s626-final-project

Saumya Mehta

2022-11-07

load data

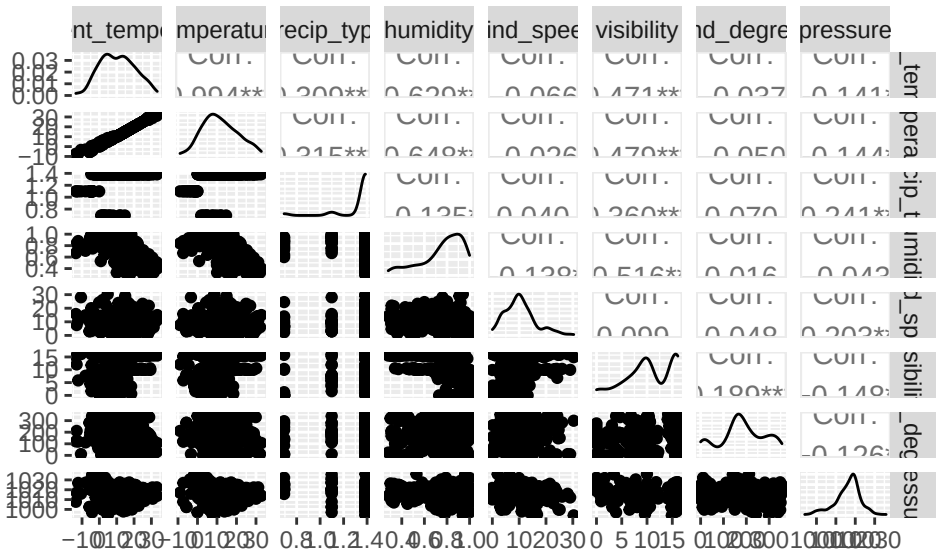
```
weather <- read.csv("data/weatherHistory.csv", header=TRUE)
```

select relevant data:

```
weather.df1 <- weather %>%
  filter(year(Formatted.Date)==2016) %>%
  mutate(Precip.Type = recode(Precip.Type,"null" = 2, "snow" = 3, "rain" = 4),
         Precip.Type = log(Precip.Type)) %>%
  dplyr::select("apparent_temperature" = Apparent.Temperature..C.,
                "temperature" = Temperature..C.,
                "precip_type" = Precip.Type,
                "humidity" = Humidity,
                "wind_speed" = Wind.Speed..km.h.,
                "visibility" = Visibility..km.,
                "wind_degrees" = Wind.Bearing..degrees.,
                "pressure" = Pressure..millibars.)
rand_ind <- sample(nrow(weather.df1), 300, replace = FALSE)

weather.df <-  weather.df1[rand_ind,]
```

```
library(GGally)
ggpairs(weather.df, columns = c("apparent_temperature","temperature","precip_type","humidity","wind_spe
```



We will try creating a model with apparent temperature as the response variable and temperature, humidity, visibility and wind speed and wind_degrees as our explanatory variables

```
weather.df  <- weather.df %>%
  dplyr::select(apparent_temperature,temperature,humidity,visibility, wind_speed, wind_degrees)
```

Bayesian Linear regression using Zellner-g prior:

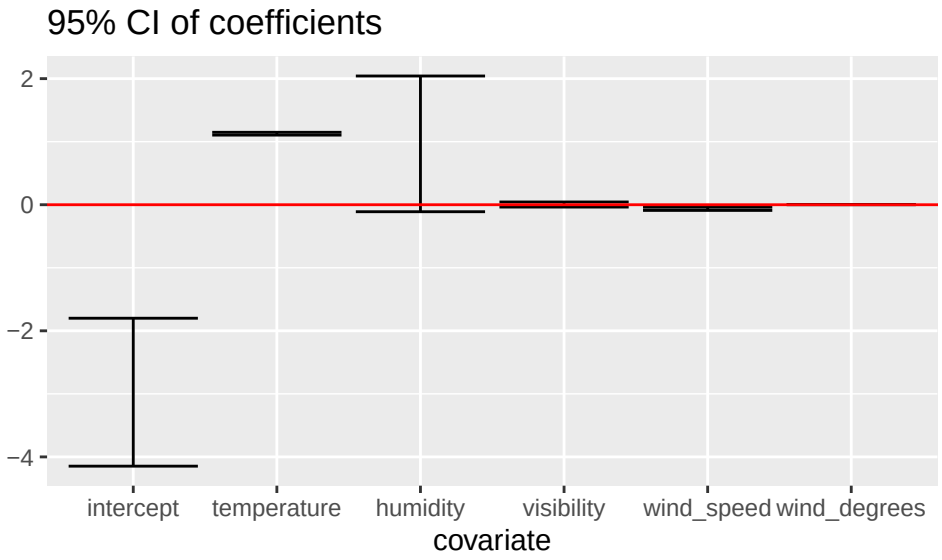
```
y <- as.matrix(weather.df[,1])
x <- model.matrix(apparent_temperature ~ ., weather.df)
n <- length(y)
g<- NROW(weather.df)
nu0 <- 1
sigma20 <- summary(lm(y ~ x[,-1], data = weather.df))$sigma^2
nSamples <- 1e3
trace <- list(s2 = numeric(nSamples), beta = array(NA, dim=c(nSamples,6)))
```

constants

```
X <- model.matrix(apparent_temperature ~ ., data = weather.df)
XtX.inv <- solve(t(X) %*% X)
H <- X %*% XtX.inv %*%t(X)
beta.ols <- XtX.inv %*%t(X)%*%weather.df$apparent_temperature
SSRg <- t(weather.df$apparent_temperature) %*% (diag(n) - g/(g+1) *H) %*% weather.df$apparent_temperatu

# collect sigma^2 and beta
for (i in 1:nSamples){
  s2 <- 1/rgamma(n=1, shape = (nu0+n)/2, rate = (nu0*sigma20 + SSRg)/2)
  beta <- mvrnorm(n=1, mu = g/(g+1)*beta.ols, Sigma=g/(g+1) * s2 * XtX.inv)
  trace$s2[i] <- s2
  trace$beta[i,] <- beta
```

```
}
signif.df <- plyr::aaply(trace$beta, 2, function(b) {
  quantile(b, c(.025, .975))
}) %>%
  as.data.frame() %>%
  dplyr::mutate(covariate = factor(c('intercept', colnames(weather.df[-1])),
                                   levels = c('intercept',
                                              colnames(weather.df[-1]))))
ggplot(signif.df) +
  geom_errorbar(aes(x = covariate, ymin = `2.5%`, ymax = `97.5%`)) +
  geom_abline(slope = 0, colour = 'red') +
  labs(title = '95% CI of coefficients')
```

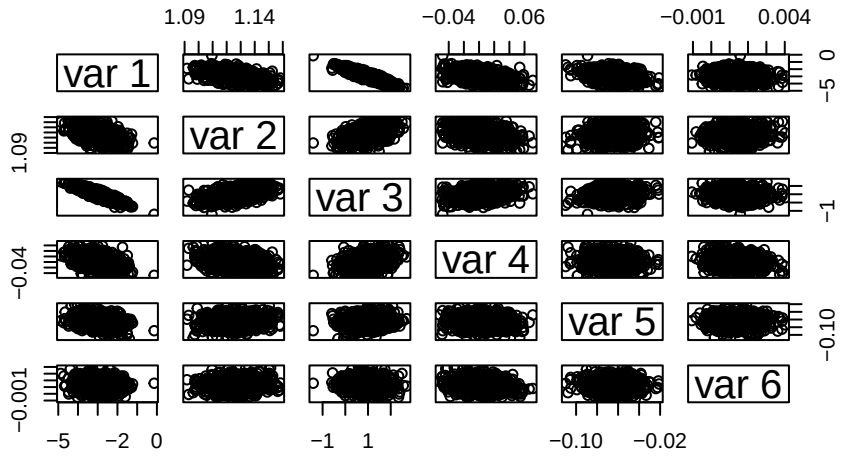


According to our analysis, only temperature, humidity and wind speed are strongly predictive variables.

```
apply(trace$beta, MARGIN = 2, FUN = mean)
```

```
[1] -2.957071750  1.126554193  0.977377007  0.001987254 -0.064699525
[6]  0.001383408
```

```
pairs(trace$beta)
```




```
# log marginal code referenced from course material on bayesian linear regression
log.marginal.y <- function(y, x, g = length(y), nu0){
  n <- length(y)
  p <- ncol(x)
  if (p == 0) {
    sigma20 <- mean(y^2)
    SSRg <- t(y) %*% y
  } else{
    tmp_lm <- lm(y~x + 0)
    sigma20 <- summary(tmp_lm)$sigma^2
    SSRg <- t(y) %*% y - g/(g+1) * t(y) %*% predict(tmp_lm)
  }
  res <- -0.5723649429247 * n + #the magic number is log(pi)/2
    lgamma(0.5*(nu0 + n)) -lgamma(0.5*nu0) +
     0.5 * ( -p * log( 1 + g ) +
              nu0 * log( nu0 * sigma20) +
              -(nu0 + n) * log(nu0 * sigma20 + SSRg)
     )
  return(res)
}

z <- as.matrix(expand.grid(0:1, 0:1, 0:1, 0:1,0:1,0:1))
dimnames(z) <- list(NULL, c('Intercept', 'temperature','humidity','visibility','wind_speed','wind_degre
cols <- apply(z, MARGIN = 1, FUN = function(x)which(x == 1))

lp <- numeric()
for (i in 1:64){
  xz <- as.matrix(x[, cols[[i]] ], nrow = length(y))
  lp[i] <- log.marginal.y(y=y, x=xz, nu0 = 1)

}
probs <- exp(lp) /sum(exp(lp))
cbind(z,lp, probs)
```

```
      Intercept temperature humidity visibility wind_speed wind_degrees
 [1,]         0           0        0          0          0            0
 [2,]         1           0        0          0          0            0
 [3,]         0           1        0          0          0            0
 [4,]         1           1        0          0          0            0
```

```
 [5,]    0    0    1    0    0    0
 [6,]    1    0    1    0    0    0
 [7,]    0    1    1    0    0    0
 [8,]    1    1    1    0    0    0
 [9,]    0    0    0    1    0    0
[10,]    1    0    0    1    0    0
[11,]    0    1    0    1    0    0
[12,]    1    1    0    1    0    0
[13,]    0    0    1    1    0    0
[14,]    1    0    1    1    0    0
[15,]    0    1    1    1    0    0
[16,]    1    1    1    1    0    0
[17,]    0    0    0    0    1    0
[18,]    1    0    0    0    1    0
[19,]    0    1    0    0    1    0
[20,]    1    1    0    0    1    0
[21,]    0    0    1    0    1    0
[22,]    1    0    1    0    1    0
[23,]    0    1    1    0    1    0
[24,]    1    1    1    0    1    0
[25,]    0    0    0    1    1    0
[26,]    1    0    0    1    1    0
[27,]    0    1    0    1    1    0
[28,]    1    1    0    1    1    0
[29,]    0    0    1    1    1    0
[30,]    1    0    1    1    1    0
[31,]    0    1    1    1    1    0
[32,]    1    1    1    1    1    0
[33,]    0    0    0    0    0    1
[34,]    1    0    0    0    0    1
[35,]    0    1    0    0    0    1
[36,]    1    1    0    0    0    1
[37,]    0    0    1    0    0    1
[38,]    1    0    1    0    0    1
[39,]    0    1    1    0    0    1
[40,]    1    1    1    0    0    1
[41,]    0    0    0    1    0    1
[42,]    1    0    0    1    0    1
[43,]    0    1    0    1    0    1
[44,]    1    1    0    1    0    1
[45,]    0    0    1    1    0    1
[46,]    1    0    1    1    0    1
[47,]    0    1    1    1    0    1
[48,]    1    1    1    1    0    1
[49,]    0    0    0    0    1    1
[50,]    1    0    0    0    1    1
[51,]    0    1    0    0    1    1
[52,]    1    1    0    0    1    1
[53,]    0    0    1    0    1    1
[54,]    1    0    1    0    1    1
[55,]    0    1    1    0    1    1
[56,]    1    1    1    0    1    1
[57,]    0    0    0    1    1    1
[58,]    1    0    0    1    1    1
```

```
[59,]         0           1        0           1           1           1
[60,]         1           1        0           1           1           1
[61,]         0           0        1           1           1           1
[62,]         1           0        1           1           1           1
[63,]         0           1        1           1           1           1
[64,]         1           1        1           1           1           1
                lp        probs
 [1,] -1230.4462 0.000000e+00
 [2,] -1118.0173 0.000000e+00
 [3,]  -639.2324 3.068766e-55
 [4,]  -524.4126 2.251956e-05
 [5,] -1162.5300 0.000000e+00
 [6,] -1046.0897 0.000000e+00
 [7,]  -545.0054 2.565612e-14
 [8,]  -523.8284 4.038839e-05
 [9,] -1080.6710 0.000000e+00
[10,] -1083.4953 0.000000e+00
[11,]  -581.2575 4.624915e-30
[12,]  -527.0484 1.613815e-06
[13,] -1077.1601 0.000000e+00
[14,] -1041.5836 0.000000e+00
[15,]  -542.0784 4.790610e-13
[16,]  -526.6695 2.357289e-06
[17,] -1160.8686 0.000000e+00
[18,] -1120.2270 0.000000e+00
[19,]  -550.2078 1.412037e-16
[20,]  -514.2593 5.781862e-01
[21,] -1156.0577 0.000000e+00
[22,] -1043.0429 0.000000e+00
[23,]  -523.5139 5.531613e-05
[24,]  -515.6614 1.422854e-01
[25,] -1081.4984 0.000000e+00
[26,] -1083.8923 0.000000e+00
[27,]  -541.3105 1.032505e-12
[28,]  -517.1119 3.335961e-02
[29,] -1079.9062 0.000000e+00
[30,] -1037.7507 0.000000e+00
[31,]  -524.9199 1.355946e-05
[32,]  -518.3929 9.265959e-03
[33,] -1155.6762 0.000000e+00
[34,] -1120.6624 0.000000e+00
[35,]  -594.1157 1.204740e-35
[36,]  -526.1239 4.067475e-06
[37,] -1154.2634 0.000000e+00
[38,] -1048.7597 0.000000e+00
[39,]  -547.7018 1.730598e-15
[40,]  -525.4247 8.184809e-06
[41,] -1080.8470 0.000000e+00
[42,] -1083.1430 0.000000e+00
[43,]  -576.8548 3.777346e-28
[44,]  -528.4135 4.120708e-07
[45,] -1079.8259 0.000000e+00
[46,] -1043.2188 0.000000e+00
[47,]  -544.6884 3.522763e-14
```

```
[48,]  -528.2470 4.867458e-07
[49,] -1148.7309 0.000000e+00
[50,] -1122.9047 0.000000e+00
[51,]  -546.0269 9.237569e-15
[52,]  -515.4446 1.767318e-01
[53,] -1151.2384 0.000000e+00
[54,] -1045.7988 0.000000e+00
[55,]  -526.1928 3.796934e-06
[56,]  -516.7918 4.594319e-02
[57,] -1082.6843 0.000000e+00
[58,] -1083.6549 0.000000e+00
[59,]  -542.8941 2.119087e-13
[60,]  -518.1837 1.142149e-02
[61,] -1082.5904 0.000000e+00
[62,] -1039.5781 0.000000e+00
[63,]  -527.0286 1.646030e-06
[64,]  -519.6439 2.651939e-03
```

```
#Posterior mode of the model posterior:
z[which(probs == max(probs)), ]
```

```
   Intercept  temperature     humidity   visibility   wind_speed wind_degrees
           1            1            0            0            1            0
```

We can confirm Humidity and Temperature are the only significant variables as we get the highest posterior density when selecting a model with only these variables

```
X <- model.matrix(apparent_temperature ~ ., weather.df %>% dplyr::select(apparent_temperature,temperatu
n <- nrow(X)

XtX.inv <- solve(t(X) %*% X)
H <- X %*% XtX.inv %*% t(X)
y <- weather.df$apparent_temperature
tmp_lm <-lm(y  ~ X + 0)
s20 <- summary(tmp_lm)$sigma^2
beta.ols <- XtX.inv %*% t(X) %*% weather.df$apparent_temperature
ssreg <- t(weather.df$apparent_temperature) %*% (diag(n) - g / (g + 1) * H ) %*% weather.df$apparent_te
trace <- list(s2 = numeric(nSamples), beta = array(NA, dim=c(nSamples,4)))

# collect sigma^2 and beta
for (i in 1:nSamples){
  s2 <- 1/rgamma(n=1, shape = (nu0+n)/2, rate = (nu0*sigma20 + SSRg)/2)
  beta <- mvrnorm(n=1, mu = g/(g+1)*beta.ols, Sigma=g/(g+1) * s2 * XtX.inv)
  trace$s2[i] <- s2
  trace$beta[i,] <- beta
}
```

```
# create a test matrix:
test.df.sample <- weather.df1[-rand_ind, ] %>% dplyr::select(apparent_temperature,temperature, humidity
rand_ind1 <- sample(nrow(test.df.sample), 300, replace = FALSE)
test.df.sample <- test.df.sample[rand_ind1,]
test.model.matrix <- model.matrix(apparent_temperature ~ ., test.df.sample)
```

```
# fit to test data
beta.means <- apply(trace$beta, 2, mean)
yhat.test <- test.model.matrix %*% beta.means
# prediction error on test data
mean((test.df.sample$apparent_temperature - yhat.test) ** 2)
```

```
[1] 0.8614714
```

```
ggplot() +
  geom_point(aes(x = test.df.sample$apparent_temperature, y = yhat.test)) +
  geom_abline(colour = 'red') +
  labs(x = 'observed', y = 'predicted')
```