DATA PREPARATION
WHITESPACES

Write a topic or a highlight here.

OUR TEAM :

Andhika Utomo

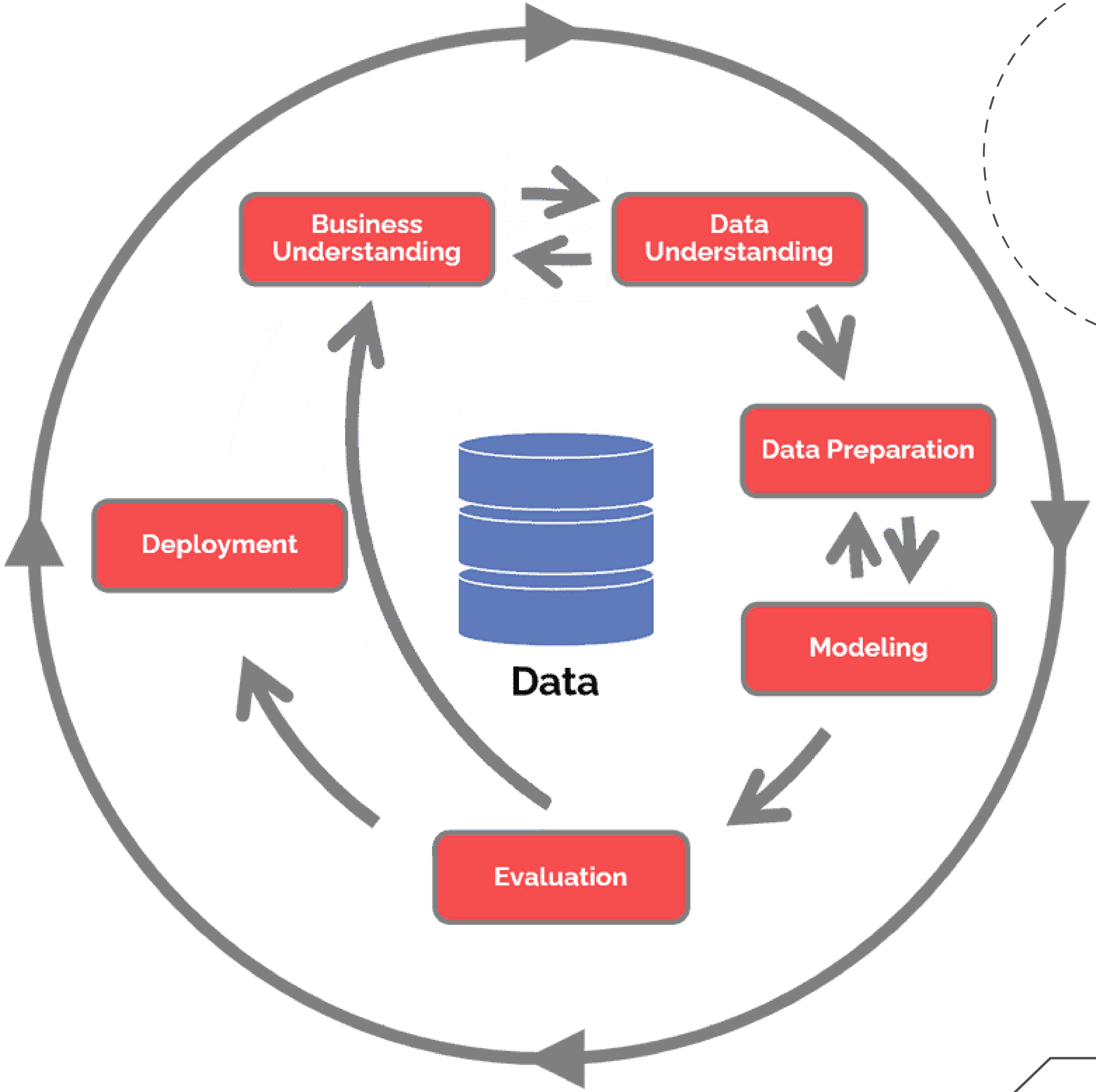
Dean Setyawan

Heru Stiawan

M Haikal Febrian p

Raisya Amanah N

CRISP DM


Business Understanding
Data Understanding
Data Preparation
Modeling
Evaluation
Deployment
Data

BUSSINESS UNDERSTANDING

Understand the project objectives and requirements from a business perspective, and then convert this knowledge into a data mining problem definition and a preliminary plan designed to achieve the objectives.

