Hypothesis testing and linear regression

BitTiger DS501 Week 2 HW Meina Wang

Question 1

Suppose we have two samples from two population with sample size n and m, one with mean μ_1 , variance as σ_2^2 , the other one with mean μ_2 , variance σ_2^2 . When detecting the difference in sample mean: $\delta = \mu_1 - \mu_2$, we want power is at least 0.8. If we know δ = 1, $\sigma_2^2 = \sigma_2^2$ = 1, n = m,

1) Can you calculate minimal n?
2) How n change along with δ ?
3) How n change along with σ_2^2 ?

Answer 1

1) Based on the sample size calculation equation, and z value of 1.28 correspondong to 80% power, and 1.96 corresponding to 95% significance level,

$$n = \frac{2\sigma^2(z_\beta + z_\alpha)^2}{\delta^2} = \frac{2 * 1^2(1.28 + 1.96)^2}{1^2} = 21$$

2) Based on the equation above, n increases with δ decreases.

3) Based on the equation above, n increases with σ_2^2 increase.

Question 2

A new casino game involves rolling 3 dice. The winnings are directly proportional to the total number of sixes rolled. Suppose a gambler plays the game 101 times, with the following observed counts:
Number of Sixes Number of Rolls
0 48
1 35
2 15
3 3
Test if this is fair dice. What test to use? Calculate stats and p value

Answer 2

To test if this is fair dice, we can test the observed results with the expected results from fair dice.

If the dice are fair, then the probability of having a 6 on any roll is $\frac{1}{6}$.

Assume the three dice are independent. The null hypothesis, ie., the expected values for 0,1,2,3 number of 6s are:

P(number of 6 = 0) =$(\frac{5}{6})^3$ = 0.579
P(number of 6 = 1) =$C_3^1 * \frac{1}{6} * (\frac{5}{6})^2$ = 0.347
P(number of 6 = 2) =$C_3^2 * \frac{5}{6} * (\frac{1}{6})^2$ = 0.069
P(number of 6 = 3) =$(\frac{1}{6})^3$ = 0.005

Given the above values, if the gambler plays the game 101 time, the expect number of rolls for each number of sixes would be: Number of Sixes Number of Rolls Expected number of rolls
0 48 58
1 35 35
2 15 7
3 3 1

Using the Chi square test, $\chi^2 = \sum_{i=0}^{n} \frac{(observed-expected)^2}{expected} = \frac{(48-58)^2}{58} + \frac{(35-35)^2}{35} + \frac{(15-7)^2}{7} + \frac{(3-1)^2}{1} = 14.87$

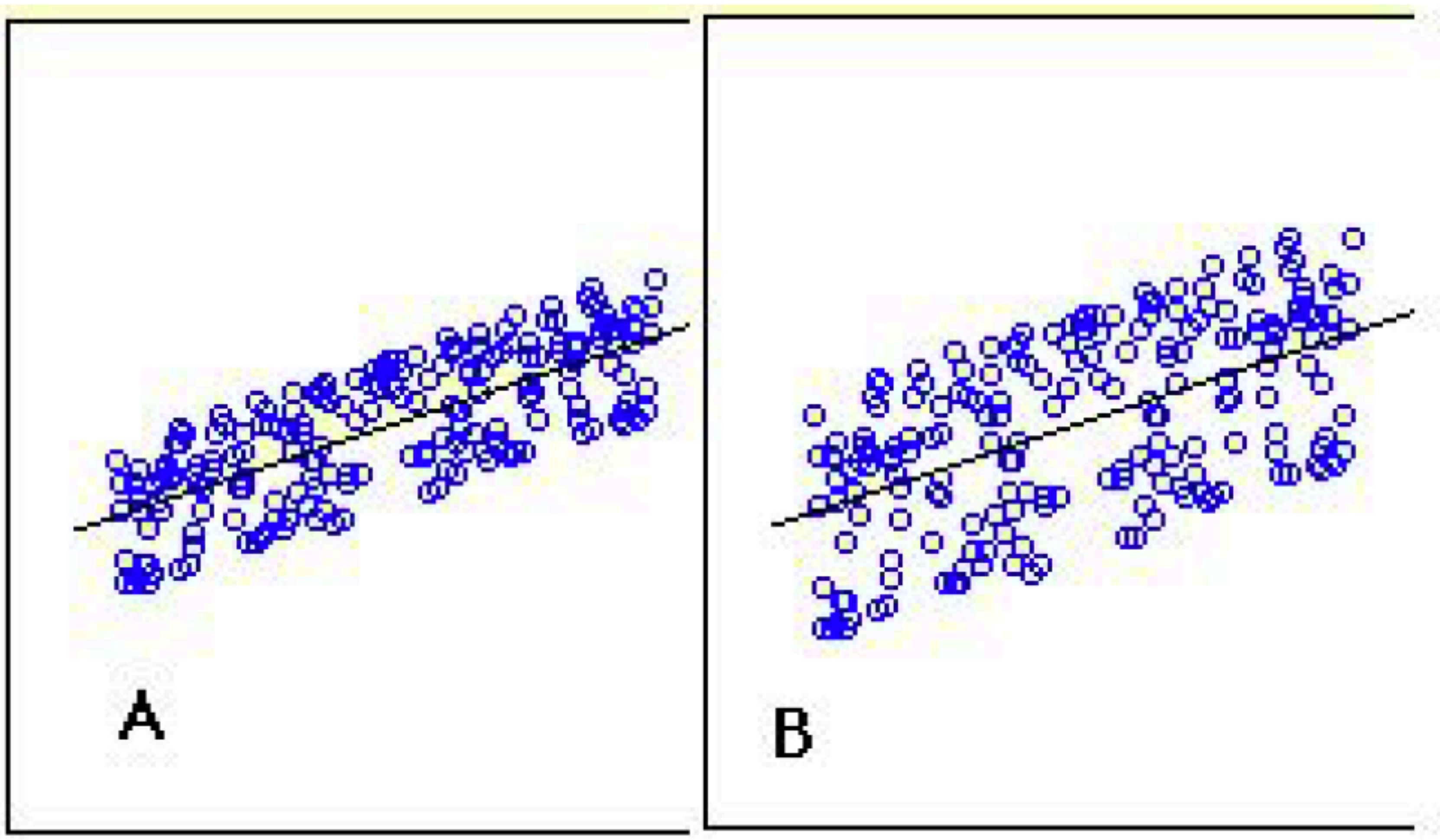
Given the chi square test value of 14.87, with degree of freedom of 3, if we use the significance level of 0.05, the threshold value to reject the numm hypothesis is 7.815.

So, based on the above calculation, with the Chi square value of 14.87 greater than the threshold value of 7.815, we can reject the null hypothesis at a significance level 0.05, and come to the conclusion that the dice are **unfair**.

