written_report_kap_4_umut

Umut Arslan

2023-02-17

Klassificering/predektion av vin

Jag har ett datasett bestående av rött och vitt vin med diverse "kemiska egenskaper". Min uppgift är att försöka skapa en modell som predikterar/klassificerar en responsvariabel, quality, så bra som möjligt.

Jag har skapat en ny variabel som definerar vad "Excellent" och "Not Excellent" är utifrån responsvariabeln quality. Om quality är större eller lika med 7 så får min nya responsvariabel, qualitydiff, värdet "Excellent" och om quality är lägre än 7 så får qualitydiff värdet "Not Excellent".

Datasettet kommer från: http://archive.ics.uci.edu/ml/datasets/Wine+Quality

Variabler som finns i datat

Vi kommer att använda oss utav alla förklaringsvariabler nedan i en GLM modell för att undersöka vilka av dessa variabler som har ett samband med responsvariabeln, qualitydiff. När/om vi får variabler som ej har ett samband med qualitydiff så kommer vi att ta bort dessa förklaringsvariabler, sedan kommer vi att använda oss utav de X variabler som har ett samband i resterande modeller

- quality - omkodad till qualitydiff, som jag skrev ovan (responsvariabel)
- fixed acidity - most acids involved with wine or fixed or nonvolatile (do not evaporate readily)
- volatile acidity - the amount of acetic acid in wine, which at too high of levels can lead to an unpleasant, vinegar taste
- citric acid - found in small quantities, citric acid can add 'freshness' and flavor to wines
- residual sugar - the amount of sugar remaining after fermentation stops, it's rare to find wines with less than 1 gram/liter and
- chlorides - the amount of salt in the wine
- free sulfur dioxide - the free form of SO2 exists in equilibrium between molecular SO2 (as a dissolved gas) and bisulfite ion; it prevents
- total sulfur dioxide - amount of free and bound forms of S02; in low concentrations, SO2 is mostly undetectable in wine, but at free SO2
- density - the density of water is close to that of water depending on the percent alcohol and sugar content
- pH - describes how acidic or basic a wine is on a scale from 0 (very acidic) to 14 (very basic); most wines are between 3-4 on the
- sulphates - a wine additive which can contribute to sulfur dioxide gas (S02) levels, wich acts as an antimicrobial and
- alcohol - % alcohol content, scale is from 8 - 15 % alcohol content.

Utvärdering av datat

Vi ser att datat är negativt skevt mot "Excellent" där cirka 20% av datat består av just "Excellent" och resterande 80% består av "Not Excellent". Kollar vi på Andeln röda viner som har fått stämpeln "Excellent"

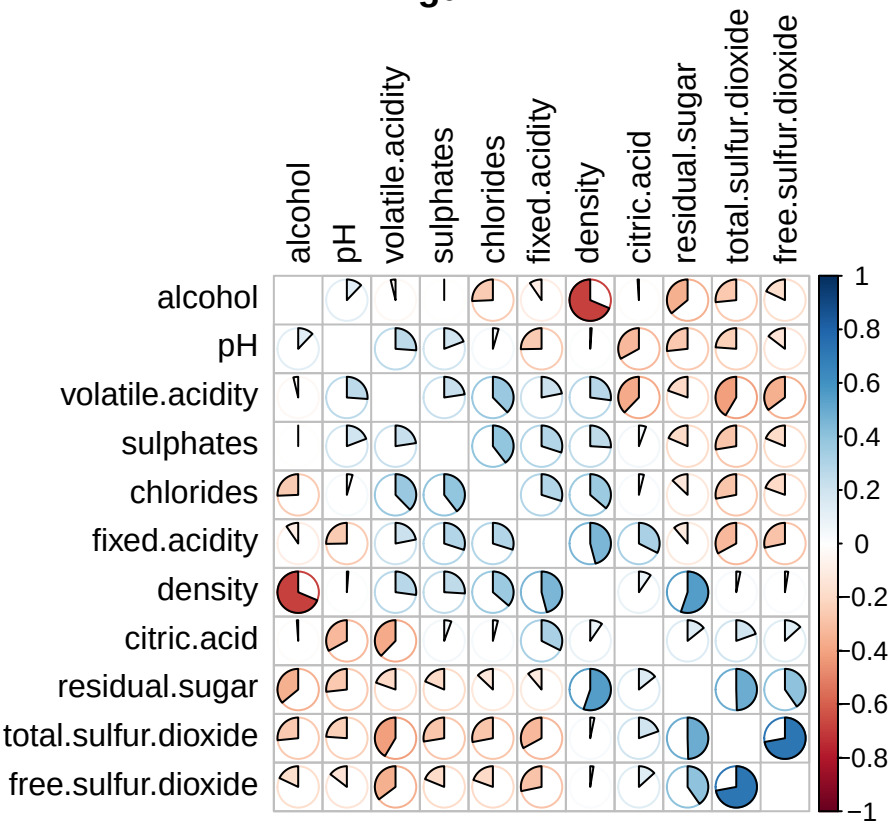
så ser vi att det är 13.5% och andeln vita viner som har fått samma stämpel är drygt 21.6%

Vi undersöker om det finns några NAs, datapunkter som saknas. I detta fall finns det ej några saknade datapunkter. Detta är bra, om det hade varit data som saknades så kan man antigen ta bort alla dessa rader eller använda oss utav "impute" med "knnimpute" eller liknande.

Utifrån det vi har sett i orginaldatat så förväntar jag mig att mina modeller kommer kunna predikera "Not Excellent" rätt oftare än vad modellerna kommer lyckas ha rätt på "Excellent". Detta på grund utav mestadels av orginaldatat består av "Not Excellent".

Kikar vi på Figur 1, korrelationsplotten, så ser vi att en del variabler har rätt så hög korrelation mellan varandra, detta bör även orsaka problem för NaiveBays som antar oberoende.

Figur 1



Modelltestning

Vi kommer att testa fem olika modelltyper.

- Logistisk regression (GLM) - Generalized linear model, där vi anger familjeparametern som en bionomialfördelning för att kunna klassa våran responsvariabel. Det finns ett antal länkfunktioner till familjeparametern, vi definierade ej någon specifik länkfunktion i koden som kommer köras för GLM, därav kommer koden automatiskt antaR "logit" som länk. Med andra ord beräknas log-oddsen.
- LDA - Linear Discriminant Analysis, kortfattat så reducerar den dimensioner, variabler, i ett datasett men samtidigt försöker modellen bibehålla så mycket information som möjligt. Liknar GLM. LDA antar normalfördelning med samma väntevärde och varians för alla klasser, LDA antar även en linjär beslutsgräns. $LDA \sim \mathsf{N}(\mathfrak{t}_k, \sum)$

- QDA - Quadratic Discriminant Analysis, samma som LDA men modellen tillåter att variansen är olika mellan varje klass och tillåter varje klass att ha sin egna kovariansmatris, funktionen är även kvadratisk istället för linjär, därav kan även QDA anta icke-linjära beslutsgränser. $QDA \sim \mathsf{N}(\mathsf{t}_k, \sum_k)$
- NaiveBayes - Baserad på Bayes' Teori, med antagande av oberoende mellan mina variabler. $NaiveBayes \sim \mathsf{N}(\mathsf{t}_k, \sigma_k \perp\!\!\!\perp)$
- kNN - icke-parametrisk algoritm, gör inga antaganden alls. I klassificering så kollar modellen efter de k närmaste grannarna med hjälp av euklidiska avståndet, k väljer jag själv.

Alla modellerna ovan förutom kNN gör diverse antaganden. jag tror ändå att kNN alltid presterar relativt bra utifrån den förutsättningen att kNN inte antar något.

GLM

jag har valt 75% av datat som träningsdata och resterande av datat är valideringsdata Vi börjar med att testa med hela datatsettet, alltså använder vi oss utav qualitydiff som responsvariabel och alla förklaringsvariabler är med i modellen.

Vi har en gräns på 1% signifikansnivå, om någon av förklaringsvariablerna i modellen överstiger denna gräns så kommer vi att plocka bort de variablerna.

När vi summerar GLM modellen så ser vi att tre variabel överstiger min gräns på 1% signifikansnivå, alltså att det ej finns ett samband mellan de variablerna och qualitydiff på 1% signifikans. Vi misslyckas alltså att förkasta nollhypotesen och att coeffisienten är lika med noll, dvs de förklaringsvariablerna som överstiger 1% signifikans har ej "non-zero" effekt på responsvariabeln.

Detta betyder alltså att citric.acid, chlorides och total.sulfur.dioxide ryker från vårat data. kan tilläggas att free.sulfur.dioxide ligger nära gränsen.

Test error om man väljer att klassa all data som majoritetsklassen är med andra ord betyder det att våra modeller vi ska koda i kapitlet nedan bör ha en lägre test error rate än 0.198099 för att vara "bättre" än slumpen.

GLM med "dåliga" variabler borttagna

Vi gör om testet ovan fast vi tar nu bort de "dåliga" variablerna från modellen

Det slutade alltså i en ökning på Test error rate med 0.05% när man tog bort de förklaringsvariabler som hade en signifikansnivå över 1%, detta kommer oavsett bli de slutgiltiga variablerna som ska användas för att jämföra diverse modeller mot varandra.

