Wine Qaulity Analysis

Narendra Bandi

08/04/2021

Wine quality prediction using Multinomial LOgistic regression, Decision Tree and Random Forest methods. Quality is assigned discrete values from 1 to 10.

```
red_data <- read_delim("winequality-red.csv",delim  = ";")
dim(red_data)
```

Read data from file, the delimiter is ";"

```
## [1] 1599   12
```

```
knitr::kable(head(red_data, 10), align = "c")
```

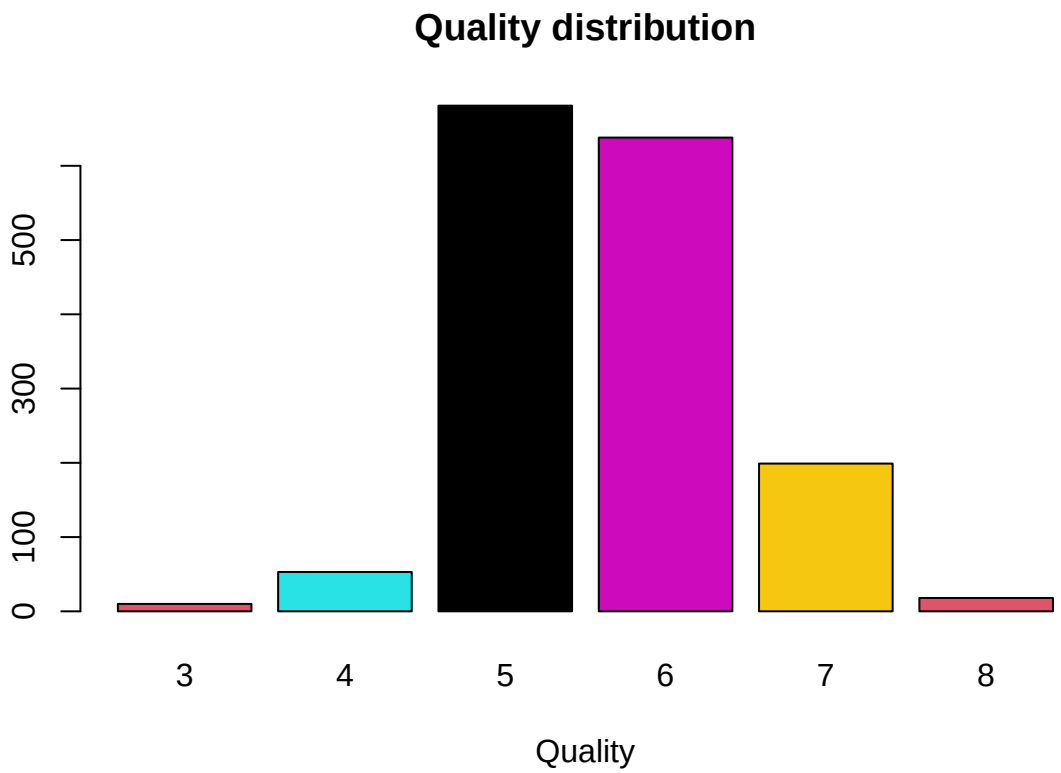
fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
7.4	0.70	0.00	1.9	0.076	11	34	0.9978	3.51	0.56	9.4	5
7.8	0.88	0.00	2.6	0.098	25	67	0.9968	3.20	0.68	9.8	5
7.8	0.76	0.04	2.3	0.092	15	54	0.9970	3.26	0.65	9.8	5
11.2	0.28	0.56	1.9	0.075	17	60	0.9980	3.16	0.58	9.8	6
7.4	0.70	0.00	1.9	0.076	11	34	0.9978	3.51	0.56	9.4	5
7.4	0.66	0.00	1.8	0.075	13	40	0.9978	3.51	0.56	9.4	5
7.9	0.60	0.06	1.6	0.069	15	59	0.9964	3.30	0.46	9.4	5
7.3	0.65	0.00	1.2	0.065	15	21	0.9946	3.39	0.47	10.0	7
7.8	0.58	0.02	2.0	0.073	9	18	0.9968	3.36	0.57	9.5	7
7.5	0.50	0.36	6.1	0.071	17	102	0.9978	3.35	0.80	10.5	5

```r
quality.tbl <- table(red_data$quality)
knitr::kable(quality.tbl, align = "c")
```

