$$L_{\chi}(\theta) = \sum_{j=1}^{n} \left[\chi_{j} \log (\rho_{j}) + (1-\chi_{j}) \log (1-\rho_{j}) \right]$$

$$\frac{1}{(B_{0}, B_{j}, \dots, B_{30})} = \sum_{j=1}^{n} \left[\chi_{j} \left[B_{0} + B_{j} \chi_{j} + \dots + B_{30} \chi_{j} \right] - \log \left(1 + e^{(\beta_{0})} + \frac{1}{1000} \right) \right]$$

$$\frac{1}{\sqrt{1}} \left[B_{0} + B_{j} \chi_{j} + \dots + B_{30} \chi_{j} \right] - \log \left(1 + e^{(\beta_{0})} + \frac{1}{1000} \right)$$

$$\frac{1}{\sqrt{1}} \left[B_{0} + B_{j} \chi_{j} + \dots + B_{30} \chi_{j} \right] - \log \left(1 + e^{(\beta_{0})} + \frac{1}{1000} \right)$$

$$= \chi^{2} \left[(Y_{j} - P_{j}) \right]$$

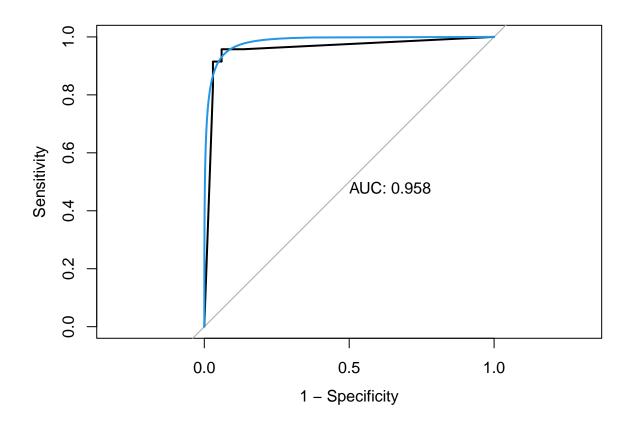
$$= \chi$$

Logistic Newton Raphson Full Model

Hun

3/17/2022

```
cancer_df <- read.csv("~/Downloads/breast-cancer.csv") %>% janitor::clean_names()
data <-
  cancer_df %>% dplyr::select(-id, -x) %>%
 mutate(diagnosis = ifelse(diagnosis == "M", 1, 0)) %>% distinct()
set.seed(7777)
split <- initial_split(data, prop = 0.8)</pre>
training_df <- split %>% training()
testing_df <- split %>% testing()
training_df_5p <- training_df %>% dplyr::select(1:5)
training_df_31p <- training_df</pre>
model_5p <- glm(diagnosis ~ ., data = training_df_5p, family = "binomial")</pre>
model_31p <- glm(diagnosis ~ ., data = training_df_31p, family = "binomial")</pre>
beta1 <- model_5p$coefficients %>% round(digits = 3) %>% broom::tidy()
beta2 <- model_31p$coefficients %>% round(digits = 3) %>% broom::tidy()
test_pred_prob <- predict(model_31p, testing_df, type = "response")</pre>
roc.glm <- roc(testing_df$diagnosis, test_pred_prob)</pre>
plot(roc.glm, legacy.axes = TRUE, print.auc = TRUE)
plot(smooth(roc.glm), col = 4, add = TRUE)
```



Function for log likelihood, gradient, and Hessian

```
logisticstuff <- function(X, y, beta) {
  p <- exp(X %*% beta) / (1+ exp(X %*% beta)) %>% as.vector()
  for (i in 1:length(p)) {
    if (p[i] == 1) {
       p[i] <- 1-1e-8
     }
  }
  loglik <- t(y) %*% log(p) + t(1-y) %*% log(1-p)
  grad <- t(X) %*% (y-p)
  W <- diag(c(p*(1-p)))
  Hess <- -t(X) %*% W %*% X
  return(list(loglik = loglik, grad = grad, Hess = Hess))
}</pre>
```

Newton Raphson with 5 parameters

```
X <- model.matrix(diagnosis~., training_df_5p)</pre>
y <- as.matrix(training_df$diagnosis)</pre>
NewtonRaphson <- function(X, y, logit_func, start, tol=1e-10, maxiter = 200) {</pre>
   i <- 0
   cur_beta <- start</pre>
   stuff <- logit_func(X, y, cur_beta)</pre>
   asc_dir_check <- -t(stuff$grad) %*% solve(stuff$Hess) %*% stuff$grad
   res <- c(i, stuff$loglik, asc dir check, cur beta)
   prevloglik <- -Inf # To make sure it iterates</pre>
   while (i < maxiter && abs(stuff$loglik - prevloglik) > tol) {
     i <- i + 1
     prevloglik <- stuff$loglik</pre>
     prev beta <- cur beta
     cur_beta <- prev_beta - (solve(stuff$Hess) %*% stuff$grad) #update beta
     stuff <- logit_func(X, y, cur_beta) #update log likelihood, gradient, Hessia</pre>
     asc_dir_check <- -t(stuff$grad) %*% solve(stuff$Hess) %*% stuff$grad
     res <- rbind(res, c(i, stuff$loglik, asc_dir_check, cur_beta))</pre>
     colnames(res) <- c("Number of trial", "Log_likelihood", "asc_dir_check", paste0("Beta", 0:4))</pre>
   }
   return(res)
coef <- rep(0,ncol(X)) # Randomly assigned coefficients (starting point)</pre>
ans <- NewtonRaphson(X, y, logisticstuff, coef) %>% data.frame() %>% `rownames<-`( NULL )
ans %>% kbl(caption = "Newton Raphson result with 5 parameters") %>%
  kable_styling(font_size = 8, latex_options = "HOLD_position")
```

