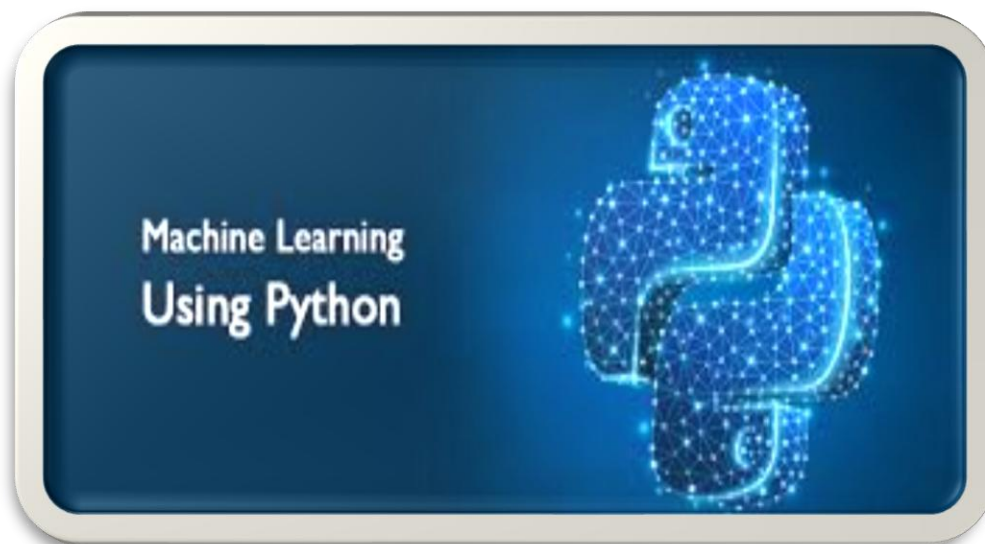


All India Eminent Faculty Council of Engineering, Management & Technology



SUBJECT - ANALYSIS OF TITANIC DATASET

SUBMITTED BY:- 1. MUNTAZIR ALAM
2. RAJANI GIRI
3. VIPIN KUMAR
4. AASTHA
5. NABANITA DAS
6. PARAMITA DAS

SUBMITTED TO:- MR.SAYANTAN
CHAKRABORTY

C O N T E N T S

- Acknowledgement
- Students Profile
- Introduction
- Objective
- Hardware & Software requirements
- Snapshots
- Coding
- Future scopes
- Advantages
- Conclusion
- Bibliography

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to the creators and maintainers of the Titanic dataset, originally made available through Kaggle. This rich and widely recognized dataset has served as an ideal resource for learning and applying machine learning and data science concepts.

We also extend our appreciation to the broader Python data science community for their invaluable tools, documentation, and resources. These contributions played a crucial role in the successful implementation of statistical models, regression analysis, and time series forecasting techniques throughout this project.

Lastly, we take this opportunity to express our profound gratitude and deep respect to our faculty mentor, **Mr. Sayantan Chakraborty**, for his exemplary guidance, consistent monitoring, and unwavering encouragement throughout the course of this project. His support and mentorship have been instrumental in shaping the direction and outcome of our work.

The knowledge, blessings, and support received from him will undoubtedly guide us through the future paths we take in life and learning.

Finally, we extend our heartfelt apologies to anyone whose contributions may have gone unnamed. We are sincerely grateful to all who supported us in various ways throughout this training journey.

Students Profile



SEMESTER – 7th

COURSE – B-TECH (CSE)

**COLLEGE – BUDGE BUDGE INSTITUTE
OF TECHNOLOGY**

INTRODUCTION

Being extremely interested in everything having a relation with the Machine Learning, the independent project was a great occasion to give us the time to learn and confirm our interest for this field. The fact that we can make estimations, predictions and give the ability for machines to learn by themselves is both powerful and limitless in term of application possibilities.

This project analyzes the Titanic passenger dataset using statistical modeling, logistic regression, and time series forecasting. The goal is to understand how factors like gender, passenger_class, age, and fare influenced survival rates. Logistic regression is used to build a predictive model for passenger survival, while stats models provides insight into the statistical significance of features.

To explore temporal trends, a simulated timeline is created and analyzed using Facebook Prophet to forecast survival rates over time. This end-to-end analysis blends classical statistics with modern machine learning and time series techniques for a comprehensive exploration of historical data.

PROJECT OBJECTIVE

- ✓ To explore and understand the structure and key variables in the Titanic dataset, including demographic and travel-related features.
- ✓ To perform data cleaning and preprocessing, including handling missing values, encoding categorical variables, and normalizing numerical data.
- ✓ To apply statistical modeling techniques (such as logistic regression using stats models) to identify significant predictors of survival and understand their impact.
- ✓ To build predictive models using machine learning algorithms, such as logistic regression (for survival classification) and linear regression (optional: predicting fare or age).
- ✓ To evaluate model performance using accuracy, confusion matrix, and classification metrics such as precision, recall, and F1-score.
- ✓ To simulate and analyze time-based trends in survival data using generated boarding dates, enabling the use of Facebook Prophet for time series forecasting.
- ✓ To visualize data insights and model results through graphs, statistical summaries, and forecast plots for better interpretation and presentation.

HARDWARE & SOFTWARE REQUIREMENTS

Hardware requirements

- ❑ **Processor:** Intel Core i3th, 5th Generation (2.4 GHz)
The Intel Core i3 processor (5th Gen) has been used due to its balance of performance and efficiency. It offers a reliable and stable working environment, suitable for data analysis and model training tasks.
- ❑ **RAM:** 4 GB
A 4 GB RAM configuration ensures smooth performance during data processing and model execution, enabling faster read/write operations and efficient memory handling.
- ❑ **Storage:** Minimum 256 GB Hard Disk
At least 256 GB of storage is recommended to accommodate the dataset, project files, software packages, and forecasting models without performance degradation.

Software requirements

- ❑ Python version should be 3.13
- ❑ Google Colab
- ❑ Operating system should be equivalent or higher version of Windows 7 and Mac or Linux also supportable.

SNAPSHOTS

```
[1] import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
```

```
[2] titanic='https://raw.githubusercontent.com/mattdeley/kaggle-titanic/master/Data/train.csv'
titanic
```

```
↳ 'https://raw.githubusercontent.com/mattdeley/kaggle-titanic/master/Data/train.csv'
```

Read datasets

```
▶ titanic=pd.read_csv('https://raw.githubusercontent.com/mattdeley/kaggle-titanic/master/Data/train.csv')
titanic
```

↳

	survived	pclass		name	sex	age	sibsp	parch		ticket	fare	cabin	embark
0	0	3		Braund, Mr. Owen Harris	male	22.0	1	0		A/5 21171	7.2500	NaN	
1	1	1	Cummings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0			PC 17599	71.2833	C85	
2	1	3	Heikkinen, Miss. Laina	female	26.0	0	0			STON/O2. 3101282	7.9250	NaN	
3	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0			113803	53.1000	C123	
4	0	3	Allen, Mr. William Henry	male	35.0	0	0			373450	8.0500	NaN	
...	
886	0	2	Montvila, Rev. Juozas	male	27.0	0	0			211536	13.0000	NaN	
887	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0			112053	30.0000	B42	
888	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2			W./C. 6607	23.4500	NaN	
889	1	1	Behr, Mr. Karl Howell	male	26.0	0	0			111369	30.0000	C148	
890	0	3	Dooley, Mr. Patrick	male	32.0	0	0			370376	7.7500	NaN	

891 rows × 11 columns

✓ [4] titanic.info()

