Homework #3

Build a logistic regression model to predict heart disease

Import data and preview

```
data1 = readtable("../cleveland_data_revised.xlsx");
head(data1,5)
```

```
ans = 5×14 table
```

...

	Age	SEX	chestPain	restingBP	cholest	highBloodSugar	ECG
1	63	1	1	145	233	1	2
2	67	1	4	160	286	0	2
3	67	1	4	120	229	0	2
4	37	1	3	130	250	0	0
5	41	0	2	130	204	0	2

Build model that predicts whether individual has heart disease.

```
data1 = rmmissing(data1);
```

```
X = data1{:,1:13};
Y = double(data1.diseaseSeverity > 0);    % convert logical to double
Y_cat = categorical(Y);      % convert double to category for mnrfit
[mdl1,~,stats1] = mnrfit(X,Y_cat);
```

Build model that predicts disease severity.

```
Y2 = data1.diseaseSeverity;
mdl2 = fitlm(X,Y2);
```

Evaluate both models using three-fold cross validation.

```
indices = crossvalind('kfold',size(data1,1),3);

accuracy = zeros(1,3);
precision = zeros(1,3);
recall = zeros(1,3);

rank = zeros(1,3);
RMSE = zeros(1,3);
prcnt_error = zeros(1,3);

for i = 1:3
    test = indices == i;
    train = ~test;
```

```
    Xtest = X(test,:);
    Ytest = Y(test,:);
    Ytest2 = Y2(test,:);

    probability = mnrval(mdl1,Xtest);
    Ypred = round(probability(:,2));
    accuracy(i) = sum(Ypred==Ytest)/length(Ypred);
    precision(i) = sum(Ypred==1 & Ytest==1)/sum(Ypred==1);
    recall(i) = sum(Ypred==1 & Ytest==1)/sum(Ytest==1);

    Ypred2 = predict(mdl2, Xtest);
    rank(i) = corr(Ytest2,Ypred2,"type","Spearman");
    RMSE(i) = sqrt(mean((Ypred2-Ytest2).^2));
    %prcnt_error(i) = mean((Ypred2-Ytest2) ./ Ytest2 * 100)
end
```

Model 1 Performance

```
mean(accuracy)
```

```
ans = 0.8485
```

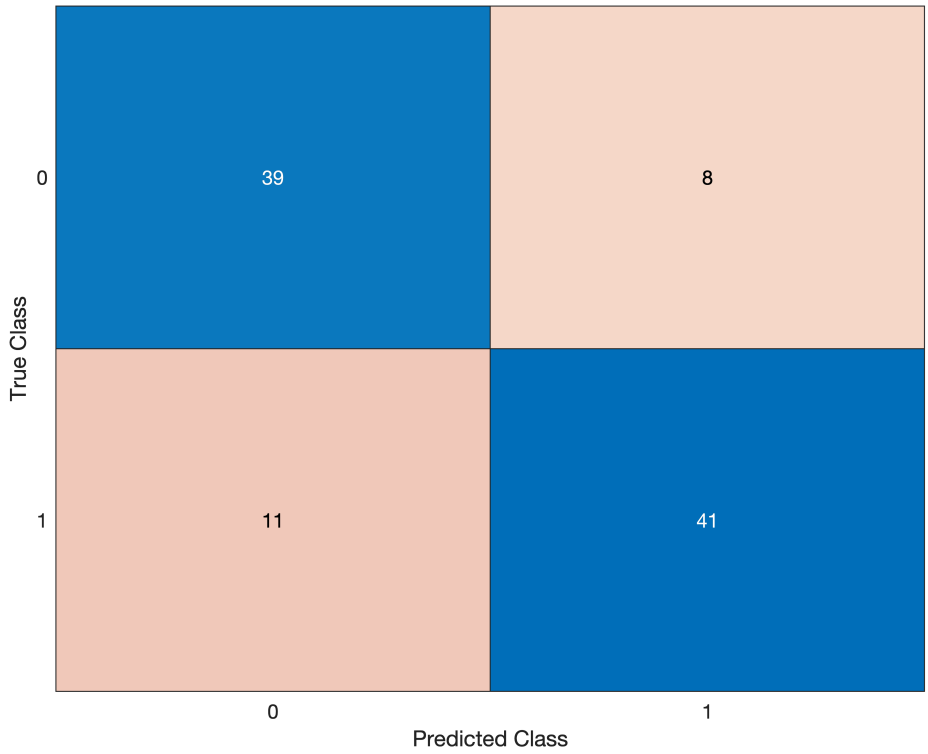
```
mean(precision)
```

```
ans = 0.8496
```

```
mean(recall)
```

```
ans = 0.8175
```

```
confusionchart(Ytest,Ypred);   % uses last cross-validation iteration
```



