

**A MINI PROJECT REPORT**  
**ON**  
**“Spaceship Titanic Data Analysis”**

Submitted in the partial fulfillment of the requirements for

The degree of

**BACHELOR OF ENGINEERING IN COMPUTER ENGINEERING**

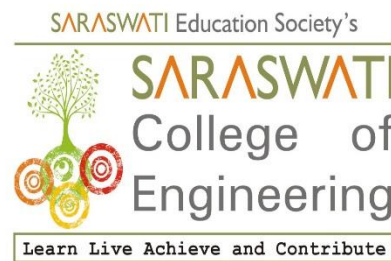
**By**

- 1) NAVNATH AUTI
- 2) SAHIL NAZARE
- 3) BINITDEV PANDEY
- 4) ADNAN SHAIKH

**UNDER THE GUIDANCE OF**

**Dr. Anjali Dadhich**

**Prof. Roshani Rajput**



Department of Computer Engineering  
Saraswati College of Engineering, Kharghar, Navi-Mumbai  
University of Mumbai  
2022-23

# **Saraswati College of Engineering, Kharghar**

## **Vision:**

To be universally accepted as autonomous centre of learning in Engineering Education and Research.

## **Mission:**

- To educate students to become responsible and quality technocrats to fulfil society and industry needs.
- To nurture student's creativity and skills for taking up challenges in all facets of life.

---

## **Department of Computer Engineering**

## **Vision:**

To be among renowned institution in Computer Engineering Education and Research by developing globally competent graduates.

## **Mission:**

- To produce quality Engineering graduates by imparting quality training, hands on experience and value education.
- To pursue research and new technologies in Computer Engineering and across interdisciplinary areas that extends the scope of Computer Engineering and benefit humanity.
- To provide stimulating learning ambience to enhance innovative ideas, problem solving ability, leadership qualities, team-spirit and ethical responsibilities.



SARASWATI Education Society's  
**SARASWATI College of Engineering**

Learn Live Achieve and Contribute

Kharghar, Navi Mumbai - 410 210.

## **DEPARTMENT OF COMPUTER ENGINEERING**

### **PROGRAM EDUCATIONAL OBJECTIVE'S**

1. To embed a strong foundation of Computer Engineering fundamentals to identify, solve, analyze and design real time engineering problems as a professional or entrepreneur for the benefit of society.
2. To motivate and prepare students for lifelong learning & research to manifest global competitiveness.
3. To equip students with communication, teamwork and leadership skills to accept challenges in all the facts of life ethically.



SARASWATI Education Society's  
**SARASWATI College of Engineering**

Learn Live Achieve and Contribute

Kharghar, Navi Mumbai - 410 210.

## **DEPARTMENT OF COMPUTER ENGINEERING**

### **PROGRAM OUTCOMES**

1. Apply the knowledge of Mathematics, Science and Engineering Fundamentals to solve complex Computer Engineering Problems.
2. Identify, formulate and analyze Computer Engineering Problems and derive conclusion using First Principle of Mathematics, Engineering Science and Computer Science.
3. Investigate Complex Computer Engineering problems to find appropriate solution leading to valid conclusion.
4. Design a software System, components, Process to meet specified needs with appropriate attention to health and Safety Standards, Environmental and Societal Considerations.
5. Create, select and apply appropriate techniques, resources and advance Engineering software to analyze tools and design for Computer Engineering Problems.
6. Understand the Impact of Computer Engineering solution on society and environment for Sustainable development.
7. Understand Societal, health, Safety, cultural, Legal issues and Responsibilities relevant to Engineering Profession.
8. Apply Professional ethics, accountability and equity in Engineering Profession.
9. Work Effectively as a member and leader in multidisciplinary team for a common goal.
10. Communicate effectively within a Profession and Society at large.
11. Appropriately incorporate principles of Management and Finance in one's own Work.
12. Identify educational needs and engage in lifelong learning in a Changing World of Technology.



SARASWATI Education Society's  
**SARASWATI College of Engineering**

Learn Live Achieve and Contribute

Kharghar, Navi Mumbai - 410 210.

**DEPARTMENT OF COMPUTER ENGINEERING**  
**PROGRAMME SPECIFIC OUTCOME**

1. Formulate and analyze complex engineering problems in computer engineering (Networking/Big data/ Intelligent Systems/Cloud Computing/Real time systems).
2. Plan and develop efficient, reliable, secure and customized application software using cost effective emerging software tools ethically.



SARASWATI Education Society's  
**SARASWATI College of Engineering**

Learn Live Achieve and Contribute

Kharghar, Navi Mumbai - 410 210.

(Approved by AICTE, regd. By Maharashtra Govt. DTE ,Affiliated to Mumbai University)

PLOT NO. 46/46A, SECTOR NO 5, BEHIND MSEB SUBSTATION, KHARGHAR, NAVI MUMBAI-410210

Tel. : 022-27743706 to 11 \* Fax : 022-27743712 \* Website: www.sce.edu.in

## **CERTIFICATE**

*This is to certify that the requirements for the project report entitled "Spaceship Titanic Data Analysis" have been successfully completed by the following students:*

| Roll numbers | Name            |
|--------------|-----------------|
| 05           | Navnath Auti    |
| 46           | Sahil Nazare    |
| 48           | Binitdev Pandey |
| 68           | Adnan Shaikh    |

In partial fulfillment of Sem –VII , **Bachelor of Engineering of Mumbai University in Computer Engineering** of Saraswati college of Engineering , Kharghar during the academic year 2022-23.

### **Internal Guide**

Dr. Anjali Dadhich  
Prof. Roshani Rajput

### **External Examiner**

### **Project Co-ordinator**

Dr. Anjali Dadhich

### **Head of Department**

Prof. Sujata Bhairnallykar

### **Principal**

Dr. Manjusha Deshmukh

# DECLARATION

I declare that this written submission represents my ideas in my own words and where others ideas or words have been included. I have adequately cited and referenced the original sources. I also declare that I have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea/data/fact/source in my submission. I understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

1. Navnath Auti
2. Sahil Nazare
3. Binitdev Pandey
4. Adnan Shaikh

Date:

# **ACKNOWLEDGEMENT**

After the completion of this work, words are not enough to express feelings about all those who helped us to reach goal.

It's a great pleasure and moment of immense satisfaction for us to express my profound gratitude to **Project Guide, Dr. Anjali Dadhich**, whose constant encouragement enabled us to work enthusiastically. His perpetual motivation, patience and excellent expertise in discussion during progress of the project work have benefited us to an extent, which is beyond expression.

