Applied Analytics and Predictive Modeling

Spring 2021

Lecture-3

Lydia Manikonda

manikl@rpi.edu

Rensselaer

Some of the slides adapted from Intro to Data Mining Tan et al. 2nd edition

Today’s agenda

- Previous week – Data Mining; Python Basics; Numpy
- Data Preprocessing
- Python Packages – Pandas
- Class exercises

Overview

Core Ideas of Data Mining

- Data and Dimensionality Reduction
- Data Exploration
- Data Visualization
- Association Rules & Recommendation systems
- Classification
- Clustering
- Prediction

Data Exploration

- Data sets are typically large, complex & messy
- Based on the task at hand, we have to process the data
- Use techniques of Reduction and Visualization

Dimensionality Reduction

- Shrinking the complex/large data into simpler/smaller data
- Reducing the number of variables/columns (e.g., principal components) – **Dimensionality Reduction**
- Reducing the number of records/rows (e.g., clustering) -- **Sampling**

Data Visualization

- Very important to understand the data – in particular, to examine the relationships between the attributes
- Graphs and plots of data
- Histograms, boxplots, bar charts, scatterplots

- Research translation exercise helped get some understanding about this

Association Rules

- To identify rules that define “what goes with what” in transactions
- Example: “If X was purchased, Y was also purchased” given a set of transactions
- Very useful in recommendation systems – “Our records show you bought X, you may also like Y”
- Also called as “affinity analysis”

Recommender Systems

- Collaborative filtering – Technique used by recommendation systems
- The main goal is to recommend items that we may like
- Various aspects that customers view, select, purchase, rate, etc
- User-based recommendation: Recommend products that “customers like you” purchase
- Item-based recommendation: Recommend products that share a “product purchaser profile” with your purchases

Supervised Learning

- Given training data (where the target value is known), the goal is to predict a single “target” or “outcome” variable
- Can be classified into two types – Classification and Prediction

Supervised Learning – Classification

- Main aim is to predict an outcome variable (or target variable)
- The target variable can be binary or multi-class

- Examples: Fraudulent or Non-fraudulent transaction; Pass or Fail; Rainy or Sunny; etc.

Supervised Learning – Prediction

- Main aim is to predict the outcome variable (usually in terms of a probability value)
- Common methods that could perform prediction are – Regression
- Examples: performance evaluation, revenue estimation, sales percentage, etc

Unsupervised Learning

- Main aim is to segment data into meaningful segments or detect patterns
- There is no target (outcome) variable to predict or classify
- Common methods include clustering.

Handling data

Pandas

- Most popular python library for data analysis
- Highly optimized performance

- Using: 1) **Series**; 2) **DataFrames**

Pandas – Series

- One dimensional array to store any data type

```
>> import pandas as pd
>> a = pd.Series(data, index = Index)
```

- **data** can be:
  - Scalar value – integer, string
  - Dictionary – <key, value> pair
  - Ndarray
- **Index** by default is from 0, 1, 2, … (*n*-1) where *n* is the length of the data

Pandas – Series

```
>> data = [1, 2, 3, 4, 5, 6, 7]
>> s = pd.Series(data)
```

```
0    1
1    2
2    3
3    4
4    5
5    6
6    7

dtype: int64
```

```
>> Index = [‘a’, ’b’, ‘c’, ‘d’, ‘e’, ‘f’, ‘g’]
>> s1 = pd.Series(data, Index)
```

```
a    1
b    2
c    3
d    4
e    5
f    6
g    7

dtype: int64
```

Pandas – Series

```
>> diction = {‘a’: 1, ‘b’:2, ‘c’:3, ‘d’:4, ‘e’:5, ‘f’:6, ‘g’:7}
>> s = pd.Series(dictionary)
```

```
a    1
b    2
c    3
d    4
e    5
f    6
g    7
dtype: int64
```

Pandas – Dataframes

- DataFrames is two-dimensional data structure that consists of rows and columns

```
>> import pandas as pd
>> s = pd.DataFrame(data)
```

data can be:

- One or more dictionaries
- One or more series
- 2D-numpy Ndarray

Pandas – DataFrames

```
>> diction1 ={'a':1, 'b':2, 'c':3, 'd':4}
>> diction2 ={'a':5, 'b':6, 'c':7, 'd':8, 'e’:9}
>> data = {'first':diction1, 'second':diction2}
>> df = pd.DataFrame(data)
```

	first	second
a	1.0	5
b	2.0	6
c	3.0	7
d	4.0	8
e	NaN	9

Pandas – DataFrames

```
>> import pandas as pd
>> s1 = pd.Series([1, 3, 5, 7, 9, 11, 13])
>> s2 = pd.Series([1.1, 2.2, 3.3, 4.4, 5.5, 6.6])
>> s3 = pd.Series(['a', 'b', 'c', 'd', 'e'])

>> data ={'first':s1, 'second':s2, 'third':s3}
>> series = pd.DataFrame(data)
```

	first	second	third
0	1	1.1	a
1	3	2.2	b
2	5	3.3	c
3	7	4.4	d
4	9	5.5	e
5	11	6.6	NaN
6	13	NaN	NaN

Pandas – Series vs DataFrames

- Series is 1-D whereas a DataFrame is 2-D.
- A one column DataFrame can have a name for that one column but a Series cannot have a column name.
- Each column of a DataFrame can be converted to a series.