Table 1: Newton Raphson result with 5 parameters

Number.of.trial	Log_likelihood	asc_dir_check	Beta0	Beta1	Beta2	Beta3	Beta4
0	-315.38197	297.4027390	0.0000000	0.000000	0.0000000	0.0000000	0.0000000
1	-141.92524	64.4620788	-8.2741164	-1.259642	0.0795466	0.2861187	-0.0034815
2	-100.99962	25.2067494	-12.8134881	-2.880231	0.1358457	0.5775260	-0.0044236
3	-85.08700	9.5366502	-13.7360236	-4.995863	0.1916826	0.8709006	-0.0001595
4	-79.07361	3.3687360	-8.6473294	-7.581528	0.2344300	1.1188065	0.0123355
5	-77.06852	0.4321716	-0.2645781	-10.005234	0.2577985	1.2831011	0.0289108
6	-76.83723	0.0057185	2.8702542	-10.980122	0.2685915	1.3503646	0.0357635
7	-76.83435	0.0000010	3.1959332	-11.094010	0.2700253	1.3588134	0.0365377
8	-76.83435	0.0000000	3.2001911	-11.095519	0.2700442	1.3589267	0.0365478
9	-76.83435	0.0000000	3.2001918	-11.095520	0.2700442	1.3589267	0.0365478

Fitted glm model Beta0: -3.2 , Beta1: -11.096 , Beta2: 0.27 , Beta3: 1.359 , Beta4: 0.037

Newton Raphson with all 31 parameters

```
logisticstuff <- function(X, y, beta) {</pre>
  p <- exp(X%*%beta) / (1 + exp(X%*%beta)) %>% as.vector()
  for (i in 1:length(p)) {
    if (p[i] == 1) {
      p[i] \leftarrow 1-2e-8
  }
  loglik \leftarrow t(y) %*% log(p) + t(1-y) %*% log(1-p)
  grad <- t(X) %*% (y-p)
  W \leftarrow diag(c(p*(1-p)))
  Hess <- -t(X) %*% W %*% X
  return(list(loglik = loglik, grad = grad, Hess = Hess))
X <- model.matrix(diagnosis~., training_df_31p)</pre>
y <- as.matrix(training_df$diagnosis)</pre>
NewtonRaphson <- function(X, y, logit_func, start, tol=1e-10, maxiter = 43) {</pre>
   i <- 0
   cur_beta <- start</pre>
   stuff <- logit_func(X, y, cur_beta)</pre>
   asc_dir_check <- -t(stuff$grad) %*% solve(stuff$Hess) %*% stuff$grad
   res <- c(i, stuff$loglik, asc_dir_check, cur_beta)</pre>
   prevloglik <- -Inf # To make sure it iterates</pre>
   while (i < maxiter && abs(stuff$loglik - prevloglik) > tol) {
     i < -i + 1
     prevloglik <- stuff$loglik</pre>
     prev_beta <- cur_beta</pre>
     cur_beta <- prev_beta - (solve(stuff$Hess) %*% stuff$grad) #update beta</pre>
     stuff <- logit_func(X, y, cur_beta) #update log likelihood, gradient, Hessia</pre>
     asc_dir_check <- -t(stuff$grad) %*% solve(stuff$Hess) %*% stuff$grad
     res <- rbind(res, c(i, stuff$loglik, asc_dir_check, cur_beta))</pre>
     colnames(res) <- c("Number_of_trial", "Log_likelihood", "asc_dir_check", paste0("Beta", 0:30))</pre>
   }
   return(res)
}
coef <- rep(0,ncol(X)) # Randomly assigned coefficients (starting point)</pre>
ans <- NewtonRaphson(X, y, logisticstuff, coef) %>% data.frame() %>% `rownames<-`( NULL )
ans %>% kbl(caption = "Newton Raphson result with 31 parameters") %>%
   kable_styling(font_size = 8, latex_options = "HOLD_position")
```

