Aim:

Build a machine learning model that predicts the type of people who survived the Titanic shipwreck using passenger data (i.e., name, age, gender, socio-economic class, etc.)

Introduction:

The goal of the project was to predict the survival of passengers based off a set of data. We used Kaggle competition "Titanic: Machine Learning from Disaster" (see https://www.kaggle.com/c/titanic/data) to retrieve necessary data and evaluate accuracy of our predictions.

The historical data has been split into two groups, a 'training set' and a 'test set'. For the training set, we are provided with the outcome (whether or not a passenger survived). We used this set to build our model to generate predictions for the test set.

For each passenger in the test set, we had to predict whether or not they survived the sinking. Our score was the percentage of correctly predictions. In our work, we learned

- Programming language Python and its libraries NumPy (to perform matrix operations) and SciKit-Learn (to apply machine learning algorithms)
- Several machine learning algorithms ('Support Vector Machines', 'Logistic Regression', 'Random Forest')
- Feature Engineering techniques.

Software used:

- Python 2.7.6 with the libraries numpy, sklearn, and matplotlib
- Microsoft Excel

Work flow:

1. Learn programming language Python
2. Learn Shannon Entropy and write Python code to compute Shannon Entropy
3. Get familiar with Kaggle project and try using Pivot Tables in Microsoft Excel to analyze the data.
4. Learn to use SciKit-Learn library in Python, including
   a) Building decision tree
   b) Building Random Forests
   c) Building Extra Trees
   d) Using Linear Regression algorithm
5. Performing Feature Engineering, applying machine learning algorithms, and analyzing result

Training and Test Data:

CSV file and contain the following fields:

a. Passenger ID
b. Passenger Class
c. Name
d. Sex
e. Age
f. Number of passenger's siblings and spouses on board
g. Number of passenger's parents and children on board
h. Ticket
i. Fare

Feature Engineering:

Since the data can have missing fields, incomplete fields, or fields containing hidden information, a crucial step in building any prediction system is Feature Engineering. For instance, the fields Age, Fare, and Embarked in the training and test data, had missing values that had to be filled in. The field Name while being useless itself, contained passenger's Title (Mr., Mrs., etc.), we also used passenger's surname to distinguish families on board of Titanic.

Stage 1: Importing the Libraries and getting data:

We start by importing all the important packages/libraries that would be required for building our model as well as to analyze the given dataset.

```
import numpy as np
import pandas as pd
import seaborn as sns
from matplotlib import pyplot as plt
from sklearn.linear_model import LogisticRegression
from sklearn.ensemble import RandomForestClassifier
from sklearn.svm import SVC
```

```
test_df = pd.read_csv("test.csv")
train_df = pd.read_csv("train.csv")
train_df.head()
```

	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

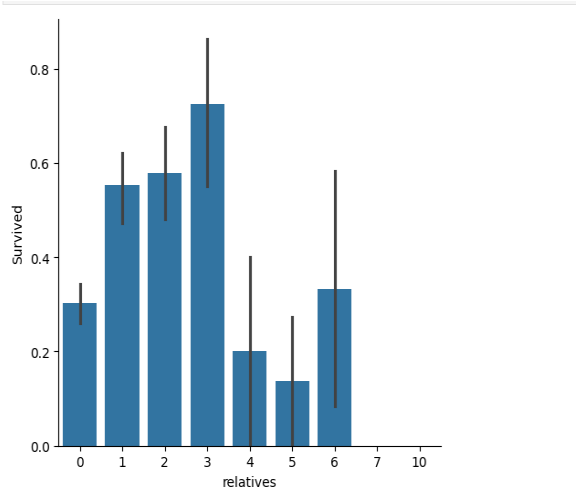
Stage 2: Data Exploration/Analysis and Visualization

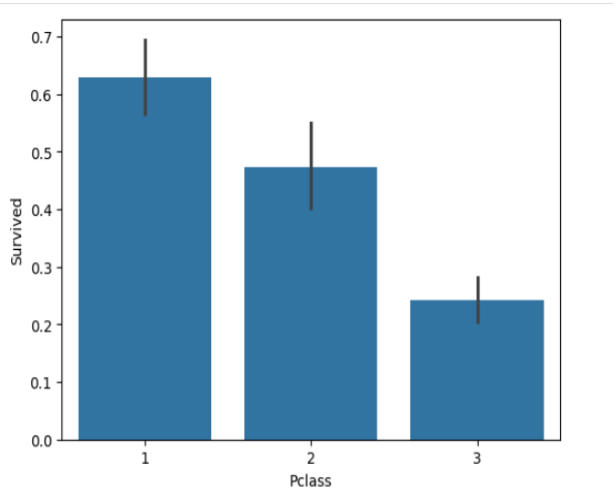
The next step is to start analyzing the given train dataset to find out patterns between the features and finding relations of essential features with the target feature (Survived or not).

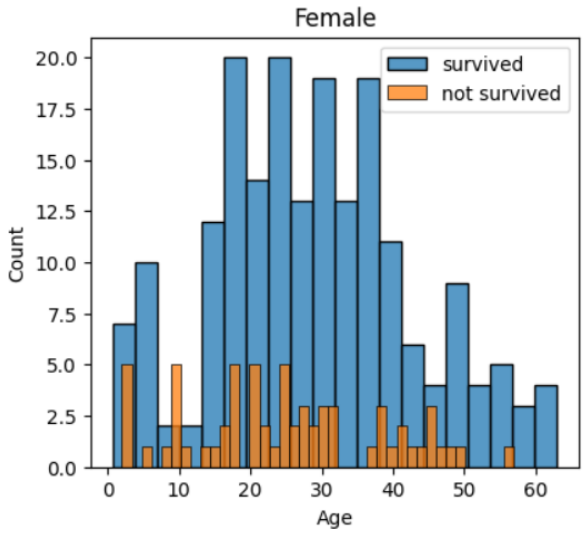
We observe that the training dataset contains approx 891 rows and 12 columns (features). We could also check the entire description of the dataset at once to get a better understanding of the dataset.

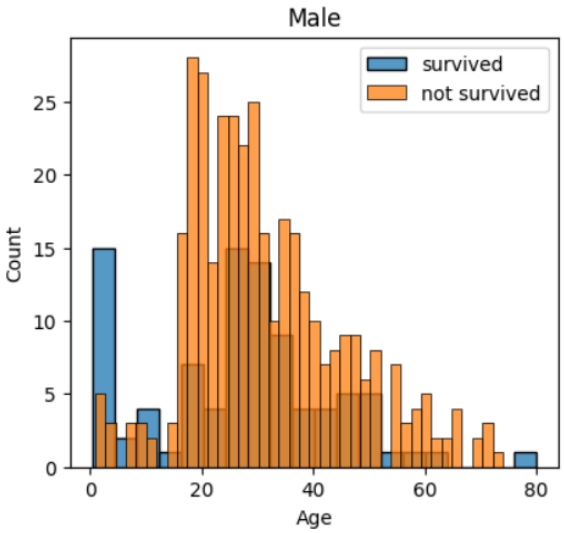
Now we could start comparing individual features with the target feature and find out the effect of individual features on the target label i.e. how individual features determine whether the person survived or not.

Thus, we would be comparing and visualizing certain important labels of the dataset and dropping less important ones to find patterns for our prediction.









Stage 3: Data Preprocessing

Now we saw that there were approx 12 different feature columns provided in the dataset. Now not all features have an impact on the required target feature (Survival in our case). So, it would be better to select the features of major importance and drop certain features that have a minor impact on our target column. This process is referred to as feature selection in machine learning.

Stage 5 : Model Building and Training

Now import the train_test_split from the sklearn library and then split the dataset into train and test specifying a test_size.

[27]:
```
X_train = train_df.drop("Survived", axis=1)
Y_train = train_df["Survived"]
X_test  = test_df.drop("PassengerId", axis=1).copy()
```

Logistic regression model:

[29]:
```
logreg = LogisticRegression(max_iter=5000)
logreg.fit(X_train, Y_train)

Y_pred = logreg.predict(X_test)

acc_log = round(logreg.score(X_train, Y_train) * 100, 2)
```

SVM model:

[30]:
```
linear_svc = SVC()
linear_svc.fit(X_train, Y_train)

Y_pred = linear_svc.predict(X_test)

acc_linear_svc = round(linear_svc.score(X_train, Y_train) * 100, 2)
```

Random Forest model:

```
[28]: random_forest = RandomForestClassifier(n_estimators=100)
      random_forest.fit(X_train, Y_train)

      Y_prediction = random_forest.predict(X_test)

      random_forest.score(X_train, Y_train)
      acc_random_forest = round(random_forest.score(X_train, Y_train) * 100, 2)
```

Now, let's compare their scores to find the most suitable model for our problem.

```
[31]: results = pd.DataFrame({
          'Model': ['Support Vector Machines', 'Logistic Regression', 'Random Forest'],
          'Score': [acc_linear_svc, acc_log, acc_random_forest]
      })
      result_df = results.sort_values(by='Score', ascending=False)
      result_df = result_df.set_index('Score')
      result_df
```

[31]:

Score	Model
92.59	Random Forest
82.27	Support Vector Machines
81.26	Logistic Regression

In our case, the maximum accuracy score was obtained from the **Random Forest** model hence we would be using it to train our model and predict the survival chances of the passengers.

Conclusion:

Hence, we successfully built a machine learning model that predicts the type of people who survived the Titanic shipwreck using passenger data.