Pandas – DataFrames – Example1

- Create a dataframe using a list

```
>> import pandas as pd
>> strlist = [‘I’, ‘love’, ‘data’, ‘mining’]

>> df = pd.DataFrame(strlist)
>> print(df)
```

	0
0	I
1	love
2	data
3	mining

Pandas – DataFrames – Example2

```
>> import pandas as pd
>> data = {'Name':[‘John', ‘William', ‘Ian', ‘Noah’],
                'Age’:[12, 15, 13, 12]}

df = pandas.DataFrame(data)
print(df)
```

	Name	Age
0	John	12
1	William	15
2	Ian	13
3	Noah	12

Pandas – Dataframes

- Given a small dataset, create a dataframe and print only two columns Name and Address.

Name	Age	Address	Qualification
John	24	New York	MS
Jim	25	Arizona	BS
Ashley	22	Minnesota	BS
Aimee	23	California	MS
Jeff	27	New York	PhD

Pandas – DataFrames – nba.csv

Name	Team	Number	Position	Age	Height	Weight	College	Salary
Avery Bradley	Boston Celtics	0	PG	25	2-Jun	180	Texas	7730337
Jae Crowder	Boston Celtics	99	SF	25	6-Jun	235	Marquette	6796117
R.J. Hunter	Boston Celtics	28	SG	22	5-Jun	185	Georgia State	1148640
Jonas Jerebko	Boston Celtics	8	PF	29	10-Jun	231		5000000
Amir Johnson	Boston Celtics	90	PF	29	9-Jun	240		12000000
Jordan Mickey	Boston Celtics	55	PF	21	8-Jun	235	LSU	1170960
Kelly Olynyk	Boston Celtics	41	C	25	Jul-00	238	Gonzaga	2165160
Terry Rozier	Boston Celtics	12	PG	22	2-Jun	190	Louisville	1824360

Pandas – DataFrames

Retrieving a player's information

```
>> Import pandas as pd
>> data = pd.read_csv("nba.csv", index_col="Name")
>> first = data.loc["Avery Bradley"]
>> second = data.loc["R.J. Hunter"]

>> print(first)
>> print(second)
```

```
Team         Boston Celtics
Number                    0
Position                 PG
Age                      25
Height                  6-2
Weight                  180
College               Texas
Salary          7.73034e+06
Name: Avery Bradley, dtype: object
```

```
Team         Boston Celtics
Number                   28
Position                 SG
Age                      22
Height                  6-5
Weight                  185
College       Georgia State
Salary          1.14864e+06
Name: R.J. Hunter, dtype: object
```

Pandas – DataFrames

- Retrieving a single column

```
>> Import pandas as pd
>> data = pd.read_csv("nba.csv", index_col="Name")

>> first = data["Age"]
```

```
Jonas Jerebko               29.0
Amir Johnson                29.0
Jordan Mickey               21.0
Kelly Olynyk                25.0
Terry Rozier                22.0
Marcus Smart                22.0
Jared Sullinger             24.0
Isaiah Thomas               27.0
Evan Turner                 27.0
James Young                 20.0
Tyler Zeller                26.0
Bojan Bogdanovic            27.0
Markel Brown                24.0
Wayne Ellington             28.0
Rondae Hollis-Jefferson     21.0
Jarrett Jack                32.0
Sergey Karasev              22.0
Sean Kilpatrick             26.0
Shane Larkin                23.0
Brook Lopez                 28.0
Chris McCullough            21.0
Willie Reed                 26.0
Thomas Robinson             25.0
Henry Sims                  26.0
Donald Sloan                28.0
Thaddeus Young              27.0
                            ...
Al-Farouq Aminu             25.0
Pat Connaughton             23.0
Allen Crabbe                24.0
Ed Davis                    27.0
```

Pandas – DataFrames

- How can we print the entire dataframe?

```
for i, j in df.iterrows():
        print(i, j)
```

Pandas – Missing values

- To check if there are any missing values

```
import pandas as pd
import numpy as np
dict1 = {'First Score':[100, 90, np.nan, 95],
        'Second Score': [30, 45, 56, np.nan],
        'Third Score':[np.nan, 40, 80, 98]}
df = pd.DataFrame(dict1)
df.isnull()
```

Pandas – Fill Missing Values

```
dict1 = {'First Score':[100, 90, np.nan, 95],
 'Second Score': [30, 45, 56, np.nan],
 'Third Score':[np.nan, 40, 80, 98]}

df = pd.DataFrame(dict1)
df.fillna(0)
```

Pandas – Drop the rows with missing values

```
dict1 = {'First Score':[100, 90, np.nan, 95],
 'Second Score': [30, 45, 56, np.nan],
 'Third Score':[np.nan, 40, 80, 98]}

df = pd.DataFrame(dict1)

df.dropna()
```

Pandas – ffill()

- Missing values are replaced with the previous row’s column value

College
Texas
Marquette
Boston University
Georgia State
No College
No College
LSU
Gonzaga

College
Texas
Marquette
Boston University
Georgia State
Georgia State
Georgia State
LSU
Gonzaga

- nba["College"].fillna(method ='ffill', inplace = True)

Pandas – ffill()

- We can set a limit (by using *limit*) on successful replacement of NaN values.

