monthly_static_features_for_baseline.Rmd

0. Preparation

Load Libraries

```
# Load libraries
library(tidyverse)
library(Hmisc)
library(gridExtra)
library(finalfit)
library(stargazer)
# To create and work with tidy temporal data
library(tsibble)
# To work with date-times and time-spans
library(lubridate)
# Provides a collection of commonly used univariate/multivariate TS models
library(fable)
## To interact directly with the Quandl API and download data
library(Quandl)
# For analyzing tidy time series data.
library(feasts)
# Provides methods and tools for displaying and analyzing univariate time series forecasts library(fore
# For estimation, lag selection, diagnostic testing, forecasting, and impulse response functions of VAR
#provides tools for statistical calculations
library(stats)
# To assist the quantitative trader in the development,
#testing and deployment of statistically based trading models.
library(quantmod)
# For statistical analysis
library(car)
## To retrieve and display the information returned online by Google Trends
library(gtrendsR)
# To do time series analysis and computational finance.
library(tseries)
```

Import data from csv file

Rows: 19015 Columns: 62

Link to the handmade codebook https://docs.google.com/spreadsheets/d/1-FQsF_sxnA6iBMNpHmGovkj9Xev6BCYg2j3Poj
edit?usp=sharing

Overview of monthly dataset

```
raw_df <- read_csv("../../../data/datasets/data_monthly_v1_0.csv")
glimpse(raw_df)
```

```
## Rows: 19,015
## Columns: 62
## $ SITE_ID           <chr> "AR-SLu", "AR-SLu", "AR-SLu", "AR-SLu", "AR-SLu", "A~
## $ year              <dbl> 2010, 2010, 2010, 2010, 2010, 2010, 2010, 2010, 2010~
## $ month             <dbl> 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 1, 2, 3, 3, 4~
## $ TIMESTAMP         <dbl> 201001, 201002, 201003, 201004, 201005, 201006, 2010~
## $ dataset           <chr> "FLUXNET", "FLUXNET", "FLUXNET", "FLUXNET", "FLUXNET~
## $ SITE_IGBP         <chr> "MF", "MF", "MF", "MF", "MF", "MF", "MF", "MF", "MF"~
## $ LOCATION_LAT      <dbl> -33.4648, -33.4648, -33.4648, -33.4648, -33.4648, -3~
## $ LOCATION_LONG     <dbl> -66.4598, -66.4598, -66.4598, -66.4598, -66.4598, -6~
## $ TA_F              <dbl> 28.493, 26.673, 25.744, 18.450, 13.493, 11.273, 8.13~
## $ VPD_F             <dbl> 23.378, 14.369, 15.167, 9.185, 5.823, 5.219, 4.949, ~
## $ P_F               <dbl> 0.903, 1.986, 0.371, 0.100, 1.852, 0.030, 0.003, 0.1~
## $ NETRAD            <dbl> 188.59881, 144.21620, 125.64314, 71.50069, 41.24915,~
## $ NEE_VUT_REF       <dbl> -5.6327800, -4.4743300, -3.8928800, -3.1115900, -1.7~
## $ NEE_VUT_REF_QC    <dbl> 0.944892, 0.969494, 0.938844, 0.962500, 0.895833, 0.~
## $ NEE_CUT_REF       <dbl> -5.627700, -4.453580, -3.884050, -3.107050, -1.55985~
## $ NEE_CUT_REF_QC    <dbl> 0.948253, 0.970982, 0.938844, 0.962500, 0.913978, 0.~
## $ GPP_NT_VUT_REF    <dbl> 10.20950, 8.16307, 7.06222, 5.72781, 3.47763, 2.7738~
## $ GPP_DT_VUT_REF    <dbl> 11.91330, 9.97563, 9.00824, 6.54333, 4.15484, 3.7793~
## $ GPP_NT_CUT_REF    <dbl> 10.08900, 8.09051, 7.07681, 5.65260, 3.56473, 2.9719~
## $ GPP_DT_CUT_REF    <dbl> 11.92320, 10.16630, 9.00492, 6.60730, 4.16886, 3.691~
## $ RECO_NT_VUT_REF   <dbl> 4.46072, 3.62522, 3.18909, 2.55268, 1.78421, 2.86141~
## $ RECO_DT_VUT_REF   <dbl> 7.03163, 5.68557, 6.51721, 4.14082, 3.35165, 3.07628~
## $ RECO_NT_CUT_REF   <dbl> 4.45634, 3.61530, 3.18613, 2.54582, 1.84822, 2.83905~
## $ RECO_DT_CUT_REF   <dbl> 7.06081, 6.02964, 6.61985, 4.04346, 3.54627, 3.19315~
## $ time              <chr> "1/31/10", "2/28/10", "3/31/10", "4/30/10", "5/31/10~
## $ ET                <dbl> 9.014540, 7.677973, 5.890317, 2.345664, 2.208000, 1.~
## $ `BESS-PAR`        <dbl> 154, 120, 107, 81, 56, 48, 58, 72, 92, 124, 147, 157~
## $ `BESS-PARdiff`    <dbl> 40, 46, 31, 27, 19, 15, 15, 23, 30, 35, 39, 38, 42, ~
## $ `BESS-RSDN`       <dbl> 336, 258, 231, 175, 122, 105, 129, 158, 202, 271, 32~
## $ `CSIF-SIFdaily`   <dbl> 0.20432499, 0.14553030, 0.10980482, 0.07672890, 0.06~
## $ `CSIF-SIFinst`    <dbl> 0.5166268, 0.3872625, 0.3072417, 0.2238720, 0.200635~
## $ PET               <dbl> -0.013386652, -0.008937791, -0.008132122, -0.0067583~
## $ Ts                <dbl> 302.4697, 298.7886, 297.5482, 291.6960, 287.0565, 28~
## $ Tmean             <dbl> 300.1098, 297.2751, 296.4367, 290.6138, 286.8832, 28~
## $ prcp              <dbl> 0.002115019, 0.003131761, 0.002206154, 0.000209161, ~
## $ vpd               <dbl> 2.0661800, 1.0901145, 1.1686398, 0.9461956, 0.716290~
## $ `prcp-lag3`       <dbl> 0.008738702, 0.009724296, 0.007452934, 0.005547076, ~
## $ `ESACCI-sm`       <dbl> 0.1515208, 0.1665578, 0.1640767, 0.1240165, 0.142726~
## $ MODIS_LC          <dbl> 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 7, 9, 9, 9~
## $ b1                <dbl> 0.08442580, 0.09180000, 0.08556129, 0.08740333, 0.07~
## $ b2                <dbl> 0.2687742, 0.2524464, 0.2304000, 0.2158133, 0.198193~
## $ b3                <dbl> 0.04531612, 0.04803214, 0.04453871, 0.04530000, 0.03~
## $ b4                <dbl> 0.08052903, 0.08092143, 0.07463870, 0.07454334, 0.06~
## $ b5                <dbl> 0.3005774, 0.2924500, 0.2694322, 0.2600900, 0.231483~
## $ b6                <dbl> 0.2505258, 0.2522143, 0.2411645, 0.2348300, 0.202816~
## $ b7                <dbl> 0.15535806, 0.15945713, 0.15363870, 0.14688666, 0.12~
```

```
## $ EVI               <dbl> 0.3212592, 0.2783001, 0.2568952, 0.2292145, 0.223529~
## $ GCI               <dbl> 2.349203, 2.121655, 2.087431, 1.895962, 2.061106, 1.~
## $ NDVI              <dbl> 0.5227052, 0.4668434, 0.4583453, 0.4235885, 0.445367~
## $ NDWI              <dbl> 0.035415366, 0.000400779, -0.022856813, -0.042024087~
## $ NIRv              <dbl> 0.14050612, 0.11781442, 0.10565167, 0.09139932, 0.08~
## $ kNDVI             <dbl> 0.26745087, 0.21459042, 0.20721813, 0.17754844, 0.19~
## $ Percent_Snow      <dbl> 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0~
## $ Fpar              <dbl> 0.49, 0.43, 0.41, 0.36, 0.37, 0.33, 0.32, 0.26, 0.25~
## $ Lai               <dbl> 1.2, 0.9, 0.8, 0.5, 0.5, 0.4, 0.4, 0.3, 0.4, 0.6, 0.~
## $ LST_Day           <dbl> 313.84, 309.86, 309.18, 303.24, 296.20, 293.18, 292.~
## $ LST_Night         <dbl> 293.58, 292.96, 290.52, 286.34, 277.82, 276.80, 271.~
## $ MODIS_IGBP        <chr> "OSH", "OSH", "OSH", "OSH", "OSH", "OSH", "OSH", "OS~
## $ MODIS_PFT         <chr> "SH", "SH", "SH", "SH", "SH", "SH", "SH", "SH", "SH"~
## $ koppen_sub        <chr> "BSk", "BSk", "BSk", "BSk", "BSk", "BSk", "BSk", "BS~
## $ koppen            <chr> "Arid", "Arid", "Arid", "Arid", "Arid", "Arid", "Ari~
## $ CO2_concentration <dbl> 387.110, 387.675, 388.195, 388.905, 389.320, 389.160~
```

