AI/ML Projects/Internships

Data Understanding

- Visualization
- Feature to Target Relationship

Data Pre-processing

- Handing Missing data
- Feature Selection
- Text data encoding
- Feature Scaling
- Data split

Data Modelling

- ML Models (Logistic Regression, Decision Tree, KNN…)
- Parameter tuning
- Prediction

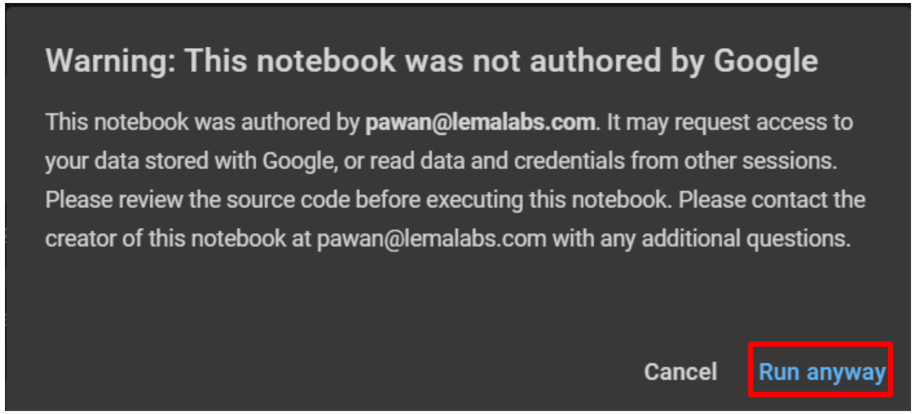
Model Evaluation

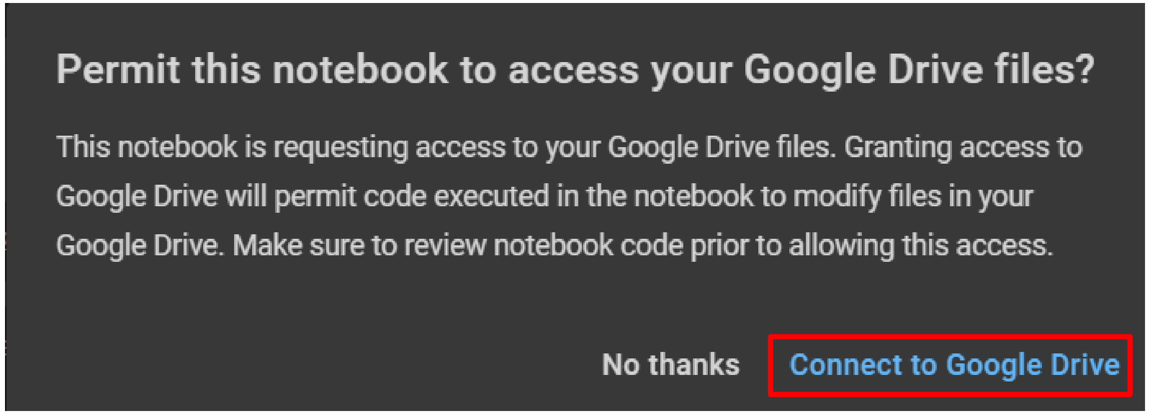
- Confusion Matrix
- Classification Report (Accuracy, Precision, Recall)
- AUC and ROC

Loading data

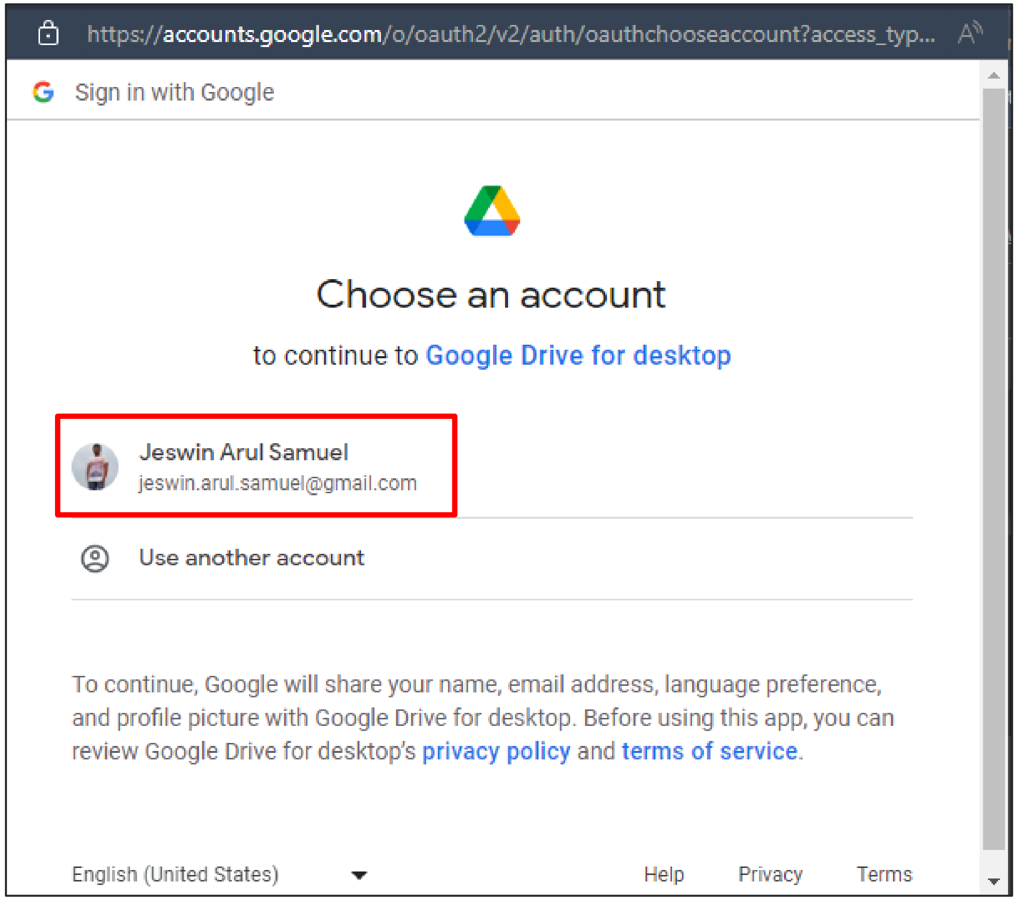
Mounting gdrive

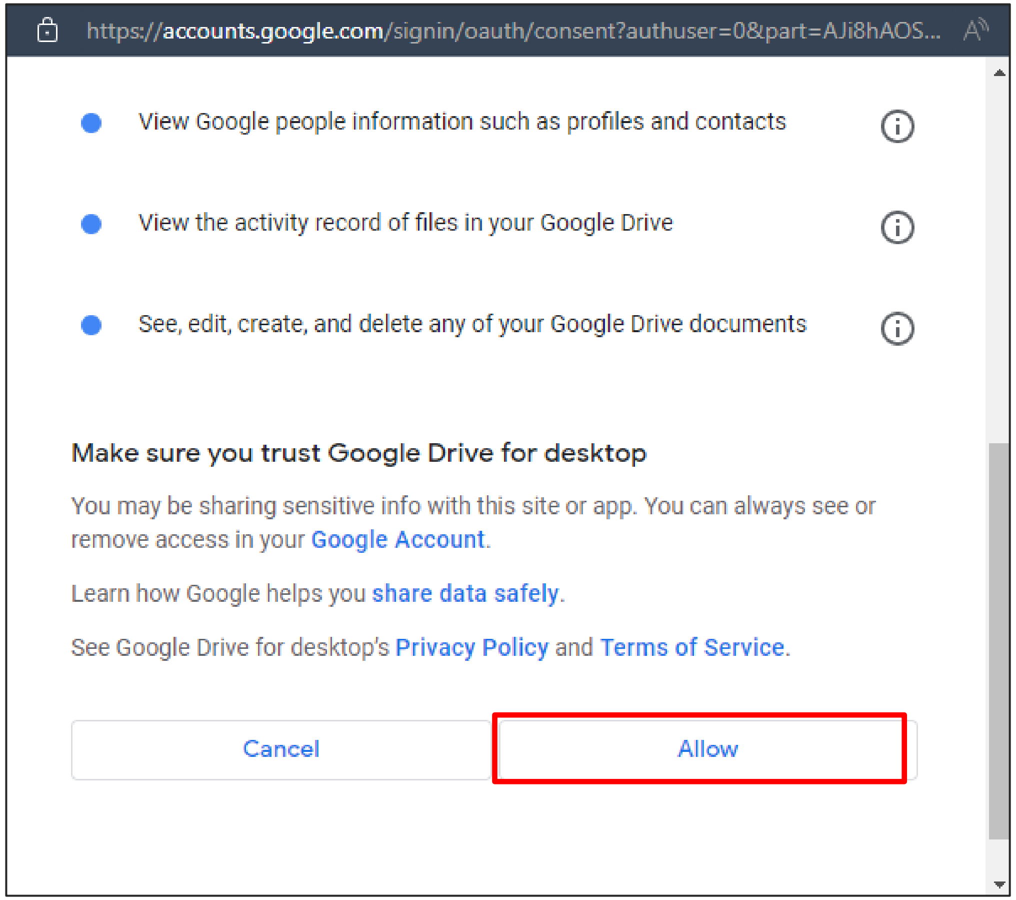
```
from google.colab import drive
drive.mount('/gdrive')
```





Mounting gdrive





Importing the libraries

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

NumPy – to handle numerical calculations
pandas – to manipulate and process data
matplotlib.pylplot – to plot graphical representation of data
seaborn – to visualize the data graphically

Reading .csv file

```
path='/gdrive/My Drive/Taplingua/Testing/'

raw_data=pd.read_csv(path+'titanic.csv')

raw_data.head()
```

Define the path where the file is stored

pd.read_csv – read csv files and stores them as dataframe

Display the first 5 rows of data

	PassengerId	Survived	Pclass	Name	Sex	Age	SibSp	Parch	Ticket	Fare	Cabin	Embarked
0	1	0	3	Braund, Mr. Owen Harris	male	22.0	1	0	A/5 21171	7.2500	NaN	S
1	2	1	1	Cumings, Mrs. John Bradley (Florence Briggs Th...	female	38.0	1	0	PC 17599	71.2833	C85	C
2	3	1	3	Heikkinen, Miss. Laina	female	26.0	0	0	STON/O2. 3101282	7.9250	NaN	S
3	4	1	1	Futrelle, Mrs. Jacques Heath (Lily May Peel)	female	35.0	1	0	113803	53.1000	C123	S
4	5	0	3	Allen, Mr. William Henry	male	35.0	0	0	373450	8.0500	NaN	S

Observe data

Observe data

```
[4]  raw_data.info()

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 891 entries, 0 to 890
Data columns (total 12 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          714 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  Cabin        204 non-null    object 
 11  Embarked     889 non-null    object 
dtypes: float64(2), int64(5), object(5)
memory usage: 83.7+ KB
```

Observe data

```
raw_data.describe()
```

	PassengerId	Survived	Pclass	Age	SibSp	Parch	Fare
count	891.000000	891.000000	891.000000	714.000000	891.000000	891.000000	891.000000
mean	446.000000	0.383838	2.308642	29.699118	0.523008	0.381594	32.204208
std	257.353842	0.486592	0.836071	14.526497	1.102743	0.806057	49.693429
min	1.000000	0.000000	1.000000	0.420000	0.000000	0.000000	0.000000
25%	223.500000	0.000000	2.000000	20.125000	0.000000	0.000000	7.910400
50%	446.000000	0.000000	3.000000	28.000000	0.000000	0.000000	14.454200
75%	668.500000	1.000000	3.000000	38.000000	1.000000	0.000000	31.000000
max	891.000000	1.000000	3.000000	80.000000	8.000000	6.000000	512.329200

Handling Missing data

Why does data sometimes go missing?

- What might missing data mean?
  - No data available (e.g., no work phone number)
  - Decided not to answer
  - Forgot to answer
  - Data accidentally deleted
  - Zero value (0)
  - It’s hard to know why any particular data might be missing!
- Does it matter if some data is missing?
  - Some data mining techniques do not tolerate any missing data
  - Is the missingness random or biased?

Patterns of missing data

- Missing Completely at Random (MCAR)
- Missing Partially at Random (MPAR)
    - Actually, statisticians call this Missing at Random (MAR)
- Missing Not at Random (MNAR)

Missing Completely at Random (MCAR)

- Any missing data is truly, completely random – missing pattern does not depend on anything that we know
  - E.g., some ages are missing, but no discernable pattern

Gender	Age	Home	Occupation	Income	Children
F		Rent	Salaried	16000	1
M	21	Own	Salaried	71000	5
M	53	Rent	Salaried	48000	2
F	24	Rent	Hourly	22000	0
M	33	Own	Hourly	90000	2
F	66	Own	Retired	62000	3
F	40	Rent	Unemployed	91000	3
M	47	Rent	Unemployed	79000	1
M		Own	Retired	38000	0
F	40	Rent	Salaried	73000	4

Missing Partially at Random (MPAR)

- Y is MPAR if whenever some other variable X has some certain values, then Y is more likely to be missing
  - E.g., age is more likely to be missing for females

Gender	Age	Home	Occupation	Income	Children
F	35	Rent	Salaried	16000	1
M	21	Own	Salaried	71000	5
M	53	Rent	Salaried	48000	2
F	24	Rent	Hourly	22000	0
M	33	Own	Hourly	90000	2
F		Own	Retired	62000	3
F		Rent	Unemployed	91000	3
M	47	Rent	Unemployed	79000	1
M	79	Own	Retired	38000	0
F		Rent	Salaried	73000	4

Missing Not at Random (MNAR)

- Y is MNAR if when it is more likely to be missing when Y itself is actually within certain ranges
  - E.g., age is more likely to be missing for older (higher age) people
  - E.g., missing income for wealthier (higher income) customers

Gender	Age	Home	Occupation	Income	Children
F	35	Rent	Salaried	16000	1
M	21	Own	Salaried	71000	5
M	53	Rent	Salaried	48000	2
F	24	Rent	Hourly	22000	0
M	33	Own	Hourly	90000	2
F		Own	Retired	62000	3
F	40	Rent	Unemployed	91000	3
M	47	Rent	Unemployed	79000	1
M		Own	Retired	38000	0
F	40	Rent	Salaried	73000	4

Options for handling (fixing) missing data

- Listwise deletion
- Variable removal
- Missing data imputation
- Do nothing

Listwise deletion

- Delete all rows with missing data in any variable in the model
- OK if you have enough data and relatively few rows with missing data

Gender	Age	Home	Occupation	Income	Children
F		Rent	Salaried	16000	1
M	21	Own	Salaried	71000	5
M	53	Rent	Salaried	48000	2
F	24	Rent	Hourly	22000	0
M	33	Own	Hourly	90000	2
F	66	Own	Retired	62000	3
F	40	Rent	Unemployed	91000	3
M	47	Rent	Unemployed	79000	1
M		Own	Retired	38000	0
F	40	Rent	Salaried	73000	4

Variable removal

- Delete the entire variable (column or attribute) if it has any missing data
- Extreme: loses valuable data; other solutions are usually preferred

Gender	Age	Home	Occupation	Income	Children
F		Rent	Salaried	16000	1
M	21	Own	Salaried	71000	5
M	53	Rent	Salaried	48000	2
F	24	Rent	Hourly	22000	0
M	33	Own	Hourly	90000	2
F	66	Own	Retired	62000	3
F	40	Rent	Unemployed	91000	3
M	47	Rent	Unemployed	79000	1
M		Own	Retired	38000	0
F	40	Rent	Salaried	73000	4

Missing data imputation

- Guess the missing value and fill it in
  - mean (for numbers)
  - mode (for categories)
    - But never just replace with zero!
- Conserves as much data as possible for small datasets

Gender	Age	Home	Occupation	Income	Children
F	40.5	Rent	Salaried	16000	1
M	21	Own	Salaried	71000	5
M	53	Rent	Salaried	48000	2
F	24	Rent	Hourly	22000	0
M	33	Own	Hourly	90000	2
F	66	Own	Retired	62000	3
F	40	Rent	Unemployed	91000	3
M	47	Rent	Unemployed	79000	1
M	40.5	Own	Retired	38000	0
F	40	Rent	Salaried	73000	4

Do nothing

- Sometimes the variable is not critical for the analysis

Gender	Age	Home	Occupation	Income	Children
F		Rent	Salaried	16000	1
M	21	Own	Salaried	71000	5
M	53	Rent	Salaried	48000	2
F	24	Rent	Hourly	22000	0
M	33	Own	Hourly	90000	2
F	66	Own	Retired	62000	3
F	40	Rent	Unemployed	91000	3
M	47	Rent	Unemployed	79000	1
M		Own	Retired	38000	0
F	40	Rent	Salaried	73000	4

Missing values

```
raw_data.isnull().sum()
```

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

Imputing missing values

```
from sklearn.impute import SimpleImputer

imputer = SimpleImputer(missing_values = np.nan, strategy = "mean")

imputer = imputer.fit(data[['Age']])

data['Age'] = imputer.transform(data[['Age']])
```

Column deletion

- If the missing values in the columns doesn't affect the target.
- If the missing values are greater than the existing data

```
data = data.drop(['Cabin'], axis =1)
```

Row deletion

- Not a lot of missing data
- Losing few rows is affordable for the dataset

```
data = data.dropna()
```

Feature Selection

Feature selection

- Based on Intuition
  - Serial numbers
  - Names of people (except if the problem is more personalised)
  - Identification values (Invoice no, ticket no, etc..)
  - Identical features with same meaning but different data

```
data = data.drop(['PassengerId','Name','Ticket','Embarked'], axis=1)
```

Assignment

- In the heart disease dataset
  - Handle missing values – Justify your actions in a text markdown
  - Decide which features to keep and drop – Justify your actions
- In general
  - A small report on the types of data in the dataset (Continuous, Categorical, etc..)
  - Read about heatmap, pair plot, cluster map and their interpretations.