VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

PROBABILITY AND STATISTICS (MT2013)

Assignment

Advisor: Nguyễn Tiến Dũng

Student(s): Lê Bá Thành - 1852739
Nguyễn Xuân Thành - 2152285
Nguyễn Thân Kiên - 2053160
Nguyễn Thanh Ngân - 2053255
Lê Gia Khánh - 2152120

Ho Chi Minh City, 4/2023

Contents

1 Introduction **3**
1.1 Content of data . . . 3
1.2 Objective . . . 3
1.3 Member list & Workload . . . 3

2 Data description **3**
2.1 Variables . . . 3
2.2 How Global Solar Irradiance (GHI) is measured . . . 4
2.3 Unused columns in our research . . . 5

3 Import necessary library and data **5**

4 Data cleaning **6**

5 Data visualization **7**
5.1 Helper functions . . . 7
5.1.1 Purpose . . . 7
5.1.2 Summarise and plot changes of one variable by stratifying . . . 7
5.1.3 Summarise and plot changes of one variable in time of day (hour and minute) 7
5.1.4 Summarise and plot one variable changes in one month . . . 8
5.1.5 Examine the distribution of one variable . . . 8
5.2 Difference in how data are recorded . . . 8
5.3 Descriptive graph for several factors . . . 9
5.3.1 Energy production . . . 9
5.3.2 GHI . . . 11
5.3.3 Temperature . . . 13
5.3.4 Pressure . . . 15
5.3.5 Humidity . . . 15
5.3.6 Wind speed . . . 18
5.3.7 Rain . . . 18
5.3.8 Snow . . . 19
5.3.9 Cloud . . . 20
5.3.10 Length of day . . . 21

6 Normality test for temperature and pressure **23**
6.1 Normal distribution and normality test . . . 23
6.2 Temperature . . . 24
6.3 Pressure . . . 26

7 Linear model to predict global horizontal irradiance (GHI) based on weather parameters **27**
7.1 Purpose . . . 27
7.2 Data selection . . . 27
7.3 Correlation between two variables . . . 28
7.4 Linear regression model . . . 29
7.5 Hypothesis Tests In Multivariate Linear Regression . . . 31
7.5.1 Test for significance of regression . . . 31
7.5.2 t-statistic, Pr(>|t|) and Signif. codes . . . 31
7.5.3 Multiple R-squared and adjusted R-squared . . . 32
7.5.4 Interpreting our result . . . 32
7.6 Rebuild the model . . . 33
7.7 Test for the normality distribution of errors . . . 34
7.7.1 Residuals vs Fitted plot . . . 34

7.7.2 Normal Quantile-Quantile plot 35
7.7.3 The histogram of residuals 36
7.7.4 One-sample Kolmogorov-Smirnov normality test 37

8 Three renewable energy solutions **39**
8.1 Solar energy 39
8.2 Hydro electricity 39
8.3 Wind energy 42

1 Introduction

1.1 Content of data

The data is contributed by a user on **Kaggle** whose username is *AI Maverick*. The URL for the data is https://www.kaggle.com/datasets/samanemami/renewable-energy-and-weather-conditions. This comprehensive data set provides an in-depth look at the complex relationship between weather conditions and renewable energy generation, solar in particular. By measuring key weather parameters such as solar radiation, temperature, humidity, and precipitation hourly, the data set allows our team to explore the impact of weather on energy production in the area.

1.2 Objective

1. Study the distribution of several weather indicators and how they change over time.
2. Predict energy output following the effects of weather patterns.
3. Give practical conclusion on how and why three potential renewable energy solutions: solar, hydro electricity and wind can or cannot be implemented in the area.

Through this project, we learn how to visualize the distribution of the energy produced and weather parameters versus time by utilizing R functionality. We also obtain knowledge on how to construct a multivariate linear model and conduct hypothesis tests.

1.3 Member list & Workload

No.	Fullname	Student ID	Tasks	Percentage
1	Lê Bá Thành	1852739	Build linear regression model	15%
2	Nguyễn Xuân Thành	2152285	Select, analyse and visualise data Code and interpret result Editor	35%
3	Nguyễn Thân Kiên	2053160	Hypothesis tests in multivariate linear regression	15%
4	Nguyễn Thanh Ngân	2053255	Data description Conclusion on data effect	15%
5	Lê Gia Khánh	2152120	Select data Normality test	20%

2 Data description

2.1 Variables

The "solar_weather.csv" file contains 196776 observations over the course of 68 months from 01/01/2017 to 31/08/2022. There are 17 columns in the file, including:

1. Time: The timestamp of the recorded data in the format of YYYY-MM-DD HH:MM:SS.
2. Energy delta (Wh): The difference in the amount of electricity generated by a solar panel from the previous timestamp to the current timestamp.
3. GHI (W / m^2): Global Horizontal Irradiance measured by a pyranometer, which is the amount of solar radiation received by a horizontal surface.
4. temp (°C): The temperature in degrees Celsius measured at the same height as the pyranometer.

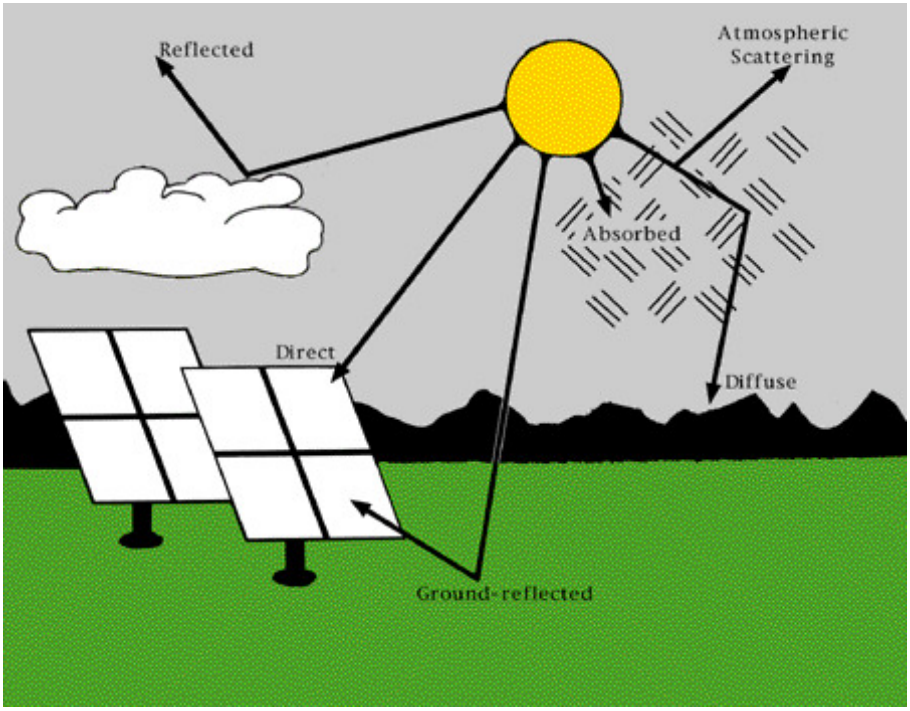
5. pressure (hPa): The atmospheric pressure in hectopascals measured at the same height as the pyranometer.
6. humidity (%): The relative humidity in percentage measured at the same height as the pyranometer.
7. wind_speed (m/s): The wind speed in meters per second measured at the same height as the pyranometer.
8. rain_1h (mm): The amount of precipitation in millimeters measured over the past hour.
9. snow_1h (mm): The amount of snowfall in millimeters.
10. clouds_all: The cloud situation where 0 indicates cloudless condition and 100 for cloudy.
11. isSun: Binary value that indicates the presence of sunlight.
12. sunlightTime: The amount of time during which sunlight is available.
13. dayLength: The difference in minutes between sunrise and sunset in a day.
14. SunlightTime/daylength: The quotient of two columns "sunlightTime" and "dayLength".
15. weather_type: Interger from 1 to 5 which provides information on the overall weather conditions such as clear, cloudy, or rainy.
16. hour
17. month

The dataset is organized by hour and month, making it ideal for studying the relationship between renewable energy generation and weather patterns over time.

2.2 How Global Solar Irradiance (GHI) is measured

Global Horizontal Irradiance (GHI) is the total solar radiation incident on a horizontal surface. The radiation is strongest when the sun is directly overhead at a 'solar zenith angle' (θ) and the thickness of the atmosphere is at its minimum.
As the sun moves down, the Direct Normal Irradiance (DNI) beam strikes the Earth's surface obliquely, and spreads out, reducing the amount of energy per unit area as a cosine function.
When passing through the atmosphere, the solar radiation is scattered, reflected, and absorbed by air molecules, aerosol particles, water droplets and ice crystals in clouds. This produces diffuse solar radiation.
Global Horizontal Irradiance (GHI), from the hemisphere above a horizontal plane surface, is a combination of DNI, corrected for the angle of incidence of the beam (θ), and Diffuse Horizontal Irradiance (DHI).

$$GHI = DNI * cos(\theta) + DHI \ (W/m^2)$$



Pyranometers are defined by ISO 9060:1990 as the instruments for the measurement of hemispherical (global) solar radiation for solar energy.
To conclude, GHI is significantly affected by several weather parameters such as humidity, wind, rain and cloud.

2.3 Unused columns in our research

- sunlightTime: This variable is redundant because we can retrieve the same amount of information by combining two columns "Time" and "isSun".
- SunlightTime/daylength
- weather_type: The source where we download the data doesn't provide labels / descriptions for each weather type. Therefore, we decided to drop the column to avoid giving vague results.

3 Import necessary library and data

```
library(tidyverse)
options(dplyr.summarise.inform = FALSE)
library(ggrepel) # to add label on plot
library(corrplot) # to plot correlation matrix
library(RColorBrewer) # add more options to graph color
library(broom) # to convert linear model from "lm" to tidy form
solar_weather <- as_tibble(read.csv("solar_weather.csv", header = TRUE))
head(solar_weather)
```

```
## # A tibble: 6 x 17
##   Time       Energ~1   GHI  temp press~2 humid~3 wind_~4 rain_1h snow_1h cloud~5
##   <chr>        <int> <dbl> <dbl>   <int>   <int>   <dbl>   <dbl>   <dbl>   <int>
## 1 1/1/2017 ~       0     0   1.6    1021     100     4.9       0       0     100
## 2 1/1/2017 ~       0     0   1.6    1021     100     4.9       0       0     100
```

```
## 3 1/1/2017 ~       0     0   1.6    1021     100     4.9       0       0     100
## 4 1/1/2017 ~       0     0   1.6    1021     100     4.9       0       0     100
## 5 1/1/2017 ~       0     0   1.7    1020     100     5.2       0       0     100
## 6 1/1/2017 ~       0     0   1.7    1020     100     5.2       0       0     100
## # ... with 7 more variables: isSun <int>, sunlightTime <int>, dayLength <int>,
## #   SunlightTime.daylength <dbl>, weather_type <int>, hour <int>, month <int>,
## #   and abbreviated variable names 1: Energy.delta.Wh., 2: pressure,
## #   3: humidity, 4: wind_speed, 5: clouds_all
```

4 Data cleaning

- Identify missing data

```
apply(is.na(solar_weather), 2, sum)
```

```
##                   Time       Energy.delta.Wh.                    GHI
##                      0                      0                      0
##                   temp               pressure               humidity
##                      0                      0                      0
##             wind_speed                 rain_1h                snow_1h
##                      0                      0                      0
##             clouds_all                  isSun            sunlightTime
##                      0                      0                      0
##              dayLength SunlightTime.daylength           weather_type
##                      0                      0                      0
##                   hour                  month
##                      0                      0
```

No missing data.

In case of missing data, we can treat these N/A values by two following ways:

1. Delete / Ignore observations where N/A values are found.
2. Replace N/A values by the mean / median / mode value corresponding with each variable.

- Rename and add new columns to aid computation
- Convert the time of observation from character type to datetime
- Rename several variables
- Extract year, day and minute to separate columns
- Drop columns which we don't consider

```
solar_weather <- solar_weather %>%
  rename(
    "E" = "Energy.delta.Wh.",
    "rain" = "rain_1h",
    "snow" = "snow_1h",
    "cloud" = "clouds_all"
  ) %>%
  mutate(
    Time = mdy_hm(Time),
    year = year(Time),
    day = day(Time),
    minute = minute(Time)
  ) %>%
  select(-c(sunlightTime, SunlightTime.daylength, weather_type))
```

5 Data visualization

5.1 Helper functions

5.1.1 Purpose

- Enhance the robustness of our code
- Minimize the time for bug fixing and typing
- Provide an elegant interface for people adopting our research

5.1.2 Summarise and plot changes of one variable by stratifying

This function selects two variables x and y from the given data and plots the change of y with respect to x.

```
strat_plot = function(dat, x, y, func, ...) {
  dat %>%
    relocate(any_of(x), any_of(y)) %>%
    rename("x" = 1, "y" = 2) %>%
    group_by(x) %>%
    summarise(
      y = func(y, ...)
    ) %>%
    ggplot(aes(x, y)) +
    xlab(x) +
    ylab(y)
}
```

5.1.3 Summarise and plot changes of one variable in time of day (hour and minute)

This function group the observations by the time of day in which it was recorded, calculate and plot the statistic of the variable in interest.

```
change_tod = function(dat, var, func, ...) {
  dat %>%
    relocate(any_of(var), "hour", "minute") %>%
    rename("var" = 1) %>%
    group_by(hour, minute) %>%
    summarise(
      var = func(var, ...)
    ) %>%
    mutate(
      time = str_c(hour, ":", minute)
    ) %>%
    mutate(
      time = factor(time, levels = time)
    ) %>%
    ggplot(aes(time, var)) +
    geom_point(aes(group = 1)) +
    theme(axis.text.x = element_blank()) +
    ylab(var)
}
```

5.1.4 Summarise and plot one variable changes in one month

This function is quite similar to the above, except that it groups observations by year and month.

```
change_month = function(dat, var, func, ...) {
  dat %>%
    relocate(any_of(var), "year", "month") %>%
    rename("var" = 1) %>%
    group_by(year, month) %>%
    summarise(
      var = func(var, ...)
    ) %>%
    mutate(
      time = str_c(month, "/", str_sub(year, 3))
    ) %>%
    mutate(
      time = factor(time, levels = time)
    ) %>%
    ggplot(aes(time, var, group = 1)) +
    geom_line() +
    geom_point() +
    geom_text_repel(aes(label = time)) +
    theme(axis.text.x = element_blank()) +
    ylab(var)
}
```

5.1.5 Examine the distribution of one variable

The function plots a histogram to show how one variable is distributed.

```
distribution = function(dat, var, ...) {
  dat %>%
    relocate(any_of(var)) %>%
    rename("var" = 1) %>%
    ggplot(aes(var)) +
    geom_histogram(...) +
    xlab(var)
}
```

5.2 Difference in how data are recorded

In the original data, several variables, typically time, energy production and GHI, usually differ when we iterate over the observations, which were taken every 15 minutes. Others, including many weather indicators, stay the same within an hour.

As discussed in the data description, we know that there are variables that are measured in hour, not 15 minutes. Therefore, in order to avoid overestimating the values when we examine the distribution of some variables, we will detect and construct a separate data frame for variables measured in hour.

```
test <- function(x) {
  tmp <- solar_weather %>%
    relocate(any_of(x)) %>%
    rename("x" = 1) %>%
    select(x, year, month, day, hour) %>%
    distinct() %>%
```

```
    mutate(time = ymd_h(str_c(year, "-", month, "-", day, " ", hour))) %>%
    pull(time)
  any(duplicated(tmp))
}
sapply(names(solar_weather[2:12]), function(x) test(x))
```

```
##          E        GHI       temp   pressure   humidity wind_speed       rain
##       TRUE       TRUE      FALSE      FALSE      FALSE      FALSE      FALSE
##       snow      cloud      isSun  dayLength
##      FALSE      FALSE       TRUE      FALSE
```

The value TRUE indicates the variable is recorded every 15 minutes, and FALSE for 1 hour.

