



LOVELY
PROFESSIONAL
UNIVERSITY

**SIX WEEKS SUMMER TRAINING
REPORT**

on

PYTHON for DATA SCIENCE

Submitted by-

Rishindra Mani Katiyar

Registration No-11715013

Program Name-B.Tech(Computer Science and Engineering)

Under the Guidance of

Ms. Shivani Kapania

(Data Science Instructor at upGrad)

School of Computer Science & Engineering

Lovely Professional University, Phagwara

(April-July, 2019)

DECLARATION

I hereby declare that I have completed my six weeks summer training at upGrad from 1st April to 30th July under the guidance of Ms. Shivani Kapania.

I declare that I have worked with full dedication during these six weeks of training and my learning outcomes fulfill the requirements of training for the award of degree of (B. Tech CSE), Lovely Professional University, Phagwara.

Rishindra Mani Katiyar

11715013

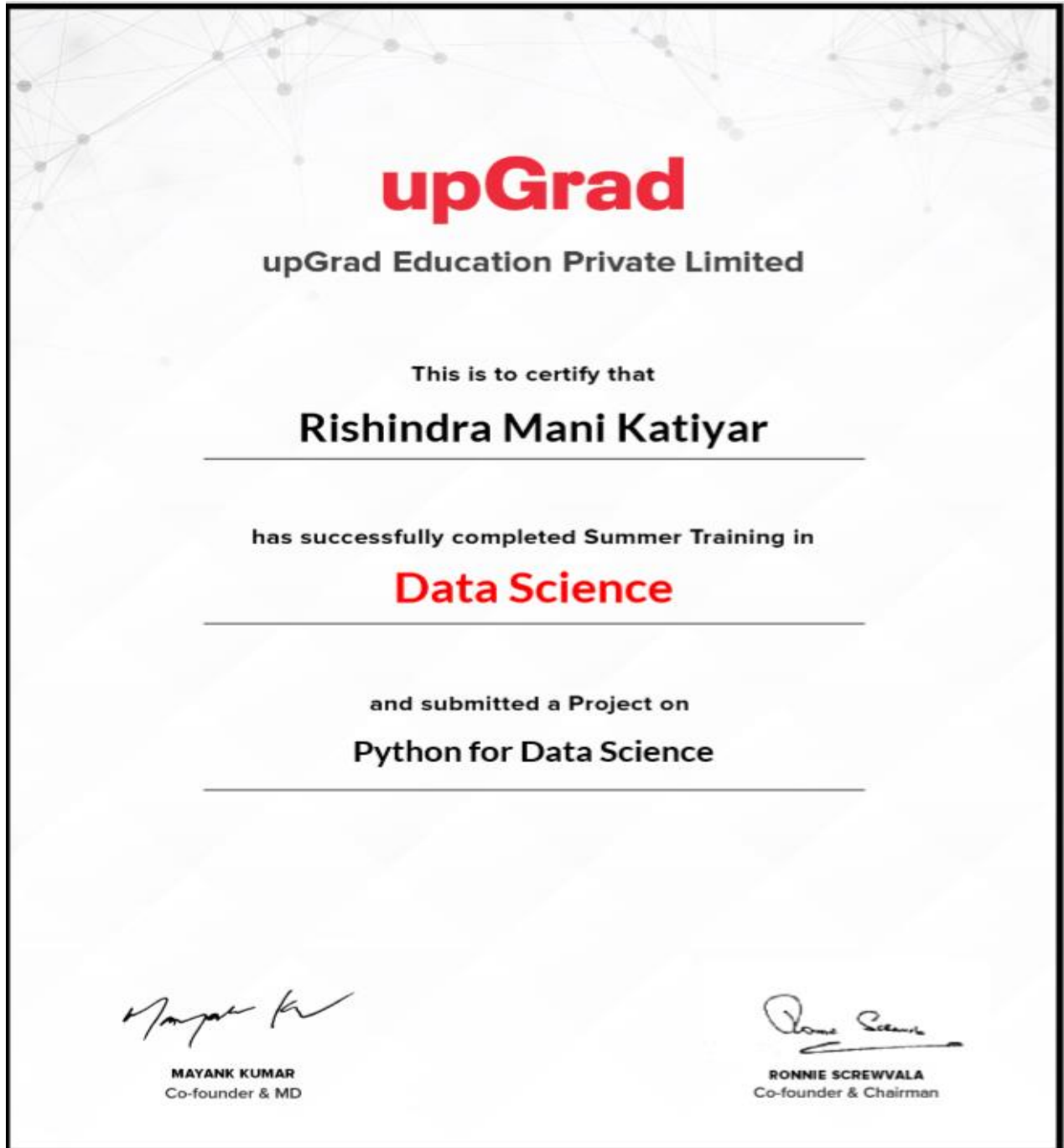
Date: 01/08/2019

ACKNOWLEDGEMENT

The summer training opportunity with upGrad was a great chance for learning as well as to upgrade myself to industry required skills in the field of Data Science. I owe special debt of gratitude to my Instructor, Ms. Shivani Kapania for her constant support and guidance throughout the training period. Her sincerity, thoroughness and perseverance have been a constant source of inspiration to me. It is only because of her cognizant efforts that I was able to complete my summer training and project with outstanding grades.

I am also thankful to my mentor and peers who gave me valuable suggestions and motivated me to complete the assignments and project on time. I perceive this opportunity as a big milestone in my career development. I will strive to use gained skills and knowledge in the best possible way, and will continue to work on their improvement, in order to attain desired career objectives.

TRAINING CERTIFICATE



Credential URL: <https://www.credential.net/ogdh34wz>



UPGRAD EDUCATION PRIVATE LIMITED

22nd July, 2019

Dear **Rishindra Mani Katiyar**,

This letter is to acknowledge that you are currently pursuing Certification Program in **Data Science** offered by upGrad commenced in **April 2019**.

We wish you all the best for the remainder of the program. Please accept our best wishes for your continued success in all future endeavours.

With best wishes,

A handwritten signature in blue ink, appearing to read "Mayank Kumar".

Mayank Kumar
(Co-Founder & MD)

TABLE OF CONTENTS

S.No.	Contents	Page No.
1	Declaration	2
2.	Acknowledgement	3
3.	Summer Training Certificate	4-5
4.	Index	6
5.	Introduction	7-9
6.	Why Data Science	10
7.	Profile of the Problem	11
8.	Code Implementation	12-25
9.	Learning Outcomes	26-28
10.	Gantt Chart	28
11.	Bibliography	29

INTRODUCTION

Data science is the field of study that combines domain expertise, programming skills, and knowledge of math and statistics to extract meaningful insights from data. Data science practitioners apply machine learning algorithms to numbers, text, images, video, audio, and more to produce artificial intelligence (AI) systems that perform tasks which ordinarily require human intelligence. In turn, these systems generate insights that analysts and business users translate into tangible business value.

Data science – discovery of data insight

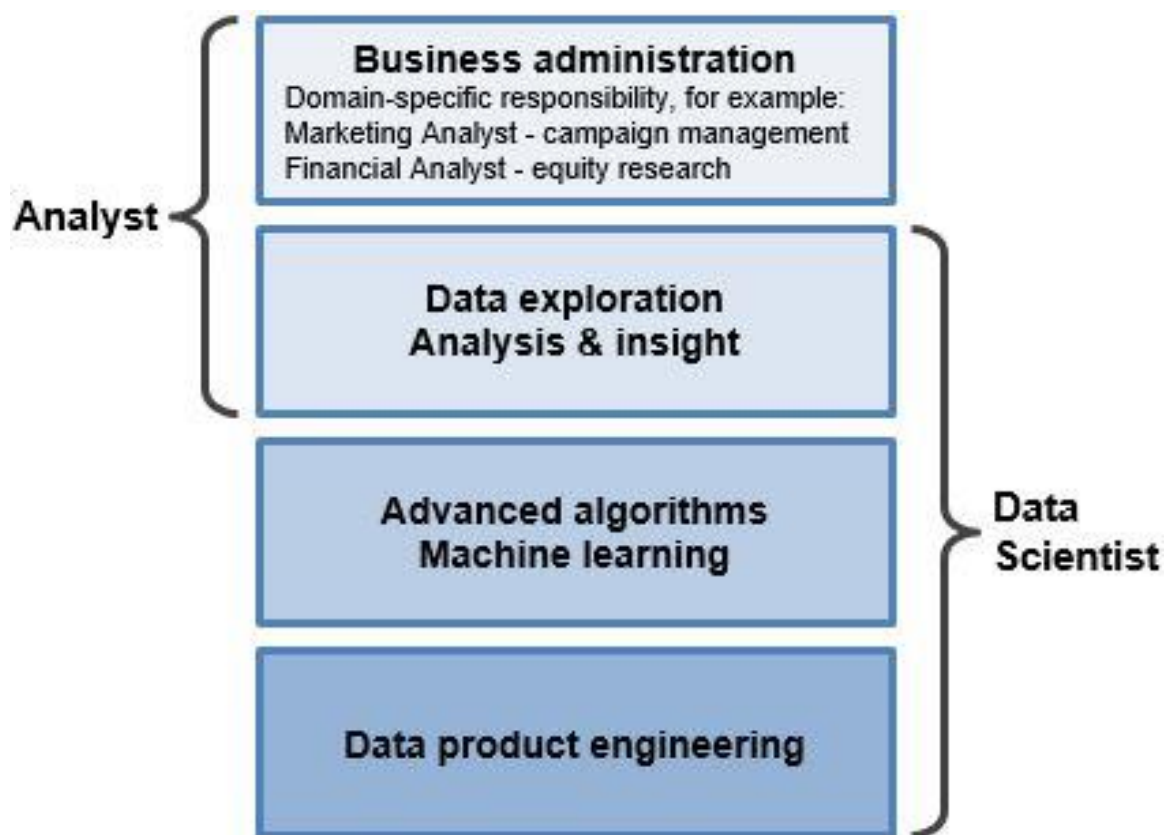
This aspect of data science is all about uncovering findings from data. Diving in at a granular level to mine and understand complex behaviors, trends, and inferences. It's about surfacing hidden insight that can help enable companies to make smarter business decisions. For example:

- Netflix data mines movie viewing patterns to understand what drives user interest, and uses that to make decisions on which Netflix original series to produce.
- Target identifies what are major customer segments within its base and the unique shopping behaviors within those segments, which helps to guide messaging to different market audiences.
- Proctor & Gamble utilizes time series models to more clearly understand future demand, which help plan for production levels more optimally.

How do data scientists mine out insights? It starts with data exploration. When given a challenging question, data scientists become detectives. They investigate leads and try to understand pattern or characteristics within the data. This requires a big dose of analytical creativity.

Then as needed, data scientists may apply quantitative technique in order to get a level deeper – e.g. inferential models, segmentation analysis, time series forecasting, synthetic control experiments, etc. The intent is to scientifically piece together a forensic view of what the data is really saying.

This data-driven insight is central to providing strategic guidance. In this sense, data scientists act as consultants, guiding business stakeholders on how to act on findings.



Python: The Meaning of Life in Data Science

The name is appropriated from Monty Python, which creator Guido Van Rossum selected to indicate that Python should be fun to use. It's common to find obscure Monty Python sketches referenced in Python code examples and documentation.

For this reason and others, Python is much beloved by programmers. Data scientists coming from engineering or scientific backgrounds might feel like the barber turned axe-man in The Lumberjack Song the first time they try to use it for data analysis—a little bit out of place.

But Python’s inherent readability and simplicity make it relatively easy to pick up and the number of dedicated analytical libraries available today mean that data scientists in almost every sector will find packages already tailored to their needs freely available for download.

Because of Python’s extensibility and general-purpose nature, it was inevitable as its popularity exploded that someone would eventually start using it for data analytics. As a jack of all trades, Python is not especially well-suited to statistical analysis, but in many cases, organizations already heavily invested in the language saw advantages to standardizing on it and extending it to that purpose.



Fig: -Application of Data Science

WHY DATA SCIENCE

Without the expertise of professionals who turn cutting-edge technology into actionable insights, Big Data is nothing. Today, more and more organizations are opening up their doors to big data and unlocking its power—increasing the value of a data scientist who knows how to tease actionable insights out of gigabytes of data.

What Does a Data Scientist Do?

Most data scientists in the industry have advanced and training in statistics, math, and computer science. Their experience is a vast horizon that also extends to data visualization, data mining, and information management. It is fairly common for them to have previous experience in infrastructure design, cloud computing, and data warehousing. Here are some advantages of data science in business:

- **Mitigating risk and fraud:**

Data scientists are trained to identify data that stands out in some way. They create statistical, network, path, and big data methodologies for predictive fraud propensity models and use those to create alerts that help ensure timely responses when unusual data is recognized.

- **Delivering relevant products:**

One of the advantages of data science is that organizations can find when and where their products sell best. This can help deliver the right products at the right time—and can help companies develop new products to meet their customers' needs.

- **Personalized customer experiences.**

One of the most buzzworthy benefits of data science is the ability for sales and marketing teams to understand their audience on a very granular level. With this knowledge, an organization can create the best possible customer experiences.

PROFILE OF THE PROBLEM

In this assignment, we will try to find some interesting insights into a few movies released between 1916 and 2016, using Python. This is a assignment wherein we download a movie dataset, write Python code to explore the data, gain insights into the movies, actors, directors, and collections, and execute it. 'Movie Assignment.ipynb' file is a commented Jupyter IPython Notebook file in which all the instructions and tasks to be performed are mentioned.

Basic Tips: -

- 1.** Identify the task to be performed correctly, and only then proceed to write the required code. Don't perform any incorrect analysis or look for information that isn't required for the assignment.
- 2.** In some cases, the variable names have already been assigned, and you just need to write code against them. In other cases, the names to be given are mentioned in the instructions. We strongly advise you to use the mentioned names only.
- 3.** Always keep inspecting your data frame after you have performed a particular set of operations.
- 4.** There are some checkpoints given in the IPython notebook provided. They're just useful pieces of information you can use to check if the result you have obtained after performing a particular task is correct or not.

CODE IMPLEMENTATION

```
# In[1]:
```

```
# Suppress Warnings
```

```
import warnings
```

```
warnings.filterwarnings('ignore')
```

```
# In[2]:
```

```
# Import the numpy and pandas packages
```

```
import numpy as np
```

```
import pandas as pd
```

```
# ## Task 1: Reading and Inspection
```

```
#
```

```
# - ### Subtask 1.1: Import and read
```

```
#
```

```
# Import and read the movie database. Store it in a variable called `movies`.
```

```
# In[3]:
```

```
movies = pd.read_csv('Movie+Assignment+Data.csv')# Write your code for importing  
the csv file here
```

```
# - ### Subtask 1.2: Inspect the dataframe
```

```
#
```

```
# Inspect the dataframe's columns, shapes, variable types etc.
```

```
# In[4]:
```

Write your code for inspection here

print(movies.shape)#Return the Shape of DataFrame (Rows , Columns)

print(movies.size) #Return the number of element in dataFrame

print(movies.columns)#Return the Index/Label of Columns

print(movies.dtypes) #Return the datatype of each column in the form of series

rc=len(movies.axes[0]) #returns total rows in initial datafrme....used in further computation

Task 2: Cleaning the Data

#

- ### Subtask 2.1: Inspect Null values

#

Find out the number of Null values in all the columns and rows. Also, find the percentage of Null values in each column. Round off the percentages upto two decimal places.

In[5]:

Write your code for column-wise null count here

movies.isnull().sum()

In[6]:

Write your code for row-wise null count here

movies.isnull().sum(axis=1)

In[7]:

Write your code for column-wise null percentages here

```
round(movies.isnull().mean() * 100,2)
```

- ### Subtask 2.2: Drop unnecessary columns

```
#
```

For this assignment, you will mostly be analyzing the movies with respect to the ratings, gross collection, popularity of movies, etc. So many of the columns in this dataframe are not required. So it is advised to drop the following columns.

```
# - color
```

```
# - director_facebook_likes
```

```
# - actor_1_facebook_likes
```

```
# - actor_2_facebook_likes
```

```
# - actor_3_facebook_likes
```

```
# - actor_2_name
```

```
# - cast_total_facebook_likes
```

```
# - actor_3_name
```

```
# - duration
```

```
# - facenumber_in_poster
```

```
# - content_rating
```

```
# - country
```

```
# - movie_imdb_link
```

```
# - aspect_ratio
```

```
# - plot_keywords
```

Write your code for dropping the columns here. It is advised to keep inspecting the #dataframe after each set of operations

```
movies.drop(['color', 'director_facebook_likes',  
'actor_1_facebook_likes', 'actor_2_facebook_likes', 'actor_3_facebook_likes',  
'actor_2_name', 'cast_total_facebook_likes', 'actor_3_name', 'duration', 'facenumber_in_p  
oster', 'content_rating', 'country',
```

```
'movie_imdb_link', 'aspect_ratio', 'plot_keywords'], axis=1, inplace=True)
```

movies.shape #new shape after dropping columns

- ### Subtask 2.3: Drop unnecessary rows using columns with high Null #percentages

#

Now, on inspection you might notice that some columns have large percentage (greater than 5%) of Null values. Drop all the rows which have Null values for such columns.

In[11]:

inspection to check which columns have greater than 5% of Null values.

```
round(movies.isnull().mean() * 100, 2) > 5
```

on inspection we find that gross and budget are two columns which have greater than 5% of Null values.

In[12]:

Write your code for dropping the rows here

```
movies.dropna(axis=0, subset=['budget', 'gross'], inplace=True)
```

#dropping rows which have null values for columns budget and gross

In[14]:

```
movies.shape
```

```
# - ### Subtask 2.4: Drop unnecessary rows
```

```
#
```

```
# Some of the rows might have greater than five NaN values. Such rows aren't of  
much use for the analysis and hence, should be removed.
```

```
# In[15]:
```

```
# Write your code for dropping the rows here
```

```
a = np.where(movies.isnull().sum(axis=1)>5)
```

```
movies=movies.drop(movies.index[a])
```

```
#Dropping rows which have greater than five NaN values
```

```
# In[16]:
```

```
movies.shape
```

```
#However after performing Subtask 2.3 no such rows exist which is evident from the  
shape of the dataframe.
```

```
# - ### Subtask 2.5: Fill NaN values
```

```
#
```

```
# You might notice that the `language` column has some NaN values. Here, on  
inspection, you will see that it is safe to replace all the missing values with `English`.
```

```
# In[17]:
```

```
# inspection to check which columns still have Null values.
```

```
movies.isnull().sum()
```

```
# In[18]:
```


Write your code for filling the NaN values in the 'language' column here

```
movies['language']=movies['language'].fillna('English')
```

In[19]:

Now language column has no missing values

```
movies.isnull().sum()
```

- **Subtask 2.6: Check the number of retained rows**

#

You might notice that two of the columns viz. `num_critic_for_reviews` and `actor_1_name` have small percentages of NaN values left. You can let these columns as it is for now. Check the number and percentage of the rows retained after completing all the tasks above.

In[20]:

Write your code for checking number of retained rows here

```
round(movies.shape[0]/rc*100,2) #original rows in movies dataset were 5043  
i.e.inspected in subtask 1.2 and stored in variable rc
```

Checkpoint 1: You might have noticed that we still have around `77%` of the rows!

Task 3: Data Analysis

#

- **Subtask 3.1: Change the unit of columns**

#

Convert the unit of the `budget` and `gross` columns from `\$` to `million \$`.

In[21]:

```
# Write your code for unit conversion here
```

```
movies.budget = movies.budget.apply(lambda x: x/1000000)
```

```
movies.gross = movies.gross.apply(lambda x: x/1000000)
```

```
# - ### Subtask 3.2: Find the movies with highest profit
```

```
#
```

```
# 1. Create a new column called `profit` which contains the difference of the two  
columns: `gross` and `budget`.
```

```
# 2. Sort the dataframe using the `profit` column as reference.
```

```
# 3. Extract the top ten profiting movies in descending order and store them in a  
new dataframe - `top10`
```

```
# In[22]:
```

```
# Write your code for creating the profit column here
```

```
movies['profit']=movies['gross']-movies['budget']
```

```
# In[23]:
```

```
# Write your code for sorting the dataframe here
```

```
movies = movies.sort_values(by=['profit'], ascending=False)
```

```
# In[24]:
```

```
top10 =movies.head(10) # Write your code to get the top 10 profiting movies here
```

```
# - ### Subtask 3.3: Drop duplicate values
```

```
#
```

```
# After you found out the top 10 profiting movies, you might have notice a duplicate  
#value. So, it seems like the dataframe has duplicate values as well. Drop the  
#duplicate values from the dataframe and repeat `Subtask 3.2`.
```

```

# In[25]:

# To inspect if top10 has duplicated values

top10.duplicated()

#result shows that duplicate rows might exist in movies dataframe too so we need to
#remove them

# In[26]:

# Write your code for dropping duplicate values here

movies.drop_duplicates(inplace=True)

# In[27]:

# Write code for repeating subtask 2 here

movies = movies.sort_values(by=['profit'], ascending=False)

top10 =movies.head(10)

# In[28]:

top10 # displayong top ten profiting movies

# **Checkpoint 2:** You might spot two movies directed by `James Cameron` in the
#list.

# - ### Subtask 3.4: Find IMDb Top 250

#

# 1. Create a new dataframe `IMDb_Top_250` and store the top 250 movies with
the highest IMDb Rating (corresponding to the column: `imdb_score`). Also make
sure that for all of these movies, the `num_voted_users` is greater than 25,000.

# Also add a `Rank` column containing the values 1 to 250 indicating the ranks of the
corresponding films.

```

```
# 2. Extract all the movies in the `IMDb_Top_250` dataframe which are not in the English language and store them in a new dataframe named `Top_Foreign_Lang_Film`.
```

```
# In[29]:
```

```
# Write your code for extracting the top 250 movies as per the IMDb score here. Make sure that you store it in a new dataframe
```

```
# and name that dataframe as 'IMDb_Top_250'
```

```
IMDb_Top_250=(movies.sort_values(by='imdb_score',ascending=False).where(movies.num_voted_users>25000)).dropna().head(250)
```

```
IMDb_Top_250.insert(0, 'Rank', range(1,251))
```

```
# In[30]:
```

```
IMDb_Top_250 #displaying top 250 movies
```

```
# In[31]:
```

```
# Write your code to extract top foreign language films from 'IMDb_Top_250' here
```

```
Top_Foreign_Lang_Film = IMDb_Top_250[(IMDb_Top_250['language']!='English')]
```

```
# In[32]:
```

```
Top_Foreign_Lang_Film #displaying top foreign language films from 'IMDb_Top_250'
```

```
# **Checkpoint 3:** Can you spot `Veer-Zaara` in the dataframe?
```

```
# - ### Subtask 3.5: Find the best directors
```

```
#
```

```
# 1. Group the dataframe using the `director_name` column.
```

2. Find out the top 10 directors for whom the mean of `imdb_score` is the highest and store them in a new dataframe `top10director`.

In[33]:

Write your code for extracting the top 10 directors here

```
top10director=movies.groupby(['director_name']).agg({'imdb_score' : np.mean})
```

```
top10director=top10director.sort_values(by=['imdb_score'],  
ascending=[False]).head(10)
```

In[34]:

top10director #displaying top 10 directors for whom the mean of imdb_score is the
#highest

**Checkpoint 4: No surprises that `Damien Chazelle` (director of Whiplash
#and La La Land) is in this list.**

- ### Subtask 3.6: Find popular genres

#

You might have noticed the `genres` column in the dataframe with all the genres of the movies separated by a pipe (`|`). Out of all the movie genres, the first two are most significant for any film.

1. Extract the first two genres from the `genres` column and store them in two new columns: `genre_1` and `genre_2`. Some of the movies might have only one genre. In such cases, extract the single genre into both the columns, i.e. for such movies the `genre_2` will be the same as `genre_1`.

2. Group the dataframe using `genre_1` as the primary column and `genre_2` as the secondary column.

3. Find out the 5 most popular combo of genres by finding the mean of the gross values using the `gross` column and store them in a new dataframe named `PopGenre`.

In[35]:

Write your code for extracting the first two genres of each movie here

```
ext = movies["genres"].str.split("|", n = 2, expand = True)#extracting first two genres
```

```
movies["genre_1"] = ext[0]
```

```
movies["genre_2"] = ext[1]
```

#Filling 'genre_2' column missing values with 'genre_1'

```
movies['genre_2'] = movies['genre_2'].fillna(movies['genre_1'])
```

In[36]:

Write your code for grouping the dataframe here

```
movies_by_segment = movies.groupby(['genre_1', 'genre_2'])
```

In[37]:

Write your code for getting the 5 most popular combo of genres here

```
PopGenre = movies_by_segment.mean().nlargest(5, columns='gross')
```

In[38]:

```
PopGenre #displaying 5 most popular combo of genres
```

Checkpoint 5: Well, as it turns out. `Family + Sci-Fi` is the most popular combo of genres out there!

- **### Subtask 3.7: Find the critic-favorite and audience-favorite actors**

#

```
# 1. Create three new dataframes namely, `Meryl_Streep`, `Leo_Caprio`, and  
`Brad_Pitt` which contain the movies in which the actors: 'Meryl Streep', 'Leonardo  
DiCaprio', and 'Brad Pitt' are the lead actors. Use only the `actor_1_name` column  
for extraction. Also, make sure that you use the names 'Meryl Streep', 'Leonardo  
DiCaprio', and 'Brad Pitt' for the said extraction.  
  
# 2. Append the rows of all these dataframes and store them in a new dataframe  
named `Combined`.  
  
# 3. Group the combined dataframe using the `actor_1_name` column.  
  
# 4. Find the mean of the `num_critic_for_reviews` and `num_user_for_review` and  
identify the actors which have the highest mean.
```

```
# In[39]:
```

```
# Write your code for creating three new dataframes here
```

```
Meryl_Streep = movies[movies['actor_1_name'] == 'Meryl Streep']
```

```
# Include all movies in which Meryl_Streep is the lead
```

```
# In[40]:
```

```
Leo_Caprio = movies[movies['actor_1_name'] == 'Leonardo DiCaprio']
```

```
# Include all movies in which Leo_Caprio is the lead
```

```
# In[41]:
```

```
Brad_Pitt = movies[movies['actor_1_name'] == 'Brad Pitt']
```

```
# Include all movies in which Brad_Pitt is the lead
```

```
# In[42]:
```

```
# Write your code for combining the three dataframes here
```

```
Combined= Meryl_Streep.append([Leo_Caprio, Brad_Pitt])
```

```
# In[43]:
```

```
# Write your code for grouping the combined dataframe here
```

```
Combined=Combined.groupby(['actor_1_name'])
```

```
# In[44]:
```

```
# Write the code for finding the mean of critic reviews and audience reviews here
```

```
print(Combined.agg({'num_critic_for_reviews' :  
np.mean}).sort_values(by=['num_critic_for_reviews'], ascending=[False]))
```

```
print(Combined.agg({'num_user_for_reviews' :  
np.mean}).sort_values(by=['num_user_for_reviews'], ascending=[False]))
```

```
# **Checkpoint 6:** `Leonardo` has aced both the lists!
```

Typical jupyter code execution:

The screenshot shows a Jupyter Notebook interface with the following content:

Subtask 2.4: Drop unnecessary rows

Some of the rows might have greater than five NaN values. Such rows aren't of much use for the analysis and hence, should be removed.

```
In [15]: # Write your code for dropping the rows here  
a = np.where(movies.isnull().sum(axis=1)>5)  
movies=movies.drop(movies.index[a])  
#Dropping rows which have greater than five NaN values
```

```
In [16]: movies.shape  
#However after performing Subtask 2.3 no such rows exist which is evident from the shape of the dataframe.
```

```
Out[16]: (3891, 13)
```

Subtask 2.5: Fill NaN values

You might notice that the `language` column has some NaN values. Here, on inspection, you will see that it is safe to replace all the missing values with `'English'`.

```
In [17]: # inspection to check which columns still have Null values.  
movies.isnull().sum()
```

```
Out[17]: director_name      0  
num_critic_for_reviews    1  
gross                    0  
genres                   0  
actor_1_name             3  
movie_title              0  
num_voted_users          0  
num_user_for_reviews     0  
language                 3  
budget                  0  
title_year              0  
imdb_score               0  
movie_facebook_likes     0  
dtype: int64
```

```
In [18]: # Write your code for filling the NaN values in the 'language' column here  
movies['language']=movies['language'].fillna('English')
```


• Subtask 3.7: Find the critic-favorite and audience-favorite actors

1. Create three new dataframes namely `Meryl_Streep`, `Leo_Caprio`, and `Brad_Pitt` which contain the movies in which the actors: 'Meryl Streep', 'Leonardo DiCaprio', and 'Brad Pitt' are the lead actors. Use only the `actor_1_name` column for extraction. Also, make sure that you use the names 'Meryl Streep', 'Leonardo DiCaprio', and 'Brad Pitt' for the said extraction.
2. Append the rows of all these dataframes and store them in a new dataframe named `Combined`.
3. Group the combined dataframe using the `actor_1_name` column.
4. Find the mean of the `num_critic_for_reviews` and `num_user_for_review` and identify the actors which have the highest mean.

```
In [39]: # Write your code for creating three new dataframes here
Meryl_Streep = movies[movies['actor_1_name'] == 'Meryl Streep']# Include all movies in which Meryl_Streep is the Lead

In [40]: Leo_Caprio = movies[movies['actor_1_name'] == 'Leonardo DiCaprio']# Include all movies in which Leo_Caprio is the Lead

In [41]: Brad_Pitt = movies[movies['actor_1_name'] == 'Brad Pitt']# Include all movies in which Brad_Pitt is the Lead

In [42]: # Write your code for combining the three dataframes here
Combined= Meryl_Streep.append([Leo_Caprio, Brad_Pitt])

In [43]: # Write your code for grouping the combined dataframe here
Combined=Combined.groupby(['actor_1_name'])


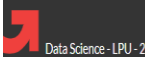
In [44]: # Write the code for finding the mean of critic reviews and audience reviews here
print(Combined.agg({'num_critic_for_reviews' : np.mean}).sort_values(by=['num_critic_for_reviews'], ascending=[False]))
print(Combined.agg({'num_user_for_reviews' : np.mean}).sort_values(by=['num_user_for_reviews'], ascending=[False]))
```

	num_critic_for_reviews
actor_1_name	
Leonardo DiCaprio	330.190476
Brad Pitt	245.000000
Meryl Streep	181.454545

	num_user_for_reviews
actor_1_name	
Leonardo DiCaprio	914.476190
Brad Pitt	742.352941
Meryl Streep	297.181818

Checkpoint 6: Leonardo has aced both the lists!

Assignment Score:




Module 7 > Session 1 > Python Assignment > Feedback

← Python Assignment Feedback

ASSIGNMENT SCORE

149 / 150

YOUR SUBMISSION

 submitfinal.ipynb

ASSESSMENT CRITERIA

Data Reading and Inspection

4/5

Feedback:

You can use commands:

df.shape - To find the shape of the data (rows and columns)

df.info() - To get a concise summary of the dataframe.

df.describe() - To see basic statistical details like percentile, mean, std etc

1.1. Reading the Data

Feedback: You have successfully loaded the dataset.

1.2. Inspecting the Data

Feedback:

You haven't used info and describe methods to inspect the dataframe. It is a good practice to inspect the data before further analysis.

Data Cleaning

25/25

Feedback:

Overall superb work!

LEARNING OUTCOMES

- **DEALING WITH MISSING VALUES IN PYTHON:**

When no data value is stored for feature for a particular observation, we say this feature has a missing value. Usually missing value in data set appears as question mark and a zero or just a blank cell. There are many ways to deal with missing values and this is regardless of language used.

- Check the source.
- Drop the missing value - Either drop the variable or the drop the data entry.
- Replace the missing value – Replace it with average or frequency or replace it based on some other function.
- Leave missing data as it is.

- **DATA FORMATTING IN PYTHON**

Data formatting means bringing data into a common standard of expression that allows users to make meaningful comparisons. As a part of dataset cleaning, data formatting ensures the data is consistent and easily understandable. It includes renaming columns to appropriate names and changing to appropriate data types.

- **NumPy**

NumPy is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays. The ancestor of NumPy, Numeric, was originally created by Jim Hugunin with contributions from several other developers. In 2005, Travis Oliphant created NumPy by incorporating features of the

competing Numarray into Numeric, with extensive modifications. NumPy is open-source software and has many contributors.

Example: array creation

```
>>> import numpy as np
```

```
>>> x = np.array([1, 2, 3])
```

```
>>> x
```

```
array([1, 2, 3])
```

```
>>> y = np.arange(10) # like Python's range, but returns an array
```

```
>>> y
```

```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

- **PANDAS**

In computer programming, pandas is a software library written for the Python programming language for data manipulation and analysis. In particular, it offers data structures and operations for manipulating numerical tables and time series. The name is derived from the term "panel data", an econometrics term for data sets that include observations over multiple time periods for the same individuals.

Library Features:

- DataFrame object for data manipulation with integrated indexing.
- Tools for reading and writing data between in-memory data structures and different file formats.
- Data alignment and integrated handling of missing data.
- Reshaping and pivoting of data sets.
- Label-based slicing, fancy indexing, and subsetting of large data sets.

- Group by engine allowing split-apply-combine operations on data sets.
- Data set merging and joining.
- Hierarchical axis indexing to work with high-dimensional data in a lower-dimensional data structure.
- Provides data filtration.

• JUPYTER NOTEBOOK

In this Project, we use Jupyter Notebook in python 3 environment to run the queries for each subtask. Notebook documents are documents produced by the Jupyter Notebook App, which contain both computer code (e.g. python) and rich text elements (paragraph, equations, figures, links, etc.). Notebook documents are both human-readable documents containing the analysis description and the results (figures, tables, etc..) as well as executable documents which can be run to perform data analysis. The Jupyter Notebook App is a server-client application that allows editing and running notebook documents via a web browser. The Jupyter Notebook App can be executed on a local desktop requiring no internet access (as described in this document) or can be installed on a remote server and accessed through the internet.

GANTT CHART

Date	Introduction to Data Science	Data Analysis using SQL	Basics of Python	NumPy and Pandas	Cleaning Data in Python	Project Work
01/04-11/04						
12/04-30/04						
01/05-22/05						
23/05-20/06						
21/06-09/07						
10/07-25/07						

BIBLIOGRAPHY

- <https://learn.upgrad.com>
- <https://www.guru99.com/numpy-tutorial.html>
- <https://towardsdatascience.com>
- <https://docs.scipy.org/doc>
- <https://pandas.pydata.org/pandas-docs/stable/>