1. Quick EDA

Source of each observation

All the observation seems to have origin in four datasets

```
raw_df %>%
  group_by(dataset) %>%
  summarise(count=n())
```

```
## # A tibble: 4 x 2
##   dataset   count
##   <chr>     <int>
## 1 AmeriFlux  3703
## 2 FLUXNET    6614
## 3 ICOS2018    336
## 4 ICOS2020   8362
```

Sites are linked to dataset

```
raw_df %>%
  group_by(SITE_ID, dataset) %>%
  summarise(count=n())
```

```
## `summarise()` has grouped output by 'SITE_ID'. You can override using the
## `.groups` argument.
```

```
## # A tibble: 243 x 3
## # Groups:   SITE_ID [243]
##    SITE_ID dataset count
##    <chr>   <chr>   <int>
##  1 AR-SLu  FLUXNET    15
##  2 AR-Vir  FLUXNET    16
##  3 AT-Neu  FLUXNET   121
```

```
##  4 AU-Ade  FLUXNET    17
##  5 AU-ASM  FLUXNET    51
##  6 AU-Cpr  FLUXNET    48
##  7 AU-Cum  FLUXNET    25
##  8 AU-DaP  FLUXNET    55
##  9 AU-DaS  FLUXNET    74
## 10 AU-Dry  FLUXNET    45
## # ... with 233 more rows
```

