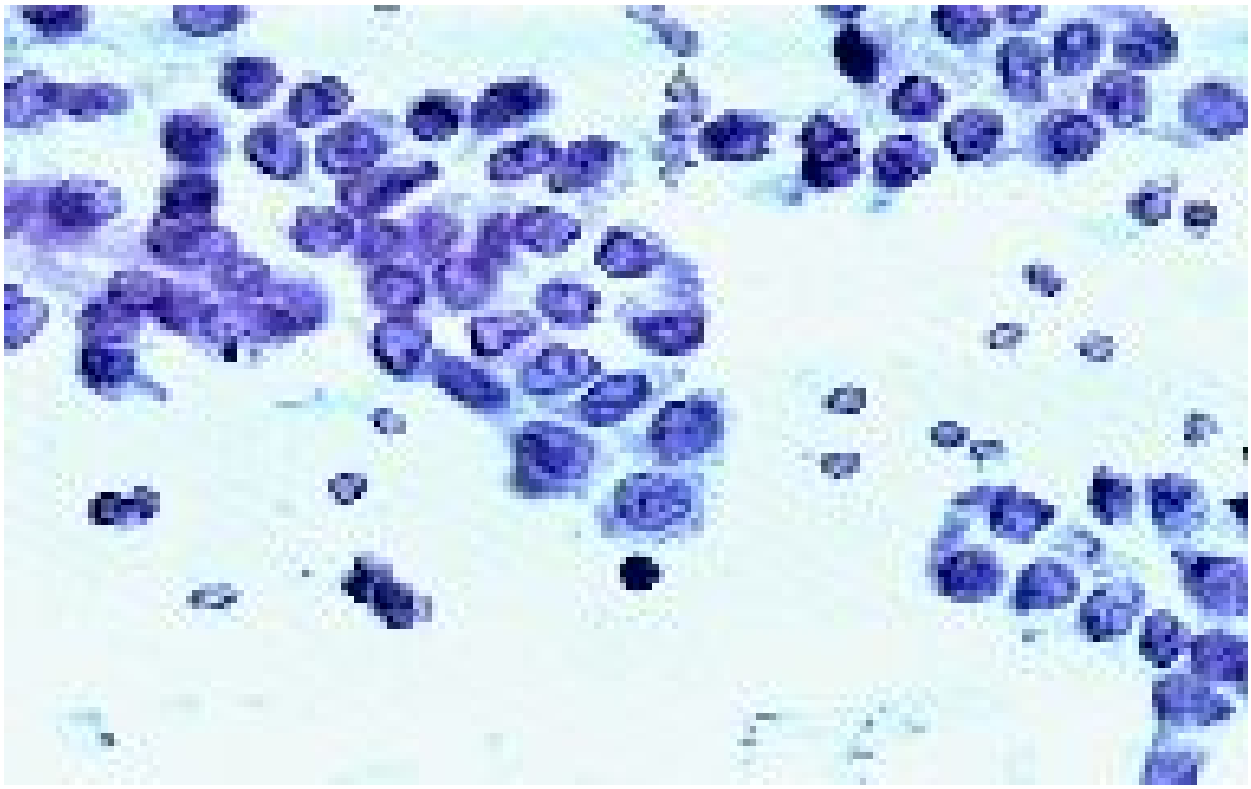
Springboard

Using Machine Learning to Diagnosis Breast Cancer

Capstone Project 1 by Chris Woods



Introduction

Breast cancer is a serious issue in the US. Here are a few statistic from breastcancer.org:

- About 1 in 8 US women, ~12%, will develop invasive breast cancer over the course of her lifetime.
- In 2019, an estimated 268,600 new cases of invasive breast cancer are expected to be diagnosed in women in the U.S., along with 62,930 new cases of non-invasive (in situ) breast cancer.

Presently in the US if a suspicious breast mass is detected either through digital palpation or mammogram, the patient is sent to have a biopsy performed on the mass. The tissue sample is sent to a lab for examination under a microscope. In breast cancer screening, needle biopsy or fine needle aspiration (FNA) is used to extract tissue from the core of the suspected mass.