Question 3

Below graphs show two fitted regression lines (A & B) on randomly generated data. Now, I want to find the sum of residuals in both cases A and B. Note: Scale is same in both graphs for both axis. X axis is independent variable and Y-axis is dependent variable.

Which of the following statement is true about sum of residuals of A and B?
A) A has higher than B
B) A has lower than B
C) Both have same
D) None of these



Answer 3

C) Both have same

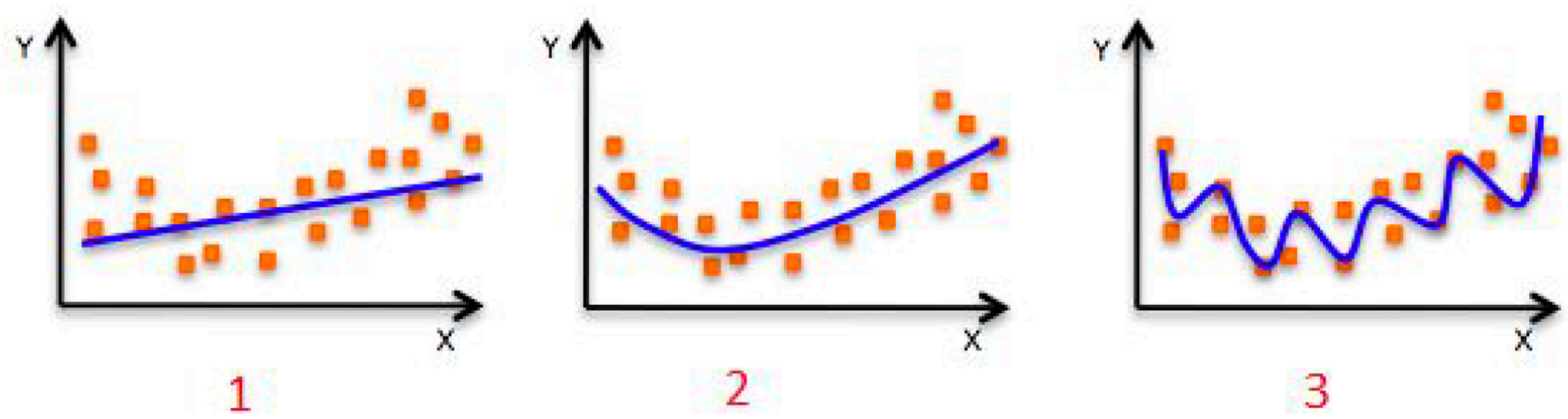
Sum of residuals are both 0 for A and B. Therefore C is correct.

Question 4

The following visualization shows the fit of three different models (in blue line) on same training data. What can you conclude from these visualizations?

1. The training error in first model is higher when compared to second and third model.
2. The best model for this regression problem is the last (third) model, because it has minimum training error.
3. The second model is more robust than first and third because it will perform better on unseen data.
4. The third model is overfitting data as compared to first and second model.
5. All models will perform same because we have not seen the test data.

A. 1 and 3
B. 1 and 3
C. 1, 3 and 4
D. Only 5



Answer 4

C. 1, 3 and 4

Model 1: Underfit

1: True, since model 1 underfits training data.

Model 2: "Just right"

3: True, since model 2 is about right fitting the training data.

Model 3: Overfit

2: False, since model 3 overfits training data.
4: True.
5: False, they will not perform the same with test data.

Therefore C.

Question 5

Using MLE (maximum likelihood estimation) to achieve coefficient estimator for multiple linear regression (i.e., more than one feature in model)

Answer 5

Assume multiple linear regression model is:

$$y = X\beta + \varepsilon,$$

The errors are normally and independently distributed with variance σ^2, or can be written as $\varepsilon \sim N(0, \sigma^2)$. The normal density function for the errors is,

$$f(\varepsilon_i) = \frac{1}{\sigma\sqrt{2\pi}} \exp\left(-\frac{1}{2\sigma^2}\varepsilon_i^2\right).$$

The likelihood function is the joint density of $\varepsilon_1 \dots \varepsilon_n$.
Therefore, the likelihood function is

$$L(\varepsilon, \beta, \sigma^2) = \prod_{i=1}^{n} f(\varepsilon_i) = \frac{1}{\sigma^n (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2\sigma^2}\varepsilon'\varepsilon\right).$$

Since the ε term can be written as $\varepsilon = Y - X\beta$, the above likelihood function then becomes,

$$L(y, X, \beta, \sigma^2) = \frac{1}{\sigma^n (2\pi)^{\frac{n}{2}}} \exp\left(-\frac{1}{2\sigma^2}(y - X\beta)'(y - X\beta)\right).$$

Similar as in the simple linear regression case, we take the log of the above equation.

$$\log L(y, X, \beta, \sigma^2) = -\frac{n}{2}\log(2\pi) - n\log\sigma - \frac{1}{2\sigma^2}(y - X\beta)'(y - X\beta).$$

Therefore, for a fixed value of σ, the log-likelihood is maximized if the term $(y - X\beta)'(y - X\beta)$ is minimized.
So, the maximum-likelihood estimator of β under the normal error condition is equivalent to the least-square estimator, where $\hat{\beta} = X'y / X'X = \frac{Cov[X, Y]}{Var[X]}$, and the maximum likelihood estimator of σ^2 is $\hat{\sigma}^2 = \frac{(y - X\hat{\beta})'(y - X\hat{\beta})}{n}$

Expand this step, showing more details:
To maximize $\log(L)$, differentiate it with respect to β and σ^2, we have,

$$\frac{\partial \log(L)}{\partial \beta} = -\frac{1}{2\sigma^2}(-2X'y - 2X'X\beta) = 0$$

$$\Rightarrow X'y = X'X\beta$$

$$\Rightarrow X'y = X'X\beta$$

$$\Rightarrow \hat{\beta} = \frac{X'y}{X'X}$$

$$\frac{\partial \log(L)}{\partial \sigma^2} = -\frac{n}{2\sigma^2} + \frac{(y-X\beta)'(y-X\beta)}{2\sigma^4} = 0$$

$$\Rightarrow \hat{\sigma}^2 = \frac{(y-X\beta)'(y-X\beta)}{n}$$

Question 6

Think about if/how you would process old features and what new features to be generated. Build the best linear regression model to explain interest rate.