Vi kommer vidare testa de resterande fyra modeller mot varandra och undersöka vilken av dessa som har bäst "AUC", man kan även använda sig utav måttet "Accuracy", där Accuracy är 1 - Test Error Rate, för att jämföra modellerna. Jag anser att AUC representerar detta data bättre eftersom att datat inte är balancerad mellan "Excellent" och "Not Excellent"

kNN

När det kommer till kNN så behöver vi göra lite annorlunda jämfört med datat ovan. Vi behöver gå in i datat och ta bort filer lite annorlunda jämfört med regressionerna

Nedan testar vi med olika antal grannar, jag testade från 1 till 15 grannar och kom fram till att endast kolla på den närmsta grannen, k = 1, gav mig lägst Test Error rate. Det kan tilläggas att ha k = 1 brukar generellt sett tendera till att göra en "over-fitting" av datat, det är inte optimalt. Vi får se lite senare i labben när jag jämför alla modeller mot varandra, Accuracy samt AUC.

Nedan så använder vi samma k = 1 men vi kör på datat där vi har tagit bort en del variabler som inte var signifikanta

Vi gör även samma sak som ovan fast vi skalar om varians till 1 och standardavvikelsen till 0 så att det euklidiska avståndet blir "jämnare". Vi hade två förklaringsvariabler som hade riktigt hög varians jämfört med resten. Jag döper dessa modeller till KNNZ och KNNZ*2*

Jämför Accuracy och AUC mellan modellerna

Vi kommer att skapa Confusion matriser, en tabell som evaluerar och summerar andeln korrekta och inkorrekt predektioner gjorde av respektive modell, där vi med hjälp av paketet caret kunna beräkna intressanta värden som Accuracy, Sensitivity och Specificity.

- Modellnamn *2* är alltså modellerna där de "dåliga" förklaringsvariablerna är borttagna

```
##          Accuracy Sensitivity Specificity Test Error Rate
## GLM     0.8104052   0.5515152   0.8336968       0.1895948
## GLM*2*  0.8099050   0.5470588   0.8343357       0.1900950
## LDA     0.8119060   0.5467290   0.8436975       0.1880940
## LDA*2*  0.8089045   0.5330189   0.8416340       0.1910955
## QDA     0.7553777   0.4258373   0.9059767       0.2446223
## QDA*2*  0.7943972   0.4797844   0.8660934       0.2056028
## NB      0.7228614   0.3820896   0.8946576       0.2771386
## NB*2*   0.7708854   0.4337607   0.8739386       0.2291146
## KNN     0.8224112   0.5484634   0.8959391       0.1775888
## KNN*2*  0.8359180   0.5867347   0.8967019       0.1640820
## KNNZ    0.8504252   0.6185819   0.9100629       0.1495748
## KNNZ*2* 0.8444222   0.6054591   0.9047619       0.1555778
```

Här ser vi beroende på modell att Accuracy, Sensitivity, Specificity och Test Error Rate ökar/sänks. Jag antog utifrån korrelationsdatat att NaiveBays skulle ha den sämsta predekteringen. Resterande modeller var svårare att förutspå, förvisso har LDA och GLM mycket i likhet när det kommer till antaganden.

Vi ser även att KNNZ har den högsta Accuracy, Sensitivity och lägst Test Error Rate av de alla 12 modellerna vi testade. Beroende på vad målet med klassningen är så kan man alltid justera och vikta om prediktionerna mot exempelvis Sensitivity. Nackdelen då är att Specifity kommer minska, och i följd Accuracy och Test Error Rate.

Sensitivity = andelen "Excellent" som blev klassad rätt (TP / TP + FN)