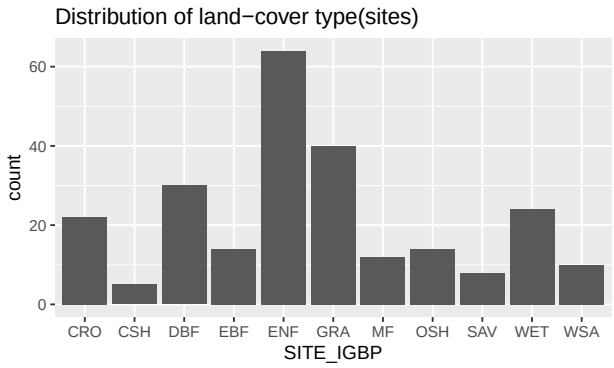
Distribution of Land-cover Type by Site

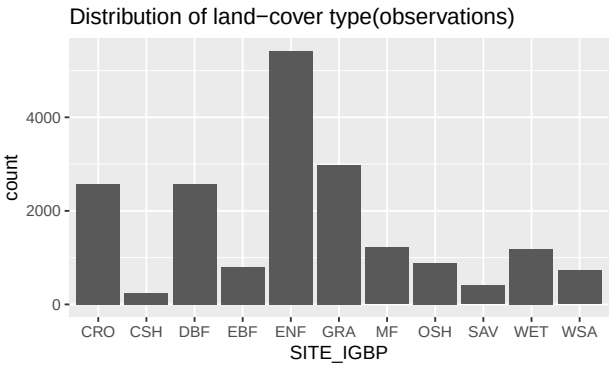
`SITE_IGBP`(Land-cover Type): 11 `SITE_ID`(Sites): 243

```
raw_df %>%
  count(SITE_ID)
```

```
## # A tibble: 243 x 2
##    SITE_ID     n
##    <chr>   <int>
##  1 AR-SLu     15
##  2 AR-Vir     16
##  3 AT-Neu    121
##  4 AU-Ade     17
##  5 AU-ASM     51
##  6 AU-Cpr     48
##  7 AU-Cum     25
##  8 AU-DaP     55
##  9 AU-DaS     74
## 10 AU-Dry     45
## # ... with 233 more rows
```

```
library(dplyr)
site_igbp_distribution <- raw_df %>%
  dplyr::select(SITE_ID, SITE_IGBP) %>%
  group_by(SITE_IGBP) %>%
  summarise(count= n())
fig1 <- raw_df %>%
  dplyr::select(SITE_ID,SITE_IGBP) %>%
  unique() %>%
  group_by(SITE_IGBP) %>%
  summarise(count=n()) %>%
  ggplot(aes(x=SITE_IGBP, y=count)) +
  geom_bar(stat='identity') +
  labs(title = "Distribution of land-cover type(sites)")
fig2 <- site_igbp_distribution %>%
  ggplot(aes(x=SITE_IGBP, y=count)) +
  geom_bar(stat='identity') +
  labs(title = "Distribution of land-cover type(observations)")
grid.arrange(fig1, fig2, nrow = 1, ncol = 2)
```






```
site_igbp_distribution
```

```
## # A tibble: 11 x 2
##    SITE_IGBP count
##    <chr>     <int>
##  1 CRO        2574
##  2 CSH         252
##  3 DBF        2582
##  4 EBF         796
##  5 ENF        5422
##  6 GRA        2972
##  7 MF         1217
##  8 OSH         886
##  9 SAV         403
## 10 WET        1186
## 11 WSA         725
```

Add features that distinguish northern/southern hemisphere

```
raw_df_hemisphere <- raw_df %>%
  mutate(hemisphere = ifelse(LOCATION_LAT >= 0, "N", "S"))
raw_df_hemisphere
```

```
## # A tibble: 19,015 x 63
##    SITE_ID  year month TIMESTAMP dataset SITE_IGBP LOCATION_LAT LOCATION_LONG
##    <chr>   <dbl> <dbl>     <dbl> <chr>   <chr>            <dbl>         <dbl>
##  1 AR-SLu   2010     1    201001 FLUXNET MF               -33.5         -66.5
##  2 AR-SLu   2010     2    201002 FLUXNET MF               -33.5         -66.5
##  3 AR-SLu   2010     3    201003 FLUXNET MF               -33.5         -66.5
##  4 AR-SLu   2010     4    201004 FLUXNET MF               -33.5         -66.5
##  5 AR-SLu   2010     5    201005 FLUXNET MF               -33.5         -66.5
##  6 AR-SLu   2010     6    201006 FLUXNET MF               -33.5         -66.5
##  7 AR-SLu   2010     7    201007 FLUXNET MF               -33.5         -66.5
##  8 AR-SLu   2010     8    201008 FLUXNET MF               -33.5         -66.5
##  9 AR-SLu   2010     9    201009 FLUXNET MF               -33.5         -66.5
## 10 AR-SLu   2010    10    201010 FLUXNET MF               -33.5         -66.5
## # ... with 19,005 more rows, and 55 more variables: TA_F <dbl>, VPD_F <dbl>,
## #   P_F <dbl>, NETRAD <dbl>, NEE_VUT_REF <dbl>, NEE_VUT_REF_QC <dbl>,
```

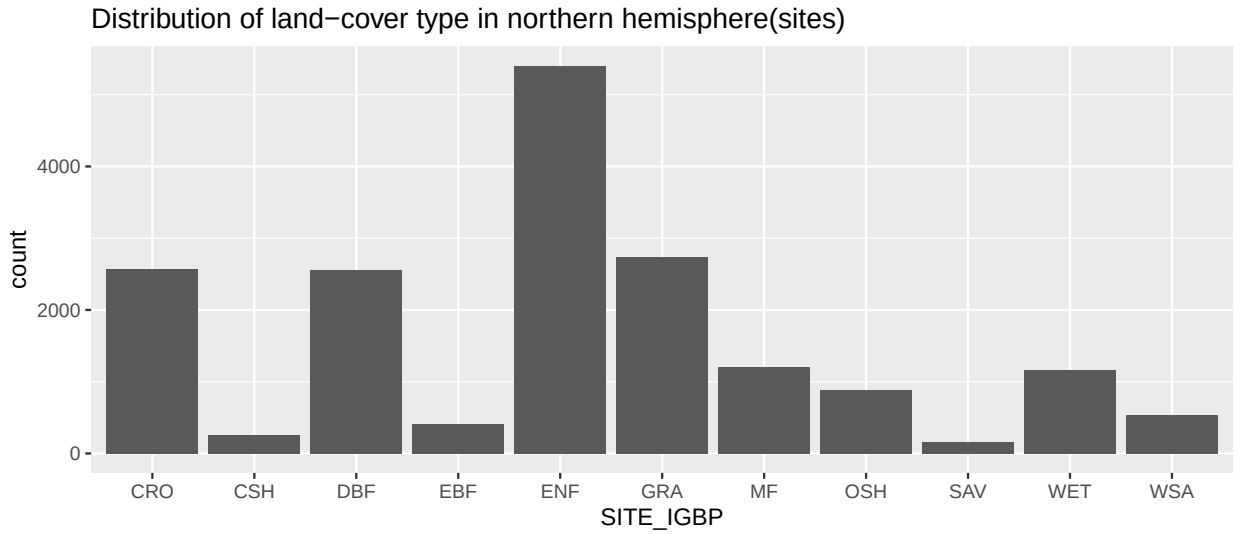
```
## #   NEE_CUT_REF <dbl>, NEE_CUT_REF_QC <dbl>, GPP_NT_VUT_REF <dbl>,
## #   GPP_DT_VUT_REF <dbl>, GPP_NT_CUT_REF <dbl>, GPP_DT_CUT_REF <dbl>,
## #   RECO_NT_VUT_REF <dbl>, RECO_DT_VUT_REF <dbl>, RECO_NT_CUT_REF <dbl>,
## #   RECO_DT_CUT_REF <dbl>, time <chr>, ET <dbl>, `BESS-PAR` <dbl>,
## #   `BESS-PARdiff` <dbl>, `BESS-RSDN` <dbl>, `CSIF-SIFdaily` <dbl>, ...
```

