Tetouan Power Consumption Prediction Model

Antoni Rakowski

In this notebook we will evaluate two data sets:

1. A data set containing weather conditions from Tetouan from 2017
2. A data set containing power consumption information from the same time and place

Then we will train appropriate ML models

Loading necessary data and libraries

```
library(dplyr)
```

```
##
## Attaching package: 'dplyr'

## The following objects are masked from 'package:stats':
##
##     filter, lag

## The following objects are masked from 'package:base':
##
##     intersect, setdiff, setequal, union
```

```
library(ggplot2)
library(png)
library(grid)
weather <- read.csv("TetouanWeather2017.csv", skip = 3)
power_consumption <- read.csv("TetouanPowerConsumption2017.csv")
```

Inspecting power consumption data for its structure and missing values

```
head(power_consumption)
```

```
##        Datetime GeneralDiffuseFlows DiffuseFlows PowerConsumption_Zone1
## 1 1/1/2017 0:00               0.051        0.119               34055.70
## 2 1/1/2017 0:10               0.070        0.085               29814.68
## 3 1/1/2017 0:20               0.062        0.100               29128.10
## 4 1/1/2017 0:30               0.091        0.096               28228.86
## 5 1/1/2017 0:40               0.048        0.085               27335.70
## 6 1/1/2017 0:50               0.059        0.108               26624.81
##   PowerConsumption_Zone2 PowerConsumption_Zone3
## 1               16128.88               20240.96
## 2               19375.08               20131.08
## 3               19006.69               19668.43
## 4               18361.09               18899.28
## 5               17872.34               18442.41
## 6               17416.41               18130.12
```

```
summary(power_consumption)
```

```
##    Datetime         GeneralDiffuseFlows  DiffuseFlows
##  Length:52416       Min.   :   0.004    Min.   :  0.011
##  Class :character   1st Qu.:   0.062    1st Qu.:  0.122
##  Mode  :character   Median :   5.035    Median :  4.456
##                     Mean   : 182.697    Mean   : 75.028
##                     3rd Qu.: 319.600    3rd Qu.:101.000
##                     Max.   :1163.000    Max.   :936.000
##  PowerConsumption_Zone1 PowerConsumption_Zone2 PowerConsumption_Zone3
##  Min.   :13896          Min.   : 8560          Min.   : 5935
##  1st Qu.:26311          1st Qu.:16981          1st Qu.:13129
##  Median :32266          Median :20823          Median :16415
##  Mean   :32345          Mean   :21043          Mean   :17835
##  3rd Qu.:37309          3rd Qu.:24714          3rd Qu.:21624
##  Max.   :52204          Max.   :37409          Max.   :47598
```

No missing values

Inspecting weather data for its structure and missing values

```r
head(weather)
```

```
##                time temperature_2m...C. relative_humidity_2m....
## 1 2017-01-01T00:00                10.9                       88
## 2 2017-01-01T01:00                11.0                       77
## 3 2017-01-01T02:00                10.7                       79
## 4 2017-01-01T03:00                10.0                       82
## 5 2017-01-01T04:00                 9.7                       83
## 6 2017-01-01T05:00                 9.5                       83
##   dew_point_2m...C. apparent_temperature...C. precipitation..mm. rain..mm.
## 1               9.0                       8.1                  0         0
## 2               7.1                       8.8                  0         0
## 3               7.3                       8.7                  0         0
## 4               7.2                       8.3                  0         0
## 5               7.0                       8.0                  0         0
## 6               6.7                       7.8                  0         0
##   pressure_msl..hPa. surface_pressure..hPa. cloud_cover.... cloud_cover_low....
## 1             1028.5                 1014.9              10                  11
## 2             1028.7                 1015.1               4                   4
## 3             1028.3                 1014.7              23                  20
## 4             1027.9                 1014.2              26                  19
## 5             1027.5                 1013.8              16                  11
## 6             1027.0                 1013.3              38                  25
##   cloud_cover_mid.... cloud_cover_high.... et0_fao_evapotranspiration..mm.
## 1                   0                    0                            0.01
## 2                   0                    0                            0.01
## 3                   0                   17                            0.01
## 4                   0                   30                            0.00
## 5                   0                   21                            0.00
## 6                   0                   51                            0.00
##   vapour_pressure_deficit..kPa. wind_speed_10m..km.h. wind_speed_100m..km.h.
## 1                          0.16                  17.9                   28.1
## 2                          0.30                  10.3                   21.9
## 3                          0.27                   9.1                   19.8
## 4                          0.22                   7.2                   17.4
## 5                          0.20                   6.6                   16.4
## 6                          0.21                   6.4                   16.5
```

```
##   wind_direction_10m.... wind_direction_100m.... wind_gusts_10m..km.h.
## 1                    100                     100                  36.7
## 2                    115                     110                  27.0
## 3                    124                     115                  25.6
## 4                    127                     117                  22.7
## 5                    131                     119                  18.7
## 6                    133                     122                  18.0
##   soil_temperature_0_to_7cm...C.
## 1                           11.8
## 2                           11.7
## 3                           11.5
## 4                           11.2
## 5                           10.8
## 6                           10.6
```

```r
summary(weather)
```

```
##      time           temperature_2m...C. relative_humidity_2m....
##  Length:8760        Min.   : 2.50       Min.   : 15.00
##  Class :character   1st Qu.:13.70       1st Qu.: 68.00
##  Mode  :character   Median :17.60       Median : 80.00
##                     Mean   :17.59       Mean   : 76.88
##                     3rd Qu.:21.50       3rd Qu.: 91.00
##                     Max.   :36.50       Max.   :100.00
##  dew_point_2m...C. apparent_temperature...C. precipitation..mm.
##  Min.   :-2.20     Min.   :-0.50             Min.   :0.00000
##  1st Qu.: 9.40     1st Qu.:11.60             1st Qu.:0.00000
##  Median :12.70     Median :17.20             Median :0.00000
##  Mean   :13.01     Mean   :17.28             Mean   :0.05153
##  3rd Qu.:17.50     3rd Qu.:23.00             3rd Qu.:0.00000
##  Max.   :23.10     Max.   :37.70             Max.   :7.40000
##    rain..mm.       pressure_msl..hPa. surface_pressure..hPa. cloud_cover....
##  Min.   :0.00000   Min.   : 997       Min.   : 983.9         Min.   :  0.00
##  1st Qu.:0.00000   1st Qu.:1015       1st Qu.:1002.3         1st Qu.:  1.00
##  Median :0.00000   Median :1018       Median :1004.8         Median : 25.00
##  Mean   :0.05153   Mean   :1019       Mean   :1005.4         Mean   : 34.98
##  3rd Qu.:0.00000   3rd Qu.:1022       3rd Qu.:1008.2         3rd Qu.: 62.00
##  Max.   :7.40000   Max.   :1038       Max.   :1024.0         Max.   :100.00
##  cloud_cover_low.... cloud_cover_mid.... cloud_cover_high....
##  Min.   :  0.00      Min.   :  0.000     Min.   :  0.00
##  1st Qu.:  0.00      1st Qu.:  0.000     1st Qu.:  0.00
##  Median :  7.00      Median :  0.000     Median :  0.00
##  Mean   : 28.43      Mean   :  8.037     Mean   : 21.16
##  3rd Qu.: 50.00      3rd Qu.:  1.000     3rd Qu.: 33.00
##  Max.   :100.00      Max.   :100.000     Max.   :100.00
##  et0_fao_evapotranspiration..mm. vapour_pressure_deficit..kPa.
##  Min.   :0.0000                  Min.   :0.0000
##  1st Qu.:0.0000                  1st Qu.:0.1500
##  Median :0.0400                  Median :0.3700
##  Mean   :0.1381                  Mean   :0.5529
##  3rd Qu.:0.2400                  3rd Qu.:0.7200
##  Max.   :0.8300                  Max.   :4.9100
##  wind_speed_10m..km.h. wind_speed_100m..km.h. wind_direction_10m....
##  Min.   : 0.00         Min.   : 0.00          Min.   :  2
##  1st Qu.: 6.60         1st Qu.:11.30          1st Qu.: 88
```

```
##  Median :10.90        Median :19.00         Median :103
##  Mean   :11.75        Mean   :19.12         Mean   :156
##  3rd Qu.:16.00        3rd Qu.:25.60         3rd Qu.:243
##  Max.   :37.80        Max.   :60.10         Max.   :360
##  wind_direction_100m.... wind_gusts_10m..km.h. soil_temperature_0_to_7cm...C.
##  Min.   :  2.0          Min.   :  1.80        Min.   : 5.40
##  1st Qu.: 91.0          1st Qu.: 19.10        1st Qu.:13.70
##  Median :103.0          Median : 29.90        Median :18.30
##  Mean   :154.9          Mean   : 31.57        Mean   :18.36
##  3rd Qu.:248.0          3rd Qu.: 42.10        3rd Qu.:22.90
##  Max.   :360.0          Max.   :106.20        Max.   :33.90
```

