MODELLI CLASSIFICATIVI

Baciu - Comelli – Jimenez

Elenco variabili

Il dataset che abbiamo preso in considerazione è composto da 6819 osservazioni e 96 variabili e riguarda la previsione di bancarotta per le aziende sulla base di alcuni indicatori economico-finanziari.

Y - Bankrupt?: Class label
X1 - ROA(C) before interest and depreciation before interest: Return On Total Assets(C)
X2 - ROA(A) before interest and % after tax: Return On Total Assets(A)
X3 - ROA(B) before interest and depreciation after tax: Return On Total Assets(B)
X4 - Operating Gross Margin: Gross Profit/Net Sales
X5 - Realized Sales Gross Margin: Realized Gross Profit/Net Sales
X6 - Operating Profit Rate: Operating Income/Net Sales
X7 - Pre-tax net Interest Rate: Pre-Tax Income/Net Sales
X8 - After-tax net Interest Rate: Net Income/Net Sales
X9 - Non-industry income and expenditure/revenue: Net Non-operating Income Ratio
X10 - Continuous interest rate (after tax): Net Income-Exclude Disposal Gain or Loss/Net Sales
X11 - Operating Expense Rate: Operating Expenses/Net Sales
X12 - Research and development expense rate: (Research and Development Expenses)/Net Sales
X13 - Cash flow rate: Cash Flow from Operating/Current Liabilities
X14 - Interest-bearing debt interest rate: Interest-bearing Debt/Equity
X15 - Tax rate (A): Effective Tax Rate
X16 - Net Value Per Share (B): Book Value Per Share(B)
X17 - Net Value Per Share (A): Book Value Per Share(A)
X18 - Net Value Per Share (C): Book Value Per Share(C)
X19 - Persistent EPS in the Last Four Seasons: EPS-Net Income
X20 - Cash Flow Per Share
X21 - Revenue Per Share (Yuan ¥): Sales Per Share
X22 - Operating Profit Per Share (Yuan ¥): Operating Income Per Share
X23 - Per Share Net profit before tax (Yuan ¥): Pretax Income Per Share
X24 - Realized Sales Gross Profit Growth Rate
X25 - Operating Profit Growth Rate: Operating Income Growth
X26 - After-tax Net Profit Growth Rate: Net Income Growth
X27 - Regular Net Profit Growth Rate: Continuing Operating Income after Tax Growth
X28 - Continuous Net Profit Growth Rate: Net Income-Excluding Disposal Gain or Loss Growth
X29 - Total Asset Growth Rate: Total Asset Growth
X30 - Net Value Growth Rate: Total Equity Growth
X31 - Total Asset Return Growth Rate Ratio: Return on Total Asset Growth
X32 - Cash Reinvestment %: Cash Reinvestment Ratio
X33 - Current Ratio
X34 - Quick Ratio: Acid Test
X35 - Interest Expense Ratio: Interest Expenses/Total Revenue
X36 - Total debt/Total net worth: Total Liability/Equity Ratio
X37 - Debt ratio %: Liability/Total Assets
X38 - Net worth/Assets: Equity/Total Assets
X39 - Long-term fund suitability ratio (A): (Long-term Liability+Equity)/Fixed Assets
X40 - Borrowing dependency: Cost of Interest-bearing Debt
X41 - Contingent liabilities/Net worth: Contingent Liability/Equity
X42 - Operating profit/Paid-in capital: Operating Income/Capital
X43 - Net profit before tax/Paid-in capital: Pretax Income/Capital
X44 - Inventory and accounts receivable/Net value: (Inventory+Accounts Receivables)/Equity
X45 - Total Asset Turnover

X46 - Accounts Receivable Turnover
X47 - Average Collection Days: Days Receivable Outstanding
X48 - Inventory Turnover Rate (times)
X49 - Fixed Assets Turnover Frequency
X50 - Net Worth Turnover Rate (times): Equity Turnover
X51 - Revenue per person: Sales Per Employee
X52 - Operating profit per person: Operation Income Per Employee
X53 - Allocation rate per person: Fixed Assets Per Employee
X54 - Working Capital to Total Assets
X55 - Quick Assets/Total Assets
X56 - Current Assets/Total Assets
X57 - Cash/Total Assets
X58 - Quick Assets/Current Liability
X59 - Cash/Current Liability
X60 - Current Liability to Assets
X61 - Operating Funds to Liability
X62 - Inventory/Working Capital
X63 - Inventory/Current Liability
X64 - Current Liabilities/Liability
X65 - Working Capital/Equity
X66 - Current Liabilities/Equity
X67 - Long-term Liability to Current Assets
X68 - Retained Earnings to Total Assets
X69 - Total income/Total expense
X70 - Total expense/Assets
X71 - Current Asset Turnover Rate: Current Assets to Sales
X72 - Quick Asset Turnover Rate: Quick Assets to Sales
X73 - Working capitcal Turnover Rate: Working Capital to Sales
X74 - Cash Turnover Rate: Cash to Sales
X75 - Cash Flow to Sales
X76 - Fixed Assets to Assets
X77 - Current Liability to Liability
X78 - Current Liability to Equity
X79 - Equity to Long-term Liability
X80 - Cash Flow to Total Assets
X81 - Cash Flow to Liability
X82 - CFO to Assets
X83 - Cash Flow to Equity
X84 - Current Liability to Current Assets
X85 - Liability-Assets Flag: 1 if Total Liability exceeds Total Assets, 0 otherwise
X86 - Net Income to Total Assets
X87 - Total assets to GNP price
X88 - No-credit Interval
X89 - Gross Profit to Sales
X90 - Net Income to Stockholder's Equity
X91 - Liability to Equity
X92 - Degree of Financial Leverage (DFL)
X93 - Interest Coverage Ratio (Interest expense to EBIT)
X94 - Net Income Flag: 1 if Net Income is Negative for the last two years, 0 otherwise
X95 - Equity to Liability

Step 0

Inizialmente abbiamo verificato l'assenza di dati mancanti. Abbiamo inoltre eliminato le variabili più collineari e risolto il problema della near zero variance, rimuovendo le uniche due covariate binarie (che vedevano la presenza quasi esclusiva di una delle due modalità).

In seguito, abbiamo ricavato i dati di score (10% del totale) e suddiviso la parte restante del dataset iniziale in training (66%) e validation (34%), stratificando per il target.
Le compagnie in bancarotta sono 220 sulle 6819 all'interno del dataset, cioè il 3,2% del totale. La distribuzione del target è realistica, ma trattandosi di un evento raro, abbiamo deciso di procedere con un oversampling per agevolare l'apprendimento dei modelli classificativi sul dataset di training.

