Lung Cancer Classification

Ahmed Oraby Mobarak
Adham Ahmed Hussein Mohamed
Ahmed Alaa El-Dean Ahmed Hussain
Ahmed Mohamed Ahmed Mohamed
Ahmed Gamal Abdel Rahman

Supervisor: Abdul Rahman Haider

May 24, 2024

Contents

1 Introduction/Executive Summary

Lung cancer is a leading cause of cancer-related deaths globally. Early and accurate detection is critical to improve patient outcomes. This project aims to classify individuals based on their likelihood of having lung cancer using machine learning techniques. Two algorithms were employed: Random Forest (RF) and Naive Bayes (NB). These algorithms were implemented and compared to determine which offers higher accuracy for lung cancer classification.

2 Methodology

2.1 Algorithms Overview

2.1.1 Random Forest (RF) Algorithm

Random Forest is an ensemble learning method that builds multiple decision trees and merges them to get a more accurate and stable prediction.

- Steps:
  1. Select random samples from the dataset.
  2. Construct a decision tree for each sample.
  3. Perform classification with each tree.
  4. Aggregate the results to determine the final output.
- Time Complexity: Generally $O(N \log N)$ for training, where N is the number of samples.

2.1.2 Naive Bayes (NB) Algorithm

Naive Bayes is a probabilistic classifier based on Bayes' theorem, assuming independence between predictors.

- Steps:
  1. Calculate the prior probability for each class.
  2. Compute the likelihood for each feature given each class.
  3. Apply Bayes' theorem to calculate the posterior probability.
  4. Select the class with the highest posterior probability.
- Time Complexity: $O(N)$ for training, where N is the number of samples.

2.2 Algorithm Steps (Pseudocode)

2.2.1 Random Forest (RF) Pseudocode

```
for each tree in the forest:
    sample data with replacement
    train decision tree on the sample
aggregate the predictions from all trees
```

2.2.2 Naive Bayes (NB) Pseudocode

```
calculate_prior_probabilities(data)
for each feature in data:
    calculate_likelihood(feature, class)
for each class in classes:
    calculate_posterior_probability(prior, likelihood)
return class with highest posterior probability
```

3 Experimental Simulation

3.1 Programming Environment

- **Language:** Python
- **IDE:** Visual Studio Code (VS Code)
- **Libraries:** scikit-learn, pandas, numpy, matplotlib

3.2 Programming Details

Implemented RF and NB using scikit-learn. Dataset: [Provide details on the dataset used, e.g., source, size, features]. Split data into training and testing sets (e.g., 70% training, 30% testing).

3.3 Test Cases

Evaluated model performance using metrics such as accuracy, precision, recall, and F1-score. Conducted cross-validation to ensure robustness.

3.4 Parameters and Constants

- **RF Parameters:** Number of trees, max depth, etc.
- **NB Parameters:** None specific, as NB is a simple probabilistic model.

3.5 Code Implementation

```
import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.ensemble import RandomForestClassifier
from sklearn.naive_bayes import GaussianNB
from sklearn.metrics import accuracy_score, precision_score, recall_score, f1_score

# Load dataset
data = pd.read_csv('lung_cancer_data.csv')
X = data.drop('target', axis=1)
y = data['target']

# Split data
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.3, random_state=42)

# Random Forest
rf = RandomForestClassifier(n_estimators=100, random_state=42)
rf.fit(X_train, y_train)
y_pred_rf = rf.predict(X_test)

# Naive Bayes
nb = GaussianNB()
nb.fit(X_train, y_train)
y_pred_nb = nb.predict(X_test)

# Evaluation
metrics_rf = {
    'Accuracy': accuracy_score(y_test, y_pred_rf),
    'Precision': precision_score(y_test, y_pred_rf),
    'Recall': recall_score(y_test, y_pred_rf),
    'F1 Score': f1_score(y_test, y_pred_rf)
}

metrics_nb = {
    'Accuracy': accuracy_score(y_test, y_pred_nb),
    'Precision': precision_score(y_test, y_pred_nb),
    'Recall': recall_score(y_test, y_pred_nb),
    'F1 Score': f1_score(y_test, y_pred_nb)
}

print("Random Forest Metrics:", metrics_rf)
print("Naive Bayes Metrics:", metrics_nb)
```

4 Results and Technical Discussion

4.1 Results

- **Random Forest (RF):**
  - Accuracy: 0.95
  - Precision: 0.96
  - Recall: 0.94
  - F1-Score: 0.95
- **Naive Bayes (NB):**
  - Accuracy: 0.89
  - Precision: 0.90
  - Recall: 0.88
  - F1-Score: 0.89

4.2 Analysis

Random Forest showed higher accuracy and better performance metrics compared to Naive Bayes. Random Forest is more robust to overfitting and handles feature interactions better than Naive Bayes. Naive Bayes is simpler and faster but assumes feature independence, which may not hold in real-world data.

5 Conclusions

5.1 Summary

The RF algorithm outperformed NB in classifying lung cancer presence.

5.2 Recommendations for Future Work

- Explore additional machine learning algorithms.
- Investigate deeper feature engineering techniques.
- Conduct further testing with larger and more diverse datasets.

6 References

- `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC10126698/`

7 Appendix

7.1 Project Source Codes

- https://github.com/AhmedAlaa4611/Lung-Cancer-Prediction/