**Berkeley AI/ML Certificate Program**

<https://student.emeritus.org>

Eposta / H..2[0-9]{4}

kaggle - sirket paypalinc google ile.

“there is an instance of anaconda navigator already running”

killall python

**Git:**

cache the given record in your computer to remembers the token:

$ git config --global credential.helper cache

If needed, anytime you can delete the cache record by:

$ git config --global --unset credential.helper

$ git config --system --unset credential.helper

**Token:** ghp\_BnTo8mZxi60GmqiUwGJBkYEsKgj2bL0EA5ds

Participants must complete all assignments by **September 14, 2022,** to receive a certificate of completion for this program.

* Introduce yourself
* What attracted you to this program?
* What is your experience with ML or AI?
* What are you hoping to do professionally with ML and AI skills?

I am an accomplished senior database architect and software engineer with over 20 years of experience in all aspects of SQL and NoSQL databases, database design and implementation as well low latency, high throughput software development by Kafka streaming.

Over the recent years, I have seen increasing demand for ML in data-centric solutions.

I have limited exposure to ML with BigQuery ML.

I would like to understand ML and AI better and use them on our incoming projects.

<https://www.linkedin.com/in/aykanerdenizmenli>

Database subject matter expert and liaison to all departments with cross-functional teams. Analytical thinker, team player, influencer, proven track record of solving for large, complex problems while balancing both tactical and strategical success.

An outline of the BH-PCMLAI program calendar

|  |  |  |  |
| --- | --- | --- | --- |
| **Module #** | **Module Title** | **Week #** | **Date** |
| **0** | Program Orientation | 0 | **Wednesday, March 02, 2022** |
| **1** | Introduction to Machine Learning | 1 | **Wednesday, March 09, 2022** |
| **2** | Fundamentals of Machine Learning | 2 | **Wednesday, March 16, 2022** |
| **3** | Introduction to Data Analysis | 3 | **Wednesday, March 23, 2022** |
| **4** | Fundamentals of Data Analysis | 4 | **Wednesday, March 30, 2022** |
| **5** | Practical Application 1 | 5 | **Wednesday, April 06, 2022** |
| Break Week |  |  | **Wednesday, April 13, 2022** |
| **6** | Clustering and Principal Component Analysis (PCA) | 6 | **Wednesday, April 20, 2022** |
| **7** | Linear and Multiple Regressions | 7 | **Wednesday, April 27, 2022** |
| **8** | Feature Engineering and Overfitting | 8 | **Wednesday, May 04, 2022** |
| **9** | Model Selection and Regularization | 9 | **Wednesday, May 11, 2022** |
| **10** | Time Series Analysis and Forecasting | 10 | **Wednesday, May 18, 2022** |
| **11** | Practical Application 2 | 11 | **Wednesday, May 25, 2022** |
| Break Week |  |  | **Wednesday, June 01, 2022** |
| **12** | Classification and k-Nearest Neighbors (KNN) | 12 | **Wednesday, June 08, 2022** |
| **13** | Logistic Regression | 13 | **Wednesday, June 15, 2022** |
| **14** | Decision Trees | 14 | **Wednesday, June 22, 2022** |
| **15** | Gradient Descent and Optimization | 15 | **Wednesday, June 29, 2022** |
| **16** | Support Vector Machines (SVMs) | 16 | **Wednesday, July 06, 2022** |
| **17** | Practical Application 3 | 17 | **Wednesday, July 13, 2022** |
| Break Week |  |  | **Wednesday, July 20, 2022** |
| **18** | Natural Language Procession (NLP) | 18 | **Wednesday, July 27, 2022** |
| **19** | Recommendation Systems | 19 | **Wednesday, August 03, 2022** |
| **20** | Capstone 1 | 20 | **Wednesday, August 10, 2022** |
| **21** | Ensemble Techniques (GBM, XGB, and Random Forest) | 21 | **Wednesday, August 17, 2022** |
| **22** | Deep Neural Networks 1 | 22 | **Wednesday, August 24, 2022** |
| **23** | Deep Neural Networks 2 | 23 | **Wednesday, August 31, 2022** |
| **24** | Capstone 2 | 24 | **Wednesday, September 07, 2022** |

**Module 1**

Machine learning is the study of all of the different ways in which models can be built from data.

AI -

Black Box:

Vector

Loss function: square of difference of actual result is y, predicted result is ŷ

The function which takes the actual result y and our predicted result ŷ and scores it as the square of the difference is known as the loss function L.

Probability and statistics

actual result is y, predicted result is ŷ

Y generic, y sample of output. Y ~ py Y has distribution py. [y]10 data set of sample size 10 = [y1, y2,…y10]

Discrete or continuous distribution

*You are correct! The answer “Non-deterministic system*” *is correct because such a system exhibits a different behavior every time it is sampled.*

*You are correct! The terms ‘distribution’ and ‘probability density function’ refer to the same thing.*

*That is correct! The answer “discrete distribution” is correct because in this system, the data can be only certain values such as integers, (0,1,2, and 3, for example).*

*You are correct! The answer “Continuous distribution“ is correct because in continuous distribution, the random variable X can have any value since there are infinite values X can take.*

*You are correct! The answer “True” is correct because the capital letter Y is used to denote the generic output of a system, and the samples of the output are denoted with a small letter y.*

*You are correct! The answer “Y is distributed according to a distribution Fy” is correct because Y is sampled using the function Fy.*

*You are correct! The answer “12” is correct because the subscript number in the question denotes how many unique outputs to return.*

E[Y] expected value, expectation, mean of Y or Py. Mean is avg, median is middle number

Small variance, large variance:graph will be wider and shorter.

stddev

*You are correct! The answer “One” is correct because the sum of the entire area under the curve will always equal one. Any area between two vertical lines running through the distribution will always be less than one.*

*You are correct! The answers “E[Y]” and “*μy*” are correct because these are the correct representations for the expected value of Y.*

*You are correct! The answer “True” is correct because although the mean and median measure different things, they can be the same value.*

*You are correct! The answer “True” is correct because as n grows, there are more data points to base the average (mean) calculations on, and therefore, it becomes a better representation of the expected value.*

*You are correct! The answer “True” is correct because variance is a measure of how widely distributed the values are. The larger the variance grows, the wider the graph will be.*

*You are correct! The answer “The square root of the variance” is correct because standard deviation is the square root of the variance.*

*You are correct! The answer “*𝛔y” *is correct because this is the correct symbol for the standard deviation.*

*The answer “*.loc[slice] *when the index is integers” is incorrect because when the index is integer the function*.loc[]*is preferable.*

Jupyter Notebook, key operations in pandas, including how to use arrays, create dataframes from dictionaries, and change indexes.

The statement .loc[[0:3],[’column1’,’column4’]] will return indexes 0 to 3 and columns from column1 and column4.

locinfo = data.loc[0]

teamdata = data.loc[data.Name == 'LeBron James', 'Name':'Team']

fr = data.iloc[0]

teamdata = data.loc[data.Name == 'LeBron James', 'Name':'Team']

sr = data.iloc[1]

lr = data.iloc[-1]

fc = data.iloc[:,0]

sc = data.iloc[:,1]

lc = data.iloc[:,-1]

firstname = data['Name'].str.split(expand=**True**).iloc[0,0]

firstname1 = data['Name'].str.split().str[1].iloc[0]

lastname = data['Name'].str.split(expand=**True**).iloc[0,1]

lastname1 = data['Name'].str.split().str[1].iloc[0]

**Fishing**

Almost any equipment or gear used for fishing can be called fishing tackle. Some examples are [hooks](https://en.wikipedia.org/wiki/Fishing_hook), [lines](https://en.wikipedia.org/wiki/Fishing_line), [sinkers](https://en.wikipedia.org/wiki/Fishing_sinker), [floats](https://en.wikipedia.org/wiki/Fishing_float), [rods](https://en.wikipedia.org/wiki/Fishing_rod), [reels](https://en.wikipedia.org/wiki/Fishing_reel), [baits](https://en.wikipedia.org/wiki/Fishing_bait), [lures](https://en.wikipedia.org/wiki/Fishing_lure), [spears](https://en.wikipedia.org/wiki/Spearfishing), [nets](https://en.wikipedia.org/wiki/Fishing_net), [gaffs](https://en.wikipedia.org/wiki/Fishing_gaff), [traps](https://en.wikipedia.org/wiki/Fishing_trap), [waders](https://en.wikipedia.org/wiki/Waders_(footwear)) and tackle boxes.

Why a fish bites a baited hook or lure involves several factors related to the sensory physiology, behaviour, feeding ecology, and biology of the fish as well as the environment and characteristics of the bait/hook/lure.[28] There is an intricate link between various fishing techniques and knowledge about the fish and their behaviour including [migration](https://en.wikipedia.org/wiki/Fish_migration), [foraging](https://en.wikipedia.org/wiki/Forage_fish) and [habitat](https://en.wikipedia.org/wiki/Oceanic_habitats). The effective use of fishing techniques often depends on this additional knowledge.[29] Some fishers follow [fishing folklores](https://en.wikipedia.org/wiki/Solunar_theory) which claim that fish feeding patterns are influenced by the position of the sun and the moon.

Fishing is another example of “black box” even though even though there are specific fishing tackles are used to target certain fish kinds like lines by weight, hook size, sinkers to adjust how close to bottom, floats to adjust how close to surface, baits and lures to target certain population. There are also other factors like time of day, environmental factor, feeding behavior, fish sensory, as well as the sun and moon positions to affect a fish bite. The end result is still not predictable.

**Datasets**

[**https://student.emeritus.org/courses/4765/discussion\_topics/257482**](https://student.emeritus.org/courses/4765/discussion_topics/257482)

* A short description of the content of the data
* One or two possible business uses of the data
* The location (URL) of the dataset
* A summary of the characteristics of the dataset using the describe() function in pandas

<https://www.kaggle.com/prasertk/internet-broadband-and-mobile-speeds-by-country>

<https://www.kaggle.com/vivovinco/san-francisco-incident-reports-2018present>

**Context**

The Nobel Prize is a set of annual international awards bestowed in several categories by Swedish and Norwegian institutions in recognition of academic, cultural, or scientific advances. The will of the Swedish chemist, engineer, and industrialist Alfred Nobel established the five Nobel prizes in 1895. The prizes in Chemistry, Literature, Peace, Physics, and Physiology or Medicine were first awarded in 1901. The prizes are widely regarded as the most prestigious awards available in their respective fields.

Between 1901 and 2020, the Nobel Prizes and the Prize in Economic Sciences were awarded to 949 people . The Nobel Prize is an international award administered by the Nobel Foundation in Stockholm, Sweden, and based on the fortune of Alfred Nobel, Swedish inventor and entrepreneur. In 1968, Sveriges Riksbank established The Sveriges Riksbank Prize in Economic Sciences in Memory of Alfred Nobel, founder of the Nobel Prize. Each Prize consists of a medal, a personal diploma, and a cash award.

A person or organization awarded the Nobel Prize is called Nobel Laureate. The word "laureate" refers to being signified by the laurel wreath. In ancient Greece, laurel wreaths were awarded to victors as a sign of honor.

**Dataset Description**

This dataset includes a record for every individual that was awarded the Nobel Prize between 1901 and 2020.

**Business Use Case**

The analysis can be used to identify which country invest in science, subject, and/or what institutions are more credible.

**URL**

<https://www.kaggle.com/bahramjannesarr/nobel-prize-from-1901-till-2020>

**Dataset Characteristic**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | year | share | age | age\_get\_prize |
| count | 923.000000 | 923.000000 | 923.000000 | 923.000000 |
| mean | 1971.102925 | 2.020585 | 79.346696 | 59.823402 |
| std | 33.606425 | 0.943415 | 11.134606 | 12.571109 |
| min | 1901.000000 | 1.000000 | 23.000000 | 17.000000 |
| 25% | 1947.000000 | 1.000000 | 73.000000 | 51.000000 |
| 50% | 1977.000000 | 2.000000 | 80.000000 | 60.000000 |
| 75% | 2000.000000 | 3.000000 | 87.500000 | 69.000000 |
| max | 2019.000000 | 4.000000 | 103.000000 | 97.000000 |

**Content**

+120.000 rows and 10 columns. Columns' description are listed below.

* DateTime : Datetime in "dd.mm.yyyy hh:mm" format
* Temperature : Temperature at 2 m in °C
* Sunshine Duration : Sunshine duration in min
* Shortwave Radiation : Shortwave radiation in W/m²
* Relative Humidity : Relative Humidity at 2 m in %
* Mean Sea Level Pressure : Mean Sea Level Pressure (MSL) in hPa
* Soil Temperature : Soil temperature at 0-10 cm down in °C
* Soil Moisture : Soil moisture at 0-10 cm down in m³/m³
* Wind Speed : Wind speed at 10 m in km/h
* Wind Direction : Wind direction at 10 m in degrees

**Acknowledgements**

Data from [Meteoblue](https://www.meteoblue.com/). Image from [Anadolu Agency](https://www.aa.com.tr/en/asia-pacific/turkeys-gallipoli-anniversary-marked-in-pakistan/1089591).

If you're reading this, please upvote.

**Dataset Description**

This dataset contains hourly weather data between 2008 and 2021 for Gallipoli, Turkey.

**Business Use Case**

Throughout day/week/month/year wind direction and wind speed can be analyzed for possible wind power installations.

This information is also useful for construction business to decide regarding building strength against wind.

**URL**

<https://www.kaggle.com/vivovinco/hourly-weather-data-in-gallipoli-20082021>

**Dataset Characteristic**

|  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Temperature | Sunshine Duration | Shortwave Radiation | Relative Humidity | Mean Sea Level Pressure | Soil Temperature | Soil Moisture | Wind Speed | Wind Direction |  |
| count | 122733.000000 | 122733.000000 | 122733.000000 | 122733.000000 | 122733.000000 | 122733.000000 | 122733.000000 | 122733.000000 | 122733.000000 |
| mean | 15.670196 | 21.084680 | 202.333411 | 69.683924 | 1015.073141 | 16.336311 | 0.210705 | 19.391254 | 108.071393 |
| std | 8.192027 | 27.030718 | 272.710423 | 19.079325 | 6.795215 | 8.054375 | 0.067094 | 11.141315 | 91.270893 |
| min | -13.800000 | 0.000000 | 0.000000 | 13.000000 | 981.300000 | -5.200000 | 0.102000 | 0.000000 | 0.310000 |
| 25% | 9.700000 | 0.000000 | 0.000000 | 55.000000 | 1010.600000 | 10.100000 | 0.153000 | 10.400000 | 44.090000 |
| 50% | 15.500000 | 0.000000 | 16.020000 | 72.000000 | 1014.400000 | 15.900000 | 0.214000 | 18.300000 | 60.360000 |
| 75% | 21.400000 | 56.000000 | 379.140000 | 86.000000 | 1019.200000 | 22.400000 | 0.263000 | 26.800000 | 195.020000 |
| max | 40.800000 | 60.000000 | 950.520000 | 100.000000 | 1043.600000 | 40.100000 | 0.433000 | 93.300000 | 360.000000 |

<https://www.kaggle.com/andrewmvd/citymapper-mobility-index>

<https://www.kaggle.com/susant4learning/crime-in-los-angeles-data-from-2020-to-present>

**Context**

This Data set contains various information about a set of cars that were manufactured with set of factory parameters like cylinder size, number of cylinders, fuel consumption, Carbon dioxide emissions etc.

**Content**

Feature set includes model, make, vehicle type, engine size,Transmission, fuel consumption etc. The target set is Carbon dioxide Emission.

**Inspiration**

The inspiration was to learn and help understand the concept behind data science's basics.

**Dataset Description**

This dataset contains make, model, vehicle type, engine size, and carbon dioxide emission.

**Business Use Case**

This carbon footprint data can be used for consumer awareness about what car companies, make, model are more environmentally friendly.

**URL**

<https://www.kaggle.com/prathamtripathi/co2-emissions-by-cars-in-canada>

**Dataset Characteristic**

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
|  | MODEL | ENGINE\_SIZE | CYLINDERS | FUEL\_CONSUMPTION\* | Unnamed: 9 | Unnamed: 10 | Unnamed: 11 | CO2\_EMISSIONS |
| count | 679.0 | 679.000000 | 679.000000 | 679.000000 | 679.000000 | 679.000000 | 679.000000 | 679.000000 |
| mean | 2001.0 | 3.252577 | 5.798233 | 14.591900 | 10.613844 | 12.802798 | 23.107511 | 293.656848 |
| std | 0.0 | 1.203751 | 1.531073 | 3.025654 | 2.357724 | 2.685590 | 5.308083 | 60.372456 |
| min | 2001.0 | 1.000000 | 3.000000 | 4.900000 | 4.000000 | 4.500000 | 14.000000 | 104.000000 |
| 25% | 2001.0 | 2.200000 | 4.000000 | 12.700000 | 9.000000 | 11.000000 | 19.000000 | 253.000000 |
| 50% | 2001.0 | 3.000000 | 6.000000 | 14.300000 | 10.100000 | 12.300000 | 23.000000 | 283.000000 |
| 75% | 2001.0 | 4.200000 | 6.000000 | 16.650000 | 12.500000 | 14.850000 | 26.000000 | 340.000000 |
| max | 2001.0 | 8.000000 | 12.000000 | 23.200000 | 18.000000 | 20.800000 | 63.000000 | 478.000000 |

* A short description of the content of the data
* One or two possible business uses of the data
* The location (URL) of the dataset
* A summary of the characteristics of the dataset using the describe() function in pandas

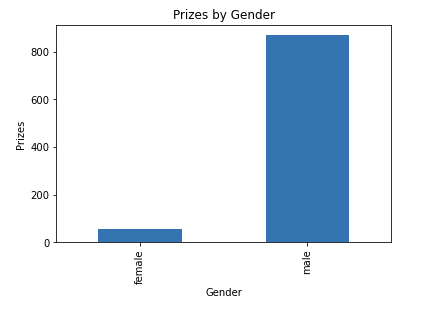
Using personally-sourced data (see Discussion 1.2), create a visualization using the pandas plot() function (scatter, bar chart, line) and post your most interesting results. For the visualization that you selected, please describe why you chose that plot type and any transformations to the data that you had to make in order to generate your visualization. Additionally, please describe any interesting trends (i.e., increased monthly sales) that you observe in your results.

