

Department of Computer Engineering

University of Peradeniya

Data Mining and Machine Learning
Lab 02

November 8, 2018

1 Introduction

NumPy, short for Numerical Python, is the fundamental package required for high-performance scientific computing and data analysis. Most of the frameworks and libraries such as Scikit-Learn, Tensorflow which require numerical calculation use NumPy.

Pandas contains high-level data structures and manipulation tools designed to make data analysis fast and easy in Python. This is built on top of NumPy and makes it easy to use in NumPy-centric applications.

Objective of this lab is to provide students hands on experience on NumPy and Pandas

2 NumPy

2.1 Creation

```
import numpy as np # import numpy module as np

matrix = np.array([np.arange(3), [i for i in range(3,6)], [6,7,8]])
matrix
Out[14]:
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

2.2 Initialization

```
np.zeros([4,5,6]) # create an array of zeros
np.ones([3,4], dtype=float) # create an array of ones
np.arange(1,10,1).reshape(3,3) # create an array evenly spaced values
np.linspace(0,2,9) # create an array of zeros
np.full([3,3], 4) # create a constant array
np.eye(3) # create an identity matrix
```

2.3 Copying, Sorting, Slicing

```
np.copy(matrix)
matrix.copy() # deep copy
matrix.view() # shallow copy
matrix.sort() # return a sorted copy of an array reference to same
#location.
matrix.sort(axis=1)
matrix[0:, :1]
matrix[:2, 1:]
matrix[:2, :]
```

2.3.1 Try Out

```
matrix[1,0]
matrix[0] = 42
matrix[1:3]
matrix[]
matrix[1:]
matrix[1:100]
matrix[: ]
matrix[1:, :2]
matrix[:2, 1:]
matrix.ravel()
matrix[:, 1].copy()
matrix[1].tolist()
matrix.reshape(-1)
```

Now see Fig. 1 for an illustration.

2.4 Apply elements wise operations and functions of NumPy

2.4.1 Try Out

```
np.sqrt(matrix)
np.exp(matrix)
np.min(np.maximum(np.random.randn(6), np.random.randn(6)))
np.mean(matrix)
np.mean(matrix, axis=0)
np.sum(matrix)
np.invert(matrix)
np.random.randint(0, 1)
```

Hope by now, you have the basic understanding about how to deal with NumPy. Try to solve this problem. Let's assume you are to implement basic version of random walk. Walk would start at any point(e.g. 0 or 10) step of 0 or 1. Current position should be deducted by 1 for 0 value occurrence. Try to implement single random walk with 500 steps.





	Expression	Shape
	<code>arr[:2, 1:]</code>	<code>(2, 2)</code>
	<code>arr[2]</code>	<code>(3,)</code>
	<code>arr[2, :]</code>	<code>(3,)</code>
	<code>arr[2:, :]</code>	<code>(1, 3)</code>
	<code>arr[:, :2]</code>	<code>(3, 2)</code>
	<code>arr[1, :2]</code>	<code>(2,)</code>
	<code>arr[1:2, :2]</code>	<code>(1, 2)</code>

Figure 1: Two dimensional array slicing

3 Python Classes

3.1 Class definition syntax

```
class RandomWalk:
    <statement-1>
    .
    .
    .
    <statement-N>
```

3.2 Class Objects

```
class RandomWalk:
    def __init__(self, position):
        self.position = position

    def walk(self):
        return walked_path
```

```
random_walker = RandomWalk(200)
```

4 Pandas

4.1 Importing Pandas

```
import pandas as pd
```

4.2 Series and Data Frames

Series and Data Frames are the primary objects which provided by Pandas. A Series object is the base data structure which operates similar to NumPy arrays with one more additional feature, i.e., indexing capabilities. When it is required more than one Series of data that is aligned by a common index pandas DataFrame can be employed.

4.2.1 Creating Series and Data Frame

There are numerous ways to construct Series and Data Frame.

```
# create a series with a list
s = pd.Series([2,4,1,-4,'home'], index=['a', 'b', 'c', 'd', 'e'])
# TODO What is the data type of s? Can it be changed?
# dtype=object

# create a Data Frame with a dictionary
data = {'population': [1.5, 1.7, 3.6, 2.4, 2.9],
        'state': ['Ohio', 'Ohio', 'Ohio', 'Nevada', 'Nevada'],
        'year': [2000, 2001, 2002, 2001, 2002]}
df = pd.DataFrame(data, columns=['year', 'state', 'population',
                                'debt'], index=['one', 'two', 'three', 'four', 'five'])
```

4.2.2 Accessing and Modifying

```
s[['a', 0]]
s.values[2:]
df[['population', 'state']]
df.population
df.ix[0:,]
df.ix[2:4:,2]
df.iloc[2:4:,2]
df.loc['one']
df.debt = 34.67
df.debt = [ df.ix[:,2][i]*5 for i in range(0, df.shape[0])]
df.head()
df.tail()
df.sample(n=5)
df['newColumn'] = pd.Series(np.random.randn(df.shape[0]),
                             index=df.index)
```

4.2.3 Loading Data from CSV File

```
df = pd.read_csv('sampleDataSet.csv')
df.shape
df = pd.read_csv('sampleDataSet.csv', names = ['a', 'b', 'c',
'd', 'e', 'f', 'g', 'h', 'i'])
df.shape
# TODO Comment on result of df.shape() with and without setting
# names
# TODO Try adding header=None parameter without setting names
# parameter
```

4.2.4 Dealing with Missing Data

```
df.isnull().a
df.isnull().sum()
df = df[df.isnull().a != True]
df.dropna(axis=0).isnull().sum()
df.dropna(axis=1)
df.dropna(axis=1, how='all')
df.dropna(axis=1, thresh=1)
df.drop('a', axis=1)
df.fillna(0)
df.replace(0, 5)
df.replace('.', np.nan)
df[np.random.rand(df.shape[0]) > 0.5] = 1.5
```

4.2.5 Applying Functions

Function can be written as a lambda expression or a ordinary function df

```
f = lambda df: df.max() - df.min()

def f(x):
    return x.max() - x.min()

df.ix[:,3:5].apply(f) # applying function element-wise
```

4.2.6 Group Operations

```
grouped = df[['a', 'b']].groupby(df['i'])
grouped.mean()
grouped = df[['a', 'b']].groupby([df['i'], df['c']]).mean()
grouped.unstack()
```

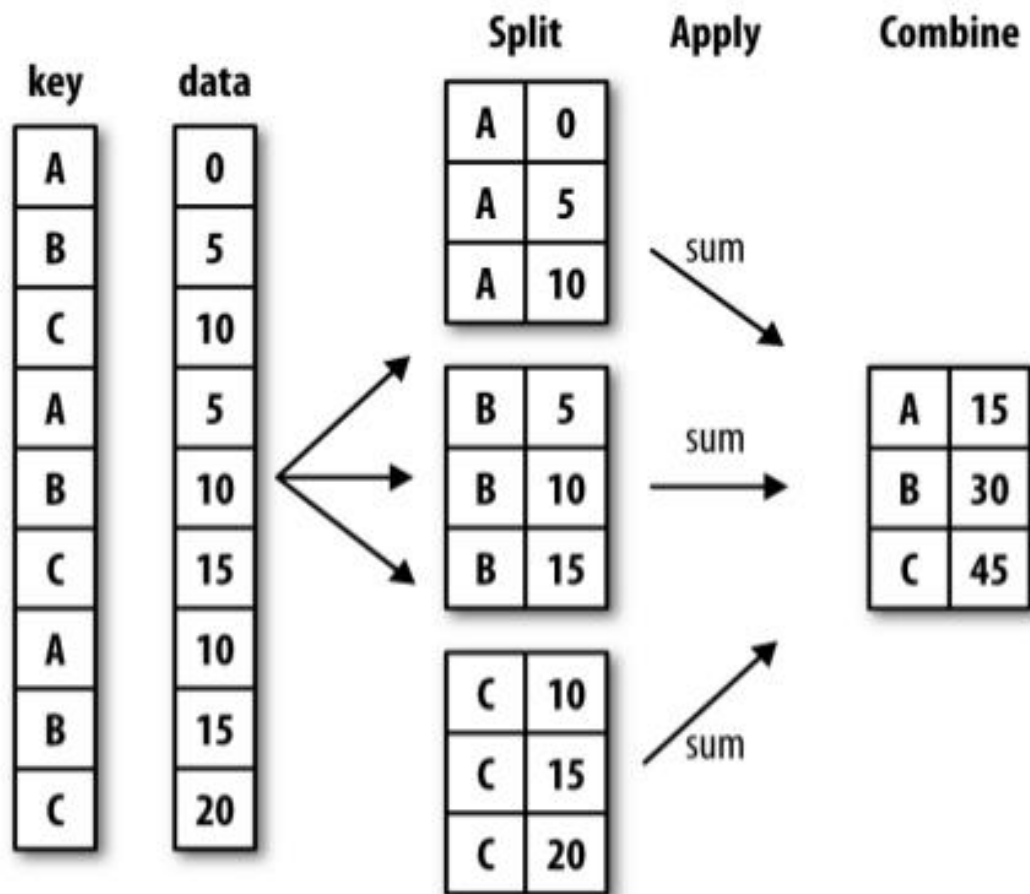


Figure 2: Group Aggregation

4.2.7 Summarize Data

```
df['a'].unique()
df['a'].value_counts() # count the number of rows with each unique
# value of variable
df.describe() # basic descriptive statistics for each columns
df.mean()
df.median()
df.sort_index().head()
```

4.2.8 Visualization

```
df.plot(kind='bar')
df.plot(kind='hist')
df.boxplot()
```

4.3 Try Out

Data wrangling is the process of transforming and mapping data from *raw* data into another format with the intent of making it more appropriate and valuable for a variety of downstream purposes such as analytics. Let try to do wrangling on the lab03Exercise dataset which is the same dataset used in lab 01, but this time it comprises of missing values and there is no way of knowing about the dataset. Once you proceed with below steps you would able to understand how Pandas can be used for data wrangling.

1. Apply what you have done with exercises in section 02 on lab02Exercise01 dataset and comment on the result.
2. Load the lab02Exercise04 dataset while specifying column names (channel1, channel2, channel3, channel4 and channel5) using *pandas.read_csv* function.
3. If there are any missing values in each column fill those values with the mean of the corresponding column.
4. To see the correlation between one column to all other columns, use following code segment and comment on diagonal plot.

```
from pandas.tools.plotting import scatter_matrix
scatter_matrix(data, alpha=0.2, figsize=(6, 6), diagonal='kde')
```

5. Add a new column named as **"class"** on lab02Exercise04 dataset. Values of this column either 1 or 0 which should be derived based on the following condition.

```
if ((column1 + column5)/2 < (column2 + column3 + column4)/3) then
    value ← 1
else
    value ← 0
end if
where columni ∈ {1, ..., 5} is the mean value of ith column.
```

5 Lab Exercise

You have to practice all the commands and exercises in the lab and implement random walker in the section 2.4.1 using NumPy package and Data wrangling using Pandas in section 4.3 for the submission.

6 Submission

Submit two text files for two sections(i.e NumPy and Pandas). Your files should contain all your commands you have tried out in the lab with the answers for TODO sections(as comments in the text) and Try Out .Rename it as **14xxxlab02np.txt** and **14xxxlab02pd.txt** where xxx is your registration number.

7 Important

Make sure that now you have the basic understand what we did during the lab. This lab is really important for successive labs as well. If you do not understand any concepts, make sure you get some help from instructors.

8 Deadline

The deadline: November 14, 23:55:00 GMT+5:30.