Logistic Regression

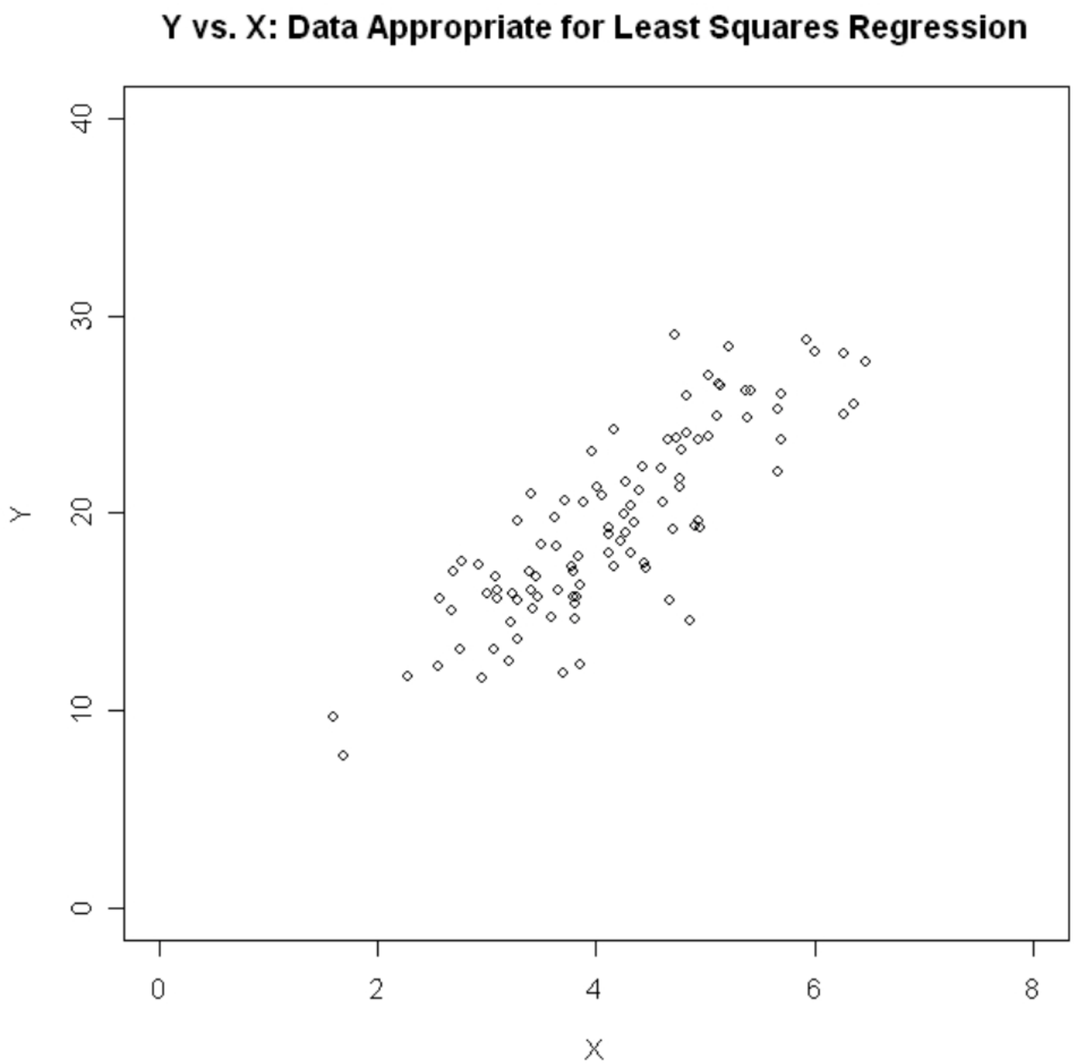
Classification Models

Logistic Regression is one type of Classification Models

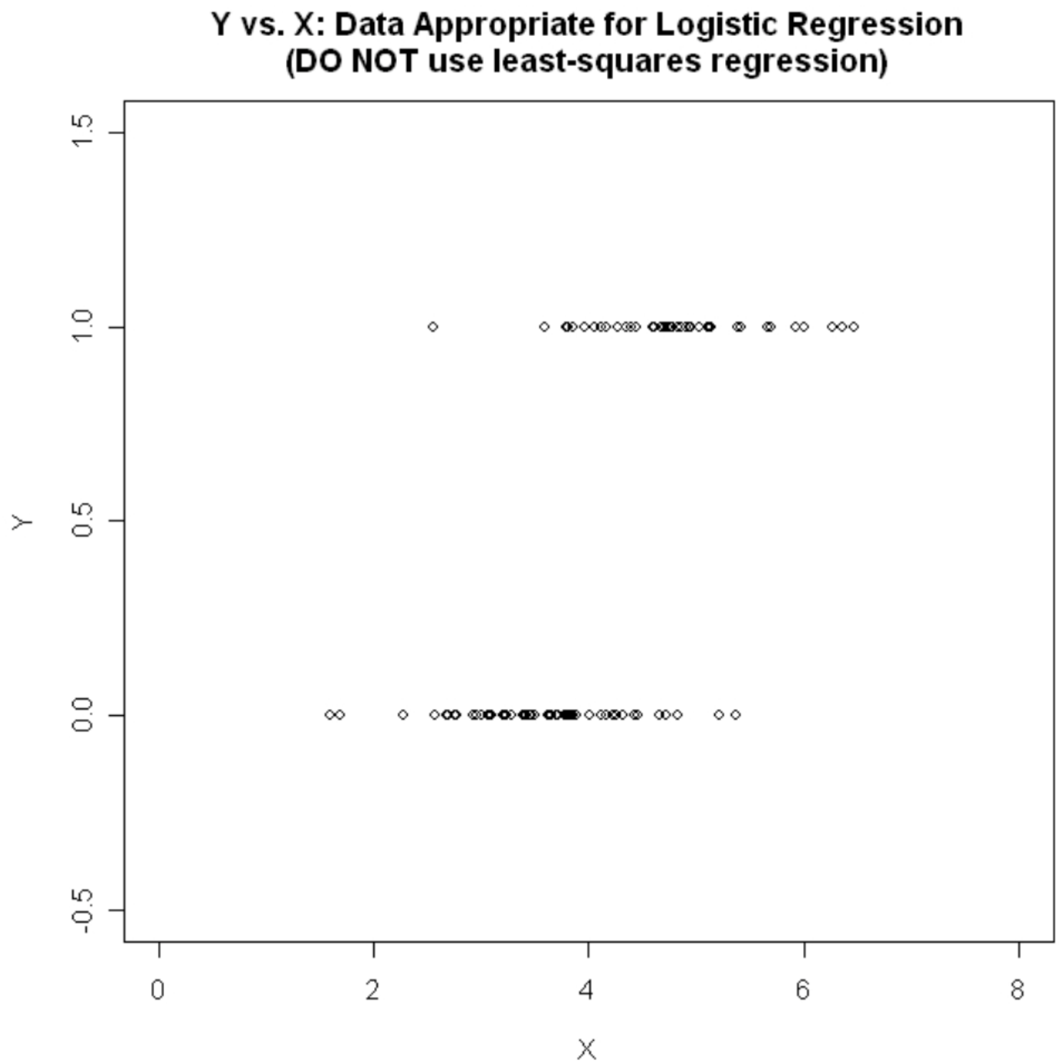
https://www.youtube.com/watch?v=zAULhNrnuL4

Why Regular Regression Does NOT Work

Recall that in least-squares regression, we model the Y-variable as a linear function of the X-variables plus a random error that is assumed to have a normal distribution. Then, if you use regular least-squares regression when you have a binary dependent variable (and should be using logistic regression) you are violating the least-squares requirement that the regression errors have a normal distribution.
When the assumptions that underlie the least-squares regression model are violated, you can no longer rely on the statistical inference (e.g., which regression coefficients are significant) or predictions that are made based on the least-squares model.
The first figure shows the kind of data that is appropriate for regular least-squares regression:



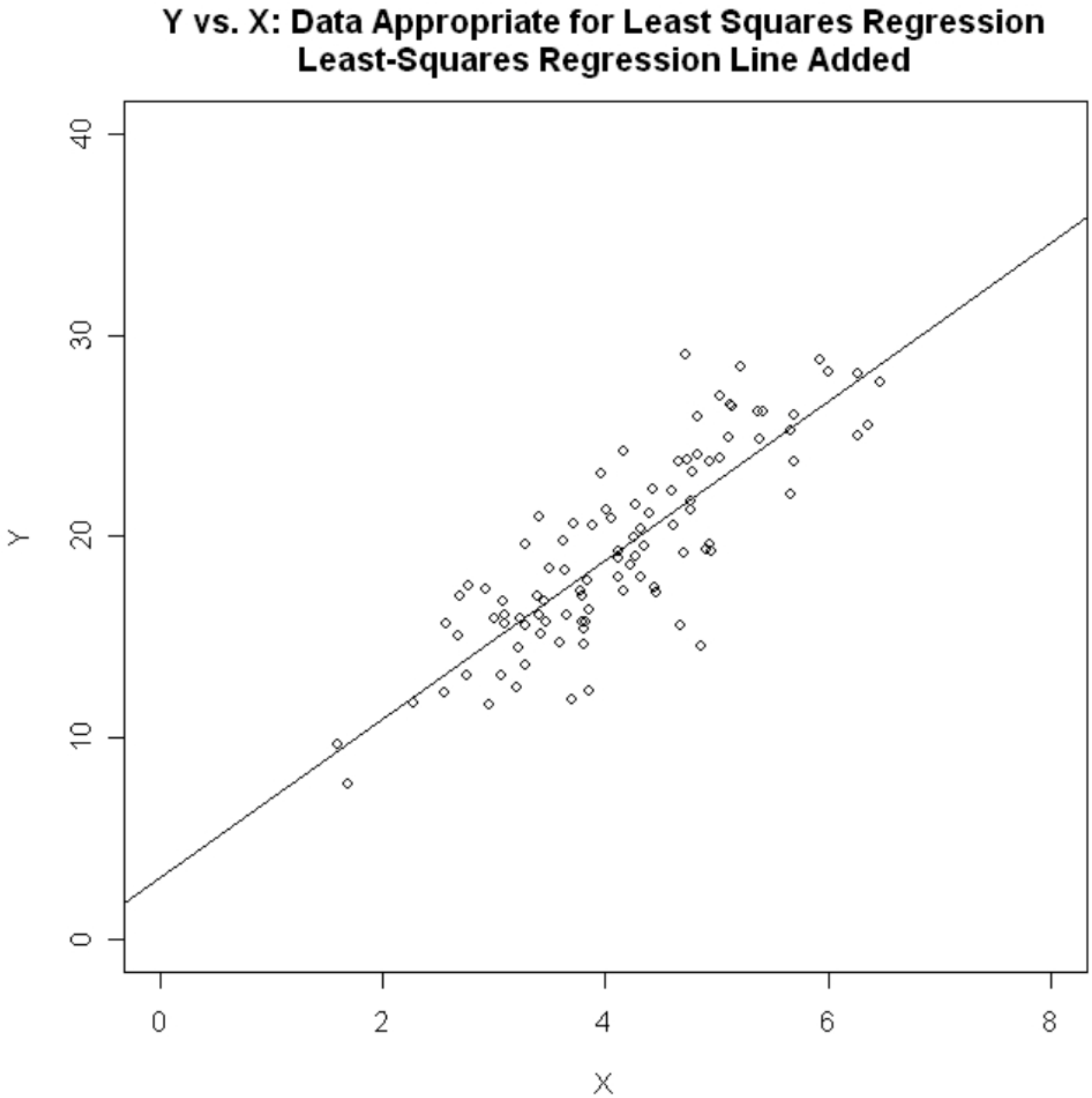
In this figure, you can see that the Y-variable takes on continuous values.The figure below shows data that is appropriate for logistic regression:



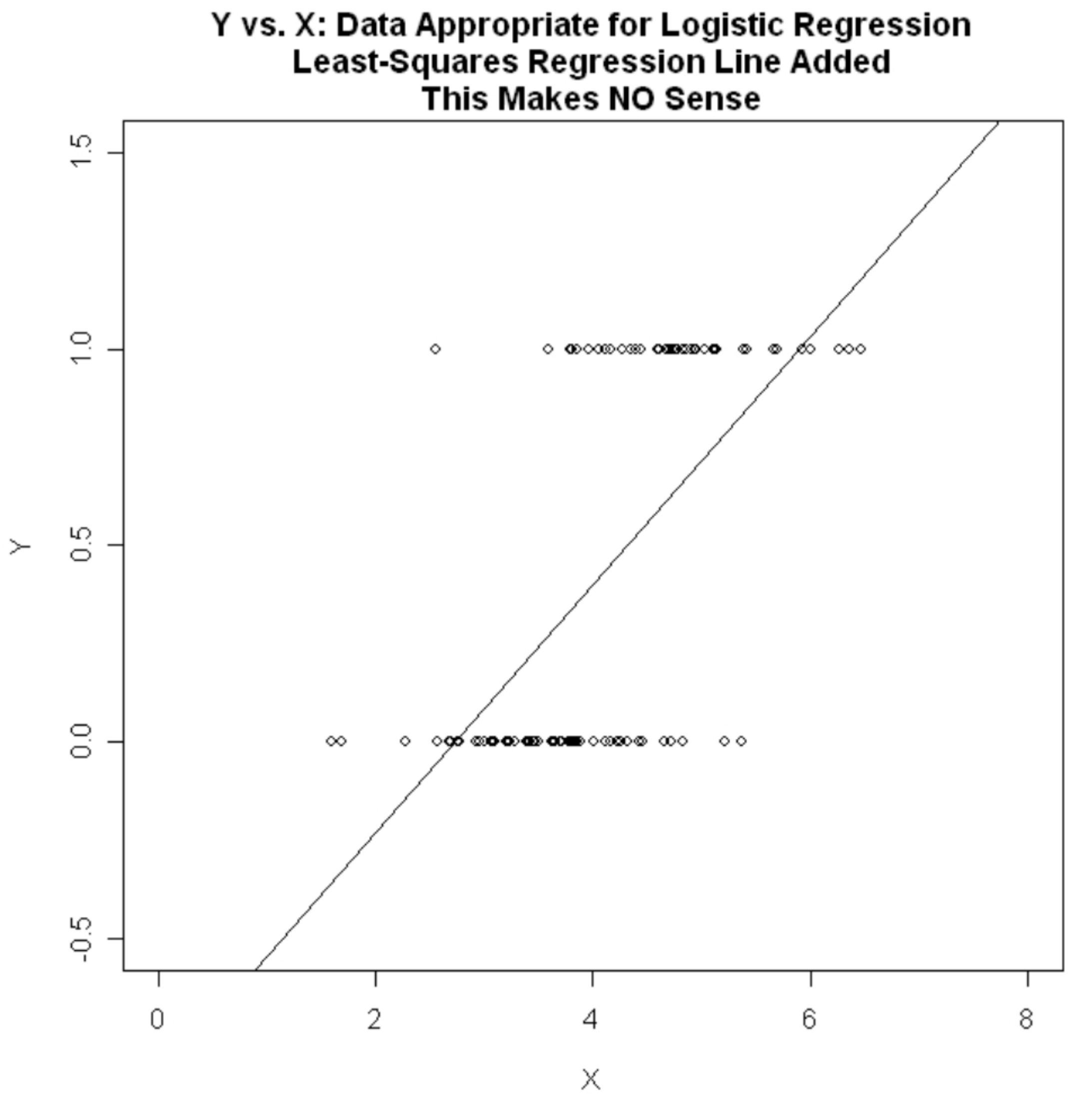
In this figure, you can see that the Y-variable only takes on two values, 0 and 1. This means that the data appear to be on two horizontal parallel lines, one at 0 and the other at 1. If you look carefully, you can see that the probability that Y is 1 increases as the value of X increases.

You can also see by looking at this picture that the equation above for the least-squares regression must give silly predictions for Y when Y takes on only binary values. The equation is linear. For any regression coefficient that is positive, increasing the corresponding X value will cause the prediction for Y to increase. You can make the predicted value of Y as large as you want just by moving the X value far enough. Thus, there will be X values for which the predicted Y value will far exceed 1. Similarly, there will be other X values for which the predicted Y value will negative and far below 0. Such predictions make no sense when the only values that the Y-variable can take on are 0 and 1.

To show this, I have added the least squares regression line to the two figures shown above. Here is the first one which was for data that is appropriate for least-squares regression.

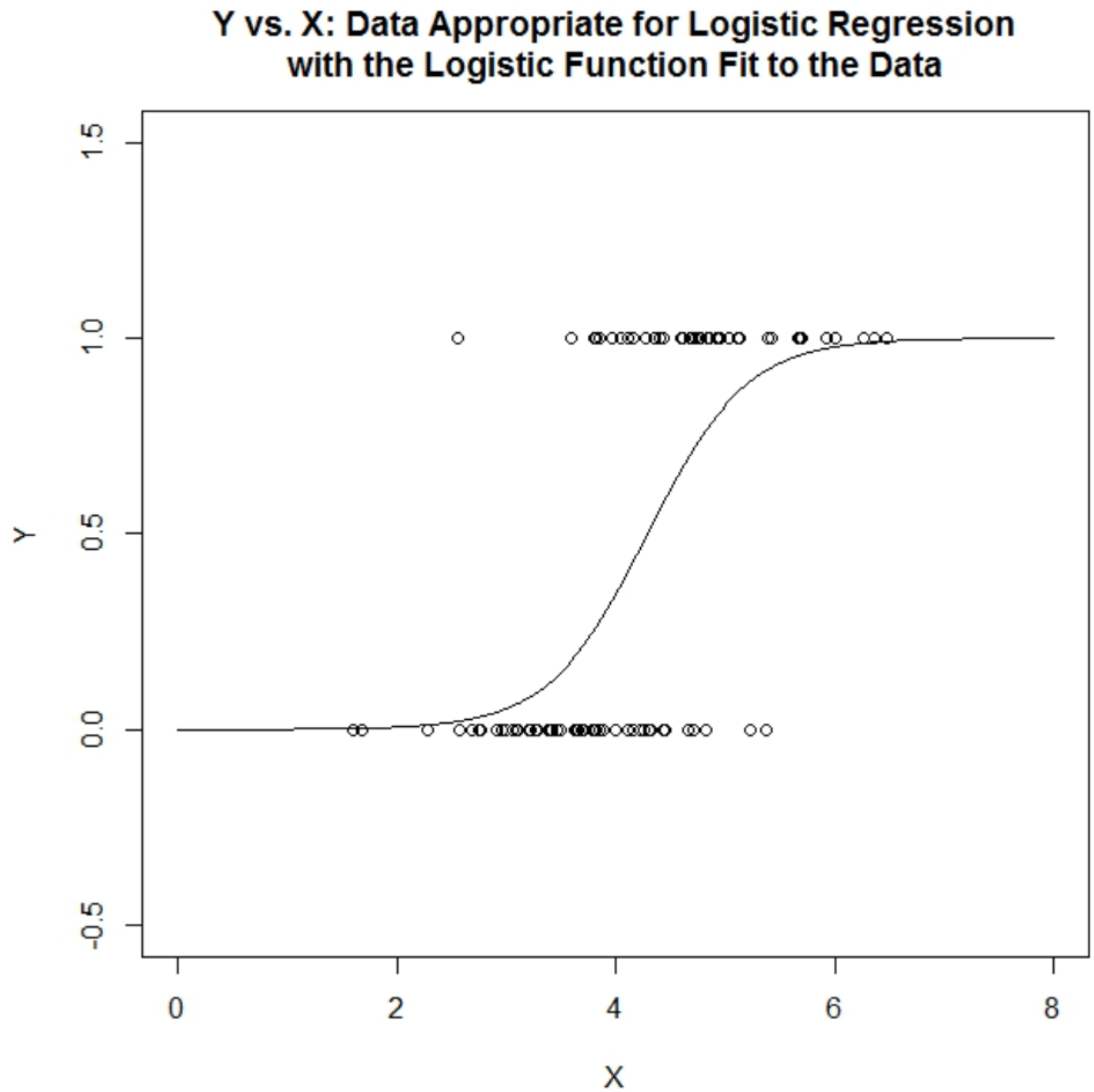


For this first figure, the regression line makes perfect sense and gives very reasonable predictions. Next, the figure for the data with the binary Y-variable is shown.

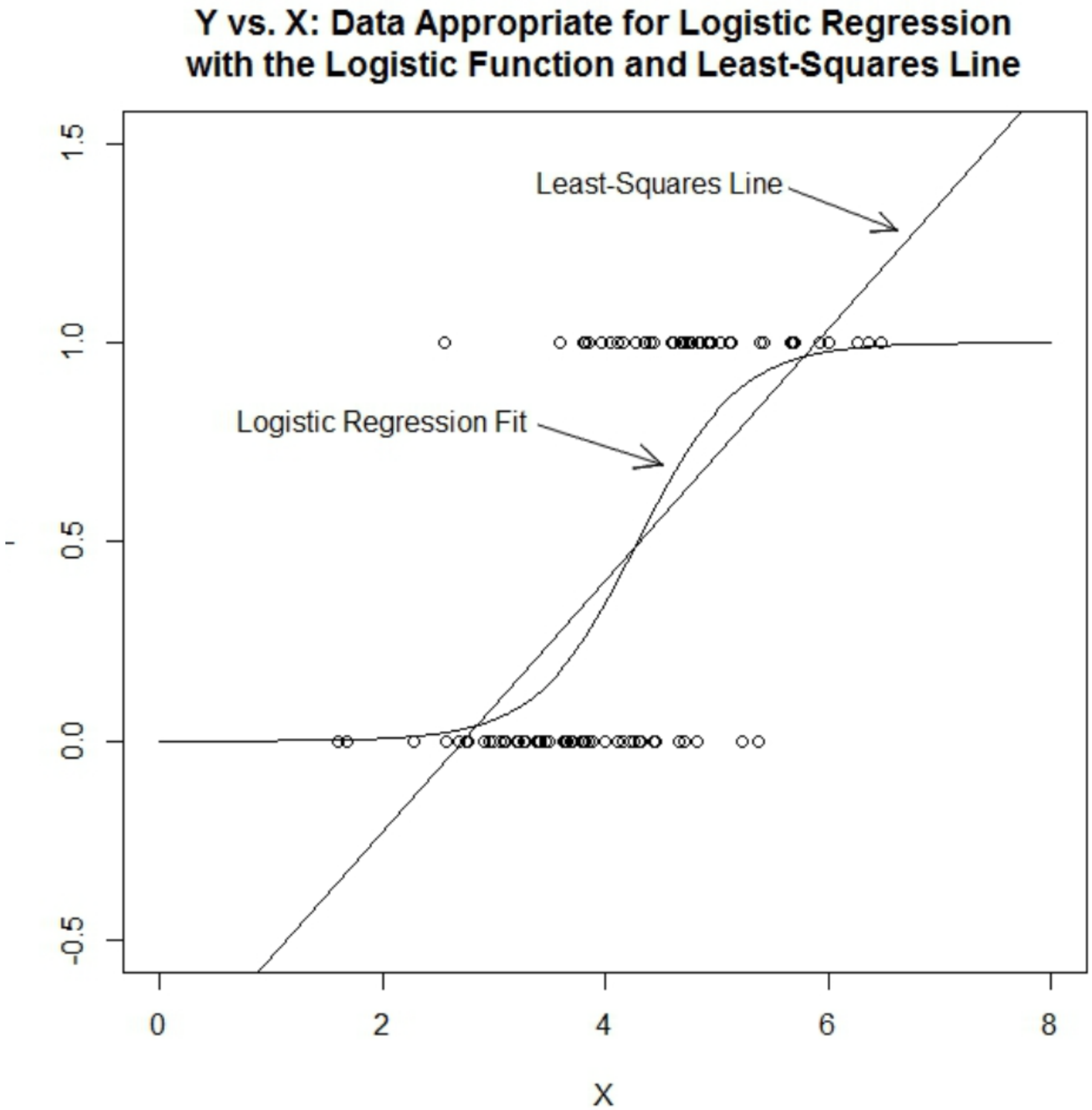


As you can see, the least-squares regression line gives predictions that make no sense. For example, for an X value of 8, the least-square regression line predicts that Y will be above 1.5. But Y can only take on values of 0 and 1.

So now I will show you in a graph what the logistic regression equation is doing.



In this figure, the smooth s-shaped trace shows the logistic function that is fit to the binary data. This function is an estimate of the probability that Y is one. As you can see, the probability that Y is 1 is very small on the left hand side of the figure. It increases through the middle of the figure and is nearly 1 on the right hand side of the figure. Just to contrast the logistic regression fit with the regular least-squares regression line, I will now add the least-squares line to the figure.



This figure clearly shows how silly the least-squares line is for this binary data and how well the logistic curve estimates the probability that the dependent Y variable is 1.

Odds Ratio

Odds are the number of times success occurred compared to the number of times failure occurred.

Probability(success) = number of successes/total number of trials
Odds(success) = number of successes/number of failures = Pr(success)/Pr(failure)

Log Odd

$log(Odds(success)) = log(Pr(success)/Pr(failure))$

False Positive

		Reality	
		True	**False**
Measured / Perceived	**True**	Correct	Type I False Positive
	False	Type II False Negative	Correct

		Disease	
		Present	**Absent**
Test	**Positive**	TP	FP
	Negative	FN	TN

TP = (Sensitivity)(Pretest probability)
FP = (1−Specificity)(1−Pretest probability)
FN = (1−Sensitivity)(Pretest probability)
TN = (Specificity)(1−Pretest probability)

$$\textit{Sensitivity} = \frac{\textit{Number of diseased patients with positive test}}{\textit{Number of diseased patients}} = \frac{TP}{TP + FN}$$

$$\textit{Specificity} = \frac{\textit{Number of nondiseased patients with negative test}}{\textit{Number of nondiseased patients}} = \frac{TN}{TN + FP}$$

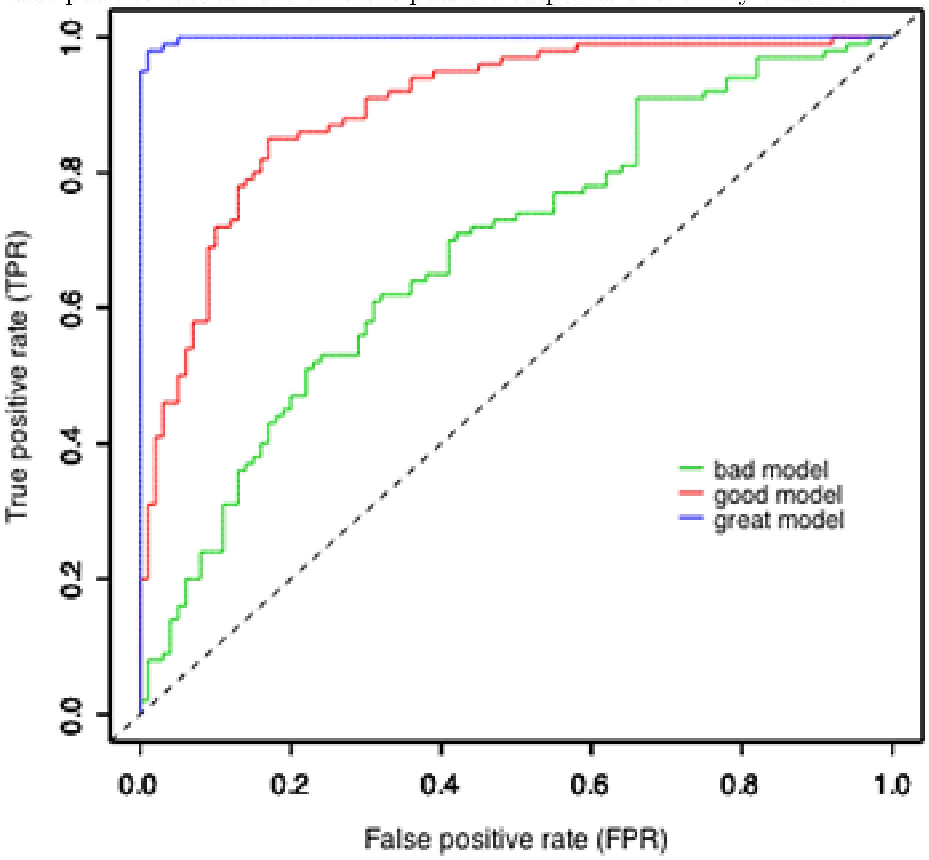
$$\textit{Posttest probability after positive test} = \textit{Probability of disease if test positive} = \frac{TP}{TP + FP}$$

$$= \frac{(\text{Sensitivity})(\text{Pretest probability})}{(\text{Sensitivity})(\text{Pretest probability}) + (1-\text{Specificity})(1-\text{Pretest probability})}$$

Source: McPhee SJ, Papadakis MA: *Current Medical Diagnosis and Treatment 2011*, 50th Edition: http://www.accessmedicine.com

ROC Curve

Receiver Operating Characteristic curve (or ROC curve). It is a plot of the true positive rate against the false positive rate for the different possible cutpoints of a binary classifier.




```r
binary <- read.csv("https://stats.idre.ucla.edu/stat/data/binary.csv")
head(binary)
```

```
##   admit gre  gpa rank
## 1     0 380 3.61    3
## 2     1 660 3.67    3
## 3     1 800 4.00    1
## 4     1 640 3.19    4
## 5     0 520 2.93    4
## 6     1 760 3.00    2
```

```r
str(binary)
```

```
## 'data.frame':    400 obs. of  4 variables:
##  $ admit: int  0 1 1 1 0 1 1 0 1 0 ...
##  $ gre  : int  380 660 800 640 520 760 560 400 540 700 ...
##  $ gpa  : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...
##  $ rank : int  3 3 1 4 4 2 1 2 3 2 ...
```

```r
summary(binary)
```

```
##      admit             gre             gpa             rank      
##  Min.   :0.0000   Min.   :220.0   Min.   :2.260   Min.   :1.000  
##  1st Qu.:0.0000   1st Qu.:520.0   1st Qu.:3.130   1st Qu.:2.000  
```

```
##  Median :0.0000   Median :580.0   Median :3.395   Median :2.000
##  Mean   :0.3175   Mean   :587.7   Mean   :3.390   Mean   :2.485
##  3rd Qu.:1.0000   3rd Qu.:660.0   3rd Qu.:3.670   3rd Qu.:3.000
##  Max.   :1.0000   Max.   :800.0   Max.   :4.000   Max.   :4.000
```

```r
## convert rank to a factor
binary$rank = factor(binary$rank)
str(binary)
```

```
## 'data.frame':    400 obs. of  4 variables:
##  $ admit: int  0 1 1 1 0 1 1 0 1 0 ...
##  $ gre  : int  380 660 800 640 520 760 560 400 540 700 ...
##  $ gpa  : num  3.61 3.67 4 3.19 2.93 3 2.98 3.08 3.39 3.92 ...
##  $ rank : Factor w/ 4 levels "1","2","3","4": 3 3 1 4 4 2 1 2 3 2 ...
```

```r
###########################################
####### Explore and Visualize Data  #######
###########################################
require(dplyr)
```

```
## Loading required package: dplyr
```

```
##
## Attaching package: 'dplyr'
```

```
## The following objects are masked from 'package:stats':
##
##     filter, lag
```

```
## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

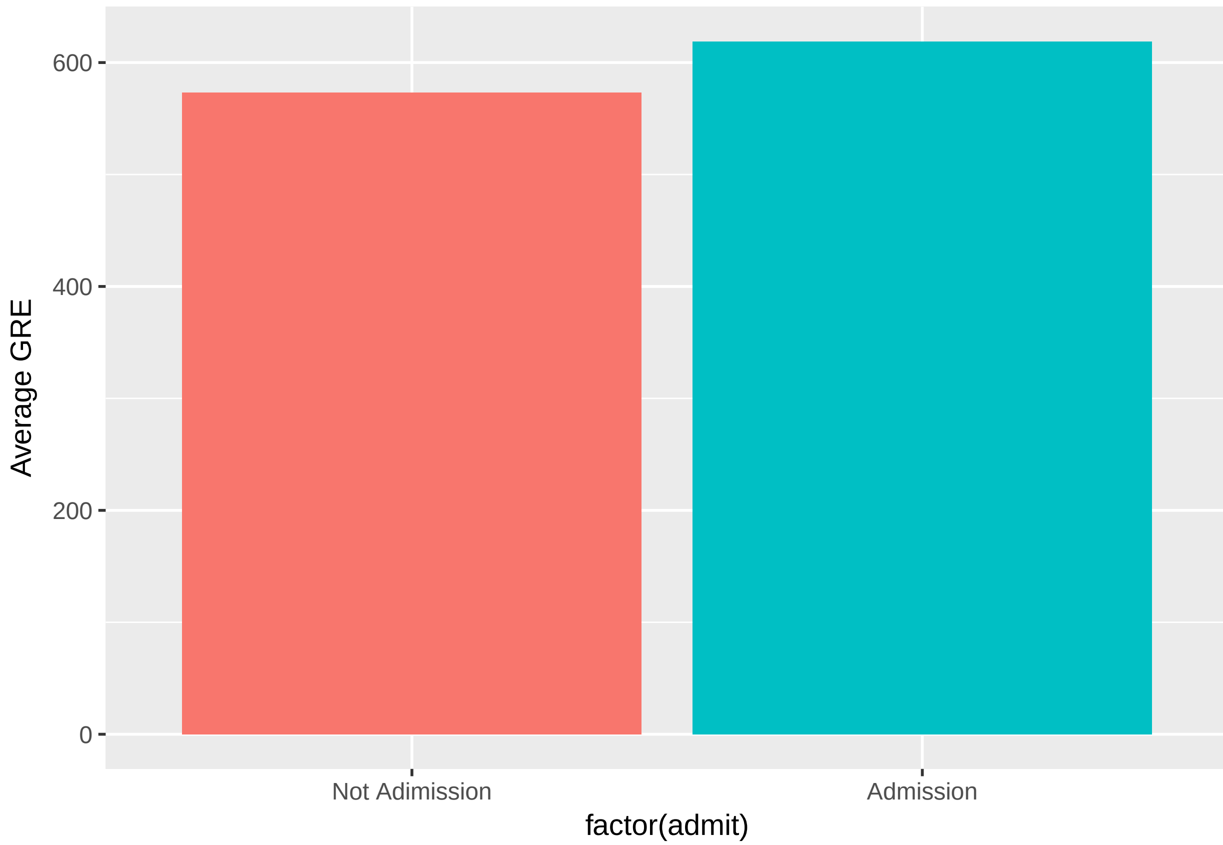
```r
require(ggplot2)
```

```
## Loading required package: ggplot2
```

```r
#### Admit vs. GRE
binary %>%
  group_by(admit) %>%
  summarise(gre=mean(gre))
```

```
## # A tibble: 2 x 2
##   admit      gre
##   <int>    <dbl>
## 1     0 573.1868
## 2     1 618.8976
```

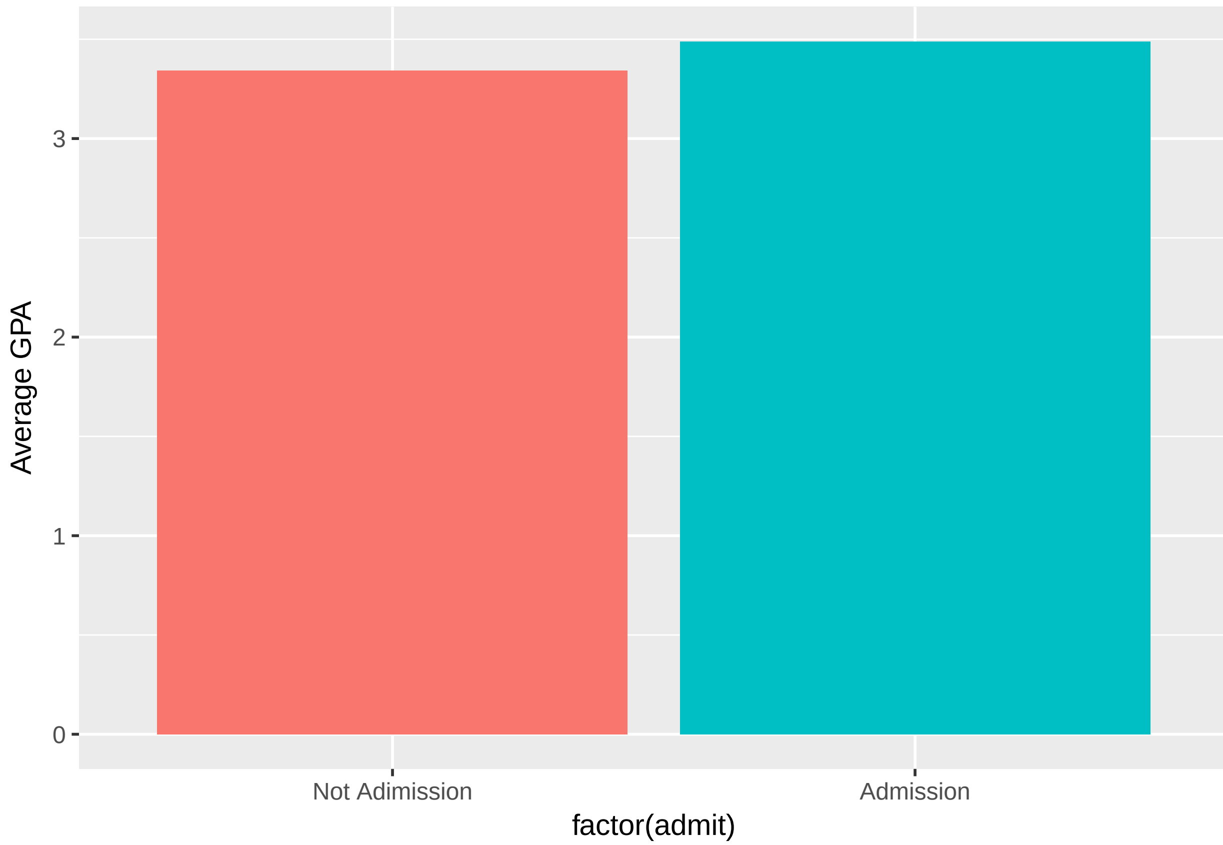
```r
ggplot(binary, aes(x=factor(admit), y=gre)) +
  geom_bar(stat="summary", fun.y=mean, aes(fill=factor(admit))) +
  ylab("Average GRE") +
  scale_x_discrete(label=c("0"="Not Adimission", "1"="Admission")) +
  theme(legend.position="none")
```




```
# Average GRE is higher for students who have an offer.



#### Admit vs. GPA
binary %>%
  group_by(admit) %>%
  summarise(gpa=mean(gpa))
```

```
## # A tibble: 2 x 2
##   admit      gpa
##   <int>    <dbl>
## 1     0 3.343700
## 2     1 3.489213
```

```
ggplot(binary, aes(x=factor(admit), y=gpa)) +
  geom_bar(stat="summary", fun.y=mean, aes(fill=factor(admit))) +
  ylab("Average GPA") +
  scale_x_discrete(label=c("0"="Not Adimission", "1"="Admission")) +
  theme(legend.position="none")
```




```
# Average GRE is slightly higher for students who have an offer.


## Logistic Regression
model = glm(admit ~., data = binary, family = "binomial")
summary(model)
```

```
##
## Call:
## glm(formula = admit ~ ., family = "binomial", data = binary)
##
## Deviance Residuals:
##     Min       1Q   Median       3Q      Max
## -1.6268  -0.8662  -0.6388   1.1490   2.0790
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -3.989979   1.139951  -3.500 0.000465 ***
## gre          0.002264   0.001094   2.070 0.038465 *
## gpa          0.804038   0.331819   2.423 0.015388 *
## rank2       -0.675443   0.316490  -2.134 0.032829 *
## rank3       -1.340204   0.345306  -3.881 0.000104 ***
## rank4       -1.551464   0.417832  -3.713 0.000205 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
```

```
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 499.98  on 399  degrees of freedom
## Residual deviance: 458.52  on 394  degrees of freedom
## AIC: 470.52
##
## Number of Fisher Scoring iterations: 4
```

```
## Deviance residuals: a measure of model fit.
## Coefficients interpretation:
## The log odds of admission increases by 0.002 when gre increase in 1 unit;
## The log odds of admission increases by 0.804 when gpa increase in 1 unit;
## Having attended an institution with rank of 2 versus an instituion with rank of 1,
## changes the log odds of admission by -0.675;
## Having attended an institution with rank of 3 versus an institution with rank of 1,
## changes the log odds of admission by -1.34;
## Having attended an institution with rank of 4 versus an institution with rank of 1,
## changes the log odds of admission by -1.551;

## odds ratio
OR = exp(coef(model))
OR
```

```
## (Intercept)         gre         gpa       rank2       rank3       rank4
##   0.0185001   1.0022670   2.2345448   0.5089310   0.2617923   0.2119375
```

```
## OR interpretation:
## one unit increase in gre, the odds of being admitted (versus not being admitted) increase by a factor
## one unit increase in gpa, the odds of being admitted increase by a factor of 2.234
## rank 2 school versus rank 1 school,
## the odds of being admitted increase by a fator of 0.5089
## rank 3 school versus rank 1 school,
## the odds of being admitted increase by a factor of 0.2618
## rank 4 school versus rank 1 school,
## the odds of being admitted increase by a factor of 0.2119

## Misclassification Rate
pred = predict(model, newdata = binary, type = "response")
pred2 = ifelse(pred > 0.5, 1, 0)
accuracy = table(pred2, binary$admit)
accuracy
```

```
##
## pred2   0   1
##     0 254  97
##     1  19  30
```

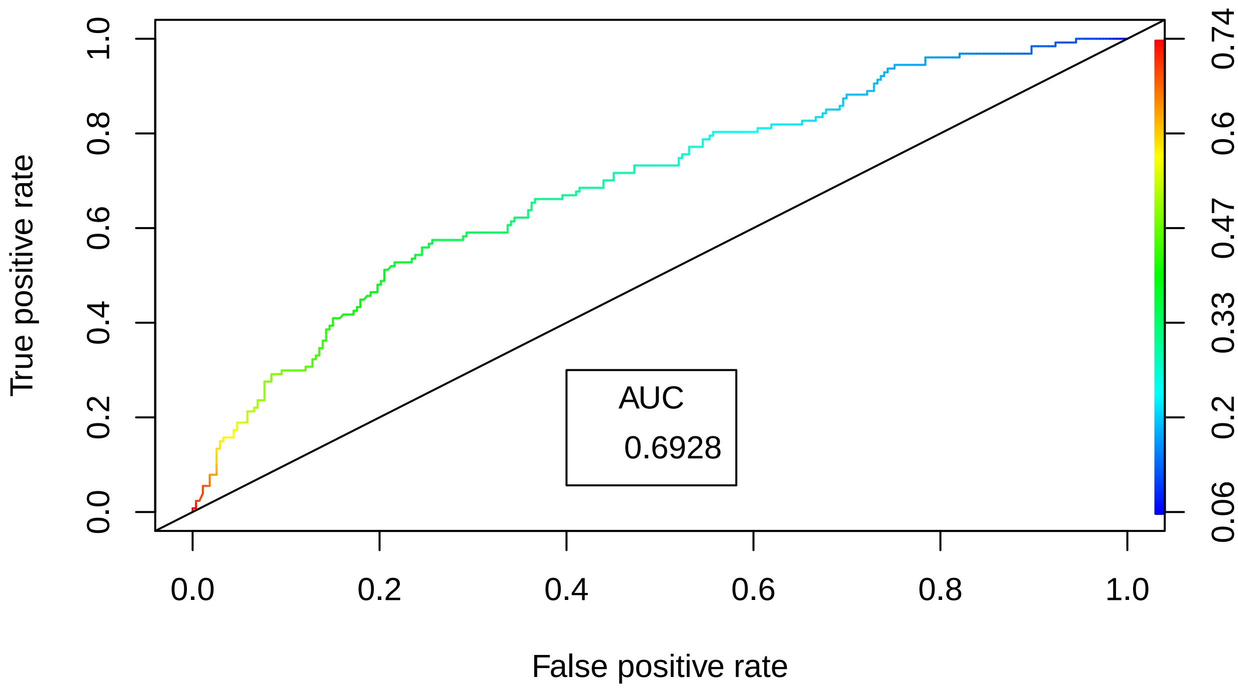
```
1 - sum(diag(accuracy))/sum(accuracy)
```

```
## [1] 0.29
```

```
## Model Performance Evaluation
## ROC Curve
## We are concerned about the area under the ROC curve (AUROC)
## The metric ranges from 0.5 to 1
## Values above 0.8 indicate that the model does a good job
require("ROCR")
```

```
## Loading required package: ROCR

## Loading required package: gplots

##
## Attaching package: 'gplots'

## The following object is masked from 'package:stats':
##
##     lowess
```

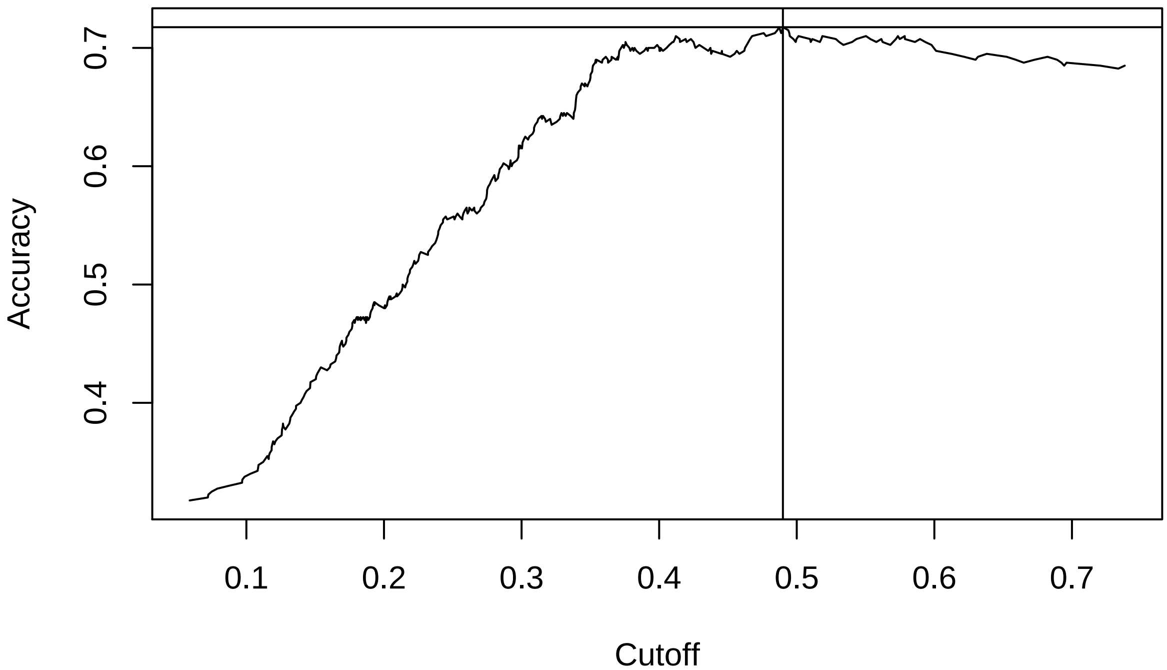
```
pred3 = prediction(pred, binary$admit)
roc = performance(pred3, "tpr", "fpr")
plot(roc,colorize=T)
abline(0,1)

## AUROC
auc = performance(pred3,'auc')
auc = slot(auc, 'y.values')[[1]]
legend(0.4,0.3,round(auc,4), title="AUC", cex=1)
```




```
## Identify Best classifier
eval = performance(pred3, "acc")
plot(eval)
max = which.max(slot(eval, "y.values")[[1]])
acc = max(slot(eval, "y.values")[[1]])
cutoff = slot(eval, "x.values")[[1]][max]
cutoff
```

```
##       271
## 0.4899411
```

```
abline(h=acc, v=cutoff)
```




```
## use the best classifier to re-classify the outcome
pred_best = ifelse(pred > cutoff, 1, 0)
cutoff
```

```
##       271
## 0.4899411
```

```
accuracy = table(pred_best, binary$admit)
accuracy
```

```
##
## pred_best   0   1
##         0 252  93
##         1  21  34
```

```
1-sum(diag(accuracy))/sum(accuracy)
```

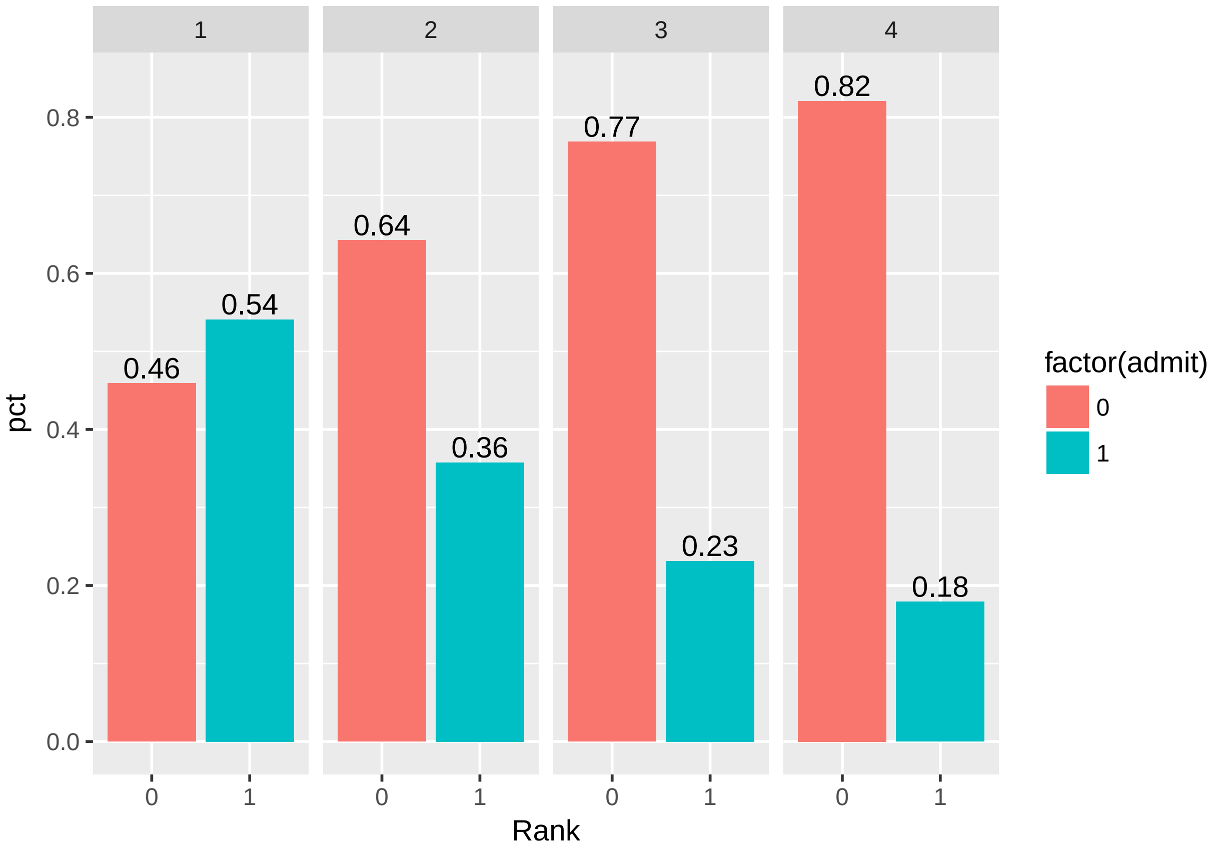
```
## [1] 0.285
```

```
#### Rank vs. Admission Rate
p2 = binary %>%
  count(rank, admit) %>%
  ungroup %>%
  group_by(rank) %>%
  mutate(pct = n/sum(n))
p2
```

```
## # A tibble: 8 x 4
## # Groups:   rank [4]
##     rank admit     n       pct
##   <fctr> <int> <int>     <dbl>
## 1      1     0    28 0.4590164
## 2      1     1    33 0.5409836
## 3      2     0    97 0.6423841
## 4      2     1    54 0.3576159
```

```
## 5      3     0    93 0.7685950
## 6      3     1    28 0.2314050
## 7      4     0    55 0.8208955
## 8      4     1    12 0.1791045
```

```r
ggplot(p2, aes(x=factor(admit), y=pct)) +
  geom_bar(stat='identity', aes(fill=factor(admit))) +
  facet_grid(~rank) +
  geom_text(aes(label=round(pct,2), y=pct+0.02)) +
  xlab("Rank")
```




```r
# Rank 1 has the highest admission rate;
# Rank 4 has the lowest admission rate.
```