Credit Logistic Regression

Lei Deng, ti7597, STAT6620

May 18, 2017

## Step 1 - Colleting data

Read the data to the R

# load the necessary library  
library(Amelia)

## Loading required package: Rcpp

## ##   
## ## Amelia II: Multiple Imputation  
## ## (Version 1.7.4, built: 2015-12-05)  
## ## Copyright (C) 2005-2017 James Honaker, Gary King and Matthew Blackwell  
## ## Refer to http://gking.harvard.edu/amelia/ for more information  
## ##

library(ROCR)

## Loading required package: gplots

##   
## Attaching package: 'gplots'

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

credit <- read.csv("credit.csv")

## Step 2 - Exploring and preparing the data

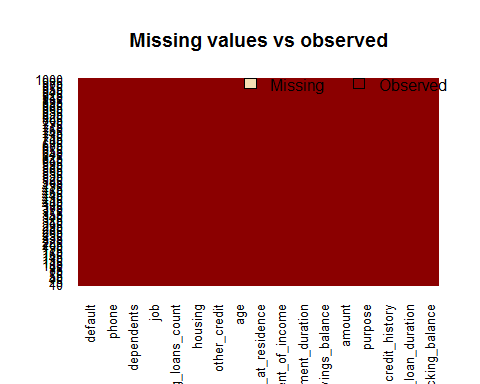
# examine the launch data  
str(credit)

## 'data.frame': 1000 obs. of 17 variables:  
## $ checking\_balance : Factor w/ 4 levels "< 0 DM","> 200 DM",..: 1 3 4 1 1 4 4 3 4 3 ...  
## $ months\_loan\_duration: int 6 48 12 42 24 36 24 36 12 30 ...  
## $ credit\_history : Factor w/ 5 levels "critical","good",..: 1 2 1 2 4 2 2 2 2 1 ...  
## $ purpose : Factor w/ 6 levels "business","car",..: 5 5 4 5 2 4 5 2 5 2 ...  
## $ amount : int 1169 5951 2096 7882 4870 9055 2835 6948 3059 5234 ...  
## $ savings\_balance : Factor w/ 5 levels "< 100 DM","> 1000 DM",..: 5 1 1 1 1 5 4 1 2 1 ...  
## $ employment\_duration : Factor w/ 5 levels "< 1 year","> 7 years",..: 2 3 4 4 3 3 2 3 4 5 ...  
## $ percent\_of\_income : int 4 2 2 2 3 2 3 2 2 4 ...  
## $ years\_at\_residence : int 4 2 3 4 4 4 4 2 4 2 ...  
## $ age : int 67 22 49 45 53 35 53 35 61 28 ...  
## $ other\_credit : Factor w/ 3 levels "bank","none",..: 2 2 2 2 2 2 2 2 2 2 ...  
## $ housing : Factor w/ 3 levels "other","own",..: 2 2 2 1 1 1 2 3 2 2 ...  
## $ existing\_loans\_count: int 2 1 1 1 2 1 1 1 1 2 ...  
## $ job : Factor w/ 4 levels "management","skilled",..: 2 2 4 2 2 4 2 1 4 1 ...  
## $ dependents : int 1 1 2 2 2 2 1 1 1 1 ...  
## $ phone : Factor w/ 2 levels "no","yes": 2 1 1 1 1 2 1 2 1 1 ...  
## $ default : Factor w/ 2 levels "no","yes": 1 2 1 1 2 1 1 1 1 2 ...

