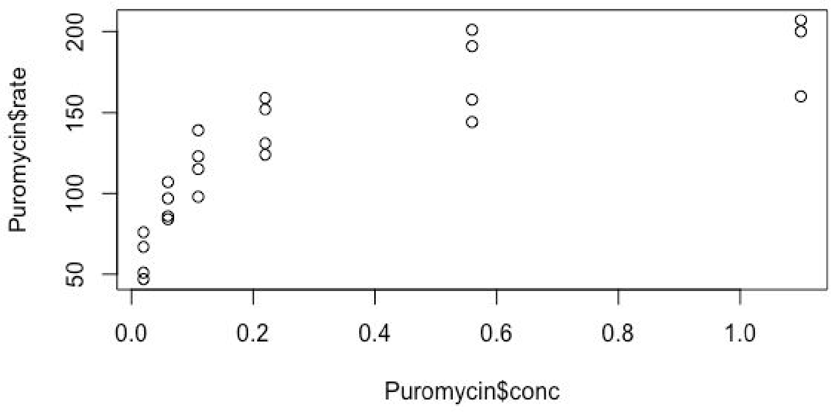
Alexander Rasho

1. Consider the built-in dataset, `Puromycin`, which shows "reaction velocity versus substrate concentration in an enzymatic reaction involving untreated cells or cells treated with Puromycin." The goal is to model reaction velocity given concentration (and whether cells are treated or untreated). Answer the following questions with the corresponding R code.

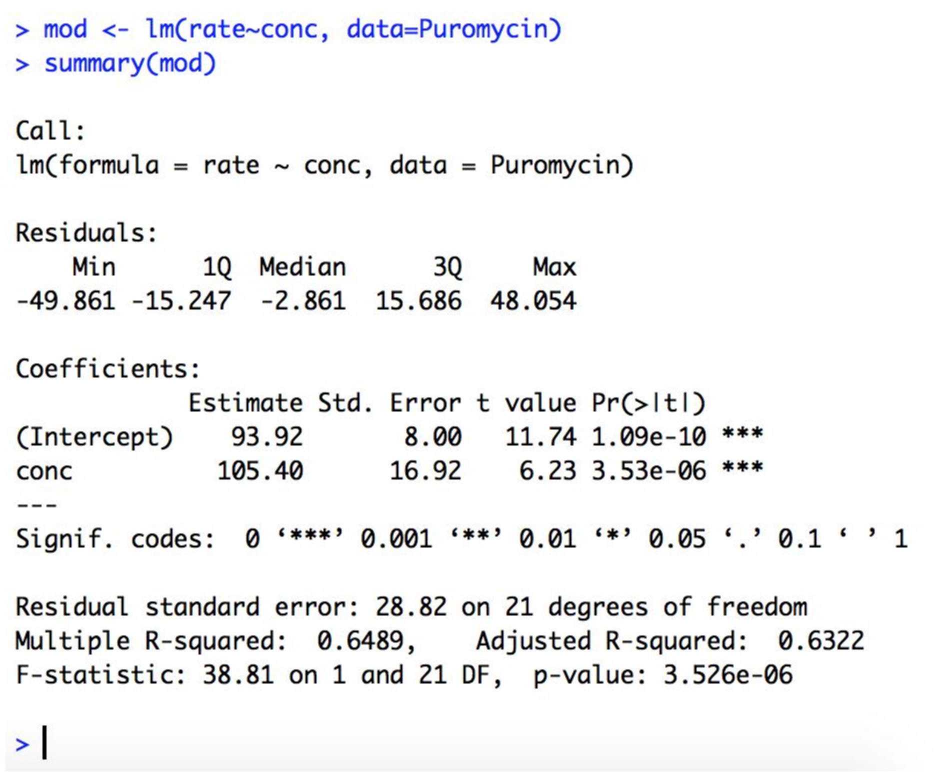
Plot `rate` vs. `conc`.

```
> plot(Puromycin$conc, Puromycin$rate)
```



Create a linear model of rate vs. conc (only one independent variable). What is the R^2?

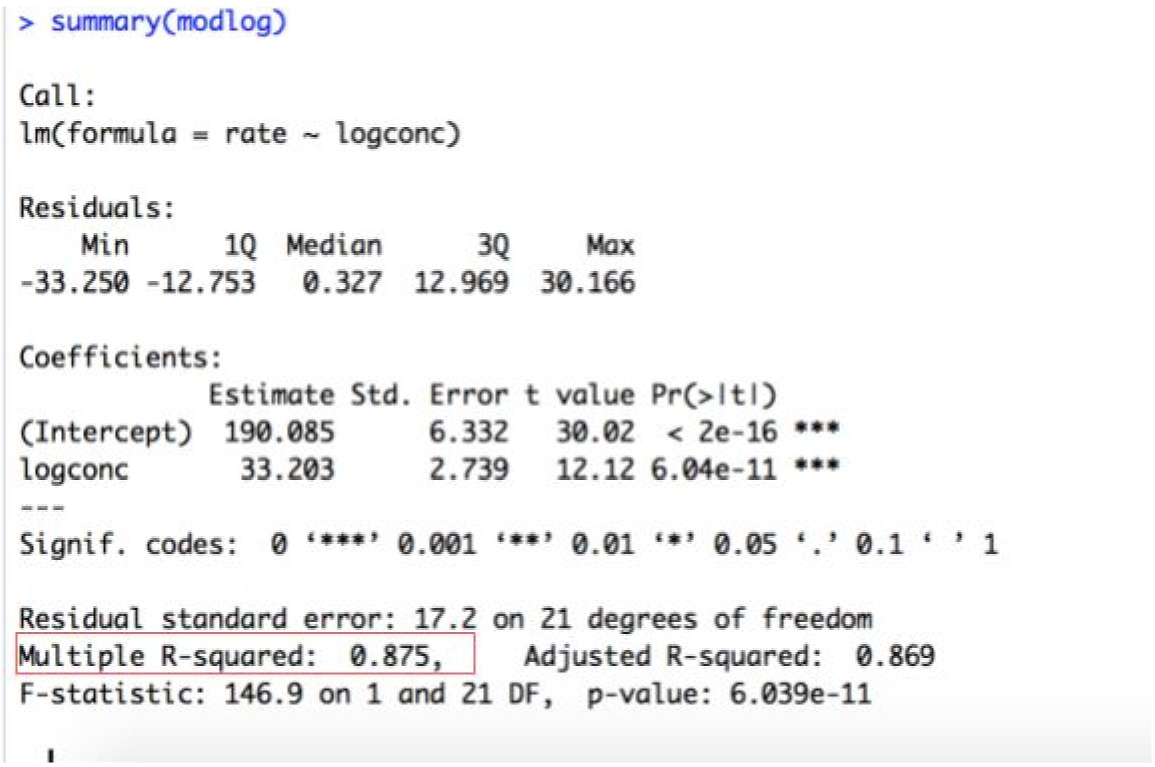
```
> mod <- lm(rate~conc, data=Puromycin)
> summary(mod)
```


```
> mod <- lm(rate~conc, data=Puromycin)
> summary(mod)

Call:
lm(formula = rate ~ conc, data = Puromycin)

Residuals:
    Min      1Q  Median      3Q     Max 
-49.861 -15.247  -2.861  15.686  48.054 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)    93.92       8.00   11.74 1.09e-10 ***
conc          105.40      16.92    6.23 3.53e-06 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 28.82 on 21 degrees of freedom
Multiple R-squared:  0.6489,	Adjusted R-squared:  0.6322 
F-statistic: 38.81 on 1 and 21 DF,  p-value: 3.526e-06

> |
```

Create a new transformed variable that is log(conc). Create a linear model of rate vs. log(conc).

i. What is the R^2?

```
> logconc <- log(Puromycin$conc)

> rate <- Puromycin$rate

> modlog <- lm(rate~logconc)

> summary(modlog)
```

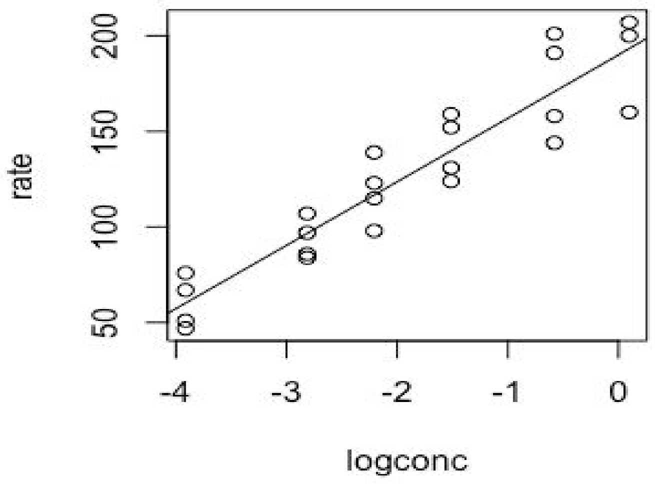


```
> summary(modlog)

Call:
lm(formula = rate ~ logconc)

Residuals:
    Min      1Q  Median      3Q     Max 
-33.250 -12.753   0.327  12.969  30.166 

Coefficients:
            Estimate Std. Error t value Pr(>|t|)    
(Intercept)  190.085      6.332   30.02  < 2e-16 ***
logconc       33.203      2.739   12.12 6.04e-11 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 17.2 on 21 degrees of freedom
Multiple R-squared:  0.875,	Adjusted R-squared:  0.869 
F-statistic: 146.9 on 1 and 21 DF,  p-value: 6.039e-11
```

ii. Is this a better fit than in part (b)?
Since this is greater than the untransformed variable, R^2 = .875 is better than R^2 = 0.6489 in B.

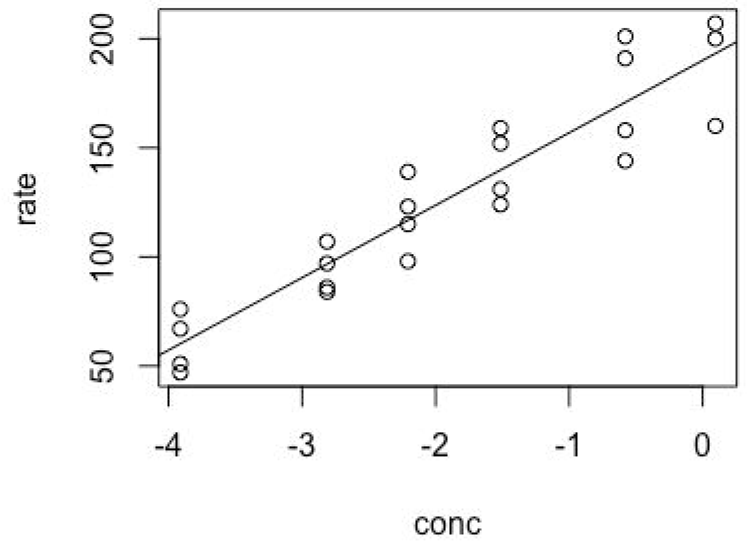
iii. Plot `rate` vs. `log(conc)` and overlay the best fit model as a straight line.

```
> plot((rate~logconc))

> abline(lm(rate~logconc))
```



iv. Plot `rate` vs. `conc` and overlay the best fit model as a curve.

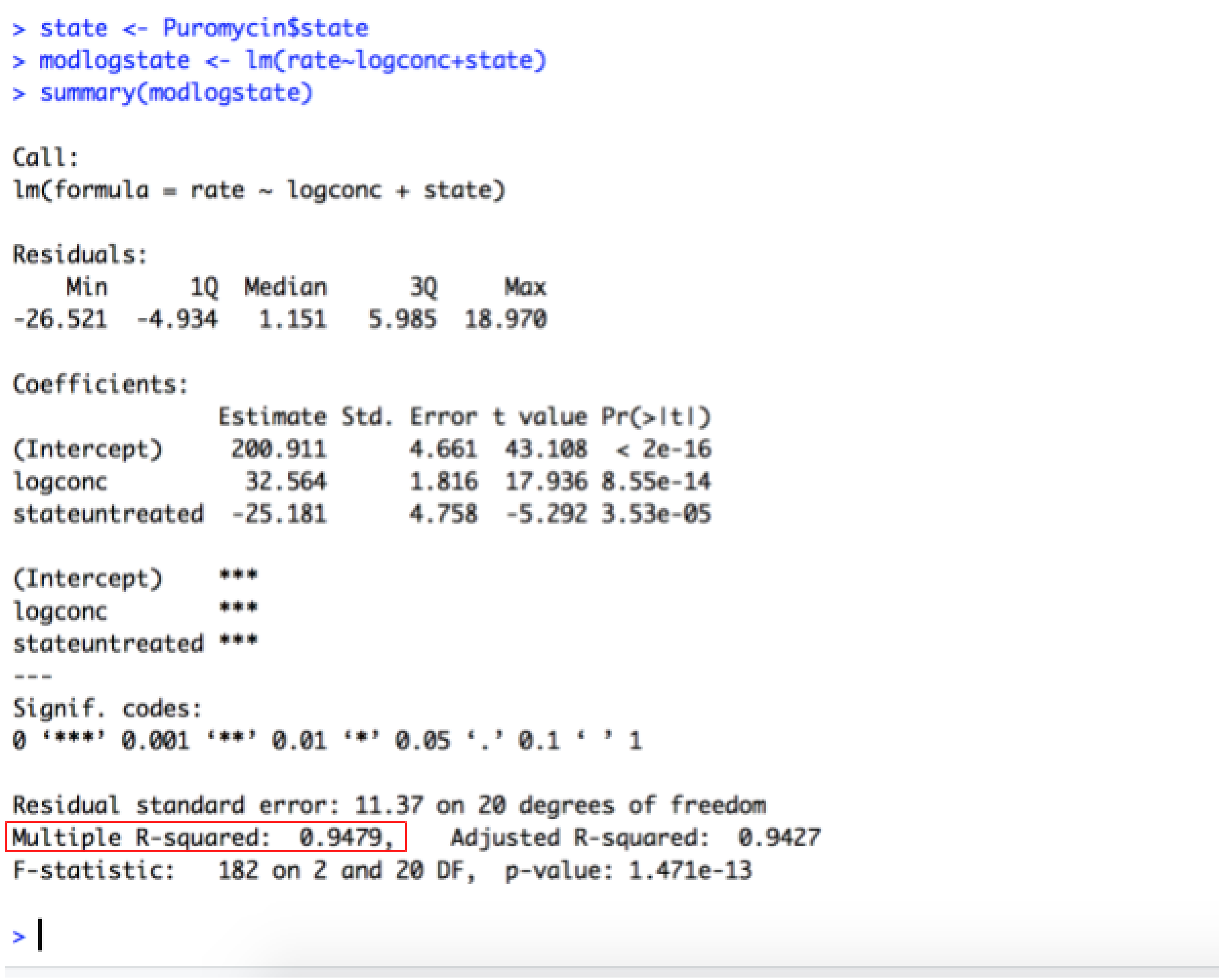
```
> conc <- log(Puromycin$conc)

> rate <- Puromycin$rate

> mod <- lm(rate~conc)

> summary(mod)

    > plot(rate~conc)

    > abline(lm(rate~conc))
```



Create a linear model of rate vs. log(conc) and state (i.e., treated/untreated). There are two independent variables.

v. What is the R^2?

```
> state <- Puromycin$state
> modlogstate <- lm(rate~logconc+state)
> summary(modlogstate)
```

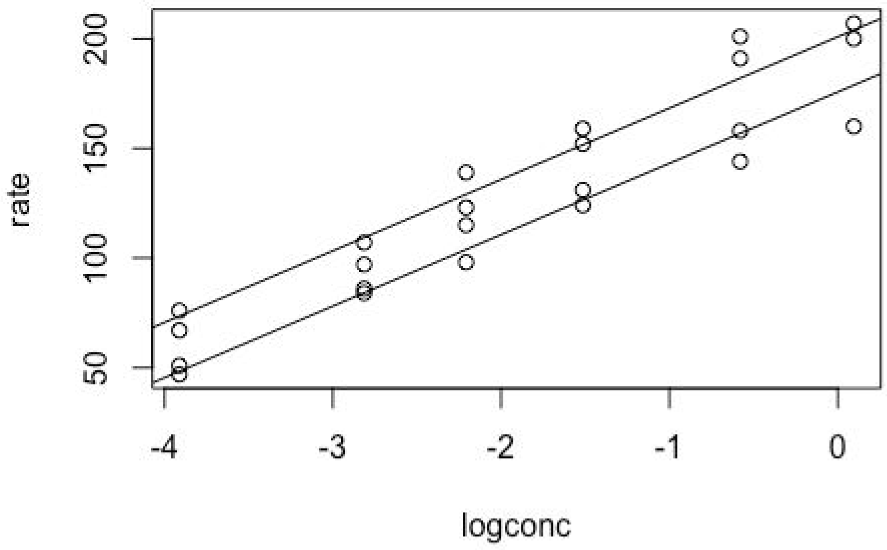


```
> state <- Puromycin$state
> modlogstate <- lm(rate~logconc+state)
> summary(modlogstate)

Call:
lm(formula = rate ~ logconc + state)

Residuals:
    Min      1Q  Median      3Q     Max 
-26.521  -4.934   1.151   5.985  18.970 

Coefficients:
               Estimate Std. Error t value Pr(>|t|)
(Intercept)     200.911      4.661  43.108  < 2e-16
logconc          32.564      1.816  17.936 8.55e-14
stateuntreated  -25.181      4.758  -5.292 3.53e-05
                  
(Intercept)    ***
logconc        ***
stateuntreated ***
---
Signif. codes:  
0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 11.37 on 20 degrees of freedom
Multiple R-squared:  0.9479,	Adjusted R-squared:  0.9427 
F-statistic:   182 on 2 and 20 DF,  p-value: 1.471e-13

> |
```

vi. Is this a better fit than in part (c)?
Since this is greater than the untransformed variable, R^2 = .9479 is better than R^2 = 0.875 in C.

vii. Plot `rate` vs. `log(conc)` and overlay the **two** linear fit lines: one for `state=treated` and the other for `state=untreated`.

```
> plot(logconc, rate)
> abline(modlogstate$coefficients[1], modlogstate$coefficients[2])
> abline(modlogstate$coefficients[1]+modlogstate$coefficients[3],
modlogstate$coefficients[2])
>
```



2. Consider the toy dataset below which shows if 4 subjects have diabetes or not, along with two diagnostic measurements.

Preg	**BP**	**HasDiabetes**	**Preg.Norm**	**BP.Norm**
2	74	No	.5	1
3	58	Yes	1	.2
2	58	Yes	.5	.2
1	54	No	0	0
2	70	?	.5	.8

a. Which variable is the “Class” variable?
   HasDiabetes

b. Normalize the Preg and BP values by scaling the minimum-maximum range of each column to 0-1. Fill in the empty columns in the table.
   Formula: x-min(x)/max(x)-min(x)

c. Predict whether a subject with Preg=2, BP=70 will have diabetes using the 1-NN algrotithm and

   i. Using Euclidean distance on the original variables:
      Closest no = sqrt((2-2)^2(74-70)^2) = 4
      Closest yes = sqrt((2-2)^2(74-58)^2) = 16
      4<16

      Predicted not to have diabetes.

   ii. Using Manhattan distance on the original variables:
      Closest no = |2-2| -|74-70| = 4
      Closest yes= |2-2| - |74-58| = 16
      4<16

      Predicted not to have diabetes.

   iii. Using Euclidean distance on the normalized variables:

      Closest no = sqrt((.5-.5)^2 + sqrt(1-.8)^2) = .2
      Closest yes = sqrt(.5-.5)^2 + (.8-.2)^2) = .6
      .2<.6

      Predicted not to have diabetes.

   iv. Using Manhattan distance on the normalized variables:

      Closest no = |.5-.5| + |1-.8| = .2
      Closest yes = |.5-.5| + |.8-.2| = .6
      .2<.6

      Predicted not to have diabetes.

3. The `pima-indians-diabetes-resampled.csv` file attached on Titanium contains records indicating whether the subjects have diabetes or not, along with certain diagnostic measurements. All subjects are of Pima Indian heritage and this dataset is called the Pima Indian Diabetes Database[1]. The goal is to see if it is possible to predict if a subject has diabetes given some of the diagnostic measurements.

a. Load and pre-process the data. Write code to:

   i. Load the data file on Titanium. How many rows and columns are there?
      > mydata <- read.csv("pima_indians_diabetes_resampled")
      > nrow(pima_indians_diabetes_resampled)

[1] https://github.com/jbrownlee/Datasets/blob/master/pima-indians-diabetes.names

[1] 768 rows
> ncol(pima_indians_diabetes_resampled)
[1] 9 columns

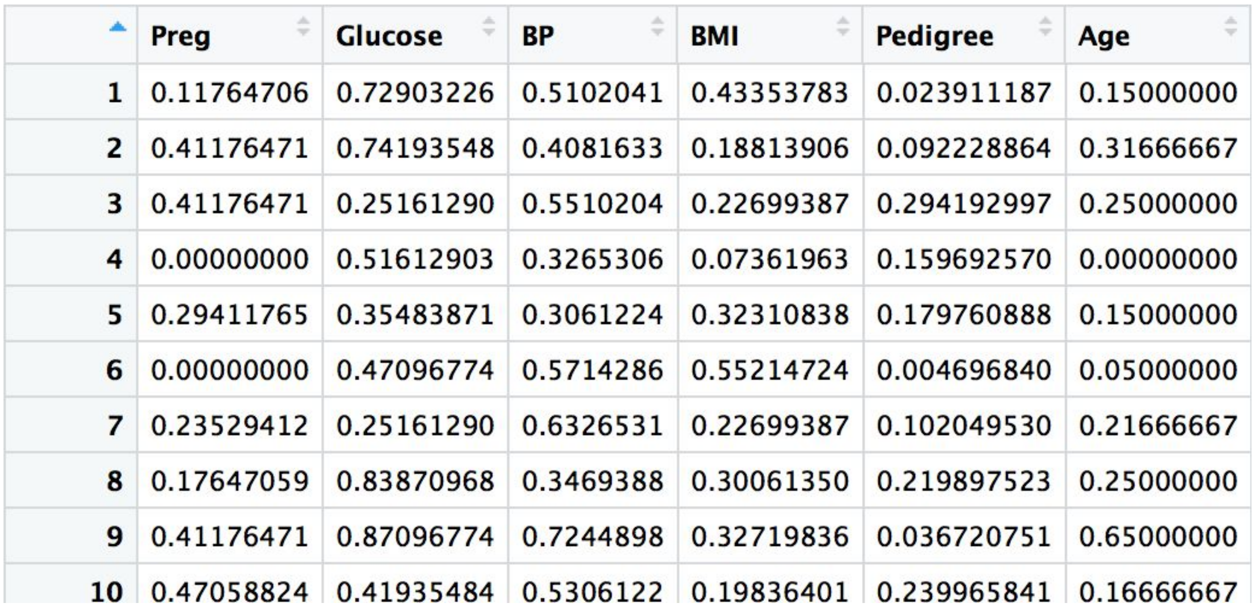
ii. What does Skin stand for in the dataset? What is its unit? (Search for the Pima Indian Diabetes Database online and read up on its background.)

    It's the thickness of the skin of a tricep fold in millimeters.
    From: https://www.kaggle.com/rishpande/pima-indians-diabetes-beginner

iii. The data contains missing values. These are indicated by 0 values in the Glucose, BP, Skin, Insulin, BMI, Pedigree, and Age columns.
    1. Remove the Skin and Insulin columns (they contain a large number of missing values)
       >pima_indians_diabetes_resampled$Skin <- NULL
       >pima_indians_diabetes_resampled$Insulin <- NULL
    2. Remove rows which contain missing values in the Glucose, BP, BMI, Pedigree, or Age columns
       > mydata[2:6][mydata[,2:6] == 0] <- NA
       > mydata2 <- drop_na(mydata, 2:6)
       > view(mydata2)
    3. Check that you have 724 rows remaining.
       > nrow(mydata2)
       [1] 724

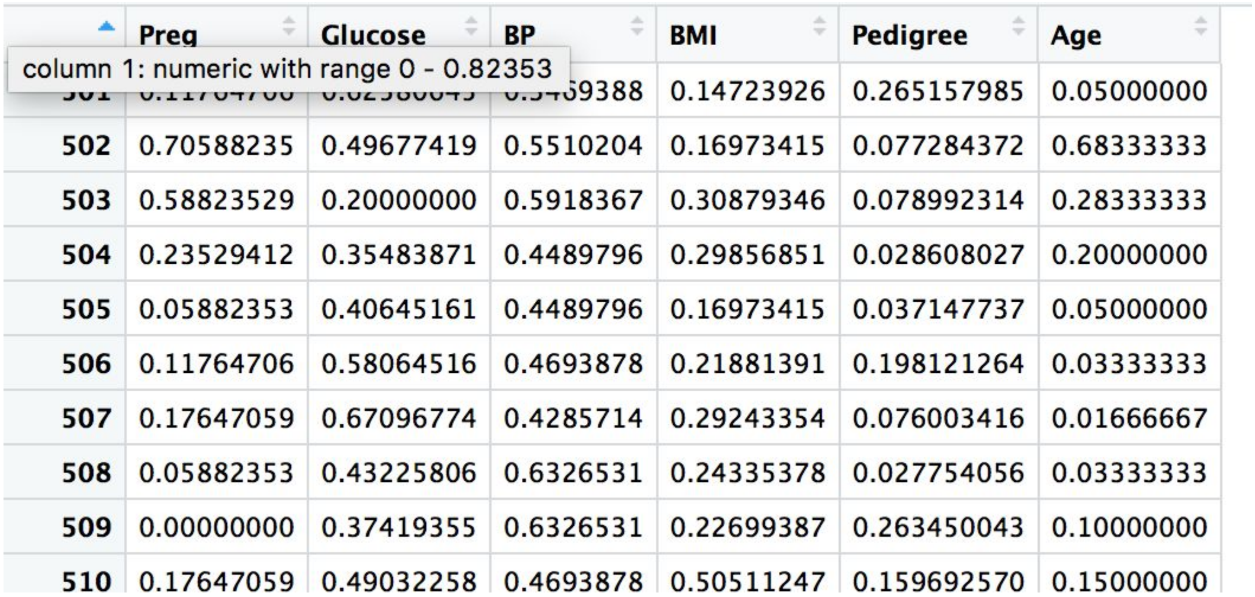
b. Normalize each column by scaling the `scale()` range of each column to 0-1. (Hint: the built-in R function `scale()` can be used for this)
   > normalize <- function(x) { return ((x - min(x)) / (max(x) - min(x)))}
   > normalized <- as.data.frame(lapply(mydata, normalize))
   > view(normalized)
c. Split the dataset into train and test datasets with the *first 500 rows* for training, and the remaining rows for test. Do NOT randomly sample the data (though resampling is usually done, this hw problem does not use this step for ease of grading).

   training <- normalized[1:500, 1:6]



	Preg	Glucose	BP	BMI	Pedigree	Age
1	0.11764706	0.72903226	0.5102041	0.43353783	0.023911187	0.15000000
2	0.41176471	0.74193548	0.4081633	0.18813906	0.092228864	0.31666667
3	0.41176471	0.25161290	0.5510204	0.22699387	0.294192997	0.25000000
4	0.00000000	0.51612903	0.3265306	0.07361963	0.159692570	0.00000000
5	0.29411765	0.35483871	0.3061224	0.32310838	0.179760888	0.15000000
6	0.00000000	0.47096774	0.5714286	0.55214724	0.004696840	0.05000000
7	0.23529412	0.25161290	0.6326531	0.22699387	0.102049530	0.21666667
8	0.17647059	0.83870968	0.3469388	0.30061350	0.219897523	0.25000000
9	0.41176471	0.87096774	0.7244898	0.32719836	0.036720751	0.65000000
10	0.47058824	0.41935484	0.5306122	0.19836401	0.239965841	0.16666667

test <- normalized[501:724, 1:6]



column 1: numeric with range 0 - 0.82353

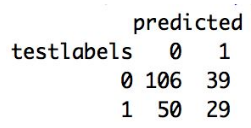
	Preg	Glucose	BP	BMI	Pedigree	Age
501	0.11764706	0.62580645	0.5469388	0.14723926	0.265157985	0.05000000
502	0.70588235	0.49677419	0.5510204	0.16973415	0.077284372	0.68333333
503	0.58823529	0.20000000	0.5918367	0.30879346	0.078992314	0.28333333
504	0.23529412	0.35483871	0.4489796	0.29856851	0.028608027	0.20000000
505	0.05882353	0.40645161	0.4489796	0.16973415	0.037147737	0.05000000
506	0.11764706	0.58064516	0.4693878	0.21881391	0.198121264	0.03333333
507	0.17647059	0.67096774	0.4285714	0.29243354	0.076003416	0.01666667
508	0.05882353	0.43225806	0.6326531	0.24335378	0.027754056	0.03333333
509	0.00000000	0.37419355	0.6326531	0.22699387	0.263450043	0.10000000
510	0.17647059	0.49032258	0.4693878	0.50511247	0.159692570	0.15000000

d. Train and test a k-nearest neighbor classifier with the above datasets. *Consider only Pred and Pedigree columns*. Set k=1. What is the error rate (number of misclassifications)?

e. library(class)

   > training <- normalized[1:500, c(1,5)]

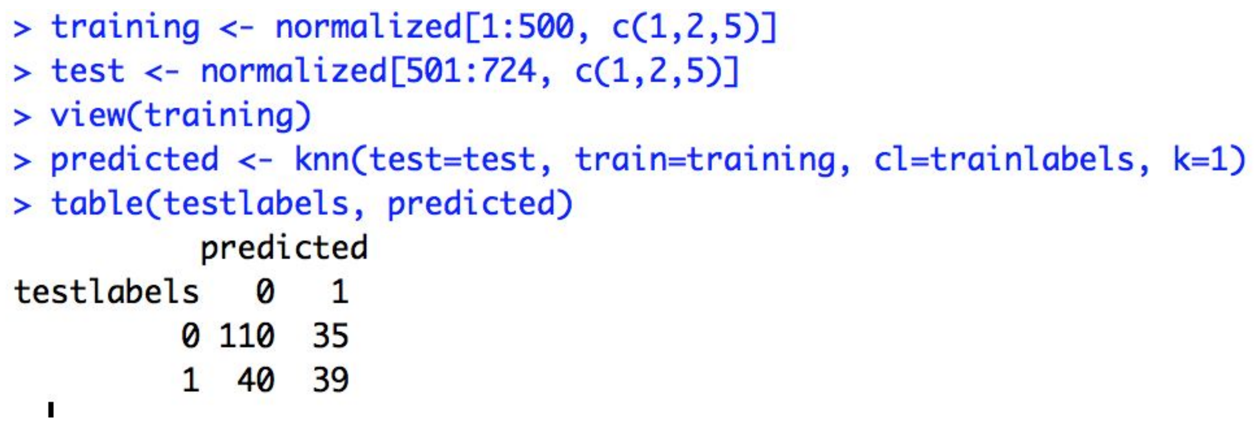
```
> test <- normalized[501:724, c(1,5)]
> trainlabels <- normalized[1:500, c(7)]
> testlabels <- normalized[501:724, c(7)]
> predicted <- knn(test=test, train=training, cl=trainlabels, k=1)
> table(testlabels, predicted)
```



```
          predicted
testlabels   0   1
         0 106  39
         1  50  29
```

Error rate = 39 + 50 = 89/224 = .397

f. Repeat part (d) but *consider only Pred, Pedigree, and Glucose columns*. Set k=1. What is the error rate? Will the error rate always decrease with larger number of parameters? Why or why not: answer in 2-3 sentences?

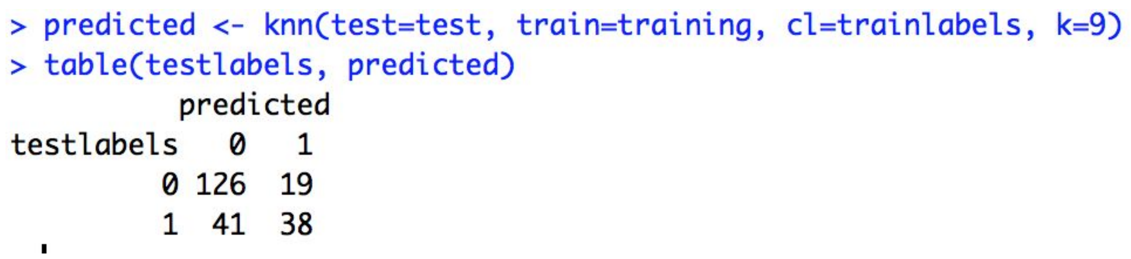


```
> training <- normalized[1:500, c(1,2,5)]
> test <- normalized[501:724, c(1,2,5)]
> view(training)
> predicted <- knn(test=test, train=training, cl=trainlabels, k=1)
> table(testlabels, predicted)
          predicted
testlabels   0   1
         0 110  35
         1  40  39
```

Error rate = 35 + 40 = 75/224 = .335

Yes, because more information is given. When more information is given the prediction will become more accurate.

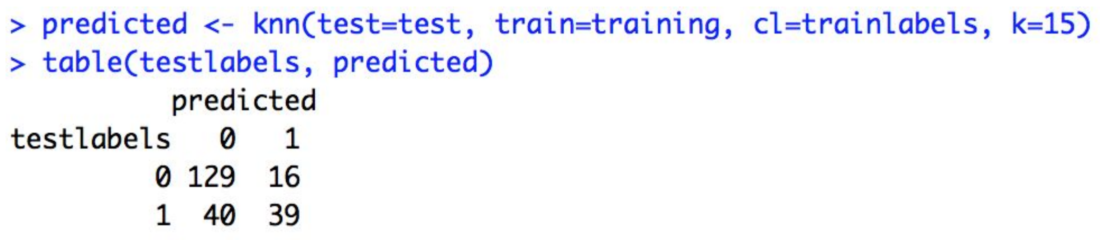
g. Repeat part (e) but set k=9. What is the error rate?



```
> predicted <- knn(test=test, train=training, cl=trainlabels, k=9)
> table(testlabels, predicted)
          predicted
testlabels   0   1
         0 126  19
         1  41  38
```

Error rate = 19+41 = 60/224 = .268

h. Repeat part (e) but set k=15. What is the error rate? Considering your observations from (e)-(g), which is the best value for k?



```
> predicted <- knn(test=test, train=training, cl=trainlabels, k=15)
> table(testlabels, predicted)
          predicted
testlabels   0   1
         0 129  16
         1  40  39
```

Error rate = 16+40 = 56/228 = .246

The best value for k is 15.