Distribution of quality:

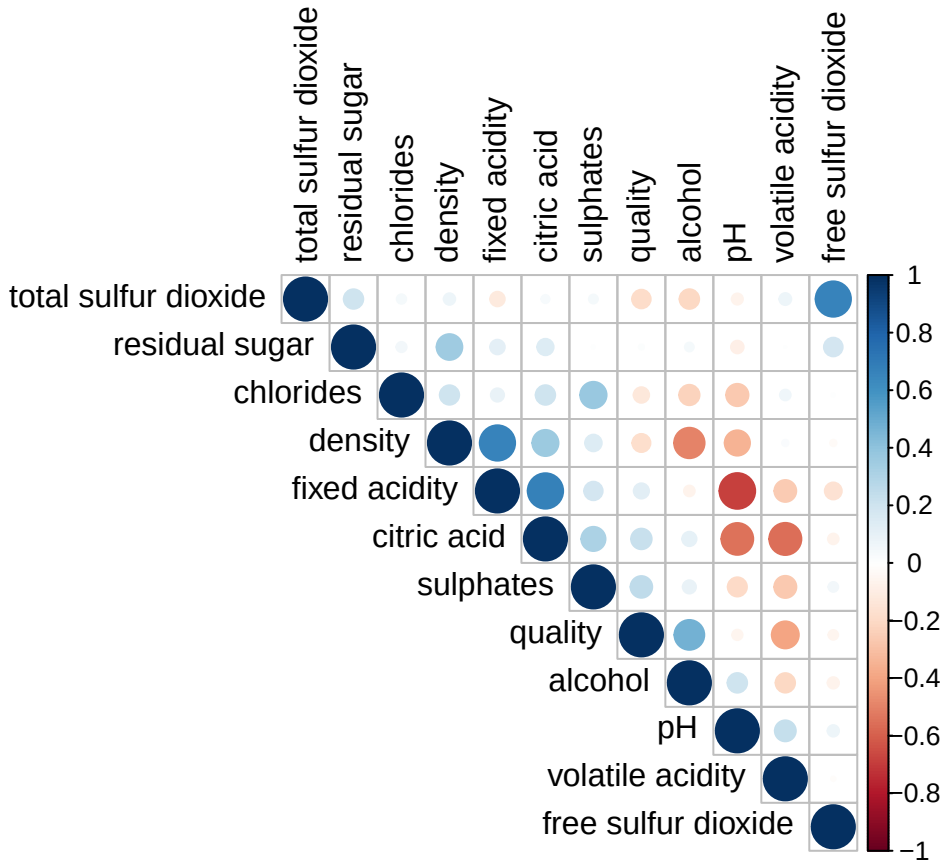
Var1	Freq
3	10
4	53
5	681
6	638
7	199
8	18

```r
barplot(quality.tbl,horiz = F,col = quality.tbl,  xlab = "Quality", main= "Quality distr
```



```
corr_mtrx <- cor(red_data, method = "pearson")

corrplot(corr_mtrx, type = "upper",order = "AOE",
         tl.col = "black", tl.srt = 90)
```



Linear correlation :

```
knitr::kable(round(corr_mtrx,2), align = "c")
```

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
fixed acidity	1.00	-0.26	0.67	0.11	0.09	-0.15	-0.11	0.67	-0.68	0.18	-0.06	0.12
volatile acidity	-0.26	1.00	-0.55	0.00	0.06	-0.01	0.08	0.02	0.23	-0.26	-0.20	-0.39
citric acid	0.67	-0.55	1.00	0.14	0.20	-0.06	0.04	0.36	-0.54	0.31	0.11	0.23
residual sugar	0.11	0.00	0.14	1.00	0.06	0.19	0.20	0.36	-0.09	0.01	0.04	0.01