Answer 6

In [1]:

```
# load the data
loan <- read.csv("/users/meinawang/Documents/bittiger/DS501/lending-club-loan-data/l
loanT <- loan
```

In [2]:

```
# remove col if 80% of the data is na
num.NA <- sort(sapply(loan, function(x){sum(is.na(x))}), decreasing = TRUE)
remain.col <- names(num.NA)[which(num.NA <= 0.8 * dim(loan)[1])]
loan <- loan[, remain.col]
```

In [3]:

```
summary(loan)
```

```
 mths_since_last_major_derog mths_since_last_delinq  tot_coll_amt
 Min.    :  0.0              Min.    :  0.0          Min.    :      0
 1st Qu.: 27.0               1st Qu.: 15.0           1st Qu.:      0
 Median : 44.0               Median : 31.0           Median :      0
 Mean   : 44.1               Mean   : 34.1           Mean   :    226
 3rd Qu.: 61.0               3rd Qu.: 50.0           3rd Qu.:      0
 Max.   :188.0               Max.   :188.0           Max.   :9152545
 NA's   :665676              NA's   :454312          NA's   :70276
  tot_cur_bal       total_rev_hi_lim    revol_util
 Min.   :      0   Min.   :      0   Min.   :  0.00
 1st Qu.:  29853   1st Qu.:  13900   1st Qu.: 37.70
 Median :  80559   Median :  23700   Median : 56.00
 Mean   : 139458   Mean   :  32069   Mean   : 55.07
 3rd Qu.: 208205   3rd Qu.:  39800   3rd Qu.: 73.60
 Max.   :8000078   Max.   :9999999   Max.   :892.30
 NA's   :70276     NA's   :70276     NA's   :502
 collections_12_mths_ex_med  delinq_2yrs      inq_last_6mths       ope
n_acc
 Min.   : 0.00000           Min.   : 0.0000   Min.   : 0.0000   Min.
```

To build a multivariate linear regression model, I will start with the variables (both numerical and categorical) that I discovered to be most significantly correlated with int_rate from last week's HW Q5.

Numerical:
last_pymnt_amnt
inq_last_6mths
total_rev_hi_lim
revol_bal
installment

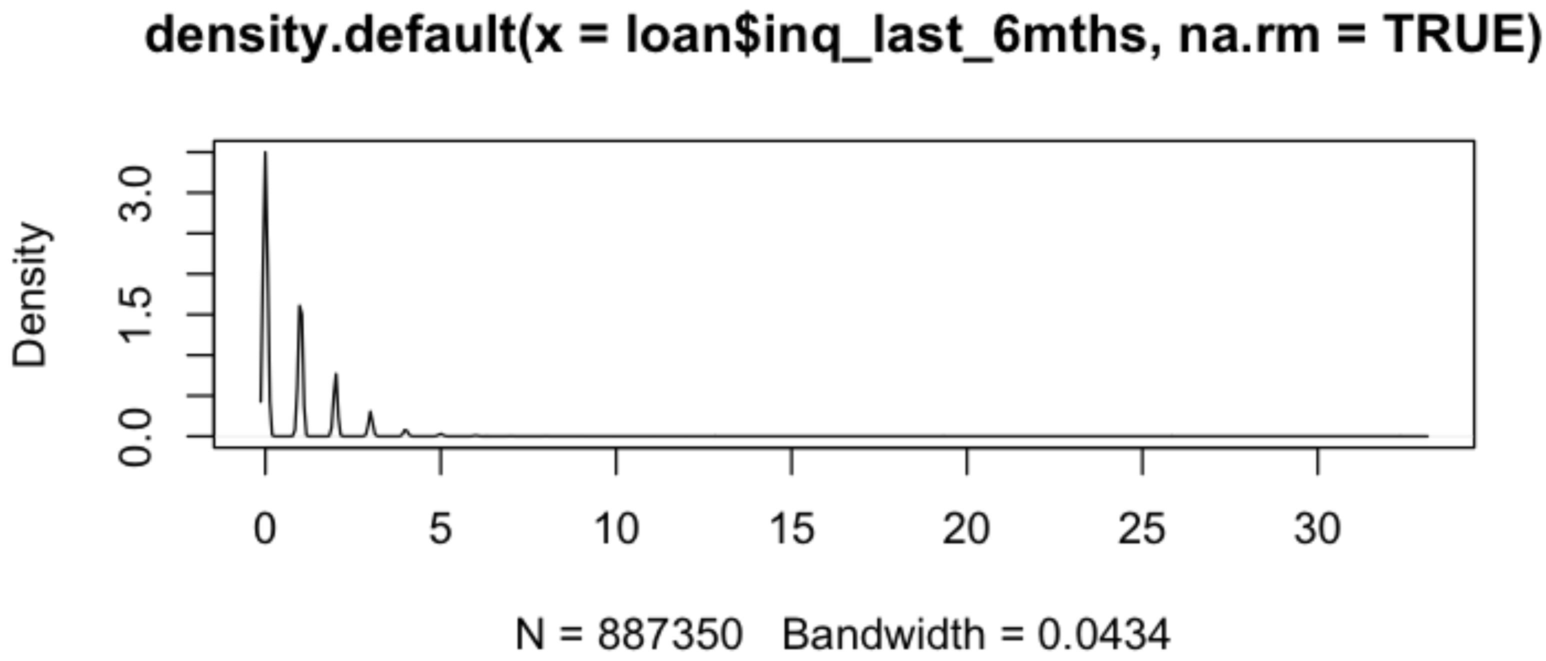
Categorical:
issue_year
loan_status
initial_list_status

One of the categorical features issue_year was removed due to the NA exceeds 80%.
grade, and sub_grade are non-available features to predict int_rate, so can not be used in the model.

So I'll start with the rest 7 features, and add more features based on intuition.

