Homework 3

Andey Nunes, Mengyu Li, Jordan Hilton, Peter Boss

2/2/2019

Document and Exercise Set Up

```
Claims_and_Days <- read_csv("Claims_and_Days.csv",
col_types = cols(placesvc = col_factor(),
                 dsfs = col_factor(),
                 PrimaryConditionGroup = col_factor(),
                 CharlsonIndex = col_character(),
                 sex = col_factor(),
                 Risk_Level = col_factor()))
```

```
## Warning: The following named parsers don't match the column names:
## Risk_Level
```

```
names(Claims_and_Days)
```

```
##  [1] "ID"                    "MemberID"
##  [3] "ProviderID"            "vendor"
##  [5] "pcp"                   "Year"
##  [7] "specialty"             "placesvc"
##  [9] "paydelay"              "LengthOfStay"
## [11] "dsfs"                  "PrimaryConditionGroup"
## [13] "CharlsonIndex"         "sex"
## [15] "AgeAtFirstClaim"       "DaysInHospital_Y2"
```

```
# fix this chunk so it doesn't print output for file loading
```

```
Claims_and_Days <- Claims_and_Days %>%
  mutate(CharlsonIndex = case_when(
    CharlsonIndex == "0" ~ "0" ,
    CharlsonIndex == "2-Jan" ~ "1-2",
    CharlsonIndex == "4-Mar" ~ "3-4",
    CharlsonIndex == "5+" ~ "5+"),
    AgeAtFirstClaim = case_when(
      AgeAtFirstClaim == "19-Oct" ~ "10-19",
      AgeAtFirstClaim != "19-Oct" ~ AgeAtFirstClaim),
    LengthOfStay = if_else(is.na(LengthOfStay), "0 or unknown", LengthOfStay)) %>%
  mutate(CharlsonIndex = ordered(CharlsonIndex, levels =
                                   c("0","1-2", "3-4", "5+")),
         AgeAtFirstClaim = ordered(AgeAtFirstClaim, levels = c(
           "0-9", "10-19", "20-29", "30-39", "40-49", "50-59", "60-69", "70-79",  "80+")),
         LengthOfStay = ordered(LengthOfStay, levels = c(
           "0 or unknown", "1 day", "2 days", "3 days","4 days", "5 days", "6 days", "1- 2 weeks",
           "2- 4 weeks", "4- 8 weeks", "8-12 weeks", "12-26 weeks", "26+ weeks"))
)
```