Abbiamo tunato un albero decisionale come strumento di model selection. Abbiamo quindi creato un secondo dataset di training con le variabili importanti individuate dall'albero da utilizzare per i modelli che richiedono model selection.

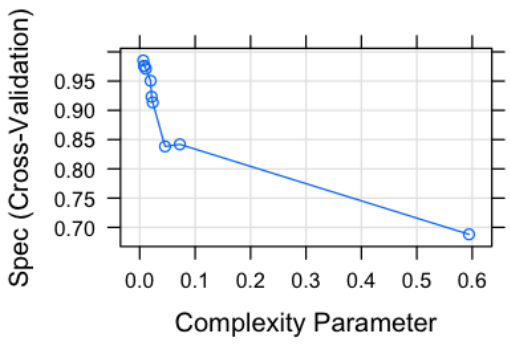
Step 1: tuning

Abbiamo considerato come evento la non bancarotta ("no") e quindi scelto come metrica trainante per la nostra analisi la specificity, in modo da minimizzare il false positive rate. Volevamo quindi massimizzare la corretta classificazione dei casi di bancarotta, minimizzando quelli previsti erroneamente. Per il tuning dei modelli abbiamo utilizzato una ten-fold cross-validation.

- DECISION TREE

Lo stesso albero usato per la model selection è stato tunato massimizzando la specificity in modo da poterlo considerare anche come uno dei modelli classificativi. Questo modello non richiede alcun tipo di pre-processing.

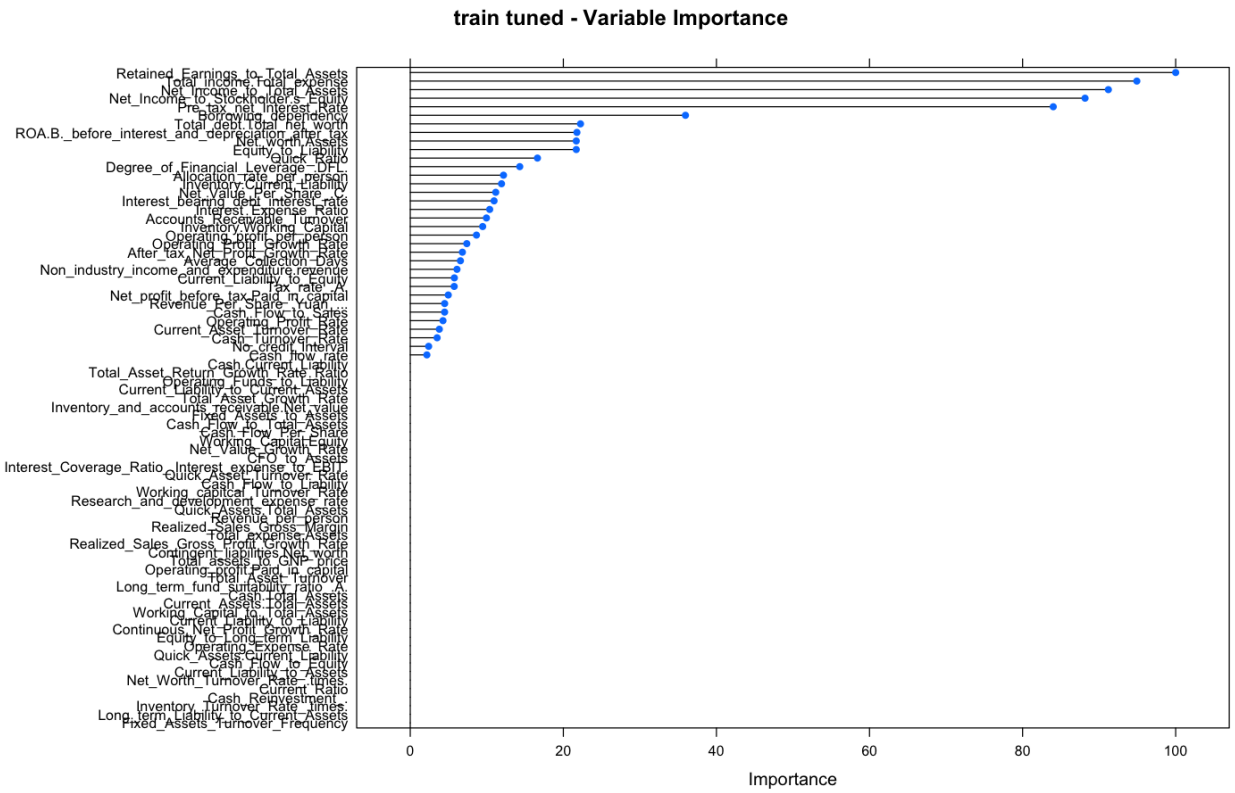
cp	ROC	Sens	Spec
0.006375925	0.9493771	0.8865159	0.9852073
0.007651109	0.9406766	0.8755511	0.9757718
0.008416220	0.9380771	0.8704491	0.9757718
0.010966590	0.9232011	0.8436685	0.9709275
0.019637847	0.8996788	0.8171548	0.9505271
0.021423106	0.8869863	0.8217401	0.9232312
0.023463402	0.8823577	0.8250565	0.9135373
0.045651619	0.8484738	0.8309212	0.8383056
0.072430502	0.8195225	0.7804110	0.8418770
0.594236164	0.7366691	0.7852878	0.6880504




```
Spec was used to select the optimal model using the largest value.
The final value used for the model was cp = 0.006375925.
```

L'albero migliore risulta quello con parametro di complessità pari a 0,006, che garantisce una specificity di 0,985 sui dati di training.

```
          Reference
Prediction   no  yes
       no  44.3  0.7
       yes  5.7 49.3

 Accuracy (average) : 0.9359
```



- LOGISTIC MODEL

Per tunare il modello logistico siamo partiti dal dataset risultante dalla model selection, pre-processato per risolvere eventuali problemi di collinearità rimasti. Abbiamo inoltre scalato le variabili per rendere il loro range più omogeneo.

ROC	Sens	Spec
0.9091169	0.850812	0.8461994

Prediction \ Reference	no	yes
no	42.5	7.7
yes	7.5	42.3

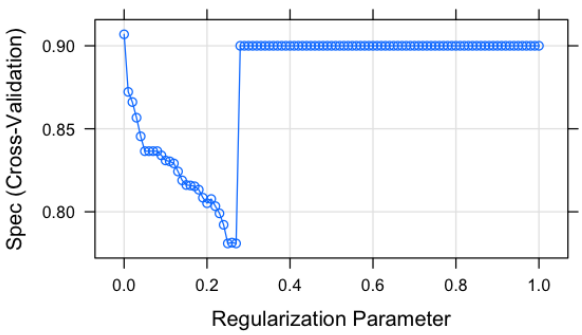
Accuracy (average) : 0.8485

Il modello logistico garantisce una specificity di 0,846 sui dati di training.

