Introduction

With their respective specialties in data analysis, Tableau and Python are crucial tools for data scientists and business analysts. Tableau is great at making interactive, aesthetically pleasing dashboards. It also has an easy-to-use interface for ad hoc analysis and helps with teamwork. It is perfect for sharing insights and facilitating visual exploration because it connects to a variety of data sources with ease. Python, on the other hand, is a strong programming language that is frequently used for data preprocessing, machine learning, and advanced analytics. Custom analytics, algorithm creation, and system integration are its strong points. End-to-end analytics is made possible by the potent synergy between Tableau and Python when used in cooperation. Analysts may use Python for sophisticated analytics and Tableau for user-friendly visuals, offering a complete solution for making decisions based on information. This cooperative method improves effectiveness, adaptability, and capacity to handle various data analysis needs.

Tableau

I first started using Tableau when I needed to display certain visualizations while working on a project in previous company. From that day, I developed a certain interest into this software. It's a boon who want to work on databases and extract important information from it. Some of its uses are:

Dashboards: With Tableau, analysts can build dynamic dashboards that let users examine and engage with data in real time. This interactivity helps stakeholders comprehend the data more deeply and improves the exchange of insights.

Investigating and Analyzing Data: Without requiring a lot of coding, Tableau enables researchers to examine datasets, connect to different data sources, and do ad hoc analysis. To examine habits, trends, and outliers, users can drag and drop fields.

At first, I took a free course for it from YouTube. It was 9.5 hours long coursre that covered basics to expert level. Later I took a 6 hours long Tableau course from datacamp. I am including the Datacamp's certification below.

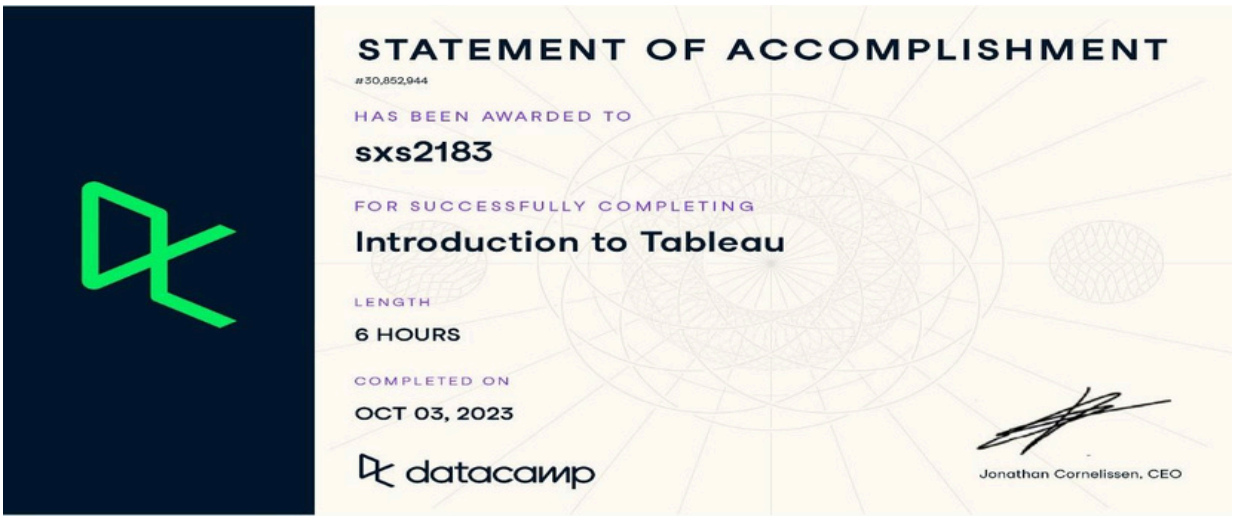


Tableau is a potent business intelligence and data visualization application that assists companies in transforming unstructured data into meaningful insights. These are a few of Tableau's main attributes and advantages:

Linking and Integrating Data: Tableau has the ability to link to numerous data sources, including as databases, spreadsheets, cloud-based data, and more. Provides easy interface with widely used data sources, including Salesforce, SQL, Excel, and others.

Simple Drag & Drop Interface: With an easy drag-and-drop interface, users can build dynamic dashboards and reports that are suitable for both non-technical and technical users.

Strong Visualization Features: Offers numerous visualization choices, such as heat maps, bar charts, line charts, scatter plots, and more. It facilitates the mapping and analysis of geographic data using geospatial graphics.

Making a Dashboard: Permits the development of dynamic, interactive dashboards that let users examine and evaluate data in real time. Dashboards can offer a thorough view of the data with a variety of filters and visualizations.

Analyzing Data in Real Time: Enables real-time data analysis and visualization for users, guaranteeing that insights are based on the most recent data.

Benefits of using Tableau

Simpleness of Use: Without requiring complex technical knowledge, users can easily create, modify, and interact with visualizations due to Tableau's straightforward design and user-friendly interface.

Quick Deployment: Organizations can quickly begin extracting insights from their data through fast setup and deployment.

Economical: Provides affordable options for businesses of all sizes with a variety of license options, including scalable offerings and free versions.

Scalability: Scales well from small-scale tasks to enterprise-wide installations, meeting the expanding data needs of a company.

Community Assistance: A sizable and vibrant user base that exchanges information, resources, and guidelines, offering Tableau users invaluable support.

Project

I am showcasing a project I undertook during my post graduation named as ‘Coniston Project’. I analyzed the dataset given to me and made sheets and two dashboards out of it that comments upon different aspects of the dataset. I will explain the knowledge that it conveys and process that I used to make the Dashboards and Stories.