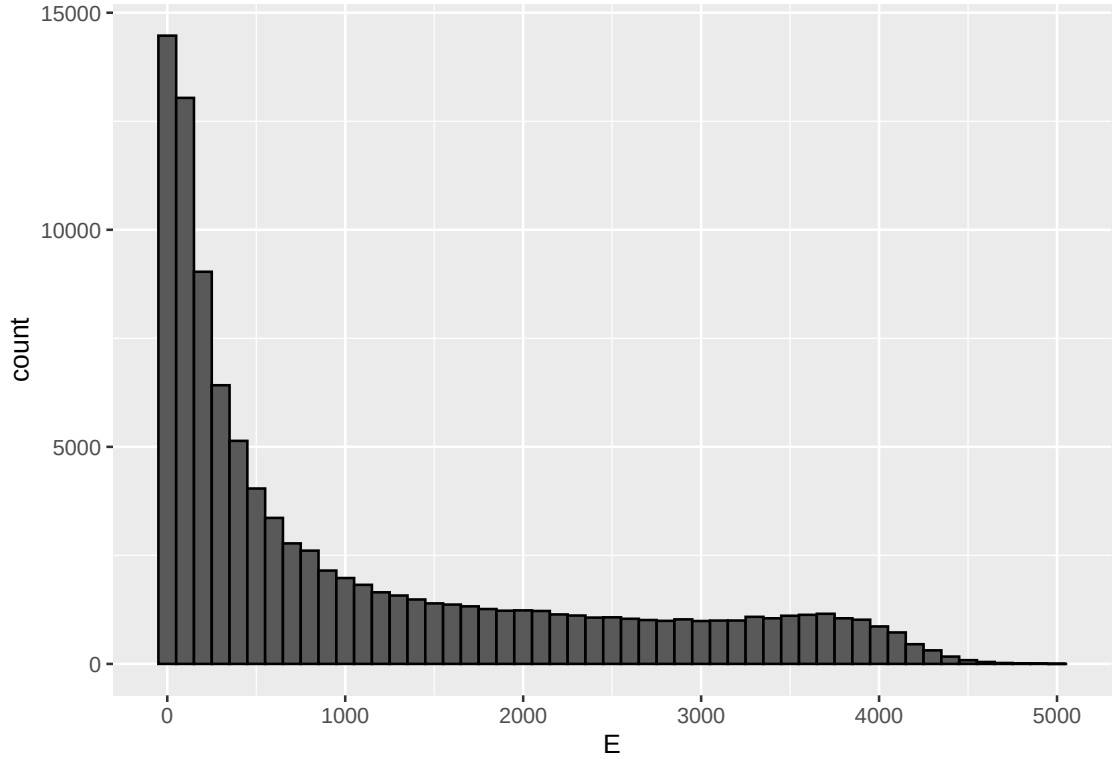
```
solar_weather_hour <- solar_weather %>%
  select(-c("E", "GHI", "isSun", "Time", "minute")) %>%
  distinct()
```

5.3 Descriptive graph for several factors

5.3.1 Energy production

- Distribution

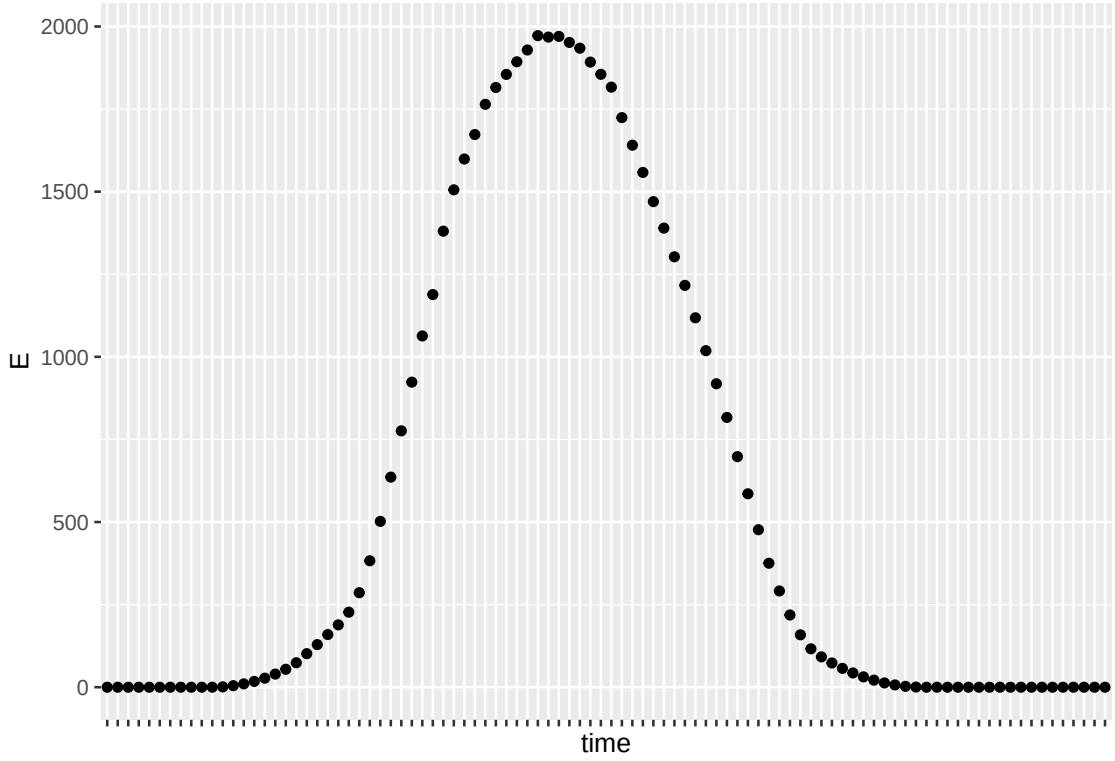
```
distribution(solar_weather %>% filter(isSun == 1), "E", binwidth = 100, col = "black")
```



Be aware that we only consider observations at which there is sun

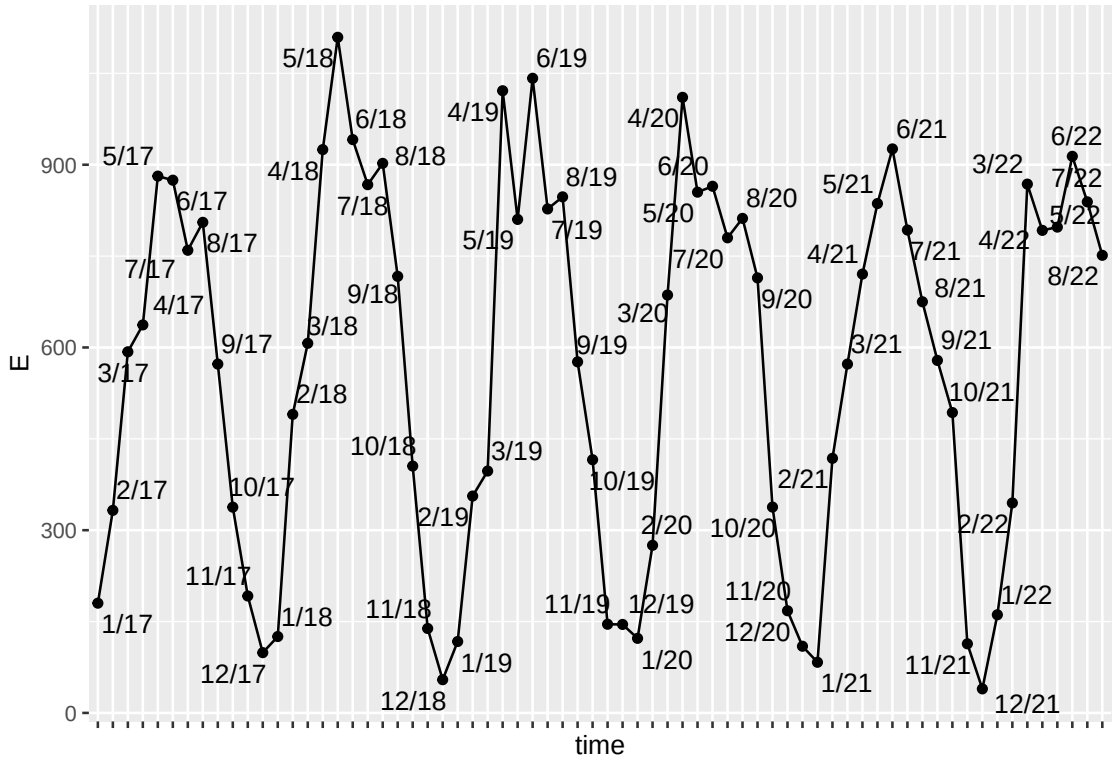
- Change in energy production throughout the day

```
change_tod(solar_weather, "E", mean)
```



- Mean energy production every month

```
change_month(solar_weather, "E", mean)
```



We can consider that the energy production is always high in three months such as April, May and

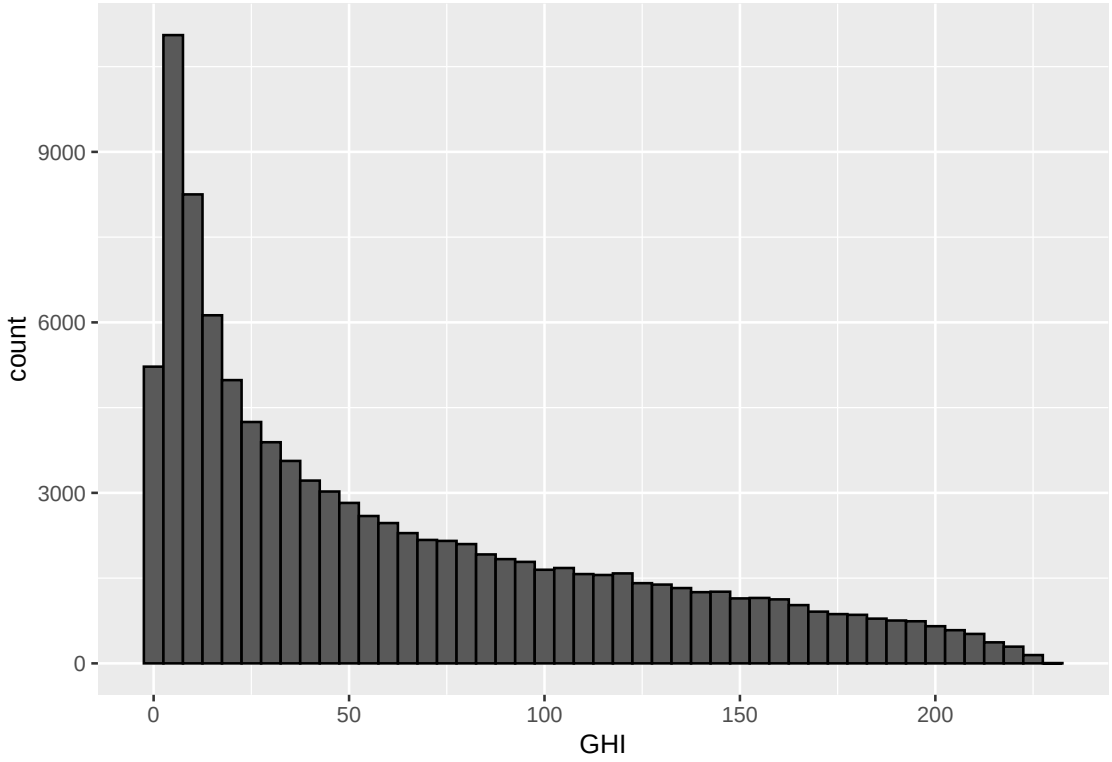
June. Beside that, the smallest energy production by sun are November, December and January.

5.3.2 GHI

- Distribution

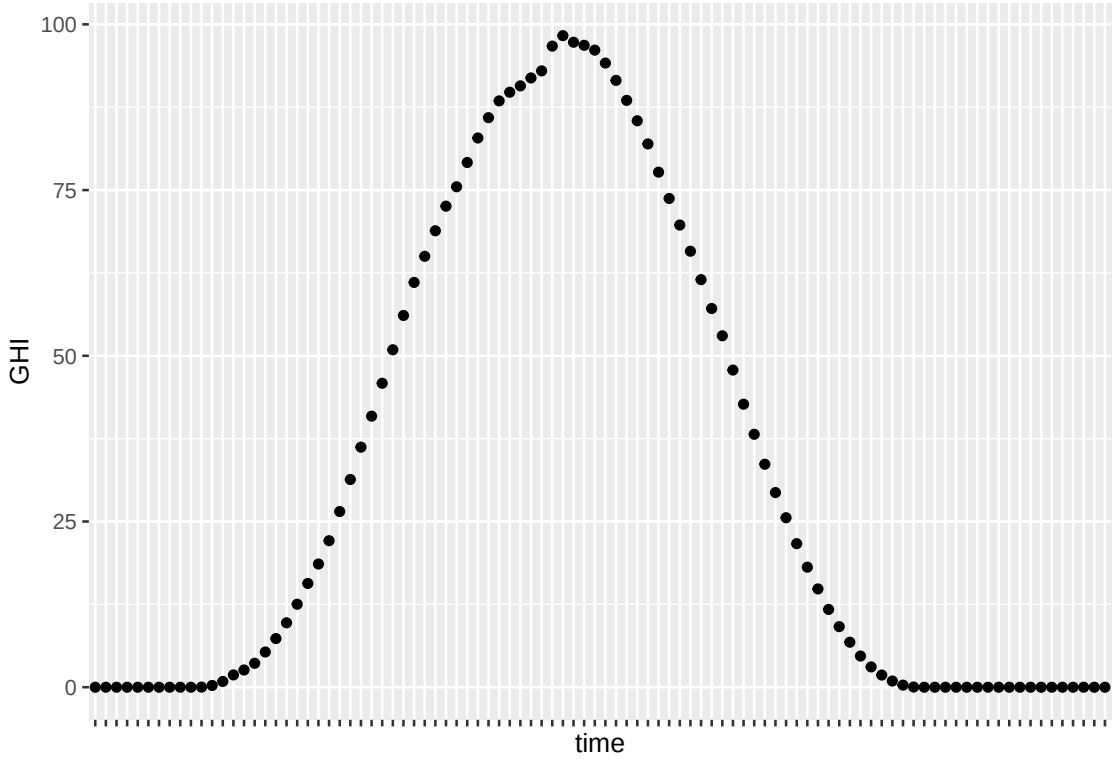
We filter the observations at which there is no sun, based on how GHI is calculated

```
distribution(solar_weather %>% filter(isSun == 1), "GHI", binwidth = 5, col = "black")
```



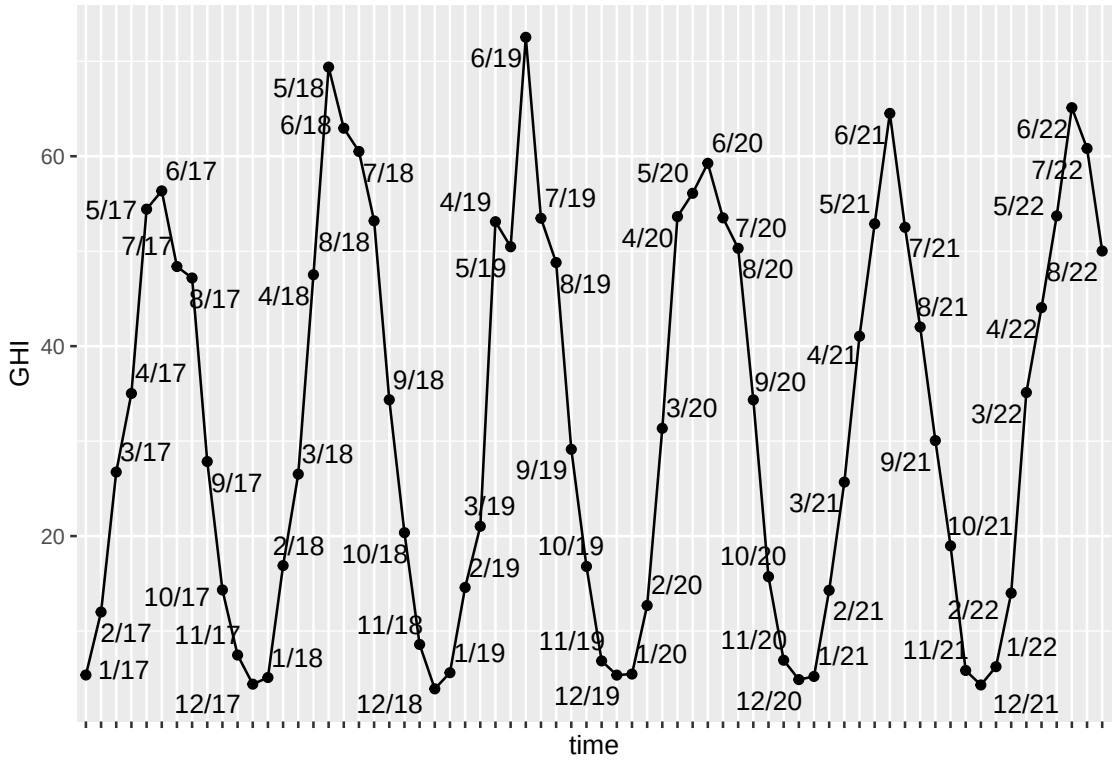
- Change in GHI throughout the day

```
change_tod(solar_weather, "GHI", mean)
```



- Mean GHI every month

```
change_month(solar_weather, "GHI", mean)
```

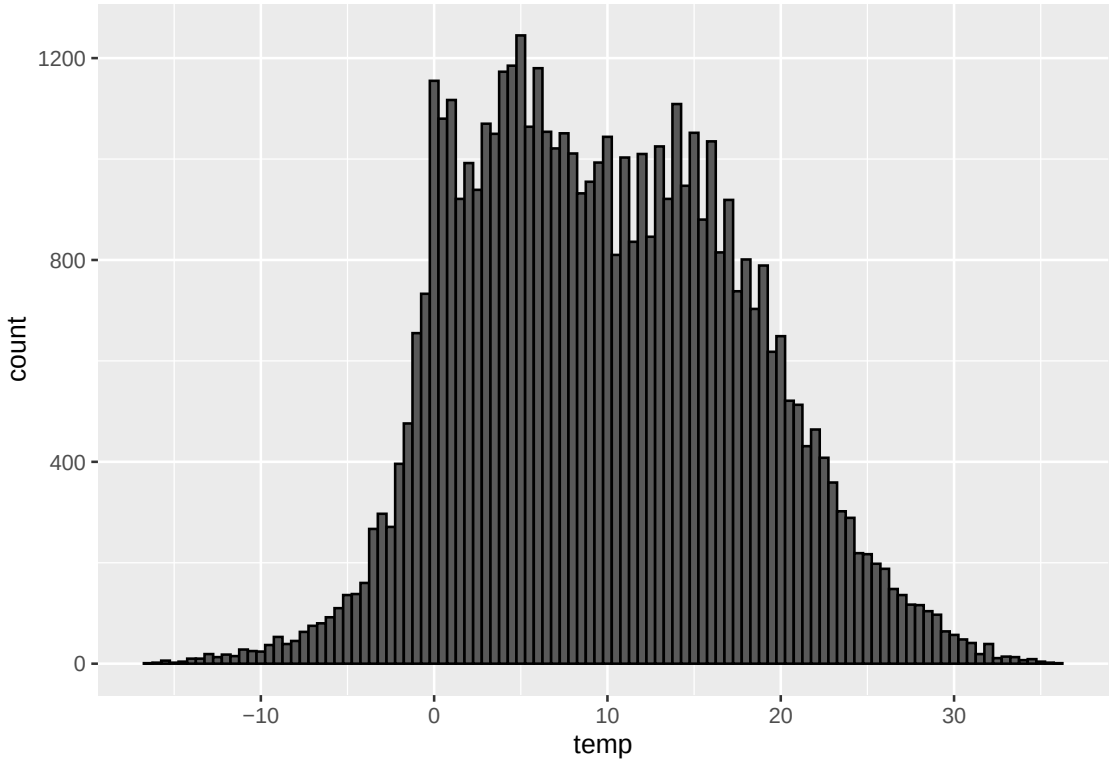


We can observe that the plots for energy production and GHI have similar structure: shape, slope, peak... This discovery, together with the fact that solar panels absorb solar radiation to generate energy, suggests a positive influence of one variable to another. This will be examined further in section 8.1.

