TEAM AAP_HW2

Sonakshi Malhotra | Prasad Gujela | Shan Lin | Taeyong Kim

PURDUE UNIVERSITY

---------- SQL ----------

1. Make create table and copy command statements in your Postgres database to load these files in their own respective tables. Provide a screenshot of the SQL codes and proof of the created table in your db.

Ans)

Creating table tr

```
CREATE TABLE tr (
    msno char(256),
    is_churn numeric,
    sub_churn numeric,
    city numeric,
    bd numeric,
    gender char(50),
    registered_via numeric,
    registration_init_time date,
    avgpayment numeric,
    latest_renew numeric,
    latest_cancel numeric,
    pct_cancel numeric,
    methother numeric,
    meth36 numeric,
    meth37 numeric,
    meth38 numeric,
    meth39 numeric,
    meth40 numeric,
    meth41 numeric,
    plan410 numeric,
    plan195 numeric,
    plan7 numeric,
    plan31 numeric,
    plan0  numeric,
    plan30 numeric,
    planother numeric,
    pric149 numeric,
    pric99 numeric,
    pric0 numeric,
    pric129 numeric,
    pric180 numeric,
    pric150 numeric,
    pricother numeric,
    numdays numeric,
    daysrange numeric,
    num25 numeric,
```

```
COPY tr FROM 'C:/Users/sonak/Desktop/Work/Falll7/Data Mining/Course Content/2_Oct30/DataFiles/tr.csv' CSV HEADER DELIMITER '|';
```

```
51  SELECT * FROM tr LIMIT 20;
52
```

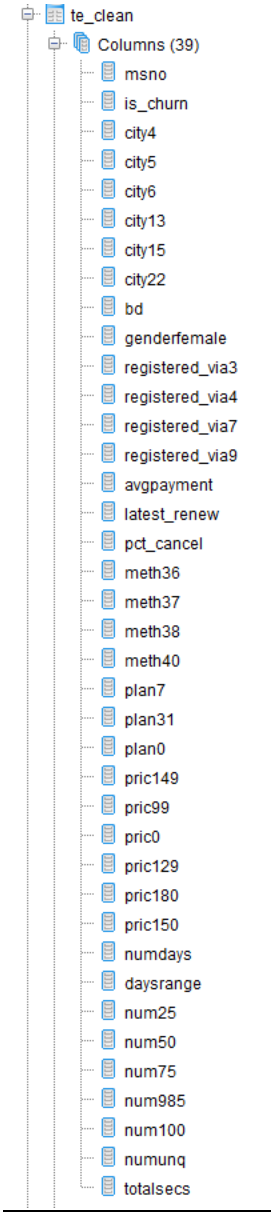
Data Output | Explain | Messages | History

	msno character (256)	is_churn numeric	sub_churn numeric	city numeric	bd numeric	gender character (50)	registered_via numeric	registration_init_time date	avgpayment numeric	latest_renew numeric	latest_cancel numeric	pct_cancel numeric	methother numeric	meth36 numeric	meth37 numeric	meth38 numeric
1	kRwY/BAWpTL0...	0	0	1	0	[null]	7	2016-11-17	000000000000	1	0	00000000000	0	0	0	
2	7Oea1ZIin+Jrq...	0	0	5	21	male	7	2015-04-18	000000000000	1	0	00000000000	0	0	0	
3	9rWSKUch5tc3...	0	0	1	18	female	3	2015-08-31	000000000000	0	0	00000000000	1	0	0	
4	WoWpXM7hZhR...	0	0	1	0	[null]	9	2016-06-22	000000000000	1	0	00000000000	1	0	0	
5	8b+0IwL8ek84...	0	0	5	34	male	9	2011-01-16	000000000000	1	0	00000000000	0	0	0	
6	4HfuSRIFN+L1A...	0	0	1	0	[null]	7	2015-07-11	545454545455	1	0	45454545455	0	0	0	
7	fX/qPe82QloRP...	0	0	1	0	[null]	4	2017-01-03	000000000000	1	0	00000000000	1	0	0	
8	WcdDDDB0xo1c...	1	0	1	0	[null]	7	2011-11-25	000000000000	1	0	00000000000	0	0	0	
9	T5hAEO64W0jr...	0	1	11	39	male	9	2007-11-01	000000000000	1	0	00000000000	1	0	0	
10	J4Dm3Yr0IOW4...	0	1	17	27	male	9	2004-07-30	000000000000	1	0	00000000000	1	0	0	
11	cZWEijTN6rApC...	0	1	9	24	male	9	2008-02-11	000000000000	1	0	00000000000	1	0	0	
12	Ad2uOmhYk/GG...	0	0	1	0	[null]	7	2015-11-02	000000000000	1	0	00000000000	0	0	0	
13	5O7Ea32olo2/J...	0	0	1	0	[null]	7	2015-11-28	000000000000	1	0	00000000000	0	0	0	

Creating te_clean table:

```
CREATE TABLE te_clean (
    msno char(256),
    is_churn numeric,
    city4   numeric,
    city5   numeric,
    city6   numeric,
    city13  numeric,
    city15  numeric,
    city22  numeric,
    bd  numeric,
    genderfemale numeric,
    registered_via3 numeric,
    registered_via4 numeric,
    registered_via7 numeric,
    registered_via9 numeric,
    avgpayment  numeric,
    latest_renew numeric,
    pct_cancel numeric,
    meth36 numeric,
    meth37  numeric,
    meth38  numeric,
    meth40  numeric,
    plan7   numeric,
    plan31   numeric,
    plan0   numeric,
    pric149 numeric,
    pric99  numeric,
    pric0   numeric,
    pric129 numeric,
    pric180 numeric,
    pric150 numeric,
    numdays numeric,
    daysrange   numeric,
    num25   numeric,
    num50   numeric,
    num75   numeric,
```

```
COPY te_clean FROM 'C:/Users/sonak/Desktop/Work/Fall17/Data Mining/Course Content/2_Oct30/DataFiles/te_clean.csv' CSV HEADER DELIMITER '|';
```

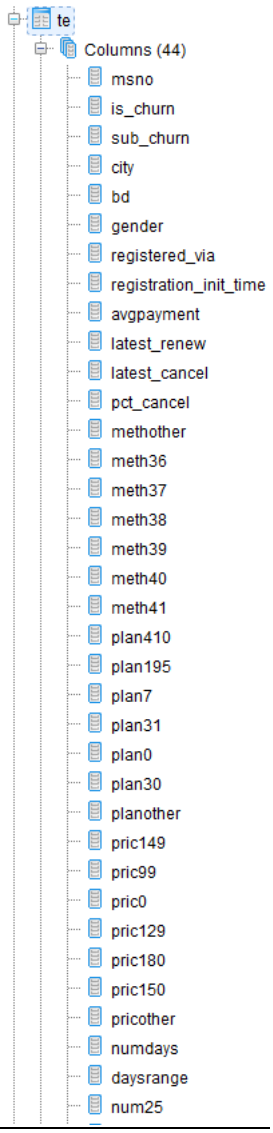



```
96
97  SELECT * FROM te_clean LIMIT 10;
```

Data Output | Explain | Messages | History

	msno character (256)	is_churn numeric	city4 numeric	city5 numeric	city6 numeric	city13 numeric	city15 numeric	city22 numeric	bd numeric	genderfemale numeric	registered_via3 numeric	registered_via4 numeric	registered_via7 numeric	registered_via9 numeric	avgpayment numeric	latest_ren numeric
1	2DiqpoRLcC4Li...	0	0	1	0	0	0	0	33	0	0	0	1	0	149	
2	b7ZwpXDkz+Rf...	0	0	0	0	0	0	0	0	1	0	0	1	0	99	
3	zkGeiFR/WL159...	0	0	0	0	0	0	0	0	0	0	0	1	0	149	
4	sl6cmmnT6vCzn...	0	1	0	0	0	0	0	0	1	0	0	0	1	041666666667	
5	NEhMZO1FRPn...	0	0	1	0	0	0	0	0	0	0	0	0	1	149	
6	5xy2VqndFMDy...	0	0	0	1	0	0	0	0	0	0	0	0	0	74.5	
7	a+ijzs2MRIgMvI...	0	0	0	0	0	0	0	0	0	0	0	1	0	99	
8	o0G4BHzfKq66...	0	0	0	0	0	0	0	0	1	0	0	1	0	99	
9	GmIB3JZPKNZO...	0	0	0	0	0	0	0	0	0	0	0	1	0	560975609756	
10	AMBC+lrNQNEQ...	0	0	0	0	0	0	1	21	0	0	0	0	1	163.9	

Creating te table:




```
CREATE TABLE te (
   msno char(256),
   is_churn numeric,
   sub_churn numeric,
   city numeric,
   bd numeric,
   gender char(50),
   registered_via numeric,
   registration_init_time d
   avgpayment numeric,
   latest_renew numeric,
   latest_cancel numeric,
   pct_cancel numeric,
   methother numeric,
   meth36 numeric,
   meth37 numeric,
   meth38 numeric,
   meth39 numeric,
   meth40 numeric,
   meth41 numeric,
   plan410 numeric,
   plan195 numeric,
   plan7 numeric,
   plan31 numeric,
   plan0  numeric,
   plan30 numeric,
   planother numeric,
   pric149 numeric,
   pric99 numeric,
   pric0 numeric,
   pric129 numeric,
   pric180 numeric,
   pric150 numeric,
   pricother numeric,
   numdays numeric,
   daysrange numeric,
```

```
COPY te FROM 'C:/Users/sonak/Desktop/Work/Fall17/Data Mining/Course Content/2_Oct30/DataFiles/te.csv' CSV HEADER DELIMITER '|';
```

```
SELECT * FROM te LIMIT 10;
```

Data Output | Explain | Messages | History

msno character (256)	is_churn numeric	sub_churn numeric	city numeric	bd numeric	gender character (50)	registered_via numeric	registration_init_time date	avgpayment numeric	latest_renew numeric	latest_cancel numeric	pct_cancel numeric	methother numeric	meth36 numeric	meth37 numeric	me nu
hFV9J8yeZfOeN...	0	1	13	18	male	3	2013-05-13	000000000000	0	0	00000000000	0	0	0	
hEKTxdawhtCyj...	0	0	1	0	[null]	7	2016-10-05	000000000000	1	0	00000000000	0	0	0	
OiVNvaxupAlCls...	0	0	1	0	[null]	7	2016-04-27	307692307692	1	0	92307692308	0	0	0	
/H5dO8s1MVaM...	0	0	1	0	[null]	7	2014-01-06	000000000000	1	0	00000000000	0	0	0	
FXi6xdHersnJ4T...	0	1	14	37	female	9	2005-01-13	000000000000	1	0	00000000000	1	0	0	
woO/HSrEzczBv...	0	1	11	56	female	9	2011-06-07	000000000000	1	0	00000000000	1	0	0	
Yi/GJ+X2d6Oso...	0	0	1	0	[null]	7	2017-01-21	000000000000	1	0	00000000000	0	0	0	
hVW0DdFTOcY5...	0	0	1	0	[null]	7	2011-02-08	000000000000	1	0	00000000000	0	0	0	
Dpt60Q625pb6...	0	0	[null]	[null]	[null]	[null]	[null]	000000000000	1	0	00000000000	0	0	0	
PIikabXeKbKnL...	0	1	4	30	male	9	2011-01-16	000000000000	1	0	66666666667	1	0	1	

---------- EDA and Preprocessing ----------

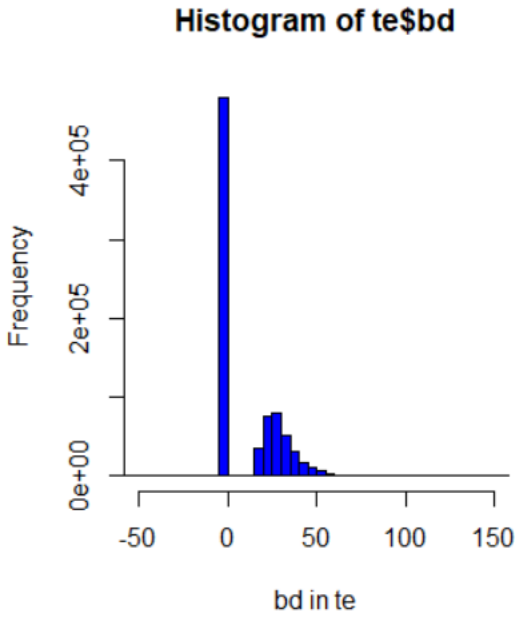
Q2 [5 points]: Write code in either R that compares the distribution of each feature in the te_clean and te tables to see how well the model-based imputation approach worked. Provide figures of these distributions with statistical summaries. You do not need to provide a screenshot of your codes. If there are any features that you believe are not representative of the original features (pre-imputation), indicate those. Do this in Python as well to earn 0.5 extra credit.

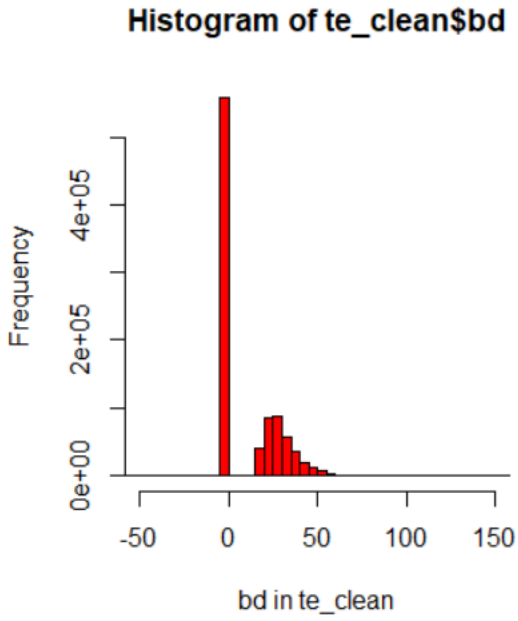
Ans)

Statistical Summaries:

bd

te$bd						
Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	NA's
-3152.00	0.00	0.00	11.94	26.00	2016.00	112381
te_clean$bd						
Min.	1st Qu.	Median	Mean	3rd Qu.	Max.	
-3152.0	0.0	0.0	11.6	25.0	2016.0	

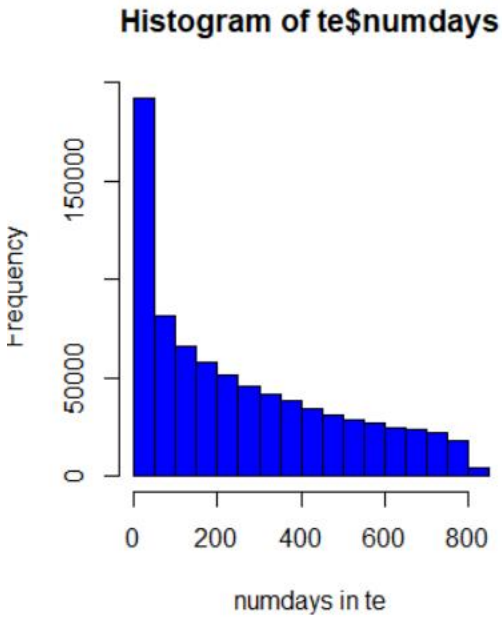


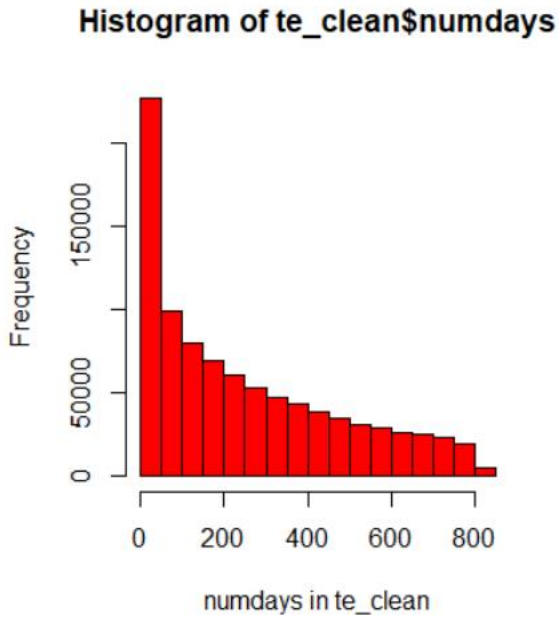


Numdays:

summary(te$numdays)

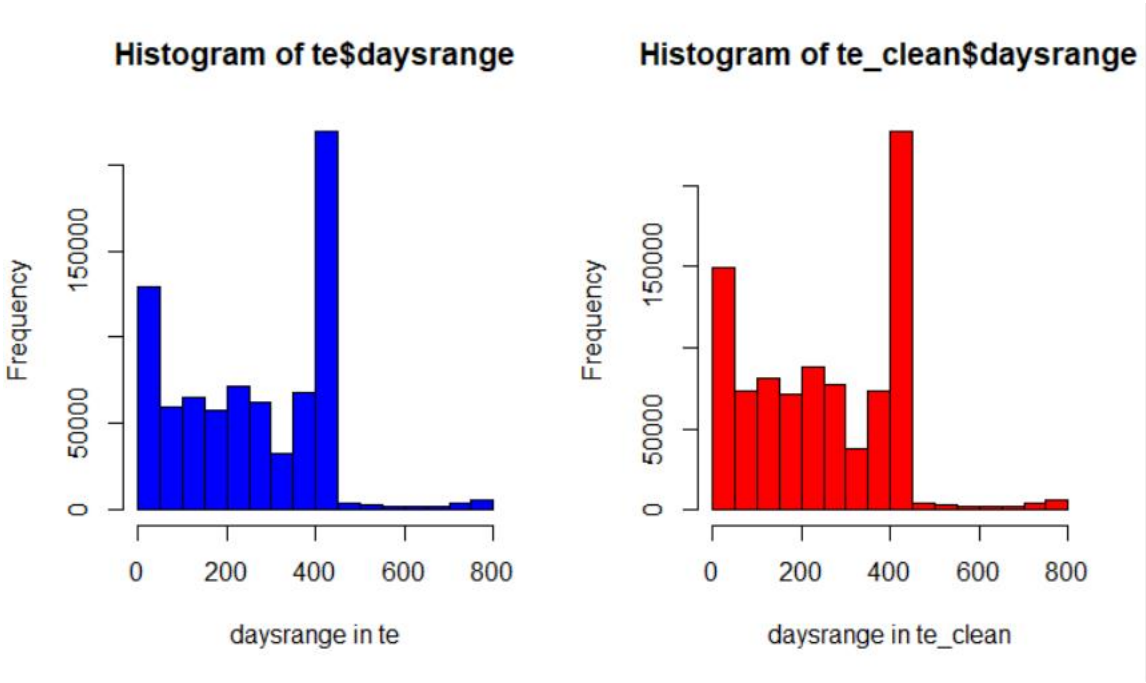
```
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
    1.0    53.0   197.0   259.9   425.0   821.0  120055
> summary(te_clean$numdays)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
      1      51     185     250     404     821
```





Daysrange:

```
> summary(te$daysrange)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
    0.0   107.0   256.5   252.0   406.0   789.0  120055
> summary(te_clean$daysrange)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.
    0.0   103.0   245.5   245.4   404.5   789.0
```



Num25:

```
> summary(te$num25)
   Min. 1st Qu.  Median    Mean 3rd Qu.    Max.    NA's
   0.00    2.34    4.05    5.78    7.00 1560.00  120055
> summary(te_clean$num25)
    Min.  1st Qu.   Median     Mean  3rd Qu.     Max.
   0.000    2.340    4.061    5.838    7.037 1560.000
```

Q4 [1 point]: Provide one positive and one negative aspect of using the mean or median to impute missing values.

Positive:

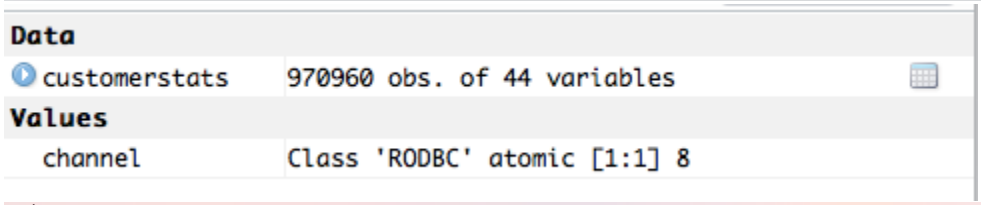
The data computation takes less time, it is easy to understand, and it eliminates the possibility of an outlier.

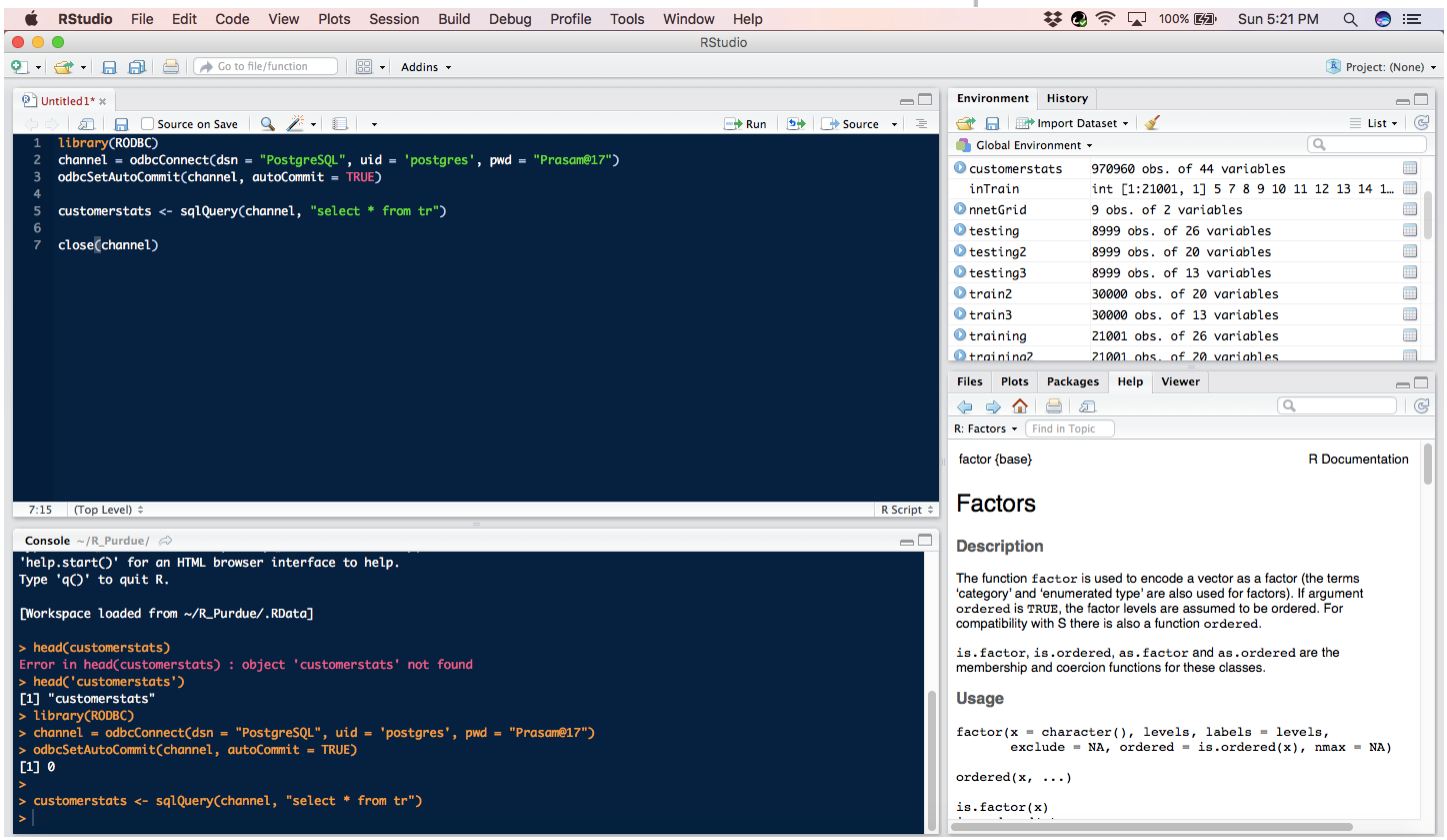
Negative:

The confidence intervals will be overly optimistic, as measures of spread will be artificially reduced.

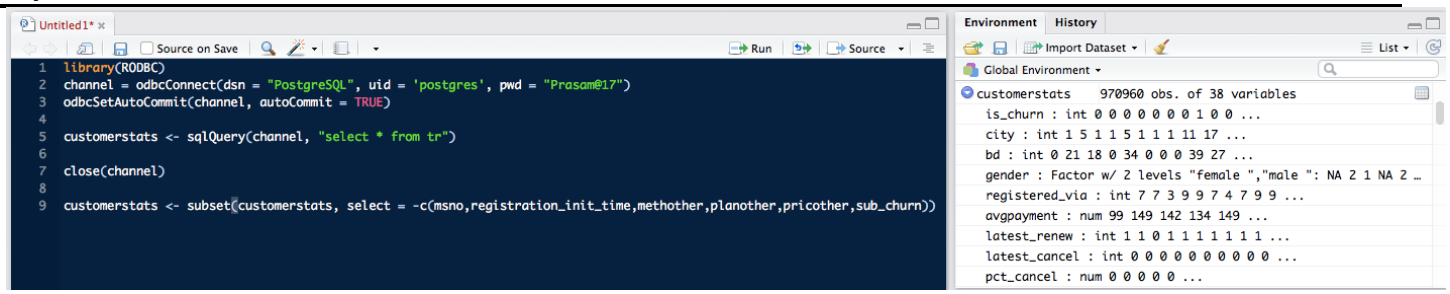
Q5 [1 point]: Pull in the tr data into R from your database using code. Provide a screenshot of your code. Do this in Python as well to earn 0.5 extra credit.

```
> head(customerstats)
                                        msno is_churn sub_churn city bd gender registered_via registration_init_time
1 xyBWyUoXBy2vfwuM9ByKG+hzn63fmWiI7JTFQr0rZ0w=        0         0    1  0   <NA>             7              2013-07-20
2 p0jwdQ20TPCoIm6M5N2K0MdZBwkYOJobmsHGJKvfLJQ=        0         0   NA NA   <NA>            NA                    <NA>
3 h5k7ADY4KxssRzNFEyBNR5CQAiryJ2ROnvsJdxm5pOA=        0         0    1  0   <NA>             7              2016-01-17
4 c4wBaT0G4W9dwXFtx52npGpx7tM25x7/Mbct0I3S9Bg=        0         0    1  0   <NA>             7              2013-09-01
5 rmlTEwuhSXljc2hrTwbr3jhzxAW6i+CnDFXREhBDx3A=        0         0    1  0   <NA>             7              2015-09-09
6 5yh7kC2yZfBoL4DYLZ3e+5+11WKl9UaurHVvCeCSNuw=        0         0    1  0   <NA>             7              2016-11-26
  avgpayment latest_renew latest_cancel pct_cancel methother meth36 meth37 meth38 meth39 meth40 meth41 plan410 plan195
1         99            1             0          0         0      0      0      0      0      0      1       0       0
2         99            1             0          0         0      0      0      0      0      0      1       0       0
3         99            1             0          0         0      0      0      0      0      0      1       0       0
4         99            1             0          0         0      0      0      0      0      0      1       0       0
5         99            1             0          0         0      0      0      0      0      0      1       0       0
6         99            1             0          0         0      0      0      0      0      0      1       0       0
  plan7 plan31 plan0 plan30 planother pric149 pric99 pric0 pric129 pric180 pric150 pricother numdays daysrange  num25
1     0      0     0      1         1       0      1     0       0       0       0         1      24     292.0 2.0000
2     0      0     0      1         1       0      1     0       0       0       0         1      NA       NA     NA
3     0      0     0      1         1       0      1     0       0       0       0         1     348     217.5 4.1841
4     0      0     0      1         1       0      1     0       0       0       0         1      13     205.5 3.6111
5     0      0     0      1         1       0      1     0       0       0       0         1      47     263.0 4.1778
6     0      0     0      1         1       0      1     0       0       0       0         1      54      60.5 4.2560
```





Q6 [1 point]: Remove the msno, registration_init_time, methother, planother, pricother, and sub_churn features. Show your code.

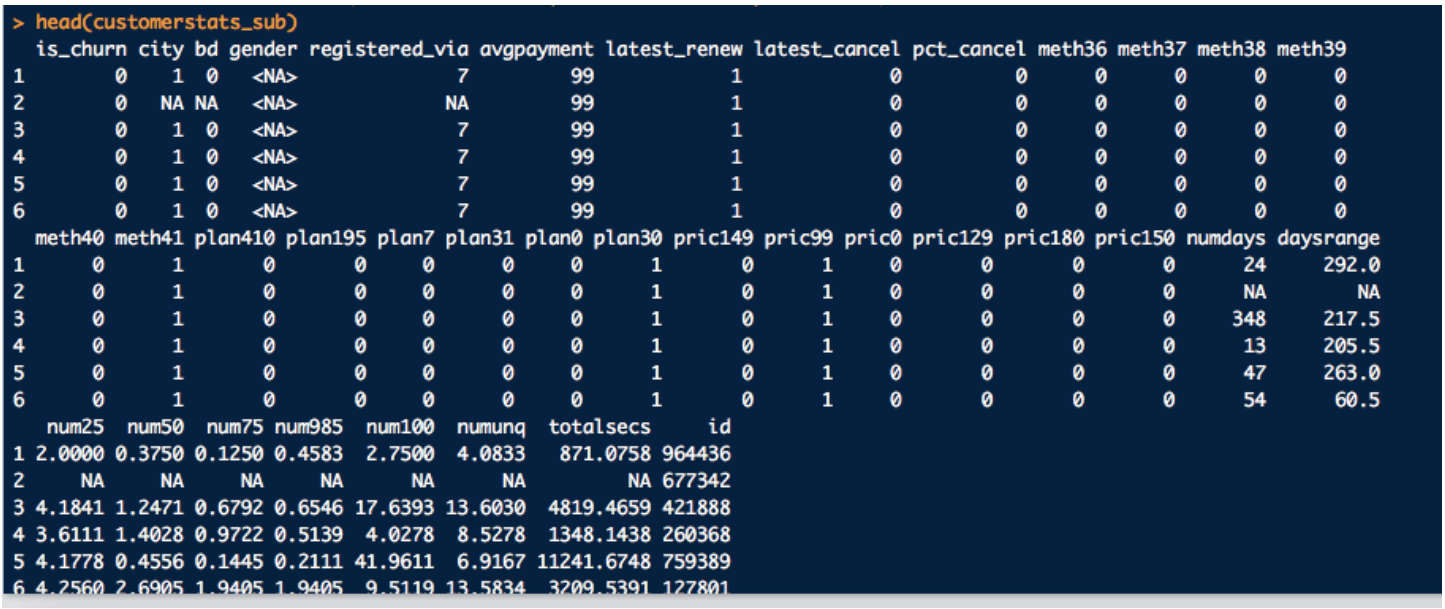


```
library(RODBC)
channel = odbcConnect(dsn = "PostgreSQL", uid = 'postgres', pwd = "Prasam@17")
odbcSetAutoCommit(channel, autoCommit = TRUE)

customerstats <- sqlQuery(channel, "select * from tr")

close(channel)

customerstats <- subset(customerstats, select = -c(msno,registration_init_time,methother,planother,pricother,sub_churn))
```

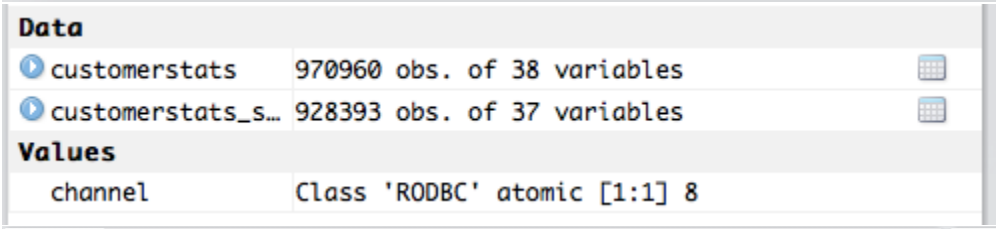
```
customerstats    970960 obs. of 38 variables
 is_churn : int 0 0 0 0 0 0 0 0 1 0 0 ...
 city : int 1 5 1 1 5 1 1 1 11 17 ...
 bd : int 0 21 18 0 34 0 0 0 39 27 ...
 gender : Factor w/ 2 levels "female ","male ": NA 2 1 NA 2 ...
 registered_via : int 7 7 3 9 9 7 4 7 9 9 ...
 avgpayment : num 99 149 142 134 149 ...
 latest_renew : int 1 1 0 1 1 1 1 1 1 1 ...
 latest_cancel : int 0 0 0 0 0 0 0 0 0 0 ...
 pct_cancel : num 0 0 0 0 0 ...
```

```
> head(customerstats)
  is_churn city bd gender registered_via avgpayment latest_renew latest_cancel pct_cancel meth36 meth37 meth38 meth39
1        0    1  0   <NA>              7         99            1             0          0      0      0      0      0
2        0   NA NA   <NA>             NA         99            1             0          0      0      0      0      0
3        0    1  0   <NA>              7         99            1             0          0      0      0      0      0
4        0    1  0   <NA>              7         99            1             0          0      0      0      0      0
5        0    1  0   <NA>              7         99            1             0          0      0      0      0      0
6        0    1  0   <NA>              7         99            1             0          0      0      0      0      0
  meth40 meth41 plan410 plan195 plan7 plan31 plan0 plan30 pric149 pric99 pric0 pric129 pric180 pric150 numdays daysrange
1      0      1       0       0     0      0     0      1       0      1     0       0       0       0      24     292.0
2      0      1       0       0     0      0     0      1       0      1     0       0       0       0      NA        NA
3      0      1       0       0     0      0     0      1       0      1     0       0       0       0     348     217.5
4      0      1       0       0     0      0     0      1       0      1     0       0       0       0      13     205.5
5      0      1       0       0     0      0     0      1       0      1     0       0       0       0      47     263.0
6      0      1       0       0     0      0     0      1       0      1     0       0       0       0      54      60.5
   num25  num50  num75 num985  num100  numunq  totalsecs useforpredictions     id
1 2.0000 0.3750 0.1250 0.4583  2.7500  4.0833   871.0758             keep 964436
2     NA     NA     NA     NA      NA      NA         NA             keep 677342
3 4.1841 1.2471 0.6792 0.6546 17.6393 13.6030  4819.4659             keep 421888
4 3.6111 1.4028 0.9722 0.5139  4.0278  8.5278  1348.1438             keep 260368
5 4.1778 0.4556 0.1445 0.2111 41.9611  6.9167 11241.6748             keep 759389
```

Q7 [1 point]: Subset your tr dataset so that you only keep records that have the “keep” flag in the useforpredictions column. Then delete this column. Show your code.



```
> head(customerstats_sub)
  is_churn city bd gender registered_via avgpayment latest_renew latest_cancel pct_cancel meth36 meth37 meth38 meth39
1        0    1  0   <NA>              7         99            1             0          0      0      0      0      0
2        0   NA NA   <NA>             NA         99            1             0          0      0      0      0      0
3        0    1  0   <NA>              7         99            1             0          0      0      0      0      0
4        0    1  0   <NA>              7         99            1             0          0      0      0      0      0
5        0    1  0   <NA>              7         99            1             0          0      0      0      0      0
6        0    1  0   <NA>              7         99            1             0          0      0      0      0      0
  meth40 meth41 plan410 plan195 plan7 plan31 plan0 plan30 pric149 pric99 pric0 pric129 pric180 pric150 numdays daysrange
1      0      1       0       0     0      0     0      1       0      1     0       0       0       0      24     292.0
2      0      1       0       0     0      0     0      1       0      1     0       0       0       0      NA        NA
3      0      1       0       0     0      0     0      1       0      1     0       0       0       0     348     217.5
4      0      1       0       0     0      0     0      1       0      1     0       0       0       0      13     205.5
5      0      1       0       0     0      0     0      1       0      1     0       0       0       0      47     263.0
6      0      1       0       0     0      0     0      1       0      1     0       0       0       0      54      60.5
   num25  num50  num75 num985  num100  numunq  totalsecs     id
1 2.0000 0.3750 0.1250 0.4583  2.7500  4.0833   871.0758 964436
2     NA     NA     NA     NA      NA      NA         NA 677342
3 4.1841 1.2471 0.6792 0.6546 17.6393 13.6030  4819.4659 421888
4 3.6111 1.4028 0.9722 0.5139  4.0278  8.5278  1348.1438 260368
5 4.1778 0.4556 0.1445 0.2111 41.9611  6.9167 11241.6748 759389
6 4.2560 2.6905 1.9405 1.9405  9.5119 13.5834  3209.5391 127801
```

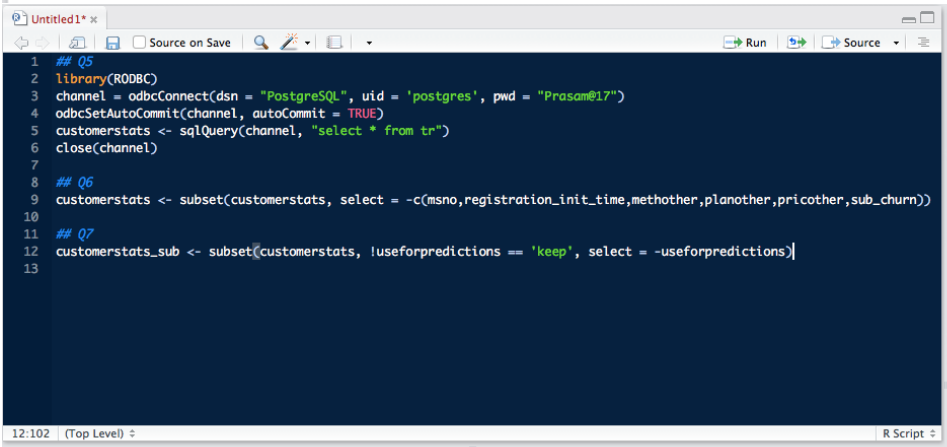


Data

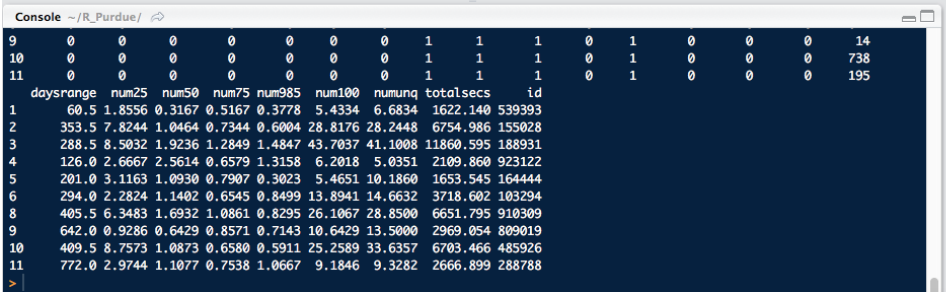
customerstats	970960 obs. of 38 variables
customerstats_s…	928393 obs. of 37 variables

Values

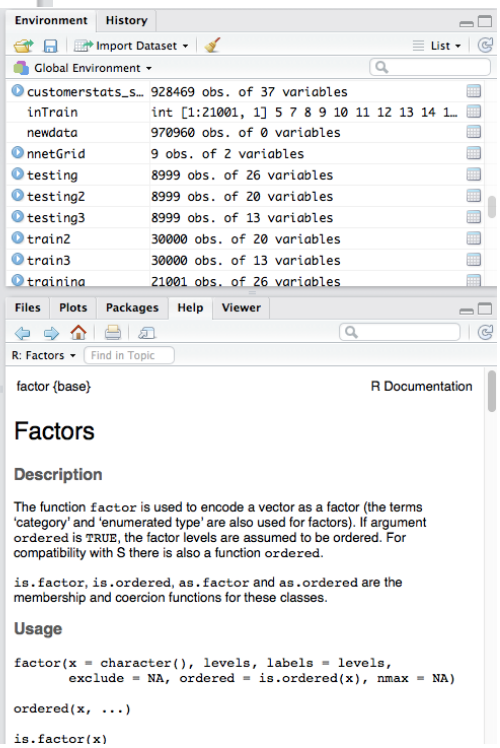
channel	Class 'RODBC' atomic [1:1] 8



```
Untitled1*
Source on Save | Run | Source
1  ## Q5
2  library(RODBC)
3  channel = odbcConnect(dsn = "PostgreSQL", uid = 'postgres', pwd = "Prasam@17")
4  odbcSetAutoCommit(channel, autoCommit = TRUE)
5  customerstats <- sqlQuery(channel, "select * from tr")
6  close(channel)
7
8  ## Q6
9  customerstats <- subset(customerstats, select = -c(msno,registration_init_time,methother,planother,pricother,sub_churn))
10
11 ## Q7
12 customerstats_sub <- subset(customerstats, !useforpredictions == 'keep', select = -useforpredictions)
13
12:102 (Top Level)                                                                                    R Script
```

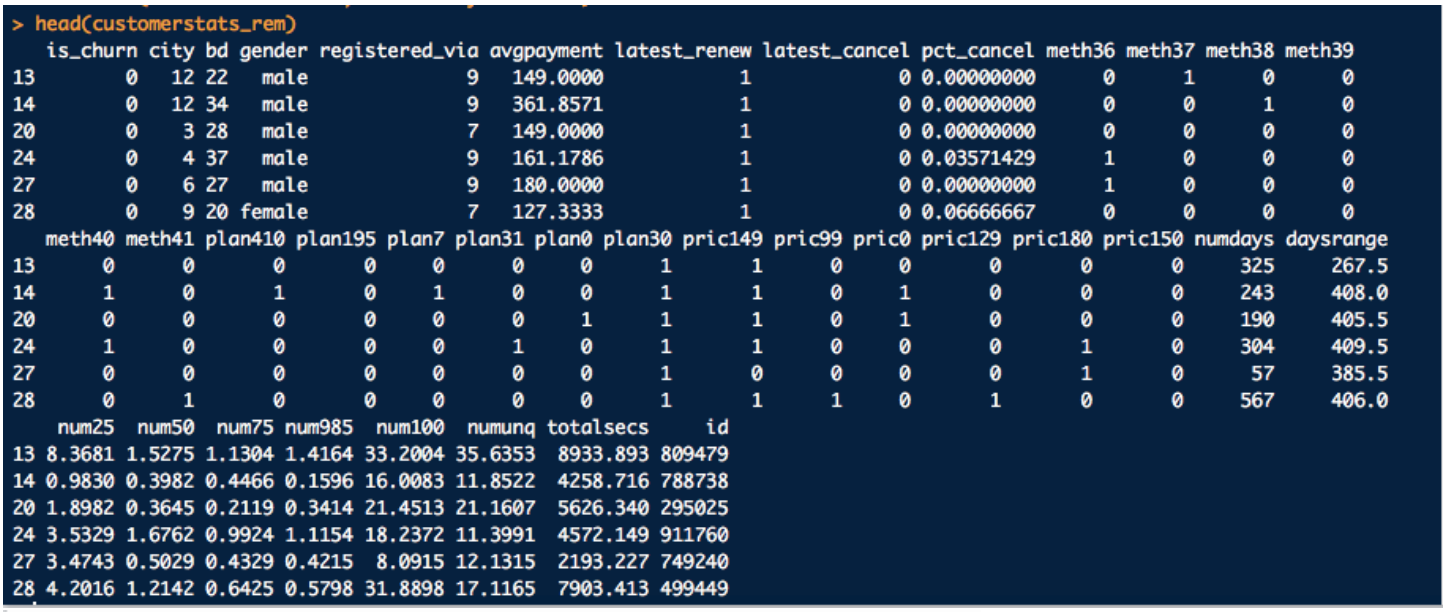


```
Console ~/R_Purdue/
9      0     0     0     0     0     0     0     1     1     1     0     1     0     0     0     14
10     0     0     0     0     0     0     0     1     1     1     0     1     0     0     0    738
11     0     0     0     0     0     0     0     1     1     1     0     1     0     0     0    195
   daysrange  num25  num50  num75 num985  num100  numunq totalsecs     id
1       60.5 1.8556 0.3167 0.5167 0.3778  5.4334  6.6834  1622.140 539393
2      353.5 7.8244 1.0464 0.7344 0.6004 28.8176 28.2448  6754.986 155028
3      288.5 8.5032 1.9236 1.2849 1.4847 43.7037 41.1008 11860.595 188931
4      126.0 2.6667 2.5614 0.6579 1.3158  6.2018  5.0351  2109.860 923122
5      201.0 3.1163 1.0930 0.7907 0.3023  5.4651 10.1860  1653.545 164444
6      294.0 2.2824 1.1402 0.6545 0.8499 13.8941 14.6632  3718.602 103294
8      405.5 6.3483 1.6932 1.0861 0.8295 26.1067 28.8500  6651.795 910309
9      642.0 0.9286 0.6429 0.8571 0.7143 10.6429 13.5000  2969.054 809019
10     409.5 8.7573 1.0873 0.6580 0.5911 25.2589 33.6357  6703.466 485926
11     772.0 2.9744 1.1077 0.7538 1.0667  9.1846  9.3282  2666.899 288788
>
```



```
Environment | History
Import Dataset | List
Global Environment
customerstats_s… 928469 obs. of 37 variables
inTrain          int [1:21001, 1] 5 7 8 9 10 11 12 13 14 1…
newdata          970960 obs. of 0 variables
nnetGrid         9 obs. of 2 variables
testing          8999 obs. of 26 variables
testing2         8999 obs. of 20 variables
testing3         8999 obs. of 13 variables
train2           30000 obs. of 20 variables
train3           30000 obs. of 13 variables
training         21001 obs. of 26 variables

Files | Plots | Packages | Help | Viewer
R: Factors | Find in Topic
factor {base}                                                    R Documentation

Factors

Description

The function factor is used to encode a vector as a factor (the terms 'category' and 'enumerated type' are also used for factors). If argument ordered is TRUE, the factor levels are assumed to be ordered. For compatibility with S there is also a function ordered.

is.factor, is.ordered, as.factor and as.ordered are the membership and coercion functions for these classes.

Usage

factor(x = character(), levels, labels = levels,
       exclude = NA, ordered = is.ordered(x), nmax = NA)

ordered(x, ...)

is.factor(x)
```

Q8 [1 point]: Now remove all records in tr dataset that having any missing values. Show your code.



```
> head(customerstats_rem)
   is_churn city bd gender registered_via avgpayment latest_renew latest_cancel pct_cancel meth36 meth37 meth38 meth39
13        0   12 22   male             9   149.0000           1             0 0.00000000      0      1      0      0
14        0   12 34   male             9   361.8571           1             0 0.00000000      0      0      1      0
20        0    3 28   male             7   149.0000           1             0 0.00000000      0      0      0      0
24        0    4 37   male             9   161.1786           1             0 0.03571429      1      0      0      0
27        0    6 27   male             9   180.0000           1             0 0.00000000      1      0      0      0
28        0    9 20 female             7   127.3333           1             0 0.06666667      0      0      0      0
   meth40 meth41 plan410 plan195 plan7 plan31 plan0 plan30 pric149 pric99 pric0 pric129 pric180 pric150 numdays daysrange
13      0      0       0       0     0      0     0      1       1      0     0       0       0       0     325     267.5
14      1      0       1       0     1      0     0      1       1      0     1       0       0       0     243     408.0
20      0      0       0       0     0      0     1      1       1      0     1       0       0       0     190     405.5
24      1      0       0       0     0      1     0      1       1      0     0       0       1       0     304     409.5
27      0      0       0       0     0      0     0      1       0      0     0       0       1       0      57     385.5
28      0      1       0       0     0      0     0      1       1      1     0       1       0       0     567     406.0
    num25  num50  num75 num985  num100  numunq totalsecs     id
13 8.3681 1.5275 1.1304 1.4164 33.2004 35.6353  8933.893 809479
14 0.9830 0.3982 0.4466 0.1596 16.0083 11.8522  4258.716 788738
20 1.8982 0.3645 0.2119 0.3414 21.4513 21.1607  5626.340 295025
24 3.5329 1.6762 0.9924 1.1154 18.2372 11.3991  4572.149 911760
27 3.4743 0.5029 0.4329 0.4215  8.0915 12.1315  2193.227 749240
28 4.2016 1.2142 0.6425 0.5798 31.8898 17.1165  7903.413 499449
```

Data

customerstats	970960 obs. of 38 variables
customerstats_r…	264260 obs. of 37 variables
customerstats_s…	928393 obs. of 37 variables

Values

channel	Class 'RODBC' atomic [1:1] 8

```
1  ## Q5
2  library(RODBC)
3  channel = odbcConnect(dsn = "PostgreSQL", uid = 'postgres', pwd = "Prasam@17")
4  odbcSetAutoCommit(channel, autoCommit = TRUE)
5  customerstats <- sqlQuery(channel, "select * from tr")
6  close(channel)
7
8  ## Q6
9  customerstats <- subset(customerstats, select = -c(msno,registration_init_time,methother,planother,pricother,sub_churn))
10
11 ## Q7
12 customerstats_sub <- subset(customerstats, !useforpredictions == 'keep', select = -useforpredictions)
13
14 ## Q8
15 customerstats_rem <- na.omit(customerstats_sub)
16
```

Environment	History
customerstats_r…	264268 obs. of 37 variables
customerstats_s…	928469 obs. of 37 variables
inTrain	int [1:21001, 1] 5 7 8 9 10 11 12 13 14 1…
newdata	970960 obs. of 0 variables
nnetGrid	9 obs. of 2 variables
testing	8999 obs. of 26 variables
testing2	8999 obs. of 20 variables
testing3	8999 obs. of 13 variables
train2	30000 obs. of 20 variables
train3	30000 obs. of 13 variables

```
> customerstats_rem <- na.omit(customerstats_sub)
> head(customerstats_rem)
   is_churn city bd     gender registered_via avgpayment latest_renew latest_cancel pct_cancel meth36 meth37 meth38
2         0    5 21 male                   7     149.00           1             0          0      0      0      0
3         0    1 18 female                 3     141.55           0             0          0      0      0      1
5         0    5 34 male                   9     149.00           1             0          0      0      0      0
9         0   11 39 male                   9     149.00           1             0          0      0      0      0
10        0   17 27 male                   9     149.00           1             0          0      0      0      0
11        0    9 24 male                   9     149.00           1             0          0      0      0      0
   meth39 meth40 meth41 plan410 plan195 plan7 plan31 plan0 plan30 pric149 pric99 pric0 pric129 pric180 pric150 numdays
2       0      0      1       0       0     0      0     0      1       1      0     0       0       0       0     604
3       0      0      0       0       0     1      0     0      1       1      0     1       0       0       0     566
5       0      1      0       0       0     0      1     1      1       1      0     1       0       0       0      43
9       0      0      0       0       0     0      0     1      1       1      0     1       0       0       0      14
10      0      0      0       0       0     0      0     1      1       1      0     1       0       0       0     738
```

factor {base} R Documentation

Factors

Description

The function `factor` is used to encode a vector as a factor (the terms 'category' and 'enumerated type' are also used for factors). If argument `ordered` is `TRUE`, the factor levels are assumed to be ordered. For compatibility with S there is also a function `ordered`.

`is.factor`, `is.ordered`, `as.factor` and `as.ordered` are the membership and coercion functions for these classes.

Usage

```
factor(x = character(), levels, labels = levels,
       exclude = NA, ordered = is.ordered(x), nmax = NA)

ordered(x, ...)

is.factor(x)
```

Q9 [1 point]: Make sure the following features are set up as factors: gender, is_churn, city, registered_via. You might need to coerce them. If they are defined properly, just prove their datatype. Show your code.

```
> is.factor(customerstats_rem$gender)
[1] TRUE
> is.factor(customerstats_rem$is_churn)
[1] FALSE
> is.factor(customerstats_rem$city)
[1] FALSE
> is.factor(customerstats_rem$registered_via)
[1] FALSE
> customerstats_rem$is_churn = as.factor(customerstats_rem$is_churn)
> customerstats_rem$city = as.factor(customerstats_rem$city)
> customerstats_rem$registered_via = as.factor(customerstats_rem$registered_via)
> is.factor(customerstats_rem$is_churn)
[1] TRUE
> is.factor(customerstats_rem$city)
[1] TRUE
> is.factor(customerstats_rem$registered_via)
[1] TRUE
>
```

Untitled1* ×

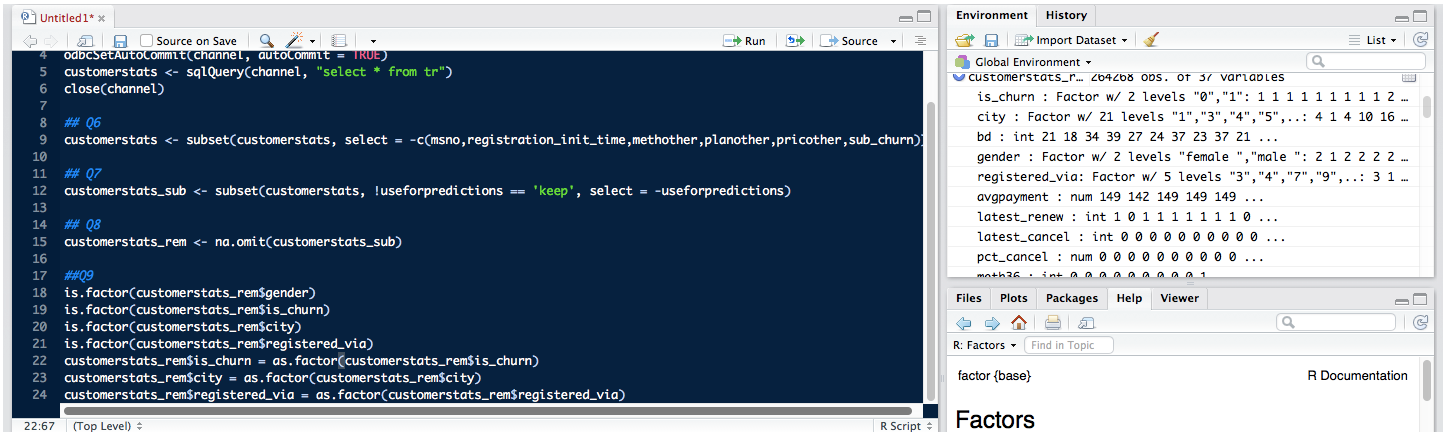
Source on Save | Run | Source

```
2  library(RODBC)
3  channel = odbcConnect(dsn = "PostgreSQL", uid = 'postgres', pwd = "Prasam@17")
4  odbcSetAutoCommit(channel, autoCommit = TRUE)
5  customerstats <- sqlQuery(channel, "select * from tr")
6  close(channel)
7
8  ## Q6
9  customerstats <- subset(customerstats, select = -c(msno,registration_init_time,methother,planother,pricother,sub_churn))
10
11 ## Q7
12 customerstats_sub <- subset(customerstats, !useforpredictions == 'keep', select = -useforpredictions)
13
14 ## Q8
15 customerstats_rem <- na.omit(customerstats_sub)
16
17 ##Q9
18 is.factor(customerstats$gender)
19 is.factor(customerstats$is_churn)
20 is.factor(customerstats$city)
21 is.factor(customerstats$registered_via)
22
```

18:1 (Top Level) R Script

Console ~/R_Purdue/

```
> is.factor(gender)
Error in is.factor(gender) : object 'gender' not found
> is.factor(customerstats$gender)
[1] TRUE
> is.factor(c(customerstats$gender, customerstats$is_churn, customerstats$city, customerstats$registered_via))
[1] FALSE
> is.factor(customerstats$gender)
[1] TRUE
> is.factor(customerstats$is_churn)
[1] FALSE
> is.factor(customerstats$city)
[1] FALSE
> is.factor(customerstats$registered_via)
[1] FALSE
>
```



Q10 [1 point]: Create dummy variables for all categorical features. This can be done easily using the **dummyVars()** function in R, but you may use whatever function you choose. Show your R code. Do this in Python as well to earn 0.5 extra credit.

```
> head(dumb)
   customerstats_rem$is_churn city1 city3 city4 city5 city6 city7 city8 city9 city10 city11 city12 city13 city14 city15
13                          0     0     0     0     0     0     0     0     0      0      0      1      0      0      0
14                          0     0     0     0     0     0     0     0     0      0      0      1      0      0      0
20                          0     0     1     0     0     0     0     0     0      0      0      0      0      0      0
24                          0     0     0     1     0     0     0     0     0      0      0      0      0      0      0
27                          0     0     0     0     0     1     0     0     0      0      0      0      0      0      0
28                          0     0     0     0     0     0     0     0     1      0      0      0      0      0      0
   city16 city17 city18 city19 city20 city21 city22 bd genderfemale gendermale registered_via3 registered_via4
13      0      0      0      0      0      0      0 22            0          1              0              0
14      0      0      0      0      0      0      0 34            0          1              0              0
20      0      0      0      0      0      0      0 28            0          1              0              0
24      0      0      0      0      0      0      0 37            0          1              0              0
27      0      0      0      0      0      0      0 27            0          1              0              0
28      0      0      0      0      0      0      0 20            1          0              0              0
   registered_via7 registered_via9 registered_via13 avgpayment latest_renew latest_cancel pct_cancel meth36 meth37 meth38
13               0               1                0   149.0000           1            0 0.00000000      0      1      0
14               0               1                0   361.8571           1            0 0.00000000      0      0      1
20               1               0                0   149.0000           1            0 0.00000000      0      0      0
24               0               1                0   161.1786           1            0 0.03571429      1      0      0
27               0               1                0   180.0000           1            0 0.00000000      1      0      0
28               1               0                0   127.3333           1            0 0.06666667      0      0      0
```

Data

customerstats	970960 obs. of 38 variables
customerstats_r…	264260 obs. of 37 variables
customerstats_s…	928393 obs. of 37 variables
dumb	264260 obs. of 62 variables

```
25
26  ## Q10
27  library(caret)
28  dmy <- caret::dummyVars("~ .", data=customerstats_rem)
29  cust <- data.frame(predict(dmy, newdata = customerstats_rem))
30
31
```

32:1 (Top Level) R Script

Console ~/R_Purdue/

```
> head(cust)
   is_churn.0 is_churn.1 city.1 city.3 city.4 city.5 city.6 city.7 city.8 city.9 city.10 city.11 city.12 city.13 city.14
2           1          0      0      0      0      0      1      0      0      0      0      0      0      0      0      0
3           1          0      0      1      0      0      0      0      0      0      0      0      0      0      0      0
5           1          0      0      0      0      0      1      0      0      0      0      0      0      0      0      0
9           1          0      0      0      0      0      0      0      0      0      0      0      1      0      0      0
10          1          0      0      0      0      0      0      0      0      0      0      0      0      0      0      0
11          1          0      0      0      0      0      0      0      0      0      1      0      0      0      0      0
   city.15 city.16 city.17 city.18 city.19 city.20 city.21 city.22 bd gender.female.... gender.male......
2        0       0       0       0       0       0       0       0 21                 0                1
3        0       0       0       0       0       0       0       0 18                 1                0
5        0       0       0       0       0       0       0       0 34                 0                1
9        0       0       0       0       0       0       0       0 39                 0                1
10       0       0       0       1       0       0       0       0 27                 0                1
11       0       0       0       0       0       0       0       0 24                 0                1
   registered_via.3 registered_via.4 registered_via.7 registered_via.9 registered_via.13 avgpayment latest_renew
```

Q11 [1 point]: Remove any features that have zero- or near-zero variances using the **nearZeroVar()** function. Show your R code. Do this in Python as well to earn 0.5 extra credit.

```
> head(filteredcust)
   city4 city5 city6 city13 city15 city22 bd genderfemale gendermale registered_via3 registered_via4 registered_via7
13     0     0     0      0      0      0 22            0          1              0              0              0
14     0     0     0      0      0      0 34            0          1              0              0              0
20     0     0     0      0      0      0 28            0          1              0              0              1
24     1     0     0      0      0      0 37            0          1              0              0              0
27     0     0     1      0      0      0 27            0          1              0              0              0
28     0     0     0      0      0      0 20            1          0              0              0              1
   registered_via9 avgpayment latest_renew pct_cancel meth36 meth37 meth38 meth40 meth41 plan7 plan31 plan0 pric149
13               1   149.0000            1 0.00000000      0      1      0      0      0     0      0     0       1
14               1   361.8571            1 0.00000000      0      0      1      1      0     1      0     0       1
20               0   149.0000            1 0.00000000      0      0      0      0      0     0      0     1       1
24               1   161.1786            1 0.03571429      1      0      0      1      0     0      1     0       1
27               1   180.0000            1 0.00000000      1      0      0      0      0     0      0     0       0
28               0   127.3333            1 0.06666667      0      0      0      0      1     0      0     0       1
   pric99 pric0 pric129 pric180 pric150 numdays daysrange  num25  num50  num75 num985  num100  numunq totalsecs     id
13      0     0       0       0       0     325     267.5 8.3681 1.5275 1.1304 1.4164 33.2004 35.6353  8933.893 809479
14      0     1       0       0       0     243     408.0 0.9830 0.3982 0.4466 0.1596 16.0083 11.8522  4258.716 788738
20      0     1       0       0       0     190     405.5 1.8982 0.3645 0.2119 0.3414 21.4513 21.1607  5626.340 295025
24      0     0       0       1       0     304     409.5 3.5329 1.6762 0.9924 1.1154 18.2372 11.3991  4572.149 911760
27      0     0       0       1       0      57     385.5 3.4743 0.5029 0.4329 0.4215  8.0915 12.1315  2193.227 749240
28      1     0       1       0       0     567     406.0 4.2016 1.2142 0.6425 0.5798 31.8898 17.1165  7903.413 499449
```

customerstats	970960 obs. of 38 variables
customerstats_r…	264260 obs. of 37 variables
customerstats_s…	928393 obs. of 37 variables
dumb	264260 obs. of 62 variables
filteredcust	264260 obs. of 40 variables

Assign2.R*

```
14  ## Q8
15  customerstats_rem <- na.omit(customerstats_sub)
16
17  ## Q9
18  is.factor(customerstats_rem$gender)
19  is.factor(customerstats_rem$is_churn)
20  is.factor(customerstats_rem$city)
21  is.factor(customerstats_rem$registered_via)
22  customerstats_rem$is_churn = as.factor(customerstats_rem$is_churn)
23  customerstats_rem$city = as.factor(customerstats_rem$city)
24  customerstats_rem$registered_via = as.factor(customerstats_rem$registered_via)
25
26  ## Q10
27  library(caret)
28  dmy <- caret::dummyVars("~ .", data=customerstats_rem)
29  cust <- data.frame(predict(dmy, newdata = customerstats_rem))
30
31  ## Q11
32  cust_nzero = nearZeroVar(cust)
33  filteredcust <- customerstats_rem[, -cust_nzero]
34
```

Environment | History

```
Global Environment
filteredcust     264268 obs. of 18 variables
registered_via: Factor w/ 5 levels "3","4","7","9",..: 3 1 …
avgpayment : num 149 142 149 149 149 ...
latest_renew : int 1 0 1 1 1 1 1 1 1 0 ...
meth40 : int 0 0 1 0 0 0 1 1 0 0 ...
plan410 : int 0 0 0 0 0 0 0 0 0 0 ...
pric99 : int 0 0 0 0 0 0 0 0 0 0 ...
pric0 : int 0 1 1 1 1 1 1 0 0 0 ...
pric129 : int 0 0 0 0 0 0 0 0 0 0 ...
pric180 : int 0 0 0 0 0 0 0 0 0 1 ...
```

Files | Plots | Packages | Help | Viewer

R: Create A Full Set of Dummy Variables — Find in Topic

dummyVars {caret} R Documentation

Q12 [1 point]: Identify and remove any features having a greater than 90% correlation using the **findCorrelation()** function. Show your R code. Do this in Python as well to earn 0.5 extra credit.

```
35  ## Q12
36  correlations <- cor(filteredcust[,c(11:16)])
37  highCorr <- findCorrelation(correlations, cutoff = .9, names = FALSE)
38  new_list <- filteredcust[, -highCorr]
39
40
```

40:1 (Top Level) R Script

Console ~/R_Purdue/

```
> correlations
            numdays     daysrange       num25       num75      num985       numunq
numdays   1.0000000  0.4906392768  0.12006344  0.16801150  0.15374512  0.2762320809
daysrange 0.4906393  1.0000000000 -0.07158229 -0.04729206 -0.01007986 -0.0004370325
num25     0.1200634 -0.0715822911  1.00000000  0.56817803  0.33588483  0.3583558235
num75     0.1680115 -0.0472920637  0.56817803  1.00000000  0.49863831  0.3280281147
num985    0.1537451 -0.0100798578  0.33588483  0.49863831  1.00000000  0.3137305857
numunq    0.2762321 -0.0004370325  0.35835582  0.32802811  0.31373059  1.0000000000
> highCorr <- findCorrelation(correlations, cutoff = .9, names = FALSE)
> highCorr
integer(0)
>
```

```
> head(filteredcust)
   city4 city5 city6 city13 city15 city22 bd genderfemale registered_via3 registered_via4 registered_via7 registered_via9
13     0     0     0      0      0      0 22            0              0              0              0              1
14     0     0     0      0      0      0 34            0              0              0              0              1
20     0     0     0      0      0      0 28            0              0              0              1              0
24     1     0     0      0      0      0 37            0              0              0              0              1
27     0     0     1      0      0      0 27            0              0              0              0              1
28     0     0     0      0      0      0 20            1              0              0              1              0
   avgpayment latest_renew pct_cancel meth36 meth37 meth38 meth40 plan7 plan31 plan0 pric149 pric99 pric0 pric129 pric180
13   149.0000            1 0.00000000      0      1      0      0     0      0     0       1      0     0       0       0
14   361.8571            1 0.00000000      0      0      1      1     1      0     0       1      0     1       0       0
20   149.0000            1 0.00000000      0      0      0      0     0      0     1       1      0     1       0       0
24   161.1786            1 0.03571429      1      0      0      1     0      1     0       1      0     0       0       1
27   180.0000            1 0.00000000      1      0      0      0     0      0     0       0      0     0       0       1
28   127.3333            1 0.06666667      0      0      0      0     0      0     0       1      1     0       1       0
   pric150 numdays daysrange  num25  num50  num75 num985  num100  numunq totalsecs     id
13       0     325     267.5 8.3681 1.5275 1.1304 1.4164 33.2004 35.6353  8933.893 809479
14       0     243     408.0 0.9830 0.3982 0.4466 0.1596 16.0083 11.8522  4258.716 788738
20       0     190     405.5 1.8982 0.3645 0.2119 0.3414 21.4513 21.1607  5626.340 295025
24       0     304     409.5 3.5329 1.6762 0.9924 1.1154 18.2372 11.3991  4572.149 911760
27       0      57     385.5 3.4743 0.5029 0.4329 0.4215  8.0915 12.1315  2193.227 749240
28       0     567     406.0 4.2016 1.2142 0.6425 0.5798 31.8898 17.1165  7903.413 499449
```

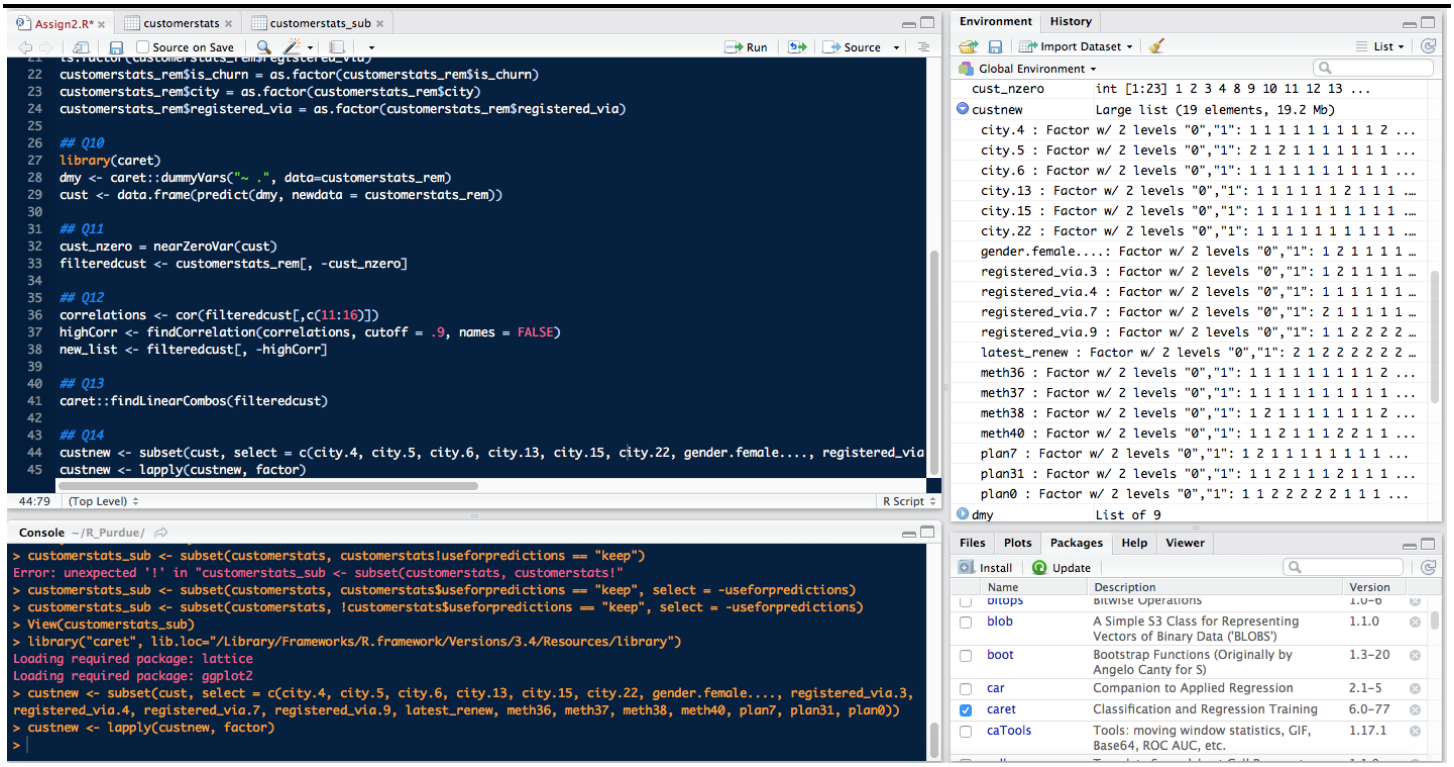
correlations	num [1:40, 1:40] 1 -0.1648 -0.0901 -0.193…
customerstats	970960 obs. of 38 variables
customerstats_r…	264260 obs. of 37 variables
customerstats_s…	928393 obs. of 37 variables
dumb	264260 obs. of 62 variables
filteredcust	264260 obs. of 38 variables
Values	
channel	Class 'RODBC' atomic [1:1] 8
dmy	List of 9
highCorr	int [1:2] 21 9

Q13 [1 point]: Identify and remove any features that are linear combinations of each other using the **findLinearCombos()** function. Show your R code. Do this in Python as well to earn 0.5 extra credit.

```
40  ## Q13
41  caret::findLinearCombos(filteredcust)
42
43
44:1  (Top Level)  R Script

Console ~/R_Purdue/
numunq     0.2762321 -0.0004370325  0.35835582  0.32802811  0.31373059  1.0000000000
> highCorr <- findCorrelation(correlations, cutoff = .9, names = FALSE)
> highCorr
integer(0)
> caret::findLinearCombos(filteredcust)
$linearCombos
list()

$remove
NULL

>
```

```
linearcust        List of 2
    linearCombos: list()
    remove : NULL
```

Q14 [1 point]: Coerce the following features to factors: city4, city5, city6, city13, city15, city22, genderfemale, registered_via3, registered_via4, registered_via7, registered_via9, latest_renew, meth36, meth37, meth38, meth40, plan7, plan31, and plan0. Show your R code. Do this in Python as well to earn 0.5 extra credit.



Q15 [1 point]: Use the **preprocess()** function to center and scale your tr dataset. Then use the **predict()** function to actually transform your variables. Show your R code. Do this in Python as well to earn 0.5 extra credit.

```r
## Q15
# preprocess values by centering and scaling (typical standardization)
preProcValues <- preProcess(filteredcust, method = c("center", "scale"))

# predict() actually transforms the variables
transformed <- predict(preProcValues, filteredcust)
```

Modeling

To answer the following questions, make sure you are using your transformed/standardized tr data from the previous questions.

Q16 [1 point]: Load the caret package and set a seed of 1234. Use make.names() and relevel() on your target variable to ensure the response works for all methods. You might need to define 1s or churners as "X1". Show your R code.

```r
## Q16
library("plyr")
transformed <- cbind(customerstats_rem$is_churn, transformed)
names <- colnames(transformed)
colnames(transformed) <- make.names(names, unique = TRUE)
transformed <- subset(transformed, select = -c(id))
str(transformed)
```

Q17 [1 point]: Use the **createDataPartition()** function to create an 80/20 train and test datasets. Show your R code. Do this in Python as well to earn 0.5 extra credit.

```r
## Q17
set.seed(12346)
# select randomly 80% of the dataset to serve as training data
library(caret)
inTrain <- createDataPartition(y = transformed$city4, times =1, p=0.8, list=FALSE)

# train and test sets having the input features
training <- transformed[inTrain,]
test <- transformed[-inTrain,]
```

Q18 [1 point]: Plot the distribution of your target variable in the train and test set. Show your R code. Do this in Python as well to earn 0.5 extra credit.

```r
## Q18
library(ggplot2)
#training$customerstats_rem.is_churn <- as.numeric(training$customerstats_rem.is_churn)
ggplot(data.frame(training),aes(x=customerstats_rem.is_churn)) + geom_bar()
ggplot(data.frame(test),aes(x=customerstats_rem.is_churn)) + geom_bar()
```

Q19 [1 point]: Use the **trainControl()** function to specify 5-fold cross-validation, as well as anything else that is important to indicate you are doing classification-type modeling. Show your R code.

```r
## Q19
library(rpart)
train_control<- trainControl(method="cv", number=5, savePredictions = TRUE)
```

Q20 [1 point]: Use the **train()** function to build any model you want. Show your R code. Do this in Python as well to earn 0.5 extra credit.

```r
## Q20
model<- train(customerstats_rem.is_churn~., data=training, trControl=train_control, method="glm", family ="binomial")
```

Model Evaluation

Q21 [1 point]: Use the **predict()** function to generate predicted probabilities and predicted classes for both the train and test sets. Show your R code. Do this in Python as well to earn 0.5 extra credit.

```r
## Q21
predict_train<- predict(model,training)
predict_test<- predict(model,test)
```

Q22 [1 point]: Use the **confusionMatrix()** function to evaluate your model. Compare the train and test statistics and describe if you have overfit your model. Show your R code. Do this in Python as well to earn 0.5 extra credit.

```
## Q22
training<- cbind(training,predict_train)
confusionMatrix_train<- confusionMatrix(training$predict_train,training$customerstats_rem.is_churn)
test <- cbind(test,predict_test)
confusionMatrix_test<- confusionMatrix(test$predict_test,test$customerstats_rem.is_churn)
```

Q23 [1 point]: Use the **roc()** function to create an ROC curve on the test set predictions, as well as the **auc()** function to capture the area-under-the-curve statistc. Show your R code. Do this in Python as well to earn 0.5 extra credit.

```
## Q23
library(ROCR)
library(pROC)
training$customerstats_rem.is_churn <- as.numeric(training$customerstats_rem.is_churn)
training$predict_train <- as.numeric(training$predict_train)

auc(training$customerstats_rem.is_churn, training$predict_train)
roc(training$customerstats_rem.is_churn, training$predict_train)

test$customerstats_rem.is_churn <- as.numeric(test$customerstats_rem.is_churn)
test$predict_train <- as.numeric(test$predict_test)

auc(test$customerstats_rem.is_churn, test$predict_test)
roc(test$customerstats_rem.is_churn, test$predict_test)
```

Deployment

Q24 [1 point]: Use the **mgcv** R package and **saveRDS()** function to save your predictive model for use later. Show your R code. Do this in Python as well to earn 0.5 extra credit.

```
library(mgcv)
save(model, file = "mymodel.rda")
saveRDS(model, "rpart.rds")
```

Q25 [1 point]: Pull the te_clean dataset from your database into R. Coerce the factor variables in this dataset exactly like you did in Q14. Show your R code.

```r
## Q25
library(RODBC)
library(caret)
channel = odbcConnect(dsn = "PostgreSQL", uid = 'postgres', pwd = "Prasam@17")
odbcSetAutoCommit(channel, autoCommit = TRUE)
te_clean <- sqlQuery(channel, "select * from te_clean")
close(channel)
te_clmsno <- te_clean
te_clean$city4 <- as.factor(te_clean$city4)
te_clean$city5 <- as.factor(te_clean$city5)
te_clean$city6 <- as.factor(te_clean$city6)
te_clean$city13 <- as.factor(te_clean$city13)
te_clean$city15 <- as.factor(te_clean$city15)
te_clean$city22 <- as.factor(te_clean$city22)
te_clean$genderfemale <- as.factor(te_clean$genderfemale)
te_clean$registered_via3 <- as.factor(te_clean$registered_via3)
te_clean$registered_via4 <- as.factor(te_clean$registered_via4)
te_clean$registered_via7 <- as.factor(te_clean$registered_via7)
te_clean$registered_via9 <- as.factor(te_clean$registered_via9)
te_clean$latest_renew <- as.factor(te_clean$latest_renew)
te_clean$meth36 <- as.factor(te_clean$meth36)
te_clean$meth37 <- as.factor(te_clean$meth37)
te_clean$meth38 <- as.factor(te_clean$meth38)
te_clean$meth40 <- as.factor(te_clean$meth40)
te_clean$plan7 <- as.factor(te_clean$plan7)
te_clean$plan31 <- as.factor(te_clean$plan31)
te_clean$plan0 <- as.factor(te_clean$plan0)
te_clean$is_churn <- as.factor(te_clean$is_churn)
te_clean <- subset(te_clean, select = -c(msno))
te_clean <- subset(te_clean, select = -c(is_churn))
```

Q26 [1 point]: Use the **preprocess()** function to center and scale your te_clean dataset just like you did to your tr dataset in Q15. Then use the **predict()** function to actually transform your variables. Show your R code.

```r
## Q26
preProcValues <- preProcess(te_clean, method = c("center", "scale"))
transformed2 <- predict(preProcValues, te_clean)
```

Q27 [1 point]: Use the **predict()** function to score your predictions on the te_clean dataset. This might take a few minutes given the type of model you used in Q20. For example, k-nearest neighbors (method="knn") will take a long time because of the distance calculations that it will have to make. Show your R code.

```r
## Q27
predict_clean<- predict(model,transformed2)
```

Q28 [1 point]: Use the **write.table()** function to write out the msno and predictions from Q27 and upload those predictions to the Kaggle competition. Show your R code. Also provide a screenshot of your new rank.

```r
## Q28
predict_clean <- as.numeric(predict_clean)
dfp = data.frame(te_clmsno$msno, predict_clean)
#a = cbind(te_clmsno$msno, predict_clean)
write.table(dfp, 'Prasad1.csv', col.names=NA)
```

Applied Projects

Q29 [1 point]: Provide and update on your project and what you have done so far, also provide what you plan to accomplish by November 28th.

We have had our first meeting with the client.

Currently we are reading the data as shared by the client and the 7 steps of buying shared by the client.

We plan to prepare the data to create inventory model.

Sonakshi Malhotra | Prasad Gujela | Shan Lin | Taeyong Kim

PURDUE UNIVERSITY

---------- EDA and Preprocessing ----------

Q2 [5 points]: Write code in either R that compares the distribution of each feature in the te_clean and te tables to see how well the model-based imputation approach worked. Provide figures of these distributions with statistical summaries. You do not need to provide a screenshot of your codes. If there are any features that you believe are not representative of the original features (pre-imputation), indicate those. Do this in Python as well to earn 0.5 extra credit.

Ans)

Python

"bd" column: Statistic summary

```
In [20]: Stats_te_clean = te_clean.describe()
         print Stats_te_clean
```

```
In [21]: Stats_te_clean.iloc[ : , 7]  #column "bd"
```

```
Out[21]: count    907471.000000
         mean         11.598318
         std          18.796884
         min       -3152.000000
         25%           0.000000
         50%           0.000000
         75%          25.000000
         max        2016.000000
         Name: bd, dtype: float64
```

```
In [22]: Stats_te = te.describe()
         print Stats_te
```

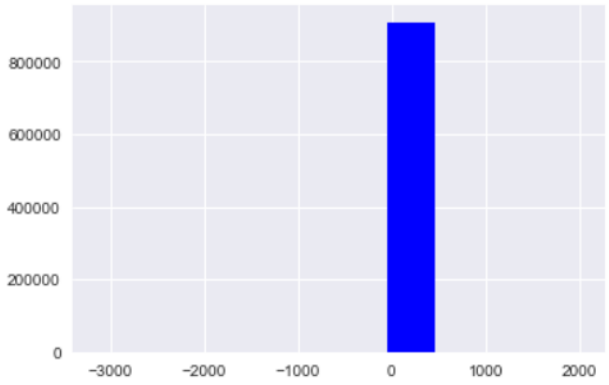
```
In [23]: Stats_te.iloc[ : , 3]  #column "bd"
```

```
Out[23]: count    795090.000000
         mean         11.944989
         std          18.900119
         min       -3152.000000
         25%           0.000000
         50%           0.000000
         75%          26.000000
         max        2016.000000
         Name: bd, dtype: float64
```

```
In [135]: import seaborn as sns
          import matplotlib.pylab as plt
          %matplotlib inline
```

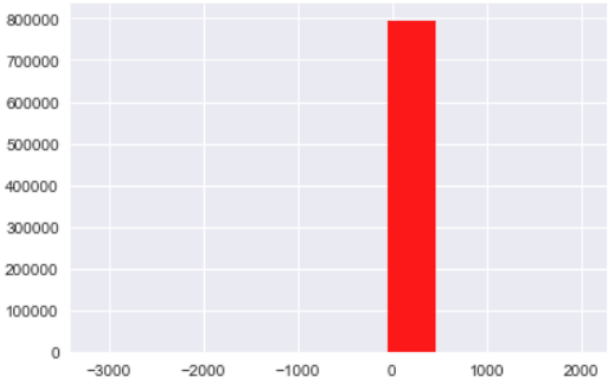
```
In [90]: plt.hist(te_clean.bd, color='blue')
```

```
Out[90]: (array([  1.00000000e+00,   0.00000000e+00,   0.00000000e+00,
                   0.00000000e+00,   0.00000000e+00,   0.00000000e+00,
                   9.07392000e+05,   2.20000000e+01,   5.30000000e+01,
                   3.00000000e+00]),
          array([-3152. , -2635.2, -2118.4, -1601.6, -1084.8,  -568. ,   -51.2,
                   465.6,   982.4,  1499.2,  2016. ]),
          <a list of 10 Patch objects>)
```




```
In [84]: te['bd'].hist(alpha=0.9,color='red')
```

```
Out[84]: <matplotlib.axes._subplots.AxesSubplot at 0x3e5f5400>
```



"Numday" column: Statistic summary

```
In [16]: Stats_te_clean.iloc[ : , 29] #column "numdays"
```

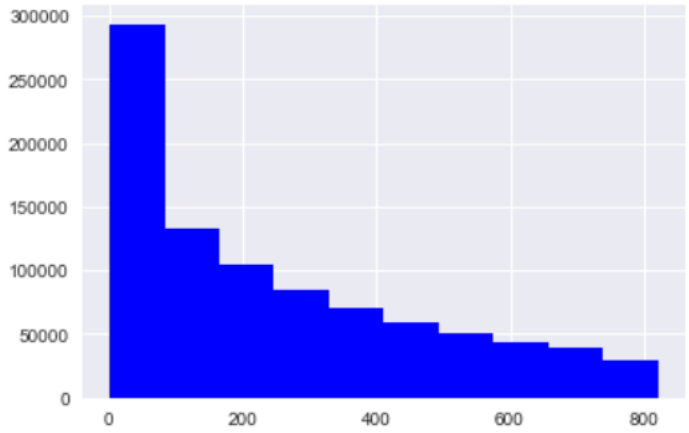
```
Out[16]: count    907471.000000
         mean        249.987905
         std         225.928408
         min           1.000000
         25%          51.000000
         50%         185.000000
         75%         404.000000
         max         821.000000
         Name: numdays, dtype: float64
```

In [18]:
```
Stats_te.iloc[ : , 30]  #column "numdays"
```

Out[18]:
```
count    787416.000000
mean        259.884112
std         230.441958
min           1.000000
25%          53.000000
50%         197.000000
75%         425.000000
max         821.000000
Name: numdays, dtype: float64
```

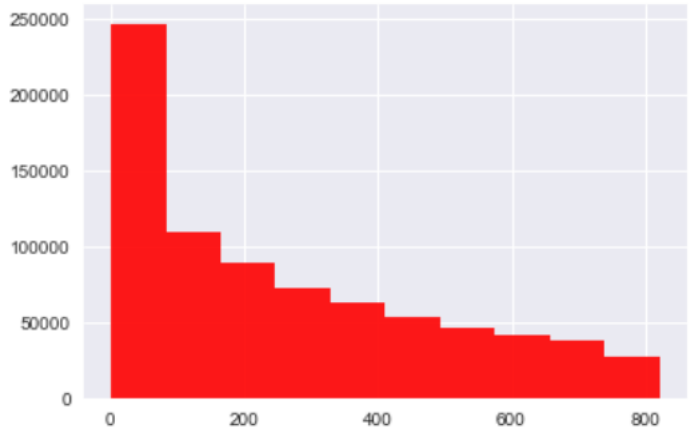
In [89]:
```
plt.hist(te_clean.numdays,color = 'blue')
```

Out[89]:
```
(array([ 293117.,  132537.,  105079.,   84113.,   70766.,   59153.,
          50015.,   43926.,   39974.,   28791.]),
 array([   1.,   83.,  165.,  247.,  329.,  411.,  493.,  575.,  657.,
         739.,  821.]),
 <a list of 10 Patch objects>)
```



In [83]:
```
te['numdays'].hist(alpha=0.9,color='red')
#dfj2_MARKET1['VSPD1_perc'].hist(ax=axes[0], alpha=0.9, color='blue')
```

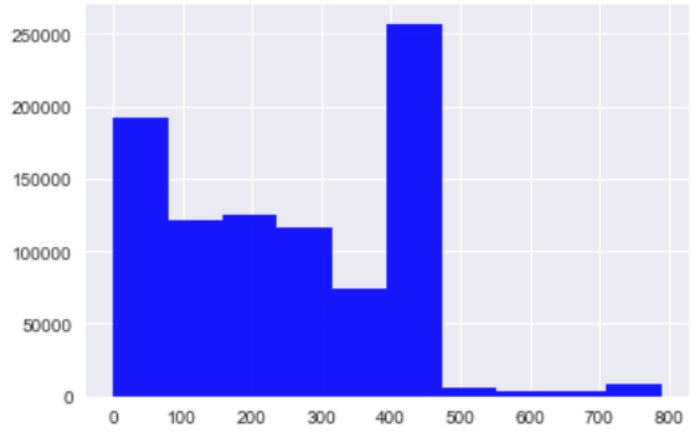
Out[83]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x31cd8080>
```



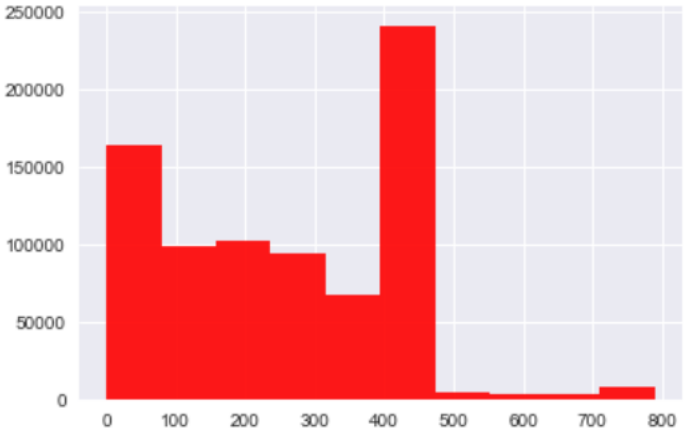
Daysrange: Statistic summary

```
In [21]: Stats_te.iloc[ : , 31]   #column "daysrange"

Out[21]: count    787416.000000
         mean        252.042102
         std         161.020163
         min           0.000000
         25%         107.000000
         50%         256.500000
         75%         406.000000
         max         789.000000
         Name: daysrange, dtype: float64
```

```
In [88]: te_clean['daysrange'].hist(alpha=0.9,color='blue')

Out[88]: <matplotlib.axes._subplots.AxesSubplot at 0x15e60668>
```




```
In [87]: te['daysrange'].hist(alpha=0.9,color='red')

Out[87]: <matplotlib.axes._subplots.AxesSubplot at 0x51a97cf8>
```



Num25: : Statistic summary

In [22]:
```
Stats_te_clean.iloc[ : , 31]  #column "num25"
```

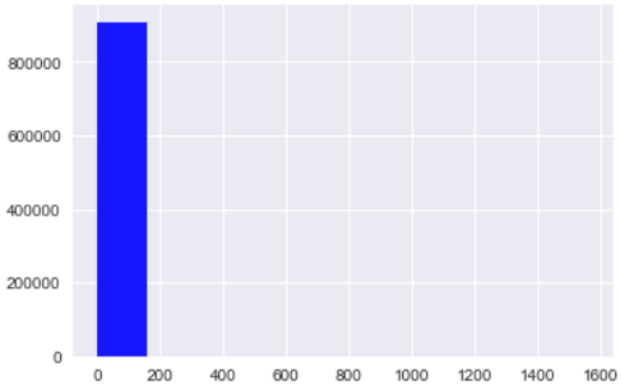
Out[22]:
```
count    907471.000000
mean          5.837622
std           6.896805
min           0.000000
25%           2.339500
50%           4.061100
75%           7.037000
max        1560.313400
Name: num25, dtype: float64
```

In [23]:
```
Stats_te.iloc[ : , 32]   #column "num25"
```

Out[23]:
```
count    787416.000000
mean          5.783492
std           6.745339
min           0.000000
25%           2.336300
50%           4.053600
75%           7.003300
max        1560.313400
Name: num25, dtype: float64
```

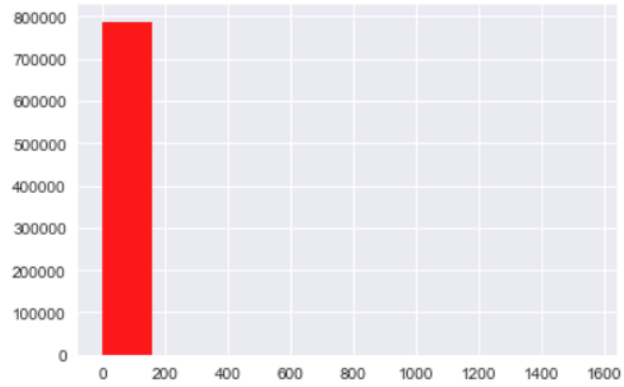
In [92]:
```
te_clean['num25'].hist(alpha=0.9,color='blue')
```

Out[92]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x3f271828>
```



In [93]:
```
te['num25'].hist(alpha=0.9,color='red')
```

Out[93]:
```
<matplotlib.axes._subplots.AxesSubplot at 0x2b537780>
```



Q5 [1 point]: Pull in the tr data into R from your database using code. Provide a screenshot of your code. Do this in Python as well to earn 0.5 extra credit.

Python:

Q5. connect Python to postgres

check this website for code: https://www.youtube.com/watch?v=NYcGqiQMws4

```
In [2]: # install this packages in the command prompt of Canopy
        # pip install psycopg2
        import psycopg2 as p

In [3]: conn = p.connect(dbname = 'postgres', user='postgres', host='localhost', password='Sunny723')

In [4]: cur = conn.cursor()
```

Q6 [1 point]: Remove the msno, registration_init_time, methother, planother, pricother, and sub_churn features. Show your code.

Python:

Q6: remove some columns, done

```
In [138]: tr_new = tr.drop(tr.columns[[0, 2, 7, 12, 25, 32]], axis = 1)
          print tr_new.head()
```

...

```
In [139]: tr_new.columns.values

Out[139]: array(['is_churn', 'city', 'bd', 'gender', 'registered_via', 'avgpayment',
                 'latest_renew', 'latest_cancel', 'pct_cancel', 'meth36', 'meth37',
                 'meth38', 'meth39', 'meth40', 'meth41', 'plan410', 'plan195',
                 'plan7', 'plan31', 'plan0', 'plan30', 'pric149', 'pric99', 'pric0',
                 'pric129', 'pric180', 'pric150', 'numdays', 'daysrange', 'num25',
                 'num50', 'num75', 'num985', 'num100', 'numunq', 'totalsecs',
                 'useforpredictions', 'id'], dtype=object)
```

Q7 [1 point]: Subset your tr dataset so that you only keep records that have the "keep" flag in the useforpredictions column. Then delete this column. Show your code.

Python:

Q7: keep a part of rows, done

```
In [18]: tr_new['useforpredictions'][0:10]
```

```
In [140]: tr_new = tr_new[tr_new['useforpredictions'] == 'keep']
          print tr_new['useforpredictions'][0:10]
```

```
0     keep
1     keep
2     keep
3     keep
4     keep
5     keep
7     keep
8     keep
9     keep
10    keep
Name: useforpredictions, dtype: object
```

Q8 [1 point]: Now remove all records in tr dataset that having any missing values. Show your code.

Python:

Q8: remove rows with NaN, done

```
In [29]: tr_new.shape
Out[29]: (928393, 38)

In [30]: tr_new.dropna(inplace=True)

In [31]: tr_new.shape
Out[31]: (264260, 38)
```

Q10 [1 point]: Create dummy variables for all categorical features. This can be done easily using the **dummyVars()** function in R, but you may use whatever function you choose. Show your R code. Do this in Python as well to earn 0.5 extra credit.

Python:

Q10: creat dummy, done

In [32]:
```
#find out numeric columns
numeric_cols = tr_new._get_numeric_data().columns
print numeric_cols
```

```
Index([u'is_churn', u'city', u'bd', u'registered_via', u'avgpayment',
       u'latest_renew', u'latest_cancel', u'pct_cancel', u'meth36', u'meth37',
       u'meth38', u'meth39', u'meth40', u'meth41', u'plan410', u'plan195',
       u'plan7', u'plan31', u'plan0', u'plan30', u'pric149', u'pric99',
       u'pric0', u'pric129', u'pric180', u'pric150', u'numdays', u'daysrange',
       u'num25', u'num50', u'num75', u'num985', u'num100', u'numunq',
       u'totalsecs', u'id'],
      dtype='object')
```

In [33]:
```
all_cols = tr_new.columns
print all_cols
```

```
Index([u'is_churn', u'city', u'bd', u'gender', u'registered_via',
       u'avgpayment', u'latest_renew', u'latest_cancel', u'pct_cancel',
       u'meth36', u'meth37', u'meth38', u'meth39', u'meth40', u'meth41',
       u'plan410', u'plan195', u'plan7', u'plan31', u'plan0', u'plan30',
       u'pric149', u'pric99', u'pric0', u'pric129', u'pric180', u'pric150',
       u'numdays', u'daysrange', u'num25', u'num50', u'num75', u'num985',
       u'num100', u'numunq', u'totalsecs', u'useforpredictions', u'id'],
      dtype='object')
```

In [34]:
```
#find out categorical columns
cat_list = list(set(all_cols) - set(numeric_cols))
print cat_list
```

```
['gender', 'useforpredictions']
```

In [35]:
```
tr_gender = pd.get_dummies(tr_new['gender'])
print tr_gender[0:10]
```

. . .

In [36]:
```
tr_city = pd.get_dummies(tr_new['city'], prefix= 'city')
print tr_city.columns
```

. . .

In [37]:
```
tr_reg = pd.get_dummies(tr_new['registered_via'], prefix= 'registered_via')
print tr_reg.columns
```

. . .

In [38]:
```
tr_new = pd.concat([tr_new, tr_gender, tr_city, tr_reg], axis=1)
print tr_new[0:10]
```

. . .

In [39]:
```
tr_new.shape
```

Out[39]: (264260, 66)

In [40]:
```
tr_new_header = tr_new.columns.values
print tr_new_header
```

```
['is_churn' 'city' 'bd' 'gender' 'registered_via' 'avgpayment'
 'latest_renew' 'latest_cancel' 'pct_cancel' 'meth36' 'meth37' 'meth38'
 'meth39' 'meth40' 'meth41' 'plan410' 'plan195' 'plan7' 'plan31' 'plan0'
 'plan30' 'pric149' 'pric99' 'pric0' 'pric129' 'pric180' 'pric150'
 'numdays' 'daysrange' 'num25' 'num50' 'num75' 'num985' 'num100' 'numunq'
 'totalsecs' 'useforpredictions' 'id' 'female' 'male  ' 'city_1.0'
 'city_3.0' 'city_4.0' 'city_5.0' 'city_6.0' 'city_7.0' 'city_8.0'
 'city_9.0' 'city_10.0' 'city_11.0' 'city_12.0' 'city_13.0' 'city_14.0'
 'city_15.0' 'city_16.0' 'city_17.0' 'city_18.0' 'city_19.0' 'city_20.0'
 'city_21.0' 'city_22.0' 'registered_via_3.0' 'registered_via_4.0'
 'registered_via_7.0' 'registered_via_9.0' 'registered_via_13.0']
```

In [41]:
```
#drop columns of string type: city, gender, registered_via, id, useforpredictions
tr2 = tr_new.drop(tr_new.columns[[1, 3, 4, 36, 37]], axis = 1)
```

In [42]:
```
tr2_col = tr2.columns.values
print tr2_col
```

```
['is_churn' 'bd' 'avgpayment' 'latest_renew' 'latest_cancel' 'pct_cancel'
 'meth36' 'meth37' 'meth38' 'meth39' 'meth40' 'meth41' 'plan410' 'plan195'
 'plan7' 'plan31' 'plan0' 'plan30' 'pric149' 'pric99' 'pric0' 'pric129'
 'pric180' 'pric150' 'numdays' 'daysrange' 'num25' 'num50' 'num75' 'num985'
 'num100' 'numunq' 'totalsecs' 'female' 'male  ' 'city_1.0' 'city_3.0'
 'city_4.0' 'city_5.0' 'city_6.0' 'city_7.0' 'city_8.0' 'city_9.0'
 'city_10.0' 'city_11.0' 'city_12.0' 'city_13.0' 'city_14.0' 'city_15.0'
 'city_16.0' 'city_17.0' 'city_18.0' 'city_19.0' 'city_20.0' 'city_21.0'
 'city_22.0' 'registered_via_3.0' 'registered_via_4.0' 'registered_via_7.0'
 'registered_via_9.0' 'registered_via_13.0']
```

Q11 [1 point]: Remove any features that have zero- or near-zero variances using the **nearZeroVar()** function. Show your R code. Do this in Python as well to earn 0.5 extra credit.

Python:

Q11: variance, done

In [43]:
```
from sklearn.feature_selection import VarianceThreshold
```

In [44]:
```
sel = VarianceThreshold(threshold = (0.8 * (1-0.8)))
```

In [45]:
```
tr2.head()
```

...

In [46]:
```
sel.fit_transform(tr2)
columns = tr2.columns
labels = [columns[x] for x in sel.get_support(indices = True) if x]
```

In [47]:
```
pd.DataFrame(sel.fit_transform(tr2), columns=labels)
```

Out[47]:

	bd	avgpayment	meth40	meth41	plan31	plan0	pric0	numdays	daysrange	num25	...	num985	num100	numunq	totalsecs	fem
0	21.0	149.000000	0.0	1.0	0.0	0.0	0.0	604.0	353.5	7.8244	...	0.6004	28.8176	28.2448	6.754986e+03	0.0
1	18.0	141.550000	0.0	0.0	0.0	0.0	1.0	566.0	288.5	8.5032	...	1.4847	43.7037	41.1008	1.186060e+04	1.0
2	34.0	149.000000	1.0	0.0	1.0	1.0	1.0	43.0	201.0	3.1163	...	0.3023	5.4651	10.1860	1.653545e+03	0.0
3	39.0	149.000000	0.0	0.0	0.0	1.0	1.0	14.0	642.0	0.9286	...	0.7143	10.6429	13.5000	2.969054e+03	0.0
4	27.0	149.000000	0.0	0.0	0.0	1.0	1.0	738.0	409.5	8.7573	...	0.5911	25.2589	33.6357	6.703466e+03	0.0
5	24.0	149.000000	0.0	0.0	0.0	1.0	1.0	195.0	772.0	2.9744	...	1.0667	9.1846	9.3282	2.666899e+03	0.0

In [48]:
```
is_churn = pd.DataFrame(tr2['is_churn']).reset_index()
del is_churn['index']
```

In [49]:
```
tr3 = pd.concat([pd.DataFrame(sel.fit_transform(tr2), columns=labels),is_churn], axis=1)
```

In [50]:
```
tr3.tail()
```

Out[50]:

daysrange	num25	...	num100	numunq	totalsecs	female	male	city_13.0	registered_via_3.0	registered_via_7.0	registered_via_9.0	is_churn
409.5	10.4197	...	22.1812	30.8049	5.999273e+03	0.0	1.0	1.0	0.0	0.0	1.0	0
408.0	1.1526	...	23.8438	25.2739	6.964145e+03	0.0	1.0	0.0	0.0	0.0	1.0	0
284.0	13.5486	...	18.9995	31.8911	5.188767e+03	0.0	1.0	0.0	1.0	0.0	0.0	1
401.5	1.2602	...	22.7560	22.3818	-1.376623e+13	0.0	1.0	1.0	0.0	0.0	1.0	0
408.0	4.9474	...	20.3867	19.1343	-6.414028e+12	1.0	0.0	1.0	1.0	0.0	0.0	0

In [51]:
```
tr3.shape
```

Out[51]:
```
(264260, 23)
```

Q12 [1 point]: Identify and remove any features having a greater than 90% correlation using the **findCorrelation()** function. Show your R code. Do this in Python as well to earn 0.5 extra credit.

Python:

In [53]:
```
pd.DataFrame.corr(tr3) > 0.9    #we should remove "meth41"
```

Out[53]:

	bd	avgpayment	meth40	meth41	plan31	pla
bd	True	False	False	False	False	Fal
avgpayment	False	True	False	False	False	Fal
meth40	False	False	True	False	False	Fal
meth41	False	False	False	True	False	Fal
plan31	False	False	False	False	True	Fal
plan0	False	False	False	False	False	Tru
pric0	False	False	False	False	False	Fal
numdays	False	False	False	False	False	Fal
daysrange	False	False	False	False	False	Fal
num25	False	False	False	False	False	Fal
num50	False	False	False	False	False	Fal
num75	False	False	False	False	False	Fal
num985	False	False	False	False	False	Fal
num100	False	False	False	False	False	Fal
numunq	False	False	False	False	False	Fal
totalsecs	False	False	False	False	False	Fal
female	False	False	False	False	False	Fal
male	False	False	False	False	False	Fal
city_13.0	False	False	False	False	False	Fal
registered_via_3.0	False	False	False	False	False	Fal
registered_via_7.0	False	False	False	True	False	Fal
registered_via_9.0	False	False	False	False	False	Fal
is_churn	False	False	False	False	False	Fal

23 rows × 23 columns

```
In [54]: #remove "meth41" column
         tr4 = tr3.drop(tr3.columns[[3]], axis = 1)
         print tr4.columns

Index([u'bd', u'avgpayment', u'meth40', u'plan31', u'plan0', u'pric0',
       u'numdays', u'daysrange', u'num25', u'num50', u'num75', u'num985',
       u'num100', u'numunq', u'totalsecs', u'female', u'male  ', u'city_13.0',
       u'registered_via_3.0', u'registered_via_7.0', u'registered_via_9.0',
       u'is_churn'],
      dtype='object')
```

Q13 [1 point]: Identify and remove any features that are linear combinations of each other using the **findLinearCombos()** function. Show your R code. Do this in Python as well to earn 0.5 extra credit.

Python:

Q13: Linear Combination

```
In [55]: import sympy

In [56]: #be careful!!!!
         #this code will take 20 minutes to run.
         reduced_form, inds = sympy.Matrix(tr4.values).rref()

In [57]: inds  #we should keep all the columns except "is_churn"

Out[57]: [0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21]

In [58]: tr4.columns

Out[58]: Index([u'bd', u'avgpayment', u'meth40', u'plan31', u'plan0', u'pric0',
                u'numdays', u'daysrange', u'num25', u'num50', u'num75', u'num985',
                u'num100', u'numunq', u'totalsecs', u'female', u'male  ', u'city_13.0',
                u'registered_via_3.0', u'registered_via_7.0', u'registered_via_9.0',
                u'is_churn'],
               dtype='object')
```

Q14 [1 point]: Coerce the following features to factors: city4, city5, city6, city13, city15, city22, genderfemale, registered_via3, registered_via4, registered_via7, registered_via9, latest_renew, meth36, meth37, meth38, meth40, plan7, plan31, and plan0. Show your R code. Do this in Python as well to earn 0.5 extra credit.

Python:

Q9 & Q14: covert some features to factors

In [59]:
```
#astype
tr5 = tr4
tr5['city_13.0'] = tr5["city_13.0"].astype('category')
tr5['registered_via_3.0'] = tr5["registered_via_3.0"].astype('category')
tr5['registered_via_7.0'] = tr5["registered_via_7.0"].astype('category')
tr5['registered_via_9.0'] = tr5["registered_via_9.0"].astype('category')
tr5['meth40'] = tr5["meth40"].astype('category')
tr5['plan31'] = tr5["plan31"].astype('category')
tr5['plan0'] = tr5["plan0"].astype('category')
tr5.dtypes
```

Out[59]:
```
bd                   float64
avgpayment           float64
meth40              category
plan31              category
plan0               category
pric0                float64
numdays              float64
daysrange            float64
num25                float64
num50                float64
num75                float64
num985               float64
num100               float64
numunq               float64
totalsecs            float64
female               float64
male                 float64
city_13.0           category
```

Q15 [1 point]: Use the **preprocess()** function to center and scale your tr dataset. Then use the **predict()** function to actually transform your variables. Show your R code. Do this in Python as well to earn 0.5 extra credit.

Python:

Q15: center and scale and predict

In [60]:
```
import sklearn
from sklearn import preprocessing
```

In [61]:
```
tr4_test = tr5.iloc[:,0:21]
tr4_target = pd.DataFrame(tr5.iloc[:,21])
```

In [62]:
```
tr4_test.head(1)
```

...

In [63]:
```
labels = tr4_test.columns
tr6 = pd.DataFrame(sklearn.preprocessing.scale(tr4_test), columns=labels)
```

```
C:\Users\Sunny Lin\AppData\Local\Enthought\Canopy\edm\envs\User\lib\site-packages\sklearn\preprocessing\data.py:160: UserWarnin
g: Numerical issues were encountered when centering the data and might not be solved. Dataset may contain too large values. You
may need to prescale your features.
  warnings.warn("Numerical issues were encountered "
```

In [64]:
```
tr7 = pd.concat([tr6,tr4_target],axis =1 )
```

In [65]:
```
tr7.head()
```

Out[65]:

	bd	avgpayment	meth40	plan31	plan0	pric0	numdays	daysrange	num25	num50	...	num100	numunq	totalsecs
0	-0.443779	-0.101982	-0.566915	-0.580077	-0.961076	-1.068820	0.942997	0.120892	0.336989	-0.297346	...	0.013329	0.026975	0.095182
1	-0.598319	-0.236475	-0.566915	-0.580077	-0.961076	0.935611	0.777221	-0.367969	0.454154	0.288776	...	0.597707	0.726219	0.095182
2	0.225896	-0.101982	1.763933	1.723909	1.040500	0.935611	-1.504385	-1.026051	-0.475661	-0.266209	...	-0.903413	-0.955253	0.095182

Modeling

To answer the following questions, make sure you are using your transformed/standardized tr data from the previous questions.

Q16 [1 point]: Load the caret package and set a seed of 1234. Use make.names() and relevel() on your target variable to ensure the response works for all methods. You might need to define 1s or churners as "X1". Show your R code.

Python:

Q16: set seed of 1234 ¶

In [66]:
```
import random
```

In [67]:
```
random.seed(1234)
```

Q17 [1 point]: Use the **createDataPartition()** function to create an 80/20 train and test datasets. Show your R code. Do this in Python as well to earn 0.5 extra credit.

Python:

Q17: data partition

In [68]:
```
from sklearn.cross_validation import train_test_split
```

```
C:\Users\Sunny Lin\AppData\Local\Enthought\Canopy\edm\envs\User\lib\site-packages\sklearn\cross_validation.py:44: DeprecationWarning: This module was deprecated in version 0.18 in favor of the model_selection module into which all the refactored classes and functions are moved. Also note that the interface of the new CV iterators are different from that of this module. This module will be removed in 0.20.
  "This module will be removed in 0.20.", DeprecationWarning)
```

In [70]:
```
Y = tr7["is_churn"]
X = tr6
```

In [71]:
```
x_train, x_test, y_train, y_test = train_test_split(X, Y, test_size=0.2)
```

In [72]:
```
x_train.head()
```

Out[72]:

	bd	avgpayment	meth40	plan31	plan0	pric0	numdays	daysrange	num25	num50	...	num985	num100	nun
228293	0.483463	-0.101982	-0.566915	-0.580077	1.040500	0.935611	-1.390959	3.215758	2.120879	0.029926	...	-0.592088	0.135413	0.92
75533	0.328922	0.225549	-0.566915	-0.580077	-0.961076	-1.068820	-0.138911	-0.100976	-0.661282	-0.335498	...	-0.407213	0.708721	0.59
78732	-0.289239	1.746631	-0.566915	-0.580077	-0.961076	-1.068820	-0.357038	-0.815465	-0.652980	-0.728250	...	-0.460732	-0.260051	-0.3
213852	0.019842	-0.418438	1.763933	1.723909	1.040500	0.935611	-0.038573	0.372843	0.681408	0.908239	...	-0.220468	0.578381	1.02
204409	-0.083185	-0.008709	-0.566915	1.723909	1.040500	0.935611	1.645365	0.542065	0.104764	0.362141	...	0.163598	1.528400	2.25

5 rows × 21 columns

Q19 [1 point]: Use the **trainControl()** function to specify 5-fold cross-validation, as well as anything else that is important to indicate you are doing classification-type modeling. Show your R code.

Python:

Q19: cross-validation

```
In [73]: from sklearn import cross_validation
         from sklearn.model_selection import KFold

In [75]: kf = KFold(n_splits = 5, shuffle = True)
```

Q20 [1 point]: Use the **train()** function to build any model you want. Show your R code. Do this in Python as well to earn 0.5 extra credit.

Python:

Q20: SVM model

```
In [260]: from sklearn.svm import SVC

In [261]: clf = sklearn.svm.SVC(kernel='linear', C=1).fit(x_train, y_train)

In [262]: #this is the model
          clf

Out[262]: SVC(C=1, cache_size=200, class_weight=None, coef0=0.0,
            decision_function_shape=None, degree=3, gamma='auto', kernel='linear',
            max_iter=-1, probability=False, random_state=None, shrinking=True,
            tol=0.001, verbose=False)
```

Model Evaluation

Q21 [1 point]: Use the **predict()** function to generate predicted probabilities and predicted classes for both the train and test sets. Show your R code. Do this in Python as well to earn 0.5 extra credit.

Python:

Q21: predict

```
In [276]: tr5[0:10]

In [281]: #feed in a row and get is_churn result = 0 or 1
          p = clf.predict(x_test)

In [282]: p

Out[282]: array([0, 0, 0, ..., 0, 0, 0], dtype=int64)
```

Q22 [1 point]: Use the **confusionMatrix()** function to evaluate your model. Compare the train and test statistics and describe if you have overfit your model. Show your R code. Do this in Python as well to earn 0.5 extra credit.

Python:

Q22: evaluate model

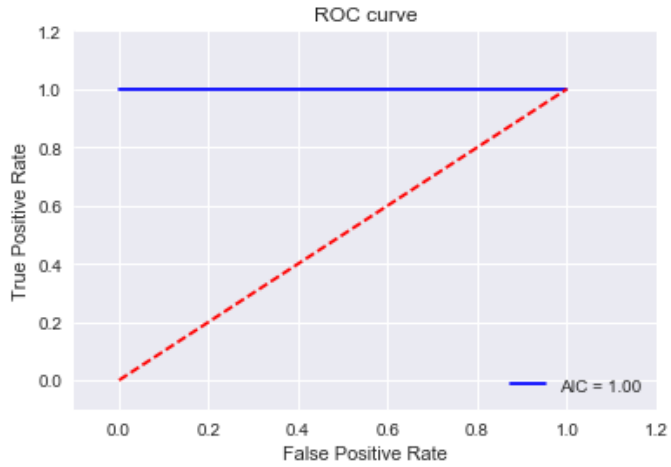
```
In [285]: from sklearn.metrics import confusion_matrix

In [286]: confusion_matrix(y_test, p)

Out[286]: array([[51166,     0],
                 [    0,  1686]])
```

Q23 [1 point]: Use the **roc()** function to create an ROC curve on the test set predictions, as well as the **auc()** function to capture the area-under-the-curve statistc. Show your R code. Do this in Python as well to earn 0.5 extra credit.

Python:

Q23: create ROC curve and get the area-under-the-curve

```
In [287]: from sklearn.metrics import roc_curve, auc

In [290]: actual = y_test
          predictions = p

In [291]: false_positive_rate, true_positive_rate, thresholds = roc_curve(actual, predictions)

In [292]: roc_auc = auc(false_positive_rate, true_positive_rate)

In [296]: plt.title('ROC curve')
          plt.plot(false_positive_rate, true_positive_rate, 'b',
          label = 'AIC = %0.2f' % roc_auc)
          plt.legend(loc='lower right')
          plt.plot([0,1],[0,1],'r--')
          plt.xlim([-0.1,1.2])
          plt.ylim([-0.1,1.2])
          plt.ylabel('True Positive Rate')
          plt.xlabel('False Positive Rate')
          plt.show()
```



Deployment

Q24 [1 point]: Use the **mgcv** R package and **saveRDS()** function to save your predictive model for use later. Show your R code. Do this in Python as well to earn 0.5 extra credit.

Python:

Q24: save model

```
In [297]: import pickle

In [299]: filename = "DataMiningHW2_AAPTeam.sav"

In [301]: pickle.dump(clf,open(filename, 'wb'))
```

Q25 [1 point]: Pull the te_clean dataset from your database into R. Coerce the factor variables in this dataset exactly like you did in Q14. Show your R code.

Python:

Q25: coerce factor variables in te_clean

```
In [303]: te_clean.columns
```

...

```
In [305]: te_clean1 = te_clean
te_clean1['city4'] = te_clean1["city4"].astype('category')
te_clean1['city5'] = te_clean1["city5"].astype('category')
te_clean1['city6'] = te_clean1["city6"].astype('category')
te_clean1['city13'] = te_clean1["city13"].astype('category')
te_clean1['city15'] = te_clean1["city15"].astype('category')
te_clean1['city22'] = te_clean1["city22"].astype('category')
te_clean1['registered_via3'] = te_clean1["registered_via3"].astype('category')
te_clean1['registered_via4'] = te_clean1["registered_via4"].astype('category')
te_clean1['registered_via7'] = te_clean1["registered_via7"].astype('category')
te_clean1['registered_via9'] = te_clean1["registered_via9"].astype('category')
te_clean1['meth38'] = te_clean1["meth38"].astype('category')
te_clean1['meth40'] = te_clean1["meth40"].astype('category')
te_clean1['plan7'] = te_clean1["plan7"].astype('category')
te_clean1['plan31'] = te_clean1["plan31"].astype('category')
te_clean1['plan0'] = te_clean1["plan0"].astype('category')
te_clean1['latest_renew'] = te_clean1["latest_renew"].astype('category')
te_clean1.dtypes
```

```
Out[305]: msno                object
is_churn              int64
city4              category
city5              category
city6              category
city13             category
city15             category
```

Q26 [1 point]: Use the **preprocess()** function to center and scale your te_clean dataset just like you did to your tr dataset in Q15. Then use the **predict()** function to actually transform your variables. Show your R code.

Python:

Q26: preproess te_clean

```
In [307]: te_clean2 = te_clean1.drop(te_clean1.columns[0], axis = 1)

In [313]: te_clean2.shape

Out[313]: (907471, 38)

In [317]: sklearn.preprocessing.scale(te_clean2)

Out[317]: array([[ 0.        , -0.227615  ,  3.55332237, ...,  0.90065231,
                   1.45317015,  0.06208955],
                 [ 0.        , -0.227615  , -0.28142676, ..., -0.66388934,
                  -0.60502941,  0.06208954],
                 [ 0.        , -0.227615  , -0.28142676, ..., -0.64877931,
                  -0.85378268,  0.06208954],
                 ...,
                 [ 0.        , -0.227615  , -0.28142676, ..., -0.68480528,
                  -0.8496938 ,  0.06208954],
                 [ 0.        , -0.227615  , -0.28142676, ...,  0.22619196,
                   0.2733079 ,  0.06208955],
                 [ 0.        , -0.227615  , -0.28142676, ..., -0.7505984 ,
                  -0.5401772 ,  0.06208954]])

In [318]: labels = te_clean2.columns
          te_clean3 = pd.DataFrame(sklearn.preprocessing.scale(te_clean2), columns=labels)
```