- LASSO

Per il modello Lasso abbiamo effettuato lo stesso pre-processing del modello logistico ma siamo partiti dal dataset completo, in quanto Lasso è in grado di porre autonomamente alcuni coefficienti uguali a zero (nel nostro caso 5).

lambda	ROC	Sens	Spec
0.00	0.9430619	0.8576933	0.9069098
0.01	0.9366869	0.8500422	0.8722250
0.02	0.9328295	0.8508062	0.8661019
0.03	0.9291038	0.8477463	0.8566671
0.04	0.9258518	0.8472367	0.8454452
0.05	0.9210953	0.8482558	0.8365172



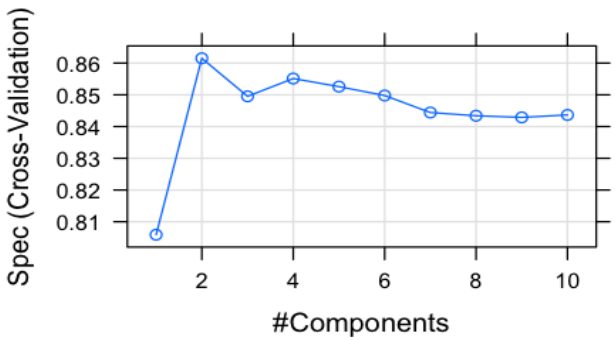
Il modello migliore risulta essere quello con il parametro di regolarizzazione λ=0, che garantisce una specificity di 0,907 sui dati di training.

```
          Reference
Prediction   no  yes
       no  42.9  4.7
       yes  7.1 45.3

 Accuracy (average) : 0.8823
```

- PLS

Per il modello PLS abbiamo effettuato ancora una volta lo stesso pre-processing, a partire dal dataset ottenuto con la model selection.

```
  ncomp  ROC        Sens       Spec
   1     0.8966880  0.8416238  0.8059108
   2     0.9106568  0.8240270  0.8615101
   3     0.9194517  0.8377979  0.8495223
   4     0.9210868  0.8375428  0.8551332
   5     0.9214362  0.8400926  0.8525822
   6     0.9210854  0.8383094  0.8497780
   7     0.9218052  0.8413700  0.8444189
   8     0.9219455  0.8423898  0.8434005
   9     0.9217984  0.8411143  0.8428915
  10     0.9218746  0.8413681  0.8436556

Spec was used to select the optimal model using the largest value.
The final value used for the model was ncomp = 2.
```



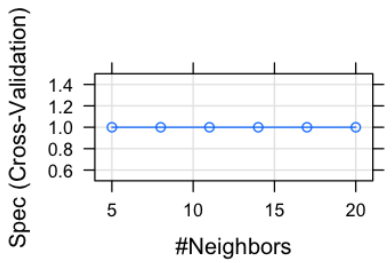
Il modello migliore risulta essere quello con due componenti PLS, che garantisce una specificity di 0,862 sui dati di training.

```
          Reference
Prediction   no  yes
       no  41.2  6.9
       yes  8.8 43.1

 Accuracy (average) : 0.8428
```

- NEAREST NEIGHBOUR

Per il modello knn abbiamo utilizzato il dataset derivante dalla model selection e pre-processato i dati scalando e centrando le variabili in modo da avere un range più omogeneo.

```
  k   ROC        Sens       Spec
   5  0.9887791  0.9035961  1
   8  0.9887791  0.8714630  1
  11  0.9887791  0.8403438  1
  14  0.9887791  0.8138190  1
  17  0.9883371  0.7918900  1
  20  0.9866320  0.7748007  1

Spec was used to select the optimal model using the largest value.
The final value used for the model was k = 20.
```



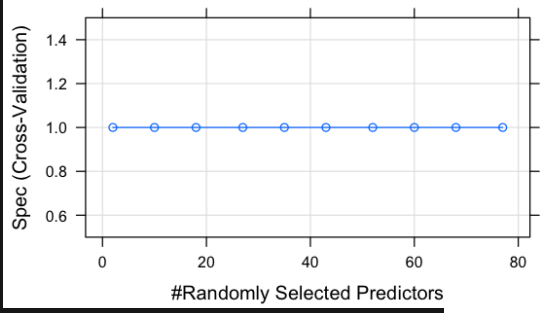
Il modello migliore risulta essere quello con 20 vicini, perché a parità di specificity (in questo caso sempre 1), il modello con più vicini tende a overfittare in misura minore.

```
          Reference
Prediction   no  yes
       no  38.7  0.0
       yes 11.3 50.0

 Accuracy (average) : 0.8874
```

- RANDOM FOREST

Per questo modello che non richiede alcun tipo di pre-processing, siamo partiti dal dataset completo.

