STAT346: Statistical Data Science I

(YOUR NAME AND STUDENT ID)

Instructions

1. This exam covers material from **Introduction to Data Science** (https://rafalab.github.io/dsbook/), Chapter 20–31.

2. You may use any books or online resources you want during this examination, but you may not communicate with any person other than your examiner.

3. You are required to use the RStudio IDE for this exam. You may use either the desktop edition or rstudio.cloud as you prefer.

4. You should work on the provided exam template. When you finalize your exam, you should submit your paper in pdf as well as its .rmd source file. They should have the following name:

   - `stat346_final_yourID.pdf`
   - `stat346_final_yourID.rmd`

5. You should submit your paper no later than 4:50 p.m. If you are late, you will get 20% penalty per 10 minutes.

Problem Set #1 (30 Points)

Load the `admissions` data set, which contains admission information for men and women across six majors and keep only the admitted percentage column:

```
library(dslabs)
data(admissions)
dat <- admissions %>% select(-applicants)
```

(a) [6 points] If we think of an observation as a major, and that each observation has two variables (men admitted percentage and women admitted percentage) then this is not tidy. Use the `spread` function to wrangle into tidy shape: one row for each major.

- (YOUR ANSWER HERE)

```
dat<-dat %>% spread(gender,admitted)
dat
```

```
##   major men women
## 1     A  62    82
## 2     B  63    68
## 3     C  37    34
## 4     D  33    35
## 5     E  28    24
## 6     F   6     7
```

(b) [6 points] Now we want to wrangle the admissions data so that for each major we have 4 observations: `admitted_men`, `admitted_women`, `applicants_men` and `applicants_women`. Use the `gather` function to create a `tmp` data.frame with a column containing the type of observation `admitted` or `applicants`. Call the new columns `key` and `value`.

- (YOUR ANSWER HERE)

```
tmp<-admissions %>% gather(key,value,3:4)
tmp
```

```
##    major gender        key value
## 1      A    men   admitted    62
## 2      B    men   admitted    63
## 3      C    men   admitted    37
## 4      D    men   admitted    33
## 5      E    men   admitted    28
## 6      F    men   admitted     6
## 7      A  women   admitted    82
## 8      B  women   admitted    68
```

```
## 9      C  women   admitted    34
## 10     D  women   admitted    35
## 11     E  women   admitted    24
## 12     F  women   admitted     7
## 13     A    men applicants   825
## 14     B    men applicants   560
## 15     C    men applicants   325
## 16     D    men applicants   417
## 17     E    men applicants   191
## 18     F    men applicants   373
## 19     A  women applicants   108
## 20     B  women applicants    25
## 21     C  women applicants   593
## 22     D  women applicants   375
## 23     E  women applicants   393
## 24     F  women applicants   341
```

(c) [6 points] Now you have an object `tmp` with columns `major`, `gender`, `key` and `value`. Note that if you combine the `key` and `gender`, we get the column names we want: `admitted_men`, `admitted_women`, `applicants_men` and `applicants_women`. Use the function `unite` to create a new column called `column_name`.

- (YOUR ANSWER HERE)

```
tmp<-tmp %>% unite(column_name, key,gender)
tmp
```

```
##    major      column_name value
## 1      A     admitted_men    62
## 2      B     admitted_men    63
## 3      C     admitted_men    37
## 4      D     admitted_men    33
## 5      E     admitted_men    28
## 6      F     admitted_men     6
## 7      A   admitted_women    82
## 8      B   admitted_women    68
## 9      C   admitted_women    34
## 10     D   admitted_women    35
## 11     E   admitted_women    24
## 12     F   admitted_women     7
## 13     A   applicants_men   825
## 14     B   applicants_men   560
## 15     C   applicants_men   325
## 16     D   applicants_men   417
## 17     E   applicants_men   191
## 18     F   applicants_men   373
## 19     A applicants_women   108
```

```
## 20     B applicants_women    25
## 21     C applicants_women   593
## 22     D applicants_women   375
## 23     E applicants_women   393
## 24     F applicants_women   341
```

(d) [6 points] Now use the `spread` function to generate the tidy data with four variables for each major.

- (YOUR ANSWER HERE)

```
tmp  %>% spread(column_name,value)
```

```
##   major admitted_men admitted_women applicants_men applicants_women
## 1     A           62             82            825              108
## 2     B           63             68            560               25
## 3     C           37             34            325              593
## 4     D           33             35            417              375
## 5     E           28             24            191              393
## 6     F            6              7            373              341
```

(e) [6 points] Now use the pipe to write a line of code that turns `admissions` to the table produced in (d).

- (YOUR ANSWER HERE)

```
admissions<-admissions %>% gather(key,value,3:4) %>% unite(col,key,gender) %>% spread(col,
admissions
```

```
##   major admitted_men admitted_women applicants_men applicants_women
## 1     A           62             82            825              108
## 2     B           63             68            560               25
## 3     C           37             34            325              593
## 4     D           33             35            417              375
## 5     E           28             24            191              393
## 6     F            6              7            373              341
```

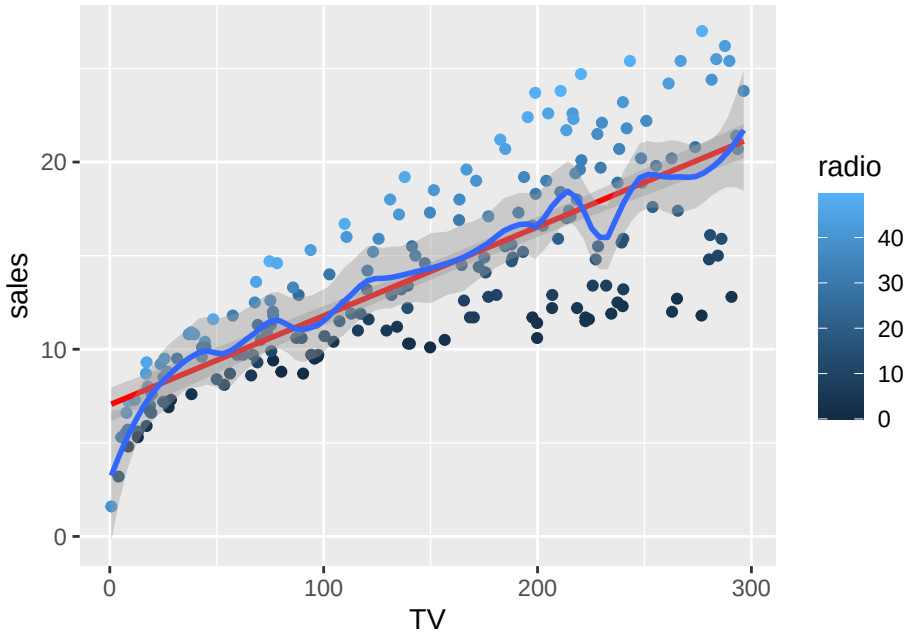
Problem Set #2 (50 Points)

For this problem, we will use a dataset containing information on sales of a product and the amount spent on advertising using different media channels. The data are available from: `http://faculty.marshall.usc.edu/gareth-james/ISL/Advertising.csv`.

(a) [6 points] Read the dataset and generate a scatterplot of sales against the amount of TV advertising. Color the points by the mount of radio advertising. Then, add a linear fit line (in red) and a loess curve (in blue) with 20% span rate. Your plot shall look as follows. Comments on this plot.

- (YOUR ANSWER HERE)

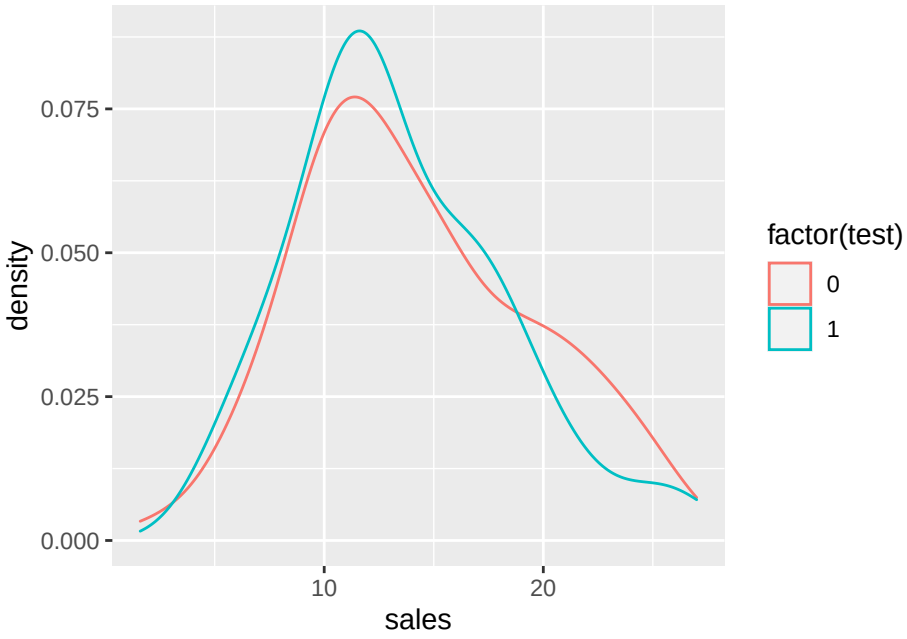
```
dat<-read_csv('http://faculty.marshall.usc.edu/gareth-james/ISL/Advertising.csv')
dat %>% ggplot(aes(TV,sales,col=radio))+geom_point()+
  geom_smooth(method='lm',col='red')+
  geom_smooth(method='loess',span=0.2)
```



(b) [6 points] The dataset has 200 rows. Use the `sample` function to divide it into a train set with 150 observations and a test set with 50 observations. Create a new `test` variable that takes 0 for train set and 1 for test set. Then generate two smoothed density curves of `sales` in a single figure, permitting stratification by `test`. Use `set.seed(123)` to fix randomness.

- (YOUR ANSWER HERE)

```
set.seed(123)
test_index<-sample(200,50)
dat<-dat %>% mutate(test=ifelse(X1 %in% test_index,1,0))
dat %>% ggplot(aes(x=sales,col=factor(test)))+geom_density()
```



(c) [6 points] Fit a linear model to the training set, where the sales values are predicted by the amount of TV advertising. Print the summary of the fitted model. Then, predict the sales values for the test set and evaluate the test model accuracy in terms of root mean squared error (RMSE), which measures the average level of error between the prediction and the true response:

$$\text{RMSE} = \sqrt{\frac{1}{n}\sum_{i=1}^{n}(\hat{y}_i - y_i)^2}$$

- (YOUR ANSWER HERE)

```
train_index<-setdiff(1:200,test_index)
fitted<-lm(sales~TV,data=dat,subset=train_index)
fitted %>% summary()
```

```
##
## Call:
## lm(formula = sales ~ TV, data = dat, subset = train_index)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.5639 -1.6254 -0.1633  1.9727  6.8986
```

```
##
## Coefficients:
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept) 6.915977   0.537327   12.87   <2e-16 ***
## TV          0.049701   0.003167   15.69   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 3.287 on 148 degrees of freedom
## Multiple R-squared:  0.6246, Adjusted R-squared:  0.622
## F-statistic: 246.2 on 1 and 148 DF,  p-value: < 2.2e-16
```

```
y_hat<-predict(fitted,dat[test_index,])

mean((y_hat-dat[test_index,]$sales)^2)
```

```
## [1] 10.37809
```

(d) [6 points] Fit a multiple linear regression model including all the variables `TV`, `radio`, `newspaper` to model the `sales` in the training set. Then, compute the predicted sales for the test set with the new model and evaluate the RMSE. Did the error decrease from the one corresponding to the previous model?

- (YOUR ANSWER HERE)

```
fitted<-lm(sales~TV+radio+newspaper,data=dat,subset=train_index)

y_hat<-predict(fitted,dat[test_index,])

mean((y_hat-dat[test_index,]$sales)^2)
```

```
## [1] 2.867617
```

Yes. The error decreases to the previous model.

(e) [6 points] Look at the summary output for the multiple regression model and note which of the coefficient in the model is significant. Are all of them significant? If not refit the model including only the features found significant. Which of the models should you choose in view of RMSE?

- (YOUR ANSWER HERE)

```
fitted %>% summary() #newspaper variable is not significant to the model.
```

```
##
## Call:
## lm(formula = sales ~ TV + radio + newspaper, data = dat, subset = train_index)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -8.5702 -0.6798  0.1893  1.2629  2.6830
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept)  2.626963   0.363310   7.231 2.49e-11 ***
## TV           0.047717   0.001650  28.915  < 2e-16 ***
## radio        0.189920   0.009898  19.188  < 2e-16 ***
## newspaper   -0.001260   0.006599  -0.191    0.849
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.694 on 146 degrees of freedom
## Multiple R-squared:  0.9016, Adjusted R-squared:  0.8995
## F-statistic: 445.7 on 3 and 146 DF,  p-value: < 2.2e-16
```

```
fitted2<-lm(sales~TV+radio,data=dat,subset=train_index)

y_hat<-predict(fitted2,dat[test_index,])

mean((y_hat-dat[test_index,]$sales)^2)
```

```
## [1] 2.863947
```

Model without (predict variable) newspaper is significant in terms of RMSE.

(f) [5 points] Now use the `rpart` function to fit a regression tree to the `sales` data set, where the sales values are predicted by the amount of TV advertising. Use the `train` function to estimate the accuracy. Try out `cp` values of `seq(0, 0.05, 0.01)`. Plot the accuracy to report the results of the best model. Use `set.seed(123)` to fix randomness.

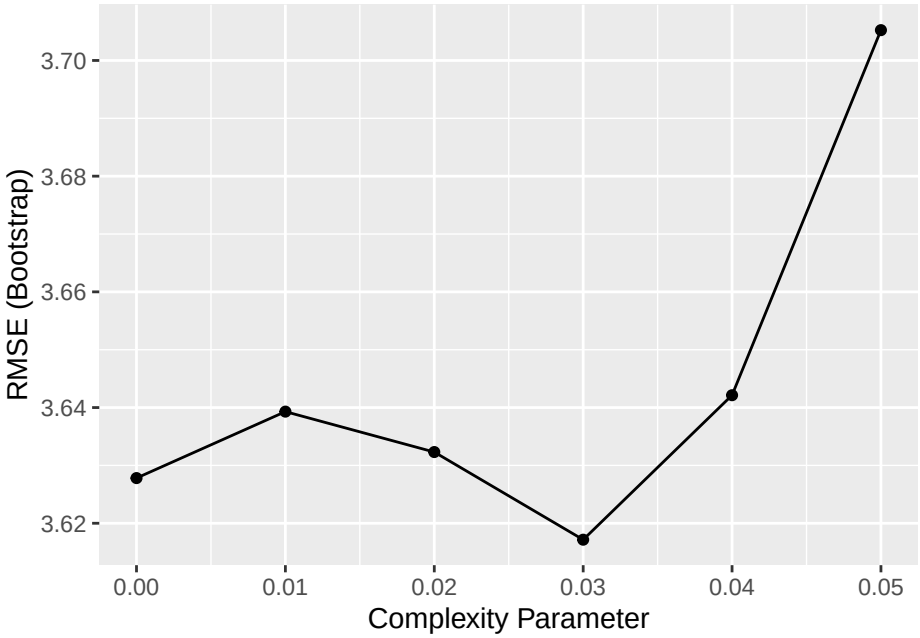
- (YOUR ANSWER HERE)

```
fit_reg<-rpart(sales~TV,data=dat,subset=train_index)
fit_reg
```

```
## n= 150
##
## node), split, n, deviance, yval
##       * denotes terminal node
##
```

```
##  1) root 150 4258.16600 14.220670
##    2) TV< 122.05 62  487.50770  9.729032
##      4) TV< 29.5 19   69.94526  6.615789 *
##      5) TV>=29.5 43  152.03910 11.104650 *
##    3) TV>=122.05 88 1638.55100 17.385230
##      6) TV< 240.9 69 1006.35900 16.397100
##       12) TV< 181.7 24  161.25830 14.791670 *
##       13) TV>=181.7 45  750.25200 17.253330
##         26) TV>=221.35 17  276.02470 15.582350 *
##         27) TV< 221.35 28  397.94110 18.267860
##           54) TV< 210.75 18  272.42500 17.250000 *
##           55) TV>=210.75 10   73.30000 20.100000 *
##      7) TV>=240.9 19  320.15680 20.973680 *
```

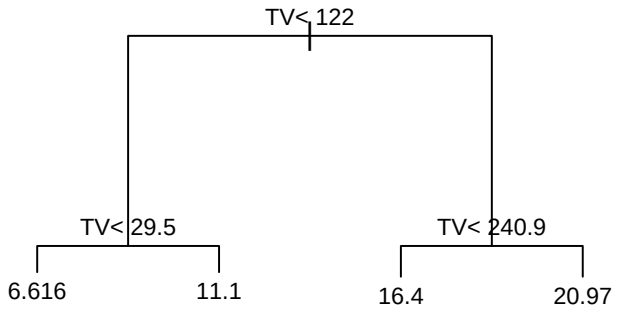
```
set.seed(123)
train_rpart<-train(sales~TV,method='rpart',
                   tuneGrid=data.frame(cp=seq(0,0.05,0.01)),data=dat, subset=train_index)
ggplot(train_rpart)
```



(g) [5 points] Draw the tree plot for the resulting regression tree from (f).

- (YOUR ANSWER HERE)

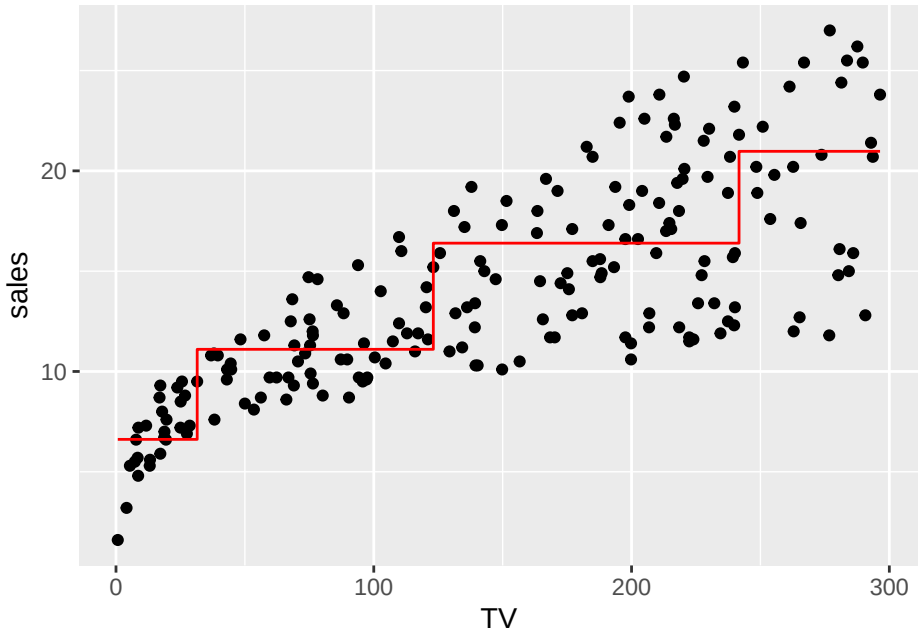
```
plot(train_rpart$finalModel,margin=0.1)
text(train_rpart$finalModel,cex=0.75)
```



(h) [5 points] As in (a), generate a scatterplot of sales against the amount of TV advertising and add the prediction curve from the regression tree from (f). Comment on it.

- (YOUR ANSWER HERE)

```
dat %>% mutate(y_hat=predict(train_rpart)) %>%
  ggplot()+geom_point(aes(TV,sales))+
  geom_step(aes(TV,y_hat),col='red')
```

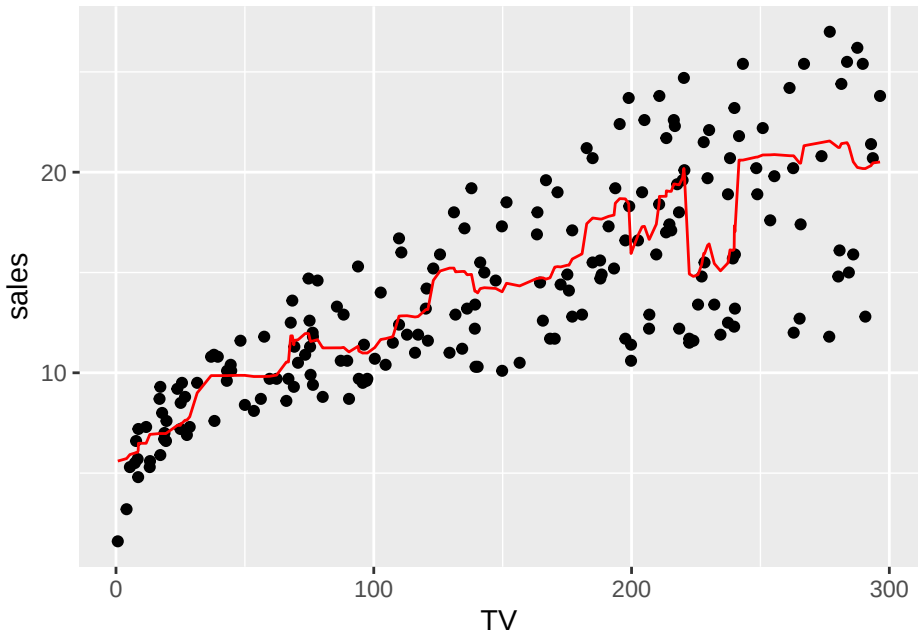


Since we controlled cp, the results has fewer nodes as we can see in the line.

(i) [5 points] Now, use `randomForest` function with `nodesize=20` and add the prediction curve into the scatter plot. Use `set.seed(123)` to fix randomness. Comment on it.

- (YOUR ANSWER HERE)

```
set.seed(123)
train<-randomForest(sales~TV,data=dat,subset=train_index,nodesize=20)
dat %>% mutate(y_hat_rf=predict(train,newdata=dat)) %>%
  ggplot()+geom_point(aes(TV,sales))+
  geom_line(aes(TV,y_hat_rf),col='red')
```



This randmoforest estimate is much smoother than previous approach(regression tree).We can say this is over-trained.(wiggly)

Problem Set #3 (20 Points)

From the following wikipedia page,

`https://en.wikipedia.org/wiki/List_of_metropolitan_statistical_areas`

you should find a table for the list of metropolitan statistical areas. Write a code to read this table into R. Display the first four columns of this table by changing their column names as `Rank`, `Metropolitan`, `Est2019` and `Cen2010`. Parse `Est2019` and `Cen2010` into numbers. Use `head` to print out first few rows.

- (YOUR ANSWER HERE)

```
library(rvest)
library(stringr)
url<-'https://en.wikipedia.org/wiki/List_of_metropolitan_statistical_areas'
 Sys.setlocale("LC_ALL", "English")
```

```
## [1] "LC_COLLATE=English_United States.1252;LC_CTYPE=English_United States.1252;LC_MONET
```

```
read_html(url) %>% html_nodes('table') %>% html_table(fill=T) ->raw
data<-raw[[2]]
data<-data[-6]
data %>% rename(Rank=Rank, Metropolitan=2,Est2019=3, Cen2010=4,Change=5)->data
data$Est2019<-str_replace_all(data$Est2019,',','')
data$Cen2010<-str_replace_all(data$Cen2010,',','')
data<-data %>% mutate(Est2019=as.numeric(Est2019),
                      Cen2010=as.numeric(Cen2010))
data$Change %>% head()
```

```
## [1] "+1.69%"  "+3.01%"  "-0.03%"  "+18.95%" "+19.35%" "+11.17%"
```

```
str_replace_all(data$Change,'%','')->data$Change

str_replace_all(data$Change,'\\+','')->data$Change


str_replace_all(data$Change,'\\-','')->data$Change
data$Change<-as.numeric(data$Change)
data %>% glimpse()
```

```
## Rows: 384
## Columns: 5
## $ Rank         <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 14, 15, 16,...
## $ Metropolitan <chr> "New York City-Newark-Jersey City, NY-NJ-PA MSA", "Los...
## $ Est2019      <dbl> 19216182, 13214799, 9458539, 7573136, 7066141, 6280487...
## $ Cen2010      <dbl> 18897109, 12828837, 9461105, 6366542, 5920416, 5649540...
## $ Change       <dbl> 1.69, 3.01, NA, 18.95, 19.35, 11.17, 10.82, 2.30, 13.8...
```