Compare Different Machine Learning Models on Breast Cancer Data

Hanh Vo

July 4, 2021

I) Introduction and Overview of Dataset

    1) Introduction (Rafay Alam)

Breast cancer (BC) is one of the most common cancers among women in the world today. Currently, the average risk of a woman in the United States developing breast cancer sometime in her life is about 13%, which means there is a 1 in 8 chance she will develop breast cancer. There are two types of breast cancer: M type -**malignant** , and B type-benign Benign tumors are noncancerous. They tend to grow slowly and do not spread while **malignant** (cancerous) can grow rapidly, invade and destroy nearby normal tissues, and spread throughout the body.

Breast cancer screening is performed using mammograms, clinical breast exam (CBE), and MRI (magnetic resonance imaging) tests giving 65% to 98% accuracy. However there are some limitations such as false negative results in mammograms. According to cancer.org "overall, screening mammograms do not find about 1 in 5 breast cancers" . MRI is also a very expensive and time consuming investigation.

Technique	Accuracy
Surgical Biopsy	100%
Mammography	68-79%
FNA	65-98%

Table 1: Accuracy of few breast cancer analysis techniques

In health science, artificial intelligence (AI) techniques like machine learning are starting to take hold for better cancer forecasting. It is because machine learning algorithms have become so good at recognizing patterns and because of the large adaption of computer-based techniques into different forms of healthcare.

In this project, we will analyze the diagnostic Wisconsin Breast Cancer Database and implement different classification algorithms to compare their accuracy in classifying breast cancer type.

2a) Overview of Dataset

The dataset consists of 30 individual cell features including patient ID and diagnosis type. Ten real-valued features mentioned are computed for each cell nucleus associated with three independent values such as mean, standard error (SE), and "worst" (mean of three largest values) resulting from 30 features. The ten values are:

1. Radius
2. Texture
3. Perimeter
4. Area
5. Smoothness
6. Compactness
7. Concavity
8. Concave Points
9. Symmetry
10. Fractal Dimension

II) Data Exploratory and analysis(by sri ganesh ganta)

DATA EXPLORATION

Data exploration is the initial step in **data** analysis, where users explore a large **data** set in an unstructured way to uncover initial patterns, characteristics, and points of interest.

Data exploration helps create a more straightforward view of datasets rather than pouring over thousands of figures in unstructured data

We used data.head() to see the number of different variables present in the data and their values

	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	symmetry_r
0	842302	M	17.99	10.38	122.80	1001.0	0.11840	0.27760	0.3001	0.14710	0.2
1	842517	M	20.57	17.77	132.90	1326.0	0.08474	0.07864	0.0869	0.07017	0.1
2	84300903	M	19.69	21.25	130.00	1203.0	0.10960	0.15990	0.1974	0.12790	0.2
3	84348301	M	11.42	20.38	77.58	386.1	0.14250	0.28390	0.2414	0.10520	0.2
4	84358402	M	20.29	14.34	135.10	1297.0	0.10030	0.13280	0.1980	0.10430	0.1

```
[ ] df.info()
```

We can also use df.describe() to Print some basic statistics of our data and df.info() to Print some general information about our data.

```
[ ] df.describe()
```

	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	symmetry_mean	fractal_di
count	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	569.000000	
mean	14.127292	19.289649	91.969033	654.889104	0.096360	0.104341	0.088799	0.048919	0.181162	
std	3.524049	4.301036	24.298981	351.914129	0.014064	0.052813	0.079720	0.038803	0.027414	
min	6.981000	9.710000	43.790000	143.500000	0.052630	0.019380	0.000000	0.000000	0.106000	
25%	11.700000	16.170000	75.170000	420.300000	0.086370	0.064920	0.029560	0.020310	0.161900	
50%	13.370000	18.840000	86.240000	551.100000	0.095870	0.092630	0.061540	0.033500	0.179200	
75%	15.780000	21.800000	104.100000	782.700000	0.105300	0.130400	0.130700	0.074000	0.195700	
max	28.110000	39.280000	188.500000	2501.000000	0.163400	0.345400	0.426800	0.201200	0.304000	

We can use df.shape to see the number of observations and number of variables present in the data

```
[ ] df.shape
```

```
(569, 31)
```

From the data we can see there are 569 different observations and 31 different variables
We should also check if there is any missing data