Table 2: Newton Raphson result with 31 parameters

0 - 315_3819672 363_386380 0.00000 0.000000 0.0000000 0.0000000 0.000000	Number of trial	Log likelihood	asc dir check	Beta0	Beta1	Beta2	Beta3	Beta4	Beta5
2 - 58.3133687	0	-315.3819672	363.3863080	0.00000	0.0000000	0.0000000	0.0000000	0.0000000	0.00000
3	1	-105.0649679	72.6858805	-10.22033	-0.9360000	0.0252396	0.0861849	0.0023765	3.64103
3	2	-58.3133087	34.7575625	-17.49844	-0.9011090	0.0147350	0.0982375	0.0023645	11.36727
4 -23.2348007 12.0995319 -36.83889 -1.3710840 -0.0871143 0.239244 0.001870 24.70603 5 -1.5.0488775 10.031818 -56.77922 1.9465817 -0.0753329 0.0299581 0.0028944 44.85133 6 -8.2985550 7.2261060 -91.13019 -3.8130106 0.0045095 0.2249788 0.0152424 101.57375 7 -3.6072702 3.4217703 -143.73478 -6.9905355 0.1232558 0.3089762 0.0326588 173.14779 8 -1.4255469 1.3794906 -0.0419664 1.02675614 0.1853259 0.0466945 68.69662 9 -0.5481015 0.5370347 -205.838951 -0.147424919 0.2491523 0.7650496 0.0641200 380.75184 10 -0.208142 0.0357374 -265.83920 -14.7424919 0.2491523 0.7569496 0.0641200 380.75184 11 -0.0781367 0.0285672 -448.15100 -37.385849 0.28174121 1.141114 0.0839141 1.1419404 0.0032424 </td <td>3</td> <td></td> <td></td> <td></td> <td>-0.9217526</td> <td>-0.0361649</td> <td></td> <td>0.0014547</td> <td>19.33863</td>	3				-0.9217526	-0.0361649		0.0014547	19.33863
6 -8.2955550 7.2261060 -91.13019 -3.8130106 0.0045095 0.2249788 0.0152424 101.57375 7 7 -3.6072702 3.4217703 -143.73478 -6.9205335 0.1232558 0.3089762 0.0326358 173.14779 8 -1.4255469 1.3794906 -204.19664 -10.2275614 0.1985325 0.4651119 0.0460945 268.69662 9 -0.5489105 0.5370347 265.83920 -14.7424919 0.2491523 0.7565496 0.0614260 380.75184 10 -0.2081442 0.2050172 -327.39849 -20.8328954 0.291716 1.1944761 0.0883038 507.67285 11 -0.0781367 0.0769959 -387.87498 -228.53494951 0.3403048 1.7432944 0.1223028 643.62385 12 -0.0225994 0.0225672 -448.15100 -373.885878 0.4003243 2.382190 1.625516 781.60031 13 -0.0110700 0.0105491 -511.49230 -47.1248196 0.4628012 3.2150915 0.1999765 924.99196 14 -0.0042974 0.0039108 -581.39421 -55.5908834 0.5219444 4.996386 0.2274242 1080.59892 15 -0.0017537 0.001675 -659.95049 -71.8607992 0.570416 -3.997565 924.99196 16 -0.0007773 0.0005637 -743.86905 -848.716500 0.5851627 8.7517464 0.2063349 1330.26572 17 -0.0003896 0.0002290 -825.33800 -961.825506 0.53805127 8.7517464 0.2063349 1360.26572 18 -0.0001580 0.000209 -825.33800 -961.825506 0.53805127 8.7517464 0.2063349 1360.26572 20 -0.0001088 0.0000362 -989.42316 -115.9561635 0.3381674 1.2189731 0.1563408 1449.67579 20 -0.0001080 0.0000577 -948.24398 -111.1803630 0.3844162 14.8097315 0.0154078 1601.37158 21 -0.0000888 0.00000362 -989.42316 -115.9561635 0.3380692 -0.0004920 1609.90557 24 -0.0000757 0.0000147 1.070.34397 125.2189261 0.2862867 18.0036925 -0.0004920 1609.9057 24 -0.0000757 0.000017 1.005.47077 1.191.702744 0.2862300 18.807306 -0.0165144 1996.90544 25 -0.0000045 0.000017 1.106.87767 1.212141741 0.2821007 18.807306 -0.0165144 1996.90544 26 -0.0000553 0.0000067 1.118.18717 1.136.0944755 0.288342 19.118748 -0.0093852 1664.14261 25 -0.0000045 0.000007 1.118.05877 1.2187471 0.2821007 18.807306 -0.0165144 1996.90557 24 -0.0000108 0.000007 1.118.05877 1.2187471 0.2821007 18.807306 -0.0165144 1996.90557 25 -0.00000405 0.000007 1.118.05877 1.2187471 0.2821007 18.807309 0.0106282 1746.0703 25 -	4	-23.2348067	12.0995319		-1.3710840	-0.0871143	0.2339244	0.0016870	24.70603
7 -3.6072702 3.4217703 -1.43.73478 -6.9205335 0.1232558 0.0809762 0.036358 17314779 8 -1.4255469 1.3794906 -204.19664 -10.2275614 0.1985325 0.4651119 0.046945 268.69662 9 -0.5489105 0.5370347 -265.83920 -14.7424919 0.2491701 1.194761 0.0803038 507.67285 10 -0.2081442 0.2050172 -327.39849 -28.8328954 0.2917701 1.194761 0.083038 507.67285 12 -0.0232964 0.0285672 -48.15400 -37.3858578 0.4003243 2.3821901 0.1625516 781.60031 14 -0.0042974 0.0039108 -581.39421 -58.5908834 0.5219444 4.4926386 0.2274242 1080.59892 15 -0.0017537 0.0016675 -69.95049 -71.8607992 0.5704151 6.3975175 0.2331574 1235.62022 16 -0.000773 0.000537 -743.86695 -84.8716800 0.581670 0.5757155 0.2331574 1235.6202	5	-15.0488775	10.0318418	-56.77922	-1.9465817	-0.0753320	0.2490561	0.0028944	44.85133
8 -1.4255469	6	-8.2985550	7.2261060	-91.13019	-3.8130106	0.0045095	0.2249788	0.0152424	101.57375
9	7	-3.6072702	3.4217703	-143.73478	-6.9205335	0.1232558	0.3089762	0.0326358	173.14779
10	8	-1.4255469	1.3794906	-204.19664	-10.2275614	0.1985325	0.4651119	0.0466945	268.69662
11	9	-0.5489105	0.5370347	-265.83920	-14.7424919	0.2491523	0.7565496	0.0641260	