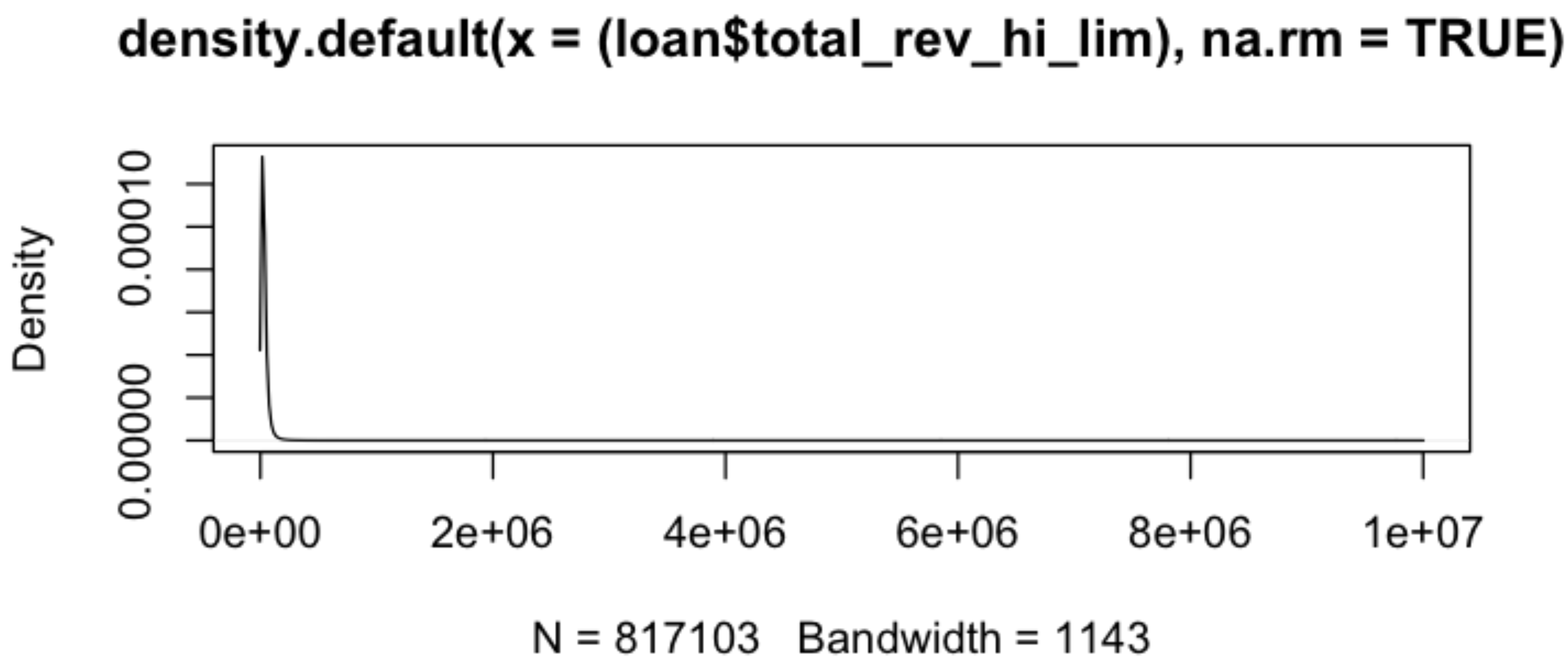
In [4]:

```
# check the pdf of features, and see if they are normal.
library(repr)

# Change plot size to 6 x 3
options(repr.plot.width=6, repr.plot.height=3)
plot(density(loan$inq_last_6mths, na.rm=TRUE))
```



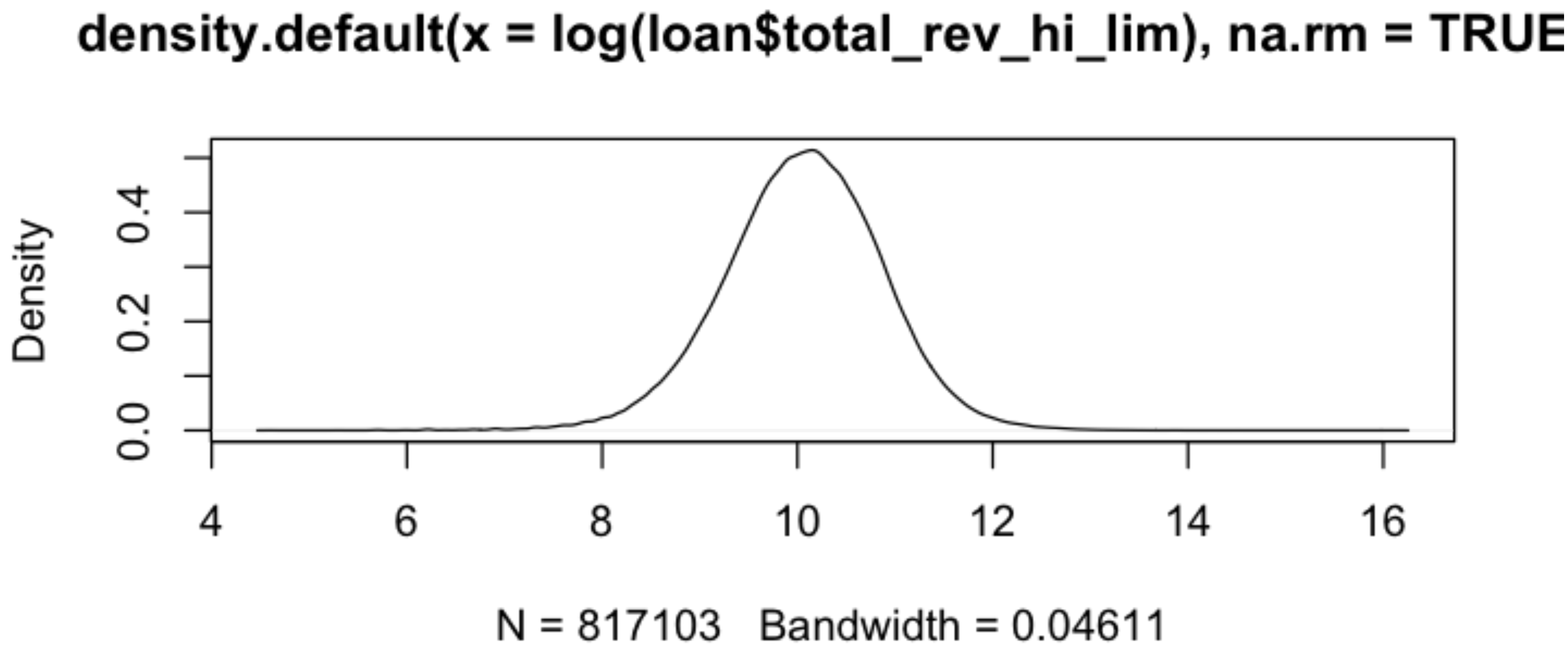
In [5]:

```
plot(density((loan$total_rev_hi_lim), na.rm=TRUE))
```