`[6]` `df.isnull()`

|	id	diagnosis	radius_mean	texture_mean	perimeter_mean	area_mean	smoothness_mean	compactness_mean	concavity_mean	concave points_mean	symmetry_me
0	False	False	False	False	False	False	False	False	False	False	Fal
1	False	False	False	False	False	False	False	False	False	False	Fal
2	False	False	False	False	False	False	False	False	False	False	Fal
3	False	False	False	False	False	False	False	False	False	False	Fal
4	False	False	False	False	False	False	False	False	False	False	Fal
...	...	...	...	...	...	...	...	...	...	...	
564	False	False	False	False	False	False	False	False	False	False	Fal
565	False	False	False	False	False	False	False	False	False	False	Fal
566	False	False	False	False	False	False	False	False	False	False	Fal
567	False	False	False	False	False	False	False	False	False	False	Fal
568	False	False	False	False	False	False	False	False	False	False	Fal

569 rows × 33 columns

From the data we can see all of our columns are showing false so there is no missing data, if there was any missing data it would showcase as true in any column, there is a null column in the end showing there is missing data, so we will drop it to avoid any accuracy predictability

Data exploration can use a combination of manual methods and automated tools such as data visualizations, charts, and initial reports.
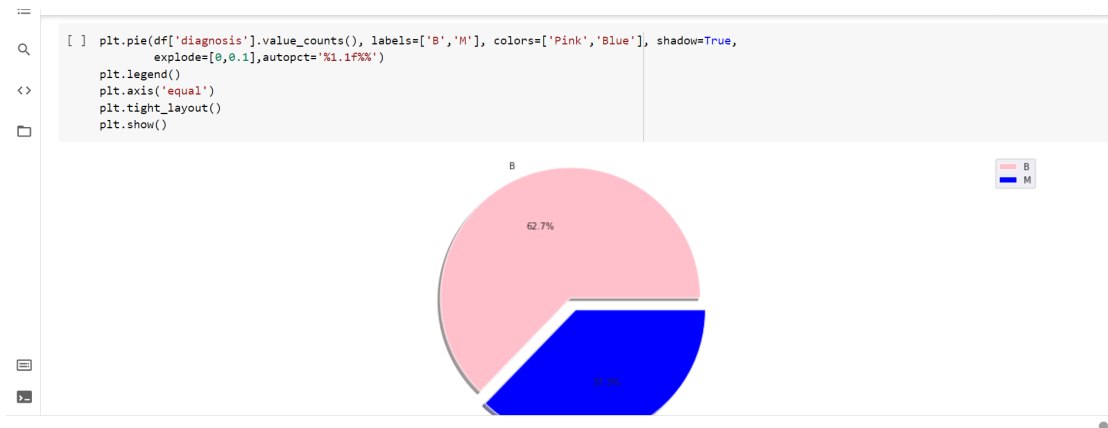
From the pairpots we can see that the two groups are well separated. There is little overlap between the two variables. For example from the concave points_mean density plot, the peaks which represent where the data points are concentrated clearly separated. The M type breast cancer has concave point means below 0.05. We can also see there are strong positive linear relationships on the top left of the pairplot which mean these variables are highly correlated to each other. (Hanh Vo)

The target variable is diagnosis, let us make a count plot and pie chart to see how the data is scattered


```
Double-click (or enter) to edit

[ ]  sns.countplot(x=df['diagnosis'])
     plt.show()
```
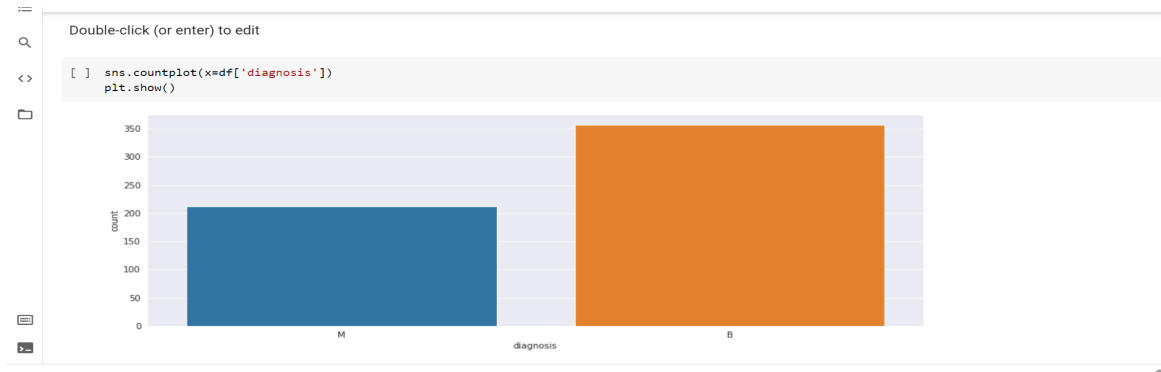
From the countplot we can see the count of people with diagnosis B is more than of with Diagnosis M


```
[ ]  plt.pie(df['diagnosis'].value_counts(), labels=['B','M'], colors=['Pink','Blue'], shadow=True,
             explode=[0,0.1],autopct='%1.1f%%')
     plt.legend()
     plt.axis('equal')
     plt.tight_layout()
     plt.show()
```

Now we can see the percentages of the people with diagnosis B(62.7%) and M(37.3%) , the pie chart is helpful in giving a scale based on percentages

Now let us make some more graphs to see how well the data is performed between its variables

```
sns.scatterplot(x = 'area_mean', y = 'smoothness_mean', hue = 'diagnosis', data = df)
plt.show()
```


 There is a complex non linear relationship between area mean and smoothness_mean. However, the data points are well separated. We can see from the plot that Most B type cancer has area_mean under 550. Most M type cancer scatters evenly between the (550,1500) interval and scatter loosely between (1500,2500) intervals.(Hanh)

```
] sns.histplot(data=df, x="radius_mean", kde=True)
```

```
<matplotlib.axes._subplots.AxesSubplot at 0x7fd36f17b6d0>
```
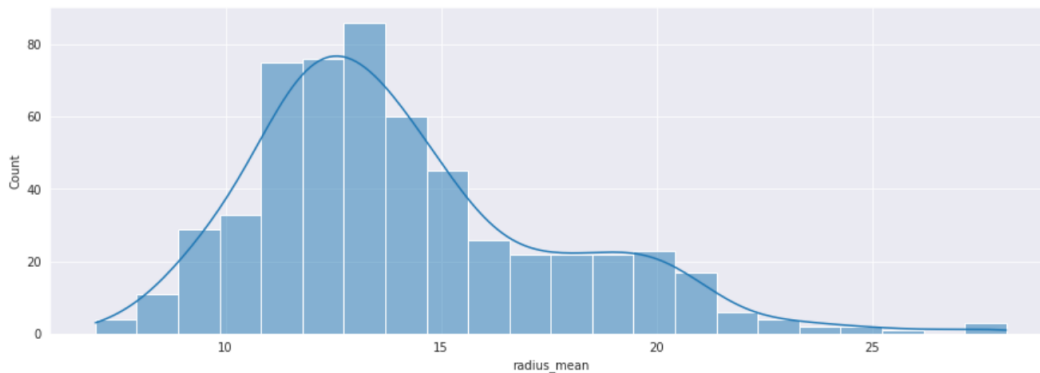

From the histplot we can see how the radius_mean values vary by the count, it gives us a good understanding on how radius_mean is being performed
Data exploration can also assist by reducing work time and finding more useful and actionable insights from the start alongside presenting clear paths to perform better analysis.

III) Data pre-processing and models overview (Noe Bastidas)

    1) Data Preprocessing

Data Preprocessing is the process where the data gets transformed to the point where it can easily be taken in by a model. One of the first steps is checking for null values in our dataset.

Handling Null Values:

```
df.isnull().sum()
```

id	0	compactness_se	0
diagnosis	0	concavity_se	0
radius_mean	0	concave points_se	0
texture_mean	0	symmetry_se	0
perimeter_mean	0	fractal_dimension_se	0
area_mean	0	radius_worst	0
smoothness_mean	0	texture_worst	0
compactness_mean	0	perimeter_worst	0
concavity_mean	0	area_worst	0
concave points_mean	0	smoothness_worst	0
symmetry_mean	0	compactness_worst	0
fractal_dimension_mean	0	concavity_worst	0
radius_se	0	concave points_worst	0
texture_se	0	symmetry_worst	0
perimeter_se	0	fractal_dimension_worst	0
area_se	0	Unnamed: 32	569
smoothness_se	0	dtype: int64	

Since a column is completely full of null values, it proves to be non valuable so the feature can be dropped. Another step that needs to get taken is dealing with categorical variables.

Handling Categorical Variables:

```
df.head()
```

	id	diagnosis	r
0	842302	M	
1	842517	M	
2	84300903	M	

```
#handling null column
if 'id' and 'Unnamed: 32' in df:
    df.drop(['id','Unnamed: 32'],axis=1,inplace=True)


#hanlding categorical variables:

#Label Encoding using map function
df['diagnosis'] = df['diagnosis'].map({'M':1,'B':0})
```

Since the feature id is unique to each record and is not in any way involved in the diagnosis we can drop this feature. However, since diagnosis is our output/classifier variable we can use label encoding to assign the classes into numbers. This will now allow our models to handle the categorical variable. Another step of preprocessing is to check for multicollinearity. Having Multicollinearity can mean having certain features being statistically more important than they really are when fed into the model. This can be identified with a heatmap.

Multicollinearity:
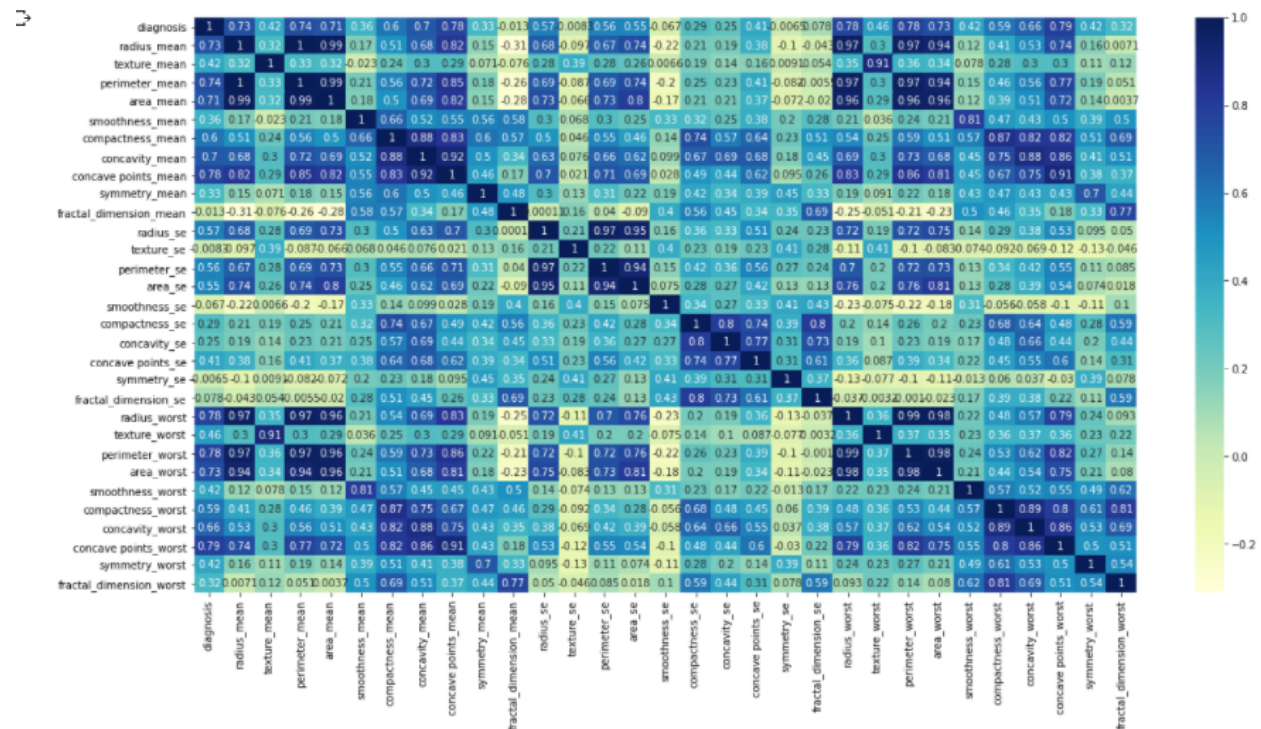
By looking at a heatmap identifying variables with correlations of about 7> are indicators of collinearity. The solution to this would be to drop a few variables in order to put less statistical significance on certain features that may not realistically have much significance. Another way to make sure certain features carry a more accurate statistical significance is to scale the data. After the data is ready and transformed, the next step is to split the data into training sets and

testing sets. The training set will be used to create the models and the testing set will be used to validate the model created.

2) Models Overview

We are comparing 5 different models: (Hanh Vo)

Logistic Regression (Softmax):

- Logistic Regression acts somewhat very similar to linear regression. It also calculates the linear output, followed by a stashing function over the regression output. Sigmoid function is the frequently used logistic function. It can only support linear solutions.

KNeighborsClassifier (KNN):

- The K-nearest neighbors are simple and basic machine learning algorithms, which are used for both regression and classification problems. These algorithms are popular as lazy learners because of passive nature in the resonance of large datasets. But these are highly useful to classify same-targeted outcomes.

Decision Tree Classifier (Dtree):

- Decision Tree Classifier is a simple supervised machine learning where the data is continuously split based on parameters.This can be applied to both classification and regression problems. The main idea is to part the data into clusters through a divide and conquer manner until all are in the same class.

Support Vector Machine (SVM):

- Support vector machines are famous machine learning algorithms to conduct outcome classification. The objective behind the SVM algorithms is to identify a hyperplane in N-dimensional space that randomly classifies the data points [17]. The hyperplanes are decision boundaries, which help to classify the data.

GaussianNB():

- Assumes that values linked with each class are dispersed according to the Gaussian (Normal Distribution). Predictions are made for every class to see if there is an association with another class using conditional probability.

Linear Discriminant Analysis (LDA) :

- Linear Discriminant Analysis is known for its dimensionality reduction. It assumes all classes are linearly separable and creates several hyperplanes in the feature space in order to separate the classes.

III) Result Analysis (Hanh Vo)

a) Performance measures

We are interested in seeing how the model can predict the patient with a malignant tumor. Thus we use performance measures such as True Positive Rate, Accuracy, precision, AUC score. Below are the definitions of the metrics we used:

Confusion matrix:

- True Positive(TP): Numbers of Patients who are positive (have malignant breast cancer) and were classified as positive(1)
- True Negative (TN): Number of Patients who are negative(0) with malignant breast cancer and were classified as negative (0).
- False positive (FP): Number of patient who are negative with malignant breast cancer but were classified as positive(1)
- False Negative (FN): Number of patient who are positive with malignant breast cancer were classified as negative (0)

Classification report:

Metrics	Meaning	Measurement
Accuracy	The proportion of the total number of predictions that were correct	(TP+TN)/(TP+TN+FP+FN)
Precision	Aka "positive predictive value" It's the proportion of positive cases that were correctly identified.	TP / (TP + FP)
Negative Predictive Value	The proportion of negative cases that were correctly identified	TN/(TN+FN)
Recall (sensitivity)	It's the true positive rate also called the recall. It is the number of instances from the positive (first) class that actually predicted correctly.	TP/(TP+FN)
Specificity	It's the true negative rate. Is the number of instances from the negative class (second) class that were actually predicted correctly.	(TP+TN)/(TN+FP)
f1-score	Weighted harmonic mean of the precision and recall, used to compare classifier models, not global accuracy F-beta score reaches its best value at 1 and worst score at 0.	2*(Recall * Precision) / (Recall + Precision)
Support	the number of occurrences of each class in y_true Imbalanced **support** in the training data may indicate structural weaknesses in the reported **scores** of the classifier and could indicate the need for stratified sampling or rebalancing	

Predicted Class

		Positive	Negative	
		Positive	**Negative**	
Actual Class	**Positive**	True Positive (TP)	False Negative (FN) **Type II Error**	**Sensitivity** $\frac{TP}{(TP+FN)}$
	Negative	False Positive (FP) **Type I Error**	True Negative (TN)	**Specificity** $\frac{TN}{(TN+FP)}$
		Precision $\frac{TP}{(TP+FP)}$	**Negative Predictive Value** $\frac{TN}{(TN+FN)}$	**Accuracy** $\frac{TP+TN}{(TP+TN+FP+FN)}$

Fig: Classification report visualized

Other measurements:

- Outer ACC(accuracy): Average cross-validation score obtained by cross_val_score function
- Training score: How the model generalized or fitted in the **training** data
- Testing score: How the model generalized or fitted in the **testing** data
- AUC score: The Area Under the Curve (**AUC**) is the measure of the ability of a classifier to distinguish between classes and is used as a summary of the ROC curve. The higher the **AUC**, the better the performance of the model at distinguishing between the positive and negative classes.

b) Models comparison result analysis

❖ **Our approach:**

First, setting up multiple GridSearchCV objects, 1 for each algorithm to find the best parameter of each classifier. Then we use cross validation of 10 folds to split X_train data and calculate the average testing score of 10 splits using the best parameter. Next, we loop

over each classifier and calculate the testing score, training score, sensitivity, recall, AUC...etc (using their best parameters).

*Fig: Visualization of cross validation*
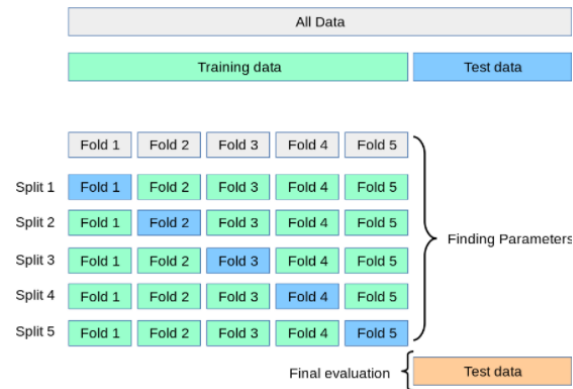
❖ **Result:**

| Model	Cross validation score	Accuracy	Specificity (true negative rate)	Sensitivity (true positive rate)	Negative Predictive Value	precision	Training score	Testing score	AUC ROC
Softmax	98.99 ± 2.03	94	98	88	93	97	99.75	94.15	92.82
KNN	96.98 ± 2.95	95	98	91	95	97	100	95.32	94.38
DTree	93.72 ± 3.21	91	97	81	90	95	97.24	91.23	89.22
SVM	99.24 ± 1.63	96	99	91	95	98	99.50	95.91	94.85
NB(Naive Bayes)	93.47 ± 4.20	93	98	84	91	96	94.22	92.98	91.25
LDA	96.72 ± 3.00	93	98	83	91	96	97.74	92.40	90.47

From this table above, we can see that Support Vector Machine (SVM) is the best classifier for this dataset. **SVM** has the highest average cross validation score (99.24) followed by Logistic Regression (98.99%), and KNN (96.98%). All of the models have an average cross validation score over 90% which is consistent with our expectation from looking at the charts (The pairplots show that the data were well separated). Naive Bay

and Decision Tree have the lowest cross validation score (about 93%). SVM also has the smallest average standard deviation. It means that the cross validation score of each run is consistent.
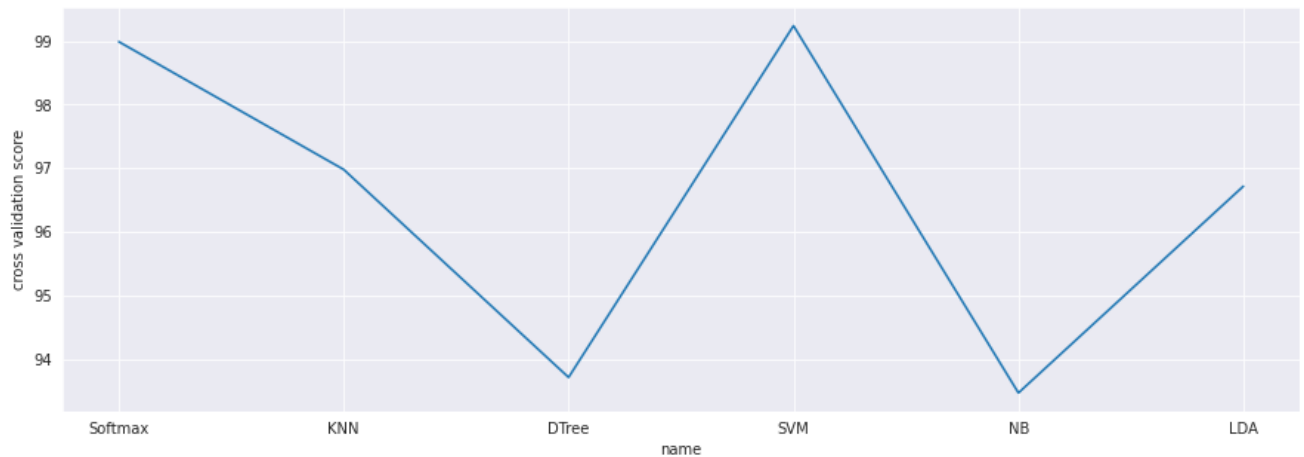
Fig: Cross validation score of each models

As mentioned earlier, accuracy is the total correct prediction/total predictions. SVM has the highest accuracy (96%) followed by KNN and Logistic Regression. Decision Tree has the lowest accuracy (91%), followed by Naive Bay and LDA (93%).
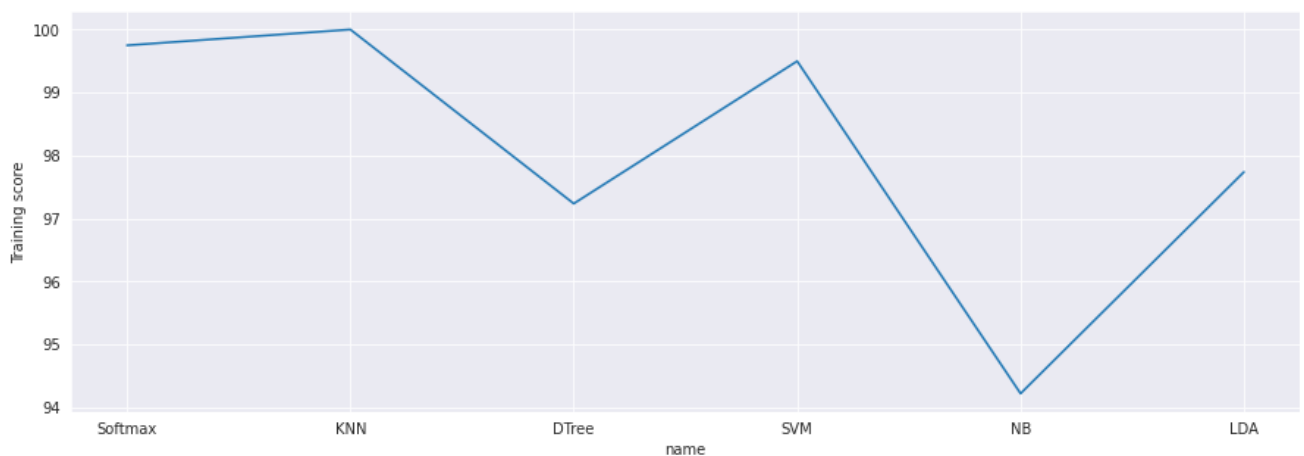
We are also interested in the Area under ROC Curve (or AUC for short).The ability to discriminate between malignant and benign tumors is the highest in SVM based on AUC score of 94.85% followed by KNN (94.38%) and Logistic Regression (92.82%)

Decision Tree classifier's ability to distinguish between the positive and negative is the weakest (89.22%) followed by LDA (90.47%) and Naive Bay((91.25%)

ROC can be broken down into sensitivity and specificity. A binary classification problem is really a trade-off between sensitivity and specificity. All modes gave almost identical and high Specificity (true negative rate). However, SVM, KNN, and LR still have the highest sensitivity (true positive rate) respectively.

- Comparison of confusion matrix of three best classifiers:
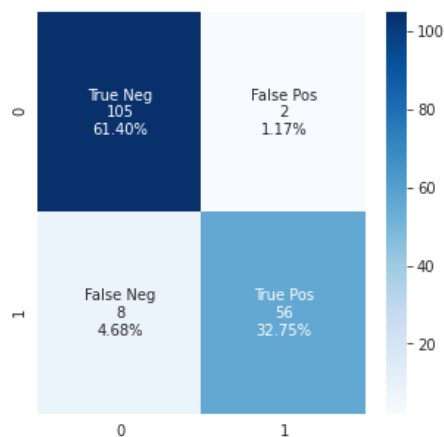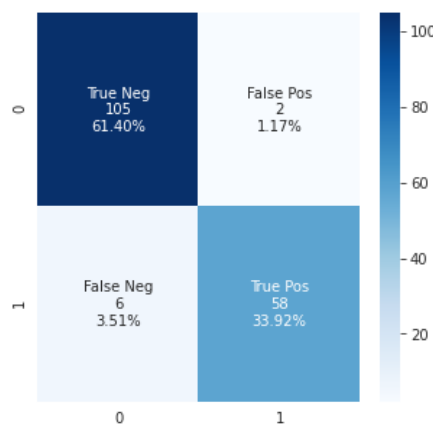
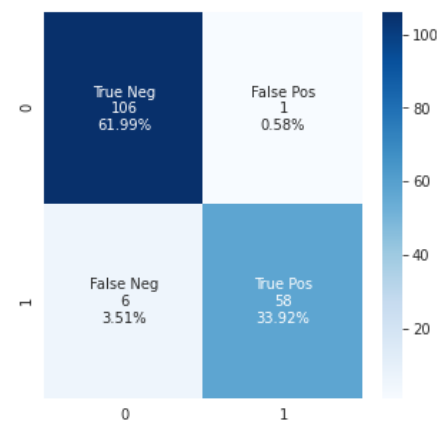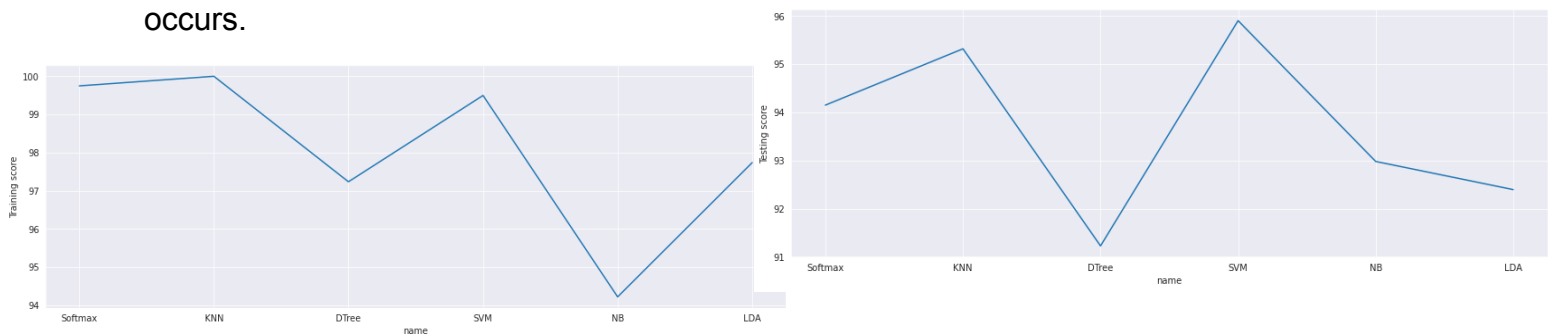We can see from the matrix that SVM gives the highest True Positive and True Negative Rate. It also gives the smallest false negative and false positive rate. Comparison of overall training score and testing score shows that no overfitting occurs.

Dealing with multicollinearity results:

We perform a second experiment in which we select only features that have a correlation <0.75 with the output. (Heatmap of selected features)

Result:

Model	Cross validation score	Accuracy	Specificity (true negative rate)	Sensitivity (true positive rate)	Negative Predictive Value	precision	Training score	Testing score	AUC
Softmax	100 ± 0.00	99	100	98	99	100	100	99.42	99.22
KNN	99.24 ± 1.61	99	100	98	99	100	99.75	99.42	99.22
DTree	100 ± 0.00	100	100	100	100	100	100	100	100
SVM	100 ± 0.00	100	100	100	100	100	100	100	100
NB(Naive Bayes)	100 ± 0.00	100	100	100	100	100	100	100	100
LDA	100 ± 0.00	100	100	100	100	100	100	100	100

Using the table above all the models perform extremely well with cross validation score, sensitivity, specificity, ...etc reach 100% or almost a 100%. KNN performed the worst out of all the models. This can be explained with KNN's sensitivity to noise/outliers.
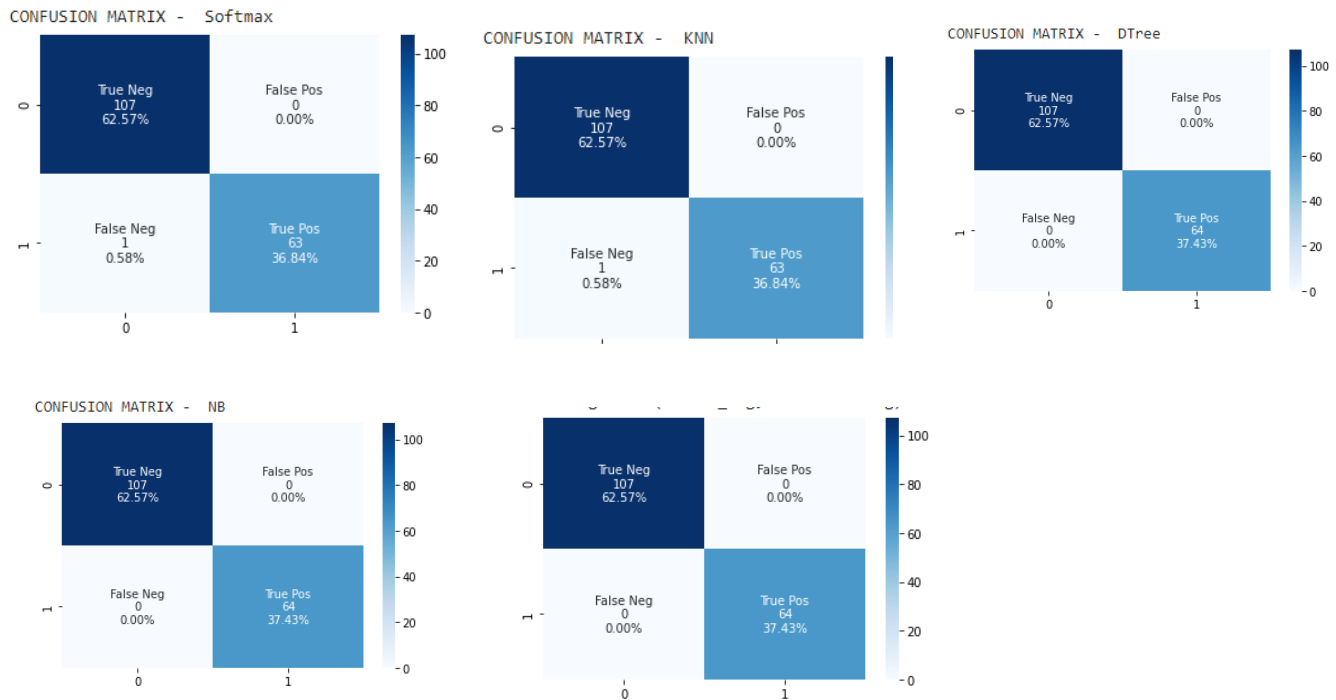
The confusion matrix of the data with limited features show a significant improvement in the true positive and true negative rate while reducing the false negative and false positive.

V) Conclusion (Hanh Vo)

In conclusion, this project presents five supervised machine learning classifiers. All five algorithms give the accuracy score over 90% and AUC score over 89%. The three best algorithms are SVM, KNN, and Logistic Regression respectively. We conduct two experiments, one with full features and one with limited features. Result shows that with limited features, the accuracy scores and sensitivity score were improved to 100%. SVM is still the best algorithm among other classifiers.

Compared to traditional screening breast cancer methods which give accuracy between 65% and 100%, supervised machine learning classifiers such as Support Vector Machine, Logistic Regression can accurately diagnose malignant breast cancer tumors with a high accuracy up to 100%. With a selection of the suitable machine learning algorithm and attributes (features) can help the doctors in the early diagnosis of malignant tumors and ultimately save the patient's life.