# logisitic regression  
  
# set up trainning and test data sets  
  
indx = sample(1:nrow(credit), as.integer(0.9\*nrow(credit)))  
indx

## [1] 263 401 20 967 641 451 709 699 962 570 111 248 571 974  
## [15] 127 886 829 185 892 31 387 475 357 906 691 663 396 409  
## [29] 993 725 317 604 381 969 620 840 128 900 79 533 146 44  
## [43] 247 861 173 167 497 729 761 104 286 67 466 981 812 672  
## [57] 738 788 878 471 7 307 125 319 797 834 283 715 464 806  
## [71] 63 169 265 724 39 105 84 176 46 66 876 750 355 339  
## [85] 100 435 2 579 655 109 284 161 139 290 142 956 29 712  
## [99] 133 516 706 894 698 253 377 681 580 17 214 708 15 921  
## [113] 957 920 763 589 572 771 360 336 789 353 512 187 849 195  
## [127] 966 600 262 234 610 623 346 243 112 514 481 524 509 21  
## [141] 350 461 791 551 238 746 152 445 642 298 721 452 51 833  
## [155] 447 847 374 97 359 825 668 320 308 293 667 904 926 562  
## [169] 856 897 741 397 755 441 488 717 707 294 277 807 337 857  
## [183] 292 354 743 414 785 804 494 915 827 941 573 181 786 587  
## [197] 227 342 408 569 944 138 64 246 62 945 625 478 627 383  
## [211] 679 987 267 585 382 885 220 705 798 433 12 5 484 814  
## [225] 132 323 45 869 588 343 349 576 429 697 953 578 154 619  
## [239] 773 334 266 204 196 448 41 546 213 157 723 808 762 436  
## [253] 548 412 306 482 385 6 487 276 52 845 822 507 970 333  
## [267] 784 268 758 403 596 49 362 225 875 88 656 70 882 470  
## [281] 658 465 982 733 837 170 237 614 689 534 288 937 130 87  
## [295] 410 99 287 609 922 711 296 802 257 224 297 459 754 83  
## [309] 1000 986 23 836 239 677 549 914 700 108 649 888 965 249  
## [323] 212 918 898 647 477 749 954 416 629 527 200 19 561 134  
## [337] 76 842 386 688 65 795 874 769 835 368 768 36 446 879  
## [351] 424 60 538 678 485 858 796 275 889 515 809 838 818 702  
## [365] 787 605 486 910 190 543 612 172 887 730 933 820 722 450  
## [379] 145 863 372 158 325 547 972 662 940 309 348 137 984 565  
## [393] 828 973 419 299 976 928 114 3 473 654 38 537 583 676  
## [407] 648 369 162 156 775 476 989 198 748 903 522 215 261 826  
## [421] 281 960 390 713 326 767 73 131 997 988 592 909 95 618  
## [435] 222 498 777 832 810 356 710 765 599 535 801 351 643 830  
## [449] 597 22 314 864 780 819 907 331 77 122 616 783 197 121  
## [463] 322 149 859 116 530 236 232 463 813 329 831 694 186 540  
## [477] 975 209 404 126 318 872 640 61 160 123 81 995 233 443  
## [491] 979 384 392 943 69 151 395 218 949 739 11 631 159 143  
## [505] 636 201 696 598 947 793 639 703 686 432 999 10 821 428  
## [519] 684 240 942 324 594 380 431 615 532 462 202 646 273 442  
## [533] 673 80 417 977 552 939 78 693 223 264 687 189 413 68  
## [547] 35 399 206 665 675 171 932 680 628 303 499 525 526 635  
## [561] 16 259 998 295 352 411 129 8 550 719 347 626 50 474  
## [575] 18 744 500 492 272 745 430 584 913 366 40 208 782 963  
## [589] 980 692 911 56 43 313 444 732 258 231 752 365 803 92  
## [603] 47 53 848 75 144 772 207 86 601 790 871 251 510 501  
## [617] 245 407 210 520 657 536 901 148 107 567 611 338 528 992  
## [631] 489 48 370 493 235 269 542 301 135 521 304 606 671 155  
## [645] 978 457 895 964 766 846 91 948 182 496 760 426 345 508  
## [659] 315 865 434 90 704 252 422 335 800 300 344 229 115 670  
## [673] 634 843 256 439 453 529 118 582 363 884 244 633 776 950  
## [687] 378 564 379 373 42 815 557 458 581 591 923 958 539 188  
## [701] 736 716 305 563 608 523 72 194 490 651 991 955 734 24  
## [715] 211 844 799 420 241 994 747 163 682 103 645 935 340 659  
## [729] 310 221 37 823 781 853 140 203 899 388 255 756 728 867  
## [743] 312 737 469 454 794 89 136 230 556 983 358 147 192 285  
## [757] 555 389 4 519 817 85 860 394 260 851 505 449 364 168  
## [771] 792 622 102 28 929 332 495 726 757 577 720 55 479 985  
## [785] 33 912 120 502 574 664 367 862 718 660 893 666 852 934  
## [799] 406 925 504 714 361 931 58 575 890 425 57 418 177 1  
## [813] 936 685 553 805 455 952 855 217 468 254 165 153 150 59  
## [827] 946 271 905 531 9 164 607 26 184 778 811 30 870 193  
## [841] 480 880 279 883 779 316 328 554 951 854 731 873 178 518  
## [855] 14 816 398 219 180 250 742 841 603 740 908 896 632 968  
## [869] 13 483 566 405 101 282 113 590 174 124 166 291 727 321  
## [883] 690 226 94 669 674 34 866 191 661 919 558 438 274 280  
## [897] 938 824 376 637

credit\_train = credit[indx,]  
credit\_test = credit[-indx,]  
  
credit\_train\_labels = credit[indx,17]  
credit\_test\_labels = credit[-indx,17]   
  
# Check if there are any missing values  
  
missmap(credit, main = "Missing values vs observed")



# number of missing values in each column  
  
sapply(credit,function(x) sum(is.na(x)))

## checking\_balance months\_loan\_duration credit\_history   
## 0 0 0   
## purpose amount savings\_balance   
## 0 0 0   
## employment\_duration percent\_of\_income years\_at\_residence   
## 0 0 0   
## age other\_credit housing   
## 0 0 0   
## existing\_loans\_count job dependents   
## 0 0 0   
## phone default   
## 0 0

# number of unique values in each column  
  
sapply(credit, function(x) length(unique(x)))

## checking\_balance months\_loan\_duration credit\_history   
## 4 33 5   
## purpose amount savings\_balance   
## 6 921 5   
## employment\_duration percent\_of\_income years\_at\_residence   
## 5 4 4   
## age other\_credit housing   
## 53 3 3   
## existing\_loans\_count job dependents   
## 4 4 2   
## phone default   
## 2 2

## Step3 - Training a model on the data

# fit the logistic regression model, with all predictor variables  
  
model <- glm(default ~.,family=binomial(link='logit'),data=credit\_train)  
model

##   
## Call: glm(formula = default ~ ., family = binomial(link = "logit"),   
## data = credit\_train)  
##   
## Coefficients:  
## (Intercept) checking\_balance> 200 DM   
## -1.119e+00 -9.669e-01   
## checking\_balance1 - 200 DM checking\_balanceunknown   
## -3.622e-01 -1.706e+00   
## months\_loan\_duration credit\_historygood   
## 3.305e-02 9.526e-01   
## credit\_historyperfect credit\_historypoor   
## 1.213e+00 7.580e-01   
## credit\_historyvery good purposecar   
## 1.309e+00 1.266e-01   
## purposecar0 purposeeducation   
## -2.308e+00 5.139e-01   
## purposefurniture/appliances purposerenovations   
## -3.248e-01 4.476e-01   
## amount savings\_balance> 1000 DM   
## 7.435e-05 -9.860e-01   
## savings\_balance100 - 500 DM savings\_balance500 - 1000 DM   
## -1.396e-01 -5.926e-01   
## savings\_balanceunknown employment\_duration> 7 years   
## -1.146e+00 -5.251e-01   
## employment\_duration1 - 4 years employment\_duration4 - 7 years   
## -3.927e-01 -9.950e-01   
## employment\_durationunemployed percent\_of\_income   
## 1.590e-02 2.460e-01   
## years\_at\_residence age   
## 4.691e-02 -1.577e-02   
## other\_creditnone other\_creditstore   
## -7.985e-01 -3.075e-01   
## housingown housingrent   
## -2.478e-01 1.866e-01   
## existing\_loans\_count jobskilled   
## 3.637e-01 1.329e-01   
## jobunemployed jobunskilled   
## -3.021e-01 -2.136e-01   
## dependents phoneyes   
## -1.717e-02 -1.958e-01   
##   
## Degrees of Freedom: 899 Total (i.e. Null); 864 Residual  
## Null Deviance: 1098   
## Residual Deviance: 843.6 AIC: 915.6

summary(model)

##   
## Call:  
## glm(formula = default ~ ., family = binomial(link = "logit"),   
## data = credit\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.9501 -0.7540 -0.3980 0.7841 2.5454   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.119e+00 9.684e-01 -1.156 0.247807   
## checking\_balance> 200 DM -9.669e-01 3.861e-01 -2.504 0.012273 \*   
## checking\_balance1 - 200 DM -3.622e-01 2.174e-01 -1.666 0.095652 .   
## checking\_balanceunknown -1.706e+00 2.379e-01 -7.171 7.45e-13 \*\*\*  
## months\_loan\_duration 3.305e-02 9.433e-03 3.503 0.000459 \*\*\*  
## credit\_historygood 9.526e-01 2.671e-01 3.566 0.000362 \*\*\*  
## credit\_historyperfect 1.213e+00 4.426e-01 2.741 0.006130 \*\*   
## credit\_historypoor 7.580e-01 3.455e-01 2.194 0.028261 \*   
## credit\_historyvery good 1.309e+00 4.397e-01 2.977 0.002908 \*\*   
## purposecar 1.266e-01 3.291e-01 0.385 0.700523   
## purposecar0 -2.308e+00 1.194e+00 -1.934 0.053134 .   
## purposeeducation 5.139e-01 4.541e-01 1.132 0.257777   
## purposefurniture/appliances -3.248e-01 3.247e-01 -1.000 0.317216   
## purposerenovations 4.476e-01 5.884e-01 0.761 0.446803   
## amount 7.435e-05 4.489e-05 1.656 0.097671 .   
## savings\_balance> 1000 DM -9.860e-01 5.107e-01 -1.931 0.053513 .   
## savings\_balance100 - 500 DM -1.396e-01 2.788e-01 -0.501 0.616536   
## savings\_balance500 - 1000 DM -5.926e-01 4.336e-01 -1.367 0.171690   
## savings\_balanceunknown -1.146e+00 2.777e-01 -4.126 3.69e-05 \*\*\*  
## employment\_duration> 7 years -5.251e-01 2.961e-01 -1.773 0.076198 .   
## employment\_duration1 - 4 years -3.927e-01 2.403e-01 -1.634 0.102158   
## employment\_duration4 - 7 years -9.950e-01 3.053e-01 -3.259 0.001118 \*\*   
## employment\_durationunemployed 1.590e-02 4.260e-01 0.037 0.970221   
## percent\_of\_income 2.460e-01 8.851e-02 2.779 0.005445 \*\*   
## years\_at\_residence 4.691e-02 8.894e-02 0.527 0.597901   
## age -1.577e-02 9.336e-03 -1.689 0.091230 .   
## other\_creditnone -7.985e-01 2.465e-01 -3.239 0.001199 \*\*   
## other\_creditstore -3.075e-01 4.370e-01 -0.704 0.481643   
## housingown -2.478e-01 3.015e-01 -0.822 0.411153   
## housingrent 1.866e-01 3.458e-01 0.540 0.589440   
## existing\_loans\_count 3.637e-01 1.959e-01 1.857 0.063326 .   
## jobskilled 1.329e-01 2.939e-01 0.452 0.651153   
## jobunemployed -3.021e-01 6.743e-01 -0.448 0.654167   
## jobunskilled -2.136e-01 3.607e-01 -0.592 0.553736   
## dependents -1.717e-02 2.495e-01 -0.069 0.945118   
## phoneyes -1.958e-01 2.032e-01 -0.963 0.335393   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1097.86 on 899 degrees of freedom  
## Residual deviance: 843.57 on 864 degrees of freedom  
## AIC: 915.57  
##   
## Number of Fisher Scoring iterations: 5

anova(model, test="Chisq")

## Analysis of Deviance Table  
##   
## Model: binomial, link: logit  
##   
## Response: default  
##   
## Terms added sequentially (first to last)  
##   
##   
## Df Deviance Resid. Df Resid. Dev Pr(>Chi)   
## NULL 899 1097.86   
## checking\_balance 3 114.252 896 983.60 < 2.2e-16 \*\*\*  
## months\_loan\_duration 1 38.584 895 945.02 5.246e-10 \*\*\*  
## credit\_history 4 23.225 891 921.79 0.0001142 \*\*\*  
## purpose 5 9.798 886 912.00 0.0811561 .   
## amount 1 0.007 885 911.99 0.9315398   
## savings\_balance 4 23.289 881 888.70 0.0001108 \*\*\*  
## employment\_duration 4 11.902 877 876.80 0.0180974 \*   
## percent\_of\_income 1 7.348 876 869.45 0.0067128 \*\*   
## years\_at\_residence 1 0.615 875 868.83 0.4327736   
## age 1 3.654 874 865.18 0.0559367 .   
## other\_credit 2 10.821 872 854.36 0.0044702 \*\*   
## housing 2 4.158 870 850.20 0.1250388   
## existing\_loans\_count 1 3.279 869 846.92 0.0701717 .   
## job 3 2.418 866 844.50 0.4902619   
## dependents 1 0.005 865 844.50 0.9462279   
## phone 1 0.933 864 843.57 0.3341736   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

# drop the insignificant predictors, alpha = 0.10  
  
model <- glm(default ~ checking\_balance + months\_loan\_duration + credit\_history + percent\_of\_income + age,family=binomial(link='logit'),data=credit\_train)  
model

##   
## Call: glm(formula = default ~ checking\_balance + months\_loan\_duration +   
## credit\_history + percent\_of\_income + age, family = binomial(link = "logit"),   
## data = credit\_train)  
##   
## Coefficients:  
## (Intercept) checking\_balance> 200 DM   
## -1.42901 -1.10572   
## checking\_balance1 - 200 DM checking\_balanceunknown   
## -0.45149 -1.86236   
## months\_loan\_duration credit\_historygood   
## 0.03649 0.58632   
## credit\_historyperfect credit\_historypoor   
## 1.50872 0.63716   
## credit\_historyvery good percent\_of\_income   
## 1.31977 0.16181   
## age   
## -0.01314   
##   
## Degrees of Freedom: 899 Total (i.e. Null); 889 Residual  
## Null Deviance: 1098   
## Residual Deviance: 914.2 AIC: 936.2

summary(model)

##   
## Call:  
## glm(formula = default ~ checking\_balance + months\_loan\_duration +   
## credit\_history + percent\_of\_income + age, family = binomial(link = "logit"),   
## data = credit\_train)  
##   
## Deviance Residuals:   
## Min 1Q Median 3Q Max   
## -1.8197 -0.8064 -0.4763 0.9310 2.3747   
##   
## Coefficients:  
## Estimate Std. Error z value Pr(>|z|)   
## (Intercept) -1.429005 0.436360 -3.275 0.001057 \*\*   
## checking\_balance> 200 DM -1.105717 0.361745 -3.057 0.002238 \*\*   
## checking\_balance1 - 200 DM -0.451486 0.196425 -2.299 0.021533 \*   
## checking\_balanceunknown -1.862360 0.218350 -8.529 < 2e-16 \*\*\*  
## months\_loan\_duration 0.036486 0.006720 5.430 5.65e-08 \*\*\*  
## credit\_historygood 0.586324 0.208274 2.815 0.004875 \*\*   
## credit\_historyperfect 1.508715 0.412864 3.654 0.000258 \*\*\*  
## credit\_historypoor 0.637159 0.317131 2.009 0.044523 \*   
## credit\_historyvery good 1.319767 0.379567 3.477 0.000507 \*\*\*  
## percent\_of\_income 0.161811 0.074459 2.173 0.029770 \*   
## age -0.013141 0.007501 -1.752 0.079806 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1  
##   
## (Dispersion parameter for binomial family taken to be 1)  
##   
## Null deviance: 1097.86 on 899 degrees of freedom  
## Residual deviance: 914.23 on 889 degrees of freedom  
## AIC: 936.23  
##   
## Number of Fisher Scoring iterations: 4

anova(model, test="Chisq")

## Analysis of Deviance Table  
##   
## Model: binomial, link: logit  
##   
## Response: default  
##   
## Terms added sequentially (first to last)  
##   
##   
## Df Deviance Resid. Df Resid. Dev Pr(>Chi)   
## NULL 899 1097.86   
## checking\_balance 3 114.252 896 983.60 < 2.2e-16 \*\*\*  
## months\_loan\_duration 1 38.584 895 945.02 5.246e-10 \*\*\*  
## credit\_history 4 23.225 891 921.79 0.0001142 \*\*\*  
## percent\_of\_income 1 4.425 890 917.37 0.0354112 \*   
## age 1 3.141 889 914.23 0.0763555 .   
## ---  
## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

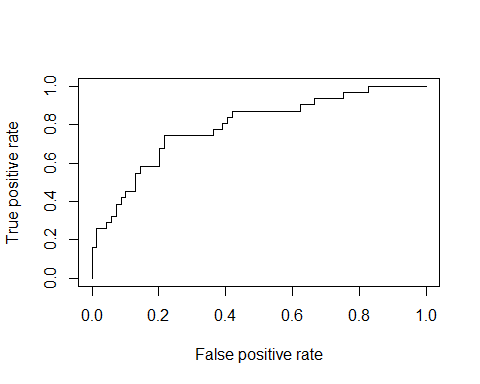
## step4 - Evaluating model performance

# check Accuracy  
  
fitted.results <- predict(model,newdata=credit\_test,type='response')  
fitted.results <- ifelse(fitted.results > 0.5,1,0)  
  
misClasificError <- mean(fitted.results != credit\_test$default)  
print(paste('Accuracy',1-misClasificError))

## [1] "Accuracy 0"

## step5 - Improving model performance

# Because this data set is so small, it is possible that the test data set  
# does not contain both 0 and 1 values. If this happens the code will not  
# run. And since the test data set is so small the ROC is not useful here  
# but the code is provided.  
  
p <- predict(model, newdata=credit\_test, type="response")  
pr <- prediction(p, credit\_test$default)  
prf <- performance(pr, measure = "tpr", x.measure = "fpr")  
plot(prf)



auc <- performance(pr, measure = "auc")  
auc <- auc@y.values[[1]]  
auc

## [1] 0.7900888