Call:

randomForest(formula = neighbourhood\_group ~ ., data = train, )

Type of random forest: classification

Number of trees: 500

No. of variables tried at each split: 2

OOB estimate of error rate: 0.12%

Confusion matrix:

Bronx Brooklyn Manhattan Queens Staten Island class.error

Bronx 654 0 6 0 0 0.0090909091

Brooklyn 0 12034 1 12 0 0.0010791068

Manhattan 2 1 12984 0 0 0.0002310002

Queens 0 13 1 3418 0 0.0040792541

Staten Island 0 0 0 0 211 0.0000000000

Random Forest

29337 samples

4 predictor

5 classes: 'Bronx', 'Brooklyn', 'Manhattan', 'Queens', 'Staten Island'

No pre-processing

Resampling: Cross-Validated (10 fold)

Summary of sample sizes: 26403, 26403, 26405, 26402, 26402, 26404, ...

Resampling results across tuning parameters:

mtry Accuracy Kappa

2 0.9983298 0.9973106

3 0.9987048 0.9979145

5 0.9986707 0.9978597

Accuracy was used to select the optimal model using the largest value.

The final value used for the model was mtry = 3.

Random Forest

29337 samples

4 predictor

5 classes: 'Bronx', 'Brooklyn', 'Manhattan', 'Queens', 'Staten Island'

No pre-processing

Resampling: Cross-Validated (10 fold)

Summary of sample sizes: 26403, 26404, 26402, 26403, 26404, 26403, ...

Resampling results across tuning parameters:

mtry Accuracy Kappa

1 0.9200327 0.8677874

2 0.9976481 0.9962118

3 0.9986707 0.9978598

4 0.9987047 0.9979146

5 0.9983980 0.9974206

6 0.9983639 0.9973658

7 0.9983298 0.9973108

8 0.9983639 0.9973657

9 0.9985002 0.9975851

10 0.9983980 0.9974206

Accuracy was used to select the optimal model using the largest value.

The final value used for the model was mtry = 4.

Call:

summary.resamples(object = results\_mtry)

Models: 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

Number of resamples: 10

Accuracy

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

5 0.8656666 0.8715479 0.8765976 0.8750376 0.8801125 0.8813906 0

6 0.8663484 0.8721446 0.8774499 0.8761284 0.8810802 0.8830948 0

7 0.8666894 0.8724003 0.8772794 0.8759579 0.8811350 0.8827539 0

8 0.8895329 0.8954995 0.8999485 0.8986256 0.9035446 0.9052488 0

9 0.9246505 0.9288271 0.9319925 0.9316558 0.9342252 0.9389911 0

10 0.9243354 0.9291681 0.9328447 0.9319967 0.9346512 0.9389911 0

11 0.9246505 0.9293386 0.9335263 0.9325419 0.9366214 0.9389911 0

12 0.9376278 0.9402489 0.9429007 0.9447785 0.9464894 0.9618399 0

13 0.9556919 0.9608762 0.9616499 0.9614479 0.9621645 0.9652352 0

14 0.9577369 0.9623349 0.9636891 0.9639358 0.9644683 0.9741056 0

15 0.9676210 0.9700895 0.9718766 0.9717761 0.9729039 0.9764906 0

Kappa

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

5 0.7802606 0.7908466 0.7982034 0.7960216 0.8043544 0.8063111 0

6 0.7813602 0.7918119 0.7997104 0.7978040 0.8060856 0.8090974 0

7 0.7819219 0.7922254 0.7994336 0.7975420 0.8057028 0.8085374 0

8 0.8201100 0.8306541 0.8372179 0.8353279 0.8431493 0.8459381 0

9 0.8770940 0.8841116 0.8893748 0.8887770 0.8930896 0.9006201 0

10 0.8769022 0.8846327 0.8907274 0.8893128 0.8937624 0.9006149 0

11 0.8770695 0.8849294 0.8918706 0.8902240 0.8970968 0.9005266 0

12 0.8985794 0.9027588 0.9071456 0.9101798 0.9129655 0.9381335 0

13 0.9280203 0.9365554 0.9375866 0.9373426 0.9383817 0.9434528 0

14 0.9312982 0.9387597 0.9408976 0.9413020 0.9420383 0.9580619 0

15 0.9471849 0.9513132 0.9541719 0.9540525 0.9559104 0.9618177 0

Call:

summary.resamples(object = results\_mtry)