12	10	-0.2081442	0.2050172	-327.39849	-20.8328954	0.2917701	1.1944761	0.0883038	507.67285
13	11	-0.0781367	0.0769959	-387.87498	-28.5404951	0.3403048	1.7432844	0.1223028	643.62385
14	12	-0.0292694	0.0285672	-448.15400	-37.3858578	0.4003243	2.3821901	0.1625516	781.60031
15	13	-0.0110700	0.0105491	-511.49230	-47.1248196	0.4628012	3.2150915	0.1999765	924.99196
16	14	-0.0042974	0.0039108	-581.39421	-58.5908834	0.5219444	4.4926386	0.2274242	1080.59892
17	15	-0.0017537	0.0014675	-659.95049	-71.8607992	0.5704151		0.2331574	1235.62022
18	16	-0.0007773	0.0005637	-743.86695	-84.8716500	0.5851627	8.7517464	0.2063349	1360.26572
19	17	-0.0003896	0.0002290	-825.33800	-96.1825906	0.5380378	11.2189731	0.1563408	1449.67579
20 -0.0001088 0.0000362 -989.42316 -115.9561635 0.3380406 15.9423143 0.0450139 1601.37158 21 -0.0000838 0.0000251 -1021.77072 -119.7020744 0.3101736 16.8003212 0.0254486 1629.61455 22 -0.0000657 0.0000147 -1070.34397 -125.2189261 0.2862967 18.0036925 -0.0004920 1669.30957 24 -0.000471 0.0000120 -1089.57958 -127.3208215 0.2827493 18.4417318 -0.0093852 1684.14261 25 -0.0000353 0.00000101 -1106.57767 -129.1241741 0.2821007 18.8073966 -0.0165144 1696.96844 26 -0.0000353 0.000007 -1121.84171 -130.6944755 0.2833342 19.1173456 -0.0223029 1708.34843 27 -0.0000311 0.0000077 -1135.72707 -133.0787108 0.2858292 19.3833668 -0.0270463 1718.63472 28 -0.0000247 0.0000062 -1160.33592 -134.4193253 0.2931695 19.8157464	18	-0.0002265	0.0001058	-894.49775	-104.8654488	0.4549612	13.2799599	0.1084977	1515.13021
21 -0.0000838 0.0000251 -1021.77072 -119.7020744 0.3101736 16.8003212 0.0254486 1629.61455 22 -0.0000673 0.0000187 -1048.12706 -122.7243020 0.2944886 17.4688006 0.0107932 1651.59365 23 -0.0000471 0.0000120 -1089.57958 -127.3208215 0.2827493 18.4417318 -0.0093852 1684.14261 25 -0.0000405 0.0000101 -1106.57767 -129.1241741 0.282007 18.8073906 -0.0165144 1696.96844 26 -0.0000353 0.0000087 -1121.84171 -130.6944755 0.2833342 19.1173456 -0.0223029 1708.34483 27 -0.0000311 0.0000069 -1148.49342 -133.3115746 0.289129 19.6140282 -0.0309581 1728.08208 28 -0.0000249 0.0000662 -1160.33592 -134.4193253 0.2931695 19.8157464 -0.0341966 1736.85646 30 -0.0000249 0.0000053 -1181.88530 -136.3696944 0.3029300 20.1582094	19	-0.0001500	0.0000577	-948.24398	-111.1803630	0.3844162	14.8097315	0.0717586	1564.34692
22 -0.0000673 0.000187 -1048.12706 -122.7243020 0.2944686 17.4688006 0.0107932 1651.59365 23 -0.0000557 0.0000147 -1070.34397 -125.2189261 0.2862967 18.0036925 -0.0004920 1669.30957 24 -0.0000405 0.000010 -106.57767 -129.1241741 0.2821007 18.8073906 -0.0165144 1696.96844 26 -0.0000353 0.00000087 -1121.84171 -130.6944755 0.2833342 19.1173456 -0.0223029 1708.34483 27 -0.0000311 0.0000077 -1135.72707 -132.0787108 0.2858292 19.3833668 -0.0270463 1718.63472 28 -0.0000277 0.0000069 -1148.49342 -133.3115746 0.2891929 19.6140282 -0.039581 1728.08208 29 -0.0000249 0.0000057 -1171.40474 -135.4222510 0.297586 19.9934334 -0.0368822 1745.07935 31 -0.000024 0.0000053 -1181.88530 -136.3696944 0.302930 20.1582094	20	-0.0001088	0.0000362	-989.42316	-115.9561635	0.3380406	15.9423143	0.0450139	1601.37158
23 -0.0000557 0.0000147 -1070.34397 -125.2189261 0.2862967 18.0036925 -0.0004920 1669.30957 24 -0.0000471 0.0000120 -1089.57958 -127.3208215 0.2827493 18.4417318 -0.0093852 1684.14261 25 -0.0000405 0.0000101 -1106.57767 -129.1241741 0.2821007 18.8073906 -0.0165144 1696.96844 26 -0.0000353 0.0000077 -131.84171 -130.6944755 0.2833342 19.1173456 -0.0223029 1708.34483 27 -0.0000311 0.0000077 -1135.72707 -132.0787108 0.2858292 19.3833668 -0.0270463 1718.63472 28 -0.0000277 0.0000069 -1148.49342 -133.3115746 0.2891929 19.6140282 -0.0309581 1728.08208 29 -0.0000249 0.0000062 -1160.33592 -134.4193253 0.2931695 19.8157464 -0.0341966 1736.85646 30 -0.0000249 0.0000050 -1191.80551 -135.4222510 0.2975886 19.9934334	21	-0.0000838	0.0000251	-1021.77072	-119.7020744	0.3101736	16.8003212	0.0254486	1629.61455
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	22	-0.0000673	0.0000187	-1048.12706	-122.7243020	0.2944686	17.4688006	0.0107932	1651.59365
25 -0.0000405 0.0000101 -1106.57767 -129.1241741 0.2821007 18.8073906 -0.0165144 1696.96844 26 -0.0000353 0.0000087 -1121.84171 -130.6944755 0.2833342 19.1173456 -0.0223029 1708.34483 27 -0.0000311 0.0000067 -1135.72707 -132.0787108 0.2858292 19.3833668 -0.0270463 1718.63472 28 -0.0000247 0.000069 -1148.49342 -133.3115746 0.2891929 19.6140282 -0.039581 1728.08208 29 -0.0000249 0.0000062 -1160.33592 -134.4193253 0.2931695 19.8157464 -0.0341966 1736.85646 30 -0.0000225 0.0000057 -1171.40474 -135.4222510 0.2975886 19.9934334 -0.0341966 1745.07935 31 -0.0000244 0.0000053 -1181.88530 -135.4222510 0.2975886 19.9934334 -0.0392283 1752.73140 32 -0.0000187 0.0000050 -1191.80581 -137.2400652 0.3081387 20.3055441	23	-0.0000557	0.0000147	-1070.34397	-125.2189261	0.2862967	18.0036925	-0.0004920	1669.30957
26 -0.0000353 0.0000087 -1121.84171 -130.6944755 0.2833342 19.1173456 -0.023029 1708.34483 27 -0.0000311 0.0000077 -1135.72707 -132.0787108 0.2858292 19.3833668 -0.0270463 1718.63472 28 -0.0000277 0.0000069 -1148.49342 -133.3115746 0.2891929 19.6140282 -0.0309581 1728.08208 29 -0.0000249 0.0000062 -1160.33592 -134.4193253 0.2931695 19.8157464 -0.0341966 1736.85646 30 -0.0000225 0.0000057 -1171.40474 -135.4222510 0.2975886 19.9934334 -0.0392283 1752.73140 31 -0.0000187 0.0000053 -1181.88530 -136.3696944 0.3029300 20.1582094 -0.0392283 1752.73140 32 -0.0000187 0.0000055 -191.80581 -137.2400652 0.3081387 20.3055441 -0.0411818 1760.10577 33 -0.0000171 0.0000047 -1201.23849 -138.041882 0.313401 20.4380993	24	-0.0000471	0.0000120	-1089.57958	-127.3208215	0.2827493	18.4417318	-0.0093852	1684.14261
27 -0.0000311 0.0000077 -1135.72707 -132.0787108 0.2858292 19.3833668 -0.0270463 1718.63472 28 -0.0000277 0.0000069 -1148.49342 -133.3115746 0.2891929 19.6140282 -0.039581 1728.08208 29 -0.0000249 0.0000062 -1160.33592 -134.4193253 0.2931695 19.8157464 -0.0341966 1736.85646 30 -0.0000225 0.0000057 -1171.40474 -135.4222510 0.2975886 19.9934334 -0.0388822 1745.07935 31 -0.0000204 0.0000053 -1181.88530 -136.3696944 0.3029300 20.1582094 -0.0392283 1752.73140 32 -0.0000187 0.0000055 -1191.80581 -137.2400652 0.3081387 20.3055441 -0.0411818 1760.10577 33 -0.0000171 0.00000047 -1201.23849 -138.0441882 0.3134401 20.4380993 -0.0428086 1767.14707 34 -0.0000158 0.0000044 -1210.24761 -138.7906760 0.3188614 20.5579467 <td>25</td> <td>-0.0000405</td> <td>0.0000101</td> <td>-1106.57767</td> <td>-129.1241741</td> <td>0.2821007</td> <td>18.8073906</td> <td>-0.0165144</td> <td>1696.96844</td>	25	-0.0000405	0.0000101	-1106.57767	-129.1241741	0.2821007	18.8073906	-0.0165144	1696.96844
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	26	-0.0000353	0.0000087	-1121.84171	-130.6944755	0.2833342	19.1173456	-0.0223029	1708.34483
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	27	-0.0000311	0.0000077	-1135.72707	-132.0787108	0.2858292	19.3833668	-0.0270463	1718.63472
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	28	-0.0000277	0.0000069	-1148.49342	-133.3115746	0.2891929	19.6140282	-0.0309581	1728.08208
$\begin{array}{cccccccccccccccccccccccccccccccccccc$		-0.0000249	0.0000062	-1160.33592		0.2931695	19.8157464	-0.0341966	1736.85646
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	30	-0.0000225	0.0000057	-1171.40474	-135.4222510	0.2975886	19.9934334	-0.0368822	1745.07935
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$		-0.0000204	0.0000053	-1181.88530		0.3029300	20.1582094	-0.0392283	1752.73140
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	32	-0.0000187	0.0000050		-137.2400652	0.3081387	20.3055441	-0.0411818	1760.10577
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$		-0.0000171		-1201.23849	-138.0441882	0.3134401	20.4380993	-0.0428086	1767.14707
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	34	-0.0000158	0.0000044	-1210.24761	-138.7906760	0.3188614	20.5579467	-0.0441587	1773.88132
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	35	-0.0000146	0.0000042		-139.4865934		20.6667661	-0.0452721	
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$		-0.0000136			-140.1378168	0.3300422	20.7659482	-0.0461820	1786.55470
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	37	-0.0000127	0.0000039	-1235.22547	-140.7492787	0.3357891	20.8566603	-0.0469160	1792.54497
$\begin{array}{c ccccccccccccccccccccccccccccccccccc$	38	-0.0000119	0.0000037	-1243.04230	-141.3264902	0.3406743	20.9406962	-0.0475476	1798.31437
41 -0.0000099 0.0000034 -1265.13967 -142.8796968 0.3576664 21.1557643 -0.0487040 1814.51722 42 -0.0000093 0.0000033 -1272.14572 -143.3456263 0.3636008 21.2171900 -0.0488987 1819.60084	39	-0.0000111	0.0000036	-1250.61131	-141.8726644	0.3461432	21.0181468	-0.0480439	1803.88539
42 -0.0000093 0.0000033 -1272.14572 -143.3456263 0.3636008 21.2171900 -0.0488987 1819.60084	40	-0.0000105	0.0000035		-142.3895255	0.3518382	21.0896134	-0.0484238	1809.28189
430_0000089	42	-0.0000093	0.0000033	-1272.14572	-143.3456263	0.3636008	21.2171900	-0.0488987	1819.60084
10 -0.0000000 0.00000000 -1210.00000 -140.1004012 0.000000000 21.2144002 -0.0430201 1024.04000	43	-0.0000089	0.0000032	-1279.00343	-143.7894612	0.3696339	21.2744032	-0.0490201	1824.54059

Table 3: Failed Newton Rapshon for 31 parameters

values

Result

Beta25

Beta26

Beta27

Beta28 Beta29

Beta30

Number_of_trial -0.000009341111 Log_likelihood 0.000003304651 asc_dir_check -1272.146 Beta0 Beta1 -143.3456 Beta2 0.3636008 21.21719 Beta3 Beta4 -0.04889868 Beta5 1819.601 Beta6 -1630.311 Beta7 929.9978 Beta8 342.2007 Beta9 -442.6396 Beta10 2262.871 Beta11 -28.39532 Beta12 -22.86274 Beta13 -3.83439 Beta14 2.290926Beta15 12389.41Beta165063.34Beta17 -1606.132 Beta18 471.5062 Beta19 -5288.991 Beta20 -45056.05 Beta21 121.76 Beta22 5.032851Beta23 -5.146386 Beta24 -0.7058646

-923.8527

-379.2067

 $\frac{124.5956}{502.6894}$

703.1726

2613.257

Table 4: Fitted glm model coefficients of 31 parameters

names	x			
(Intercept)	-1293.417			
radius_mean	-172.482			
texture_mean	1.066			
perimeter_mean	13.475			
area_mean	0.718			
smoothness_mean	2785.956			
compactness_mean	-1775.940			
concavity_mean	1424.757			
concave_points_mean	473.400			
symmetry_mean	-345.545			
fractal_dimension_mean	3672.362			
radius_se	-254.806			
texture_se	-42.316			
perimeter_se	-7.104			
area_se	4.773			
smoothness_se	15391.485			
compactness_se	5711.477			
concavity_se	-2162.101			
concave_points_se	4968.934			
symmetry_se	-7212.775			
fractal_dimension_se	-55050.387			
radius_worst	178.104			
texture_worst	6.587			
perimeter_worst	-6.294			
area_worst	-1.114			
smoothness_worst	-1623.965			
compactness_worst	-402.553			
concavity_worst	160.429			
concave_points_worst	134.278			
symmetry_worst	875.003			
fractal_dimension_worst	3118.667			

Modified Newton Raphson with 5 number of parameters

```
logisticstuff <- function(X, y, beta) {</pre>
  p <- exp(X%*%beta) / (1+ exp(X%*%beta)) %>% as.vector()
  for (i in 1:length(p)) {
    if (p[i] == 1) {
      p[i] <- 1-1e-8
  }
  loglik \leftarrow t(y) %*% log(p) + t(1-y) %*% log(1-p)
  grad <- t(X) %*% (y-p)
  W \leftarrow diag(c(p*(1-p)))
  Hess <- -t(X) %*% W %*% X
  return(list(loglik = loglik, grad = grad, Hess = Hess))
X <- model.matrix(diagnosis~., training_df_5p)</pre>
y <- as.matrix(training_df$diagnosis)</pre>
NewtonRaphson_mod <- function(X, y, logit_func, start, tol=1e-10, maxiter = 200) {</pre>
   i <- 0
   cur_beta <- start</pre>
   stuff <- logit_func(X, y, cur_beta)</pre>
   asc_dir_check <- -t(stuff$grad) %*% solve(stuff$Hess) %*% stuff$grad
   lambda <- 1 #initial random lambda
   res <- c(i, stuff$loglik, asc_dir_check, cur_beta)
   prevloglik <- -Inf # To make sure it iterates</pre>
   while (i < maxiter && abs(stuff$loglik - prevloglik) > tol) {
      i < -i + 1
      prev_beta <- cur_beta</pre>
      #checking if direction is ascent. If not, transform Hessian into negative definite.
      if (asc dir check < 0) {</pre>
         stuff$Hess = stuff$Hess - (max(stuff$Hess) + 5)
         prev_beta <- prev_beta - lambda * (solve(stuff$Hess) %*% stuff$grad)</pre>
          stuff <- logit_func(X, y, prev_beta)</pre>
         prevloglik <- stuff$loglik</pre>
      }
      else {
         prev_beta <- prev_beta - lambda * (solve(stuff$Hess) %*% stuff$grad)</pre>
          stuff <- logit_func(X, y, prev_beta)</pre>
          prevloglik <- stuff$loglik</pre>
      }
      cur2 beta <- prev beta - lambda * (solve(stuff$Hess) %*% stuff$grad)</pre>
      stuff2 <- logit_func(X, y, cur2_beta)</pre>
      #condition check before step halving process
      if (stuff2$loglik > prevloglik) {
          cur_beta = cur2_beta
         stuff = stuff2
```

```
asc_dir_check <- -t(stuff$grad) %*% solve(stuff$Hess) %*% stuff$grad
      }
      #step halving process
      else {
         repeat {
         lambda = lambda/2
         cur_beta = prev_beta - lambda * (solve(stuff$Hess) %*% stuff$grad)
         stuff <- logit_func(X, y, cur_beta)</pre>
         if (stuff$loglik > prevloglik) {
         cur_beta = cur_beta
         stuff = stuff
         asc_dir_check <- -t(stuff$grad) %*% solve(stuff$Hess) %*% stuff$grad
         break}
         }
      res <- rbind(res, c(i, stuff$loglik, asc_dir_check, cur_beta))</pre>
      colnames(res) <- c("Number of trial", "Log_likelihood", "asc_dir_check", paste0("Beta", 0:4))</pre>
  }
  return(res)
coef <- rep(0,ncol(X)) # Randomly assigned coefficients (starting point)</pre>
ans <- NewtonRaphson_mod(X, y, logisticstuff, coef) %>% data.frame() %>% `rownames<-`( NULL )
ans %>% kbl(caption = "Newton Raphson result with 5 parameters") %>%
  kable_styling(font_size = 8, latex_options = "HOLD_position")
```

Table 5: Newton Raphson result with 5 parameters

Number.of.trial	Log_likelihood	asc_dir_check	Beta0	Beta1	Beta2	Beta3	Beta4
0	-315.38197	297.4027390	0.000000	0.000000	0.0000000	0.000000	0.0000000
1	-100.99962	25.2067494	-12.813488	-2.880231	0.1358457	0.577526	-0.0044236
2	-79.07361	3.3687360	-8.647329	-7.581528	0.2344300	1.118807	0.0123355
3	-76.83723	0.0057185	2.870254	-10.980122	0.2685915	1.350365	0.0357635
4	-76.83435	0.0000000	3.200191	-11.095519	0.2700442	1.358927	0.0365478
5	-76.83435	0.0000000	3.200192	-11.095520	0.2700442	1.358927	0.0365478

Modified Newton Raphson with all 31 parameters

```
logisticstuff <- function(X, y, beta) {
  p <- exp(X%*%beta) / (1 + exp(X%*%beta)) %>% as.vector()
  for (i in 1:length(p)) {
    if (p[i] == 1) {
       p[i] <- 1-2e-8
    }
  }
  loglik <- t(y) %*% log(p) + t(1-y) %*% log(1-p)
  grad <- t(X) %*% (y-p)
  W <- diag(c(p*(1-p)))
  Hess <- -t(X) %*% W %*% X</pre>
```

```
return(list(loglik = loglik, grad = grad, Hess = Hess))
}
X <- model.matrix(diagnosis~., training_df_31p)</pre>
y <- as.matrix(training_df$diagnosis)</pre>
NewtonRaphson_mod <- function(X, y, logit_func, start, tol=1e-10, maxiter = 21) {</pre>
   cur beta <- start
   stuff <- logit_func(X, y, cur_beta)</pre>
   asc_dir_check <- -t(stuff$grad) %*% solve(stuff$Hess) %*% stuff$grad
   res <- c(i, stuff$loglik, asc_dir_check, cur_beta)</pre>
   prevloglik <- -Inf # To make sure it iterates</pre>
   lambda <- 1 #initial random lambda
   while (i < maxiter && abs(stuff$loglik - prevloglik) > tol) {
      i <- i + 1
      prev_beta <- cur_beta</pre>
      #checking if direction is ascent. If not, transform Hessian into negative definite.
      if (asc_dir_check < 0) {</pre>
         stuff$Hess = stuff$Hess - (max(stuff$Hess) + 5)
         prev_beta <- prev_beta - lambda * (solve(stuff$Hess) %*% stuff$grad)</pre>
         stuff <- logit_func(X, y, prev_beta)</pre>
         prevloglik <- stuff$loglik</pre>
      }
      else {
         prev_beta <- prev_beta - lambda * (solve(stuff$Hess) %*% stuff$grad)</pre>
         stuff <- logit_func(X, y, prev_beta)</pre>
         prevloglik <- stuff$loglik</pre>
      }
      cur2_beta <- prev_beta - lambda * (solve(stuff$Hess) %*% stuff$grad)</pre>
      stuff2 <- logit_func(X, y, cur2_beta)</pre>
      #condition check before step halving process
      if (stuff2$loglik > prevloglik) {
         cur_beta = cur2_beta
         stuff = stuff2
         asc_dir_check <- -t(stuff$grad) %*% solve(stuff$Hess) %*% stuff$grad
      }
      #step halving process
      else {
         repeat {
         lambda = lambda/2
         cur_beta = prev_beta - lambda2 * (solve(stuff$Hess) %*% stuff$grad)
         stuff <- logit_func(X, y, cur_beta)</pre>
         if (stuff$loglik > prevloglik) {
         cur_beta = cur_beta
         stuff = stuff
         asc_dir_check <- -t(stuff$grad) %*% solve(stuff$Hess) %*% stuff$grad
```

```
    break}
    break}
}

res <- rbind(res, c(i, stuff$loglik, asc_dir_check, cur_beta))
    colnames(res) <- c("Number of trial", "Log_likelihood", "asc_dir_check", pasteO("Beta", 0:30))
}

return(res)
}

coef <- rep(0,ncol(X)) # Randomly assigned coefficients (starting point)

ans <- NewtonRaphson_mod(X, y, logisticstuff, coef) %>% data.frame() %>% `rownames<-`( NULL )

ans %>% kbl(caption = "Newton Raphson result with 5 parameters") %>%
    kable_styling(font_size = 8, latex_options = "HOLD_position")
```