**Noble Prizes**

I analyzed Nobel Prize Winners dataset by gender, category, age, institutions and number of institutions in country by using kaggle dataset: <https://www.kaggle.com/bahramjannesarr/nobel-prize-from-1901-till-2020>.

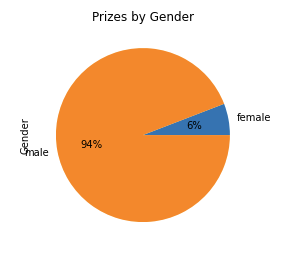
Data reveals there is a huge gap between genders, majority of Noble Prize winners are male by using bar chart:

data.groupby('gender').size().plot(kind='bar',title='Prizes by Gender', xlabel='Gender', ylabel='Prizes')



It is an uneven distribution among gender, only 6% is female by using pie chart for better visibility:

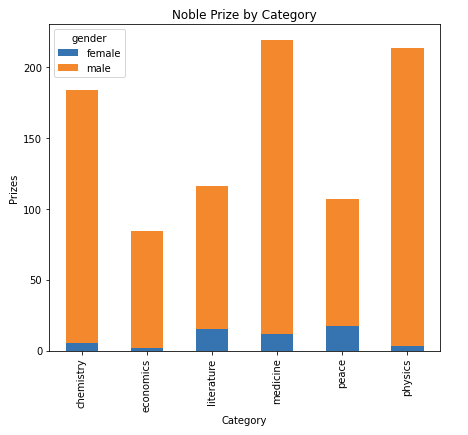
data.groupby('gender').size().plot(kind='pie', title='Prizes by Gender', ylabel='Gender', autopct='%1.0f%%')



Further break gender down by category reveals that females participate more in Peace, Medicine and Literature categories and less in Chemistry, Economics and Physics categories. I chose the stacked bar representation for comparison:

cnt=data.groupby(['category','gender']).size().unstack()

cnt.plot(kind='bar',stacked=True, title='Noble Prize by Category', xlabel='Category', ylabel='Prizes', figsize=(7,6))



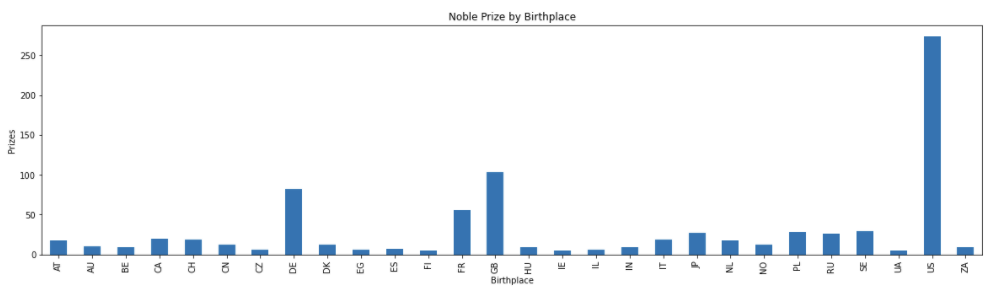
Also, there are more participants in Medicine, Physics and Chemistry as they tend to be team effort than the rest.

Birthplace of Noble Prize winners concentrate only in a few countries where majority in US followed by United Kingdom, Germany and France:

bornset=data.groupby('born\_country\_code', as\_index=False).size()

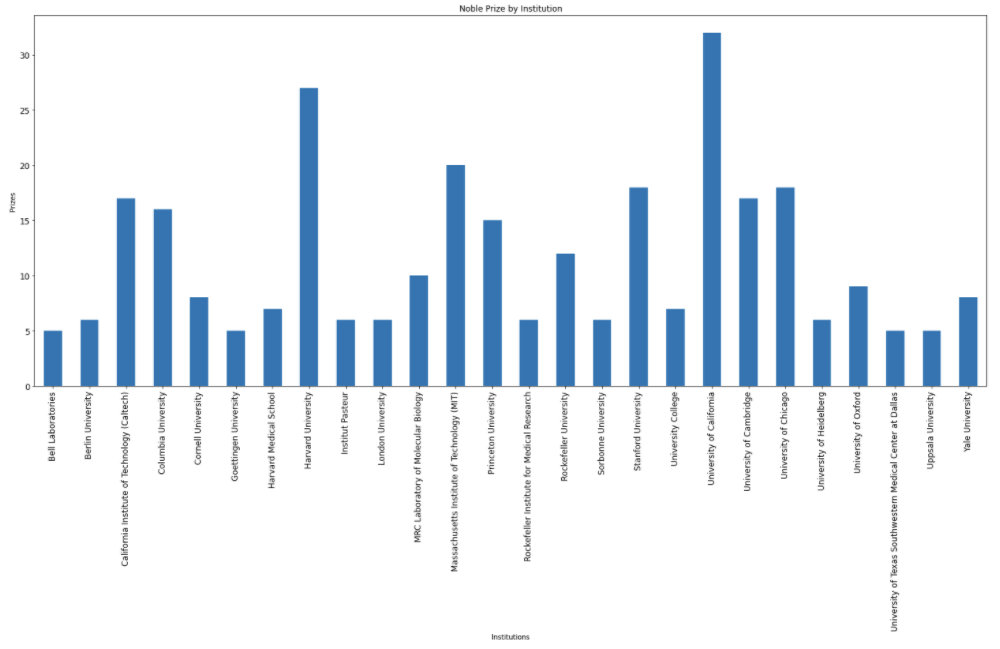
bornset=bornset[bornset['size'] > 4]

bornset.plot(kind='bar', figsize=(20,5), x='born\_country\_code', title='Noble Prize by Birthplace', xlabel='Birthplace', ylabel='Prizes', legend=False)



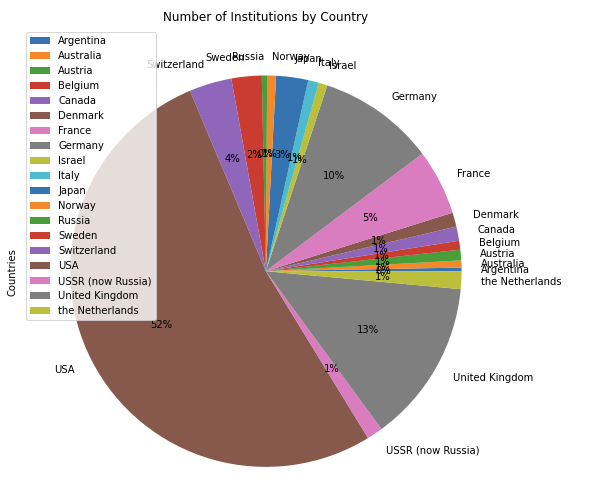
As well as top institutions hosted noble prize winners, University of California is at the top:

subset[subset['size'] > 4].plot(kind='bar', figsize=(25,10), x='name\_of\_university', title='Noble Prize by Institution', xlabel='Institutions', ylabel='Prizes', fontsize=12, legend=False)



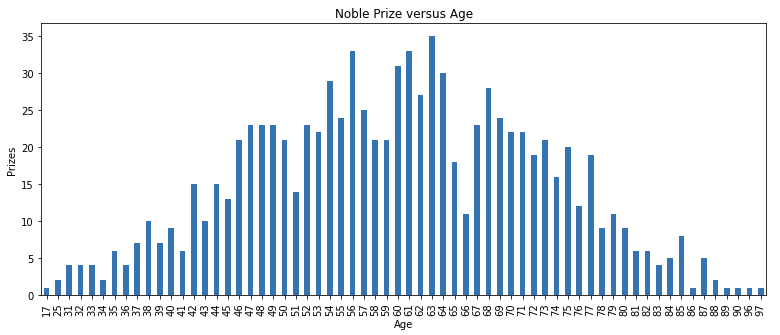
These information yield where those institutions concentrated. More than half of prize winners are located in US followed by United Kingdom, Germany and France just like their birthplace which points those brilliant folks do not migrate:

subset.plot(kind='pie', title='Number of Institutions by Country', x='country\_of\_university', y='size', ylabel='Countries', autopct='%1.0f%%', figsize=(9, 9), fontsize=10, rot=180)



Final look at the data is age breakdown by prize, not surprisingly peak at between 54 and 64:

data.groupby('age\_get\_prize').size().plot(kind='bar', figsize=(13,5), title='Noble Prize versus Age', xlabel='Age', ylabel='Prizes'



**Module 2**

Here are a few helpful downloads for this module:

* [Video Transcripts](https://student.emeritus.org/courses/4765/files/2878716?wrap=1)
* [Download Video Transcripts](https://student.emeritus.org/courses/4765/files/2878716/download?download_frd=1)
* [Quick Reference Guide](https://student.emeritus.org/courses/4765/files/2878719?wrap=1)

Loc, scale

Dot . & Tab gives all available functions!

from scipy.stats import uniform

from scipy.stats import norm

import matplotlib.pyplot as plt

import numpy as np

Gaussian - normal - bell curve

mu\_X

**Coding\_Activity\_2.1**

dist1 = uniform(loc=10, scale=3)

dist1\_mean = dist1.mean()

dist1\_var = dist1.var()

dist1\_std = dist1.std()

x = np.linspace(9,14,100)

plt.plot(x,dist1.pdf(x))

p\_9 = dist1.pdf(9)

p\_11 = dist1.pdf(11)

p\_11\_or\_12\_ = dist1\_.pdf(11) + dist1\_.pdf(12)

p\_less\_than\_12 = dist1.cdf(12)

p\_between\_11\_and\_13 = dist1.cdf(13) - dist1.cdf(11)

p\_greater\_than\_12 = 1 - dist1.cdf(12)

dist1 = norm(loc=5, scale=10)

**for** i **in** range(1,501,1):

*#generate samples*

*#remember the random\_state*

samples = dist1.rvs(i,random\_state = 22)

*#find sample mean*

smean = np.mean(samples)

*#append mean to sample\_means*

sample\_means.append(smean)

ans3 = sample\_means[400] - 10 < .1

samples\_30\_or\_more = sample\_means[30:]

samples\_mean = np.mean(samples\_30\_or\_more)

samples\_std = np.std(samples\_30\_or\_more)

gauss\_dist = norm(loc = 5, scale = 10)

sample\_means\_gauss = []

**for** i **in** range(30,501,1):

*#generate samples*

*#remember the random\_state*

samples = gauss\_dist.rvs(i,random\_state = 22)

*#find sample mean*

smean = np.mean(samples)

*#append mean to sample\_means*

sample\_means\_gauss.append(smean)

gauss\_mean = np.mean(sample\_means\_gauss)

gauss\_standard\_deviation = np.std(sample\_means\_gauss)

nyc\_salary\_data = pd.read\_csv('data/nyc\_salaries.csv')

ans\_2 = nyc\_salary\_data['base\_salary'].mean()

ans\_3 = nyc\_salary\_data['base\_salary'].median()

first\_quartile = nyc\_salary\_data['base\_salary'].quantile(0.25)

third\_quartile = nyc\_salary\_data['base\_salary'].quantile(0.75)

iqr = third\_quartile - first\_quartile

lower = first\_quartile - 1.5\*iqr

upper = third\_quartile + 1.5\*iqr

salaries\_no\_outlier = nyc\_salary\_data.loc[((nyc\_salary\_data['base\_salary'] > lower) & (nyc\_salary\_data['base\_salary'] < upper))]

salaries\_no\_outlier = nyc\_salary\_data.loc[(nyc\_salary\_data['base\_salary']>(first\_quartile - 1.5\*iqr)) & (nyc\_salary\_data['base\_salary']<(third\_quartile + 1.5\*iqr))]

mean\_no\_outliers = salaries\_no\_outlier['base\_salary'].mean()

median\_no\_outliers = salaries\_no\_outlier['base\_salary'].median()

std\_numpy\_outliers = np.std(nyc\_salary\_data['base\_salary'])

std\_numpy\_no\_outliers = np.std(salaries\_no\_outlier['base\_salary'])

std\_pandas\_outliers = nyc\_salary\_data['base\_salary'].std()

std\_pandas\_no\_outliers = salaries\_no\_outlier['base\_salary'].std()

smolt\_mean = smolt['Reflectance'].mean()

smolt\_median = smolt['Reflectance'].median()

smolt\_std = smolt['Reflectance'].std()

smolt\_first\_quartile = smolt['Reflectance'].quantile(0.25)

smolt\_third\_quartile = smolt['Reflectance'].quantile(0.75)

first\_class = len(titanic.loc[titanic['class']=='First'])

first\_class\_over\_40 = len( titanic.loc[((titanic['class']=='First') & (titanic['age'] > 40))] )

p\_over\_40\_given\_first\_class = first\_class\_over\_40 / first\_class

second\_class = len(titanic.loc[titanic['class']=='Second'])

second\_class\_over\_40 = len( titanic.loc[((titanic['class']=='Second') & (titanic['age'] > 40))] )

p\_over\_40\_given\_second\_class = second\_class\_over\_40 / second\_class

first\_class = titanic.loc[titanic['class']=='First']['age']

second\_class = titanic.loc[titanic['class']=='Second']['age']

sns.histplot(first\_class)

sns.histplot(second\_class)

num\_survived = len( titanic.loc[(titanic['survived']==1)] )

survived\_over\_30 = len( titanic.loc[((titanic['survived']==1) & (titanic['age'] > 30))] )

p\_over\_30\_given\_survived = survived\_over\_30 / num\_survived

sns.histplot(titanic, x='age', hue='survived')

survived = **True**

**Quizes**

Uniform distributions model situations in which the outcomes are between two values, a and b, and all outcomes are equally probable. : True

When a fair, six-sided die is rolled, there are six possible outcomes: 1, 2, 3, 4, 5, and 6.

What is the probability of each outcome? : 1/6

What is the formula for the expected value E[X] for discrete and continuous uniform distributions? : (a+b)/2

The formula for the variance Var[X] for a discrete uniform distribution is (n2-1)/12. : True

What is the formula for the variance Var[X] of a continuous uniform distribution? : (b-a)2/12

To import uniform distribution packages from SciPy into Python, the statement used is

from scipy.stats import uniform -: True

*You are correct! The answer “True” is correct because to import uniform distribution packages from the library SciPy, we explicitly mention uniform type in the statement.*

A uniform distribution ranging from a to b is made using this Python statement: U = uniform(loc,scale).

What does the constructor loc represent? -: Loc = a

*You are correct! The answer “loc=a” is correct because the constructor loc in the function represents the starting range of the uniform distribution, so it is represented as “a”.*

A uniform distribution ranging from a to b is made using this Python statement:  U = uniform(loc,scale).

What does the constructor scale represent? -: scale=b-a

*You are correct! The answer “scale=b-a” is correct because the constructor scale in the function represents half of the width of the uniform distribution, so it is represented as “b-a”.*

If a uniform distribution is formulated in Python as U = uniform(loc=10, scale=5),

what is the output of the statement U.mean()? -: 12.5

*You are correct! The answer “12.5” is correct because the statement returns the mean of the uniform distribution. Here a=10, b=15, and the formula for the mean is (a+b)/2, which gives the output as “12.5”.*

If a uniform distribution is formulated in Python as U = uniform(loc=10, scale=5),

what does the statement U.var() represent? -: Variance

*You are correct! The answer “Variance” is correct because this function is used to get the variance of a uniform distribution.*

In Python, for a uniform distribution U the function U.rvs(size)

is used to sample from the distribution U. -: True

*You are correct! The answer “True” is correct because the function*rvs(size)*provided with the size constructor is used to provide samples from a distribution object of count “size”.*

The uniform distribution is often referred to as the “bell curve”. : False

*False because the graph of uniform distribution is a straight line, so it cannot be referred to as a bell curve. A normal distribution looks like a bell curve.*

The probability density function for normal distribution is

1

2

π

σ

2

e

−

(

x

−

μ

)

2

2

σ

2

True!

*True because the formula shown for normal distribution is correct.*

How do you represent the mean and variance of a random variable “**x̄n**”? -: Mean= μ**x̄n**

Variance= σ2**x̄n**

*You are correct! The answer “*Mean= μ**x̄n,**Variance= σ2**x̄n***” is correct because the symbol for mean and variance is “*μ” *and* “σ2”. *For the random variable* “**x̄n**” *the symbols are used together with the symbol of the variable for which the measures are used.*

The equation σ2**x̄n**=**(**σ2**x)**/n

states that the larger the sample size, the smaller the variance of the random variable **x̄n. -: True**

*You are correct! The answer “True” is correct because in the formula “*σ2**x̄n**=**(**σ2**x)**/n*”, the sample size “n” is in the denominator, which states that the increase in “n” will reduce the variance of the random variable* “**x̄n**”.

