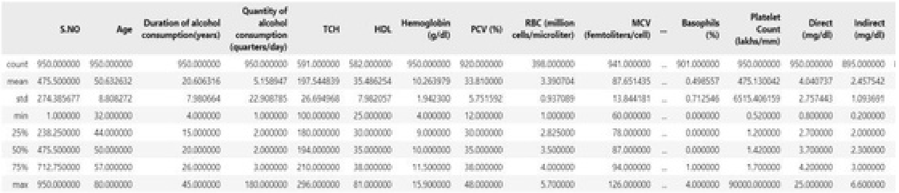
Data Collection and Preprocessing Phase

Date	٤th June ٢٠٢٤
Team ID	LTVIP٢٠٢٥TMID٤٣٩١٥
Project Title	Revolutionizing Liver Care: Predicting Liver Cirrhosis Using Advanced Machine Learning Techniques.
Maximum Marks	

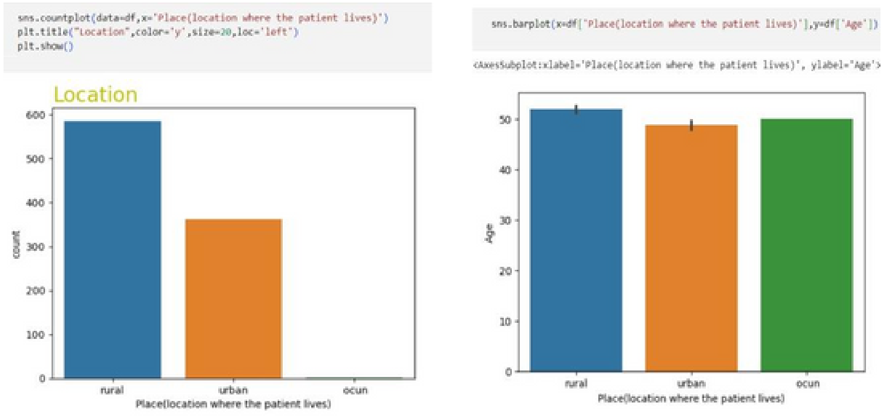
Data Exploration and Preprocessing Template

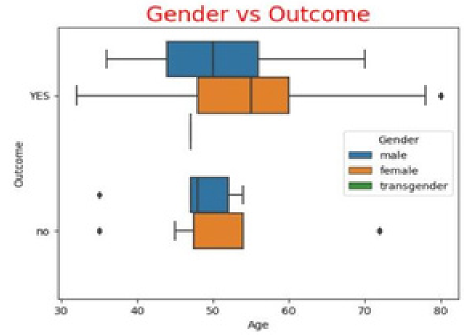
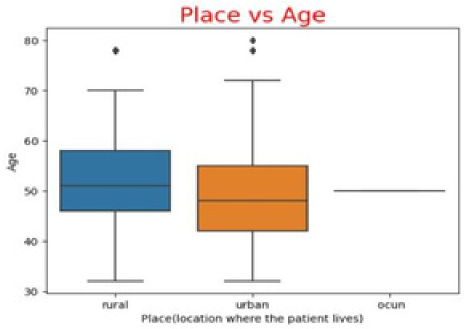
Dataset variables will be statistically analyzed to identify patterns and outliers، with Python employed for preprocessing tasks like normalization and feature engineering. Data cleaning will address missing values and outliers، ensuring quality for subsequent analysis and modeling، and forming a strong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

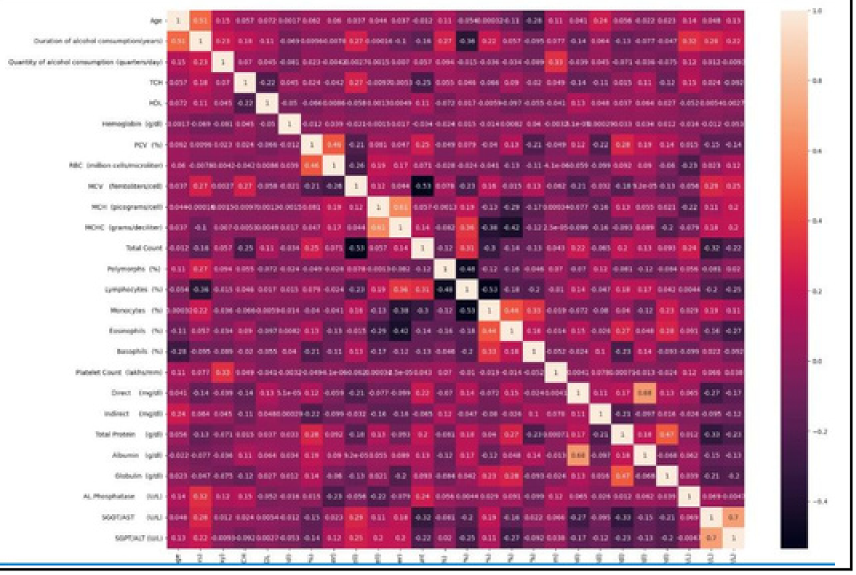
Section	Description
Data Overview	Dimension: ٩٤٩ rows ×٣٩ columns Descriptivestatistics: 

	S.NO	Age	Duration of alcohol consumption(years)	Quantity of alcohol consumption (quarters/day)	TCH	HDL	Hemoglobin (g/dl)	PCV (%)	RBC (million cells/microliter)	MCV (femtoliters/cell)	...	Basophils (%)	Platelet Count (lakhs/mm)	Direct (mg/dl)	Indirect (mg/dl)
count	950.000000	950.000000	950.000000	950.000000	591.000000	582.000000	950.000000	920.000000	398.000000	941.000000	...	901.000000	950.000000	950.000000	895.000000
mean	475.500000	50.632632	20.606316	5.158947	197.544839	35.486254	10.263979	33.810000	3.390704	87.651435	...	0.498557	475.130042	4.040737	2.457542
std	274.385677	8.808272	7.980664	22.908785	26.694968	7.982057	1.942300	5.751592	0.937089	13.844181	...	0.712546	6515.406159	2.757443	1.093691
min	1.000000	32.000000	4.000000	1.000000	100.000000	25.000000	4.000000	12.000000	1.000000	60.000000	...	0.000000	0.520000	0.800000	0.200000
25%	238.250000	44.000000	15.000000	2.000000	180.000000	30.000000	9.000000	30.000000	2.825000	78.000000	...	0.000000	1.200000	2.700000	2.000000
50%	475.500000	50.000000	20.000000	2.000000	194.000000	35.000000	10.000000	35.000000	3.500000	87.000000	...	0.000000	1.420000	3.700000	2.300000
75%	712.750000	57.000000	26.000000	3.000000	210.000000	38.000000	11.500000	38.000000	4.000000	94.000000	...	1.000000	1.700000	4.200000	3.000000
max	950.000000	80.000000	45.000000	180.000000	296.000000	81.000000	15.900000	48.000000	5.700000	126.000000	...	4.000000	90000.000000	25.000000	6.600000

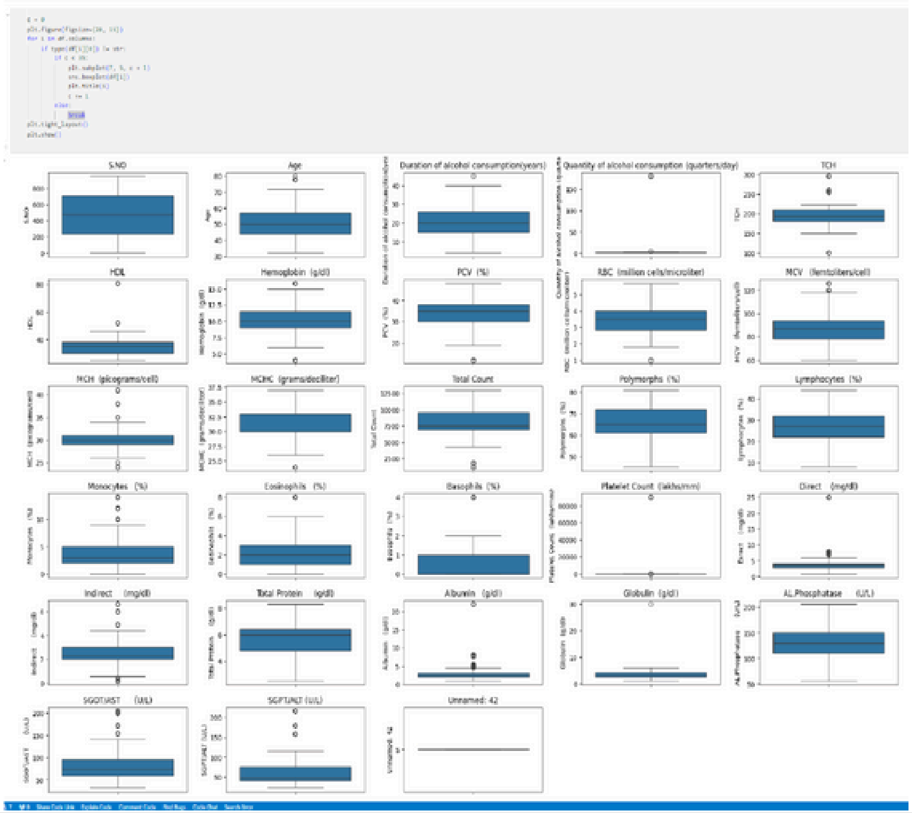
عليه السلام

Univariate Analysis	
	`sns.countplot(data=df,x='Place(location where the patient lives)')`
	`plt.title("Location",color='y',size=20,loc='left')`
	`plt.show()`
	`sns.barplot(x=df['Place(location where the patient lives)'],y=df['Age'])`
	`<AxesSubplot:xlabel='Place(location where the patient lives)', ylabel='Age'>`
	 Location
	600
	500
	400
	300
	200
	100
	0
	count
	rural
	urban
	ocun
	Place(location where the patient lives)
	50
	40
	30
	20
	10
	0
	Age
	rural
	urban
	ocun
	Place(location where the patient lives)

Bivariate Analysis	`sns.boxplot(x='Age',y='Outcome',data=df,hue='Gender')`
	`plt.title('Gender vs Outcome',color='red',size=20)`
	`plt.show()`
	
	
	`sns.boxplot(x='Place(location where the patient lives)',y='Age',data=df)`
	`plt.title('Place vs Age',color='red',size=20)`
	`Text(0.5, 1.0, 'Place vs Age')`
	

Multivariate Analysis	`plt.figure(figsize=(20,15))`
	`sns.heatmap(df.corr(),annot=True)`
	`plt.show()`
	

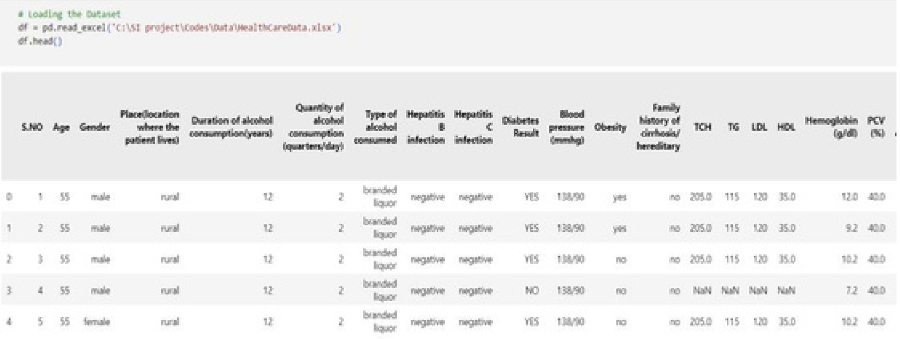
Outliers and Anomalies



```
c = 0
plt.figure(figsize=(20, 15))
for i in df.columns:
    if type(df[i][0]) != str:
        if c < 35:
            plt.subplot(7, 5, c + 1)
            sns.boxplot(df[i])
            plt.title(i)
            c += 1
        else:
            break
plt.tight_layout()
plt.show()
```

Data Preprocessing Code Screenshots

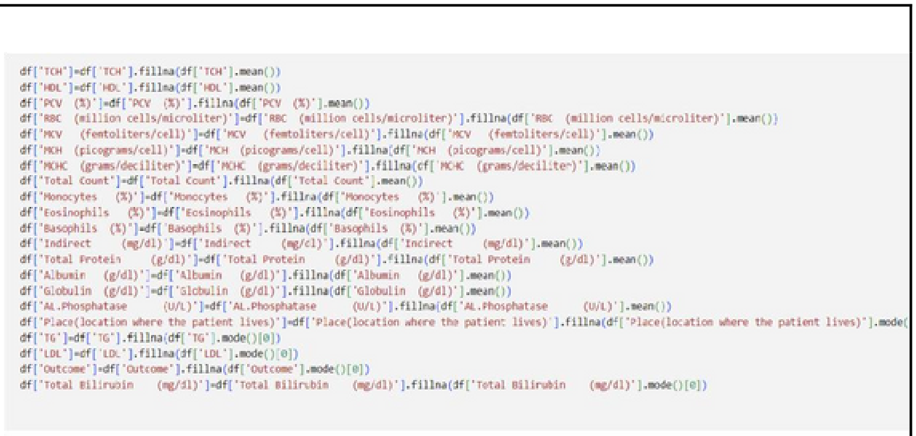
Loading Data



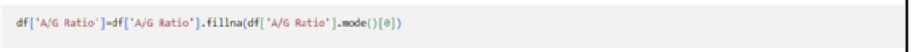
```
# Loading the Dataset
df = pd.read_excel('C:\SI project\Codes\Data\HealthCareData.xlsx')
df.head()
```

	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	Blood pressure (mmhg)	Obesity	Family history of cirrhosis/ hereditary	TCH	TG	LDL	HDL	Hemoglobin (g/dl)	PCV (%)
0	1	55	male	rural	12	2	branded liquor	negative	negative	YES	138/90	yes	no	205.0	115	120	35.0	12.0	40.0
1	2	55	male	rural	12	2	branded liquor	negative	negative	YES	138/90	yes	no	205.0	115	120	35.0	9.2	40.0
2	3	55	male	rural	12	2	branded liquor	negative	negative	YES	138/90	no	no	205.0	115	120	35.0	10.2	40.0
3	4	55	male	rural	12	2	branded liquor	negative	negative	NO	138/90	no	no	NaN	NaN	NaN	NaN	7.2	40.0
4	5	55	female	rural	12	2	branded liquor	negative	negative	YES	138/90	no	no	205.0	115	120	35.0	10.2	40.0

<table>
<tr>
<td>Handling Missing Data</td>
<td>

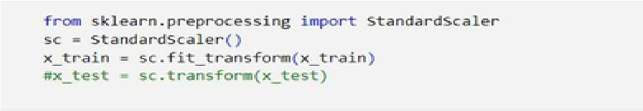


```
df['TCH']=df['TCH'].fillna(df['TCH'].mean())
df['HDL']=df['HDL'].fillna(df['HDL'].mean())
df['PCV  (%)']=df['PCV  (%)'].fillna(df['PCV  (%)'].mean())
df['RBC  (million cells/microliter)']=df['RBC  (million cells/microliter)'].fillna(df['RBC  (million cells/microliter)'].mean())
df['MCV   (femtoliters/cell)']=df['MCV   (femtoliters/cell)'].fillna(df['MCV   (femtoliters/cell)'].mean())
df['MCH  (picograms/cell)']=df['MCH  (picograms/cell)'].fillna(df['MCH  (picograms/cell)'].mean())
df['MCHC  (grams/deciliter)']=df['MCHC  (grams/deciliter)'].fillna(cf['MCHC  (grams/deciliter)'].mean())
df['Total Count']=df['Total Count'].fillna(df['Total Count'].mean())
df['Monocytes   (%)']=df['Monocytes   (%)'].fillna(df['Monocytes   (%)'].mean())
df['Eosinophils   (%)']=df['Eosinophils   (%)'].fillna(df['Eosinophils   (%)'].mean())
df['Basophils  (%)']=df['Basophils  (%)'].fillna(df['Basophils  (%)'].mean())
df['Indirect     (mg/dl)']=df['Indirect     (mg/dl)'].fillna(df['Indirect     (mg/dl)'].mean())
df['Total Protein     (g/dl)']=df['Total Protein     (g/dl)'].fillna(df['Total Protein     (g/dl)'].mean())
df['Albumin   (g/dl)']=df['Albumin   (g/dl)'].fillna(df['Albumin   (g/dl)'].mean())
df['Globulin  (g/dl)']=df['Globulin  (g/dl)'].fillna(df['Globulin  (g/dl)'].mean())
df['AL.Phosphatase      (U/L)']=df['AL.Phosphatase      (U/L)'].fillna(df['AL.Phosphatase      (U/L)'].mean())
df['Place(location where the patient lives)']=df['Place(location where the patient lives)'].fillna(df['Place(location where the patient lives)'].mode(
df['TG']=df['TG'].fillna(df['TG'].mode()[0])
df['LDL']=df['LDL'].fillna(df['LDL'].mode()[0])
df['Outcome']=df['Outcome'].fillna(df['Outcome'].mode()[0])
df['Total Bilirubin    (mg/dl)']=df['Total Bilirubin    (mg/dl)'].fillna(df['Total Bilirubin    (mg/dl)'].mode()[0])
```

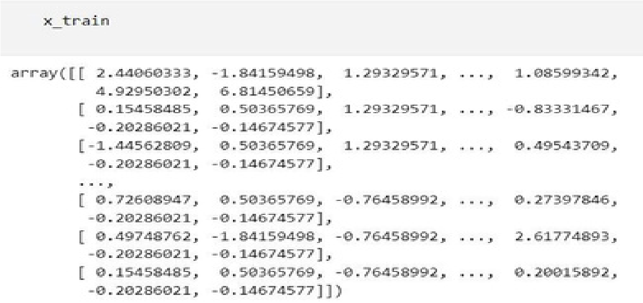


```
df['A/G Ratio']=df['A/G Ratio'].fillna(df['A/G Ratio'].mode()[0])
```

</td>
</tr>
<tr>
<td>Data Transformation</td>
<td>

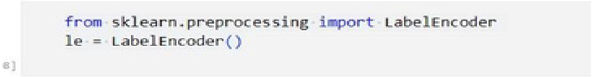


```
from sklearn.preprocessing import StandardScaler
sc = StandardScaler()
x_train = sc.fit_transform(x_train)
#x_test = sc.transform(x_test)
```

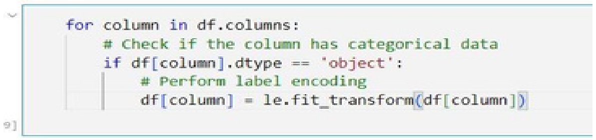


```
x_train
```

```
array([[ 2.44060333, -1.84159498,  1.29329571, ...,  1.08599342,
         4.92950302,  6.81450659],
       [ 0.15458485,  0.50365769,  1.29329571, ..., -0.83331467,
        -0.20286021, -0.14674577],
       [-1.44562809,  0.50365769,  1.29329571, ...,  0.49543709,
        -0.20286021, -0.14674577],
       ...,
       [ 0.72608947,  0.50365769, -0.76458992, ...,  0.27397846,
        -0.20286021, -0.14674577],
       [ 0.49748762, -1.84159498, -0.76458992, ...,  2.61774893,
        -0.20286021, -0.14674577],
       [ 0.15458485,  0.50365769, -0.76458992, ...,  0.20015892,
        -0.20286021, -0.14674577]])
```



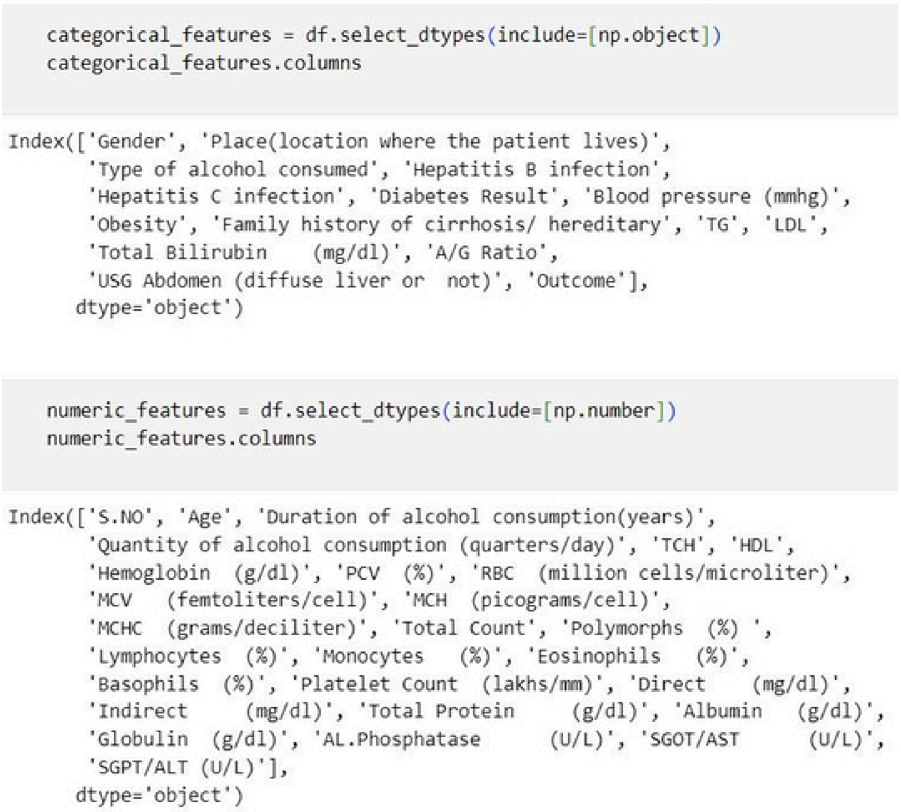
```
from sklearn.preprocessing import LabelEncoder
le = LabelEncoder()
```



```
for column in df.columns:
    # Check if the column has categorical data
    if df[column].dtype == 'object':
        # Perform label encoding
        df[column] = le.fit_transform(df[column])
```

</td>
</tr>
</table>

Feature Engineering



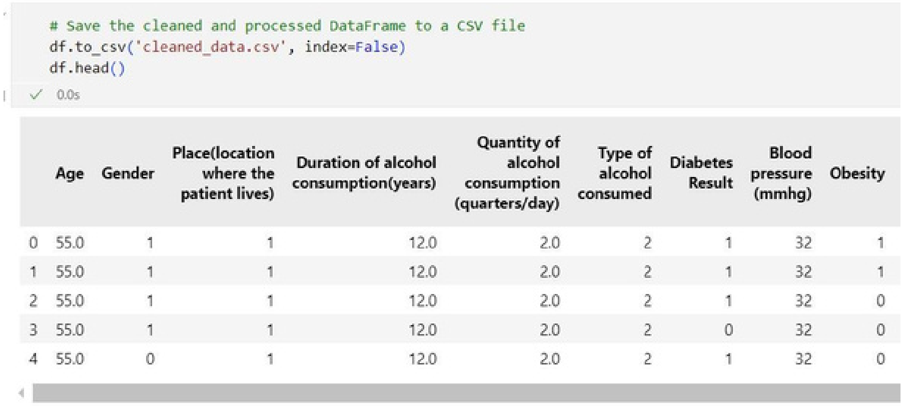
```
categorical_features = df.select_dtypes(include=[np.object])
categorical_features.columns
```

```
Index(['Gender', 'Place(location where the patient lives)',
       'Type of alcohol consumed', 'Hepatitis B infection',
       'Hepatitis C infection', 'Diabetes Result', 'Blood pressure (mmhg)',
       'Obesity', 'Family history of cirrhosis/ hereditary', 'TG', 'LDL',
       'Total Bilirubin    (mg/dl)', 'A/G Ratio',
       'USG Abdomen (diffuse liver or  not)', 'Outcome'],
      dtype='object')
```

```
numeric_features = df.select_dtypes(include=[np.number])
numeric_features.columns
```

```
Index(['S.NO', 'Age', 'Duration of alcohol consumption(years)',
       'Quantity of alcohol consumption (quarters/day)', 'TCH', 'HDL',
       'Hemoglobin  (g/dl)', 'PCV  (%)', 'RBC  (million cells/microliter)',
       'MCV   (femtoliters/cell)', 'MCH  (picograms/cell)',
       'MCHC  (grams/deciliter)', 'Total Count', 'Polymorphs  (%) ',
       'Lymphocytes  (%)', 'Monocytes   (%)', 'Eosinophils   (%)',
       'Basophils  (%)', 'Platelet Count  (lakhs/mm)', 'Direct    (mg/dl)',
       'Indirect     (mg/dl)', 'Total Protein     (g/dl)', 'Albumin   (g/dl)',
       'Globulin  (g/dl)', 'AL.Phosphatase      (U/L)', 'SGOT/AST      (U/L)',
       'SGPT/ALT (U/L)'],
      dtype='object')
```

Save Processed Data



```
# Save the cleaned and processed DataFrame to a CSV file
df.to_csv('cleaned_data.csv', index=False)
df.head()
```

✓ 0.0s

	Age	Gender	Place(location where the patient lives)	Duration of alcohol consumption(years)	Quantity of alcohol consumption (quarters/day)	Type of alcohol consumed	Diabetes Result	Blood pressure (mmhg)	Obesity
0	55.0	1	1	12.0	2.0	2	1	32	1
1	55.0	1	1	12.0	2.0	2	1	32	1
2	55.0	1	1	12.0	2.0	2	1	32	0
3	55.0	1	1	12.0	2.0	2	0	32	0
4	55.0	0	1	12.0	2.0	2	1	32	0