

# CART, RF, Boosting

12/13/2019

```
library(dplyr) # data manipulation

##
## Attaching package: 'dplyr'
## The following objects are masked from 'package:stats':
##
##   filter, lag
## The following objects are masked from 'package:base':
##
##   intersect, setdiff, setequal, union

library(caTools) # splits
library(ggplot2) # plot graph
library(randomForest) # Random Forest

## randomForest 4.6-14
## Type rfNews() to see new features/changes/bug fixes.
##
## Attaching package: 'randomForest'
## The following object is masked from 'package:ggplot2':
##
##   margin
## The following object is masked from 'package:dplyr':
##
##   combine

library(rpart)
library(rpart.plot)
library(caret)

## Loading required package: lattice

library(lubridate)

##
## Attaching package: 'lubridate'
## The following object is masked from 'package:base':
##
##   date

library(gbm)

## Loaded gbm 2.1.5

library(boot)

##
## Attaching package: 'boot'
```

```
## The following object is masked from 'package:lattice':
##
##      melanoma
```

```
OSR2 <- function(predictions, test, train) {
  SSE <- sum((test - predictions)^2)
  SST <- sum((test - mean(train))^2)
  r2 <- 1 - SSE/SST
  return(r2)
}
```

```
us <- read.csv("us_suicides_merged_no_na.csv")
```

```
suicide_us <- us %>% select(year, sex, suicides_no, population, suicides.100k.pop, HDI.for.year, gdp_f
```

```
suicide_us$year <- as.factor(suicide_us$year)
```

```
# split data for us
set.seed(377)
train.ids_us = sample(nrow(suicide_us), 0.70*nrow(suicide_us))
train_us <- suicide_us[train.ids_us,]
test_us <- suicide_us[-train.ids_us,]
```

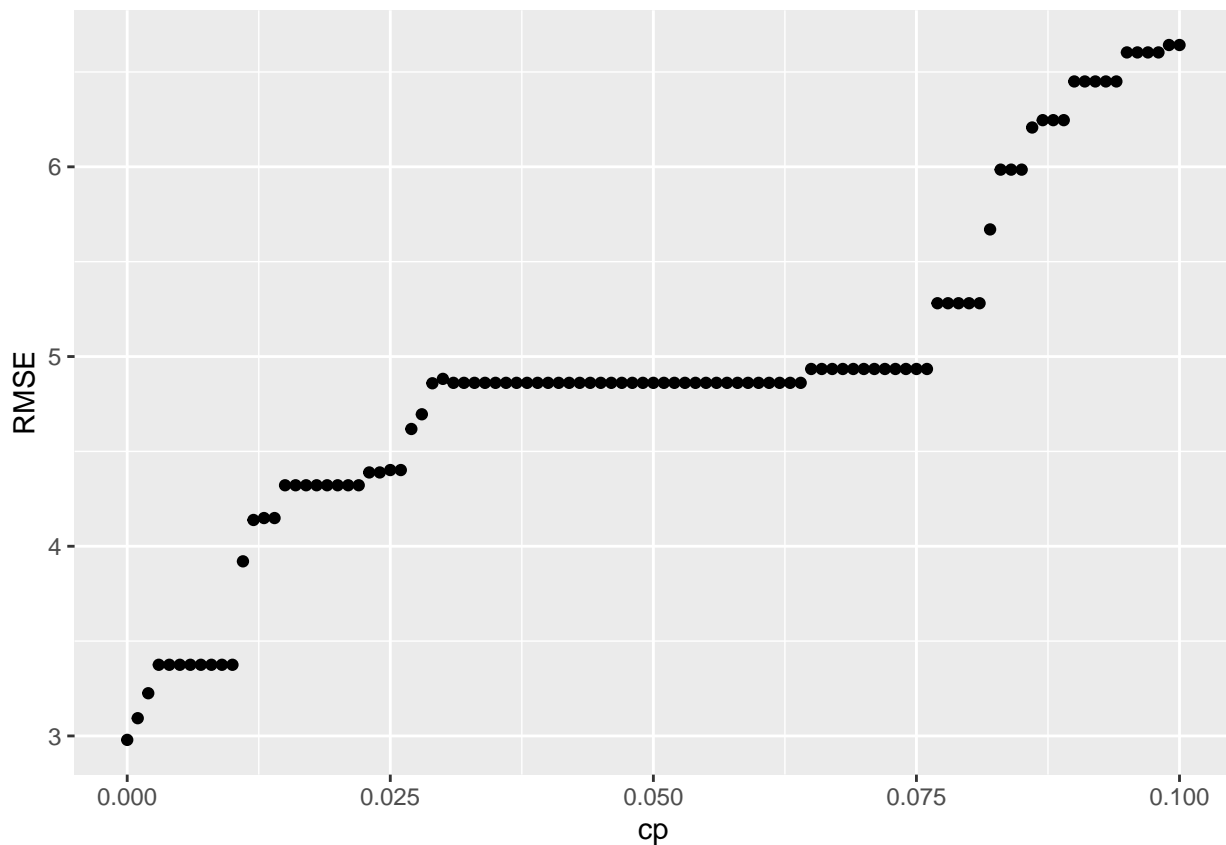
## CART

```
set.seed(377)
```

```
us_train.cart = train(suicides.100k.pop ~ .,
  data = train_us,
  method = "rpart",
  tuneGrid = data.frame(cp=seq(0, 0.1, 0.001)),
  trControl = trainControl(method="cv", number=10),
  metric = "RMSE")
us_train.cart$bestTune
```

```
##      cp
## 1  0
```

```
ggplot(us_train.cart$results, aes(x=cp, y=RMSE)) + geom_point()
```



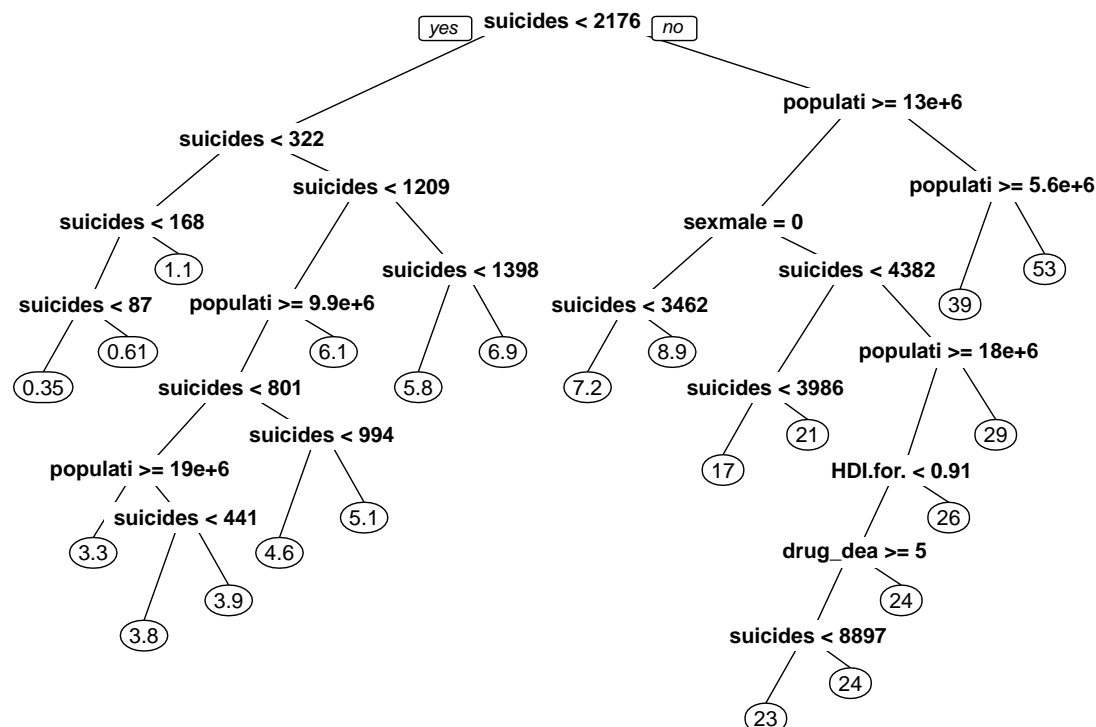
```
us_train.cart$results
```

##	cp	RMSE	Rsquared	MAE	RMSESD	RsquaredSD	MAESD
## 1	0.000	2.979329	0.9299221	1.199689	3.225845	0.1279826	0.7136592
## 2	0.001	3.093575	0.9277967	1.404896	3.194623	0.1285093	0.6913812
## 3	0.002	3.225177	0.9234075	1.554537	3.217963	0.1315820	0.7476110
## 4	0.003	3.375140	0.9194188	1.738446	3.157827	0.1314394	0.7601279
## 5	0.004	3.375140	0.9194188	1.738446	3.157827	0.1314394	0.7601279
## 6	0.005	3.375140	0.9194188	1.738446	3.157827	0.1314394	0.7601279
## 7	0.006	3.375140	0.9194188	1.738446	3.157827	0.1314394	0.7601279
## 8	0.007	3.375140	0.9194188	1.738446	3.157827	0.1314394	0.7601279
## 9	0.008	3.375140	0.9194188	1.738446	3.157827	0.1314394	0.7601279
## 10	0.009	3.375140	0.9194188	1.738446	3.157827	0.1314394	0.7601279
## 11	0.010	3.375140	0.9194188	1.738446	3.157827	0.1314394	0.7601279
## 12	0.011	3.920498	0.9041719	2.305750	2.903154	0.1267028	0.6481499
## 13	0.012	4.138473	0.8952126	2.487929	2.919139	0.1304562	0.7615112
## 14	0.013	4.148571	0.8954976	2.485649	2.928927	0.1289138	0.7723281
## 15	0.014	4.148571	0.8954976	2.485649	2.928927	0.1289138	0.7723281
## 16	0.015	4.321436	0.8799357	2.518585	3.284802	0.1667361	0.8338564
## 17	0.016	4.321436	0.8799357	2.518585	3.284802	0.1667361	0.8338564
## 18	0.017	4.321436	0.8799357	2.518585	3.284802	0.1667361	0.8338564
## 19	0.018	4.321436	0.8799357	2.518585	3.284802	0.1667361	0.8338564
## 20	0.019	4.321436	0.8799357	2.518585	3.284802	0.1667361	0.8338564
## 21	0.020	4.321436	0.8799357	2.518585	3.284802	0.1667361	0.8338564
## 22	0.021	4.321436	0.8799357	2.518585	3.284802	0.1667361	0.8338564
## 23	0.022	4.321436	0.8799357	2.518585	3.284802	0.1667361	0.8338564
## 24	0.023	4.388922	0.8755937	2.607405	3.422550	0.1734971	1.0382248

## 25	0.024	4.388922	0.8755937	2.607405	3.422550	0.1734971	1.0382248
## 26	0.025	4.401650	0.8750325	2.635935	3.449401	0.1748987	1.0841685
## 27	0.026	4.401650	0.8750325	2.635935	3.449401	0.1748987	1.0841685
## 28	0.027	4.618113	0.8704746	2.761382	3.361284	0.1724605	1.0469658
## 29	0.028	4.695637	0.8680869	2.780571	3.338454	0.1714471	1.0421989
## 30	0.029	4.859037	0.8660092	2.899967	3.245508	0.1704052	0.9986216
## 31	0.030	4.882301	0.8666749	2.910465	3.225638	0.1708483	0.9861274
## 32	0.031	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 33	0.032	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 34	0.033	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 35	0.034	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 36	0.035	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 37	0.036	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 38	0.037	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 39	0.038	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 40	0.039	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 41	0.040	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 42	0.041	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 43	0.042	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 44	0.043	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 45	0.044	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 46	0.045	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 47	0.046	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 48	0.047	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 49	0.048	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 50	0.049	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 51	0.050	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 52	0.051	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 53	0.052	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 54	0.053	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 55	0.054	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 56	0.055	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 57	0.056	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 58	0.057	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 59	0.058	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 60	0.059	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 61	0.060	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 62	0.061	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 63	0.062	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 64	0.063	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 65	0.064	4.861378	0.8679781	2.899746	3.240825	0.1715778	0.9970946
## 66	0.065	4.934361	0.8606146	3.059289	3.398389	0.1905633	1.3213671
## 67	0.066	4.934361	0.8606146	3.059289	3.398389	0.1905633	1.3213671
## 68	0.067	4.934361	0.8606146	3.059289	3.398389	0.1905633	1.3213671
## 69	0.068	4.934361	0.8606146	3.059289	3.398389	0.1905633	1.3213671
## 70	0.069	4.934361	0.8606146	3.059289	3.398389	0.1905633	1.3213671
## 71	0.070	4.934361	0.8606146	3.059289	3.398389	0.1905633	1.3213671
## 72	0.071	4.934361	0.8606146	3.059289	3.398389	0.1905633	1.3213671
## 73	0.072	4.934361	0.8606146	3.059289	3.398389	0.1905633	1.3213671
## 74	0.073	4.934361	0.8606146	3.059289	3.398389	0.1905633	1.3213671
## 75	0.074	4.934361	0.8606146	3.059289	3.398389	0.1905633	1.3213671
## 76	0.075	4.934361	0.8606146	3.059289	3.398389	0.1905633	1.3213671
## 77	0.076	4.934361	0.8606146	3.059289	3.398389	0.1905633	1.3213671
## 78	0.077	5.280960	0.8462010	3.362222	3.443466	0.1883127	1.5240413

```
## 79 0.078 5.280960 0.8462010 3.362222 3.443466 0.1883127 1.5240413
## 80 0.079 5.280960 0.8462010 3.362222 3.443466 0.1883127 1.5240413
## 81 0.080 5.280960 0.8462010 3.362222 3.443466 0.1883127 1.5240413
## 82 0.081 5.280960 0.8462010 3.362222 3.443466 0.1883127 1.5240413
## 83 0.082 5.670020 0.8203833 3.656730 3.311591 0.1868571 1.4931868
## 84 0.083 5.985219 0.8024132 3.897906 3.138200 0.1804592 1.3748995
## 85 0.084 5.985219 0.8024132 3.897906 3.138200 0.1804592 1.3748995
## 86 0.085 5.985219 0.8024132 3.897906 3.138200 0.1804592 1.3748995
## 87 0.086 6.207410 0.7891493 4.068535 2.992776 0.1745661 1.2807708
## 88 0.087 6.245957 0.7861506 4.156334 3.060970 0.1780620 1.3742501
## 89 0.088 6.245957 0.7861506 4.156334 3.060970 0.1780620 1.3742501
## 90 0.089 6.245957 0.7861506 4.156334 3.060970 0.1780620 1.3742501
## 91 0.090 6.450051 0.7771554 4.303759 2.948063 0.1716057 1.2971819
## 92 0.091 6.450051 0.7771554 4.303759 2.948063 0.1716057 1.2971819
## 93 0.092 6.450051 0.7771554 4.303759 2.948063 0.1716057 1.2971819
## 94 0.093 6.450051 0.7771554 4.303759 2.948063 0.1716057 1.2971819
## 95 0.094 6.450051 0.7771554 4.303759 2.948063 0.1716057 1.2971819
## 96 0.095 6.603046 0.7687192 4.419024 2.789900 0.1629593 1.1658689
## 97 0.096 6.603046 0.7687192 4.419024 2.789900 0.1629593 1.1658689
## 98 0.097 6.603046 0.7687192 4.419024 2.789900 0.1629593 1.1658689
## 99 0.098 6.603046 0.7687192 4.419024 2.789900 0.1629593 1.1658689
## 100 0.099 6.642118 0.7712096 4.477140 2.754341 0.1657459 1.0781541
## 101 0.100 6.642118 0.7712096 4.477140 2.754341 0.1657459 1.0781541
```

```
mod.us_cart <- us_train.cart$finalModel
prp(mod.us_cart)
```



```
us_test.cart = as.data.frame(model.matrix(suicides.100k.pop ~ . + 0, data=test_us))

predcart_us = predict(mod.us_cart, newdata=us_test.cart)
#predcart_us$results
```

```

cart.tab.us <- table(test_us$suicides.100k.pop, predcart_us)
cart.tab.us

```

```

##      predcart_us
##      0.35125 0.608571428571429 1.09722222222222 3.26 3.75555555555556
## 0.28      1      0      0      0
## 0.31      1      0      0      0
## 0.37      1      0      0      0
## 0.39      1      0      0      0
## 0.41      1      0      0      0
## 0.42      1      0      0      0
## 0.43      0      1      0      0
## 0.44      1      0      0      0
## 0.45      0      1      0      0
## 0.74      0      1      0      0
## 0.78      0      1      0      0
## 0.83      0      0      1      0
## 0.88      0      0      1      0
## 0.92      0      0      1      0
## 1.02      0      0      2      0
## 1.14      0      0      2      0
## 1.24      0      0      1      0
## 1.3       0      0      1      0
## 1.33      0      0      1      0
## 3.03      0      0      0      1
## 3.12      0      0      0      0
## 3.32      0      0      0      0
## 3.61      0      0      0      0
## 3.72      0      0      0      0
## 3.77      0      0      0      0
## 3.85      0      0      0      0
## 3.88      0      0      0      0
## 4         0      0      0      0
## 4.08      0      0      0      0
## 4.23      0      0      0      0
## 4.32      0      0      0      0
## 4.36      0      0      0      0
## 4.59      0      0      0      0
## 4.73      0      0      0      0
## 4.77      0      0      0      0
## 4.81      0      0      0      0
## 4.95      0      0      0      0
## 5.06      0      0      0      0
## 5.07      0      0      0      0
## 5.17      0      0      0      0
## 5.38      0      0      0      0
## 5.47      0      0      0      0
## 5.61      0      0      0      0
## 5.77      0      0      0      0
## 6.02      0      0      0      0
## 6.17      0      0      0      0
## 6.21      0      0      0      0
## 6.29      0      0      0      0
## 6.4       0      0      0      0

```

##	6.76	0	0	0	0	0
##	6.8	0	0	0	0	0
##	6.91	0	0	0	0	0
##	7.06	0	0	0	0	0
##	7.11	0	0	0	0	0
##	7.3	0	0	0	0	0
##	7.38	0	0	0	0	0
##	7.48	0	0	0	0	0
##	7.83	0	0	0	0	0
##	7.9	0	0	0	0	0
##	8.99	0	0	0	0	0
##	16.09	0	0	0	0	0
##	16.15	0	0	0	0	0
##	16.16	0	0	0	0	0
##	16.64	0	0	0	0	0
##	17.2	0	0	0	0	0
##	17.5	0	0	0	0	0
##	18.01	0	0	0	0	0
##	19.57	0	0	0	0	0
##	20	0	0	0	0	0
##	20.98	0	0	0	0	0
##	21.26	0	0	0	0	0
##	21.89	0	0	0	0	0
##	21.92	0	0	0	0	0
##	22.24	0	0	0	0	0
##	22.25	0	0	0	0	0
##	22.41	0	0	0	0	0
##	22.61	0	0	0	0	0
##	22.62	0	0	0	0	0
##	23.3	0	0	0	0	0
##	23.45	0	0	0	0	0
##	23.57	0	0	0	0	0
##	23.88	0	0	0	0	0
##	24	0	0	0	0	0
##	24.01	0	0	0	0	0
##	24.03	0	0	0	0	0
##	24.12	0	0	0	0	0
##	24.62	0	0	0	0	0
##	24.76	0	0	0	0	0
##	24.78	0	0	0	0	0
##	24.86	0	0	0	0	0
##	25.02	0	0	0	0	0
##	25.06	0	0	0	0	0
##	25.48	0	0	0	0	0
##	25.52	0	0	0	0	0
##	25.61	0	0	0	0	0
##	25.62	0	0	0	0	0
##	26.34	0	0	0	0	0
##	26.41	0	0	0	0	0
##	26.52	0	0	0	0	0
##	26.71	0	0	0	0	0
##	27.05	0	0	0	0	0
##	27.93	0	0	0	0	0
##	28.11	0	0	0	0	0

##	36.53	0	0	0	0	0
##	37.11	0	0	0	0	0
##	45.15	0	0	0	0	0
##	52.33	0	0	0	0	0
##	57.85	0	0	0	0	0
##	predcart_us					
##	3.93727272727273	4.59416666666667	5.132	5.78153846153846	6.07375	
##	0.28	0	0	0	0	0
##	0.31	0	0	0	0	0
##	0.37	0	0	0	0	0
##	0.39	0	0	0	0	0
##	0.41	0	0	0	0	0
##	0.42	0	0	0	0	0
##	0.43	0	0	0	0	0
##	0.44	0	0	0	0	0
##	0.45	0	0	0	0	0
##	0.74	0	0	0	0	0
##	0.78	0	0	0	0	0
##	0.83	0	0	0	0	0
##	0.88	0	0	0	0	0
##	0.92	0	0	0	0	0
##	1.02	0	0	0	0	0
##	1.14	0	0	0	0	0
##	1.24	0	0	0	0	0
##	1.3	0	0	0	0	0
##	1.33	0	0	0	0	0
##	3.03	0	0	0	0	0
##	3.12	1	0	0	0	0
##	3.32	1	0	0	0	0
##	3.61	0	0	0	0	0
##	3.72	1	0	0	0	0
##	3.77	0	0	0	0	0
##	3.85	0	0	0	0	0
##	3.88	0	1	0	0	0
##	4	0	1	0	0	0
##	4.08	1	0	0	0	0
##	4.23	1	0	0	0	0
##	4.32	0	1	0	0	0
##	4.36	0	1	0	0	0
##	4.59	1	0	0	0	0
##	4.73	0	1	0	0	0
##	4.77	0	1	1	0	0
##	4.81	0	0	0	0	1
##	4.95	0	0	1	0	0
##	5.06	0	0	1	0	0
##	5.07	0	0	0	0	1
##	5.17	1	0	0	0	0
##	5.38	0	0	2	0	0
##	5.47	0	0	0	0	1
##	5.61	0	0	0	1	0
##	5.77	0	0	0	0	0
##	6.02	0	0	0	0	0
##	6.17	0	0	0	0	1
##	6.21	0	0	0	1	0



##	6.29	0	0	0	1	0
##	6.4	0	0	0	0	1
##	6.76	0	0	0	0	0
##	6.8	0	0	0	0	0
##	6.91	0	0	0	0	0
##	7.06	0	0	0	0	0
##	7.11	0	0	0	0	0
##	7.3	0	0	0	0	0
##	7.38	0	0	0	0	0
##	7.48	0	0	0	0	0
##	7.83	0	0	0	0	0
##	7.9	0	0	0	0	0
##	8.99	0	0	0	0	0
##	16.09	0	0	0	0	0
##	16.15	0	0	0	0	0
##	16.16	0	0	0	0	0
##	16.64	0	0	0	0	0
##	17.2	0	0	0	0	0
##	17.5	0	0	0	0	0
##	18.01	0	0	0	0	0
##	19.57	0	0	0	0	0
##	20	0	0	0	0	0
##	20.98	0	0	0	0	0
##	21.26	0	0	0	0	0
##	21.89	0	0	0	0	0
##	21.92	0	0	0	0	0
##	22.24	0	0	0	0	0
##	22.25	0	0	0	0	0
##	22.41	0	0	0	0	0
##	22.61	0	0	0	0	0
##	22.62	0	0	0	0	0
##	23.3	0	0	0	0	0
##	23.45	0	0	0	0	0
##	23.57	0	0	0	0	0
##	23.88	0	0	0	0	0
##	24	0	0	0	0	0
##	24.01	0	0	0	0	0
##	24.03	0	0	0	0	0
##	24.12	0	0	0	0	0
##	24.62	0	0	0	0	0
##	24.76	0	0	0	0	0
##	24.78	0	0	0	0	0
##	24.86	0	0	0	0	0
##	25.02	0	0	0	0	0
##	25.06	0	0	0	0	0
##	25.48	0	0	0	0	0
##	25.52	0	0	0	0	0
##	25.61	0	0	0	0	0
##	25.62	0	0	0	0	0
##	26.34	0	0	0	0	0
##	26.41	0	0	0	0	0
##	26.52	0	0	0	0	0
##	26.71	0	0	0	0	0
##	27.05	0	0	0	0	0

##	27.93	0		0	0		0	0
##	28.11	0		0	0		0	0
##	36.53	0		0	0		0	0
##	37.11	0		0	0		0	0
##	45.15	0		0	0		0	0
##	52.33	0		0	0		0	0
##	57.85	0		0	0		0	0
##	predcart_us							
##	6.88636363636364	7.1625	8.865	16.9827272727273	21.1638888888889			
##	0.28	0	0	0	0			0
##	0.31	0	0	0	0			0
##	0.37	0	0	0	0			0
##	0.39	0	0	0	0			0
##	0.41	0	0	0	0			0
##	0.42	0	0	0	0			0
##	0.43	0	0	0	0			0
##	0.44	0	0	0	0			0
##	0.45	0	0	0	0			0
##	0.74	0	0	0	0			0
##	0.78	0	0	0	0			0
##	0.83	0	0	0	0			0
##	0.88	0	0	0	0			0
##	0.92	0	0	0	0			0
##	1.02	0	0	0	0			0
##	1.14	0	0	0	0			0
##	1.24	0	0	0	0			0
##	1.3	0	0	0	0			0
##	1.33	0	0	0	0			0
##	3.03	0	0	0	0			0
##	3.12	0	0	0	0			0
##	3.32	0	0	0	0			0
##	3.61	0	0	0	0			0
##	3.72	0	0	0	0			0
##	3.77	0	0	0	0			0
##	3.85	0	0	0	0			0
##	3.88	0	0	0	0			0
##	4	0	0	0	0			0
##	4.08	0	0	0	0			0
##	4.23	0	0	0	0			0
##	4.32	0	0	0	0			0
##	4.36	0	0	0	0			0
##	4.59	0	0	0	0			0
##	4.73	0	0	0	0			0
##	4.77	0	0	0	0			0
##	4.81	0	0	0	0			0
##	4.95	0	0	0	0			0
##	5.06	0	0	0	0			0
##	5.07	0	0	0	0			0
##	5.17	0	0	0	0			0
##	5.38	0	0	0	0			0
##	5.47	0	0	0	0			0
##	5.61	0	0	0	0			0
##	5.77	1	0	0	0			0
##	6.02	1	0	0	0			0

##	6.17	0	0	0	0	0
##	6.21	0	0	0	0	0
##	6.29	0	0	0	0	0
##	6.4	0	0	0	0	0
##	6.76	0	1	0	0	0
##	6.8	0	1	0	0	0
##	6.91	1	0	0	0	0
##	7.06	1	0	0	0	0
##	7.11	0	1	0	0	0
##	7.3	0	1	0	0	0
##	7.38	0	1	0	0	0
##	7.48	0	1	0	0	0
##	7.83	0	1	0	0	0
##	7.9	1	0	0	0	0
##	8.99	0	0	1	0	0
##	16.09	0	0	0	1	0
##	16.15	0	0	0	1	0
##	16.16	0	0	0	1	0
##	16.64	0	0	0	1	0
##	17.2	0	0	0	1	0
##	17.5	0	0	0	1	0
##	18.01	0	0	0	0	1
##	19.57	0	0	0	1	0
##	20	0	0	0	1	0
##	20.98	0	0	0	0	1
##	21.26	0	0	0	0	1
##	21.89	0	0	0	0	1
##	21.92	0	0	0	0	1
##	22.24	0	0	0	0	1
##	22.25	0	0	0	0	0
##	22.41	0	0	0	0	0
##	22.61	0	0	0	0	0
##	22.62	0	0	0	0	0
##	23.3	0	0	0	0	0
##	23.45	0	0	0	0	1
##	23.57	0	0	0	0	0
##	23.88	0	0	0	0	0
##	24	0	0	0	0	0
##	24.01	0	0	0	0	0
##	24.03	0	0	0	0	0
##	24.12	0	0	0	0	0
##	24.62	0	0	0	0	0
##	24.76	0	0	0	0	0
##	24.78	0	0	0	0	0
##	24.86	0	0	0	0	0
##	25.02	0	0	0	0	0
##	25.06	0	0	0	0	0
##	25.48	0	0	0	0	0
##	25.52	0	0	0	0	0
##	25.61	0	0	0	0	0
##	25.62	0	0	0	0	0
##	26.34	0	0	0	0	0
##	26.41	0	0	0	0	0
##	26.52	0	0	0	0	0

##	26.71	0	0	0	0	0
##	27.05	0	0	0	0	0
##	27.93	0	0	0	0	0
##	28.11	0	0	0	0	0
##	36.53	0	0	0	0	0
##	37.11	0	0	0	0	0
##	45.15	0	0	0	0	0
##	52.33	0	0	0	0	0
##	57.85	0	0	0	0	0
##	predcart_us					
##	23.0911764705882 24.0785714285714 24.47 26.1814285714286					
##	0.28	0		0	0	0
##	0.31	0		0	0	0
##	0.37	0		0	0	0
##	0.39	0		0	0	0
##	0.41	0		0	0	0
##	0.42	0		0	0	0
##	0.43	0		0	0	0
##	0.44	0		0	0	0
##	0.45	0		0	0	0
##	0.74	0		0	0	0
##	0.78	0		0	0	0
##	0.83	0		0	0	0
##	0.88	0		0	0	0
##	0.92	0		0	0	0
##	1.02	0		0	0	0
##	1.14	0		0	0	0
##	1.24	0		0	0	0
##	1.3	0		0	0	0
##	1.33	0		0	0	0
##	3.03	0		0	0	0
##	3.12	0		0	0	0
##	3.32	0		0	0	0
##	3.61	0		0	0	0
##	3.72	0		0	0	0
##	3.77	0		0	0	0
##	3.85	0		0	0	0
##	3.88	0		0	0	0
##	4	0		0	0	0
##	4.08	0		0	0	0
##	4.23	0		0	0	0
##	4.32	0		0	0	0
##	4.36	0		0	0	0
##	4.59	0		0	0	0
##	4.73	0		0	0	0
##	4.77	0		0	0	0
##	4.81	0		0	0	0
##	4.95	0		0	0	0
##	5.06	0		0	0	0
##	5.07	0		0	0	0
##	5.17	0		0	0	0
##	5.38	0		0	0	0
##	5.47	0		0	0	0
##	5.61	0		0	0	0

##	5.77	0	0	0	0
##	6.02	0	0	0	0
##	6.17	0	0	0	0
##	6.21	0	0	0	0
##	6.29	0	0	0	0
##	6.4	0	0	0	0
##	6.76	0	0	0	0
##	6.8	0	0	0	0
##	6.91	0	0	0	0
##	7.06	0	0	0	0
##	7.11	0	0	0	0
##	7.3	0	0	0	0
##	7.38	0	0	0	0
##	7.48	0	0	0	0
##	7.83	0	0	0	0
##	7.9	0	0	0	0
##	8.99	0	0	0	0
##	16.09	0	0	0	0
##	16.15	0	0	0	0
##	16.16	0	0	0	0
##	16.64	0	0	0	0
##	17.2	0	0	0	0
##	17.5	0	0	0	0
##	18.01	0	0	0	0
##	19.57	0	0	0	0
##	20	0	0	0	0
##	20.98	0	0	0	0
##	21.26	0	0	0	0
##	21.89	0	0	0	0
##	21.92	0	0	0	0
##	22.24	0	0	0	0
##	22.25	1	0	0	0
##	22.41	1	0	0	0
##	22.61	1	0	0	0
##	22.62	0	1	0	0
##	23.3	1	0	0	0
##	23.45	0	0	0	0
##	23.57	0	1	0	0
##	23.88	0	1	0	0
##	24	0	0	0	1
##	24.01	1	0	0	0
##	24.03	0	0	1	0
##	24.12	0	1	0	0
##	24.62	0	0	0	0
##	24.76	0	0	1	0
##	24.78	0	0	1	0
##	24.86	0	0	0	1
##	25.02	0	0	1	0
##	25.06	0	0	0	0
##	25.48	0	0	1	0
##	25.52	0	0	0	1
##	25.61	0	0	1	0
##	25.62	0	0	1	0
##	26.34	0	0	0	1

##	26.41	0	0	0	0
##	26.52	0	0	0	1
##	26.71	0	0	0	0
##	27.05	0	0	0	1
##	27.93	0	0	0	1
##	28.11	0	0	0	1
##	36.53	0	0	0	0
##	37.11	0	0	0	0
##	45.15	0	0	0	0
##	52.33	0	0	0	0
##	57.85	0	0	0	0
##	predcart_us				
##	29.0328571428571	38.67125	53.46		
##	0.28	0	0	0	
##	0.31	0	0	0	
##	0.37	0	0	0	
##	0.39	0	0	0	
##	0.41	0	0	0	
##	0.42	0	0	0	
##	0.43	0	0	0	
##	0.44	0	0	0	
##	0.45	0	0	0	
##	0.74	0	0	0	
##	0.78	0	0	0	
##	0.83	0	0	0	
##	0.88	0	0	0	
##	0.92	0	0	0	
##	1.02	0	0	0	
##	1.14	0	0	0	
##	1.24	0	0	0	
##	1.3	0	0	0	
##	1.33	0	0	0	
##	3.03	0	0	0	
##	3.12	0	0	0	
##	3.32	0	0	0	
##	3.61	0	0	0	
##	3.72	0	0	0	
##	3.77	0	0	0	
##	3.85	0	0	0	
##	3.88	0	0	0	
##	4	0	0	0	
##	4.08	0	0	0	
##	4.23	0	0	0	
##	4.32	0	0	0	
##	4.36	0	0	0	
##	4.59	0	0	0	
##	4.73	0	0	0	
##	4.77	0	0	0	
##	4.81	0	0	0	
##	4.95	0	0	0	
##	5.06	0	0	0	
##	5.07	0	0	0	
##	5.17	0	0	0	
##	5.38	0	0	0	

##	5.47	0	0	0
##	5.61	0	0	0
##	5.77	0	0	0
##	6.02	0	0	0
##	6.17	0	0	0
##	6.21	0	0	0
##	6.29	0	0	0
##	6.4	0	0	0
##	6.76	0	0	0
##	6.8	0	0	0
##	6.91	0	0	0
##	7.06	0	0	0
##	7.11	0	0	0
##	7.3	0	0	0
##	7.38	0	0	0
##	7.48	0	0	0
##	7.83	0	0	0
##	7.9	0	0	0
##	8.99	0	0	0
##	16.09	0	0	0
##	16.15	0	0	0
##	16.16	0	0	0
##	16.64	0	0	0
##	17.2	0	0	0
##	17.5	0	0	0
##	18.01	0	0	0
##	19.57	0	0	0
##	20	0	0	0
##	20.98	0	0	0
##	21.26	0	0	0
##	21.89	0	0	0
##	21.92	0	0	0
##	22.24	0	0	0
##	22.25	0	0	0
##	22.41	0	0	0
##	22.61	0	0	0
##	22.62	0	0	0
##	23.3	0	0	0
##	23.45	0	0	0
##	23.57	0	0	0
##	23.88	0	0	0
##	24	0	0	0
##	24.01	0	0	0
##	24.03	0	0	0
##	24.12	0	0	0
##	24.62	1	0	0
##	24.76	0	0	0
##	24.78	0	0	0
##	24.86	0	0	0
##	25.02	0	0	0
##	25.06	1	0	0
##	25.48	0	0	0
##	25.52	0	0	0
##	25.61	0	0	0

```
##    25.62          0          0          0
##    26.34          0          0          0
##    26.41          1          0          0
##    26.52          0          0          0
##    26.71          1          0          0
##    27.05          0          0          0
##    27.93          0          0          0
##    28.11          0          0          0
##    36.53          0          1          0
##    37.11          0          1          0
##    45.15          0          1          0
##    52.33          0          0          1
##    57.85          0          0          1
```

```
print("CART OSR2:")
```

```
## [1] "CART OSR2:"
```

```
OSR2(predcart_us, test_us$suicides.100k.pop, train_us$suicides.100k.pop)
```

```
## [1] 0.9883818
```

## Random Forest

```
set.seed(377)
```

```
mod.rf.us <- randomForest(suicides.100k.pop ~ ., data = train_us, mtry = 5, nodesize = 5, ntree = 500)
```

```
pred.rf.us <- predict(mod.rf.us, newdata = test_us) # just to illustrate
pred.rf.us[1:5]
```

```
##          1          7          16          21          26
## 4.859536 1.050132 24.295192  7.247386 20.566757
```

```
importance(mod.rf.us)
```

```
##              IncNodePurity
## year              721.5390
## sex              6873.8985
## suicides_no      20017.7699
## population       12227.8713
## HDI.for.year      248.8317
## gdp_for_year....   268.8887
## gdp_per_capita.... 222.9683
## generation        2319.3296
## depression_percentage 4389.2658
## drug_death_rate    1101.7067
```

```
set.seed(377)
```

```
train.rf.us <- train(suicides.100k.pop ~ .,
                     data = train_us,
                     method = "rf",
                     tuneGrid = data.frame(mtry=1:5),
                     trControl = trainControl(method="cv", number=5, verboseIter = TRUE),
                     metric = "RMSE")
```



```

## + Fold1: mtry=1
## - Fold1: mtry=1
## + Fold1: mtry=2
## - Fold1: mtry=2
## + Fold1: mtry=3
## - Fold1: mtry=3
## + Fold1: mtry=4
## - Fold1: mtry=4
## + Fold1: mtry=5
## - Fold1: mtry=5
## + Fold2: mtry=1
## - Fold2: mtry=1
## + Fold2: mtry=2
## - Fold2: mtry=2
## + Fold2: mtry=3
## - Fold2: mtry=3
## + Fold2: mtry=4
## - Fold2: mtry=4
## + Fold2: mtry=5
## - Fold2: mtry=5
## + Fold3: mtry=1
## - Fold3: mtry=1
## + Fold3: mtry=2
## - Fold3: mtry=2
## + Fold3: mtry=3
## - Fold3: mtry=3
## + Fold3: mtry=4
## - Fold3: mtry=4
## + Fold3: mtry=5
## - Fold3: mtry=5
## + Fold4: mtry=1
## - Fold4: mtry=1
## + Fold4: mtry=2
## - Fold4: mtry=2
## + Fold4: mtry=3
## - Fold4: mtry=3
## + Fold4: mtry=4
## - Fold4: mtry=4
## + Fold4: mtry=5
## - Fold4: mtry=5
## + Fold5: mtry=1
## - Fold5: mtry=1
## + Fold5: mtry=2
## - Fold5: mtry=2
## + Fold5: mtry=3
## - Fold5: mtry=3
## + Fold5: mtry=4
## - Fold5: mtry=4
## + Fold5: mtry=5
## - Fold5: mtry=5
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 5 on full training set

```

```
train.rf.us$results
```

```
##      mtry      RMSE Rsquared      MAE      RMSESD RsquaredSD      MAESD
## 1      1 9.824508 0.8696212 7.994422 0.9381334 0.040652349 0.5209140
## 2      2 6.502285 0.9248426 4.913252 0.8379313 0.007879139 0.4761299
## 3      3 4.769440 0.9495409 3.396341 0.7533864 0.010192692 0.4333536
## 4      4 3.711369 0.9646292 2.495179 0.6784323 0.007789490 0.3832441
## 5      5 3.069317 0.9735388 2.018823 0.5291298 0.005027661 0.3227183
```

```
best.rf.us <- train.rf.us$finalModel
```

```
us.test_rf = as.data.frame(model.matrix(suicides.100k.pop ~ . + 0, data = test_us))
```

```
pred.best.rf_us <- predict(best.rf.us, newdata = us.test_rf)
pred.best.rf_us[1:5]
```

```
##           1           7           16           21           26
## 6.398268  3.707426 23.008092  7.826827 21.054450
```

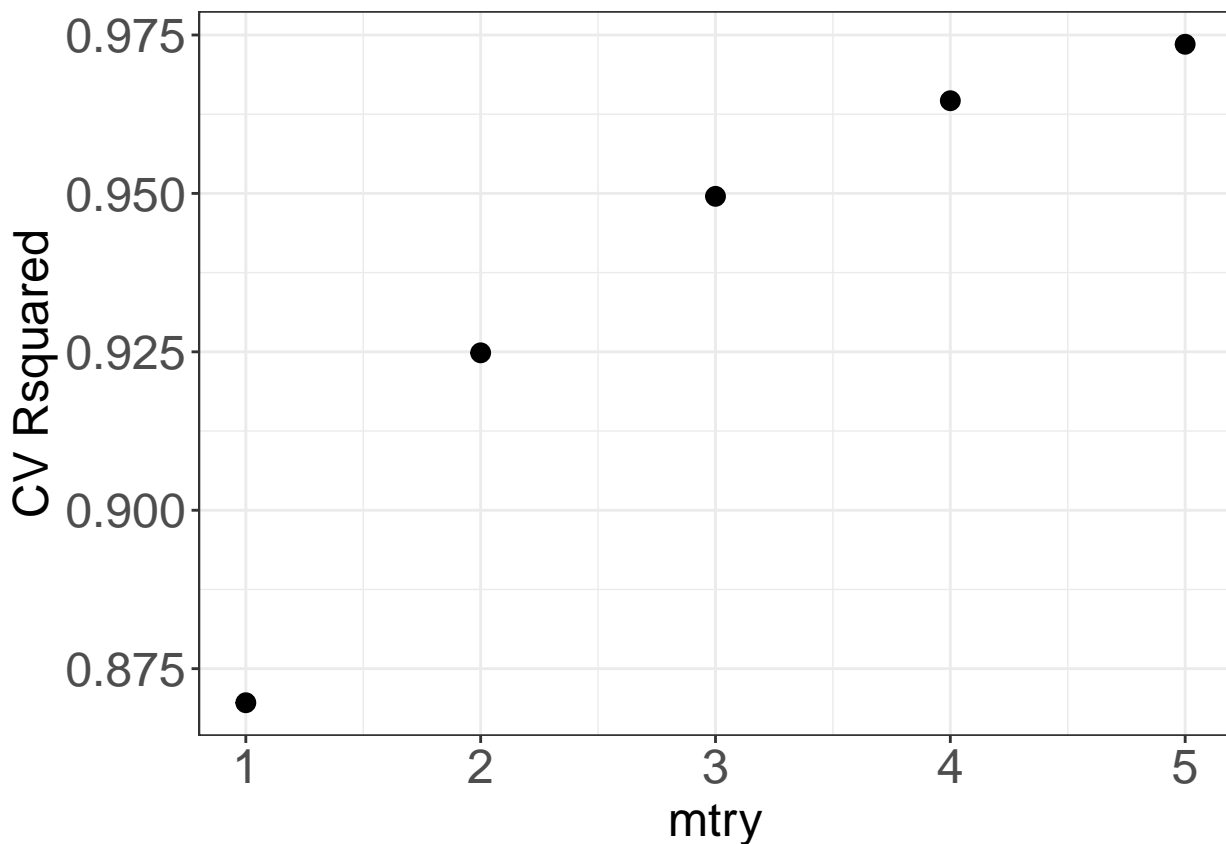
```
print("Random Forests OSR2:")
```

```
## [1] "Random Forests OSR2:"
```

```
OSR2(pred.best.rf_us, test_us$suicides.100k.pop, train_us$suicides.100k.pop)
```

```
## [1] 0.9645385
```

```
ggplot(train.rf.us$results, aes(x = mtry, y = Rsquared)) + geom_point(size = 3) +
  ylab("CV Rsquared") + theme_bw() + theme(axis.title=element_text(size=18), axis.text=element_text(size=12))
```



mtry = 10

```
set.seed(377)
train.rf.us_mtryTen <- train(suicides.100k.pop ~ .,
                             data = train_us,
                             method = "rf",
                             tuneGrid = data.frame(mtry=1:10),
                             trControl = trainControl(method="cv", number=5, verboseIter = TRUE),
                             metric = "RMSE")
```

```
## + Fold1: mtry= 1
## - Fold1: mtry= 1
## + Fold1: mtry= 2
## - Fold1: mtry= 2
## + Fold1: mtry= 3
## - Fold1: mtry= 3
## + Fold1: mtry= 4
## - Fold1: mtry= 4
## + Fold1: mtry= 5
## - Fold1: mtry= 5
## + Fold1: mtry= 6
## - Fold1: mtry= 6
## + Fold1: mtry= 7
## - Fold1: mtry= 7
## + Fold1: mtry= 8
## - Fold1: mtry= 8
## + Fold1: mtry= 9
## - Fold1: mtry= 9
## + Fold1: mtry=10
## - Fold1: mtry=10
## + Fold2: mtry= 1
## - Fold2: mtry= 1
## + Fold2: mtry= 2
## - Fold2: mtry= 2
## + Fold2: mtry= 3
## - Fold2: mtry= 3
## + Fold2: mtry= 4
## - Fold2: mtry= 4
## + Fold2: mtry= 5
## - Fold2: mtry= 5
## + Fold2: mtry= 6
## - Fold2: mtry= 6
## + Fold2: mtry= 7
## - Fold2: mtry= 7
## + Fold2: mtry= 8
## - Fold2: mtry= 8
## + Fold2: mtry= 9
## - Fold2: mtry= 9
## + Fold2: mtry=10
## - Fold2: mtry=10
## + Fold3: mtry= 1
## - Fold3: mtry= 1
## + Fold3: mtry= 2
## - Fold3: mtry= 2
```

```
## + Fold3: mtry= 3
## - Fold3: mtry= 3
## + Fold3: mtry= 4
## - Fold3: mtry= 4
## + Fold3: mtry= 5
## - Fold3: mtry= 5
## + Fold3: mtry= 6
## - Fold3: mtry= 6
## + Fold3: mtry= 7
## - Fold3: mtry= 7
## + Fold3: mtry= 8
## - Fold3: mtry= 8
## + Fold3: mtry= 9
## - Fold3: mtry= 9
## + Fold3: mtry=10
## - Fold3: mtry=10
## + Fold4: mtry= 1
## - Fold4: mtry= 1
## + Fold4: mtry= 2
## - Fold4: mtry= 2
## + Fold4: mtry= 3
## - Fold4: mtry= 3
## + Fold4: mtry= 4
## - Fold4: mtry= 4
## + Fold4: mtry= 5
## - Fold4: mtry= 5
## + Fold4: mtry= 6
## - Fold4: mtry= 6
## + Fold4: mtry= 7
## - Fold4: mtry= 7
## + Fold4: mtry= 8
## - Fold4: mtry= 8
## + Fold4: mtry= 9
## - Fold4: mtry= 9
## + Fold4: mtry=10
## - Fold4: mtry=10
## + Fold5: mtry= 1
## - Fold5: mtry= 1
## + Fold5: mtry= 2
## - Fold5: mtry= 2
## + Fold5: mtry= 3
## - Fold5: mtry= 3
## + Fold5: mtry= 4
## - Fold5: mtry= 4
## + Fold5: mtry= 5
## - Fold5: mtry= 5
## + Fold5: mtry= 6
## - Fold5: mtry= 6
## + Fold5: mtry= 7
## - Fold5: mtry= 7
## + Fold5: mtry= 8
## - Fold5: mtry= 8
## + Fold5: mtry= 9
## - Fold5: mtry= 9
```

```
## + Fold5: mtry=10
## - Fold5: mtry=10
## Aggregating results
## Selecting tuning parameters
## Fitting mtry = 10 on full training set

train.rf.us_mtryTen$results

##      mtry      RMSE Rsquared      MAE      RMSESD RsquaredSD      MAESD
## 1      1 9.882207 0.8543650 8.035307 1.0023155 0.036784552 0.6113408
## 2      2 6.620567 0.9187419 4.985777 0.9989036 0.023054109 0.5563487
## 3      3 4.875914 0.9456037 3.456866 0.7017972 0.007218984 0.4125931
## 4      4 3.752569 0.9659937 2.568780 0.6596917 0.006132700 0.3727510
## 5      5 3.000041 0.9743157 1.975137 0.5458457 0.005464792 0.3490734
## 6      6 2.617097 0.9784615 1.669313 0.5002927 0.004488329 0.3125495
## 7      7 2.349818 0.9807843 1.408615 0.4467046 0.006388239 0.2691800
## 8      8 2.211275 0.9820666 1.316461 0.3986787 0.005339521 0.2288858
## 9      9 1.990433 0.9844981 1.167108 0.4789708 0.007932799 0.2326738
## 10     10 1.889257 0.9860799 1.087407 0.3959739 0.005604698 0.2008353

best.rf.us_mtryTen <- train.rf.us_mtryTen$finalModel

pred.best.rf.us_mtryTen <- predict(best.rf.us_mtryTen, newdata = us.test_rf)
pred.best.rf.us_mtryTen[1:5]

##           1           7           16           21           26
## 5.047776 1.655800 24.156776 7.160834 20.078536

print("Random Forests OSR2:")

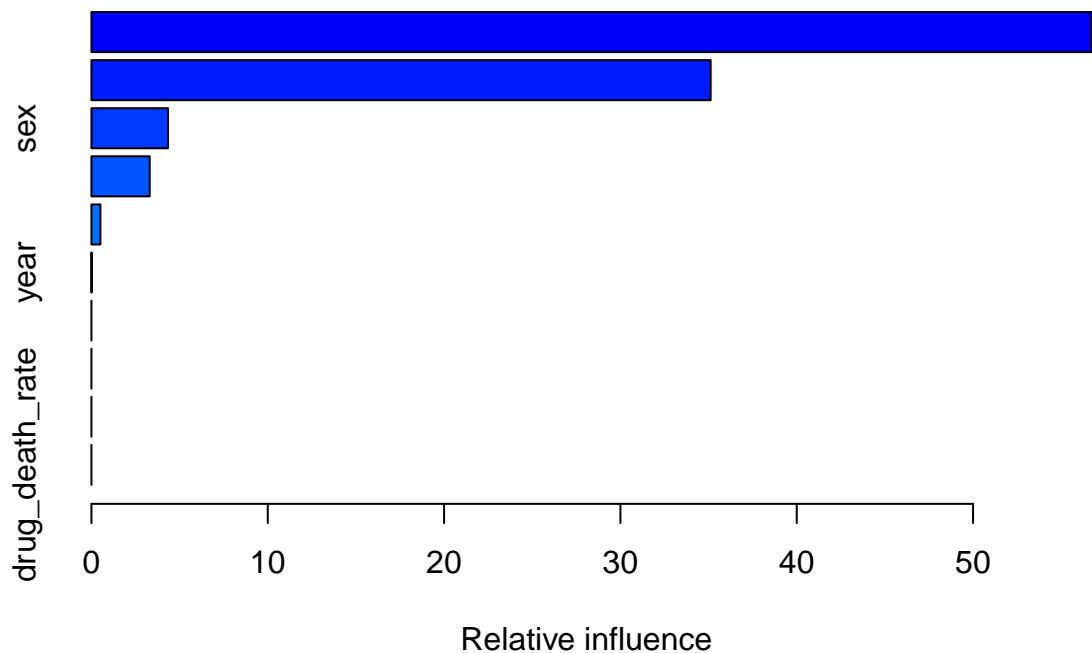
## [1] "Random Forests OSR2:"

OSR2(pred.best.rf.us_mtryTen, test_us$suicides.100k.pop, train_us$suicides.100k.pop)

## [1] 0.9928406
```

## Boosting

```
mod.boost <- gbm(suicides.100k.pop ~ .,
  data = train_us,
  distribution = "gaussian",
  n.trees = 1000,
  shrinkage = 0.001,
  interaction.depth = 2)
summary(mod.boost)
```



```
##               var      rel.inf
## suicides_no    suicides_no 56.71060064
## population     population 35.11841374
## sex            sex        4.34138880
## depression_percentage depression_percentage 3.30468990
## generation     generation 0.51025724
## year           year       0.01464967
## HDI.for.year   HDI.for.year 0.00000000
## gdp_for_year.... gdp_for_year.... 0.00000000
## gdp_per_capita.... gdp_per_capita.... 0.00000000
## drug_death_rate drug_death_rate 0.00000000
```

```
pred.boost <- predict(mod.boost, newdata = test_us, n.trees=1000)
OSR2(pred.boost, test_us$suicides.100k.pop, train_us$suicides.100k.pop)
```

```
## [1] 0.7628486
```

```
## took a while to run -- not super amazing OSR~2
# test_us_mm = as.data.frame(model.matrix(suicides.100k.pop ~ . + 0, data = test_us))
#
# gbmGrid <- expand.grid(interaction.depth = c(1,2,4,6,8,10),
#                         n.trees = (1:75)*500,
#                         shrinkage = 0.001,
#                         n.minobsinnode = 10)
# fitControl <- trainControl(## 10-fold CV
#                             method = "repeatedcv",
#                             number = 5,
#                             ## repeated ten times
#                             repeats = 5)
# set.seed(377)
# gbmFit2 <- train(suicides.100k.pop ~ ., data = train_us,
#                  method = "gbm",
#                  trControl = fitControl,
#                  verbose = FALSE,
```

```

#           tuneGrid = gbmGrid)
#
# gbm.best <- gbmFit2$finalModel
# gbm.pred.best.boost <- predict(gbm.best, newdata = test_us_mm, n.trees = 11500)
# OSR2(gbm.pred.best.boost, test_us$suicides.100k.pop, train_us$suicides.100k.pop)

## same results as above
# tGrid = expand.grid(n.trees = 1000, interaction.depth = 2, shrinkage = 0.001, n.minobsinnode = 10)tGr
#
# set.seed(377)
# train.boost <- train(suicides.100k.pop ~ .,
#                       data = train_us,
#                       method = "gbm",
#                       tuneGrid = tGrid,
#                       trControl = trainControl(method="cv", number=5,
#                                                verboseIter = FALSE),
#                       metric = "RMSE",
#                       distribution = "gaussian",
#                       verbose = FALSE)
# train.boost
# best.boost <- train.boost$finalModel
# pred.best.boost <- predict(best.boost, newdata = test_us_mm, n.trees = 11500) # can use same model ma
#
# ggplot(train.boost$results, aes(x = n.trees, y = Rsquared, colour = as.factor(interaction.depth))) +
#   ylab("CV Rsquared") + theme_bw() + theme(axis.title=element_text(size=18), axis.text=element_text(s
#   scale_color_discrete(name = "interaction.depth")

# OSR2(pred.best.boost, test_us$suicides.100k.pop, train_us$suicides.100k.pop)
# #Out-of-sample MAE:
# sum(abs(test_us$suicides.100k.pop - pred.best.boost))/nrow(test_us_mm)

```