14            1      NaN        1.87  2.38               12.0        102
```

```
15            1      NaN        1.81  2.70               17.2        112
16            1      NaN        1.92  2.72               20.0        120
17            1      NaN        1.57  2.62               20.0        115
18            1      NaN        1.59  2.48               16.5        108
19            1      NaN        3.10  2.56               15.2        116
20            1    14.06        1.63  2.28               16.0        126
21            1    12.93        3.80  2.65               18.6        102
22            1    13.71        1.86  2.36               16.6        101
23            1    12.85        1.60  2.52               17.8         95
24            1    13.50        1.81  2.61               20.0         96
25            1    13.05        2.05  3.22               25.0        124
26            1    13.39        1.77  2.62               16.1         93
27            1    13.30        1.72  2.14               17.0         94
28            1    13.87        1.90  2.80               19.4        107
```

2)

[5]:
```
from sklearn.impute import SimpleImputer

imp1 = SimpleImputer(missing_values = np.nan, strategy = 'median')
imputed_data = imp1.fit_transform(df_wine.values)

df_wine_imputed = pd.DataFrame(data = imputed_data)
df_wine_imputed.columns = ['Class label', 'Alcohol', 'Malic acid', 'Ash',␣
 ↪'Alcalinity of ash', 'Magnesium', 'Total phenols', 'Flavanoids',␣
 ↪'Nonflavanoid phenols', 'Proanthocyanins', 'Color intensity', 'Hue', 'OD280/
 ↪OD315 of diluted wines', 'Proline']

print(df_wine_imputed.iloc[0:29,0:6])
```

```
    Class label  Alcohol  Malic acid   Ash  Alcalinity of ash  Magnesium
0           1.0   12.855        1.71  2.43               15.6      127.0
1           1.0   12.855        1.78  2.14               11.2      100.0
2           1.0   12.855        2.36  2.67               18.6      101.0
3           1.0   12.855        1.95  2.50               16.8      113.0
4           1.0   12.855        2.59  2.87               21.0      118.0
5           1.0   12.855        1.76  2.45               15.2      112.0
6           1.0   12.855        1.87  2.45               14.6       96.0
7           1.0   12.855        2.15  2.61               17.6      121.0
8           1.0   12.855        1.64  2.17               14.0       97.0
9           1.0   12.855        1.35  2.27               16.0       98.0
10          1.0   12.855        2.16  2.30               18.0      105.0
11          1.0   12.855        1.48  2.32               16.8       95.0
12          1.0   12.855        1.73  2.41               16.0       89.0
13          1.0   12.855        1.73  2.39               11.4       91.0
14          1.0   12.855        1.87  2.38               12.0      102.0
15          1.0   12.855        1.81  2.70               17.2      112.0
16          1.0   12.855        1.92  2.72               20.0      120.0
```

17	1.0	12.855	1.57	2.62	20.0	115.0
18	1.0	12.855	1.59	2.48	16.5	108.0
19	1.0	12.855	3.10	2.56	15.2	116.0
20	1.0	14.060	1.63	2.28	16.0	126.0
21	1.0	12.930	3.80	2.65	18.6	102.0
22	1.0	13.710	1.86	2.36	16.6	101.0
23	1.0	12.850	1.60	2.52	17.8	95.0
24	1.0	13.500	1.81	2.61	20.0	96.0
25	1.0	13.050	2.05	3.22	25.0	124.0
26	1.0	13.390	1.77	2.62	16.1	93.0
27	1.0	13.300	1.72	2.14	17.0	94.0
28	1.0	13.870	1.90	2.80	19.4	107.0

3) It's not recommended to impute the whole dataset first and then to split it into training and test sets. If we do so, we use all other rows (including part of the test data) to select the median and use that to impute missing values. As we mentioned in the lecture, it's not recommended that we use the test data to impute the missing values. Thus, what we should do is to split the unimputed dataset into training and test sets first, then use the training data to fit the model, and then impute the missing values in training and test dataset using that model.

[6]:
```python
from sklearn.model_selection import train_test_split

# split the whole data into training and test sets
df_wine_train, df_wine_test = \
    train_test_split(df_wine,
                     test_size=0.3,
                     random_state=1,
                     stratify=y)
# use the training data to fit the model
imp2 = SimpleImputer(missing_values = np.nan, strategy = 'median')
imp2 = imp2.fit(df_wine_train.values)

# use the trained model to impute the missing values in training and test dataset
df_wine_train_imputedValues = imp2.transform(df_wine_train.values)
df_wine_test_imputedValues = imp2.transform(df_wine_test.values)

# transform the results to DataFrame and add columns
df_wine_train_imputed = pd.DataFrame(data = df_wine_train_imputedValues)
df_wine_test_imputed = pd.DataFrame(data = df_wine_test_imputedValues)
df_wine_train_imputed.columns = ['Class label', 'Alcohol', 'Malic acid', 'Ash', 'Alcalinity of ash', 'Magnesium', 'Total phenols', 'Flavanoids', 'Nonflavanoid phenols', 'Proanthocyanins', 'Color intensity', 'Hue', 'OD280/OD315 of diluted wines', 'Proline']
```

```
df_wine_test_imputed.columns = ['Class label', 'Alcohol', 'Malic acid', 'Ash',␣
 ↪'Alcalinity of ash', 'Magnesium', 'Total phenols', 'Flavanoids',␣
 ↪'Nonflavanoid phenols', 'Proanthocyanins', 'Color intensity', 'Hue', 'OD280/
 ↪OD315 of diluted wines', 'Proline']

# generate X_train, y_train, X_test and y_test
X_train, y_train = df_wine_train_imputed.iloc[:, 1:].values,␣
 ↪df_wine_train_imputed.iloc[:, 0].values
X_test, y_test = df_wine_test_imputed.iloc[:, 1:].values, df_wine_test_imputed.
 ↪iloc[:, 0].values

print("df_wine_train_imputed: \n",df_wine_train_imputed.iloc[0:10,0:6],"\n\n")
print("df_wine_test_imputed: \n",df_wine_test_imputed.iloc[0:10,0:6])
```

```
df_wine_train_imputed:
    Class label  Alcohol  Malic acid   Ash  Alcalinity of ash  Magnesium
0          1.0    12.87        1.95  2.50               16.8      113.0
1          2.0    12.77        3.43  1.98               16.0       80.0
2          3.0    13.17        5.19  2.32               22.0       93.0
3          2.0    12.72        1.75  2.28               22.5       84.0
4          1.0    12.87        2.36  2.67               18.6      101.0
5          3.0    13.58        2.58  2.69               24.5      105.0
6          2.0    12.70        3.87  2.40               23.0      101.0
7          1.0    13.56        1.73  2.46               20.5      116.0
8          1.0    13.28        1.64  2.84               15.5      110.0
9          1.0    13.58        1.66  2.36               19.1      106.0


df_wine_test_imputed:
    Class label  Alcohol  Malic acid   Ash  Alcalinity of ash  Magnesium
0          3.0    13.52        3.17  2.72               23.5       97.0
1          3.0    13.11        1.90  2.75               25.5      116.0
2          3.0    14.16        2.51  2.48               20.0       91.0
3          2.0    11.84        0.89  2.58               18.0       94.0
4          2.0    12.29        3.17  2.21               18.0       88.0
5          1.0    13.72        1.43  2.50               16.7      108.0
6          2.0    12.08        2.08  1.70               17.5       97.0
7          1.0    13.05        1.65  2.55               18.0       98.0
8          2.0    12.37        1.17  1.92               19.6       78.0
9          3.0    12.25        4.72  2.54               21.0       89.0
```