“Location, location, location!”

Boston

* Capital of the state of Massachusetts, USA
* First settled in 1630
* 5 million people in greater Boston Area, some of the highest population densities in America

Housing Data

* A paper was written on the relationship between **house prices** and **clean air** in the late 1970s by David Harrison of Harvard and Daniel Rubinfeld of U. of Michigan
* Data set widely used to evaluate algorithms

The R in CART

* Trees can also be used for **regression** - the output at each leaf of the tree is no longer a category, but a number
* Just like classification trees, **regression trees** can capture **nonlinearities** that linear regression can’t

Regression Trees

* With Classification Trees we report the average outcome at each leaf of our tree, e.g. if the outcome is “true” 15 times, and “false” 5 times, the value at the leaf is :

1515+5=0.75≥0.5−>true1515+5=0.75≥0.5−>true

* With Regression Trees, we have continuous variables, so we simply report the average of the values at that leaf:

3,4,5=53,4,5=5

Chart

Description automatically generated

Housing Data

* We will explore the dataset with the aid of trees
* Compute linear regression with regression trees
* Discussing what the “cp” parameter means
* Apply cross-validation to regression trees

Understanding the data

* Each entry to a census **tract**, a statistical division of the area that is used by researchers to break down towns and cities
* There will usually be multiple census tracts per **town**
* **LON** and **LAT** are the longitude and latitude of the center of the census tract
* **MEDV** is the median value of owner-occupied homes, in thousands of dollars
* **CRIM** is the per capita crime rate
* **ZN** is related to how much of the land is zoned for large residential properties
* **INDUS** is proportion of area used for industry
* **CHAS** is 1 if the census tract is next to the Charles River
* **NOX** is the concentration of nitrous oxides in the air
* **RM** is the average number of rooms per dwelling
* **AGE** is the proportion of owner-occupied units built before 1940
* **DIS** is a measure of how far the tract is from centers of employment in Boston
* **RAD** is a measure of closeness to important highways
* **TAX** is the property tax rate per $10,000 of value
* **PTRATIO** is the pupil-teacher ratio by town

The “cp” parameter

* “cp” stands for “complexity parameter”
* Intuition: having too many splits is bad for generalization, so we should penalize the **complexity**
* Our goal when building the tree is to minimize the RSS by making splits, but we want to penalize too many splits. Define **S** to be number of splits, and lambda is the penalty. Our goal is to find the tree that minimizes:

∑Leaves(RSSateachleaf)+λS∑Leaves(RSSateachleaf)+λS

* lambda = 0.5

Table

Description automatically generated

* If we pick a large value of lambda, we won’t make many splits because we pay a big price for every additional split that outweighs the decrease in “error”
* If we pick a small (or zero) value of lambda, we’ll make splits until it no longer decreases error
* The definition of “cp” is closely related to lambda
* Consider a tree with no splits - we simply take the average of the data. Calculate RSS for that tree, let us call it **RSS(no splits)**

cp=λRSS(nosplits)cp=λRSS(nosplits)

Unit 4, Recitation

Read in data

*# Read in data*

boston = **read.csv**("boston.csv")

*# Output structure*

**str**(boston)

## 'data.frame': 506 obs. of 16 variables:

## $ TOWN : Factor w/ 92 levels "Arlington","Ashland",..: 54 77 77 46 46 46 69 69 69 69 ...

## $ TRACT : int 2011 2021 2022 2031 2032 2033 2041 2042 2043 2044 ...

## $ LON : num -71 -71 -70.9 -70.9 -70.9 ...

## $ LAT : num 42.3 42.3 42.3 42.3 42.3 ...

## $ MEDV : num 24 21.6 34.7 33.4 36.2 28.7 22.9 22.1 16.5 18.9 ...

## $ CRIM : num 0.00632 0.02731 0.02729 0.03237 0.06905 ...

## $ ZN : num 18 0 0 0 0 0 12.5 12.5 12.5 12.5 ...

## $ INDUS : num 2.31 7.07 7.07 2.18 2.18 2.18 7.87 7.87 7.87 7.87 ...

## $ CHAS : int 0 0 0 0 0 0 0 0 0 0 ...

## $ NOX : num 0.538 0.469 0.469 0.458 0.458 0.458 0.524 0.524 0.524 0.524 ...

## $ RM : num 6.58 6.42 7.18 7 7.15 ...

## $ AGE : num 65.2 78.9 61.1 45.8 54.2 58.7 66.6 96.1 100 85.9 ...

## $ DIS : num 4.09 4.97 4.97 6.06 6.06 ...

## $ RAD : int 1 2 2 3 3 3 5 5 5 5 ...

## $ TAX : int 296 242 242 222 222 222 311 311 311 311 ...

## $ PTRATIO: num 15.3 17.8 17.8 18.7 18.7 18.7 15.2 15.2 15.2 15.2 ...

Plot observations

*# Plot observations*

**plot**(boston**$**LON, boston**$**LAT)

*# Tracts alongside the Charles River*

**points**(boston**$**LON[boston**$**CHAS**==**1], boston**$**LAT[boston**$**CHAS**==**1], col="blue", pch=19)

*# Plot MIT*

**points**(boston**$**LON[boston**$**TRACT**==**3531],boston**$**LAT[boston**$**TRACT**==**3531],col="red", pch=20)

*# Plot polution*

**summary**(boston**$**NOX)

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## 0.3850 0.4490 0.5380 0.5547 0.6240 0.8710

**points**(boston**$**LON[boston**$**NOX**>=**0.55], boston**$**LAT[boston**$**NOX**>=**0.55], col="green", pch=20)

Chart, scatter chart

Description automatically generated

*# Plot prices*

**plot**(boston**$**LON, boston**$**LAT)

**summary**(boston**$**MEDV)

## Min. 1st Qu. Median Mean 3rd Qu. Max.

## 5.00 17.02 21.20 22.53 25.00 50.00

**points**(boston**$**LON[boston**$**MEDV**>=**21.2], boston**$**LAT[boston**$**MEDV**>=**21.2], col="red", pch=20)

Chart, scatter chart

Description automatically generated

Linear Regression

*# Linear Regression using LAT and LON*

**plot**(boston**$**LAT, boston**$**MEDV)

Chart, scatter chart

Description automatically generated

**plot**(boston**$**LON, boston**$**MEDV)

Chart, scatter chart

Description automatically generated

latlonlm = **lm**(MEDV **~** LAT **+** LON, data=boston)

**summary**(latlonlm)

##

## Call:

## lm(formula = MEDV ~ LAT + LON, data = boston)

##

## Residuals:

## Min 1Q Median 3Q Max

## -16.460 -5.590 -1.299 3.695 28.129

##

## Coefficients:

## Estimate Std. Error t value Pr(>|t|)

## (Intercept) -3178.472 484.937 -6.554 1.39e-10 \*\*\*

## LAT 8.046 6.327 1.272 0.204

## LON -40.268 5.184 -7.768 4.50e-14 \*\*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 8.693 on 503 degrees of freedom

## Multiple R-squared: 0.1072, Adjusted R-squared: 0.1036

## F-statistic: 30.19 on 2 and 503 DF, p-value: 4.159e-13

*# Visualize regression output*

**plot**(boston**$**LON, boston**$**LAT)

**points**(boston**$**LON[boston**$**MEDV**>=**21.2], boston**$**LAT[boston**$**MEDV**>=**21.2], col="red", pch=20)

latlonlm**$**fitted.values

## 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22

## 18.75633 18.81648 18.21651 17.97483 17.77344 17.60024 18.32916 18.49416 18.32904 18.20015 18.44176 18.81222 19.00560 19.43658 19.69836 19.93589 20.39492 19.92388 20.48766 20.26703 20.08099 20.05277

## 23 24 25 26 27 28 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44

## 19.85547 19.67426 19.54619 19.35208 19.20306 19.16685 19.03801 18.78031 19.43265 19.29173 19.61388 19.91187 19.79915 20.68910 21.04745 20.98293 21.63527 21.55856 22.33163 21.37316 21.24036 20.21512

## 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 63 64 65 66

## 19.75853 19.88344 19.58142 19.25924 19.25115 19.18508 19.23088 19.74790 19.60931 20.38652 22.21046 20.07213 18.70708 18.66683 18.82408 18.77184 18.40938 18.33295 18.02687 17.51943 15.65495 23.10462

## 67 68 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88

## 24.13141 24.84590 25.60131 25.61752 25.93170 25.34631 26.20561 25.81913 25.29164 24.85675 24.42188 23.90640 24.53454 24.63514 23.75323 23.82969 23.85779 23.38269 22.72237 22.94390 22.29957 22.40842

## 89 90 91 92 93 94 95 96 97 98 99 100 101 102 103 104 105 106 107 108 109 110

## 22.45762 22.14675 21.91325 22.41258 22.92793 23.39494 23.27825 23.80985 24.17226 24.28506 24.63140 23.89449 23.25432 23.74970 23.81005 23.62081 23.29062 23.13361 22.97737 22.72687 22.74697 22.97647

## 111 112 113 114 115 116 117 118 119 120 121 122 123 124 125 126 127 128 129 130 131 132

## 22.86770 22.52944 22.52548 22.31604 22.13247 21.99392 22.06474 21.77238 21.74020 21.37777 21.49303 21.78858 22.02217 21.96582 21.72822 21.52770 21.78464 22.54575 22.74307 22.99111 23.22626 23.46786

## 133 134 135 136 137 138 139 140 141 142 143 144 145 146 147 148 149 150 151 152 153 154

## 23.45175 23.66916 23.51215 23.43729 23.12160 22.92429 22.84380 22.65451 22.46283 22.52568 22.23739 22.40088 22.34452 22.45326 22.67069 22.53941 22.60221 22.64652 22.81242 22.79553 22.63851 22.84389

## 155 156 157 158 159 160 161 162 163 164 165 166 167 168 169 170 171 172 173 174 175 176

## 22.92200 22.98483 22.97271 23.13372 23.02100 22.92276 23.11686 23.31252 23.42771 23.60892 24.03575 23.66526 23.47196 23.89072 23.37207 23.52911 23.68372 23.62894 23.96395 23.93091 23.86243 24.37376

## 177 178 179 180 181 182 183 184 185 186 187 188 189 190 191 192 193 194 195 196 197 198

## 25.09862 24.80630 24.38995 24.48101 24.41018 24.24272 24.40379 24.65586 24.87733 25.13904 24.97145 25.49323 26.23821 26.48860 25.58575 26.72127 26.15347 27.07142 27.53046 28.15904 29.56023 30.37365

## 199 200 201 202 203 204 205 206 207 208 209 210 211 212 213 214 215 216 217 218 219 220

## 29.78157 30.29335 30.89753 28.88834 28.88817 28.39678 28.18598 26.42353 26.41397 26.34152 26.35525 26.05888 26.04683 25.85673 25.70288 25.94843 25.52159 25.36218 25.00625 24.54316 24.05995 24.54320

## 221 222 223 224 225 226 227 228 229 230 231 232 233 234 235 236 237 238 239 240 241 242

## 24.66565 25.13194 25.26726 25.13756 24.60370 24.18094 25.03862 24.75439 24.43473 24.76255 25.26424 25.24407 25.72003 25.55405 25.55558 25.79638 26.19426 26.15406 28.30049 28.21593 27.75302 28.44966

## 243 244 245 246 247 248 249 250 251 252 253 254 255 256 257 258 259 260 261 262 263 264

## 28.80955 29.24691 29.63356 30.18767 30.42681 29.83324 30.20688 29.95153 29.63737 29.91678 30.88568 31.20960 31.44166 30.54133 28.66323 22.91885 23.11536 23.26677 23.37955 23.32319 23.48831 23.12993

## 265 266 267 268 269 270 271 272 273 274 275 276 277 278 279 280 281 282 283 284 285 286

## 23.00909 22.94468 23.01719 23.48836 23.91107 23.70209 23.15454 23.39217 23.63371 24.44305 25.11943 25.46811 25.48752 25.92879 25.59288 25.98907 26.70586 27.45326 27.06033 26.66993 26.65411 27.91862

## 287 288 289 290 291 292 293 294 295 296 297 298 299 300 301 302 303 304 305 306 307 308

## 26.91216 25.10000 24.60860 25.54682 25.08756 25.00528 23.87704 23.79102 24.24758 24.72601 24.56256 24.21390 23.93226 23.01827 23.27587 22.80456 21.85417 22.96544 21.55585 22.03901 21.60809 20.90341

## 309 310 311 312 313 314 315 316 317 318 319 320 321 322 323 324 325 326 327 328 329 330

## 20.32355 20.61345 20.57708 20.25089 20.46440 20.11653 19.74773 18.96246 19.22830 19.82432 20.12628 20.53701 19.89691 19.49419 19.21390 18.95456 19.11170 19.37440 19.44197 19.89697 20.53328 21.11313

## 331 332 333 334 335 336 337 338 339 340 341 342 343 344 345 346 347 348 349 350 351 352

## 20.31192 19.67577 19.07574 18.49177 17.91995 18.23393 18.58014 17.93585 18.14436 18.36659 18.31661 15.16523 16.38123 17.16657 16.78823 16.96581 17.17110 16.23690 16.11270 13.72375 12.71715 12.29463

## 353 354 355 356 357 358 359 360 361 362 363 364 365 366 367 368 369 370 371 372 373 374

## 11.34041 12.06130 13.01563 12.81849 23.60656 24.04794 24.26138 23.92718 23.79674 23.68957 23.45200 23.72980 22.58058 22.58221 22.34222 22.22785 22.14326 22.22781 21.94108 21.93382 21.87098 21.65754

## 375 376 377 378 379 380 381 382 383 384 385 386 387 388 389 390 391 392 393 394 395 396

## 21.66964 21.79684 21.70421 21.86931 21.93617 21.95789 22.21961 22.01023 21.28298 21.20890 21.30154 21.24358 21.22987 21.19120 21.11629 21.02849 20.80619 20.53234 21.09620 20.80642 20.92563 21.00858

## 397 398 399 400 401 402 403 404 405 406 407 408 409 410 411 412 413 414 415 416 417 418

## 21.16724 21.15191 21.12448 21.48050 21.44749 21.34601 21.32429 21.44671 21.51596 21.46034 21.85567 21.91777 21.92585 22.00638 22.04664 22.12558 22.04264 21.95407 21.80510 21.92029 22.01853 22.13689

## 419 420 421 422 423 424 425 426 427 428 429 430 431 432 433 434 435 436 437 438 439 440

## 22.23113 22.48482 22.60643 22.75782 22.77960 22.57667 22.43736 22.42283 22.32218 22.27706 22.17560 22.01454 22.24007 22.06290 22.10238 21.97033 21.91152 21.87123 21.87201 21.85105 21.86795 21.56755

## 441 442 443 444 445 446 447 448 449 450 451 452 453 454 455 456 457 458 459 460 461 462

## 21.35816 21.39446 21.40651 21.56114 21.69239 21.74879 21.59015 21.48709 21.59180 21.72467 21.82537 21.61196 21.32605 21.55157 21.77144 22.00257 21.99455 21.95834 21.78922 21.60801 21.63859 21.33012

## 463 464 465 466 467 468 469 470 471 472 473 474 475 476 477 478 479 480 481 482 483 484

## 21.12881 21.30601 21.64832 22.05905 22.08721 22.60660 22.68721 22.68315 22.84829 23.20505 23.27911 22.94481 22.20387 22.08625 22.30451 22.18770 22.30287 22.77962 23.97576 23.73415 23.48051 23.73667

## 485 486 487 488 489 490 491 492 493 494 495 496 497 498 499 500 501 502 503 504 505 506

## 22.99336 22.70752 22.64302 22.42955 21.16374 21.31355 21.40855 21.07754 21.68151 21.00096 21.03154 20.98882 20.58857 20.31154 20.69332 20.41146 20.10948 19.81316 19.98473 20.12569 19.81563 19.59015

**points**(boston**$**LON[latlonlm**$**fitted.values **>=** 21.2], boston**$**LAT[latlonlm**$**fitted.values **>=** 21.2], col="blue", pch="$")

Chart, scatter chart

Description automatically generated

CART

*# Load CART packages*

**library**(rpart)

**library**(rpart.plot)

*# CART model*

latlontree = **rpart**(MEDV **~** LAT **+** LON, data=boston)

**prp**(latlontree)

Diagram

Description automatically generated with medium confidence

*# Visualize output*

**plot**(boston**$**LON, boston**$**LAT)

**points**(boston**$**LON[boston**$**MEDV**>=**21.2], boston**$**LAT[boston**$**MEDV**>=**21.2], col="red", pch=20)

fittedvalues = **predict**(latlontree)

**points**(boston**$**LON[fittedvalues**>**21.2], boston**$**LAT[fittedvalues**>=**21.2], col="blue", pch="$")

Chart, scatter chart

Description automatically generated

*# Simplify tree by increasing minbucket*

latlontree = **rpart**(MEDV **~** LAT **+** LON, data=boston, minbucket=50)

**plot**(latlontree)

**text**(latlontree)

Diagram, schematic

Description automatically generated

*# Visualize Output*

**plot**(boston**$**LON,boston**$**LAT)

**abline**(v=**-**71.07)

**abline**(h=42.21)

**abline**(h=42.17)

**points**(boston**$**LON[boston**$**MEDV**>=**21.2], boston**$**LAT[boston**$**MEDV**>=**21.2], col="red", pch=20)

Chart, scatter chart

Description automatically generated

Split the data

*# Split the data*

**library**(caTools)

**set.seed**(123)

split = **sample.split**(boston**$**MEDV, SplitRatio = 0.7)

train = **subset**(boston, split**==**TRUE)

test = **subset**(boston, split**==**FALSE)

Linear Regression

*# Create linear regression*

linreg = **lm**(MEDV **~** LAT **+** LON **+** CRIM **+** ZN **+** INDUS **+** CHAS **+** NOX **+** RM **+** AGE **+** DIS **+** RAD **+** TAX **+** PTRATIO, data=train)

**summary**(linreg)

##

## Call:

## lm(formula = MEDV ~ LAT + LON + CRIM + ZN + INDUS + CHAS + NOX +

## RM + AGE + DIS + RAD + TAX + PTRATIO, data = train)

##

## Residuals:

## Min 1Q Median 3Q Max

## -14.511 -2.712 -0.676 1.793 36.883

##

## Coefficients:

## Estimate Std. Error t value Pr(>|t|)

## (Intercept) -252.290511 436.710080 -0.578 0.5638

## LAT 1.543965 5.191959 0.297 0.7664

## LON -2.987111 4.785834 -0.624 0.5329

## CRIM -0.180802 0.043905 -4.118 0.0000477248 \*\*\*

## ZN 0.032503 0.018773 1.731 0.0843 .

## INDUS -0.042968 0.084731 -0.507 0.6124

## CHAS 2.904178 1.220052 2.380 0.0178 \*

## NOX -21.612794 5.413707 -3.992 0.0000797779 \*\*\*

## RM 6.284395 0.482704 13.019 < 2e-16 \*\*\*

## AGE -0.044305 0.017854 -2.482 0.0135 \*

## DIS -1.577356 0.284182 -5.551 0.0000000563 \*\*\*

## RAD 0.245089 0.097284 2.519 0.0122 \*

## TAX -0.011123 0.005452 -2.040 0.0421 \*

## PTRATIO -0.983488 0.193886 -5.072 0.0000006382 \*\*\*

## ---

## Signif. codes: 0 '\*\*\*' 0.001 '\*\*' 0.01 '\*' 0.05 '.' 0.1 ' ' 1

##

## Residual standard error: 5.595 on 350 degrees of freedom

## Multiple R-squared: 0.665, Adjusted R-squared: 0.6525

## F-statistic: 53.43 on 13 and 350 DF, p-value: < 2.2e-16

*# Make predictions*

linreg.pred = **predict**(linreg, newdata=test)

linreg.sse = **sum**((linreg.pred **-** test**$**MEDV)**^**2)

linreg.sse

## [1] 3037.088

CART

*# Create a CART model*

tree = **rpart**(MEDV **~** LAT **+** LON **+** CRIM **+** ZN **+** INDUS **+** CHAS **+** NOX **+** RM **+** AGE **+** DIS **+** RAD **+** TAX **+** PTRATIO, data=train)

**prp**(tree)

Diagram

Description automatically generated with low confidence

*# Make predictions*

tree.pred = **predict**(tree, newdata=test)

tree.sse = **sum**((tree.pred **-** test**$**MEDV)**^**2)

tree.sse

## [1] 4328.988

Cross-Validation

*# Load libraries for cross-validation*

**library**(caret)

**library**(e1071)

*# Number of folds*

tr.control = **trainControl**(method = "cv", number = 10)

*# cp values*

cp.grid = **expand.grid**( .cp = (0**:**10)**\***0.001)

*# What did we just do?*

1**\***0.001

## [1] 0.001

10**\***0.001

## [1] 0.01

0**:**10

## [1] 0 1 2 3 4 5 6 7 8 9 10

0**:**10 **\*** 0.001

## [1] 0.000 0.001 0.002 0.003 0.004 0.005 0.006 0.007 0.008 0.009 0.010

*# Cross-validation*

tr = **train**(MEDV **~** LAT **+** LON **+** CRIM **+** ZN **+** INDUS **+** CHAS **+** NOX **+** RM **+** AGE **+** DIS **+** RAD **+** TAX **+** PTRATIO, data = train, method = "rpart", trControl = tr.control, tuneGrid = cp.grid)

*# Extract tree*

best.tree = tr**$**finalModel

**prp**(best.tree)

Chart

Description automatically generated with low confidence

*# Make predictions*

best.tree.pred = **predict**(best.tree, newdata=test)

best.tree.sse = **sum**((best.tree.pred **-** test**$**MEDV)**^**2)

best.tree.sse

## [1] 3660.149