No missing values

Adjusting the date format to enable inner joining

```r
power_consumption %>% mutate(Datetime =
format(strptime(Datetime, "%m/%d/%Y %H:%M"), "%Y-%m-%dT%H:00")) ->
power_date_converted
```

```r
head(power_date_converted)
```

```
##           Datetime GeneralDiffuseFlows DiffuseFlows PowerConsumption_Zone1
## 1 2017-01-01T00:00               0.051        0.119               34055.70
## 2 2017-01-01T00:00               0.070        0.085               29814.68
## 3 2017-01-01T00:00               0.062        0.100               29128.10
## 4 2017-01-01T00:00               0.091        0.096               28228.86
## 5 2017-01-01T00:00               0.048        0.085               27335.70
## 6 2017-01-01T00:00               0.059        0.108               26624.81
##   PowerConsumption_Zone2 PowerConsumption_Zone3
## 1               16128.88               20240.96
## 2               19375.08               20131.08
## 3               19006.69               19668.43
## 4               18361.09               18899.28
## 5               17872.34               18442.41
## 6               17416.41               18130.12
```

We can see that duplicate datetime values have been created due to the rounding of minutes. Let's take the mean values of each hour

```r
power_date_converted %>%
group_by(Datetime) %>% summarise(across(everything(), mean)) -> power_mean
```

```r
head(power_mean)
```

```
## # A tibble: 6 x 6
##   Datetime         GeneralDiffuseFlows DiffuseFlows PowerConsumption_Zone1
##   <chr>                          <dbl>        <dbl>                  <dbl>
## 1 2017-01-01T00:00              0.0635       0.0988                 29198.
## 2 2017-01-01T01:00              0.0568       0.112                  24657.
## 3 2017-01-01T02:00              0.063        0.129                  22083.
## 4 2017-01-01T03:00              0.0598       0.141                  20811.
## 5 2017-01-01T04:00              0.058        0.123                  20476.
## 6 2017-01-01T05:00              0.0658       0.119                  20807.
## # i 2 more variables: PowerConsumption_Zone2 <dbl>,
## #   PowerConsumption_Zone3 <dbl>
```

Inner joining the weather and power data

```r
joined_data <- inner_join(power_mean, weather, by = c("Datetime" = "time"))
```

```r
head(joined_data)
```

```
## # A tibble: 6 x 26
##   Datetime         GeneralDiffuseFlows DiffuseFlows PowerConsumption_Zone1
##   <chr>                          <dbl>        <dbl>                  <dbl>
## 1 2017-01-01T00:00              0.0635       0.0988                 29198.
## 2 2017-01-01T01:00              0.0568       0.112                  24657.
## 3 2017-01-01T02:00              0.063        0.129                  22083.
## 4 2017-01-01T03:00              0.0598       0.141                  20811.
## 5 2017-01-01T04:00              0.058        0.123                  20476.
## 6 2017-01-01T05:00              0.0658       0.119                  20807.
## # i 22 more variables: PowerConsumption_Zone2 <dbl>,
## #   PowerConsumption_Zone3 <dbl>, temperature_2m...C. <dbl>,
## #   relative_humidity_2m.... <int>, dew_point_2m...C. <dbl>,
## #   apparent_temperature...C. <dbl>, precipitation..mm. <dbl>, rain..mm. <dbl>,
## #   pressure_msl..hPa. <dbl>, surface_pressure..hPa. <dbl>,
## #   cloud_cover.... <int>, cloud_cover_low.... <int>,
## #   cloud_cover_mid.... <int>, cloud_cover_high.... <int>, ...
```

Let's study the correlations between the power consumption values and weather conditions

```r
names(joined_data)
```

```
##  [1] "Datetime"                          "GeneralDiffuseFlows"
##  [3] "DiffuseFlows"                      "PowerConsumption_Zone1"
##  [5] "PowerConsumption_Zone2"            "PowerConsumption_Zone3"
##  [7] "temperature_2m...C."               "relative_humidity_2m...."
##  [9] "dew_point_2m...C."                 "apparent_temperature...C."
## [11] "precipitation..mm."                "rain..mm."
## [13] "pressure_msl..hPa."                "surface_pressure..hPa."
## [15] "cloud_cover...."                   "cloud_cover_low...."
## [17] "cloud_cover_mid...."               "cloud_cover_high...."
## [19] "et0_fao_evapotranspiration..mm." "vapour_pressure_deficit..kPa."
## [21] "wind_speed_10m..km.h."             "wind_speed_100m..km.h."
## [23] "wind_direction_10m...."            "wind_direction_100m...."
## [25] "wind_gusts_10m..km.h."             "soil_temperature_0_to_7cm...C."
```

```r
check_correlation <- function(data, i1, i2, i3, i4) {
  for (x in names(data)[i1:i2]) {
    for (y in names(data)[i3:i4]) {
      correlation <- cor(data[[x]], data[[y]], method = "spearman")
      if (is.na(correlation)) next
      if (abs(correlation) >= 0.50) {
        cat(sprintf("Correlation of %f between %s and %s.\n",
        correlation, x, y))
        next
      }
    }
  }
}
```

```r
check_correlation(joined_data, 2, 6, 7, 26)
```

```
## Correlation of 0.542703 between GeneralDiffuseFlows and temperature_2m...C..
```

```
## Correlation of -0.535597 between GeneralDiffuseFlows and relative_humidity_2m.....
## Correlation of 0.826942 between GeneralDiffuseFlows and et0_fao_evapotranspiration..mm..
## Correlation of 0.634602 between GeneralDiffuseFlows and vapour_pressure_deficit..kPa..
## Correlation of 0.757191 between DiffuseFlows and et0_fao_evapotranspiration..mm..
## Correlation of 0.547380 between DiffuseFlows and vapour_pressure_deficit..kPa..
```

Conclusion 1: General diffuse flows seem to be influenced by the evapotranspiration, vapour pressure deficits,temperature and relative humidity

Conclusion 2: Diffuse flows seem to be influenced by the evapotranspiration rate and vapour pressure deficits

Due to more and stronger correlations we will focus on general diffuse flows

```
joined_data %>% select(GDF = GeneralDiffuseFlows,
ET = et0_fao_evapotranspiration..mm.,
VPD = vapour_pressure_deficit..kPa., TEMP = temperature_2m...C.,
HUM = relative_humidity_2m....) -> correlated_data
```

```
head(correlated_data)
```

```
## # A tibble: 6 x 5
##      GDF    ET   VPD  TEMP   HUM
##    <dbl> <dbl> <dbl> <dbl> <int>
## 1 0.0635  0.01  0.16  10.9    88
## 2 0.0568  0.01  0.3   11      77
## 3 0.063   0.01  0.27  10.7    79
## 4 0.0598  0     0.22  10      82
## 5 0.058   0     0.2    9.7    83
## 6 0.0658  0     0.21   9.5    83
```

Creating new features to find new correlations and strengthen the existing ones (that is: VPD, TEMP, HUM)

```
new_features <- correlated_data %>%
  mutate(
    # Sum of two features
    VPD_TEMP_sum = VPD + TEMP,
    VPD_HUM_sum = VPD + HUM,
    TEMP_HUM_sum = TEMP + HUM,

    # Difference of two features
    VPD_TEMP_diff = VPD - TEMP,
    VPD_HUM_diff = VPD - HUM,
    TEMP_HUM_diff = TEMP - HUM,

    # Product of two features
    VPD_TEMP_product = VPD * TEMP,
    VPD_HUM_product = VPD * HUM,
    TEMP_HUM_product = TEMP * HUM,

    # Ratio of two features (adding a small constant to avoid division by 0)
    VPD_TEMP_ratio = VPD / (TEMP + 1e-6),
    VPD_HUM_ratio = VPD / (HUM + 1e-6),
    TEMP_HUM_ratio = TEMP / (HUM + 1e-6),

    # Sum of all three features
    all_three_sum = VPD + TEMP + HUM,
```

```r
    # Difference between the sum of two features and the third feature
    VPD_TEMP_sum_minus_HUM = (VPD + TEMP) - HUM,
    VPD_HUM_sum_minus_TEMP = (VPD + HUM) - TEMP,
    TEMP_HUM_sum_minus_VPD = (TEMP + HUM) - VPD,

    # Product of all three features
    all_three_product = VPD * TEMP * HUM,

    # Average of two features
    VPD_TEMP_average = (VPD + TEMP) / 2,
    VPD_HUM_average = (VPD + HUM) / 2,
    TEMP_HUM_average = (TEMP + HUM) / 2,

    # Square of a feature
    VPD_square = VPD * VPD,
    TEMP_square = TEMP * TEMP,
    HUM_square = HUM * HUM,

    # Interaction term (adding a small constant to avoid division by 0)
    VPD_TEMP_interaction = (VPD * TEMP) / (HUM + 1e-6),
    VPD_HUM_interaction = (VPD * HUM) / (TEMP + 1e-6),
    TEMP_HUM_interaction = (TEMP * HUM) / (VPD + 1e-6),

    # Data from previous hour
    GDF_prev = lag(GDF, 1),
    DF_prev = lag(joined_data$DiffuseFlows, 1)
  ) %>% slice(-1)
```

```r
head(new_features)
```

```
## # A tibble: 6 x 33
##      GDF    ET   VPD  TEMP   HUM VPD_TEMP_sum VPD_HUM_sum TEMP_HUM_sum
##    <dbl> <dbl> <dbl> <dbl> <int>        <dbl>       <dbl>        <dbl>
## 1 0.0568  0.01  0.3   11      77        11.3         77.3         88
## 2 0.063   0.01  0.27  10.7    79        11.0         79.3         89.7
## 3 0.0598  0     0.22  10      82        10.2         82.2         92
## 4 0.058   0     0.2    9.7    83         9.9         83.2         92.7
## 5 0.0658  0     0.21   9.5    83         9.71        83.2         92.5
## 6 0.0617  0     0.21   9.3    82         9.51        82.2         91.3
## # i 25 more variables: VPD_TEMP_diff <dbl>, VPD_HUM_diff <dbl>,
## #   TEMP_HUM_diff <dbl>, VPD_TEMP_product <dbl>, VPD_HUM_product <dbl>,
## #   TEMP_HUM_product <dbl>, VPD_TEMP_ratio <dbl>, VPD_HUM_ratio <dbl>,
## #   TEMP_HUM_ratio <dbl>, all_three_sum <dbl>, VPD_TEMP_sum_minus_HUM <dbl>,
## #   VPD_HUM_sum_minus_TEMP <dbl>, TEMP_HUM_sum_minus_VPD <dbl>,
## #   all_three_product <dbl>, VPD_TEMP_average <dbl>, VPD_HUM_average <dbl>,
## #   TEMP_HUM_average <dbl>, VPD_square <dbl>, TEMP_square <dbl>, ...
```

```r
names(new_features)
```

```
##  [1] "GDF"                    "ET"                     "VPD"
##  [4] "TEMP"                   "HUM"                    "VPD_TEMP_sum"
##  [7] "VPD_HUM_sum"            "TEMP_HUM_sum"           "VPD_TEMP_diff"
## [10] "VPD_HUM_diff"           "TEMP_HUM_diff"          "VPD_TEMP_product"
## [13] "VPD_HUM_product"        "TEMP_HUM_product"       "VPD_TEMP_ratio"
## [16] "VPD_HUM_ratio"          "TEMP_HUM_ratio"         "all_three_sum"
```

```
## [19] "VPD_TEMP_sum_minus_HUM" "VPD_HUM_sum_minus_TEMP" "TEMP_HUM_sum_minus_VPD"
## [22] "all_three_product"      "VPD_TEMP_average"       "VPD_HUM_average"
## [25] "TEMP_HUM_average"       "VPD_square"             "TEMP_square"
## [28] "HUM_square"             "VPD_TEMP_interaction"   "VPD_HUM_interaction"
## [31] "TEMP_HUM_interaction"   "GDF_prev"               "DF_prev"
```

```r
check_correlation(new_features, 1, 1, 3,33)
```

```
## Correlation of 0.634598 between GDF and VPD.
## Correlation of 0.542643 between GDF and TEMP.
## Correlation of -0.535587 between GDF and HUM.
## Correlation of 0.560203 between GDF and VPD_TEMP_sum.
## Correlation of -0.528005 between GDF and VPD_HUM_sum.
## Correlation of -0.521096 between GDF and VPD_TEMP_diff.
## Correlation of 0.542899 between GDF and VPD_HUM_diff.
## Correlation of 0.628027 between GDF and TEMP_HUM_diff.
## Correlation of 0.662873 between GDF and VPD_TEMP_product.
## Correlation of 0.651205 between GDF and VPD_HUM_product.
## Correlation of 0.535142 between GDF and VPD_TEMP_ratio.
## Correlation of 0.620269 between GDF and VPD_HUM_ratio.
## Correlation of 0.657121 between GDF and TEMP_HUM_ratio.
## Correlation of 0.628444 between GDF and VPD_TEMP_sum_minus_HUM.
## Correlation of -0.627582 between GDF and VPD_HUM_sum_minus_TEMP.
## Correlation of 0.668113 between GDF and all_three_product.
## Correlation of 0.560203 between GDF and VPD_TEMP_average.
## Correlation of -0.528005 between GDF and VPD_HUM_average.
## Correlation of 0.634598 between GDF and VPD_square.
## Correlation of 0.542643 between GDF and TEMP_square.
## Correlation of -0.535587 between GDF and HUM_square.
## Correlation of 0.654411 between GDF and VPD_TEMP_interaction.
## Correlation of 0.529118 between GDF and VPD_HUM_interaction.
## Correlation of -0.535138 between GDF and TEMP_HUM_interaction.
## Correlation of 0.931844 between GDF and GDF_prev.
## Correlation of 0.765385 between GDF and DF_prev.
```

The correlation between GDF and values from the previous hour is strong, but VPD, HUM, TEMP and their derivatives are insufficiently correlated. Thus, we will use ET, GDF_prev and DF_prev to predict GDF.

```r
new_features %>% select(GDF, ET, GDF_prev, DF_prev) -> correlated_data2
```

```r
head(correlated_data2)
```

```
## # A tibble: 6 x 4
##      GDF    ET GDF_prev DF_prev
##    <dbl> <dbl>    <dbl>   <dbl>
## 1 0.0568  0.01   0.0635  0.0988
## 2 0.063   0.01   0.0568  0.112
## 3 0.0598  0      0.063   0.129
## 4 0.058   0      0.0598  0.141
## 5 0.0658  0      0.058   0.123
## 6 0.0617  0      0.0658  0.119
```

Studying the influence of outliers on the correlation

```r
check_correlation_by_delta <- function(delta, i1, i2, i3, i4) {

  lower <- lapply(correlated_data2[2:3],
```

```
  function(x) quantile(x, probs = 0.00 + delta / 100))
  upper <- lapply(correlated_data2[2:3],
  function(x) quantile(x, probs = 1.00 - delta / 100))
  data <- correlated_data2

  for (x in names(lower)) {
    data <- data %>% filter(data[[x]] >= lower[[x]] & data[[x]] <= upper[[x]])
  }

  cat(sprintf("Delta = %d\n", delta))

  check_correlation(data, i1, i2, i3, i4)

  cat("\n")
}
```

```
for (delta in 0:5) check_correlation_by_delta(delta, 1, 1, 2, 3)
```

```
## Delta = 0
## Correlation of 0.826969 between GDF and ET.
## Correlation of 0.931844 between GDF and GDF_prev.
##
## Delta = 1
## Correlation of 0.821210 between GDF and ET.
## Correlation of 0.927872 between GDF and GDF_prev.
##
## Delta = 2
## Correlation of 0.816820 between GDF and ET.
## Correlation of 0.924175 between GDF and GDF_prev.
##
## Delta = 3
## Correlation of 0.811891 between GDF and ET.
## Correlation of 0.920134 between GDF and GDF_prev.
##
## Delta = 4
## Correlation of 0.807086 between GDF and ET.
## Correlation of 0.915369 between GDF and GDF_prev.
##
## Delta = 5
## Correlation of 0.802791 between GDF and ET.
## Correlation of 0.912848 between GDF and GDF_prev.
```

The effect of outliers is marginal, but slightly negative. Hence, we will not remove outliers. Let's normalise the data on account of plotting and fitting ML algorithms

```
min_max <- function(data) {
  (data - min(data)) / (max(data) - min(data))
}
```

```
correlated_data2 %>% mutate(GDF = min_max(GDF), ET = min_max(ET),
GDF_prev = min_max(GDF_prev), DF_prev = min_max(DF_prev)) -> normalised_data
```

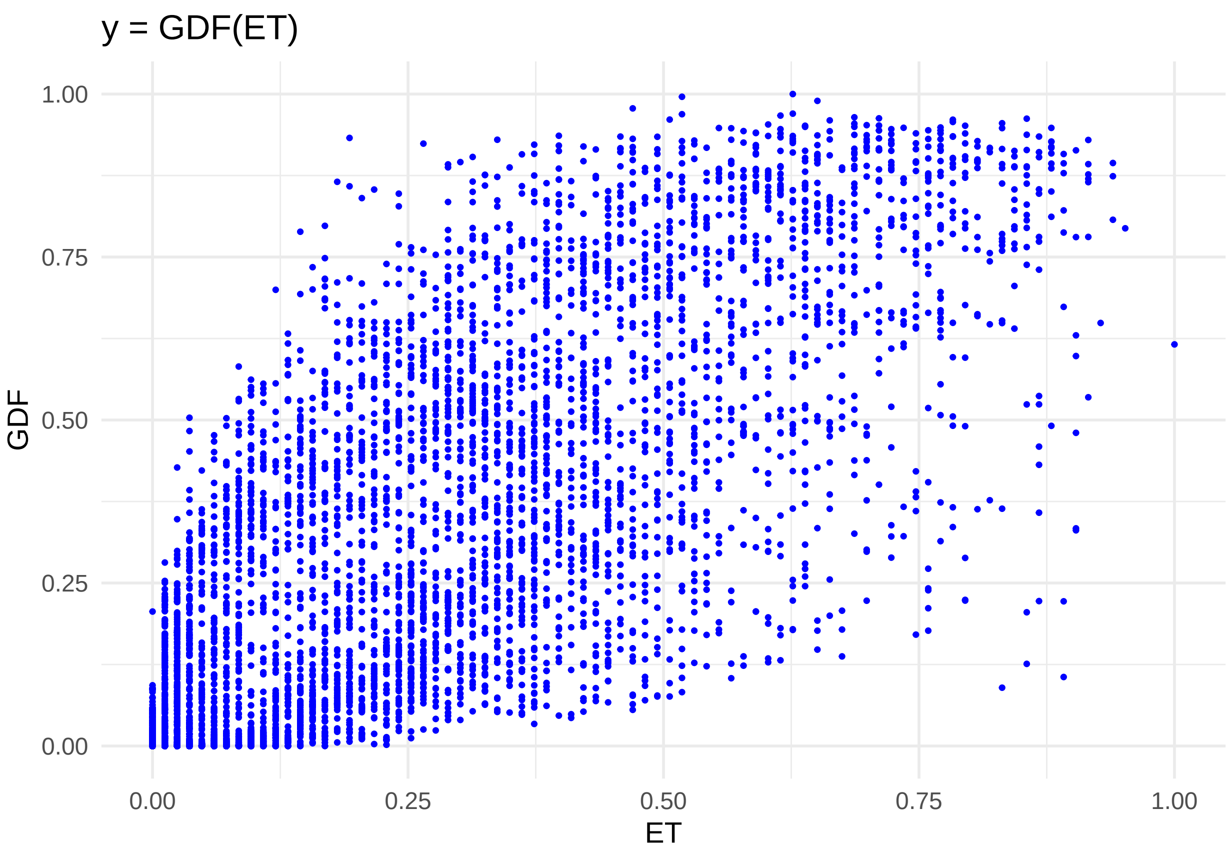
```
head(normalised_data)
```

```
## # A tibble: 6 x 4
##         GDF     ET   GDF_prev   DF_prev
```

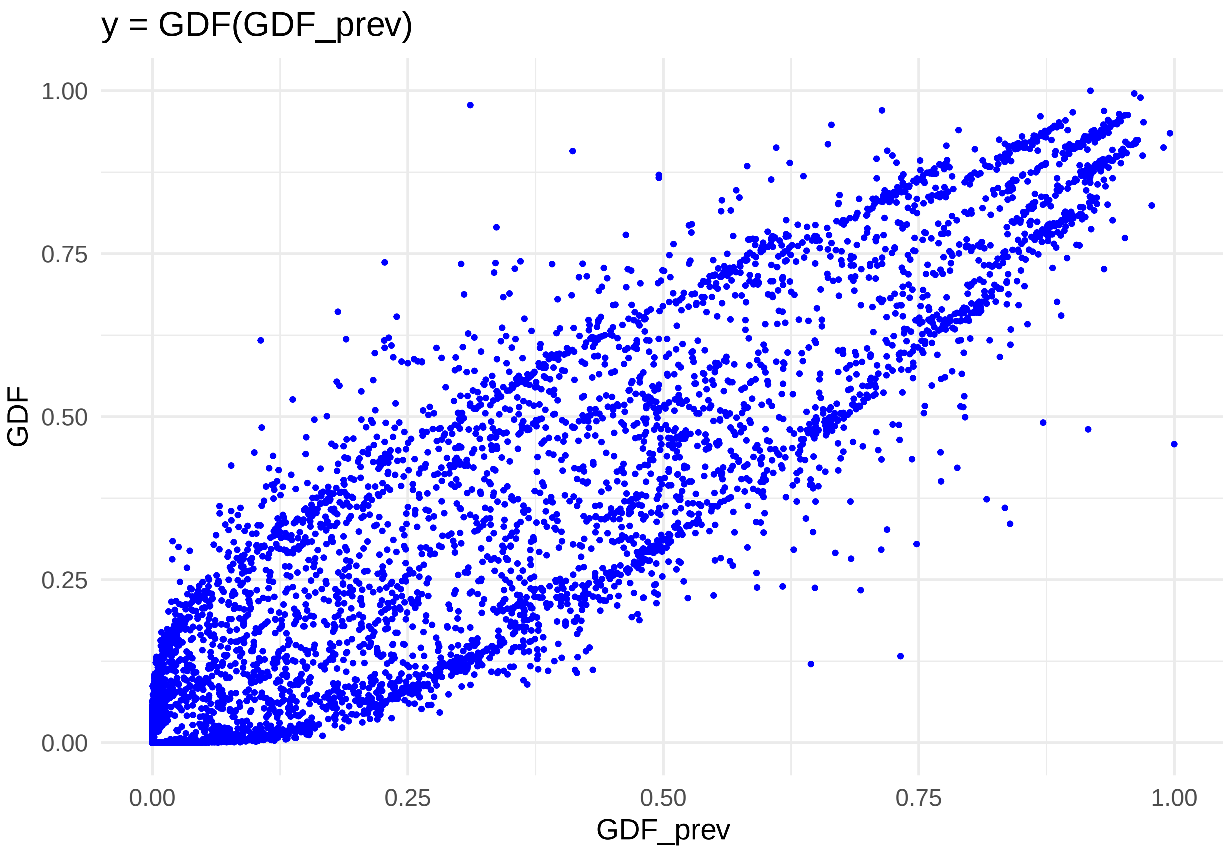
```
##       <dbl>  <dbl>     <dbl>     <dbl>
## 1 0.0000397 0.0120 0.0000467 0.0000683
## 2 0.0000462 0.0120 0.0000397 0.0000842
## 3 0.0000428 0      0.0000462 0.000104
## 4 0.0000409 0      0.0000428 0.000117
## 5 0.0000491 0      0.0000409 0.0000962
## 6 0.0000448 0      0.0000491 0.0000916
```

Plotting the data

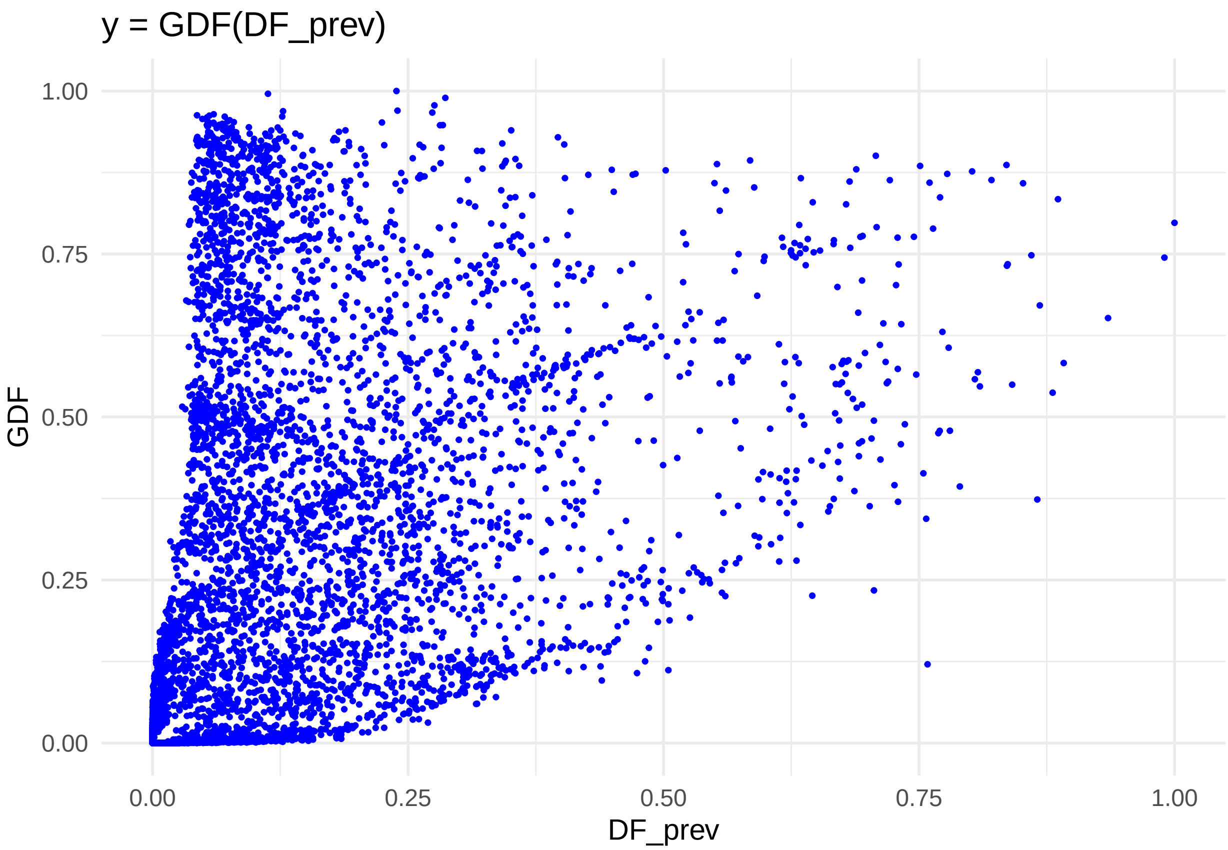
```
plot1 <- ggplot(normalised_data, aes(x = ET, y = GDF)) +
geom_point(size = 0.5, color = "blue") +
labs(title = "y = GDF(ET)", x ="ET", y = "GDF") +
theme_minimal()
print(plot1)
```




```
plot2 <- ggplot(normalised_data, aes(x = GDF_prev, y = GDF)) +
geom_point(size = 0.5, color = "blue") +
labs(title = "y = GDF(GDF_prev)", x ="GDF_prev", y = "GDF") +
theme_minimal()
print(plot2)
```




```
plot3 <- ggplot(normalised_data, aes(x = DF_prev, y = GDF)) +
geom_point(size = 0.5, color = "blue") +
labs(title = "y = GDF(DF_prev)", x ="DF_prev", y = "GDF") +
theme_minimal()
print(plot3)
```



The data is clustered, therefore we will try the following: random forest regressor, gradient boosting, k-nearest neighbours regressor and support vector regressor. First we must save the data to a csv file and switch to Python

```
normalised_data %>% mutate(GDF = format(GDF, scientific = FALSE),
ET = format(ET, scientific = FALSE),
GDF_prev = format(GDF_prev, scientific = FALSE),
DF_prev = format(DF_prev, scientific = FALSE)) -> data_to_save
```

```
head(data_to_save)
```

```
## # A tibble: 6 x 4
##   GDF             ET         GDF_prev        DF_prev
##   <chr>           <chr>      <chr>           <chr>
## 1 0.0000396854118 0.01204819 0.0000466784359 0.0000683345723
## 2 0.0000461539591 0.01204819 0.0000396854118 0.0000842083256
## 3 0.0000428322727 0.00000000 0.0000461539591 0.0001035665614
## 4 0.0000409091910 0.00000000 0.0000428322727 0.0001173109088
## 5 0.0000491259944 0.00000000 0.0000409091910 0.0000962104318
## 6 0.0000447553543 0.00000000 0.0000491259944 0.0000915644552
```

```
write.csv(data_to_save, "NormalisedData.csv", row.names = FALSE)
```

Switching to Python and importing necessary libraries

```
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.neighbors import KNeighborsRegressor
from sklearn.svm import SVR
from sklearn.model_selection import train_test_split
```

```
from sklearn.metrics import mean_squared_error, r2_score
import pandas as pd
import matplotlib.pyplot as plt
```

Loading the data into a Panda's dataframe

```
data = pd.read_csv("NormalisedData.csv", header = 0)
data.head()
```

```
##         GDF        ET  GDF_prev   DF_prev
## 0  0.000040  0.012048  0.000047  0.000068
## 1  0.000046  0.012048  0.000040  0.000084
## 2  0.000043  0.000000  0.000046  0.000104
## 3  0.000041  0.000000  0.000043  0.000117
## 4  0.000049  0.000000  0.000041  0.000096
```

Creating features and targets

```
Target = data[["GDF"]].values.ravel()
Target
```

```
## array([3.96854118e-05, 4.61539591e-05, 4.28322727e-05, ...,
##        5.68183209e-05, 4.16084934e-05, 4.33567495e-05])
```

```
Feature = data[["ET", "GDF_prev", "DF_prev"]]
Feature.head()
```

```
##          ET  GDF_prev   DF_prev
## 0  0.012048  0.000047  0.000068
## 1  0.012048  0.000040  0.000084
## 2  0.000000  0.000046  0.000104
## 3  0.000000  0.000043  0.000117
## 4  0.000000  0.000041  0.000096
```

Splitting data into training and testing sets

```
X_train, X_test, y_train, y_test = train_test_split(Feature, Target,
test_size = 0.2, random_state = 666)
```

Training models

```
knn_regressor = KNeighborsRegressor(n_neighbors = 5)
knn_regressor.fit(X_train, y_train)
```

```
## KNeighborsRegressor()
```

```
knn_predictions = knn_regressor.predict(X_test)

gb_regressor = GradientBoostingRegressor(n_estimators = 100, random_state = 666)
gb_regressor.fit(X_train, y_train)
```

```
## GradientBoostingRegressor(random_state=666)
```

```
gb_predictions = gb_regressor.predict(X_test)

rf_regressor = RandomForestRegressor(n_estimators = 100, random_state = 666)
rf_regressor.fit(X_train, y_train)
```

```
## RandomForestRegressor(random_state=666)
```

```
rf_predictions = rf_regressor.predict(X_test)

sv_regressor = SVR(kernel="rbf")
sv_regressor.fit(X_train, y_train)
```

```
## SVR()
```

```
sv_predictions = sv_regressor.predict(X_test)
```

Evaluating models

```
knn_mse = mean_squared_error(y_test, knn_predictions)
knn_r2 = r2_score(y_test, knn_predictions)
print(f'KNeighborsRegressor MSE: {knn_mse}')
```

```
## KNeighborsRegressor MSE: 0.008764658054074377
```

```
print(f'KNeighborsRegressor R2: {knn_r2}')
```

```
## KNeighborsRegressor R2: 0.883063493319165
```

```
print()
```

```
gb_mse = mean_squared_error(y_test, gb_predictions)
gb_r2 = r2_score(y_test, gb_predictions)
print(f'GradientBoostingRegressor MSE: {gb_mse}')
```

```
## GradientBoostingRegressor MSE: 0.008346674902375012
```

```
print(f'GradientBoostingRegressor R2: {gb_r2}')
```

```
## GradientBoostingRegressor R2: 0.8886401500819977
```

```
print()
```

```
rf_mse = mean_squared_error(y_test, rf_predictions)
rf_r2 = r2_score(y_test, rf_predictions)
print(f'RandomForestRegressor MSE: {rf_mse}')
```

```
## RandomForestRegressor MSE: 0.008361810377589658
```

```
print(f'RandomForestRegressor R2: {rf_r2}')
```

```
## RandomForestRegressor R2: 0.8884382152674692
```

```
print()
```

```
sv_mse = mean_squared_error(y_test, sv_predictions)
sv_r2 = r2_score(y_test, sv_predictions)
print(f'SupportVectorRegressor MSE: {sv_mse}')
```

```
## SupportVectorRegressor MSE: 0.012535954827181289
```

```
print(f'SupportVectorRegressor R2: {sv_r2}')
```

```
## SupportVectorRegressor R2: 0.832747523479495
```

KNeighborsRegressor, GradientBoostingRegressor, RandomForestRegressor are similarly accurateand SupportVectorRegressor is less accurate than the former three. The GradientBoostingRegressor proved to be most accurate with a slight upperhand. Let's train it again with more n_estimators

```
final_model = GradientBoostingRegressor(n_estimators = 1000, random_state = 666)
final_model.fit(X_train, y_train)
```