We would also like to give our sincere thanks to **Prof. Sujata Bhairnallykar, Head of Department**, and **Dr. Anjali Dadhich, Prof. Roshani Rajput** Internal guide from Department of Computer Engineering, Saraswati college of Engineering, Kharghar, Navi Mumbai, for their guidance, encouragement, and support during a project.

I am thankful to **Dr. Manjusha Deshmukh, Principal**, Saraswati College of Engineering, Kharghar, Navi Mumbai for providing an outstanding academic environment, also for providing the adequate facilities.

Last but not the least we would also like to thank all the staffs of Saraswati college of Engineering (Computer Engineering Department) for their valuable guidance with their interest and valuable suggestions brightened us.

1. Navnath Auti
2. Sahil Nazare
3. Binitdev Pandey
4. Adnan Shaikh



## ABSTRACT

Welcome to the year 2912, where your data science skills are needed to solve a cosmic mystery. We've received a transmission from four light years away and things aren't looking good.

The spaceship titanic was an interstellar passenger liner launched a month ago. With almost 13,000 passengers on board, the vessel set out on its maiden voyage transporting emigrants from our solar system to three newly habitable exoplanets orbiting nearby stars.

While rounding alpha centauri en route to its first destination—the torrid 55 cancri e—the unwary spaceship titanic collided with a space-time anomaly hidden within a dust cloud. Sadly, it met a similar fate as its namesake from 1000 years before. Though the ship stayed intact, almost half of the passengers were transported to an alternate dimension!

We have done Summarization and EDA of Spaceship Titanic Data using PySpark (Python interface for Spark) pandas, numpy and seaborn modules to extract insight on passengers' behaviour those are transported and those aren't and found relationship between various different features using feature engineering.

# Table of Contents

|  |           |
|--|-----------|
|  |           |
| <b>List of Figures .....</b>                       | <b>1</b>  |
| <b>1. Introduction .....</b>                       | <b>2</b>  |
| <b>1.1 General .....</b>                           | <b>2</b>  |
| <b>1.2 Objective and problem statement .....</b>   | <b>3</b>  |
| <b>2. Methodology .....</b>                        | <b>6</b>  |
| <b>2.1 Algorithmic details.....</b>                | <b>6</b>  |
| <b>2.2 Hardware and Software requirements.....</b> | <b>12</b> |
| <b>2.3 Design Details.....</b>                     | <b>13</b> |
| <b>3. Implementation and Results .....</b>         | <b>15</b> |
| <b>4. Conclusion and Future Scope.....</b>         | <b>18</b> |
| <b>5. References.....</b>                          | <b>19</b> |

## List of Figures

| Figure No. | Name                      | Page No. |
|------------|---------------------------|----------|
| 2.1        | Flow Chart                |          |
| 3.1        | Implementation and Result |          |

# CHAPTER 1

## INTRODUCTION

### 1.1 GENERAL

Data analysis is about asking and answering questions about our data. The objective of the data analysis step is to increase the understanding of the problem by better understanding the problems data. This involves providing multiple different ways to describe the data as an opportunity to review and capture observations and assumptions that can be tested in later experiments.

#### 1. Summarize Data

Summarizing the data is about describing the actual structure of the data. The minimum aspects of the data to summarize are the structure and the distributions.

##### Data Structure

Summarizing the data structure is about describing the number and data types of attributes. For example, going through this process highlights ideas for transforms in the Data Preparation step for converting attributes from one type to another (such as real to ordinal or ordinal to binary).

Some motivating questions for this step include:

How many attributes and instances are there?

What are the data types of each attribute (e.g. nominal, ordinal, integer, real, etc.)?

##### Data Distributions

Summarizing the distributions of each attributes can also flush out ideas for possible data transforms in the Data Preparation step, such as the need and effects of Discretization, Normalization and Standardization.

A summary of the distribution of each real valued attribute typically includes the minimum, maximum, median, mode, mean, standard deviation and number of missing values.

Some motivating questions for this step include:

Create a five-number summary of each real-valued attribute.

What is the distribution of the values for the class attribute?

Knowing the distribution of the class attribute (or mean of a regression output variable) is useful because we can use it to define the minimum accuracy of a predictive model.

## **2. Visualize Data**

Visualizing the data is about creating graphs that summarize the data, capturing them and studying them for interesting structure that can be described.

There are seemingly an infinite number of graphs we can create (especially in software like Python data analysis libraries such as Matplotlib and Seaborn, R, MATLAB, WEKA), we focused on Bar plots, Histograms, Box plots, Heat maps, Kernel density plots, Scatter plots and Pair plot.

## **PySpark**

PySpark is an interface for Apache Spark in Python. It not only allows us to write Spark applications using Python APIs, but also provides the PySpark shell for interactively analyzing our data in a distributed environment. PySpark supports most of Spark's features such as Spark SQL, DataFrame, Streaming, MLlib (Machine Learning) and Spark Core.

## **Pandas API on Spark**

Pandas API on Spark allows us to scale our pandas workload out. With this package, we can:

- Be immediately productive with Spark, with no learning curve, if already familiar with pandas.
- Have a single codebase that works both with pandas (tests, smaller datasets) and with Spark (distributed datasets).
- Switch to pandas API and PySpark API contexts easily without any overhead.

## **1.2 PROBLEM STATEMENT AND OBJECTIVE**

### **1.2.1 PROBLEM STATEMENT**

The Spaceship Titanic was an interstellar passenger liner launched a month ago. With almost 13,000 passengers on board, the vessel set out on its maiden voyage transporting emigrants from our solar system to three newly habitable exoplanets orbiting nearby stars.

While rounding Alpha Centauri en route to its first destination—the torrid 55 Cancri E—the unwary Spaceship Titanic collided with a spacetime anomaly hidden within a dust cloud. Sadly, it met a similar fate as its namesake from 1000 years before. Though the ship stayed intact, almost half of the passengers were transported to an alternate dimension!

Our task is to determine what factors are related to transportation of passengers to alternate dimension and how different features are related of given data through data analysis concepts and tools.

### **1.2.2 OBJECTIVE**

- Load data from online/offline source
- Create Data Frame of data using PySpark pandas class
- Check and correct data types of features
- Treat NULL values using suitable statistic
- Summarization of features
- Univariate Exploratory Data Analysis (EDA)
- Multi-variate EDA
- Extracting meaningful insight from Summarization and EDA

# CHAPTER 2

## METHODOLOGY

### 2.1 ALGORITHMIC DETAILS

**Algorithm:**

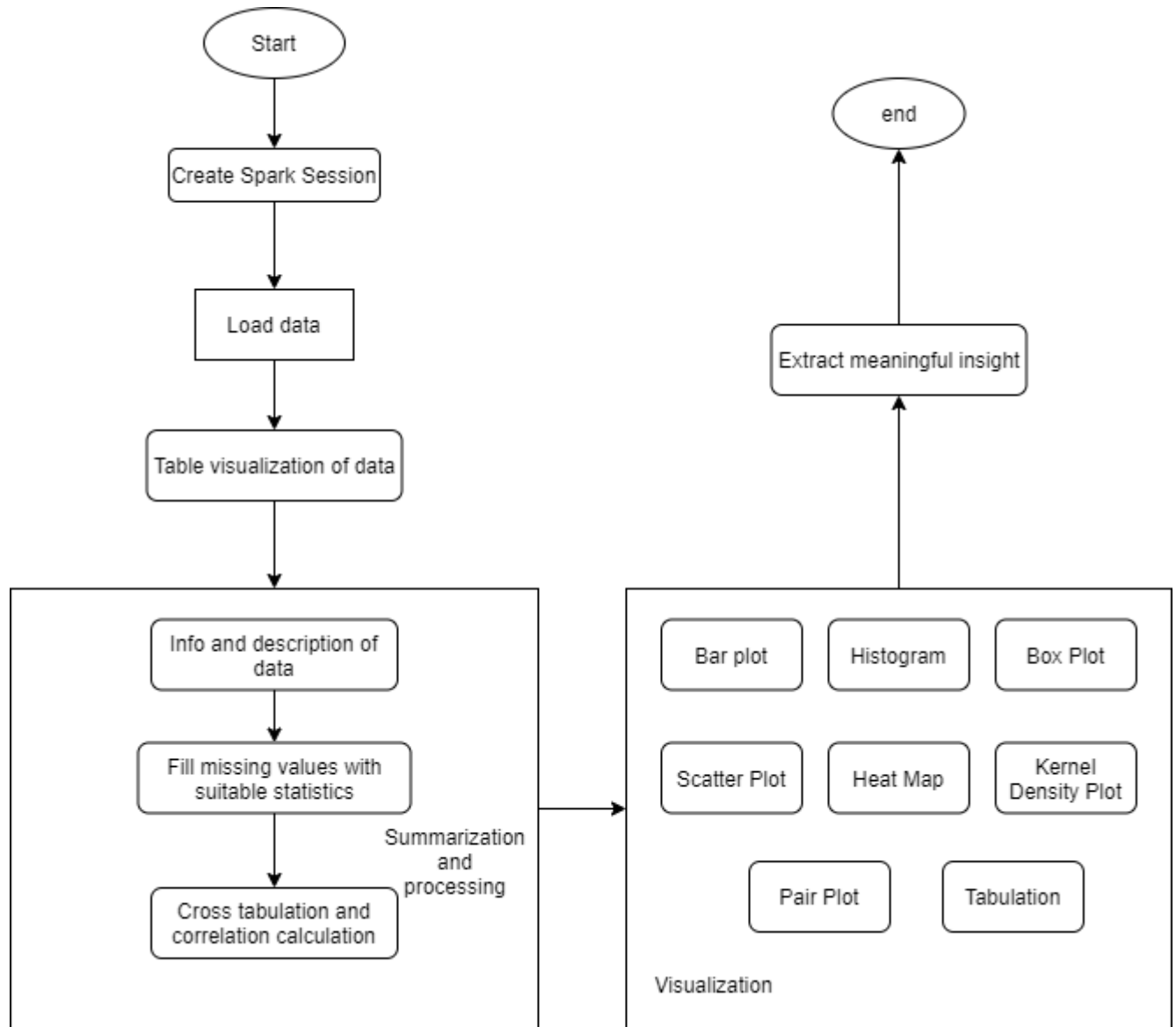
#### **Summarization and Wrangling**

1. **Info:** Data types of each feature
2. **Describe:** Statistics such as min, max, mean, median, mode, count, standard deviation and quantiles of suitable features
3. **Value counts:** Unique values in each feature with their respective count
4. **Nan with sum:** To count NULL values in each feature
5. **Cross Tabulation:** To get count of different combination of unique values in 2 features.
6. **Correlation:** To check how closely related different real features are.
7. **Normalization:** To get uniform scale for comparison
8. **Fill Nan:** To fill NULL value with suitable statistics
9. **Broadcasting:** To apply arithmetic operation and above specified operation on each value of corresponding feature

#### **Exploratory Data Analysis (EDA)**

1. **Head and Tail:** To get the values of subset of starting and ending records in frame format
2. **Bar Plot:** To visualize count of different value in features, if univariate no stacking is applied if bivariate stacking is applied and for trivariate third value is taken as a hue i.e. color. Mostly applied on categorical data type.
3. **Box Plot:** To visualize five descriptive statistics of real value features.
4. **Histogram:** To visualize count of data for real value features, automatic binning is applied.
5. **Scatter Plot:** To visualize relation between two features, if three features are used then third feature is set a hue.
6. **Kernel Density Plot:** Fit kernel suitable for data and show curve visualization accordingly
7. **Heat Map:** Visualize correlation between multiple real features.
8. **Pair Plot:** Visualize scatter (bivariate and trivariate) as well kernel density estimate between multiple features in single graph.

## Flowchart:



**Figure 2.1. Flow Chart**



## 2.2 HARDWARE AND SOFTWARE REQUIREMENTS

### 2.2.1 HARDWARE REQUIREMENTS

1. RAM : 4GB RAM
2. Hard Drive : 40 GB Hard Drive
3. Processor : Intel Dual Core Processor

### 2.2.2 SOFTWARE REQUIREMENTS

1. Anaconda
2. Jupyter Notebook
3. Python
4. PySpark
5. Pandas
6. Numpy
7. Matplotlib
8. Seaborn

## 2.3 DESIGN DETAILS:

Spark session is created using PySpark which return spark instance, using spark instance data is loaded from local repository (can be Hadoop cluster or local directory). Pandas is loaded using PySpark pandas module, numpy, matplotlib and seaborn loaded from their respective packages.

Loaded data is converted pandas data frame using inbuilt method of spark frame and it is visualize in tabular format using head method of frame.

