Common Python (PANDAS) commands

|  |  |
| --- | --- |
| import pandas as pd |  |
| import numpy as np |  |
| import seaborn as sns |  |
| import matplotlib.pyplot as plt |  |
| df = pd.read\_csv("dfname.csv") | Import a csv file |
| df=pd.read\_excel("filename.xlsx", sheet\_name="tabname") | Imports a spread sheet but only the relevant tab |
| df.head(x) | Displays the top x rows of data |
| df.tail(x) | Displays the last x rows of data |
| df.describe() | Gives a summary of the data, along with mean, corr, std etc |
| df.describe(include="object") | Gives different detail including text data eg names, addresses, dates |
| df.dtypes | Gives the type of data for each row |
| df.info() | Summary of the columns and data types |
| df.sort\_values(by = 'Column') | Eg:  df.sort\_values(by='first\_name', ascending = False).head() |
| df.sort\_values(by=column, inplace=True) | Inplace will save the data in order |
| df.isnull().sum() or *df.isna().sum()* | Shows blanks or NAs etc in data frame |
| df.to\_csv('dfname.csv') | Saves the df with a name chosen |
| df[column] > 5.0 | Filters the column to just these returns |
| df[store\_id\_filter & first\_name\_filter] | Filters to store id and first name |
| customer[(customer['store\_id'] == 2) & (customer['first\_name'].str.upper() == 'WILLIE')] | Will show customers from store No2 whose first name is Willie |
| df[df[column] > 5.0] | Will return the table in this format after filtering to responses >5 |
| df['Year'] = df['year'].dt.year | Adds a new column called Year using year data but as year only |
| df['column'].unique() | Lists the answers in the relevant column |
| df[df[‘column’]=='x'] | Lists the row where the answer is ‘x’ |
| df[df.isnull().any(axis=1)]?? | Shows all rows with one or more nulls (may be wrong) |
| df[df['column']=='x'].describe() | Summarises all rows where answer is x |
| df.dropna() | Removes any rows with an NA or Nan or Null etc |
| clean\_df= df.dropna() | Creates a new df called clean\_df without nulls |
| df.fillna({'age': df['age']. mean(),  'satisfaction': df['satisfaction'].mean()}) | replaces the missing ages with the average age of the others and then the same with any other columns (eg satisfaction). Only if not object |
| df['column'] = df['column'].astype("U") | Changes column data type to string (object) |
| df[['Col1', 'Col2']] | Selects particular columns from the df  The syntax for choosing a Column from the DataFrame and returning it as a Series is:  DataFrame['Column']  The syntax for choosing more than one column from the DataFrame, returning a DataFrame is:  DataFrame[['Column 1', 'Column 2']]  The syntax for choosing a Column from the DataFrame and returning a DataFrame is:  DataFrame[['Column']]  customer[['first\_name', 'last\_name']] |
| df.loc[78, :] | Retrieves data from row 78 |
| df.loc[0:78] | Retrieves data from rows 0 to 78 |
| df.loc[16:] | Retrieves data from rows 16 to end |
| df[[‘Col1’, Col2]].groupby(‘Col1’).mean() | Lists Col1 then Col2 (mean figures) |
| df1.pivot\_table(index='Default', columns = 'Term',) | From    To |
| df.plot(x = 'Col1', y = 'Col2') | Create a graph with the basic information |
| df.plot(kind = 'scatter', x = 'Col1', y = 'Col2') | Creates a scatter plot in pandas (Matplotlib). Can change kind to ‘hist’ |
| newname=pd.concat([df1, df2], axis=1) | Joins two dataframes together, second after the first. Axis 1 is by columns, axis 0 by rows |
| search=df[df['column']=='search string'] | Finds all columns where the search string is the entry |
| df[‘column’].round() | Rounds the result to x decimal places (blank = zero) |
| df[‘column’] = df[‘column’]. astype(int) | If the data is date based but we only want the year as an integer |
| df['column].mean() | Gives the mean result of the data |
| mean\_price = df['Y house price of unit area'].mean()  df['above\_mean\_price']= np.where(df['Y house price of unit area']>mean\_price, 'yes', 'no')  df | Adds a new column with the Boolean result |
| cols = ['User\_id', 'item\_id', 'rating', 'timestamp']  user\_ratings = pd.read\_csv('u.data', names = cols, sep= '\t') | Creates a list of column headers ready to import a data frame  Then data is imported from an existing file to show |