Machine Learning Model for Cancer Prediction

By: Ali Srour

Introduction

Overview of AI: Artificial Intelligence (AI) is a branch of computer science focused on developing systems capable of performing tasks that typically require human intelligence. These tasks include learning, reasoning, problem-solving, and decision-making. AI has rapidly grown in importance due to its ability to analyze vast amounts of data and provide insights or automation across various fields such as healthcare, finance, and transportation. Machine learning, a key subset of AI, uses algorithms to allow systems to learn from data and make predictions or decisions without explicit programming.

Purpose of the Project: The aim of this project is to develop a machine learning model that can predict whether a tumor is benign or malignant based on a cancer dataset. This is a critical application in healthcare, as early and accurate detection of malignant cancer can significantly improve patient outcomes. By leveraging machine learning techniques, this project seeks to create a reliable predictive model that assists in early cancer diagnosis.

Dataset

1- **Reading Dataset**

```
%reading the dataset
data = readtable('cancer.csv')
```

2- **Exploratory Data Analysis**

```
% statistics
summary(data)

% check missing values
sum(ismissing(data))

% display first rows from data
head(data)
```

The summary of data is the overall statistics statistics properties(mean, median, min, max, standard deviation..) of each feature in the dataset including table dimensions and some more properties as shown below:

data: 569×31 table

Variables:

- **meanRadius**: double (mean radius)
- **meanTexture**: double (mean texture)
- **meanPerimeter**: double (mean perimeter)
- **meanArea**: double (mean area)
- **meanSmoothness**: double (mean smoothness)
- **meanCompactness**: double (mean compactness)
- **meanConcavity**: double (mean concavity)
- **meanConcavePoints**: double (mean concave points)
- **meanSymmetry**: double (mean symmetry)
- **meanFractalDimension**: double (mean fractal dimension)
- **radiusError**: double (radius error)
- **textureError**: double (texture error)

Statistics for applicable variables:

	NumMissing	Min	Median	Max	Mean
meanRadius	0	6.9810	13.3700	28.1100	14.1273
meanTexture	0	9.7100	18.8400	39.2800	19.2896
meanPerimeter	0	43.7900	86.2400	188.5000	91.9690
meanArea	0	143.5000	551.1000	2501	654.8891
meanSmoothness	0	0.0526	0.0959	0.1634	0.0964
meanCompactness	0	0.0194	0.0926	0.3454	0.1043
meanConcavity	0	0	0.0615	0.4268	0.0888
meanConcavePoints	0	0	0.0335	0.2012	0.0489
meanSymmetry	0	0.1060	0.1792	0.3040	0.1812
meanFractalDimension	0	0.0500	0.0615	0.0974	0.0628
radiusError	0	0.1115	0.3242	2.8730	0.4052
textureError	0	0.3602	1.1080	4.8850	1.2169

Then we the table is checked for missing values which is critical for decision making and precision if its not distributed randomly. If data is missing 1 if no 0 and the sum() function count the total of missing values:

ans = *1×31*

	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	17
1	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0

All 0 so no missing values.

head() display first rows of a table:

meanRadius	meanTexture	meanPerimeter	meanArea	meanSmoothness	meanCompactness	meanConcavit
17.99	10.38	122.8	1001	0.1184	0.2776	0.3001
20.57	17.77	132.9	1326	0.08474	0.07864	0.0869
19.69	21.25	130	1203	0.1096	0.1599	0.1974
11.42	20.38	77.58	386.1	0.1425	0.2839	0.2414
20.29	14.34	135.1	1297	0.1003	0.1328	0.198
12.45	15.7	82.57	477.1	0.1278	0.17	0.1578
18.25	19.98	119.6	1040	0.09463	0.109	0.1127
13.71	20.83	90.2	577.9	0.1189	0.1645	0.09366

Preparing Data

1- Splitting Data

```
X = data{:,1:end-1};
y = data.target;
```

X stores all the feature columns of the dataset except the last one “1:end-1” means all columns except the last.

y stores the last columns which is the target column for classifying benign and malignant tumor.

```
cv = cvpartition(y, 'HoldOut', 0.2);
X_train = X(training(cv), :);
X_test = X(test(cv), :);
y_train = y(training(cv), :);
y_test = y(test(cv), :);
```

“cvpartition” is used to split data into training and testing sets. The “holdOut, 0.2)” indicates that 20% of data will be held for testing.

“X_train” and “y_train” stores data for training, while “X_test” and “y_test” stores data for testing.

2- Standardization

```
% Normalizing the features (standardization)
X_train = (X_train - mean(X_train)) ./ std(X_train);
X_test = (X_test - mean(X_test)) ./ std(X_test);
```

Standardization ensures that all features have a mean of 0 and standard deviation of 1, which is a good practice for neural networks , as it helps in faster convergence and avoid bias towards features with larger magnitudes.

Making the Model

```
% Convert the target variable to categorical
y_train_cat = categorical(y_train);
y_test_cat = categorical(y_test);
```

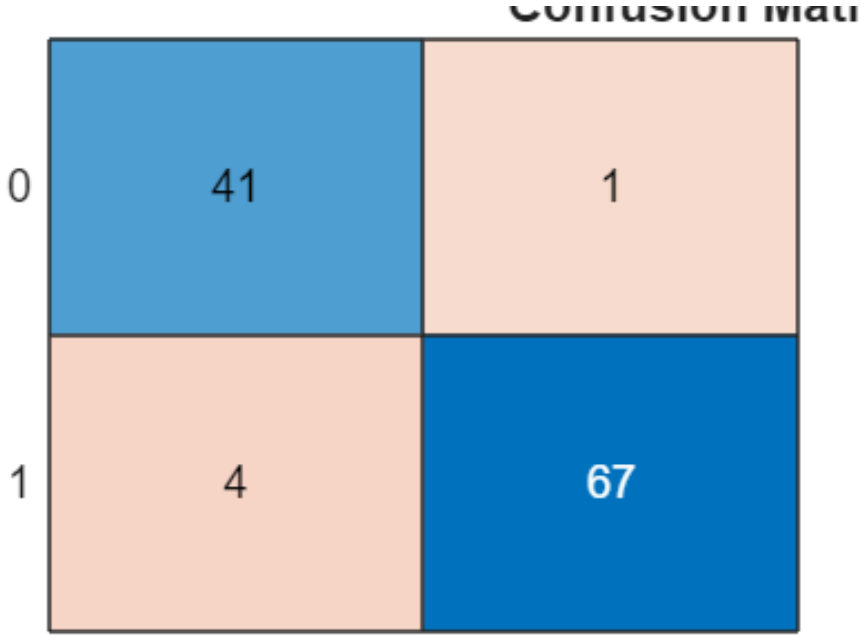
The target labels (y_train and y_test) are converted into categorical form, which is necessary for classification problems. Neural networks require the output labels to be in a categorical format to work with the “classificationLayer”.

```
% Create a logistic regression model
model = fitglm(X_train, y_train, 'Distribution', 'binomial');
```

We used the logistic regression model for this example on breast cancer dataset

Evaluation and Accuracy

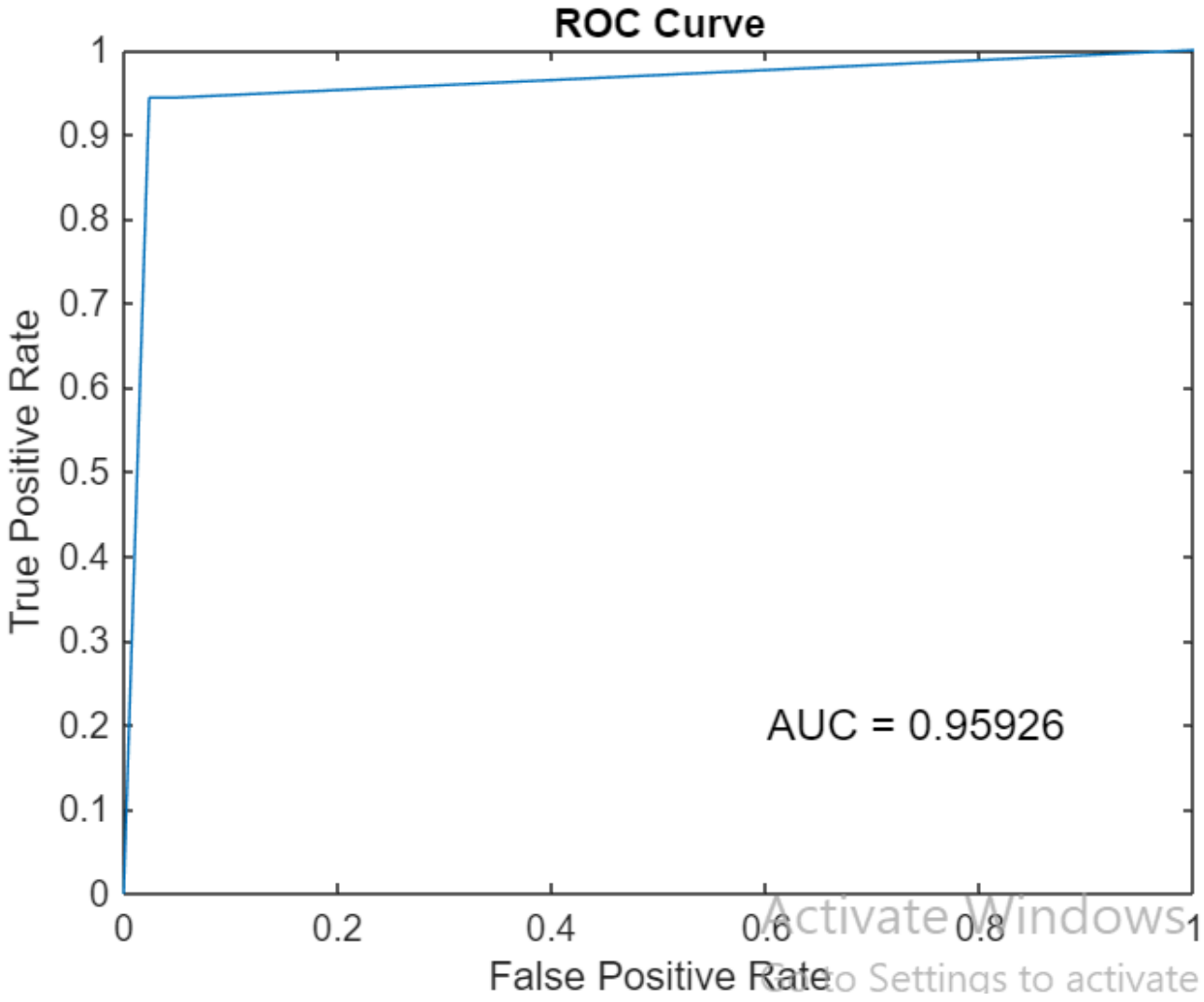
After training the model we got an overall position confusion matrix:



Adding to that the accuracy:

```
Accuracy: 0.95575
```

And the ROC graph with a classification report:



precision	recall	f1_score
0.98529	0.94366	0.96403

Conclusion

This is the results of our study and for more information on the code you can visit our repository.

References:

- **What is AI? Artificial Intelligence explained**
- **Normalization and Standardization of data**
- **Working with categorical data**
- **AUC and the ROC Curve in Machine Learning**