> Exploratory analysis is what you do to understand the data and figure out what might be noteworthy or interesting to highlight to others. When we do exploratory analysis, it's like hunting for pearls in oysters.
>
> — Cole Nussbaumer Knaflic (Knaflic, 2015)

Data Science for Operational Researchers using R

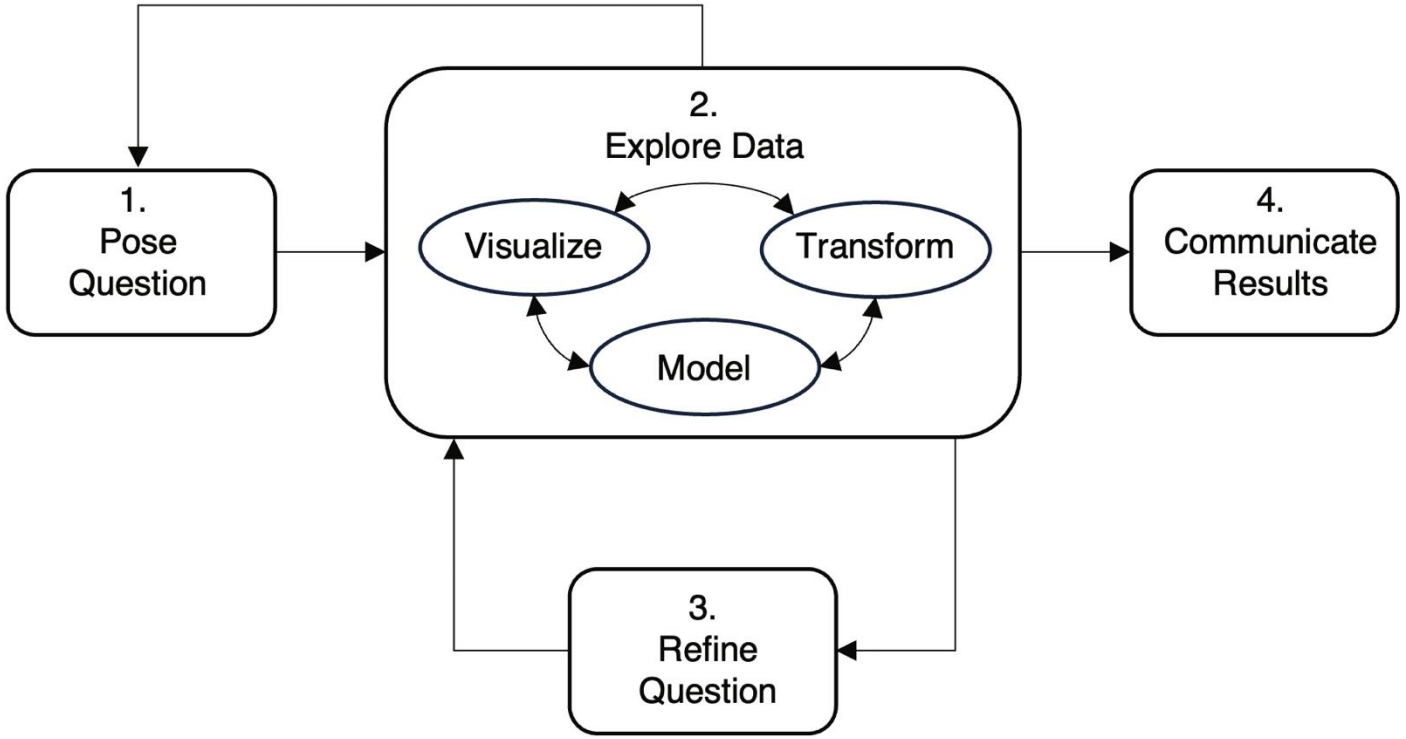
09 – Exploratory Data Analysis

https://github.com/JimDuggan/explore_or

Exploratory Data Analysis

- Exploratory data analysis (EDA) involves reviewing the features and characteristics of a dataset with an “open mind”, and is frequently used upon “first contact with the data” (EDA, 2008).
- A convenient way to pursue EDA is to use questions as a means to guide your exploration, as this process focuses your attention on specific aspects of the dataset (Wickham et al., 2023).
- An attractive feature of EDA is that there are no constraints on the type of question posed, and therefore it can be viewed as a creative process

Process for EDA



- Can a plant's dimensions help us uniquely identify a species?
- Based on observations from 2012, 2013, and 2014, is there a potential association between temperature and energy demand in the state of Victoria, Australia?
- For a Boston dataset from the 1970s, can we find possible relationships between house value and pupil–teacher ratios across different suburbs?
- What passengers had the greatest chance of survival on the Titanic?
- During the winter season in Ireland, is there initial evidence to suggest that wind direction has an impact on temperature?

(1) The iris dataset

- A collection of 150 observations based on data collected by the American botanist Edgar Anderson.
- For three species of the iris flower (setosa, versicolor, and virginica) measurements (in cm) relating to the length and width of both the sepal, and the petal are recorded.

http://www.lac.inpe.br/~rafael.santos/Docs/CAP394/WholeStory-Iris.html

```
library(dplyr)
iris_tb <- dplyr::as_tibble(iris)
dplyr::slice(iris_tb,c(1:2,51:52,101:102))
#> # A tibble: 6 x 5
#>   Sepal.Length Sepal.Width Petal.Length Petal.Width Species
#>          <dbl>       <dbl>        <dbl>       <dbl> <fct>
#> 1          5.1         3.5          1.4         0.2 setosa
#> 2          4.9         3            1.4         0.2 setosa
#> 3          7           3.2          4.7         1.4 versicolor
#> 4          6.4         3.2          4.5         1.5 versicolor
#> 5          6.3         3.3          6           2.5 virginica
#> 6          5.8         2.7          5.1         1.9 virginica
```

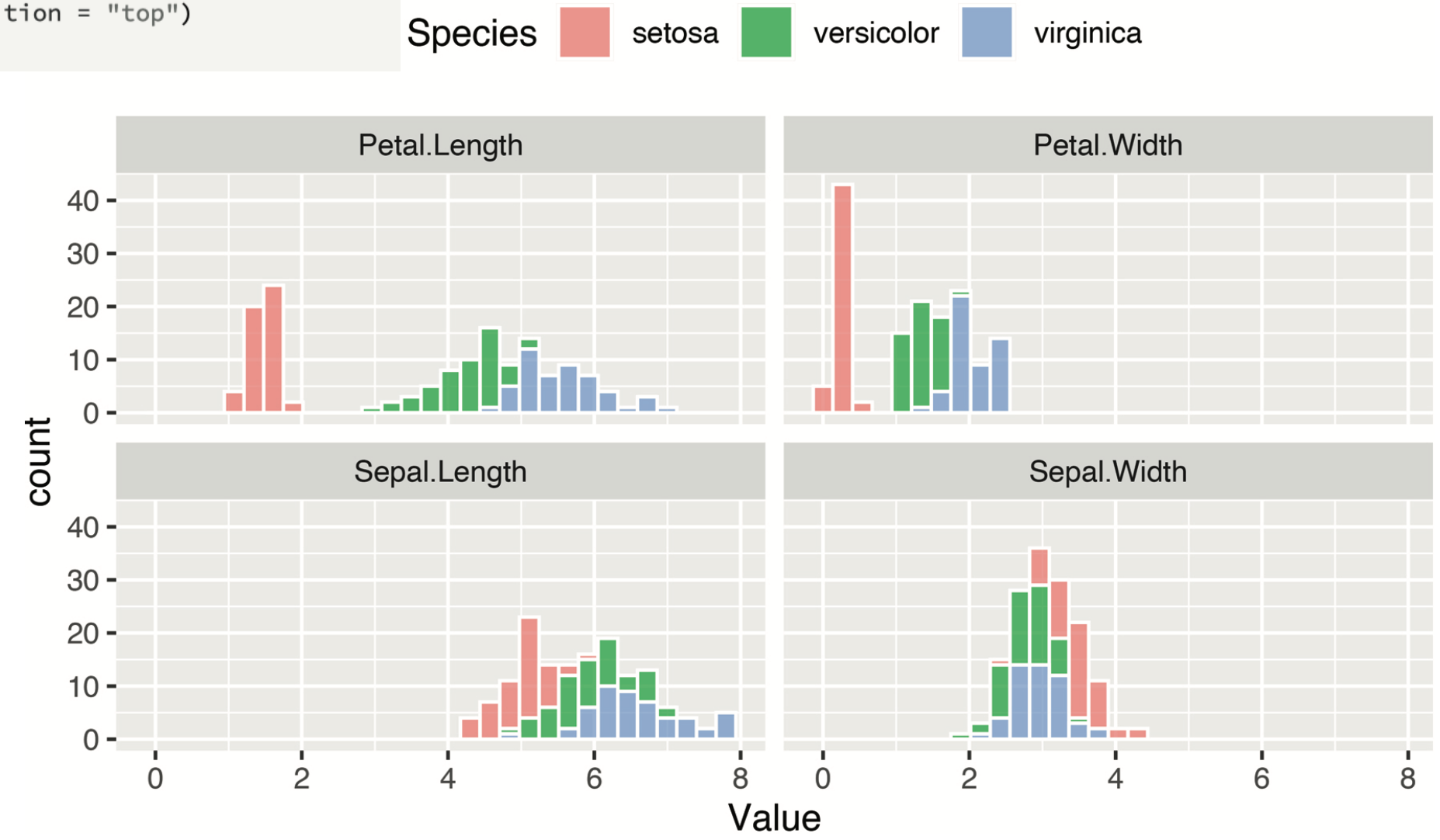
Convert to longer format to support visualisation

```
library(tidyr)
iris_long <- tidyr::pivot_longer(iris_tb,
                                 names_to = "Measurement",
                                 values_to = "Value",
                                 -Species)

head(iris_long)
#> # A tibble: 6 x 3
#>   Species Measurement  Value
#>   <fct>   <chr>        <dbl>
#> 1 setosa  Sepal.Length   5.1
#> 2 setosa  Sepal.Width    3.5
#> 3 setosa  Petal.Length   1.4
#> 4 setosa  Petal.Width    0.2
#> 5 setosa  Sepal.Length   4.9
#> 6 setosa  Sepal.Width    3
```

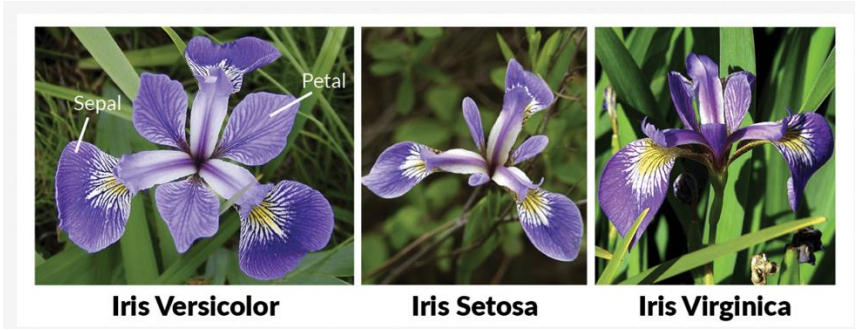
```
p1 <- ggplot(iris_long,aes(x=Value,fill=Species))+
      geom_histogram(color="white", alpha=0.7)+
      facet_wrap(~Measurement,ncol=2)+
      theme(legend.position = "top")
p1
```

Visualise

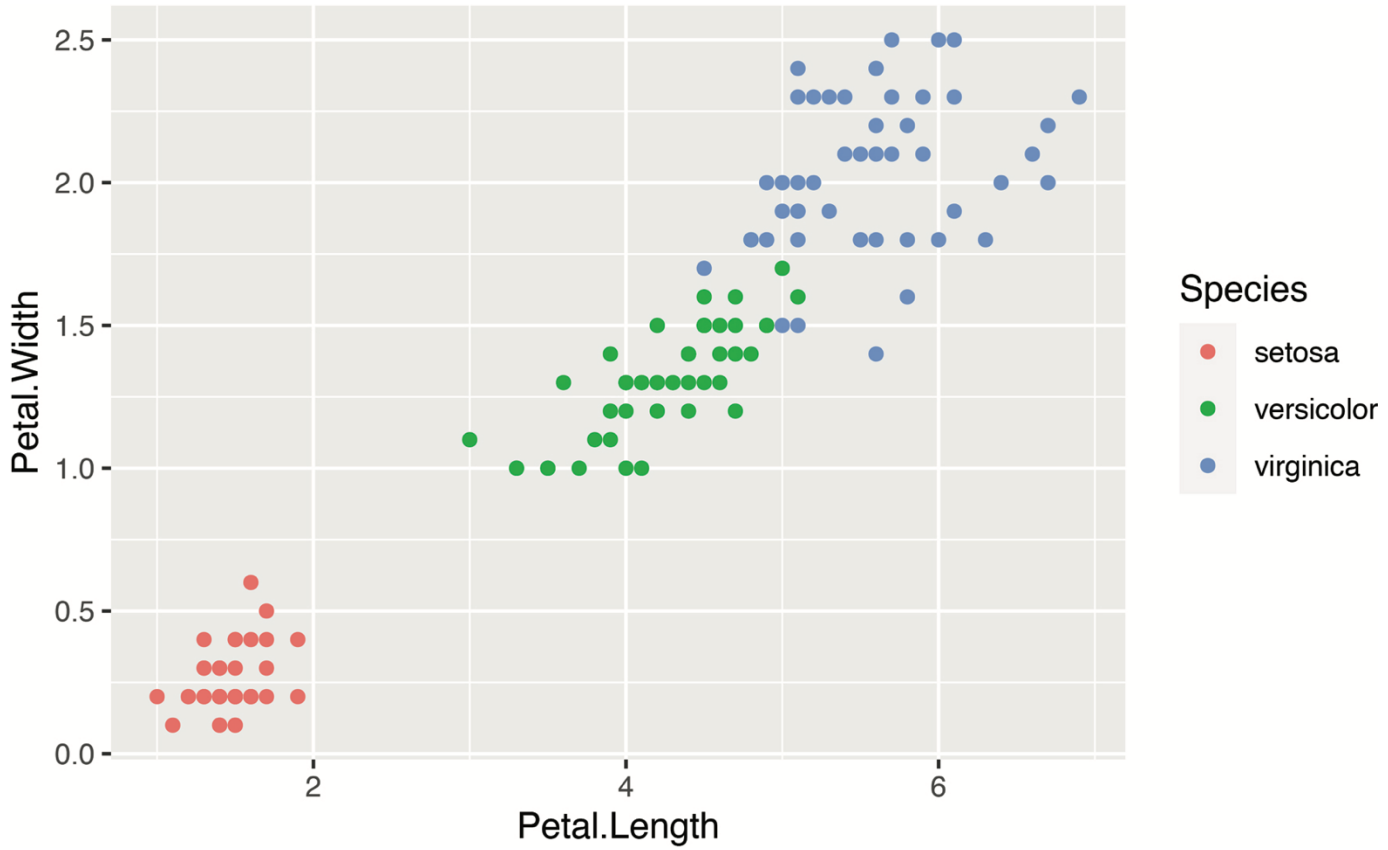


Initial comments…

- The petal variables visualized on the top row show a clearer separation between the species.
- Specifically, for both the petal length and width, the setosa histogram plot is fully distinct from the other two.
- There also seems to be a difference between the versicolor and virginica species.



