# Machine Learning in Breast Cancer Prediction

**Rohit Midha, Saadhana Lakshmi Narasimhan**

Department of Computer Science,
SSN College of Engineering

***Abstract***

In this paper, we explore the applications of Machine Learning to breast cancer by focusing on predicting possibility of a person having breast cancer. We use a variety of methods including neural networks and thereby propose a hybrid model following the No Free Lunch Theorem. On average our best model is able to predict with an accuracy of 99.7%.

## 1. Introduction

According to a WHO report, 'Breast cancer is the top cancer in women worldwide and is increasing particularly in developing countries where the majority of cases are diagnosed in late stages.'
The American Cancer Society's estimates for breast cancer in the United States for 2018 are:

- About 266,120 new cases of invasive breast cancer will be diagnosed in women.
- About 63,960 new cases of carcinoma in situ (CIS) will be diagnosed (CIS is non-invasive and is the earliest form of breast cancer).
- About 40,920 women will die from breast cancer.

In this paper we will explore the applications of Machine Learning in the field of Breast Cancer, using The University of **Wisconsin Breast Cancer Diagnosis Dataset (WBCD)** which consists of 569 cases, 357 classified as benign and 212 as malignant.

After a patient who is a possible victim of breast cancer visits a doctor, he or she might want a second opinion, to confirm the results. Given that the model we are building here would be for an extremely sensitive issue, we have set the target accuracy as 99%.

## 2. Dataset and Features

### 1. Data collection

The data collected so far can be classified into two groups: benign and malignant cases.
The data being used was found at the UC Irvine Machine Learning Repository. The features are computed from a digitized image of a fine needle aspirate (FNA) of a breast mass. They describe characteristics of the cell nuclei present in the image.

### 2. Feature building

The following features were selected to build the models.

- radius(mean of distances from center to points on the perimeter)
- texture(standard deviation of gray-scale values)
- perimeter
- area
- smoothness(local variation in radius lengths)
- compactness
- concavity(severity of concave portions of the contour)
- Concave points (number of concave portions of the contour)
- symmetry

- fractal dimension

## 3. METHODS

We used many python libraries such as Scikit-learn, Matplotlib, Keras for the models that were built.

**Decision tree**:
A decision tree is a flowchart-like structure in which each internal node represents a "test" on an attribute, each branch represents the outcome of the test, and each leaf node represents a class label (decision taken after computing all attributes). The paths from root to leaf represent classification rules. **Fig. 1** shows the decision tree classifier built for breast cancer. The series of questions are answered by test data and accordingly a particular branch of the tree is chosen to proceed. The final leaf node arrived at, represents the class the sample belongs to. Hence each sample is classified as benign or malignant.