College
Texas
Marquette
Boston University
Georgia State
No College
No College
LSU
Gonzaga

College
Texas
Marquette
Boston University
Georgia State
Georgia State
NaN
LSU
Gonzaga
Louisville

- nba["College"].fillna(method ='ffill', limit = 1, inplace = True)

Pandas – Groupby()

- It is used to split the data into groups based on some criteria.
- For example, use the nba.csv to group the data based on the “Team”

```
>> import pandas as pd
>> df = pd.read_csv("nba.csv")
>> df
>> gbdata = df.groupby(‘Team’)
>> gbdata.first()
>> gbmean = df.groupby(‘age’).mean()
>> gbmean
```

Summary of the dataframe

```
>> print(df.info())
```

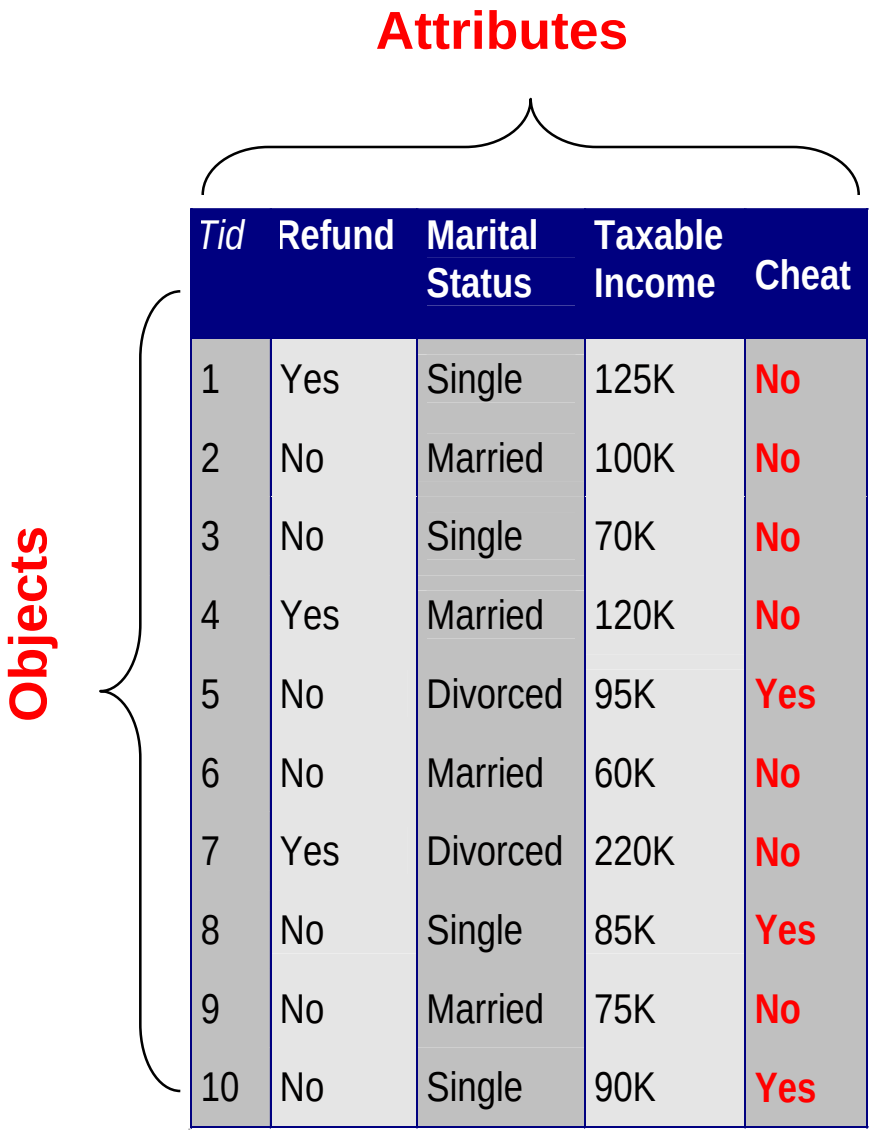
Pandas – Class Exercise

1. How can we retrieve a row by their index number? For example, index=3?
2. How do you find the number of rows and number of columns in the dataframe?
3. For nba.csv dataset, find the number of rows that have missing values.
4. Replace all the missing values in nba.csv with 0.

What is data?

- Collection of **data objects** and their **attributes**
- According to Tan et al.,

- An **attribute** is a property or characteristic of an object
  - Also known as variable, field, characteristic, dimension, or feature
- A collection of attributes describe an **object**
  - Also known as tuple, record, point, case, sample, etc.



More views of data

- Data may have parts
- The different parts of data may have relationships
- More generally, data may have structure
- Data can be incomplete