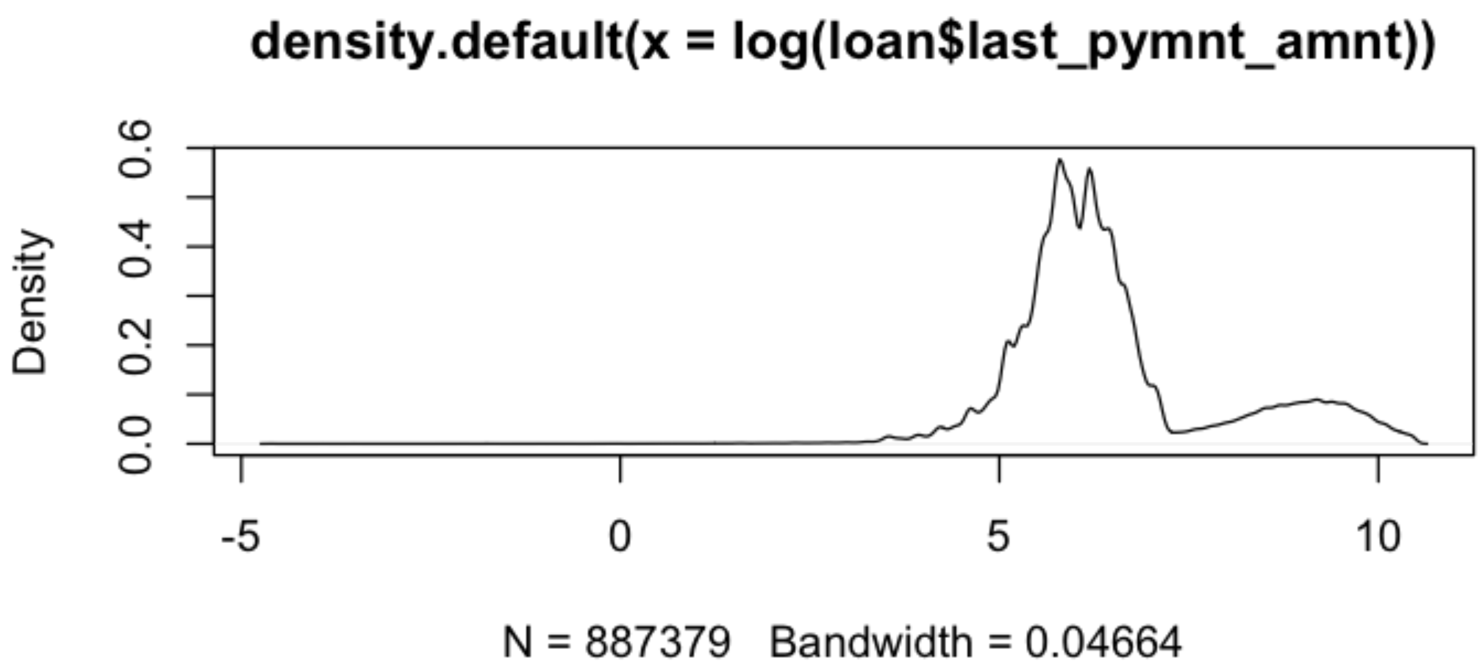
In [6]:

```
plot(density(log(loan$total_rev_hi_lim), na.rm=TRUE))
#take the log transform of the total_rev_hi_lim feature so that it's normal.
loan$total_rev_hi_lim_log = log(loan$total_rev_hi_lim + 1)
# + 1 here is to avoid taking log on 0 value, leading to inf.
```



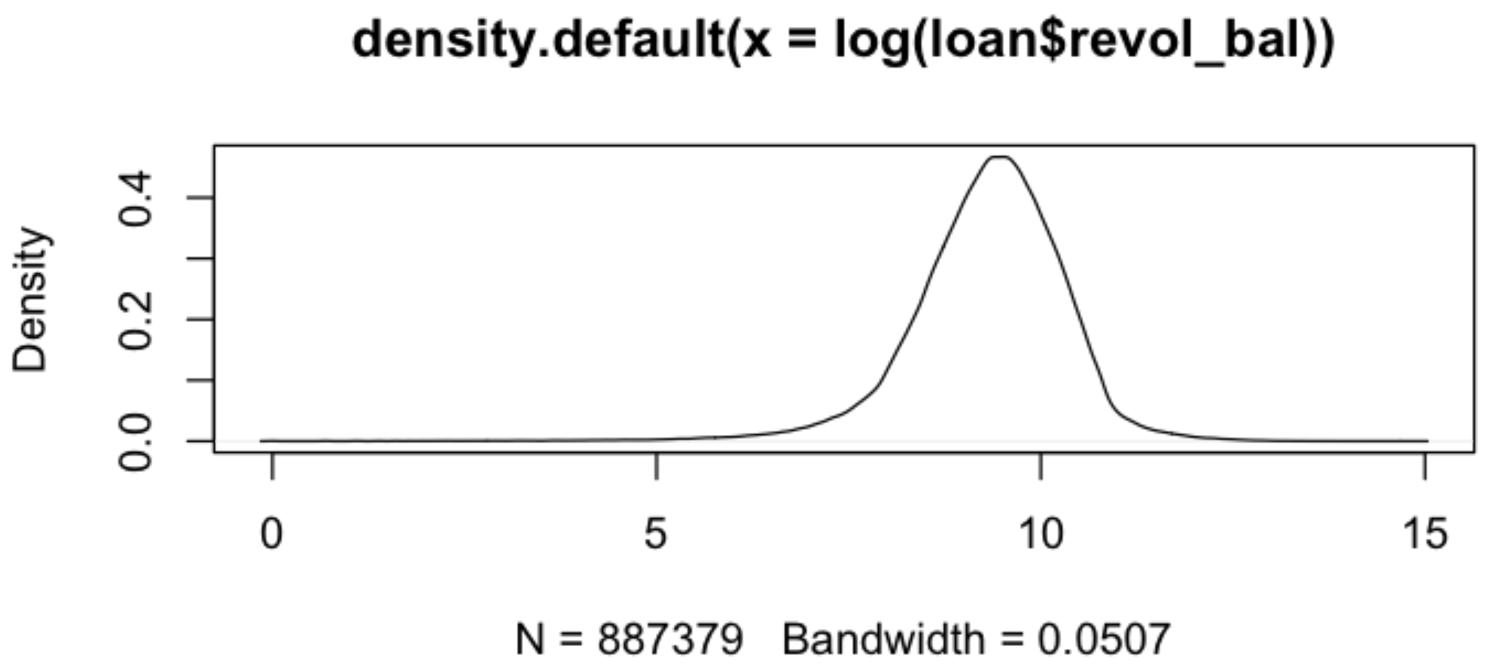
In [7]:

```
plot(density(log(loan$last_pymnt_amnt)))
#take the log transform of the last_pymnt_amnt feature so that it's closer to norma
loan$last_pymnt_amnt_log = log(loan$last_pymnt_amnt + 1)
```