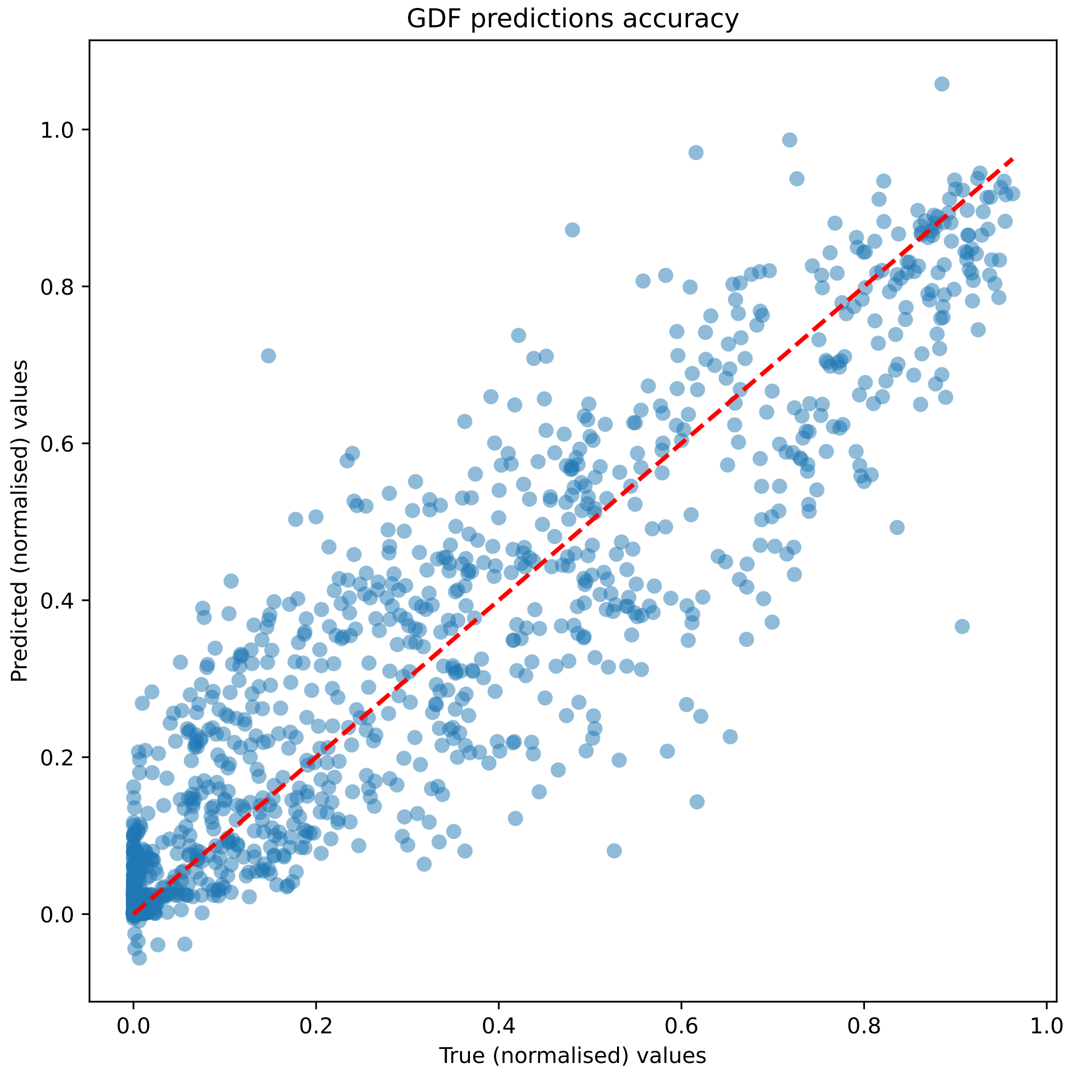
```
## GradientBoostingRegressor(n_estimators=1000, random_state=666)
```

```
final_predictions = final_model.predict(X_test)
```

Now let's create a scatter plot to demonstrate the results

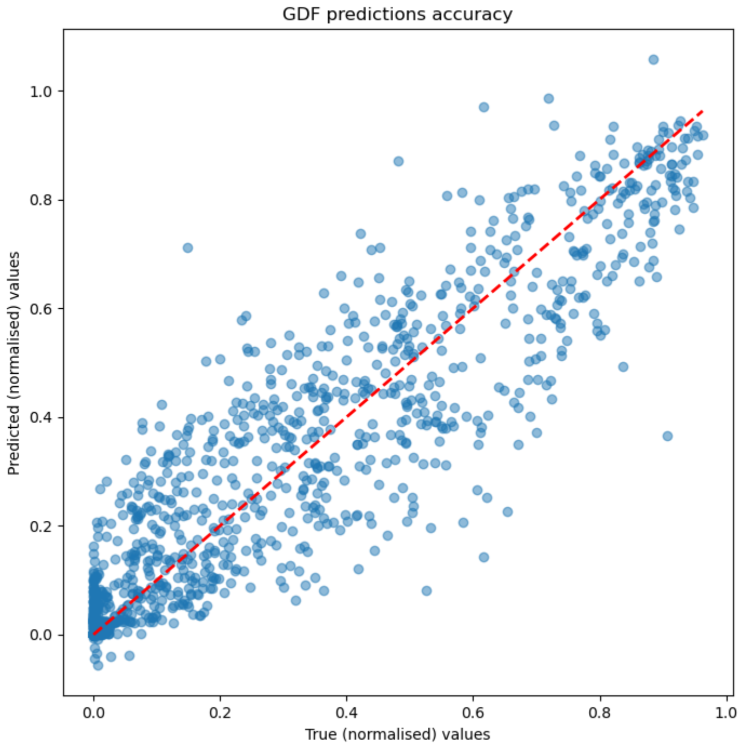
```
def plot_predictions(y_true, y_pred, title, fname):
  plt.figure(figsize=(8, 8))
  plt.scatter(y_true, y_pred, alpha = 0.5)
  plt.plot([y_true.min(), y_true.max()], [y_true.min(), y_true.max()], "r--",
  lw = 2)
  plt.xlabel("True (normalised) values")
  plt.ylabel("Predicted (normalised) values")
  plt.title(title)
  plt.savefig(fname)
```

```
plot_predictions(y_test, final_predictions, "GDF predictions accuracy",
"GDFPredictions")
```



Displaying the image in RNotebook

```
img <- readPNG("GDFPredictions.png")
grid.newpage()
grid.raster(img)
```


GDF predictions accuracy
Predicted (normalised) values
True (normalised) values
0.0
0.2
0.4
0.6
0.8
1.0