The below task is done exclusively by me:

https://public.tableau.com/app/profile/sukhman.singh1298/viz/MicroAnalytics/EffectandShareofGreenhousesGases?publish=yes

I am also adding Coniston Group Task by me and my group members:

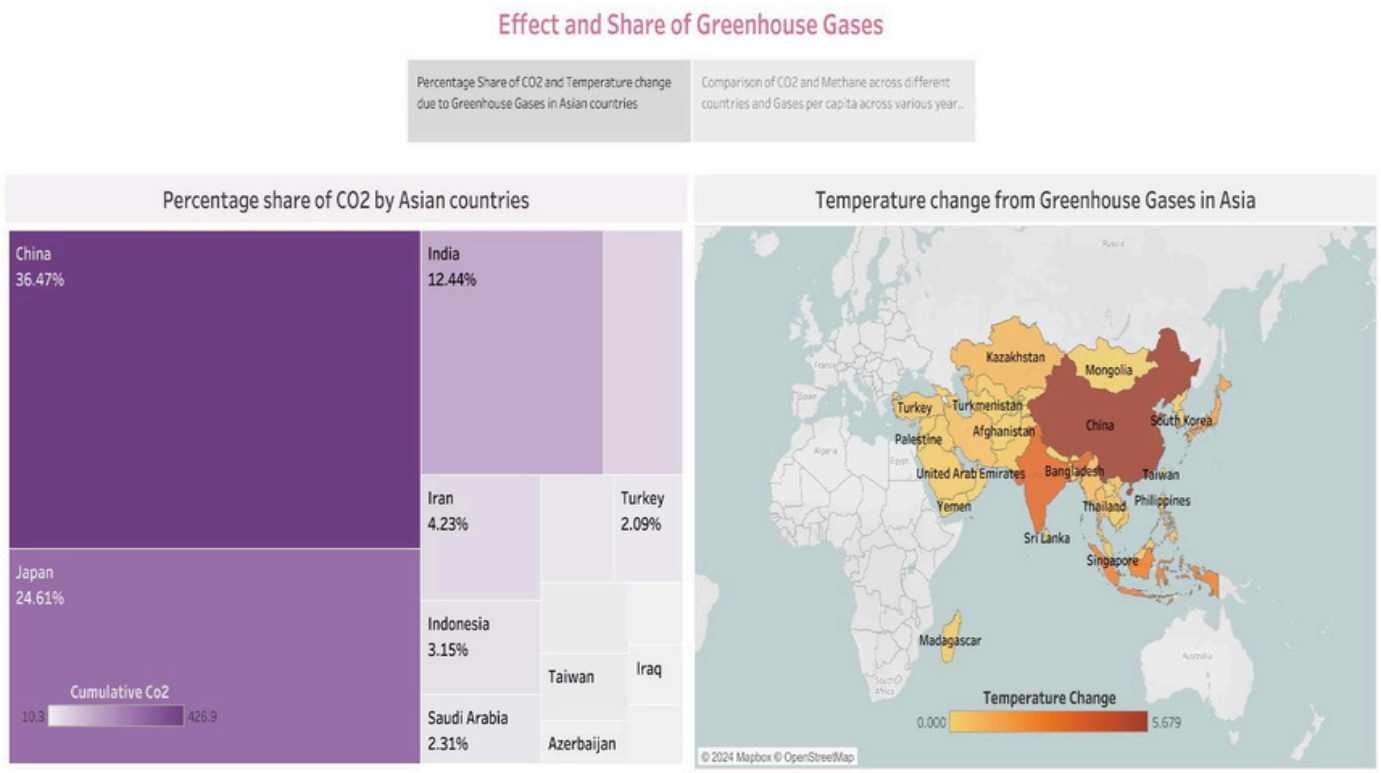
https://public.tableau.com/views/Coniston_group/Story1?:language=en-US&:display_count=n&:origin=viz_share_link

In the task done by me, I made **4 sheets,** then made **2 Dashboards** out of it and **1 story** containg **2 dashboards** to cover each feature and convey knowledge through dashboard.

I will be sharing **2 images of the story, each image represents 1 dashboard**.

For assessment, I am including links of Tableau Public Gallaries down so that they can be accessed and commented upon:

Dashboard 1



Percentage share of CO2 by Asian countries

The above graph is a **Treemap** that states the percentage **share of CO2 by top 15 countries** in **Asia**. For construction of this graph, I selected **'Country'** and **CO2 Cumulative share** from the Tables section in the Tableau. Then I selected Treemap graph option under **'Show Me'** section on Tableau on the top right corner.

I performed **'Quick Table Calculation'** to **'Culmulative CO2 Share'** and made it **'Percentage of Total'**. Therefore the graph represents the percentage share of total CO2 with all the countries labelled.

Reflective Analysis: Here I selected share global cumulative CO2 from measures and countries from dimensions to make this graph. **Color grading** and **size** of the blocks makes it easier to compare different countries, that can be re-checked with the mentioned percentages too.

Temperature change from Greenhouse Gases in Asia

This graph represents temperature change from Greenhouse gases in the Asian countries. For constructing this graph I double clicked on the **country** tab and put countries tab in the **filter** to aggregate the Asian countries. Later I put **'Temperature change from Ghg'** into the color section. The darker the color gets, more temperature change it represents.

Reflective Analysis: Color grading helps us to easily compare different countries. Tableau enabled me to give a impressive visual presentation of world map and temperature change on it. Labelling of countries added further feathers. I was able to put **Filter of Top 20 countries** easily that helped me in analysis. The features Tableau provides with data visualization are quite impressive and handy to use.