Problem Statement

One issue with this procedure is the high rate of false negative test results. A false negative happens when a test result indicates there is no disease present when there actually is disease. For cancer, this would mean a test or biopsy did not find cancer when, in fact, there is cancer. This can delay diagnosis and ultimately lead to longer and more extensive treatment.

The medical community seems satisfied with the false positive rate of the needle biopsy procedure. “However, even with needle biopsies, false negative results are not common. One study looking at nearly 1,000 core needle biopsies found a false negative result rate of 2.2%. That’s just over 2 out of 100 biopsies.” Biopsy Source To me this is way too high, and I believe we can do better!

The goal of this project would be to increase the overall accuracy of breast tumor diagnosis using machine learning principles. But more specifically the rate of false negative test results. The problem with false negative results is that treatment for the tumor may be delayed six months, a year, two years, or possibly longer, with the cancer getting worse. This may result in longer more difficult treatments, or possibly unnecessary death. If we were able to, simply by using machine learning techniques, reduce the false negative test result, we could positively affect the lives of thousands of women.

Data Collection and Wrangling Summary

For this project I used the Breast Cancer Wisconsin (Diagnostic) Data Set . In this dataset 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 and given a diagnosis of benign or malignant. Ten valued features were computed for each cell nucleus:

1. **Radius** - mean of distances from center to points on the perimeter
2. **Texture** - standard deviation of gray-scale values
3. **Perimeter**
4. **Area**
5. **Smoothness** - local variation in radius lengths
6. **Compactness** - Perimeter^2 / Area - 1.0)
7. **Concavity** - severity of concave portions of the contour
8. **Concave points** - number of concave portions of the contour
9. **Symmetry**
10. **Fractal dimension** - "coastline approximation" - 1

The mean, standard error and "worst" or largest (mean of the three largest values) of these features were computed for each image, resulting in 30 features. 357 benign and 212 malignant.

This data was taken from the following study in 1992, Nuclear Feature Extraction for Breast Tumor Diagnosis. At the time diagnosis of breast tumors had usually been performed by a full surgical biopsy, an invasive surgical procedure. A relatively new procedure at the time, fine needle aspirations(FNA) provided a less invasive method, but diagnosis met with mixed success.

Overall, this was a very clean data set with no missing or NaN values in the data, with the exception of the “Unnamed: 32” column, which was removed. Additionally, the “id” column was also removed.

Recall that there are 10 feature measurements on the tumor cell nuclei. With the mean, standard error, and worst values calculated for each of these features, bringing the total number of features to 30.

When dealing with a large number of data features, 30 in this instance, it is sometimes easier to work with smaller logical blocks. I broke down the features into four groups for later analysis:

1. All Data Columns
2. Mean Data Columns
3. Standard Error Data Columns
4. Worst, or Largest Data Columns

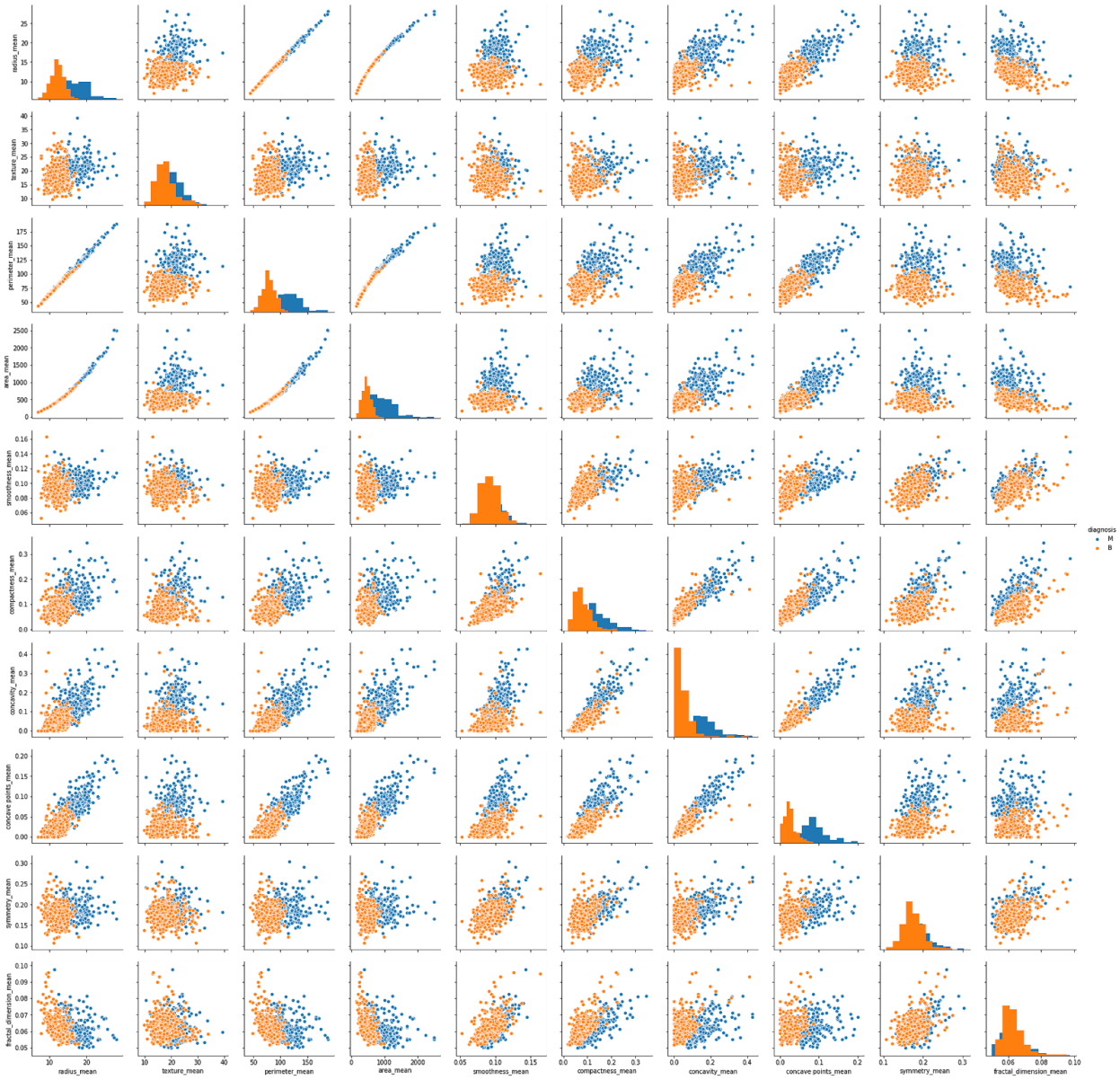
Exploratory Data Analysis and Visualization Summary

Initially, I used the **.describe()** function to get a sense of the basic statistics of the features. The major takeaway here is that the scale of some features are magnitudes of order larger than others. For example the mean of the '***area_mean***' feature is 654.889104, while the mean of the '***concave points_mean***' feature is 0.048919. Because of this I had to standardize the data for some visualizations and machine learning algorithms to make sense.

Next I wanted to get an idea of the distribution of the features. Because there are 30 features, making one chart that would clearly display the relationships between all features would be impossible. And, making 30 different charts would make comparisons cumbersome. To remedy this I grouped the 30 features into 3 groups of 10. One set representing the ***Mean*** data features, another representing the ***Standard Error*** data features and the last with ***Worst, or Largest*** data features.

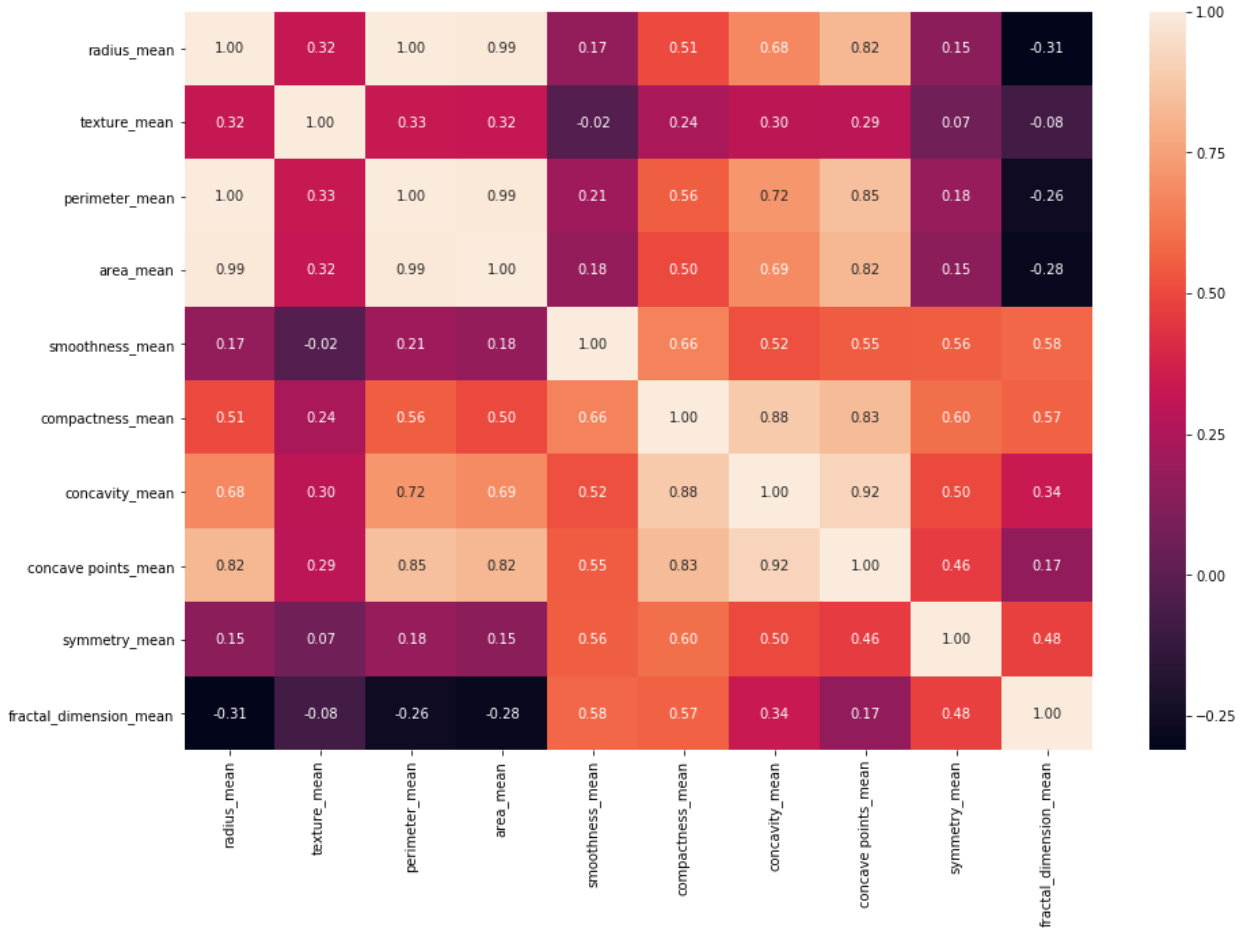
One of the most efficient way to examine the distributions of this type of data is to use histograms, scatterplots and box plots. The first visualization will be a Seaborn pairplot. It provides much the same information as a Pandas scatter matrix, a histogram to show the general distribution and scatter plots of each feature.

Additionally, the Seaborn pairplot allows us to color the scatterplot points by the diagnosis feature, **"B"** or **"M"**. And overlay the histograms by **"B"** and **"M".** Here I looked for signs of collinearity between the features and the histograms showing the distributions between **"B"** Benign and **"M"** Malignant tumors. See the Mean features pairplot below:



From the scatter plots we can clearly see several features are highly collinear. Radius, Perimeter and Area are obvious but others look highly correlated as well. In the individual feature histograms you can start to see which variables would make better predictor features. The more divergence between the Benign and Malignant distributions the better.

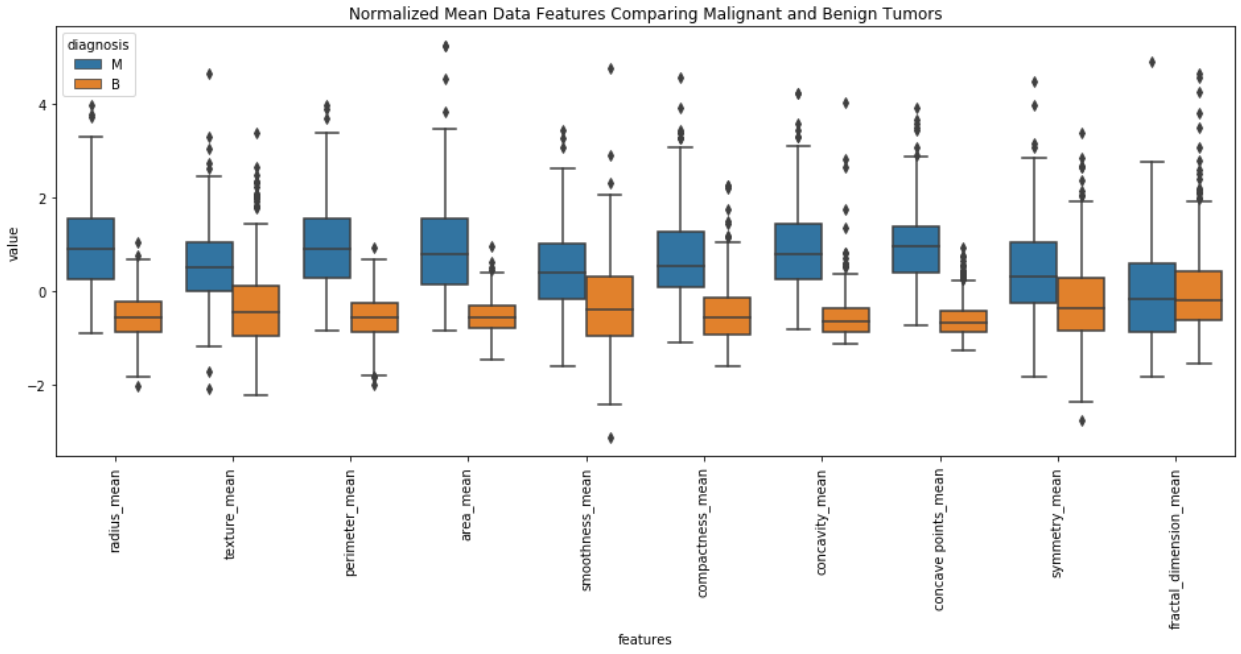
To confirm correlation I looked at the Pearson's correlation coefficients for the feature pairs.



As suspected Radius, Perimeter, and Area are almost perfectly correlated! Additionally, there are many other pairs of features that are also highly correlated. Concave Points has correlation coefficients > 0.8 with five other features.

Histograms are nice and give you a general idea of the normality and distribution of your data, to see if there is a real difference in two populations of data I prefer to see box plots.

As was mentioned earlier, because of differences in the measurement scales of the different features, I standardized the grouped features in order to make direct comparisons.



Here we can see some of the same patterns as noted before. Notice how the box plots for Radius, Perimeter, and Area are very close to the same. As well as Concavity and Concave points. You can also see that Smoothness, Symmetry, and Fractal_Dimension will not make good predictor features because their IQR's (Interquartile Range) boxes overlap. Meaning their p-value is much higher than 0.05.

Machine Learning Analysis

Analyzing the data we can see that a Supervised Machine Learning model is appropriate because all of the features are labelled. Additionally we see that the the dependant variable, 'diagnosis' is binary, B (benign) or M (malignant). This makes it a Classification problem.

For this project I will started with a basic k-NN Classifier to get a baseline feel for the accuracy we can expect from the model. Then follow-up using Logistic Regression, Random Forests, and Gradient Boosting in order to determine the best model.

In ranking or grading the models I will use two measures: Recall and AUC_ROC.

- **Recall** - Recall is the number of True Positives divided by the number of True Positives plus the number of False Negatives. Put another way, it is the number of positive predictions divided by the number of positive class values in the test data. It is also called Sensitivity or the True Positive Rate.
- **AUC-ROC** - AUC is the area under the curve of Receiver Operator Characteristic (ROC) plot False Positive Rate vs True Positive Rate at different points. The AUC represents a model's ability to discriminate between positive and negative classes. An area of 1.0 represents a model that made all predictions perfectly. An area of 0.5 represents a model as good as random.

The AUC-ROC is used to get a general sense of the accuracy of a model. But the factor that will ultimately determine the best model is *Recall of the Malignant responses*. In this project we are dealing with human beings where the outcome of our model is Benign/Malignant, or Life/Death. For this reason the 'accuracy' of our prediction is not sufficient.

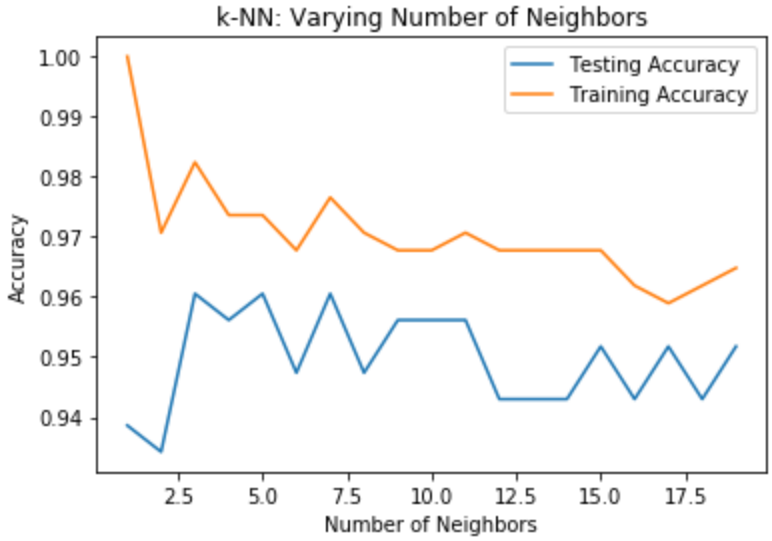
Recall was chosen as the primary scoring method because it considers False Negatives. A False Negative in our model would mean that we diagnosed a woman's tumor as being benign when in fact it was malignant. We strive for a Recall as close to 100% as possible with our model.

k-NN Classifier

The k-Nearest-Neighbors (k-NN) method of classification is one of the simplest methods in machine learning. At its most basic level, it is classification by finding the most similar data points in the training data, and making an educated guess based on their classifications.

k-NN falls under lazy learning, which means that there is no explicit training phase before classification. k-NN tends to work best on smaller data-sets that do not have many features.

In order to generate the best model I had to first determine the optimal number of "nearest neighbors" for classification purposes by comparing the model accuracy over several different k values.



Visually we can see that 3 is the optimum value for k. 6 and 7 yield the same accuracy, however would unnecessarily complicate the model.

k-NN Summary:

- **Malignant Recall = 0.8941**
- **AUC = 0.9728**

Logistic Regression

Logistic Regression is used when the dependent variable is categorical or binary (B or M). Logistic regression uses an equation as the representation, very much like linear regression.

Input values (x) are combined linearly using weights or coefficient values (referred to as the Greek capital letter Beta) to predict an output value (y). A key difference from linear regression is that the output value being modeled is a binary values (0 or 1) rather than a numeric value.

Logistic regression models the probability of the default class (e.g. Malignant). Put another way, we are modeling the probability that an input (X) belongs to the default class (Y=1), we can write this formally as:

$$P(X) = P(Y=1 \mid X)$$

In order to yield the best model you must tune the hyperparameters of the logistic regression model. Like the alpha parameter of lasso and ridge regularization, logistic regression also has a regularization parameter: C. C controls the inverse of the regularization strength. A large C can lead to an overfit model, while a small C can lead to an underfit model. In addition to C, logistic regression has a 'penalty' hyperparameter which specifies whether to use 'l1' or 'l2' regularization.

Logistic Regression Summary:

- **Malignant Recall = 0.9875**
- **AUC = 0.9986**

Top 10 Logistic Regression Features by Coefficient	
Feature	Coefficient Value
texture_worst	0.943562
symmetry_worst	0.855121
radius_se	0.809812
concave points_mean	0.783062
concavity_worst	0.712591
concave points_worst	0.651508
concavity_mean	0.641614
area_se	0.616604
radius_worst	0.602004
area_worst	0.574298

Random Forests

To say it simply: Random forest builds multiple decision trees and merges them together to get a more accurate and stable prediction. Individually, predictions made by decision trees may not be accurate, but combined together, the predictions will be closer to the mark on average.

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.

Therefore, in Random Forest, only a random subset of the features is taken into consideration by the algorithm for splitting a node. You can even make trees more random, by additionally using random thresholds for each feature rather than searching for the best possible thresholds like a normal decision tree does.

In hyper tuning the parameters of the Random Forest I used a slightly different approach. First, I used RandomizedSearchCV to get our parameter values "in the neighborhood". Then I followed up with GridSearchCV to determine the final values of the parameters.

Random Forest Summary:

- **Malignant Recall = 0.9375**
- **AUC = 0.9975**

Top 10 Random Forest Features by Importance	
Feature	Importance
concave points_mean	0.152267
perimeter_worst	0.118365
concave_points_worst	0.099178
area_worst	0.088694
radius_worst	0.080277
concavity_mean	0.078913
perimeter_mean	0.056
area_mean	0.042276
area_se	0.03218
concavity_worst	0.0298

Gradient Boosting

The term 'Boosting' refers to a family of algorithms which converts weak learner to strong learners. In our data we have 30 features and are trying to classify a tumor as benign or malignant. Individually, each of these features is not powerful enough to classify a tumor mass as benign or malignant and are therefore known as weak learners.

Gradient boosting, trains many model sequentially. Each new model gradually minimizes the loss function of the whole system using Gradient Descent method. The learning procedure consecutively fit new models to provide a more accurate estimate of the response variable.

The principle idea behind this algorithm is to construct new base learners which can be maximally correlated with negative gradient of the loss function, associated with the whole ensemble.

Gradient Boosting Summary:

- **Malignant Recall = 0.9500**
- **AUC = 0.9972**

Top 10 Gradient Boosting Features by Importance	
Feature	Importance
concave_points_worst	0.013168
symmetry_worst	0.013088
concave points_mean	0.012143
texture_worst	0.011579
concavity_worst	0.011134
texture_mean	0.010392
area_worst	0.010049
radius_worst	0.009698
perimeter_mean	0.008518
perimeter_worst	0.00831

Conclusion

As previously stated, ***The goal of this project would be to increase the overall accuracy of breast tumor diagnosis using machine learning principles.*** The table summarizes the results.

	Machine Learning Algorithm Performance			
Metric	**k-NN**	**Logistic Recression**	**Random Forest**	**Gradient Boosting**
Recall	**0.8941**	***0.9875***	**0.9375**	**0.9500**
ROC-AUC	**0.9728**	***0.9986***	**0.9975**	**0.9972**

Using logistic regression, on a relatively small data sample I was to reduce the false negative rate for malignant tumor diagnosis from 2.2% to 1.25%, or a ~ 57% reduction. This a very promising first step with regard to integrating machine learning and medical diagnosis.

The data used in this project was relatively small, 569 observations. It would be interesting to see if the conclusion found here would be confirmed with additional observations, or if a different ML model would perform better than Logistic Regression.