STAT 685: Dr. Suojin Wang's Group

Modeling Seoul Bike Sharing Demand

Nam Tran, Bai Zou

Fall 2020

Background

Many urban cities including Seoul have adopted bicycle rental schemes for the enhancement of mobility and leisure for their citizens and visitors. The Seoul city government wishes to make the rental bicycles available and accessible to the public at the right time in order to operate this business smoothly and to lessen the waiting time. Therefore, it is of strong interest for the city to provide a stable supply of rental bicycles. This leads to the important statistical question of how to make accurate predictions of bicycle count required at each hour for the stable supply of rental bicycles.

Data

```
# Loading the Data
datPath = paste0(dirname(getwd()), "/data/SeoulBikeData.csv")
colNames = c("Date", "RentedBikeCount", "Hour", "Temp", "Humidity",
             "WindSpeed", "Visibility", "DewPointTemp", "SolarRadiation",
             "Rainfall", "Snowfall", "Seasons", "Holiday", "FunctionalDay")
dat = read_csv(datPath, col_names = colNames, skip=1)

# Setting up Factors
dat$Seasons = as_factor(dat$Seasons)
dat$Holiday = as_factor(dat$Holiday)
dat$FunctionalDay = as_factor(dat$FunctionalDay)

# Creating DateTime that incorporates both Date and Hours and dropping Hour
dat$Date = AsDateTime(dat$Date) + hours(dat$Hour)
dat = dat %>%
  rename(DateTime = Date) %>%
  select(-Hour)
```

Seoul Bike Sharing Demand Data

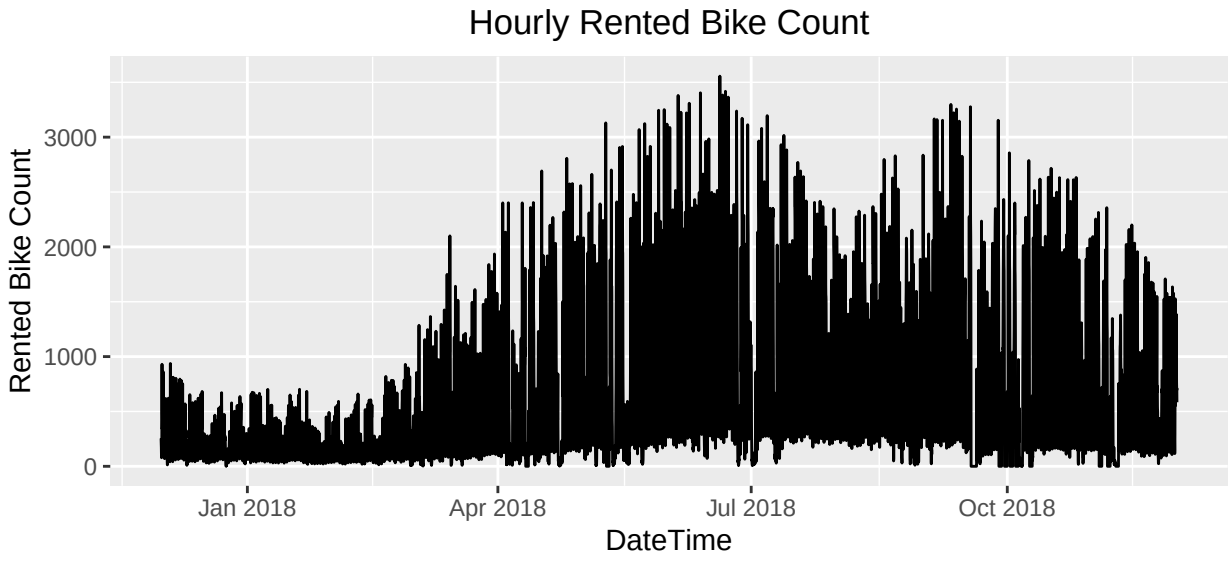
- Downloaded the data from the UCI Machine Learning Repo.
- Contains 8760 measurements of number of bikes rented over 364.9583333 days.

Features of the data are,

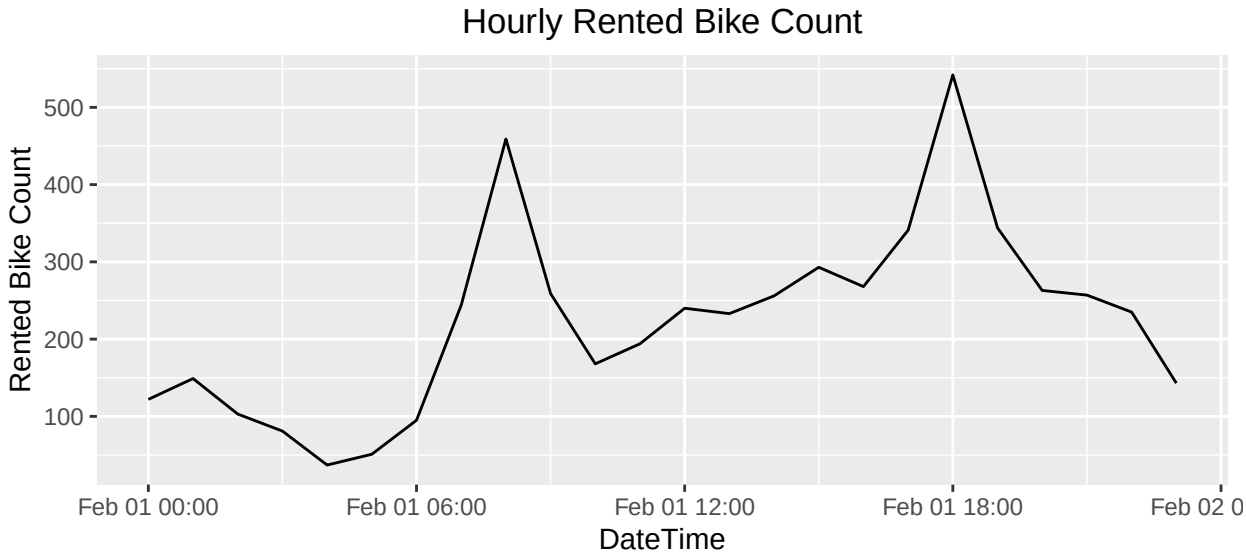
- `DateTime`
- `RentedBikeCount`
- `Temp`, in Celsius.
- `Humidity`, in percent, max of 100.

- `Windspeed`
- `Visibility` out to 10 meters.
- `DewPointTemp`, in Celsius.
- `SolarRadiation`
- `Rainfall`, in mm.
- `Snowfall`, in cm.
- `Seasons`, a factor with levels {Winter, Spring, Summer, Autumn}.
- `Holiday`, a factor with levels {Holiday, No holiday}.
- `FunctionalDay`, a factor with levels {NoFunc(Non Functional Hours), Fun(Functional hours)}

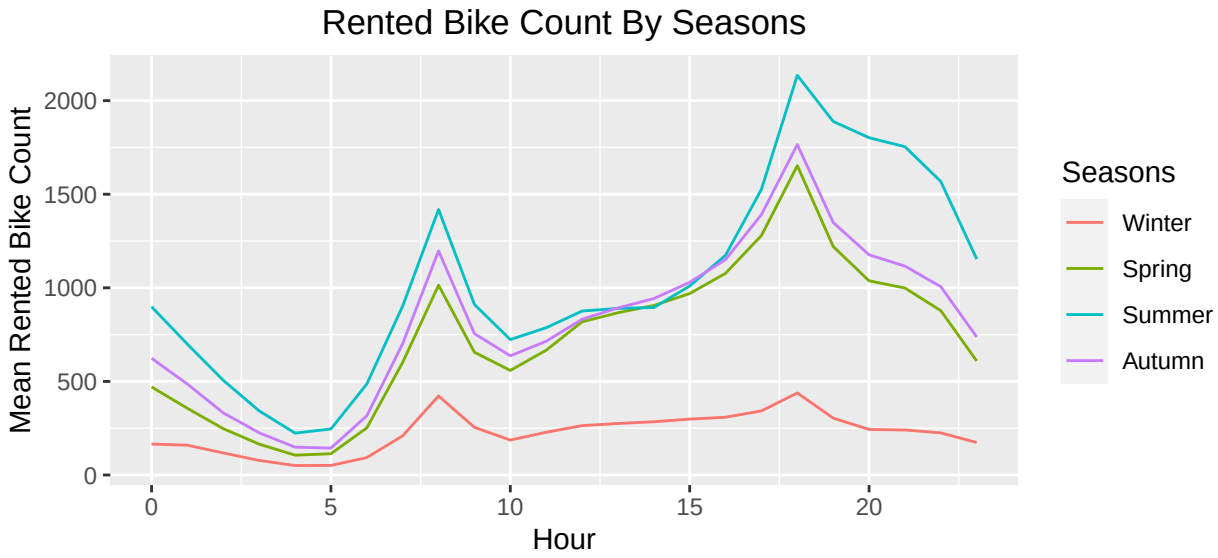
```
# plotting over entire time period
dat %>%
  ggplot(aes(x=DateTime, y=RentedBikeCount)) +
  geom_line() +
  labs(y="Rented Bike Count", x="DateTime", title="Hourly Rented Bike Count")
```




```
# plotting over one day
dat %>%
  filter(date(DateTime) == "2018-02-01") %>%
  ggplot(aes(x=DateTime, y=RentedBikeCount)) +
  geom_line() +
  labs(y="Rented Bike Count", x="DateTime", title="Hourly Rented Bike Count")
```




```r
# plotting "effect" of seasons
dat %>%
  mutate(Hour = hour(DateTime)) %>%
  group_by(Seasons, Hour) %>%
  summarise(MeanRentedBikeCount = mean(RentedBikeCount)) %>%
  ggplot(aes(x=Hour, y=MeanRentedBikeCount, color=Seasons)) +
  geom_line() +
  labs(y="Mean Rented Bike Count", x="Hour", title="Rented Bike Count By Seasons")
```



Fundamentally, our data is time-series data. As such, let x_t be the time series we're working to model, i.e., Seoul's bike sharing data.

Stationarity

It's arguable that there might be a strong seasonality component (less in winter more in summer), but that's hard to ascertain here since we only have one year's of data and only have one cycle. Further, there might be strong seasonality on an intraday basis (less in early morning and ramp up afterwards). If there was a strong seasonality component, we'd say our data isn't stationary, since on a first order basis, $E(x_t)$ will be dependent on t. Stationarity is important for a multitude of reasons, including *averaging being meaningful*

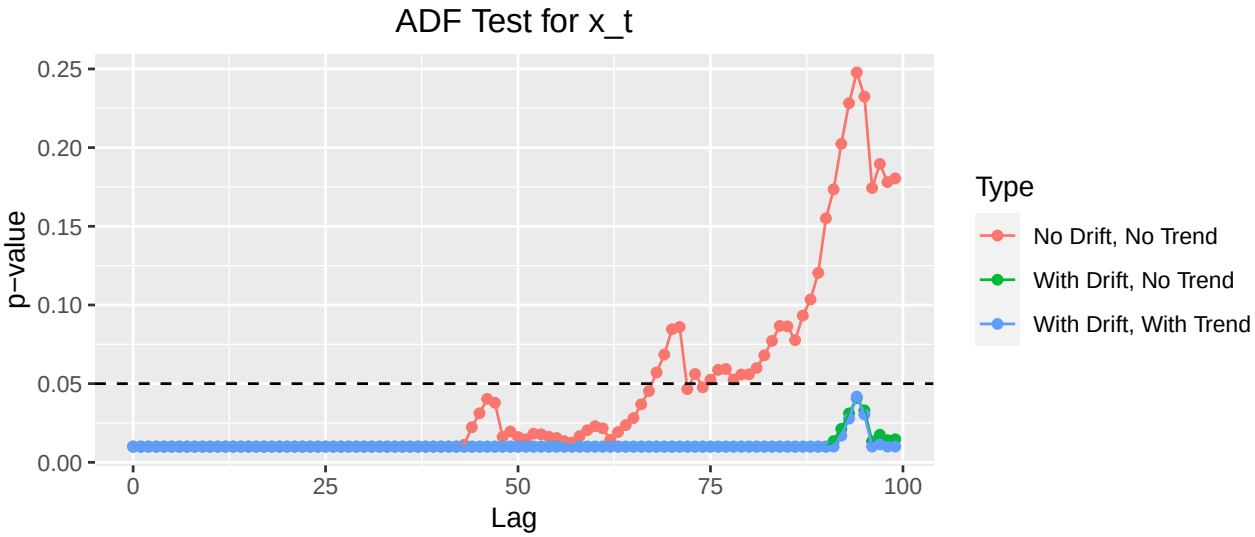
and any *conditional expectation model we build is stable.*

Note, we can still incorporate terms to make a time series stationary, e.g., trend-stationary.

We can test this directly using the Augmented Dickey-Fuller (ADF) Test, which intuitively tests for the presence of a unit root, which implies non-stationarity. H_0 for ADF is that x_t is non-stationary, and H_a is that x_t is stationary. Note there are different types of stationarity, e.g., in presence of drift (μ) or linear trend (βt).

```
zz = adf.test(dat$RentedBikeCount, 100, output=FALSE)
adf1 = zz[[1]] %>% data.frame %>% mutate(Type="No Drift, No Trend")
adf2 = zz[[2]] %>% data.frame %>% mutate(Type="With Drift, No Trend")
adf3 = zz[[3]] %>% data.frame %>% mutate(Type="With Drift, With Trend")
adfDat = bind_rows(adf1, adf2, adf3)

adfDat %>% ggplot(aes(x=lag, y=p.value, color=Type)) +
  geom_point() +
  geom_line() +
  geom_hline(yintercept=0.05, linetype="dashed", color = "black") +
  labs(x="Lag", y="p-value", title="ADF Test for x_t")
```



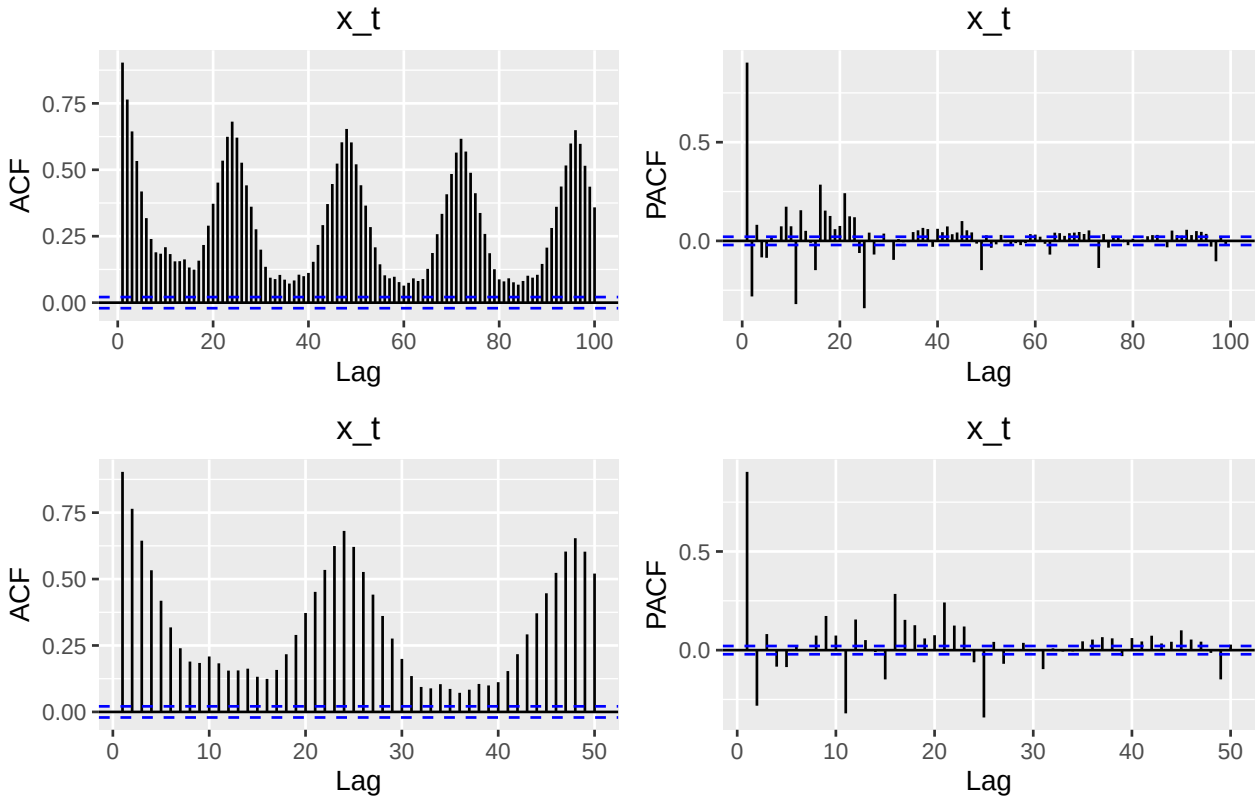
From this, we can note that we're relatively stationarity up to lag 40 under the relaxed condition of having drift.

ACF and PACF of x_t

We look at ACF and PACF of y_t up to 100 and 50 lags, to see if any simple models come to mind.

```
selfACF   = ggAcf(dat$RentedBikeCount, lag.max = 100) + labs(title="x_t")
selfPACF  = ggPacf(dat$RentedBikeCount, lag.max = 100) + labs(title="x_t")
selfACF2  = ggAcf(dat$RentedBikeCount, lag.max = 50) + labs(title="x_t")
selfPACF2 = ggPacf(dat$RentedBikeCount, lag.max = 50) + labs(title="x_t")

plots = list(selfACF, selfPACF, selfACF2, selfPACF2)
gridExtra::grid.arrange(grobs = plots, ncol=2)
```



Math theory states that an AR(p) model would have a hard cutoff to zero in the PACF plot for $h > p$, and a MA(q) model would have a hard cutoff to zero in the ACF plot for $h > q$. From the ACF plot and seeing statistically significant autocorrelations all the way out, a simple MA(q) model will not suffice. Looking at the PACF plot, we see a strong "cut-off" at around lag 25, suggesting an AR(25) model. Needless to say, an AR(25) model isn't very palatable and doesn't seem parsimonious. As such, we seemingly can't get away with a simple MA(q) nor a simple AR(p) model.

Additional Covariates

Fortunately, there's additional covariates we can leverage, both quantitative and qualitative features.

Quantitative features we have include,

- `DateTime`
- `Temp`, in Celsius.
- `Humidity`, in percent, max of 100.
- `Windspeed`
- `Visibility` out to 10 meters.
- `DewPointTemp`, in Celsius.
- `SolarRadiation`
- `Rainfall`, in mm.
- `Snowfall`, in cm.

Qualtitative features we have include,

- `Seasons`, a factor with levels {Winter, Spring, Summer, Autumn}.
- `Holiday`, a factor with levels {Holiday, No holiday}.
- `FunctionalDay`, a factor with levels {NoFunc(Non Functional Hours), Fun(Functional hours)}

CCF of Covariates

We looked at CCF of the quantitative covariates with respect to x_t. We used this information to help choose the features to use in the model.

Note, we're simply looking at the cross-correlations of the **untransformed covariates** with respect to x_t. There might be a transform that would induce more linear correlations, which will be investigated later. These transformations can include but aren't limited to `log` and `percent change`. Or, we may even consider Box-Cox transformations downstream.