	fixed acidity	volatile acidity	citric acid	residual sugar	chlorides	free sulfur dioxide	total sulfur dioxide	density	pH	sulphates	alcohol	quality
chlorides	0.09	0.06	0.20	0.06	1.00	0.01	0.05	0.20	-0.27	0.37	-0.22	-0.13
free sulfur dioxide	-0.15	-0.01	-0.06	0.19	0.01	1.00	0.67	-0.02	0.07	0.05	-0.07	-0.05
total sulfur dioxide	-0.11	0.08	0.04	0.20	0.05	0.67	1.00	0.07	-0.07	0.04	-0.21	-0.19
density	0.67	0.02	0.36	0.36	0.20	-0.02	0.07	1.00	-0.34	0.15	-0.50	-0.17
pH	-0.68	0.23	-0.54	-0.09	-0.27	0.07	-0.07	-0.34	1.00	-0.20	0.21	-0.06
sulphates	0.18	-0.26	0.31	0.01	0.37	0.05	0.04	0.15	-0.20	1.00	0.09	0.25
alcohol	-0.06	-0.20	0.11	0.04	-0.22	-0.07	-0.21	-0.50	0.21	0.09	1.00	0.48
quality	0.12	-0.39	0.23	0.01	-0.13	-0.05	-0.19	-0.17	-0.06	0.25	0.48	1.00

```
red_data$quality <-  factor(red_data$quality, levels=c(1:10), ordered=TRUE)
```

Split data into Train and Testing using stratification sampling

```
row_idx <- createDataPartition(red_data$quality,p = 0.25,list = F)
```

```
## Warning in createDataPartition(red_data$quality, p = 0.25, list = F): Some
## classes have no records ( 1, 2, 9, 10 ) and these will be ignored
```

```
train.df <- red_data[-row_idx[,1],]
validation.df <- red_data[row_idx[,1],]
```

```
model_ord_log <- polr(quality ~ ., data = train.df, Hess=TRUE)
summary(model_ord_log)
```

Ordered Multinomial Logistc Regression approach (quality is ordered)

```
## Call:
## polr(formula = quality ~ ., data = train.df, Hess = TRUE)
##
## Coefficients:
##                           Value Std. Error t value
## `fixed acidity`         0.13289   0.059757  2.2238
## `volatile acidity`     -3.42224   0.466289 -7.3393
## `citric acid`          -0.97228   0.532698 -1.8252
## `residual sugar`        0.08656   0.042728  2.0258
## chlorides              -4.63140   1.648791 -2.8090
## `free sulfur dioxide`   0.01578   0.007693  2.0516
## `total sulfur dioxide` -0.01006   0.002670 -3.7671
## density                 1.20109   3.524506  0.3408
## pH                     -0.77153   0.572354 -1.3480
## sulphates               2.98168   0.415477  7.1765
## alcohol                 0.94088   0.069836 13.4726
##
## Intercepts:
##      Value          Std. Error    t value
## 1|2  -3.773300e+00  3.584100e+00 -1.052800e+00
## 2|3  -2.897000e+00  3.559200e+00 -8.139000e-01
## 3|4   3.029800e+00  3.467400e+00  8.738000e-01
## 4|5   4.983600e+00  3.466700e+00  1.437500e+00
## 5|6   8.738500e+00  3.469900e+00  2.518400e+00
## 6|7   1.163160e+01  3.475500e+00  3.346700e+00
## 7|8   1.467460e+01  3.488000e+00  4.207200e+00
## 8|9   1.991556e+11  3.488000e+00  5.709702e+10
## 9|10  1.991556e+11  3.488000e+00  5.709702e+10
##
## Residual Deviance: 2278.139
## AIC: 2318.139
```

predict the quality for test data.

```
predicted_quality <- predict(model_ord_log, newdata = validation.df)
```

```
confusion_mtrx <- confusionMatrix(predicted_quality, validation.df$quality)
confusion_mtrx
```

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   2   3   4   5   6   7   8   9  10
##         1    0   0   0   0   0   0   0   0   0   0
```

```
##         2    0   0   0   0   0   0   0   0   0   0
##         3    0   0   0   0   0   0   0   0   0   0
##         4    0   0   0   0   1   0   0   0   0   0
##         5    0   0   3  10 125  56   4   0   0   0
##         6    0   0   0   4  43  92  34   1   0   0
##         7    0   0   0   0   1  12  12   4   0   0
##         8    0   0   0   0   1   0   0   0   0   0
##         9    0   0   0   0   0   0   0   0   0   0
##         10   0   0   0   0   0   0   0   0   0   0
##
## Overall Statistics
##
##                Accuracy : 0.5682
##                  95% CI : (0.5183, 0.6172)
##     No Information Rate : 0.4243
##     P-Value [Acc > NIR] : 4.405e-09
##
##                   Kappa : 0.2934
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 4 Class: 5 Class: 6
## Sensitivity                NA       NA 0.000000 0.000000   0.7310   0.5750
## Specificity                 1        1 1.000000 0.997429   0.6853   0.6626
## Pos Pred Value             NA       NA      NaN 0.000000   0.6313   0.5287
## Neg Pred Value             NA       NA 0.992556 0.965174   0.7756   0.7031
## Prevalence                  0        0 0.007444 0.034739   0.4243   0.3970
## Detection Rate              0        0 0.000000 0.000000   0.3102   0.2283
## Detection Prevalence        0        0 0.000000 0.002481   0.4913   0.4318
## Balanced Accuracy          NA       NA 0.500000 0.498715   0.7082   0.6188
##                      Class: 7 Class: 8 Class: 9 Class: 10
## Sensitivity           0.24000 0.000000       NA        NA
## Specificity           0.95184 0.997487        1         1
## Pos Pred Value        0.41379 0.000000       NA        NA
## Neg Pred Value        0.89840 0.987562       NA        NA
## Prevalence            0.12407 0.012407        0         0
## Detection Rate        0.02978 0.000000        0         0
## Detection Prevalence  0.07196 0.002481        0         0
## Balanced Accuracy     0.59592 0.498744       NA        NA
```

```
model.dt <- rpart::rpart(quality ~., data= train.df)
predicted.dt <- predict(model.dt, validation.df, type = "class")
confusion.dt <- confusionMatrix(predicted.dt, validation.df$quality)
confusion.dt
```

Decision Tree approach

```
## Confusion Matrix and Statistics
##
##           Reference
## Prediction   1   2   3   4   5   6   7   8   9  10
##         1    0   0   0   0   0   0   0   0   0   0
##         2    0   0   0   0   0   0   0   0   0   0
##         3    0   0   0   0   0   0   0   0   0   0
##         4    0   0   0   0   0   0   0   0   0   0
##         5    0   0   3  10 116  53   6   0   0   0
##         6    0   0   0   4  52  89  26   0   0   0
##         7    0   0   0   0   3  18  18   5   0   0
##         8    0   0   0   0   0   0   0   0   0   0
##         9    0   0   0   0   0   0   0   0   0   0
##         10   0   0   0   0   0   0   0   0   0   0
##
## Overall Statistics
##
##                Accuracy : 0.5533
##                  95% CI : (0.5033, 0.6026)
##     No Information Rate : 0.4243
##     P-Value [Acc > NIR] : 1.281e-07
##
##                   Kappa : 0.2796
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 4 Class: 5 Class: 6
## Sensitivity                NA       NA 0.000000  0.00000   0.6784   0.5563
## Specificity                 1        1 1.000000  1.00000   0.6897   0.6626
## Pos Pred Value             NA       NA      NaN      NaN   0.6170   0.5205
## Neg Pred Value             NA       NA 0.992556  0.96526   0.7442   0.6940
## Prevalence                  0        0 0.007444  0.03474   0.4243   0.3970
## Detection Rate              0        0 0.000000  0.00000   0.2878   0.2208
## Detection Prevalence        0        0 0.000000  0.00000   0.4665   0.4243
```