use case:

01 data mining

with business understanding we can mining only necessary data to process so not waste time mining data that not used

02 data processing

data can be process to make an insight to expand the company and linear toward company business

DATA PREPARATION

library for data preparation

Pandas

pandas is a software library written for the Python programming language for data manipulation and analysis

numpy

numpy is a Python library used for working with arrays

Pandas

matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python

seaborn

seaborn Seaborn is a Python data visualization library based on matplotlib

DATA CLEANSING

Data cleansing or data cleaning is the process of identifying and correcting corrupt, incomplete, duplicated, incorrect, and irrelevant data from a reference set, table, or database.

basic step of data cleansing

01 find null value in data set using info() method

02 fill that null value if not too much null value otherwie can removed that column

basic method for handling missing value

There is 2 basic handling null value such as

- case deletion
- filling missing value using mean, median or modus

need to note that this isn't only way to handle missing value but there is many more such as regression method, K-Nearest Neighbour Imputation (KNN) and many other

case deletion

case deletion is method to deleted one column from dataset. this method only use if missing value in that variable is too much to avoid any artificial increase in relationships with independent variables.

example case deletion using titanic dataset

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
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: 83.5+ KB
```

missing value

```
df.isnull().sum()
```

```
Survived      0
Pclass        0
Name          0
Sex           0
Age         177
SibSp         0
Parch         0
Ticket        0
Fare          0
Cabin       687
Embarked      2
dtype: int64
```

we can see that cabin column have too much missing value so we can drop cabin column

code program

```
df.drop("Cabin", axis=1,inplace=True)
```

output

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
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: 83.5+ KB
```

after

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 10 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   Embarked  889 non-null    object
dtypes: float64(2), int64(4), object(4)
memory usage: 76.6+ KB
```

imputation using mean/median/modus

If the missing values in a column or feature are numerical, the values can be imputed by the mean of the complete cases of the variable. Mean can be replaced by median if the feature is suspected to have outliers. For a categorical feature, the missing values could be replaced by the mode of the column. The major drawback of this method is that it reduces the variance of the imputed variables. This method also reduces the correlation between the imputed variables and other variables because the imputed values are just estimates and will not be related to other values inherently.

example imputation using modus in titanic dataset

code program

```
val=df.Embarked.mode().values[0]
df.Embarked=df["Embarked"].fillna(val)
```

output

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          891 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Embarked     889 non-null    object
dtypes: float64(2), int64(5), object(4)
memory usage: 76.7+ KB
```

after

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 11 columns):
 #   Column       Non-Null Count  Dtype
---  ------       --------------  -----
 0   PassengerId  891 non-null    int64
 1   Survived     891 non-null    int64
 2   Pclass       891 non-null    int64
 3   Name         891 non-null    object
 4   Sex          891 non-null    object
 5   Age          891 non-null    float64
 6   SibSp        891 non-null    int64
 7   Parch        891 non-null    int64
 8   Ticket       891 non-null    object
 9   Fare         891 non-null    float64
 10  Embarked     891 non-null    object
dtypes: float64(2), int64(5), object(4)
memory usage: 76.7+ KB
```

example imputation using median in titanic dataset

code program

```
val=df.Age.median()
df["Age"]=df.Age.fillna(val)
```

output

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 10 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   Embarked  891 non-null    int64
dtypes: float64(2), int64(5), object(3)
memory usage: 76.6+ KB
```

after

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 10 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       891 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   Embarked  891 non-null    int64
dtypes: float64(2), int64(5), object(3)
memory usage: 76.6+ KB
```

EXPLANATORY DATA ANALYSIS (EDA)

EDA is applied to investigate the data and summarize the key insights. It will give you the basic understanding of your data, it's distribution, null values and much more. You can either explore data using graphs or through some python functions.

basic step of EDA

01 cek every column and if column is not important or not helping insight. removed that column from dataset

02 fix oulier or anomali data set using graph then using coraltion between variable to make a graph insight

Example Sex Column in titanic dataset

to know how many kind is unique data can use command below.

```
df.Sex.nunique()
```

```
2
```

and if want to describe detail what is unique data aviable in data set can use command below.

```
df.Sex.unique()
```

```
array(['male', 'female'], dtype=object)
```

```
df.Sex.value_counts()
```

```
male      577
female    314
Name: Sex, dtype: int64
```

to kow how many row and column from data set can use command below.

```
df.shape
```

```
(891, 11)
```

for checking duplicated in data set we can use command below and if we want to removed in we can use drop it using drop_duplicated() method.

```
df[df.duplicated()]
```

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
PassengerId											

```
df.drop_duplicates()
```

	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
PassengerId											
1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S
...	...	...	...	...	...	...	...	...	...	...	...
887	0	2	Montvila, Rev. Juozas	male	27.0	0	0	211536	13.0000	NaN	S
888	1	1	Graham, Miss. Margaret Edith	female	19.0	0	0	112053	30.0000	B42	S
889	0	3	Johnston, Miss. Catherine Helen "Carrie"	female	NaN	1	2	W./C. 6607	23.4500	NaN	S
890	1	1	Behr, Mr. Karl Howell	male	26.0	0	0	111369	30.0000	C148	C
891	0	3	Dooley, Mr. Patrick	male	32.0	0	0	370376	7.7500	NaN	Q

891 rows × 11 columns

Extra in example sex coloumn

Sex Column

we can make data inside these into number using dictionary so if we make machine learning data is ready to use.

Example Embarked Column in titanic dataset

Programing Code and Output :

```
df.Sex=df.Sex.map({"male":0,"female":1})
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 9 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  891 non-null    int64
 1   Pclass    891 non-null    int64
 2   Sex       891 non-null    int64
 3   Age       891 non-null    float64
 4   SibSp     891 non-null    int64
 5   Parch     891 non-null    int64
 6   Ticket    891 non-null    object
 7   Fare      891 non-null    float64
 8   Embarked  891 non-null    int64
dtypes: float64(2), int64(6), object(1)
memory usage: 69.6+ KB
```

Programing Code and Output :

```
df[df.Embarked.isnull()]
```

PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
62	1	1	Icard, Miss. Amelie	female	38.0	0	0	113572	80.0	B28	NaN
830	1	1	Stone, Mrs. George Nelson (Martha Evelyn)	female	62.0	0	0	113572	80.0	B28	NaN

```
df.Embarked.value_counts()
```

```
S    644
C    168
Q     77
Name: Embarked, dtype: int64
```

Programing Code and Output :

```
val=df.Embarked.mode().values[0]
df["Embarked"]=df.Embarked.fillna(val)
```

```
df.Embarked.value_counts()
```

```
S    646
C    168
Q     77
Name: Embarked, dtype: int64
```

Programing Code and Output :

```
df.Embarked=df.Embarked.map({"S":0,"C":1,"Q":2})
```

```
df.info()
```

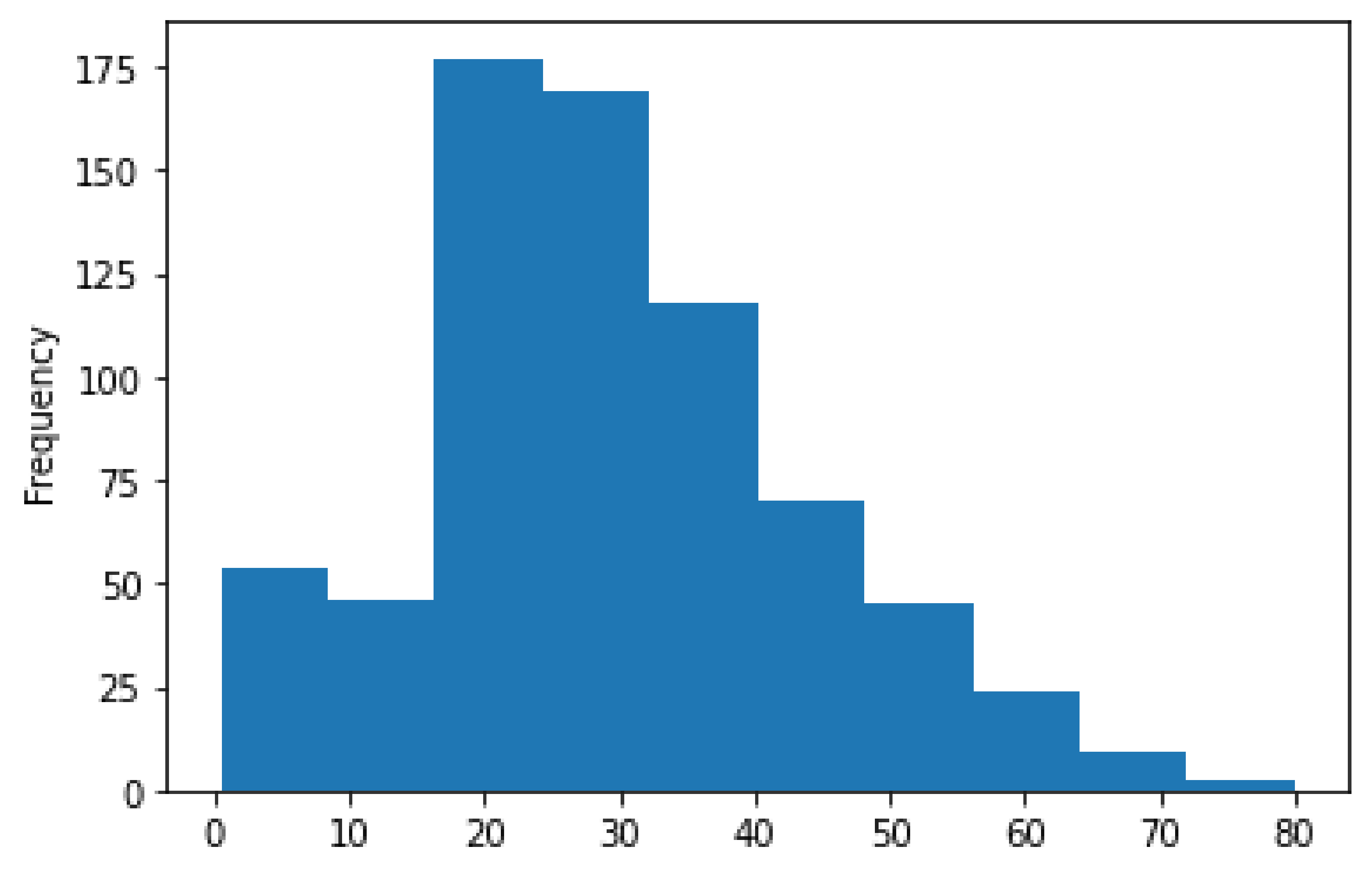
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 11 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  891 non-null    int64
 1   Pclass    891 non-null    int64
 2   Name      891 non-null    object
 3   Sex       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  891 non-null    int64
dtypes: float64(2), int64(5), object(4)
memory usage: 83.5+ KB
```

Example Age Column in titanic dataset

Programing Code and Output :

```
df.Age.plot(kind="hist")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f39312768d0>
```

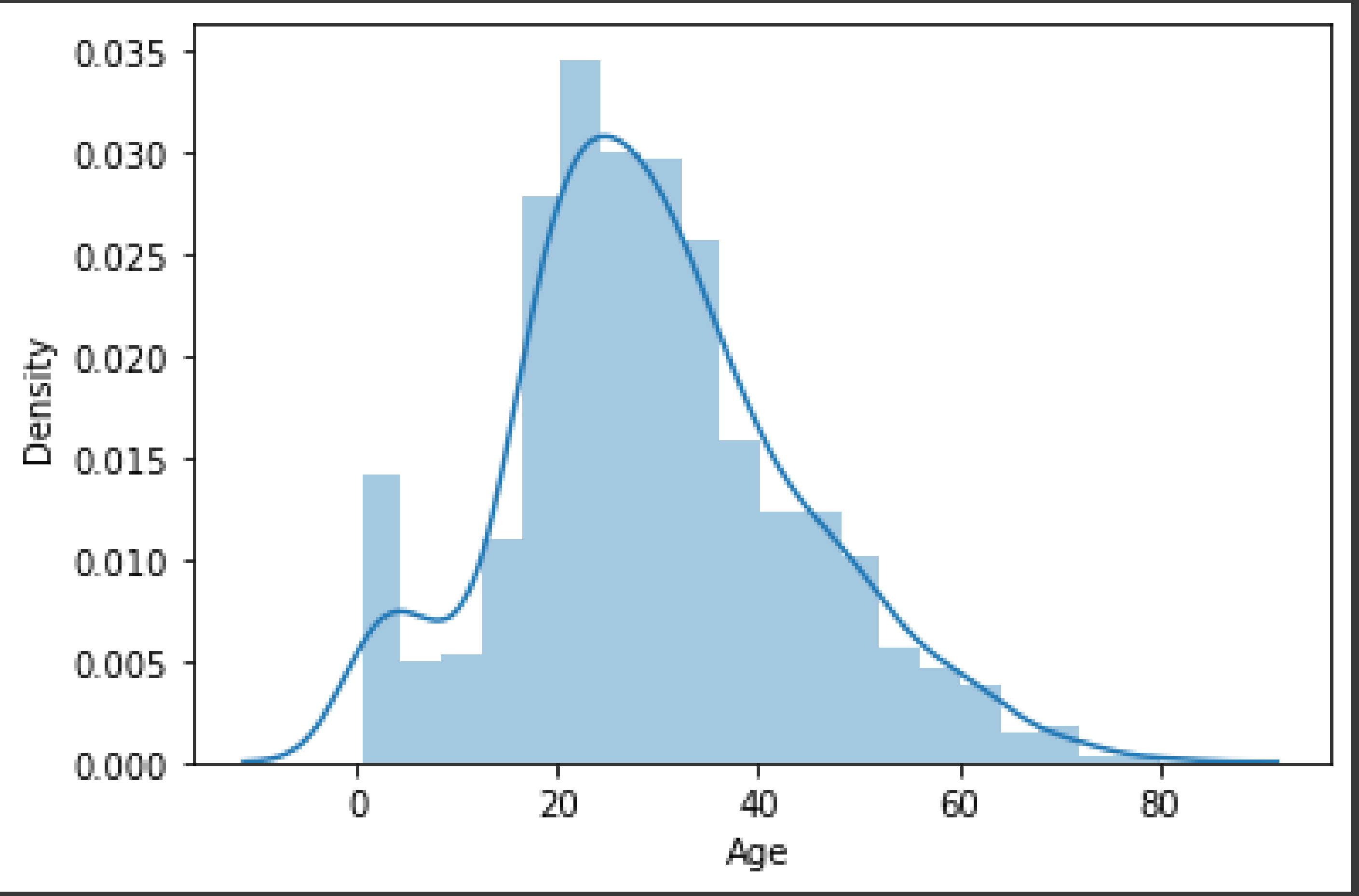


Programing Code and Output :

```
[25] import seaborn as sns

     sns.distplot(df["Age"])

     /usr/local/lib/python3.7/dist-packages/seaborn/distributio
       warnings.warn(msg, FutureWarning)
     <matplotlib.axes._subplots.AxesSubplot at 0x7f3914a531d0>
```



Programing Code and Output :

```
val=df.Age.median()
df["Age"]=df.Age.fillna(val)
```

```
df.info()
```

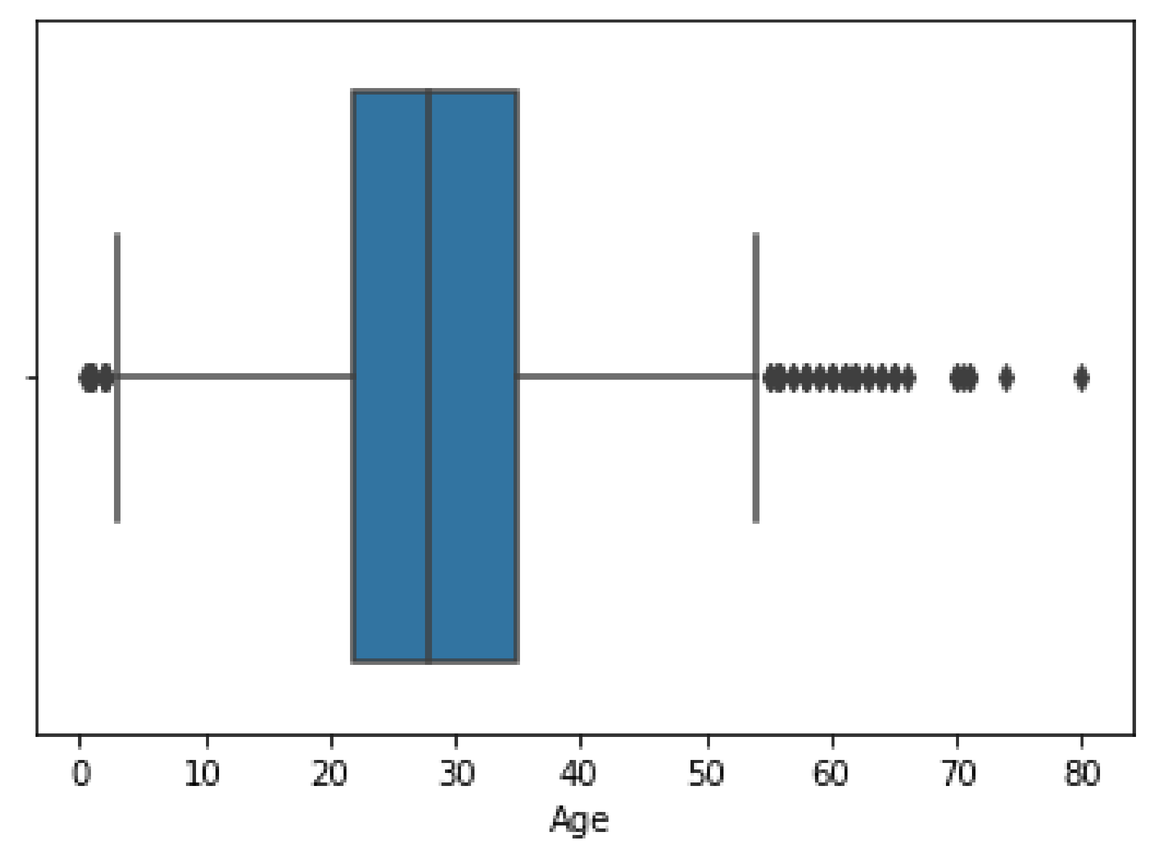
```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 11 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  891 non-null    int64
 1   Pclass    891 non-null    int64
 2   Name      891 non-null    object
 3   Sex       891 non-null    object
 4   Age       891 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  891 non-null    int64
dtypes: float64(2), int64(5), object(4)
memory usage: 83.5+ KB
```

need to be note that in data usually there is anomaly and outlier data. anomaly data mean it should be impossible to get that data using logic and outlier if it's still possible to get that data but have significant difference between that data and the rest of data. in command below we found outlier data cause it is possible for human life till 80 but data have majority passanger in range age 20-40 years

Programing Code and Output :

```
sns.boxplot(df["Age"])
```

```
/usr/local/lib/python3.7/dist-packages/seaborn/_decorators
  FutureWarning
<matplotlib.axes._subplots.AxesSubplot at 0x7f39141a9a50>
```



if too much data null can be removed to prevent false/wrong insight that can make a big loss

Programing Code and Output :

```
df.drop("Cabin", axis=1,inplace=True)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 10 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       891 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   Embarked  891 non-null    int64
dtypes: float64(2), int64(5), object(3)
memory usage: 76.6+ KB
```

Example Name Column in titanic dataset

example Name Column in titanic dataset

we can drop name column data column because that column have to many unique value and not informative for our purpose. for example in bussiness startegy we want to make a campaign for make more profit. we don't need to know what name is the most buying our product because if we only make campaign for one person. we won't yeild max profit we can get beacuse person still have limit in their fund.
need to be note: campaign here more like an limited event from company to their target market(with widen target like for males or females, for children, for spesific day customers and ect) and never to one spesific person but can spesific for one institute like university or school

Programing Code and Output :

```
df.drop("Name",axis=1,inplace=True)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 9 columns):
 #   Column    Non-Null Count  Dtype  
---  ------    --------------  -----  
 0   Survived  891 non-null    int64  
 1   Pclass    891 non-null    int64  
 2   Sex       891 non-null    object 
 3   Age       891 non-null    float64
 4   SibSp     891 non-null    int64  
 5   Parch     891 non-null    int64  
 6   Ticket    891 non-null    object 
 7   Fare      891 non-null    float64
 8   Embarked  891 non-null    int64  
dtypes: float64(2), int64(5), object(2)
memory usage: 69.6+ KB
```

Example Ticket Column in titanic dataset

Ticket Column

ticket column case is same as name because ticket is too unique to get insight from it so we can drop it.