Model 2 Performance

```
mean(rank)
```

```
ans = 0.7501
```

```
mean(RMSE)
```

```
ans = 0.8095
```

```
%mean(prcnt_error)
```

Compare models to distribution of 100 random guesses using a t-test

```
guess_accuracy = zeros(100,1);
guess_RMSE = zeros(100,1);
for i = 1:100
    mdl1_guess = Y(randperm(length(Ytest)));
    mdl2_guess = Y2(randperm(length(Ytest2)));

    guess_accuracy(i,1) = sum(mdl1_guess == Ytest)/length(Ytest);
    guess_RMSE(i,1) = sqrt(mean((Ypred2-Ytest2).^2));
    %accuracy_guess2(i,1) = sum(mdl2_guess == Ytest2)/length(Ytest2);
end
mean(guess_accuracy)
```

```
ans = 0.4905
```

```
mean(guess_RMSE)
```

```
ans = 0.7990
```

```
[h_mdl1, p_mdl1] = ttest2(accuracy,guess_accuracy)
```

```
h_mdl1 = 1
p_mdl1 = 7.5711e-23
```

```
[h_mdl2, p_mdl2] = ttest2(RMSE,guess_RMSE)
```

```
h_mdl2 = 1
p_mdl2 = 1.5664e-06
```

Most important features from each model, based on p-value

Model 1

```
[p1,m1] = mink(stats1.p,5)
```

```
p1 = 5×1
    0.0000
    0.0006
    0.0026
    0.0072
    0.0105
m1 = 5×1
    13
    14
     4
     3
     1
```

```
% Must subtract 1 from index to account for intercept term
data1.Properties.VariableNames([12,13,3,2])
```

```
ans = 1×4 cell array
    {'fluoroscopy'}    {'thalliumTest'}    {'chestPain'}    {'SEX'}
```

Model 2

```
[p2,m2] = mink(mdl2.Coefficients.pValue,5)
```

```
p2 = 5×1
    0.0000
    0.0000
    0.0008
    0.0011
    0.0348
m2 = 5×1
    13
    14
     4
    11
     9
```

```
data1.Properties.VariableNames([12,13,3,10,8])
```

```
ans = 1×5 cell array
    {'fluoroscopy'}    {'thalliumTest'}    {'chestPain'}    {'ECGDepression'}    {'maxHR'}
```

Brain cancer survival data set

Import and inspect data

```
data2 = readtable("Top 100 Genes.xlsx");
tail(data2,5)
```

```
ans = 5×104 table
```

...

	PATIENT_ID	SurvivalDays	SurvivalDaysPF	Test	LIX1L	NEU4	TTC12
1	'GSM1912975'	313	313	0	0.1122	-1.5424	1.1803
2	'GSM1912976'	962	277	0	0.8193	0.9344	-0.7882
3	'GSM1912977'	826	826	0	0.3891	1.7826	-1.4738
4	'GSM1912978'	257	257	0	-0.3570	-1.5359	1.1740
5	'GSM1912979'	593	395	1	0.4702	2.6611	-2.0240

Create train and tests sets X and Y

```
Xtrain = data2{data2.Test==0,5:end};
Xtest = data2{data2.Test==1,5:end};

Ytrain = data2{data2.Test==0,3};
Ytest = data2{data2.Test==1,3};
```

Build lasso model

```
[B1, Fit] = lasso(Xtrain,Ytrain,'CV',10);
B1_coeff = B1(:,Fit.IndexMinMSE);
B1_intercept = Fit.Intercept(Fit.IndexMinMSE)
```

```
B1_intercept = 459.5216
```

Build stepwise model

```
[B2,~,~,inmodel,stats] = stepwisefit(Xtrain,Ytrain);
```

```
Initial columns included: none
Step 1, added column 1, p=1.14656e-05
Step 2, added column 87, p=0.000780954
Step 3, added column 67, p=0.00203177
Step 4, added column 16, p=0.0131333
Step 5, added column 94, p=0.021837
Final columns included:  1 16 67 87 94
    'Coeff'        'Std.Err.'    'Status'    'P'
    [ 331.2930]    [ 61.5834]    'In'        [3.7580e-06]
```

[12.6648]	[29.8070]	'Out'	[0.6733]
[-23.1072]	[37.5795]	'Out'	[0.5423]
[7.9953]	[33.8602]	'Out'	[0.8146]
[44.1486]	[72.8926]	'Out'	[0.5483]
[75.1289]	[71.2068]	'Out'	[0.2981]
[-84.6673]	[84.8098]	'Out'	[0.3244]
[27.2049]	[38.5893]	'Out'	[0.4851]
[-52.8482]	[29.3072]	'Out'	[0.0793]
[-17.4918]	[88.8986]	'Out'	[0.8451]
[-28.4037]	[65.8510]	'Out'	[0.6687]
[-39.2505]	[43.8732]	'Out'	[0.3766]
[-23.6395]	[54.2047]	'Out'	[0.6652]
[41.0555]	[57.3742]	'Out'	[0.4786]
[-6.2000]	[41.0978]	'Out'	[0.8809]
[272.6832]	[74.9647]	'In'	[7.9575e-04]
[23.2529]	[75.3805]	'Out'	[0.7594]
[54.9158]	[56.7711]	'Out'	[0.3395]
[-97.0980]	[77.9745]	'Out'	[0.2207]
[85.3805]	[76.0550]	'Out'	[0.2686]
[-109.8801]	[72.3577]	'Out'	[0.1371]
[-14.0607]	[29.7101]	'Out'	[0.6387]
[-65.9509]	[67.0209]	'Out'	[0.3313]
[-1.4072]	[17.8184]	'Out'	[0.9375]
[-33.6010]	[44.3738]	'Out'	[0.4536]
[63.1477]	[75.2082]	'Out'	[0.4064]
[26.3232]	[44.0971]	'Out'	[0.5541]
[15.1599]	[54.5696]	'Out'	[0.7827]
[-84.5155]	[86.9530]	'Out'	[0.3372]
[-61.5976]	[64.3388]	'Out'	[0.3444]
[14.1874]	[42.4973]	'Out'	[0.7403]
[27.5928]	[43.9999]	'Out'	[0.5343]
[-4.0651]	[41.1456]	'Out'	[0.9218]
[66.5157]	[103.9025]	'Out'	[0.5259]
[-11.5037]	[61.1865]	'Out'	[0.8519]
[-1.5527]	[22.5276]	'Out'	[0.9454]
[-75.1915]	[71.8999]	'Out'	[0.3023]
[1.4089]	[16.4612]	'Out'	[0.9322]
[6.5972]	[62.8855]	'Out'	[0.9170]
[17.6227]	[68.2993]	'Out'	[0.7978]
[-2.3037]	[65.8195]	'Out'	[0.9723]
[-37.0691]	[39.3440]	'Out'	[0.3521]
[42.3775]	[85.4106]	'Out'	[0.6226]
[105.0859]	[75.3442]	'Out'	[0.1712]
[6.6225]	[27.0814]	'Out'	[0.8081]
[22.0274]	[74.4685]	'Out'	[0.7690]
[49.7224]	[56.8578]	'Out'	[0.3873]
[26.3420]	[65.3027]	'Out'	[0.6889]
[29.1268]	[50.0003]	'Out'	[0.5636]
[10.0925]	[121.9010]	'Out'	[0.9345]
[-7.1528]	[59.9455]	'Out'	[0.9056]
[72.6795]	[89.4912]	'Out'	[0.4218]
[33.5334]	[75.0535]	'Out'	[0.6576]
[-13.7545]	[45.2840]	'Out'	[0.7630]
[-101.1998]	[100.8522]	'Out'	[0.3220]
[15.3976]	[60.9681]	'Out'	[0.8020]
[-20.5558]	[53.5616]	'Out'	[0.7033]
[15.8468]	[19.6565]	'Out'	[0.4252]
[118.9658]	[78.5162]	'Out'	[0.1380]
[21.2026]	[65.7408]	'Out'	[0.7488]
[6.4107]	[61.4893]	'Out'	[0.9175]
[79.4708]	[80.2296]	'Out'	[0.3282]
[-25.5018]	[24.8898]	'Out'	[0.3120]
[58.7712]	[75.3145]	'Out'	[0.4400]
[13.2043]	[65.6129]	'Out'	[0.8416]
[-15.1778]	[19.7297]	'Out'	[0.4465]

```
[ -68.6378]    [ 20.4231]    'In'        [    0.0017]
[  23.5800]    [ 62.5281]    'Out'       [    0.7082]
[ -32.5209]    [ 38.2286]    'Out'       [    0.4003]
[ -22.4227]    [ 90.0085]    'Out'       [    0.8046]
[  48.6536]    [ 59.2750]    'Out'       [    0.4169]
[  11.8101]    [ 62.0225]    'Out'       [    0.8500]
[  88.6327]    [ 46.3177]    'Out'       [    0.0632]
[   8.1990]    [ 86.4273]    'Out'       [    0.9249]
[  13.2463]    [ 76.6554]    'Out'       [    0.8637]
[  12.8070]    [117.7579]    'Out'       [    0.9140]
[ -34.6839]    [ 31.5018]    'Out'       [    0.2778]
[  38.2428]    [ 34.9757]    'Out'       [    0.2811]
[   8.6532]    [ 71.3582]    'Out'       [    0.9041]
[  57.3700]    [ 82.9868]    'Out'       [    0.4936]
[ -42.9050]    [ 38.6773]    'Out'       [    0.2743]
[  34.1271]    [ 70.2194]    'Out'       [    0.6298]
[  -3.8694]    [ 20.9028]    'Out'       [    0.8541]
[  37.1350]    [ 86.8685]    'Out'       [    0.6714]
[ -66.9142]    [124.3847]    'Out'       [    0.5937]
[ -34.1969]    [ 71.8780]    'Out'       [    0.6370]
[-177.3820]    [ 42.2309]    'In'        [1.4977e-04]
[  19.4743]    [ 70.0543]    'Out'       [    0.7825]
[   6.4093]    [ 71.5706]    'Out'       [    0.9291]
[ -37.7863]    [ 72.3943]    'Out'       [    0.6047]
[  49.8952]    [ 63.9588]    'Out'       [    0.4402]
[ 110.1510]    [ 59.5287]    'Out'       [    0.0720]
[  43.5849]    [ 86.6252]    'Out'       [    0.6178]
[-174.3336]    [ 72.9783]    'In'        [    0.0218]
[ -96.7626]    [ 68.1004]    'Out'       [    0.1635]
[  43.7342]    [ 65.1730]    'Out'       [    0.5062]
[ -59.0546]    [ 59.1841]    'Out'       [    0.3247]
[ -18.2813]    [ 26.5689]    'Out'       [    0.4956]
[  14.6770]    [ 52.3346]    'Out'       [    0.7807]
[ -38.5626]    [ 27.3709]    'Out'       [    0.1670]
```

Use models to predict survival (PFS). Calculate correlation and mean absolute error

Lasso

```
Ypred_lasso = Xtest * B1_coeff + B1_intercept;
r_lasso = corr(Ypred_lasso,Ytest)
```

```
r_lasso = 0.2259
```

```
avg_error_lasso = mean(abs(Ypred_lasso-Ytest))
```

```
avg_error_lasso = 281.1800
```

Stepwise

```
Ypred_step = Xtest(:,find(stats.PVAL < 0.05))*B2(find(stats.PVAL < 0.05)) + stats.inter
r_stepwise = corr(Ypred_step,Ytest)
```

```
r_stepwise = 0.4530
```

```
avg_error_stepwise = mean(abs(Ypred_step-Ytest))
```

```
avg_error_stepwise = 275.6031
```

Compare regression methods with linear regession

Build linear regression model with top 15 correlated genes

```
% Find correlation of all 100 genes and then extract data for top 15 genes
r_100 = corr(Xtrain,Ytrain);
[r_15,index_15] = maxk(abs(r_100),15);

% Create new training and test sets with top 15 genes
Xtrain_15 = Xtrain(:,index_15);
Xtest_15 = Xtest(:,index_15);

mdl = fitlm(Xtrain_15,Ytrain)
```

```
mdl =
Linear regression model:
    y ~ 1 + x1 + x2 + x3 + x4 + x5 + x6 + x7 + x8 + x9 + x10 + x11 + x12 + x13 + x14 + x15

Estimated Coefficients:
                   Estimate      SE       tStat       pValue
                   ________    _______    ________    __________

    (Intercept)      457.33     56.354      8.1153    5.9897e-09
    x1               213.75     120.73      1.7704      0.087166
    x2              -160.74     114.06     -1.4093       0.16939
    x3              -33.507     159.69    -0.20982       0.83527
    x4                174.4      195.9     0.89022       0.38068
    x5               31.944     170.59     0.18725       0.85277
    x6               180.64     128.12      1.4099        0.1692
    x7              -80.843        193    -0.41887       0.67839
    x8               215.96     213.79      1.0102       0.32077
    x9                48.88     44.087      1.1087       0.27666
    x10              8.2189     69.133     0.11888       0.90619
    x11             -41.904     167.72    -0.24985       0.80447
    x12              63.079     59.104      1.0673       0.29466
    x13             -120.49     156.75    -0.76867       0.44831
    x14               55.38     60.463     0.91593       0.36726
    x15             -94.028     173.23    -0.54279       0.59142


Number of observations: 45, Error degrees of freedom: 29
Root Mean Squared Error: 231
R-squared: 0.674,  Adjusted R-Squared: 0.505
F-statistic vs. constant model: 3.99, p-value = 0.000688
```

Predict survival (PFS) in test set and measure model accuracy

```
Ypred_norm = predict(mdl,Xtest_15);

r_norm = corr(Ytest,Ypred_norm)
```

```
r_norm = 0.0677
```

```
avg_error = mean(abs(Ypred_norm-Ytest))
```

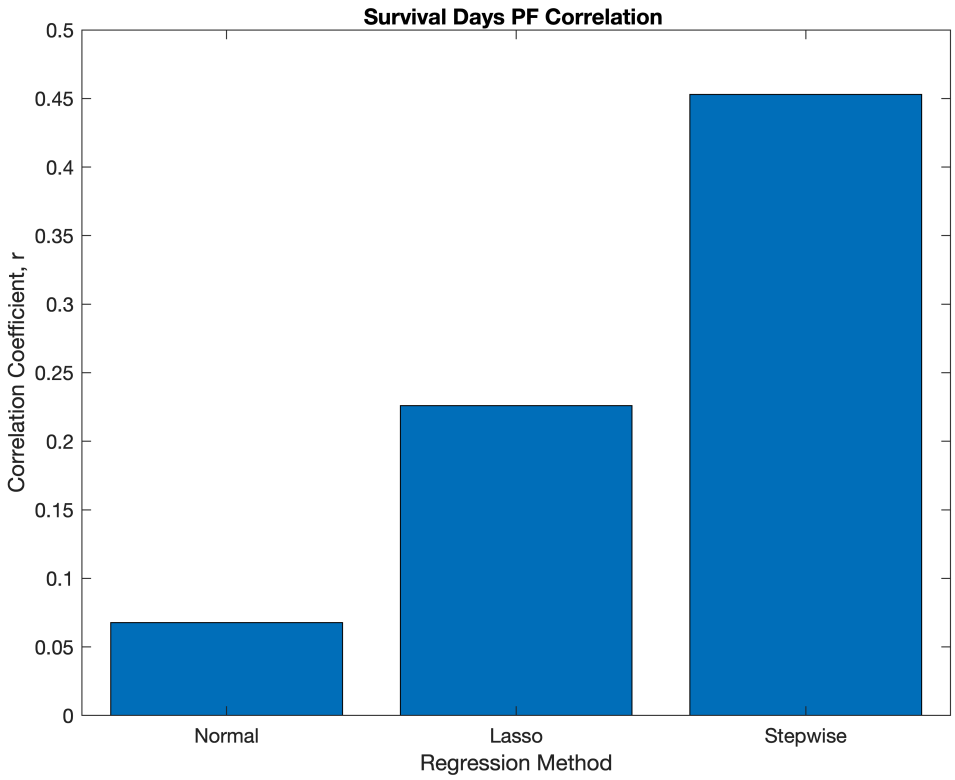
```
avg_error = 279.0284
```

Compare correlation and number of features of 3 methods

```
% Create labels for bar graphs
```

```
x = categorical({'Normal','Lasso','Stepwise'});
x = reordercats(x,{'Normal','Lasso','Stepwise'});

% Correlation bar graph.
bar(x,[r_norm,r_lasso,r_stepwise])
xlabel("Regression Method")
ylabel("Correlation Coefficient, r")
title("Survival Days PF Correlation ")
```




```
% Find number of features in each model
size_lin_reg = size(mdl.Coefficients,1)-1    % subtract 1 for intercept term
```

```
size_lin_reg = 15
```

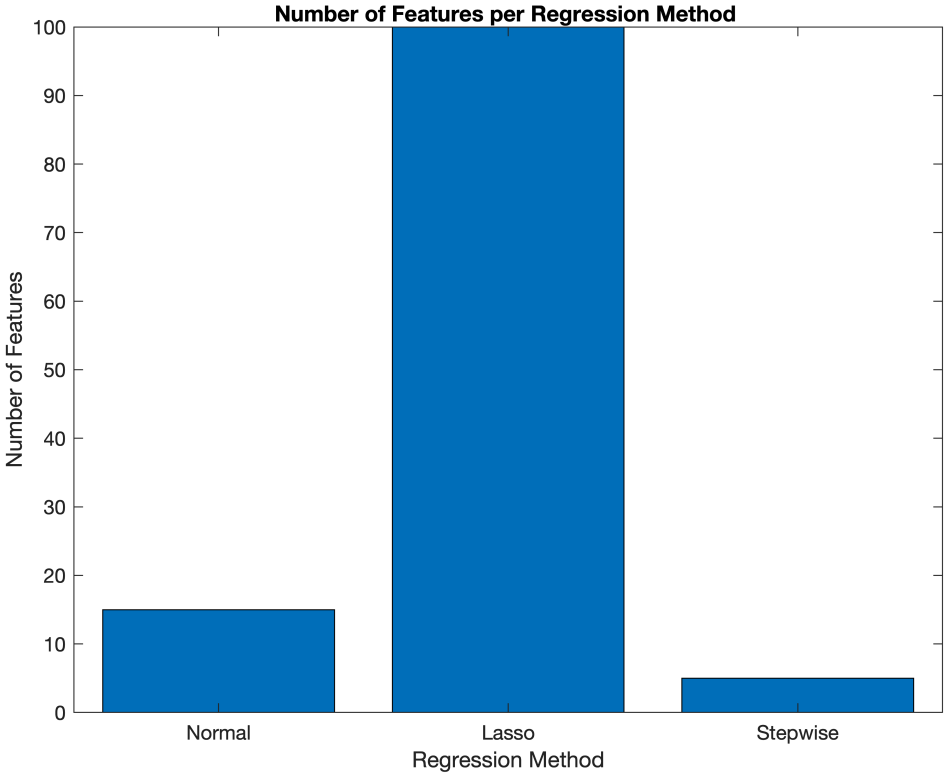
```
size_lasso = length(B1_coeff)
```

```
size_lasso = 100
```

```
size_stepwise = length(find(stats.PVAL < 0.05))
```

```
size_stepwise = 5
```

```
% Number of features bar graph
bar(x,[size_lin_reg,size_lasso,size_stepwise])
xlabel("Regression Method")
ylabel("Number of Features")
title("Number of Features per Regression Method ")
```


Number of Features per Regression Method
100
90
80
70
60
50
40
30
20
10
0
Number of Features
Normal
Lasso
Stepwise
Regression Method