↳

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 11 columns):
#   Column      Non-Null Count  Dtype
---  -
0   survived    891 non-null    int64
1   pclass      891 non-null    int64
2   name        891 non-null    object
3   sex         891 non-null    object
4   age         714 non-null    float64
5   sibsp       891 non-null    int64
6   parch       891 non-null    int64
7   ticket      891 non-null    object
8   fare        891 non-null    float64
9   cabin       204 non-null    object
10  embarked    889 non-null    object
dtypes: float64(2), int64(4), object(5)
memory usage: 76.7+ KB
```

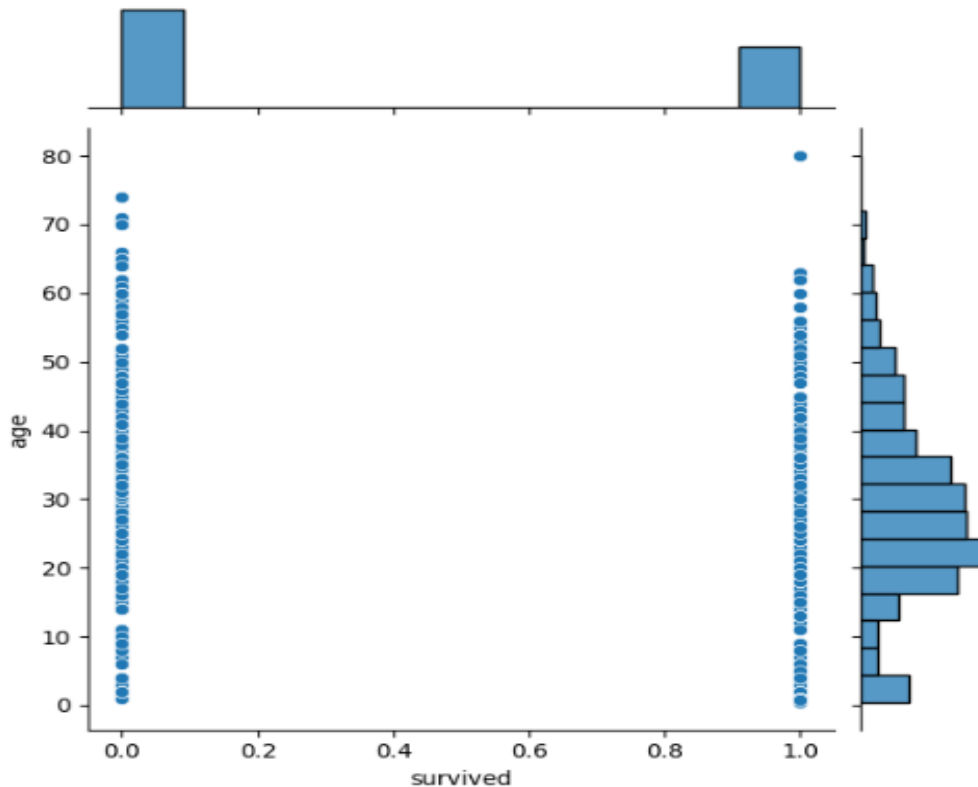

1. Statistical Modeling

(Titanic Dataset)

Statistical modeling is the process of using mathematical models to summarize and understand relationships between variables. Here are some statistical analysis of Titanic data samples:

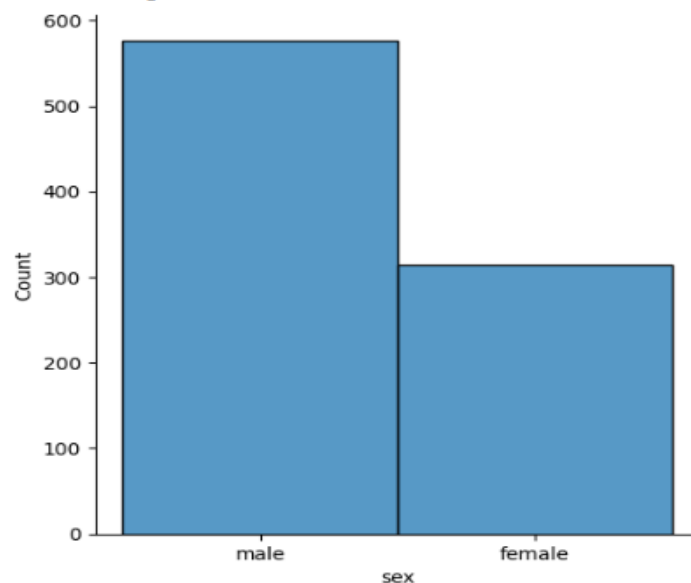
```
sns.jointplot(x='survived',y='age', data=titanic)
```

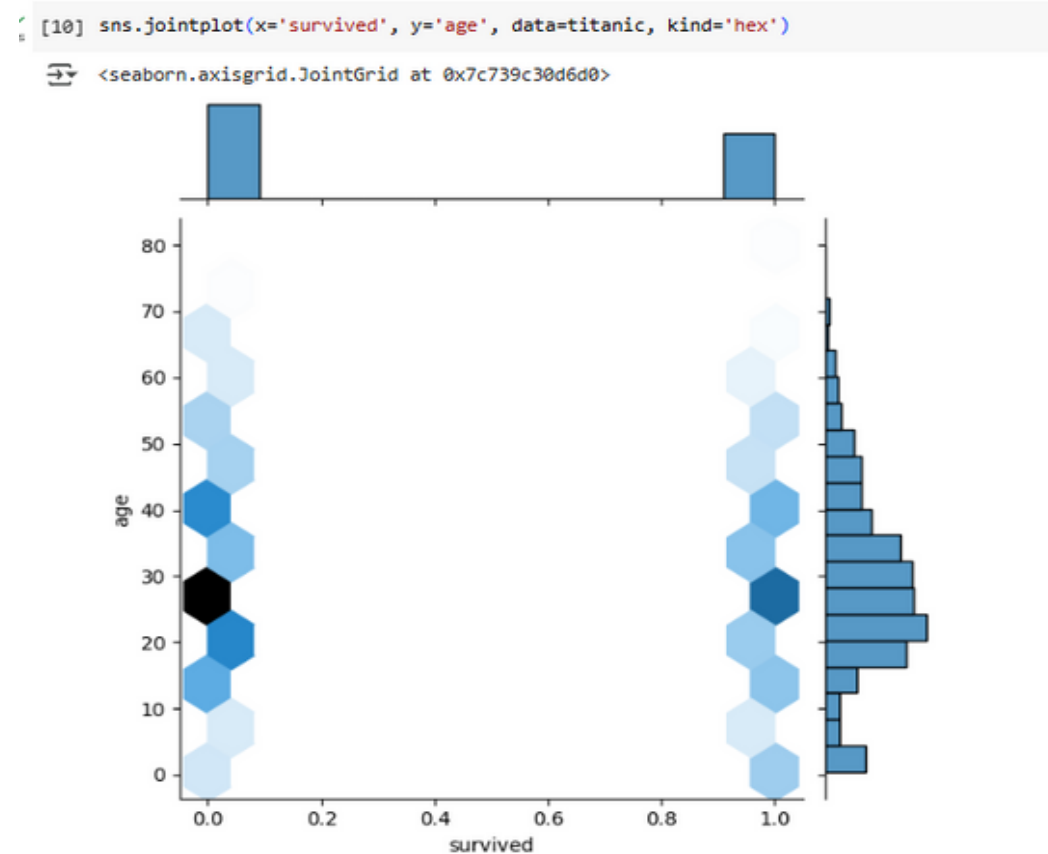
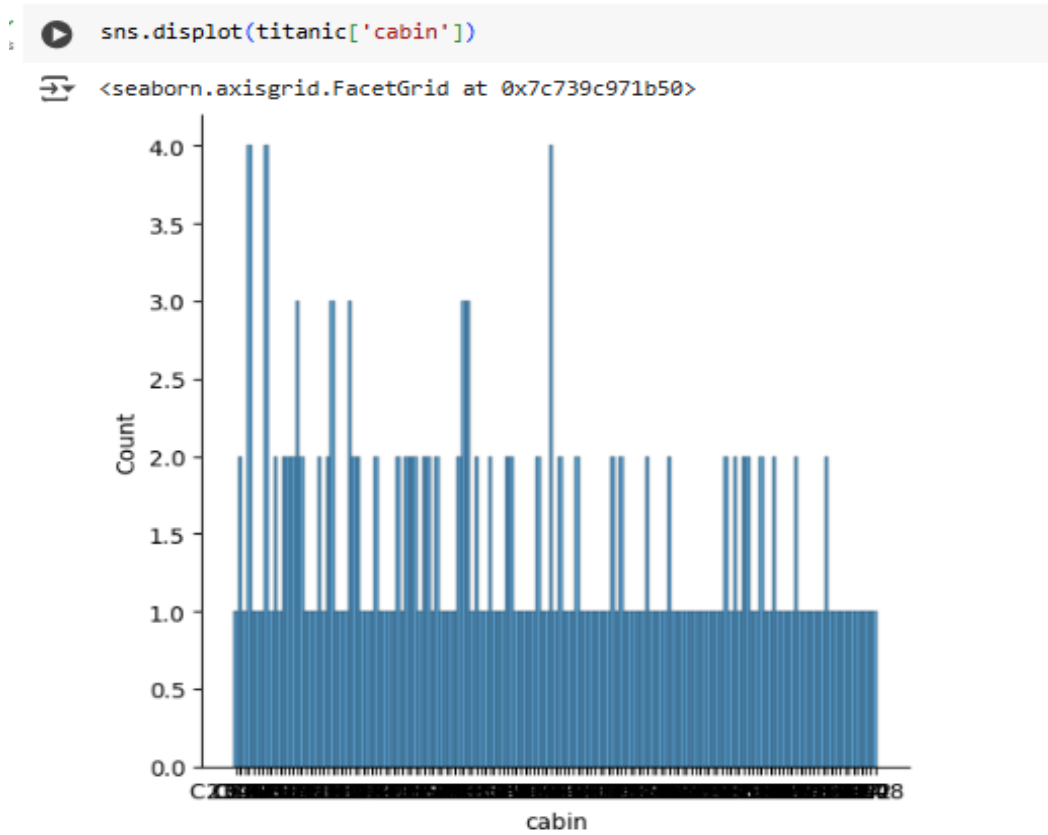
```
<seaborn.axisgrid.JointGrid at 0x7c739c952550>
```



```
sns.displot(titanic['sex'])
```

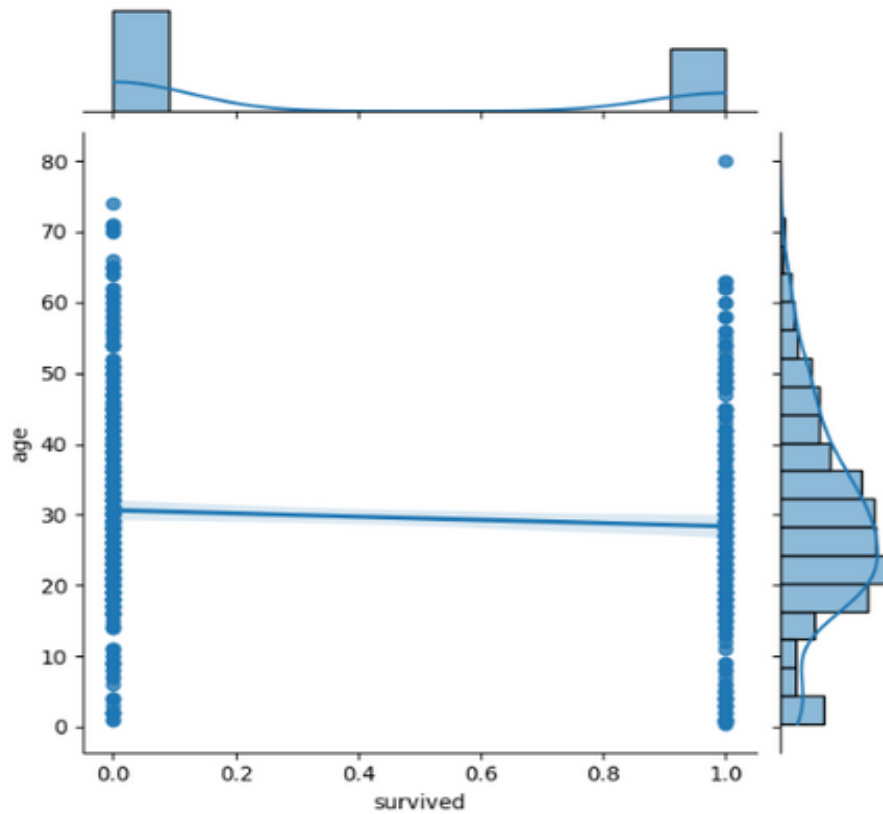
```
<seaborn.axisgrid.FacetGrid at 0x7c739c894d10>
```





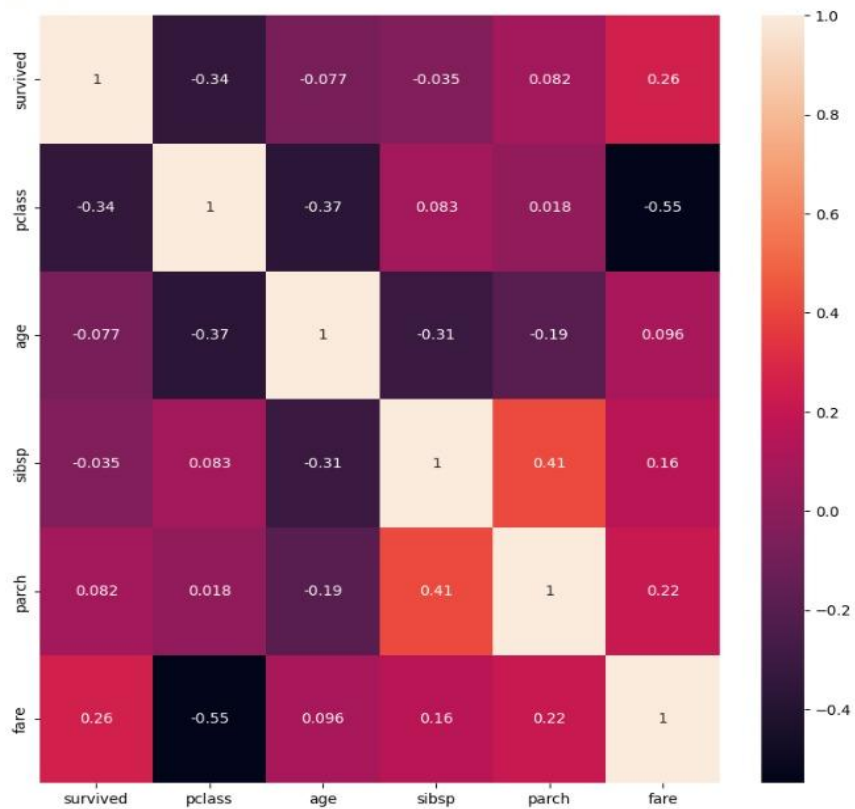
```
sns.jointplot(x='survived', y='age', data=titanic, kind='reg')
```

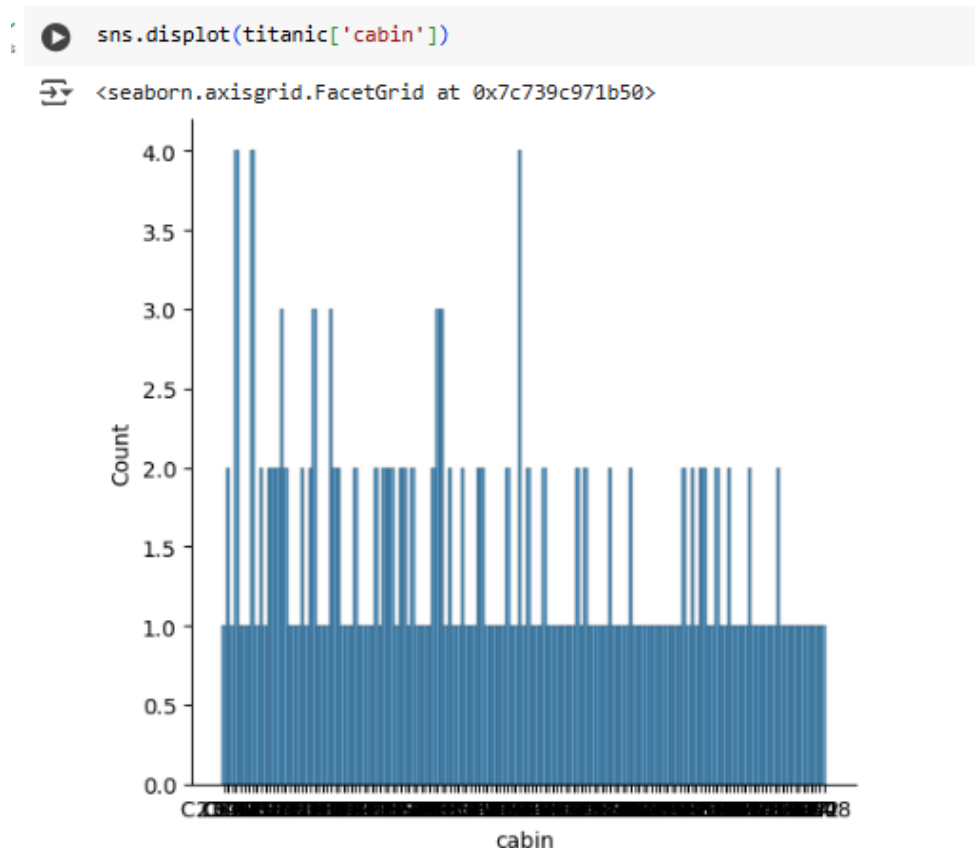
```
<seaborn.axisgrid.JointGrid at 0x7c739c442550>
```