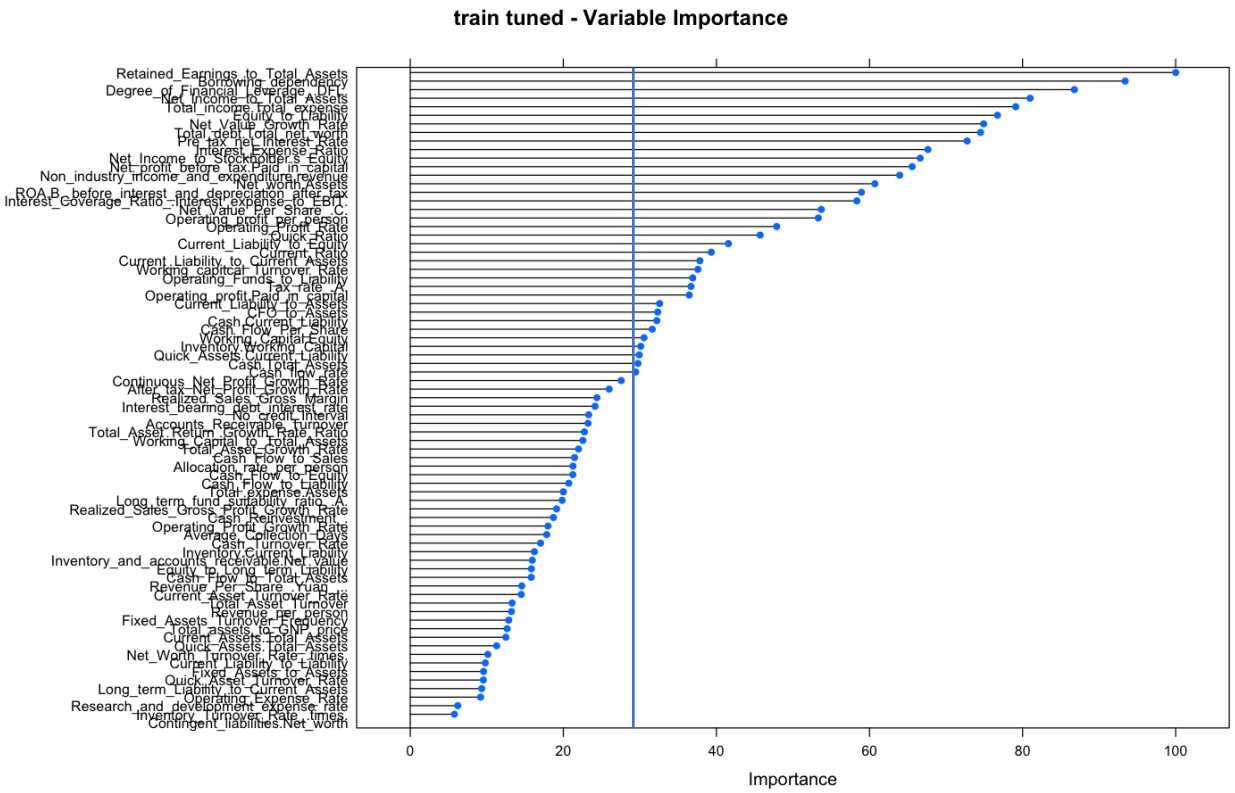
```
  mtry  ROC  Sens       Spec
   2    1    0.9964299  1
  10    1    0.9913278  1
  18    1    0.9895421  1
  27    1    0.9869917  1
  35    1    0.9864815  1
  43    1    0.9862271  1
  52    1    0.9849522  1
  60    1    0.9836774  1
  68    1    0.9846971  1
  77    1    0.9846971  1

Spec was used to select the optimal model using the largest value.
The final value used for the model was mtry = 2.
```



Il modello migliore, a parità di specificity (sempre 1), risulta essere quello con due predittori selezionati casualmente per ogni split.

```
          Reference
Prediction   no  yes
       no  49.8  0.0
       yes  0.2 50.0

 Accuracy (average) : 0.9982
```

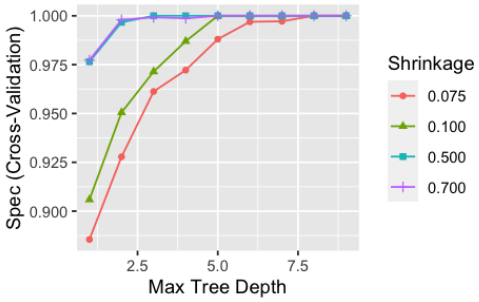
La variabile più importante per la random forest (così come per l'albero decisionale) risulta essere "Retained earnings to total assets", un rapporto che aiuta a misurare la redditività delle attività di un'azienda.



- GRADIENT BOOSTING

Anche questo modello non richiede né pre-processing né model selection.

```
  0.500      1                   0.9757621  0.8969712  0.9765345
  0.500      2                   0.9900092  0.9339487  0.9966843
  0.500      3                   0.9940604  0.9477171  1.0000000
Tuning parameter 'n.trees' was held constant at a value of 50
Tuning
 parameter 'n.minobsinnode' was held constant at a value of 20
Spec was used to select the optimal model using the largest value.
The final values used for the model were n.trees = 50, interaction.depth = 3,
 shrinkage = 0.5 and n.minobsinnode = 20.
```



Il modello migliore, a parità di specificity (massima), risulta essere quello con una minore complessità e una maggiore capacità di generalizzazione (parametro di shrinkage più basso).

```
          Reference
Prediction   no  yes
       no  47.4  0.0
       yes  2.6 50.0

 Accuracy (average) : 0.9739
```

- LINEAR DISCRIMINANT ANALYSIS

Per questo modello siamo partiti dal dataset risultante dalla model selection e abbiamo pre-processato i dati per normalizzare le covariate e risolvere eventuali problemi di collinearità residui.

```
          Reference
Prediction   no  yes
       no  42.1  7.8
       yes  7.9 42.2

 Accuracy (average) : 0.8429
```

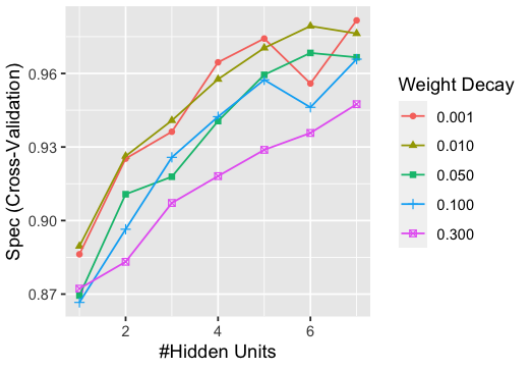
```
ROC        Sens       Spec
0.9220167  0.8423898  0.8434005
```

Il modello LDA garantisce una specificity pari a 0,843.

- NEURAL NETWORK

Per la rete neurale abbiamo utilizzato il dataset ottenuto con la model selection e abbiamo svolto un pre-processing per risolvere eventuali problemi di collinearità rimasti, oltre a riscalare le variabili in un range tra 0 e 1.