```
## Balanced Accuracy          NA       NA 0.500000  0.50000   0.6840   0.6094
##                       Class: 7 Class: 8 Class: 9 Class: 10
## Sensitivity            0.36000  0.00000       NA        NA
## Specificity            0.92635  1.00000        1         1
## Pos Pred Value         0.40909      NaN       NA        NA
## Neg Pred Value         0.91086  0.98759       NA        NA
## Prevalence             0.12407  0.01241        0         0
## Detection Rate         0.04467  0.00000        0         0
## Detection Prevalence   0.10918  0.00000        0         0
## Balanced Accuracy      0.64317  0.50000       NA        NA
```

```
train.col.df <- train.df
valid.col.df <- validation.df
col_names <- colnames(train.col.df)
colnames(train.col.df) <-   gsub(" ", "_", col_names)
colnames(valid.col.df) <- gsub(" " , "_", col_names)
```

```
library(randomForest)
train.col.df$quality <- droplevels(train.col.df$quality)
model.rf <- randomForest(quality ~., data= train.col.df, ntree=300, mtry=4, importance=T
```

```
predicted.rf <- predict(model.rf, valid.col.df, type = "class")
predicted.prob.rf <- predict(model.rf, valid.col.df,type="prob")
confusion.rf <- confusionMatrix(predicted.rf, valid.col.df$quality)
```

Random Forest approach

```
## Warning in levels(reference) != levels(data): longer object length is not a
## multiple of shorter object length
```

```
## Warning in confusionMatrix.default(predicted.rf, valid.col.df$quality): Levels
## are not in the same order for reference and data. Refactoring data to match.
```

```
confusion.rf
```

```
## Confusion Matrix and Statistics
##
```

```
##           Reference
## Prediction   1   2   3   4   5   6   7   8   9  10
##         1    0   0   0   0   0   0   0   0   0   0
##         2    0   0   0   0   0   0   0   0   0   0
##         3    0   0   0   1   0   0   0   0   0   0
##         4    0   0   2   0   0   1   0   0   0   0
##         5    0   0   1  10 138  33   4   0   0   0
##         6    0   0   0   2  32 113  28   2   0   0
##         7    0   0   0   1   1  13  18   3   0   0
##         8    0   0   0   0   0   0   0   0   0   0
##         9    0   0   0   0   0   0   0   0   0   0
##         10   0   0   0   0   0   0   0   0   0   0
##
## Overall Statistics
##
##                Accuracy : 0.6675
##                  95% CI : (0.6192, 0.7134)
##     No Information Rate : 0.4243
##     P-Value [Acc > NIR] : < 2.2e-16
##
##                   Kappa : 0.4623
##
##  Mcnemar's Test P-Value : NA
##
## Statistics by Class:
##
##                      Class: 1 Class: 2 Class: 3 Class: 4 Class: 5 Class: 6
## Sensitivity                NA       NA 0.000000 0.000000   0.8070   0.7063
## Specificity                 1        1 0.997500 0.992288   0.7931   0.7366
## Pos Pred Value             NA       NA 0.000000 0.000000   0.7419   0.6384
## Neg Pred Value             NA       NA 0.992537 0.965000   0.8479   0.7920
## Prevalence                  0        0 0.007444 0.034739   0.4243   0.3970
## Detection Rate              0        0 0.000000 0.000000   0.3424   0.2804
## Detection Prevalence        0        0 0.002481 0.007444   0.4615   0.4392
## Balanced Accuracy          NA       NA 0.498750 0.496144   0.8001   0.7214
##                      Class: 7 Class: 8 Class: 9 Class: 10
## Sensitivity           0.36000  0.00000       NA        NA
## Specificity           0.94901  1.00000        1         1
## Pos Pred Value        0.50000      NaN       NA        NA
## Neg Pred Value        0.91281  0.98759       NA        NA
## Prevalence            0.12407  0.01241        0         0
## Detection Rate        0.04467  0.00000        0         0
## Detection Prevalence  0.08933  0.00000        0         0
## Balanced Accuracy     0.65450  0.50000       NA        NA
```