Liver Cirrhosis Stage Detection

1. Introduction:

The purpose of this project is to create a machine learning model capable of predicting the stage of liver cirrhosis based on medical data. The dataset used contains a range of features, including age, platelets, cholesterol, and various medical conditions like ascites, hepatomegaly, and edema, that can be used to determine the stage of liver cirrhosis in patients.

The project follows the steps of data preprocessing, feature engineering, model training, evaluation, and optimization, aiming to achieve high prediction accuracy. Multiple machine learning algorithms were tested, including Random Forest, Decision Tree, Support Vector Machine (SVM), and Naive Bayes. The best performing model will be used for making predictions.

2. Dataset Overview:

The dataset ***liver_cirrhosis.csv*** contains the following key attributes:

- **Numerical Features:**
    - ***Age***: The age of the patient in years.
    - ***Platelets***: The platelet count in the patient's blood.
    - ***Albumin***: A protein in the blood, indicating liver function.
    - ***Cholesterol***: The cholesterol level in the patient's blood.
- **Categorical Features:**
    - ***Drug***: The drug used by the patient (categorized into different types).
    - ***Sex***: The gender of the patient (Male/Female).
    - ***Ascites***: Presence or absence of ascites (fluid buildup in the abdomen).
    - ***Hepatomegaly***: Presence or absence of hepatomegaly (enlargement of the liver).
    - ***Spiders***: Presence or absence of spider-like blood vessels under the skin.
    - ***Edema***: Presence or absence of edema (swelling due to fluid retention).
    - ***Status***: Health status of the patient, categorized as 1 (healthy) or 0 (not healthy).
    - ***Stage***: The target variable, representing the stage of liver cirrhosis (ranging from 1 to 3).

3. Data Exploration & Preprocessing:

Upon loading the dataset, it was observed that there were no missing or null values in any of the columns. The next step involved encoding the categorical features (e.g., ***Drug***, ***Sex***, ***Ascites***, etc.) into numeric values using **Label Encoding**.

- **Label Encoding:** The categorical features were transformed using ***LabelEncoder()***, which converts the string labels into numeric values. For example, ***Drug***, ***Sex***, ***Ascites***, ***Hepatomegaly***, ***Spiders***, ***Edema***, and ***Status*** were label-encoded to make them compatible with machine learning algorithms.

```
le = LabelEncoder()
for column in df.select_dtypes(include=['object']).columns:
    df[column] = le.fit_transform(df[column])
```

- **Data Visualization:** Various data visualizations were created to understand the distribution of the features:
    - **Value Counts:** The distribution of the categorical features (*Drug, Sex, Ascites,* etc.) was visualized using value counts.
    - **Heatmaps & Correlation Matrix:** A heatmap of the correlation matrix was plotted to visualize the relationship between different numerical features and the target variable *Stage*.
    - **Pairplot & Scatterplots:** Pairplots were used to visualize relationships between features, and scatterplots showed the relationship between key features like *Albumin* and *Cholesterol* across different stages of liver cirrhosis.

4. Model Selection:

A variety of machine learning algorithms were trained to predict the stage of liver cirrhosis. The models evaluated were:

- **Random Forest Classifier**: This is an ensemble method that creates multiple decision trees and combines their results for prediction.
- **Decision Tree Classifier**: A basic yet interpretable model that makes decisions based on splitting the data into subsets.
- **Support Vector Machine (SVM)**: A powerful classifier that finds the optimal hyperplane to separate the classes.
- **Naive Bayes Classifier**: A probabilistic model based on Bayes' Theorem, used for classification problems.

Each model was trained using the training data, and their performance was evaluated using the test data.

- **Training & Testing Split:** The data was split into 80% training and 20% testing sets using ***train_test_split*** from ***sklearn***:

```
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random_state=42)
```

- **Model Training and Evaluation:** The models were trained using their respective ***fit*** methods, and their accuracy was evaluated using ***accuracy_score***, ***classification_report***, and visualizations such as confusion matrices.

Example for Random Forest Classifier:

```
model = RandomForestClassifier()
model.fit(X_train, y_train)
training_score = model.score(X_train, y_train)
testing_score = model.score(X_test, y_test)
```

5. Hyperparameter Optimization:

To improve the performance of the Random Forest model, hyperparameter tuning was performed using **RandomizedSearchCV**. The hyperparameters optimized were:

- ***n_estimators***: The number of trees in the forest.
- ***max_depth***: The maximum depth of the tree.
- ***min_samples_split***: The minimum number of samples required to split an internal node.
- ***min_samples_leaf***: The minimum number of samples required to be at a leaf node.

```
rf_grid = {"n_estimators": np.arange(10, 100, 50),
           "max_depth": [None, 3, 5, 10],
           "min_samples_split": np.arange(2, 20, 2),
           "min_samples_leaf": np.arange(1, 20, 2)}

rs_rf = RandomizedSearchCV(RandomForestClassifier(), param_distributions=rf_grid, cv=5,
rs_rf.fit(X_train, y_train)
```

After hyperparameter optimization, the best model configuration was determined and its accuracy was evaluated on the test set.

6. Final Model and Evaluation:

After training several models and performing hyperparameter tuning, **Random Forest** was chosen as the final model due to its high performance.

- **Model Evaluation:** The final model was evaluated using accuracy scores, precision, recall, and F1-score. Confusion matrices were used to visualize misclassifications, and feature importances were plotted to identify the most influential features in predicting the stage of liver cirrhosis.

```
feature_importances = Final_model.feature_importances_
indices = np.argsort(feature_importances)
plt.barh(range(len(indices)), feature_importances[indices], align='center')
```

- **Confusion Matrix:** A confusion matrix was visualized to assess how well the model performed across different classes of liver cirrhosis stages:

```
sns.heatmap(confusion_matrix(y_test, preds), annot=True)
```

7. Model Deployment:

The final model was serialized and saved to disk using ***pickle*** to facilitate its future use:

```
pickle.dump(Final_model, open("Liver_Cirrhosis_Stage_Detection.pkl", "wb"))
```

8. Predictions & Final Output:

The model was used to make predictions on the test set, and the predictions were visualized in various ways, including scatter plots and heatmaps. A final prediction DataFrame was created showing the predicted stages alongside features like ***Age*** and ***Platelets*** to give a deeper understanding of how the model predicts liver cirrhosis stages.

```
predictions1 = pd.DataFrame()
predictions1["Platelets"] = X_test["Platelets"]
predictions1["Stage"] = preds
```

9. Conclusion & Future Work:

The project successfully developed a machine learning model to predict the stage of liver cirrhosis, achieving good performance across multiple evaluation metrics. The model can assist healthcare professionals in diagnosing the disease at various stages, enabling early intervention and better treatment planning.

- **Future Enhancements:**
    - Exploring additional models like XGBoost or Neural Networks.
    - Performing further feature engineering to improve model accuracy.
    - Expanding the dataset to improve generalizability.