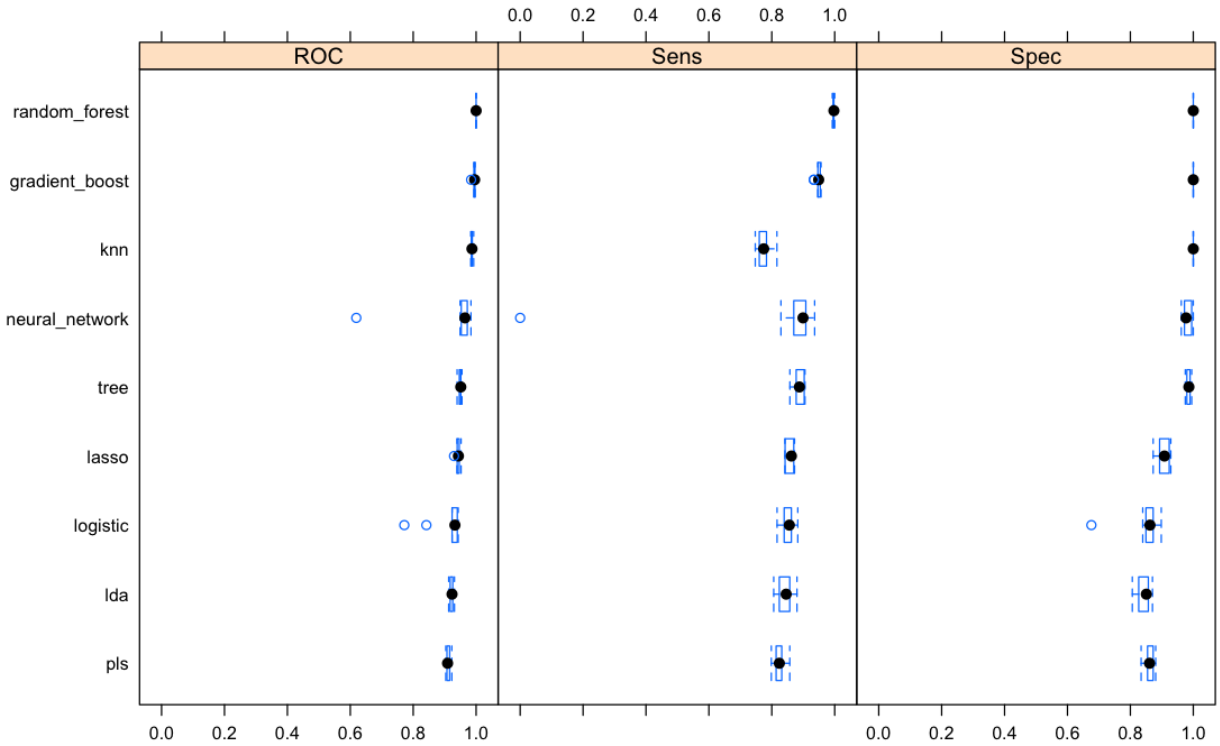
```
  7     0.001  0.9314401  0.8063970  0.9816424
  7     0.010  0.9687220  0.9048787  0.9762891
  7     0.050  0.9664865  0.8972309  0.9665914
  7     0.100  0.9639579  0.8882984  0.9658313
  7     0.300  0.9579607  0.8770733  0.9474626

Spec was used to select the optimal model using the largest value.
The final values used for the model were size = 7 and decay = 0.001.
```



Il modello migliore risulta essere quello con 7 neuroni nascosti e un decay di 0,001, che garantisce una specificity pari a 0,982.

```
          Reference
Prediction   no  yes
       no  40.3  0.9
       yes  9.7 49.1

 Accuracy (average) : 0.894
```

Prima di procedere con lo step successivo, per un primo confronto tra i nove modelli costruiti abbiamo guardato inizialmente la distribuzione delle metriche cross-validate. In base al grafico ottenuto, il modello migliore sembrerebbe essere la random forest, che presenta valori molto elevati e stabili per tutte le metriche di interesse.

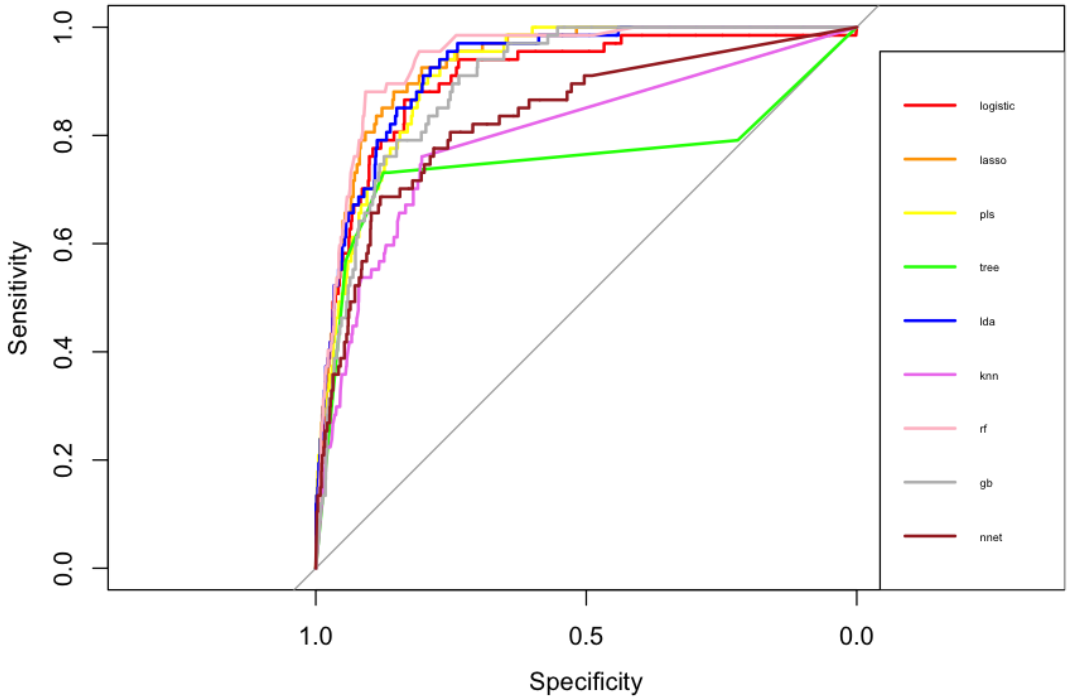


Step 2: assessment

Abbiamo confrontato le performance dei vari modelli attraverso i valori dell'AUC (area sotto le curve ROC). In base a questa metrica il modello migliore sembra essere ancora una volta la random forest.

```
Data: random_forest in 2019 controls (Bankrupt no) > 67 cases (Bankrupt yes).
Area under the curve: 0.941
```

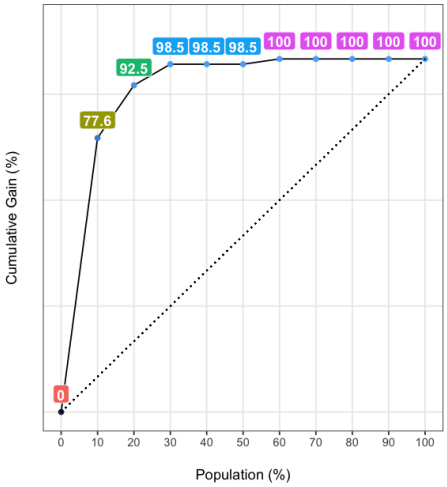
Tuttavia, abbiamo anche rappresentato graficamente le curve ROC di tutti i modelli per vedere se si intersecassero tra loro. Come si può notare dal grafico sottostante, la curva relativa alla random forest risulta quasi sempre sopra tutte le altre, ma in corrispondenza dei valori più elevati di sensitivity e false positive rate si interseca con le curve dei modelli PLS, gradient boosting e Lasso.

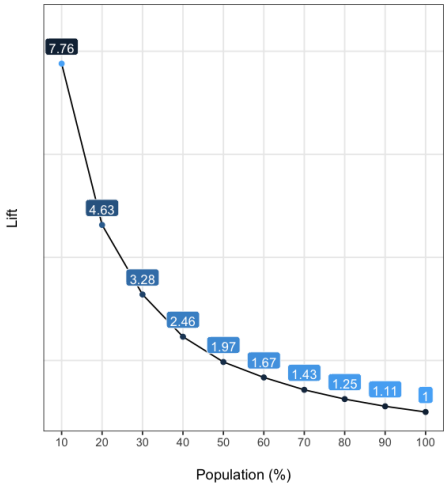


Abbiamo quindi deciso, dopo aver ricavato le posteriors aggiustate, di svolgere un confronto più approfondito costruendo le curve lift relative a questi quattro modelli:

- Random forest

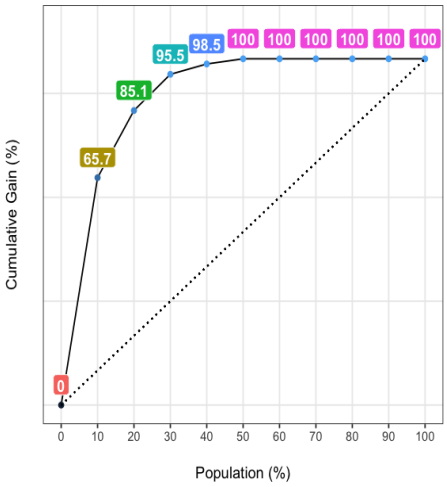
	Population	Gain	Lift	Score.Point
1	10	77.61	7.76	0.00398656380
2	20	92.54	4.63	0.00151917964
3	30	98.51	3.28	0.00067419518
4	40	98.51	2.46	0.00040135122
5	50	98.51	1.97	0.00019950456
6	60	100.00	1.67	0.00013274483
7	70	100.00	1.43	0.00006624381
8	80	100.00	1.25	0.00000000000
9	90	100.00	1.11	0.00000000000
10	100	100.00	1.00	0.00000000000

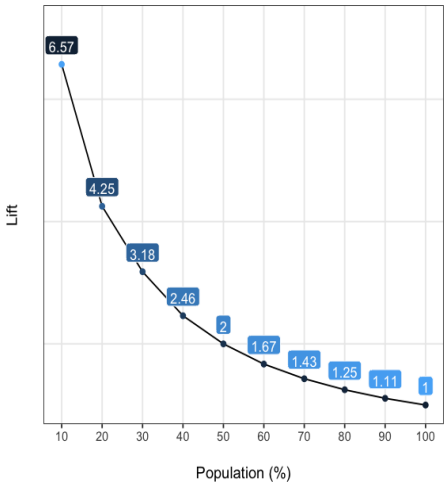




- PLS

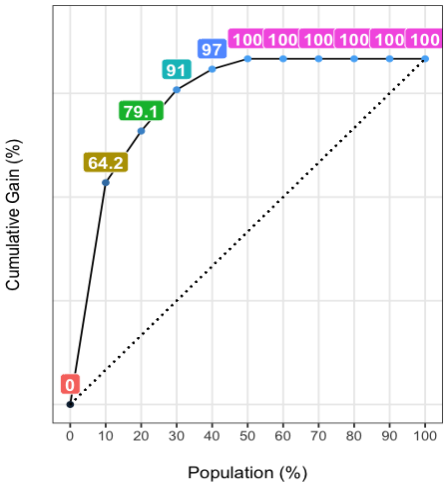
	Population	Gain	Lift	Score.Point
1	10	65.67	6.57	0.03971584118
2	20	85.07	4.25	0.03155690992
3	30	95.52	3.18	0.02701085395
4	40	98.51	2.46	0.02377691512
5	50	100.00	2.00	0.02123389819
6	60	100.00	1.67	0.01881909981
7	70	100.00	1.43	0.01662224044
8	80	100.00	1.25	0.01435878289
9	90	100.00	1.11	0.01156822417
10	100	100.00	1.00	0.00000000000

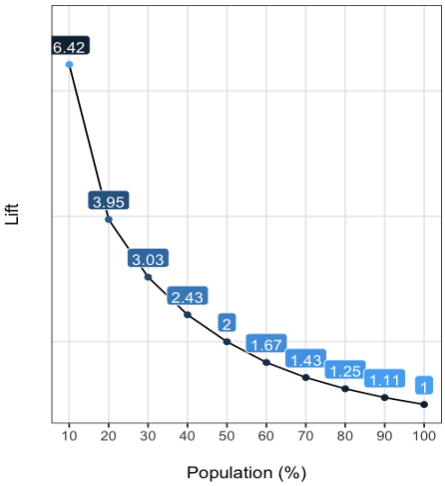




- Gradient boosting

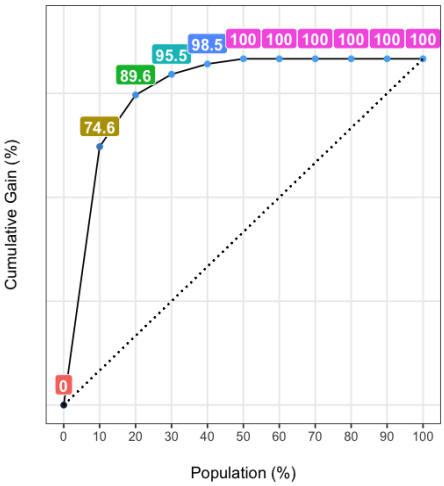
	Population	Gain	Lift	Score.Point
1	10	64.18	6.42	0.011218524256
2	20	79.10	3.95	0.002441453109
3	30	91.04	3.03	0.000988397823
4	40	97.01	2.43	0.000457186658
5	50	100.00	2.00	0.000262622781
6	60	100.00	1.67	0.000161489476
7	70	100.00	1.43	0.000115224693
8	80	100.00	1.25	0.000084638142
9	90	100.00	1.11	0.000058273749
10	100	100.00	1.00	0.000009685455

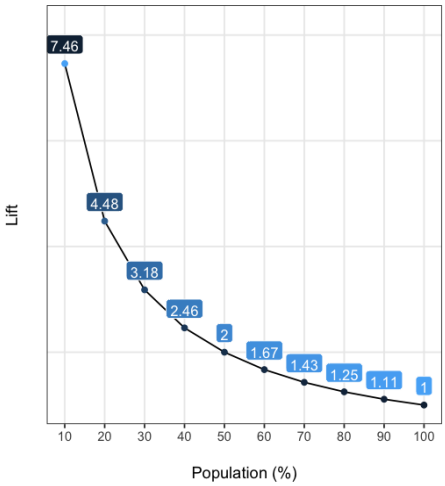




- Lasso

	Population	Gain	Lift	Score.Point
1	10	74.63	7.46	0.06874684213
2	20	89.55	4.48	0.02071913275
3	30	95.52	3.18	0.00894826698
4	40	98.51	2.46	0.00427846116
5	50	100.00	2.00	0.00207909889
6	60	100.00	1.67	0.00091632070
7	70	100.00	1.43	0.00047466270
8	80	100.00	1.25	0.00015210498
9	90	100.00	1.11	0.00002975378
10	100	100.00	1.00	0.00000000000





