Datorlaboration 3

Josef Wilzén

August 30, 2024

732G12 Data Mining HT2024

Allmänt

Datorlaborationerna kräver att ni har R och Rstudio installerat.

- Kodmanual: länk
  - Utvärdering av klassificeringsproblem, se här.
- **ISL**: An Introduction to Statistical Learning,
  - Boken: länk
  - R-kod till labbar: länk
  - Dataset: länk och länk
- **IDM**: Introduction to Data Mining
  - Kod till boken finns här
  - Sample chapters
- Dataset till vissa uppgifter finns här.

Notera att ni inte behöver göra alla delar på alla uppgifter. Det viktiga är att ni får en förståelse för de olika principerna och modellerna som avhandlats. Dessa uppgifter ska inte lämnas in, utan är till för er övning.

Datauppdelning

För att motverka överanpassning bör ni dela upp data till träning-, validering-, (och testmängd). Detta kan göras med `createDataPartition()` från `caret`-paketet. Argument till den funktionen som är av vikt här är p som hur stor andel av observationerna som ska användas till träningsmängden. Ni kan också använda `subset()` för att göra detta också, men det blir svårare att tydligt ange de observationer som ska tilldelas till valideringsmängden. Denna uppdelning ska ske slumpmässigt. Notera att om en testmängd ska skapas måste uppdelningen ske en gång till från valideringsmängden eller träningsmängden.

Del 1: Trädmodeller

ISL

Gå igenom laborationerna 8.3.1 och 8.3.2 i **ISL**.

Begränsningar av beslutsträd

Läs in datamaterialet "issue.csv" som innehåller två förklarande variabler och en binär responsvariabel. Målet med denna övning är att ni ska skapa ett beslutsträd på detta data och undersöka kvaliten av denna modell. Använd R-paketet rpart, se kodmanualen för exempel, se även IDM, för att göra följande:

1. Läs in datamaterialet och se till att alla variabler följer rätt skala och dela sedan upp datamaterialet i 50 procent träning och 50 procent validering.
2. Skatta ett beslutsträd med där maxdjupet är satt till 5 och antal korsvalideringar är satt till 0 (i övrigt standardinställningar enligt kodmanualen).
3. Ta fram en plot över trädet med funktionen `rpart.plot` från paketet med samma namn. Se här för exempel på användning: länk.
4. Utvärdera den skattade modellen genom att:
   (a) Beräkna felkvoten av både tränings och valideringsmängden och tolka. Kolla tex här.
   (b) Utforska modellen, hur komplex är den, hur många löv har valts ut och vilket utav de är den största?
   (c) Hur djupt är det resulterande trädet?
5. Visualisera det ursprungliga datamaterialet och dess klasser. Hur ser beslutsgränsen ut?
6. Visualisera en liknande figur men nu endast på de predikterade värdena från valideringsmängden. (Ledning: För att prediktera nya utfall från en skattad modell används `predict(model, newdata = data, type = "class")`.) Verkar det som att beslutsgränsen som hittades i 4) också hittas med modellen? Vad för struktur har den nuvarande gränsen?
7. Skatta ett nytt beslutsträd där maxdjupet istället är satt till 10. Vad händer med den resulterande modellen och de predikterade värdena?
8. Sammanfatta era iakttagelser och svara på följande frågor:
   (a) Varför kan det vara svårt att använda de två skattade modellerna?
   (b) Varför kan det vara svårt att hitta en bra beslutsträdsmodell för detta data?

9. Skapa nya variabler i issue-materialet genom följande formler, $Z1 = X1 + X2$ och $Z2 = X1 - X2$. Visualisera datamaterialet i det nya koordinatsystemet och kommentera på beslutsgränsen.

10. Skatta ett beslutsträd likt den från uppgift 2) men använd istället det nya koordinatsystemet som förklarande variabler. Vad blir resultated av denna modell, dess felkvoter i träning och valideringsmängden, antalet löv och djup? Varför skiljer sig resultatet här gentemot tidigare modeller?

Regression med beslutsträd

1. Återvänd till datasetet `lab2_data_1.csv` från laboration 2.

2. Dela upp data i 70 % träning och 30 % validering.

3. Skatta bestlutsträd på träningsdata. Testa några olika träd (minst tre olika) där ni ändrar maxdjup och/eller minsta antal observationer som behövs för en uppdelning. Ta fram MSE för träning och valdiering för de olika modellerna. Skapa plottar över träningsdata och anpassade värden från modellerna på träningsdata. Jämför och analysera. Hur påverkar de olika hyperparametrarna anpassningen? Den modell som ni är mest nöjd med: Ta fram en plot över det skattade trädet. Jämför det plottade trädet mer er plot över anpassade värden på träningsdata.

4. Upprepa 3) men cost complexity pruning tillsammans med korsvalidering[1] för att välja modell. Hur komplext/djupt blev ert träd? Gör sedan prediktioner på valideringsdata och jämför validerings MSE med era resultat från 3).

Identifiering av spam-mail med hjälp av beslutsträd

Filen "spambase.csv" innehåller information om frekvensen av ett antal ord, symboler osv. från 4601 olika mail. Dessa har också klassificerats som antingen spam (spam = 1) eller vanliga mail (spam = 0). Er uppgift är nu att skatta ett beslutsträd som kan användas som någon form av spamfilter för nya mail som anländer till inkorgen. Mer detaljerad information om dessa attribut är angivna i bilagan.

1. Läs in materialet och se över variablernas skalor enligt dess beskrivning

2. Dela upp materialet i en tränings- och valideringsmängd i följande kombinationer, 10/90, 30/70, 50/50, 70/30 och 90/10. Skatta ett beslutsträd med standardinställningarna (xval = 0) för vardera uppdelning och jämför hur felkvoterna (på träning och validering) och trädets komplexitet (djup och antalet löv) förändras. Vilken datauppdelning verkar vara bäst?