The Central Limit Theorem states that if you have a population with mean μ and standard deviation σ and take sufficiently large random samples from the population, then the distribution of the sample means will be approximately normally distributed. -: True

*You are correct! The answer “True” is correct because the Central Limit Theorem states that the sampling distribution of the mean approaches a normal distribution as the size of the sample increases.*

What is the general threshold decided by statisticians for random variable sample size? -: 30

*You are correct! The answer “30” is correct because this is the threshold decided by*statisticians for random variable sample size.

Which of the following is a multivariate random variable? : X=[D,L]

*You are correct! The answer “X=[D,L]” is correct because a multivariate random variable is a collection of random variables.*

The mean of a multivariate random variable is the scalar of the means of the individual components. : False

*You are correct! The answer “False” is correct because the mean of a multivariate random variable is the vector of the means of the individual components.*

What do you call the matrix that is used to compute the variances and covariances of a multivariate random variable? : Covariance matrix

*You are correct! The answer “Covariance matrix” is correct because it summarizes the variances and covariances of a set of vectors.*

What is the formula to calculate covariance? : Covar[X] = E[(X-E[X])(X-E[X])2]

*You are correct! The answer “*Covar[X] = E[(X-E[X])(X-E[X])2]*” is correct because X minus the expectation of X multiplied by X minus expectation of X whole squared is the formula for covariance.*

What does the covariance between variables “A” and “B” represent? :

The relationship between “B” and “A”

The relationship between “A” and “B”

*You are correct! The answers “*The relationship between “A” and “B”,” *and “*The relationship between “B” and “A”*” are correct because covariance is a measure of the relationship between two variables and the variance is the same, regardless of direction.*

Consider the following covariance matrix:

∑

=

[

|  |  |  |
| --- | --- | --- |
| **σ**  00  2 | **σ**  01  2 | **σ**  02  2 |
| **σ**  10  2 | **σ**  11  2 | **σ**  12  2 |
| **σ**  20  2 | **σ**  21  2 | **σ**  22  2 |

]

  The entries in the diagonal, “σ200”, “σ211” and “ σ222”, represent the variances of the variables 1, 2, and 3, respectively. : True

*You are correct! The answer “*True*” is correct because in a covariance matrix the values in the diagonal are the variances of the respective variables.*

The covariance of “X1” with “X2” is equal to the covariance of “X2” with “X1”. : True

*You are correct! The answer “*True*” is correct because the covariance* *matrix has an axis of symmetry along its diagonal, hence these are the same.*

In Python, import seaborn as sns is used for statistical graphics. : True

*You are correct! The answer “*True*” is correct because seaborn is a data visualization library built on top of matplotlib and therefore is used for statistical graphics.*

In Python, what would you use to generate a covariance matrix on a dataframe? : df.cov()

*You are correct! The answer “*df.cov()*” is correct because the function “*cov()*” is used to build a covariance matrix for a dataframe.*

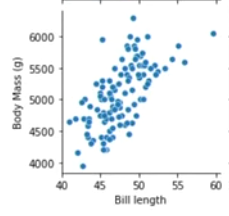
In Python what is the output of the given statement df.corr()? : Correlation matrix

*You are correct! The answer “*Correlation matrix*” is correct because the function “*corr()*” is used to build a correlation matrix for a dataframe in Python.*

In Python, to plot a correlation matrix, the function “sns()” is used. : False

*You are correct! The answer “*False*” is correct because to plot a correlation matrix the function “*sns.pairplot(df)*” is used. The “*sns*” itself is just the alias of the Python library seaborn.*

The plot below shows a weak correlation between the variable “Body Mass” and “Bill length”.



: False

*You are correct! The answer “*False*” is correct because the correlation between the variables*“Body Mass” *and*“Bill length” *is strong. The increase in the value of one variable results in an increase in the value of the second variable as well.*

What is the symbol used to represent correlation measures in a correlation matrix? : Rho,⍴

*You are correct! The answer “*rho,⍴*” is correct because the symbol is used for correlation measures.*

Consider the following correlation matrix:

C

o

r

r

[

X

]

=

[

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | **ρ**  1  ,  2 | … | **ρ**  1  ,  n |
| **ρ**  2  ,  1 | 1 | … | **ρ**  2  ,  n |
| ⋮ | ⋮ | ⋱ | ⋮ |
| **ρ**  n  ,  1 | **ρ**  n  ,  2 | … | 1 |

]

What is the correct formula to get the value of ⍴1,2? : ⍴1,2 = 𝜎1,22/𝜎1𝜎2

*You are correct! The answer “*⍴1,2 = 𝜎1,22/𝜎1𝜎2*” is correct because correlation “*⍴1,2*” is obtained by taking the 1,2 covariance and dividing it by the standard deviations of X1 and X2.*

A positive correlation means that when one goes up, the other tends to go down, and when one goes down, the other tends to go up. : False

*You are correct! The answer “*False*” is correct because a positive correlation means that they both go up and down together.*

What is the equation to get the conditional probability of variable Y given X? : Py(Y|X=x)= Px,y(X,Y)/Px(X)

*You are correct! The answer “*Py(Y|X=x)= Px,y(X,Y)/Px(X)*” is correct because* *the conditional probability of Y given X is equal to the joint pdf of X and Y divided by the marginal distribution of X.*

In Python statement

sns.histplot(data=gentoo, x=’Bill Length’, hue=’sex’),

the constructor “hue” is used to display bar heights. : False

*You are correct! The answer “*False*” is correct because “*hue*” is used as a semantic variable that is mapped to determine the color of plot elements.*

If two variables ‘X’ and ‘Y’ are independent of each other, then what is the joint distribution PX,Y(X,Y)? : PX,Y(X,Y) =PX(X)PY(Y)

*You are correct! The answer “*PX,Y(X,Y) =PX(X)PY(Y)*” is correct because if the variables are independent, their joint distribution is equal to the multiplication of their marginal distributions.*

If two variables ‘X’ and ‘Y’ are said to be independent, then

Py(Y|X=x1)=Py(Y|X=x2)=Py(Y). : True

*You are correct! The answer “*True*” is correct because the distribution of Y given X=x1 or the distribution of Y given X=x2 is equal to the marginal distribution of Y if the variables are independent.*

*Quiz*

If two variables ‘X’ and ‘Y’ are said to be independent, then

Py(Y|X=x1)=Py(Y|X=x2)=Py(Y). True

What does the covariance between variables “A” and “B” represent?

The relationship between “B” and “A”

The relationship between “A” and “B”

The uniform distribution is often referred to as the “bell curve”. False

The plot below shows a weak correlation between the variable “Body Mass” and “Bill length”.

False

In Python statement

sns.histplot(data=gentoo, x=’Bill Length’, hue=’sex’),

the constructor “hue” is used to display bar heights.

False

A uniform distribution ranging from a to b is made using this Python statement: U = uniform(loc,scale).

What does the constructor scale represent?

scale=b-a

In Python, what is the output of the given statement df.corr()?

Correlation matrix

What is the equation to get the conditional probability of variable Y given X?

Py(Y|X=x)= Px,y(X,Y)/Px(X)

For a correlation matrix

C

o

r

r

[

X

]

=

[

|  |  |  |  |
| --- | --- | --- | --- |
| 1 | **ρ**  1  ,  2 | … | **ρ**  1  ,  n |
| **ρ**  2  ,  1 | 1 | … | **ρ**  2  ,  n |
| ⋮ | ⋮ | ⋱ | ⋮ |
| **ρ**  n  ,  1 | **ρ**  n  ,  2 | … | 1 |

]

The correct formula to get the value of ⍴1,2 is:

⍴1,2 = 𝜎1,22/𝜎1𝜎2

Uniform distributions model situations in which the outcomes are between two values, a and b, and all outcomes are equally probable.

True

The covariance of “X1” with “X2” is equal to the covariance of “X2” with “X1”.

True

Which of the following is a multivariate random variable?

X=[D,L]

A uniform distribution ranging from a to b is made using this Python statement U = uniform(loc,scale).

What does the constructor loc represent?

Loc = a

Consider the following covariance matrix:

∑

=

[

|  |  |  |
| --- | --- | --- |
| **σ**  00  2 | **σ**  01  2 | **σ**  02  2 |
| **σ**  10  2 | **σ**  11  2 | **σ**  12  2 |
| **σ**  20  2 | **σ**  21  2 | **σ**  22  2 |

]

  The entries in the diagonal, “σ200”, “σ211” and “ σ222”, represent the variances of the variables 1, 2, and 3, respectively.

True

The matrix that is used to compute the variances and covariances of a multivariate random variable is called

What is the formula to calculate covariance?

Covar[X] = E[(X-E[X])(X-E[X])2]

When a fair, six-sided die is rolled, there are six possible outcomes: 1, 2, 3, 4, 5, and 6.

What is the probability of each outcome?

1/6

How do we represent the mean and variance of a random variable “**x̄n**”?

Mean= μ**x̄n**

Variance= σ2**x̄n**

The Central Limit Theorem states that if you have a population with mean μ and standard deviation σ and take sufficiently large random samples from the population, then the distribution of the sample means will be approximately normally distributed.

True

What is the symbol used to represent correlation measures in a correlation matrix?

Rho,⍴

The formula for the variance Var[X] for a discrete uniform distribution is (n2-1)/12.

True

A positive correlation means that when one goes up, the other tends to go down, and when one goes down, the other tends to go up.

False

To import uniform distribution packages from SciPy into Python, the statement used is

from scipy.stats import uniform

True

The equation σ2**x̄n**=**(**σ2**x)**/n

states that the larger the sample size, the smaller the variance of the random variable **x̄n.**

**True**

What is the general threshold decided by statisticians for random variable sample size?

30

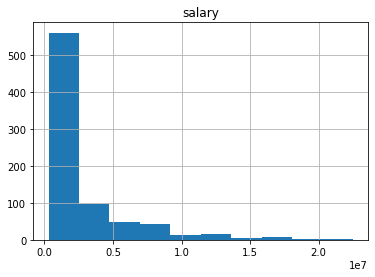
**Discussion Activity**

**2.1**

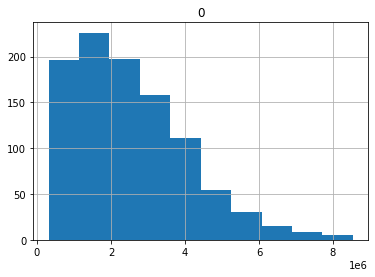
Dear Fellow Coworker,

I came across a dataset of Major League Baseball player salaries which was gathered from USA Today’s database. I thought I could see The *Central Limit Theorem* in action as work shown below.

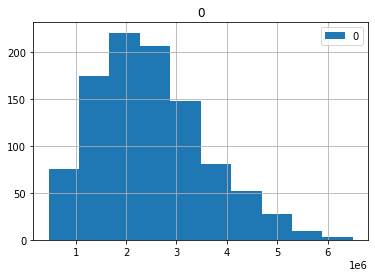
Please note that the dataset is right-skewed showing number of low earners are majority and there is a long tail towards high earners as shown in the diagram also reflected in the median and mean. The range of salaries from $300000 through $22500000, the median is $800000 and the mean is $2497668.6850690087, not linear, the standard deviation is 3535924.969930462.



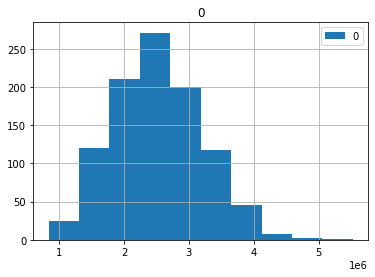
First, I started with 5 sample size, the mean is 2522718.052, it is close to the dataset’s mean value! But, the sample data shape is still skewed, the standard deviation is 1528540.9580613254 smaller than the dataset’s standard deviation value.



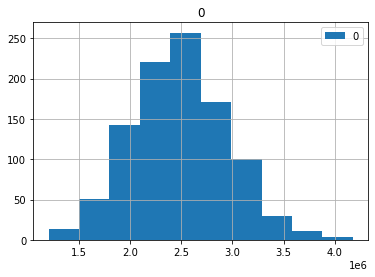
When the sample size is 10, the histogram becomes looking normal distribution, mean is more close to the original dataset’s mean value, 2481489.6061000004 now, the standard deviation is smaller 1089048.6400678856.



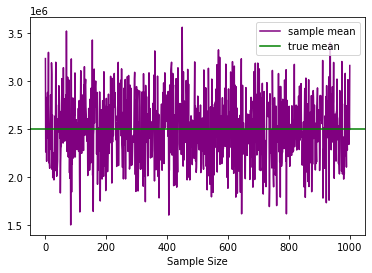
Starting with sample size 25, the histogram started looking better, however, the mean is a little off at 2516991.16008 but the standard deviation is more smaller 680793.4292964822.



However, at sample size 50, the histogram getting better, however, the mean is more closer to the original value at 2488753.0537199997, the standard deviation is more smaller 465761.1101933828.



At sample size 100, the mean at 2500692.7811100003 converges to the original dataset’s value as the *Central Limit Theorem* dictates the mean of a sample of data will be closer to the mean of the overall population as the sample size increases, see below the mean comparison chart, also note that the standard deviation is at its smallest 335622.5646845928.



So, as the histogram approaches to normal distribution, the standard deviation gets smaller as an indicator.

I just wanted to share my observations with you.

Regards,

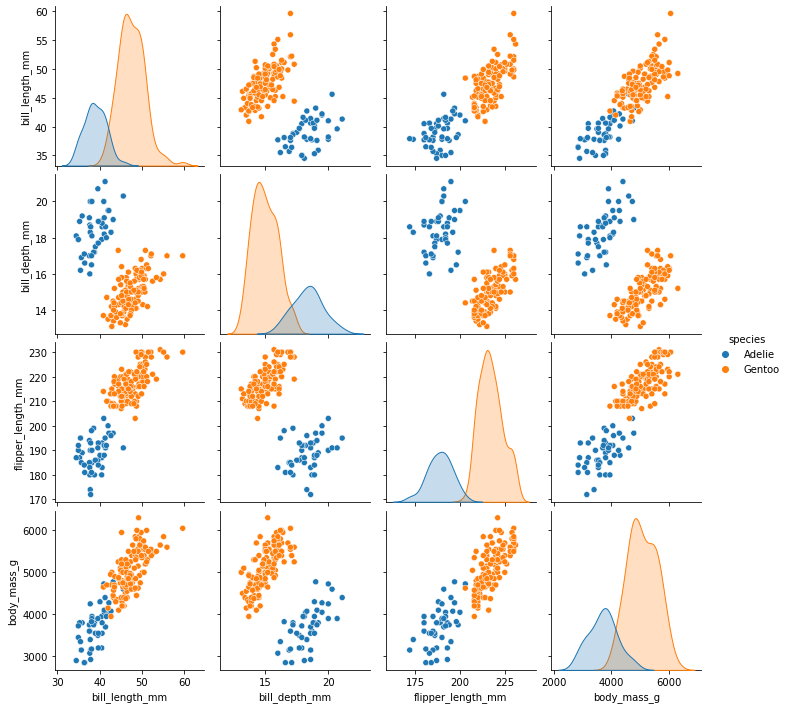
Aykan Erdenizmenli

**2.2**

**Penguins:**

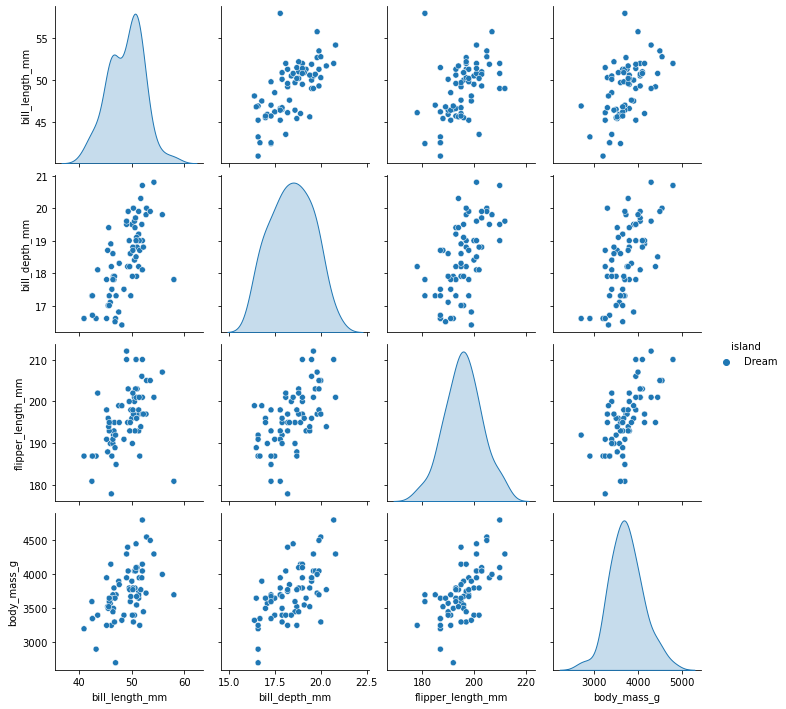
In penguins dataset, besides species there is also island information, I thought I could split data by islands to see if geography has a role. When I drew pair plot it is clear that penguins leave on Biscoe Island has larger range on all anatomical measurements. So, I could split data by Biscoe and other islands to start with.

There is strong positive correlation between body mass with all other measurements by each species which are Adelie and Gentoo and every other measurement as shown below.



In contrast to that there is weak positive correlation in the other islands dataset between those numeric features even for Adelie specie on Torgersen and Dream islands.

Next step was to separating **Chinstrap** from this other island dataset to analyze further: There is weak positive correlation between bill length with bill depth, flipper length, body mass; and between all other numeric measurements for this specie.



**Cars:**

What is axle ratio? It's a ratio that represents the number of revolutions the driveshaft must make to spin the axle one full turn. The higher the ratio, the higher the performance. The lower the ratio, the higher the fuel economy.

**First set by categorical feature of engine type (vs):**

Negative correlation mpg with engine size, higher the engine size less mileage per gallon.

Negative correlation mpg with hp, more the horse power is less mileage per gallon.

Positive correlation mpg with rear axle ratio, the higher the ratio is more mileage per gallon.

Negative correlation mpg with weight, more the weight is less mileage per gallon.

No correlation mpg with 1/4 mile time.

Positive correlation engine size with horse power.

Negative correlation engine size with rear axle ratio, higher the engine size is smaller the ratio.

Positive correlation engine size with weight.

Negative correlation engine size with 1/4 mile time.

Negative correlation horse power with rear axle ratio, higher the horse power is smaller the ratio.

Positive correlation horse power with weight.

Negative correlation horse power with 1/4 mile time.

Negative correlation rear axle ratio with weight, higher the ration is lighter weight

Positive correlation ear axle ratio with 1/4 mile time.

No correlation weight with 1/4 mile time.

**Second set by categorical feature of manual/automatic transmission (am):**

Negative correlation mpg with engine size, higher the engine size less mileage per gallon.

Negative correlation mpg with hp, more the horse power is less mileage per gallon.

Positive correlation mpg with rear axle ratio, the higher the ratio is more mileage per gallon.

Negative correlation mpg with weight, more the weight is less mileage per gallon.

No correlation mpg with 1/4 mile time.

Positive correlation engine size with horse power.

Negative correlation engine size with rear axle ratio, higher the engine size is smaller the ratio.

Positive correlation engine size with weight.

Negative correlation engine size with 1/4 mile time.

Negative correlation horse power with rear axle ratio, higher the horse power is smaller the ratio.

Positive correlation horse power with weight.

Negative correlation horse power with 1/4 mile time.

Negative correlation rear axle ratio with weight, higher the ration is lighter weight

Positive correlation ear axle ratio with 1/4 mile time.

No correlation weight with 1/4 mile time.

**Third set by categorical feature of number of forward gears (gear):**

Negative correlation mpg with engine size, higher the engine size less mileage per gallon.

Negative correlation mpg with hp, more the horse power is less mileage per gallon.

Positive correlation mpg with rear axle ratio, the higher the ratio is more mileage per gallon.

Negative correlation mpg with weight, more the weight is less mileage per gallon.

No correlation mpg with 1/4 mile time.

Positive correlation engine size with horse power.

Negative correlation engine size with rear axle ratio, higher the engine size is smaller the ratio.

Positive correlation engine size with weight.

Negative correlation engine size with 1/4 mile time.

Negative correlation horse power with rear axle ratio, higher the horse power is smaller the ratio.

Positive correlation horse power with weight.

Negative correlation horse power with 1/4 mile time.

Negative correlation rear axle ratio with weight, higher the ratio is lighter weight

Positive correlation ear axle ratio with 1/4 mile time.

No correlation weight with 1/4 mile time.

**Fourth set by categorical feature of number of carburetors (carb): (except 3, 6 and 8 carburetors)**

Negative correlation mpg with engine size, higher the engine size less mileage per gallon.

Negative correlation mpg with hp, more the horse power is less mileage per gallon.

Positive correlation mpg with rear axle ratio, the higher the ratio is more mileage per gallon.

Negative correlation mpg with weight, more the weight is less mileage per gallon.

No correlation mpg with 1/4 mile time.

Positive correlation engine size with horse power.

Negative correlation engine size with rear axle ratio, higher the engine size is smaller the ratio.

Positive correlation engine size with weight.

Negative correlation engine size with 1/4 mile time.

Negative correlation horse power with rear axle ratio, higher the horse power is smaller the ratio.

Positive correlation horse power with weight.

Negative correlation horse power with 1/4 mile time.

Negative correlation rear axle ratio with weight, higher the ratio is lighter weight

Positive correlation ear axle ratio with 1/4 mile time.

No correlation weight with 1/4 mile time.

**Fifth set by categorical feature of number of cylinders (cyl):**

Negative correlation mpg with engine size, higher the engine size less mileage per gallon.

Negative correlation mpg with hp, more the horse power is less mileage per gallon.

Positive correlation mpg with rear axle ratio, the higher the ratio is more mileage per gallon.

Negative correlation mpg with weight, more the weight is less mileage per gallon.

No correlation mpg with 1/4 mile time.

Positive correlation engine size with horse power.

Negative correlation engine size with rear axle ratio, higher the engine size is smaller the ratio.

Positive correlation engine size with weight.

Negative correlation engine size with 1/4 mile time.

Negative correlation horse power with rear axle ratio, higher the horse power is smaller the ratio.

Positive correlation horse power with weight.

Negative correlation horse power with 1/4 mile time.

Negative correlation rear axle ratio with weight, higher the ratio is lighter weight

Positive correlation ear axle ratio with 1/4 mile time.

No correlation weight with 1/4 mile time.

——

**Cars:**

**MPG Correlation with:**

Negative: engine size, hp, weight

Positive: rear axle ratio

None with 1/4 mile time.

**Engine Size Correlation with:**

Positive: horse power, weight.

Negative: rear axle ratio, 1/4 mile time.

**Horse Power Correlation:**

Negative: rear axle ratio, 1/4 mile time.

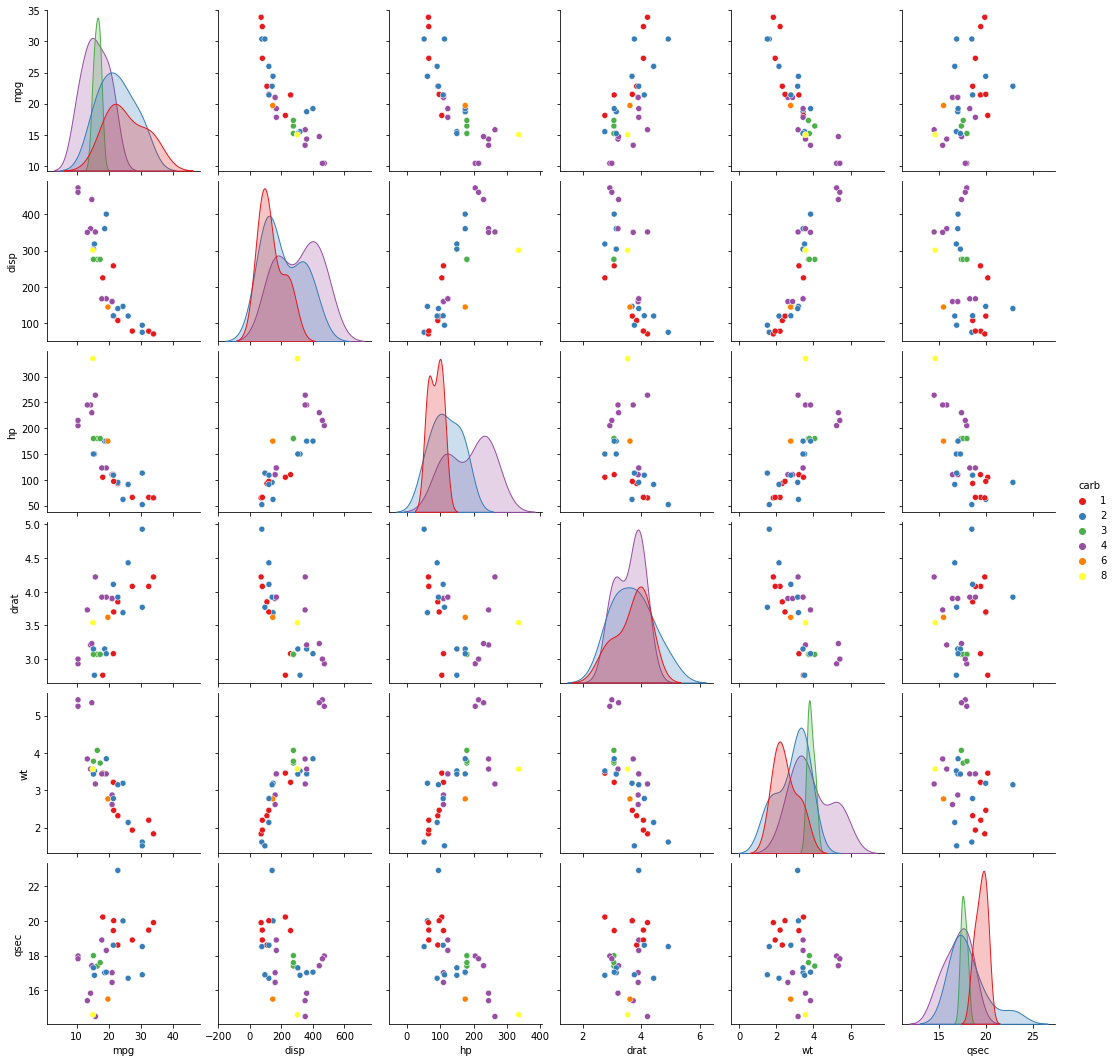
Positive: weight.

**Rear Axle Ratio Correlation:**

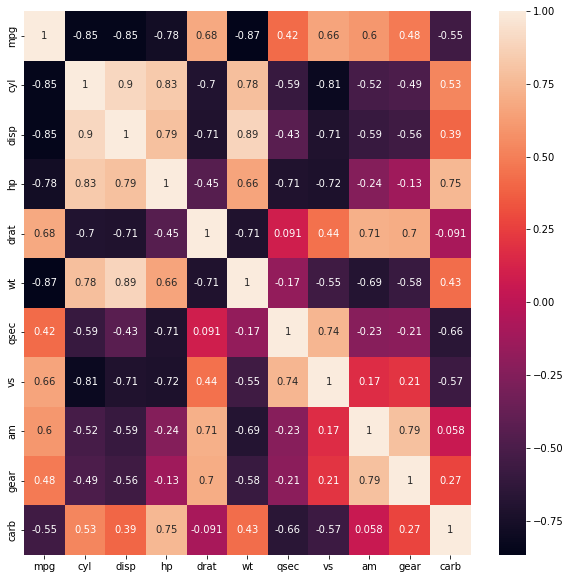
Negative: weight.

Positive: 1/4 mile time.

**No correlation weight with 1/4 mile time.**



All 5 datasets go with same correlation except (carb) categorical feature of 3, 6 and 8 as there are not many data points to conclude!



 ————— o —————

**Module 3**

Here are a few helpful downloads for this module:

* [Video Transcripts](https://student.emeritus.org/courses/4765/files/2911992?wrap=1)
* [Download Video Transcripts](https://student.emeritus.org/courses/4765/files/2911992/download?download_frd=1)
* [Quick Reference Guide](https://student.emeritus.org/courses/4765/files/2911994?wrap=1)

**plotly**

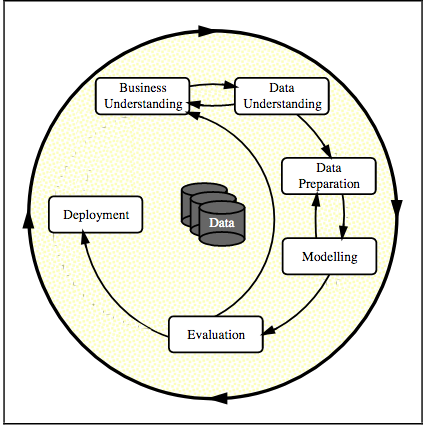
pip install --trusted-host pypi.org --trusted-host files.pythonhosted.org plotly

Done! Successfully installed plotly-5.6.0 tenacity-8.0.1

Cross-Industry Standard Process for Data Mining (CRISP-DM)

The CRISP-DM framework is important to data scientists for a variety of reasons, including:

* The methodology includes several processes that take care of simple data mining tasks
* It promotes best practices and facilitates the replication of projects
* It provides a uniform framework for planning and managing projects
* It is a cross-industry standard to be used by any data science project, regardless of its domain



df.sample(5)

df[df["column name"].isin(name of list)] or df.query(‘Entity in @list\_of\_countries’)

ans\_4 = df.query('minutes > 30 and points > 25')

df.query('team.str.contains("k")', engine = "python")

ans1 = gapminder.groupby('year')[['lifeExp']].agg('mean')

ans2 = gapminder.groupby('continent')[['gdpPercap']].agg('median')

ans3 = gapminder.groupby('continent')[['gdpPercap']].agg(['mean','median','std'])

ans4 = gapminder.groupby(gapminder['pop'] > 500\_000\_000)[['lifeExp']].agg('mean')

ans5a = gapminder.query('continent in ("Americas", "Europe")')

ans5b = ans5a.groupby(['continent', 'country'])[['lifeExp']].agg('mean')

churn\_pct = churn\_df['churn'].value\_counts(normalize = True)

**Issues**

1. Activity 3.2: Problem 8: fig.write\_image('images/plotly\_hist.png') requires missing kaleido package:

$ pip install -U kaleido. Skip

1. Activity 3.3: advanced, not straight forward
2. Activity 3.4 Problem 3 goal is not clear, Problem 4 has wrong input, Problem 5 is misleading
3. Activity 3.6 Instructions are not clear to keep what columns or not
4. Activity 3.8 Problem 8, plot is misleading, not clear instructions, Problem 9 is misleading too. Grading is not done, scores 0.
5. Activity 3.8 fails to grade!

**Jupiter environment update:**

#import sys

#!{sys.executable} -m pip install -U kaleido

**Install plotly**

CERTIFICATE\_VERIFY\_FAILED

import ssl

ssl.\_create\_default\_https\_context = ssl.\_create\_unverified\_context

!jupyter kernelspec list

gapminder['lifeExp'].plot(kind='hist', bins=15, edgecolor='black', title='Histogram of Life Expectency')

sns.displot(gapminder,kind='kde',x='lifeExp', hue='continent', multiple="stack")

sns.boxplot(data=gapminder, x='lifeExp', y='gdpPercap', hue='continent')

#create seaborn boxplots by group

sns.boxplot(x='variable', y='value', data=df\_melted).set(title='Points by Team')

#modify axis labels

plt.xlabel('Team')

plt.ylabel('Points')

**Quizes**

In Python, what is the correct syntax for loading a dataframe from a csv file? : df=pd.read\_csv(filename)

*You are correct! The answer “*df=pd.read\_csv(filename)*” is correct because the statement calls pandas (pd) as well as the function [read\_csv()] and passes the filename inside the function parentheses.*

The function head() is used to show the last rows of the dataframe. : False

*That is correct! The answer “False” is correct because*head()*is used to show the top rows of the dataframe.*

In Python, what will df.sample(10) return? : Ten random rows of the dataframe df

*You are correct! The answer “*Ten random rows of the dataframe df*” is correct because the function*sample()*is used to print random rows of the data, and when given a parameter ‘10’, it will return ten random rows.*

For the dataframe shown in Video 3.2, you want to show only the rows of data from China. Which would be the correct syntax? : df[df[“Entity”]==”China”]

Dataframe

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | **Entity** | **Code** | **Year** | **GDP (constant 2010 US$)** |
| **6856** | Saint Lucia | LCA | 2009 | 1.404756e+09 |
| **1479** | Canada | CAN | 1983 | 8.031318e+11 |
| **6277** | Papua New Guinea | PNG | 1962 | 2.579601e+09 |
| **6073** | Oman | OMN | 1974 | 7.085347e+09 |
| **8279** | Uganda | UGA | 1992 | 5.751489e+09 |

*You are correct! The answer “*df[df[“Entity”]==”China”]*” is correct because the column Entity has the name of the countries, and the filtering syntax is correct.*

In Python, the function df.query() is used to apply filters on a dataframe. - True

*You are correct! The answer “True” is correct because the*query()*function is used to query the columns of a dataframe with a Boolean expression to filter the data.*

In Python, given the list

list\_of\_countries=[“UK”,”USA”,”China”,”Egypt”], what is the output of the statement

df.query(‘Entity in @list\_of\_countries’)? : Return all the entries within the dataframe where Entity is in list\_of\_countries

*You are correct! The answer “*Return all the entries within the dataframe where Entity is in list\_of\_countries*” is correct because the statement will return all the data in the dataframe that meets the filter criteria that the Entity column equals to any of the country names from the list list\_of\_countries.*

Which statement do you need to use in Python for the dataframe df to build a bar plot between the columns Entity and GDP? : df.plot(x=”Entity”,y=”gdp”,kind=”bar”)

*You are correct! The answer “*df.plot(x=”Entity”,y=”gdp”,kind=”bar”)*” is correct because the correct syntax to build a plot with the required x- and y-axes and plot are provided correctly.*

What is the function that rotates the x-axis labels of a plot built with Seaborn? : xticks(rotation)

*You are correct! The answer “*xticks(rotation)*” is correct because the function is used for the rotation of the x-axis labels of a plot.*

Which of the following is the function to plot a bar chart using Plotly? : fig=px.bar(df,x=”Entity”,y=”gdp”,color=”Entity”)

*You are correct! The answer “*fig=px.bar(df,x=”Entity”,y=”gdp”,color=”Entity”)*” is correct because this is the function for the Python library Plotly to draw a bar chart.*

Which function alters font size in a plot built with Plotly? : fig.update\_layout(font\_size=””)

!!!Ambigues!