Summarization and Pre-processing is done using built-in pandas, pandas dataframe and numpy methods and functions described in **section 2.1 summarization.**

Visualization is done using pandas data frame in-built methods and seaborn functions described in **section 2.1 EDA.** Meaningful insight is gained from these summarization and visualization of data.

# CHAPTER 3

## IMPLEMENTATION AND RESULTS

### Analysis of Spaceship Titanic Data

```
In [1]: from pyspark.sql import SparkSession
import seaborn as sns
import matplotlib.pyplot as plt
spark = SparkSession.builder.appName("Spaceship_Transportation_Analytics").getOrCreate()
spark.conf.set('spark.sql.repl.eagerEval.enabled', True)
spark
```

22/10/13 16:42:49 WARN Utils: Your hostname, adnan-System-Product-Name resolves to a loopback address: 127.0.1.1; using 192.168.1.109 instead (on interface enp2s0)  
22/10/13 16:42:49 WARN Utils: Set SPARK\_LOCAL\_IP if you need to bind to another address

Setting default log level to "WARN".  
To adjust logging level use sc.setLogLevel(newLevel). For SparkR, use setLogLevel(newLevel).

22/10/13 16:42:49 WARN NativeCodeLoader: Unable to load native-hadoop library for your platform... using builtin-java classes where applicable

Out[1]: **SparkSession - in-memory**  
**SparkContext**

[Spark UI](#)

**Version**

v3.3.0

**Master**

local[\*]

**AppName**

Spaceship\_Transportation\_Analytics

```
In [2]: df= spark.read.csv("spaceship/train.csv",header=True,inferSchema=True)
```

```
In [3]: import pyspark.pandas as ps
import numpy as np
import pandas as pd
psdf = df.toPandas().set_index("PassengerId")
psdf.drop("Name",axis=1,inplace=True)
psdf.head(5)
```

WARNING:root:'PYARROW\_IGNORE\_TIMEZONE' environment variable was not set. It is required to set this environment variable to '1' in both driver and executor sides if you use pyarrow>=2.0.0. pandas-on-Spark will set it for you but it does not work if there is a Spark context already launched.

Out[3]:

|             | HomePlanet | CryoSleep | Cabin | Destination | Age  | VIP   | RoomService | FoodCourt | ShoppingMall | Spa    | VRDeck | Transported |
|-------------|------------|-----------|-------|-------------|------|-------|-------------|-----------|--------------|--------|--------|-------------|
| PassengerId |            |           |       |             |      |       |             |           |              |        |        |             |
| 0001_01     | Europa     | False     | B/0/P | TRAPPIST-1e | 39.0 | False | 0.0         | 0.0       | 0.0          | 0.0    | 0.0    | False       |
| 0002_01     | Earth      | False     | F/0/S | TRAPPIST-1e | 24.0 | False | 109.0       | 9.0       | 25.0         | 549.0  | 44.0   | True        |
| 0003_01     | Europa     | False     | A/0/S | TRAPPIST-1e | 58.0 | True  | 43.0        | 3576.0    | 0.0          | 6715.0 | 49.0   | False       |
| 0003_02     | Europa     | False     | A/0/S | TRAPPIST-1e | 33.0 | False | 0.0         | 1283.0    | 371.0        | 3329.0 | 193.0  | False       |
| 0004_01     | Earth      | False     | F/1/S | TRAPPIST-1e | 16.0 | False | 303.0       | 70.0      | 151.0        | 565.0  | 2.0    | True        |

```
In [4]: psdf.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Index: 8693 entries, 0001_01 to 9280_02
Data columns (total 12 columns):
#   Column             Non-Null Count  Dtype
---  -
0   HomePlanet         8492 non-null   object
1   CryoSleep          8476 non-null   object
2   Cabin              8494 non-null   object
3   Destination         8511 non-null   object
4   Age                8514 non-null   float64
5   VIP                8490 non-null   object
6   RoomService        8512 non-null   float64
7   FoodCourt          8510 non-null   float64
8   ShoppingMall       8485 non-null   float64
9   Spa                8510 non-null   float64
10  VRDeck             8505 non-null   float64
11  Transported        8693 non-null   bool
dtypes: bool(1), float64(6), object(5)
memory usage: 823.5+ KB
```

```
In [5]: psdf.describe()
```

```
Out[5]:
```

|       | Age         | RoomService  | FoodCourt    | ShoppingMall | Spa          | VRDeck       |
|-------|-------------|--------------|--------------|--------------|--------------|--------------|
| count | 8514.000000 | 8512.000000  | 8510.000000  | 8485.000000  | 8510.000000  | 8505.000000  |
| mean  | 28.827930   | 224.687617   | 458.077203   | 173.729169   | 311.138778   | 304.854791   |
| std   | 14.489021   | 666.717663   | 1611.489240  | 604.696458   | 1136.705535  | 1145.717189  |
| min   | 0.000000    | 0.000000     | 0.000000     | 0.000000     | 0.000000     | 0.000000     |
| 25%   | 19.000000   | 0.000000     | 0.000000     | 0.000000     | 0.000000     | 0.000000     |
| 50%   | 27.000000   | 0.000000     | 0.000000     | 0.000000     | 0.000000     | 0.000000     |
| 75%   | 38.000000   | 47.000000    | 76.000000    | 27.000000    | 59.000000    | 46.000000    |
| max   | 79.000000   | 14327.000000 | 29813.000000 | 23492.000000 | 22408.000000 | 24133.000000 |

```
In [6]: psdf.value_counts()
```

```
Out[6]:
```

| HomePlanet | CryoSleep | Cabin    | Destination   | Age  | VIP   | RoomService | FoodCourt | ShoppingMall | Spa  | VRDeck | Transported |
|------------|-----------|----------|---------------|------|-------|-------------|-----------|--------------|------|--------|-------------|
| Europa     | True      | C/25/S   | 55 Cancr i    | 30.0 | False | 0.0         | 0.0       | 0.0          | 0.0  | 0.0    | True        |
| 2          |           |          |               |      |       |             |           |              |      |        |             |
| Mars       | True      | F/1553/P | TRAPPIST-1e   | 3.0  | False | 0.0         | 0.0       | 0.0          | 0.0  | 0.0    | True        |
| 2          |           |          |               |      |       |             |           |              |      |        |             |
| Europa     | True      | B/19/S   | 55 Cancr i    | 18.0 | False | 0.0         | 0.0       | 0.0          | 0.0  | 0.0    | True        |
| 2          |           |          |               |      |       |             |           |              |      |        |             |
| Mars       | False     | F/1050/P | TRAPPIST-1e   | 4.0  | False | 0.0         | 0.0       | 0.0          | 0.0  | 0.0    | True        |
| 2          |           |          |               |      |       |             |           |              |      |        |             |
| Earth      | True      | G/577/P  | TRAPPIST-1e   | 0.0  | False | 0.0         | 0.0       | 0.0          | 0.0  | 0.0    | True        |
| 2          |           |          |               |      |       |             |           |              |      |        |             |
| ..         |           |          |               |      |       |             |           |              |      |        |             |
|            | False     | G/556/S  | TRAPPIST-1e   | 29.0 | False | 0.0         | 4.0       | 0.0          | 0.0  | 904.0  | False       |
| 1          |           | G/556/P  | PSO J318.5-22 | 37.0 | False | 308.0       | 35.0      | 415.0        | 10.0 | 0.0    | True        |
| 1          |           | G/553/S  | PSO J318.5-22 | 26.0 | False | 6.0         | 0.0       | 400.0        | 2.0  | 373.0  | False       |
| 1          |           | G/551/S  | TRAPPIST-1e   | 74.0 | False | 0.0         | 193.0     | 581.0        | 0.0  | 0.0    | True        |
| 1          |           |          |               |      |       |             |           |              |      |        |             |
| Mars       | True      | F/999/P  | TRAPPIST-1e   | 15.0 | False | 0.0         | 0.0       | 0.0          | 0.0  | 0.0    | True        |
| 1          |           |          |               |      |       |             |           |              |      |        |             |

Length: 6755, dtype: int64

```
In [7]: np.sum(psdf.isna())
```

```
Out[7]:
```

|              |     |
|--------------|-----|
| HomePlanet   | 201 |
| CryoSleep    | 217 |
| Cabin        | 199 |
| Destination  | 182 |
| Age          | 179 |
| VIP          | 203 |
| RoomService  | 181 |
| FoodCourt    | 183 |
| ShoppingMall | 208 |
| Spa          | 183 |
| VRDeck       | 188 |
| Transported  | 0   |

dtype: int64

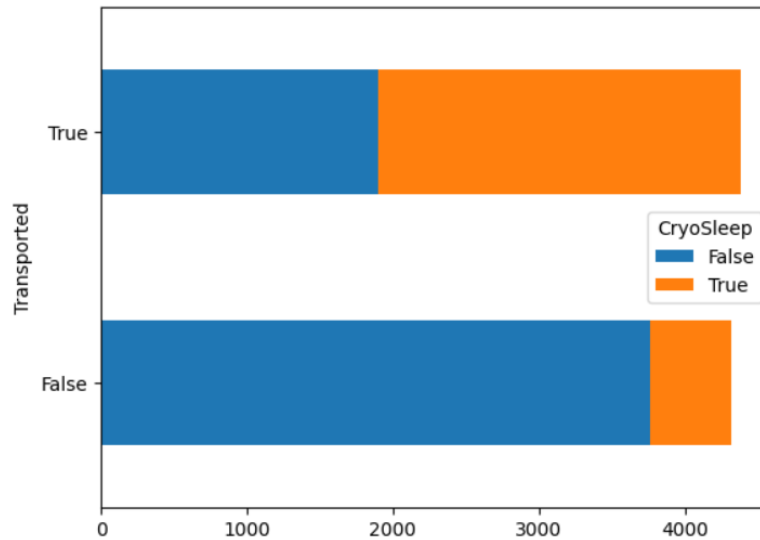
```
In [8]: psdf.HomePlanet = psdf.HomePlanet.fillna("UnkownPlanet")
np.sum(psdf.HomePlanet.isna())
```

Out[8]: 0

```
In [9]: psdf["CryoSleep"].fillna(False,inplace=True)
np.sum(psdf["CryoSleep"].isna())
```

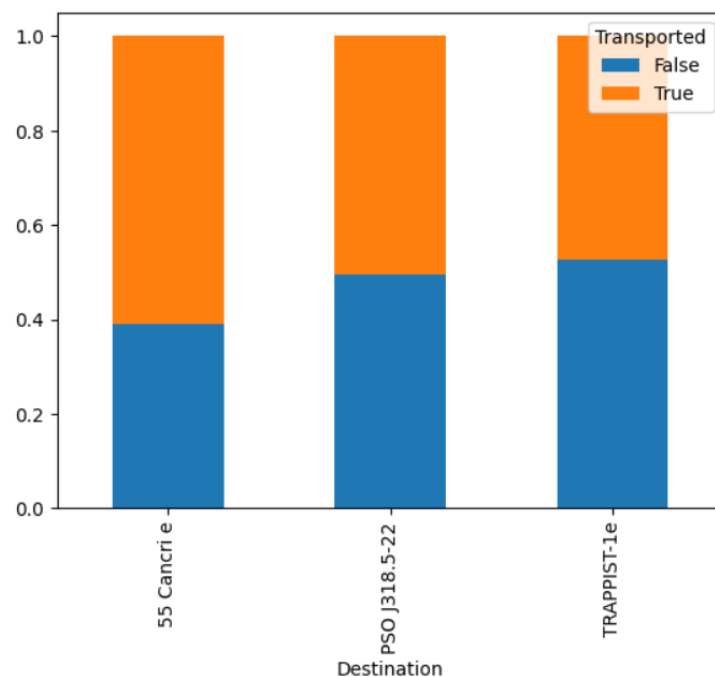
Out[9]: 0

```
In [10]: pd.crosstab(index=psdf["Transported"],columns=psdf["CryoSleep"]).plot.barh(stacked=True)
plt.show()
```

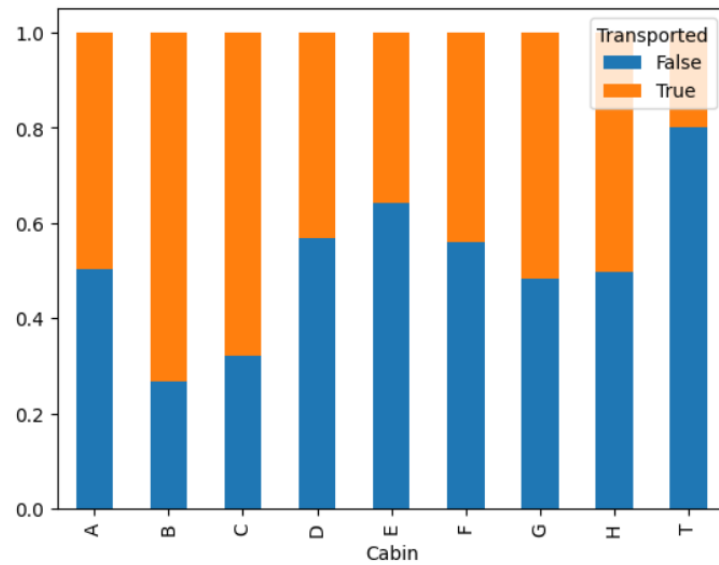


```
In [11]: psdf["Cabin"] = psdf.Cabin.map(lambda x:x[0] if x else "H")
psdf['Destination'].fillna(psdf['Destination'].mode()[0],inplace=True)
```

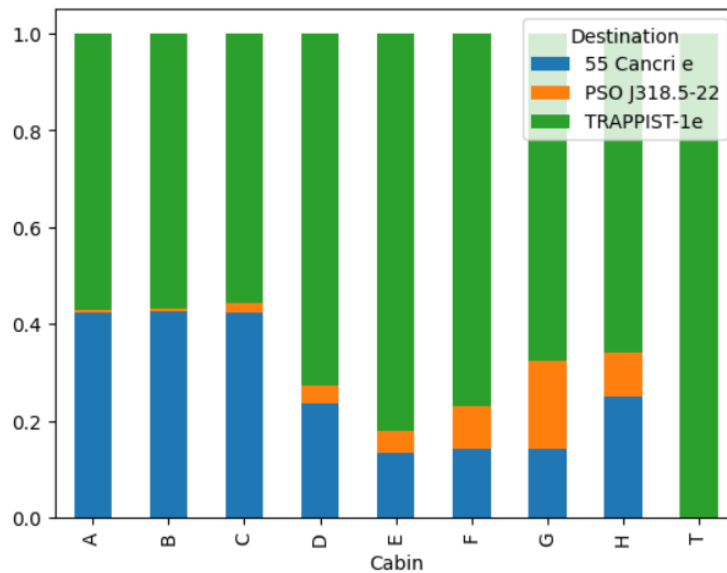
```
In [12]: temp = pd.crosstab(index=psdf["Destination"],columns=psdf["Transported"])
temp.div(np.sum(temp,axis=1),axis=0).plot.bar(stacked=True)
plt.show()
```



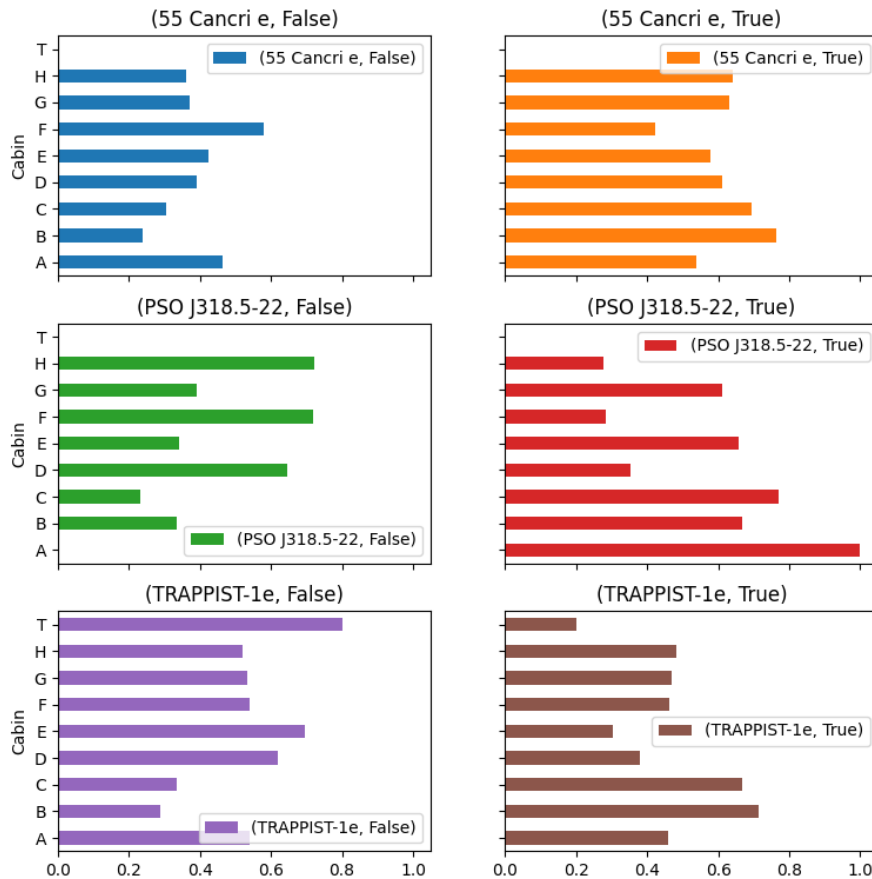
```
In [13]: temp = pd.crosstab(index=psdf["Cabin"], columns=psdf["Transported"])
temp.div(np.sum(temp,axis=1),axis=0).plot.bar(stacked=True)
plt.show()
```



```
In [14]: temp = pd.crosstab(index=psdf["Cabin"], columns=psdf["Destination"])
temp.div(np.sum(temp,axis=1),axis=0).plot.bar(stacked=True)
plt.show()
```

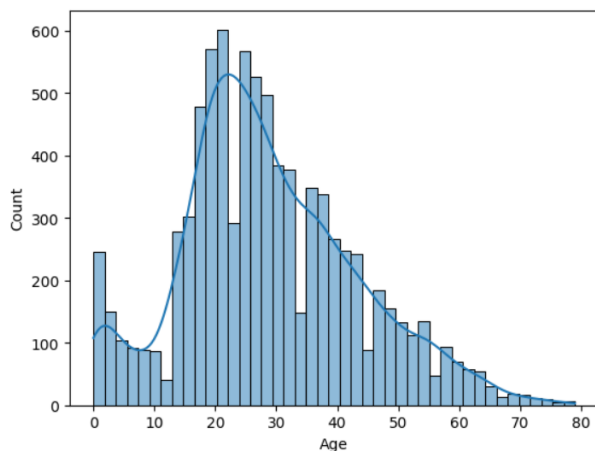


```
In [15]: temp = pd.crosstab(index=psdf["Cabin"], columns=[psdf["Transported"], psdf["Destination"]]).swaplevel(axis=1).sort_index(axis=1)
temp = temp.divide(temp.groupby(axis=1, level=0).sum(), level=0)
temp.plot(kind='barh', subplots=True, layout=(3,2), figsize=(9,9), sharey=True)
plt.show()
```



It can be Inferred from above graph Cabin B and C are more dangerous regardless of Destination and people going on Trappist less likely to select A, B and C i.e Trappist people who doesn't select Cabin A, B and C have lower chance of being teleported

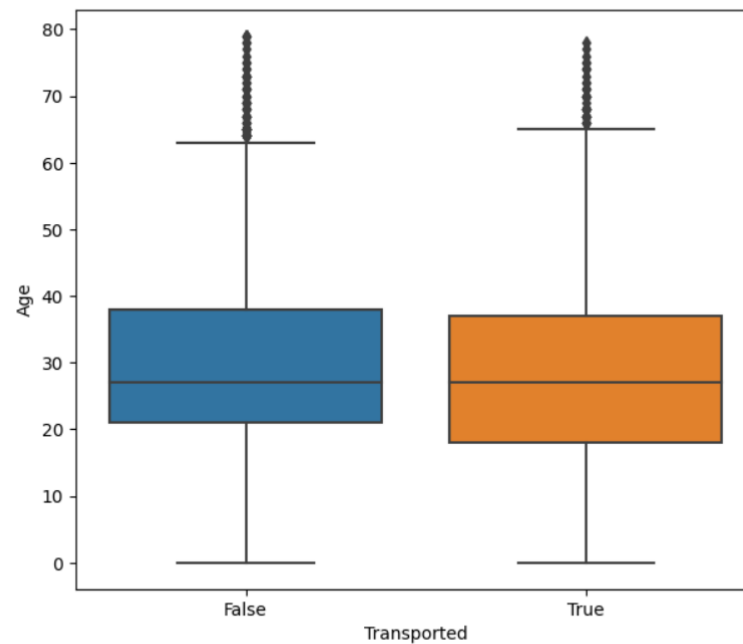
```
In [16]: sns.histplot(psdf["Age"], kde=True)
plt.show()
```



```
In [17]: psdf["Age"].fillna(psdf["Age"].median(), inplace=True)
np.sum(psdf["Age"].isna())
```

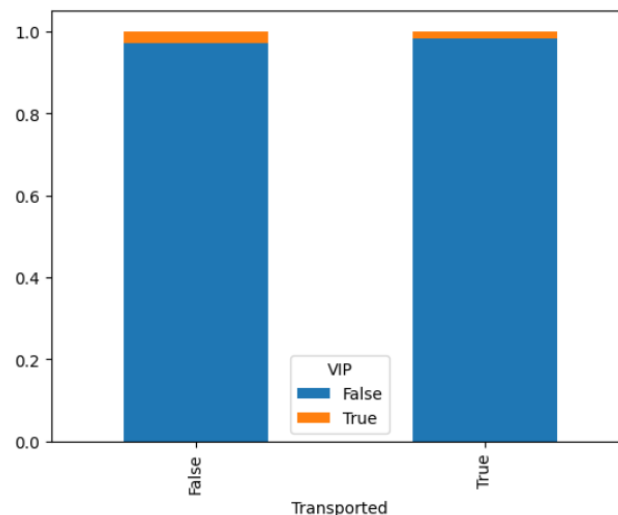
```
Out[17]: 0
```

```
In [20]: fig,ax= plt.subplots(1,1,figsize=(7,6))
sns.boxplot(psdf,x="Transported",y="Age",ax=ax)
plt.show()
```



*From above it can be seen that small kids have high chance of teleportation*

```
In [21]: psdf["VIP"].fillna(False,inplace=True)
temp = pd.crosstab(index=psdf["Transported"],columns=psdf["VIP"])
temp.div(np.sum(temp,axis=1),axis=0).plot.bar(stacked=True)
plt.show()
```



**There doesn't seems to be much relation between transported and VIP feature**

```
In [22]: temp = psdf.loc[psdf["VIP"]==True][["RoomService","FoodCourt","ShoppingMall","Spa","VRDeck"]]
temp.sum()/temp.count()
```

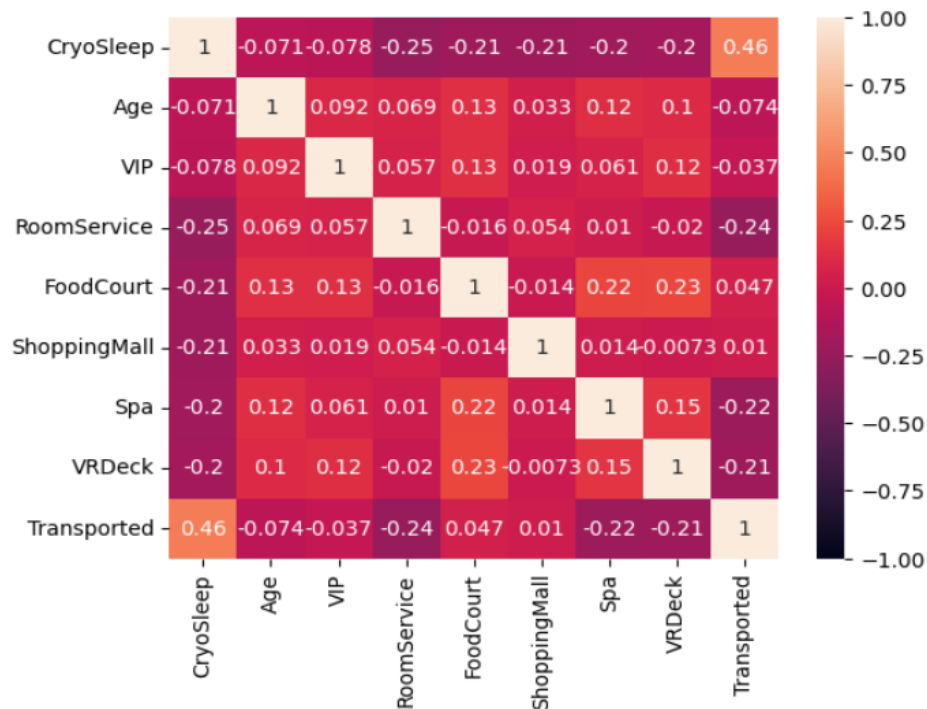
```
Out[22]: RoomService    473.615385
FoodCourt    1811.393782
ShoppingMall    247.726804
Spa           760.710660
VRDeck       1234.856410
dtype: float64
```

```
In [23]: temp = psdf.loc[psdf["VIP"]==False][["RoomService", "FoodCourt", "ShoppingMall", "Spa", "VRDeck"]]
temp.sum()/temp.count()
```

```
Out[23]: RoomService    218.851268
FoodCourt    426.672839
ShoppingMall  171.997708
Spa          300.484903
VRDeck       283.031649
dtype: float64
```