```
plt.figure(figsize=(10,10))
sns.heatmap(s,annot=True)
```

```
<Axes: >
```





In this project, a statistical model was developed using Python in the Google Colab environment to analyze survival patterns in the Titanic dataset. We used state model to perform logistic regression, which allowed us to evaluate the relationship between key features—such as gender, passenger class, age, and fare—and the probability of survival. The model helped identify which variables were statistically significant in determining survival outcomes. Google Colab provided an efficient, cloud-based platform for running our analysis with minimal setup. The results were interpreted through summary tables, p-values, and confidence intervals, offering a solid statistical foundation for further prediction models.

2. Regression (Titanic Dataset)

Regression is a predictive modeling technique used to estimate a target variable using one or more input variables.

CODES:

```
✓ [73] # Reload the original data to ensure all columns are present
0s titanic = pd.read_csv('https://raw.githubusercontent.com/mattdelhey/kaggle-titanic/master/Data/train.csv')

# Handle missing values
titanic['age'].fillna(titanic['age'].mean(), inplace=True)
titanic['embarked'].fillna(titanic['embarked'].mode()[0], inplace=True)

# Convert 'sex' to numeric
titanic['sex'] = titanic['sex'].map({'male': 0, 'female': 1})

# One-hot encode 'embarked'
titanic_encoded = pd.get_dummies(titanic, columns=['embarked'], drop_first=True, dtype=float)

# Define features (X) and target (y)
X = titanic_encoded[['sex', 'age', 'sibsp', 'parch', 'fare', 'embarked_Q', 'embarked_S']]
y = titanic_encoded['survived']

# Split data into training and testing sets
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)

# --- Logistic Regression ---
import statsmodels.api as sm

# Add constant for statsmodels
X_train_sm = sm.add_constant(X_train)
X_test_sm = sm.add_constant(X_test)

# Initialize and train Logistic Regression Model
logit_model = sm.Logit(y_train, X_train_sm).fit()
y_pred_logit = logit_model.predict(X_test_sm)
y_pred_logit_binary = (y_pred_logit > 0.5).astype(int) # Convert probabilities to binary predictions

print("Logistic Regression Performance:")
print("Accuracy:", accuracy_score(y_test, y_pred_logit_binary))
print("Classification Report:\n", classification_report(y_test, y_pred_logit_binary))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred_logit_binary))

# --- Decision Tree Model ---
from sklearn.tree import DecisionTreeClassifier

# Initialize and train Decision Tree Model
dt_model = DecisionTreeClassifier(random_state=42)
dt_model.fit(X_train, y_train)
```

Use Logistic Regression (since Survived is binary: 0 or 1)

```
[129] from sklearn.linear_model import LogisticRegression
      from sklearn.model_selection import train_test_split
      from sklearn.preprocessing import LabelEncoder
      from sklearn.metrics import classification_report
```

```
# Drop rows with missing Age
df_clean = titanic[['survived', 'pclass', 'sex', 'age', 'fare']].dropna()
df_clean
```



	survived	pclass	sex	age	fare
0	0	3	0	22.000000	7.2500
1	1	1	1	38.000000	71.2833
2	1	3	1	26.000000	7.9250
3	1	1	1	35.000000	53.1000
4	0	3	0	35.000000	8.0500
...
886	0	2	0	27.000000	13.0000
887	1	1	1	19.000000	30.0000
888	0	3	1	29.699118	23.4500
889	1	1	0	26.000000	30.0000
890	0	3	0	32.000000	7.7500




891 rows × 5 columns

```
[16] # finding duplicate
      titanic.duplicated().sum()
```

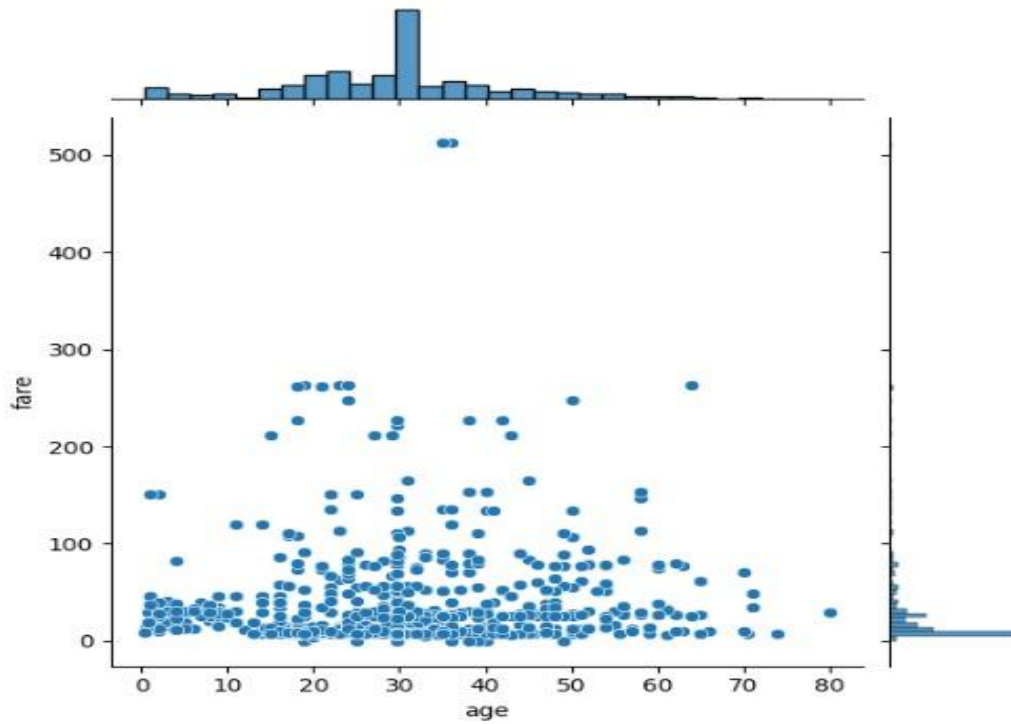
```
np.int64(112)
```

```
[17] # identifying garbage value
      for i in titanic.select_dtypes(include='object'):
          print(titanic[i].value_counts())
          print("*****" * 10)
```

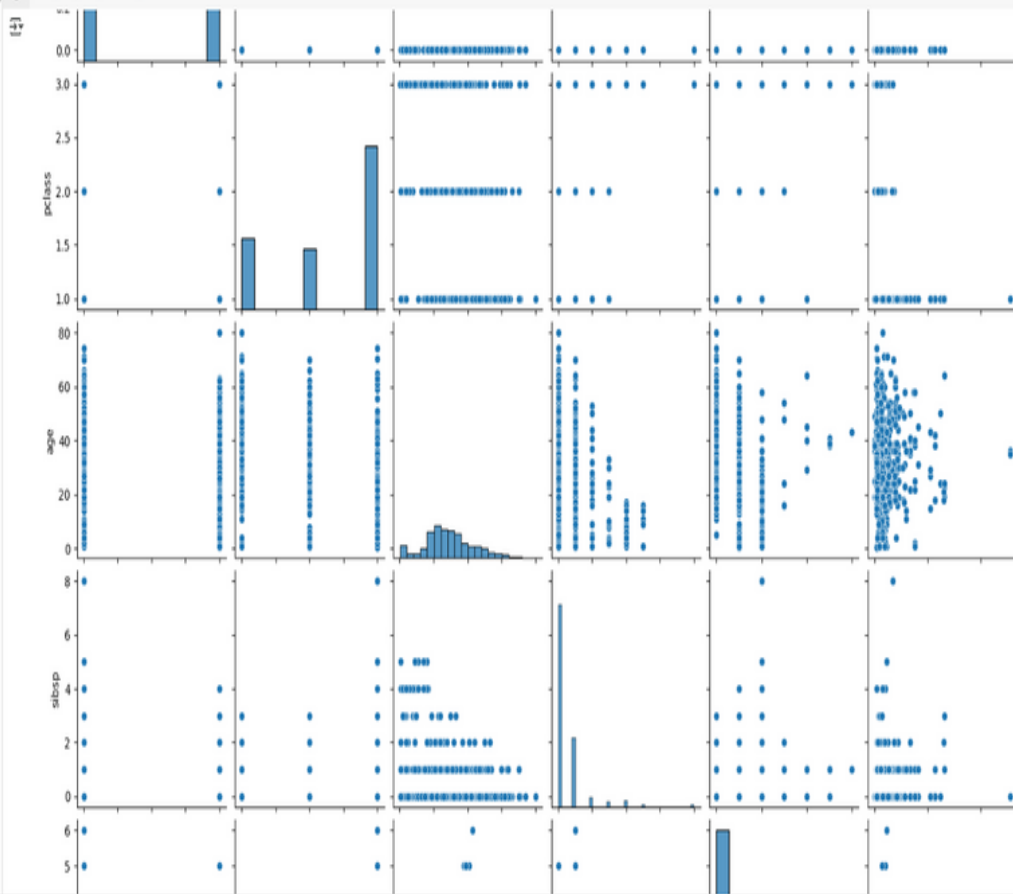


```
sex
male    577
female  314
Name: count, dtype: int64
*****
embarked
S      644
C      168
Q       77
Name: count, dtype: int64
*****
```

```
sns.jointplot(data=df_clean, x='age', y='fare')
plt.show()
```



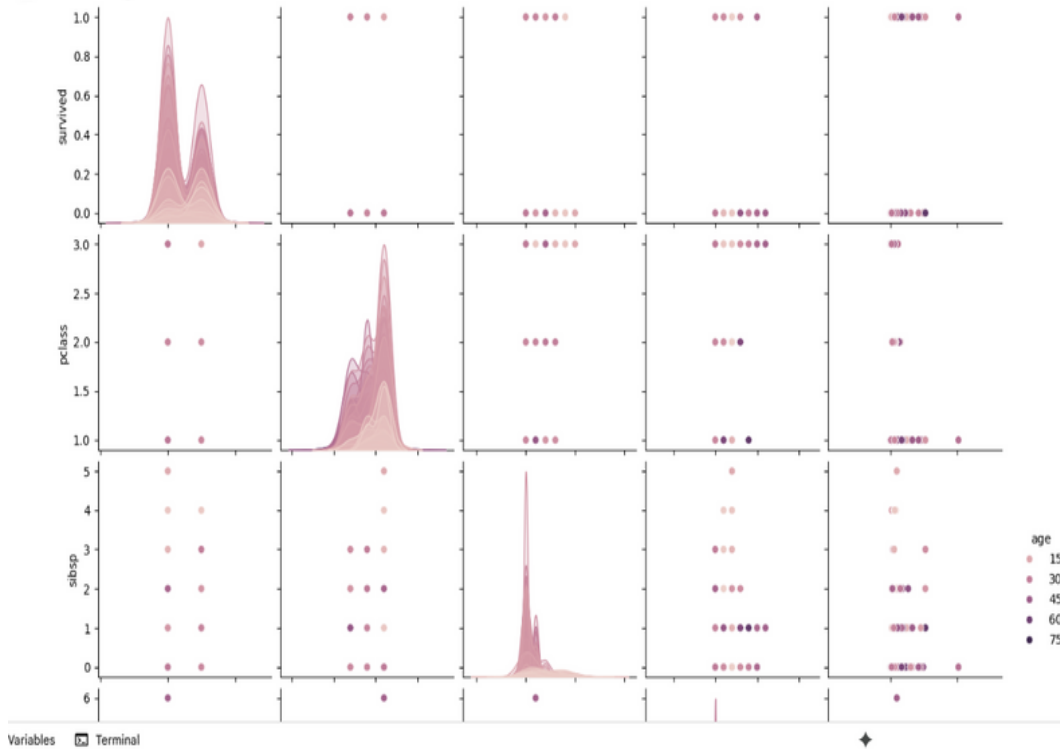
```
sns.pairplot(titanic)
```



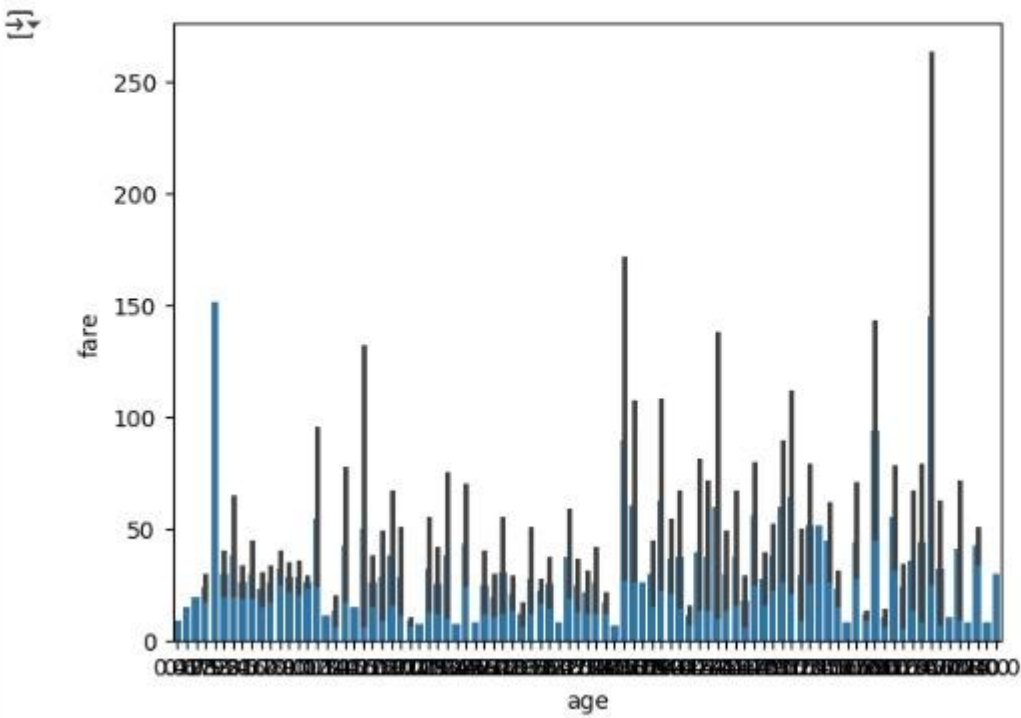
Variables Terminal


```
sns.pairplot(titanic, hue='age')
```

```
<seaborn.axisgrid.PairGrid at 0x7c739c36be10>
```



```
sns.barplot(data=df_clean, x='age', y='fare')
plt.show()
```




```
[158] sns.heatmap(df_clean.corr(), annot=True)  
plt.show()
```



Regression analysis was used to understand the influence of various features on survival. Specifically, logistic regression was applied, as the target variable—survival—is binary (survived or not). This method helped estimate the probability of survival based on predictors such as age, gender, passenger class, and fare. The regression model provided coefficients that quantify the impact of each variable, along with metrics like accuracy and confusion matrix to evaluate performance. Logistic regression proved to be a powerful tool in identifying the most influential factors in predicting survival outcomes.

3. PROPHET (TITANIC DATASET)

Prophet is an **open-source forecasting library** available in both **Python** and **R**. It is used to predict future values based on historical time series data.

CODE:

```
import pandas as pd
from prophet import Prophet
import matplotlib.pyplot as plt

url = "https://raw.githubusercontent.com/mattdehney/kaggle-titanic/master/Data/train.csv"
df = pd.read_csv(url)

df['Date'] = pd.date_range(start='1912-01-01', periods=len(df), freq='D')

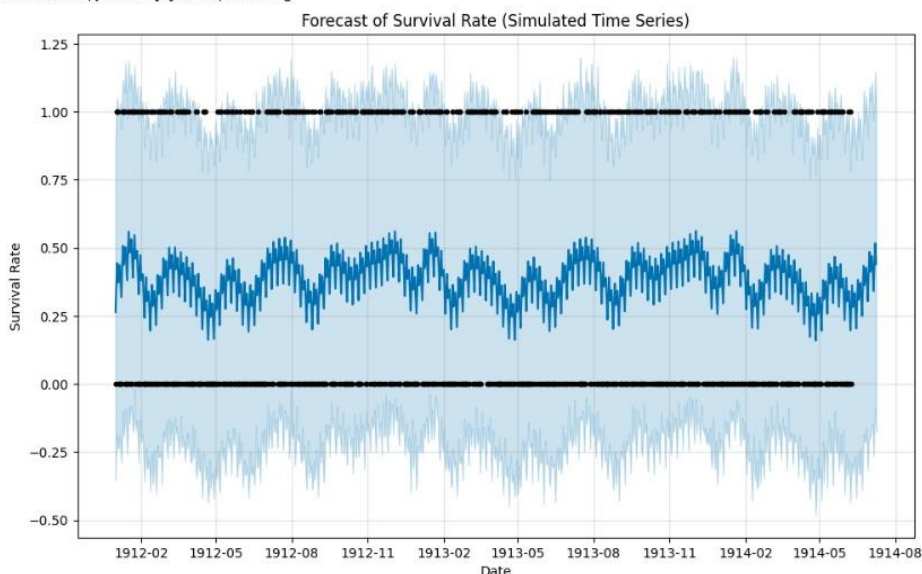
daily_survival = df.groupby('Date')['survived'].mean().reset_index()
daily_survival.columns = ['ds', 'y']

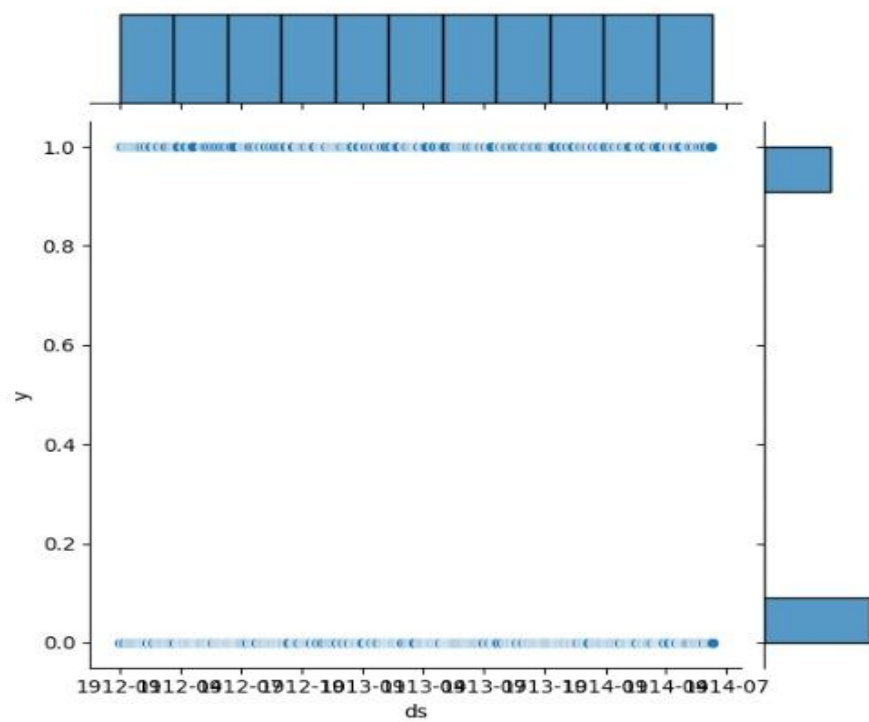
m = Prophet()
m.fit(daily_survival)

future = m.make_future_dataframe(periods=30)
forecast = m.predict(future)

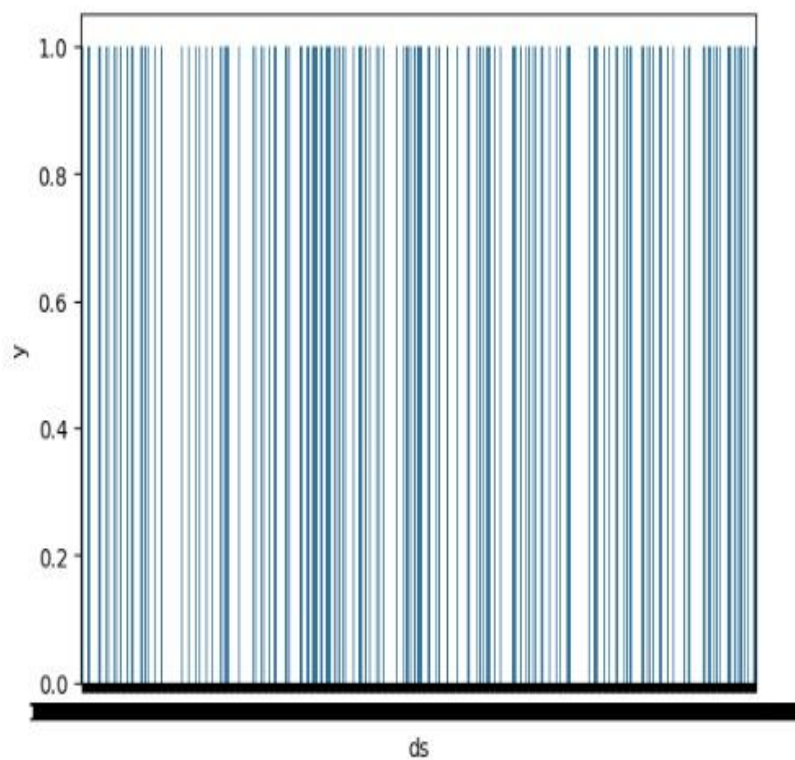
fig1 = m.plot(forecast)
plt.title("Forecast of Survival Rate (Simulated Time Series)")
plt.xlabel("Date")
plt.ylabel("Survival Rate")
plt.show()
```

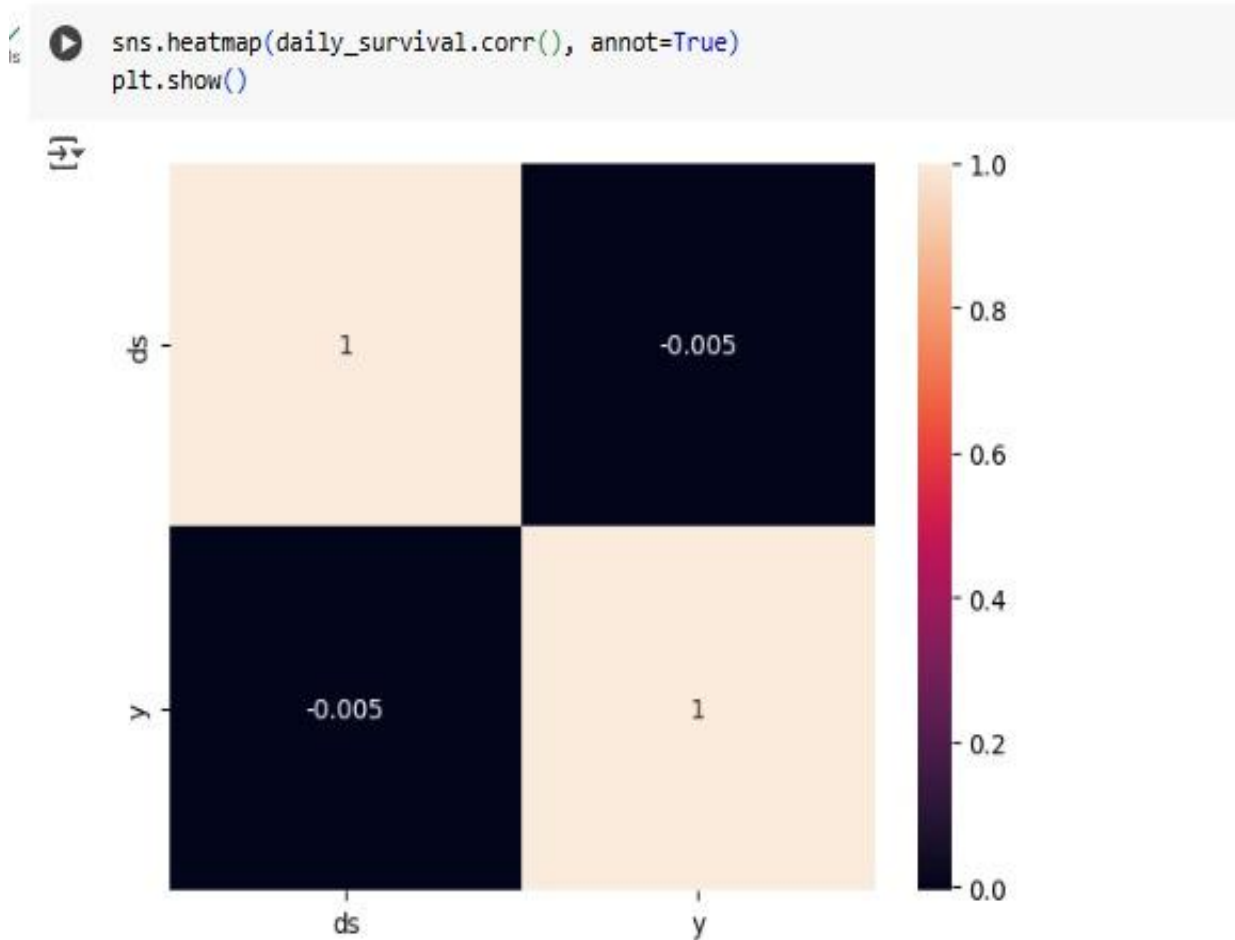
```
INFO:prophet:Disabling daily seasonality. Run prophet with daily_seasonality=True to override this.
DEBUG:cmdstanpy:input tempfile: /tmp/tmpxq838ex6/x_b_jxct.json
DEBUG:cmdstanpy:input tempfile: /tmp/tmpxq838ex6/i_or186.json
DEBUG:cmdstanpy:idx 0
DEBUG:cmdstanpy:running CmdStan, num_threads: None
DEBUG:cmdstanpy:CmdStan args: ['/usr/local/lib/python3.11/dist-packages/prophet/stan_model/prophet_model.bin', 'random', 'seed=81077', 'data', 'file=/tmp/tmpxq838ex6/x_b_jxct.json']
10:29:55 - cmdstanpy - INFO - Chain [1] start processing
INFO:cmdstanpy:Chain [1] start processing
10:29:55 - cmdstanpy - INFO - Chain [1] done processing
INFO:cmdstanpy:Chain [1] done processing
```





▶

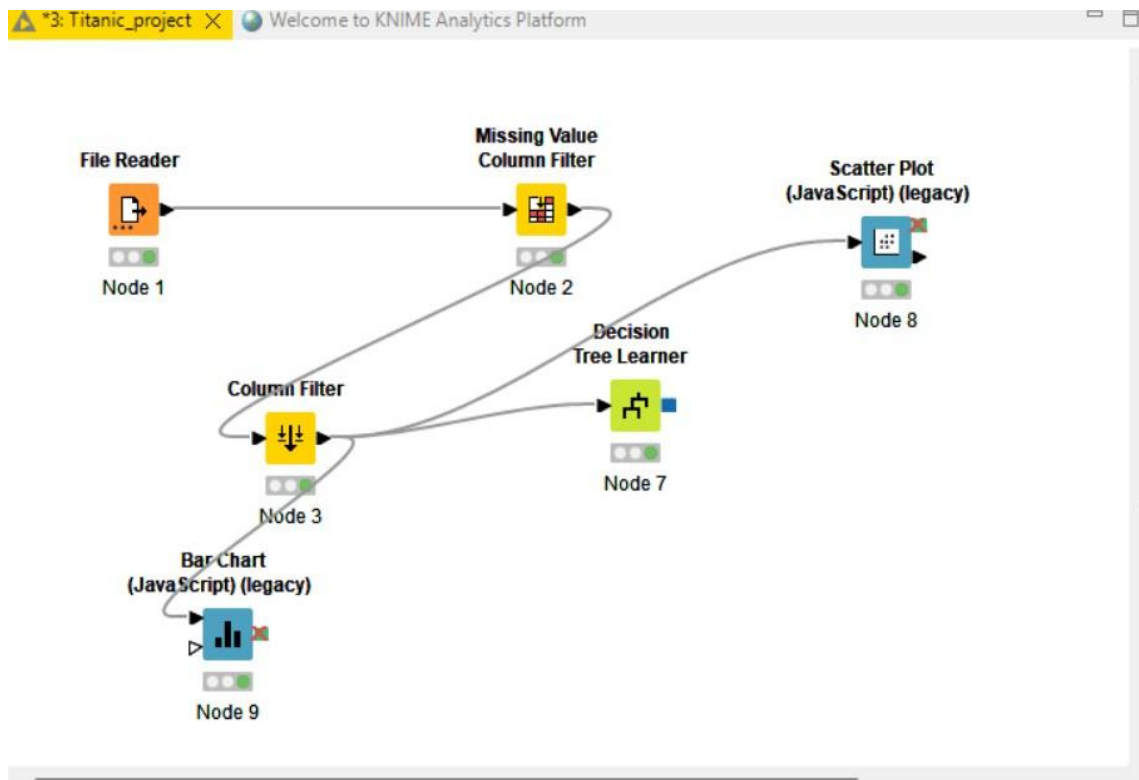




To explore time-based trends, a simulated timeline was created and analyzed using **Prophet**, a powerful open-source time series forecasting tool. Prophet was used to forecast survival rates over time, helping visualize how survival trends might have evolved under various conditions. Despite the Titanic dataset lacking a natural time component, Prophet allowed us to experiment with hypothetical scenarios and observe trend and seasonal effects. Its intuitive syntax and automatic handling of seasonality made it easy to apply in Google Colab. This analysis added a dynamic forecasting dimension to our static dataset exploration.

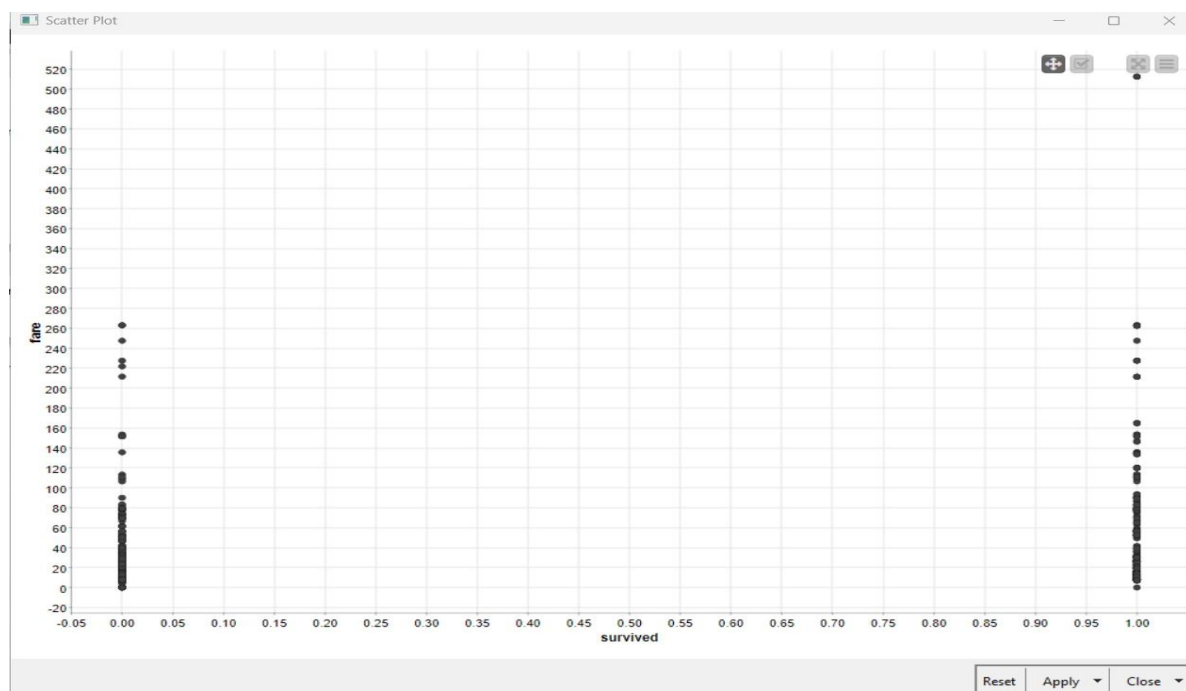
ANALYSIS OF TITANIC DATA KNIME

WORKFLOW OVERVIEW:



A KNIME workflow is a visual pipeline of data analysis tasks, where each step—like data input, preprocessing, modeling, and output—is represented by interconnected nodes.

Sunburst Chart:



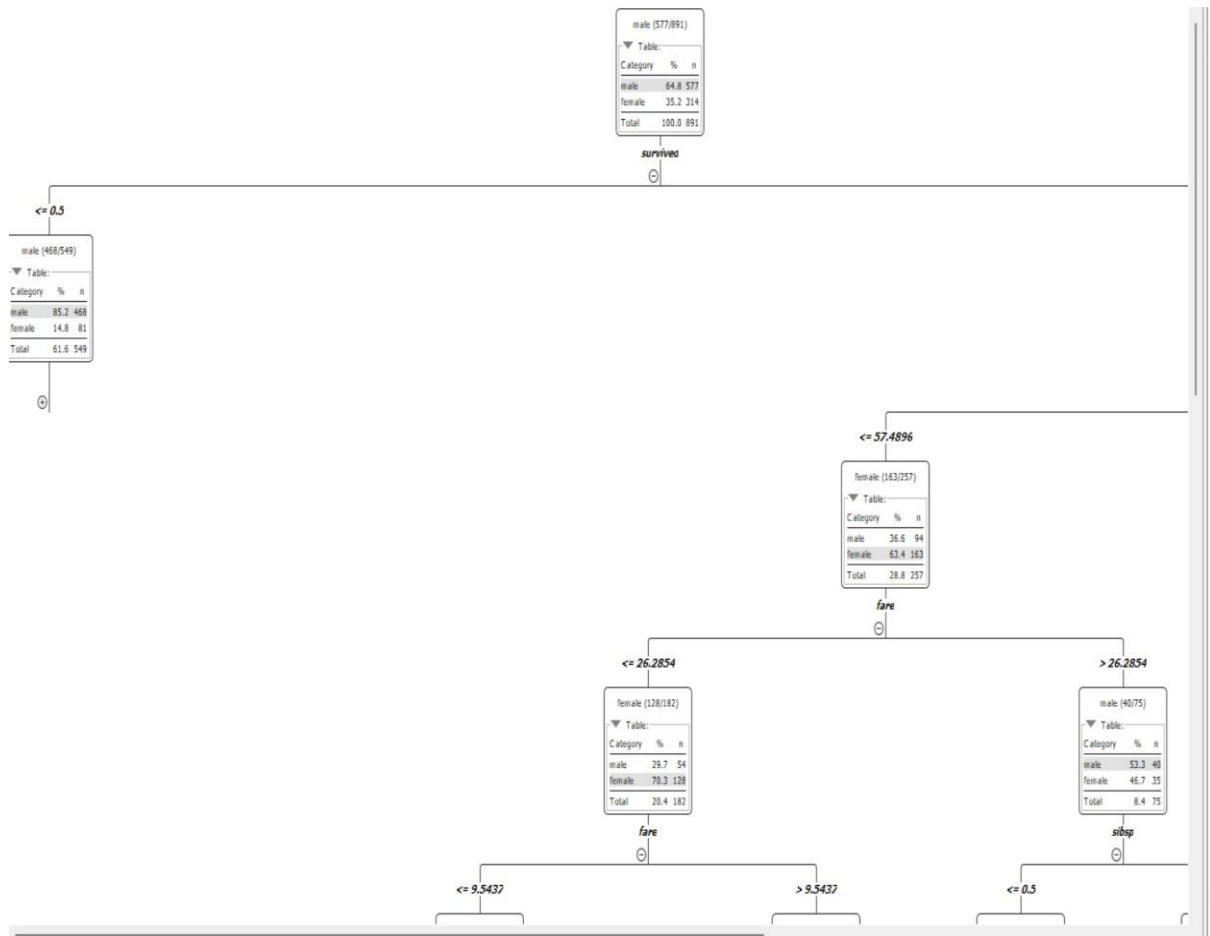
A Sunburst Chart is a circular visualization that displays hierarchical data across multiple levels using concentric rings.

TABLE VIEW:

table "default" - Rows: 891 Spec - Columns: 6 Properties Flow Variables						
Row ID	I survived	I pclass	S sex	I sibsp	I parch	D fare
Row0	0	3	male	1	0	7.25
Row1	1	1	female	1	0	71.283
Row2	1	3	female	0	0	7.925
Row3	1	1	female	1	0	53.1
Row4	0	3	male	0	0	8.05
Row5	0	3	male	0	0	8.458
Row6	0	1	male	0	0	51.862
Row7	0	3	male	3	1	21.075
Row8	1	3	female	0	2	11.133
Row9	1	2	female	1	0	30.071
Row10	1	3	female	1	1	16.7
Row11	1	1	female	0	0	26.55
Row12	0	3	male	0	0	8.05
Row13	0	3	male	1	5	31.275
Row14	0	3	female	0	0	7.854
Row15	1	2	female	0	0	16
Row16	0	3	male	4	1	29.125
Row17	1	2	male	0	0	13
Row18	0	3	female	1	0	18
Row19	1	3	female	0	0	7.225
Row20	0	2	male	0	0	26
Row21	1	2	male	0	0	13
Row22	1	3	female	0	0	8.029
Row23	1	1	male	0	0	35.5
Row24	0	3	female	3	1	21.075
Row25	1	3	female	1	5	31.387
Row26	0	3	male	0	0	7.225
Row27	0	1	male	3	2	263
Row28	1	3	female	0	0	7.879
Row29	0	3	male	0	0	7.896
Row30	0	1	male	0	0	27.721
Row31	1	1	female	1	0	146.521
Row32	1	3	female	0	0	7.75
Row33	0	2	male	0	0	10.5
Row34	0	1	male	1	0	82.171
Row35	0	1	male	1	0	52
Row36	1	3	male	0	0	7.229
Row37	0	3	male	0	0	8.05
Row38	0	3	female	2	0	18
Row39	1	3	female	1	0	11.242
Row40	0	3	female	1	0	9.475
Row41	0	2	female	1	0	21
Row42	0	3	male	0	0	7.896
Row43	1	2	female	1	2	41.579
Row44	1	3	female	0	0	7.879

The **Table View** in KNIME displays data in a spreadsheet-like format, allowing users to inspect, sort, and filter rows and columns. It helps verify data at various workflow stages for accuracy and completeness.

Decision Tree:



A **Decision Tree** is a supervised machine learning model that uses a tree-like structure to make decisions based on input features. It splits data into branches using rules, leading to predictions at the leaf nodes.

In addition to Python-based analysis, **KNIME (Konstanz Information Miner)** was explored as a no-code/low-code platform for data analytics and machine learning. KNIME's drag-and-drop interface allowed for easy data preprocessing, visualization, and model building without extensive programming. It was particularly useful for tasks like data cleaning, filtering, and running basic machine learning workflows such as logistic regression. KNIME also offered built-in nodes for data visualization and statistical analysis, making it a helpful tool for understanding the Titanic dataset from a visual and interactive perspective. Its integration with Python and R further enhanced its versatility in the data science pipeline.

FUTURE SCOPE

- Python continues to dominate as the preferred programming language for emerging technologies, particularly in the field of Machine Learning (ML). Its simplicity, readability, and vast ecosystem of libraries make it an ideal choice for both beginners and experts.
- The scope of Machine Learning with Python is extensive. Being an open-source language, it benefits from a large and active community that constantly contributes to its development and improvement. This collaborative ecosystem ensures rapid innovation and long-term support.
- With a growing developer base and increasing industry adoption, Python's role in the future of data science and artificial intelligence is only expected to strengthen

ADVANTAGES

- Python is highly effective in the development of prototypes, allowing developers to accelerate the transition from concept to creation. Its clean and readable syntax significantly reduces development time and enhances maintainability.
- In addition to Machine Learning, Python is ideal for a wide range of general-purpose tasks, including data mining, automation, and big data processing. Its extensive library support and integration capabilities make it a powerful tool for handling complex data workflows efficiently.

CONCLUSION

Python is easy, simple, powerful, and innovative due to its broader usage in a variety of contexts, some of which are not associated with data science. R is an optimized environment for data analysis, but it is difficult to learn.

It's only one way to shape the debate: considering it as a zero-sum game. The fact is that understanding both tools and utilizing them as per their respective strengths can refine you as a data scientist. Every data scientist should be versatile and should stay at the top of their game.

Data science experts expect this trend to continue with increasing development in the Python ecosystem. And while your journey to learn Python programming may be just beginning, it's nice to know that employment opportunities are abundant as well.

So, the future is bright for data science topics like Machine Learning, Deep Learning etc. and Python is just one piece of the proverbial pie. Fortunately, learning Python and other programming fundamentals is as attainable as ever.

BIBLIOGRAPHY

The contents have been gathered from following:

- ☐ Datasets: Kaggle
- ☐ Information: Google
- ☐ Coding and snapshots: Self-Performed

■

THANK YOU