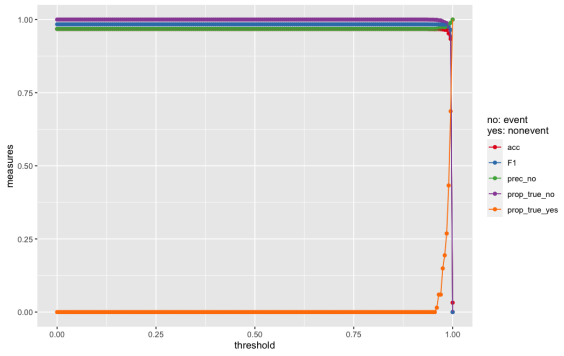
In base a questi risultati il modello più performante si conferma essere la random forest, che ha un'ottima capacità previsiva, in quanto nei primi tre decili riesce a catturare cumulativamente il 98,5% del totale dei casi di bancarotta nel dataset di validation. Utilizzando questo modello, per avere una specificity di 0,7761 sui dati di validation si dovrebbe considerare una soglia molto elevata, pari a 0,996. Tuttavia, dall'analisi delle curve lift si nota che gli altri modelli sembrano garantire risultati simili anche con una soglia più bassa.

Step 3: threshold

In seguito, sempre per i quattro modelli ritenuti più performanti, abbiamo valutato le metriche di classificazione delle unità al variare di tutte le possibili soglie tra 0 e 1 con un salto di 0,005:

- Random forest

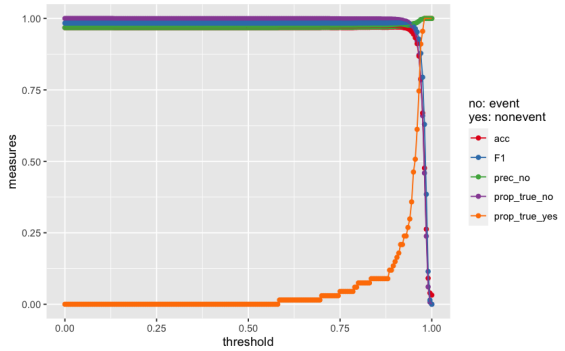
	threshold	prop_true_no	prop_true_yes	acc	prec_no	F1
191	0.950	0.9995047	0.00000000	0.96740173	0.9678657	0.9834308
192	0.955	0.9990094	0.00000000	0.96692234	0.9678503	0.9831830
193	0.960	0.9990094	0.01492537	0.96740173	0.9683149	0.9834227
194	0.965	0.9975235	0.05970149	0.96740173	0.9696678	0.9833984
195	0.970	0.9970282	0.05970149	0.96692234	0.9696532	0.9831502
196	0.975	0.9935612	0.14925373	0.96644295	0.9723703	0.9828515
197	0.980	0.9905894	0.19402985	0.96500479	0.9737098	0.9820771
198	0.985	0.9876176	0.26865672	0.96452541	0.9760157	0.9817824
199	0.990	0.9697870	0.43283582	0.95254075	0.9809619	0.9753425
200	0.995	0.9420505	0.68656716	0.93384468	0.9890796	0.9649924
201	1.000	0.0000000	1.00000000	0.03211889	1.0000000	0.0000000



In questo caso la specificity si discosta da 0 solo per valori estremamente elevati della soglia (≥0,96), mentre le altre metriche sono pressoché costanti (vicine a 1) per tutte le soglie, quindi non sembrano essere particolarmente discriminanti ai fini della scelta della finale.

- PLS

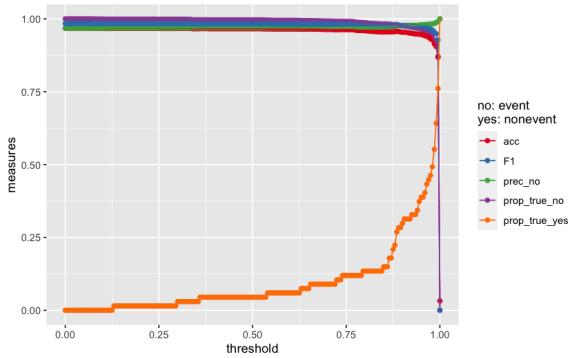
	threshold	prop_true_no	prop_true_yes	acc	prec_no	F1
191	0.950	0.961367013	0.4626866	0.94534995	0.9817906	0.97147147
192	0.955	0.946508172	0.5074627	0.93240652	0.9830247	0.96442089
193	0.960	0.921743437	0.6119403	0.91179291	0.9862215	0.95289299
194	0.965	0.872213967	0.7462687	0.86816874	0.9904387	0.92757440
195	0.970	0.785042100	0.9104478	0.78906999	0.9962288	0.87811634
196	0.975	0.659732541	0.9552239	0.66922339	0.9977528	0.79427549
197	0.980	0.459138187	1.0000000	0.47651007	1.0000000	0.62932790
198	0.985	0.238236751	1.0000000	0.26270374	1.0000000	0.38480000
199	0.990	0.060921248	1.0000000	0.09108341	1.0000000	0.11484594
200	0.995	0.007429421	1.0000000	0.03930968	1.0000000	0.01474926
201	1.000	0.000000000	1.0000000	0.03211889	1.0000000	0.00000000



In questo caso si nota come la specificity raggiunga livelli soddisfacenti già con una soglia minore, ad esempio con una soglia di 0,955 è maggiore di 0,50 (con la random forest era ancora 0). La soglia 0,97 sembra essere la scelta ideale in quanto garantisce una specificity superiore a 0,90. Aumentando ulteriormente la soglia le altre metriche diminuiscono troppo drasticamente.

- Gradient boosting

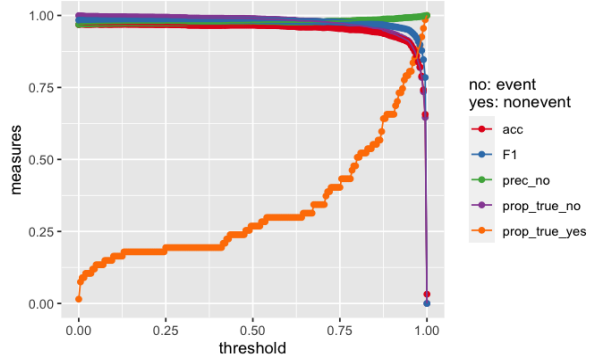
	threshold	prop_true_no	prop_true_yes	acc	prec_no	F1
191	0.950	0.9653294	0.3880597	0.94678811	0.9793970	0.9723123
192	0.955	0.9628529	0.3880597	0.94439118	0.9793451	0.9710290
193	0.960	0.9598811	0.4029851	0.94199425	0.9797776	0.9697273
194	0.965	0.9574047	0.4328358	0.94055609	0.9807204	0.9689223
195	0.970	0.9529470	0.4477612	0.93672100	0.9811321	0.9668342
196	0.975	0.9470035	0.4626866	0.93144775	0.9815195	0.9639526
197	0.980	0.9420505	0.4925373	0.92761266	0.9824380	0.9618205
198	0.985	0.9266964	0.5522388	0.91466922	0.9842188	0.9545918
199	0.990	0.9138187	0.6417910	0.90508150	0.9871589	0.9490741
200	0.995	0.8717187	0.7611940	0.86816874	0.9909910	0.9275362
201	1.000	0.0000000	1.0000000	0.03211889	1.0000000	0.0000000



In questo caso la specificity si discosta da 0 per valori molto bassi della soglia (≥0,13), però raggiunge valori accettabili con una soglia maggiore rispetto al modello PLS. Le altre metriche sono sempre costanti vicino a 1, anche se per valori molto alti tendono a diminuire leggermente.

- Lasso

	threshold	prop_true_no	prop_true_yes	acc	prec_no	F1
187	0.930	0.9217434	0.7462687	0.91610738	0.9909478	0.9550937
188	0.935	0.9192670	0.7761194	0.91466922	0.9919829	0.9542416
189	0.940	0.9153046	0.7910448	0.91131352	0.9924812	0.9523319
190	0.945	0.9128281	0.7910448	0.90891659	0.9924610	0.9509804
191	0.950	0.9068846	0.8059701	0.90364334	0.9929501	0.9479679
192	0.955	0.8979693	0.8059701	0.89501438	0.9928806	0.9430429
193	0.960	0.8870728	0.8358209	0.88542665	0.9938957	0.9374509
194	0.965	0.8756810	0.8507463	0.87488015	0.9943757	0.9312615
195	0.970	0.8603269	0.8507463	0.86001918	0.9942759	0.9224642
196	0.975	0.8380386	0.8805970	0.83940556	0.9952941	0.9099220
197	0.980	0.8187221	0.8955224	0.82118888	0.9957831	0.8986138
198	0.985	0.7845468	0.9253731	0.78906999	0.9968534	0.8780488
199	0.990	0.7350173	0.9552239	0.74209012	0.9979825	0.8465488
200	0.995	0.6458643	0.9850746	0.65675935	0.9992337	0.7845969
201	1.000	0.0000000	1.0000000	0.03211889	1.0000000	0.0000000



In questo caso la specificity non è mai nulla per alcuna soglia e ha una crescita più costante. Già considerando una soglia di 0,93 raggiunge un valore di 0,75 e con una soglia ≥0,95 è superiore a 0,80. Le altre metriche hanno una tendenza a decrescere per valori molto alti della soglia, ma risultano comunque soddisfacenti.

Poiché dalle analisi non risulta esserci un modello predominante sugli altri, abbiamo deciso di portare avanti tutti e quattro questi modelli, scegliendo delle soglie appropriate per ognuno di essi. Nello specifico:

- Per la random forest consideriamo una soglia di 0,995 (specificity = 0,687 sul validation)

Prediction \ Reference	no	yes
no	1902	21
yes	117	46

Kappa: 0,3714

- Per la PLS consideriamo una soglia di 0,970 (specificity = 0,910 sul validation)

Prediction \ Reference	no	yes
no	1585	6
yes	434	61

Kappa: 0,1701

- Per il gradient boosting consideriamo una soglia di 0,995 (specificity = 0,761 sul validation)

Prediction \ Reference	no	yes
no	1760	16
yes	259	51

Kappa: 0,2299

- Per il Lasso consideriamo varie soglie:
  - 0,930 (specificity = 0,746 sul validation)

Prediction \ Reference	no	yes
no	1861	17
yes	158	50

Kappa: 0,3311

  - 0,950 (specificity = 0,801 sul validation)

Prediction \ Reference	no	yes
no	1831	13
yes	188	54

Kappa: 0,3151

  - 0,980 (specificity = 0,896 sul validation)

Prediction \ Reference	no	yes
no	1653	7
yes	366	60

Kappa: 0,1989

  - 0,995 (specificity = 0,985 sul validation)

Prediction \ Reference	no	yes
no	1304	1
yes	715	66

Kappa: 0,1026

Alla fine, la nostra scelta è ricaduta sulla random forest, come inizialmente suggerito dalle curve ROC e lift. Questo perché, nonostante avessimo individuato altri modelli con una specificity maggiore sui dati di validation, la random forest fornisce buoni risultati su tutte le metriche. In particolare, abbiamo tenuto in considerazione il valore del Kappa, che ci ha suggerito che questo fosse il modello con la minore influenza della componente casuale. La soglia molto elevata risulta essere realistica in questa situazione, poiché bisogna essere praticamente certi per poter classificare un caso come non bancarotta.

Step 4: score

Con il modello e la soglia prescelti abbiamo infine effettuato lo score dei nuovi casi (utilizzando il dataset ricavato inizialmente).

```
          Reference
Prediction  no yes
       no  606   2
       yes  53  20

               Accuracy : 0.9192
                 95% CI : (0.8962, 0.9386)
    No Information Rate : 0.9677
    P-Value [Acc > NIR] : 1

                  Kappa : 0.3908

 Mcnemar's Test P-Value : 0.00000000001562

            Sensitivity : 0.9196
            Specificity : 0.9091
         Pos Pred Value : 0.9967
         Neg Pred Value : 0.2740
             Prevalence : 0.9677
         Detection Rate : 0.8899
   Detection Prevalence : 0.8928
      Balanced Accuracy : 0.9143

       'Positive' Class : no
```

Sui nuovi dati il modello risulta effettivamente molto performante. La specificity ottenuta è infatti superiore a 0,90, con solo 2 casi di bancarotta previsti erroneamente su 22. Inoltre, nonostante la soglia molto elevata, anche la sensitivity risulta alquanto elevata (maggiore di 0,90), quindi sono in numero contenuto anche i casi erroneamente previsti come bancarotta. Il modello sembra quindi avere una buona capacità di generalizzazione.