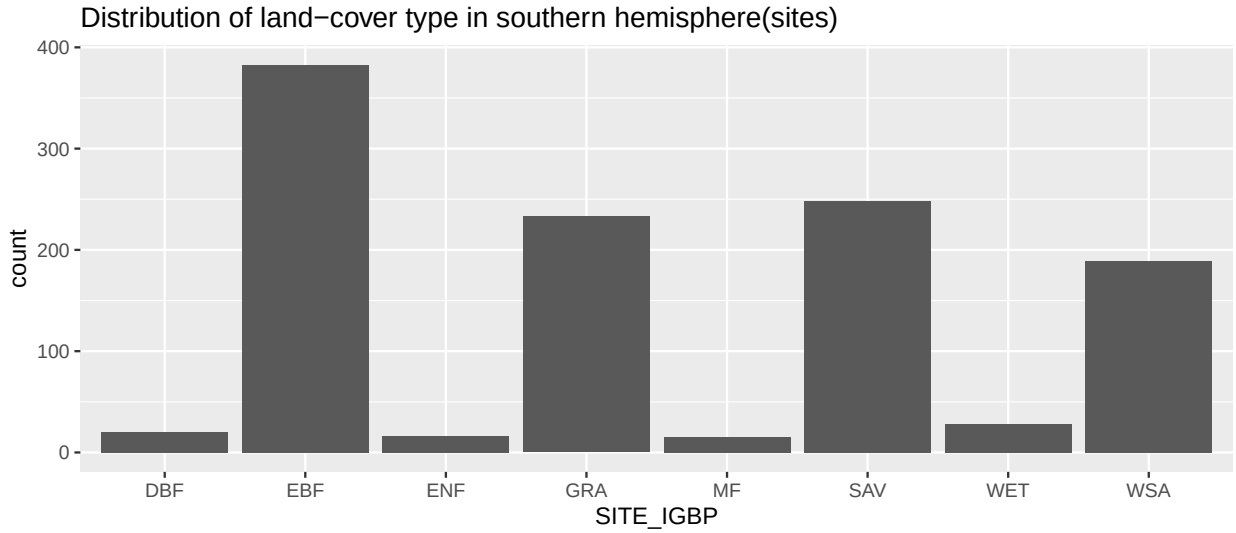
```
library(dplyr)
site_igbp_distribution <- raw_df_hemisphere %>%
  dplyr::select(SITE_ID, SITE_IGBP, hemisphere) %>%
  group_by(SITE_IGBP, hemisphere) %>%
  summarise(count= n())
```

```
## `summarise()` has grouped output by 'SITE_IGBP'. You can override using the
## `.groups` argument.
```

```
fig1 <- site_igbp_distribution %>%
subset(hemisphere == "N") %>%
ggplot(aes(x=SITE_IGBP, y=count)) +
geom_bar(stat='identity') +
labs(title = "Distribution of land-cover type in northern hemisphere(sites)")

fig2 <- site_igbp_distribution %>%
  subset(hemisphere == "S") %>%
  ggplot(aes(x=SITE_IGBP, y=count)) +
  geom_bar(stat='identity') +
  labs(title = "Distribution of land-cover type in southern hemisphere(sites)")
grid.arrange(fig1, fig2, nrow = 2, ncol = 1)
```





