Topic	**Capstone class: Data Story-1**	
Class Description	**Students create a data story by reviewing the savings of people who were reminded to save money and those who were not.**	
Class	**C112**	
Class time	**45 mins**	
Goal	● Use Colab to write a data story based on the analysis and hypothesis testing of the given dataset.	
Resources Required	● Teacher Resources ○ Google Colaboratory (Colab) ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen ● Student Resources ○ Google Colaboratory (Colab) ○ Laptop with internet connectivity ○ Earphones with mic ○ Notebook and pen	
Class structure	**Warm Up** **Teacher-led Activity** **Student-led Activity** **Wrap up**	**5 mins** **15 min** **15 min** **5 min**

CONTEXT

- **Review the concepts learned in the earlier classes**

Class Steps	**Teacher Action**	**Student Action**
Step 1: **Warm Up** **(5 mins)**	Hi <Student Name> Welcome to the Capstone Class.	**ESR:** -We learned to find mean, median and mode. -We learned about the

	What all did we learn in our previous classes?	correlation. -We learned about the sampling distribution. - We learned about the z test.
	Yes...these are very useful concepts that we learned while doing some data visualization. Today we'll take data of the people who were sent reminders to save money and people who weren't sent reminders to save the money. We'll conduct some tests on this data and write our data story. **Data story** or data **storytelling** is the practice of building a narrative around a set of **data** and its accompanying visualizations to help convey the meaning of that **data** in a powerful and compelling fashion. Sounds exciting?	**ESR:** varied
	Let's get started then.	-

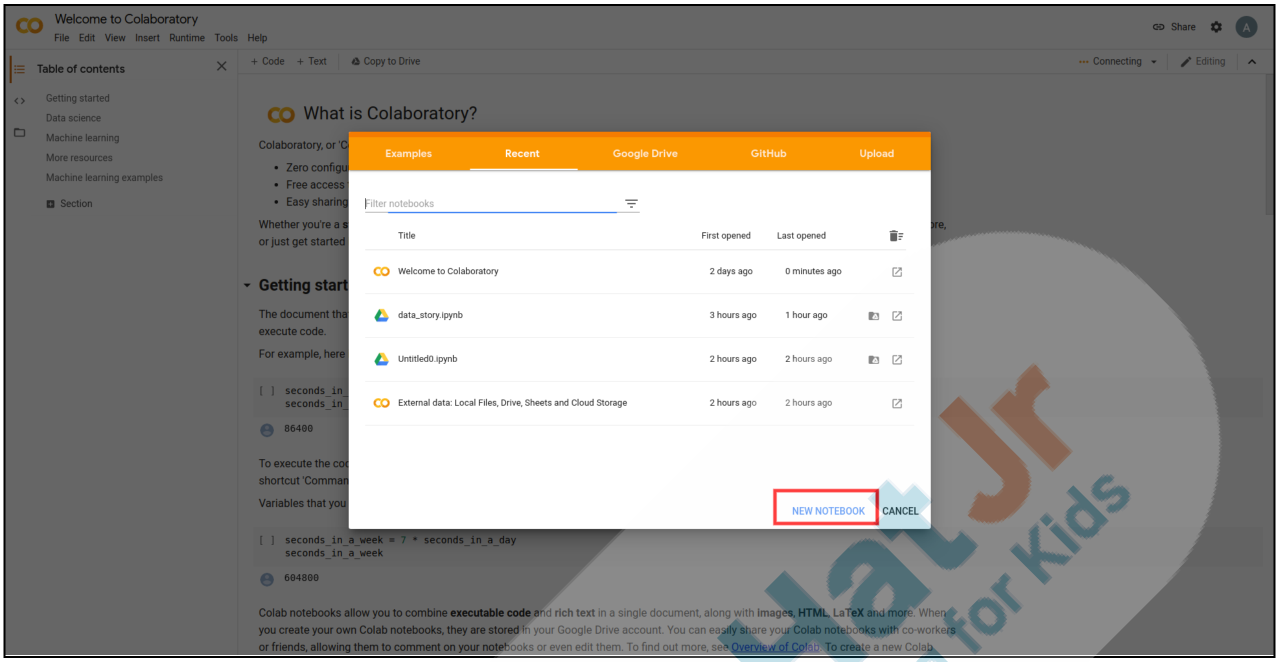
Teacher Initiates Screen Share

CHALLENGE

- **Learn the usage of colab**
- **Perform hypothesis testing on the given data set**

Step 2: **Teacher-led Activity** **(15 min)**	To write the data story we'll use the "Colaboratory" or "Colab " for short. Colab allows us to write and execute	-

	python on the browser. Let's watch a short introduction video about Colab. *<Teacher opens the link from **Teacher activity 1** and watch the video>* Now let's see how to use a Colab. *Teacher opens the link from **Teacher activity 2.***	
	In Colab every project is called a **notebook**. When we open a Colab we see a pop up where we can select our previous notebook to continue our work or create a new notebook to work on a new project. We'll create a new notebook. Here we can write python code as well as text.	

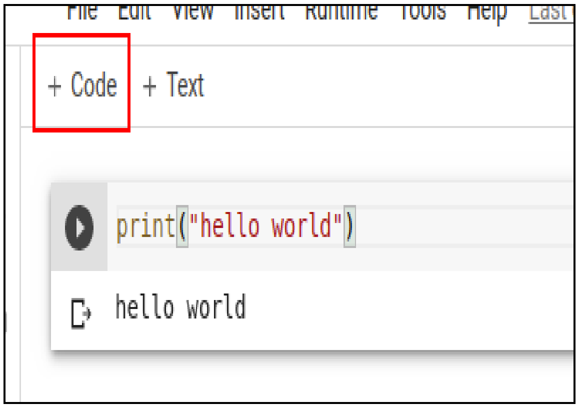
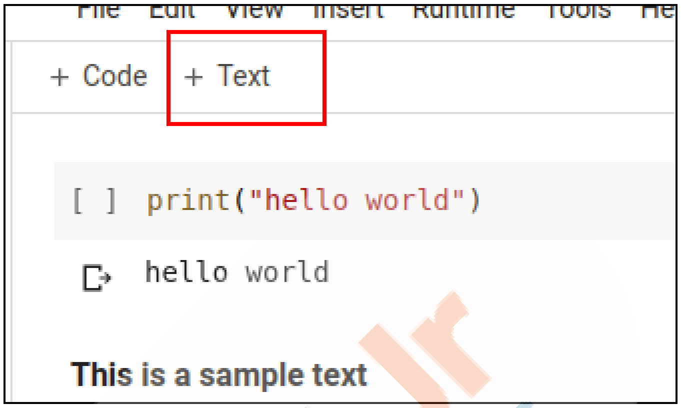


