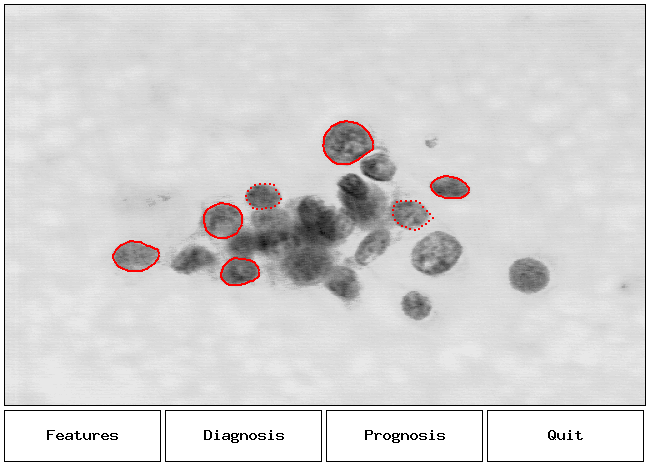
**Breast Tumor Diagnosis**

In this project, my initial goal was to create a feedforward neural network for classification of breast tumors (malignant/benign. I am using a dataset from University of California Irvine Machine Learning Repository that consists of 569 labeled samples with 32 attributes. “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.”



**Data Preprocessing**

The performance of a learning algorithm is sensitive to the quality of the data and the amount of useful information in it. Thus, it is almost always necessarily to do some data preprocessing before feeding it to a learning algorithm. For instance, removing or imputing missing values, handling categorical data, scaling and so on. Since I downloaded from University of California Irvine Machine Learning Repository, my dataset had no missing values and did not require much preprocessing work. Although, I had to encode two class labels, since my goal was classification. I also split my data in test and training parts to ensure that my algorithms generalize well.

**Information on attributes:**

1. ID number
2. Diagnosis (M = malignant, B = benign)

“All 10 features are real-valued, all of them are numerically modeled such that larger values will typically indicate a higher likelihood of malignancy”. For each attribute the mean, standard error (SE), and “worst” (mean of 3 largest values) were computed. So, in my dataset field 4 corresponds to Mean texture, field 14 – Texture SE, field 24- Worst Texture.

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

Another general requirement that I satisfied was standardizing the features to center them around 0 with a standard deviation of 1. Why is it important? In my project I used a feedforward neural network, which is constructed from sigmoid neurons and uses the gradient descent optimization algorithm. So, if features are on different scales, their weights will not update with the same rate. Another utility of feature scaling is in finding directions maximizing the variance of the data, which is done by using Linear Discriminant Analysis. Not having features on the same scale would mean emphasizing variables with larger scalars.

**Feature engineering**

Feature selection and extraction are dimensionality reduction techniques are useful for the following reasons:

Curse of Dimensionality:

The problem of CD is that the size of the training set needed to train a classifier grows exponentially with the number of dimensions. If we exceed the maximum number of features for a certain number of training examples, we will run into the problem of overfitting and therefore our classifier will show a poor out-of-sample performance. Although we lose some information by discarding some features, it is compensated by a more accurate mapping in lower-dimensional space.

Better interpretation and visualization:

Less features means a simpler explained model, data visualization in 2 or 3 dimensions enables us to see the structure, groups, outliers.

Reduced time & space complexity.

In my project, I decided to use feature extraction, namely Linear Discriminant Analysis (LDA). As a dimensionality reduction technique, it projects a dataset onto a lower-dimensional space and it also maintains good class-separability. LDA is similar to Principal Component Analysis (PCA), “but in addition to finding the component axes that maximize the variance of our data (PCA), we are additionally interested in the axes that maximize the separation between multiple classes (LDA).”