Cheatography

CodingJinxx Pandas Facts Cheat Sheet

by CodingJinxx via cheatography.com/140865/cs/30026/

Pandas Datastructures

The primary two components of pandas are the Series and DataFrame.

Dataframes and Series

Series

	apples
0	3
1	2
2	0
3	1

+ Series

	oranges
0	0
1	3
2	7
3	2

= DataFrame

	apples	oranges
0	3	0
1	2	3
2	0	7
3	1	2

A Series is One Dimensional Data

A Dataframe is N-Dimensional Data

Creating a Dataframe from Scratch

```
data = {
        'ap ples': [3, 2, 0, 1],
        'or anges': [0, 3, 7, 2]
}
purchases = pd.Dat aFr ame -
(data)
```

Turns a Dictionary into a Dataframe

Adding an Index to a Dataframe

```
purchases = pd.DataFrame(data,
index=['June', 'Robert', 'Lily',
'David'])
```

A Dataframe is a (key, value) based Data Structure

Dataframe with Index Set

	apples	oranges
June	3	0
Robert	2	3
Lily	0	7
David	1	2

Understanding your Data

```
movies_df.describe()
```

	rank	year	runtime	rating	votes	revenue_millions	metascore
count	1000.000000	1000.000000	1000.000000	1000.000000	1.000000e+03	1000.000000	936.000000
mean	500.500000	2012.783000	113.172000	6.723200	1.698083e+05	82.956376	58.985043
std	288.819436	3.205962	18.810908	0.945429	1.887626e+05	96.412043	17.194757
min	1.000000	2006.000000	66.000000	1.900000	6.100000e+01	0.000000	11.000000
25%	250.750000	2010.000000	100.000000	6.200000	3.630900e+04	17.442500	47.000000
50%	500.500000	2014.000000	111.000000	6.800000	1.107990e+05	60.375000	59.500000
75%	750.250000	2016.000000	123.000000	7.400000	2.399098e+05	99.177500	72.000000
max	1000.000000	2016.000000	191.000000	9.000000	1.791916e+06	936.630000	100.000000

movies_df.describe()

Describing individual columns
movies_df['genre'].describe()

movies_df['genre'].value_counts().head(10)

	genre
Action,Adventure,Sci-Fi	50
Drama	48
Comedy,Drama,Romance	35
Comedy	32
Drama,Romance	31
Animation,Adventure,Comedy	27
Action,Adventure,Fantasy	27
Comedy,Drama	27

movies_df.corr()

	rank	year	runtime	rating	votes	revenue_millions	metascore
rank	1.000000	-0.261605	-0.221739	-0.219555	-0.283876	-0.252996	-0.191869
year	-0.261605	1.000000	-0.164900	-0.211219	-0.411904	-0.117562	-0.079305
runtime	-0.221739	-0.164900	1.000000	0.392214	0.407062	0.247834	0.211978
rating	-0.219555	-0.211219	0.392214	1.000000	0.511537	0.189527	0.631897
votes	-0.283876	-0.411904	0.407062	0.511537	1.000000	0.607941	0.325684
revenue_millions	-0.252996	-0.117562	0.247834	0.189527	0.607941	1.000000	0.133328
metascore	-0.191869	-0.079305	0.211978	0.631897	0.325684	0.133328	1.000000

Generates a matrix describing correlation between data

Reading in Data

It's quite simple to load data from various file formats into a DataFrame. In the following examples we'll keep using our apples and oranges data, but this time it's coming from various files.

Reading CSV

```
df =
pd.read_csv('purchases.csv')
# Or with setting the index
column, needs to be set with CSV
df = pd.rea d_c sv( 'pu rch -
ase s.csv', index_ col=0)
```

Reading JSON

```
# The index is automatically set
with json
df = pd.rea d_j son ('p urc -
has es.j son')
```

Reading in SQLite

```
import sqlite3
con = sqlite 3.c onn ect ("da -
tab ase.db ")
df = pd.rea d_s ql_ que ry( " -
SELECT * FROM purcha ses ", con)
```

Setting index in post

```
df = df.set_index('index')
```

Writing Data

```
df.to_csv('new_purchases.csv')
df.to_ jso n(' new _pu rch ase -
s.j son')
df.to_ sql ('n ew_ pur cha ses',
con)
```

con being a sql connection

When we save JSON and CSV files, all we have to input into those functions is our desired filename with the appropriate file extension. With SQL, we're not creating a new file but instead inserting a new table into the database using our con variable from before.

By **CodingJinxx**

cheatography.com/codingjinxx/

Published 2nd December, 2021.
Last updated 9th December, 2021.

Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
https://apollopad.com

Cheatography

CodingJinxx Pandas Facts Cheat Sheet

by CodingJinxx via cheatography.com/140865/cs/30026/

Dataframe Slicing Extracting Selecting

Up until now we've focused on some basic summaries of our data. We've learned about simple column extraction using single brackets, and we imputed null values in a column using fillna(). Below are the other methods of slicing, selecting, and extracting you'll need to use constantly.

It's important to note that, although many methods are the same, DataFrames and Series have different attributes, so you'll need be sure to know which type you are working with or else you will receive attribute errors.

Let's look at working with columns first.