[1] Korsvalideringen gör på träningsdata.

3. Använd nu uppdelningen 70/30 och undersök istället hur valet av splitkriterium påverkar kvaliten av modellen.

4. Använd nu uppdelningen 70/30 och använd cost complexity pruning tillsammans med korsvalidering[2] för att välja modell. Gör sedan prediktioner på de sista 30 % (nu som testdata).

   (a) Beräkna förväxlingsmatrisen för tränings- och testdata. Se tex här.

   (b) Beräkna sensitivitet och specificitet för tränings- och testdata.

   (c) Blev modellen bra? Hur djupt blev trädet? Hur många lövnoder?

Teorifrågor

1. Beskriv för- och nackdelarna med för- och efterbeskärning och svara vilken av metoderna som vanligtvis föredras.

2. Vad är en beslutsgräns och vilken form följer den i beslutsträd?

3. Vilken skala är bäst anpassad för klassificeringsproblem generellt, nominal eller ordinal? Varför?

4. Hur hanteras ordinala förklarande variabler i beslutsträd?

5. Klassificeringsträd: varför är det inte så bra att använda felkvoten (misclassification rate) när nya beslutsgränser ska skapas?

6. Regressionsträd: Vilken kostnadsfunktion brukar används för att utvädera beslutsgränser?

7. Utan regularisering: vad brukar vara ett problem för trädmodeller, bias eller varians?

8. Utgå från ett kontinuerligt y och en kontinuerlig förklarande variabel x. Rita upp en funktion (på papper) $y = f(x)$ som är lätt att anpassa med ett beslutsträd, men som är problematisk för linjär regression.

9. **ISL** 8.4 Exercises Conceptual: 1, 3

[2] Korsvalideringen gör på träningsdata.

Del 2: Naive Bayes classifier

I filen "loan.csv" finns 600 observationer från en bank där egenskaper hos individer som tagit lån är registrerade. Det finns även information om lånet betalats tillbaka eller ej. Variablerna beskrivs som:

- **Y**, om lånet betalas tillbaka eller ej (0 = Ja, 1 = N ej)
- **Residence**, om individen är en medborgare eller ej (0 = N ej, 1 = Ja)
- **Age**, vilken åldersgrupp som individen tillhör (0 = [35, Infinity), 1 = [18 − 35])
- **House**, om individen äger ett hus eller ej (0 = N ej, 1 = Ja)
- **Sex**. individens kön (0 = Man, 1 = Kvinna)
- **Employment**, om individen har en anställning eller ej (0 = N ej, 1 = Ja)
- **Marital status**, om individen är gift eller ej (0 = Gift eller änka, 1 = Skild eller separerad)

1. Tanken är att ni ska "räkna för hand" i R: Klassificera, med en Naïve Bayes klassificerare, en ny individ som har följande egenskaper: En man som är 26 år gammal, är gift, har en anställning men äger inte ett eget hus och är inte en medborgare i landet. Beskriv detaljerat hur ni gått tillväga för att lösa denna uppgift. (Ledning: Använd table() och prop.table() för att beräkna sannolikheterna som behövs.)
2. Beräkna samma klassificering utan att använda Naïve Bayes på följande sätt. Sortera datamaterialet, hitta antalet individer som överensstämmer med de eftersökta egenskaperna av den nya individen och räkna hur många som betalat tillbaka sitt lån eller ej. Ett klassificeringsbeslut kan tas genom att titta på de relativa frekvenserna.
3. Jämför resultaten mellan 1) och 2). Om ni kommit fram till olika beslut, vilken utav metoderna förlitar ni er på mest och varför gör ni det?

Del 3: Ensemblemetoder

ISL

- Bootstrap:
  - Gå igenom laborationen 5.3.4
  - 5.4 Exercises Conceptual: 4, Exercises Applied: 9
- Bagging and Random Forests, Boosting
  - Gå igenom laborationerna 8.3.3 och 8.3.4
  - 8.4 Exercises Conceptual: 2, 5
  - Exercises Applied: 7, 8, 9, 10, 11

Bilaga

Attribute Information for spambase: The last column of "spambase.csv" denotes whether the e-mail was considered spam (1) or not (0), i.e. unsolicited commercial e-mail. Most of the attributes indicate whether a particular word or character was frequently occurring in the e-mail. The run-length attributes (55-57) measure the length of sequences of consecutive capital letters. For the statistical measures of each attribute, see the end of this file.

Here are the definitions of the attributes: 48 continuous real [0,100] attributes of type word_freq_WORD = percentage of words in the e-mail that match WORD, i.e. 100 * (number of times the WORD appears in the e-mail) / total number of words in e-mail. A "word" in this case is any string of alphanumeric characters bounded by non-alphanumeric characters or end-of-string.

- 6 continuous real [0,100] attributes of type char_freq_CHAR = percentage of characters in the e-mail that match CHAR, i.e. 100 * (number of CHAR occurrences) / total characters in e-mail
- 1 continuous real [1,. . .] attribute of type capital_run_length_average = average length of uninterrupted sequences of capital letters
- 1 continuous integer [1,. . .] attribute of type capital_run_length_longest = length of longest uninterrupted sequence of capital letters 1 continuous integer [1,. . .] attribute of type capital_run_length_total = sum of length of uninterrupted sequences of capital letters = total number of capital letters in the e-mail
- 1 nominal {0,1} class attribute of type spam = denotes whether the e-mail was considered spam (1) or not (0), i.e. unsolicited commercial e-mail.