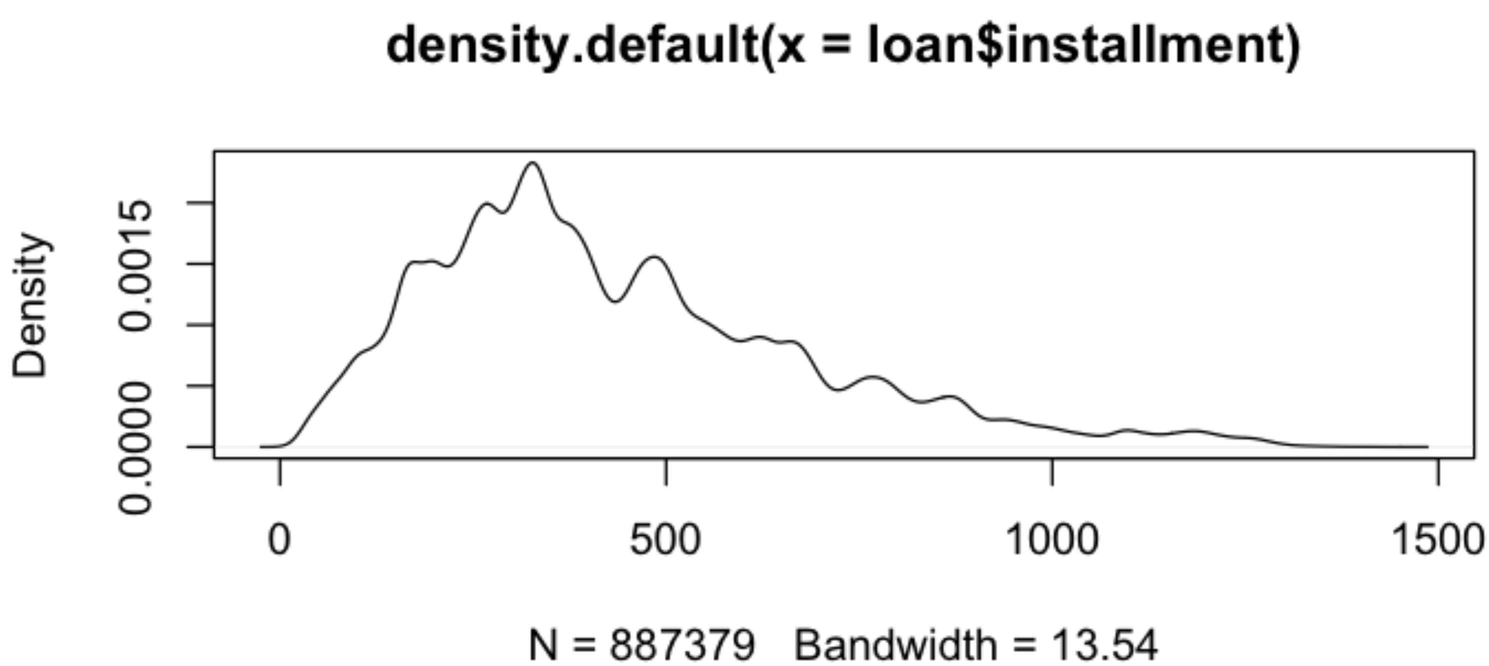
In [8]:

```
plot(density(log(loan$revol_bal)))
#take the log transform of the revol_bal feature so that it's normal.
loan$revol_bal_log = log(loan$revol_bal + 1)
```



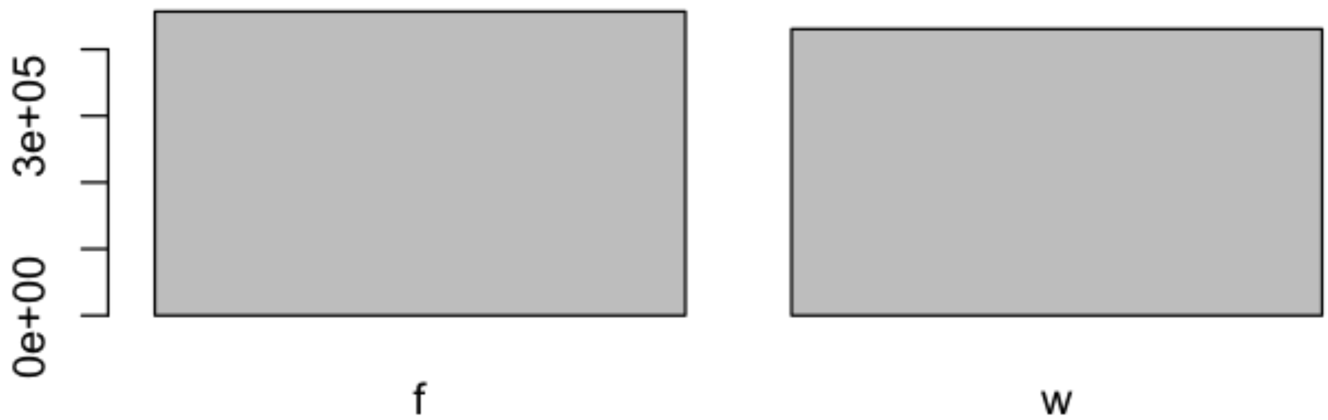
In [9]:

```
plot(density(loan$installment))
# this one looks fine
```



In [11]:

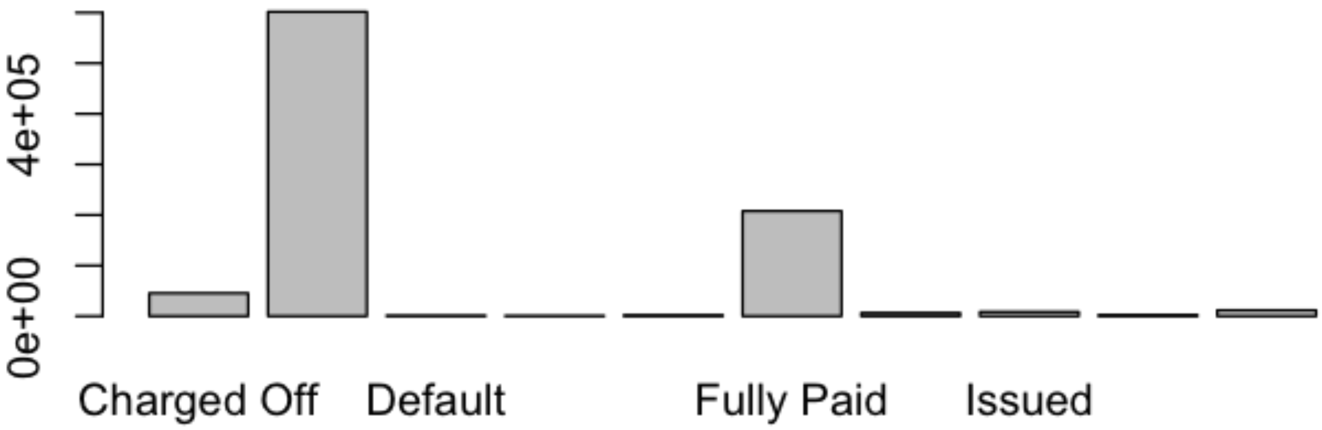
```
loan$loan_status.f <- factor(loan$loan_status)
loan$initial_list_status.f <- factor(loan$initial_list_status)

barplot(table(loan$initial_list_status.f))
```



In [12]:

```
barplot(table(loan$loan_status.f))# need to modify
levels(loan$loan_status.f)
```

'Charged Off' 'Current' 'Default' 'Does not meet the credit policy. Status:Charged Off' 'Does not meet the credit policy. Status:Fully Paid' 'Fully Paid' 'In Grace Period' 'Issued' 'Late (16-30 days)' 'Late (31-120 days)'

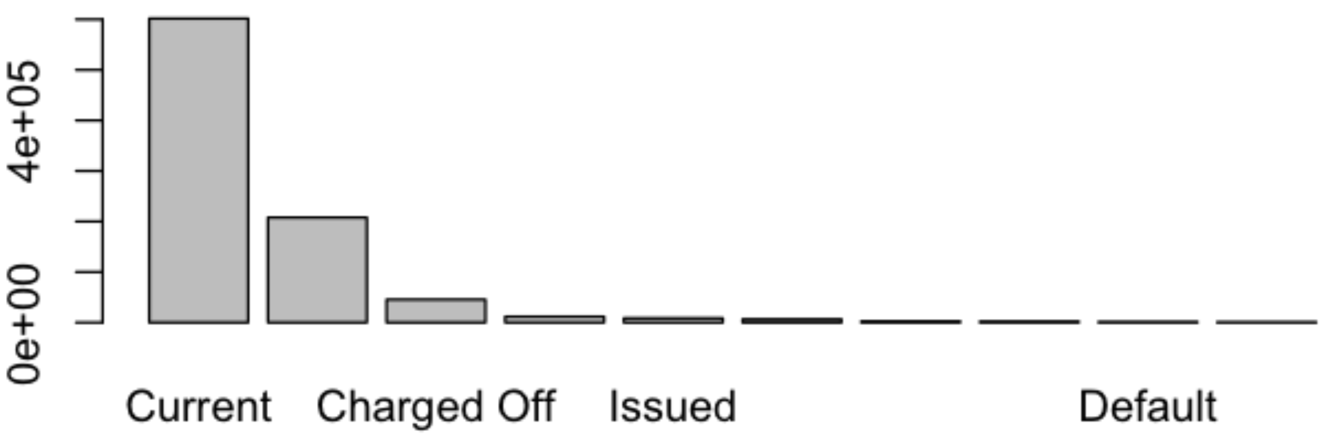


In [13]:

```
sort(table(loan$loan_status))
round(sort(table(loan$loan_status)) / dim(loan)[1] * 100, 2)
barplot(sort(table(loan$loan_status), decreasing = TRUE))
```

```
Does not meet the credit policy. Status:Charged Off
                                                761
                                            Default
                                               1219
 Does not meet the credit policy. Status:Fully Paid
                                               1988
                                  Late (16-30 days)
                                               2357
                                    In Grace Period
                                               6253
                                             Issued
                                               8460
                                 Late (31-120 days)
                                              11591
                                        Charged Off
                                              45248
                                         Fully Paid
```

```
                                                 207723
                                                Current
                                                 601779

Does not meet the credit policy. Status:Charged Off
                                                   0.09
                                                Default
                                                   0.14
 Does not meet the credit policy. Status:Fully Paid
                                                   0.22
                                     Late (16-30 days)
                                                   0.27
                                       In Grace Period
                                                   0.70
                                                 Issued
                                                   0.95
                                    Late (31-120 days)
                                                   1.31
                                            Charged Off
                                                   5.10
                                             Fully Paid
                                                  23.41
                                                Current
                                                  67.82
```



In [14]:

```
# remove certain string from loan_status
loan$loan_status <- gsub('Does not meet the credit policy. Status:',
                         '', loan$loan_status)
sort(table(loan$loan_status))
loan$loan_status_1 <- with(loan, ifelse(loan_status %in% c('Current', 'Fully Paid',
                                        1, 0))
table(loan$loan_status_1)

#train$loan_status_1 <- loan$loan_status_1[train.ind]
#test$loan_status_1 <- loan$loan_status_1[-train.ind]
```

```
           Default  Late (16-30 days)    In Grace Period             I
ssued
              1219               2357               6253
8460
Late (31-120 days)        Charged Off         Fully Paid            Cu
rrent
             11591              46009             209711             6
01779

     0      1
 67429 819950
```

Some of the features will need to be clean before use as input for the model. Impute the na value with median.

In [15]:

```
loan$annual_inc[which(is.na(loan$annual_inc))] <- median(loan$annual_inc, na.rm = T
loan$total_rev_hi_lim_log[which(is.na(loan$total_rev_hi_lim_log))] <- median(loan$to
loan$inq_last_6mths[which(is.na(loan$inq_last_6mths))] <- median(loan$inq_last_6mths
```

In [19]:

```
table(loan$home_ownership)
```

```
     ANY MORTGAGE     NONE    OTHER      OWN     RENT
       3   443557       50      182    87470   356117
```

Combine the categories into fewer levels.

In [20]:

```
loan$home_ownership <- ifelse(loan$home_ownership %in% c('ANY', 'NONE', 'OTHER'), 'C
```

In [22]:

```
int_state <- by(loan, loan$addr_state, function(x) {return(mean(x$int_rate))})
# quantile bin
loan$state_mean_int <- ifelse(loan$addr_state %in% names(int_state)
                              [which(int_state <= quantile(int_state, 0.25))],
                              'low',
                              ifelse(loan$addr_state %in% names(int_state)
                                     [which(int_state <= quantile(int_state, 0.5))],
                                     'lowmedium',
                                     ifelse(loan$addr_state %in% names(int_state)
                                            [which(int_state <= quantile(int_state,
                                            'mediumhigh','high')))
```

In [23]:

```
# after data cleaning and transformation, now train test split
set.seed(0)
train.ind <- sample(1:dim(loan)[1], 0.7* dim(loan)[1])
train <- loan[train.ind, ]
test <- loan[-train.ind, ]
```

In [24]:

```
summary(train)
```

```
 mths_since_last_major_derog mths_since_last_delinq  tot_coll_amt
 Min.   :  0.0               Min.   :  0             Min.   :      0
 1st Qu.: 27.0               1st Qu.: 15             1st Qu.:      0
 Median : 44.0               Median : 31             Median :      0
 Mean   : 44.1               Mean   : 34             Mean   :    230
 3rd Qu.: 61.0               3rd Qu.: 50             3rd Qu.:      0
 Max.   :180.0               Max.   :180             Max.   :9152545
 NA's   :465901              NA's   :317721          NA's   :49293
  tot_cur_bal       total_rev_hi_lim    revol_util
 Min.   :      0   Min.   :      0   Min.   :  0.00
 1st Qu.:  29861   1st Qu.:  13900   1st Qu.: 37.70
 Median :  80568   Median :  23700   Median : 56.00
 Mean   : 139334   Mean   :  32060   Mean   : 55.07
 3rd Qu.: 208099   3rd Qu.:  39800   3rd Qu.: 73.60
 Max.   :8000078   Max.   :9999999   Max.   :892.30
 NA's   :49293     NA's   :49293     NA's   :330
 collections_12_mths_ex_med  delinq_2yrs       inq_last_6mths       ope
n_acc
 Min.   : 0.00000           Min.   : 0.0000   Min.   : 0.0000   Min.
```

In [25]:

```
mod_hw2 <- lm(int_rate ~ last_pymnt_amnt + inq_last_6mths + total_rev_hi_lim_log
              + revol_bal_log + installment + state_mean_int+ home_ownership
              + annual_inc + loan_status_1 + initial_list_status.f+ term+ loan_amnt
              data = train, na.action=na.exclude)
```

In [26]:

```
summary(mod_hw2)
```

```
Call:
lm(formula = int_rate ~ last_pymnt_amnt + inq_last_6mths + total_rev_h
i_lim_log +
    revol_bal_log + installment + state_mean_int + home_ownership +
    annual_inc + loan_status_1 + initial_list_status.f + term +
    loan_amnt, data = train, na.action = na.exclude)

Residuals:
    Min      1Q  Median      3Q     Max
-19.544  -2.034  -0.273   1.777  32.812

Coefficients:
                           Estimate Std. Error  t value Pr(>|t|)
(Intercept)               2.245e+01  5.252e-02  427.434  < 2e-16 ***
last_pymnt_amnt           3.187e-05  7.901e-07   40.329  < 2e-16 ***
inq_last_6mths            6.516e-01  3.724e-03  174.966  < 2e-16 ***
total_rev_hi_lim_log     -1.528e+00  6.350e-03 -240.621  < 2e-16 ***
revol_bal_log             4.523e-01  4.238e-03  106.724  < 2e-16 ***
installment               4.407e-02  8.295e-05  531.337  < 2e-16 ***
state_mean_intlow        -3.349e-01  1.586e-02  -21.115  < 2e-16 ***
state_mean_intlowmedium  -9.220e-02  1.130e-02   -8.161 3.33e-16 ***
state_mean_intmediumhigh -8.341e-02  1.228e-02   -6.790 1.12e-11 ***
home_ownershipOTHER       1.011e+00  2.277e-01    4.441 8.96e-06 ***
home_ownershipOWN         4.191e-01  1.269e-02   33.029  < 2e-16 ***
home_ownershipRENT        4.831e-01  7.989e-03   60.464  < 2e-16 ***
annual_inc               -2.679e-06  5.888e-08  -45.498  < 2e-16 ***
loan_status_1            -1.224e+00  1.394e-02  -87.833  < 2e-16 ***
initial_list_status.fw   -7.359e-01  7.442e-03  -98.883  < 2e-16 ***
term 60 months            1.105e+01  1.574e-02  701.856  < 2e-16 ***
loan_amnt                -1.326e-03  2.655e-06 -499.597  < 2e-16 ***
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 2.852 on 621148 degrees of freedom
Multiple R-squared:  0.5759,    Adjusted R-squared:  0.5759
F-statistic: 5.272e+04 on 16 and 621148 DF,  p-value: < 2.2e-16
```

```
The model above has R^2 value of 0.5759, and F-score of 5.272e+04.
```

Next, I tried to applied this model to the test set, and see how it performs on unseen data.

In [27]:

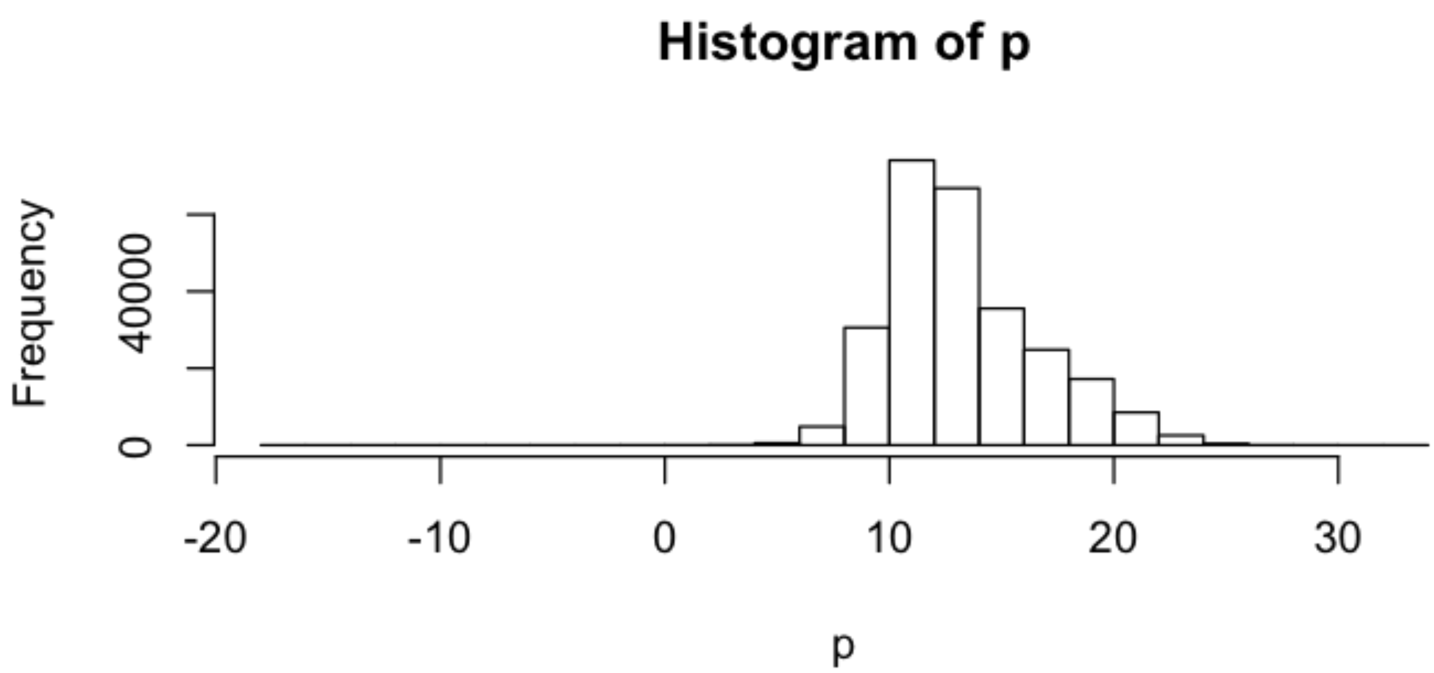
```
p <- predict(mod_hw2, test)
```

In [31]:

```
hist(p)
length(p)
length(which(is.na(p))) / length(p)
```

266214

0



In [30]:

```
which(is.na(p))
```

```
From Rstudio, there is ~8% of the predicted values are NA.  However, here it is 0.
I'm not sure why the results are different.  Will look into it.
```

In [29]:

```
actuals_preds <- data.frame(cbind(actuals=test$int_rate, predicteds=p))  # make actu
correlation_accuracy <- cor(actuals_preds)

correlation_accuracy
head(actuals_preds)
```

	actuals	predicteds
actuals	1.0000000	0.7602215
predicteds	0.7602215	1.0000000

	actuals	predicteds
7	15.96	20.53198
13	13.49	13.73807
23	11.71	11.52087
24	11.71	11.98949
26	9.91	11.29187
29	11.71	10.00997

In [ ]: