BREAST CANCER PROGNOSIS

By

SURBHI KATHIRIYA
17BIT035

DEVANSHU MATHUR
17BIT016

DEPARTMENT OF INFORMATION TECHNOLOGY
Ahmedabad 382481

BREAST CANCER PROGNOSIS

Mini Project - II

Submitted in fulfillment of the requirements

For the degree of

Bachelor of Technology in Information Technology

By

SURBHI KATHIRIYA
17BIT035

DEVANSHU MATHUR
17BIT016

Guided By
Dr. Swati Jain
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

DEPARTMENT OF INFORMATION TECHNOLOGY
Ahmedabad 382481

CERTIFICATE

This is to certify that the project entitled "BREAST CANCER PROGNOSIS" submitted by SURBHI KATHIRIYA (17BIT035) and DEVANSHU MATHUR (17BIT016), towards the partial fulfillment of the requirements for the degree of Bachelor of Technology in Information Technology of Nirma University is the record of work carried out by him/her under my supervision and guidance. In my opinion, the submitted work has reached a level required for being accepted for examination.

Dr. Swati Jain
Associate Professor
Computer Science and Engineering,
Institute Of Technology
Nirma University,
Ahmedabad

Dr. Madhuri Bhavsar
Professor and Head of Department
Computer Science and Engineering,
Institute of Technology,
Nirma University,
Ahmedabad

ACKNOWLEDGEMENT

We would like to express our deepest gratitude to all those who provided us the opportunity to complete this report. We acknowledge with thanks, the guidance provided by Dr. Swati Jain, under whose counsel we were able to complete the task with a promising result. This report could not have been possible without the coordination of the team members Surbhi Kathiriya and Devanshu Mathur. We also acknowledge the excellent suggestions given by our friends to further improve the content of the report.

At the home, we are purely thankful to our family members for the support and encouragement provided by them in completing this report.

ABSTRACT/ Outline

This project aims at constructing a model using Machine Learning Techniques which can predict the survival time of the patients showing the positive symptoms of Breast Cancer by learning from their diagnostic data including symptoms, tumour size and the hormonal balance etc.

These types of machines are very helpful in present day Medical Institutes so that the patient gets the best and Quick treatment according to his/her bodily matabolisms and the amount of the spread of the disease.

CONTENTS

1. Introduction

1.1. General

This document reports all the elements which are requiered in our project “BREAST CANCER PROGNOSIS”. This report also specifies the Scope of the work together with the implementation of each and every component. Gradually, it shows the model constructed using Python and Scikit-Learn.

1.2. Scope of Work

The project is constructed to learn from the Database provided and then predict the Survival months of the patients using Machine Learning Techniques. The data fetched is first preprocessed and then divided for testing and training after that the model predicts the survival months.

2. Literature Survey

2.1. General

The Literature survey covered papers and books that discussed Different Machine Learning Techniques for Regression Model. Papers that focused on Breast Cancer Detection and Prognosis, eventually those papers were chosen that focused on the topics discussed above.

3. SEER Database Analysis

In our project we used the SEER database for training and testing our model. This database contains approx. 4000 entries with 15 different diagnostic features.

SEER (Surveillance, Epidemiology, and End Results) program of National Cancer Institute collects cancer data from population-based cancer registries covering approximately 34.6 percent of the U.S. population. We will look at some of the important features in detail.

3.1. T-Stage

It tells us about the Size and Location of Tumor.

T1 : This means tumor size in the breast is about 20 millimeters.

T2 : This means tumor size is in between 20 mm to 50 mm.

T3 : This means tumor size is more than 50 mm.

T4 : This means tumor has spread to the chest wall.

3.2. N-Stage

It describes about the amount of lymph nodes affected by the tumor.

N0 : No lymph nodes are affected by the cancer.

N1 : It says that cancer has spread to 1 to 3 lymph nodes.

N2 : The cancer has spread to 4 to 9 lymph nodes.

N3 : The cancer has spread to the lymph node of collarbone.

3.3. Grade

Grade describes how the cells look like under a microscope. Lower grade describes slow growth of cancer and higher grade

describes fast growth of the cancer. Grading is distributed in three type.

Grade I : Cancer cells are looking like normal cells and their growth is not rapid.

Grade II : Cancer cells are not looking like normal cells and their growth is faster in compare to normal cells.

Grade III : Cancer cells that are looks abnormal and their growth is aggressive.

3.4. Estrogen and progesterone Status

Some proteins in the cells are called receptors that can attach to certain substances in the blood. Normal breast cells and some breast cancer cells have these receptors which are attach to the hormones estrogen and progesterone. And they depend on these hormones to grow. Breast cancer cells may or may not have these cells.

ER : Positive/Negative

NR : Positive/Negative

ER status and NR status are important because it helps doctor to decide whether hormonal therapy will respond or other therapy will respond.

3.5. Regional nodes examined

This data contains the total number of regional lymph nodes removed or examined.

4. Methedology

In this paper we are going to investigate several regression models and then find the best one out of them using the negative mean squared error and will use that model to finally train and test on our database.

4.1 Importing the Libraries and Loading the Dataset

In our project we are using several libraries such as numpy, pandas, Scikit-Learn etc. These libraries help us use various In-built Machine Learning and python functions.

```
In [1]: import numpy as np
        import pandas as pd
        from sklearn import datasets
        import seaborn as sns
        from sklearn.feature_selection import RFE
        from sklearn.model_selection import train_test_split
        from sklearn.model_selection import cross_val_score
        from sklearn.model_selection import KFold
        from sklearn.pipeline import Pipeline
        from sklearn.preprocessing import StandardScaler
        import matplotlib.pyplot as plt
```

Fig 4.1 Importing Libraries.

In the next step we would load the dataset using the pandas function :

Pd.read_csv()

```
In [2]: df = pd.read_csv(r"C:\Users\l\Desktop\lala.csv")
        df.head(3)
```

Out[2]:

	Age	Race	Marital Status	T Stage	N Stage	6th Stage	Grade	A Stage	Tumor Size	Estrogen Status	Progesterone Status	Regional Node Examined	Reginol Node Positive	Survival Months	Status
0	43	Other (American Indian/AK Native, Asian/Pacifi...	Married (including common law)	T2	N3	IIIC	Moderately differentiated; Grade II	Regional	40	Positive	Positive	19	11	1	Alive
1	47	Other (American Indian/AK Native, Asian/Pacifi...	Married (including common law)	T2	N2	IIIA	Moderately differentiated; Grade II	Regional	45	Positive	Positive	25	9	2	Alive
2	67	White	Married (including common law)	T2	N1	IIB	Poorly differentiated; Grade III	Regional	25	Positive	Positive	4	1	2	Dead

Fig 4.2 Loading the Dataset

4.2 Cleaning the Dataset

If our dataset contains any null value in any of the column then we need to remove that column or if any row is null than we should remove that row as Null values is the dataset negatively effects our models accuracy.

df.isnull().any()

```
In [5]: print(df.isnull().any())

Age                       False
Race                      False
Marital Status            False
T Stage                   False
N Stage                   False
6th Stage                 False
Grade                     False
A Stage                   False
Tumor Size                False
Estrogen Status           False
Progesterone Status       False
Regional Node Examined    False
Reginol Node Positive     False
Survival Months           False
Status                    False
dtype: bool
```

Fig 4.3 Checking for Null Values

4.3 Preprocessing

Preprocessing is a very important part of Machine Learning Models. Preprocessing is done to clean or structure the raw data so as to get best results.

First of all the data will be converted from object to integer as our model understands integers and not strings of data.We are going to import labelencoder() library from sklearn for that.

```
In [8]: #Labeling the data and converting to integers
        from sklearn.preprocessing import LabelEncoder
        labelencoder_Y = LabelEncoder()
        df.iloc[:,4] = labelencoder_Y.fit_transform(df.iloc[:,4].values)
        df.iloc[:,5] = labelencoder_Y.fit_transform(df.iloc[:,5].values)
        df.iloc[:,2] = labelencoder_Y.fit_transform(df.iloc[:,2].values)
        df.iloc[:,3] = labelencoder_Y.fit_transform(df.iloc[:,3].values)
        df.iloc[:,1] = labelencoder_Y.fit_transform(df.iloc[:,1].values)
        df.iloc[:,6] = labelencoder_Y.fit_transform(df.iloc[:,6].values)
        df.iloc[:,7] = labelencoder_Y.fit_transform(df.iloc[:,7].values)
        df.iloc[:,9] = labelencoder_Y.fit_transform(df.iloc[:,9].values)
        df.iloc[:,10] = labelencoder_Y.fit_transform(df.iloc[:,10].values)
        df.iloc[:,14] = labelencoder_Y.fit_transform(df.iloc[:,14].values)
        #df.iloc[:,15] = labelencoder_Y.fit_transform(df.iloc[:,15].values)
```

Fig 4.4 Label Encoding

4.4 Picking out the Relevant Features for Model Training

We will use the method of identifying the features that have direct correlations with the target variables. The target variable is 'Survival months'. We can do this by building a correlation matrix. From the matrix,we can pick the top 10 features that has relationship with the target variable.

```
In [11]: correlation = df.corr(method='pearson')
         columns = correlation.nlargest(10, 'Survival Months').index
         columns

Out[11]: Index(['Survival Months', 'Estrogen Status', 'Progesterone Status', 'A Stage',
                'Race ', 'Grade', 'Age', 'Regional Node Examined', 'Marital Status',
                'T Stage '],
               dtype='object')
```

Fig 4.5 Correlation

From our search we got the 10 Best Features that are:

```
['Survival Months', 'Estrogen Status', 'Progesterone
Status', 'A  Stage' ,'Race ', 'Grade', 'Age',
'Regional     Node Examined',    'Marital Status', 'T
Stage ']
```

We had also created a Heat Map for further clarification

```
In [15]: correlation_map = np.corrcoef(df[columns].values.T)
         sns.set(font_scale=.70)
         heatmap = sns.heatmap(correlation_map, cbar=True, annot=True, square=True, fmt='.2f',
                               yticklabels=columns.values, xticklabels=columns.values)

         plt.show()
```

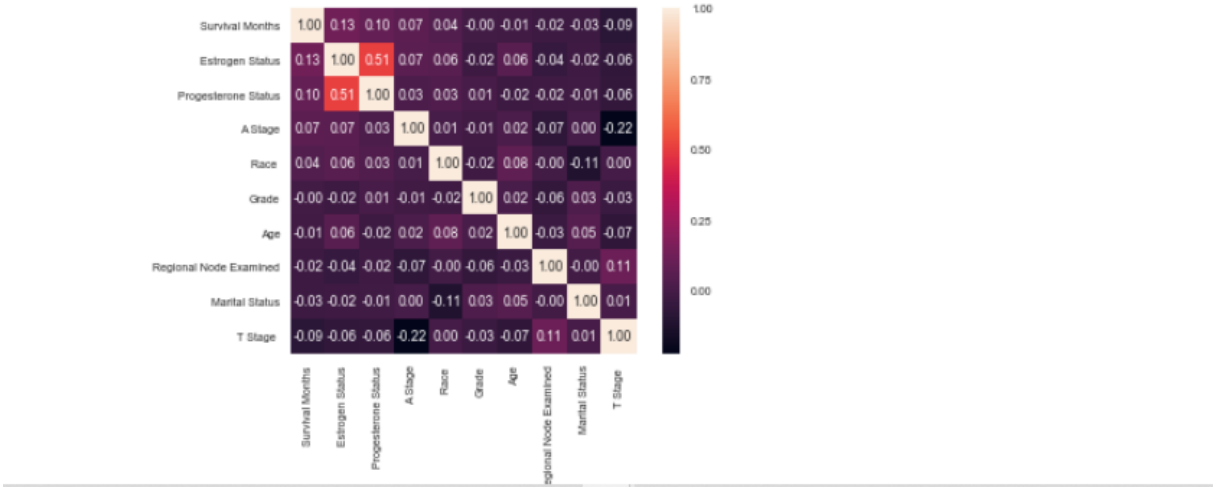

Fig 4.6 Correlation Heat Map

4.5 Baseline Algorithm Test

We have several regression models for this task but to get the best we will test each of the models and finally use the best one according to its negative mean squared error.

We will remove the target feature from the model and then split it into 80 to 20 ratio for testing and splitting.

After appending each model to different pipeline and finding their negative mean squared error we get the best model as Linear Regression.

```
In [18]: X_train, X_test, Y_train, Y_test = train_test_split (X, Y, test_size = 0.20, random_state=42)

In [19]: from sklearn.linear_model import LinearRegression
         from sklearn.linear_model import Lasso
         from sklearn.linear_model import ElasticNet
         from sklearn.tree import DecisionTreeRegressor
         from sklearn.neighbors import KNeighborsRegressor
         from sklearn.ensemble import GradientBoostingRegressor
```

Fig 4.7 Spliting Data and Importing Models

```
In [22]: pipelines = []
         pipelines.append(('ScaledLR', Pipeline([('Scaler', StandardScaler()),('LR',LinearRegression())])))
         pipelines.append(('ScaledLASSO', Pipeline([('Scaler', StandardScaler()),('LASSO', Lasso())])))
         pipelines.append(('ScaledEN', Pipeline([('Scaler', StandardScaler()),('EN', ElasticNet())])))
         pipelines.append(('ScaledKNN', Pipeline([('Scaler', StandardScaler()),('KNN', KNeighborsRegressor())])))
         pipelines.append(('ScaledCART', Pipeline([('Scaler', StandardScaler()),('CART', DecisionTreeRegressor())])))
         pipelines.append(('ScaledGBM', Pipeline([('Scaler', StandardScaler()),('GBM', GradientBoostingRegressor())])))

         results = []
         names = []
         for name, model in pipelines:
             kfold = KFold(n_splits=10, random_state=21)
             cv_results = cross_val_score(model, X_train, Y_train, cv=kfold, scoring='neg_mean_squared_error')
             results.append(cv_results)
             names.append(name)
             msg = "%s: %f (%f)" % (name, cv_results.mean(), cv_results.std())
             print(msg)

         ScaledLR: -0.215626 (0.045437)
         ScaledLASSO: -0.221461 (0.044971)
         ScaledEN: -0.221461 (0.044971)
         ScaledKNN: -0.256590 (0.042534)
         ScaledCART: -0.474807 (0.050775)
         ScaledGBM: -0.221317 (0.043006)
```

Fig 4.8 Finding best Model

4.6 Finalise and validate model

We finally train our Linear Regression Model using Grid Search to tune the hyperparameters.

```
In [24]: from sklearn.metrics import mean_squared_error
         from math import sqrt

         scaler = StandardScaler().fit(X_train)
         rescaled_X_train = scaler.transform(X_train)
         model = LinearRegression(fit_intercept=True, normalize=False, copy_X=True, n_jobs=None)
         model.fit(rescaled_X_train, Y_train)

         # transform the validation dataset
         rescaled_X_test = scaler.transform(X_test)
         predictions = model.predict(rescaled_X_test)
         error=sqrt(mean_squared_error(Y_test, predictions))
         print(error)
```

Fig 4.9 Linear Regression

4.7 Final results

We have to first de-normalize our results to see the actual results.

```
In [30]: actual_y_test = np.exp(Y_test)
         actual_predicted = np.exp(predictions)
         diff = (actual_y_test - actual_predicted)

         compare_actual = pd.DataFrame({'Test Data': actual_y_test, 'Predicted ' : actual_predicted, 'Difference' : diff})
         compare_actual = compare_actual.astype(int)
         compare_actual.head(30)
```

Fig 4.10 De-Normalization

We predicted the value of the survival months and after checking the accuracy using rms error we had an error of 19 months.The error is not very small but our future steps will be in the direction to reduce the error to a greater extent.

5. Summary and Conclusion

1. Summary

This report discussed our projects functional requirement and it's implementation. we briefly discussed about the model ad how it works. We conclude our project that resulted from analysis of various regression models.

2. Conclusion

In this project we learned to:

- practice acquired knowledge of machine learning.
- Identify the best suitable model for the dataset and prognosis of breast cancer.
- improve the accuracy of models by changing hyper parameters.
- Work in a team in development of the project.

REFERENCES

1. Arihito Endo, Takeo Shibata and Hiroshi Tanaka, "Com-parison of Seven Algorithms to Predict Breast Cancer Survival,"Biomedical Soft Computing and Human Sciences, Vol. 13,No.2, pp. 11-16, 2008.

2. Shomona G. Jacob and R. Geetha Ramani, "Efficient Clas-sifier for Classification of Prognosis Breast Cancer Data Through Data Mining Techniques," Proceedings of the World Congress on Engineering and Computer Science 2012, Vol. I, October 2012.

3. Label Encoder vs. one Hot label encoder in machine learning : https://medium.com/@contactsunny/label-encoder-vs-one-hot-encoder-in-machine-learning-3fc273365621

4. Alberto Palacios Pawlovsky , Mai Nagahashi , A Study on the Use of the kNN Algorithm for Prognosis of Breast Cancer,Toin University of Yokohama: Faculty of Biomedical Engineering, Dept. of Clinical Engineering,Dept. of Clinical Engineering .