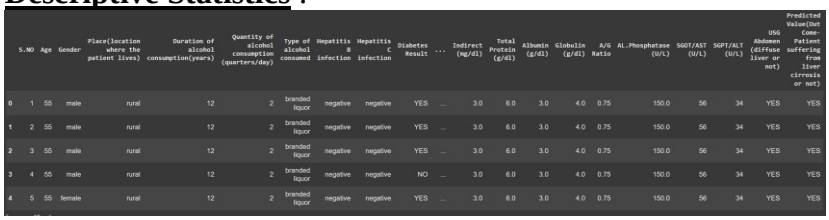
Data Collection and Preprocessing Phase

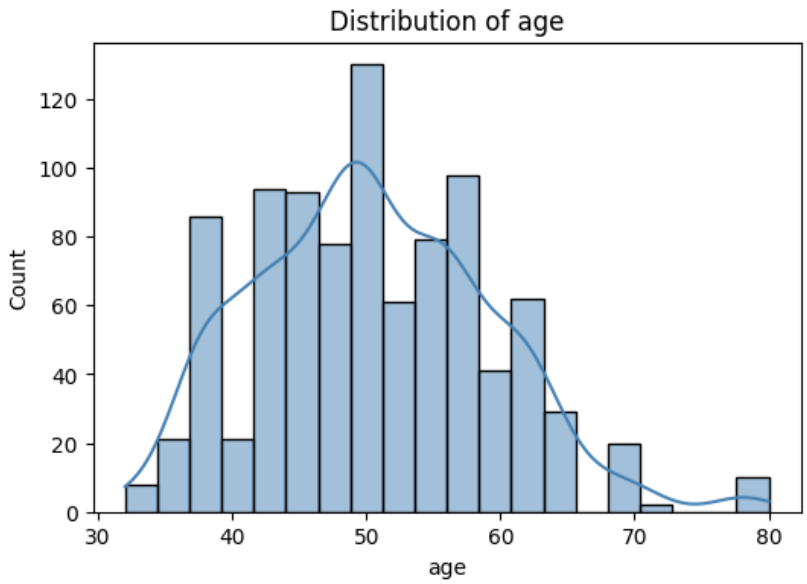
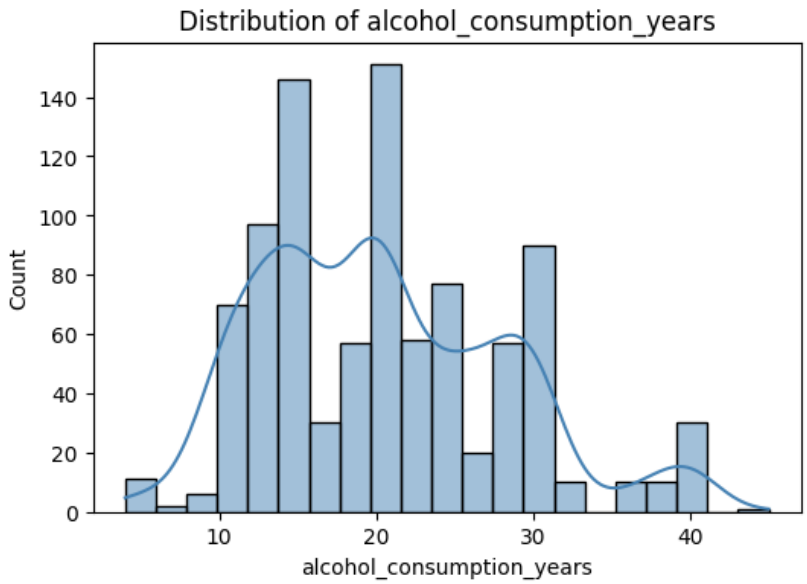
Date	24 June 2025
Team ID	SWTID1749708868
Project Title	Revolutionizing Liver Care : Predicting Liver Cirrhosis Using Advanced Machine Learning Techniques
Maximum Marks	6 Marks

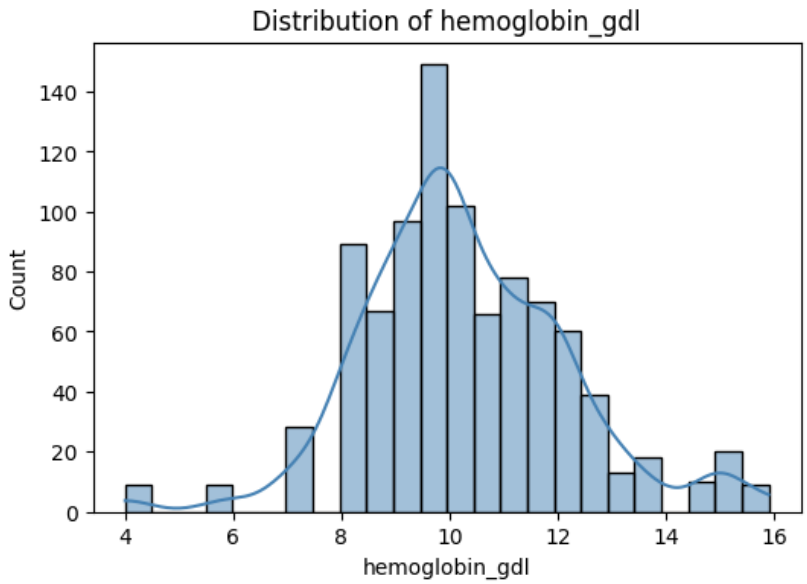
Data Exploration and Preprocessing Template

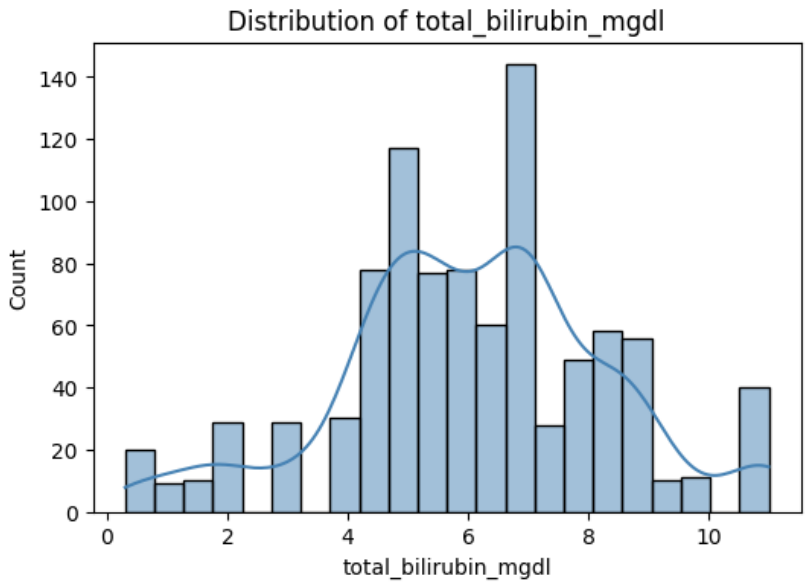
Dataset variables will be statistically analyzed to identify patterns and outliers, with Python employed for preprocessing tasks like normalization and feature engineering. Data cleaning will address missing values and outliers, ensuring quality for subsequent analysis and modeling, and forming a strong foundation for insights and predictions.

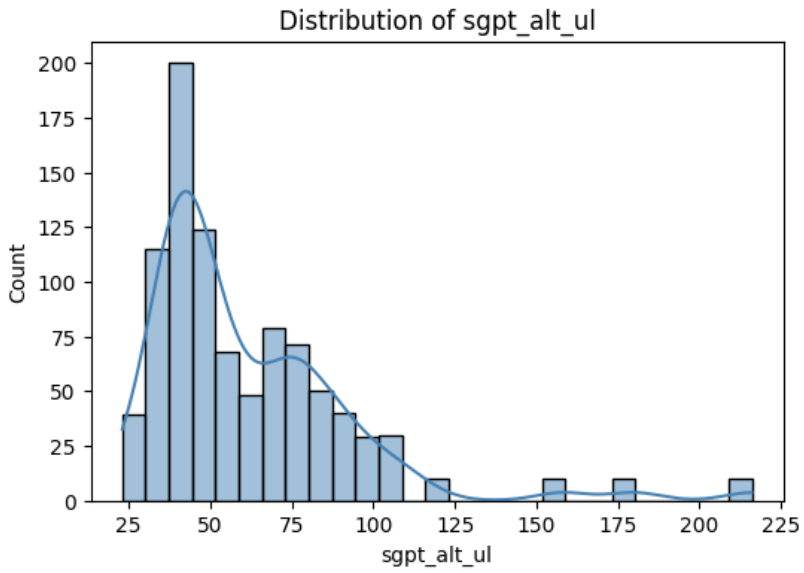
Section	**Description**
Data Overview	**<u>Dimension</u>** : 950 rows X 13 columns **<u>Descriptive Statistics</u>** : 

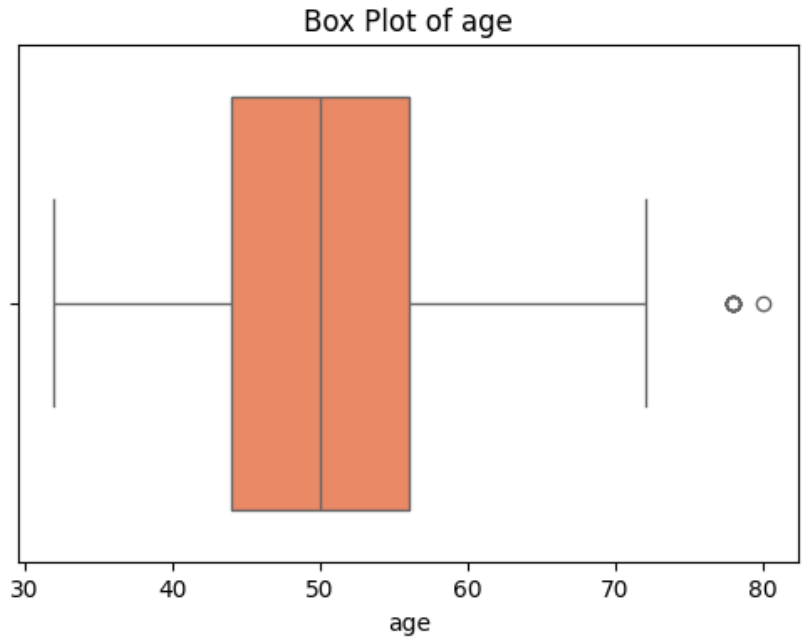
	S.NO	Age	Gender	Place(location where the patient lives)	Duration of alcohol consumption(years)	Quantity of alcohol consumption (quarters/day)	Type of alcohol consumed	Hepatitis B infection	Hepatitis C infection	Diabetes Result	...	Indirect (mg/dl)	Total Protein (g/dl)	Albumin (g/dl)	Globulin (g/dl)	A/G Ratio	Al.Phosphatase (U/L)	SGOT/AST (U/L)	SGPT/ALT (U/L)	USG Abdomen (diffuse liver or not)	Predicted Value(Outcome- Patient suffering from liver cirrosis or not)
0	1	55	male	rural	12	2	branded liquor	negative	negative	YES	...	3.0	6.0	3.0	4.0	0.75	150.0	56	34	YES	YES
1	2	55	male	rural	12	2	branded liquor	negative	negative	YES	...	3.0	6.0	3.0	4.0	0.75	150.0	56	34	YES	YES
2	3	55	male	rural	12	2	branded liquor	negative	negative	YES	...	3.0	6.0	3.0	4.0	0.75	150.0	56	34	YES	YES
3	4	55	male	rural	12	2	branded liquor	negative	negative	NO	...	3.0	6.0	3.0	4.0	0.75	150.0	56	34	YES	YES
4	5	55	female	rural	12	2	branded liquor	negative	negative	YES	...	3.0	6.0	3.0	4.0	0.75	150.0	56	34	YES	YES

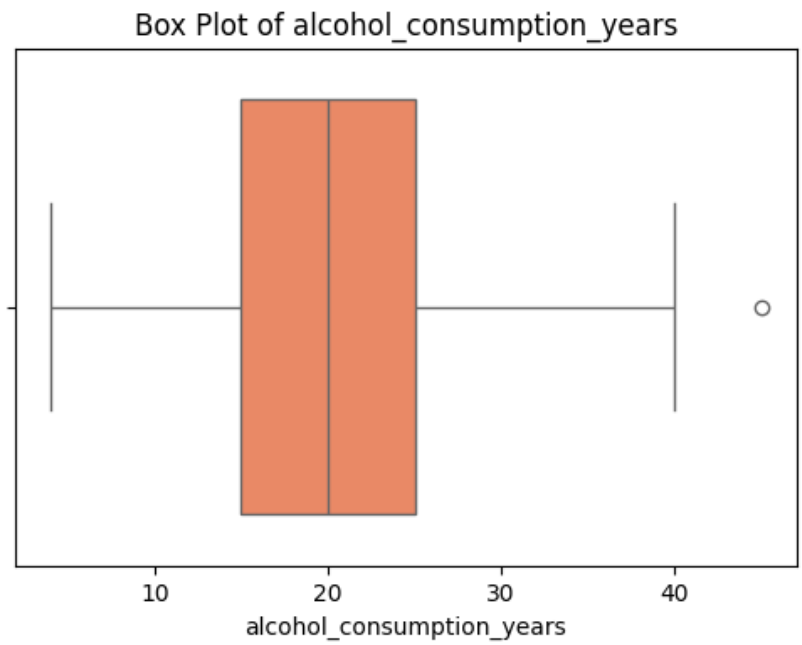
Univariate Analysis	 


Distribution of hemoglobin_gdl
Count
hemoglobin_gdl
0
20
40
60
80
100
120
140
4
6
8
10
12
14
16

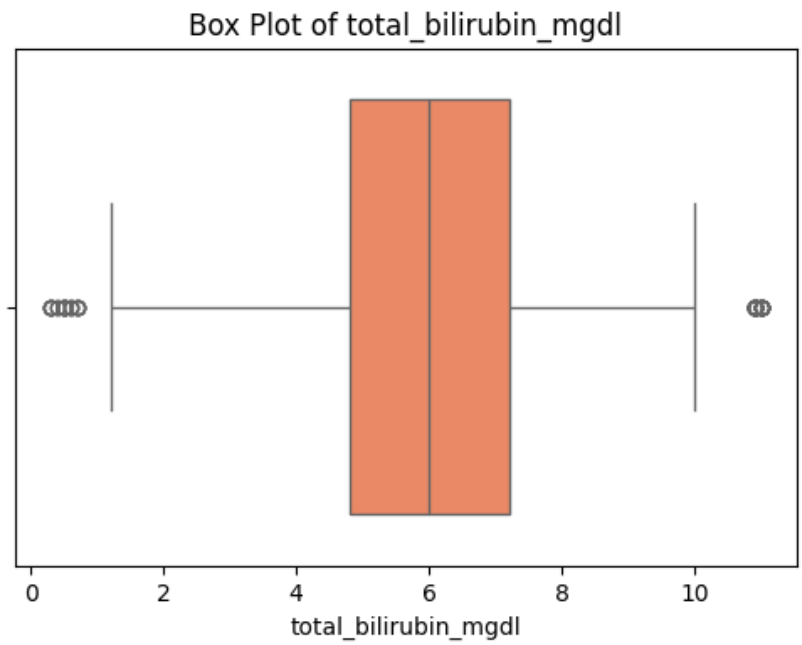

Distribution of total_bilirubin_mgdl
Count
total_bilirubin_mgdl
0
20
40
60
80
100
120
140
0
2
4
6
8
10

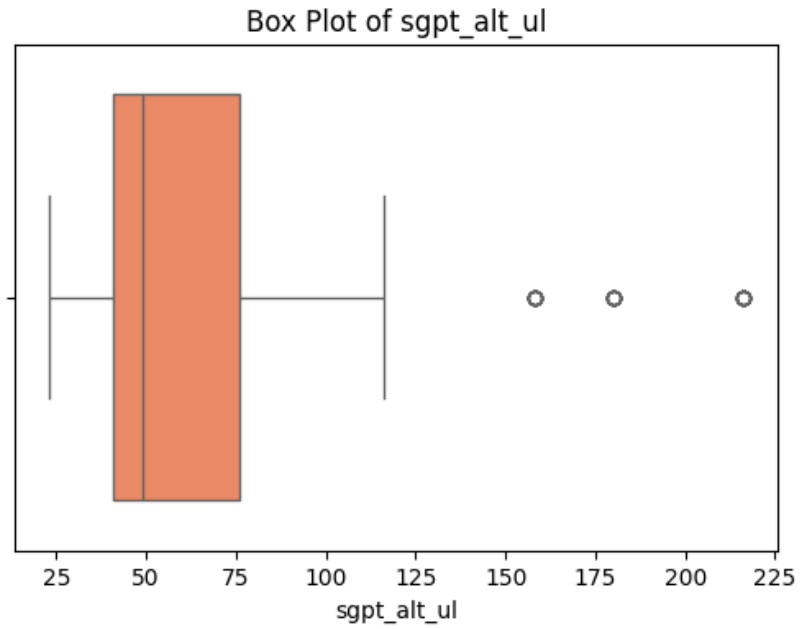

Distribution of sgpt_alt_ul
Count
sgpt_alt_ul
0
25
50
75
100
125
150
175
200
225

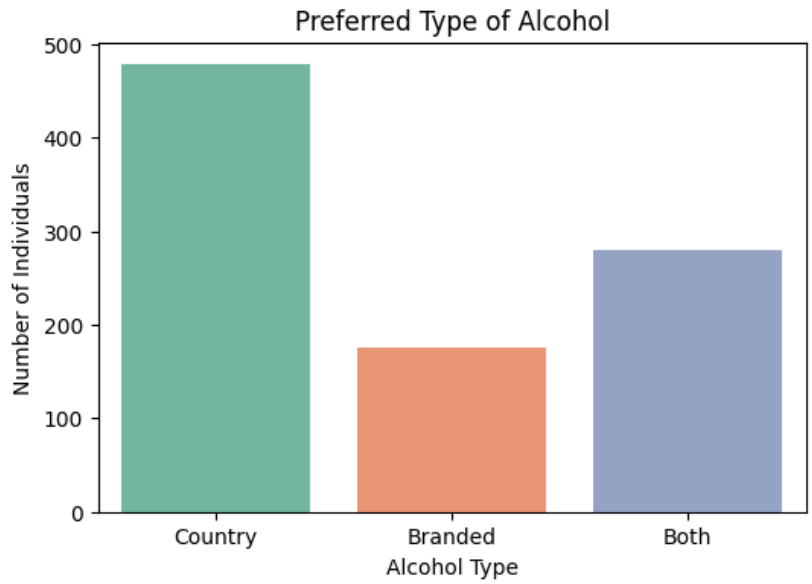

Box Plot of age
age
30
40
50
60
70
80

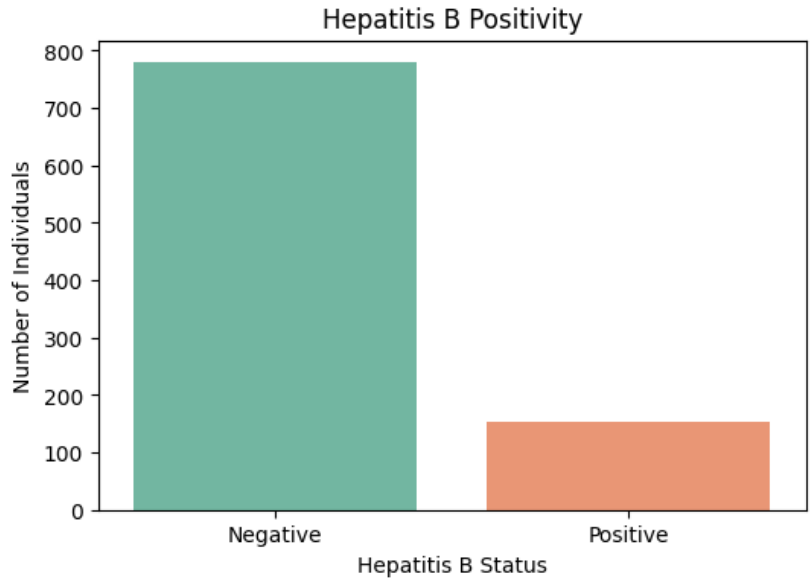

Box Plot of alcohol_consumption_years
10
20
30
40
alcohol_consumption_years

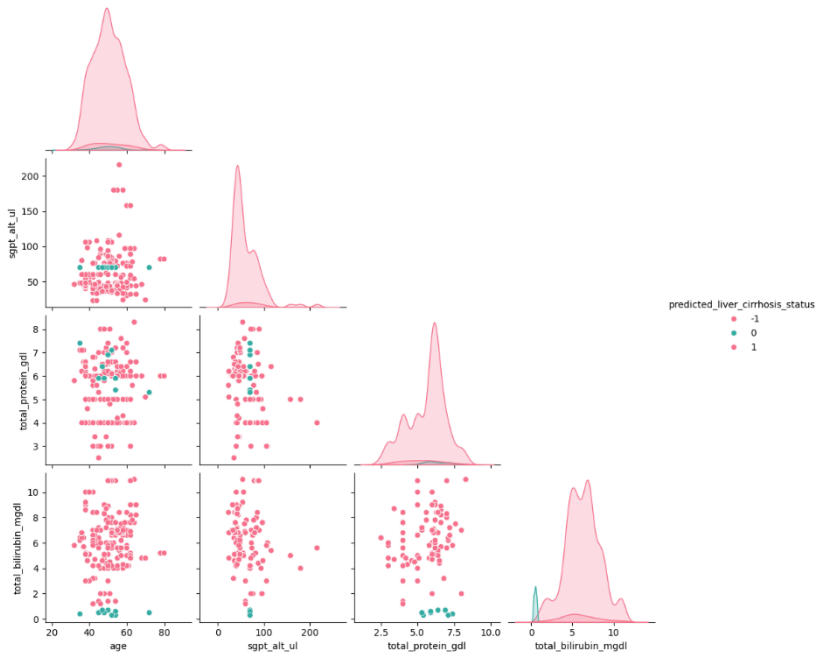
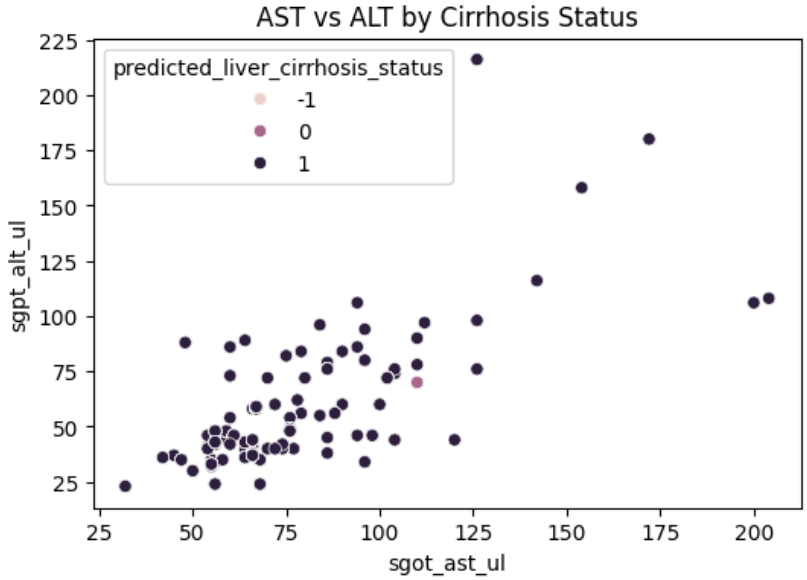

Box Plot of hemoglobin_gdl
4
6
8
10
12
14
16
hemoglobin_gdl

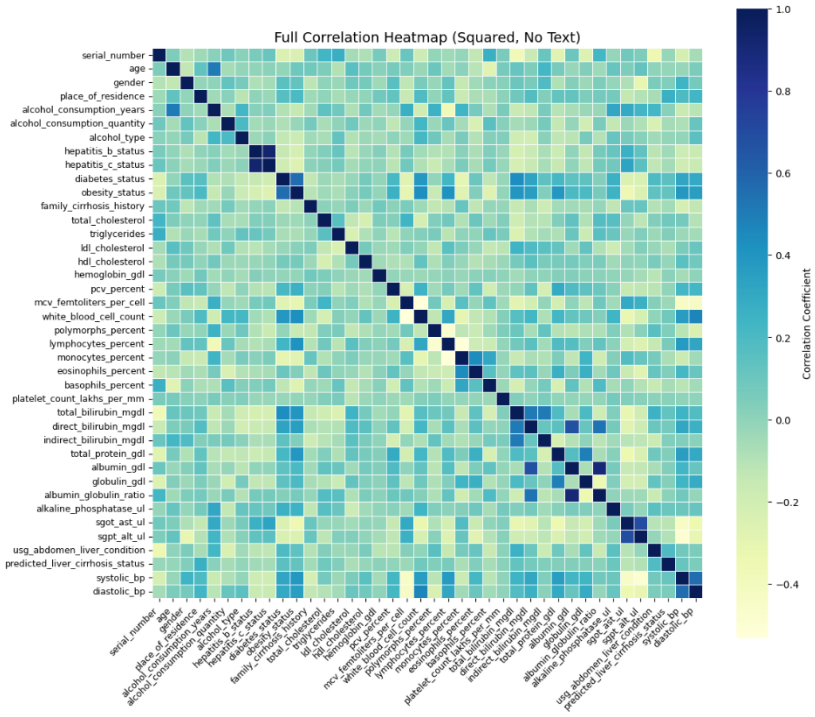

Box Plot of total_bilirubin_mgdl
0
2
4
6
8
10
total_bilirubin_mgdl


Box Plot of sgpt_alt_ul
25
50
75
100
125
150
175
200
225
sgpt_alt_ul


Preferred Type of Alcohol
Number of Individuals
500
400
300
200
100
0
Country
Branded
Both
Alcohol Type


Hepatitis B Positivity
Number of Individuals
800
700
600
500
400
300
200
100
0
Negative
Positive
Hepatitis B Status

Bivariate Analysis	 

Multivariate Analysis	
Outliers and Anomalies	-

Data Preprocessing Code Screenshots

Loading Data	

```
[7] df = pd.read_excel("HealthCareData.xlsx", sheet_name="Sheet1")
```

```
df.head()
```

	S.NO	Age	Gender	Place(location where the patient lives)	Duration of alcohol consumption(years)	Quantity of alcohol consumption (quarters/day)	Type of alcohol consumed	Hepatitis B infection	Hepatitis C infection	Diabetes Result	...	Indirect (mg/dl)	Total Protein (g/dl)	Albumin (g/dl)	Globulin (g/dl)	A/G Ratio	AL.Phosphatase (U/L)	SGOT/AST (U/L)	SGPT/ALT (U/L)	USG Abdomen (diffuse liver or not)	Predicted Value(Out Come-Patient suffering from liver cirrosis or not)
0	1	55	male	rural	12	2	branded liquor	negative	negative	YES	...	3.0	6.0	3.0	4.0	0.75	150.0	56	34	YES	YES
1	2	55	male	rural	12	2	branded liquor	negative	negative	YES	...	3.0	6.0	3.0	4.0	0.75	150.0	56	34	YES	YES
2	3	55	male	rural	12	2	branded liquor	negative	negative	YES	...	3.0	6.0	3.0	4.0	0.75	150.0	56	34	YES	YES
3	4	55	male	rural	12	2	branded liquor	negative	negative	NO	...	3.0	6.0	3.0	4.0	0.75	150.0	56	34	YES	YES
4	5	55	female	rural	12	2	branded liquor	negative	negative	YES	...	3.0	6.0	3.0	4.0	0.75	150.0	56	34	YES	YES

Handling Missing Data	

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

```
[11] # Drop columns with more than 50% missing values
     threshold = 0.5
     df = df.dropna(thresh=len(df) * threshold, axis=1)

     # Define categorical and numerical columns
     categorical_cols = df.select_dtypes(include=['object']).columns
     numeric_cols = df.select_dtypes(include=['int64', 'float64']).columns

     # Fill categorical missing values with "Unknown"
     df[categorical_cols] = df[categorical_cols].fillna("Unknown")

     # Fill numerical missing values with median
     for col in numeric_cols:
         df[col] = df[col].fillna(df[col].median())

     # Apply KNN imputation for more refined missing value handling
     imputer = KNNImputer(n_neighbors=5)
     df[numeric_cols] = imputer.fit_transform(df[numeric_cols])

     # Display final dataset info
     print(df.isnull().sum())
```

```
[12] columns_list = df.columns.tolist()
     print(columns_list)
```

```
[13] for i in columns_list:
       print(i,' \t: ',df[i].unique());
```

```
[15] gender_mapping = {'male': 1, 'female': 0, 'transgender': 2, 'female ': 0}
     df['Gender'] = df['Gender'].map(gender_mapping)
```

```
[16] df = df[df['Place(location where the patient lives)'] != 'ocun']
     place = {'rural': 1, 'urban': 0, 'Unknown': -1}
     df['Place(location where the patient lives)'] = df['Place(location where the patient lives)'].map(place)
```

```
[17] df = df.dropna(subset=['Place(location where the patient lives)'])
```

```
[18] df = df[df['Quantity of alcohol consumption (quarters/day)'] <= 10]
```

```
[19] alcohol_type = {'branded liquor':1, 'both':2, 'country liquor':0, ' branded liquor':1}
     df['Type of alcohol consumed'] = df['Type of alcohol consumed'].map(alcohol_type)
```

```
[20] test_result = {'negative':0, 'positive':1, 'Positive':1}
     df['Hepatitis B infection'] = df['Hepatitis B infection'].map(test_result)
```

```
[21] test_result = {'negative':0, 'positive':1, 'Positive':1}
     df['Hepatitis C infection'] = df['Hepatitis C infection'].map(test_result)
```

```
[22] test_result = {'NO':0, 'YES':1}
     df['Diabetes Result'] = df['Diabetes Result'].map(test_result)
```

```
[23] test_result = {'no':0, 'yes':1}
     df['Obesity'] = df['Obesity'].map(test_result)
```

```
[24] history = {'no':0, 'yes':1, 'husband':1}
     df['Family history of cirrhosis/ hereditary'] = df['Family history of cirrhosis/ hereditary'].map(history)
```

Data Transformation	[24] history = {'no':0, 'yes':1, 'husband':1} df['Family history of cirrhosis/ hereditary'] = df['Family history of cirrhosis/ hereditary'].map(history) [25] df['TG'] = df['TG'].replace("130LDL", 130) df['TG'] = pd.to_numeric(df['TG'], errors='coerce') df['TG'].fillna(df['TG'].median(), inplace=True) [26] df['LDL'] = pd.to_numeric(df['LDL'], errors='coerce') df['LDL'].fillna(df['LDL'].median(), inplace=True) [27] df['Total Bilirubin (mg/dl)'] = df['Total Bilirubin (mg/dl)'].replace("0.4", 0.4) [28] df['A/G Ratio'] = df['Albumin (g/dl)'] / df['Globulin (g/dl)'] [29] usg_result = {'no':0, 'YES':1} df['USG Abdomen (diffuse liver or not)'] = df['USG Abdomen (diffuse liver or not)'].map(usg_result) [30] result = {'no':0, 'YES':1, 'Unknown': -1} df['Predicted Value(Out Come-Patient suffering from liver cirrosis or not)'] = df['Predicted Value(Out Come-Patient suffering from liver cirrosis or not)'].map(result)
Feature Engineering	31] df[['Systolic_BP', 'Diastolic_BP']] = df['Blood pressure (mmhg)'].str.split('/', expand=True).astype(float)
Save Processed Data	[34] df.to_csv('cleaned_data.csv', index=False) df.to_excel('cleaned_data.xlsx', index=False)