Table 6: Newton Raphson result with 5 parameters

Number.of.trial Log_likelihoo	d asc dir check	Beta0	Beta1	Beta2	Beta3	Beta4	D-4-F	
		Detao	Detai	Detaz	Detas	рега4	Beta5	
0 -315.381967	2 363.3863080	0.00000	0.000000	0.0000000	0.0000000	0.0000000	0.00000	
1 -58.313308	7 34.7575625	-17.49844	-0.901109	0.0147350	0.0982375	0.0023645	11.36727	-
2 -23.234806	7 12.0995319	-36.38389	-1.371084	-0.0871143	0.2339244	0.0016870	24.70603	-
3 -8.298555	7.2261060	-91.13019	-3.813011	0.0045095	0.2249788	0.0152424	101.57375	-
4 -1.425546	1.3794906	-204.19664	-10.227561	0.1985325	0.4651119	0.0466945	268.69662	-2
5 -0.208144	2 0.2050172	-327.39849	-20.832895	0.2917701	1.1944761	0.0883038	507.67285	-3
6 -0.029269	0.0285672	-448.15400	-37.385858	0.4003243	2.3821901	0.1625516	781.60031	-5
7 -0.004297	0.0039108	-581.39421	-58.590883	0.5219444	4.4926386	0.2274242	1080.59892	-7
8 -0.000777	0.0005637	-743.86695	-84.871650	0.5851627	8.7517464	0.2063349	1360.26572	-9
9 -0.000226	0.0001058	-894.49775	-104.865449	0.4549612	13.2799599	0.1084977	1515.13021	-11
10 -0.000108	0.0000362	-989.42316	-115.956164	0.3380406	15.9423143	0.0450139	1601.37158	-12
11 -0.000067	0.0000187	-1048.12706	-122.724302	0.2944686	17.4688006	0.0107932	1651.59365	-13
12 -0.000047	0.0000120	-1089.57958	-127.320821	0.2827493	18.4417318	-0.0093852	1684.14261	-14
13 -0.000035	0.0000087	-1121.84171	-130.694476	0.2833342	19.1173456	-0.0223029	1708.34483	-14
14 -0.000027	7 0.0000069	-1148.49342	-133.311575	0.2891929	19.6140282	-0.0309581	1728.08208	-14
15 -0.000022	0.0000057	-1171.40474	-135.422251	0.2975886	19.9934334	-0.0368822	1745.07935	-15
16 -0.000018	7 0.0000050	-1191.80581	-137.240065	0.3081387	20.3055441	-0.0411818	1760.10577	-15
17 -0.000015	0.0000044	-1210.24761	-138.790676	0.3188614	20.5579467	-0.0441587	1773.88132	-15
18 -0.000013	0.0000040	-1227.20008	-140.137817	0.3300422	20.7659482	-0.0461820	1786.55470	-15
19 -0.000011	0.0000037	-1243.04230	-141.326490	0.3406743	20.9406962	-0.0475476	1798.31437	-15
20 -0.000010	0.0000035	-1257.96833	-142.389525	0.3518382	21.0896134	-0.0484238	1809.28189	-16
21 -0.000009	0.0000033	-1272.14572	-143.345626	0.3636008	21.2171900	-0.0488987	1819.60084	-16

Conundrum:

It is to be observed most of the absolute values of β_i continue to increase as Newton Raphson algorithm proceeds. This causes some of the elements in p vector to be very close to 1, leading some of the elements in log(1-p) vector to be negative infinity and hence the next log likelihood to diverge to negative infinity. As a result, Newton Raphson algorithm cannot go further till its convergence of maximum likelihood estimation.

Proof for why Newton Raphson algorithm cannot reach convergence

```
glm_model_beta_vector <- beta2[2] %>% pull()

p <- exp(X %*% glm_model_beta_vector) / (1 + exp(X %*% glm_model_beta_vector))</pre>
```

```
which(p == 1)

## [1]  3  15  19  21  25  26  29  30  33  37  38  39  41  42  44  45  48  52
## [19]  58  60  62  68  70  71  72  76  84  88  92  95  97  99  100  103  104  107
## [37]  108  109  110  111  113  117  119  121  122  128  129  133  138  139  141  146  150  153
## [55]  161  169  171  185  186  188  190  196  197  198  199  201  202  204  206  210  222  224
## [73]  227  239  244  260  261  263  265  266  268  270  271  274  277  281  285  287  288  289
## [91]  290  294  296  300  305  307  309  325  327  328  331  334  335  339  343  345  348  352
## [109]  356  361  368  371  373  374  375  381  383  387  389  391  392  394  398  399  402  403
## [127]  406  411  412  418  421  423  426  429  430  432  437  438  439  441  444  453  455

log_likelihood

## [,1]
## [1,]  NaN
```