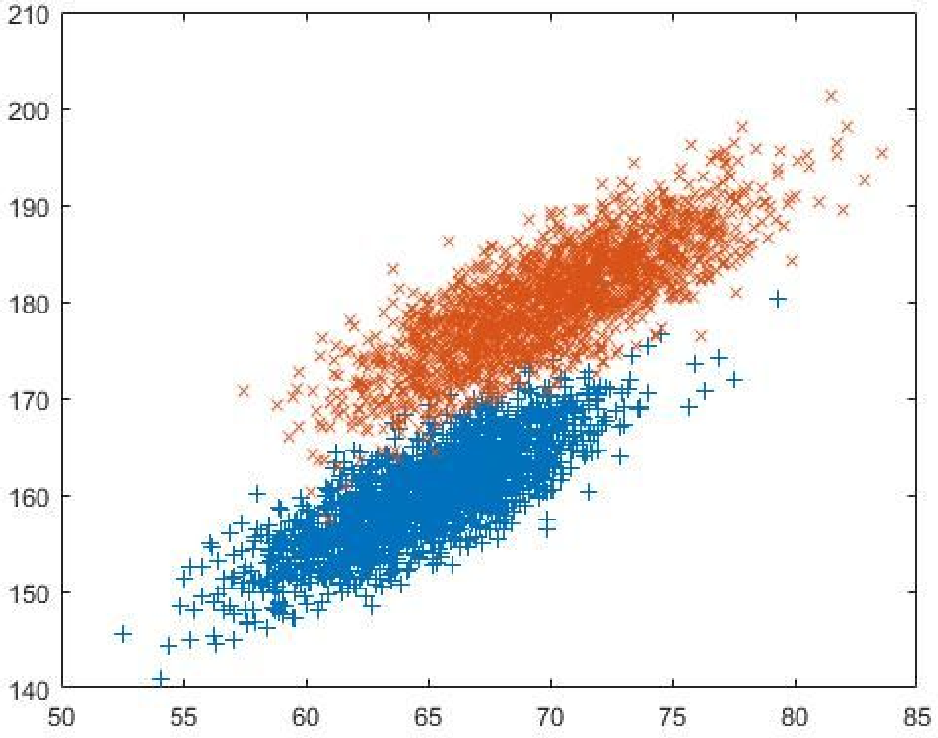
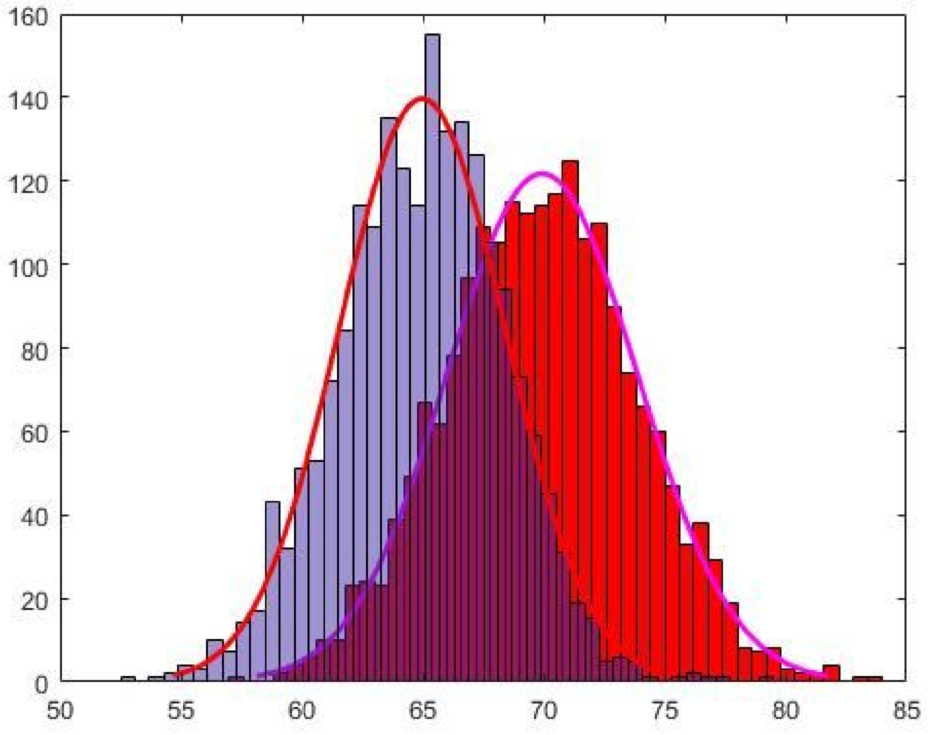
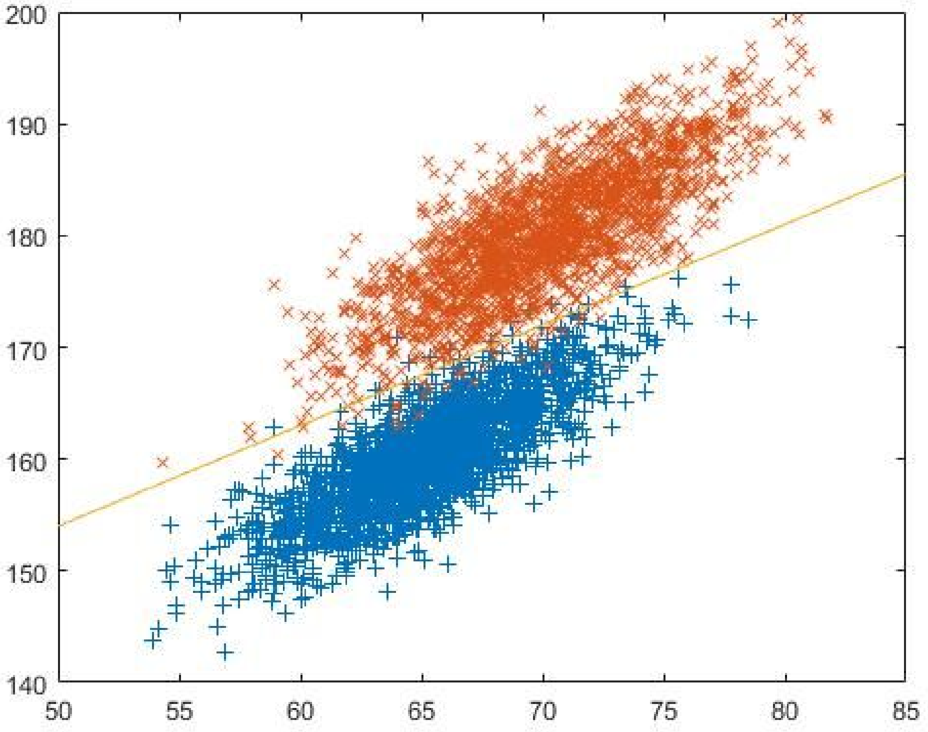
To generate a random sample of normally distributed data we created a covariance matrix and correlation matrix. This allowed us to generate a multivariable normally distributed sample of 2000 men and 2000 women around a set mean for height and weight with predetermined standard deviations. Because we used a covariance matrix we can break out any axis of data such as height and analyse it individually. We used a random function that was reproducible for the sake of presentation. From this sample we manually created a distinguishing line and calculated the accuracy, error, true positive rate, true negative rate, false positive rate, and false negative rate. It is interesting how adding a second variable substantially increased the accuracy, from .7403 to .9480.

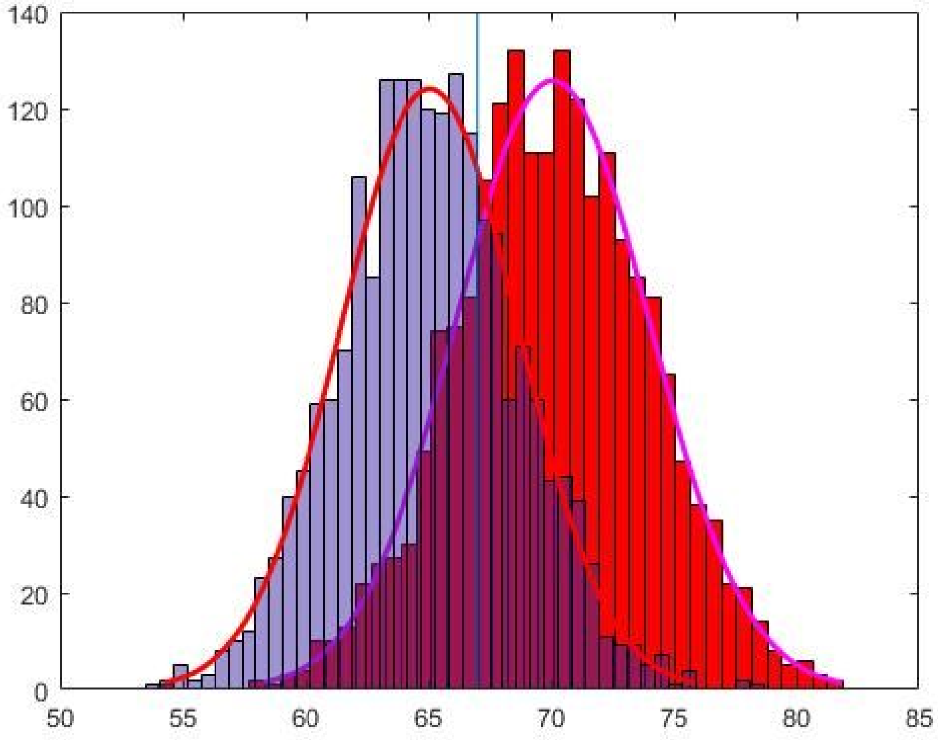
Data Without Separation Lines:





Data with Separation Lines:





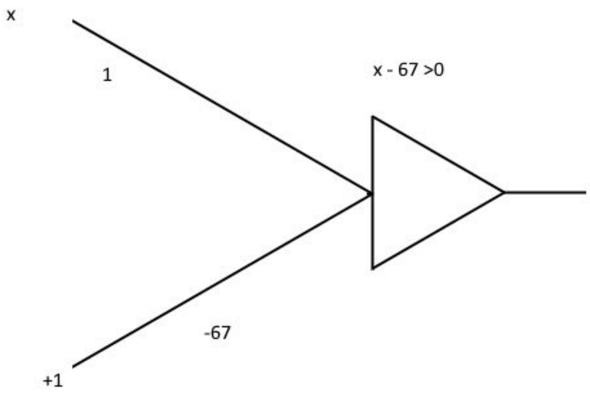
Contrast Artificial Neuron:

For our sample the equations are $x = 67$ while distinguishing based solely on height, and $y = 0.9x + 104$ when distinguishing by both height and weight

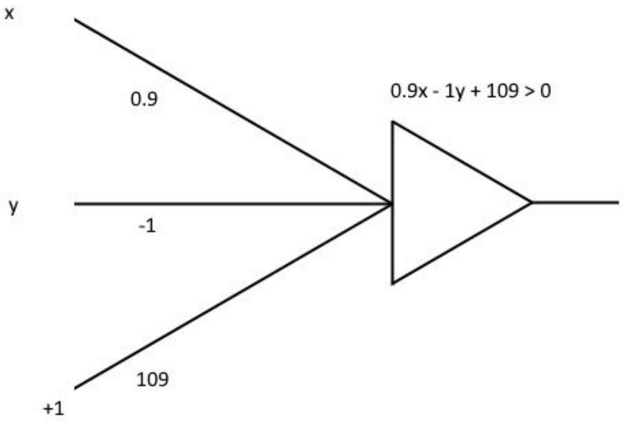
Equation Scenario A: $1x - 67 > 0$
Equation Scenario B: $0.9x -1y + 104 > 0$

Weights of Scenario A: [1 -67]
Weights of Scenario B: [1.3 1 -257]

Height Artificial Neuron:



Height and Weight Artificial Neuron:



Scenario A:

Error:	0.2597
Accuracy:	0.7403
True Positive Rate:	0.7720
True Negative Rate:	0.7085
False Positive Rate:	0.2915
False Negative Rate:	0.2280

Scenario B:

Error:	0.0520
Accuracy:	0.9480
True Positive Rate:	0.9030
True Negative Rate:	0.9930
False Positive Rate:	0.0070
False Negative Rate:	0.0970

CODE:

```
mu = [70,180];                                        %mean for height/weight in
inches/pounds - men
ExpSigma = [4  6];

ExpCorrC = [1.0 0.8

          0.8  1.0];

sigma = corr2cov(ExpSigma, ExpCorrC);                 %covariance
matrix(https://en.wikipedia.org/wiki/Covariance_matrix)
r = mvnrnd(mu,sigma,2000);



mu = [65,160];                                        %mean for height/weight in
inches/pounds - women
ExpSigma = [3.5  5];

ExpCorrC = [1.0 0.8

          0.8  1.0];

sigma = corr2cov(ExpSigma, ExpCorrC);                 %covariance
matrix(https://en.wikipedia.org/wiki/Covariance_matrix)
[standard_div] = cov2corr(sigma);

r2 = mvnrnd(mu,sigma,2000);



figure

disp("Height");                                       %create new figure(graph window)

h1 = histfit(r(:,1));                                 %histfit creates histogram with
bell curve
set(h1(1),'facecolor','r'); set(h1(2),'color','m')  %<histfit>(1) = bar color,
<histfit>(2) = line color
hold on;                                              %keep in figure

h2 = histfit(r2(:,1));

alpha(h2,.5)

line([67 67], ylim); %line equation x >= 67



figure
```

```
disp("Height v Weight");
plot(r2(:,1),r2(:,2),'+',r(:,1),r(:,2),'X')
hold on;
x = 50:1:85;
y = 0.9*x+109; %line equation y = 0.9x + 109
plot(x,y)


disp("Men");
disp(r);
disp("Women");
disp(r2);


%{
S = std(r)
M = mean(r)

S2 = std(r2)
M2 = mean(r2)
%}


%Scenario A - height only
error = 0;
tp = 0;
tn = 0;
fp = 0;
fn = 0;
for a = 1:2000
   if r(a,1) < 67      %error if men less than 67
       error = error + 1;
       fn = fn + 1;
   else
       tp = tp + 1;
   end
   if r2(a,1) >= 67    %error if women greater than 67
       error = error + 1;
       fp = fp + 1;
   else
       tn = tn + 1;
   end
end
error = error / 4000;
disp("Scenario A")
disp("Error Rate:")
disp(error)
disp("Accuracy Rate:")
disp(1-error)
disp("True Positive Rate")
disp(tp / 2000)
disp("True Negative Rate")
disp(tn / 2000)
disp("False Positive Rate")
disp(fp / 2000)
disp("False Negative Rate")
disp(fn / 2000)

%Scenario B - height and weight
error = 0;
tp = 0;
tn = 0;
```

```
fp = 0;
fn = 0;
for a = 1:2000
   x = r(a,1);
   yhat = 0.9*x+109;
   if r(a,2) < yhat
       error = error + 1;          %error if below line for men
       fn = fn + 1;
   else
       tp = tp + 1;
   end

   x = r2(a,1);
   yhat = -1.3*x+257;
   if r2(a,2) >= yhat
       error = error + 1;          %error if above line for women
       fp = fp + 1;
   else
       tn = tn + 1;
   end
end
error = error / 4000;
disp("Scenario B")
disp("Error Rate:")
disp(error)
disp("Accuracy Rate:")
disp(1-error)
disp("True Positive Rate")
disp(tp / 2000)
disp("True Negative Rate")
disp(tn / 2000)

disp("False Positive Rate")

disp(fp / 2000)

disp("False Negative Rate")

disp(fn / 2000)
```

Work Distribution:

Sample Generation:	Jacob Roberts
Data Plotting:	Jacob Roberts
Estimate/Plot Separation Line:	Jacob Wilson
Determine Equation:	Jacob Wilson
Contrast Equation With Artificial Neuron:	Jacob Roberts
Report Error, Accuracy, etc.:	Jacob Wilson

Data: