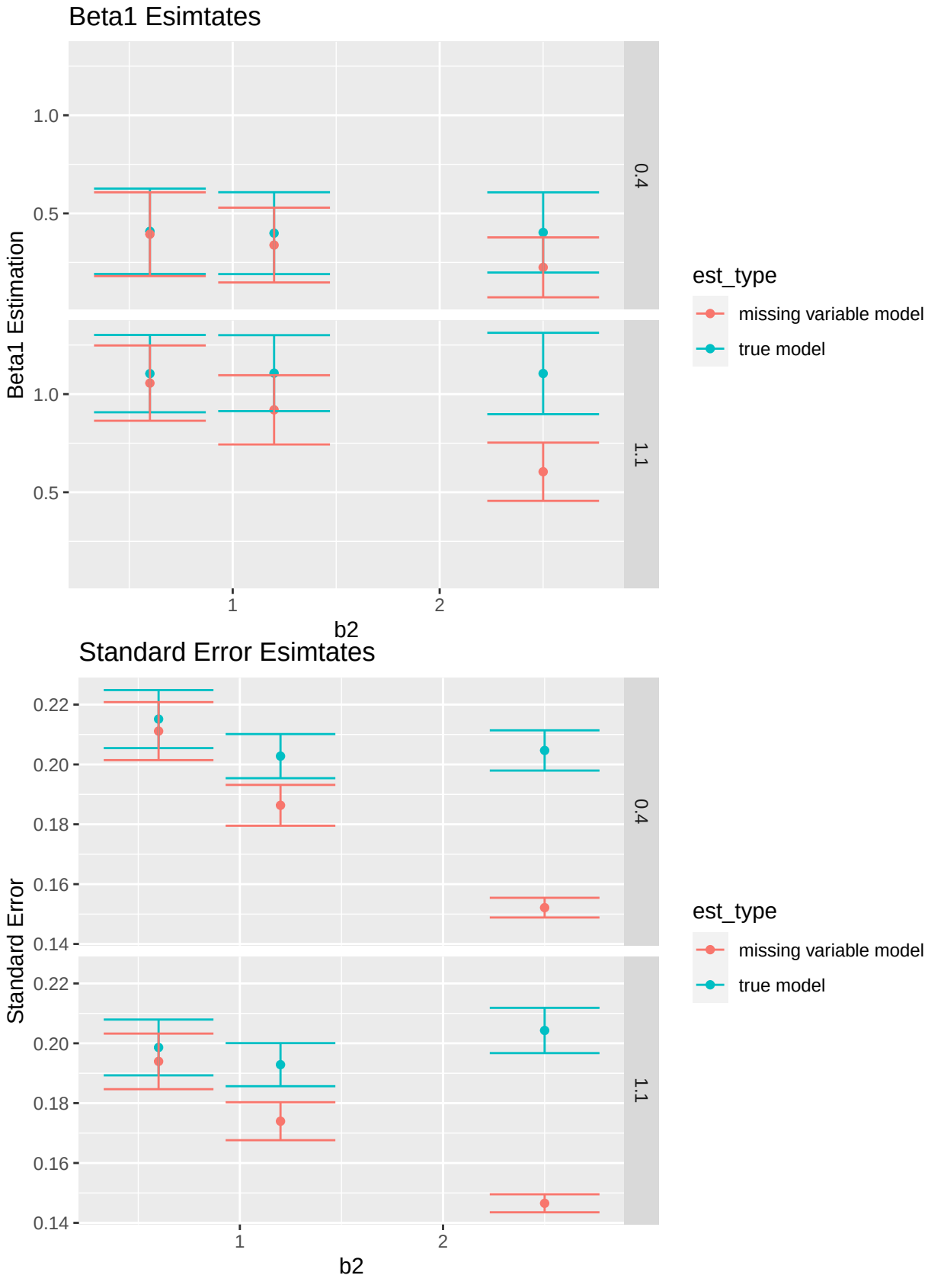
Stat 570 HW7

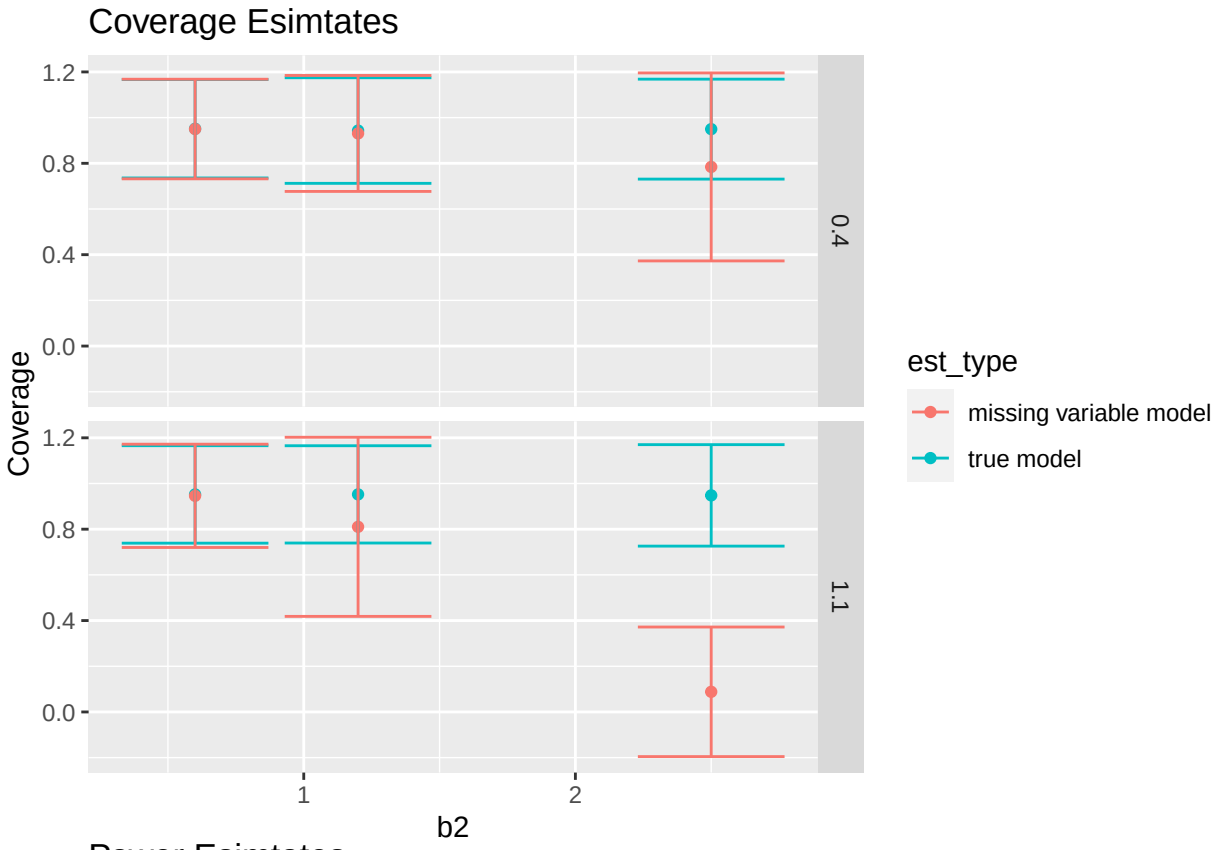
Dongyang Wang

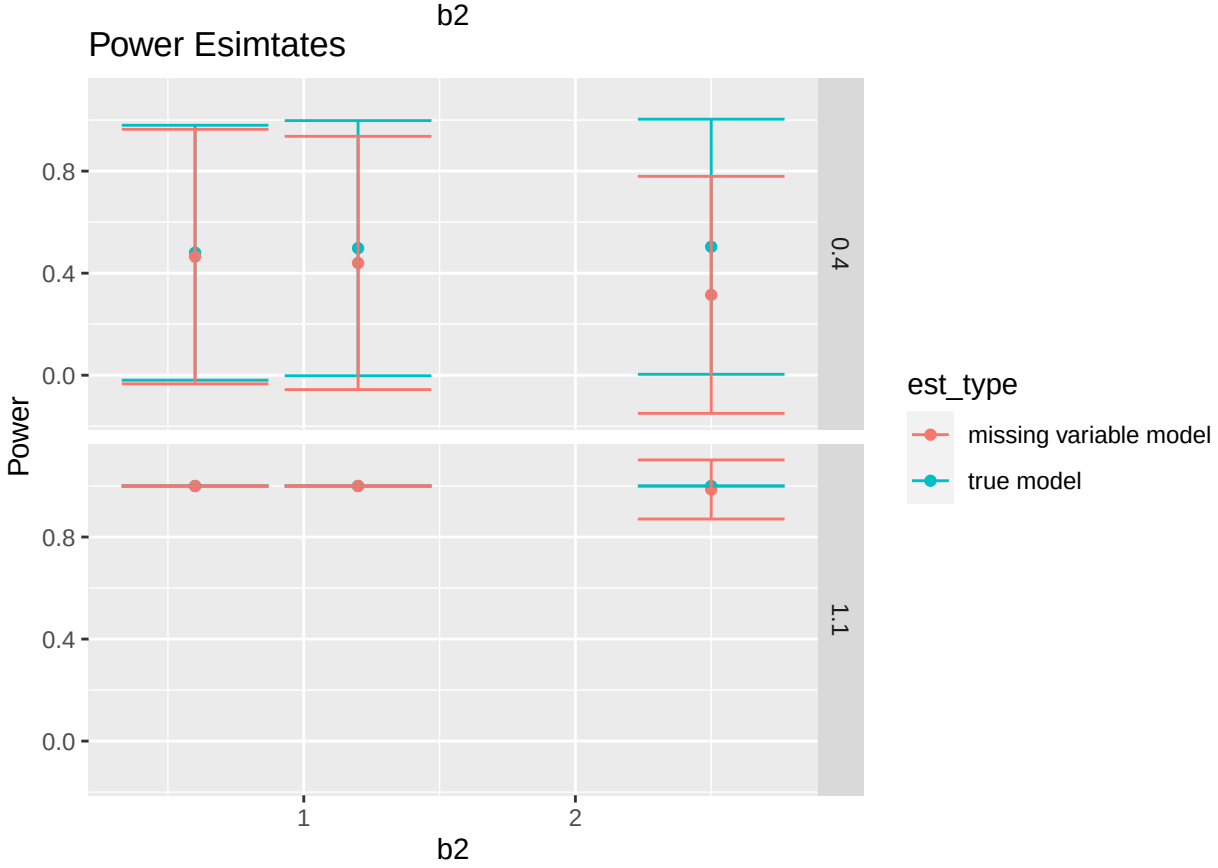
2022-11-30

Q1

a


Beta1 Esimtates
Beta1 Estimation
1.0
0.5
1.0
0.5
0.4
1.1
1
2
b2
est_type
missing variable model
true model
Standard Error Esimtates
Standard Error
0.22
0.20
0.18
0.16
0.14
0.22
0.20
0.18
0.16
0.14
0.4
1.1
1
2
b2
est_type
missing variable model
true model





After simulating with 5000 times, each time with 1000 random points, and generating the graphs with error

bars, we have obtained the following insights.

First, the glm model that is based on correct inclusion of variables performs well, in terms of accuracy of estimation, robustness of the estimate, the confidence interval (nearly 95%) all the time, as well as the power of the model for conducting Wald test.

By comparison, so we can observe from the above graphs, the omitted variable model does not perform well. It is especially the case as the importance/weight (coefficient) in the original b2 increase, for both values of b1, there will be less accurate estimate of the coefficients, decreased variance so worse reliability, poor confidence interval coverage, and slightly less power. Since the estimate is not accurate and the standard deviation is small, the interval is narrow and covers less likely the true parameters.

Compared with linear models, the omission of the variable z, which is independent of x, leads to prediction problems. In linear models, the variable z would not matter in terms of the estimation, standard error calculation, or the confidence interval.

Q2

a

```
##   pregnant glucose mass pedigree age diabetes
## 1        6     148 33.6    0.627  50      pos
## 2        1      85 26.6    0.351  31      neg
## 3        8     183 23.3    0.672  32      pos
## 4        1      89 28.1    0.167  21      neg
## 5        0     137 43.1    2.288  33      pos
## 6        5     116 25.6    0.201  30      neg
```

Removed missing values.

b

```
##             Logit Estimate    Logit SE Probit Estimate   Probit SE
## (Intercept)    -9.32278909 0.737279068    -5.473698289 0.396809311
## pregnant        0.11505791 0.032341071     0.067055633 0.018732802
## glucose         0.03594107 0.003555113     0.021036224 0.001986150
## mass            0.08752914 0.014722449     0.051903294 0.008395284
## pedigree        0.92058274 0.300831981     0.457143010 0.171883269
## age             0.01136584 0.009315052     0.007195378 0.005460427
##             CLoglog Estimate  CLoglog SE
## (Intercept)     -6.593361113 0.488891187
## pregnant         0.077851131 0.021421414
## glucose          0.023956745 0.002263167
## mass             0.059542247 0.009875868
## pedigree         0.230040553 0.193947006
## age              0.007446081 0.006539923
```

c

The logit model provides the best interpretability. One unit increase in a certain variable is associated with a unit increase in the log of odds ratio. For this model, we can show that based on the result below,

```
##
## Call:
## glm(formula = diabetes ~ ., family = binomial(link = "logit"),
##     data = data_clean)
##
## Deviance Residuals:
```

```
##     Min       1Q   Median       3Q      Max
## -2.8093  -0.7287  -0.4011   0.7275   2.4449
##
## Coefficients:
##              Estimate Std. Error z value Pr(>|z|)
## (Intercept) -9.322789   0.737279 -12.645  < 2e-16 ***
## pregnant     0.115058   0.032341   3.558 0.000374 ***
## glucose      0.035941   0.003555  10.110  < 2e-16 ***
## mass         0.087529   0.014722   5.945 2.76e-09 ***
## pedigree     0.920583   0.300832   3.060 0.002212 **
## age          0.011366   0.009315   1.220 0.222405
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 974.75  on 751  degrees of freedom
## Residual deviance: 703.24  on 746  degrees of freedom
## AIC: 715.24
##
## Number of Fisher Scoring iterations: 5
```

All terms but age are significant. So, pregnancy, glucose, mass, and pedigree are all associated with increased risk for diabetes. Their exact odds ratios are calculated in order below (other terms constant).

```
## [1] 1.121939
```

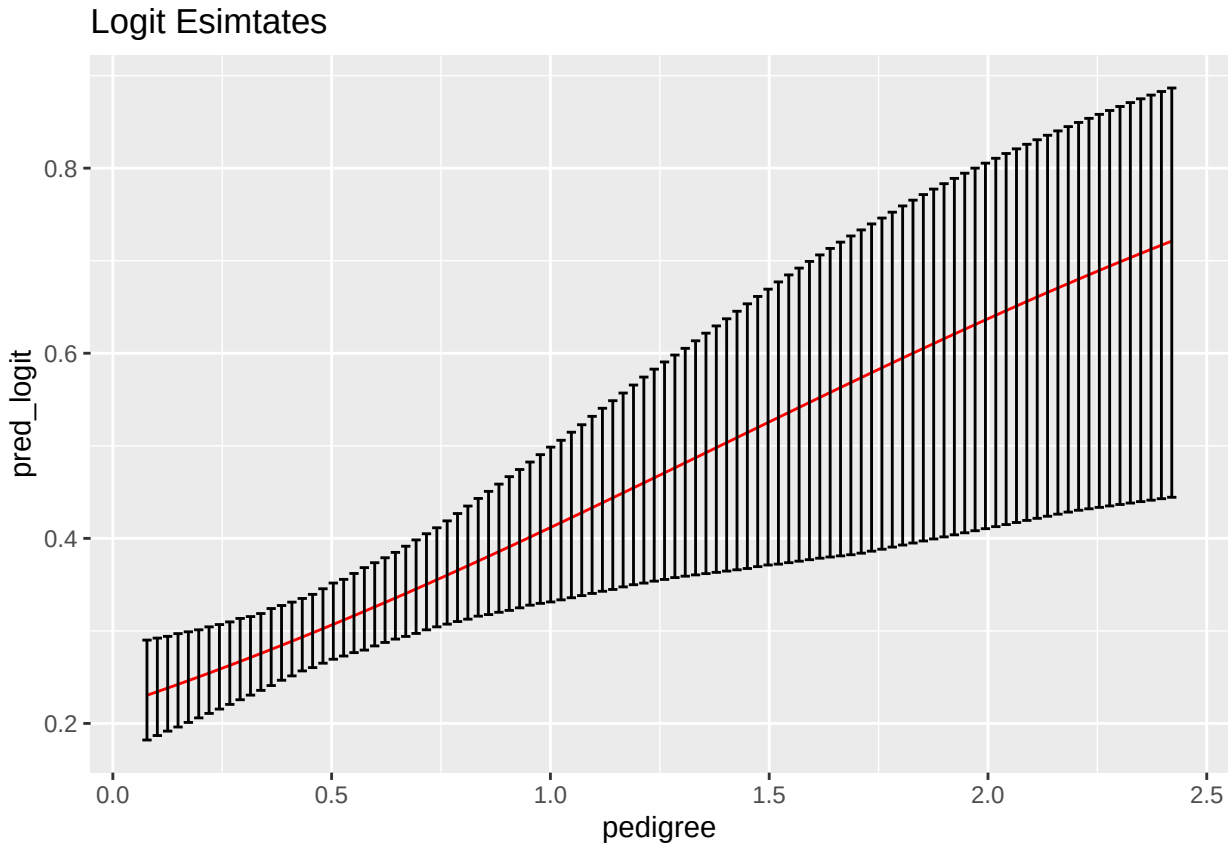
```
## [1] 1.036595
```

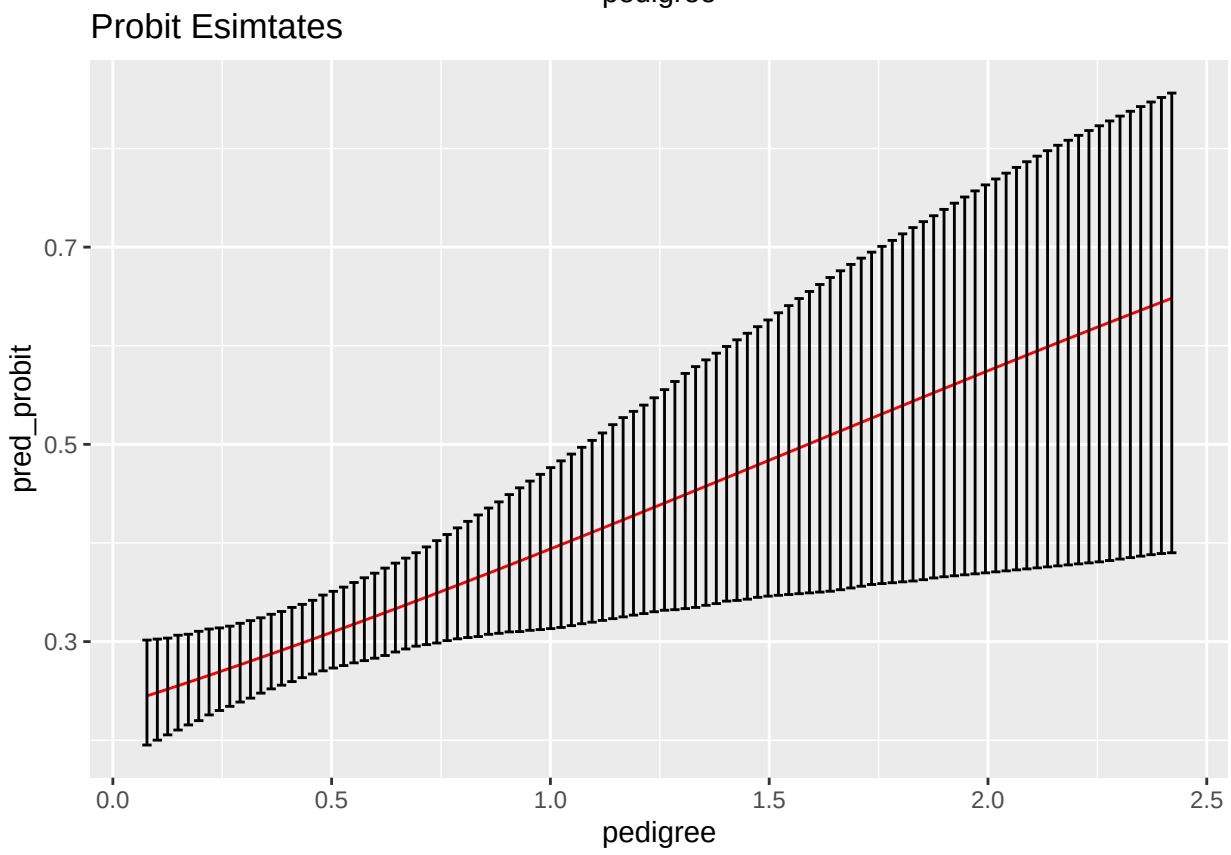
```
## [1] 1.091474
```

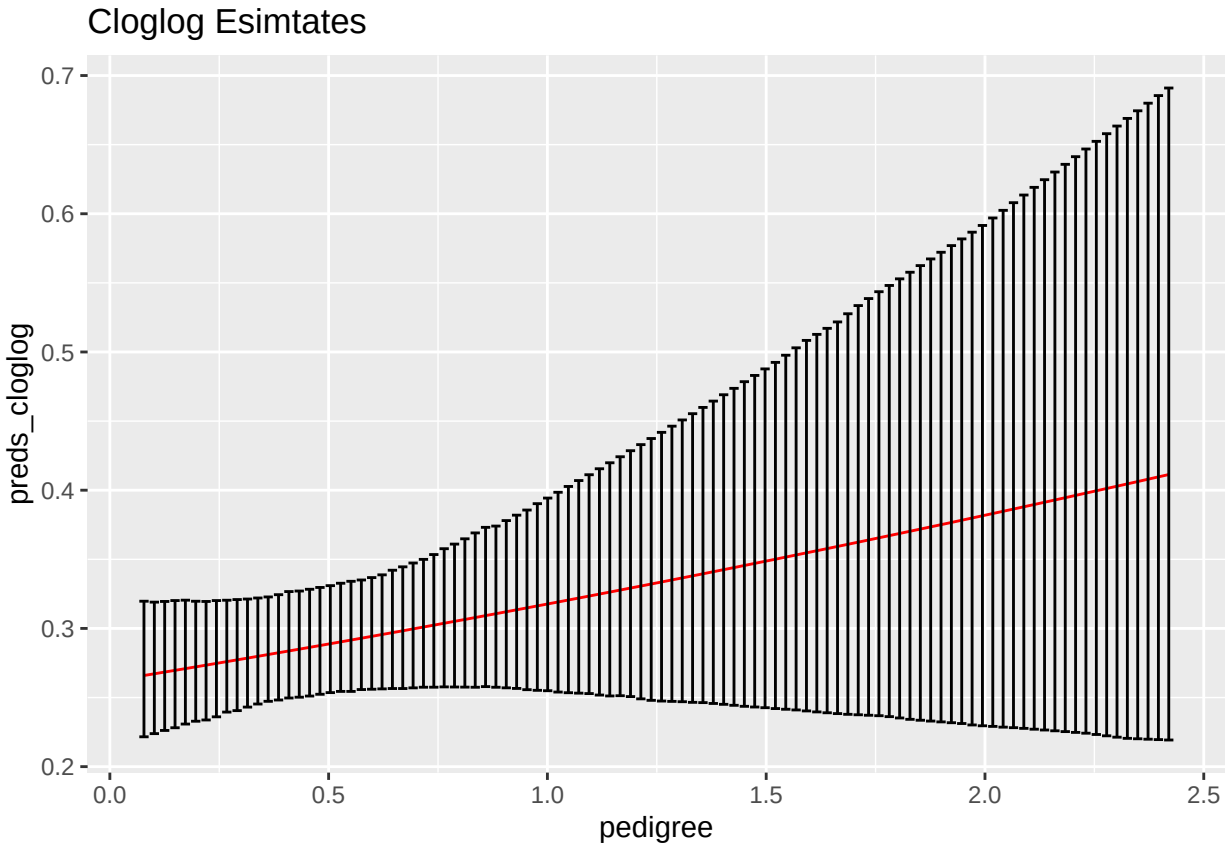
```
## [1] 2.510754
```

d

We calculate the confidence interval based on sampling from the multivariable normal with mean given by the model estimates and variance given by model variance.


Logit Esimtates
pred_logit
0.8
0.6
0.4
0.2
0.0
0.5
1.0
1.5
2.0
2.5
pedigree


Probit Esimtates
pred_probit
0.7
0.5
0.3
0.0
0.5
1.0
1.5
2.0
2.5
pedigree



All models show that the estimate and the variance get larger as pedigree grows.

e

For prediction, we only care about accuracy. we want our results to capture the reality as much as possible. We don't care about interpretability. Using which model solely depends on model performance. We can use cross validation and stepwise regression to find the best model.

Appendix

Q1

```
rm(list = ls())

b1 = c(0.4, 1.1)
b2 = c(0.6, 1.2, 2.5)
b0 = -2.5
n = 1000

sim_one <- function(n, b1, b2){
  z = rnorm(n)
  x = rbinom(n,1,0.5)
  logit1 = b0 + b1 * x + b2 * z
  p_est = exp(logit1)/(1+exp(logit1))
  y = rbinom(n,1,p_est)

  model1 = glm(y~x+z,family = "binomial")
  model2 = glm(y~x, family = "binomial")
```

```
  b1.est1 = model1$coefficients[2]
  b1.est2 = model2$coefficients[2]

  se1 = sqrt(diag(vcov(model1)))[2]
  se2 = sqrt(diag(vcov(model2)))[2]

  ci1 = b1.est1 + se1 %o% c(qnorm(0.025), qnorm(0.975))
  ci2 = b1.est2 + se2 %o% c(qnorm(0.025), qnorm(0.975))
  cover1 = b1 >= ci1[1] & b1 <= ci1[2]
  cover2 = b1 >= ci2[1] & b1 <= ci2[2]

  rej1 = 0 < ci1[1] | 0 > ci1[2]
  rej2 = 0 < ci2[1] | 0 > ci2[2]

  return(c(est1 = b1.est1, est2 = b1.est2, se1 = se1, se2 = se2,
           cov1 = cover1, cov2 = cover2, rej1=rej1, rej2=rej2))

}

#sim_one(n, b1 = 1, b2 = 1)

set.seed(42)
result = c()
sd_result = c()

for (i in 1:2){
  for(j in 1:3){
    res = replicate(5000, sim_one(n, b1[i], b2[j]))
    df = data.frame(apply(res,1,mean))
    df$b1 = b1[i]
    df$b2 = b2[j]
    df$est = rep(c("true model", "missing variable model"), 4)
    result = rbind(result, df)

    df = data.frame(apply(res,1,sd))
    df$b1 = b1[i]
    df$b2 = b2[j]
    df$est = rep(c("true model", "missing variable model"), 4)
    sd_result = rbind(sd_result, df)
  }
}

colnames(result) <- c("Estimate", "b1", "b2", "est_type")
colnames(sd_result) <- c("Estimate", "b1", "b2", "est_type")

res = vector(mode='list', length=4)
ind = c(1,3,5,7)

for (i in 1:4){

  result_sub = result[c(seq(ind[i],42+ind[i], 8), seq(ind[i]+1,42+ind[i]+1, 8) ),]
  sd_sub = sd_result[c(seq(ind[i],42+ind[i], 8), seq(ind[i]+1,42+ind[i]+1, 8) ),]
  result_sub = cbind(result_sub, sd_sub[,1])
```

```
  colnames(result_sub)[5] = "sd"

  res[[i]] = result_sub
}


ggplot(res[[1]], aes(x= b2, y = Estimate, color = est_type)) +
    geom_point() +
    geom_errorbar(aes(ymin=Estimate-sd, ymax=Estimate+sd)) +
    ylab("Beta1 Estimation") +
    ggtitle("Beta1 Esimtates") +
    facet_grid(rows = vars(b1))

ggplot(res[[2]], aes(x= b2, y = Estimate, color = est_type)) +
    geom_point() +
    geom_errorbar(aes(ymin=Estimate-sd, ymax=Estimate+sd)) +
    ylab("Standard Error") +
    ggtitle("Standard Error Esimtates") +
    facet_grid(rows = vars(b1))

ggplot(res[[3]], aes(x= b2, y = Estimate, color = est_type)) +
    geom_point() +
    geom_errorbar(aes(ymin=Estimate-sd, ymax=Estimate+sd)) +
    ylab("Coverage") +
    ggtitle("Coverage Esimtates") +
    facet_grid(rows = vars(b1))

ggplot(res[[4]], aes(x= b2, y = Estimate, color = est_type)) +
    geom_point() +
    geom_errorbar(aes(ymin=Estimate-sd, ymax=Estimate+sd)) +
    ylab("Power") +
    ggtitle("Power Esimtates") +
    facet_grid(rows = vars(b1), labeller = as_labeller(b1))
```

Q2

```
library(mlbench)
data(PimaIndiansDiabetes2)

data <- PimaIndiansDiabetes2[,-c(3,4,5)]
data_clean <- na.omit(data)

head(data_clean)

model_logit <- glm(diabetes~., data = data_clean, family = binomial(link="logit"))
model_probit <- glm(diabetes~., data = data_clean, family = binomial(link="probit"))
model_cloglog <- glm(diabetes~., data = data_clean, family = binomial(link = "cloglog"))

res <- cbind(model_logit$coefficients, sqrt(diag(vcov(model_logit))),
      model_probit$coefficients, sqrt(diag(vcov(model_probit))),
      model_cloglog$coefficients, sqrt(diag(vcov(model_cloglog))))
colnames(res) <- c("Logit Estimate", "Logit SE", "Probit Estimate",
                          "Probit SE", "CLoglog Estimate", "CLoglog SE")
```

```
res

summary(model_logit)

exp(0.115058)
exp(0.035941)
exp(0.087529)
exp(0.920583)

library(MASS)
pedigrees <- seq(min(data_clean$pedigree),max(data_clean$pedigree),length.out=100)
mean_data <- apply(data_clean[,1:5],2,mean)
sim_data <- data.frame(matrix(rep(mean_data, each=100), nrow=100))
colnames(sim_data)[1:5] <- colnames(data_clean)[1:5]
sim_data$pedigree <- pedigrees

sim_data$pred_logit <- predict.glm(model_logit, newdata=sim_data, type="response")
sim_data$pred_probit <- predict.glm(model_probit, newdata=sim_data, type="response")
sim_data$preds_cloglog <- predict.glm(model_cloglog, newdata=sim_data, type="response")

beta_logit <- mvrnorm(1000, mu=model_logit$coefficients, Sigma=vcov(model_logit))
beta_probit <- mvrnorm(1000, mu=model_probit$coefficients, Sigma=vcov(model_probit))
beta_cloglog <- mvrnorm(1000, mu=model_cloglog$coefficients, Sigma=vcov(model_cloglog))

return_y <- function(beta, x, spec){

  if (spec == 1){

    exp(beta%*%x)/(1+exp(beta%*%x))

  }
  else if (spec == 2){

    pnorm(beta%*%x)

  }

  else if (spec == 3){

    1-exp(-exp(beta%*%x))

  }

}




for (i in 1:nrow(sim_data)) {
  x <- as.numeric(c(1,sim_data[i,1:5]))

  logit_y <- apply(beta_logit, 1, function(beta) return_y(beta,x, 1))
  probit_y <- apply(beta_probit, 1, function(beta) return_y(beta,x,2))
  cloglog_y <- apply(beta_cloglog, 1, function(beta) return_y(beta,x,3))
```

```
  sim_data$lower_logit[i] <- quantile(logit_y, 0.025)
  sim_data$upper_logit[i] <- quantile(logit_y, 0.975)

  sim_data$lower_probit[i] <- quantile(probit_y, 0.025)
  sim_data$upper_probit[i] <- quantile(probit_y, 0.975)

  sim_data$lower_cloglog[i] <- quantile(cloglog_y, 0.025)
  sim_data$upper_cloglog[i] <- quantile(cloglog_y, 0.975)
}

colnames(sim_data)

library(ggplot2)
ggplot(sim_data, aes(x= pedigree, y = pred_logit)) +
    geom_line(color = "red") +
    geom_errorbar(aes(ymin= lower_logit, ymax=upper_logit)) +
    ylab("pred_logit") +
    xlab("pedigree") +
    ggtitle("Logit Esimtates")

ggplot(sim_data, aes(x= pedigree, y = pred_probit)) +
    geom_line(color = "red") +
    geom_errorbar(aes(ymin= lower_probit, ymax=upper_probit)) +
    ylab("pred_probit") +
    xlab("pedigree") +
    ggtitle("Probit Esimtates")

ggplot(sim_data, aes(x= pedigree, y = preds_cloglog)) +
    geom_line(color = "red") +
    geom_errorbar(aes(ymin= lower_cloglog, ymax=upper_cloglog)) +
    ylab("preds_cloglog") +
    xlab("pedigree") +
    ggtitle("Cloglog Esimtates")
```