Programing Code and Output :

```
df.drop("Ticket",axis=1,inplace=True)
```

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
Int64Index: 891 entries, 1 to 891
Data columns (total 8 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   Survived  891 non-null    int64
 1   Pclass    891 non-null    int64
 2   Sex       891 non-null    int64
 3   Age       891 non-null    float64
 4   SibSp     891 non-null    int64
 5   Parch     891 non-null    int64
 6   Fare      891 non-null    float64
 7   Embarked  891 non-null    int64
dtypes: float64(2), int64(6)
memory usage: 62.6 KB
```

Data visualization

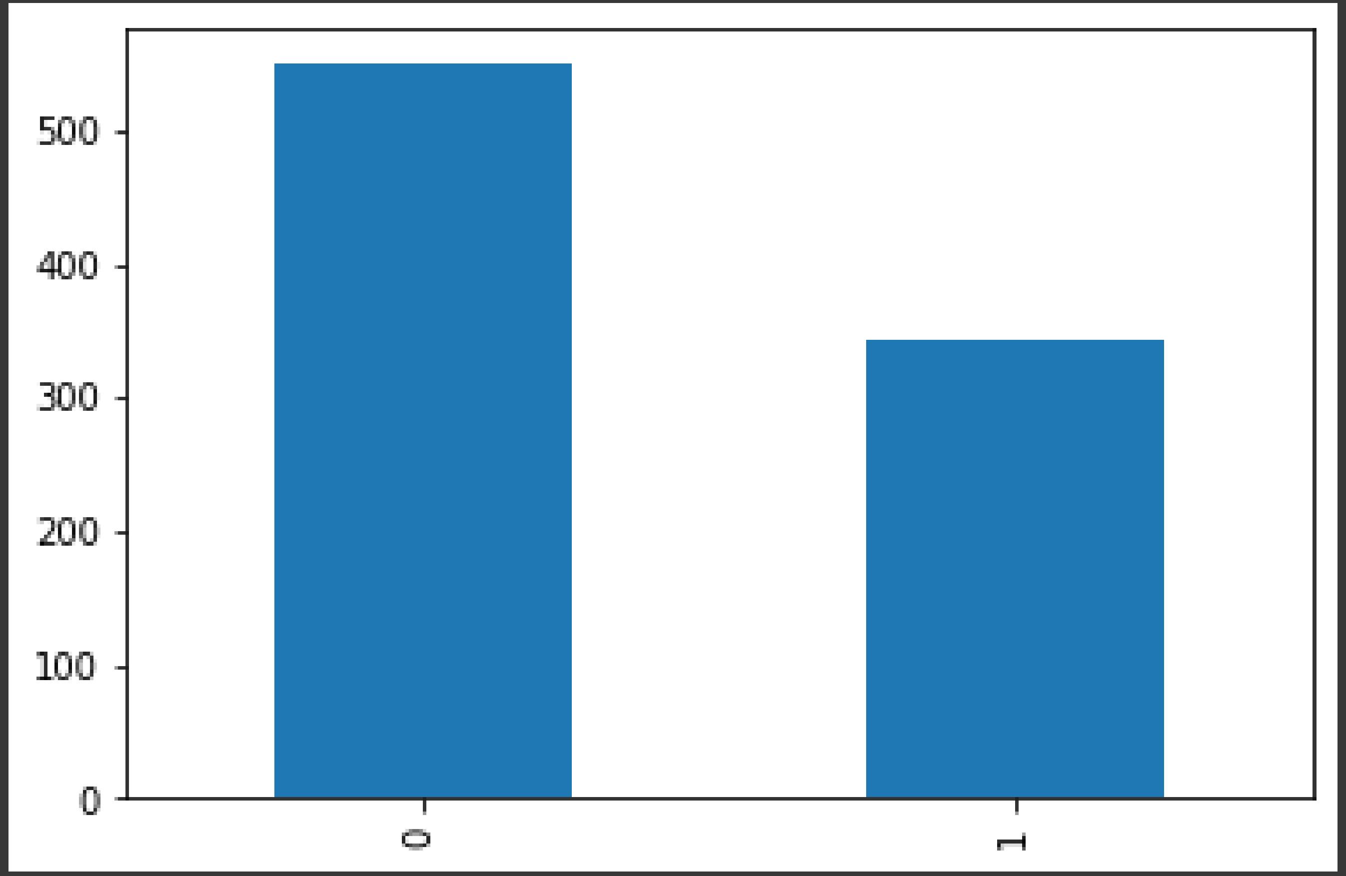
Data visualization is the graphical representation of information and data. By using visual elements like charts, graphs, and maps, data visualization tools provide an accessible way to see and understand trends, outliers, and patterns in data.

Programing Code :

```
import matplotlib.pyplot as plt
%matplotlib inline
import seaborn as sns
```

```
df.Survived.value_counts().plot(kind="bar")
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7f391211b3d0>
```

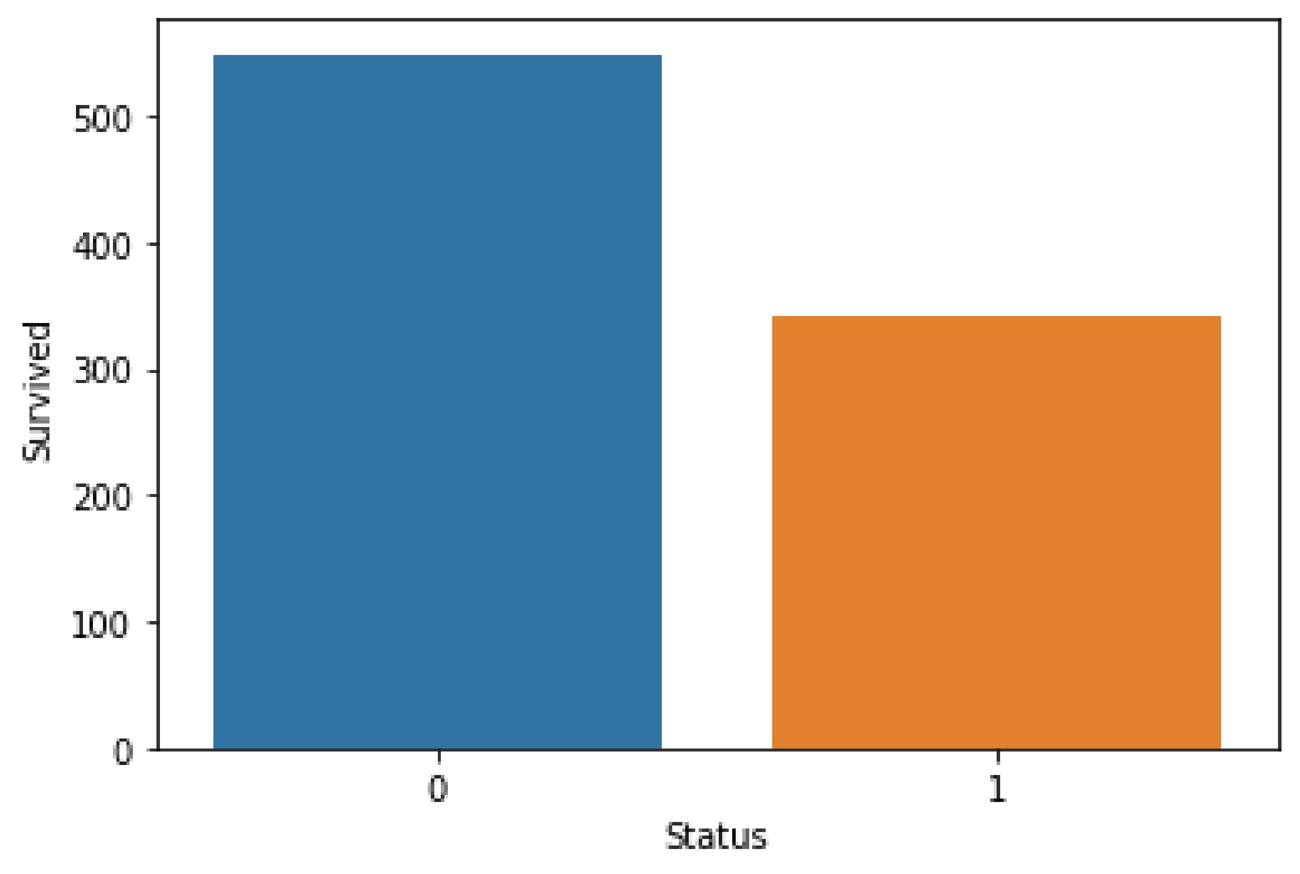


Programing Code :

```
df_Survived=pd.DataFrame(df.Survived.value_counts())

df_Survived["Status"]=[0,1]

sns.barplot(x="Status",y="Survived",data=df_Survived);
```



Programing Code :

```
df_Survived2=pd.DataFrame(df.Survived.value_counts())
df_Survived2["Status"]=["dies","alive"]
sns.barplot(x="Status",y="Survived",data=df_Survived2);
```