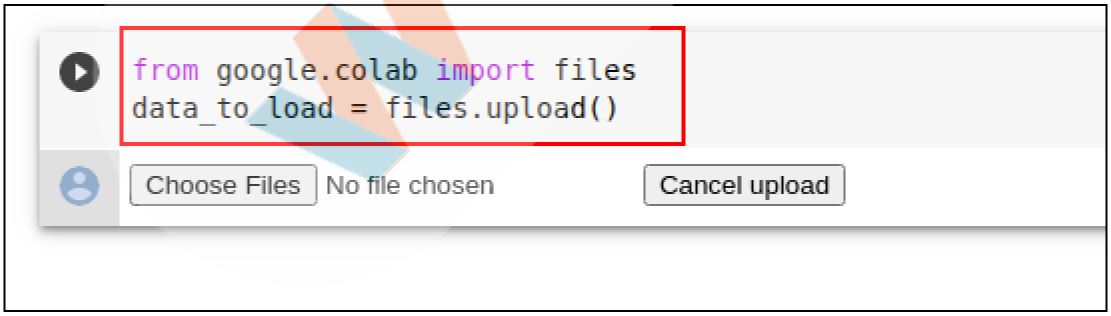
Title	First opened	Last opened
Welcome to Colaboratory	2 days ago	0 minutes ago
data_story.ipynb	3 hours ago	1 hour ago
Untitled0.ipynb	2 hours ago	2 hours ago
External data: Local Files, Drive, Sheets and Cloud Storage	2 hours ago	2 hours ago

```
[ ]  seconds_in_a_week = 7 * seconds_in_a_day
     seconds_in_a_week

     604800
```

Colab notebooks allow you to combine **executable code** and **rich text** in a single document, along with **images**, **HTML**, **LaTeX** and more. When you create your own Colab notebooks, they are stored in your Google Drive account. You can easily share your Colab notebooks with co-workers or friends, allowing them to comment on your notebooks or even edit them. To find out more, see Overview of Colab. To create a new Colab

	Can you guess how we can write code and text? Yes, To write code we click on the code button. A code cell opens up where you can write your code and press the run button to execute your code. *<The teacher clicks on the code button and types print("hello world") in the code cell and clicks on the run button>* Same way we can add the text in the notebook. Text can be used for general purpose like: -Adding a heading. -Adding an explanation on what your code block is doing.	**ESR:** We can see the + Code and + Text button in the top corner. By clicking those buttons we can add code or text to the notebook.

	-Adding instructions.	
 File Edit View Insert Runtime Tools Help Last + Code + Text print("hello world") hello world	 [ ] print("hello world") hello world This is a sample text	
	Uploading and importing files in Colab is also very easy. To upload the files in Colab we just have to write a small piece of code. *<Teacher writes the following code in code cell>* Code:- **from google.colab import files** **data_to_load = files.upload()** a choose file button will appear. by clicking on the button we can upload the files from our local system.	-

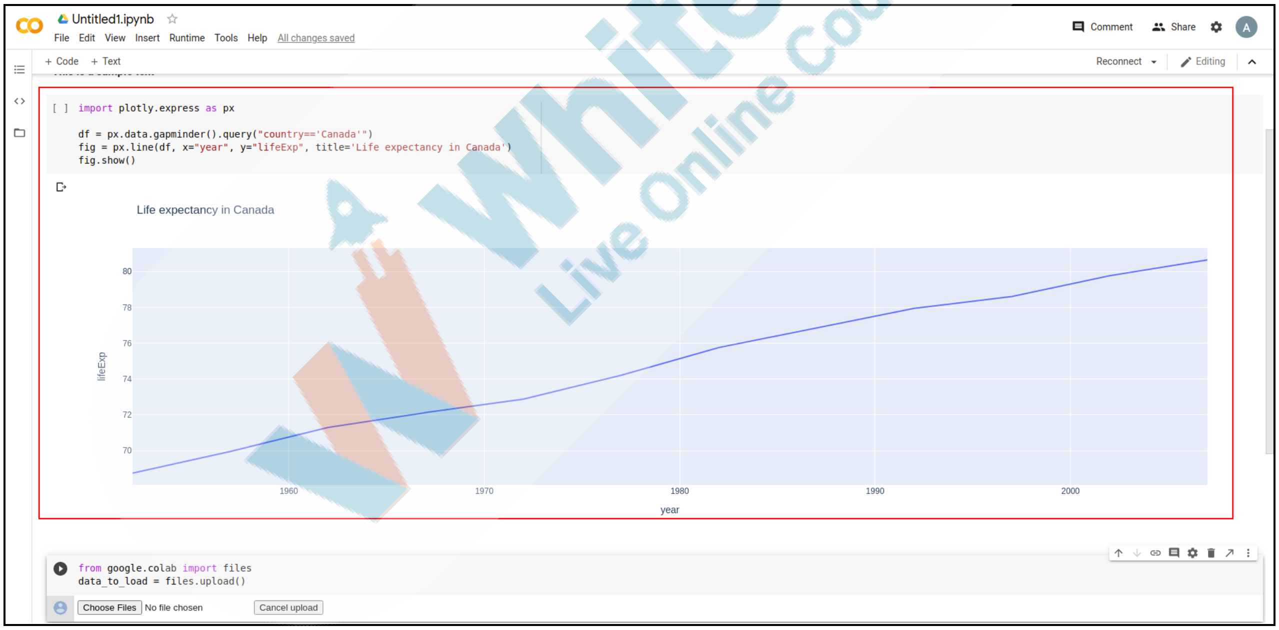

from google.colab import files
data_to_load = files.upload()		

	We have been plotting graphs to visualize data in past few classes. We can even plot the graphs in Colab using our plotly libraries.	

<Teacher writes the following sample code in the code cell and executes the code>

Code:-

```
import plotly.express as px

df = px.data.gapminder().query("country=='Canada'")
fig = px.line(df, x="year", y="lifeExp", title='Life expectancy in Canada')
fig.show()
```

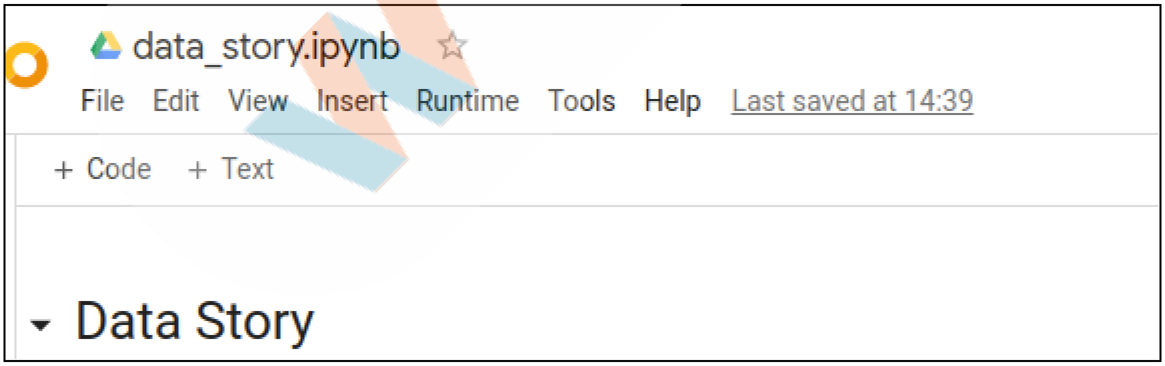


```
import plotly.express as px

df = px.data.gapminder().query("country=='Canada'")
fig = px.line(df, x="year", y="lifeExp", title='Life expectancy in Canada')
fig.show()
```

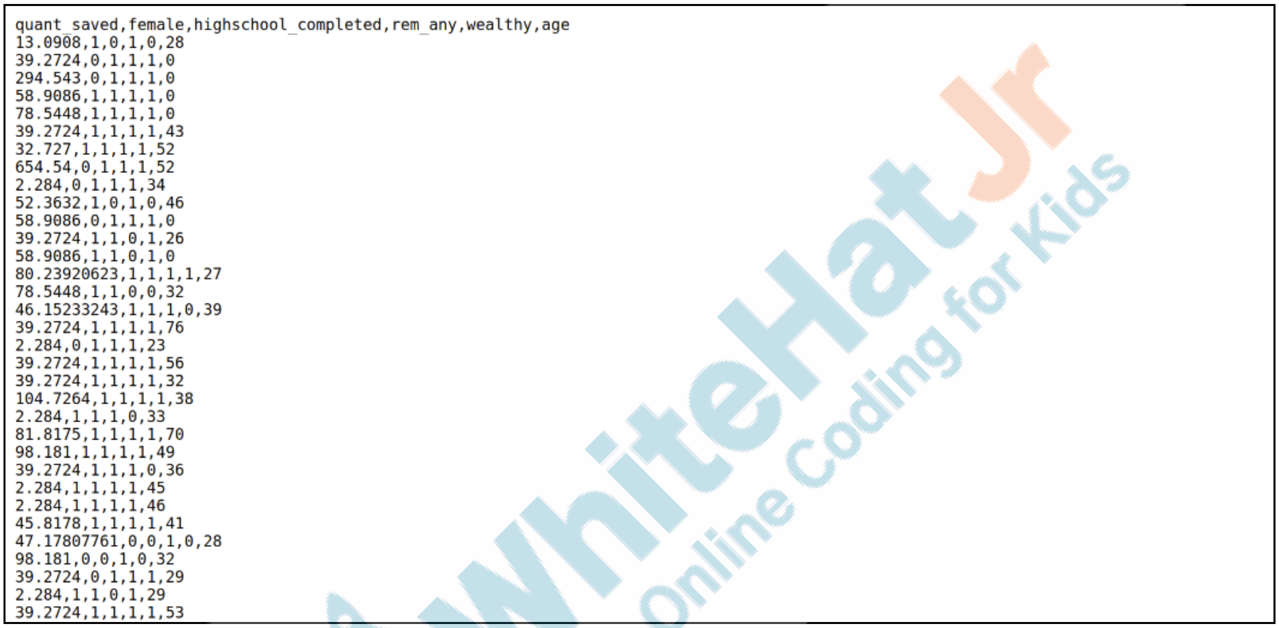
```
from google.colab import files
data_to_load = files.upload()
```

We have a set of data of people who were given reminders to save money and some people who weren't given reminders to save the money.

Let's write a data story of these data

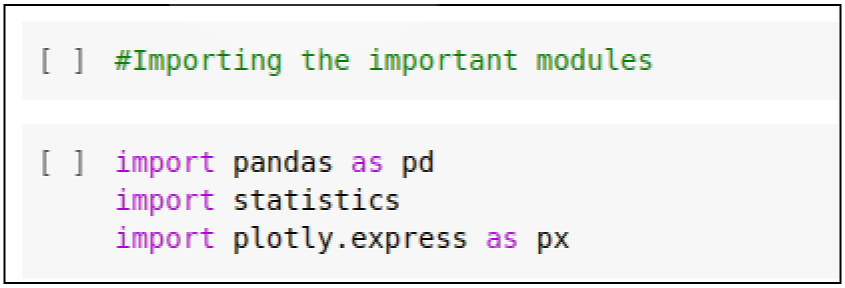
	We are writing the data story so that anyone who reads it can understand what meaning the data is trying to convey. Are you ready for this?	**ESR:** Yes.
Teacher Stops Screen Share		
	Now it's your turn. Please share your screen with me.	
● **Ask Student to press ESC key to come back to panel** ● **Guide Student to start Screen Share** ● **Teacher gets into Fullscreen**		
ACTIVITY ● **Perform multiple tests to do the analysis and write the data story on Colab**		
Step 3: **Student-Led Activity** **(15 min)**	*Teacher helps the student to open a new Colab notebook and rename it as data_story.*	*Student opens the Colab notebook from student activity 1 and renames it as data_story.*
 data_story.ipynb File Edit View Insert Runtime Tools Help Last saved at 14:39 + Code + Text Data Story		

	Now let's download the data that we will be using. *Teacher helps the student to download the data.*	*Student downloads the data from Student Activity 2.*



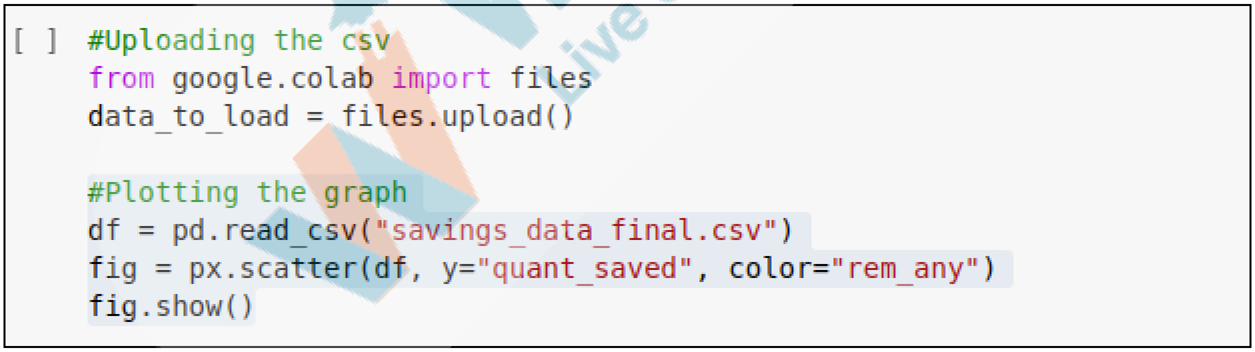
```
quant_saved,female,highschool_completed,rem_any,wealthy,age
13.0908,1,0,1,0,28
39.2724,0,1,1,1,0
294.543,0,1,1,1,0
58.9086,1,1,1,1,0
78.5448,1,1,1,1,0
39.2724,1,1,1,1,43
32.727,1,1,1,1,52
654.54,0,1,1,1,52
2.284,0,1,1,1,34
52.3632,1,0,1,0,46
58.9086,0,1,1,1,0
39.2724,1,1,0,1,26
58.9086,1,1,0,1,0
80.23920623,1,1,1,1,27
78.5448,1,1,0,0,32
46.15233243,1,1,1,0,39
39.2724,1,1,1,1,76
2.284,0,1,1,1,23
39.2724,1,1,1,1,56
39.2724,1,1,1,1,32
104.7264,1,1,1,1,38
2.284,1,1,1,0,33
81.8175,1,1,1,1,70
98.181,1,1,1,1,49
39.2724,1,1,1,0,36
2.284,1,1,1,1,45
2.284,1,1,1,1,46
45.8178,1,1,1,1,41
47.17807761,0,0,1,0,28
98.181,0,0,1,0,32
39.2724,0,1,1,1,29
2.284,1,1,0,1,29
39.2724,1,1,1,1,53
```

	First of all we need to import all the libraries like plotly, pandas and statistics to our code. *Teacher helps the student to import the libraries.*	*Student imports the libraries to the code.*

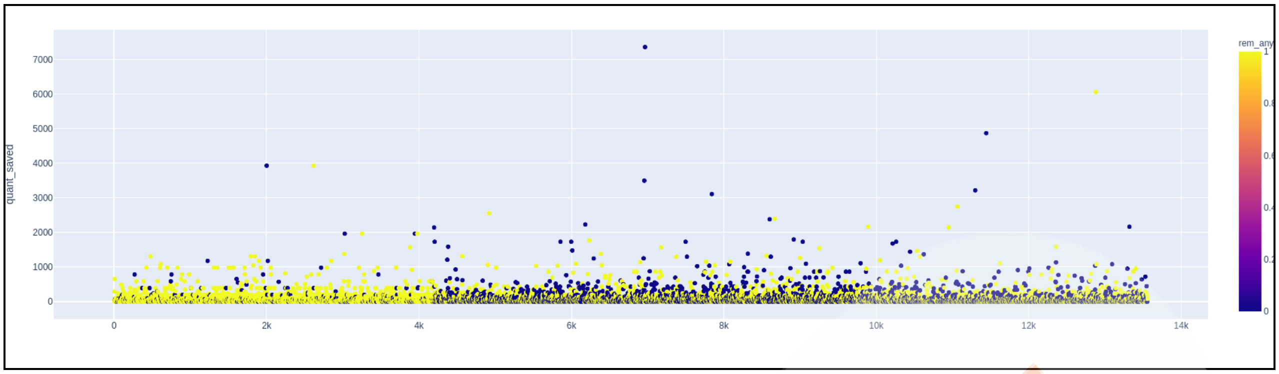


```
[ ]  #Importing the important modules

[ ]  import pandas as pd
     import statistics
     import plotly.express as px
```

	Now let's upload the data to our notebook and plot it to see if we get something from it. To upload the data to the Colab we write the following code:- **#Uploading the csv** **from google.colab import files** **data_to_load = files.upload()** Then click on the choose file button and select the data file. To plot the graph:- **#Plotting the graph** **df = pd.read_csv("savings_data_final.csv")** **fig = px.scatter(df, y="quant_saved", color="rem_any")** **fig.show()**	*Student uploads the file to the colab and then plots in a scatter plot.*



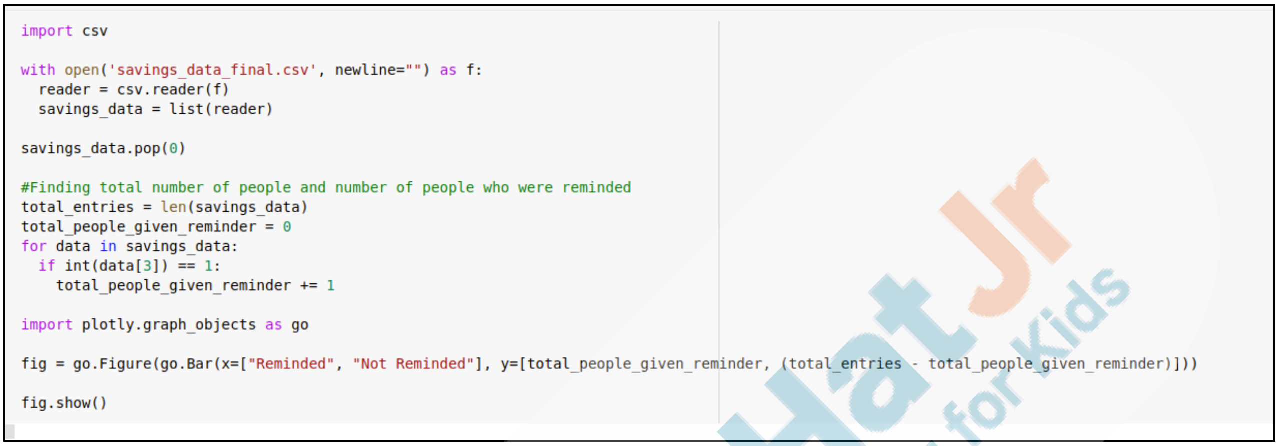
```
#Uploading the csv
from google.colab import files
data_to_load = files.upload()

#Plotting the graph
df = pd.read_csv("savings_data_final.csv")
fig = px.scatter(df, y="quant_saved", color="rem_any")
fig.show()
```



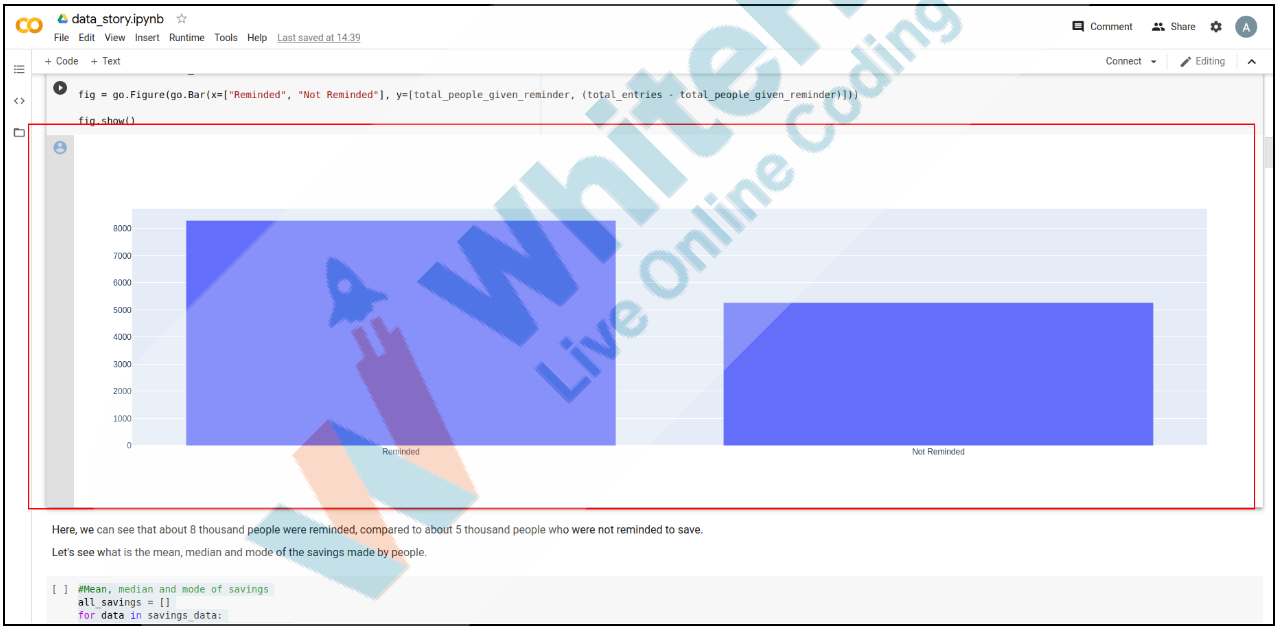
	If we look at this data, we can see that the yellow dots are the ones who were given a reminder to save (Since 1 stands for True) while the blue dots are the ones who were not given a reminder to save. What else can we see from the data? Let's try to plot and see how many people were given a reminder v/s the people who were not given a reminder.	**ESR:** We can see that most of the outliers are the Blue dots, who have saved more than others.
	Before that we need to find out the number of people who were given reminders. By looking at the data we know that the number of people who were given reminders have value 1 and those who were not given reminders have value as 0. -So first we'll read the data from the csv file using csv.reader and save the data in the savings_data variable. -Then using a for loop on the	*Student codes to read the data from the csv and save it in a saving_data variable.* *-Then creates a total_entries variable and sets the length of saving_data to it.* *-Then creates a total_people_given_reminder variable and sets its value to 0.* *-Using a for loop on the*

	savings_data, everytime when a user has value 1 we increase the count of the total_people_given_reminder by 1. -To find the number of people who were not given reminders we will subtract the total number of entries - total number of people given reminders. *Teacher helps the student with the code.* **import csv** **with open('savings_data_final.csv', newline="") as f:** **reader = csv.reader(f)** **savings_data = list(reader)** **savings_data.pop(0)** **#Finding total number of people and number of people who were reminded** **total_entries = len(savings_data)** **total_people_given_reminder = 0** **for data in savings_data:** **if int(data[3]) == 1:** **total_people_given_reminder += 1** **import plotly.graph_objects as go** **fig = go.Figure(go.Bar(x=["Reminded", "Not Reminded"], y=[total_people_given_reminder,**	*saving_data, every time we find the value of an entry to be 1 we increase the counter by 1.* *-Then using plotly plots the graph.*

	(total_entries - total_people_given_reminder)])) fig.show()	



```
import csv

with open('savings_data_final.csv', newline="") as f:
  reader = csv.reader(f)
  savings_data = list(reader)

savings_data.pop(0)

#Finding total number of people and number of people who were reminded
total_entries = len(savings_data)
total_people_given_reminder = 0
for data in savings_data:
  if int(data[3]) == 1:
    total_people_given_reminder += 1

import plotly.graph_objects as go

fig = go.Figure(go.Bar(x=["Reminded", "Not Reminded"], y=[total_people_given_reminder, (total_entries - total_people_given_reminder)]))

fig.show()
```



```
fig = go.Figure(go.Bar(x=["Reminded", "Not Reminded"], y=[total_people_given_reminder, (total_entries - total_people_given_reminder)]))

fig.show()
```

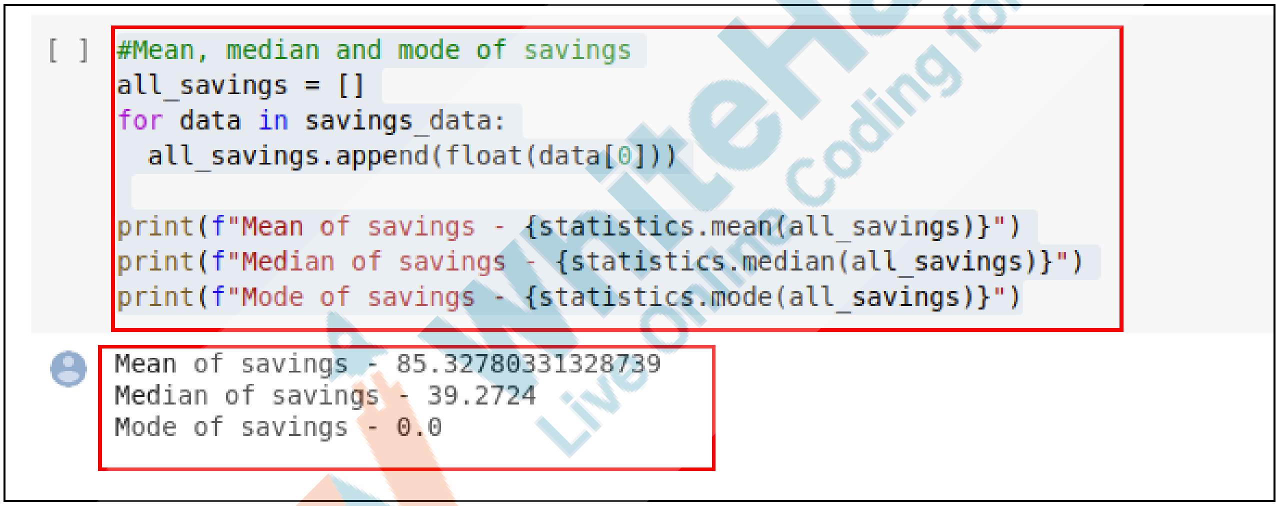
Here, we can see that about 8 thousand people were reminded, compared to about 5 thousand people who were not reminded to save.

Let's see what is the mean, median and mode of the savings made by people.

```
[ ] #Mean, median and mode of savings
    all_savings = []
    for data in savings_data:
```

	What can we see from the plot? Yes! Now let's find the mean median and mode of the savings_data. Can you tell me how to find that?	**ESR:** We can see that the number of people who were reminded is more than people not reminded.

	Code:- **#Mean, median and mode of savings** **all_savings = []** **for data in savings_data:** ** all_savings.append(float(data[0]))** **print(f"Mean of savings - {statistics.mean(all_savings)}")** **print(f"Median of savings - {statistics.median(all_savings)}")** **print(f"Mode of savings - {statistics.mode(all_savings)}")**	**ESR:** We use the mean, median and mode functions of the statistics library. *Student finds the mean median and mode of the saved_data.*



```
#Mean, median and mode of savings
all_savings = []
for data in savings_data:
  all_savings.append(float(data[0]))

print(f"Mean of savings - {statistics.mean(all_savings)}")
print(f"Median of savings - {statistics.median(all_savings)}")
print(f"Mode of savings - {statistics.mode(all_savings)}")
```

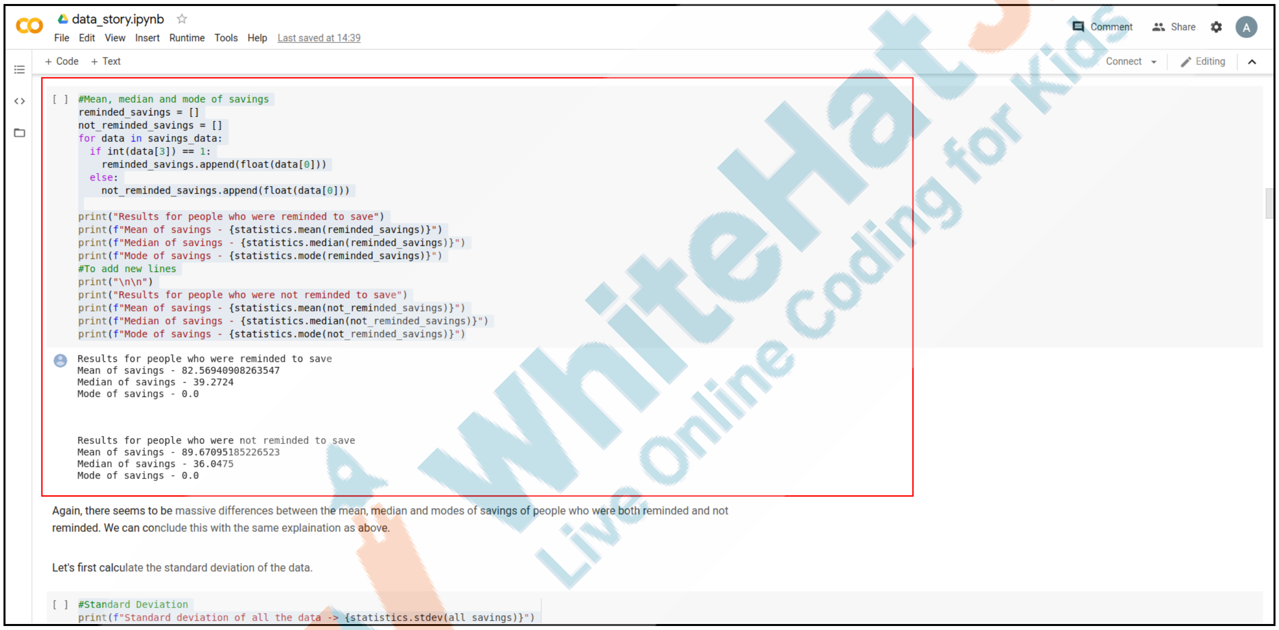
```
Mean of savings - 85.32780331328739
Median of savings - 39.2724
Mode of savings - 0.0
```

	Now these are some very interesting results! Can you guess why the mean, median and the mode are not the same and worlds apart? If we go back and look at the scatterplot we plotted before, we can see that the majority of the savings data lies between 0 to 100.	**ESR:** varied

	Now, since we have a few outliers, which are the blue dots that are away from the rest of the crowd, our mean has significantly increased from the median, since it is the sum of all values by total entries. Since the outliers lie far away from the crowd, the difference is huge. Similarly, for mode, in our data, there are a lot of people who didn't save at all. Thus, the mode of the data is 0. Mode is the value with maximum occurrences. Let's see if we have a similar massive difference between the mean, median and mode of people who got reminded and people who didn't receive reminders.	
	Before that we need to find the values of people who got reminded and people who didn't get reminded. As we know in our data we have 0 as a value to people who were not reminded to save and 1 for people who were reminded to save. We'll create two variables not_reminded_savings and reminded_savings and set empty lists as their values. To get that data we'll loop on the savings data and in the 4th column in	*Student codes to get the list of reminded_savings and not_reminded_savings.* *And then finds the mean, median and mode for both the data.*

	savings data if the value is 0 we'll append it in the not_reminded_savings list and if the value is 1 we'll append it in the reminded_savings list. *Teacher helps the student with the code* Code:- **#Mean, median and mode of savings** **reminded_savings = []** **not_reminded_savings = []** **for data in savings_data:** ** if int(data[3]) == 1:** ** reminded_savings.append(float(data[0]))** ** else:** ** not_reminded_savings.append(float(data[0]))** **print("Results for people who were reminded to save")** **print(f"Mean of savings - {statistics.mean(reminded_savings)}")** **print(f"Median of savings - {statistics.median(reminded_savings)}")** **print(f"Mode of savings - {statistics.mode(reminded_savings)}")** **#To add new lines** **print("\n\n")** **print("Results for people who were**	

	not reminded to save")
	print(f"Mean of savings - {statistics.mean(not_reminded_savings)}")
	print(f"Median of savings - {statistics.median(not_reminded_savings)}")
	print(f"Mode of savings - {statistics.mode(not_reminded_savings)}")	


data_story.ipynb
File Edit View Insert Runtime Tools Help Last saved at 14:39
+ Code + Text
Comment Share
Connect Editing

```
[ ] #Mean, median and mode of savings
    reminded_savings = []
    not_reminded_savings = []
    for data in savings_data:
      if int(data[3]) == 1:
        reminded_savings.append(float(data[0]))
      else:
        not_reminded_savings.append(float(data[0]))

    print("Results for people who were reminded to save")
    print(f"Mean of savings - {statistics.mean(reminded_savings)}")
    print(f"Median of savings - {statistics.median(reminded_savings)}")
    print(f"Mode of savings - {statistics.mode(reminded_savings)}")
    #To add new lines
    print("\n\n")
    print("Results for people who were not reminded to save")
    print(f"Mean of savings - {statistics.mean(not_reminded_savings)}")
    print(f"Median of savings - {statistics.median(not_reminded_savings)}")
    print(f"Mode of savings - {statistics.mode(not_reminded_savings)}")
```

```
Results for people who were reminded to save
Mean of savings - 82.56940908263547
Median of savings - 39.2724
Mode of savings - 0.0


Results for people who were not reminded to save
Mean of savings - 89.67095185226523
Median of savings - 36.0475
Mode of savings - 0.0
```

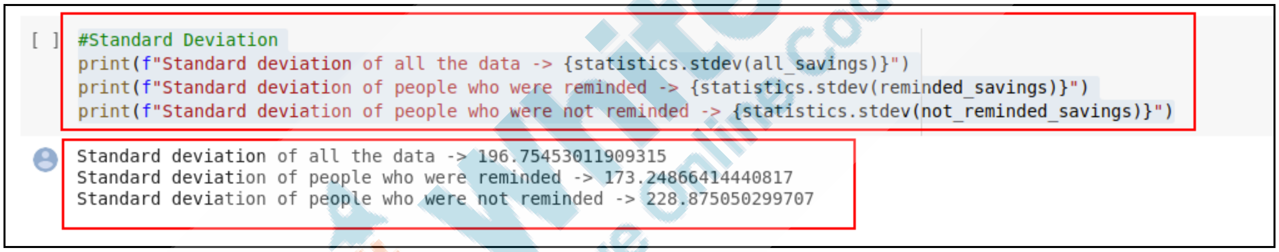
Again, there seems to be massive differences between the mean, median and modes of savings of people who were both reminded and not reminded. We can conclude this with the same explaination as above.

Let's first calculate the standard deviation of the data.

```
[ ] #Standard Deviation
    print(f"Standard deviation of all the data -> {statistics.stdev(all savings)}")
```

	Again, there seems to be massive differences between the mean, median and modes of savings of people who were reminded and people who were not reminded. We can conclude this with the same explanation as earlier. Let's first calculate the standard deviation of the data.	*Student Codes to calculate the mean median and mode of the* *-all_savings* *-reminded_Savings* *-not_reminded_savings* *and print those values*

	Teacher helps student with the code. Code:- **#Standard Deviation** **print(f"Standard deviation of all the data -> {statistics.stdev(all_savings)}")** **print(f"Standard deviation of people who were reminded -> {statistics.stdev(reminded_savings)}")** **print(f"Standard deviation of people who were not reminded -> {statistics.stdev(not_reminded_savings)}")**	



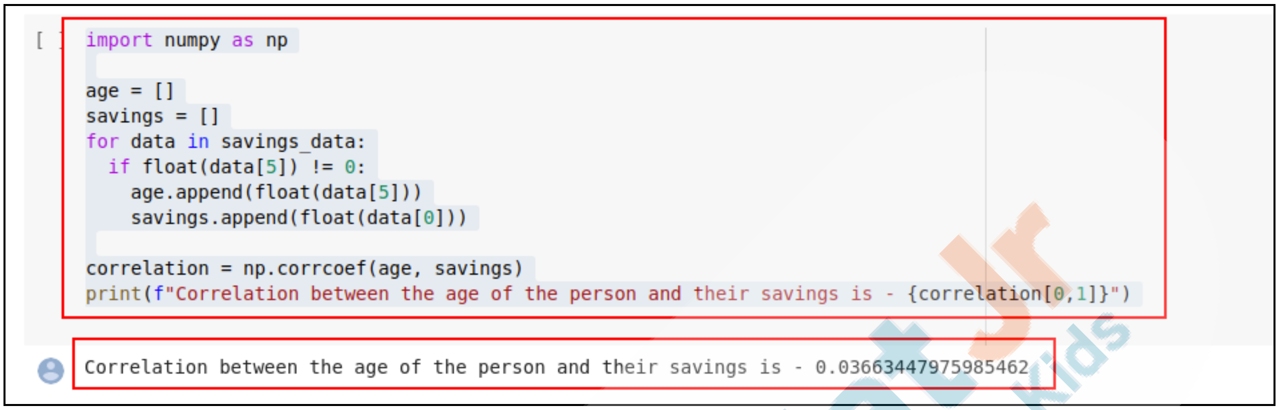
```
#Standard Deviation
print(f"Standard deviation of all the data -> {statistics.stdev(all_savings)}")
print(f"Standard deviation of people who were reminded -> {statistics.stdev(reminded_savings)}")
print(f"Standard deviation of people who were not reminded -> {statistics.stdev(not_reminded_savings)}")
```

```
Standard deviation of all the data -> 196.75453011909315
Standard deviation of people who were reminded -> 173.24866414440817
Standard deviation of people who were not reminded -> 228.875050299707
```

	Here, we can see that the standard deviation varies a lot in all three types of data. What can we tell from the standard deviation of the three types of data? Very true. Now the question is, does this data have a correlation? Let's see if the savings are correlated to the age of people.	**ESR:** It is higher for the people who were not reminded v/s the people who were reminded. **ESR:** Looking at the data uptil now, we can assume that reminding people to save did not have a significant effect.

	Note - The columns that have age as 0 will not be considered, since their age is missing. No one can be saving at the age of 0. To find the correlation between the savings and age of people we need to get these data first. In our data set the first column is the savings and the last column is the age. And we don't want a savings where the age is 0. So we'll write an if condition which will check if the value in the 5th column is 0 or not. If it's 0 we skip it and if it's not 0 we append those values in the age and savings list respectively. Using the corrcoef function of numpy we calculate the correlation between them. *Teacher helps the student with the code.* Code:- **import numpy as np** **age = []** **savings = []** **for data in savings_data:** **if float(data[5]) != 0:** **age.append(float(data[5]))** **savings.append(float(data[0]))** **correlation = np.corrcoef(age, savings)** **print(f"Correlation between the age**	From standard deviations, we can see that the people who were not reminded have much more scattered data than people who were reminded. *Student codes to find the data and of age and savings and find it's correlation.*

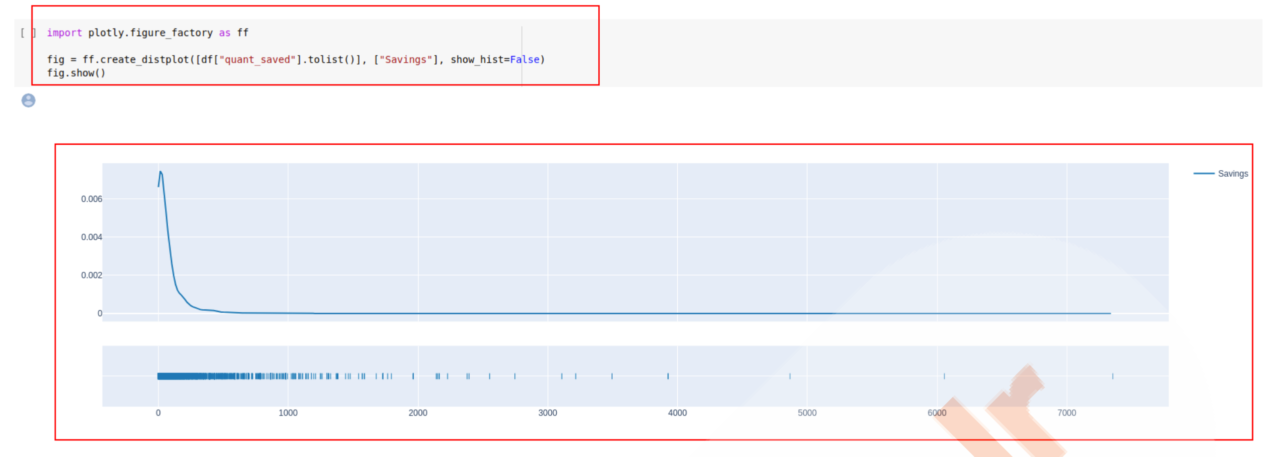
	of the person and their savings is - {correlation[0,1]}")	



```
import numpy as np

age = []
savings = []
for data in savings_data:
  if float(data[5]) != 0:
    age.append(float(data[5]))
    savings.append(float(data[0]))

correlation = np.corrcoef(age, savings)
print(f"Correlation between the age of the person and their savings is - {correlation[0,1]}")
```

```
Correlation between the age of the person and their savings is - 0.03663447975985462
```

	Here, we receive the correlation between the age and the savings to be 0.03, which means that the given data is not correlated. Let's see if this given data for savings follows a bell curve normal distribution. *Teacher helps the student with the code.* Code:- **import plotly.figure_factory as ff** **fig = ff.create_distplot([df["quant_saved"].tolist()], ["Savings"], show_hist=False)** **fig.show()**	*Student codes to plot the savings data on a distplot.*



```
import plotly.figure_factory as ff

fig = ff.create_distplot([df["quant_saved"].tolist()], ["Savings"], show_hist=False)
fig.show()
```

	What observations can we make by looking at the plot? Very good, we can also see that our chart is skewed to the left side of the chart. This means that the majority of the data, instead of lying on the center, lies in the left side of the chart. To deal with this, we can remove the outliers. There is a method known as the IQR (Interquartile Range) method. We will learn more about it in our next class.	**ESR:** -It definitely does not follow a normal distribution. -Majority of the data lies under 1000. The rest is just a very small number of data points.

Teacher Guides Student to Stop Screen Share

FEEDBACK

- **Appreciate the student for their efforts**
- **Identify 2 strengths and 1 area of progress for the student**

Step 4: **Wrap-Up** **(5 min)**	So, in this data story class we reviewed the concepts we have learned so far. How was your experience?	**ESR:** varied

	Amazing. While working on this data story , we also made sure that we are at the top of all the concepts we have acquired so far. Next class, we will be learning new concepts and building new projects.	-
	Congratulations! You have accomplished a milestone. In this Capstone project, you will analyse and visualise a given data set and write a data story. In order to achieve this, you have to apply the learnings from the past few classes.	
Project Overview	Data Story -1 **Goal of the Project:** In this project you will analyse and visualise a given data set and write a data story based on it . **Story:** In our journey of analyzing the articlele's data, you also want to understand how the results are changing after the introduction of an intervention. write a program to do the z test of a given sample.	

	I am very excited to see your project solution and I know you will do really well. Bye Bye!	
	Teacher Clicks	
Additional Activities	*Encourage the student to write reflection notes in their reflection journal using markdown.* Use these as guiding questions: ● What happened today? - Describe what happened - Code I wrote ● How did I feel after the class? ● What have I learned about programming and developing games? ● What aspects of the class helped me? What did I find difficult?	*The student uses the markdown editor to write her/his reflection in a reflection journal.*

Activity	Activity Name	Links
Teacher Activity 1	Colab Introduction	https://youtu.be/inN8seMm7UI
Teacher Activity 2	Colab notebook link	https://colab.research.google.com/notebooks/intro.ipynb#recent=true

Teacher Activity 3	Colab Reference (final code)	https://colab.research.google.com/gist/shubhamwhj/f0113b4cd0b9f31e49d5e7ff42b95c94/c112-v3-reference-code.ipynb
Student Activity1	Colab notebook link	https://colab.research.google.com/notebooks/intro.ipynb#recent=true
Student activity 2	savings data	https://raw.githubusercontent.com/whitehatjr/datasets/master/savings_data_final.csv