By Columns

```
genre_col = movies_df['genre']
genre_col = movies _df [[' gen -
re']]
subset = movies _df [[' genre',
'rating']]
subset.head()
```

To extract data as Dataframes from a dataframe pass a list key

By Rows

```
prom =
movies_df.loc["Prometheus"]
prom = movies _df.il oc[1]
movie_ subset = movies _df.lo -
c[' Pro met heu s': 'Sing']
movie_ subset = movies _df.il -
oc[1:4]
```

loc and iloc can be thought of as similar to Python list slicing. To show this even further, let's select multiple rows.

Conditional Selections

```
condition =
(movies_df['director'] ==
"Ridley Scott")
movies _df [mo vie s_d f[' dir -
ector'] == " Ridley
Scott"].head() # Returns a
Dataframe where Director is
Ridley Scott
movies _df [mo vie s_d f[' rat -
ing'] >= 8.6].h ead(3)
movies _df [(m ovi es_ df[ 'di -
rec tor'] == 'Chris topher
Nolan') | (movie s_d f[' dir -
ector'] == 'Ridley Scott' )].h -
ead()
# OR Simply
movies _df [mo vie s_d f[' dir -
ect or' ].i sin ([' Chr ist -
opher Nolan', 'Ridley
Scott' ])].head()
movies_df[
        ((m ovi es_ df[ 'year']
>= 2005) & (movie s_d f[' year']
<= 2010))
        & (movie s_d f[' rat -
ing'] > 8.0)
        & (movie s_d f[' rev -
enu e_m ill ions'] < movies -
_df ['r eve nue _mi lli -
ons '].q ua nti le( 0.25))
]
```

Similar to isnull(), this returns a Series of True and False values: True for films directed by Ridley Scott and False for ones not directed by him.

Applying Functions to Data

```
def rating_function(x):
       if x >= 8.0:
                return " goo d"
        else:
                return " bad "
movies _df ["ra tin g_c ate -
gor y"] = movies _df ["ra tin -
g"].a pp ly( rat ing _fu nction)
```

Viewing Data

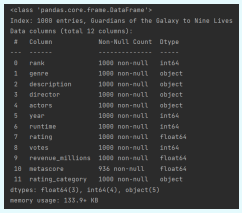
```
movies_df.head() # Print first 5
rows
movies _df.he ad(10) # Print
first 10 rows
movies _df.tail() # Print last 5
Rows
movies _df.ta il(2) # Print last
2 rows
```

Previews start or ends of dataframes

Getting metadata from a dataframe

```
movies_df.info() # Output info
regarding datatypes
movies _df.shape # Outputs
number of rows and columns
```

Dataframe Info



df.info()

By **CodingJinxx**

cheatography.com/codingjinxx/

Published 2nd December, 2021.

Last updated 9th December, 2021.

Sponsored by **ApolloPad.com**

Everyone has a novel in them. Finish Yours!

https://apollopad.com

Cheatography

CodingJinxx Pandas Facts Cheat Sheet

by CodingJinxx via cheatography.com/140865/cs/30026/

Cleaning Data

```
### Duplicate Data Handling
temp_df = temp_d f.d rop _du -
pli cates() # Drops duplicate
data
# Or
temp_d f.d rop _du pli cat es( -
inp lac e=True)
# Keeping duplicate data
temp_d f.d rop _du pli cat es( -
inp lac e=True, keep=True)
### Renaming Columns
movies _df.re nam e(c olu mns={
                'Ru ntime
(Minut es)': 'Runtime',
                'Re venue
(Milli ons)': 'Reven ue_ mil -
lions'
       }, inplac e=True)
# Can also be set directly
movies _df.co lumns = ['rank',
'genre', 'descr ipt ion',
'direc tor', 'actors', 'year',
'runtime',
                                -
          'ra ting', 'votes',
'reven ue_ mil lions', 'metas -
core']
# Setting column names lowercase
movies _df.co lumns = [col.l -
ower() for col in movies_df]
### Handling Null Values
movies _df.is null() # Returns a
column with either True or False
for null or not
movies _df.is nul l().sum() #
Returns a count of all null
entries
# Removing Nulls
```

Cleaning Data (cont)

```
> movies_df.dropna()
# Specify the axis to drop against
movies_df.dropna(axis=1)
```

Plotting with Matplotlib

Another great thing about pandas is that it integrates with Matplotlib, so you get the ability to plot directly off DataFrames and Series. To get started we need to import Matplotlib (pip install matplotlib):

Scatter plot

```
import matplotlib.pyplot as plt
plt.rc Par ams.up dat e({ 'fo -
nt.s ize': 20, 'figur e.f igs -
ize': (10, 8)}) # set font and
plot size to be larger
movies _df.pl ot( kin d=' sca -
tter', x='rat ing', y='rev enu -
e_m ill ions', title= 'Re venue
(millions) vs Rating');
```

Histogram plot

```
movies_df['rating'].plot(kind='hist',
title='Rating');
```

Boxplot

```
movies_df['rating'].plot(kind="box
```

Boxplot



By **CodingJinxx**

cheatography.com/codingjinxx/

Published 2nd December, 2021.
Last updated 9th December, 2021.

Sponsored by **ApolloPad.com**
Everyone has a novel in them. Finish Yours!
https://apollopad.com