Observe categorical variables

Since they only have 243 rows by removing all the duplicates, all the categorical variables has unique values in each site.

```
raw_df_categorical <- raw_df_hemisphere %>%
  dplyr::select(SITE_ID, SITE_IGBP, dataset, MODIS_LC, MODIS_IGBP, MODIS_PFT,
         koppen_sub, koppen, hemisphere) %>%
  distinct() #drop duplicates

head(raw_df_categorical)
```

```
## # A tibble: 6 x 9
##   SITE_ID SITE_IGBP dataset MODIS_LC MODIS_IGBP MODIS_PFT koppen_sub koppen
##   <chr>   <chr>     <chr>      <dbl> <chr>      <chr>     <chr>      <chr>
## 1 AR-SLu  MF        FLUXNET        7 OSH        SH        BSk        Arid
## 2 AR-Vir  ENF       FLUXNET        9 SAV        SA        Cfa        Temperate
```

```
## 3 AT-Neu  GRA       FLUXNET        5 MF         MF        Dfb        Cold
## 4 AU-ASM  SAV       FLUXNET        6 CSH        SH        BWh        Arid
## 5 AU-Ade  WSA       FLUXNET       10 GRA        GRA       Aw         Tropical
## 6 AU-Cpr  SAV       FLUXNET        6 CSH        SH        BWk        Arid
## # ... with 1 more variable: hemisphere <chr>
```

```
nrow(raw_df_categorical)
```

```
## [1] 243
```

```
raw_df_categorical %>%
  distinct(koppen) %>%
  as.list()
```

```
## $koppen
## [1] "Arid"      "Temperate" "Cold"      "Tropical"  "Polar"
```

```
raw_df_categorical %>%
  distinct(MODIS_LC) %>%
  as.list()
```

```
## $MODIS_LC
##  [1]  7  9  5  6 10  1  2  8  4 12 17 16 13 11
```

```
### Unique values of categorical variables

- SITE_ID: 243
- SITE_IGBP: 11
  - MF, ENF, GRA, SAV, WSA, EBF, WET, OSH, DBF, CRO, CSH
- MODIS_LC:
  - 7  9  5  6 10  1  2  8  4 12 17 16 13 11
- MODIS_IGBP: 14
  - OSH, SAV, MF, CSH, GRA, ENF, EBF, WSA, DBF, CRO, WAT, BSV, URB, WET
- MODIS_PFT: 14
  -  SH, SA, MF, GRA, ENF, EBF, DBF, CRO, Other
- koppen_sub: 21
  - BSk, Cfa, Dfb, BWh, Aw, BWk, BSh, Csa, Cfb, Am, Dfc, Dwb, Dwa, Cwa, Dwc,
    ET, Dsb, Af, Dsc, Csb, Dfa
- koppen: 5
  - Arid, Temperate, Cold, Tropical, Polar

# 3. Convert to monthly average dataset with continuous features

## 3-1. Drop NA first and obtain monthly average data

### Create a new df of monthly average across sites(SITE_ID)
(Before dropping NA: row = 2781)


‘‘‘r
SITE_month_df <- raw_df_hemisphere %>%
```

```
  group_by(SITE_ID, SITE_IGBP, month) %>%
  drop_na() %>% # drop NA beore aggregation
  summarise(TA_F_avg = mean(TA_F, na.rm = T),VPD_F_avg = mean(VPD_F, na.rm = T),
            P_F_avg = mean(P_F, na.rm = T),NETRAD_avg = mean(NETRAD, na.rm = T),
            NEE_VUT_REF_avg = mean(NEE_VUT_REF, na.rm = T),
            NEE_VUT_REF_QC_avg = mean(NEE_VUT_REF_QC, na.rm = T),
            NEE_CUT_REF_avg = mean(NEE_CUT_REF, na.rm = T),
            NEE_CUT_REF_QC_avg = mean(NEE_CUT_REF_QC, na.rm = T),
            GPP_NT_VUT_REF_avg = mean(GPP_NT_VUT_REF, na.rm = T),
            GPP_DT_VUT_REF_avg = mean(GPP_DT_VUT_REF, na.rm = T),
            GPP_NT_CUT_REF_avg = mean(GPP_NT_CUT_REF, na.rm = T),
            GPP_DT_CUT_REF_avg = mean(GPP_DT_CUT_REF, na.rm = T),
            RECO_NT_VUT_REF_avg = mean(RECO_NT_VUT_REF, na.rm = T),
            RECO_DT_VUT_REF_avg = mean(RECO_DT_VUT_REF, na.rm = T),
            RECO_NT_CUT_REF_avg = mean(RECO_NT_CUT_REF, na.rm = T),
            RECO_DT_CUT_REF_avg = mean(RECO_DT_CUT_REF, na.rm = T),
            ET_avg = mean(ET, na.rm = T),
            `BESS-PAR_avg` = mean(`BESS-PAR`, na.rm = T),
            `BESS-PARdiff_avg` = mean(`BESS-PARdiff`, na.rm = T),
            `BESS-RSDN_avg` = mean(`BESS-RSDN`, na.rm = T),
            `CSIF-SIFdaily_avg` = mean(`CSIF-SIFdaily`, na.rm = T),
            `CSIF-SIFinst_avg` = mean(`CSIF-SIFinst`, na.rm = T),
            PET_avg = mean(PET, na.rm = T),Ts_avg = mean(Ts, na.rm = T),
            Tmean_avg = mean(Tmean, na.rm = T),
            prcp_avg = mean(prcp, na.rm = T),
            vpd_avg = mean(vpd, na.rm = T),
            `prcp-lag3_avg` = mean(`prcp-lag3`, na.rm = T),
            `ESACCI-sm_avg` = mean(`ESACCI-sm`, na.rm = T),
            b1_avg = mean(b1, na.rm = T),b2_avg = mean(b2, na.rm = T),
            b3_avg = mean(b3, na.rm = T),b4_avg = mean(b4, na.rm = T),
            b5_avg = mean(b5, na.rm = T),b6_avg = mean(b6, na.rm = T),
            b7_avg = mean(b7, na.rm = T),EVI_avg = mean(EVI, na.rm = T),
            GCI_avg = mean(GCI, na.rm = T),NDVI_avg = mean(NDVI, na.rm = T),
            NDWI_avg = mean(NDWI, na.rm = T),NIRv_avg = mean(NIRv, na.rm = T),
            kNDVI_avg = mean(kNDVI, na.rm = T),
            Percent_Snow_avg = mean(Percent_Snow, na.rm = T),
            Fpar_avg = mean(Fpar, na.rm = T),Lai_avg = mean(Lai, na.rm = T),
            LST_Day_avg = mean(LST_Day, na.rm = T),
            LST_Night_avg = mean(LST_Night, na.rm = T),
            CO2_concentration_avg = mean(CO2_concentration, na.rm = T)
                        )

## `summarise()` has grouped output by 'SITE_ID', 'SITE_IGBP'. You can override
## using the `.groups` argument.
```

```
nrow(SITE_month_df)
```

```
## [1] 2213
```

```
# Check whether new df contains NA or not
SITE_month_df_NA <- SITE_month_df[rowSums(is.na(SITE_month_df)) > 0,]
# When we drop NA first and then calculate the average, there is no missing value anymore
SITE_month_df_NA
```

```
## # A tibble: 0 x 51
## # Groups:   SITE_ID, SITE_IGBP [0]
## # ... with 51 variables: SITE_ID <chr>, SITE_IGBP <chr>, month <dbl>,
## #   TA_F_avg <dbl>, VPD_F_avg <dbl>, P_F_avg <dbl>, NETRAD_avg <dbl>,
## #   NEE_VUT_REF_avg <dbl>, NEE_VUT_REF_QC_avg <dbl>, NEE_CUT_REF_avg <dbl>,
## #   NEE_CUT_REF_QC_avg <dbl>, GPP_NT_VUT_REF_avg <dbl>,
## #   GPP_DT_VUT_REF_avg <dbl>, GPP_NT_CUT_REF_avg <dbl>,
## #   GPP_DT_CUT_REF_avg <dbl>, RECO_NT_VUT_REF_avg <dbl>,
## #   RECO_DT_VUT_REF_avg <dbl>, RECO_NT_CUT_REF_avg <dbl>, ...
```

