Name: Putta Hanumantha Reddy Raghavendra Reddy
Student ID: C0851724

1. Define data science?

The study of obtaining valuable information from data using advanced analytical methods and scientific principles for business decision-making, strategic planning, and other objectives is known as data science. Data science encompasses several fields, including data preparation, data mining, predictive analytics, machine learning, data visualization, statistics, mathematics, and software development.

Data science Process and Life cycle:

1. Determine a business hypothesis to test.

2. Data Collection and Prepare for analysis

3. Experiment with various analytical models to see what you can produce.

4. Choose the best model and compare it to the data.

5. Continually use the model with new data by deploying it.

1.1 How is Data science transforming the healthcare sector?

Medicine and healthcare are the two most vital aspects of our life as humans. Traditionally, treatment relied exclusively on the doctors' discretion. A doctor, for example, would have to advise appropriate therapies depending on a patient's symptoms. However, this was not always the case, and it was susceptible to human mistakes. However, with improvements in computers and data science, precise diagnostic measures are now possible. Data science is used in various sectors in healthcare, including medical imaging, drug development, genomics, predictive diagnosis, and others.

1.2 How is data science changing the face of the banking and finance sector?

Banks are attempting to detect trends in a significant amount of available transaction data to communicate with their clients more efficiently. Banks use data from consumer transactions, previous history, trends, communication, and loyalty to implement Data Science in banking. Because this data is usually unstructured and difficult to deal with, extracting insights from such an enormous volume of information is a huge task. For this, several data analysis approaches such as data fusion and integration, machine learning, natural language processing, signal processing, and so on can be applied. Banks use data Science to accomplish various vital activities, such as fraud detection and customer segmentation.

2. What are the missing values and errors in data?

Missing data (or missing values) is the value of a variable in an observation that is not stored. Missing data is a prevalent problem in all studies, and it can significantly impact the conclusions that can be taken from the data [1]. When corporations, organizations, and people collect and evaluate data, they frequently need to know whether their figures are accurate. Statistics reliability, also known as error, can be used to determine how reliable a statistic is within a given percentage and how often that statistic is likely to be correct to that degree.

2.1 Handling missing values in data preprocessing step

The handling of missing data is critical during the preprocessing of the dataset as many machine learning algorithms do not support missing values.

1. Delete the rows or columns with null values to deal with missing values. If more than half of the rows in a column are null, the column might be eliminated. The rows with null values in one or more columns can also be removed.

2. Columns with continuous numeric values in the dataset can be replaced with the mean, median, or mode of the remaining values in the queue. In comparison to the previous way, this strategy can prevent data loss. A statistical approach to dealing with missing values replaces the above two approximations (mean, median)
3. When missing values are found in categorical columns (string or numerical), we might use the most frequent category to fill in the gaps. If there are significant missing values, we can replace them with a new variety.

2.2 Why is it important to handle? Explain with an example in detail

The quality of a machine learning model can be significantly impacted by training it using a data set that contains many missing values. Deterministic models are particularly affected by this issue. The bulk of deterministic models in use today for practical purposes are, in fact, deterministic. As a result, missing data must be dealt with before the machine learning model can be used. For example, we will consider 4 attributes such as Mobile ID, Mobile Package, Download Speed, and Data Limit Usage with values as shown in the below table.

Mobile ID	Mobile Package	Download Speed	Data Limit Usage
1	Fast+	N/A	80%
2	Lite	99	70%
3	Fast+	167	10%
4	Fast+	N/A	75%

Download Speed missing values has relation to Download Speed, Data Limit Usage, and some other unknown variables. Here value is missing beyond a data limit usage range (>=75%) but we cannot predict the value. It is difficult to predict missing values.

3. Experiment

3.1 Problem Statement

Customers' information submitted while filling out an online application form will be used to automate (in real-time) the loan qualifying process. Gender, marital status, education, dependents, income, loan amount, credit history, and other information. They are having trouble identifying the client segments that are eligible for a specific loan amount so that they can automate the process. In this scenario, they have only offered a small amount of data

3.2 Introduction

It is a project for loan forecasting. I'll anticipate which of the bank's customers are most likely to have their loan applications accepted based on a set of characteristics. This project is a classification problem in machine learning, and banks can use these models to analyze their customers' loan qualifications and fitness.

3.3 Methodology

1. **Dataset Description**

This dataset is named Loan Prediction Dataset data set. There are two data subsets, one is the training dataset, and the other is the test dataset. The dataset is taken from the Kaggle [2]. The dataset contains a set of **613** records under **13 attributes**. Table 1 describes the list of attributes that contribute to getting a loan sanction.

Table 1: List of attributes

Variable	Description
Loan_ID	Unique Loan ID
Gender	Male/ Female
Married	Applicant married (Y/N)
Dependents	Number of dependents
Education	Applicant Education (Graduate/ Under Graduate)
Self_Employed	Self employed (Y/N)
ApplicantIncome	Applicant income
CoapplicantIncome	Coapplicant income
LoanAmount	Loan amount in thousands
Loan_Amount_Term	Term of loan in months
Credit_History	credit history meets guidelines
Property_Area	Urban/ Semi Urban/ Rural
Loan_Status	Loan approved (Y/N)

2. Analysis

info () method allows us to know and learn the shape of object types of our data.

In [31]:
```
# Loading Traininig and Testing Dataset
loan_train = pd.read_csv(r"C:\Users\HP1\Downloads\train_u6lujuX_CVtuZ9i.csv",sep=',')
loan_test = pd.read_csv(r"C:\Users\HP1\Downloads\test_Y3wMUE5_7gLdaTN.csv",sep=',')
loan_train.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Loan_ID            614 non-null    object
 1   Gender             601 non-null    object
 2   Married            611 non-null    object
 3   Dependents         599 non-null    object
 4   Education          614 non-null    object
 5   Self_Employed      582 non-null    object
 6   ApplicantIncome    614 non-null    int64
 7   CoapplicantIncome  614 non-null    float64
 8   LoanAmount         592 non-null    float64
 9   Loan_Amount_Term   600 non-null    float64
 10  Credit_History     564 non-null    float64
 11  Property_Area      614 non-null    object
 12  Loan_Status        614 non-null    object
dtypes: float64(4), int64(1), object(8)
memory usage: 62.5+ KB
```

Describe () method give us summary statistics for numerical columns in our dataset

```
In [34]: loan_train.describe()
```

Out[34]:

	ApplicantIncome	CoapplicantIncome	LoanAmount	Loan_Amount_Term	Credit_History
count	614.000000	614.000000	592.000000	600.00000	564.000000
mean	5403.459283	1621.245798	146.412162	342.00000	0.842199
std	6109.041673	2926.248369	85.587325	65.12041	0.364878
min	150.000000	0.000000	9.000000	12.00000	0.000000
25%	2877.500000	0.000000	100.000000	360.00000	1.000000
50%	3812.500000	1188.500000	128.000000	360.00000	1.000000
75%	5795.000000	2297.250000	168.000000	360.00000	1.000000
max	81000.000000	41667.000000	700.000000	480.00000	1.000000

Isna().values.any() method Return a Boolean value indicating if the values are null

```
In [14]: # checking for any missing values
         loan_test.isna().values.any()
         loan_train.isna().values.any()
Out[14]: True
```

3. Data Preprocessing

Before we develop our models, we need to undertake the following preparation processes based on Exploratory Data Analysis:

3.1 Missing value imputation

1. Imputation using the mean or median for numerical variables

2. Imputation using mode for categorical variables

3.2 Identify all classified columns first, and then for object to numeric conversion, label encoding is used.

Handling Missing Values

The below code describes sum of null values present in each column for both train dataset and test dataset.

```
In [6]: # Checking We have missing data
        loan_train.isna().values.any()
        # Checking we have missing data
        loan_test.isna().values.any()
```

```
Out[6]: True
```

```
In [7]: loan_train.isna().sum()
```

```
Out[7]: Loan_ID              0
        Gender              13
        Married              3
        Dependents          15
        Education            0
        Self_Employed       32
        ApplicantIncome      0
        CoapplicantIncome    0
        LoanAmount          22
        Loan_Amount_Term    14
        Credit_History      50
        Property_Area        0
        Loan_Status          0
        dtype: int64
```

```
In [19]: # Same for test data
         loan_test.isna().sum()
```

```
Out[19]: Loan_ID              0
         Gender              11
         Married              0
         Dependents          10
         Education            0
         Self_Employed       23
         ApplicantIncome      0
         CoapplicantIncome    0
         LoanAmount           5
         Loan_Amount_Term     6
         Credit_History      29
         Property_Area        0
         dtype: int64
```

Data Cleaning and Preparation

Imputation of Numerical Variables Using Mean or Median

Imputation of Categorical Variables Using Mode

In [8]:
```
# We'll do a forward fill here, so, we get only 1 or 0 to fill the missing data
loan_train['Credit_History'].fillna(method='ffill', inplace=True)
loan_train['Credit_History'].isna().values.any()
# We'll fill this column using the median of the values
median_loan = loan_train['Loan_Amount_Term'].median()
loan_train['Loan_Amount_Term'].fillna((median_loan), inplace=True)
loan_train['Loan_Amount_Term'].isna().values.any()
# We'll fill this column using the median of the values
median_loan_amount = loan_train['LoanAmount'].median()
loan_train['LoanAmount'].fillna((median_loan_amount), inplace=True)
loan_train['LoanAmount'].isna().values.any()
# Count the values to know which occurs most frequently
loan_train['Self_Employed'].value_counts()
#Fill with mode
loan_train['Self_Employed'].fillna('No', inplace=True)
loan_train['Self_Employed'].isna().values.any()
# fill with mode
loan_train['Dependents'].fillna(0, inplace=True)
loan_train['Dependents'].isna().values.any()
loan_train['Married'].mode()
# fill with mode
loan_train['Married'].fillna('Yes', inplace=True)
loan_train['Married'].isna().values.any()
loan_train['Gender'].mode()
# fill with mode
loan_train['Gender'].fillna('Male', inplace=True)
loan_train['Gender'].isna().values.any()
```

Out[8]:
```
False
```

Identifying all categorical columns and Label Encoding for Object to Numeric Conversion

```
In [35]: #first identify all categorical columns & pass into a variable
objectlist_train = loan_train.select_dtypes(include = "object").columns
# Then Label Encoding for object to numeric conversion
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
for feature in objectlist_train:
    loan_train[feature] = le.fit_transform(loan_train[feature].astype(str))
print (loan_train.info())
# Now, repeat the same process to encode the test data
objectlist_test = loan_test.select_dtypes(include='object').columns
for feature in objectlist_test:
    loan_test[feature] = le.fit_transform(loan_test[feature].astype(str))
print (loan_test.info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 614 entries, 0 to 613
Data columns (total 13 columns):
 #   Column             Non-Null Count  Dtype
---  ------             --------------  -----
 0   Loan_ID            614 non-null    int32
 1   Gender             614 non-null    int32
 2   Married            614 non-null    int32
 3   Dependents         614 non-null    int32
 4   Education          614 non-null    int32
 5   Self_Employed      614 non-null    int32
 6   ApplicantIncome    614 non-null    int64
 7   CoapplicantIncome  614 non-null    float64
 8   LoanAmount         592 non-null    float64
 9   Loan_Amount_Term   600 non-null    float64
```

4. Generating Training and Testing Dataset

While building a Machine Learning model it is better to train the model and test with a subset of the dataset.

```
In [36]: x = loan_train.iloc[:,1:].drop('Loan_Status', axis=1) # drop loan_status column because that is what we are predicting
y = loan_train['Loan_Status']
train_x, test_x, train_y, test_y = train_test_split(x, y, test_size=0.30, random_state=0)
```

5. Model Generation

Before data cleaning, I tried to build the model. It has errored due to Null values

```
In [37]: lr_model = LogisticRegression(solver='lbfgs', multi_class='auto')
         lr_model.fit(train_x, train_y)
         predict_y_3 = lr_model.predict(test_x)
         print("Accuracy:", accuracy_score(predict_y_3, test_y))
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
~\AppData\Local\Temp/ipykernel_1560/2129396029.py in <module>
      1 lr_model = LogisticRegression(solver='lbfgs', multi_class='auto')
----> 2 lr_model.fit(train_x, train_y)
      3 predict_y_3 = lr_model.predict(test_x)
      4 print("Accuracy:", accuracy_score(predict_y_3, test_y))

F:\Anaconda\lib\site-packages\sklearn\utils\validation.py in _assert_all_finite(X, allow_nan, msg_dtype)
    101                 not allow_nan and not np.isfinite(X).all()):
    102             type_err = 'infinity' if allow_nan else 'NaN, infinity'
--> 103             raise ValueError(
    104                     msg_err.format
    105                     (type_err,

ValueError: Input contains NaN, infinity or a value too large for dtype('float64').
```

```
In [6]: # Checking We have missing data
        loan_train.isna().values.any()
        # Checking we have missing data
        loan_test.isna().values.any()
```

```
Out[6]: True
```

After data cleaning, I tried to build the model successfully

```
]: x = loan_train.iloc[:,1:].drop('Loan_Status', axis=1) # drop loan_status column because that is what we are predicting
   y = loan_train['Loan_Status']
   train_x, test_x, train_y, test_y = train_test_split(x, y, test_size=0.30, random_state=0)
```

```
]: lr_model = LogisticRegression()
   lr_model.fit(train_x, train_y)
   predict_y = lr_model.predict(test_x)
   print("Accuracy:", accuracy_score(predict_y, test_y))
```

```
Accuracy: 0.8432432432432433
```

Conclusion

For our ML model, at 84% accuracy, the Logistic Regression model is the most suitable to make this prediction.

References

1. Graham JW. Missing data analysis: making it work in the real world. Annu Rev Psychol. 2009;60:549–576. [PubMed] [Google Scholar]

2. Kaggle, Loan Prediction Dataset

https://www.kaggle.com/gavincanacam/home-loan-predictions