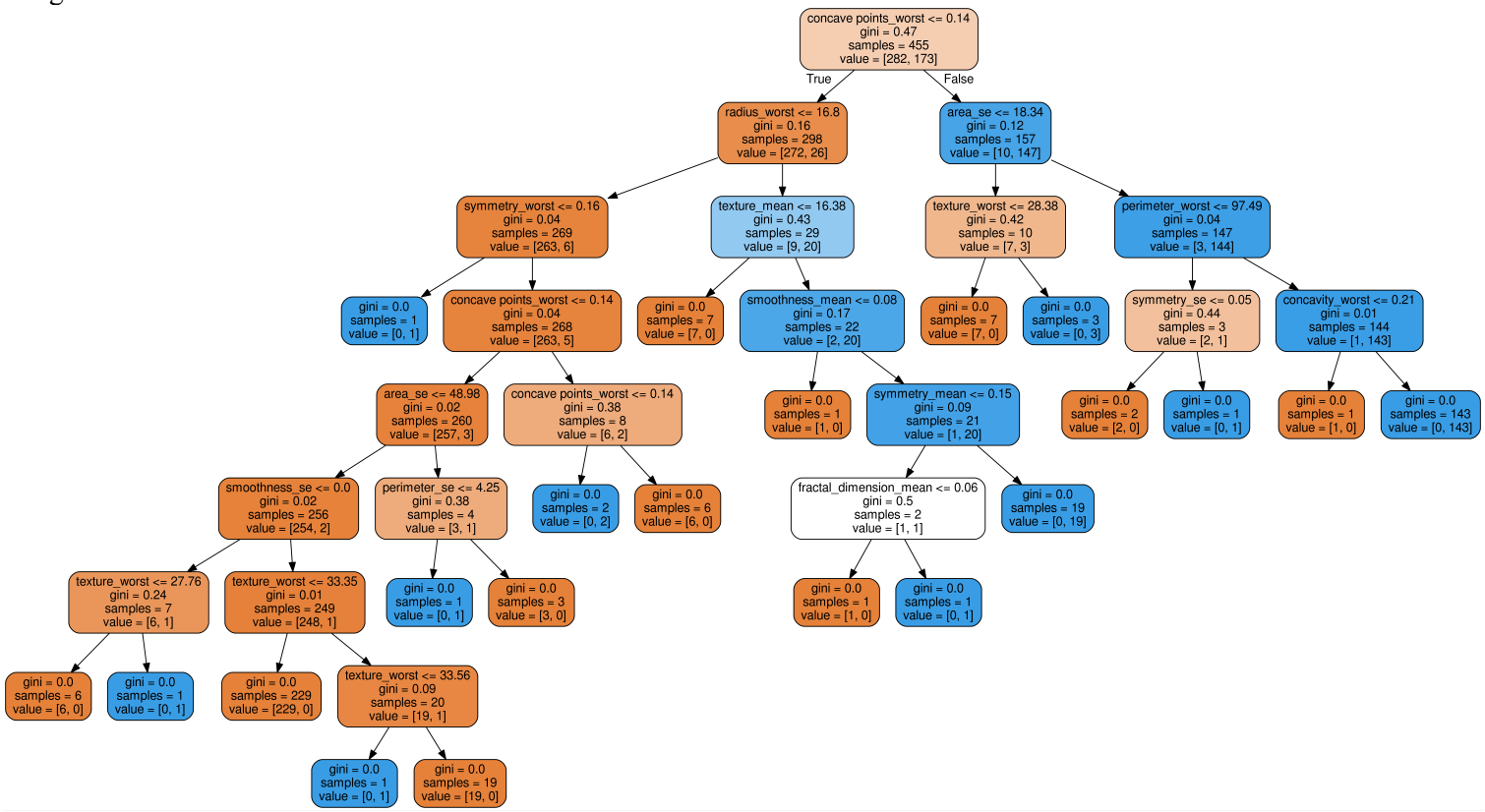



**Fig. 1 Decision Tree Classifier**

**Random Forest:**
A random forest[1] is a collection of several decision trees. This provides a more stable and accurate prediction. Random Forest adds additional randomness to the model, while growing the trees. Instead of searching for the most important feature while splitting a node, it searches for the best feature among a random subset of features. This results in a wide diversity that generally results in a better model. Decision trees might suffer from over-fitting. Random Forest prevents overfitting by creating random subsets of the features and building smaller trees using these subsets. **Fig. 2** depicts 3 decision trees. Tree 1 and Tree 3 classify a test sample as malignant, while tree 2 classifies as benign. Random forest makes a decision based on majority of votes, hence predicts the sample to be malignant.

Over-fitting refers to a model that models the training data too well. This occurs when a model learns every detail and noise in the training data to the extent that it shows very poor performance on unseen data. This is because, the noise or random fluctuations in the training data is picked up and learned as concepts by the model. But these concepts might not apply to new data and the model is no longer able to generalize.

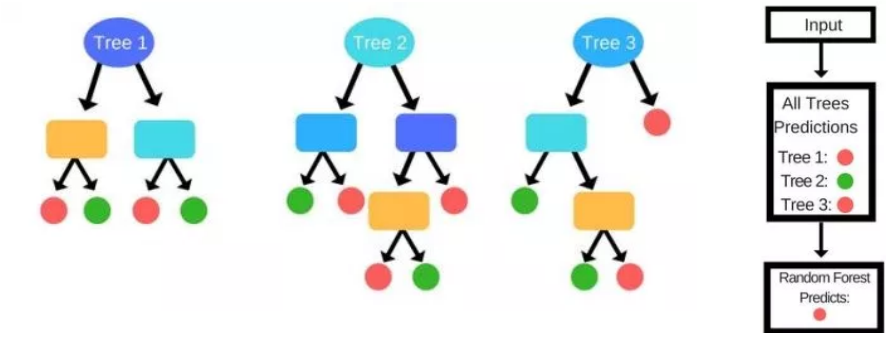



**Fig. 2 Random Forest Classifier**

**Extra Tree**

Extra tree classifier[2] is obtained by randomizing the random forest further. Each tree is trained using the whole learning sample (rather than a bootstrap sample), and the top-down splitting in the tree learner is randomized. Instead of computing the locally optimal cut-point for each feature under consideration, a random cut-point is selected. Then, of all the randomly generated splits, the split that yields the highest score is chosen to split the node.

**Support vector machine**

Each data item in the support vector machine[3] is plotted as a point in n-dimensional space (n is number of features) with the value of each feature being the value of a particular coordinate. Then perform classification by finding the hyper-plane that differentiate the two classes very well. Linear kernel finds a linear hyperplane to classify the samples. Fig. 3 shows an SVM classifier trained with features radius and texture. The red '+' represent malignant samples, while the blue circles represent benign samples. SVM classifier identifies the best hyperplane that classifies the data into their classes. This is represented by the yellow line. The model should consider accuracy as well as aim to maximize margin from samples to prevent overfitting.

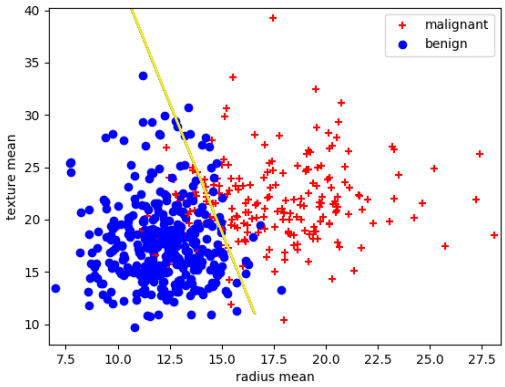



**Fig. 3 SVM classifier**

**Logistic regression**

Logistic regression[4] algorithm uses a linear equation with independent predictors to predict a value. The predicted value can be anywhere between negative infinity to positive infinity. Logistic regression produces an output which is a class variable, i.e 0-no, 1-yes. Squashing of output of the linear equation into a range of [0,1] is done. In **Fig. 4**, X axis represents the features used in breast cancer prediction. The green curve represents the sigmoid function which is used to squash the predicted value between 0 and 1. All samples below this curve(red points) are benign, and the blue ones are malignant.

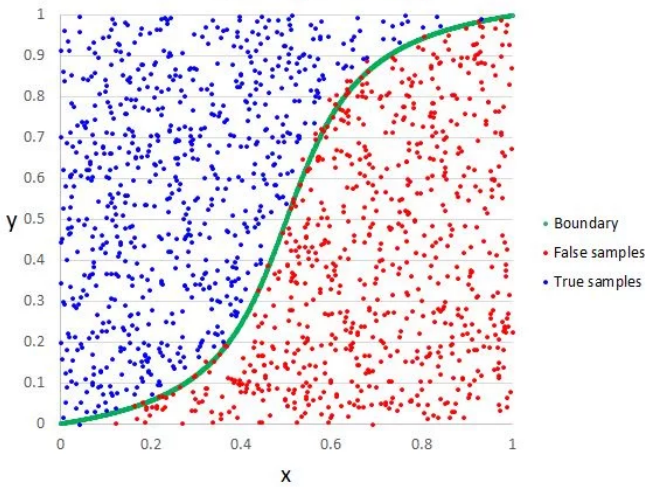



**Fig. 4 Logistic Regression Classifier**

**Naive Bayes**

It is a classification technique based on Bayes' Theorem with an assumption of independence among predictors. Naive Bayes[5] classifier assumes that the presence of a particular feature in a class is unrelated to the presence of any other feature, hence the name Naive. **Fig 5** shows probability of a sample belonging to class malignant or benign. P(c) represents this probability, while P(x) represents the probability of a feature occurring, eg. mean symmetry being<=0.15.

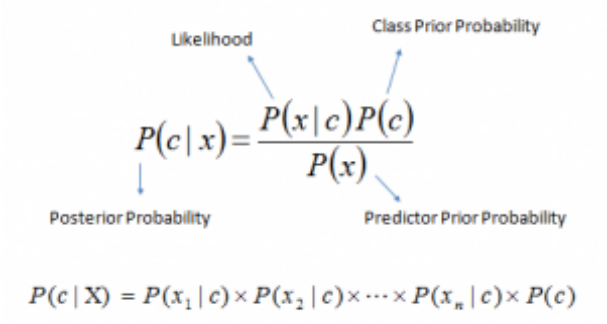



**Fig. 5 Naive Bayes Model**

- P(c/x) is the posterior probability of class c(target) given predictor x(attributes).
- P(c) is prior probability of class.
- P(x/c) is the likelihood which is the probability of predictor given class.
- P(x) is the prior probability of predictor.

**Artificial Neural Networks**

The Neural Network[6] captures information from the outcomes of previous data between cases, as during training, the network is provided the results of previous cases as input along with the features. The Neural Network has an advantage over other methods in that it is also able to take features of all cases involved as inputs. Therefore, it can draw on the outcomes of previous training examples.

The neural network used in this case looks as show below.

| Layer (type) | Output Shape | Param # |
|---|---|---|
| dense_1 (Dense) | (None, 16) | 496 |
| dropout_1 (Dropout) | (None, 16) | 0 |
| dense_2 (Dense) | (None, 16) | 272 |
| dropout_2 (Dropout) | (None, 16) | 0 |
| dense_3 (Dense) | (None, 1) | 17 |

Total params: 785
Trainable params: 785
Non-trainable params: 0

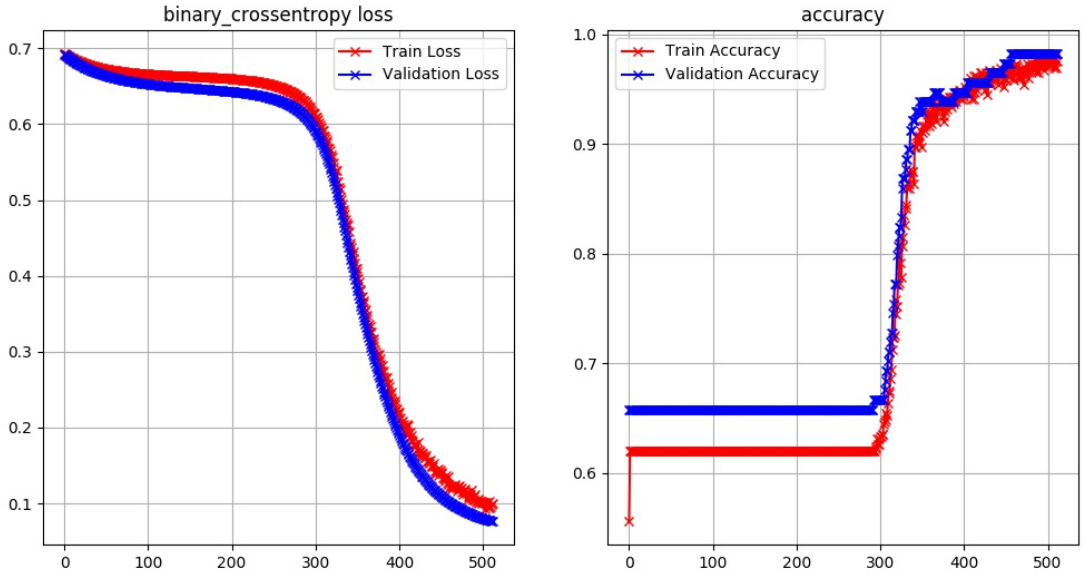



**Fig 6 ANN classifier**

### Hybrid RoSaNet Model

The No Free Lunch Theorem[8] states that any one algorithm that searches for an optimal cost or fitness solution is not universally superior to any other algorithm. In essence, different algorithms prove to be more effective for different data sets. Thus, instead of relying on a single algorithm completely, we rely equally on all of them.

RoSaNet takes into account all the above mentioned models. Every model votes whether a test data is to be classified as benign or malignant. Based on the majority of votes, a sample in the hybrid model is classified as either benign or malignant. This can reduce over fitting as it prevents complete dependence on a single classifier.

## 4. DISCUSSION

In the Support vector machine model, proper parameter selection plays an important role in correct classification. C, the penalty parameter of error term was set as 0.1. Probability is enabled true, to enable probability estimates for all samples. Linear kernel function is used to separate both the classes. Gamma should not be too high, as this can cause over-fitting.

In the logistic regression model, we have used Limited memory Broyden–Fletcher–Goldfarb–Shanno (L-BFGS)[7] solver. The algorithm starts with an initial estimate of the optimal value, and proceeds iteratively to refine that estimate with a sequence of better estimates, by identifying the direction of steepest descent. Multi-class option is set to multinomial. This uses Cross entropy loss i.e, it rewards/penalizes probabilities of correct classes only. The value is independent of how the remaining probability is split between incorrect classes. Cross-entropy loss uses a log function to measure the performance of a classification model whose output is a probability value between 0 and 1. Cross-entropy loss decreases as the predicted probability becomes closer to the actual label.
Eg, predicting a probability of 0.015 when the actual observation label is 1 is bad and results in a high loss.

The data collected was plotted into a correlation matrix, a table showing correlation coefficients between variables, as shown in **Fig 7**. Each cell in the table shows the **correlation** between two variables which helps us decide the features that we can use for training the model.

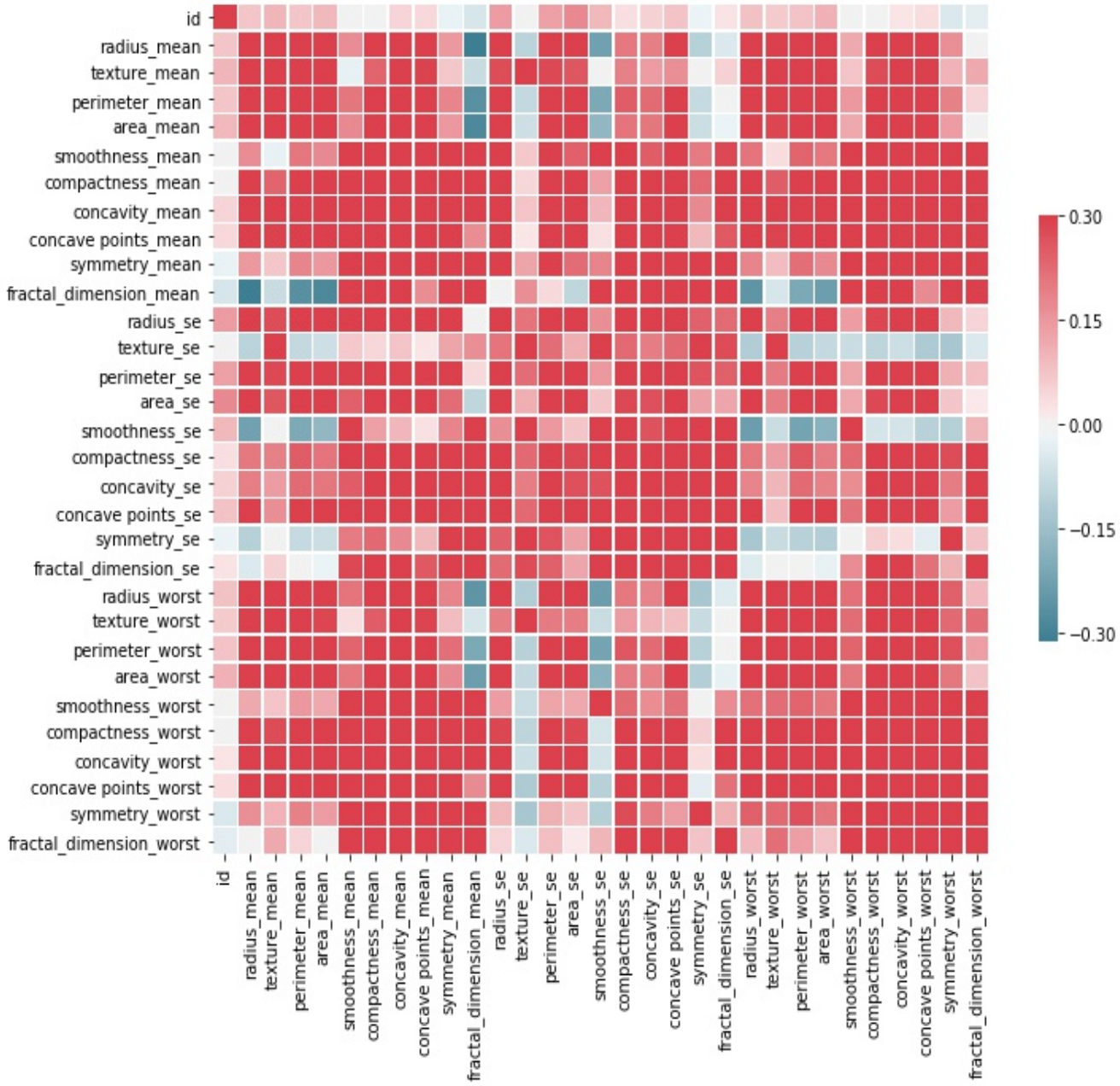


**Fig. 7 Correlation Matrix**

The correlation matrix helps us determine the correlated and uncorrelated features, some of which are seen through **Fig 8.** and **Fig 9. Fig 8** shows that the features nearly have a linear relationship, hence are positively correlated. **Fig 9** shows that the features are scattered, hence do not exhibit correlation.

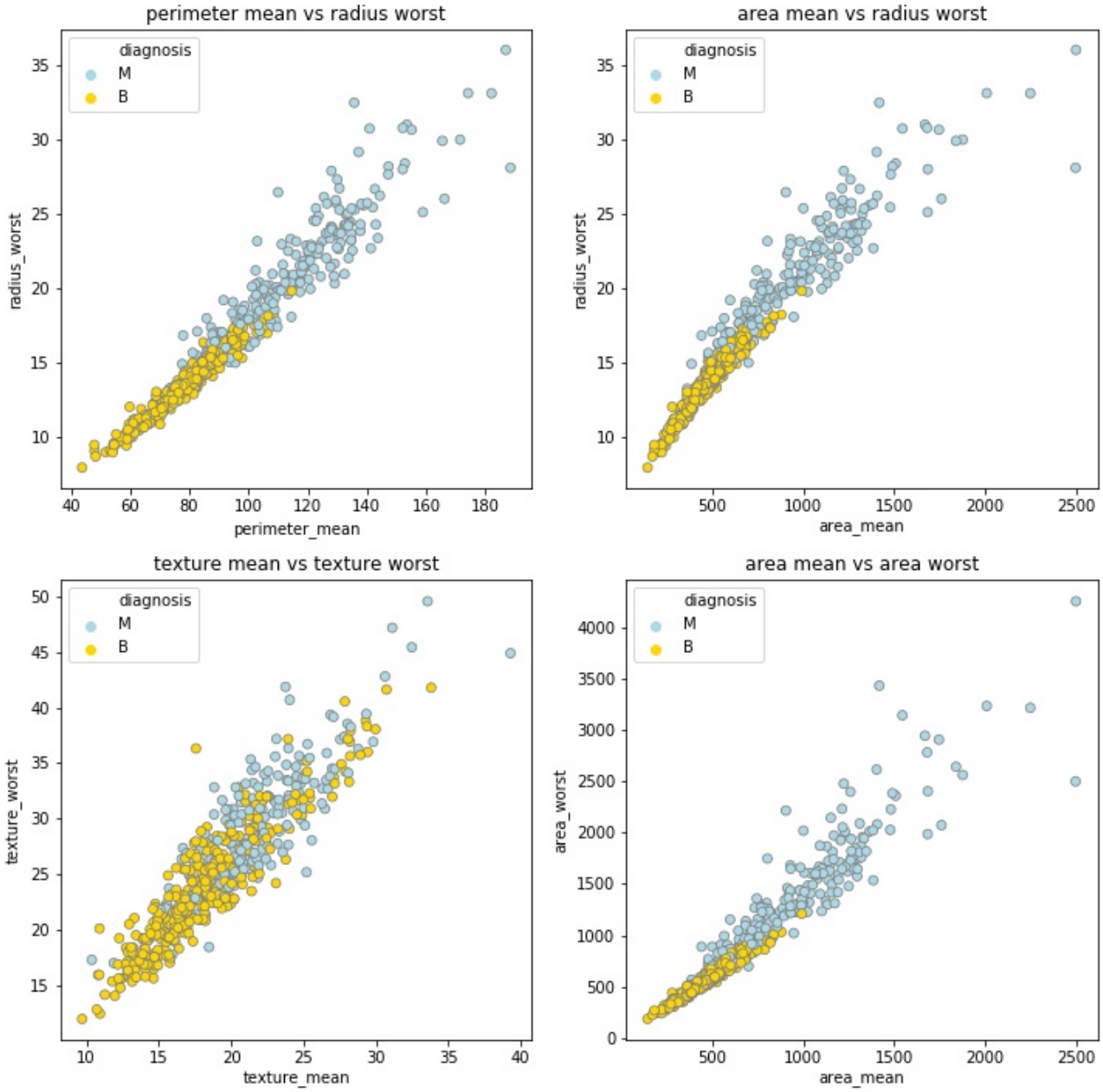
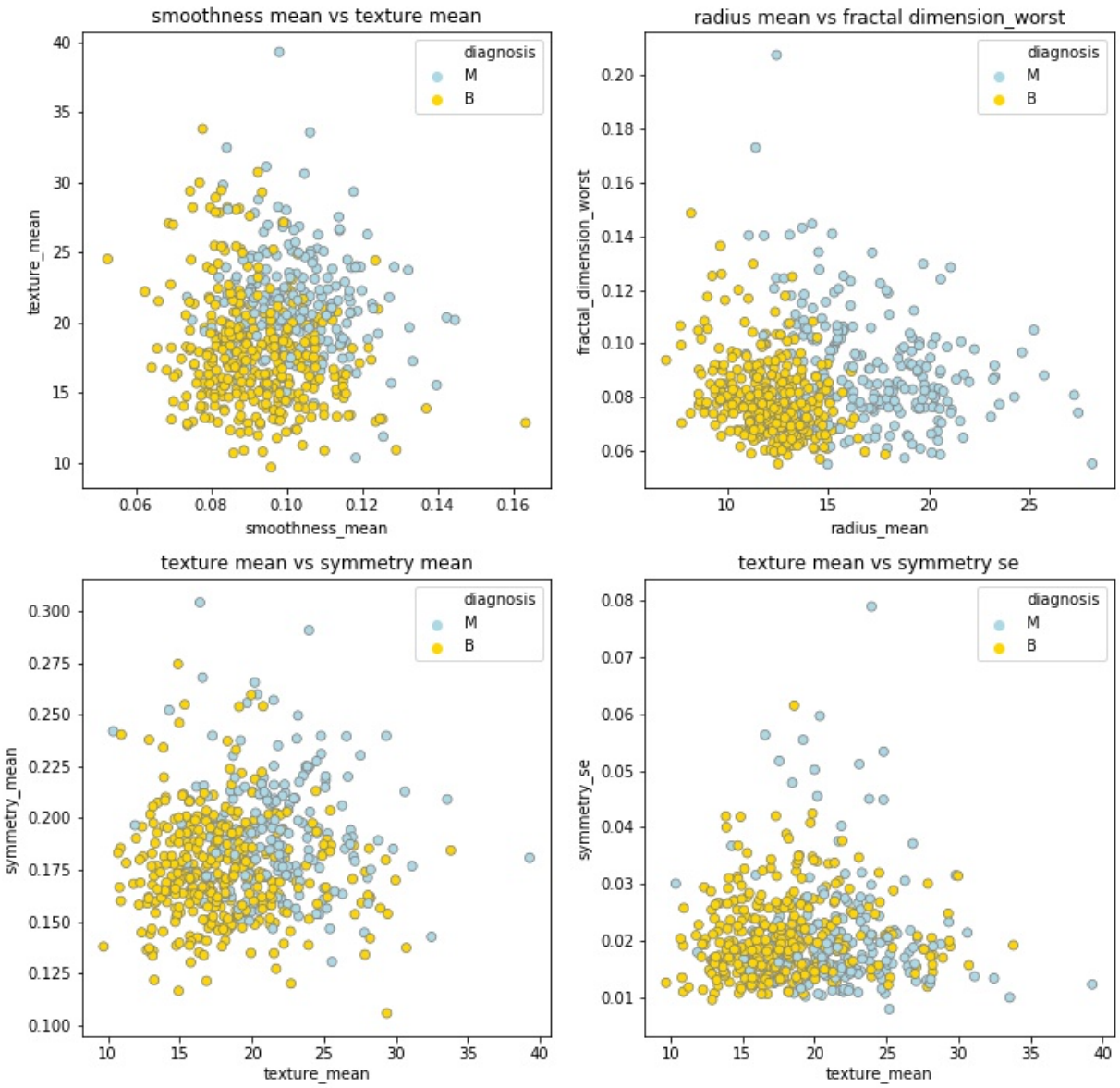



**Fig 8 Positively correlated features**

Some positively correlated features identified are :

- Perimeter mean and Radius Worst
- Area mean and Radius Worst
- Texture Mean and Texture Worst
- Area mean and Area Worst

**Fig 9 Uncorrelated features**




Some uncorrelated features identified were :

- Smoothness mean and Texture mean
- Radius mean and Worst Fractal dimensions
- Texture mean and Symmetry mean
- Texture mean vs Symmetry se

From the correlation diagrams, it was understood that radius, area and perimeter essentially contain redundant information, which describes the physical appearance of a cell. Since area and perimeter are derived from radius, it is safe to discard both those columns. All the 'worst' columns can be discarded since they are a subset of the 'mean' columns.

For the random forest classifier and extra tree classifier, we have implemented both the criteria- namely, entropy and gini impurities. Although both are often interchangeably used, for our dataset, entropy shows slightly better results. Gini prevents miscalculation, while entropy is used for exploratory analysis and can handle missing values. Entropy is apt for attributes that occur in classes.

**Gini impurity**: $\text{Gini}(E)=1 - \sum_{j=1}^{c} p_j^2$

**Entropy**: $H(E)= - \sum_{j=1}^{c} p_j log p_j$

where C is the number of classes and $p_j$ is the fraction of items labeled as class j.

## 5. COMPARISON

Our recordings were compared with literature from "Towards Data Science", as shown in **Table 1**.

| CLASSIFIER | OUR FINDINGS | LITERATURE | REASON |
|---|---|---|---|
| Naive Bayes | 95.6% | 91.6% | Apt random state was specified in our model. This is the seed used by the random number generator and gives consistent outputs. |
| Random Forest Classification | 99.1% | 98.6% | Features with high correlation were removed in our model. Random forest detects interaction between different features. But highly correlated features mask this interaction. |
| Logistic Regression | 96.4% | 95.8% | Apt random state was specified in our model. This is the seed used by the random number generator and gives consistent outputs. |

**Table 1 Comparison of models**

## 6. RESULTS

**Table 2** shows the results and accuracies obtained.

| ACCURACY | CRITERION | MODEL |
|---|---|---|
| 0.991 | Entropy | Random Forest |
| 0.982 | Gini | Random Forest |
| 0.991 | Entropy | Extra tree |
| 0.982 | Gini | Extra tree |
| 0.973 | Linear kernel | Support vector machine |
| 0.964 | - | Logistic regression |
| 0.956 | - | Naive Bayes |
| 0.999 | - | Artificial Neural Network |
| 0.991 | - | Hybrid model |

**Table 2 Results**

## REFERENCES


1. **Random Forest**- Random Decision Forest, by Tim Kam Ho
2. **Extra Tree Classifier**- Extremely randomized Trees, by Pierre Geurts , Damien Ernst, Louis Wehenkel.
3. **SVM Classifier**- Data Classification Using Support Vector Machine, by Durgesh.K.Srivatsa, Lekha Bhambhu .
4. **Logistic Regression**- An Introduction to Logistic Regression Analysis and Reporting by Chao-Ying Joanne Peng, Kuk Lida Lee, Gary M. Ingersoll
5. **Naive Bayes**- Naive Bayes for Machine Learning, by Jason Brownlee
6. **Artificial Neural Network**-
7. **L-BFGS**- Notes on CG and LM-BFGS Optimization of Logistic Regression, by Hal Daum´e III.
8. **No free lunch theorem**- Simple explanation of the no free lunch theorem of optimization, by Yu-Chi Ho, David L.Pepyne


.