Assignment No - 4

Title - Develop a Machine learning based Recommendation System for breast cancer prognosis

Objectives-

- To develop a machine learning-based recommendation system that assists in breast cancer prognosis by analyzing patient data, identifying patterns, and recommending potential treatment options or risk assessments.
- The aim is to improve prognosis accuracy, facilitate early diagnosis, and personalize treatment plans based on predictive analytics.

Outcomes-

- Students learn to clean and preprocess data, addressing missing values and normalizing features.
- Students gain hands-on experience in building and training machine learning models.
- Students learn to assess model performance using metrics like accuracy and F1-score.
- Students apply theoretical
- concepts from coursework in a practical setting.
- Students explore how predictions can inform personalized treatment options.
- Students discuss ethical issues related to data privacy and bias in medical data.
- Students develop critical thinking and troubleshooting skills throughout the project.

Theory-

Breast cancer is one of the most prevalent cancers among women worldwide. Prognosis varies significantly based on various factors such as tumor size, grade, lymph node involvement, and patient age. Machine learning techniques can analyze complex datasets to identify patterns that may not be evident through traditional statistical methods. This project aims to harness these techniques to develop a robust prognostic model that can recommend treatment pathways based on individual patient profiles.

Data Collection

Dataset Selection

- Source: Use publicly available datasets such as the Breast Cancer Wisconsin (Diagnostic) Data Set from the UCI Machine Learning Repository or the METABRIC dataset from cBioPortal.
- Features: The dataset should include features relevant to breast cancer prognosis, such as:
  - Age
  - Tumor size
  - Tumor grade
  - Hormone receptor status (ER, PR, HER2)
  - Lymph node involvement
  - Genetic markers
  - Patient demographics

Data Preprocessing

- Data Cleaning: Handle missing values, remove duplicates, and normalize the data.
- Feature Selection: Utilize techniques such as correlation analysis and recursive feature elimination to select significant features.
- Data Splitting: Split the dataset into training (70%), validation (15%), and testing (15%) subsets.

Model Development

Choice of Algorithms

Select appropriate machine learning algorithms based on the nature of the data and the prediction task. Possible algorithms include:

- Logistic Regression: For binary classification (e.g., prognosis: good vs. poor).
- Decision Trees: To understand feature importance and decision rules.
- Random Forest: To improve prediction accuracy through ensemble learning.
- Support Vector Machines (SVM): For classification tasks.
- Neural Networks: To capture complex patterns in the data.

Model Training and Evaluation
- Training: Train the models using the training dataset. Use techniques such as cross-validation to optimize hyperparameters.
- Evaluation Metrics: Evaluate the model performance using metrics like accuracy, precision, recall, F1-score, and area under the ROC curve (AUC).

Implementation of Recommendation System

Development of the Recommendation Engine
- Recommendation Logic: Based on the model predictions, design a system that recommends treatment plans based on risk stratification. For example:
- High-risk patients may be recommended more aggressive treatment options.
- Low-risk patients may be monitored with less intensive treatment.

User Interface
- Develop a simple user interface (UI) to input patient data and receive prognostic recommendations. This could be a web-based application using Flask or Streamlit.

Ethical Considerations
- Patient Privacy: Ensure compliance with ethical standards and regulations (e.g., HIPAA) concerning patient data.
- Bias Mitigation: Assess the model for biases based on demographics and strive to ensure fair treatment recommendations across diverse patient populations.

Conclusion -
In conclusion, the development of a machine learning-based recommendation system for breast cancer prognosis can significantly enhance the decision-making process in clinical settings. By utilizing advanced predictive modeling techniques, this system can offer personalized treatment recommendations, ultimately leading to improved patient outcomes and quality of care.

```python
# Step 1: Importing necessary libraries
import pandas as pd
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, classification_report,
confusion_matrix
from sklearn.datasets import load_breast_cancer
import numpy as np

# Step 2: Load the Breast Cancer dataset (UCI built-in)
data = load_breast_cancer()
df = pd.DataFrame(data.data, columns=data.feature_names)
df['target'] = data.target   # 0 = malignant, 1 = benign

print("Dataset loaded successfully.")
print("Shape:", df.shape)
print("\nSample data:\n", df.head())

# Step 3: Data Preprocessing
# (Dataset already cleaned, normalized — no missing values)
# Additional preprocessing could include normalization or feature
selection.

# Step 4: Splitting the data into train and test sets
X = df.drop(columns='target')
y = df['target']
X_train, X_test, y_train, y_test = train_test_split(X, y,
test_size=0.2, random_state=42)

print("\nTraining samples:", X_train.shape[0])
print("Testing samples:", X_test.shape[0])

# Step 5: Train the model (Random Forest)
model = RandomForestClassifier(n_estimators=100, random_state=42)
model.fit(X_train, y_train)

# Step 6: Evaluate the model
y_pred = model.predict(X_test)
accuracy = accuracy_score(y_test, y_pred)

print("\n--- Model Evaluation ---")
print("Model Accuracy: {:.2f}%".format(accuracy * 100))
print("\nClassification Report:\n", classification_report(y_test,
y_pred, target_names=["Malignant", "Benign"]))
print("Confusion Matrix:\n", confusion_matrix(y_test, y_pred))

# Step 7: Recommendation system for prognosis
def prognosis_recommendation(features):
    """
    Provides a prognosis recommendation based on model predictions.
```

```
    :param features: Array of patient features
    :return: String recommendation
    """
    prediction = model.predict([features])
    if prediction[0] == 0:
        return "⚠ High risk of malignant cancer. Immediate
consultation and further diagnostic tests recommended."
    else:
        return "✅ Benign prognosis. Routine monitoring and regular
check-ups suggested."

# Step 8: Example use case — test with one patient sample
example_patient = X_test.iloc[0].values
recommendation = prognosis_recommendation(example_patient)

print("\n--- Example Prediction ---")
print("Actual Label:", "Malignant" if y_test.iloc[0] == 0 else
"Benign")
print("Predicted Label:", "Malignant" if
model.predict([example_patient])[0] == 0 else "Benign")
print("Recommendation:", recommendation)
```

```
Dataset loaded successfully.
Shape: (569, 31)

Sample data:
    mean radius  mean texture  mean perimeter  mean area  mean
smoothness  \
0        17.99         10.38          122.80     1001.0
0.11840
1        20.57         17.77          132.90     1326.0
0.08474
2        19.69         21.25          130.00     1203.0
0.10960
3        11.42         20.38           77.58      386.1
0.14250
4        20.29         14.34          135.10     1297.0
0.10030

   mean compactness  mean concavity  mean concave points  mean
symmetry  \
0           0.27760          0.3001              0.14710
0.2419
1           0.07864          0.0869              0.07017
0.1812
2           0.15990          0.1974              0.12790
0.2069
3           0.28390          0.2414              0.10520
0.2597
4           0.13280          0.1980              0.10430
```

```
0.1809

   mean fractal dimension  ...  worst texture  worst perimeter  worst
area  \
0                 0.07871  ...          17.33           184.60
2019.0
1                 0.05667  ...          23.41           158.80
1956.0
2                 0.05999  ...          25.53           152.50
1709.0
3                 0.09744  ...          26.50            98.87
567.7
4                 0.05883  ...          16.67           152.20
1575.0

   worst smoothness  worst compactness  worst concavity  worst concave
points  \
0            0.1622             0.6656           0.7119
0.2654
1            0.1238             0.1866           0.2416
0.1860
2            0.1444             0.4245           0.4504
0.2430
3            0.2098             0.8663           0.6869
0.2575
4            0.1374             0.2050           0.4000
0.1625

   worst symmetry  worst fractal dimension  target
0          0.4601                  0.11890       0
1          0.2750                  0.08902       0
2          0.3613                  0.08758       0
3          0.6638                  0.17300       0
4          0.2364                  0.07678       0

[5 rows x 31 columns]

Training samples: 455
Testing samples: 114

--- Model Evaluation ---
Model Accuracy: 96.49%

Classification Report:
              precision    recall  f1-score   support

   Malignant       0.98      0.93      0.95        43
      Benign       0.96      0.99      0.97        71

    accuracy                           0.96       114
```

```
   macro avg       0.97      0.96      0.96       114
weighted avg       0.97      0.96      0.96       114

Confusion Matrix:
 [[40  3]
 [ 1 70]]

--- Example Prediction ---
Actual Label: Benign
Predicted Label: Benign
Recommendation:  Benign prognosis. Routine monitoring and regular
check-ups suggested.

/usr/local/lib/python3.12/dist-packages/sklearn/utils/
validation.py:2739: UserWarning: X does not have valid feature names,
but RandomForestClassifier was fitted with feature names
  warnings.warn(
/usr/local/lib/python3.12/dist-packages/sklearn/utils/validation.py:27
39: UserWarning: X does not have valid feature names, but
RandomForestClassifier was fitted with feature names
  warnings.warn(
```