CO2 and Methane gas emissions in top 10 countries in Asia

This graph visualizes **top 10 countries** by **CO2** and **Methane** gas emissions in **Asia**. China tops the list followed by Iran, Saudi Arabia and India. **Reference line** has been added that represents the average value of CO2 and Methane emitted of top 10 countries. Reference lines are good to visualize the contribution of a partiular country when compared to average. ***For counstructing this graph***, I first plotted the country, SUM Gas CO2 and SUM Methane onto the graph and change the measure category to Average. Changed the chart type of **CO2** to **Bar** and chart type of **Methane** to **Line** chart. I gave **green** color to Bar and **light yellow** color to line chart to efficiently distinguish the two. Adequate colors to reference lines have also been provided.

Reflective Analysis: Bar and Line chart gave me a good visual presentation on comparison between CO2 and Methane levels in different countries. Tableau gives us a feature of color coding. Green(CO2) and Yellow(Methane) helped me to effeciently distinguish between CO2 and Methane levels. Reference lines helped me to analyze how much more or less contribution is there from the mean value in Global Air Pollution.

Comparison of different Gases per Capita with their trend lines in Asia

Here we are comparing the Emission of **4 different gases** per capita into atmosphere from **2009 to 2019**. This graph is a line chart that showcases the **Trend lines** that gives us quite an idea about what is about to happen in future years for different gases. A probable reason we are seeing this line graph straight is because **gases per capita** means the **'Amount of Gas'/'Total population'** of all the **Asian** countries. As

country's population keeps on increasing with increasing time, along with the amount of these gases, the line on the graph remains stable.

Reflective Analysis: I was able to use good combination of colors are used to distinguish pane, rows and column's area. In this graph, we should keep in mind that as emissions of any gas increases, its population is also increasing year by year. Formula of gas per capita is: **Emissions / Total population**. Therefore, one can see almost linear line for some gases in the graph with increasing years.

Python

Python is a programming language that offers various use cases in the job for Business and Data Analysts. Some of its use cases are:

Statistical Analysis: Tools for modeling statistics, testing assumptions, and other statistical analyses are available through libraries like Statsmodels and SciPy in Python. This is essential to comprehending the fundamental connections and trends in the data.

Data Visualization: Data scientists may produce eye-catching and instructive charts and plots with the aid of Python visualization packages such as Matplotlib, Seaborn, Plotly, and others. Making decisions based on information and sharing ideas among stakeholders both require the use of visualizations.

Clear and Understandable Syntax: Python's syntax is clear and simple to understand, which makes it suitable for learners and encourages the reading of code.

Large Standard Library: Precludes the requirement for external libraries for many common features thanks to a large standard library that includes programs and modules for a variety of jobs.

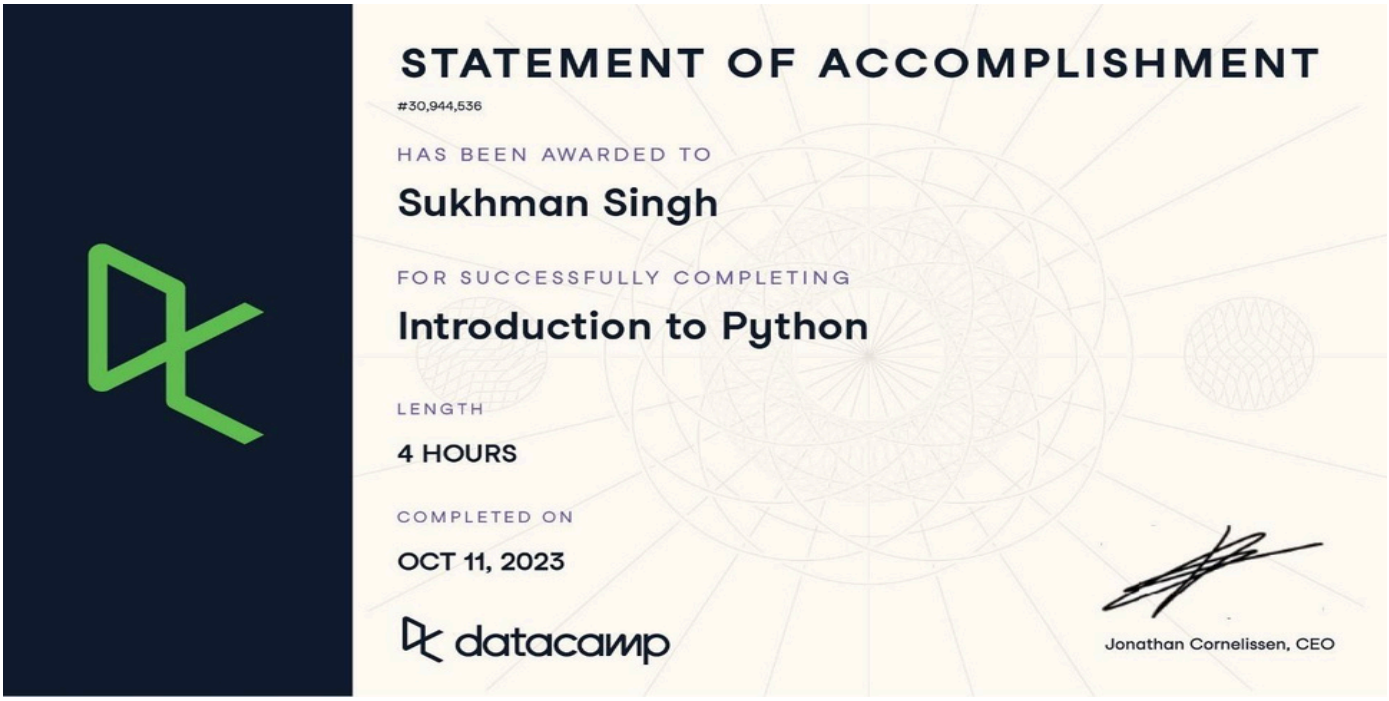
Community and Ecosystem: Python benefits from a sizable and vibrant development community that actively participates in its expansion. Python's capabilities are expanded by a multitude of third-party libraries and frameworks that are produced by this thriving ecosystem.