Models: 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30

Number of resamples: 10

Accuracy

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

5 0.8656666 0.8715479 0.8765976 0.8750376 0.8801125 0.8813906 0

6 0.8663484 0.8721446 0.8774499 0.8761284 0.8810802 0.8830948 0

7 0.8666894 0.8724003 0.8772794 0.8759579 0.8811350 0.8827539 0

8 0.8895329 0.8954995 0.8999485 0.8986256 0.9035446 0.9052488 0

9 0.9246505 0.9288271 0.9319925 0.9316558 0.9342252 0.9389911 0

10 0.9243354 0.9291681 0.9328447 0.9319967 0.9346512 0.9389911 0

11 0.9246505 0.9293386 0.9335263 0.9325419 0.9366214 0.9389911 0

12 0.9376278 0.9402489 0.9429007 0.9447785 0.9464894 0.9618399 0

13 0.9556919 0.9608762 0.9616499 0.9614479 0.9621645 0.9652352 0

14 0.9577369 0.9623349 0.9636891 0.9639358 0.9644683 0.9741056 0

15 0.9676210 0.9700895 0.9718766 0.9717761 0.9729039 0.9764906 0

16 0.9669393 0.9699191 0.9723879 0.9720147 0.9737560 0.9757927 0

17 0.9679618 0.9703401 0.9720470 0.9721169 0.9736708 0.9761499 0

18 0.9689843 0.9734083 0.9749446 0.9745712 0.9761418 0.9792164 0

19 0.9720518 0.9746912 0.9773307 0.9770255 0.9796353 0.9812479 0

20 0.9723926 0.9759694 0.9785240 0.9778435 0.9801466 0.9819298 0

21 0.9747699 0.9775885 0.9798876 0.9794797 0.9821019 0.9833049 0

22 0.9792022 0.9804823 0.9821033 0.9820361 0.9834682 0.9846678 0

23 0.9798909 0.9837239 0.9844894 0.9846608 0.9869600 0.9877342 0

24 0.9829526 0.9857691 0.9868758 0.9870129 0.9882413 0.9904535 0

25 0.9832936 0.9869632 0.9887506 0.9880696 0.9893508 0.9907944 0

26 0.9836345 0.9877301 0.9889192 0.9883423 0.9893490 0.9914763 0

27 0.9832936 0.9878153 0.9890915 0.9885468 0.9894360 0.9914763 0

28 0.9836345 0.9882413 0.9890915 0.9889557 0.9900298 0.9921635 0

29 0.9884117 0.9888349 0.9904550 0.9907624 0.9926714 0.9942078 0

30 0.9887487 0.9928406 0.9942059 0.9935235 0.9952277 0.9959100 0

Kappa

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

5 0.7802606 0.7908466 0.7982034 0.7960216 0.8043544 0.8063111 0

6 0.7813602 0.7918119 0.7997104 0.7978040 0.8060856 0.8090974 0

7 0.7819219 0.7922254 0.7994336 0.7975420 0.8057028 0.8085374 0

8 0.8201100 0.8306541 0.8372179 0.8353279 0.8431493 0.8459381 0

9 0.8770940 0.8841116 0.8893748 0.8887770 0.8930896 0.9006201 0

10 0.8769022 0.8846327 0.8907274 0.8893128 0.8937624 0.9006149 0

11 0.8770695 0.8849294 0.8918706 0.8902240 0.8970968 0.9005266 0

12 0.8985794 0.9027588 0.9071456 0.9101798 0.9129655 0.9381335 0

13 0.9280203 0.9365554 0.9375866 0.9373426 0.9383817 0.9434528 0

14 0.9312982 0.9387597 0.9408976 0.9413020 0.9420383 0.9580619 0

15 0.9471849 0.9513132 0.9541719 0.9540525 0.9559104 0.9618177 0

16 0.9460847 0.9509720 0.9549663 0.9543924 0.9572343 0.9606380 0

17 0.9477418 0.9516688 0.9543976 0.9545595 0.9570974 0.9612346 0

18 0.9494291 0.9566474 0.9591933 0.9585922 0.9611581 0.9662681 0

19 0.9544810 0.9588131 0.9631404 0.9626357 0.9668982 0.9695821 0

20 0.9550385 0.9609128 0.9650824 0.9639740 0.9677329 0.9706983 0

21 0.9589161 0.9635676 0.9673327 0.9666633 0.9709516 0.9729664 0

22 0.9662215 0.9683124 0.9709669 0.9708625 0.9731938 0.9751807 0

23 0.9673861 0.9736159 0.9748722 0.9751518 0.9788992 0.9801809 0

24 0.9723834 0.9769503 0.9787634 0.9789840 0.9809769 0.9845705 0

25 0.9729353 0.9789103 0.9817960 0.9806995 0.9827877 0.9851240 0

26 0.9734871 0.9801559 0.9820741 0.9811422 0.9827731 0.9862306 0

27 0.9729353 0.9802937 0.9823499 0.9814733 0.9829256 0.9862304 0

28 0.9734976 0.9809785 0.9823536 0.9821395 0.9838786 0.9873548 0

29 0.9812530 0.9819633 0.9845639 0.9850726 0.9881660 0.9906606 0

30 0.9818284 0.9884391 0.9906586 0.9895489 0.9923063 0.9934078 0

Call:

summary.resamples(object = results\_tree)

Models: 250, 300, 350, 400, 450, 500, 550, 600, 800, 1000, 2000

Number of resamples: 10

Accuracy

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

250 0.9826176 0.9857666 0.9867054 0.9863312 0.9878122 0.9884078 0

300 0.9826117 0.9857666 0.9865349 0.9862971 0.9875659 0.9884078 0

350 0.9826176 0.9857666 0.9865349 0.9862971 0.9877363 0.9884078 0

400 0.9839755 0.9856814 0.9865349 0.9865016 0.9876470 0.9880668 0

450 0.9836345 0.9857666 0.9868758 0.9866379 0.9879826 0.9884196 0

500 0.9839755 0.9857666 0.9867054 0.9867061 0.9879826 0.9887487 0

550 0.9839755 0.9857666 0.9870462 0.9867402 0.9876511 0.9887487 0

600 0.9843164 0.9857666 0.9870462 0.9867744 0.9876470 0.9890897 0

800 0.9843164 0.9860259 0.9868823 0.9869789 0.9882373 0.9890897 0

1000 0.9843164 0.9861075 0.9868758 0.9868425 0.9878996 0.9887487 0

2000 0.9843164 0.9864519 0.9870527 0.9870811 0.9879816 0.9887526 0

Kappa

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

250 0.9717933 0.9769595 0.9784741 0.9778731 0.9802956 0.9812810 0

300 0.9718130 0.9769595 0.9781983 0.9778203 0.9798863 0.9812669 0

350 0.9717933 0.9769480 0.9782209 0.9778170 0.9801619 0.9812669 0

400 0.9740219 0.9768000 0.9782227 0.9781537 0.9800127 0.9807336 0

450 0.9734745 0.9769470 0.9787755 0.9783747 0.9805705 0.9812766 0

500 0.9740266 0.9769470 0.9785001 0.9784851 0.9805705 0.9818339 0

550 0.9740219 0.9769595 0.9790259 0.9785409 0.9800288 0.9818339 0

600 0.9745788 0.9769595 0.9790310 0.9785958 0.9800127 0.9823815 0

800 0.9745788 0.9773733 0.9787621 0.9789286 0.9809803 0.9823815 0

1000 0.9745871 0.9774997 0.9787755 0.9787082 0.9804313 0.9818339 0

2000 0.9745934 0.9780462 0.9790601 0.9790966 0.9805729 0.9818339 0

* mtry=4: 4 features is chosen for each iteration
* maxnodes = 30: Maximum 30 nodes in the terminal nodes (leaves)
* ntree =2000: 2000 trees will be trained

> fit\_rf

Random Forest

29337 samples

4 predictor

5 classes: 'Bronx', 'Brooklyn', 'Manhattan', 'Queens', 'Staten Island'

No pre-processing

Resampling: Cross-Validated (10 fold)

Summary of sample sizes: 26402, 26403, 26404, 26404, 26403, 26402, ...

Resampling results:

Accuracy Kappa

0.9943075 0.9908197

Tuning parameter 'mtry' was held constant at a value of 4

> conf

Confusion Matrix and Statistics

Reference

Prediction Bronx Brooklyn Manhattan Queens Staten Island

Bronx 423 0 6 0 0

Brooklyn 0 8055 4 50 0

Manhattan 8 0 8664 30 0

Queens 0 2 0 2154 0

Staten Island 0 0 0 0 162

Overall Statistics

Accuracy : 0.9949

95% CI : (0.9938, 0.9958)

No Information Rate : 0.4435

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.9917

Mcnemar's Test P-Value : NA

Statistics by Class:

Class: Bronx Class: Brooklyn Class: Manhattan Class: Queens

Sensitivity 0.98144 0.9998 0.9988 0.9642

Specificity 0.99969 0.9953 0.9965 0.9999

Pos Pred Value 0.98601 0.9933 0.9956 0.9991

Neg Pred Value 0.99958 0.9998 0.9991 0.9954

Prevalence 0.02204 0.4120 0.4435 0.1142

Detection Rate 0.02163 0.4119 0.4430 0.1101

Detection Prevalence 0.02193 0.4146 0.4449 0.1102

Balanced Accuracy 0.99056 0.9975 0.9977 0.9820

Class: Staten Island

Sensitivity 1.000000

Specificity 1.000000

Pos Pred Value 1.000000

Neg Pred Value 1.000000

Prevalence 0.008283

Detection Rate 0.008283

Detection Prevalence 0.008283

Balanced Accuracy 1.000000

> conf\_default

Confusion Matrix and Statistics

Reference

Prediction Bronx Brooklyn Manhattan Queens Staten Island

Bronx 429 0 1 0 0

Brooklyn 0 8056 0 12 0

Manhattan 2 0 8673 1 0

Queens 0 1 0 2221 0

Staten Island 0 0 0 0 162

Overall Statistics

Accuracy : 0.9991

95% CI : (0.9986, 0.9995)

No Information Rate : 0.4435

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.9986

Mcnemar's Test P-Value : NA

Statistics by Class:

Class: Bronx Class: Brooklyn Class: Manhattan Class: Queens

Sensitivity 0.99536 0.9999 0.9999 0.9942

Specificity 0.99995 0.9990 0.9997 0.9999

Pos Pred Value 0.99767 0.9985 0.9997 0.9995

Neg Pred Value 0.99990 0.9999 0.9999 0.9993

Prevalence 0.02204 0.4120 0.4435 0.1142

Detection Rate 0.02193 0.4119 0.4435 0.1136

Detection Prevalence 0.02199 0.4125 0.4436 0.1136

Balanced Accuracy 0.99765 0.9994 0.9998 0.9971

Class: Staten Island

Sensitivity 1.000000

Specificity 1.000000

Pos Pred Value 1.000000

Neg Pred Value 1.000000

Prevalence 0.008283

Detection Rate 0.008283

Detection Prevalence 0.008283

Balanced Accuracy 1.000000

After normalisation:

> rf

Call:

randomForest(formula = neighbourhood\_group ~ ., data = train, )

Type of random forest: classification

Number of trees: 500

No. of variables tried at each split: 2

OOB estimate of error rate: 0.09%

Confusion matrix:

1 2 3 4 5 class.error

1 12086 1 7 0 0 0.0006614850

2 0 12954 0 0 2 0.0001543686

3 14 1 3375 0 0 0.0044247788

4 0 0 0 249 0 0.0000000000

5 0 2 0 0 646 0.0030864198

> print(rf\_default)

Random Forest

29337 samples

4 predictor

5 classes: '1', '2', '3', '4', '5'

No pre-processing

Resampling: Cross-Validated (10 fold)

Summary of sample sizes: 26404, 26403, 26404, 26403, 26405, 26402, ...

Resampling results across tuning parameters:

mtry Accuracy Kappa

2 0.9981252 0.9969794

3 0.9990456 0.9984630

5 0.9988752 0.9981884

Accuracy was used to select the optimal model using the largest value.

The final value used for the model was mtry = 3.

> print(rf\_mtry)

Random Forest

29337 samples

4 predictor

5 classes: '1', '2', '3', '4', '5'

No pre-processing

Resampling: Cross-Validated (10 fold)

Summary of sample sizes: 26403, 26403, 26402, 26402, 26403, 26405, ...

Resampling results across tuning parameters:

mtry Accuracy Kappa

1 0.9248043 0.8753899

2 0.9975799 0.9961013

3 0.9987048 0.9979144

4 0.9986366 0.9978043

5 0.9985003 0.9975847

6 0.9985003 0.9975847

7 0.9985003 0.9975847

8 0.9984662 0.9975298

9 0.9983980 0.9974200

10 0.9984662 0.9975299

Accuracy was used to select the optimal model using the largest value.

The final value used for the model was mtry = 3.

> best\_mtry <- rf\_mtry$bestTune$mtry

> best\_mtry

[1] 3

> max(rf\_mtry$results$Accuracy)

[1] 0.9987048

> summary(results\_mtry)

Call:

summary.resamples(object = results\_mtry)

Models: 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15

Number of resamples: 10

Accuracy

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

5 0.8683941 0.8733271 0.8755964 0.8754131 0.8774710 0.8827539 0

6 0.8694170 0.8734977 0.8759372 0.8762994 0.8796012 0.8834356 0

7 0.8704398 0.8747869 0.8769393 0.8782764 0.8804533 0.8929789 0

8 0.8718910 0.8777056 0.8799861 0.8795720 0.8824031 0.8847989 0

9 0.8919195 0.9026156 0.9033248 0.9071470 0.9075574 0.9366053 0

10 0.9018070 0.9091066 0.9122208 0.9123293 0.9148410 0.9205862 0

11 0.9018070 0.9090987 0.9130732 0.9138288 0.9196342 0.9263804 0

12 0.9147921 0.9203034 0.9248083 0.9259635 0.9313399 0.9400136 0

13 0.9253324 0.9329983 0.9379790 0.9373144 0.9412918 0.9464894 0

14 0.9386294 0.9432322 0.9543286 0.9526191 0.9603816 0.9706885 0

15 0.9522510 0.9658433 0.9700016 0.9688779 0.9740945 0.9775051 0

Kappa

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

5 0.7842375 0.7929166 0.7959232 0.7960143 0.7998838 0.8084979 0

6 0.7856969 0.7931405 0.7970966 0.7975344 0.8029979 0.8095044 0

7 0.7879074 0.7954388 0.7988493 0.8009820 0.8046222 0.8249925 0

8 0.7898820 0.8001803 0.8038481 0.8030977 0.8079047 0.8119374 0

9 0.8235707 0.8404990 0.8417457 0.8481216 0.8487484 0.8969779 0

10 0.8392389 0.8508083 0.8565278 0.8565027 0.8609647 0.8700467 0

11 0.8391839 0.8508408 0.8579101 0.8589608 0.8685587 0.8799960 0

12 0.8605917 0.8698211 0.8774640 0.8790847 0.8876821 0.9024045 0

13 0.8780961 0.8905853 0.8987642 0.8976785 0.9042787 0.9126805 0

14 0.8998839 0.9074719 0.9256153 0.9227285 0.9353126 0.9522949 0

15 0.9221191 0.9443419 0.9511003 0.9493170 0.9579657 0.9634437 0

> summary(results\_mtry)

Call:

summary.resamples(object = results\_mtry)

Models: 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30

Number of resamples: 10

Accuracy

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

5 0.8683941 0.8733271 0.8755964 0.8754131 0.8774710 0.8827539 0

6 0.8694170 0.8734977 0.8759372 0.8762994 0.8796012 0.8834356 0

7 0.8704398 0.8747869 0.8769393 0.8782764 0.8804533 0.8929789 0

8 0.8718910 0.8777056 0.8799861 0.8795720 0.8824031 0.8847989 0

9 0.8919195 0.9026156 0.9033248 0.9071470 0.9075574 0.9366053 0

10 0.9018070 0.9091066 0.9122208 0.9123293 0.9148410 0.9205862 0

11 0.9018070 0.9090987 0.9130732 0.9138288 0.9196342 0.9263804 0

12 0.9147921 0.9203034 0.9248083 0.9259635 0.9313399 0.9400136 0

13 0.9253324 0.9329983 0.9379790 0.9373144 0.9412918 0.9464894 0

14 0.9386294 0.9432322 0.9543286 0.9526191 0.9603816 0.9706885 0

15 0.9522510 0.9658433 0.9700016 0.9688779 0.9740945 0.9775051 0

16 0.9536153 0.9631077 0.9693250 0.9679579 0.9740880 0.9778459 0

17 0.9539720 0.9596862 0.9660873 0.9664930 0.9738366 0.9781868 0

18 0.9532901 0.9584929 0.9712095 0.9676507 0.9753707 0.9792093 0

19 0.9689949 0.9751150 0.9759590 0.9761395 0.9772495 0.9815951 0

20 0.9696763 0.9754518 0.9764705 0.9767531 0.9795501 0.9809134 0

21 0.9747871 0.9759755 0.9776679 0.9781504 0.9805726 0.9815951 0

22 0.9751278 0.9772417 0.9778346 0.9784913 0.9792945 0.9826176 0

23 0.9754685 0.9780923 0.9790317 0.9794458 0.9813395 0.9829584 0

24 0.9751278 0.9773270 0.9790281 0.9797185 0.9815951 0.9846626 0

25 0.9764906 0.9786055 0.9802217 0.9806388 0.9828732 0.9850034 0

26 0.9764906 0.9790317 0.9798807 0.9808774 0.9834697 0.9853442 0

27 0.9771721 0.9792874 0.9809036 0.9814229 0.9840661 0.9867076 0

28 0.9785349 0.9798824 0.9819328 0.9824795 0.9845759 0.9877301 0

29 0.9788756 0.9801382 0.9821002 0.9828203 0.9855999 0.9884117 0

30 0.9812479 0.9816865 0.9838077 0.9839451 0.9858542 0.9880709 0

Kappa

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

5 0.7842375 0.7929166 0.7959232 0.7960143 0.7998838 0.8084979 0

6 0.7856969 0.7931405 0.7970966 0.7975344 0.8029979 0.8095044 0

7 0.7879074 0.7954388 0.7988493 0.8009820 0.8046222 0.8249925 0

8 0.7898820 0.8001803 0.8038481 0.8030977 0.8079047 0.8119374 0

9 0.8235707 0.8404990 0.8417457 0.8481216 0.8487484 0.8969779 0

10 0.8392389 0.8508083 0.8565278 0.8565027 0.8609647 0.8700467 0

11 0.8391839 0.8508408 0.8579101 0.8589608 0.8685587 0.8799960 0

12 0.8605917 0.8698211 0.8774640 0.8790847 0.8876821 0.9024045 0

13 0.8780961 0.8905853 0.8987642 0.8976785 0.9042787 0.9126805 0

14 0.8998839 0.9074719 0.9256153 0.9227285 0.9353126 0.9522949 0

15 0.9221191 0.9443419 0.9511003 0.9493170 0.9579657 0.9634437 0

16 0.9246333 0.9400576 0.9500516 0.9479193 0.9578830 0.9640427 0

17 0.9250872 0.9345036 0.9447964 0.9455006 0.9574695 0.9645999 0

18 0.9240240 0.9325549 0.9530810 0.9474084 0.9600377 0.9662515 0

19 0.9495271 0.9594761 0.9609378 0.9612262 0.9630564 0.9701532 0

20 0.9506274 0.9601459 0.9617636 0.9622303 0.9668356 0.9690432 0

21 0.9589652 0.9608643 0.9637467 0.9645061 0.9684677 0.9701940 0

22 0.9595158 0.9630901 0.9640139 0.9650684 0.9663869 0.9718150 0

23 0.9600770 0.9644137 0.9659786 0.9666242 0.9697271 0.9724218 0

24 0.9595158 0.9631631 0.9659688 0.9670607 0.9701425 0.9751557 0

25 0.9617605 0.9652414 0.9679184 0.9685693 0.9722332 0.9757026 0

26 0.9617625 0.9659358 0.9673551 0.9689505 0.9731997 0.9762673 0

27 0.9628112 0.9663303 0.9690261 0.9698426 0.9741757 0.9784937 0

28 0.9651063 0.9673085 0.9706994 0.9715648 0.9750008 0.9801473 0

29 0.9656829 0.9677287 0.9709635 0.9721250 0.9766687 0.9812604 0

30 0.9695399 0.9702718 0.9737585 0.9739666 0.9770805 0.9806985 0

> summary(results\_tree)

Call:

summary.resamples(object = results\_tree)

Models: 250, 300, 350, 400, 450, 500, 550, 600, 800, 1000, 2000

Number of resamples: 10

Accuracy

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

250 0.9758092 0.9794562 0.9803989 0.9800593 0.9810886 0.9829642 0

300 0.9764906 0.9794562 0.9803989 0.9802297 0.9815146 0.9829642 0

350 0.9764906 0.9795415 0.9807397 0.9802638 0.9815146 0.9826117 0

400 0.9764906 0.9794562 0.9803989 0.9801275 0.9812591 0.9822828 0

450 0.9761499 0.9794596 0.9809100 0.9801957 0.9818477 0.9819359 0

500 0.9761499 0.9794596 0.9807396 0.9801958 0.9817671 0.9822707 0

550 0.9761499 0.9794596 0.9807396 0.9800594 0.9812591 0.9819298 0

600 0.9758092 0.9795415 0.9807430 0.9800935 0.9815162 0.9819298 0

800 0.9761499 0.9795415 0.9805693 0.9800593 0.9811738 0.9822828 0

1000 0.9761499 0.9795415 0.9809102 0.9802297 0.9811738 0.9833049 0

2000 0.9744463 0.9792039 0.9805627 0.9798889 0.9810886 0.9833049 0

Kappa

Min. 1st Qu. Median Mean 3rd Qu. Max. NA's

250 0.9607608 0.9665836 0.9681477 0.9676251 0.9692816 0.9723920 0

300 0.9618792 0.9665836 0.9681471 0.9679014 0.9699750 0.9723852 0

350 0.9618767 0.9667232 0.9687294 0.9679578 0.9699669 0.9718068 0

400 0.9618767 0.9665872 0.9681696 0.9677368 0.9695464 0.9712754 0

450 0.9613172 0.9665984 0.9689917 0.9678482 0.9705295 0.9707017 0

500 0.9613172 0.9665984 0.9687376 0.9678489 0.9703820 0.9712592 0

550 0.9613172 0.9665984 0.9687399 0.9676266 0.9695475 0.9707017 0

600 0.9607652 0.9667270 0.9687309 0.9676820 0.9700013 0.9707017 0

800 0.9613172 0.9667232 0.9684562 0.9676243 0.9694078 0.9712669 0

1000 0.9613268 0.9667270 0.9689819 0.9679031 0.9694179 0.9729379 0

2000 0.9585297 0.9661693 0.9684267 0.9673454 0.9692815 0.9729379 0

onfusion Matrix and Statistics

Reference

Prediction 1 2 3 4 5

1 7991 4 105 0 0

2 12 8689 156 0 47

3 7 0 2014 0 0

4 0 0 0 124 0

5 0 12 1 0 396

Overall Statistics

Accuracy : 0.9824

95% CI : (0.9805, 0.9842)

No Information Rate : 0.4451

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.9714

Mcnemar's Test P-Value : NA

Statistics by Class:

Class: 1 Class: 2 Class: 3 Class: 4 Class: 5

Sensitivity 0.9976 0.9982 0.8849 1.00000 0.89391

Specificity 0.9906 0.9802 0.9996 1.00000 0.99932

Pos Pred Value 0.9865 0.9759 0.9965 1.00000 0.96822

Neg Pred Value 0.9983 0.9985 0.9851 1.00000 0.99755

Prevalence 0.4096 0.4451 0.1164 0.00634 0.02265

Detection Rate 0.4086 0.4443 0.1030 0.00634 0.02025

Detection Prevalence 0.4142 0.4553 0.1033 0.00634 0.02091

Balanced Accuracy 0.9941 0.9892 0.9422 1.00000 0.94661

> conf\_default

Confusion Matrix and Statistics

Reference

Prediction 1 2 3 4 5

1 7995 2 3 0 0

2 3 8702 1 0 3

3 12 0 2272 0 0

4 0 0 0 124 0

5 0 1 0 0 440

Overall Statistics

Accuracy : 0.9987

95% CI : (0.9981, 0.9992)

No Information Rate : 0.4451

P-Value [Acc > NIR] : < 2.2e-16

Kappa : 0.9979

Mcnemar's Test P-Value : NA

Statistics by Class:

Class: 1 Class: 2 Class: 3 Class: 4 Class: 5

Sensitivity 0.9981 0.9997 0.9982 1.00000 0.99323

Specificity 0.9996 0.9994 0.9993 1.00000 0.99995

Pos Pred Value 0.9994 0.9992 0.9947 1.00000 0.99773

Neg Pred Value 0.9987 0.9997 0.9998 1.00000 0.99984

Prevalence 0.4096 0.4451 0.1164 0.00634 0.02265

Detection Rate 0.4088 0.4449 0.1162 0.00634 0.02250

Detection Prevalence 0.4090 0.4453 0.1168 0.00634 0.02255

Balanced Accuracy 0.9988 0.9995 0.9988 1.00000 0.99659

> fit\_rf

Random Forest

29337 samples

4 predictor

5 classes: '1', '2', '3', '4', '5'

No pre-processing

Resampling: Cross-Validated (10 fold)

Summary of sample sizes: 26404, 26403, 26404, 26403, 26404, 26402, ...

Resampling results:

Accuracy Kappa

0.9830931 0.9725758

Tuning parameter 'mtry' was held constant at a value of 3

REGRESSION PRICE – RANGER

rangerReg

Ranger result

Call:

ranger(price ~ ., data = train, write.forest = TRUE, classification = F)

Type: Regression

Number of trees: 500

Sample size: 29337

Number of independent variables: 4

Mtry: 2

Target node size: 5

Variable importance mode: none

Splitrule: variance

OOB prediction error (MSE): 0.0004944917

R squared (OOB): 0.07553202

task = makeRegrTask(data = train, target = "price")

> estimateTimeTuneRanger(task, iters = 70, num.threads = 6, num.trees = 1000)

Approximated time for tuning: 6M 44S

> res = tuneRanger(task, measure = list(mse), num.trees = 1000, num.threads = 4, iters = 70, show.info = getOption("mlrMBO.show.info", TRUE))

# Mean of best 5 % of the results

> res

Recommended parameter settings:

mtry min.node.size sample.fraction

1 2 60 0.2010013

Results:

mse exec.time

1 0.0004812156 3.446

> # Model with the new tuned hyperparameters

> res$model

Model for learner.id=regr.ranger; learner.class=regr.ranger

Trained on: task.id = train; obs = 29337; features = 4

Hyperparameters: num.threads=4,verbose=FALSE,respect.unordered.factors=order,mtry=2,min.node.size=60,sample.fraction=0.201,num.trees=1e+03,replace=FALSE

> tuned\_rangerReg <- ranger( price~ ., data = train, write.forest = TRUE, classification = F, mtry= 2,

+ min.node.size = 60, sample.fraction = 0.201,num.trees = 1000, replace= FALSE)

> tuned\_rangerReg

Ranger result

Call:

ranger(price ~ ., data = train, write.forest = TRUE, classification = F, mtry = 2, min.node.size = 60, sample.fraction = 0.201, num.trees = 1000, replace = FALSE)

Type: Regression

Number of trees: 1000

Sample size: 29337

Number of independent variables: 4

Mtry: 2

Target node size: 60

Variable importance mode: none

Splitrule: variance

OOB prediction error (MSE): 0.0004817166

R squared (OOB): 0.09941546

### DOMANDE

#### Supervised

* Il dataset va bene?
* normalizzazione del dataset da 0 a 1? Oppure scale -1 and 1?
* Variabili da aggiungere al modello? Tutte?
* Modelli RF durano 15 minuti per tuning, va bene?

#### UNSUPERVISED

* OBIETTIVO: clustering possibile? Forse clustering tra i differenti quartieri riguardo le proprietà di lusso in base a posizione e prezzo
* KNN e PAM occupano troppa memoria in R studio: PCA è la soluzione? Utilizzare solo una parte di Dataset?
* Come poter interpretare i dati dei clusters?

#### Report

* Struttura del report?
* Teoria? PAM oppure RANGER ect.
* Template va bene?
* Dataset description in entrambi anche se uguale?

Tenere conto solo dei quartieri specifici: sia per prezzo che per clustering

Tenere conto anche solo di fasce di prezzo precise per il quartiere : utente x vorrebbe una casa di lusso? E’ una famiglia?

Cluster sui singoli quartieri: trovare fasce di prezzo