Import the data

data = readmatrix('fire.csv');
x = data(:,1:3);
y = data(:,4);
m = length(y);

Visualization of the data

histogram(x(:,3),10);

plot(x(:,3),y,'o')
Normalize the features and transform the output

```matlab
y2 = log(1+y);
for i = 1:3
    x2(:,i) = (x(:,i) - min(x(:,i)))/(max(x(:,i)) - min(x(:,i)));
end
histogram(x2(:,3),10);
```
Train an artificial neural network (ANN)

xt = x2';
yt = y2;
hiddenLayerSize = 7;
net = fitnet(hiddenLayerSize);
net.divideParam.trainRatio = 70/100;
net.divideParam.valRatio = 30/100;
net.divideParam.testRatio = 0/100;
[net,tr] = train(net, xt, yt);

Performance of the ANN network

```
yTrain = exp(net(xt(:,tr.trainInd)))-1;
yTrainTrue = exp(yt(tr.trainInd))-1;
sqrt(mean((yTrain - yTrainTrue).^2))
```

ans = 7.6004

```
yVal = exp(net(xt(:,tr.valInd)))-1;
yValTrue = exp(yt(tr.valInd))-1;
sqrt(mean((yVal - yValTrue).^2))
```

ans = 7.6188

Visualize the predictions from the ANN model

```
plot(yTrainTrue,yTrain,'x'); hold on;
plot(yValTrue,yVal,'o');
plot(0:100,0:100); hold off;
```

Optimize the number of neurons in the hidden layer

```
for i = 1:60
```
% defining the architecture of the ANN
hiddenLayerSize = i;
net = fitnet(hiddenLayerSize);
net.divideParam.trainRatio = 70/100;
net.divideParam.valRatio = 30/100;
net.divideParam.testRatio = 0/100;

% training the ANN
[net,tr] = train(net, xt, yt);

% determine the error of the ANN
yTrain = exp(net(xt(:,tr.trainInd)))-1;
yTrainTrue = exp(yt(tr.valInd))-1;
yTrainTrue = exp(yt(tr.trainInd))-1;
yVal = exp(net(xt(:,tr.valInd)))-1;
rmse_train(i) = sqrt(mean((yTrain - yTrainTrue).^2)); % RMSE of training set
rmse_val(i) = sqrt(mean((yVal - yValTrue).^2)); % RMSE of validation set
end

Select the optimal number of neurons in the hidden layer

plot(1:60,rmse_train); hold on;
plot(1:60,rmse_val); hold off;