Cross-Platform Compatibility: Because Python is platform-independent, programming written in it can execute without change on a variety of operating systems.

Interpreted Language: Python is an interpreted language, which enables line-by-line execution of code, simplifying testing and debugging.

I first started learning Python on **YouTube**. I gained some knowledge of it before I took a course of Python on **Datacamp**. I learnt about different Python libraries and coding techniques on data statistics and **data visualization** with Python.

I am including **Python certificate from Datacamp** as a proof for it.



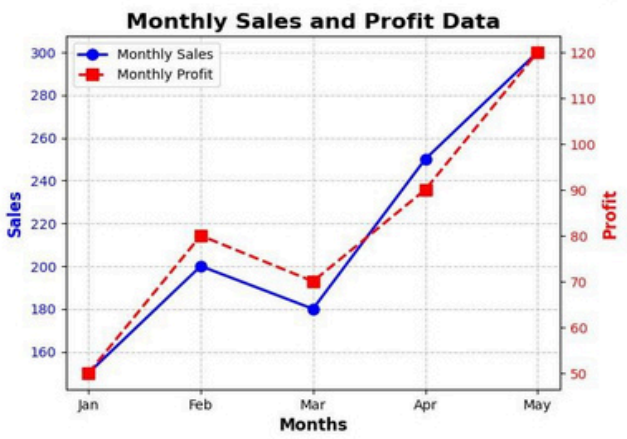
Now I will be discussing and commenting upon the tasks I learned within the datacamp and the tasks that I performed outside the course structure from new page.

- **Some of the Pyhon coding techniques that I learnt outside of course structure are:**

Task 1: Create a dual axis Line chart showcasing profit and sales of a company monthly.

Solution:

```
import matplotlib.pyplot as plt
import numpy as np

# Sample data
months = ['Jan', 'Feb', 'Mar', 'Apr', 'May']
sales_data = [150, 200, 180, 250, 300]
profit_data = [50, 80, 70, 90, 120]

# Create a figure and axis
fig, ax1 = plt.subplots()

# Plot the primary y-axis (sales data)
ax1.plot(months, sales_data, marker='o', linestyle='-', color='b', label='Monthly Sales', linewidth=2, markersize=8)

# Add labels and title for the primary y-axis
ax1.set_xlabel('Months', fontsize=12, fontweight='bold')
ax1.set_ylabel('Sales', fontsize=12, fontweight='bold', color='b')
ax1.set_title('Monthly Sales and Profit Data', fontsize=16, fontweight='bold')

# Customize tick labels for the primary y-axis
ax1.tick_params(axis='y', labelcolor='b')

# Add a secondary y-axis (profit data)
ax2 = ax1.twinx()
ax2.plot(months, profit_data, marker='s', linestyle='--', color='r', label='Monthly Profit', linewidth=2, markersize=8)

# Add labels for the secondary y-axis
ax2.set_ylabel('Profit', fontsize=12, fontweight='bold', color='r')

# Customize tick labels for the secondary y-axis
ax2.tick_params(axis='y', labelcolor='r')

# Add gridlines
ax1.grid(True, linestyle='--', alpha=0.7)

# Customize legend placement
lines, labels = ax1.get_legend_handles_labels()
lines2, labels2 = ax2.get_legend_handles_labels()
ax2.legend(lines + lines2, labels + labels2, loc='upper left')

# Save the plot as an image (optional)
plt.savefig('monthly_sales_profit_plot.png', bbox_inches='tight')

# Show the plot
plt.show()
```

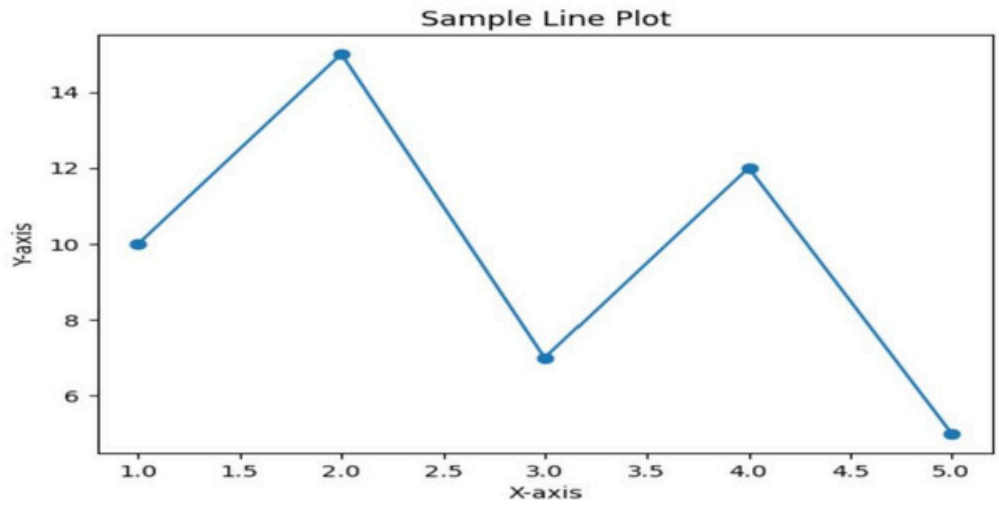


Reflective Analysis: I think Python gave a lot of **flexibility** deciding marker, color, linestyle, fontsize, fontweight etc. I was able to add **Axis labe**ls and **chart titles** to clearly define the motive of this chart. I really like the fact that I was able to add two axis in my chart. I was able to do formatting of the chart lables, gridlines and line style separately. For better uderstanding of the code I was able to add **#** and then write a short description of the code for the audience to understand every aspect of the code.

Task 2: ***Create a Line graph with mentioned specifications to showcase data visualization skills.***

Solution:

```
import matplotlib.pyplot as plt


x_values = [1, 2, 3, 4, 5]
y_values = [10, 15, 7, 12, 5]


plt.plot(x_values, y_values, marker='o', linestyle='-')


plt.title('Sample Line Plot')
plt.xlabel('X-axis')
plt.ylabel('Y-axis')

plt.show()
```



Reflective Analysis: I used **Google Collab** website to make this visualization. I used different **X** and **Y values** to plot them on the chart. I defined **marker** and **linestyle**. I was able to assign Title and Labe names of my choice too. I think Python has a very simple and readable syntax that makes it very easy to code and read the code. It allowed me to alter little details of the Line chart as per my needs. I used **Matplot library** of Python to do so. Overall I had a good experience making charts with Python.

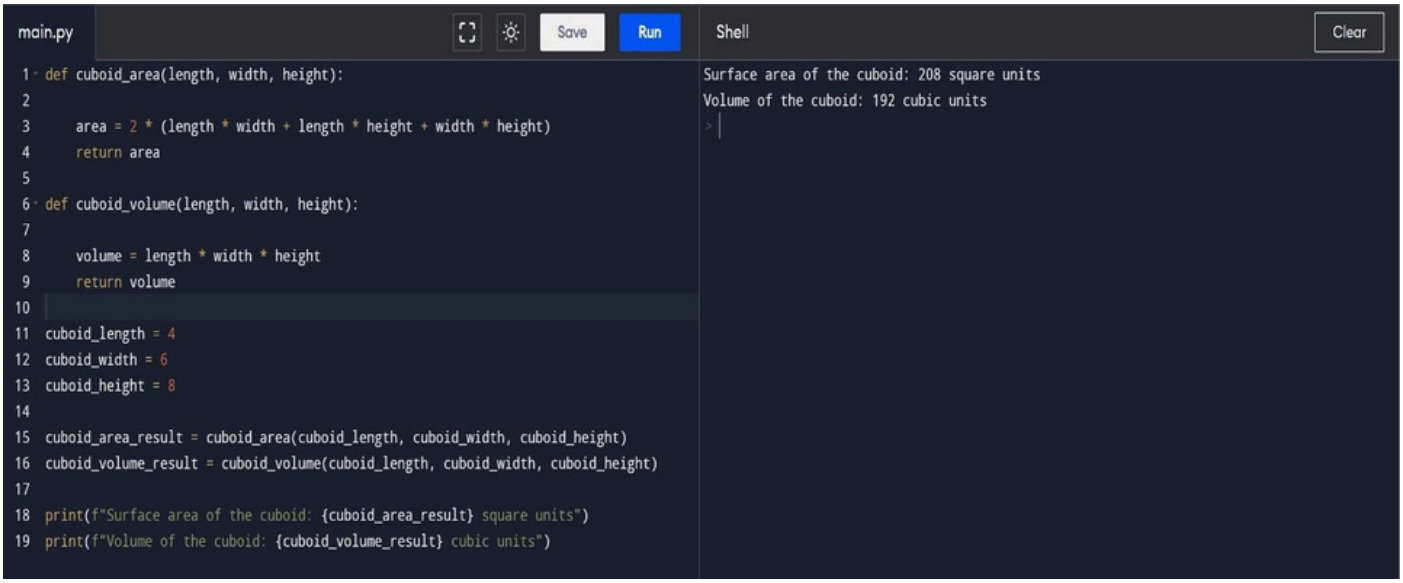
Task 3: ***Find lateral surface area and volume of cylinder and cuboid?***

Solution:

main.py | Save | Run | Shell | Clear

```
import math

def cylinder_area(radius, height):

    area = 2 * math.pi * radius * height
    return area

def cylinder_volume(radius, height):

    volume = math.pi * radius**2 * height
    return volume

cylinder_radius = 3
cylinder_height = 5

cylinder_area_result = cylinder_area(cylinder_radius, cylinder_height)
cylinder_volume_result = cylinder_volume(cylinder_radius, cylinder_height)

print(f"Area of the cylinder: {cylinder_area_result:.2f} square units")
print(f"Volume of the cylinder: {cylinder_volume_result:.2f} cubic units")
```

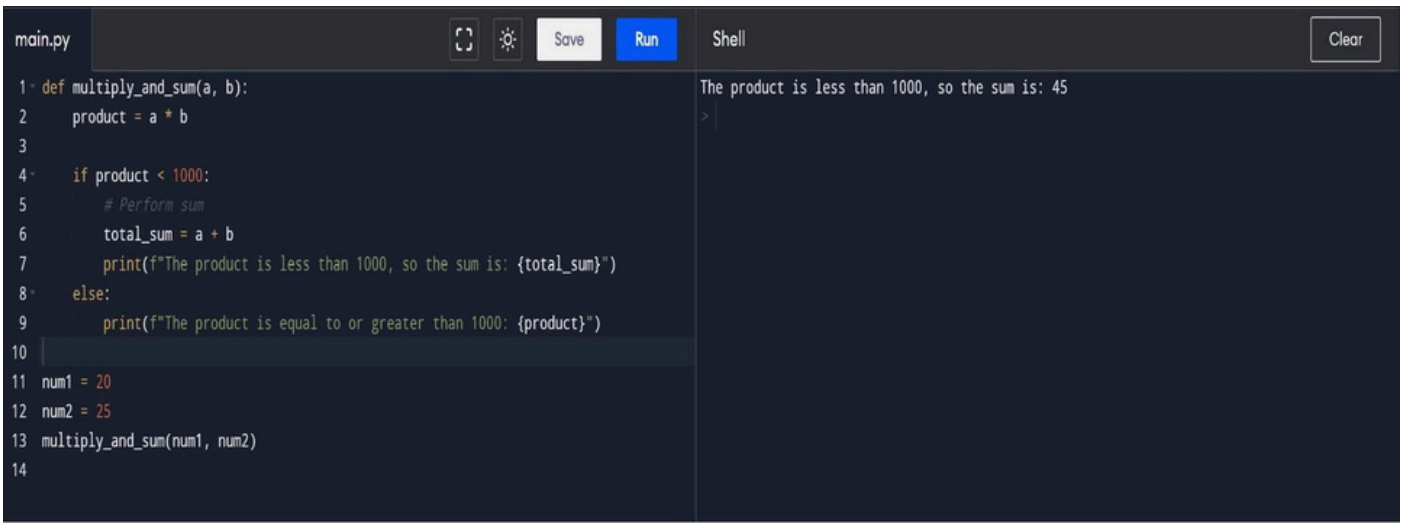
```
Area of the cylinder: 94.25 square units
Volume of the cylinder: 141.37 cubic units
>
```



Reflective Analysis: I learned here that Python is very efficient in doing complex **geometrical** calculations. With such automation complex calculations can be performed quickly and error- free. It was very simple to write a code and perform a geometrical calculation. When we have large databases to perform calculations, these type of codes become really handy and efficient to use.

Task 4: ***Multiplication or Sum of numbers, First numbers should be multiplied, if multiplication result is less than 1000, then numbers should be added.***

Solution:

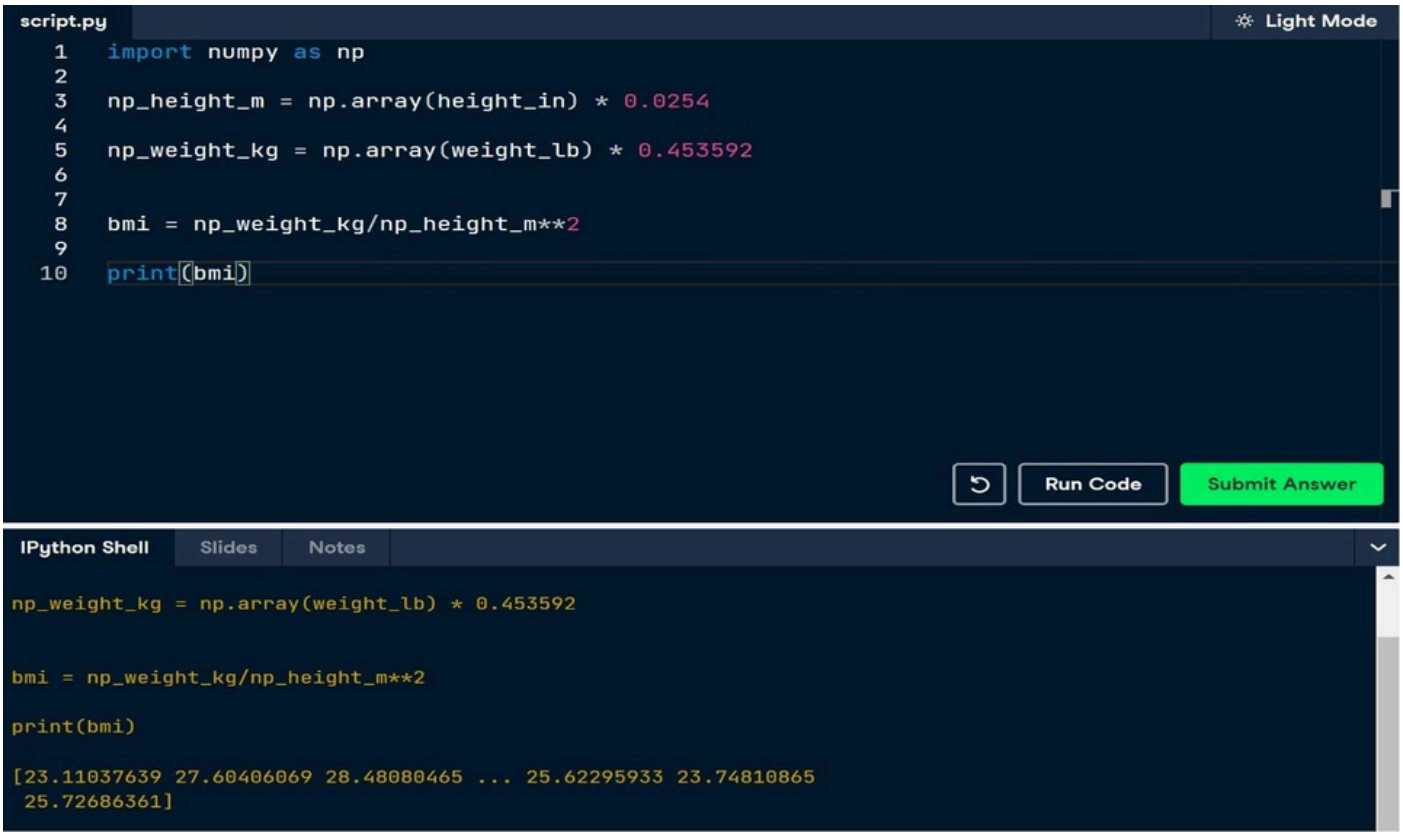


Reflective Analysis: Python made it very easy for me to perform this complex mathematical calculation. These types of calculations may look easy if perform with small numbers but imagine if the numbers are big and complex and i have large databases full of these complex numbers where doing calculations like this is a regular part of my work, it becomes easy for me to just code and get the results.

- **Tasks that I performed within the Datacamp are:**

Task 4: Calculate BMI of the players. Height in inches and weight in pounds for players is given. Array mentioning the heights and weights of players is already integerated in Datacamp.

Solution:

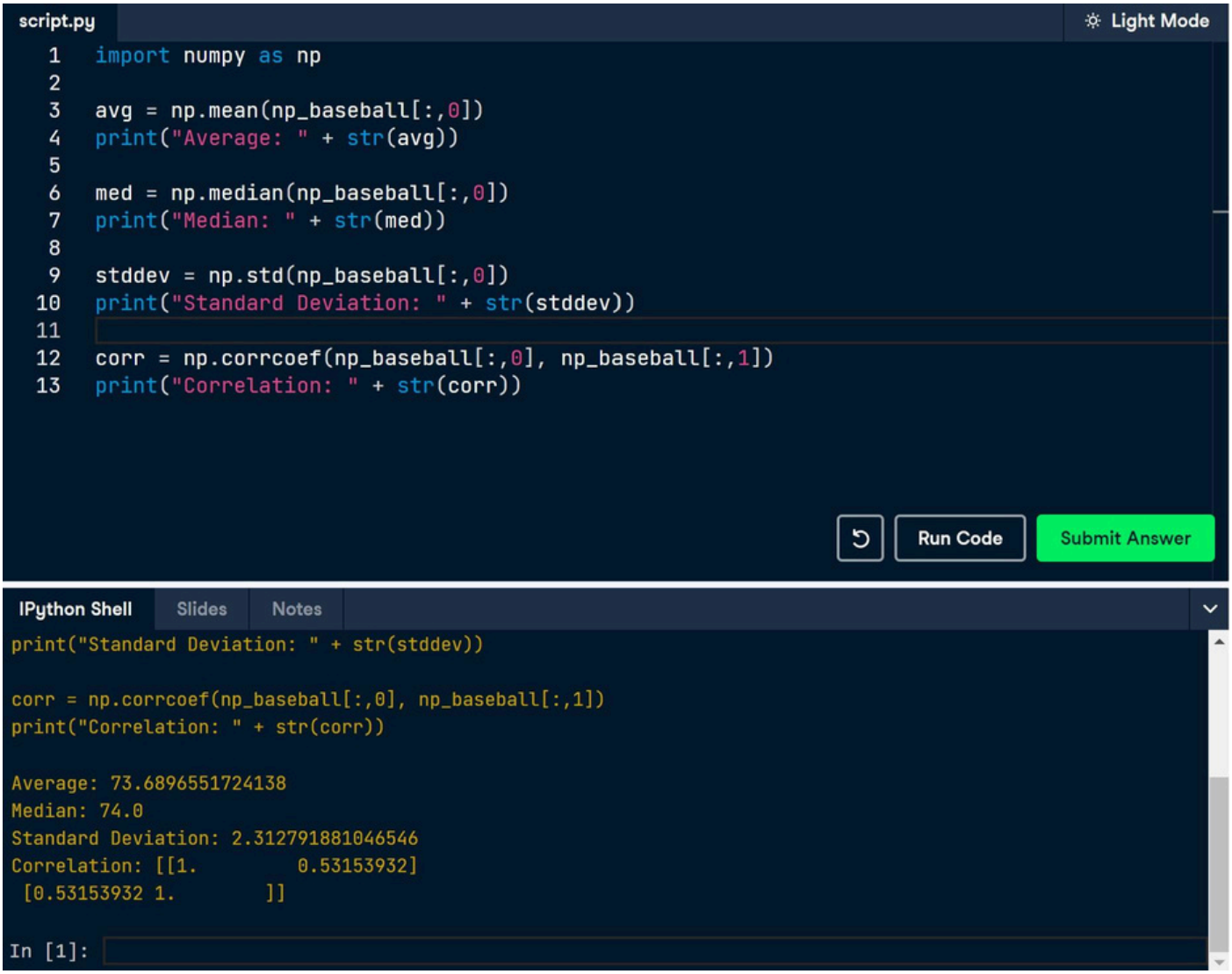


I created a **numpy** array from the **weight_lb** list with the correct units. I multiplied it by **0.453592** to go from pounds to kilograms. I stored the resulting numpy array as **np_weight_kg**. I again multiplied height_in with **0.0254** to convert heigth in inches into meters. I put the formula to calculate the BMI. Later I coded and printed out the results. Formula for BMI: **Weight(kg)/Height(m)square**

Reflective Analysis: I found it convenient to import numpy libray and thereafter perform my calculations. I do not have to write height and weight of players individually and thereafter perform my calculations. Rather I used **arrays** in one go to perform calculations of height and weight of every player. These calculations can be extended to any length of arrays and saves us the hectic procedue or re-doing these calculations again and again.

Task 5: Take out statistical details of the 'height' of players and comment upon them.

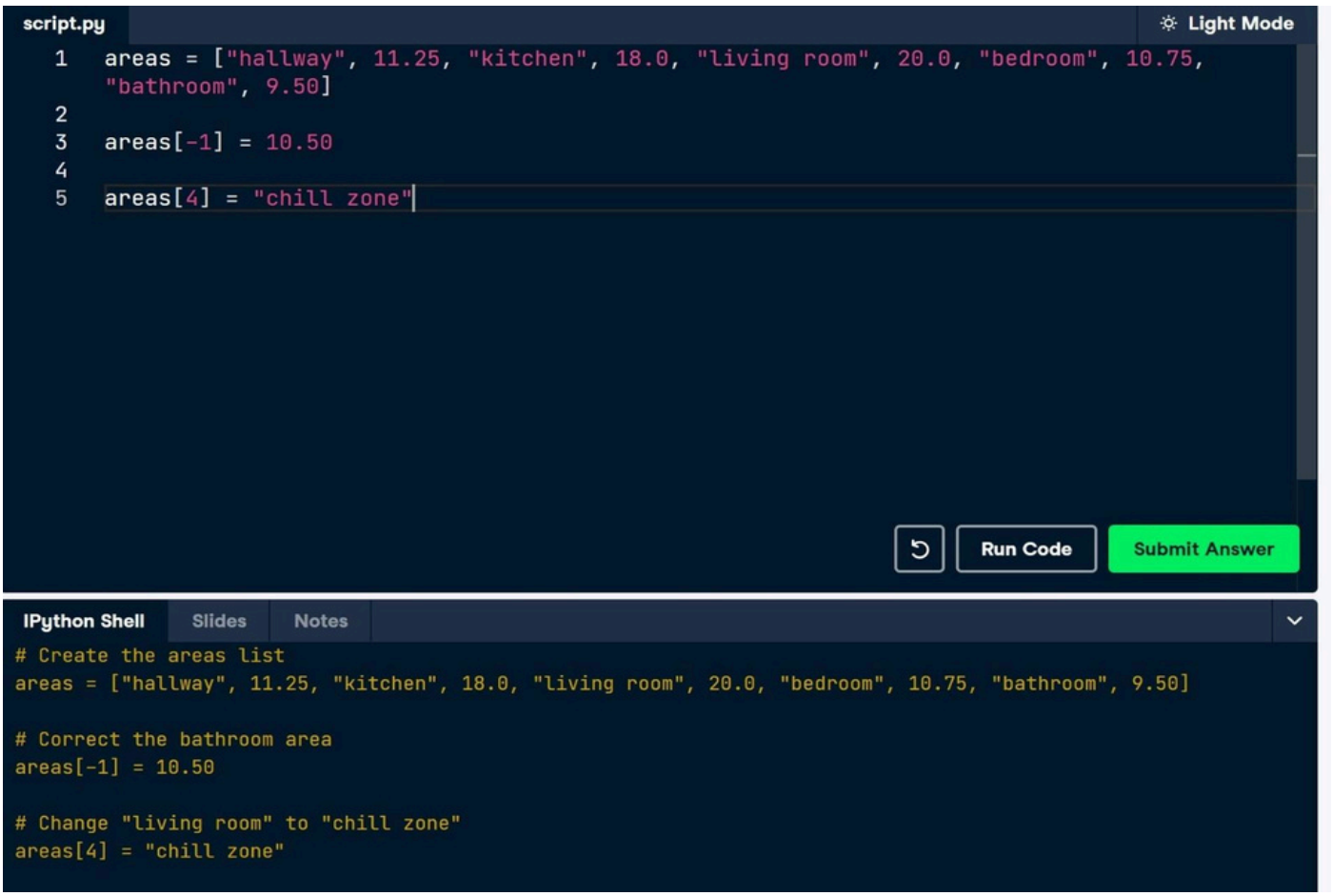
Solution:



Reflective Analysis: This task proves that Python is capable of performing Descriptive Statistics too. The are times when I need to perform descriptive statistics on a dataset. This code was easy to perform and helped me take out mean, median, standard deviation and correlation of the heights of the players. This piece of coding will be really helpful to me as many a times I need to perform descriptive statistics on large databases and this will add further feathers to my cap.

Task 6: Replace the list items. Update the area of the bathroom area to be 10.50 square meters instead of 9.50. Make the areas list more trendy! Change the "living room" to "chill zone".

Solution:



Reflective Learning: Creating list is one of the most important feature of Python coding especially for Data scientists and Business analysts. But many a times during this list creation, we need to change certain **list items** and replace them with something else. Python provides good editing options too. As seen above, every item of the list is allotted a certain index number. We can refer to particular list item through the list number and thereafter change it.

Conclusion

Incorporating Tableau and Python into a project portfolio improves the skills of business analysts and data scientists while providing an all-inclusive answer for a range of data analysis requirements. Tableau is a great tool for initial data exploration and dashboard building because of its capabilities in intuitive visualization, ad-hoc analysis, and easy to use collaboration. Python is an excellent choice for sophisticated analytics, machine learning, and data preprocessing due to its extensive library and programming skills.

The project portfolio gains from end-to-end analytics by integrating Tableau and Python, offering a comprehensive method for data-driven decision-making. Effective data pipeline growth is made possible by this partnership, from data cleaning and exploration to the use of unique algorithms and modeling. Tableau's visualization skills and Python's flexibility and extensibility combine to create a potent combination.