Specifity = andelen "Not Excellent" som blev klassad rätt

Test Error Rate = Andelen felklassade responsvariabler

Accuracy = 1 - Test Error Rate, dvs andelen rätt klassade responsvariabler

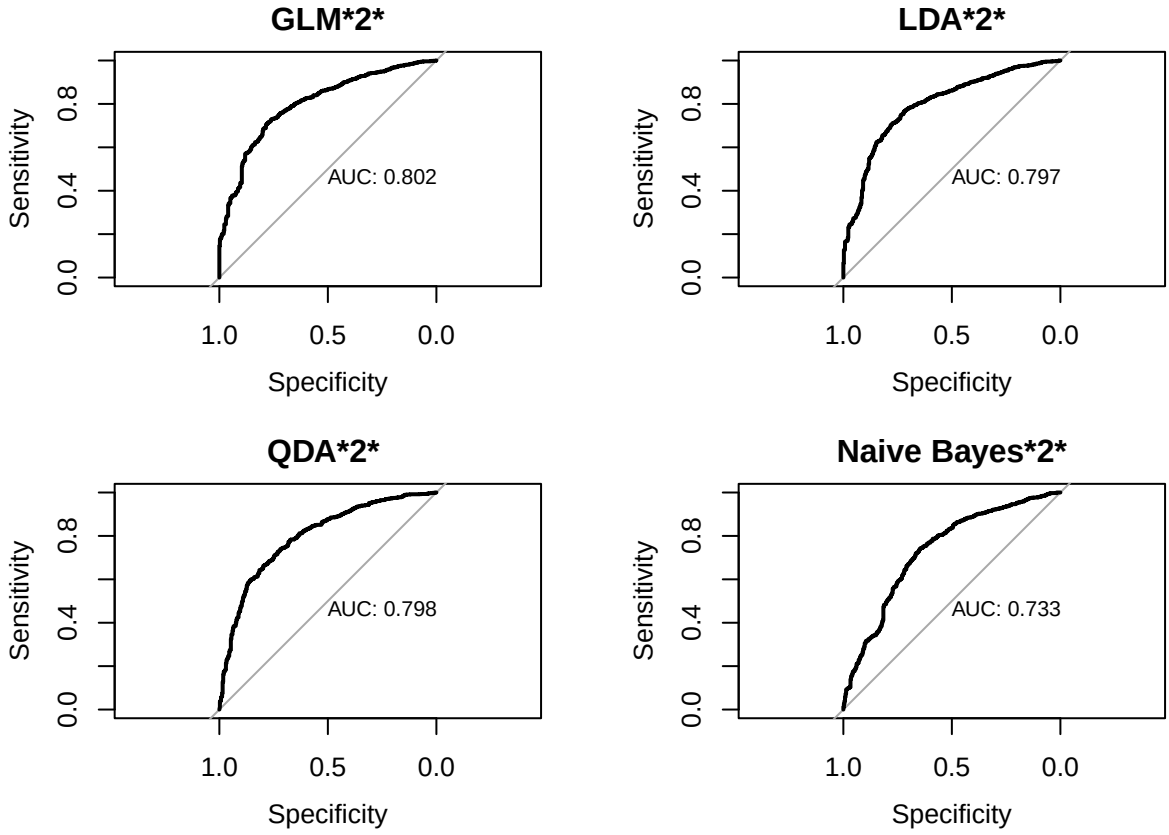
Vi kikar nu på ROC AUC

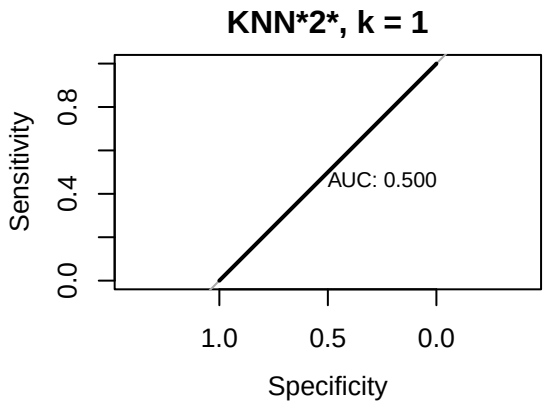
ROC "skapar" punkter baserat på y-axel = Sensitivity och x-axel = 1 - Specifity där ROC testar alla olika "thresholds". "thresholds" är alltså något man själv sätter i prediktionerna, vilken sannolikhet man vill att en av responsvariablerna ska ha, vilken "weight" alltså.

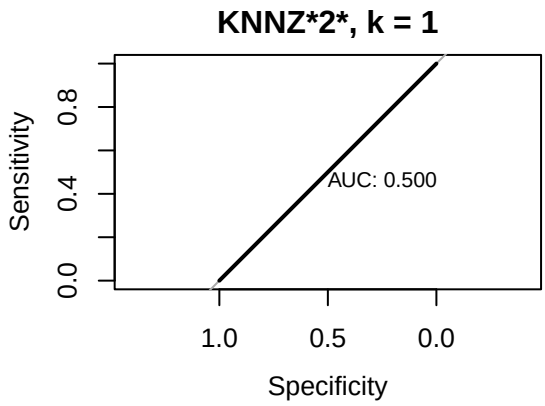
Exempel: glm.pred[glm.probs >.5] <- "Not Excellent" ersätter alla som har slh > "thresholds" med "Not excellent" där 0 < "thresholds" < 1, detta gör alltså paketet pROC med funktioen roc() med "thresholds" 0

hela vägen till 1 och plottar upp detta i en graf för varje beräkning. Sedan beräknar funktionen även arean under dessa punkter, AUC. Grafen nedan visar hur slutresultatet blev för respektive modell.

Vi får fram att GLM, LDA och QDA presterar ungefär likvärdigt medan k = 1 är lite sämre än slumpen, jag testade även med k = 3, 5, 10 och fick en mycket bättre ranking predicion med kNN då.




KNN*2*, k = 1
Sensitivity
0.0
0.4
0.8
AUC: 0.500
1.0
0.5
0.0
Specificity


KNNZ*2*, k = 1
Sensitivity
0.0
0.4
0.8
AUC: 0.500
1.0
0.5
0.0
Specificity