When we drop the row first and then calculate the average, 500 rows are lost

3-2. Impute NA with average data after the aggregation(Impute is done in Python)

```
SITE_month_df_2 <- raw_df_hemisphere %>%
  group_by(SITE_ID, SITE_IGBP, month) %>% # no drop_na
  summarise(TA_F_avg = mean(TA_F, na.rm = T),VPD_F_avg = mean(VPD_F, na.rm = T),
            P_F_avg = mean(P_F, na.rm = T),NETRAD_avg = mean(NETRAD, na.rm = T),
            NEE_VUT_REF_avg = mean(NEE_VUT_REF, na.rm = T),
            NEE_VUT_REF_QC_avg = mean(NEE_VUT_REF_QC, na.rm = T),
            NEE_CUT_REF_avg = mean(NEE_CUT_REF, na.rm = T),
            NEE_CUT_REF_QC_avg = mean(NEE_CUT_REF_QC, na.rm = T),
            GPP_NT_VUT_REF_avg = mean(GPP_NT_VUT_REF, na.rm = T),
            GPP_DT_VUT_REF_avg = mean(GPP_DT_VUT_REF, na.rm = T),
            GPP_NT_CUT_REF_avg = mean(GPP_NT_CUT_REF, na.rm = T),
            GPP_DT_CUT_REF_avg = mean(GPP_DT_CUT_REF, na.rm = T),
            RECO_NT_VUT_REF_avg = mean(RECO_NT_VUT_REF, na.rm = T),
            RECO_DT_VUT_REF_avg = mean(RECO_DT_VUT_REF, na.rm = T),
            RECO_NT_CUT_REF_avg = mean(RECO_NT_CUT_REF, na.rm = T),
            RECO_DT_CUT_REF_avg = mean(RECO_DT_CUT_REF, na.rm = T),
            ET_avg = mean(ET, na.rm = T),
            `BESS-PAR_avg` = mean(`BESS-PAR`, na.rm = T),
            `BESS-PARdiff_avg` = mean(`BESS-PARdiff`, na.rm = T),
            `BESS-RSDN_avg` = mean(`BESS-RSDN`, na.rm = T),
            `CSIF-SIFdaily_avg` = mean(`CSIF-SIFdaily`, na.rm = T),
            `CSIF-SIFinst_avg` = mean(`CSIF-SIFinst`, na.rm = T),
            PET_avg = mean(PET, na.rm = T),Ts_avg = mean(Ts, na.rm = T),
            Tmean_avg = mean(Tmean, na.rm = T),
            prcp_avg = mean(prcp, na.rm = T),
            vpd_avg = mean(vpd, na.rm = T),
            `prcp-lag3_avg` = mean(`prcp-lag3`, na.rm = T),
            `ESACCI-sm_avg` = mean(`ESACCI-sm`, na.rm = T),
            b1_avg = mean(b1, na.rm = T),b2_avg = mean(b2, na.rm = T),
            b3_avg = mean(b3, na.rm = T),b4_avg = mean(b4, na.rm = T),
            b5_avg = mean(b5, na.rm = T),b6_avg = mean(b6, na.rm = T),
            b7_avg = mean(b7, na.rm = T),EVI_avg = mean(EVI, na.rm = T),
            GCI_avg = mean(GCI, na.rm = T),NDVI_avg = mean(NDVI, na.rm = T),
            NDWI_avg = mean(NDWI, na.rm = T),NIRv_avg = mean(NIRv, na.rm = T),
            kNDVI_avg = mean(kNDVI, na.rm = T),
            Percent_Snow_avg = mean(Percent_Snow, na.rm = T),
            Fpar_avg = mean(Fpar, na.rm = T),Lai_avg = mean(Lai, na.rm = T),
```

```
            LST_Day_avg = mean(LST_Day, na.rm = T),
            LST_Night_avg = mean(LST_Night, na.rm = T),
            CO2_concentration_avg = mean(CO2_concentration, na.rm = T)
                        )
```

```
## `summarise()` has grouped output by 'SITE_ID', 'SITE_IGBP'. You can override
## using the `.groups` argument.
```