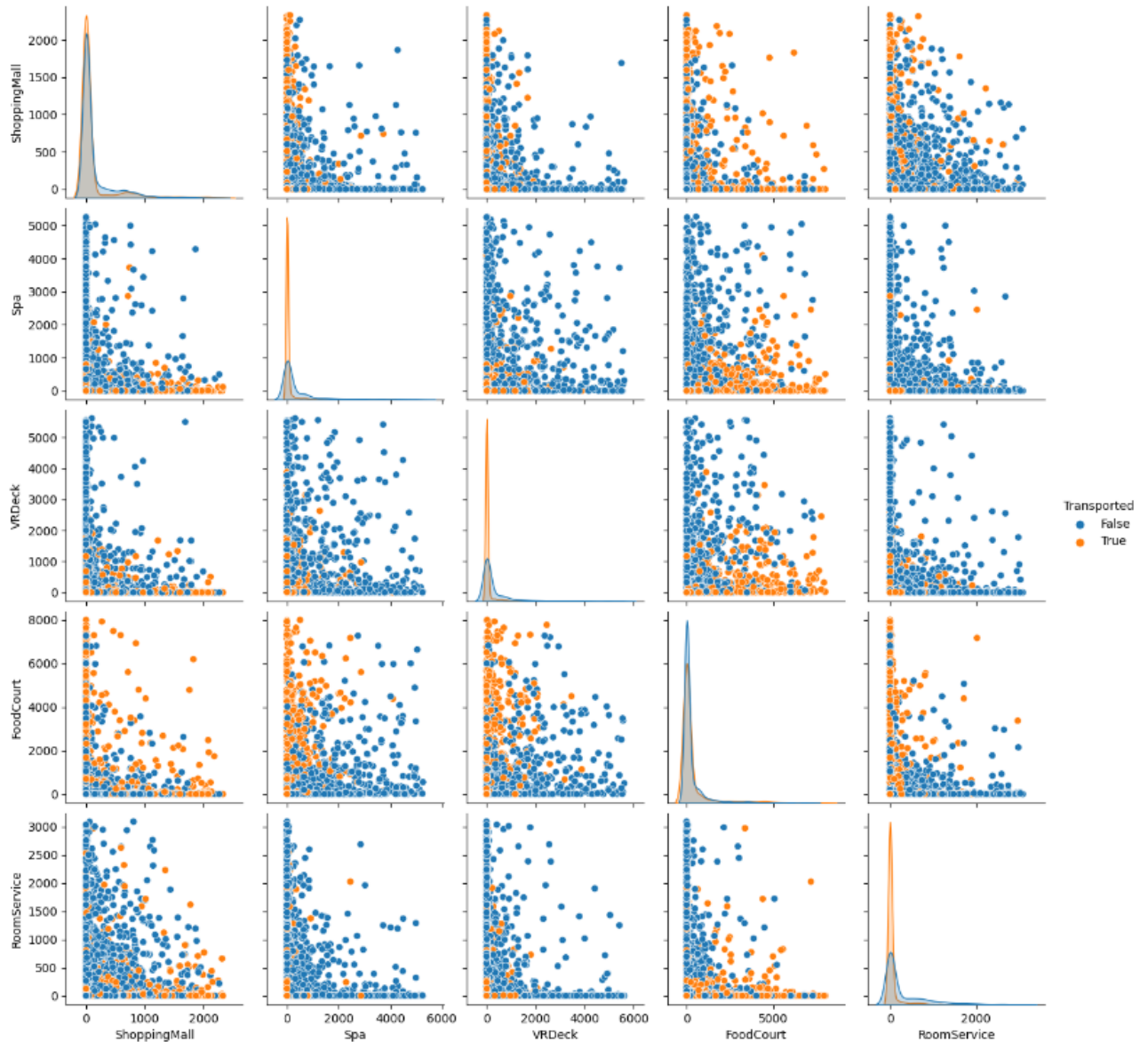
VIP people are much likely to spend on RoomService, FoodCourt, ShoppingMall, Spa and VRDeck than normal people

```
In [24]: sns.heatmap(psdf.select_dtypes(exclude="object").corr(), annot=True, vmin=-1, vmax=1)
plt.show()
```





```
In [25]: temp = psdf[["ShoppingMall", "Spa", "VRDeck", "FoodCourt", "Transported", "RoomService"]]
temp = temp.loc[
    (temp["ShoppingMall"] < temp["ShoppingMall"].quantile(0.99))
    & (temp["Spa"] < temp["Spa"].quantile(0.99))
    & (temp["VRDeck"] < temp["VRDeck"].quantile(0.99))
    & (temp["FoodCourt"] < temp["FoodCourt"].quantile(0.99))
    & (temp["RoomService"] < temp["RoomService"].quantile(0.99))
]
sns.pairplot(temp, diag_kind="kde", hue="Transported")
plt.show()
```



**Figure 3.1. Implementation and Result**

## **CHAPTER 4**

### **CONCLUSION AND FUTURE SCOPE**

Through Data pre-processing, summarization and visualization of Spaceship Titanic data we were to gained insight of related features to transported feature and how multiple features vary each other. Pyspark, pandas, numpy and seaborn came very handy for above mentioned process. From this insight we can further mixed some features to get more powerful features. This information can be used as downstream to train ML models of better accuracy than the models trained without any knowledge of data. These types of visualization can be used for different data which can come very handy for knowledge discovery.

# CHAPTER 5

## REFERENCES

- [1] <https://spark.apache.org/docs/latest/>
- [2] <https://spark.apache.org/docs/latest/api/python/>
- [3] <https://pandas.pydata.org/docs/>
- [4] <https://numpy.org/doc/stable/reference/>
- [5] <https://matplotlib.org/>
- [6] <https://seaborn.pydata.org/api.html>
- [7] <https://www.kaggle.com/competitions/spaceship-titanic>