Another view




```
res <- iris_long %>%
        dplyr::filter(Measurement %in% c("Petal.Length",
                                         "Petal.Width")) %>%
        dplyr::group_by(Species,Measurement) %>%
        dplyr::summarize(Min=min(Value),
                         Q25=quantile(Value,0.25),
                         Median=median(Value),
                         Mean=mean(Value),
                         Q75=quantile(Value,0.75),
                         Max=max(Value)) %>%
        dplyr::ungroup() %>%
        dplyr::arrange(Measurement,Mean)
res
#> # A tibble: 6 x 8
#>   Species    Measurement    Min   Q25 Median  Mean   Q75   Max
#>   <fct>      <chr>        <dbl> <dbl>  <dbl> <dbl> <dbl> <dbl>
#> 1 setosa     Petal.Length   1     1.4   1.5  1.46   1.58   1.9
#> 2 versicolor Petal.Length   3     4     4.35 4.26   4.6    5.1
#> 3 virginica  Petal.Length   4.5   5.1   5.55 5.55   5.88   6.9
#> 4 setosa     Petal.Width    0.1   0.2   0.2  0.246  0.3    0.6
#> 5 versicolor Petal.Width    1     1.2   1.3  1.33   1.5    1.8
#> 6 virginica  Petal.Width    1.4   1.8   2    2.03   2.3    2.5
```

(2) Temperature and Energy Demand

- Our next example focuses on time series data recorded in Victoria, Australia from 2012 through to 2014, and it can be accessed through CRAN (Hyndman and Athanasopoulos, 2021).
- The dataset contains electricity demand values (megawatt hours) and the corresponding temperature levels (celsius) at 30 minute intervals.

```
library(ggplot2)
library(dplyr)
library(tsibble)
library(tsibbledata)
library(lubridate)
library(tidyr)
library(ggpubr)
```

```
aus_elec <- vic_elec %>%
              tsibble::as_tibble()
aus_elec %>% dplyr::slice(1:3)
#> # A tibble: 3 x 5
#>   Time                Demand Temperature Date       Holiday
#>   <dttm>               <dbl>       <dbl> <date>     <lgl>
#> 1 2012-01-01 00:00:00  4383.        21.4 2012-01-01 TRUE
#> 2 2012-01-01 00:30:00  4263.        21.0 2012-01-01 TRUE
#> 3 2012-01-01 01:00:00  4049.        20.7 2012-01-01 TRUE
```

Available data

There are four variables in `aus_elec`:

- `Time`, which is a timestamp of the observation time. This is an important R data structure, as from this we can extract, using the package `lubridate`, addition time-related features such as the weekday.
- `Demand`, the total electricity demand in megawatt hours (MWh).
- `Temperature`, the temperature in Melbourne, which is the capital of the Australian state of Victoria.
- `Date`, the date of each observation.
- `Holiday`, a logical value indicating whether the day is a public holiday.

Variables added

As part of our exploratory data analysis, we want to create additional variables based on the variable `Time`. These can be extracted using functions from the package `lubridate`, for example:

- `wday()`, which returns the weekday and can be represented as a factor (e.g., Sun, Mon, etc.).
- `year()`, which returns the year.
- `month()`, which returns the month.
- `hour()`, which returns the hour.
- `yday()`, which returns the day number for the year (1 to 365 or 366).

```r
aus_elec <- aus_elec %>%
             dplyr::mutate(WDay=wday(Time,label=TRUE),
                           Year=year(Time),
                           Month=as.integer(month(Time)),
                           Hour=hour(Time),
                           YearDay=yday(Time),
                           Quarter=case_when(
                             Month %in% 1:3   ~ "Q1",
                             Month %in% 4:6   ~ "Q2",
                             Month %in% 7:9   ~ "Q3",
                             Month %in% 10:12 ~ "Q4",
                             TRUE ~ "Undefined"
                           ),
                           DaySegment=case_when(
                             Hour %in%  0:5  ~ "S1",
                             Hour %in% 6:11  ~ "S2",
                             Hour %in% 12:17 ~ "S3",
                             Hour %in% 18:23 ~ "S4",
                             TRUE ~ "Undefined"
                           ))

dplyr::slice(aus_elec,1:3)
#> # A tibble: 3 x 12
#>   Time                Demand Temperature Date       Holiday WDay
#>   <dttm>               <dbl>       <dbl> <date>     <lgl>   <ord>
#> 1 2012-01-01 00:00:00  4383.        21.4 2012-01-01 TRUE    Sun
#> 2 2012-01-01 00:30:00  4263.        21.0 2012-01-01 TRUE    Sun
```

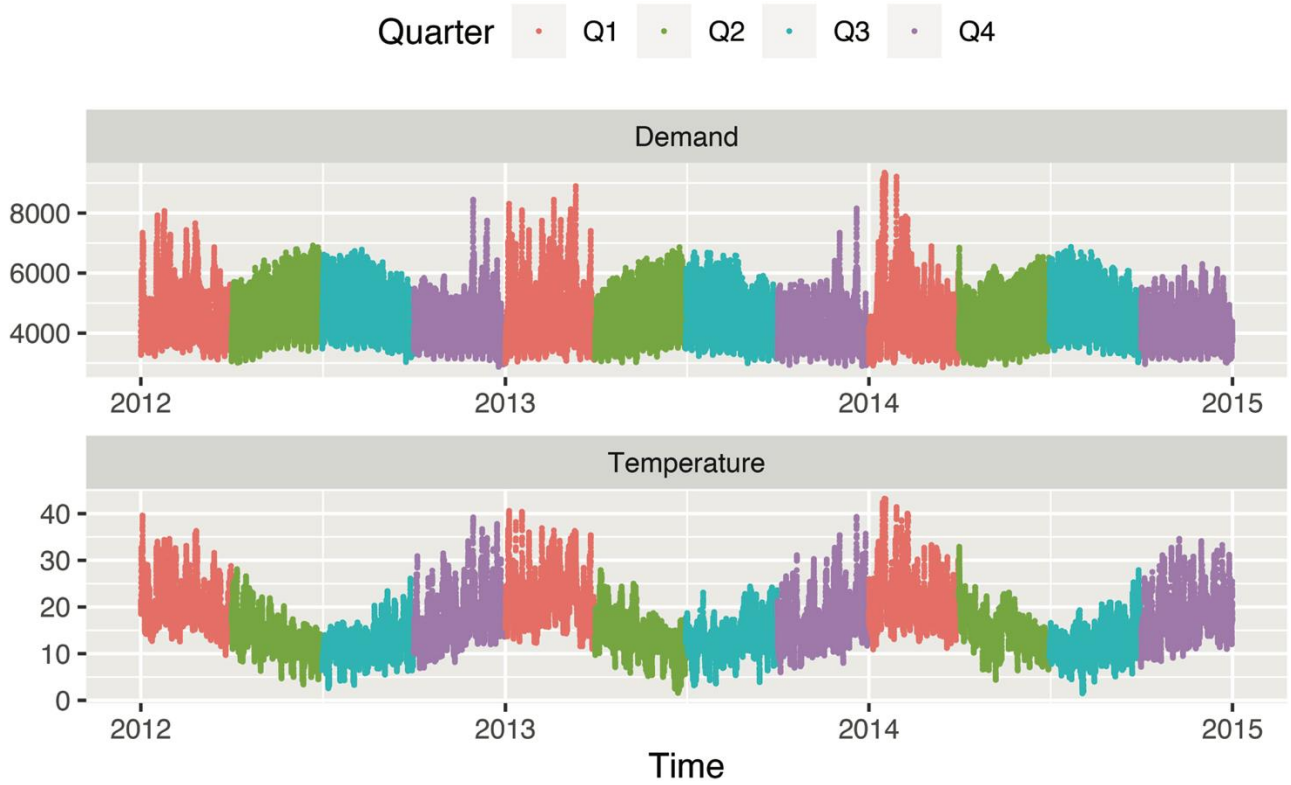
The resulting tibble…

```
dplyr::slice(aus_elec,1:3)
#> # A tibble: 3 x 12
#>   Time                Demand Temperature Date       Holiday WDay
#>   <dttm>               <dbl>       <dbl> <date>     <lgl>   <ord>
#> 1 2012-01-01 00:00:00  4383.        21.4 2012-01-01 TRUE    Sun
#> 2 2012-01-01 00:30:00  4263.        21.0 2012-01-01 TRUE    Sun
#> 3 2012-01-01 01:00:00  4049.        20.7 2012-01-01 TRUE    Sun
#> # ... with 6 more variables: Year <dbl>, Month <int>,
#> #   Hour <int>, YearDay <dbl>, Quarter <chr>, DaySegment <chr>
```

Exploring the time series…

```
aus_long <- aus_elec %>%
             dplyr::select(Time,Demand,Temperature,Quarter) %>%
             tidyr::pivot_longer(names_to="Indicator",
                                 values_to="Value",
                                 -c(Time,Quarter))
dplyr::slice(aus_long,1:3)
#> # A tibble: 3 x 4
#>   Time                Quarter Indicator   Value
#>   <dttm>              <chr>   <chr>       <dbl>
#> 1 2012-01-01 00:00:00 Q1      Demand      4383.
#> 2 2012-01-01 00:00:00 Q1      Temperature   21.4
#> 3 2012-01-01 00:30:00 Q1      Demand      4263.
```

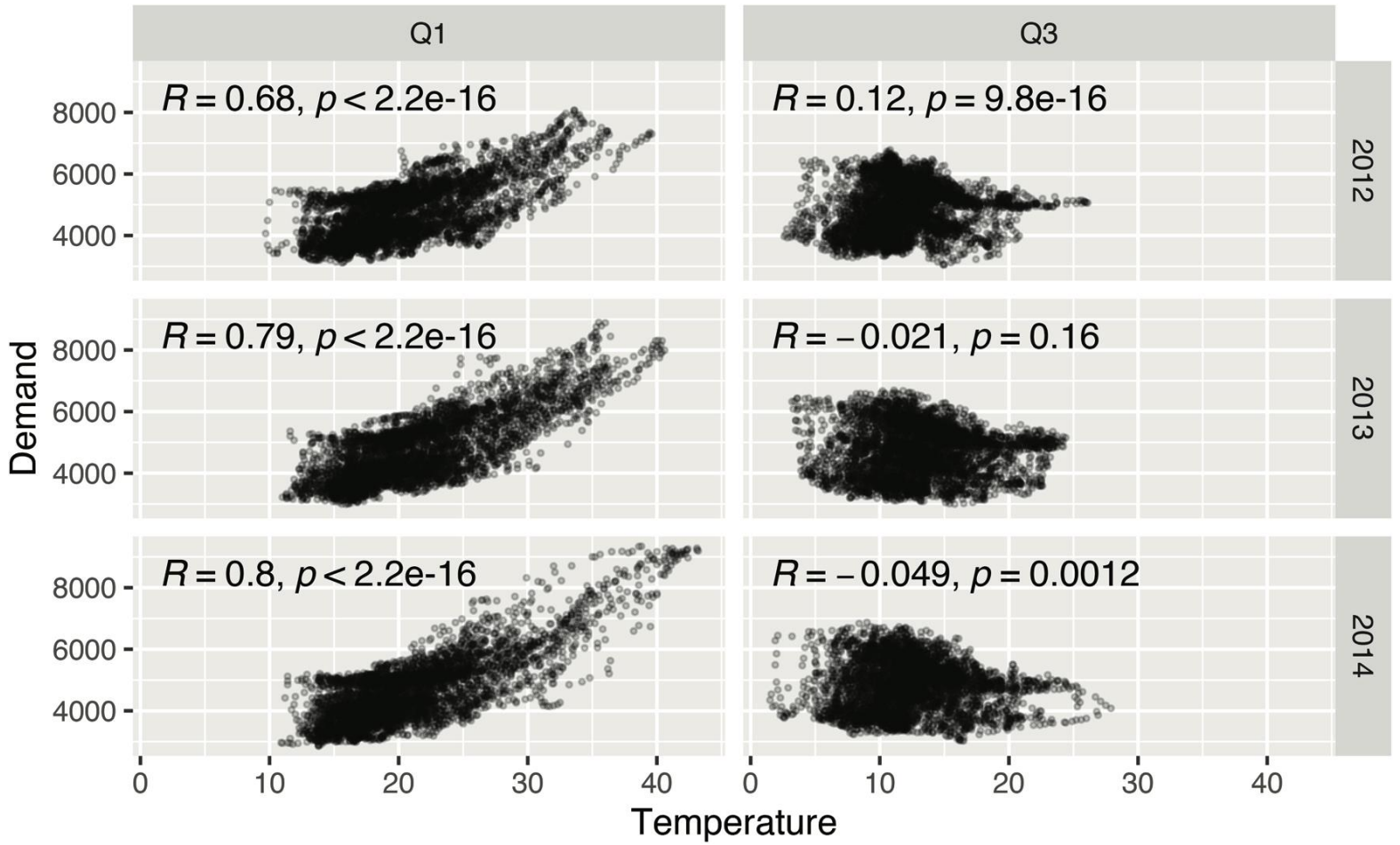
```
p3 <- ggplot(aus_long,aes(x=Time,y=Value,color=Quarter))+
       geom_point(size=0.2)+
       facet_wrap(~Indicator,scales="free",ncol = 1)+
       theme(legend.position = "top")
p3
```



Correlation coefficient: Temperature, Demand

```
aus_cor <- aus_elec %>%
            dplyr::group_by(Year, Quarter) %>%
            dplyr::summarize(CorrCoeff=cor(Temperature, Demand)) %>%
            dplyr::ungroup() %>%
            dplyr::arrange(desc(CorrCoeff))
#> `summarize()` has grouped output by 'Year'. You can override
#> using the `.groups` argument.
dplyr::slice(aus_cor,1:4)
#> # A tibble: 4 x 3
#>    Year Quarter CorrCoeff
#>   <dbl> <chr>       <dbl>
#> 1  2014 Q1          0.796
#> 2  2013 Q1          0.786
#> 3  2012 Q1          0.683
#> 4  2012 Q4          0.476
```

```
q1_out <- filter(aus_elec, Quarter %in% c("Q1","Q3"))
p4 <- ggplot(q1_out,aes(x=Temperature,y=Demand))+
       geom_point(alpha=0.2,size=0.5)+
       facet_grid(Year~Quarter)+
       stat_cor(digits = 2)
p4
```



Maximum demand information: per year, quarter

```
aus_summ <- aus_elec %>%
             dplyr::group_by(Year,Quarter) %>%
             dplyr::summarize(MaxD=max(Demand),
                              MaxIndex=which.max(Demand),
                              Time=nth(Time,MaxIndex),
                              Temp=nth(Temperature,MaxIndex),
                              Day=nth(WDay,MaxIndex),
                              DaySegment=nth(DaySegment,MaxIndex)) %>%
             dplyr::ungroup() %>%
             dplyr::arrange(desc(MaxD))
head(aus_summ)
#> # A tibble: 6 x 8
#>    Year Quarter  MaxD MaxIndex Time                Temp Day
#>   <dbl> <chr>   <dbl>    <int> <dttm>              <dbl> <ord>
#> 1  2014 Q1      9345.      755 2014-01-16 17:00:00  38.8 Thu
#> 2  2013 Q1      8897.     3395 2013-03-12 17:00:00  35.5 Tue
#> 3  2012 Q4      8443.     2865 2012-11-29 17:00:00  38.7 Thu
#> 4  2013 Q4      8156.     3824 2013-12-19 16:30:00  39   Thu
#> 5  2012 Q1      8072.     1138 2012-01-24 16:30:00  33.6 Tue
#> 6  2012 Q2      6921.     3926 2012-06-21 17:30:00   9.6 Thu
#> # ... with 1 more variable: DaySegment <chr>
```

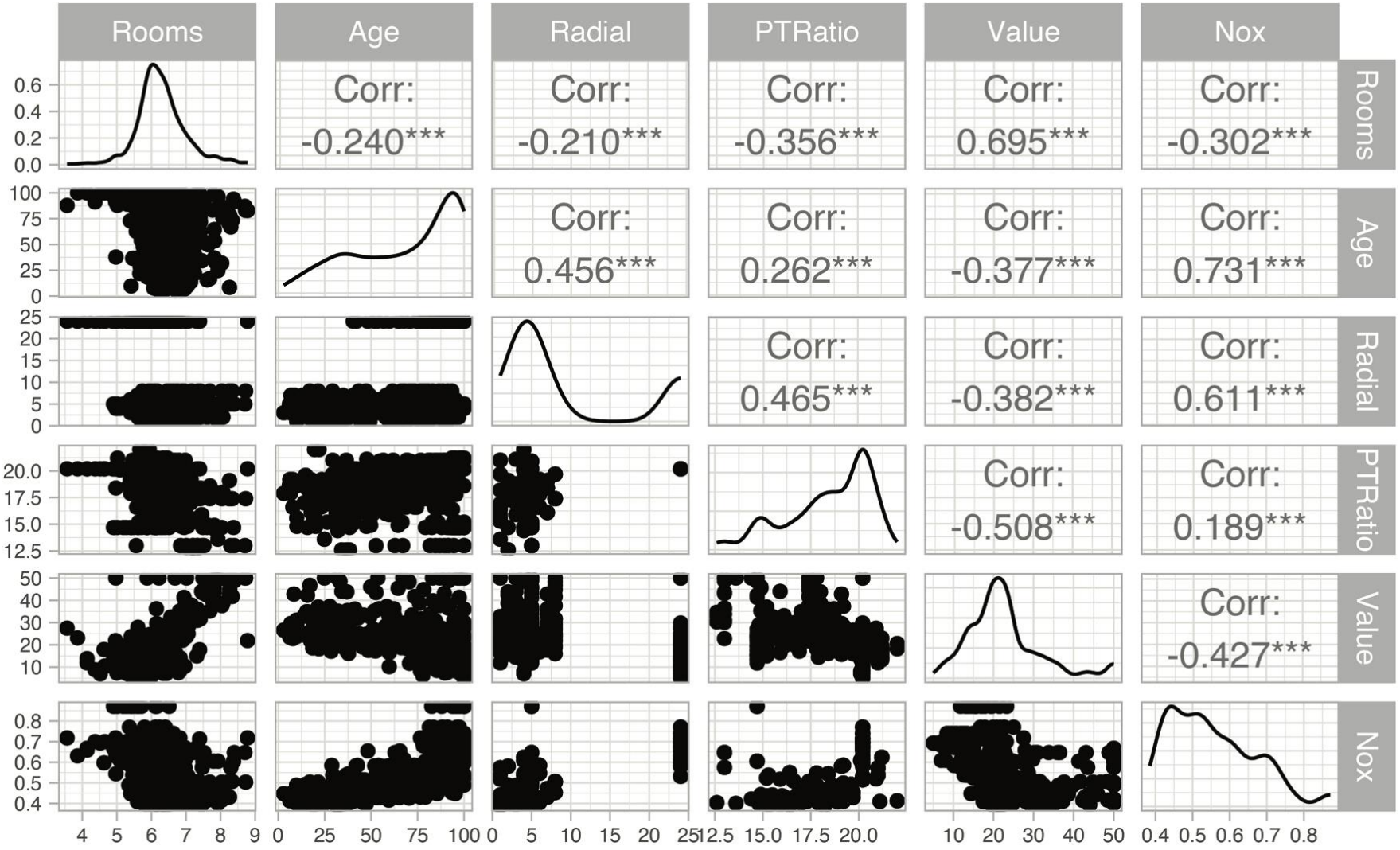
(3) Exploring Housing Values

Package MASS, Contains information on Boston suburb housing prices, and related variables, from the 1970s (Harrison Jr and Rubinfeld, 1978), including:

- **chas**, a variable that indicates whether the area bounds the Charles River (1 = bounds, 0 otherwise).
- **rm**, the average number of rooms per dwelling
- **age**, the proportion of owner-occupied units built prior to 1940
- **rad**, index of accessibility to radial highways.
- **ptratio**, pupil–teacher ratio by suburb. 280
- **medv**, the median value of owner-occupied homes (thousands of dollars).
- **nox**, the nitrogen oxide concentration (in parts per million).

```
bos <- Boston %>%
        dplyr::as_tibble() %>%
        dplyr::select(chas,rm,age,rad,ptratio,medv,nox) %>%
        dplyr::rename(PTRatio=ptratio,
                      ByRiver=chas,
                      Rooms=rm,
                      Age=age,
                      Radial=rad,
                      Value=medv,
                      Nox=nox) %>%
        dplyr::mutate(ByRiver=as.logical(ByRiver))
bos
#> # A tibble: 506 x 7
#>    ByRiver Rooms   Age Radial PTRatio Value   Nox
#>    <lgl>   <dbl> <dbl>  <int>   <dbl> <dbl> <dbl>
#>  1 FALSE    6.58  65.2      1    15.3  24   0.538
#>  2 FALSE    6.42  78.9      2    17.8  21.6 0.469
#>  3 FALSE    7.18  61.1      2    17.8  34.7 0.469
#>  4 FALSE    7.00  45.8      3    18.7  33.4 0.458
#>  5 FALSE    7.15  54.2      3    18.7  36.2 0.458
#>  6 FALSE    6.43  58.7      3    18.7  28.7 0.458
#>  7 FALSE    6.01  66.6      5    15.2  22.9 0.524
#>  8 FALSE    6.17  96.1      5    15.2  27.1 0.524
#>  9 FALSE    5.63 100        5    15.2  16.5 0.524
#> 10 FALSE    6.00  85.9      5    15.2  18.9 0.524
#> # ... with 496 more rows
```

Pair-wise associations



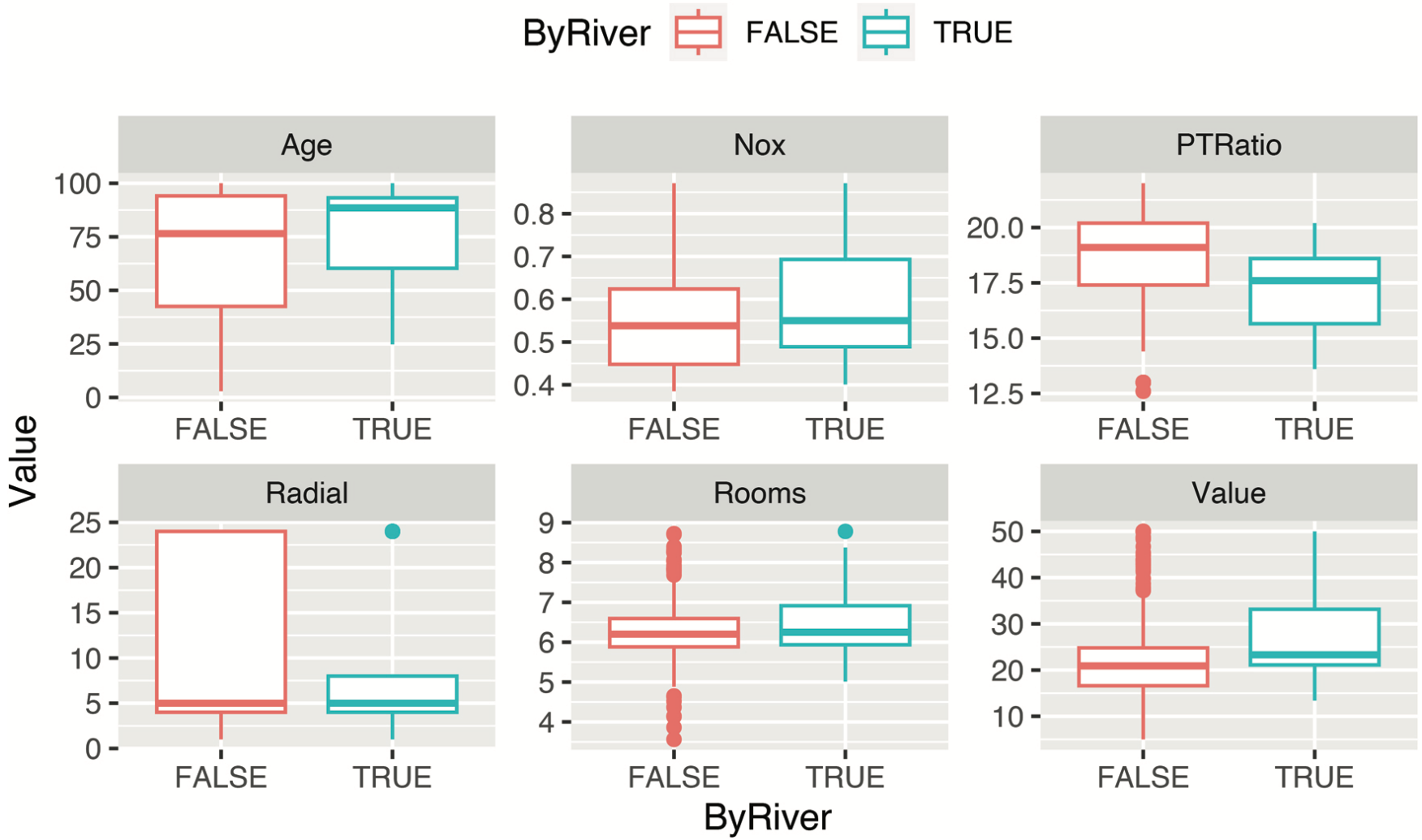
Exploring the correlation data…

```
cor(dplyr::select(bos,-ByRiver))
#>             Rooms     Age  Radial PTRatio   Value     Nox
#> Rooms     1.0000 -0.2403 -0.2098 -0.3555  0.6954 -0.3022
#> Age      -0.2403  1.0000  0.4560  0.2615 -0.3770  0.7315
#> Radial   -0.2098  0.4560  1.0000  0.4647 -0.3816  0.6114
#> PTRatio  -0.3555  0.2615  0.4647  1.0000 -0.5078  0.1889
#> Value     0.6954 -0.3770 -0.3816 -0.5078  1.0000 -0.4273
#> Nox      -0.3022  0.7315  0.6114  0.1889 -0.4273  1.0000
```

Transforming the data…

```
bos_long <- bos %>%
              tidyr::pivot_longer(names_to = "Indicator",
                                  values_to = "Value",
                                  -ByRiver)
bos_long
#> # A tibble: 3,036 x 3
#>    ByRiver Indicator  Value
#>    <lgl>   <chr>      <dbl>
#>  1 FALSE   Rooms      6.58
#>  2 FALSE   Age       65.2
#>  3 FALSE   Radial     1
#>  4 FALSE   PTRatio   15.3
#>  5 FALSE   Value     24
#>  6 FALSE   Nox        0.538
#>  7 FALSE   Rooms      6.42
#>  8 FALSE   Age       78.9
#>  9 FALSE   Radial     2
#> 10 FALSE   PTRatio   17.8
#> # ... with 3,026 more rows
```

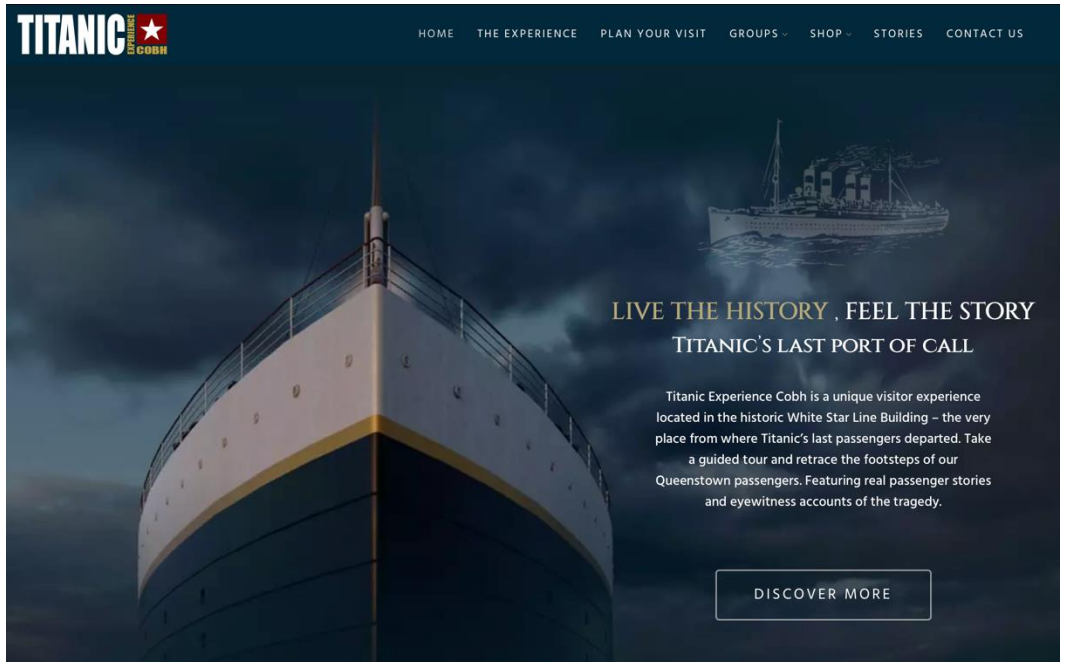
```
p5 <- ggplot(bos_long,aes(x=ByRiver,y=Value,color=ByRiver))+
        geom_boxplot()+
        facet_wrap(~Indicator,scales="free")+
        theme(legend.position = "top")
p5
```



(4) Passenger survival on the Titanic

- The Titanic was built in Belfast, and departed on its first voyage from Southampton on April 10, 1912, stopping in Cherbourg and Cobh, before setting sail across the Atlantic Ocean on route to New York.
- On April 15, she struck an iceberg off the coast of Canada, and sank to the ocean floor.
- Over 1,500 passengers and crew perished, with only about 700 surviving.
- Electronic versions of the ship's passenger list are now available, and have been used as case studies in order to explore survival outcomes of passengers

https://www.titanicexperiencecobh.ie



```
library(ggplot2)
library(dplyr)
library(titanic)
```

```
dplyr::glimpse(titanic_train)
#> Rows: 891
#> Columns: 12
#> $ PassengerId <int> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, ~
#> $ Survived    <int> 0, 1, 1, 1, 0, 0, 0, 0, 1, 1, 1, 1, 0, 0, 0~
#> $ Pclass      <int> 3, 1, 3, 1, 3, 3, 1, 3, 3, 2, 3, 1, 3, 3, 3~
#> $ Name        <chr> "Braund, Mr. Owen Harris", "Cumings, Mrs. J~
#> $ Sex         <chr> "male", "female", "female", "female", "male~
#> $ Age         <dbl> 22, 38, 26, 35, 35, NA, 54, 2, 27, 14, 4, 5~
#> $ SibSp       <int> 1, 1, 0, 1, 0, 0, 0, 3, 0, 1, 1, 0, 0, 1, 0~
#> $ Parch       <int> 0, 0, 0, 0, 0, 0, 0, 1, 2, 0, 1, 0, 0, 5, 0~
#> $ Ticket      <chr> "A/5 21171", "PC 17599", "STON/O2. 3101282"~
#> $ Fare        <dbl> 7.250, 71.283, 7.925, 53.100, 8.050, 8.458,~
#> $ Cabin       <chr> "", "C85", "", "C123", "", "", "E46", "", "~
#> $ Embarked    <chr> "S", "C", "S", "S", "S", "Q", "S", "S", "S"~
```

```r
titanic <- titanic_train %>%
            dplyr::select(PassengerId,
                          Survived,
                          Pclass,
                          Sex,
                          Age) %>%
            dplyr::mutate(Survived=as.logical(Survived),
                          Sex=factor(Sex,
                                     levels=c("male","female")),
                          Class=factor(Pclass,
                                       levels=c(1,2,3))) %>%
            dplyr::select(-Pclass) %>%
            dplyr::as_tibble()
summary(titanic)
#>    PassengerId    Survived           Sex           Age
#>  Min.   :  1   Mode :logical   male  :577   Min.   : 0.42
#>  1st Qu.:224   FALSE:549       female:314   1st Qu.:20.12
#>  Median :446   TRUE :342                    Median :28.00
#>  Mean   :446                                Mean   :29.70
#>  3rd Qu.:668                                3rd Qu.:38.00
#>  Max.   :891                                Max.   :80.00
#>                                             NA's   :177
#>  Class
#>  1:216
#>  2:184
#>  3:491
```

Overall summaries of the data.

```
sum1 <- titanic %>%
         dplyr::summarize(N=n(),
                          TSurvived=sum(Survived),
                          TPerished=sum(Survived==FALSE),
                          PropSurvived=TSurvived/N,
                          PropPerished=TPerished/N)
sum1
#> # A tibble: 1 x 5
#>       N TSurvived TPerished PropSurvived PropPerished
#>   <int>     <int>     <int>        <dbl>         <dbl>
#> 1   891       342       549        0.384         0.616
```

Exploring differences: male/female

```
sum2 <- titanic %>%
         dplyr::group_by(Sex) %>%
         dplyr::summarize(N=n(),
                          TSurvived=sum(Survived),
                          TPerished=sum(Survived==FALSE),
                          PropSurvived=TSurvived/N,
                          PropPerished=TPerished/N) %>%
                          arrange(desc(PropSurvived))
sum2
#> # A tibble: 2 x 6
#>   Sex        N TSurvived TPerished PropSurvived PropPerished
#>   <fct>  <int>     <int>     <int>        <dbl>         <dbl>
#> 1 female   314       233        81        0.742         0.258
#> 2 male     577       109       468        0.189         0.811
```

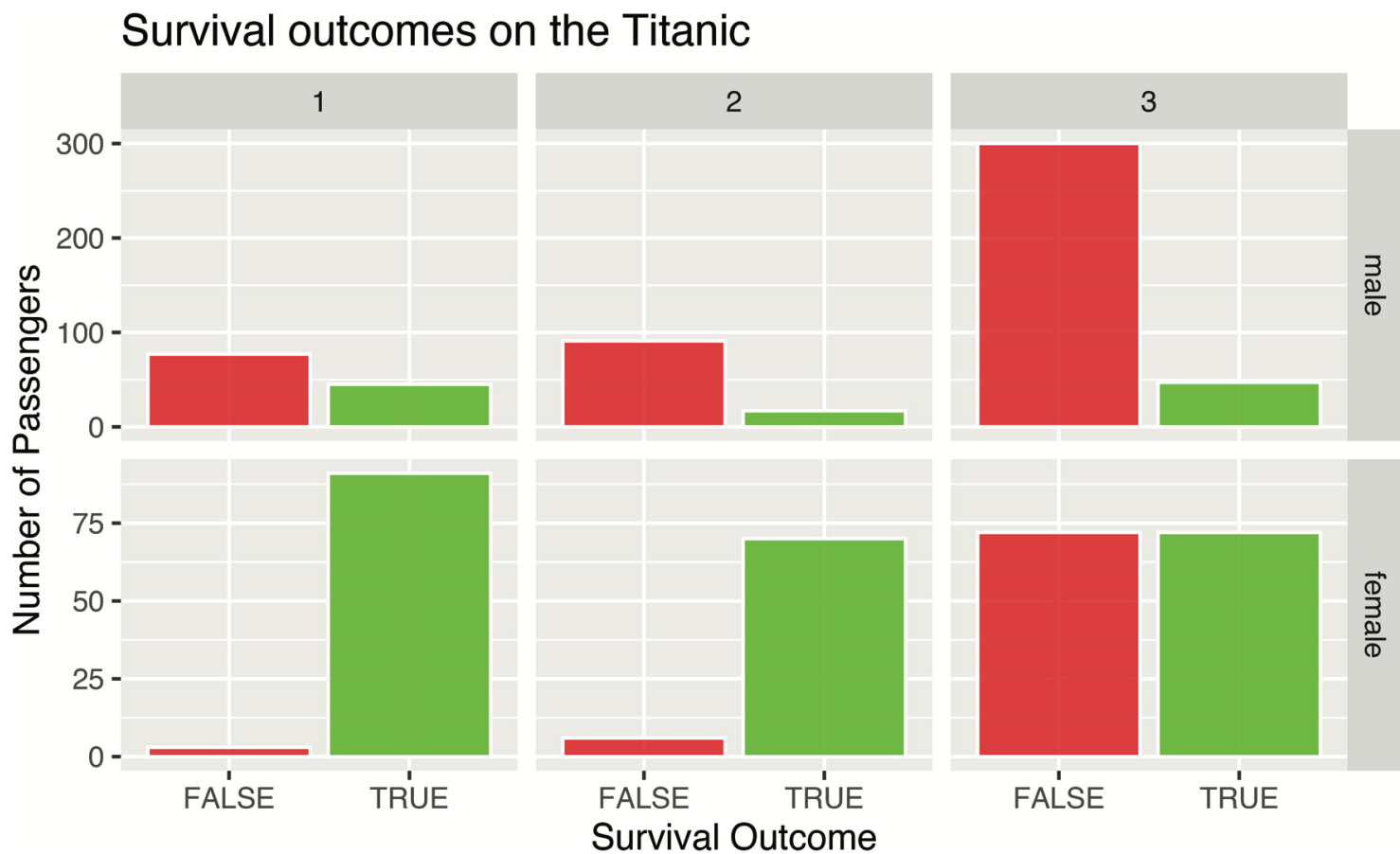
Exploring differences: 1st, 2nd, and 3rd class

```
sum3 <- titanic %>%
         dplyr::group_by(Class) %>%
         dplyr::summarize(N=n(),
                          TSurvived=sum(Survived),
                          TPerished=sum(Survived==FALSE),
                          PropSurvived=TSurvived/N,
                          PropPerished=TPerished/N) %>%
         dplyr::arrange(desc(PropSurvived))
sum3
#> # A tibble: 3 x 6
#>   Class     N TSurvived TPerished PropSurvived PropPerished
#>   <fct> <int>     <int>     <int>        <dbl>         <dbl>
#> 1 1       216       136        80        0.630         0.370
#> 2 2       184        87        97        0.473         0.527
#> 3 3       491       119       372        0.242         0.758
```

Differences by two groupings

```
sum4 <- titanic %>%
         dplyr::group_by(Class,Sex) %>%
         dplyr::summarize(N=n(),
                          TSurvived=sum(Survived),
                          TPerished=sum(Survived==FALSE),
                          PropSurvived=TSurvived/N,
                          PropPerished=TPerished/N) %>%
         dplyr::arrange(desc(PropSurvived))
sum4
#> # A tibble: 6 x 7
#> # Groups:   Class [3]
#>   Class Sex        N TSurvived TPerished PropSurvived PropPeris~1
#>   <fct> <fct>  <int>     <int>     <int>        <dbl>       <dbl>
#> 1 1     female    94        91         3        0.968      0.0319
#> 2 2     female    76        70         6        0.921      0.0789
#> 3 3     female   144        72        72        0.5        0.5
#> 4 1     male     122        45        77        0.369      0.631
#> 5 2     male     108        17        91        0.157      0.843
#> 6 3     male     347        47       300        0.135      0.865
#> # ... with abbreviated variable name 1: PropPerished
```

```r
p6 <- ggplot(titanic,aes(x=Survived,fill=Survived))+
      geom_bar(color="white",alpha=0.75)+
      facet_grid(Sex~Class,scales="free")+
      theme(legend.position = "none")+
      labs(title="Survival outcomes on the Titanic",
           x="Survival Outcome",
           y="Number of Passengers")+
      scale_fill_manual(values=c("red","green"))
p6
```



(5) Wind direction and winter temperatures

- Our final example focuses on weather data, based on the tibble observations that is part of the aimsir17 CRAN data package.
- Our question is to explore whether, in winter, wind from a southerly direction is more associated with higher temperatures.

```
library(dplyr)
library(ggplot2)
library(aimsir17)
```

```
st4 <- c("MALIN HEAD",
         "DUBLIN AIRPORT",
         "ROCHES POINT",
         "MACE HEAD")
st4
#> [1] "MALIN HEAD"     "DUBLIN AIRPORT" "ROCHES POINT"
#> [4] "MACE HEAD"
```

Filter 4 stations and modify wdsp to kmh

```
# Filter data, convert to factor, and update average hourly wind speed
# from knots to kmh
eda0 <- observations %>%
         dplyr::filter(station %in% st4) %>%
         dplyr::mutate(station=factor(station),
                       wdsp=wdsp*1.852) %>%
         dplyr::select(-year)
head(eda0)
#> # A tibble: 6 x 11
#>   station month   day  hour date                rain  temp  rhum
#>   <fct>   <dbl> <int> <int> <dttm>              <dbl> <dbl> <dbl>
#> 1 DUBLIN~     1     1     0 2017-01-01 00:00:00   0.9   5.3    91
#> 2 DUBLIN~     1     1     1 2017-01-01 01:00:00   0.2   4.9    95
#> 3 DUBLIN~     1     1     2 2017-01-01 02:00:00   0.1   5      92
#> 4 DUBLIN~     1     1     3 2017-01-01 03:00:00   0     4.2    90
#> 5 DUBLIN~     1     1     4 2017-01-01 04:00:00   0     3.6    88
#> 6 DUBLIN~     1     1     5 2017-01-01 05:00:00   0     2.8    89
#> # ... with 3 more variables: msl <dbl>, wdsp <dbl>, wddir <dbl>
```

Add wind direction as NESW

```
eda <- eda0 %>%
  dplyr::mutate(wind_dir = case_when(
                    wddir > 315    | wddir <= 45   ~ "N",
                    wddir > 45     & wddir <= 135  ~ "E",
                    wddir > 135    & wddir <= 225  ~ "S",
                    wddir > 225    & wddir <= 315  ~ "W",
                    TRUE ~ "Missing"),
                  wind_dir = ifelse(wind_dir=="Missing",
                         NA, wind_dir),
                  wind_dir= factor(wind_dir,
                                   levels=c("N","E","S","W"))) %>%
  dplyr::select(station:date,
                  wdsp,
                  wind_dir,
                  wddir,
                  everything())
```

Show the resulting tibble

```
dplyr::slice(eda,1:3)
#> # A tibble: 3 x 12
#>   station     month   day  hour date                 wdsp wind_~1
#>   <fct>       <dbl> <int> <int> <dttm>               <dbl> <fct>
#> 1 DUBLIN AIR~     1     1     0 2017-01-01 00:00:00  22.2 N
#> 2 DUBLIN AIR~     1     1     1 2017-01-01 01:00:00  14.8 W
#> 3 DUBLIN AIR~     1     1     2 2017-01-01 02:00:00  14.8 W
#> # ... with 5 more variables: wddir <dbl>, rain <dbl>,
#> #   temp <dbl>, rhum <dbl>, msl <dbl>, and abbreviated variable
#> #   name 1: wind_dir
```

Filter by winter month (10, 11, and 1)

```
winter <- eda %>%
            dplyr::filter(month %in% c(11,12,1))
dplyr::glimpse(winter)
#> Rows: 8,832
#> Columns: 12
#> $ station  <fct> DUBLIN AIRPORT, DUBLIN AIRPORT, DUBLIN AIRPORT~
#> $ month    <dbl> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
#> $ day      <int> 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1~
#> $ hour     <int> 0, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, ~
#> $ date     <dttm> 2017-01-01 00:00:00, 2017-01-01 01:00:00, 201~
#> $ wdsp     <dbl> 22.22, 14.82, 14.82, 22.22, 20.37, 22.22, 24.0~
#> $ wind_dir <fct> N, W, W, N, N, N, N, N, N, N, N, N, N, N, N, N~
#> $ wddir    <dbl> 340, 310, 310, 330, 330, 330, 330, 330, 330, 3~
#> $ rain     <dbl> 0.9, 0.2, 0.1, 0.0, 0.0, 0.0, 0.0, 0.0, 0.0, 0~
#> $ temp     <dbl> 5.3, 4.9, 5.0, 4.2, 3.6, 2.8, 1.7, 1.6, 2.0, 2~
#> $ rhum     <dbl> 91, 95, 92, 90, 88, 89, 91, 91, 89, 84, 84, 80~
#> $ msl      <dbl> 1020, 1020, 1020, 1020, 1020, 1020, 1020, 1021~
```

ggplot() call

```
p7 <- ggplot(winter,aes(x=wind_dir,y=temp,color=station))+
        geom_boxplot()+
        facet_wrap(~station,nrow = 1)+
        labs(y="Temperature",
             x="Wind Direction",
             title="Winter temperatures at weather stations",
             subtitle="Data summarized by wind direction")+
        theme(legend.position = "none")
p7
```

Visualisation