Add site-unique categorical variables to monthly average dataframe

```
# Merge categorical variables
raw_df_categorical_ <- raw_df_categorical %>%
  dplyr::select(-c(SITE_IGBP)) # drop SITE_IGBP to avoid two variables in one df
raw_df_categorical_
```

```
## # A tibble: 243 x 8
##    SITE_ID dataset MODIS_LC MODIS_IGBP MODIS_PFT koppen_sub koppen    hemisphere
##    <chr>   <chr>      <dbl> <chr>      <chr>     <chr>      <chr>     <chr>
##  1 AR-SLu  FLUXNET        7 OSH        SH        BSk        Arid      S
##  2 AR-Vir  FLUXNET        9 SAV        SA        Cfa        Temperate S
##  3 AT-Neu  FLUXNET        5 MF         MF        Dfb        Cold      N
##  4 AU-ASM  FLUXNET        6 CSH        SH        BWh        Arid      S
##  5 AU-Ade  FLUXNET       10 GRA        GRA       Aw         Tropical  S
##  6 AU-Cpr  FLUXNET        6 CSH        SH        BWk        Arid      S
##  7 AU-Cum  FLUXNET        9 SAV        SA        Cfa        Temperate S
##  8 AU-DaP  FLUXNET       10 GRA        GRA       Aw         Tropical  S
##  9 AU-DaS  FLUXNET       10 GRA        GRA       Aw         Tropical  S
## 10 AU-Dry  FLUXNET       10 GRA        GRA       Aw         Tropical  S
## # ... with 233 more rows
```

```
SITE_month_df_2 <- merge(x = SITE_month_df_2, y = raw_df_categorical_,
                         by="SITE_ID")
# Add site-unique longitude/latitude to monthly df
raw_df_geo <- raw_df_hemisphere %>%
  dplyr::select(SITE_ID, LOCATION_LAT, LOCATION_LONG) %>%
  distinct() #drop duplicates
SITE_month_df_2 <- merge(x = SITE_month_df_2, y = raw_df_geo,
                         by="SITE_ID")
head(SITE_month_df_2)
```

```
##   SITE_ID SITE_IGBP month TA_F_avg VPD_F_avg P_F_avg NETRAD_avg NEE_VUT_REF_avg
## 1  AR-SLu        MF     1  27.8660   22.5575  1.3420  189.43464       -5.630970
## 2  AR-SLu        MF     2  25.6745   13.8210  3.1785  144.70720       -4.059005
## 3  AR-SLu        MF     3  24.2735   14.1460  0.6440  128.89173       -4.032335
## 4  AR-SLu        MF     4  18.4500    9.1850  0.1000   71.50069       -3.111590
## 5  AR-SLu        MF     5  13.4930    5.8230  1.8520   41.24915       -1.716330
## 6  AR-SLu        MF     6  11.2730    5.2190  0.0300   30.25793       -0.156183
##   NEE_VUT_REF_QC_avg NEE_CUT_REF_avg NEE_CUT_REF_QC_avg GPP_NT_VUT_REF_avg
## 1           0.957661       -5.609750          0.9596775          10.079865
## 2           0.970610       -4.047950          0.9713540           8.413830
```

```
## 3           0.908938       -4.034525          0.9089380           7.418585
## 4           0.962500       -3.107050          0.9625000           5.727810
## 5           0.895833       -1.559850          0.9139780           3.477630
## 6           0.943750       -0.105029          0.9263890           2.773850
##   GPP_DT_VUT_REF_avg GPP_NT_CUT_REF_avg GPP_DT_CUT_REF_avg RECO_NT_VUT_REF_avg
## 1          10.951080           9.997845          10.948160             4.32928
## 2           9.421800           8.306475           9.545365             4.23054
## 3           8.867375           7.399075           8.893495             3.38424
## 4           6.543330           5.652600           6.607300             2.55268
## 5           4.154840           3.564730           4.168860             1.78421
## 6           3.779370           2.971990           3.691950             2.86141
##   RECO_DT_VUT_REF_avg RECO_NT_CUT_REF_avg RECO_DT_CUT_REF_avg   ET_avg
## 1             5.65853            4.366645            5.765395 7.603880
## 2             5.52517            4.243325            5.750190 7.776421
## 3             5.64335            3.368570            5.805820 5.717461
## 4             4.14082            2.545820            4.043460 2.345664
## 5             3.35165            1.848220            3.546270 2.208000
## 6             3.07628            2.839050            3.193150 1.726798
##   BESS-PAR_avg BESS-PARdiff_avg BESS-RSDN_avg CSIF-SIFdaily_avg
## 1        153.5             41.0         334.5        0.16403178
## 2        121.5             43.0         262.0        0.15997129
## 3        110.5             32.5         239.0        0.14642769
## 4         81.0             27.0         175.0        0.07672890
## 5         56.0             19.0         122.0        0.06602006
## 6         48.0             15.0         105.0        0.05599389
##   CSIF-SIFinst_avg      PET_avg   Ts_avg Tmean_avg    prcp_avg   vpd_avg
## 1        0.4149796 -0.013073508 302.5744  299.8286 0.001772498 2.0366469
## 2        0.4263628 -0.009295062 298.6399  296.9365 0.002944978 1.1516008
## 3        0.4098504 -0.008685846 297.3012  295.8245 0.001617301 1.2592478
## 4        0.2238720 -0.006758340 291.6960  290.6138 0.000209161 0.9461956
## 5        0.2006351 -0.004725121 287.0565  286.8832 0.000841617 0.7162909
## 6        0.1774079 -0.003966339 284.3210  284.7615 0.000266855 0.5791388
##   prcp-lag3_avg ESACCI-sm_avg     b1_avg    b2_avg     b3_avg     b4_avg
## 1   0.006010333     0.1533844 0.09253548 0.2569774 0.04947258 0.08471935
## 2   0.007326761     0.1750476 0.08677142 0.2524607 0.04586429 0.07913214
## 3   0.006334777     0.1647253 0.07812742 0.2421113 0.04136936 0.07207096
## 4   0.005547076     0.1240165 0.08740333 0.2158133 0.04530000 0.07454334
## 5   0.003256932     0.1427260 0.07619031 0.1981935 0.03887419 0.06480969
## 6   0.001317633     0.1531552 0.07104666 0.1795600 0.03812334 0.06306333
##      b5_avg    b6_avg    b7_avg   EVI_avg  GCI_avg  NDVI_avg     NDWI_avg
## 1 0.2958339 0.2628032 0.1735823 0.3212592 2.053723 0.4703431 -0.011007202
## 2 0.2882232 0.2484911 0.1595143 0.2783001 2.199261 0.4887292  0.007994725
## 3 0.2730613 0.2337096 0.1457161 0.2568952 2.369731 0.5112432  0.017228723
## 4 0.2600900 0.2348300 0.1468867 0.2292145 1.895962 0.4235885 -0.042024087
## 5 0.2314839 0.2028161 0.1242000 0.2235292 2.061106 0.4453674 -0.010640754
## 6 0.2149600 0.1913133 0.1239800 0.2055056 1.848606 0.4330553 -0.031492900
##     NIRv_avg kNDVI_avg Percent_Snow_avg Fpar_avg Lai_avg LST_Day_avg
## 1 0.12149290 0.2202215                0    0.440    1.00      314.20
## 2 0.12356264 0.2354717                0    0.445    0.95      309.20
## 3 0.12445301 0.2576048                0    0.465    0.95      308.53
## 4 0.09139932 0.1775484                0    0.360    0.50      303.24
## 5 0.08814689 0.1959197                0    0.370    0.50      296.20
## 6 0.07776146 0.1854466                0    0.330    0.40      293.18
##   LST_Night_avg CO2_concentration_avg dataset MODIS_LC MODIS_IGBP MODIS_PFT
```

```
## 1        292.94              388.2825 FLUXNET        7        OSH        SH
## 2        291.96              388.6475 FLUXNET        7        OSH        SH
## 3        290.36              389.0650 FLUXNET        7        OSH        SH
## 4        286.34              388.9050 FLUXNET        7        OSH        SH
## 5        277.82              389.3200 FLUXNET        7        OSH        SH
## 6        276.80              389.1600 FLUXNET        7        OSH        SH
##   koppen_sub koppen hemisphere LOCATION_LAT LOCATION_LONG
## 1        BSk   Arid          S     -33.4648      -66.4598
## 2        BSk   Arid          S     -33.4648      -66.4598
## 3        BSk   Arid          S     -33.4648      -66.4598
## 4        BSk   Arid          S     -33.4648      -66.4598
## 5        BSk   Arid          S     -33.4648      -66.4598
## 6        BSk   Arid          S     -33.4648      -66.4598
```

