

DATA CLEANING
FOR THE
KELOWNA WEATHER-CRASH PROJECT

Contents

1	Introduction	2
2	Weather Data	3
3	Crash Data	11

1 Introduction

Here I present the data cleaning for each dataset, as well as the process of combining the datasets.

2 Weather Data

Loading in weather data:

```
> fullWeather = c()
> for (i in c(2017:2021)){
+   for (j in c(1:12)){
+     temp = subset(read.csv(paste0('.././weatherdata/en_climate_hourly_BC_1123939_',
+       sprintf("%02d", j), '-', i, '_P1H.csv')),
+       select = - c(`Temp.Flag`,
+         `Dew.Point.Temp.Flag`, `Rel.Hum.Flag`,
+         `Precip..Amount.Flag`, `Wind.Dir.Flag`,
+         `Wind.Spd.Flag`, `Visibility.Flag`,
+         `Stn.Press.Flag`, `Hmdx`, `Hmdx.Flag`, `Wind.Chill.Flag`))
+     fullWeather = rbind(fullWeather, temp)
+   }
+ }
> nrow(fullWeather)
```

```
[1] 43824
```

```
> 24*365*5 + 24 #2020 was a leap year
```

```
[1] 43824
```

```
> #assigning a Julian day variable
> library(lubridate)
> fullWeather$julianday = yday(as.Date(fullWeather$Date.Time..LST., tz='LST'))
> summary(fullWeather)
```

Longitude..x.	Latitude..y.	Station.Name	Climate.ID
Min. : -119.4	Min. : 49.96	KELOWNA:43824	Min. : 1123939
1st Qu.: -119.4	1st Qu.: 49.96		1st Qu.: 1123939
Median : -119.4	Median : 49.96		Median : 1123939
Mean : -119.4	Mean : 49.96		Mean : 1123939
3rd Qu.: -119.4	3rd Qu.: 49.96		3rd Qu.: 1123939
Max. : -119.4	Max. : 49.96		Max. : 1123939

Date.Time..LST.	Year	Month	Day
2017-01-01 00:00:	1 Min. : 2017	Min. : 1.000	Min. : 1.00
2017-01-01 01:00:	1 1st Qu.: 2018	1st Qu.: 4.000	1st Qu.: 8.00
2017-01-01 02:00:	1 Median : 2019	Median : 7.000	Median : 16.00
2017-01-01 03:00:	1 Mean : 2019	Mean : 6.524	Mean : 15.73
2017-01-01 04:00:	1 3rd Qu.: 2020	3rd Qu.: 10.000	3rd Qu.: 23.00
2017-01-01 05:00:	1 Max. : 2021	Max. : 12.000	Max. : 31.00
(Other)	: 43818		

Time..LST.	Temp...C.	Dew.Point.Temp...C.	Rel.Hum....
00:00 : 1826	Min. : -28.900	Min. : -32.800	Min. : 12.00
01:00 : 1826	1st Qu.: 0.800	1st Qu.: -2.100	1st Qu.: 52.00
02:00 : 1826	Median : 7.800	Median : 2.300	Median : 74.00
03:00 : 1826	Mean : 8.511	Mean : 2.043	Mean : 69.33
04:00 : 1826	3rd Qu.: 15.700	3rd Qu.: 7.500	3rd Qu.: 89.00
05:00 : 1826	Max. : 43.800	Max. : 19.700	Max. : 100.00

(Other):32868	NA's :30	NA's :29	NA's :24
Precip..Amount..mm.	Wind.Dir..10s.deg.	Wind.Spd..km.h.	Visibility..km.
Min. :0.00000	Min. : 1.00	Min. : 0.000	Min. : 0.0
1st Qu.:0.00000	1st Qu.: 9.00	1st Qu.: 4.000	1st Qu.:16.1
Median :0.00000	Median :18.00	Median : 5.000	Median :16.1
Mean :0.02999	Mean :19.13	Mean : 8.415	Mean :15.1
3rd Qu.:0.00000	3rd Qu.:33.00	3rd Qu.:11.000	3rd Qu.:16.1
Max. :7.10000	Max. :36.00	Max. :58.000	Max. :16.1
NA's :24	NA's :13955	NA's :52	NA's :31
Stn.Press..kPa.	Wind.Chill	Weather	julianday
Min. :93.73	Min. : -34.00	Rain : 2069	Min. : 1.0
1st Qu.:96.10	1st Qu.: -11.00	Snow : 1662	1st Qu.: 92.0
Median :96.52	Median : -6.00	Haze : 1187	Median :183.0
Mean :96.55	Mean : -8.22	Fog : 873	Mean :183.1
3rd Qu.:96.96	3rd Qu.: -4.00	Rain,Fog: 233	3rd Qu.:274.0
Max. :99.34	Max. : -1.00	(Other) : 246	Max. :366.0
NA's :34	NA's :36294	NA's :37554	

There are 22 empty strings in the `Weather` variable that should be NAs.

```
> length(fullWeather[fullWeather$Weather == "" &
+           is.na(fullWeather$Weather) == FALSE, "Weather"])
```

```
[1] 22
```

```
> ## SETTING EMPTY TO NA
> fullWeather$Weather[fullWeather$Weather == "" &
+           is.na(fullWeather$Weather) == FALSE] = NA
> ## CHECKING
> length(fullWeather[fullWeather$Weather == "" &
+           is.na(fullWeather$Weather) == FALSE, "Weather"])
```

```
[1] 0
```

Need to relevel factor to remove empty string option:

```
> "" %in% levels(fullWeather$Weather)
```

```
[1] TRUE
```

```
> fullWeather$Weather = droplevels(fullWeather$Weather)
> "" %in% levels(fullWeather$Weather)
```

```
[1] FALSE
```

Changing Weather factor variable to many indicator variables.

```
> #creating columns
> lst = c()
> for (i in levels(fullWeather$Weather)){
+   temp = unlist(strsplit(as.character(i), ","))
+   lst = c(lst, temp)
+ }
> lst = lst[!(duplicated(lst))] #removing duplicates
> lst

[1] "Fog"           "Freezing Fog"  "Freezing Rain" "Snow"
[5] "Haze"          "Rain"          "Moderate Snow" "Moderate Rain"
[9] "Thunderstorms" "Heavy Rain"    "Heavy Snow"    "Blowing Snow"

> for (i in lst){
+   fullWeather[, i] = factor("0", levels = c("0", "1"))
+ }
> names(fullWeather)

[1] "Longitude..x."      "Latitude..y."      "Station.Name"
[4] "Climate.ID"         "Date.Time..LST."   "Year"
[7] "Month"              "Day"               "Time..LST."
[10] "Temp...C."          "Dew.Point.Temp...C." "Rel.Hum...."
[13] "Precip..Amount..mm." "Wind.Dir..10s.deg." "Wind.Spd..km.h."
[16] "Visibility..km."    "Stn.Press..kPa."   "Wind.Chill"
[19] "Weather"            "julianday"         "Fog"
[22] "Freezing Fog"       "Freezing Rain"     "Snow"
[25] "Haze"               "Rain"              "Moderate Snow"
[28] "Moderate Rain"      "Thunderstorms"     "Heavy Rain"
[31] "Heavy Snow"         "Blowing Snow"

> #####
>
> #assigning values
> for (i in 1:nrow(fullWeather)){
+   if (!is.na(fullWeather$Weather[i])){
+     temp = as.character(fullWeather$Weather[i])
+     temp = unlist(strsplit(temp, ","))
+     for (j in temp){
+       fullWeather[i, j] = "1"
+     }
+   }
+ }
> #####
>
> #collapsing to only: Rain, Snow, Thunderstorms, Fog, Freezing Rain
> fullWeather$Rain[fullWeather$'Moderate Rain' == 1] = 1
> fullWeather$Rain[fullWeather$'Heavy Rain' == 1] = 1
> fullWeather$Snow[fullWeather$'Moderate Snow' == 1] = 1
> fullWeather$Snow[fullWeather$'Heavy Snow' == 1] = 1
> fullWeather$Snow[fullWeather$'Blowing Snow' == 1] = 1
```

```

> fullWeather$Fog[fullWeather$'Freezing Fog' == 1] = 1
> fullWeather$Fog[fullWeather$'Haze' == 1] = 1
> fullWeather = subset(fullWeather, select = - c(`Moderate Rain`,
+         `Heavy Rain`, `Moderate Snow`, `Heavy Snow`,
+         `Blowing Snow`, `Freezing Fog`, `Haze`))
> summary(fullWeather[,c("Weather", "Rain", "Fog", "Snow",
+         "Thunderstorms", "Freezing Rain")])

```

	Weather	Rain	Fog	Snow	Thunderstorms	Freezing Rain
Rain	: 2069	0:41370	0:41429	0:42132	0:43794	0:43819
Snow	: 1662	1: 2454	1: 2395	1: 1692	1: 30	1: 5
Haze	: 1187					
Fog	: 873					
Rain,Fog:	233					
(Other)	: 224					
NA's	:37576					

```

> #converting indicator columns to numeric
> fullWeather$Fog = as.numeric(as.character(fullWeather$Fog))
> fullWeather$'Freezing Rain' = as.numeric(as.character(fullWeather$'Freezing Rain'))
> fullWeather$Snow = as.numeric(as.character(fullWeather$Snow))
> fullWeather$Rain = as.numeric(as.character(fullWeather$Rain))
> fullWeather$Thunderstorms = as.numeric(as.character(fullWeather$Thunderstorms))

```

I realize that I could have used a regex solution here to shorten the code, but the solution above is more general. This is important if I wanted to add/remove groups to/from the reduced list of weather types later on (ie. adding 'Heavy Snow').

*Note that I kept **Rain** and **Freezing Rain** separate and mutually exclusive. Therefore, if an hour had freezing rain, the indicator variables will have "1" for **Freezing Rain**, and a "0" for **Rain**.

Testing if any hours have precipitation without the appropriate `Weather` variable:

```
> precip = which(fullWeather$Precip..Amount..mm. > 0 &
+               is.na(fullWeather$Precip..Amount..mm.) == FALSE &
+               is.na(fullWeather$Weather)
+ ) #gives rows that have precip but NA for weather
> length(precip)
```

```
[1] 356
```

There are 356 cases where there is precipitation but no associated `Weather` variable!

To fix this, we want to assign the appropriate `Weather` depending on the temperature. However, this is easier said than done. At what temperature does it snow versus rain?

We can ask the data:

```
> fullWeather$actual = NA
> for (i in 1:nrow(fullWeather)){
+   if (fullWeather$Rain[i] == 1){
+     fullWeather$actual[i] = 'Rain'
+   }
+   if (fullWeather$Snow[i] == 1){
+     fullWeather$actual[i] = 'Snow'
+   }
+ }
> assign = function(x){
+   if (is.na(x)){
+     return(NA)
+   }
+   if (x > 0){
+     return("Above 0")
+   } else {
+     return("Below 0")
+   }
+ }
> #classification table
> fullWeather$expected = sapply(fullWeather$Temp...C., assign)
> rainsnow = table(fullWeather$expected, fullWeather$actual)
> rainsnow
```

	Rain	Snow
Above 0	2438	202
Below 0	3	1490

```
> #probabilities
> #chance of raining if above 0
> rainprob = rainsnow[1,1]/(rainsnow[1,1]+rainsnow[1,2])
> rainprob
```

```
[1] 0.9234848
```

```
> #chance of snowing if below 0
> snowprob = rainsnow[2,2]/(rainsnow[2,2]+rainsnow[2,1])
> snowprob
```


[1] 0.9979906

In other words, whenever the temperature is above 0, it is usually raining (92%). Similarly, whenever the temperature is below 0, it is basically always snowing (99.8%).

Therefore, we could assign our missing values based on these probabilities. However, it is intuitively true that as you get further from zero, these probabilities would change. Therefore, we can just use a random forest model to predict whether or not it will be snowing or raining based on the temperature:

```
> training = subset(fullWeather[-precip,],
+                   subset = (is.na(fullWeather$actual[-precip]) == FALSE),
+                   select=c('Temp...C.', 'actual'))
> training$actual = as.factor(training$actual)
> library(randomForest)
> RFmodel = randomForest(actual~Temp...C., data=training, importance=TRUE)
> RFmodel

Call:
randomForest(formula = actual ~ Temp...C., data = training, importance = TRUE)
      Type of random forest: classification
      Number of trees: 500
No. of variables tried at each split: 1

      OOB estimate of  error rate: 2.93%
Confusion matrix:
      Rain Snow class.error
Rain 2366   75  0.03072511
Snow  46 1646  0.02718676

> plot(training, col='red', main="Rain/Snow RF", yaxt='n', ylab="")
> test = min(training$Temp...C.):max(training$Temp...C.)
> lines(test, predict(RFmodel,
+                   data.frame(Temp...C. = test), type = "response"), lwd=2)
> axis(side = 2, at = c(1,2), labels = c('Rain', 'Snow'), cex.axis=1.2, las=1)
> #predicting missing (precip) values
> rainsnowpredict = predict(RFmodel,
+                   fullWeather[precip,], type = "response")
> #old Rain and Snow variables
> summary(as.factor(fullWeather$Rain))

      0      1
41370 2454

> summary(as.factor(fullWeather$Snow))

      0      1
42132 1692

> #adding predictions
> #where 0 = raining and 1 = snowing
> for (i in names(rainsnowpredict)){
+   if (rainsnowpredict[i] == 'Rain'){
+     fullWeather[as.numeric(i), 'Rain'] = 1
+   }
+ }
```

```

+   fullWeather[as.numeric(i), 'Weather'] = 'Rain'
+ } else {
+   fullWeather[as.numeric(i), 'Snow'] = 1
+   fullWeather[as.numeric(i), 'Weather'] = 'Snow'
+ }
+ }
> #new Rain and Snow variables
> summary(as.factor(fullWeather$Rain))

  0    1
41065 2759

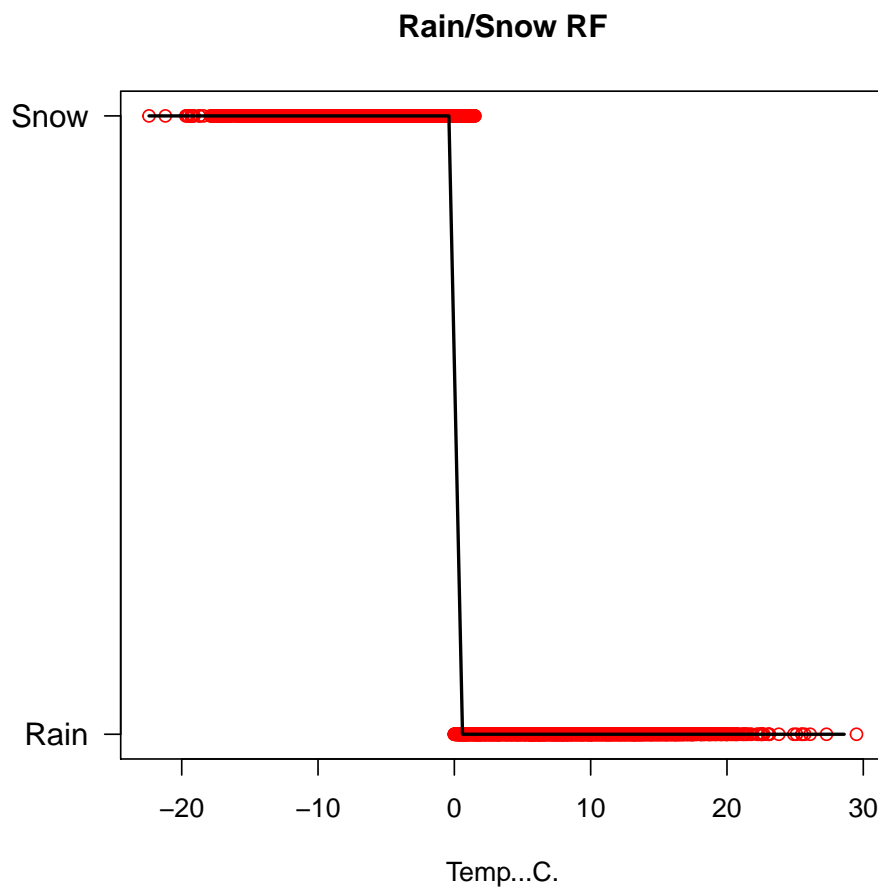
> summary(as.factor(fullWeather$Snow))

  0    1
42081 1743

> #checking that it worked
> length(which(fullWeather$Precip..Amount..mm. > 0 &
+             is.na(fullWeather$Precip..Amount..mm.) == FALSE &
+             is.na(fullWeather$Weather)))

[1] 0

```



3 Crash Data

Loading in crash data:

```
> fullCrash = subset(read.csv('.././crashdata/Southern Interior_Full Data_data.csv'),
+                      select = - c(`Crash.Breakdown.2`, `Region`,
+                                   `Municipality.Name..ifnull.`))
> summary(fullCrash)
```

Date.Of.Loss.Year	Animal.Flag	Crash.Severity	Cyclist.Flag
Min. :2017	No :54118	CASUALTY CRASH :11473	No :55725
1st Qu.:2018	Yes: 2018	PROPERTY DAMAGE ONLY:44663	Yes: 411
Median :2019			
Mean :2019			
3rd Qu.:2020			
Max. :2021			

Day.Of.Week	Derived.Crash.Configuration	Heavy.Veh.Flag
FRIDAY :9316	REAR END :13024	No :54085
MONDAY :8024	SINGLE VEHICLE :12495	Yes: 2051
SATURDAY :6753	UNDETERMINED :11453	
SUNDAY :5464	SIDE IMPACT :11369	
THURSDAY :9061	CONFLICTED : 3068	
TUESDAY :8728	SIDE SWIPE - SAME DIRECTION: 1833	
WEDNESDAY:8790	(Other) : 2894	

Intersection.Crash	Month.Of.Year	Motorcycle.Flag	Parked.Vehicle.Flag
No :31677	JULY : 5152	No :55673	No :38567
Yes:24459	DECEMBER: 5095	Yes: 463	Yes:17569
	JANUARY : 5063		
	AUGUST : 4907		
	OCTOBER : 4836		
	JUNE : 4735		
	(Other) :26348		

Parking.Lot.Flag	Pedestrian.Flag	Street.Full.Name..ifnull.
No :37189	No :55790	HWY 97 : 6019
Yes:18947	Yes: 346	HARVEY AVE : 3168
		HWY 33 : 2064
		GORDON DR : 1921
		LAKESHORE RD : 1372
		SPRINGFIELD RD: 1326
		(Other) :40266

Time.Category	Municipality.Name	Road.Location.Description
12:00-14:59:14870	KELOWNA :45943	UNKNOWN : 2211
15:00-17:59:14473	WEST KELOWNA:10193	HWY 97 : 1961
09:00-11:59:11021		HARVEY AVE : 1670
18:00-20:59: 5805		LAKESHORE RD: 932
06:00-08:59: 5489		LOUIE DR : 751
21:00-23:59: 2684		HWY 33 : 715
(Other) : 1794		(Other) :47896

Street.Full.Name	Metric.Selector	Total.Crashes	Total.Victims
HWY 97 : 6019	Min. :1.000	Min. :1.000	Min. :0.0000
HARVEY AVE : 3168	1st Qu.:1.000	1st Qu.:1.000	1st Qu.:0.0000

HWY 33	: 2064	Median	:1.000	Median	:1.000	Median	:0.0000
GORDON DR	: 1921	Mean	:1.006	Mean	:1.006	Mean	:0.2917
LAKESHORE RD	: 1372	3rd Qu.	:1.000	3rd Qu.	:1.000	3rd Qu.	:0.0000
SPRINGFIELD RD	: 1326	Max.	:3.000	Max.	:3.000	Max.	:9.0000
(Other)	:40266						

Now, while the crash data is overall much cleaner out-of-the-box than the weather data, the issue is that this crash dataset was made to be 4-anonymous, as the day of the month is not given. Instead, all we have is the month, year, and day of the week. This means we cannot perfectly know the weather on any given day. This massively reduces the power of any statistical tests that we perform with the data, but without being able to reidentify any of the observations, this is the best that we can do.

A solution to this problem would be to randomly assign each of the 4 (or 5) possible weather conditions in each month to each of the corresponding possible days. A better alternative would be to average the weather conditions on each of the possible 4 days and then assign this same (averaged) weather condition to all 4 (or 5) days in that month. The precipitation type (rain, snow, fog, etc.) would then also be averaged (2 snows + 1 rain = snow) with a priority rating (1 snow + 1 rain = snow).

In order to do this, we will need to create a new column on which to join the two datasets. For the crash data, this is easy.

```
> fullCrash$linker = paste(fullCrash$Date.Of.Loss.Year,
+                           fullCrash$Month.Of.Year,
+                           fullCrash$Day.Of.Week,
+                           fullCrash$Time.Category)
> head(fullCrash$linker)

[1] "2017 DECEMBER FRIDAY 00:00-02:59" "2017 FEBRUARY FRIDAY 00:00-02:59"
[3] "2017 APRIL FRIDAY 06:00-08:59"    "2017 JANUARY FRIDAY 06:00-08:59"
[5] "2017 JUNE FRIDAY 06:00-08:59"     "2017 OCTOBER FRIDAY 06:00-08:59"
```

For the weather data, we first collapse the hourly data to the same time periods as the crash data (00:00-02:59, 03:00-05:59, 06:00-08:59, 09:00-11:59, 12:00-14:59, 15:00-17:59, 18:00-20:59, 21:00-23:59):

```
> weatherTime = function(x){
+   #assigns the correct time category
+   if (x %in% c('00:00', '01:00', '02:00')){
+     return('00:00-02:59')
+   }
+   if (x %in% c('03:00', '04:00', '05:00')){
+     return('03:00-05:59')
+   }
+   if (x %in% c('06:00', '07:00', '08:00')){
+     return('06:00-08:59')
+   }
+   if (x %in% c('09:00', '10:00', '11:00')){
+     return('09:00-11:59')
+   }
+   if (x %in% c('12:00', '13:00', '14:00')){
+     return('12:00-14:59')
+   }
+ }
```

```

+   if (x %in% c('15:00', '16:00', '17:00')){
+     return('15:00-17:59')
+   }
+   if (x %in% c('18:00', '19:00', '20:00')){
+     return('18:00-20:59')
+   }
+   if (x %in% c('21:00', '22:00', '23:00')){
+     return('21:00-23:59')
+   }
+ }
> fullWeather$timeCategory = sapply(fullWeather$Time..LST., weatherTime)

```

Next, we need to get the day of the week. Then we can create the linker for the weather data:

```

> #weekday, in uppercase
> fullWeather$weekday = toupper(as.character(wday(as.Date(fullWeather$Date.Time..LST., tz='LST'))))
> #month
> fullWeather$monthName = toupper(as.character(month(as.Date(fullWeather$Date.Time..LST., tz='LST'))))
> #creating weather linker column
> fullWeather$linker = paste(fullWeather$Year,
+                             fullWeather$monthName,
+                             fullWeather$weekday,
+                             fullWeather$timeCategory)
> head(fullWeather$linker)

[1] "2017 JANUARY SUNDAY 00:00-02:59" "2017 JANUARY SUNDAY 00:00-02:59"
[3] "2017 JANUARY SUNDAY 00:00-02:59" "2017 JANUARY SUNDAY 03:00-05:59"
[5] "2017 JANUARY SUNDAY 03:00-05:59" "2017 JANUARY SUNDAY 03:00-05:59"

> #we now can average the weather over these time periods
> timereducedWeather = c()
> j = 1
> for (i in seq(1, nrow(fullWeather), 3)){
+   #creating a whole new dataframe
+   timereducedWeather$linker[j] = fullWeather$linker[i]
+   #avg of temp
+   timereducedWeather$Temp...C.[j] = mean(c(fullWeather$Temp...C.[i],
+       fullWeather$Temp...C.[i+1], fullWeather$Temp...C.[i+2]), na.rm = TRUE)
+   #avg of dew point temp
+   timereducedWeather$Dew.Point.Temp...C.[j] = mean(c(fullWeather$Dew.Point.Temp...C.[i],
+       fullWeather$Dew.Point.Temp...C.[i+1], fullWeather$Dew.Point.Temp...C.[i+2]), na.rm = TRUE)
+   #avg of relative humidity
+   timereducedWeather$Rel.Hum....[j] = mean(c(fullWeather$Rel.Hum....[i],
+       fullWeather$Rel.Hum....[i+1], fullWeather$Rel.Hum....[i+2]), na.rm = TRUE)
+   #avg of precipitation amount
+   timereducedWeather$Precip..Amount..mm.[j] = mean(c(fullWeather$Precip..Amount..mm.[i],
+       fullWeather$Precip..Amount..mm.[i+1], fullWeather$Precip..Amount..mm.[i+2]), na.rm = TRUE)
+   #avg of wind direction
+   timereducedWeather$Wind.Dir..10s.deg.[j] = mean(c(fullWeather$Wind.Dir..10s.deg.[i],
+       fullWeather$Wind.Dir..10s.deg.[i+1], fullWeather$Wind.Dir..10s.deg.[i+2]), na.rm = TRUE)
+   #avg of wind speed

```

```

+ timereducedWeather$Wind.Spd..km.h.[j] = mean(c(fullWeather$Wind.Spd..km.h.[i],
+   fullWeather$Wind.Spd..km.h.[i+1], fullWeather$Wind.Spd..km.h.[i+2]), na.rm = TRUE)
+ #avg of visibility
+ timereducedWeather$Visibility..km.[j] = mean(c(fullWeather$Visibility..km.[i],
+   fullWeather$Visibility..km.[i+1], fullWeather$Visibility..km.[i+2]), na.rm = TRUE)
+ #avg of stn pressure
+ timereducedWeather$Stn.Press..kPa.[j] = mean(c(fullWeather$Stn.Press..kPa.[i],
+   fullWeather$Stn.Press..kPa.[i+1], fullWeather$Stn.Press..kPa.[i+2]), na.rm = TRUE)
+ #avg of wind chill
+ timereducedWeather$Wind.Chill[j] = mean(c(fullWeather$Wind.Chill[i],
+   fullWeather$Wind.Chill[i+1], fullWeather$Wind.Chill[i+2]), na.rm = TRUE)
+ #indicator variables
+ timereducedWeather$Fog[j] = max(c(fullWeather$Fog[i], fullWeather$Fog[i+1],
+   fullWeather$Fog[i+2]))
+ timereducedWeather$'Freezing Rain'[j] = max(c(fullWeather$'Freezing Rain'[i],
+   fullWeather$'Freezing Rain'[i+1], fullWeather$'Freezing Rain'[i+2]))
+ timereducedWeather$Snow[j] = max(c(fullWeather$Snow[i], fullWeather$Snow[i+1],
+   fullWeather$Snow[i+2]))
+ timereducedWeather$Rain[j] = max(c(fullWeather$Rain[i], fullWeather$Rain[i+1],
+   fullWeather$Rain[i+2]))
+ timereducedWeather$Thunderstorms[j] = max(c(fullWeather$Thunderstorms[i],
+   fullWeather$Thunderstorms[i+1], fullWeather$Thunderstorms[i+2]))
+ #incrementing ticker
+ j = j+1
+ }
> timereducedWeather = as.data.frame(timereducedWeather)
> head(timereducedWeather)

```

			linker	Temp...C.	Dew.Point.Temp...C.	Rel.Hum....
1	2017	JANUARY	SUNDAY	00:00-02:59	-4.600000	-5.200000 95.33333
2	2017	JANUARY	SUNDAY	03:00-05:59	-4.266667	-5.500000 91.33333
3	2017	JANUARY	SUNDAY	06:00-08:59	-3.533333	-5.466667 86.66667
4	2017	JANUARY	SUNDAY	09:00-11:59	-3.033333	-6.066667 79.66667
5	2017	JANUARY	SUNDAY	12:00-14:59	-2.333333	-6.866667 71.33333
6	2017	JANUARY	SUNDAY	15:00-17:59	-2.833333	-7.733333 69.33333
				Precip..Amount..mm.	Wind.Dir..10s.deg.	Wind.Spd..km.h. Visibility..km.
1				0	35.33333	11.666667 4.833333
2				0	3.50000	4.333333 9.666667
3				0	3.00000	15.666667 13.433333
4				0	25.00000	25.666667 16.100000
5				0	35.66667	38.666667 13.966667
6				0	35.66667	36.666667 15.566667
				Stn.Press..kPa.	Wind.Chill	Fog Freezing.Rain Snow Rain Thunderstorms
1				95.57333	-9.000000	0 0 1 0 0
2				95.65000	-7.500000	0 0 1 0 0
3				95.74333	-8.666667	0 0 1 0 0
4				95.90000	-9.666667	0 0 1 0 0
5				95.96333	-10.333333	1 0 0 0 0
6				96.10333	-10.666667	0 0 0 0 0

Now, it is important to re-focus on what we want to do. We want to know the weather that was occurring during each crash (ie. for each row in the crash dataset). The only thing

preventing us from doing this is that for each unique linker in the crash data, there are 4 (or 5) equivalent linkers in the weather dataset. In order to remove this ambiguity, we need to average the weather over the weather rows that have identical linkers.

```
> #proof of principle (shows number of times each unique linker occurs)
> table(table(timereducedWeather$linker))
```

```
    4    5
2192 1168
```

```
> #getting indices of duplicates
> idx = duplicated(timereducedWeather$linker)
> finalWeather = timereducedWeather[!idx,]
```

Now, the final step. We need to join the crash and weather datasets.

```
> alldata = merge(x=fullCrash,y=finalWeather,by="linker",all.x=TRUE, sort=FALSE)
> head(alldata)
```

	linker	Date.Of.Loss.Year	Animal.Flag
1	2017 DECEMBER FRIDAY 00:00-02:59	2017	No
2	2017 DECEMBER FRIDAY 00:00-02:59	2017	No
3	2017 DECEMBER FRIDAY 00:00-02:59	2017	Yes
4	2017 DECEMBER FRIDAY 00:00-02:59	2017	No
5	2017 DECEMBER FRIDAY 00:00-02:59	2017	No
6	2017 FEBRUARY FRIDAY 00:00-02:59	2017	No

	Crash.Severity	Cyclist.Flag	Day.Of.Week	Derived.Crash.Configuration
1	CASUALTY CRASH	No	FRIDAY	SINGLE VEHICLE
2	PROPERTY DAMAGE ONLY	No	FRIDAY	UNDETERMINED
3	PROPERTY DAMAGE ONLY	No	FRIDAY	SINGLE VEHICLE
4	PROPERTY DAMAGE ONLY	No	FRIDAY	REAR END
5	PROPERTY DAMAGE ONLY	No	FRIDAY	SINGLE VEHICLE
6	PROPERTY DAMAGE ONLY	No	FRIDAY	UNDETERMINED

	Heavy.Veh.Flag	Intersection.Crash	Month.Of.Year	Motorcycle.Flag
1	No	No	DECEMBER	No
2	No	No	DECEMBER	No
3	No	Yes	DECEMBER	No
4	No	No	DECEMBER	No
5	No	No	DECEMBER	No
6	No	No	FEBRUARY	No

	Parked.Vehicle.Flag	Parking.Lot.Flag	Pedestrian.Flag
1	No	No	No
2	Yes	Yes	No
3	No	No	No
4	No	Yes	No
5	No	Yes	No
6	Yes	No	No

	Street.Full.Name..ifnull.	Time.Category	Municipality.Name
1	DOBBIN RD	00:00-02:59	KELOWNA
2	SMITH AVE	00:00-02:59	KELOWNA
3	MOUNT ROYAL DR	00:00-02:59	KELOWNA
4	BURTCH ST	00:00-02:59	KELOWNA
5	BANKS RD	00:00-02:59	KELOWNA

```

6           HWY 97    00:00-02:59           KELOWNA
      Road.Location.Description Street.Full.Name Metric.Selector Total.Crashes
1           UNKNOWN          DOBBIN RD           1           1
2           SMITH AVE          SMITH AVE           1           1
3 MOUNT ROYAL DR & ROYAL VIEW DR MOUNT ROYAL DR           1           1
4           UNKNOWN          BURTCH ST            1           1
5           BANKS RD          BANKS RD            1           1
6           HWY 97           HWY 97              1           1
Total.Victims Temp...C. Dew.Point.Temp...C. Rel.Hum.... Precip..Amount..mm.
1           2    2.233333          0.03333333    85.33333          0
2           0    2.233333          0.03333333    85.33333          0
3           0    2.233333          0.03333333    85.33333          0
4           0    2.233333          0.03333333    85.33333          0
5           0    2.233333          0.03333333    85.33333          0
6           0 -12.766667         -14.53333333    87.00000          0
Wind.Dir..10s.deg. Wind.Spd..km.h. Visibility..km. Stn.Press..kPa. Wind.Chill
1           16.5          5.333333          16.1          96.65333      NaN
2           16.5          5.333333          16.1          96.65333      NaN
3           16.5          5.333333          16.1          96.65333      NaN
4           16.5          5.333333          16.1          96.65333      NaN
5           16.5          5.333333          16.1          96.65333      NaN
6           NaN           0.000000          16.1          96.98333      NaN
Fog Freezing.Rain Snow Rain Thunderstorms
1    0           0    0    0           0
2    0           0    0    0           0
3    0           0    0    0           0
4    0           0    0    0           0
5    0           0    0    0           0
6    0           0    0    0           0

```

Saving the cleaned dataset:

```
> save(alldata, file = "../rda_files/all_data.rda")
```