In [1]:
```
library(ggplot2)
```

```
Registered S3 methods overwritten by 'ggplot2':
  method         from
  [.quosures     rlang
  c.quosures     rlang
  print.quosures rlang
```

Importing the required libraries for svm classification ¶

In [2]:
```
library(e1071)
```

```
Warning message:
"package 'e1071' was built under R version 3.6.3"
```

In [3]:
```
library(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
```

Reading the csv data and storing it in loans

In [7]:
```
loans = read.csv("/Users/sanju/Desktop/R Programming/Datasets/loan_
```

In [9]:
```
head(loans)
```

credit.policy	purpose	int.rate	installment	log.annual.inc	dti	fico	days.with.cr
1	debt_consolidation	0.1189	829.10	11.35041	19.48	737	5639
1	credit_card	0.1071	228.22	11.08214	14.29	707	2760
1	debt_consolidation	0.1357	366.86	10.37349	11.63	682	4710
1	debt_consolidation	0.1008	162.34	11.35041	8.10	712	2699
1	credit_card	0.1426	102.92	11.29973	14.97	667	4066
1	credit_card	0.0788	125.13	11.90497	16.98	727	6120

In [10]:
```
str(loans)
```

```
'data.frame':   9578 obs. of  14 variables:
 $ credit.policy    : int  1 1 1 1 1 1 1 1 1 1 ...
 $ purpose          : Factor w/ 7 levels "all_other","credit_card"
,..: 3 2 3 3 2 2 3 1 5 3 ...
 $ int.rate         : num  0.119 0.107 0.136 0.101 0.143 ...
 $ installment      : num  829 228 367 162 103 ...
 $ log.annual.inc   : num  11.4 11.1 10.4 11.4 11.3 ...
 $ dti              : num  19.5 14.3 11.6 8.1 15 ...
 $ fico             : int  737 707 682 712 667 727 667 722 682 707
...
 $ days.with.cr.line: num  5640 2760 4710 2700 4066 ...
 $ revol.bal        : int  28854 33623 3511 33667 4740 50807 3839
24220 69909 5630 ...
 $ revol.util       : num  52.1 76.7 25.6 73.2 39.5 51 76.8 68.6 5
1.1 23 ...
 $ inq.last.6mths   : int  0 0 1 1 0 0 0 0 1 1 ...
 $ delinq.2yrs      : int  0 0 0 0 1 0 0 0 0 0 ...
 $ pub.rec          : int  0 0 0 0 0 0 1 0 0 0 ...
 $ not.fully.paid   : int  0 0 0 0 0 0 1 1 0 0 ...
```

In last 4 rows we can see there are only 2 levels and stored in int datatype.We need to convert them into factors

In [12]:
```
sum(is.na(loans))
```

```
0
```

Checking for missing values using is.na()..No missing values found

In [13]:
```
summary(loans)
```

```
 credit.policy                 purpose        int.rate       insta
llment
 Min.   :0.000   all_other         :2331   Min.   :0.0600   Min.
: 15.67
 1st Qu.:1.000   credit_card       :1262   1st Qu.:0.1039   1st Qu
.:163.77
 Median :1.000   debt_consolidation:3957   Median :0.1221   Median
:268.95
 Mean   :0.805   educational       : 343   Mean   :0.1226   Mean
:319.09
 3rd Qu.:1.000   home_improvement  : 629   3rd Qu.:0.1407   3rd Qu
.:432.76
 Max.   :1.000   major_purchase    : 437   Max.   :0.2164   Max.
:940.14
                 small_business    : 619
 log.annual.inc        dti              fico       days.with.cr.li
ne
 Min.   : 7.548   Min.   : 0.000   Min.   :612.0   Min.   :  179
 1st Qu.:10.558   1st Qu.: 7.213   1st Qu.:682.0   1st Qu.: 2820
 Median :10.929   Median :12.665   Median :707.0   Median : 4140
 Mean   :10.932   Mean   :12.607   Mean   :710.8   Mean   : 4561
 3rd Qu.:11.291   3rd Qu.:17.950   3rd Qu.:737.0   3rd Qu.: 5730
 Max.   :14.528   Max.   :29.960   Max.   :827.0   Max.   :17640

   revol.bal         revol.util    inq.last.6mths    delinq.2yrs
 Min.   :      0   Min.   :  0.0   Min.   : 0.000   Min.   : 0.000
0
 1st Qu.:   3187   1st Qu.: 22.6   1st Qu.: 0.000   1st Qu.: 0.000
0
 Median :   8596   Median : 46.3   Median : 1.000   Median : 0.000
0
 Mean   :  16914   Mean   : 46.8   Mean   : 1.577   Mean   : 0.163
7
 3rd Qu.:  18250   3rd Qu.: 70.9   3rd Qu.: 2.000   3rd Qu.: 0.000
0
 Max.   :1207359   Max.   :119.0   Max.   :33.000   Max.   :13.000
0

    pub.rec        not.fully.paid
 Min.   :0.00000   Min.   :0.0000
 1st Qu.:0.00000   1st Qu.:0.0000
 Median :0.00000   Median :0.0000
 Mean   :0.06212   Mean   :0.1601
 3rd Qu.:0.00000   3rd Qu.:0.0000
 Max.   :5.00000   Max.   :1.0000
```

We can see the summary of data minimum int rate is 0.06 and maximum interest rate is 0.2164

Converting into factors from int datatype and replacing them. using factor function to convert it into factors

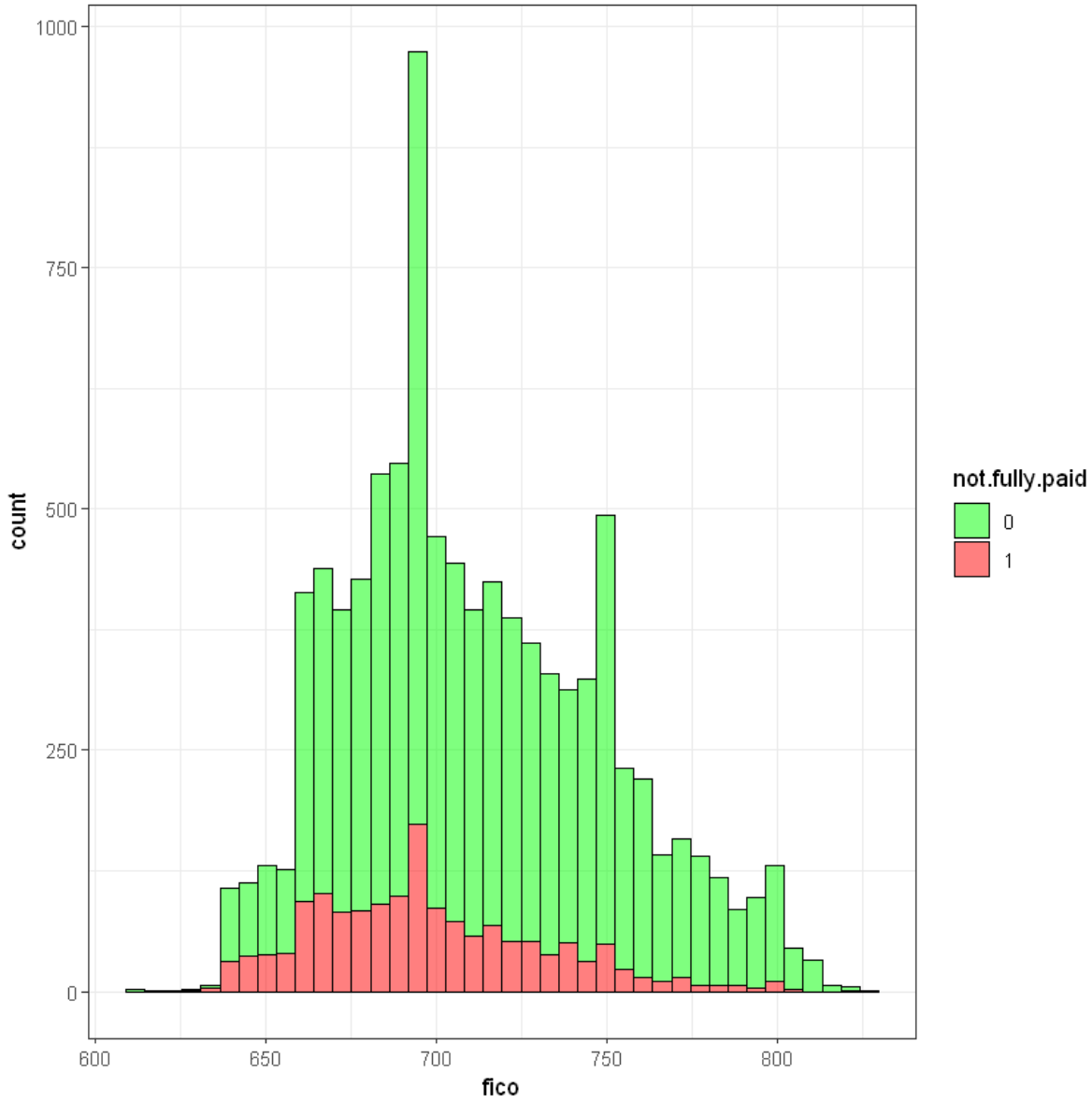
In [15]:
```
loans$credit.policy = factor(loans$credit.policy)
```

In [16]:
```
loans$inq.last.6mths = factor(loans$inq.last.6mths)
loans$delinq.2yrs = factor(loans$delinq.2yrs)
loans$pub.rec = factor(loans$pub.rec)
loans$not.fully.paid = factor(loans$not.fully.paid)
```

Exploratory Data Analysis

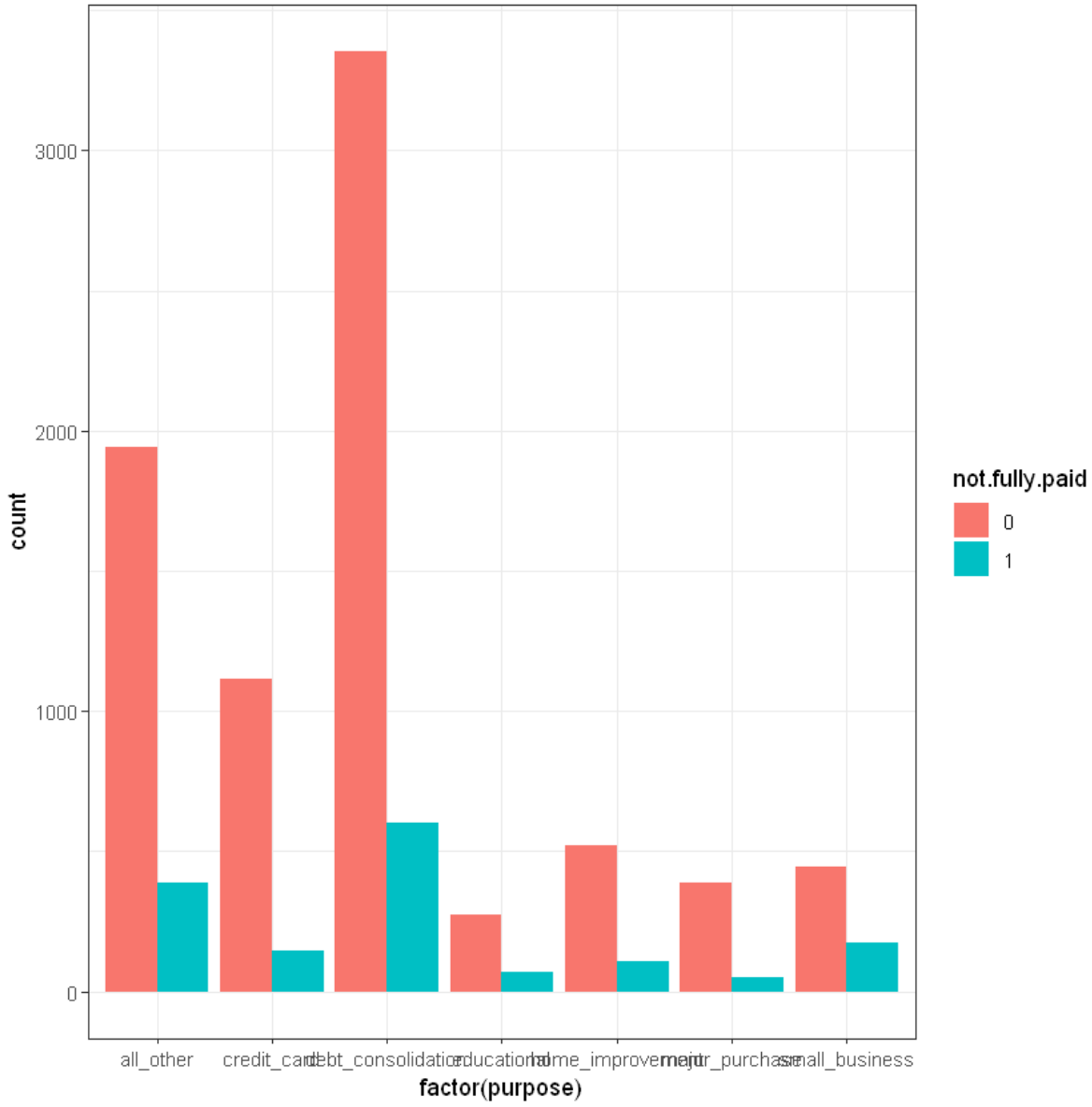
Let's use ggplot2 to visualize the data!! create a histogram of fico scores colored by not.fully.paid

In [18]:
```
pl = ggplot(loans, aes(x=fico))
pl = pl+geom_histogram(aes(fill = not.fully.paid), color = 'black',
pl + scale_fill_manual(values = c('green','red')) + theme_bw()
```



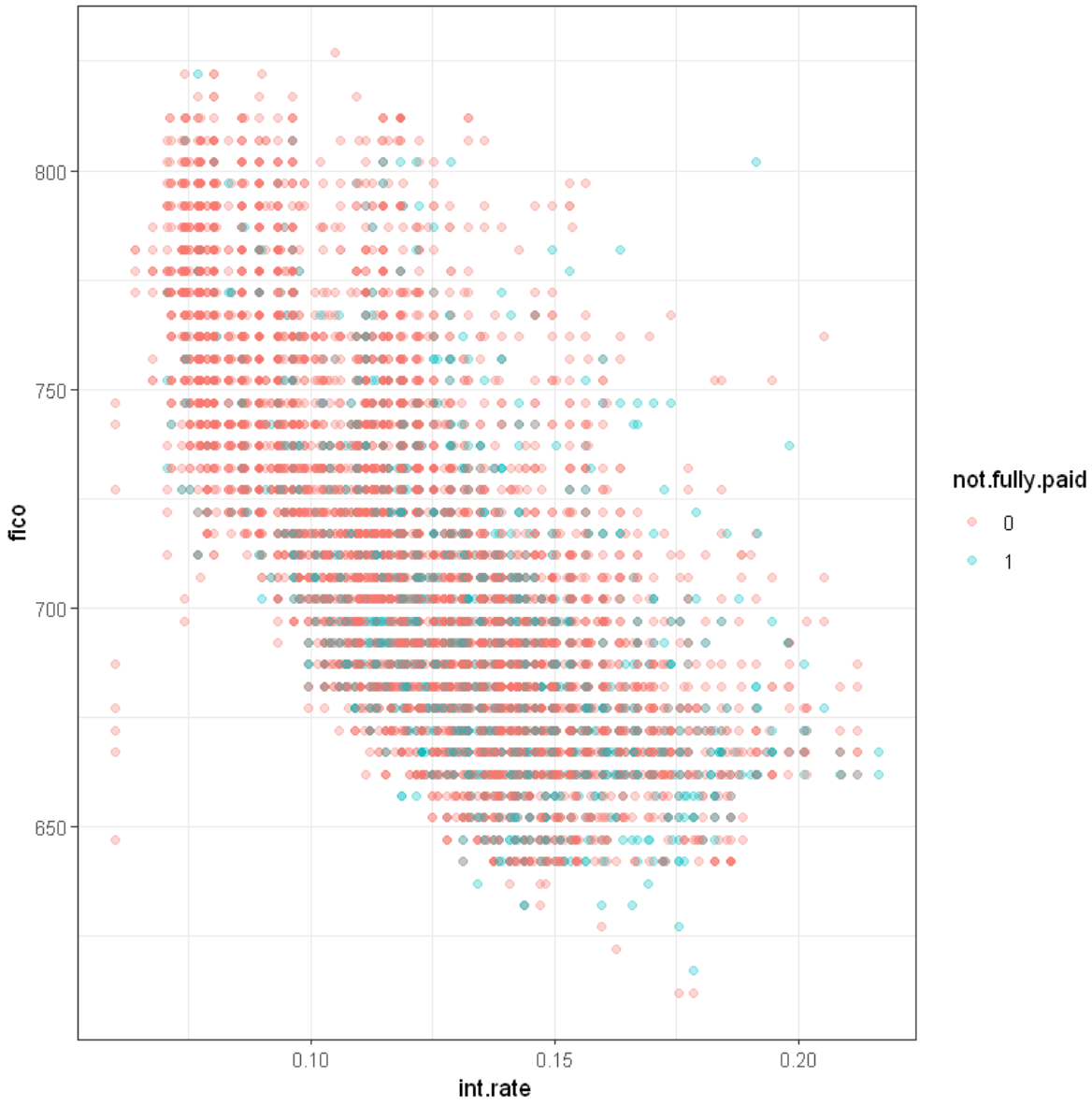
Borrowers with a high fico score tends to pay back their loans. from 660 to 750 there are a more people who haven't fully paid yet

In [20]:
```
pl = ggplot(loans, aes(x = factor(purpose)))
pl = pl + geom_bar(aes(fill = not.fully.paid), position = 'dodge')
pl + theme_bw()
```



from the graph we can see that debt_consolidation purpose customers almost 4000 of them not fully paid loans.

In [21]:
```
ggplot(loans, aes(int.rate,fico)) + geom_point(aes(color = not.full
```



In [23]:
```
library(caTools)
```

Importing required library for svm to execute

In [24]:
```
set.seed(101)
```

In [27]:
```
spl = sample.split(loans$not.fully.paid, 0.7)
```

creating the split based on target variable and split ratio is 70:30

In [29]:
```
train = subset(loans, spl = TRUE)
```

70% of data storing in train by giving spl equal to true

In [30]:
```
test = subset(loans, spl = FALSE)
```

30% of data storing in test by giving spl equal to false

TRAINING THE MODEL USING SVM FUNCTION WITHOUT TUNING THE COST AND GAMMA

In [32]:
```
model = svm(not.fully.paid ~ ., data = train)
```

In [33]:
```
summary(model)
```

```
Call:
svm(formula = not.fully.paid ~ ., data = train)


Parameters:
   SVM-Type:  C-classification
 SVM-Kernel:  radial
       cost:  1

Number of Support Vectors:  4030

 ( 2497 1533 )


Number of Classes:  2

Levels:
 0 1
```

From the summary support vectors on margin are 4030 used to divide the margins

Use predict to predict new values from the test set using model

In [34]:
```
predicted.values = predict(model, test[1:13])
```

Removing the label column from test and passing model into predict function to predict the data without labels

In [35]:
```
table(predicted.values, test$not.fully.paid)
```

```
predicted.values    0    1
               0 8045 1532
               1    0    1
```

From the confusion matrix we can see we got bad results as cost and gamma values are not properly defined. It incorrectly classified the not fully paid into paid column.

gamma and cost parameters are bad so we need to tune the results

In [36]:
```
model = svm(not.fully.paid ~ ., data = train, gamma = 1, cost = 10)
```

Training the model with better gamma and cost function

In [37]:
```
predicted.values1 = predict(model, test[1:13])
```

Predicting using predict function after tuning the model

Printing confusion matrix

In [38]:
```
table(predicted.values1, test$not.fully.paid)
```

```
predicted.values1    0    1
                0 8045    5
                1    0 1528
```

We have got good results by tuning the model. svm correctly classified. paid members are 8045 and not paid members are 1528.

Training the model with different gamma and cost function

In [43]:
```
model = svm(not.fully.paid ~ ., data = train, gamma = 0.1, cost = 2
```

In [44]:
```
predicted.values2 = predict(model, test[1:13])
```

In [45]:
```
table(predicted.values2, test$not.fully.paid)
```

```
predicted.values2    0    1
                0 8041  431
                1    4 1102
```

We have got good results by tuning the model. svm correctly classified. paid members are 8041 and not paid members are 1102.