Number of rows(rows contain NA)

```
nrow(SITE_month_df_2)
```

```
## [1] 2781
```

Number and details of NA

```
# Check whether new df contains NA or not
SITE_month_df_2_NA <- SITE_month_df_2[rowSums(is.na(SITE_month_df_2)) > 0,]
print("Number of rows with NA")
```

```
## [1] "Number of rows with NA"
```

```
print(nrow(SITE_month_df_2_NA))
```

```
## [1] 563
```

Missing values

feature	number_of NaN	feature	number_of NaN
P_F_avg	2	b3_avg	91
NETRAD_avg	178	b4_avg	91
ET_avg	6	b5_avg	91
CSIF-SIFdaily_avg	24	b6_avg	91
CSIF-SIFinst_avg	24	b7_avg	91
PET_avg	12	EVI_avg	118
Ts_avg	12	GCI_avg	101
Tmean_avg	12	NDVI_avg	102
prcp_avg	12	NDWI_avg	91
vpd_avg	12	NIRv_avg	102
prcp-lag3_avg	12	kNDVI_avg	91
ESACCI-sm_avg	236	Percent_Snow_avg	28

feature	number_of NaN	feature	number_of NaN
b1_avg	91	Fpar_avg	136
b2_avg	86	Lai_avg	136

```
# describe(SITE_month_df_2_NA)
```

```
nrow(SITE_month_df_2)
```

```
## [1] 2781
```

Export csv

```
write.csv(SITE_month_df_2,
          "../../../data/datasets/static_features_month_df_raw.csv", row.names=FALSE)
```

APPENDIX

P_F_avg

```
# SITE_month_df_2_NA %>%
#   subset(P_F_avg == "NaN")
```

```
# SITE_month_df_2_NA %>%
#   subset(SITE_ID == "FI-Ken")
```

```
# raw_df %>%
#    subset(SITE_ID == "FI-Ken")
```

```
# Calculate average of site and fill NA with average value
P_F_avg_FI_Ken <- SITE_month_df_2 %>%
  subset(SITE_ID == "FI-Ken") %>%
  drop_na(P_F_avg) %>% select(P_F_avg) %>% colMeans()
print("average of P_F_avg in FI_Ken")
```

```
## [1] "average of P_F_avg in FI_Ken"
```

```
print(P_F_avg_FI_Ken)
```

```
## P_F_avg
## 1.92055
```

```
SITE_month_df_2$P_F_avg[is.na(SITE_month_df_2$P_F_avg)] <- P_F_avg_FI_Ken
```

```
# SITE_month_df_2 %>%
#   subset(SITE_ID == "FI-Ken")
```

NETRAD_avg

```
SITE_month_df_2_NA %>%
  subset(NETRAD_avg == "NaN") %>%
  group_by(SITE_ID) %>%
  summarise(count = n())
```

```
## # A tibble: 21 x 2
##    SITE_ID count
##    <chr>   <int>
##  1 AR-Vir      1
##  2 BE-Maa     12
##  3 CA-Cbo     11
##  4 CH-Aws      6
##  5 CH-Lae     12
##  6 CH-Oe2     12
##  7 DE-RuW     12
##  8 FI-Ken     12
##  9 FI-Sii     12
## 10 FR-LGt     12
## # ... with 11 more rows
```

```
# Overall average
NETRAD_avg_all <- SITE_month_df_2 %>% select(NETRAD_avg) %>% drop_na %>% colMeans()
print("Overall average")
```

```
## [1] "Overall average"
```

```
NETRAD_avg_all
```

```
## NETRAD_avg
##    83.8256
```

```
# Site average
NETRAD_avg_AR_Vir <- SITE_month_df_2 %>%
  subset(SITE_ID == "AR-Vir") %>%
  drop_na(NETRAD_avg) %>% select(NETRAD_avg) %>% colMeans()
NETRAD_avg_CH_Aws <- SITE_month_df_2 %>%
  subset(SITE_ID == "CH-Aws") %>%
  drop_na(NETRAD_avg) %>% select(NETRAD_avg) %>% colMeans()
NETRAD_avg_GL_NuF <- SITE_month_df_2 %>%
  subset(SITE_ID == "GL-NuF") %>%
  drop_na(NETRAD_avg) %>% select(NETRAD_avg) %>% colMeans()
NETRAD_avg_GL_ZaH <- SITE_month_df_2 %>%
  subset(SITE_ID == "GL-ZaH") %>%
  drop_na(NETRAD_avg) %>% select(NETRAD_avg) %>% colMeans()
```

```
NETRAD_avg_RU_Che <- SITE_month_df_2 %>%
  subset(SITE_ID == "RU-Che") %>%
  drop_na(NETRAD_avg) %>% select(NETRAD_avg) %>% colMeans()
NETRAD_avg_SJ_Adv <- SITE_month_df_2 %>%
  subset(SITE_ID == "SJ-Adv") %>%
  drop_na(NETRAD_avg) %>% select(NETRAD_avg) %>% colMeans()
NETRAD_avg_US_GBT <- SITE_month_df_2 %>%
  subset(SITE_ID == "US-GBT") %>%
  drop_na(NETRAD_avg) %>% select(NETRAD_avg) %>% colMeans()
```

```
# Impute missing data
SITE_month_df_2[SITE_month_df_2$SITE_ID == "AR-Vir" && NETRAD_avg == "NaN"] <- NETRAD_avg_AR_Vir
```

```
## Warning in SITE_month_df_2$SITE_ID == "AR-Vir" && NETRAD_avg == "NaN":
## 'length(x) = 2781 > 1' in coercion to 'logical(1)'
```

```
# SITE_month_df_2 %>% subset(SITE_ID == "AR-Vir" & NETRAD_avg == "NaN") %>% select(NETRAD_avg)
# <- P_F_avg_FI_Ken
# NETRAD_avg_AR_Vir
```

```
# SITE_month_df_2$P_F_avg[is.na(SITE_month_df_2$NETRAD_avgP_F_avg)] <- P_F_avg_FI_Ken
```