Tidy 1 R Algorithm: Homework 3

1. Quantize the answer field: `Risk_Level`

```
VlookupSim <- Claims_and_Days %>%
  mutate(Risk_Level = case_when(
    DaysInHospital_Y2 == 0 ~ "A",
    DaysInHospital_Y2 == 1 ~ "B",
    DaysInHospital_Y2 == 2 | DaysInHospital_Y2 == 3 ~ "C",
    DaysInHospital_Y2 == 4 | DaysInHospital_Y2 == 5 ~ "D",
    DaysInHospital_Y2 >= 6 ~ "E"),
      Risk_Level_label = case_when(
        Risk_Level == "A" ~ "no risk",
        Risk_Level == "B" ~ "very low risk",
        Risk_Level == "C" ~ "low risk",
        Risk_Level == "D" ~ "medium risk",
        Risk_Level == "E" ~ "high risk")) %>%
  mutate(Risk_Level_label = ordered(Risk_Level_label, levels = c(
    "no risk", "very low risk", "low risk", "medium risk", "high risk")))
summary(VlookupSim)
```

```
##        ID             MemberID           ProviderID
##  Min.   :     1   Min.   :    25872   Min.   :    59876
##  1st Qu.:161177   1st Qu.:253788245   1st Qu.:229695343
##  Median :322354   Median :500835952   Median :540714368
##  Mean   :322354   Mean   :500341450   Mean   :499619932
##  3rd Qu.:483530   3rd Qu.:747315828   3rd Qu.:776433376
##  Max.   :644706   Max.   :999999313   Max.   :999923007
##                                       NA's   :3903
##      vendor             pcp             Year            specialty
##  Min.   :   7666   Min.   :   982   Length:644706      Length:644706
##  1st Qu.:2999365   1st Qu.:219523   Class :character   Class :character
##  Median :5550418   Median :483696   Mode  :character   Mode  :character
##  Mean   :5291877   Mean   :476158
##  3rd Qu.:7520076   3rd Qu.:708982
##  Max.   :9997631   Max.   :998831
##  NA's   :6492      NA's   :1619
##                 placesvc         paydelay           LengthOfStay
##  Office             :404309   Min.   :  0     0 or unknown:617827
##  Independent Lab    :125498   1st Qu.: 28     1 day       : 22281
##  Urgent Care        : 49748   Median : 37     2 days      :  2162
##  Outpatient Hospital: 30989   Mean   : 47     3 days      :   896
##  Inpatient Hospital : 21783   3rd Qu.: 58     4 days      :   466
##  Ambulance          :  8251   Max.   :161     1- 2 weeks  :   302
##  (Other)            :  4128   NA's   :44623   (Other)     :   772
##           dsfs        PrimaryConditionGroup CharlsonIndex sex
##  0- 1 month :177047   MSC2a3  :111499      0  :369191    F:378963
##  1- 2 months: 63077   METAB3  : 71800      1-2:263524    M:265743
##  2- 3 months: 56464   ARTHSPIN: 71711      3-4: 10780
##  3- 4 months: 52594   NEUMENT : 46305      5+ :  1211
##  4- 5 months: 46923   RESPR4  : 37533
##  5- 6 months: 46839   MISCHRT : 31421
##  (Other)    :201762   (Other) :274437
```

```
##  AgeAtFirstClaim  DaysInHospital_Y2  Risk_Level
##  70-79  :170799   Min.   : 0.0      Length:644706
##  80+    :101319   1st Qu.: 0.0      Class :character
##  60-69  :100934   Median : 0.0      Mode  :character
##  50-59  : 75146   Mean   : 1.1
##  40-49  : 71580   3rd Qu.: 0.0
##  30-39  : 44739   Max.   :15.0
##  (Other): 80189
##       Risk_Level_label
##  no risk      :494761
##  very low risk: 52853
##  low risk     : 25193
##  medium risk  : 25600
##  high risk    : 46299
##
##
```

2. Create pivot tables: field counts by risk group

proportion of observations for risk groups

```
prop_obs <- function(df = VlookupSim, x,
                     y = VlookupSim$Risk_Level_label) {
   prop <- with(df, table(x, y))
   prop <- cbind(prop, total = rowSums(prop))
   high_prop <- prop[,5]/prop[,6]*100
   prop <- cbind(prop, high_risk_proportion = high_prop)
   #proptibble <- tibble(prop) %>%
    #  arrange(desc(high_risk_proportion))

   return(prop)
}
```

Primary Condition Group pivot table

```
pcg <- prop_obs(x = VlookupSim$PrimaryConditionGroup)
kable(pcg)
```

	no risk	very low risk	low risk	medium risk	high risk	total	high_risk_proportion
MSC2a3	89147	8331	3899	3769	6353	111499	5.7
RESPR4	29576	3274	1355	1266	2062	37533	5.5
METAB3	56338	5377	2383	2662	5040	71800	7.0
NEUMENT	35789	3936	1524	1713	3343	46305	7.2
SKNAUT	21217	2150	847	903	1784	26901	6.6
GIBLEED	21002	2755	1343	1222	2524	28846	8.8
MISCHRT	24141	2585	994	1264	2437	31421	7.8
ARTHSPIN	54575	6096	3087	3080	4873	71711	6.8
MISCL1	968	125	64	46	98	1301	7.5
HEART2	8446	1124	542	716	1591	12419	12.8
CHF	2004	322	130	153	563	3172	17.8
RENAL3	9657	1070	496	554	972	12749	7.6
ROAMI	8884	1200	622	676	1453	12835	11.3
TRAUMA	14868	1752	710	607	1023	18960	5.4

	no risk	very low risk	low risk	medium risk	high risk	total	high_risk_proportion
FXDISLC	6951	710	289	262	470	8682	5.4
INFEC4	17064	1864	783	691	1139	21541	5.3
UTI	7817	822	403	433	680	10155	6.7
ODaBNCA	10050	844	368	397	657	12316	5.3
MISCL5	9330	1098	496	524	867	12315	7.0
COPD	8516	1007	409	575	1199	11706	10.2
APPCHOL	3805	461	207	213	355	5041	7.0
AMI	5919	713	483	588	1188	8891	13.4
PNEUM	1905	238	81	131	226	2581	8.8
HEMTOL	4513	461	177	286	757	6194	12.2
HIPFX	655	106	82	50	130	1023	12.7
PRGNCY	5336	411	1048	609	429	7833	5.5
CANCRB	7343	685	346	421	944	9739	9.7
GYNEC1	8793	775	734	553	379	11234	3.4
HEART4	5142	612	327	270	726	7077	10.3
GIOBSENT	2186	230	132	94	189	2831	6.7
METAB1	752	95	35	56	86	1024	8.4
PERVALV	618	61	32	54	85	850	10.0
CATAST	309	33	16	19	68	445	15.3
RENAL2	1323	178	76	97	429	2103	20.4
SEIZURE	3530	625	271	278	458	5162	8.9
FLaELEC	899	104	45	55	144	1247	11.6
STROKE	1474	207	122	110	236	2149	11.0
LIVERDZ	595	57	20	30	48	750	6.4
SEPSIS	76	12	5	10	17	120	14.2
GYNECA	1847	162	163	91	106	2369	4.5
PERINTL	128	9	8	6	10	161	6.2
PNCRDZ	179	40	0	20	20	259	7.7
CANCRM	170	21	11	7	22	231	9.5
CANCRA	866	80	19	35	101	1101	9.2
RENAL1	58	35	9	4	18	124	14.5

Charlson Index Group pivot table

Length of Stay pivot table

3. Simulate the 1R Algorithm

Ignoring Risk Level A build the 1R Rule for each of Primary Condition Group, Charlson Index Group, and Length of Stay.

```
data1R <- filter(VlookupSim, Risk_Level != "A")
str(data1R)
```

```
## Classes 'tbl_df', 'tbl' and 'data.frame':    149945 obs. of  18 variables:
##  $ ID                  : num  42 34 35 36 37 38 39 41 31 43 ...
##  $ MemberID            : num  60481 60481 60481 60481 60481 ...
##  $ ProviderID          : num  310462192 310462192 148070583 310462192 640687583 ...
##  $ vendor              : num  9380038 9380038 30539 9380038 4192922 ...
##  $ pcp                 : num  574239 574239 574239 574239 574239 ...
##  $ Year                : chr  "Y1" "Y1" "Y1" "Y1" ...
```

```
##  $ specialty            : chr  "Internal" "Internal" "Surgery" "Internal" ...
##  $ placesvc             : Factor w/ 8 levels "Independent Lab",..: 2 2 5 2 1 1 5 5 2 2 ...
##  $ paydelay             : num  78 70 55 52 36 49 26 158 84 80 ...
##  $ LengthOfStay         : Ord.factor w/ 13 levels "0 or unknown"<..: 1 1 1 1 1 1 1 1 1 1 ...
##  $ dsfs                 : Factor w/ 12 levels "0- 1 month","1- 2 months",..: 9 10 6 6 6 6 6 6 5 9 ..
##  $ PrimaryConditionGroup: Factor w/ 45 levels "MSC2a3","RESPR4",..: 3 3 2 3 1 5 8 3 3 2 ...
##  $ CharlsonIndex        : Ord.factor w/ 4 levels "0"<"1-2"<"3-4"<..: 2 2 2 2 2 2 3 2 3 2 ...
##  $ sex                  : Factor w/ 2 levels "F","M": 2 2 2 2 2 2 2 2 2 2 ...
##  $ AgeAtFirstClaim      : Ord.factor w/ 9 levels "0-9"<"10-19"<..: 7 7 7 7 7 7 7 7 7 7 ...
##  $ DaysInHospital_Y2    : num  15 15 15 15 15 15 15 15 15 15 ...
##  $ Risk_Level           : chr  "E" "E" "E" "E" ...
##  $ Risk_Level_label     : Ord.factor w/ 5 levels "no risk"<"very low risk"<..: 5 5 5 5 5 5 5 5 5 5 .
```

```r
# custom function that generates a table of counts, a vector of the risk label with highest count,
# from each field state, which is the 1R decision, then it creates a vector for error calculation

prop_obs_1R <- function(df = data1R, x,
                        y = data1R$Risk_Level_label) {
   prop <- with(df, table(x, y))
   prop <- cbind(prop, total = rowSums(prop))
   high_value <- rep(99, dim(prop)[1])  # set a vector to hold the decision
   # loop across the vector to find the most likely result, and return that name
   for(i in 1:length(high_value)){
     high_value[i] = colnames(prop)[which.max(prop[i, 1:5])]
   }
   errors <- rep(99, dim(prop)[1])  # set a vector to hold the total errors
   for(k in 1:length(errors)){
     errors[k] = prop[k,6] - prop[k,which.max(prop[k, 1:5])]
   }
   error_rate <- rep(99, dim(prop)[1])  # set a vector to hold the error rate
   # loop across the vector, summing up the non-1R talleys, and dividing by total observations
   for(j in 1:length(error_rate)){
     error_rate[j] = round(1 - max(prop[j, 1:5])/prop[j,6], 4)
   }
   # combine the table, the decision, and the error rate
   prop <- cbind(prop, decision_1R = high_value, errors, error_rate)
   # set the first col to NA since we're ignoring no-risk situations
   prop[,1] <- NA
   #proptibble <- tibble(prop) %>%
    #  arrange(desc(high_risk_proportion))

   return(prop)
}
```

Primary Condition Group 1R Decisions and Error Rates

```r
pcg_1R <- prop_obs_1R(x = data1R$PrimaryConditionGroup)
pcg_1R_tot <- sum(as.numeric(pcg_1R[,6]))
pcg_1R_err <- sum(as.numeric(pcg_1R[,8]))
pcg_1R_rate <- pcg_1R_err/pcg_1R_tot
kable(pcg_1R)
```

	no risk	very low risk	low risk	medium risk	high risk	total	decision_1R	errors	error_rate
MSC2a3	NA	8331	3899	3769	6353	22352	very low risk	14021	0.6273

	no risk	very low risk	low risk	medium risk	high risk	total	decision_1R	errors	error_rate
RESPR4	NA	3274	1355	1266	2062	7957	very low risk	4683	0.5885
METAB3	NA	5377	2383	2662	5040	15462	very low risk	10085	0.6522
NEUMENT	NA	3936	1524	1713	3343	10516	very low risk	6580	0.6257
SKNAUT	NA	2150	847	903	1784	5684	very low risk	3534	0.6217
GIBLEED	NA	2755	1343	1222	2524	7844	very low risk	5089	0.6488
MISCHRT	NA	2585	994	1264	2437	7280	very low risk	4695	0.6449
ARTHSPIN	NA	6096	3087	3080	4873	17136	very low risk	11040	0.6443
MISCL1	NA	125	64	46	98	333	very low risk	208	0.6246
HEART2	NA	1124	542	716	1591	3973	high risk	2382	0.5995
CHF	NA	322	130	153	563	1168	high risk	605	0.518
RENAL3	NA	1070	496	554	972	3092	very low risk	2022	0.6539
ROAMI	NA	1200	622	676	1453	3951	high risk	2498	0.6322
TRAUMA	NA	1752	710	607	1023	4092	very low risk	2340	0.5718
FXDISLC	NA	710	289	262	470	1731	very low risk	1021	0.5898
INFEC4	NA	1864	783	691	1139	4477	very low risk	2613	0.5836
UTI	NA	822	403	433	680	2338	very low risk	1516	0.6484
ODaBNCA	NA	844	368	397	657	2266	very low risk	1422	0.6275
MISCL5	NA	1098	496	524	867	2985	very low risk	1887	0.6322
COPD	NA	1007	409	575	1199	3190	high risk	1991	0.6241
APPCHOL	NA	461	207	213	355	1236	very low risk	775	0.627
AMI	NA	713	483	588	1188	2972	high risk	1784	0.6003
PNEUM	NA	238	81	131	226	676	very low risk	438	0.6479
HEMTOL	NA	461	177	286	757	1681	high risk	924	0.5497
HIPFX	NA	106	82	50	130	368	high risk	238	0.6467
PRGNCY	NA	411	1048	609	429	2497	low risk	1449	0.5803
CANCRB	NA	685	346	421	944	2396	high risk	1452	0.606
GYNEC1	NA	775	734	553	379	2441	very low risk	1666	0.6825
HEART4	NA	612	327	270	726	1935	high risk	1209	0.6248
GIOBSENT	NA	230	132	94	189	645	very low risk	415	0.6434
METAB1	NA	95	35	56	86	272	very low risk	177	0.6507
PERVALV	NA	61	32	54	85	232	high risk	147	0.6336
CATAST	NA	33	16	19	68	136	high risk	68	0.5
RENAL2	NA	178	76	97	429	780	high risk	351	0.45
SEIZURE	NA	625	271	278	458	1632	very low risk	1007	0.617
FLaELEC	NA	104	45	55	144	348	high risk	204	0.5862
STROKE	NA	207	122	110	236	675	high risk	439	0.6504
LIVERDZ	NA	57	20	30	48	155	very low risk	98	0.6323
SEPSIS	NA	12	5	10	17	44	high risk	27	0.6136
GYNECA	NA	162	163	91	106	522	low risk	359	0.6877
PERINTL	NA	9	8	6	10	33	high risk	23	0.697
PNCRDZ	NA	40	0	20	20	80	very low risk	40	0.5
CANCRM	NA	21	11	7	22	61	high risk	39	0.6393
CANCRA	NA	80	19	35	101	235	high risk	134	0.5702
RENAL1	NA	35	9	4	18	66	very low risk	31	0.4697

Charlson Index Group 1R Decisions and Error Rates

```
ci_1R <- prop_obs_1R(x = data1R$CharlsonIndex)
ci_1R_tot <- sum(as.numeric(ci_1R[,6]))
ci_1R_err <- sum(as.numeric(ci_1R[,8]))
ci_1R_rate <- ci_1R_err/ci_1R_tot
kable(ci_1R)
```

	no risk	very low risk	low risk	medium risk	high risk	total	decision_1R	errors	error_rate
0	NA	27731	13408	12086	16051	69276	very low risk	41545	0.5997
1-2	NA	24094	11320	12828	28407	76649	high risk	48242	0.6294
3-4	NA	902	414	627	1701	3644	high risk	1943	0.5332
5+	NA	126	51	59	140	376	high risk	236	0.6277

Length of Stay 1R Decisions and Error Rates

```
los_1R <- prop_obs_1R(x = data1R$LengthOfStay)
los_1R_tot <- sum(as.numeric(los_1R[,6]))
los_1R_err <- sum(as.numeric(los_1R[,8]))
los_1R_rate <- los_1R_err/los_1R_tot
kable(los_1R)
```

	no risk	very low risk	low risk	medium risk	high risk	total	decision_1R	errors	error_rate
0 or unknown	NA	49884	23874	24275	43751	141784	very low risk	91900	0.6482
1 day	NA	2494	1108	1066	1968	6636	very low risk	4142	0.6242
2 days	NA	237	105	129	232	703	very low risk	466	0.6629
3 days	NA	89	30	49	86	254	very low risk	165	0.6496
4 days	NA	42	19	29	40	130	very low risk	88	0.6769
5 days	NA	16	7	24	24	71	medium risk	47	0.662
6 days	NA	11	4	1	9	25	very low risk	14	0.56
1- 2 weeks	NA	35	19	15	72	141	high risk	69	0.4894
2- 4 weeks	NA	20	10	9	43	82	high risk	39	0.4756
4- 8 weeks	NA	23	17	2	68	110	high risk	42	0.3818
8-12 weeks	NA	0	0	0	3	3	high risk	0	0
12-26 weeks	NA	2	0	1	0	3	very low risk	1	0.3333
26+ weeks	NA	0	0	0	3	3	high risk	0	0

Comparison of the three 1R rules

The Primary Condition Group 1R rule has 93726 errors on 149945 observations, for an error rate of 0.63.
The Charlson Index 1R rule has 91966 errors on 149945 observations, for an error rate of 0.61.
The Length of Stay 1R rule has 96973 errors on 149945 observations, for an error rate of 0.65.

The Charlson Index has the lowest error rate, so we select that as our 1R rule. Details of the rule are in the table above.

Q4.1. Why does our selected rule work better (reference error rate).

The Charlson Index has a lower error rate (61%) than the other two options (63% and 65%).

Q4.2. For the 1R Rule is it better to have a lower or higher cardinality?

In this scenario, the lower cardinality may have contributed to a stronger rule. In general, lower cardinality will speed up calculations because it creates a smaller search space. The optimal cardinality will be whatever is most useful to the person using the analysis. For instance, if `no risk`, `very low risk`, and `low risk` were going to be treated the same, it would make sense to bin them together to speed up the computations.

Q4.3. Why are we ignoring risk level A?

We are ignoring risk level A because it is a subset of the data that is independent of the group we are trying to predict and it consists of nearly 77% of the original observations. By working with a smaller subset that contains only the observations with any risk level other than "no risk", we have a smaller search space for our algorithm.

Extra Credit

Ignoring Risk Level A rebin the risk level buckets and rebuild the 1R Rule and see if there is a difference. New categories are as follows:

- 0 DaysInHospital_Y2 group **A** "no risk"
- 1-3 DaysInHospital_Y2 group **B** "low risk"
- 4-6 DaysInHospital_Y2 group **C** "medium risk"
- more than 6 DaysInHospital_Y2 group **D** "high risk"

```
rebinRisk <- Claims_and_Days %>%
  mutate(Risk_Level = case_when(
    DaysInHospital_Y2 == 0 ~ "A",
    DaysInHospital_Y2 > 0 & DaysInHospital_Y2 < 4 ~ "B",
    DaysInHospital_Y2 > 3 & DaysInHospital_Y2 < 7 ~ "C",
    DaysInHospital_Y2 > 6 ~ "D"),
      Risk_Level_label = case_when(
        Risk_Level == "A" ~ "no risk",
        Risk_Level == "B" ~ "low risk",
        Risk_Level == "C" ~ "medium risk",
        Risk_Level == "D" ~ "high risk")) %>%
  mutate(Risk_Level_label = ordered(Risk_Level_label, levels = c(
    "no risk", "low risk", "medium risk", "high risk"))) %>%
  filter(Risk_Level != "A")
```

```
# custom function that generates a table of counts, a vector of the risk label with highest count,
# from each field state, which is the 1R decision, then it creates a vector for error calculation

rebin_1R <- function(df = rebinRisk, x,
                         y = rebinRisk$Risk_Level_label) {
   prop <- with(df, table(x, y))
   prop <- cbind(prop, total = rowSums(prop))
   high_value <- rep(99, dim(prop)[1])  # set a vector to hold the decision
   # loop across the vector to find the most likely result, and return that name
   for(i in 1:length(high_value)){
     high_value[i] = colnames(prop)[which.max(prop[i, 1:4])]
   }
   errors <- rep(99, dim(prop)[1])  # set a vector to hold the total errors
   for(k in 1:length(errors)){
     errors[k] = prop[k,5] - prop[k,which.max(prop[k, 1:4])]
   }
   error_rate <- rep(99, dim(prop)[1])  # set a vector to hold the error rate
   # loop across the vector, summing up the non-1R talleys, and dividing by total observations
   for(j in 1:length(error_rate)){
     error_rate[j] = round(1 - max(prop[j, 1:4])/prop[j,5], 3)
   }
   # combine the table, the decision, and the error rate
```

```
  prop <- cbind(prop, decision_1R = high_value, errors, error_rate)
  # set the first col to NA since we're ignoring no-risk situations
  prop[,1] <- NA
  #proptibble <- tibble(prop) %>%
   #  arrange(desc(high_risk_proportion))

  return(prop)
}
```

Primary Condition Group 1R Decisions and Error Rates on rebinned risk groups

```
pcg_rebin1R <- rebin_1R(x = rebinRisk$PrimaryConditionGroup)
pcg_rebin1R_tot <- sum(as.numeric(pcg_rebin1R[,5]))
pcg_rebin1R_err <- sum(as.numeric(pcg_rebin1R[,7]))
pcg_rebin1R_rate <- pcg_rebin1R_err/pcg_rebin1R_tot
kable(pcg_rebin1R)
```

	no risk	low risk	medium risk	high risk	total	decision_1R	errors	error_rate
MSC2a3	NA	12230	5116	5006	22352	low risk	10122	0.453
RESPR4	NA	4629	1661	1667	7957	low risk	3328	0.418
METAB3	NA	7760	3672	4030	15462	low risk	7702	0.498
NEUMENT	NA	5460	2377	2679	10516	low risk	5056	0.481
SKNAUT	NA	2997	1292	1395	5684	low risk	2687	0.473
GIBLEED	NA	4098	1648	2098	7844	low risk	3746	0.478
MISCHRT	NA	3579	1754	1947	7280	low risk	3701	0.508
ARTHSPIN	NA	9183	4107	3846	17136	low risk	7953	0.464
MISCL1	NA	189	71	73	333	low risk	144	0.432
HEART2	NA	1666	1052	1255	3973	low risk	2307	0.581
CHF	NA	452	242	474	1168	high risk	694	0.594
RENAL3	NA	1566	788	738	3092	low risk	1526	0.494
ROAMI	NA	1822	1003	1126	3951	low risk	2129	0.539
TRAUMA	NA	2462	821	809	4092	low risk	1630	0.398
FXDISLC	NA	999	364	368	1731	low risk	732	0.423
INFEC4	NA	2647	920	910	4477	low risk	1830	0.409
UTI	NA	1225	562	551	2338	low risk	1113	0.476
ODaBNCA	NA	1212	522	532	2266	low risk	1054	0.465
MISCL5	NA	1594	712	679	2985	low risk	1391	0.466
COPD	NA	1416	814	960	3190	low risk	1774	0.556
APPCHOL	NA	668	261	307	1236	low risk	568	0.46
AMI	NA	1196	827	949	2972	low risk	1776	0.598
PNEUM	NA	319	158	199	676	low risk	357	0.528
HEMTOL	NA	638	421	622	1681	low risk	1043	0.62
HIPFX	NA	188	60	120	368	low risk	180	0.489
PRGNCY	NA	1459	731	307	2497	low risk	1038	0.416
CANCRB	NA	1031	644	721	2396	low risk	1365	0.57
GYNEC1	NA	1509	653	279	2441	low risk	932	0.382
HEART4	NA	939	431	565	1935	low risk	996	0.515
GIOBSENT	NA	362	142	141	645	low risk	283	0.439
METAB1	NA	130	69	73	272	low risk	142	0.522
PERVALV	NA	93	90	49	232	low risk	139	0.599
CATAST	NA	49	28	59	136	high risk	77	0.566
RENAL2	NA	254	157	369	780	high risk	411	0.527
SEIZURE	NA	896	358	378	1632	low risk	736	0.451

	no risk	low risk	medium risk	high risk	total	decision_1R	errors	error_rate
FLaELEC	NA	149	88	111	348	low risk	199	0.572
STROKE	NA	329	144	202	675	low risk	346	0.513
LIVERDZ	NA	77	41	37	155	low risk	78	0.503
SEPSIS	NA	17	19	8	44	medium risk	25	0.568
GYNECA	NA	325	123	74	522	low risk	197	0.377
PERINTL	NA	17	13	3	33	low risk	16	0.485
PNCRDZ	NA	40	25	15	80	low risk	40	0.5
CANCRM	NA	32	7	22	61	low risk	29	0.475
CANCRA	NA	99	39	97	235	low risk	136	0.579
RENAL1	NA	44	6	16	66	low risk	22	0.333

Charlson Index Group 1R Decisions and Error Rates on rebinned risk groups

```
ci_rebin1R <- rebin_1R(x = rebinRisk$CharlsonIndex)
ci_rebin1R_tot <- sum(as.numeric(ci_rebin1R[,5]))
ci_rebin1R_err <- sum(as.numeric(ci_rebin1R[,7]))
ci_rebin1R_rate <- ci_rebin1R_err/ci_rebin1R_tot
kable(ci_rebin1R)
```

	no risk	low risk	medium risk	high risk	total	decision_1R	errors	error_rate
0	NA	41139	15765	12372	69276	low risk	28137	0.406
1-2	NA	35414	18298	22937	76649	low risk	41235	0.538
3-4	NA	1316	902	1426	3644	high risk	2218	0.609
5+	NA	177	68	131	376	low risk	199	0.529

Length of Stay 1R Decisions and Error Rates on rebinned risk groups

```
los_rebin1R <- rebin_1R(x = rebinRisk$LengthOfStay)
los_rebin1R_tot <- sum(as.numeric(los_rebin1R[,5]))
los_rebin1R_err <- sum(as.numeric(los_rebin1R[,7]))
los_rebin1R_rate <- los_rebin1R_err/los_rebin1R_tot
kable(los_rebin1R)
```

	no risk	low risk	medium risk	high risk	total	decision_1R	errors	error_rate
0 or unknown	NA	73758	33235	34791	141784	low risk	68026	0.48
1 day	NA	3602	1435	1599	6636	low risk	3034	0.457
2 days	NA	342	166	195	703	low risk	361	0.514
3 days	NA	119	65	70	254	low risk	135	0.531
4 days	NA	61	35	34	130	low risk	69	0.531
5 days	NA	23	30	18	71	medium risk	41	0.577
6 days	NA	15	3	7	25	low risk	10	0.4
1- 2 weeks	NA	54	25	62	141	high risk	79	0.56
2- 4 weeks	NA	30	17	35	82	high risk	47	0.573
4- 8 weeks	NA	40	20	50	110	high risk	60	0.545
8-12 weeks	NA	0	0	3	3	high risk	0	0
12-26 weeks	NA	2	1	0	3	low risk	1	0.333
26+ weeks	NA	0	1	2	3	high risk	1	0.333

Comparison of 1R rules on rebinned risk groups

The rebinned Primary Condition Group 1R rule has 71750 errors on 149945 observations, for an error rate of 0.48.
The rebinned Charlson Index 1R rule has 71789 errors on 149945 observations, for an error rate of 0.48.
The rebinned Length of Stay 1R rule has 71864 errors on 149945 observations, for an error rate of 0.48.

Each field now has a lower error rate of roughly 48%, which is better since now any 1R rule chosen is better (where as previously, the lowest error rate was 61%). With this set of bins on the risk level group, the Primary Condition Group had a fractionally better error rate of 47.85% error (compared to Charlson Index error rate of 47.88% and Length Of Stay error rate of 47.93%).

Appendix

Risk Group Counts

graphical inspection function

```
risk_group_counts <- ggplot(VlookupSim, aes(x = Risk_Level_label)) +
   geom_text(aes(label = ..count..), stat = "count", vjust = -0.25) +
   geom_bar()

rebin_risk_group_counts <- ggplot(rebinRisk, aes(x = Risk_Level_label)) +
   geom_text(aes(label = ..count..), stat = "count", vjust = -0.25) +
   geom_bar()
```

```
risk_group_counts +
  facet_wrap(~PrimaryConditionGroup)

rebin_risk_group_counts  +
  facet_wrap(~PrimaryConditionGroup)
```

```
risk_group_counts +
  facet_wrap(~CharlsonIndex)

rebin_risk_group_counts +
  facet_wrap(~CharlsonIndex)
```

```
risk_group_counts +
  facet_wrap(~LengthOfStay)

rebin_risk_group_counts +
  facet_wrap(~LengthOfStay)
```