5.3.3 Temperature

- Distribution

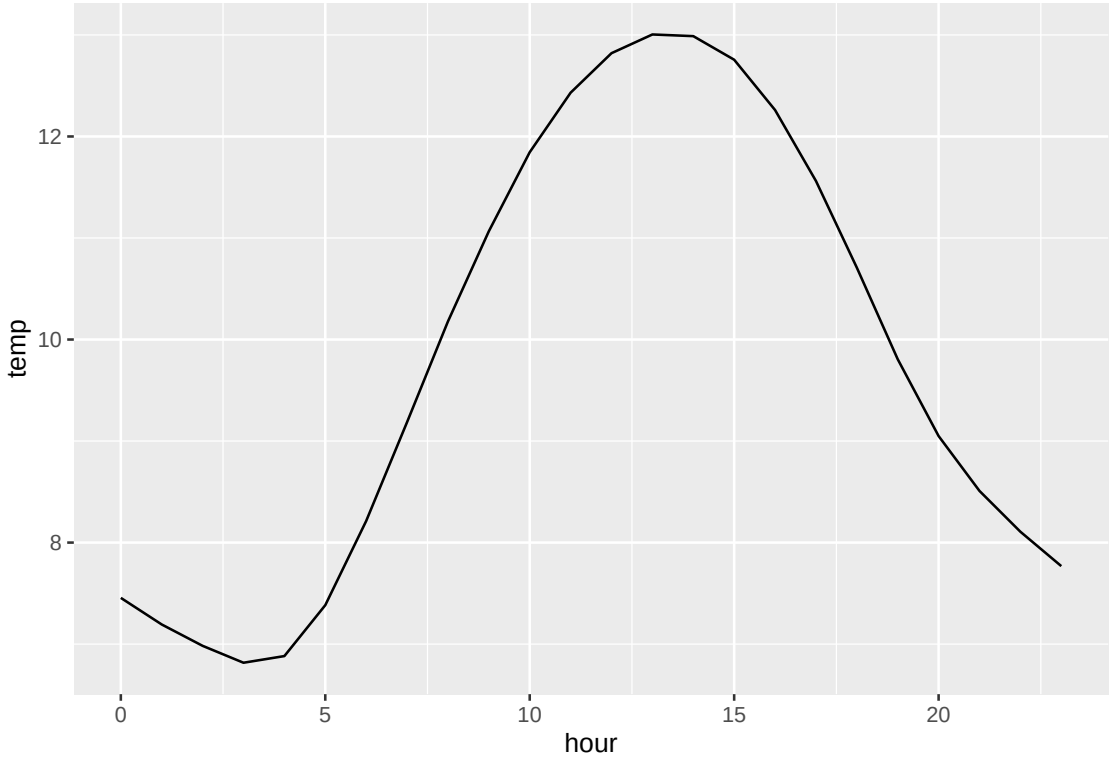
```
distribution(solar_weather_hour, "temp", binwidth = 0.5, col = "black")
```



It seems like the distribution of temperature is normal, we will test that later in section 6.2.

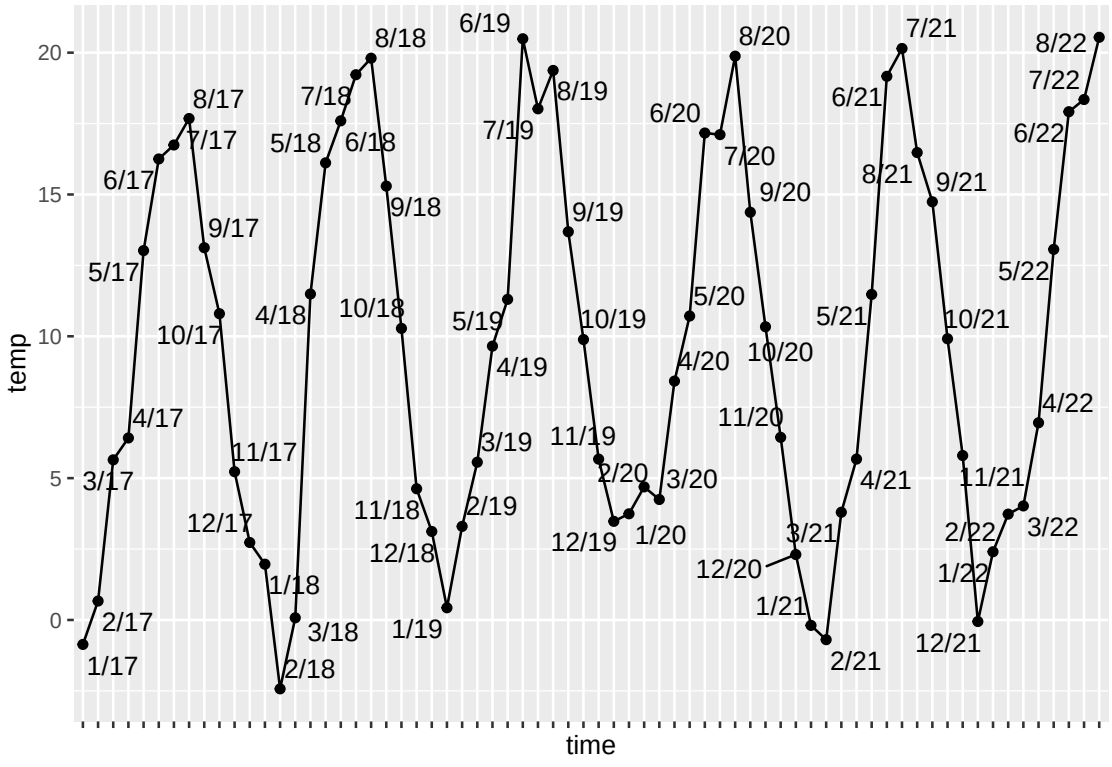
- Change in temperature throughout the day

```
strat_plot(solar_weather_hour, "hour", "temp", mean) +
  geom_line()
```



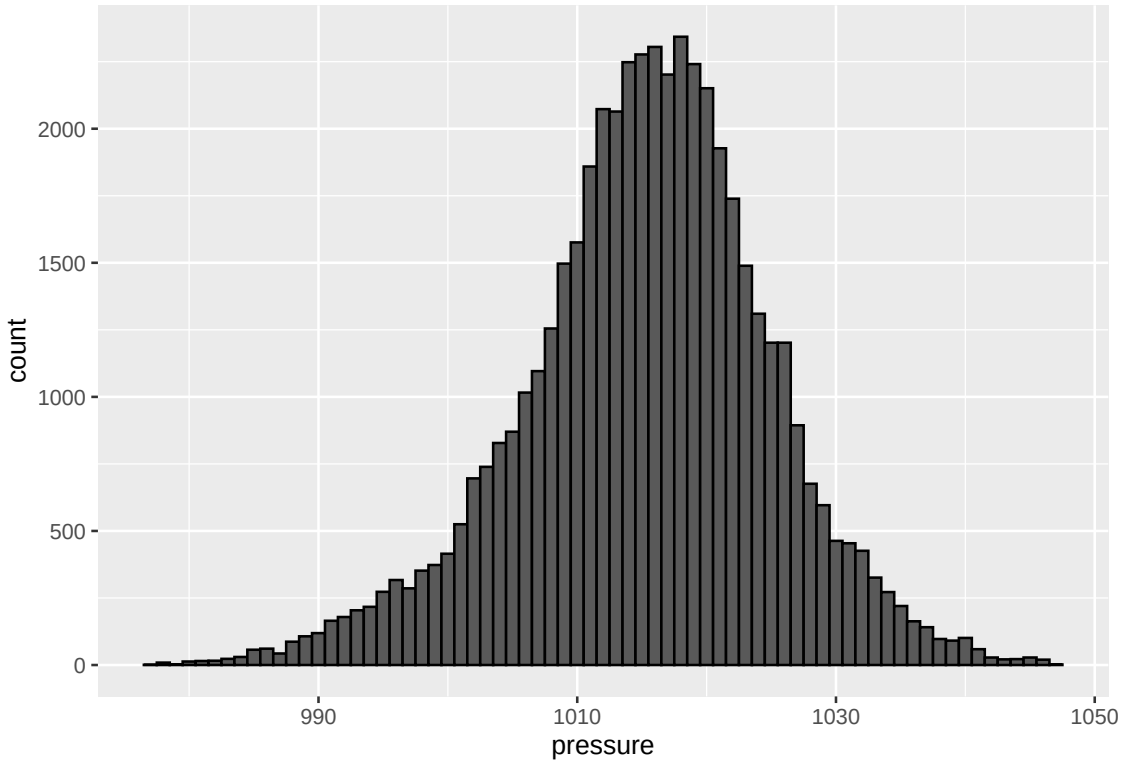
- Mean temperature every month

```
change_month(solar_weather_hour, "temp", mean)
```



5.3.4 Pressure

```
distribution(solar_weather_hour, "pressure", binwidth = 1, col = "black")
```

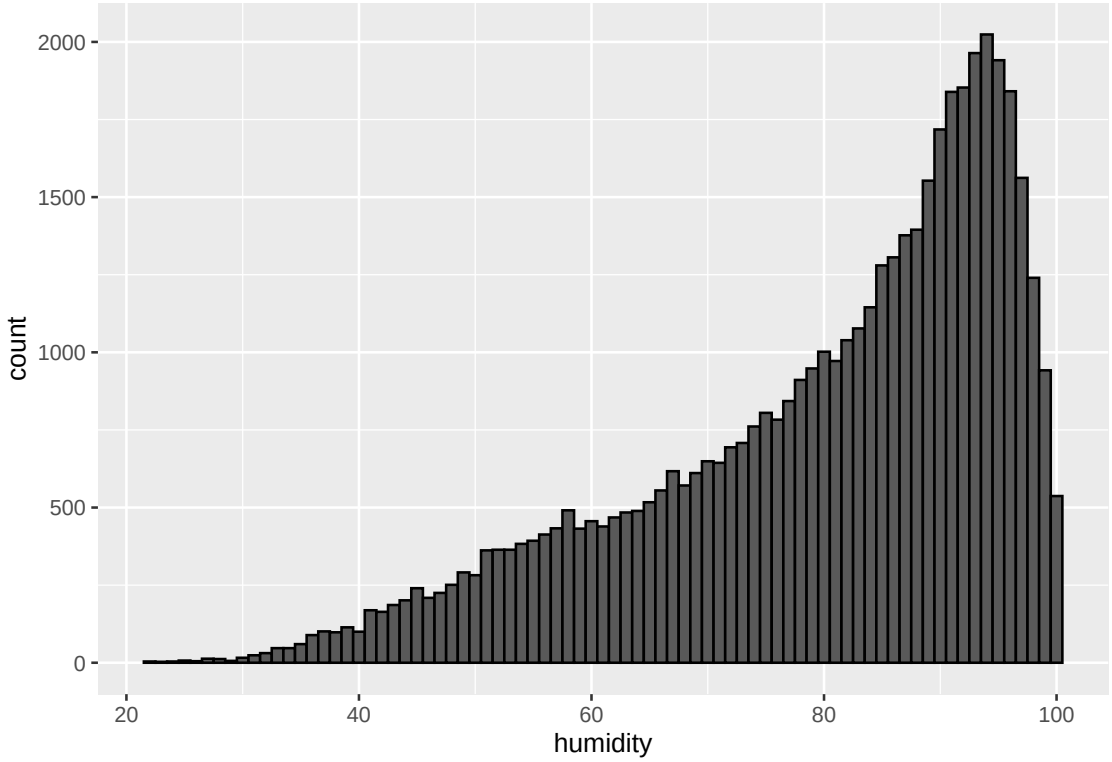


Pressure is also a prominent variable we can consider when we examine the normality of its distribution.

5.3.5 Humidity

- Distribution

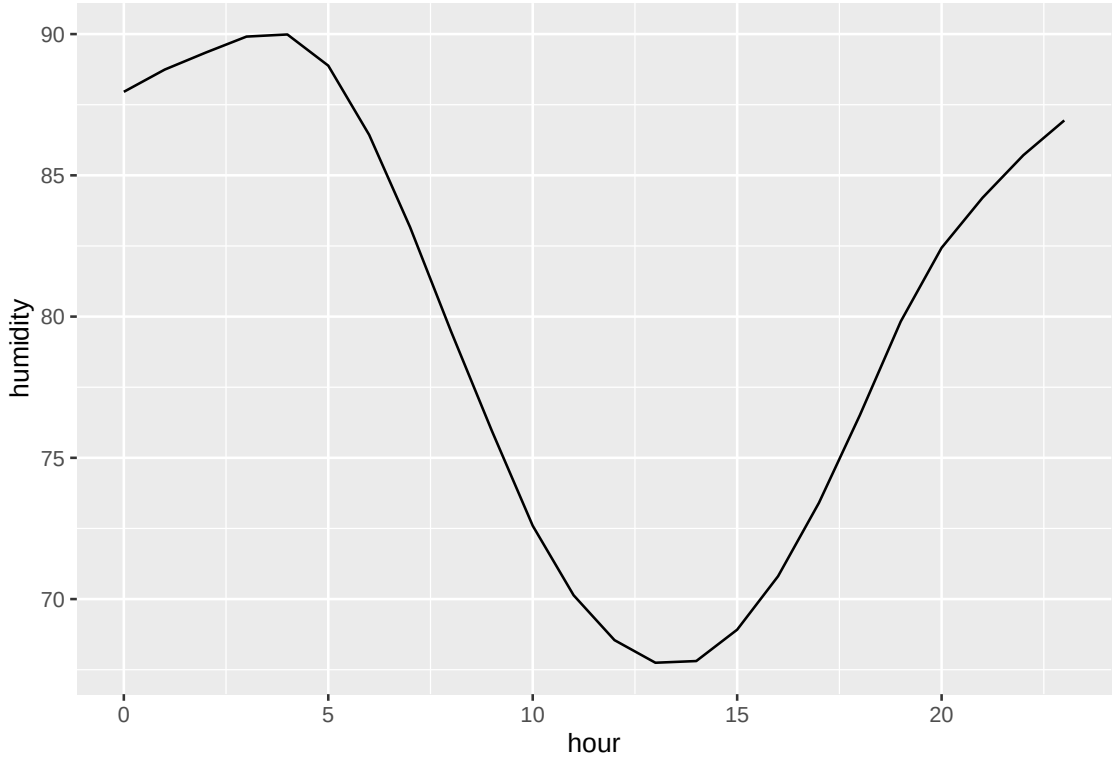
```
distribution(solar_weather_hour, "humidity", binwidth = 1, col = "black")
```



The relative humidity is high, with the mean of around 80%.

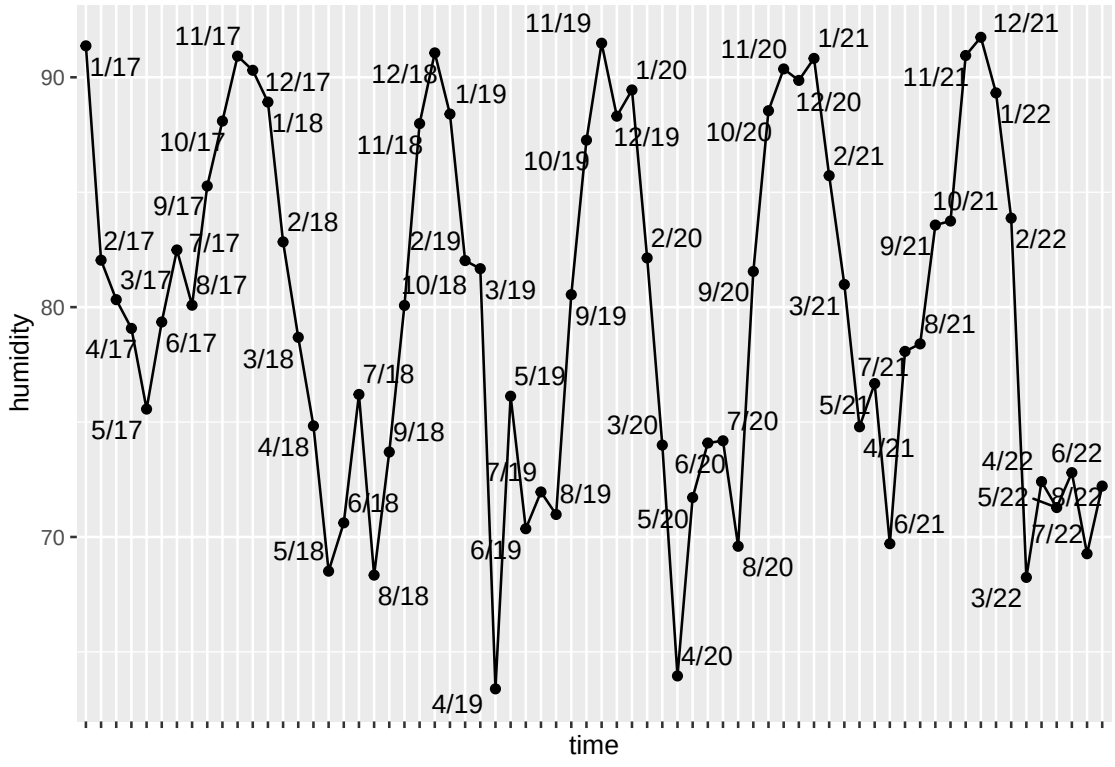
- Change in humidity throughout the day

```
strat_plot(solar_weather_hour, "hour", "humidity", mean) +
  geom_line()
```



- Mean humidity every month

```
change_month(solar_weather_hour, "humidity", mean)
```

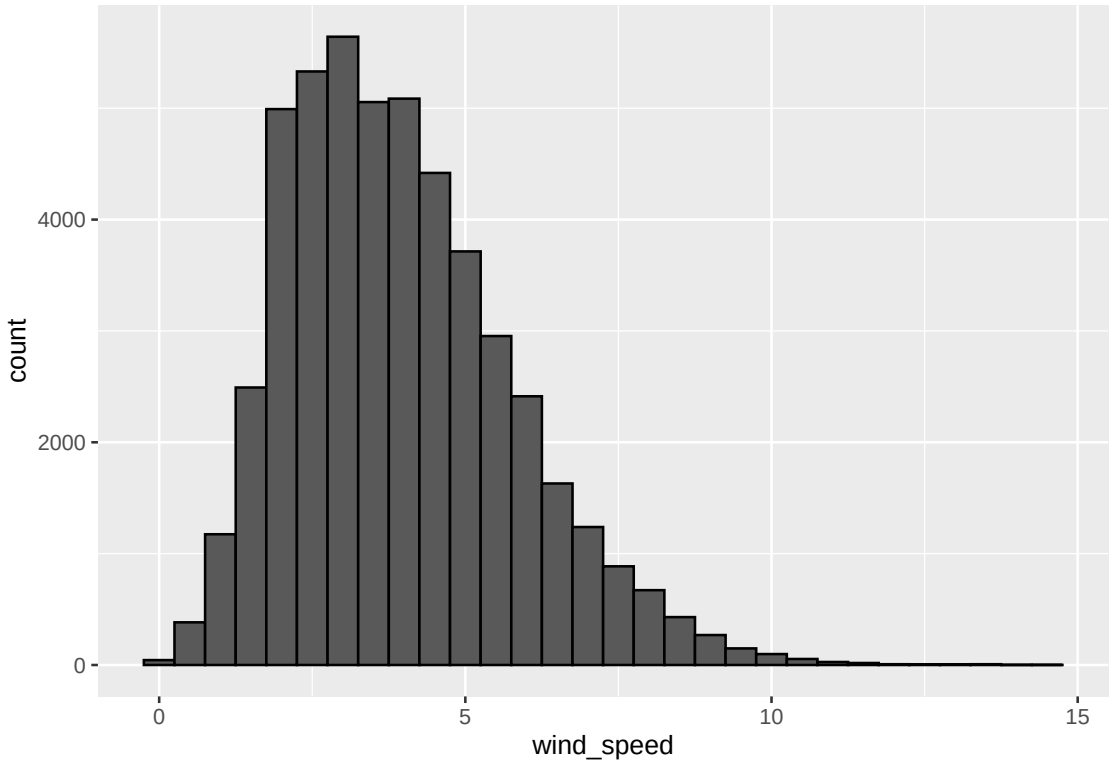


The way humidity changed is quite contrast with those of energy production, GHI and temperature.

5.3.6 Wind speed

- Distribution

```
distribution(solar_weather_hour, "wind_speed", binwidth = 0.5, col = "black")
```

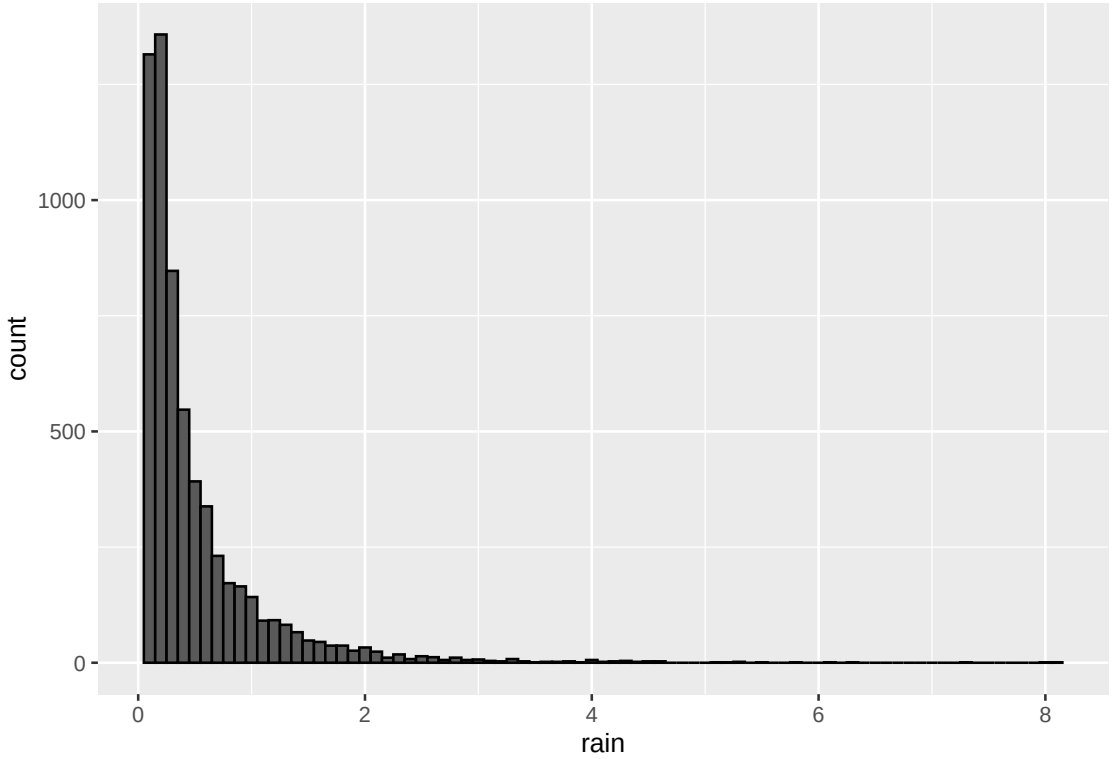


Looking at the distribution, we expected the weather was quite calm with light breeze.

5.3.7 Rain

Only consider observation when it is rainy

```
distribution(filter(solar_weather_hour, rain > 0), "rain", binwidth = 0.1, col = "black")
```

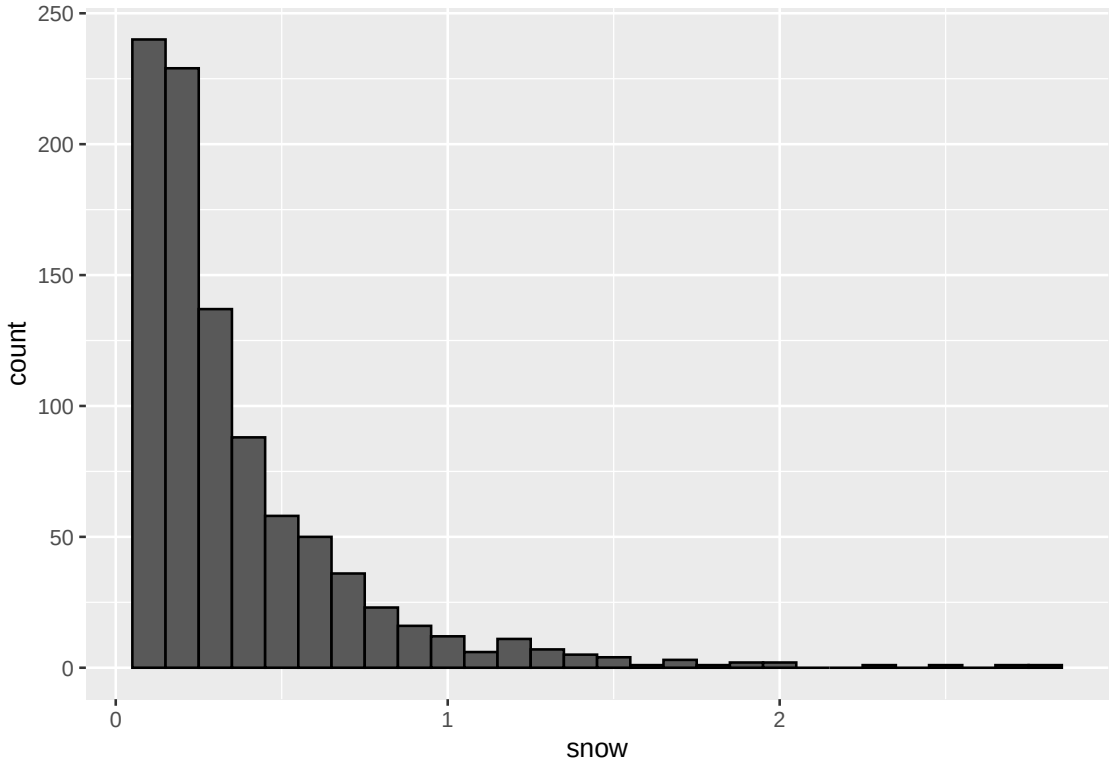


Rain was quite rare, and the amount of precipitation per hour was quite low.

5.3.8 Snow

We remove observations where there aren't snow and group them by day to ease the visualization and to estimate the scarcity of snow.

```
distribution(filter(solar_weather_hour, snow > 0), "snow", binwidth = 0.1, col = "black")
```

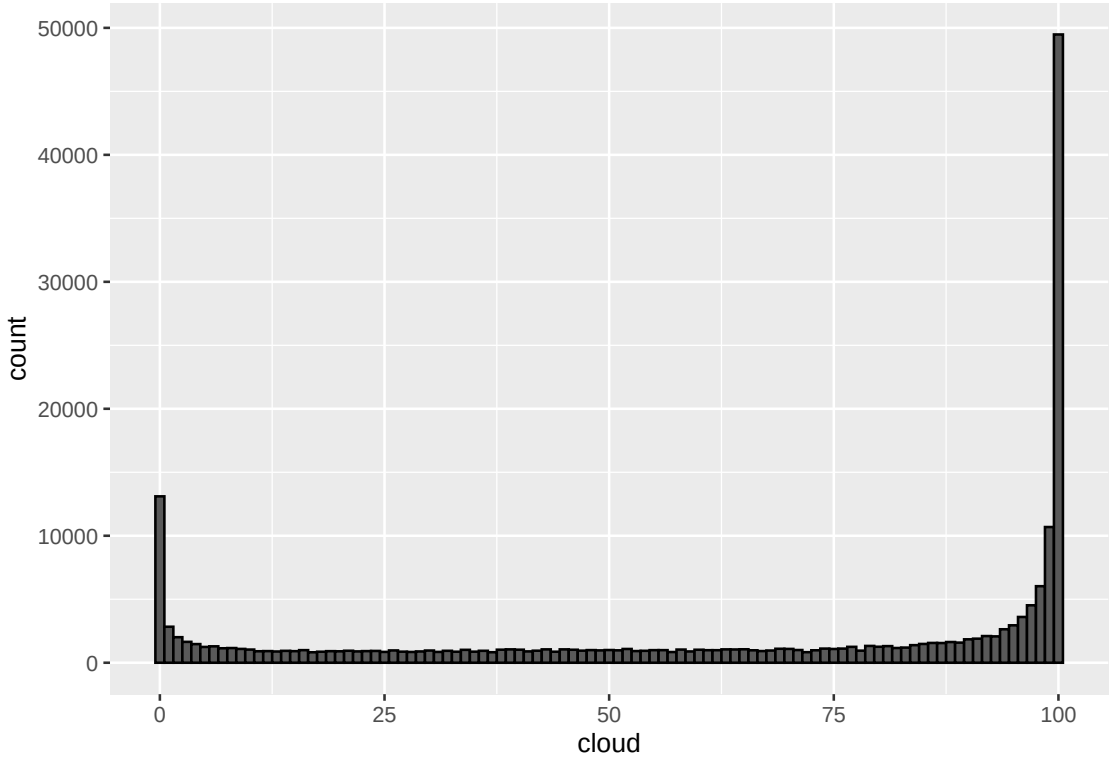


We can conclude that it was rare to see snow, and the amount of snow was quite low. Therefore, we can eliminate the weak dependence of energy production on snow.

5.3.9 Cloud

Most of observations took place when it is cloudy. A significant proportion of those were cloudless too.

```
distribution(solar_weather, "cloud", binwidth = 1, col = "black")
```



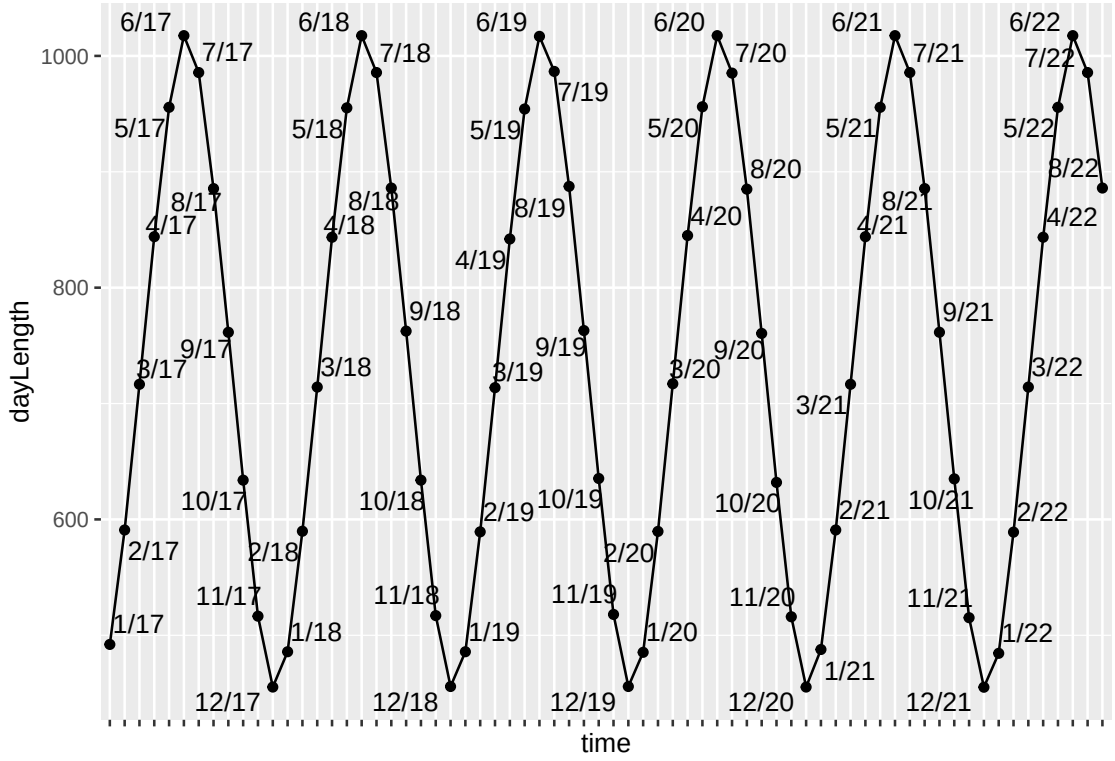
5.3.10 Length of day

Filter duplicated rows: observations taken in one day have the same length of day

```
solar_weather_day <- solar_weather %>%
  select(year, month, day, dayLength) %>%
  distinct()
```

Length of day changed periodically

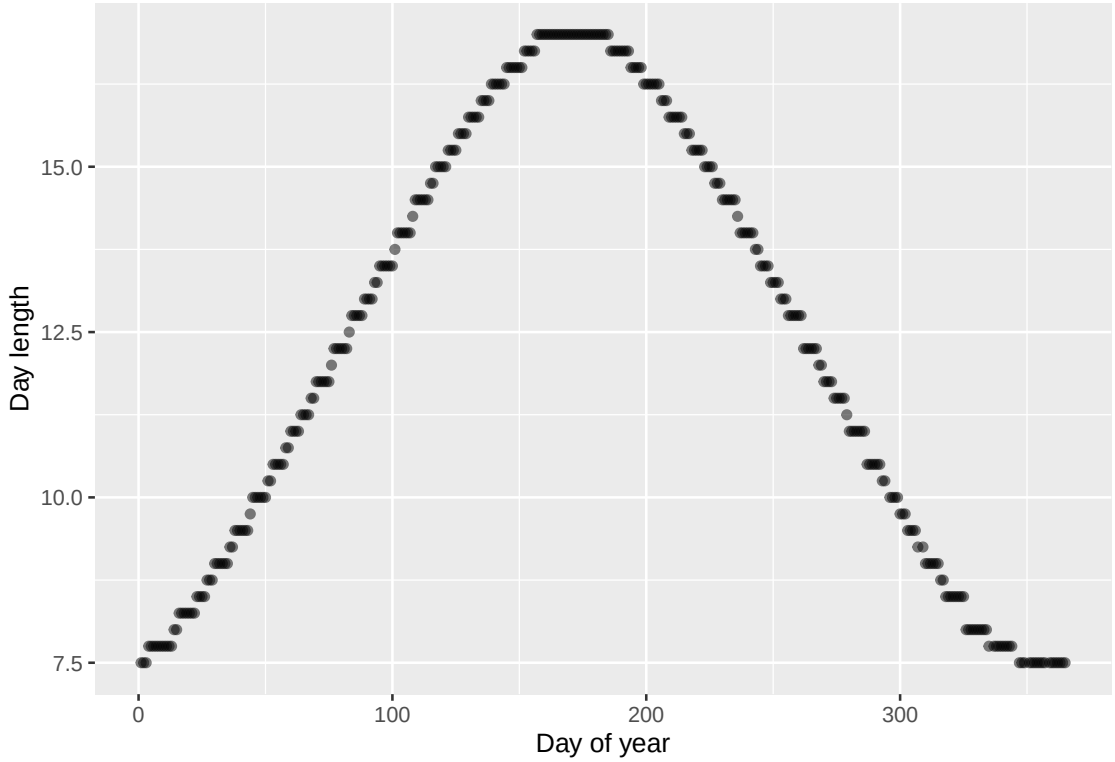
```
change_month(solar_weather_day, "dayLength", mean)
```



To sum up, the length of day is always longer in summer, and quite short in winter.

Further examination of length of day in a full year (2021) reveals:

```
dat_2021 <- solar_weather_day %>%
  filter(year == 2021) %>%
  mutate(
    doy = yday(ymd(str_c(year, "-", month, "-", day))),
    dayHour = dayLength / 60
  )
dat_2021 %>%
  ggplot(aes(doy, dayHour)) +
  geom_point(alpha = 0.5) +
  xlab("Day of year") +
  ylab("Day length")
```



It can be seen that the length of days significantly increased until nearly mid-year before reducing moderately to the end of the year.

6 Normality test for temperature and pressure

6.1 Normal distribution and normality test

We can use visual or statistical methods to test the normality assumption. Let us start with the visuals.

- Histogram: A frequency distribution shows how often each different value in a set of data occurs. A histogram is the most used graph to show frequency distributions. If the graph is seemingly bell-shaped, the data is normally distributed.
- Kernel density plot: A type of plot displays the distribution of values in a data set using one continuous curve. It is similar to a histogram, but it's better at displaying the shape of a distribution since it isn't affected by the number of bins used in the histogram.
- Normal Quantile-Quantile plot:
    1. In the normal Q-Q plot, we plot the theoretical quantiles known as the standard normal variate (a normal distribution with mean equals 0 and standard deviation equals 1) on the x-axis and the ordered values for the random variable which we want to find whether it is Gaussian distributed or not, on the y-axis. A 45-degree reference line is also plotted.
    2. Each observation is plotted as a single point. If all the points plotted on the graph perfectly lies on a straight line then we can clearly say that this distribution is normal because it is evenly aligned with the standard normal variate. This method is usually preferred since we can disregard the sample size.

A popular statistical method is Shapiro-Wilk test of normality. The null hypothesis for the test is normality, so a low p-value indicates that the observed data is unlikely under the assumption it was drawn from a normal distribution.

However, the Shapiro-Wilk test is only intended for relatively small samples. As sample sizes grow, increasingly trivial departures from normality (which are almost always present in real data) will result in small p-values. For this reason, visual tests are more useful.

6.2 Temperature

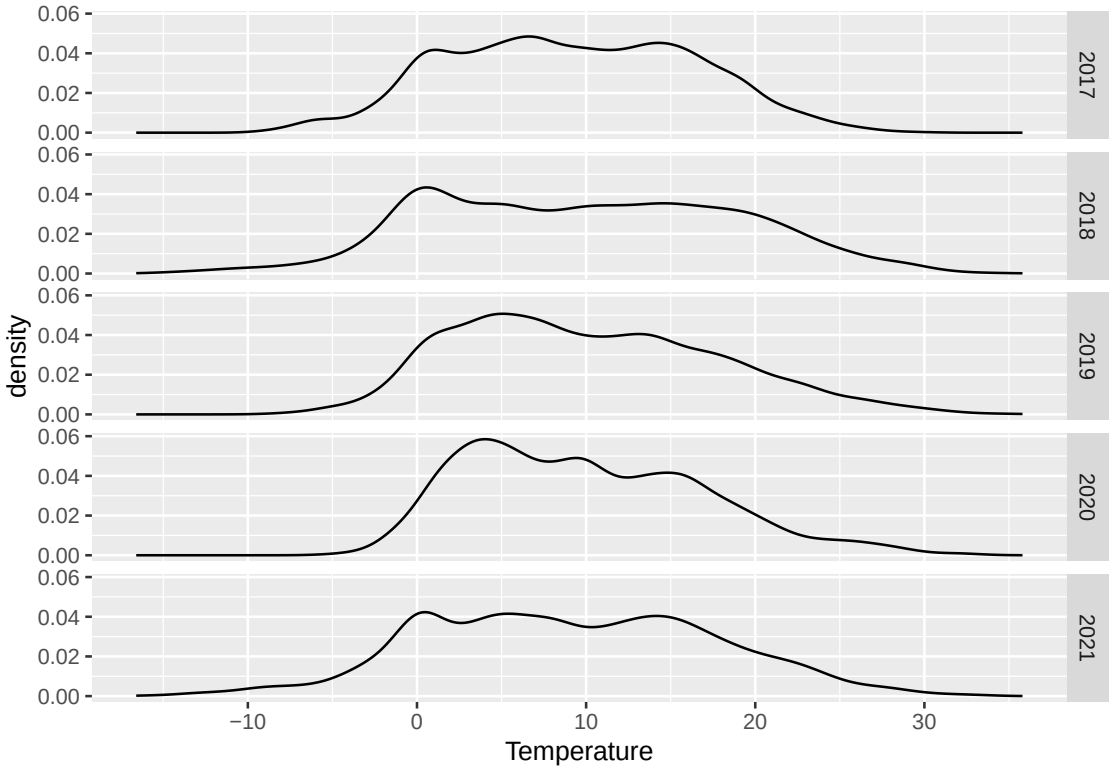
- Data selection:

As observed in the visualization, temperature changed dramatically throughout the day. Even the average temperature in two months of a year had a great difference. Therefore, we need to enlarge our sample size to a year to have a more stable evaluation. We will construct the test data by taking the temperature for each year from 2017 to 2021, as the data for 2022 is incomplete.

```
test_dat <- solar_weather_hour %>%
  filter(year != 2022)
```

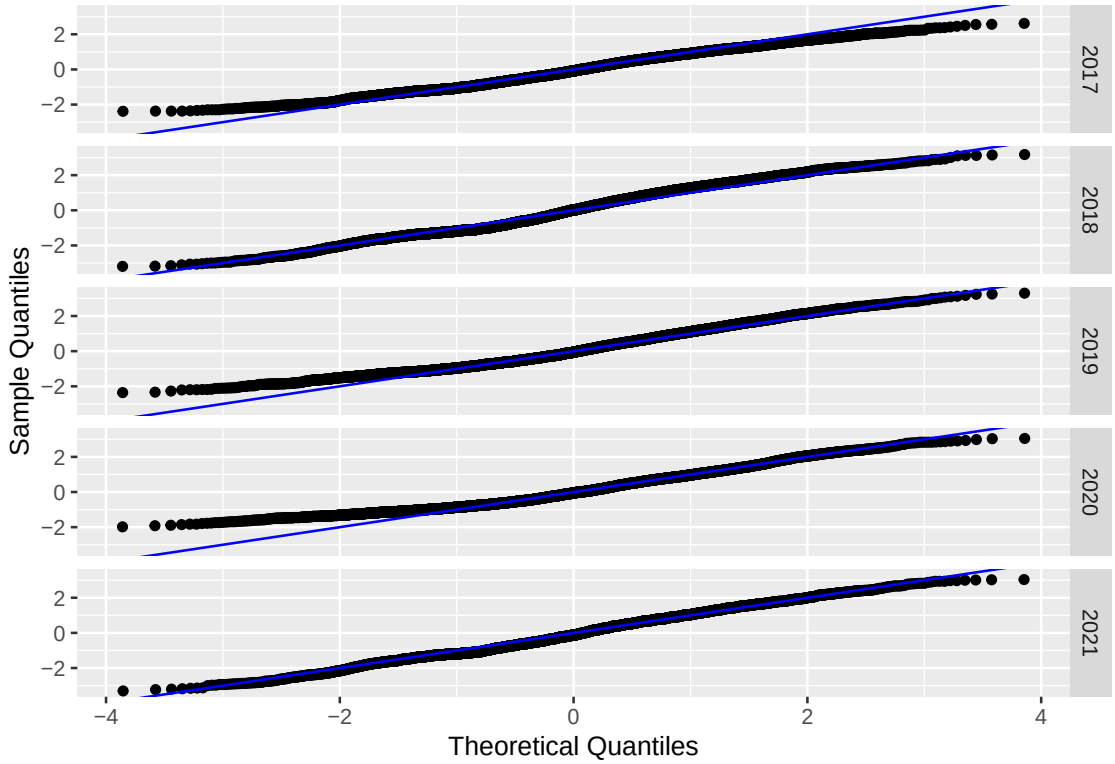
- Density plot:

```
test_dat %>%
  ggplot(aes(temp)) +
  geom_density() +
  xlab("Temperature") +
  facet_grid(year ~ .)
```



- Normal Q-Q plot:

```
test_dat %>%
  ggplot(aes(sample = scale(temp))) +
  geom_qq() +
  geom_abline(col = "blue") +
  xlab("Theoretical Quantiles") +
  ylab("Sample Quantiles") +
  facet_grid(year ~ .)
```



As all of the Q-Q plots are light-tailed, we see that the annual temperature follows normal distribution with more observations taken at extreme temperature value than those at the center of the distribution

The mean and standard deviation temperature over years weren't quite different:

```
get_mean_and_sd <- function(dat, var) {
  vec <- dat %>%
    relocate(any_of(var)) %>%
    rename("var" = 1) %>%
    pull(var)
  tibble(
    mean = mean(vec),
    sd = sd(vec)
  )
}
df <- sapply(split(test_dat, test_dat$year), get_mean_and_sd, var = "temp")
t(df)
```

```
##      mean     sd
## 2017 9.144479 7.200164
## 2018 9.83645  8.999578
```

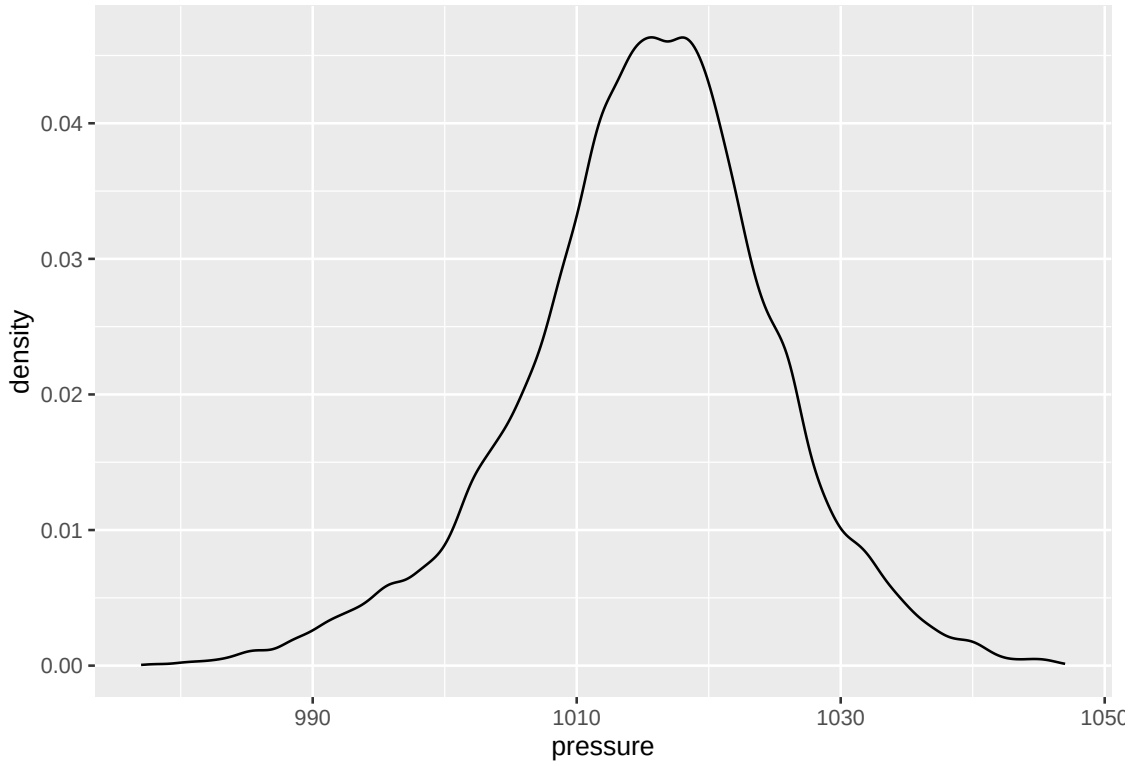
```
## 2019 10.07248 7.724225
## 2020 9.962773 7.012329
## 2021 9.036103 8.498941
```

Surprisingly, the temperature range was large in this area, as the standard deviation were just below the mean temperature.

6.3 Pressure

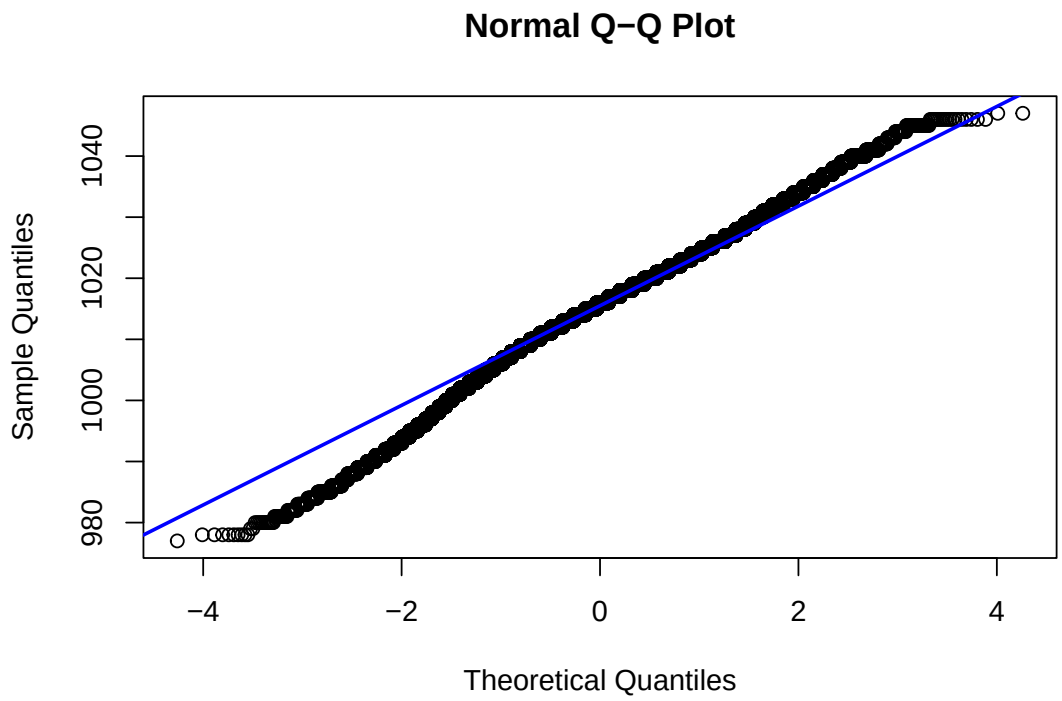
- Density plot:

```
solar_weather_hour %>%
  ggplot(aes(pressure)) +
  geom_density()
```



- Normal Q-Q plot:

```
qqnorm(solar_weather_hour$pressure)
qqline(solar_weather_hour$pressure, col = "blue", lwd = 2)
```



Pressure can be considered to have a normal distribution, though not as clear as the yearly temperature.

7 Linear model to predict global horizontal irradiance (GHI) based on weather parameters

7.1 Purpose

The production of solar energy is essentially driven by the solar irradiance reaching the Earth's surface, quantified as global horizontal irradiance (GHI). This physical quantity is highly variable for two main causes.

The first one is astronomic: the irradiance depends deterministically on the diurnal and seasonal variations of solar elevation above the horizon. This is the controllable factor that we have examined in the visualization.

The second one is meteorological: atmospheric components (water vapor, aerosols and mainly clouds) significantly attenuate the solar radiation passing through the atmosphere and reaching a solar energy production system. This is the part where we want to apply our model to.

7.2 Data selection

It is known that pyranometers can't measure GHI when there is no sun, so we have to remove observations with no sun in order not to litter our data with implicitly missing values.

Nevertheless, we will not consider time as a factor contributing to GHI in the model as we know their relationship is not linear base on the visualization in section 5.3.2. However, as some variables fluctuate significantly throughout the day, we will build our model using the average daily value.

```
solar_weather_lm <- solar_weather %>%
  filter(isSun == 1) %>%
  group_by(year, month, day) %>%
  summarise(
    E = mean(E),
    GHI = mean(GHI),
    temp = mean(temp),
    pressure = mean(pressure),
    humidity = mean(humidity),
    rain = mean(rain),
    snow = mean(snow),
    cloud = mean(cloud)
  ) %>%
  ungroup %>%
  select(c("E", "GHI", "temp", "pressure", "humidity", "rain", "snow", "cloud",
  "year"))
```

7.3 Correlation between two variables

The correlation between two variables refers to the degree to which they are related or associated with each other. This relationship can be positive, negative or neutral.

Positive correlation: When the value of one variable increases, the value of the other variable also increases. For example, there is a positive correlation between the amount of exercise a person does and their overall fitness level.

Negative correlation: When the value of one variable increases, the value of the other variable decreases. For example, there is a negative correlation between the amount of sleep a person gets and their stress levels.

Neutral correlation: When there is no relationship between the two variables. For example, there is no correlation between a person's height and their favorite color.

The strength of the correlation between two variables is measured by the correlation coefficient. The correlation coefficient ranges from -1 to +1. A correlation coefficient of +1 indicates a perfect positive correlation, while a correlation coefficient of -1 indicates a perfect negative correlation. A correlation coefficient of 0 indicates no correlation between the two variables.

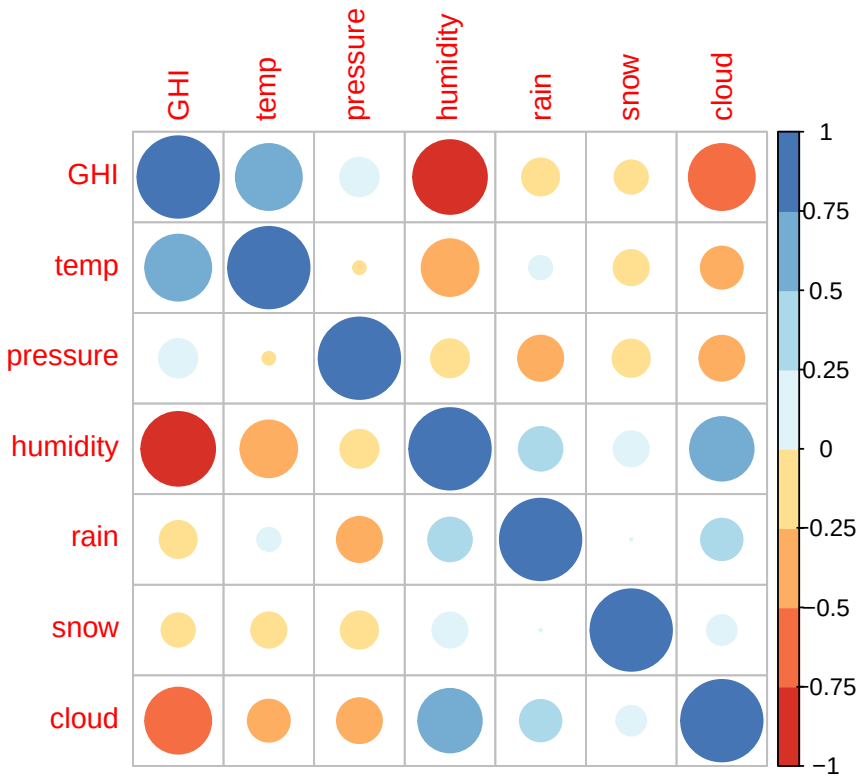
It is important to note that correlation does not imply causation. Just because two variables are correlated does not mean that one causes the other. There may be other factors or variables that are influencing the relationship between the two variables.

Calculate and plot the correlation matrix

```
M <- cor(solar_weather_lm %>% select(-c("E", "year")))
round(M, 3)
```

```
##              GHI   temp pressure humidity   rain   snow  cloud
## GHI        1.000  0.657    0.226   -0.820 -0.208 -0.168 -0.657
## temp       0.657  1.000   -0.025   -0.486  0.083 -0.188 -0.267
## pressure   0.226 -0.025    1.000   -0.219 -0.308 -0.210 -0.305
## humidity -0.820 -0.486   -0.219    1.000  0.288  0.187  0.609
## rain     -0.208  0.083   -0.308    0.288  1.000  0.001  0.260
## snow     -0.168 -0.188   -0.210    0.187  0.001  1.000  0.135
## cloud    -0.657 -0.267   -0.305    0.609  0.260  0.135  1.000
```

```
corrplot(M, col=brewer.pal(n=8, name="RdYlBu"))
```



From the matrix, it is obvious that temperature is directly proportional to GHI, while the opposite trend holds for humidity and cloud.

Also, the weather parameters are not independent. Specifically, humidity is inversely proportional to temperature. Therefore, we expect and will try to eliminate confounding.

7.4 Linear regression model

Linear regression is used to predict the value of an outcome variable Y based on one or more input predictor variables X. The aim is to establish a linear relationship (a mathematical formula) between the predictor variable(s) and the response variable, so that, we can use this formula to estimate the value of the response Y, when only the predictors (Xs) values are known.

The equation for a simple linear regression model with one independent variable is:

$$Y = \beta_0 + \beta_1 X + \epsilon$$

where Y is the dependent variable, X is the independent variable, β_0 is the intercept (the value of Y when X equals zero), β_1 is the slope (the change in Y for every unit change in X), and ϵ is the error term (the difference between the actual value of Y and the predicted value of Y based on the model).

The parameters β_0 and β_1 are estimated from the data using a method called ordinary least squares (OLS), which minimizes the sum of the squared errors between the actual and predicted values of Y. The OLS method finds the values of β_0 and β_1 that minimize the sum of the squared errors, resulting in a line that best fits the data.

Linear regression models can be extended to include multiple independent variables, which allows for more complex relationships between variables to be modeled. In this case, the equation for a

multiple linear regression model with k independent variables is:

$$Y = \beta_0 + \beta_1 X_1 + \cdots + \beta_k X_k + \epsilon$$

where $X_1, X_2, \ldots X_k$ are the k independent variables, and $\beta_1, \beta_2, \ldots \beta_k$ are the corresponding slopes.

Linear regression models can be used for prediction, as well as for understanding the relationship between variables. However, it is important to note that linear regression assumes a linear relationship between the dependent and independent variables, and may not be appropriate for non-linear relationships. Additionally, linear regression assumes that the error terms are normally distributed and have constant variance, which may not always be the case in practice.

We will build a linear model, which concerns the GHI dependent on temperature, pressure, humidity, rain, snow and cloud condition from 2017 to 2020, to predict the GHI of 2021. The reason we choose these years is because we have the data from the year 2021 to compare with our prediction.

```
lin_model <- solar_weather_lm %>%
  filter(year %in% 2017:2020) %>%
  lm(GHI ~ temp + pressure + humidity + rain + snow + cloud, data = .)
```

Our linear regression model has the form:

$$GHI = \beta_0 + \beta_1 temp + \beta_2 pressure + \beta_3 humidity + \beta_4 rain + \beta_5 snow + \beta_6 cloud$$

We can see the coefficients using **tidy**

```
coefs <- tidy(lin_model, conf.int = TRUE)
coefs
```

```
## # A tibble: 7 x 7
##   term        estimate std.error statistic   p.value conf.low conf.high
##   <chr>          <dbl>     <dbl>     <dbl>     <dbl>    <dbl>     <dbl>
## 1 (Intercept)  -69.2     47.9        -1.44 1.49e-  1 -163.       24.8
## 2 temp           1.69     0.0640     26.3  3.68e-125    1.56      1.81
## 3 pressure       0.221    0.0465      4.76 2.16e-  6    0.130     0.313
## 4 humidity      -1.32     0.0460    -28.6  6.02e-143   -1.41     -1.23
## 5 rain          -2.35     2.27       -1.03 3.02e-  1   -6.80      2.11
## 6 snow          28.3      8.76        3.23 1.28e-  3   11.1      45.5
## 7 cloud         -0.279    0.0177    -15.7  2.16e- 51   -0.313    -0.244
```

From this result, we can estimate the coefficients as:

$$\beta_0 = -69.2$$
$$\beta_1 = 1.69$$
$$\beta_2 = 0.221$$
$$\beta_3 = -1.32$$
$$\beta_4 = -2.35$$
$$\beta_5 = 28.3$$
$$\beta_6 = -0.279$$

Let construct the 95% confidence interval for the coefficients:

```
confint(lin_model)
```

```
##                    2.5 %     97.5 %
## (Intercept) -163.2226141 24.7539496
## temp           1.5604679  1.8117093
## pressure       0.1300840  0.3126304
## humidity      -1.4056248 -1.2251517
## rain          -6.8044208  2.1123703
## snow          11.0889325 45.4703510
## cloud         -0.3133961 -0.2437642
```

To sum up, the linear model is:

$$GHI = -69.2 + 1.69temp + 0.221pressure - 1.32humidity - 2.35rain + 28.3snow - 0.279cloud$$

7.5 Hypothesis Tests In Multivariate Linear Regression

7.5.1 Test for significance of regression

The test for significance of regression is a test to determine whether a linear relationship exists between the response variable y and a subset of the regressor variables x_1, x_2, x_3, ..., x_k. The appropriate hypotheses are:

$$H_0 : \beta_0 = \beta_1 = \beta_2 = \beta_3 = ...\beta_k = 0$$
$$H_1 : \exists j : \beta_j \neq 0$$

Rejection of H_0 implies that at least one of the regressor variables $x_1, x_2, ..., x_k$ contributes significantly to the model.

The test for significance of regression is a generalization of the procedure used in simple linear regression. The total sum of squares SST is partitioned into a sum of squares due to the model or to regression and a sum of squares due to error, say:

$$SS_T = SS_R + SS_E$$

Now if $H_0 : \beta_0 = \beta_1 = \beta_2 = \beta_3 = ...\beta_k = 0$ is true, SS_R/ε^2 is a chi-square random variable with k-degrees of freedom. Note that the number of degrees of freedom for this chi-square random variable is equal to the number of regressor variables in the model. We can also show that the SS_R/ε^2 is a chi-square random variable with $(n$ - $p)$ $(p = k + 1)$ degrees of freedom, and that SS_E and SS_R are independent. The test statistic for $H_0 : \beta_0 = \beta_1 = \beta_2 = \beta_3 = ...\beta_k = 0$ is:

$$F_0 = \frac{SS_R/k}{SS_E/(n-p)} = \frac{MS_R}{MS_E}$$

We should reject H_0 if the computed value of the test statistic F_0 , is greater than $f_{a,k,n-p}$.

7.5.2 t-statistic, Pr(>|t|) and Signif. codes

7.5.2.1 t-statistic The t-value is a measure of how many standard deviations our coefficient estimate is far away from 0. The larger our t-statistic is, the more certain we can be that the coefficient is not zero. The t-statistic is then used to find the p-value.

7.5.2.2 Pr(>|t|) The Pr(>|t|) acronym found in the model output relates to the probability of observing any value equal or larger than t. It help us to understand how significant our coefficient is to the model. In practice, any p-value below 0.05 is usually deemed as significant, which means we are confident that the coefficient is not zero and the predictor helps explaining the response variable.

7.5.2.3 Significant codes To the right of some p-values (under Pr(>|t|) column), we can see several asterisks (or none if the coefficient is not significant to the model). The more asterisks, the more significant the coefficient is.

7.5.3 Multiple R-squared and adjusted R-squared

The multiple R-squared ($\mathcal{R}^2$) is most often used for simple linear regression (one predictor). It tells us what percentage of the variation within our dependent variable that the independent variable is explaining. It always lies between 0 and 1 where a number near 0 represents a regression that does not explain the variance in the response variable well and a number close to 1 indicates that regression does explain the observed variance.
The adjusted R-squared value is used when running multiple linear regression and can conceptually be thought of in the same way we described multiple R-squared. The difference between these two metrics is that the adjusted ($\mathcal{R}^2$) essentially penalizes the analyst for adding terms to the model. It is an easy way to guard against overfitting, that is, including regressors that are not really useful. Consequently, it is very useful in comparing and evaluating competing regression models.

7.5.4 Interpreting our result

```
summary(lin_model)
```

```
##
## Call:
## lm(formula = GHI ~ temp + pressure + humidity + rain + snow +
##     cloud, data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -64.437 -10.107  -0.488  10.008  47.609
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -69.23433   47.91400  -1.445  0.14868
## temp          1.68609    0.06404  26.329  < 2e-16 ***
## pressure      0.22136    0.04653   4.757 2.16e-06 ***
## humidity     -1.31539    0.04600 -28.595  < 2e-16 ***
## rain         -2.34603    2.27283  -1.032  0.30215
## snow         28.27964    8.76360   3.227  0.00128 **
## cloud        -0.27858    0.01775 -15.696  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.35 on 1448 degrees of freedom
## Multiple R-squared:  0.8089, Adjusted R-squared:  0.8081
## F-statistic:  1021 on 6 and 1448 DF,  p-value: < 2.2e-16
```

From the coefficients table, we can see that the Pr(>|t|) value for rain is around 0.3, which is much greater than 0.05, so there is no sufficient evidence to reject the hypothesis $\mathcal{H}_0$ corresponding to this variable. On the other hand, other predictors are extremely significant (most of them have three asterisks). Therefore, we can rebuild our model by eliminating only the rain indicator.

The multiple $\mathcal{R}^2$ is 80.89% and the adjusted $\mathcal{R}^2$ is 80.81%. These numbers imply that about 80% of the variability in GHI is caused by the measured weather factors.

Finally, we have the $\mathcal{F}$-statistic of 1021, which is very large compared to $f_{0.05,6,1448} = 2.104832$

Our p-value is approximately zero, so we are able to reject the null hypothesis and conclude that the global solar irradiance is linearly related to at least one the weather parameters.

7.6 Rebuild the model

As discussed above, we will rebuild our linear model after discarding the insignificant rain parameter. We take the same wrangled data set as before, which only includes observations from 2017 to 2020.

```
new_lm <- solar_weather_lm %>%
  filter(year %in% 2017:2020) %>%
  lm(GHI ~ temp + pressure + humidity + snow + cloud, data = .)
```

We can see the coefficients and test statistics as:

```
summary(new_lm)
```

```
##
## Call:
## lm(formula = GHI ~ temp + pressure + humidity + snow + cloud,
##     data = .)
##
## Residuals:
##     Min      1Q  Median      3Q     Max
## -64.752  -9.949  -0.528  10.075  47.815
##
## Coefficients:
##              Estimate Std. Error t value Pr(>|t|)
## (Intercept) -78.55576   47.05635  -1.669  0.09525 .
## temp          1.67083    0.06231  26.814  < 2e-16 ***
## pressure      0.23150    0.04548   5.090 4.05e-07 ***
## humidity     -1.32729    0.04453 -29.803  < 2e-16 ***
## snow         28.78277    8.75023   3.289  0.00103 **
## cloud        -0.27988    0.01770 -15.808  < 2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 15.35 on 1449 degrees of freedom
## Multiple R-squared:  0.8087, Adjusted R-squared:  0.8081
## F-statistic:  1225 on 5 and 1449 DF,  p-value: < 2.2e-16
```

From this result, the new linear model is:

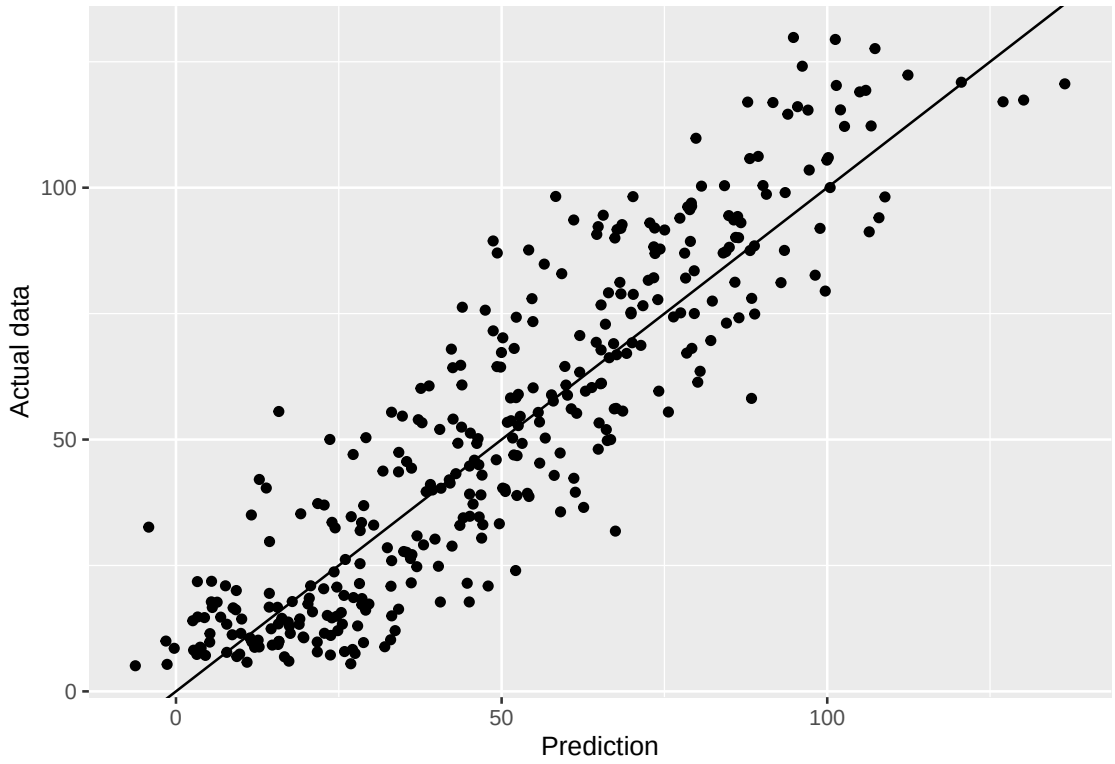
$$GHI = -78.55576 + 1.67083\, temp + 0.2315\, pressure - 1.32729\, humidity + 28.78277\, snow - 0.27988\, cloud$$

For the new model, we can see that all the predictors are significant. Furthermore, the new adjusted R-square is 80.81%, which is the same as the adjusted R-square before the removal. This implies that the rain parameter is not helpful in predicting GHI.

To see how well our model actually predicts energy production, we can predict the daily average energy generated in 2021, then make a plot to compare with the actual data.

```
test_dat <- solar_weather_lm %>%
  filter(year == 2021) %>%
  mutate(GHI_hat = predict(new_lm, newdata = .))
test_dat %>%
  ggplot(aes(GHI_hat, GHI)) +
  geom_point() +
```

```
  geom_abline() +
  xlab("Prediction") +
  ylab("Actual data")
```



The points spread evenly and not far away from the identity line. The model works moderately well.

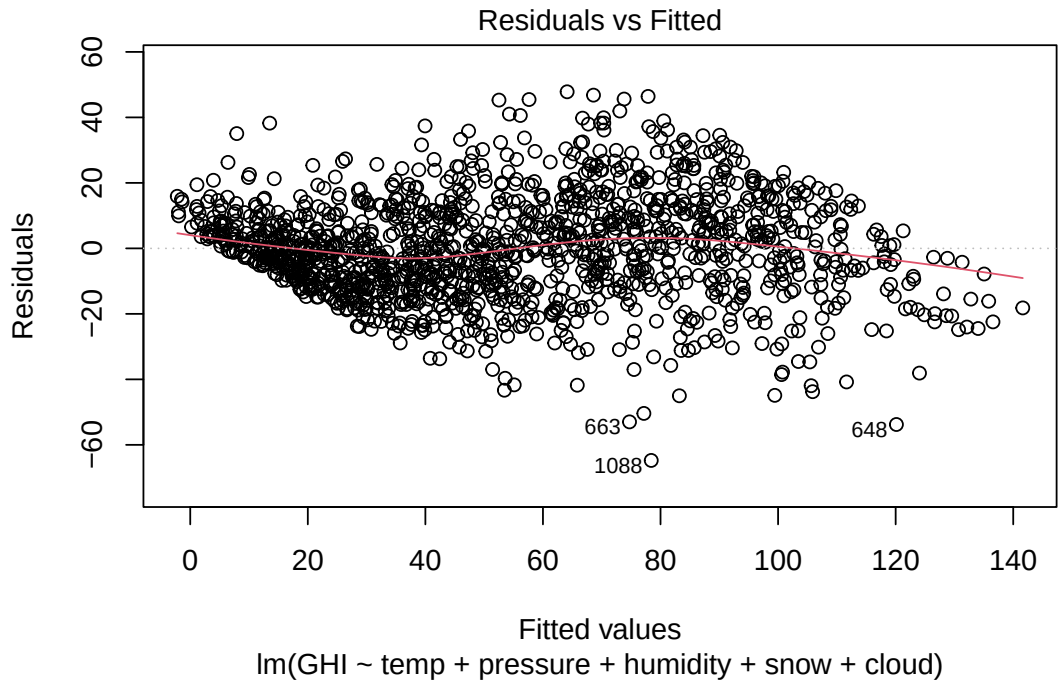
7.7 Test for the normality distribution of errors

Now that we have no other adjustments to the model, let's run some tests to verify that the errors are normally distributed with mean 0.

7.7.1 Residuals vs Fitted plot

The first method to check the normality assumption of residuals is by creating a scatter plot of fitted (predicted) values with the corresponding residual (error) values. For the normality assumption to hold, the residuals should spread randomly around 0, which gives us the uniform variance, and form a horizontal band.

```
plot(new_lm, which = 1)
```

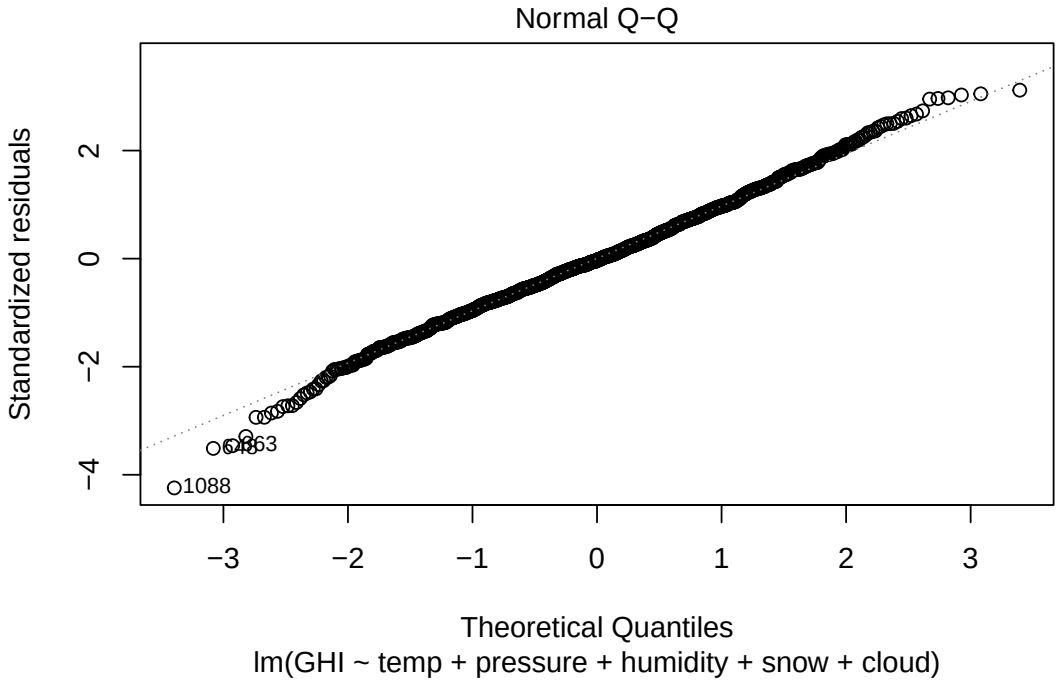


The plot shows both a line around 0, as well as a red trend line. Looking closely, we see that the red line is quite curvy and unstable but it surrounds the line y = 0 in general. As a result, we shall conclude that the normality of the data is relatively satisfied.

7.7.2 Normal Quantile-Quantile plot

We will plot the standardized residuals on the y-axis, together with the theoretical quantiles on the x-axis.

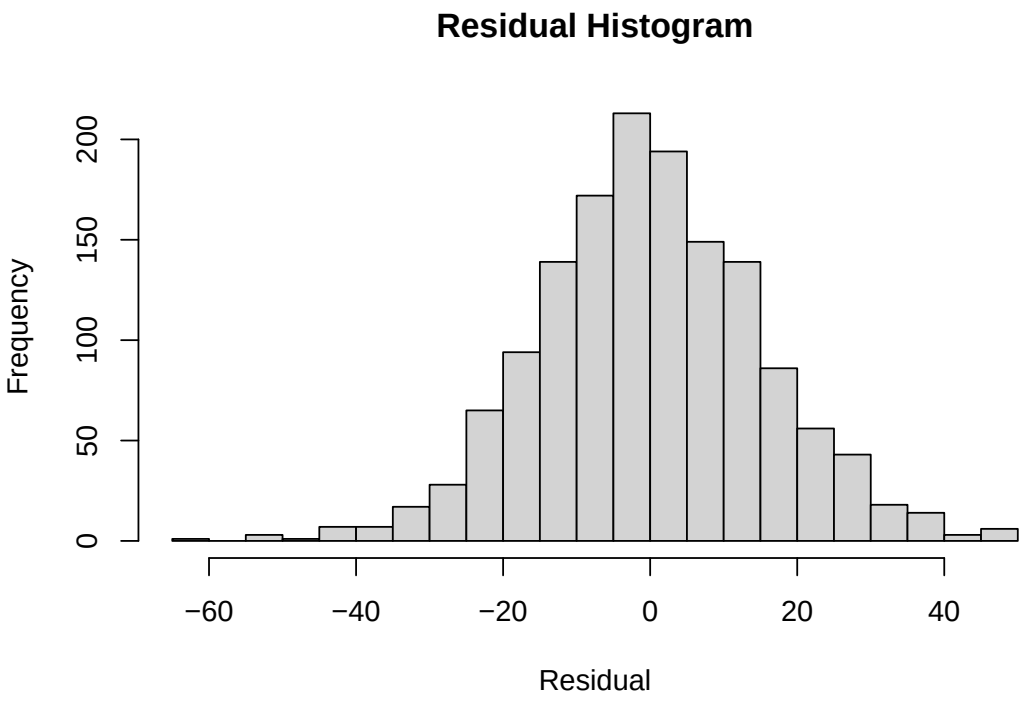
```
plot(new_lm, which = 2)
```



Most of the residual points lie on the identity line, so we can be certain that the errors follows the normal distribution.

7.7.3 The histogram of residuals

```
hist(new_lm$residuals, breaks = 25, main = "Residual Histogram", xlab = "Residual")
```



The histogram is centered around zero and show a bell-shaped curve with low density at the extremes, so the model doesn't violate the normality assumption.

7.7.4 One-sample Kolmogorov-Smirnov normality test

The One-sample Kolmogorov-Smirnov (KS) normality test is a statistical test used to determine whether a given data set follows a normal distribution. It is a non-parametric test that makes no assumptions about the mean or variance of the population. Instead, it compares the empirical cumulative distribution function (ECDF) of the data to the cumulative distribution function (CDF) of a normal distribution with the same mean and standard deviation.

The KS test statistic, denoted by D, is calculated as the maximum absolute difference between the ECDF and the CDF. The null hypothesis for the test is that the data is normally distributed. If the test statistic D is smaller than the critical value at a chosen significance level, then the null hypothesis is not rejected, and the data is assumed to be normally distributed. On the other hand, if D is greater than the critical value, then the null hypothesis is rejected, and the data is assumed to deviate significantly from a normal distribution.

It should be noted that the test may have limited power for small sample sizes (<50 samples) and may not be sensitive to other types of departures from normality, such as skewness or kurtosis.

Given that the number of observations that we use to train the model is 1455, the one-sample Kolmogorov-Smirnov test is most suitable.

```
error <- new_lm$residuals
fun_ecdf <- ecdf(error)
m = mean(error)
s = sd(error)
ks.test(error, "pnorm", mean = m, sd = s)
```

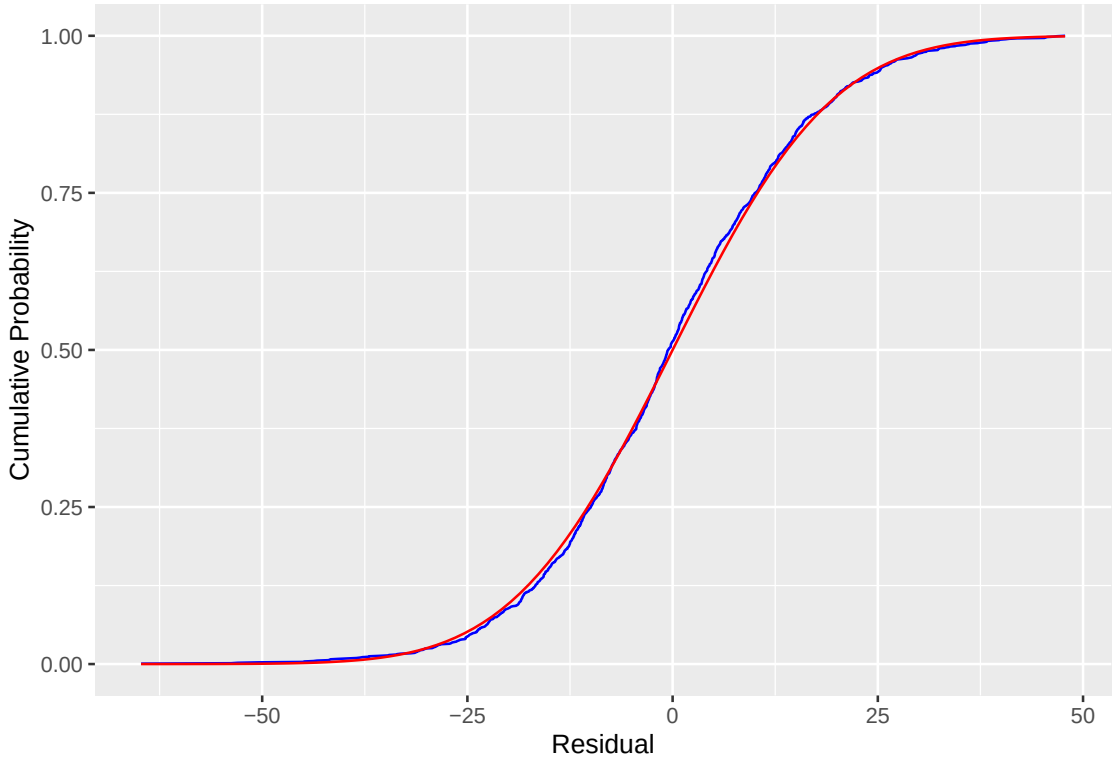
```
##
```

```
##  Asymptotic one-sample Kolmogorov-Smirnov test
##
## data:  error
## D = 0.024075, p-value = 0.368
## alternative hypothesis: two-sided
```

The p-value returned from the test is a measure of the strength of evidence against the null hypothesis that the data follows a normal distribution. Specifically, the p-value is the probability of obtaining a test statistic D that is as extreme or more extreme than the observed D, assuming the null hypothesis is true.

In our case, the p-value (0.368) is much larger than the chosen significance level (0.05), so there is insufficient evident to reject the null hypothesis. We illustrate this fact by drawing the CDF for the residuals

```
data.frame(x = error, y = fun_ecdf(error)) %>%
  ggplot(aes(x)) +
  geom_line(aes(y = y), col = "blue") +
  stat_function(fun = pnorm, args = list(mean = m, sd = s), col = "red") +
  xlab("Residual") +
  ylab("Cumulative Probability")
```



The red line is the CDF of normal distribution, while the blue line is an empirical CDF of the calculated residuals.

8 Three renewable energy solutions

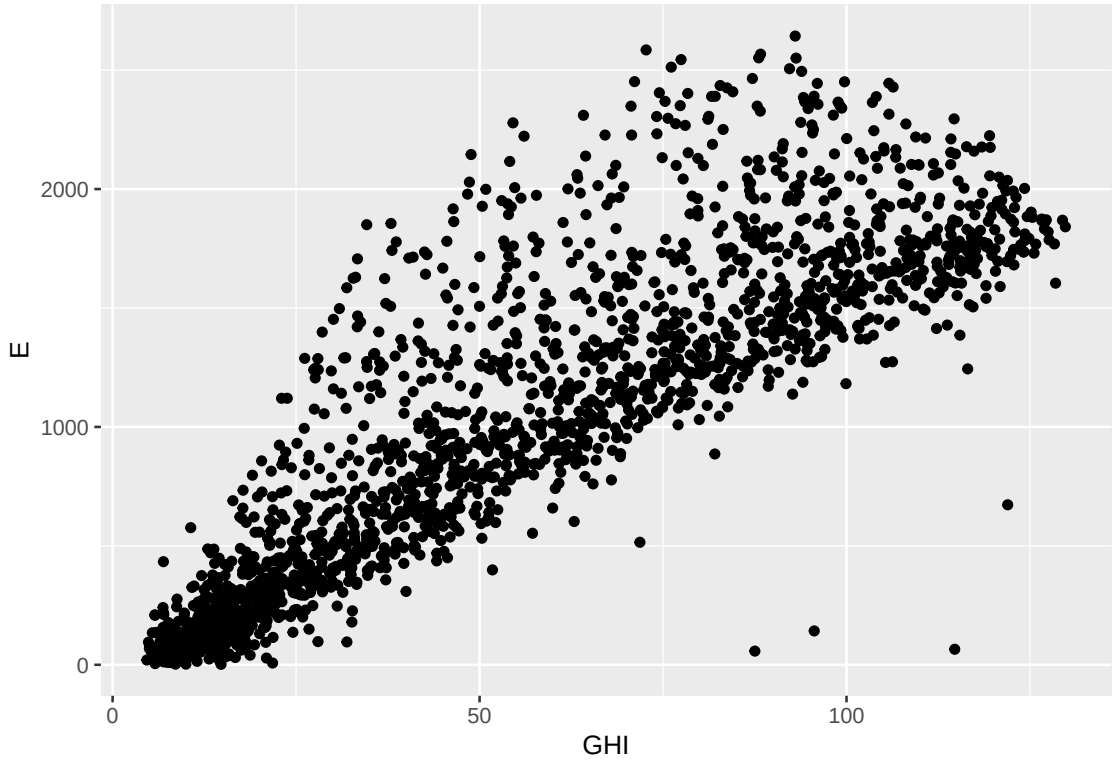
8.1 Solar energy

We reuse the wrangled data from the linear model as it accounts for sun availability and fluctuation throught the day of GHI.

```
cat("The correlation between  GHI  and  energy production is: ",
with(solar_weather_lm, cor(E, GHI)))
```

```
## The correlation between  GHI  and  energy production is:  0.8698023
```

```
solar_weather_lm %>%
  ggplot(aes(GHI, E)) +
  geom_point()
```



Clearly, there is a strong relationship between GHI and energy production in this area, which may assist the development of solar energy. Nevertheless, the higher value of GHI, the more variability seen in the amount of energy generated. This implies that we need to consider other configurations other than measured GHI to evaluate the energy efficiency.

8.2 Hydro electricity

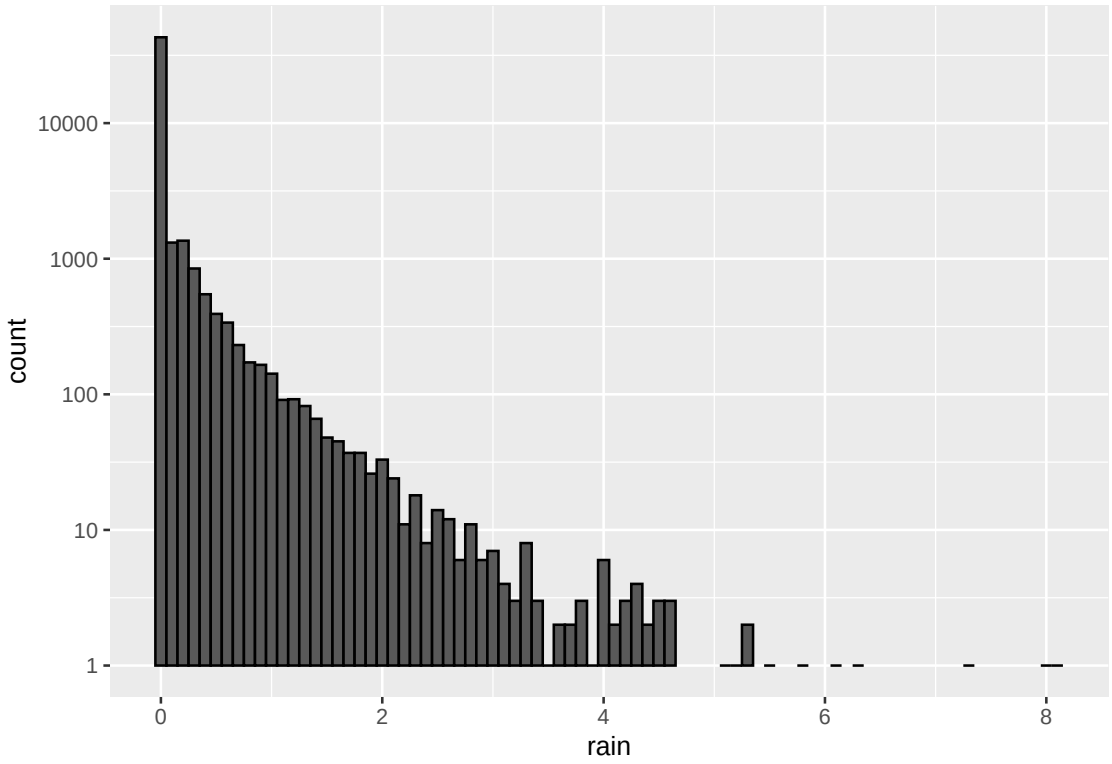
To evaluate the potential of building a dam to generate electricity, rain is unquestionably the most important indicator. Let reexamine the distribution of recipitation in millimeters per hour. This time, we use decimal logarithm to scale the y-axis to ease the visualization.

```
dat <- solar_weather %>%
  select(year, month, day, hour, rain) %>%
  distinct()
distribution(dat, "rain", binwidth = 0.1, col = "black") +
```

```
  scale_y_continuous(trans = "log10")
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

```
## Warning: Removed 25 rows containing missing values (`geom_bar()`).
```



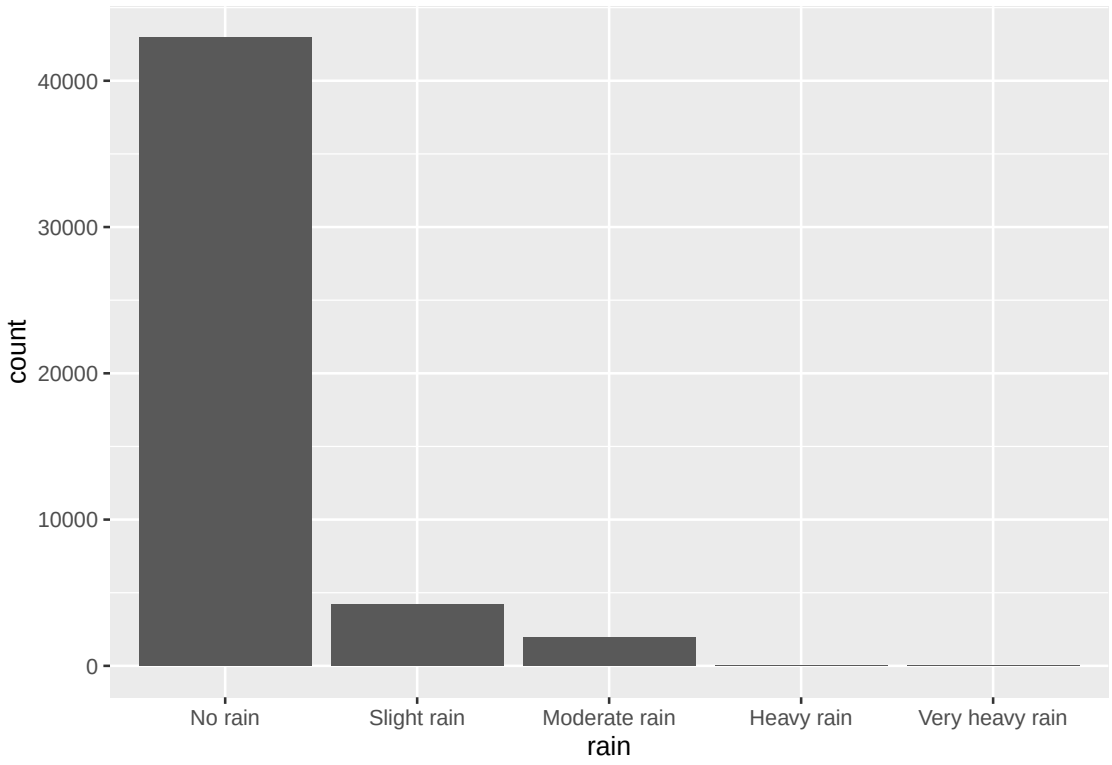
If we categories the data by the rates of rainfall [3] below:

Slight rain	Less than 0.5 mm per hour
Moderate rain	Greater than 0.5 mm per hour, but less than 4.0 mm per hour
Heavy rain	Greater than 4 mm per hour, but less than 8 mm per hour
Very heavy rain	Greater than 8 mm per hour

we will have:

```
rain_rate <- dat %>%
  mutate(
    rain = case_when(
      rain == 0 ~ "No rain",
      rain < 0.5 ~ "Slight rain",
      rain < 4.0 ~ "Moderate rain",
      rain < 8.0 ~ "Heavy rain",
      TRUE ~ "Very heavy rain"
    )
  ) %>%
  mutate(
    rain = factor(rain, levels = c("No rain", "Slight rain", "Moderate rain",
    "Heavy rain", "Very heavy rain"))
  )
rain_rate %>%
```

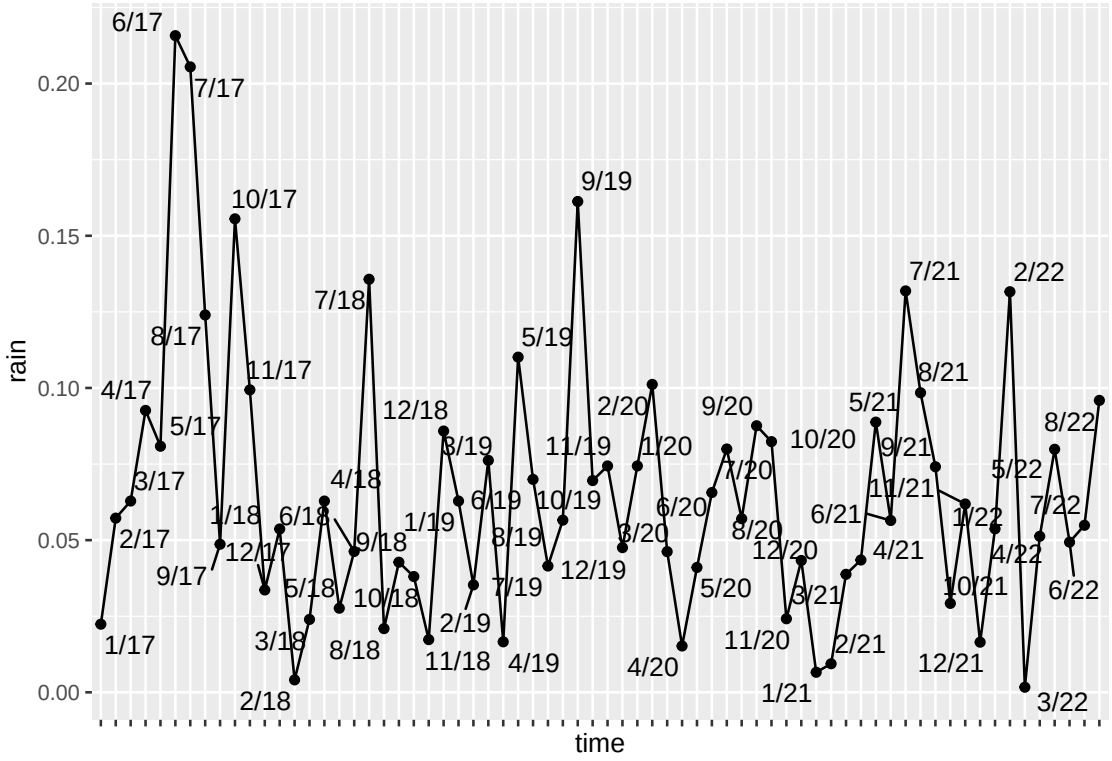
```
ggplot(aes(rain)) +
geom_bar()
```



The amount of precipitation in millimeters (mm) exceeding 0.5 were few and vary. It was also extremely rare to see heavy rain.

- The avarage amount of rain every month

```
change_month(solar_weather, "rain", mean)
```



It fluctuated greatly over the course of time with no clear pattern.

Base on the visualization, we can confirm that hydro electricity is not applicable in the area.

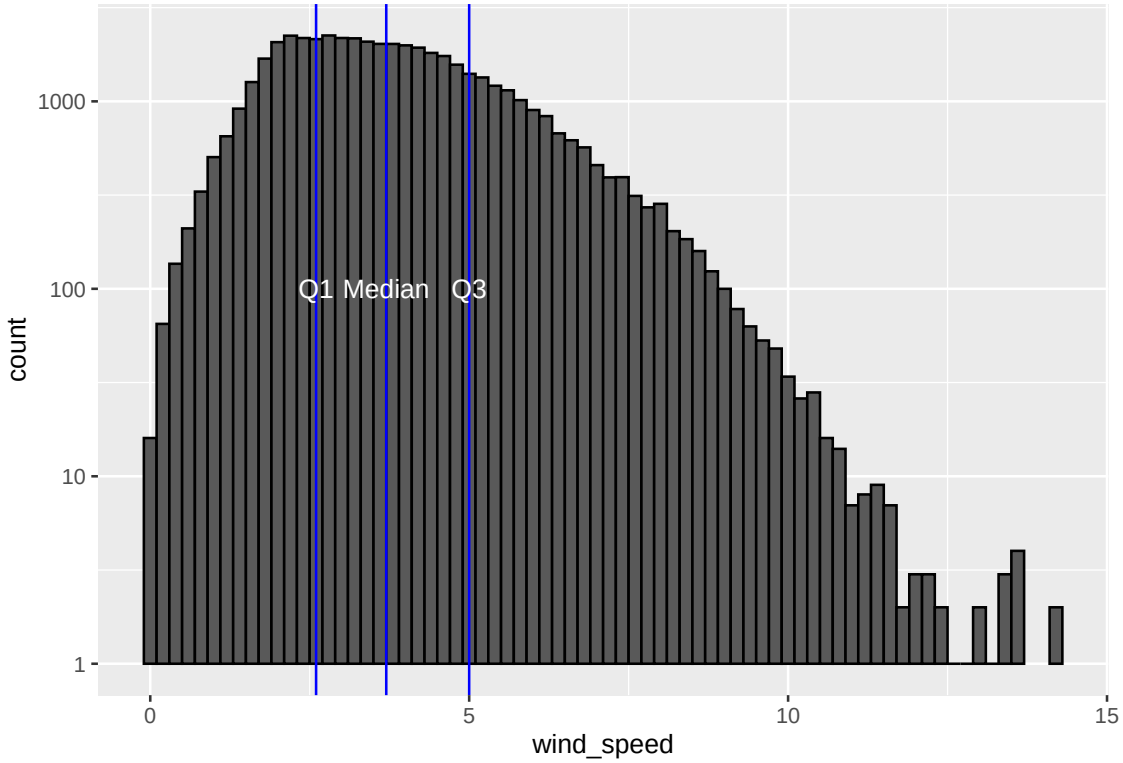
8.3 Wind energy

Similar to how rain affects hydro electricity, wind turbines rely on appropriate wind speed to run.

```
tmp <- solar_weather %>%
  select(year, month, day, hour, wind_speed) %>%
  distinct()
qt <- quantile(tmp$wind_speed, c(0.25, 0.5, 0.75))
distribution(tmp, "wind_speed", binwidth = 0.2, col = "black") +
  scale_y_continuous(trans = "log10") +
  geom_vline(xintercept = qt[["25%"]], col = "blue") +
  annotate("text", x=qt[["25%"]], y=100, label="Q1", angle=0, col = "white") +
  geom_vline(xintercept = qt[["50%"]], col = "blue") +
  annotate("text", x=qt[["50%"]], y=100, label="Median", angle=0, col = "white") +
  geom_vline(xintercept = qt[["75%"]], col = "blue") +
  annotate("text", x=qt[["75%"]], y=100, label="Q3", angle=0, col = "white") +
  ylab("count")
```

```
## Warning: Transformation introduced infinite values in continuous y-axis
```

```
## Warning: Removed 2 rows containing missing values (`geom_bar()`).
```



Only 25% of observations have wind speed of at least 5 milliliter per second. Consequently, wind energy is not applicable.

References

[1] Daryl Ronald Myers
Solar Radiation: Practical Modeling for Renewable Energy Applications

[2] Rafael A. Irizarry
Introduction to Data Science: Data Analysis and Prediction Algorithms with R

[3] Douglas C. Montgomery, George C. Runger
Applied Statistics and Probability for Engineers 6th Edition