*You are correct! The answer “*fig.update\_layout(font\_size=””)*” is correct because this is the function used to alter font size in a chart built by the Python library Plotly.*

The constructor “Title” in the Python statement

fig=px.kbar(df,x=”EntityPPp”,color=”Entity”,Title=”XYZ”)

is used to set the title of the legends in the chart. : False

*You are correct! The answer “False” is correct because the constructor “Title” is used to set the title of the whole chart, not the legends.*

The Python dataframe function groupby(’Entity’) allows splitting data into separate groups to perform computations for better analysis. : True

*You are correct! The answer “True” is correct because the function*groupby(’Entity’)*recognizes data so that rows with the same entity are clustered together.*

The agg(sum) followed by groupby(‘Entity’) sums all the values in the columns where the group of entities is matched. : True

*You are correct! The answer “True” is correct because the*agg(sum) *followed by* groupby(‘Entity’) sums*the values in all the columns of a dataframe for each group of Entity.*

What will be the output of the last column for df.groupby(‘Entity’).agg(sum)” when Entity is equal to China? : 86.88

**groupby("Entity")**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| China |  | 1960 |  | 1 |
|  |  |  |  |  |
| India |  | 1960 |  | 1 |
|  |  |  |  |  |
| USA |  | 1960 |  | 1 |
|  |  |  |  |  |
| China |  | 1990 |  | 6.48 |
|  |  |  |  |  |
| India |  | 1990 |  | 3.41 |
|  |  |  |  |  |
| USA |  | 1990 |  | 2.94 |
|  |  |  |  |  |
| China |  | 2017 |  | 79.4 |
|  |  |  |  |  |
| India |  | 2017 |  | 19.2 |
|  |  |  |  |  |
| USA |  | 2017 |  | 5.62 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| China |  | 1960 |  | 1 |
|  |  |  |  |  |
| China |  | 1990 |  | 6.48 |
|  |  |  |  |  |
| China |  | 2017 |  | 79.4 |
|  |  |  |  |  |

**agg(sum)**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| China |  | 5961 |  |  |

Group of answer choices

*You are correct! The answer “86.88” is correct because for all the rows where Entity is equal to China, the sum of the last column can be calculated as 1 + 6.48 + 79.4 = 86.88, which is the correct answer.*

The function df.sort\_values(“Column”) returns the random values of a column. : False

*You are correct! The answer “False” is correct because the function is used to sort a dataframe in ascending or descending order on a passed column.*

For a dataframe df, given a column name gdp, what would be the statement in Python to sort the dataframe on gdp? : df.sort\_values(“gdp”)

*You are correct! The answer “*df.sort\_values(“gdp”)*” is correct because this is the correct syntax for applying a sorting function on the column gdp.*

A function in Python is a named section of a code that performs a specific task. : True

*You are correct! The answer “True” is correct because a function is a specific section of a code that involves taking some input, manipulating the input, and returning an output.*

Suppose you have the Python statement

df.groupby(“column”).agg(sum). Instead of using .agg(sum), what would be the built-in function to aggregate on a summation? : .sum()

*You are correct! The answer “*.sum()*” is correct because this is the built-in function used for aggregation on a summation.*

Given this dataframe, what is the Python statement to get the total GDP per year? : df.groupby(“Year”).agg(sum)

!!!Ambigues! Choices

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Entity** | **Code** | **Year** | **GDP (constant 2010 US$)** | **gdp** |
| **0** | Afghanistan | AFG | 2002 | 8.013233e+09 | 8.013233 |
| **1** | Afghanistan | AFG | 2003 | 8.689884e+09 | 8.689884 |
| **2** | Afghanistan | AFG | 2004 | 8.781610e+09 | 8.781610 |
| **3** | Afghanistan | AFG | 2005 | 9.762979e+09 | 9.762979 |
| **4** | Afghanistan | AFG | 2006 | 1.030523e+10 | 10.305228 |
| **...** | ... | ... | ... | ... | ... |
| **8864** | Zimbabwe | ZWE | 2013 | 1.418193e+10 | 14.181927 |
| **8865** | Zimbabwe | ZWE | 2014 | 1.448359e+10 | 14.483588 |
| **8866** | Zimbabwe | ZWE | 2015 | 1.472830e+10 | 14.728302 |
| **8867** | Zimbabwe | ZWE | 2016 | 1.481899e+10 | 14.818986 |
| **8868** | Zimbabwe | ZWE | 2017 | 1.532981e+10 | 15.329811 |

*You are correct! The answer “*df.groupby(“Year”).agg(sum)*” is correct because the statement groups the data on the column Year and aggregates the sum for each year tuple on all remaining columns.*

If dataframe df1 has an index Entity, and dataframe df2 has an index Year, will the statement df1/df2 be valid? : No

!!!Ambigues! The statement is true, semantics wrong!

*You are correct! The answer “No” is correct because the two dataframes have separate indexes that do not match. Hence, the division of the two dataframes will return NaN values, which is not a valid result.*

The Python function set\_index(“Column”) changes the original dataframe. : False

*You are correct! The answer “False” is correct because the function set\_index() does not actually change the original dataframe, just creates a copy.*

Suppose you have a dataframe df1 that has the columns year and gdp, with the index set as year. You also have a dataframe df2 that has columns labeled Year, Entity, and gdp.

To get the fraction of the GDP that each entity generates per year, what should be the index of df2? : set\_index([“Entity”,”Year”])

*You are correct! The answer “*set\_index([“Entity”,”Year”])*” is correct because setting the index at Year and Entity will generate a result for the fraction of the GDP that each entity generates per year.*

The Python function dropna() is used to drop all the rows that have NaN values. : True

*You are correct! The answer “True” is correct because in the pandas dataframe, the*dropna()*function is used to remove rows and columns with null/NaN values.*

Given this dataframe and the Python statement px.line(df,x=”Year”,y=”gdp”,color=”?”), what should be inside the quotes for the color parameter to get separate colored lines for each country? : Entity

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
|  | **Entity** | **Code** | **Year** | **GDP (constant 2010 US$)** | **gdp** |
| **0** | Afghanistan | AFG | 2002 | 8.013233e+09 | 8.013233 |
| **1** | Afghanistan | AFG | 2003 | 8.689884e+09 | 8.689884 |
| **2** | Afghanistan | AFG | 2004 | 8.781610e+09 | 8.781610 |
| **3** | Afghanistan | AFG | 2005 | 9.762979e+09 | 9.762979 |
| **4** | Afghanistan | AFG | 2006 | 1.030523e+10 | 10.305228 |
| **...** | ... | ... | ... | ... | ... |
| **8864** | Zimbabwe | ZWE | 2013 | 1.418193e+10 | 14.181927 |
| **8865** | Zimbabwe | ZWE | 2014 | 1.448359e+10 | 14.483588 |
| **8866** | Zimbabwe | ZWE | 2015 | 1.472830e+10 | 14.728302 |
| **8867** | Zimbabwe | ZWE | 2016 | 1.481899e+10 | 14.818986 |
| **8868** | Zimbabwe | ZWE | 2017 | 1.532981e+10 | 15.329811 |

*You are correct! The answer “Entity” is correct because the constructor “color” should be divided into Entity to show separate color codes for each country.*

In Python, the function filter() returns a sequence from the iterable elements for which the function returns True. : True

*You are correct! The answer “True” is correct because Python's*filter()*is a built-in function that allows you to process an iterable and extract those items that satisfy a given condition.*

Given the dataframe and the Python statement df.groupby(“Entity”).filter(max\_gdp\_ratio\_gt\_10), which of the following entities of the data will be in the output? : China, India

**def max\_gdp\_ratio\_gt\_10(s):** **max\_gdp\_ratio = max(s["gdp\_ratio"])** **return max\_gdp\_ratio > 10**

**Dataframe**

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| China |  | 1960 |  | 1 |
|  |  |  |  |  |
| India |  | 1960 |  | 1 |
|  |  |  |  |  |
| USA |  | 1960 |  | 1 |
|  |  |  |  |  |
| China |  | 1990 |  | 6.48 |
|  |  |  |  |  |
| India |  | 1990 |  | 3.41 |
|  |  |  |  |  |
| USA |  | 1990 |  | 2.94 |
|  |  |  |  |  |
| China |  | 2017 |  | 79.4 |
|  |  |  |  |  |
| India |  | 2017 |  | 19.2 |
|  |  |  |  |  |
| USA |  | 2017 |  | 5.62 |

*You are correct! The answer “China” and “India” are correct because in the dataframe, both countries have a GDP ratio greater than ten for at least one row of data.*

The Python function set\_index(“Column”) changes the original dataframe. : False

The agg(sum) followed by groupby(‘Entity’) sums all the values in the columns where the group of entities is matched. : True

Suppose you have a dataframe df1 that has the columns year and gdp, with the index set as year. You also have a dataframe df2 that has columns labeled Year, Entity, and gdp.

To get the fraction of the GDP that each entity generates per year, what should be the index of df2? : set\_index([“Entity”,”Year”])

In Python, what will df.sample(10) return? : Ten random rows of the dataframe df

Suppose you have the Python statement df.groupby(“column”).agg(sum). Instead of using .agg(sum), what would be the built-in function to aggregate on a summation? : .sum()

The function df.sort\_values(“Column”) returns the random values of a column. : Flase

A function in Python is a named section of a code that performs a specific task. : True

For a dataframe df, given a column name gdp, what would be the statement in Python to sort the dataframe on gdp? : df.sort\_values(“gdp”)

In Python, what is the correct syntax for loading a dataframe from a csv file? : df=pd.read\_csv(filename)

If dataframe df1 has an index Entity, and dataframe df2 has an index Year, will the statement df1/df2 be valid? :

**Discussion Activity**

**3.1**

<http://cs.unibo.it/~danilo.montesi/CBD/Beatriz/10.1.1.198.5133.pdf>

Data projects we deal almost everyday, the methodology outlined in the paper sounded a mix of software development and project management lifecycle which made me automatically think of bunch of questions related to each phase as below:

**Business Understanding**

Describes what form of business area it is implementing, what are business objectives, success criteria? What information should be provided to help out. Should be it be reactive, proactive? Which business units should be involved, what products are affected in large scale?

**Data Understanding**

For the specific business area, do we have data? Where can we get data if missing, how do we onboard? What specific data elements is needed? What are the quality measures of this data?

**Data Preparation**

After listing what data elements is needed, what transformation logic is needed to form data, data mapping rules? Split, merge, normalization, transform, data standardization steps?

**Modeling**

How to define business understanding by Laying out prepared data, what sequences of data is defining business needs? What actions is needed to detect business need and what corrective actions should it take? What are the business rules to execute when there is a match or mismatch?

**Evaluation**

How closely modeling is serving to the business needs, feedback to business understanding if any shortfalls, or any new requirements arise

**Deployment**

Rolling to production if satisfies business needs or within acceptable threshold per business success criteria, where are all targeted deployment areas, use cases? Are they all covered?

Obviously, the process is standardization of project management just for data mining, however, it is a broader scale can be applied to any project management.

**3.2**

* Attach your visualization
* Explain the method you used to create the visualization
* Share what library you used
* Describe the results you found and what they tell you about the dataset you chose

Exploring

Get gapminder dataset

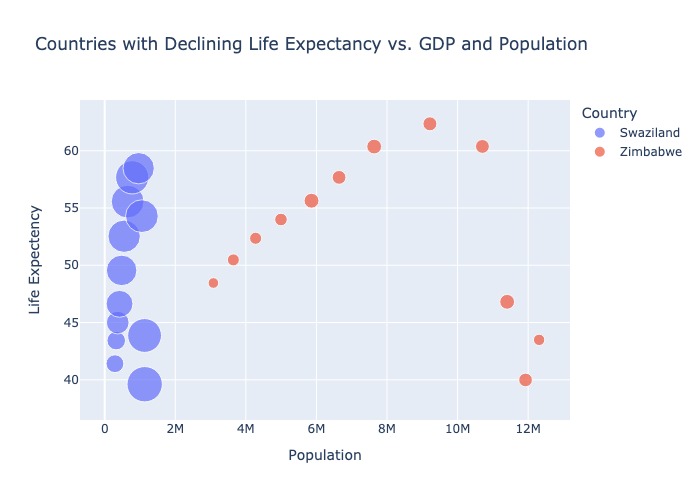
Show which countries having life expectancy decreasing over time

Correlate this finding to poverty! Look up if GDP is decreasing too!

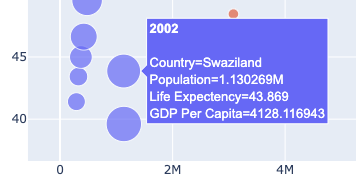
I found the gapminder dataset very intriguing, it contains year, continent, country, life expectancy, population and location. I analyzed data to see if there are any trends in country’s declining life expectancy with other features over time, I located 2 countries following the suit: Swaziland and Zimbabwe, both are in Africa.

px.scatter(gapminder.query('country in @list\_of\_countries'), x='pop', y='lifeExp', color='country', size='gdpPercap', size\_max=25, hover\_name='year', labels=dict(pop="Population", gdpPercap="GDP Per Capita", lifeExp="Life Expectency", country="Country"), title="Countries with Declining Life Expectancy vs. GDP and Population")

Visualization



I used plotly library to draw above scatter plot with hovering functionality to reveal all attributes for highlighting those features among declining life expectancy in two countries. The size of bubbles show how big their GDP per capita is, the x-axis is Population and y-axis is Life Expectancy. It is interactive displaying a pop up like:



Although, there is subtle or no correlation among those numeric features, there is an intriguing factor per country by looking at the x-axis and its hovering detail: For Swaziland, life expectancy started declining after the country’s population reached at 1M people and for Zimbabwe after reaching 10M people, perhaps these thresholds highlighting a point that each country’s infrastructure has a limit on how many people a country can serve effectively because the fluctuations in GDP per capita does not seem have an affect on life expectancy.

 ————— o —————

**Module 4**

Here are a few helpful downloads for this module:

* [Video Transcripts](https://student.emeritus.org/courses/4765/files/2949437?wrap=1)
* [Download Video Transcripts](https://student.emeritus.org/courses/4765/files/2949437/download?download_frd=1)
* [Quick Reference Guide](https://student.emeritus.org/courses/4765/files/2949448?wrap=1)

*#raise NotImplementedError()*

site\_visits\_df = pd.merge(left=site, right=visited, left\_on='name', right\_on='site', how='inner')

*#raise NotImplementedError()*

visited\_renamed = visited.rename(columns = {'site' : 'name'})

site\_visits\_df2 = pd.merge(left=site, right=visited\_renamed, on='name', how='inner')

*#raise NotImplementedError()*

survey\_renamed = survey.rename(columns = {'taken' : 'id'})

survey\_site\_visits = pd.merge(left=site\_visits\_df2, right=survey\_renamed, on='id', how='inner')

survey\_site\_visits\_ = pd.merge(survey\_.rename({'taken': 'id'}, axis = 1),

site\_visits\_df2\_, on = 'id')

*#raise NotImplementedError()*

survey\_site\_visits\_renamed = survey\_site\_visits.rename(columns = {'person' : 'person\_id'})

person\_renamed = person.rename(columns = {'id' : 'person\_id'})

full\_name\_df = pd.merge(left=survey\_site\_visits\_renamed, right=person\_renamed, on='person\_id', how='inner')

left\_ = survey\_site\_visits\_.rename({'person': 'person\_id'}, axis = 1)

right\_ = person\_.rename({'id': 'person\_id'}, axis = 1)

full\_name\_df\_ = pd.merge(left\_, right\_, on = 'person\_id')

*#raise NotImplementedError()*

ans5 = pd.merge(left=df1, right=df2, on='name', how='left')

*# Answer check*

print(type(ans5))

ans5

geofin = pd.merge(left = demographics, right = financials, on = 'id')

ans1 = geofin.query('country == "Kenya"')[['funded\_amount']].agg('mean').values[0]

ans1\_ = pd.merge(demographics\_.loc[demographics\_['country'] == 'Kenya'], financials\_, on = 'id')[['funded\_amount']].mean().values[0]

geofinuse = pd.merge(left = geofin, right = use, on = 'id')

ans2 = geofinuse.query('country == "El Salvador"').groupby('sector')[['funded\_amount']].agg('sum').reset\_index().sort\_values(by='funded\_amount',ascending=**False**)['sector'][0]

ans2\_ = pd.merge(demographics\_.loc[demographics\_['country'] == 'El Salvador'], use\_, on = 'id')['activity'].value\_counts().index[0]

ans3 = geofinuse.query('country == "Pakistan" and sector == "Agriculture"').groupby('sector')[['funded\_amount']].agg('sum').reset\_index()['funded\_amount'][0]

use\_ = pd.read\_csv('data/kiva/use.csv')

p1\_ = pd.merge(use\_, demographics\_, on = 'id')

a\_ = pd.merge(p1\_, financials\_, on = 'id')

b\_ = a\_.loc[a\_['country'] == 'Pakistan']

ans3\_ = b\_.loc[b\_['activity'] == 'Agriculture'][['funded\_amount']].sum().values[0]

ans4 = geofinuse.groupby('sector')[['funded\_amount']].agg('sum').reset\_index().sort\_values(by='funded\_amount',ascending=**False**)['sector'][0]

ans4\_ = pd.merge(financials\_, use\_, on = 'id').groupby('activity')[['funded\_amount']].sum().sort\_values(by = 'funded\_amount', ascending = **False**).index[0]

geofinusecrowd = pd.merge(left = geofinuse, right = crowdsource, on = 'id')

geofinusecrowd['dollar\_to\_lender\_ratio'] = geofinusecrowd['funded\_amount'] / geofinusecrowd['lender\_count']

ans5 = geofinusecrowd.groupby('sector')[['dollar\_to\_lender\_ratio']].agg('max').reset\_index().sort\_values(by='dollar\_to\_lender\_ratio',ascending=**False**)['sector'][0]

ans5\_ = b\_.groupby('activity')[['ratio']].sum().sort\_values('ratio', ascending = **False**).index[0]

px.scatter(gapminder, x='gdpPercap', y='lifeExp')

px.scatter(gapminder, x='gdpPercap', y='lifeExp', color='country')

px.scatter(gapminder, x='gdpPercap', y='lifeExp', color='country', size='pop')

px.scatter(gapminder, x='gdpPercap', y='lifeExp', color='country', size='pop', log\_x=True)

px.scatter(gapminder[gapminder['year'] == 2007], x='gdpPercap', y='lifeExp', color='country', size='pop', log\_x=True)

px.box(gapminder, x='gdpPercap', y='continent', color='continent')

*#raise NotImplementedError()*

ans1 = russian\_states[russian\_states['Economic region'].str.contains('Siberian')]

df.isnull().sum().sort\_values().plot(kind = 'bar')

plt.savefig('images/missing\_plot\_.png')

plt.close();

*#raise NotImplementedError()*

df.isna().sum().sort\_values().plot(kind='bar')

plt.savefig(‘images/missing\_plot.png')

plt.close()

*#raise NotImplementedError()*

ans4 = ans3.apply(**lambda** x : x[-1])

print(type(ans4))

ans4.head()

**Issues**

Activity 4.1 - Problem 5 rename variables:

print(type(ans5))

ans5

Activity 4.4 - Problem 4, asking “***float***”, however, it is expecting float64 to be more specific!

Activity 4.6 @ Problem 6

**Quizes**

The pandas function rename() is used to rename a column in a dataframe. : True

*You are correct! The answer “True” is correct because the*rename()*function is used to change column names in pandas dataframes.*

What is the statement to rename a column named Code to Country in dataframe df? : df.rename(columns ={“Code” : “Country”})

*You are correct! The answer “*df.rename(columns ={“Code” : “Country”})*” is correct because the parameters of the function were formed correctly, with the column names in quotation marks, separated by a colon, and in parentheses.*

The function merge() is used to join two datasets into one and align the rows from each based on their common columns. : True

*You are correct! The answer “True” is correct because the function*merge()*joins two datasets into one new dataframe based on the similar columns in the source datasets.*

Given two dataframes (df1 and df2) with common Entity columns, what will be the Python statement for joining them into a new dataframe (df3)? : df3=pd.merge(left=df1, right=df2, left\_on=”Entity”,right\_on=”Entity”)

*You are correct! The answer “*df3=pd.merge(left=df1, right=df2, left\_on=”Entity”,right\_on=”Entity”)*” is correct because all four parameters are correctly identified (i.e.,*left*,*right*,*left\_on*, and*right\_on*).*

The function merge() is used to join two datasets into one. Which constructor is used to declare the type of join? : how

*You are correct! The answer “how” is correct because the constructor how is used to declare the type of join to be made.*

The left join returns all rows from a table declared using right= and only the matching rows from a table declared using left=. : False

*You are correct! The answer “False” is correct because the left join returns all rows from the*left=*table and only the matching rows from the*right=*table.*

What is the type of join where the resulting table contains only rows from both tables where the joining condition is met? :  Inner join

*You are correct! The answer “Inner join” is correct because in this type of join, the final table has only those rows from both tables where the joining condition is met.*

df.sort\_values(“Column”) sorts a dataframe in ascending or descending order using Column. : True

*You are correct! The answer “True” is correct because the pandas*sort\_values()*function sorts a dataframe in ascending or descending order using Column.*

Using merge(), the join condition of two separate dataframes cannot use more than one column. : False

*You are correct! The answer “False” is correct because a join in two separate dataframes can use more than one column.*

Given two dataframes (df1 and df2) with two join conditions on columns Entity and Year, what is the statement for joining them into a new dataframe (df3)? : df3=pd.merge(left=df1, right=df2, left\_on=[”Entity”,”Year”],right\_on=[”Entity”,”Year”])

*You are correct! The answer “*df3=pd.merge(left=df1, right=df2,left\_on=[”Entity”,”Year”],right\_on=[”Entity”,”Year”])*” is correct because this is the correct syntax in Python to merge df1 and df2 on the columns Entity and Year.*

Consider the following line of code:

df3=pd.merge(left=df1, right=df2, left\_on=”Entity”,right\_on=”Entity”,how=?)

What would you put in the how= parameter to include all the rows from the left table and the rows matching the join condition from the right table? : Left

*You are correct! The answer “Left” is correct because this type of join returns all records from the left table and the matched records from the right table.*

The function reset\_index()is used to set an index of a dataframe. : False

*You are correct! The answer “False” is correct because*reset\_index()*is used to reset the index of a dataframe object to default indexing.*

What function is used to drop all NaN values from a dataframe? : dropna()

*You are correct! The answer “*dropna()*” is correct because the function in pandas is used to drop all the rows from a dataframe that have a NaN value.*

The scatterplot’s primary use is to show relationships between two numeric variables. : True

*You are correct! The answer “True” is correct because the dots in a scatterplot not only report the values of individual data points, but also the patterns when the data is taken as a whole. That shows the relationship between the variables.*

Given a dataframe with the columns Year, Entity, and gdp\_ratio, how would you draw a scatterplot to show the trends of GDP ratio per year for each entity? : px.scatter(df, x= “Year”, y = “gdp\_ratio” , color = “Entity”)

*You are correct! The answer “*px.scatter(df, x= “Year”, y = “gdp\_ratio” , color = “Entity”)*” is correct because this is the correct syntax to create a scatterplot to see the trends of GDP ratio per year for each entity.*

In the Python function px.scatter(), size= is used to set the size of the graph. : False

*You are correct! The answer “False” is correct because in the function px.scatter(), the constructor size= is used to set the size of the markers in the plot according to the given values.*

Given a dataframe df1 with the columns Entity, Code, Year, and Expectancy, how can it be filtered into a dataframe without the column Code? : df1=df1[[“Entity”,”Year”,”Expectancy”]]

*You are correct! The answer “*df1=df1[[“Entity”,”Year”,”Expectancy”]]*” is correct because this is the correct syntax to select all the columns except the column Code.*

To make a scatterplot using a logarithmic scale, which constructor in function px.scatter() is set to true? : log\_x = True

*You are correct! The answer “*log\_x = True*” is correct because this is the constructor that is used to turn on logarithmic scales in a scatterplot.*

In which library is px.histogram(df[“column”]) used to create a histogram? : Plotly

*You are correct! The answer “*Plotly*” is correct because the function px.histogram() is used to create a histogram in the Python library Plotly.*

sns.displot() is used to create a scatterplot in Seaborn. : False

*You are correct! The answer “*False*” is correct because the function sns.displot() in Seaborn is used to create a histogram.*

Using sns.displot() in Seaborn, what constructor would you use to also display the kernel density estimate curve on the histogram? : kde = True

*You are correct! The answer “*kde = True*” is correct because this is the constructor used in function*sns.displot()*to set the kernel density estimate curve on the histogram.*

The violin plot tells the kernel density estimate of a variable in a dataframe. : True

*You are correct! The answer “*True*” is correct because the violin plot shows the sidewise kernel density estimate of a column in a dataframe.*

What does the middle line of a box plot represent? : Median

*You are correct! The answer “*Median*” is correct because the middle line of a box plot is used to show the median of a dataset.*

What are the constructors used in the function px.scatter() to create x- and y-axes marginal plots? : **marginal\_x, marginal\_y**

*You are correct!*

In Seaborn, which plot function is used to quickly visualize and analyze the relationship between two variables and describe their individual distributions on the same plot? : sns.jointplot()

*You are correct! The answer “*sns.jointplot()” *is correct because this type of plot function is used to visualize the relationship between two variables and describe their individual distributions on the same plot.*

The statement df1[df1[“Entities”].str.contains(“in”)] gives only entities whose name contains the substring in. : True

*You are correct! The answer “*True” is*correct because the function*str.contains()*is used to get a subset of the data where the given value contains a substring.*

Which statement would return rows only for entities that start with “F”? : df1[df1[“Entities”].str.startswith(“F”)]

*“*df1[df1[“Entities”].str.startswith(“F”)]” is*correct because the syntax is correct, and the function startswith() is used to filter the data where values start with a given letter.*

str.upper() is used to get all the values that are in uppercase letters. : False

*You are correct! The answer “False*” is*correct because the function upper() is used to convert the values into uppercase letters.*

Which function is used to replace a string value? : str.replace()

*You are correct! The answer “*str.replace()” is*correct because the function replace() is used to*replace a value.

to\_numeric(“column”) converts the data type of column into (blank). : int64

*You are correct! The answer “*int64” is*correct because the function*to\_numeric(“column”)*in Python converts the data type of column into int64.*

**Discussion Activity**

**4.1**

**Does income differentiate customers who purchase wine?**

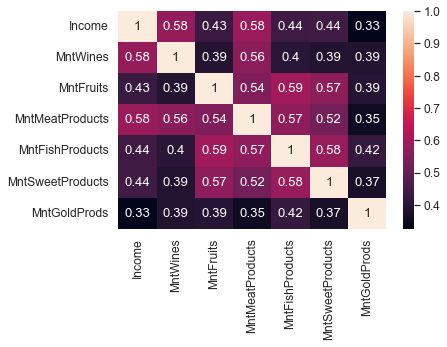
plt.tight\_layout()

sns.set(font\_scale=1.1)

sns.heatmap(data = df[['Marital\_Status', 'Income', 'MntWines', 'MntFruits',

'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',

'MntGoldProds']].corr(), annot=True)



As shown in the heat map, income plays a significant role in wine purchase, they are strongly correlated as well as meat products, there is also high correlation with other purchases too.

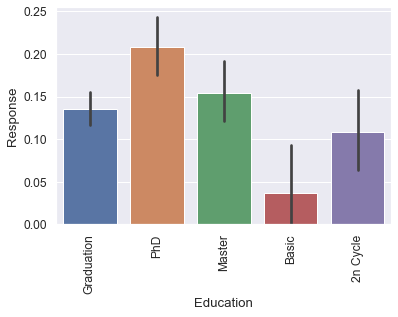
**What customers are more likely to participate in the last promotional campaign?**

sns.barplot(data = df[['Marital\_Status', 'Income', 'Education',

'AcceptedCmp3', 'AcceptedCmp4', 'AcceptedCmp5', 'AcceptedCmp1',

'AcceptedCmp2', 'Response']], x='Education', y='Response')

plt.xticks(rotation=90)



I looked at the education level of individuals, as the education gets higher the more participation in promotional campaigns.

**Income versus Monthly Web Visits**

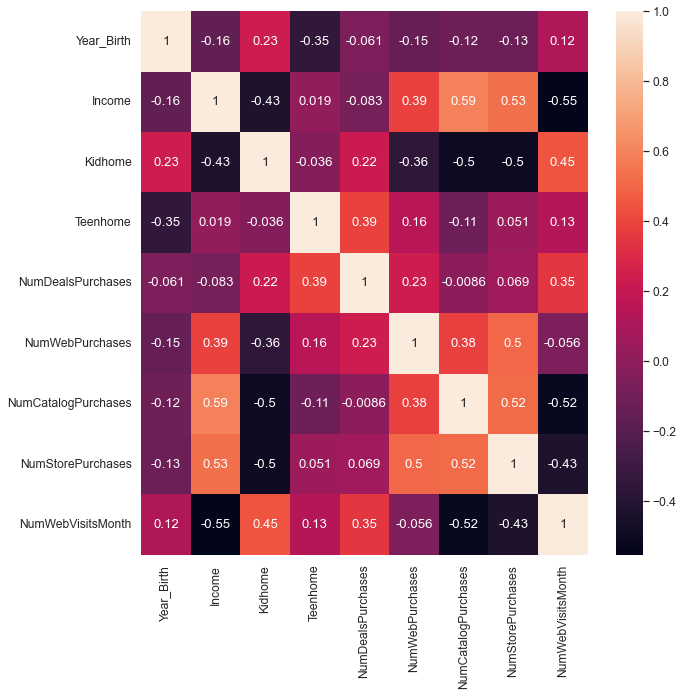
plt.tight\_layout()

plt.subplots(figsize=(10,10))

sns.heatmap(data = df[['Year\_Birth', 'Education', 'Marital\_Status', 'Income', 'Kidhome',

'Teenhome', 'Dt\_Customer','NumDealsPurchases', 'NumWebPurchases',

'NumCatalogPurchases', 'NumStorePurchases', 'NumWebVisitsMonth']].corr(), annot=True)



The result in this dataset is interesting as income increases less frequent web visits, however, catalog and in-store purchases higher as well as web purchases surprisingly.

**Complaints about Product**

sns.barplot(data=df.query('Complain == 1')[['MntWines', 'MntFruits',

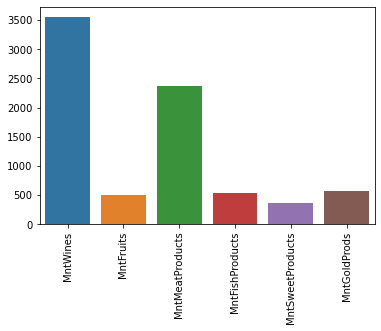
'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',

'MntGoldProds', 'Complain']].groupby('Complain').sum().reset\_index()[['MntWines', 'MntFruits',

'MntMeatProducts', 'MntFishProducts', 'MntSweetProducts',

'MntGoldProds']])

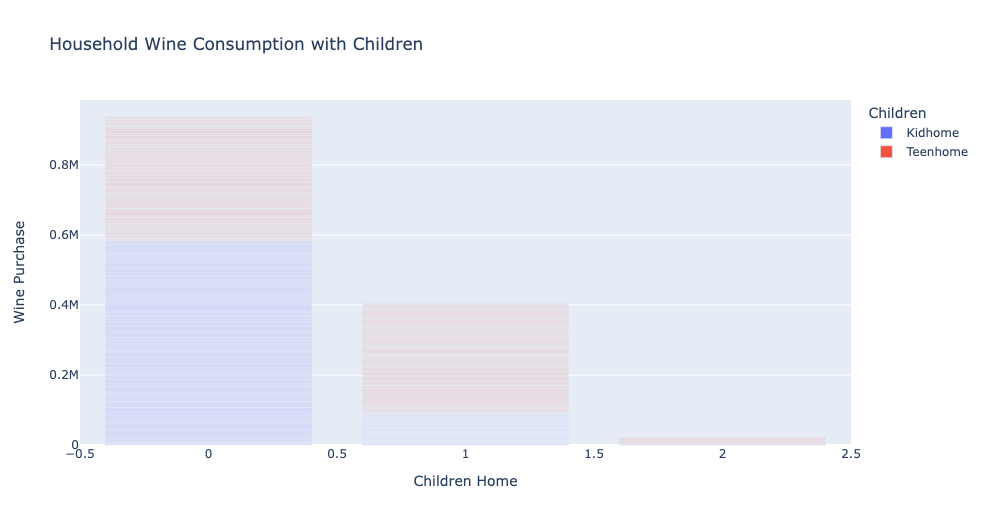
plt.xticks(rotation=90)



Wine and Meat products lead number of complaints.

**Do people with children purchase more wine?**

px.bar(df[['Marital\_Status', 'Kidhome', 'Teenhome', 'MntWines']], x=['Kidhome','Teenhome'], y='MntWines', title='Household Wine Consumption with Children', labels={"value":"Children Home", "MntWines":"Wine Purchase", "variable":"Children"})



Wine consumers are households without any children by far.

**4.2**

**Transformation**

Features in this dataset are not to start with, there are some critical information taken out (deliberately) like “*income*” which makes harder to correlate. I analyzed fields by value\_counts() method: *credit.savings\_status.value\_counts().*

**Identification**

However, I converted almost all object fields to numeric, so, I can correlate them all to see on *heat map*, I cleaned up a few fields which do not correlate:

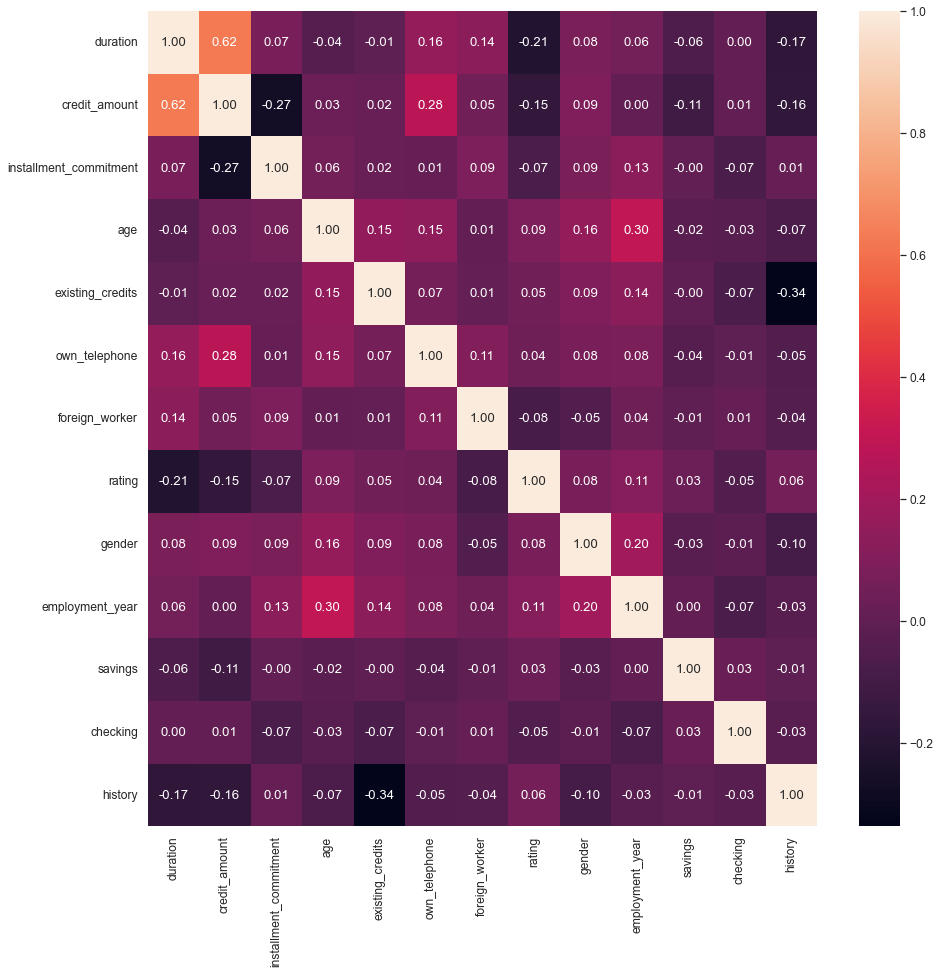
plt.tight\_layout()

plt.subplots(figsize=(15,15))

sns.set(font\_scale=1.1)

sns.heatmap(credit[['duration', 'credit\_amount', 'installment\_commitment', 'age', 'existing\_credits', 'own\_telephone',

'foreign\_worker', 'rating', 'gender', 'employment\_year', 'savings', 'checking', 'history']].corr(), annot=True, fmt='.2f')



Credit *rating* numeric bad=0 or good=1 as seen in heat map there is no strong correlation, two fields stand out *duration* and *credit\_amount* which are negative correlation, there is also weak positive correlation to *employment\_year* (length of employment in years) towards good credit. Other features have negligible impact on the credit rating!

**Deep Dive**

Next, I looked at these 3 features more closely, I used Plotly violin plot to identify thresholds of each for eyeballing.

Credit Amount, risk increases as the amount gets bigger, so, I captured the median value from good credit. $2244.

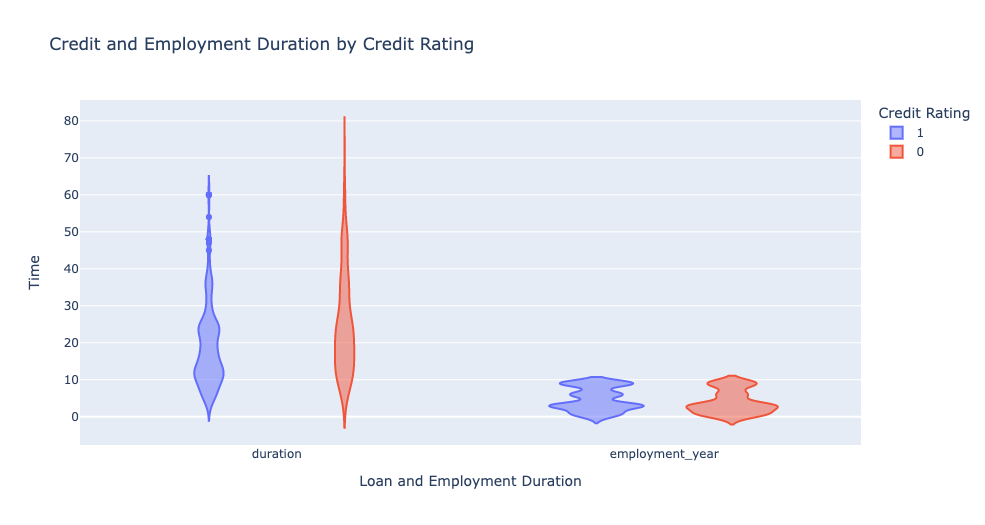
Loan duration, risk increases as loan duration gets longer, so, I captured the upper fence value from good credit. 42 months.

Employment duration, risk gets lower as employment gets longer, so, I captured 3rd quartile value 6 years from the violin plots, although, it has very less influence on correcting numbers by only 2%:

px.violin(credit[['credit\_amount','rating']], color='rating', title='Credit Amount by Credit Rating', labels={"variable":"Credit Amount", "value":"$", "rating":"Credit Rating"})



px.violin(credit[['duration','employment\_year','rating']], color='rating', title='Credit and Employment Duration by Credit Rating', labels={"variable":"Loan and Employment Duration", "value":"Time", "rating":"Credit Rating", "duration":"Loan Duration", "employment\_year":"Employment Duration"})



**Threshold Result**

Finally, I applied the threshold values to the formula to detect people with bad credit ratings:

credit[(credit['duration'] > 42) & (credit['employment\_year'] < 6) & (credit['credit\_amount'] > 2244)].rating.value\_counts(normalize=True)

0 0.588235

1 0.411765

Name: rating, dtype: float64

versus 30% bad, 70% good credit rating in the entire dataset!

 ————— o —————

**Module 5**

[Video Transcript](https://student.emeritus.org/courses/4765/files/2982971?wrap=1)

titanic['fare'].plot(kind='hist', x='fare', color = 'lightblue', **alpha=0.6**, edgecolor= 'black', bins= 30, figsize = (12, 5))

plt.show()

# Remove empty Bar rows for analysis

bars = bars.dropna(subset=['Bar'])

**Savio’s session:**

plot(kind=‘pie’, explode=[0.1,0.1], autopct=‘%1.1f%%’, colors=[‘g’, ’r’] )

pd.crosstab (ibm[‘Attrition’], ibm[‘Gender’], normalize=True)

help(pd.crosstab)

*Nominal variable, ordinal variable?*

plt.subplot(121)

sns.countplot(‘Gender’, data=ibm, hue=‘Attrition’)

sns.distplot( )

Savio’s IBM attrition dataset :<https://github.com/SavioSal/datasets/raw/master/HR-Employee-Attrition.csv>

Savio’s board: <https://colab.research.google.com/drive/1m640WoWTbTgw0ymZoIw9K0wyCfMXtbYH#scrollTo=8veffxKiwKfu>

 ————— o —————

**Module 6**

* [Video Transcripts](https://student.emeritus.org/courses/4765/files/3055193?wrap=1)
* [Download Video Transcripts](https://student.emeritus.org/courses/4765/files/3055193/download?download_frd=1)
* [Quick Reference Guide](https://student.emeritus.org/courses/4765/files/3055194?wrap=1)

**Notes:**

Principal Component Analysis (PCA)

Singular Value Decomposition (SVD)

KMeans and DBScan clustering

Density-based spatial clustering of applications with noise (DBSCAN)

**DBSCAN**

A density-based clustering non-parametric algorithm that groups together data points that are closely packed together, marking as outliers points that lie alone in low-density regions

**K-Means**

A method of vector quantization that aims to partition n observations into k clusters in which each observation belongs to the cluster with the nearest mean (cluster centers or cluster centroid), serving as a prototype of the cluster

**K-Means++**

An algorithm for choosing the initial values or ‘seeds’ for the k-means clustering algorithm

**Principal Component Analysis (PCA)**

The process of computing principal components and using them to perform a change of basis on the data, sometimes using only the first few principal components and ignoring the rest

**Singular Value Decomposition (SVD)**

The factorization of a real or complex matrix

mu = X.mean()

sigma = X.std()

Xnorm = (X - mu)/sigma

Sigma = np.diag(sigma)

U, Sigma, VT = svd(Xnorm)

reconstruct\_X = U @ Sigma @ VT

percent\_variance\_explained = sigma / sigma.sum()

print(np.cumsum(percent\_variance\_explained[:22]))

np.allclose(df['cluster label0'], df['cluster label1'])

**Module Issues:**

**Activity 6.1 Problem 5:** “Standardize the singular values in the first ten entries of Sigma by dividing each by the sum of the main diagonal”, the statement is not clear to use sum of diagonal of Sigma!

**Activity 6.2 Problem 4:** Solution does not normalize dataset before SVD which is wrong!



**Activity 6.6 Problem 2:** hue = 'label' is supposed to be 'cluster label’!

**Activity 6.7 Problem 3:** variable inertia\_8\_centers is supposed to be inertia

**Try-It activity 6.1:**

credit.csv is missing in data folder! Fixed.

'default payment next month' supposed to be 'default.payment.next.month'

**Try-It activity 6.2:**

images/segments.jpeg is missing in the zip file.

**Quizes:**

PCA is used for clustering. :- False

*You are correct! The answer “False” is correct because the principal component analysis is used for dimensionality reduction.*

PCA looks for linear combinations of existing (blank) that capture the bulk of the variance. : Columns

*You are correct! The answer “Columns” is correct because PCA looks for new columns that are linear combinations of existing columns and capture the bulk of the variation in the data*.

The “curse of dimensionality” states that the amount of data you need to train a model increases exponentially with the number of inputs. : True

*You are correct! The answer “True” is correct because the amount of data you need to train a model increases exponentially with the number of inputs is stated as the “curse of dimensionality”.*

What does running SVD on X decompose X into? : U Σ V

*You are correct! The answer “*U Σ V*” is correct because running singular value decomposition (SVD) on X will decompose it into three matrices: U,*Σ,*and V.*

What is the formula to normalize a dataset X? : Xnorm = (X−μ)/σ

*You are correct! The answer “*Xnorm = (X−μ)/σ*” is correct because this is the formula to normalize the dataset X.*

What is the function in Python library “scipy.linalg” to compute the singular value decomposition? : svd()

*You are correct! The answer “*svd()*” is correct because this is the function in Python library “scipy.linalg” to compute the singular value decomposition.*

The Python function “numpy.allclose()” is used to find whether two arrays are element-wise equal. : True

*You are correct! The answer “True” is correct because the function “*numpy.allclose()*” is used to check whether two arrays in numpy are element-wise equal or not.*

In SVD, the matrix sigma has all the diagonal values as zero. : False

*You are correct! The answer “False” is correct because the matrix sigma in SVD is a diagonal matrix in which, other than the diagonal values, all other values are zero.*

Below is the equation to represent the multiplication of the matrices for SVD: ∑Di=1 σiuivit What does the parameter *D*represent? : Principal components

*You are correct! The answer “Principal components” is correct because in the formula for the multiplication of matrices in SVD, the parameter D represents the number of iterable principal components.*

The Matrix Σ in SVD has values “σi” in the diagonal of the matrix, which represent the importance of the i’th component for the dataset. : True

*You are correct! The answer “True” is correct because the value of*“σi”*conveys the importance of the i’th principal component for the dataset.*

The formula used to project the data into desired dimensions is

“ x̄rr  = Ur ∑r ”

The parameter “r” is defined as the total number of columns. : False

*You are correct! The answer “False” is correct because the parameter “r” is defined as the number of principal components that are selected.*

What is the symbol used for matrix multiplications? : @

*You are correct! The answer “@” is correct because this symbol is used for matrix multiplications.*

Clustering is a method for creating groups out of the columns of a dataset. : False

*You are correct! The answer “False” is correct because clustering is a method for creating groups out of the rows of a dataset.*

Clustering is an unsupervised machine learning model. : True

*You are correct! The answer “True” is correct because clustering has no labeled datasets.*

How is the centroid of a cluster “k” in k-means clustering represented? : μk

*You are correct! The answer “*μk*” is correct because the mean of each cluster is declared as the centroid of that cluster, which, for cluster k, is “*μk*”.*

In the K-means clustering algorithm, how is inertia defined? : Sum of the squared distances from points to their centroids

*You are correct! The answer “Sum of the squared distances from points to their centroids” is correct because inertia in K-means is the summation of the squared distances of data points from their respective centroids.*

The stepwise sequence for the K-means algorithm is as follows:

1. Assignment
2. Updation

: False

*You are correct! The answer “False” is correct because K-means starts with updating of centroids initially and then assigning data points to the nearest centroids.*

What is the stopping criteria for K-means clustering? : Assignment step has no change of data points

*You are correct! The answer “*Assignment step has no change of data points*” is correct because the stopping criteria for K-means clustering is when assignment stops changing.*

Consider this dataframe:

Dataframe

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| **CustomerID** | **Gender** | **Age** | **Annual Income (k$)** | **Spending Score (1-100)** |
| 1 | Male | 19 | 15 | 39 |
| 2 | Male | 21 | 15 | 81 |
| 3 | Female | 20 | 16 | 6 |
| 4 | Female | 23 | 16 | 77 |
| 5 | Female | 31 | 17 | 40 |
| ... | ... | ... | ... | ... |
| 196 | Female | 35 | 120 | 79 |
| 197 | Female | 45 | 126 | 28 |
| 198 | Male | 32 | 126 | 74 |
| 199 | Male | 32 | 137 | 18 |
| 200 | Male | 30 | 137 | 83 |

What should the index of this dataframe be set to? : CustomerID

*You are correct! The answer “*CustomerID*” is correct because the index of a dataframe should be a unique value for every row.*

In the Python function “KMeans()”, the constructor ‘‘init’’ is used to select the criteria for the initialization of data points. : False

*You are correct! The answer “False” is correct because the constructor ‘init’ is used to select criteria for the initialization of the centroids of the clusters.*

In “KMeans()”, how do you generate the array that tells which cluster the data point belongs to? : kmeans.labels\_

*You are correct! The answer “*kmeans.labels\_*” is correct because the statement is used to get an array that tells which data point belongs to which cluster.*

The default initialization in K-means is random initialization. : False

*You are correct! The answer “False” is correct because the default initialization in K-means is improved initialization, which is K-means++.*

K-means++ only finds centroids once each time you run the function. : False

*You are correct! The answer “False” is correct because K-means++ finds the initial centroids and then searches again in an attempt to lower the inertia of the dataset.*

The number of clusters for DBSCAN are declared beforehand. : False

*You are correct! The answer “False” is correct because the DBSCAN algorithm is centroid-less and the number of clusters arise naturally from the algorithm.*

What is the clustering algorithm which has the ability to create curved boundaries between clusters? : DBSCAN

*You are correct! The answer “DBSCAN” is correct because it has the ability to create curved boundaries between clusters.*

Points that are sufficiently removed from other points are designated by DBSCAN as (blank). : Outliers

*You are correct! The answer “Outliers” is correct because DBSCAN has a built-in outlier detection feature*.*Points that are sufficiently removed from other points and not classified at all are called outliers.*

In the Python function “cluster.DBSCAN()”, the constructors of the function are (blank). *(Check all that apply.)* : min\_samples, eps

*You are correct! The answers “eps” and “min\_samples” are correct because these are the constructors for the function of DBSCAN.*

If the ball of radius epsilon captures less than min\_sample points, then that point is designated as a core point. : False

*You are correct! The answer “False” is correct because if the ball of radius epsilon captures at least min\_sample points, then that point is designated as a core point.*

How does DBSCAN declare a point as an outlier? : Points with no core or boundary points in their epsilon ball radius

*You are correct! The answer “Points with no core or boundary points in their epsilon ball radius” is correct because such data points that do not have any core or boundary point in their epsilon are declared as outliers.*

**Savio’s**

*sse = {}*

*for k in range(1,10):*

*kmeans = KMeans(n\_clusters = k, max\_iter = 1000).fit(df\_pca)*

*sse[k] = kmeans.inertia\_*

*plt.figure()*

*plt.plot(list(sse.keys()),list(sse.values()))*

*Feature vs categorical*

*df.groupby(‘cluster’).*

*Use standard scaler for try-it 6.2!*

**6.1: Summarizing Data with PCA - Section B**

For PCA analysis, we can take a look at the scree plot to decide how many components to keep. First, let’s normalize out dataset:

# normalize data

default\_norm = (default - default.mean()) / default.std()

We can set number of components to a large value first to analyze components visually to decide how many to keep:

# initialize to 10 components to visualize Scree Plot first

pca = PCA(n\_components = 10, random\_state = 42)

pca.fit\_transform(default\_norm)

Then, draw a scree plot to visualize components:

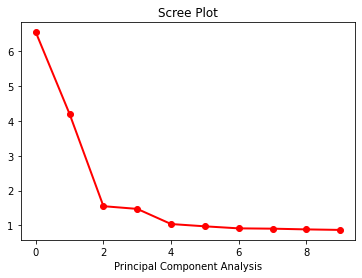
# Scree Plot

plt.plot(pca.explained\_variance\_, 'ro-', linewidth=2)

plt.title('Scree Plot')

plt.xlabel('Principal Component Analysis')

plt.show()



Most significant components are first 2, however, one may pick up to 4 components but I chose only first 2 for further scatter plot analysis:

# initialize to 2

pca = PCA(n\_components = 2, random\_state = 42)

components = pca.fit\_transform(default\_norm)

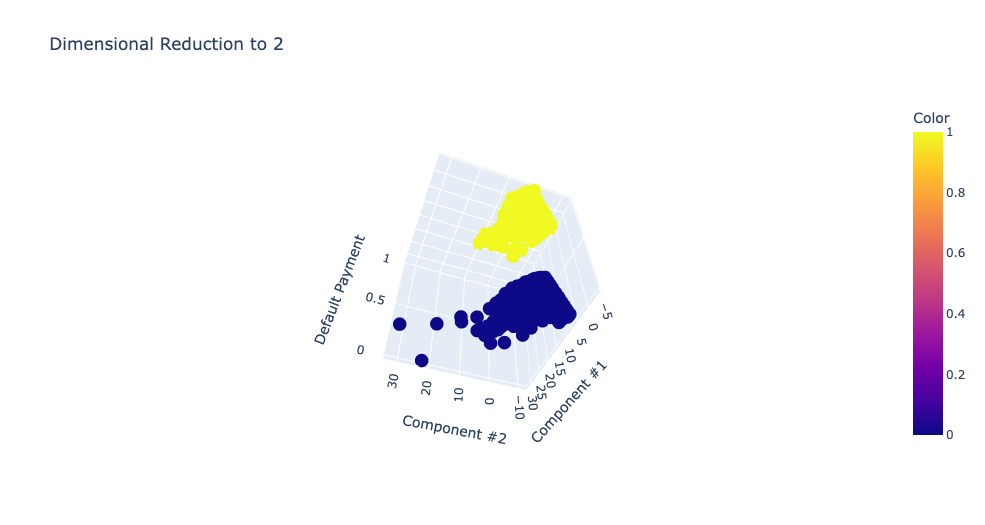
# Scatter plot by default.payment.next.month

px.scatter\_3d(x=components[:, 0], y=components[:, 1], z=default['default.payment.next.month'],

color=default['default.payment.next.month'],

title='Dimensional Reduction to 2',

labels={"z":"Default Payment", "x":"Component #1", "y":"Component #2", "color":"Color"})



As shown in the 3D scatter plot, it did not achieve the goal by r=2 since there is no a clear line between these two groups by default.payment.next.month, in fact, they are overlapping because ‘Default Payment’ is not significant enough by itself to classify data points close to each other per principle component. Also, these 2 principle components only represent 45% of dataset, but, I do not think representing more of the dataset would change the scatter plot.

# Cumulative sum of ratios

np.cumsum(pca.explained\_variance\_ratio\_)

array([0.272986 , 0.44804925, 0.51275778, 0.57423046, 0.61758316,

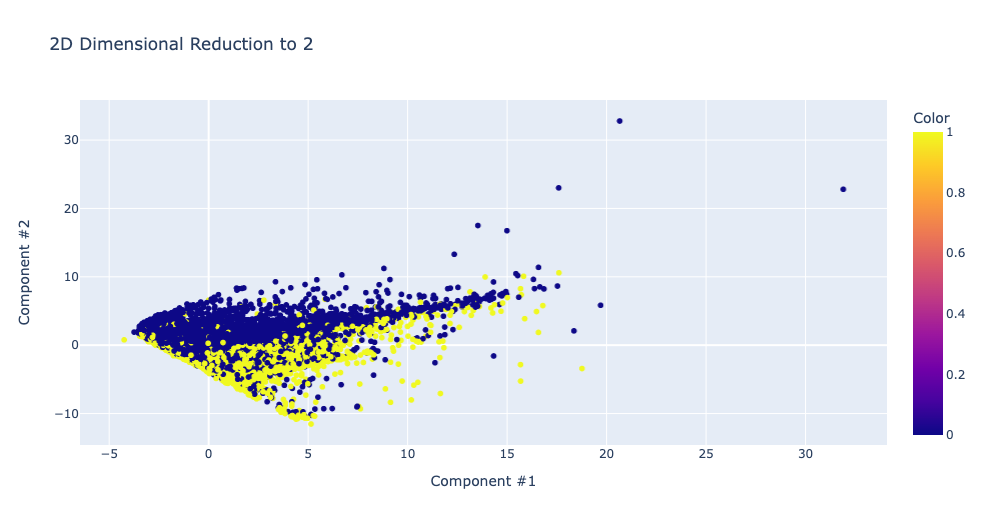
0.65818493, 0.69635315, 0.7341443 , 0.77106995, 0.8073719 ])

# Scatter plot by default.payment.next.month

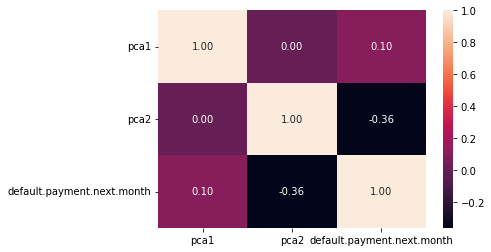
px.scatter(data\_frame=components, x=0, y=1, color=default['default.payment.next.month'],

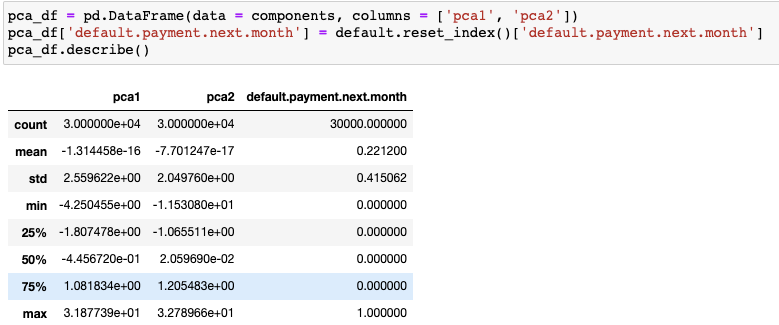
title='2D Dimensional Reduction to 2',

labels={"0":"Component #1", "1":"Component #2", "color":"Color"})



sns.heatmap(pca\_df.corr(), annot=True, fmt='.2f')





**6.2: Interpreting the Results of K-Means and PCA - Section B**

For PCA and clustering, first the dataset needs some cleaning and transformation as below.

**Data Preparation:**

* Transform all Yes/No columns to 1/0
* Transform Gender Male/Female to 1/0
* Analyze columns Offer, Internet Type, Contract, Payment Method, Churn Category and Churn Reason
* Fill null Customer Satisfaction with mean value 3.005453
* Fill null Churn Reason with ‘Don't know’
* Fill null Churn Category with ‘Other’
* Set index to Customer ID
* Drop column City since there is zip code and latitude & longitude
* Drop dependents since there is number of dependents
* Drop Under 30 and Senior Citizen since there is Age column

**Dimensional Reduction:**

Initialize r=20 to analyze PCA to decide how many to keep:

# initialize r to 20 components to visualize Scree Plot first

pca = PCA(n\_components = 20, random\_state = 42)

pca.fit\_transform(df\_norm)

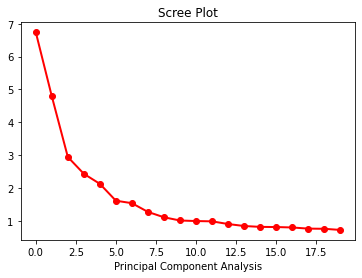
Scree plot

plt.plot(pca.explained\_variance\_, 'ro-', linewidth=2)

plt.title('Scree Plot')

plt.xlabel('Principal Component Analysis')

plt.show()



I checked cumulative ratios as well for 20:

# Draw a cumulative sum to decide how many to keep

np.cumsum(pca.explained\_variance\_ratio\_)

array([0.1644471 , 0.28131084, 0.35310188, 0.4124016 , 0.46424793,

0.50356417, 0.54105027, 0.5720512 , 0.59907129, 0.6236999 ,

0.64791467, 0.671911 , 0.69391559, 0.7144539 , 0.73434008,

0.75409989, 0.77354679, 0.79214295, 0.81060817, 0.82823602])

r=6 seems represents more than 50% of features in the dataset.

**Clustering Dataset:**

After running k-means up to 10 clusters by checking inertia, I chose 4 clusters to execute k-means:

# kmeans inertia check

inertias = []

for i in range(1, 11):

kmeans = KMeans(n\_clusters = i, init='k-means++', random\_state=42).fit(components)

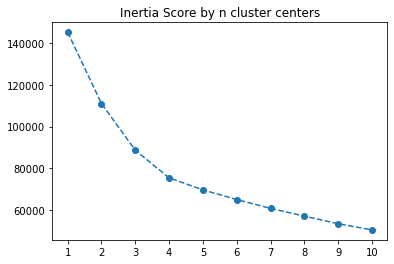
inertias.append(kmeans.inertia\_)

print(inertias)

plt.plot(list(range(1, 11)), inertias, '--o')

plt.xticks(list(range(1, 11)), list(range(1, 11)))

plt.title('Inertia Score by n cluster centers')



# kmeans

kmeans = KMeans(n\_clusters = 4, random\_state = 42).fit(components)

#df\_clean['cluster'] = kmeans.labels\_

np.unique(kmeans.labels\_)

**Results:**

There are many features in this dataset that one can find many different results, listing a few samples

**Total Charges:**

Cluster 3: Customer segment with total regular charges between $2000 and up, high long distance charges and no extra data charge.

Cluster 1: Customer segment with total regular charges up to $2000 and high long distance charges and no extra data charge.

Cluster 2: Customer segment with total regular charges up to $3000 and moderate long distance extra data charges.

Cluster 0: Customer segment with total regular charges up to $4000 and moderate long distance charges and no extra data charge.

px.scatter\_3d(data\_frame=df, x = 'Total Regular Charges',

y = 'Total Extra Data Charges', z = 'Total Long Distance Charges', color = 'kmeans',

title='Charges by Clusters',

labels={"kmeans":"Clusters"})



**Churn:**

Cluster 3: Customer segment with 2 year contract

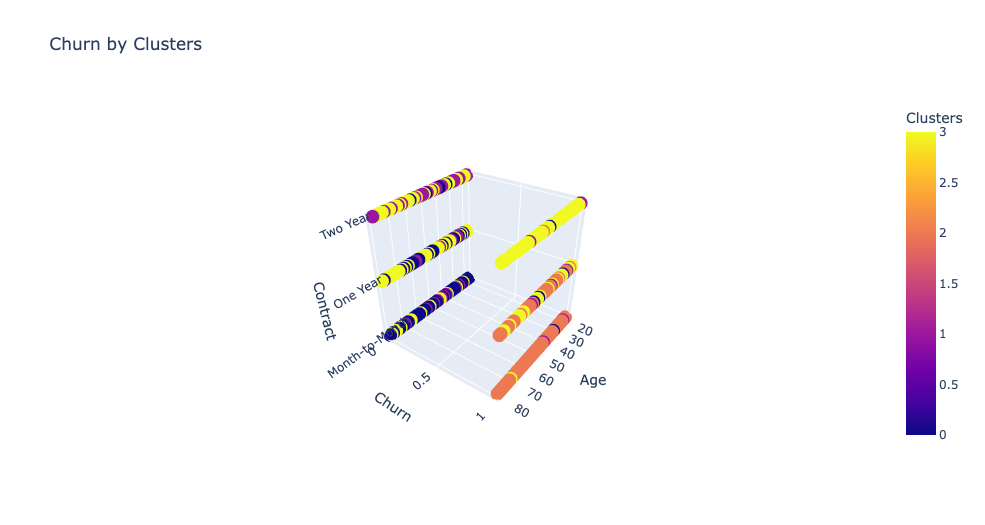
Cluster 2: Customer segment with month-to-month contract

Cluster 0: Customer segment who stays with month-to-month contract

px.scatter\_3d(data\_frame=df, x = 'Age', y = 'Churn Value', z = 'Contract', color = 'kmeans',

title='Churn by Clusters',

labels={"Churn Value":"Churn", "kmeans":"Clusters"})

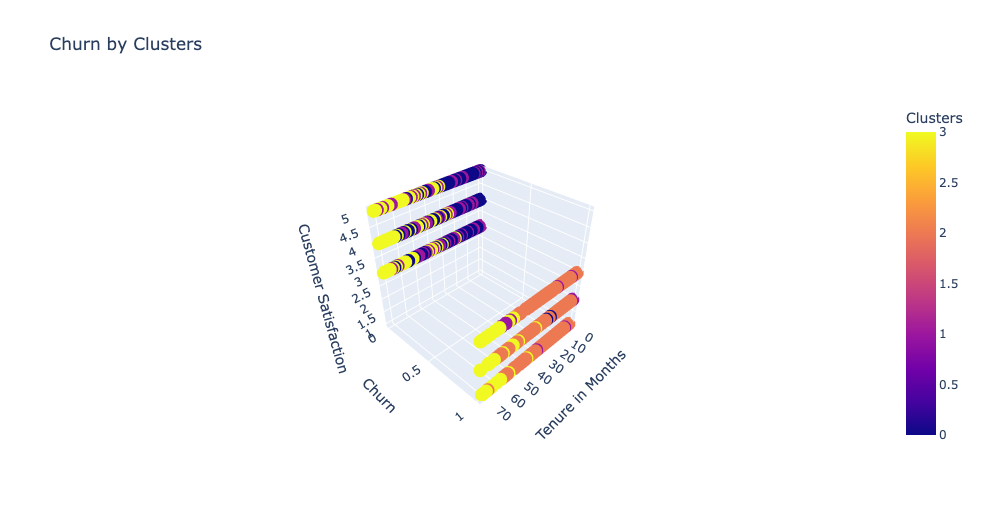


Also, Customer segment in Cluster 2 with month-to-month contract, points to low customer satisfaction which is 3 and below.

px.scatter\_3d(data\_frame=df, x = 'Tenure in Months', y = 'Churn Value', z = 'Customer Satisfaction', color = 'kmeans',

title='Churn by Clusters',

labels={"Churn Value":"Churn", "kmeans":"Clusters"})



**Total Spending:**

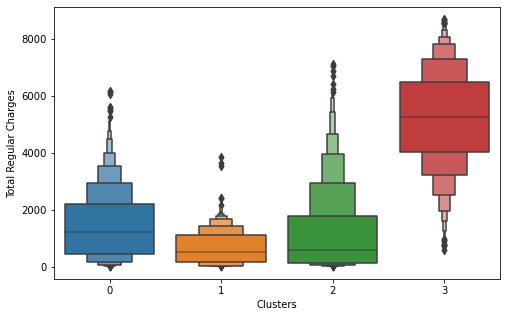
This is another angle about the customer segment in Cluster 3 which is clearly high-spending by buying all services and Cluster 1 is low-spending customer segment who buy phone service and nothing else.

plt.tight\_layout()

plt.subplots(figsize=(8,5))

sns.boxenplot(data=df, x='kmeans', y='Total Regular Charges')

plt.xlabel('Clusters')



**Break-down by Service:**

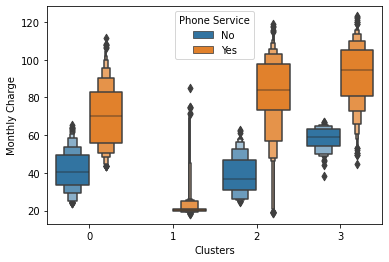
Customer segment in Cluster 1 goes with Phone Service only, they do not buy any other service, low paying customer segment.

plt.tight\_layout()

plt.subplots(figsize=(6,4))

sns.boxenplot(data=df, x='kmeans', y='Monthly Charge', hue='Phone Service')

plt.xlabel('Clusters')



**Overall Conclusion**

Checked overall dataframe by grouping it by cluster to analyze entire dataframe:

*Cluster 0* is total spending is second highest in this customer segment, lesser on phone service, on month-to-month, they go with unlimited data more, tend to be single, fewer dependents, goes with paperless billing.

*Cluster 1* is low-spending customer segment phone-only service, mid-age, tenure is longer, on typically 1 year contract, marital status is mixed, highest number of dependents, second most referrals, pay by credit card, goes with paper billing.

*Cluster 2* is high churn-rate due to attitude of support person, this segment ask most customer service requests by reporting product/service issues, low customer satisfaction, more mature folks, on month-to-month, tend to be single, none to fewer dependents, total spending is less due to short tenure but monthly charge is second highest, fewest referrals, pay by bank, goes with paperless billing.

*Cluster 3* is high-spending customer segment by buying all services, longest tenure in this segment, on typically 1 year contract, married, some dependents, they pay highest monthly charges, refer many friends, goes with paperless billing.

————— o —————

**Mentoring**

<https://ml-ops.org/>

<https://www.credly.com/skills/hadoop> -> is arama

**Resume Career Help:** [**danielle.fortier@emertius.org**](mailto:danielle.fortier@emertius.org)

 ————— o —————

**Module 7**

Notes:

Module Issues:

Quizes: