Project 2

Pokemon

One of my favorite games of all time would have to be is pokemon. At a very basic level it is very engaging to a younger audience who just like the animations and designs of the pokemon. While it also appeals to an older audience since they're tons of data that go in each pokemon and the competitive part of it. In this dataset, includes all pokemon in the game along with some key stats such as their typing,stats, and other data.

MANOVA Testing

With this dataset, I will be performing a MONOVA test in order to determine if there is mean difference between height and weight of a pokemon based on their primary type.

```
pokemon <- read.csv("pokemon.csv")
pokemon$mega<-as.factor(pokemon$mega)
man1<-manova(cbind(height_m,weight_kg)~type1, data=pokemon)
summary(man1)
```

```
##            Df  Pillai approx F num Df den Df    Pr(>F)
## type1      17 0.13727   3.3074     34   1526 6.941e-10 ***
## Residuals 763
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

According to our data, the data was significant therefore, there was mean difference between weight and height based on typing.

```
summary.aov(man1)
```

```
##  Response height_m :
##              Df Sum Sq Mean Sq F value    Pr(>F)
## type1        17  48.12  2.8306  2.5049 0.0007069 ***
## Residuals   763 862.22  1.1300
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
##  Response weight_kg :
##              Df  Sum Sq Mean Sq F value    Pr(>F)
## type1        17  969306   57018   5.205 5.944e-11 ***
## Residuals   763 8358297   10955
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## 20 observations deleted due to missingness
```

```
pokemon%>%group_by(type1)%>%summarize(mean(height_m,na.rm=T),mean(weight_kg,na.rm=T))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 18 x 3
##    type1    `mean(height_m, na.rm = T)` `mean(weight_kg, na.rm = T)`
##    <fct>                          <dbl>                        <dbl>
##  1 bug                            0.883                         33.1
##  2 dark                           1.30                          69.1
##  3 dragon                         1.94                         107.
##  4 electric                       0.982                         37.9
##  5 fairy                          0.794                         23.6
##  6 fighting                       1.20                          58.7
##  7 fire                           1.17                          66.1
##  8 flying                         1.17                          52
##  9 ghost                          1.25                          69.6
## 10 grass                          0.939                         33.3
## 11 ground                         1.34                         150.
## 12 ice                            1.21                         103.
## 13 normal                         1.02                          46.2
## 14 poison                         1.16                          33.8
## 15 psychic                        1.06                          57.3
## 16 rock                           1.30                          92.9
## 17 steel                          1.88                         189.
## 18 water                          1.28                          51.1
```

```
pairwise.t.test(pokemon$height_m,pokemon$type1,p.adj="none")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  pokemon$height_m and pokemon$type1
##
##          bug     dark    dragon  electric fairy   fighting fire    flying
## dark     0.07756 -       -       -        -       -        -       -
## dragon   1.3e-05 0.02455 -       -        -       -        -       -
## electric 0.64499 0.22987 0.00038 -        -       -        -       -
## fairy    0.75110 0.11588 0.00044 0.53858  -       -        -       -
## fighting 0.18642 0.72232 0.00998 0.41733  0.21106 -        -       -
## fire     0.14618 0.60455 0.00254 0.41541  0.20149 0.90982  -       -
## flying   0.65117 0.84039 0.23411 0.77164  0.57463 0.96325  0.99832 -
## ghost    0.12491 0.87510 0.01812 0.31276  0.15775 0.84678  0.74128 0.89528
## grass    0.74966 0.12302 3.0e-05 0.83980  0.60373 0.27277  0.23588 0.71597
## ground   0.05693 0.87652 0.03964 0.17989  0.09166 0.61489  0.49645 0.78795
## ice      0.20168 0.76731 0.01599 0.41893  0.21599 0.96730  0.87927 0.94866
## normal   0.38868 0.22526 8.2e-05 0.83106  0.39737 0.44979  0.43605 0.81981
## poison   0.23142 0.62196 0.00600 0.49215  0.24910 0.89627  0.97401 0.99174
## psychic  0.35210 0.34440 0.00056 0.71826  0.35506 0.59376  0.61969 0.87016
## rock     0.04303 0.97426 0.01666 0.17722  0.08987 0.67743  0.54129 0.82797
## steel    8.3e-05 0.04898 0.83527 0.00132  0.00116 0.02202  0.00756 0.27689
## water    0.01264 0.95234 0.00418 0.13008  0.07018 0.69842  0.52261 0.85121
##          ghost   grass   ground  ice     normal  poison  psychic rock
## dark     -       -       -       -       -       -       -       -
## dragon   -       -       -       -       -       -       -       -
## electric -       -       -       -       -       -       -       -
## fairy    -       -       -       -       -       -       -       -
## fighting -       -       -       -       -       -       -       -
```

```
## fire     -       -       -       -       -       -       -       -
## flying   -       -       -       -       -       -       -       -
## ghost    -       -       -       -       -       -       -       -
## grass    0.18857 -       -       -       -       -       -       -
## ground   0.75875 0.09146 -       -       -       -       -       -
## ice      0.88627 0.28594 0.66168 -       -       -       -       -
## normal   0.32441 0.59388 0.17046 0.45412 -       -       -       -
## poison   0.74472 0.33428 0.52176 0.86876 0.54080 -       -       -
## psychic  0.45523 0.51428 0.27185 0.58552 0.83111 0.69214 -       -
## rock     0.84055 0.07540 0.89176 0.72846 0.15514 0.57071 0.27722 -
## steel    0.03699 0.00018 0.07362 0.03202 0.00045 0.01427 0.00205 0.03725
## water    0.88999 0.02838 0.80086 0.75880 0.07546 0.57196 0.21683 0.91141
##          steel
## dark     -
## dragon   -
## electric -
## fairy    -
## fighting -
## fire     -
## flying   -
## ghost    -
## grass    -
## ground   -
## ice      -
## normal   -
## poison   -
## psychic  -
## rock     -
## steel    -
## water    0.01342
##
## P value adjustment method: none
```

```
pairwise.t.test(pokemon$weight_kg,pokemon$type1,p.adj="none")
```

```
##
##  Pairwise comparisons using t tests with pooled SD
##
## data:  pokemon$weight_kg and pokemon$type1
##
##          bug     dark    dragon  electric fairy   fighting fire    flying
## dark     0.11812 -       -       -        -       -        -       -
## dragon   0.00179 0.17466 -       -        -       -        -       -
## electric 0.81687 0.22777 0.00881 -        -       -        -       -
## fairy    0.72986 0.14745 0.00886 0.63103  -       -        -       -
## fighting 0.27261 0.70716 0.08652 0.42671  0.26706 -        -       -
## fire     0.08705 0.90228 0.10112 0.21176  0.13964 0.76396  -       -
## flying   0.75914 0.78774 0.38706 0.82288  0.66310 0.91642  0.82082 -
## ghost    0.12281 0.98650 0.18777 0.23032  0.14892 0.69965  0.88949 0.78274
## grass    0.99198 0.11643 0.00166 0.82128  0.72343 0.27146  0.08447 0.76097
## ground   9.1e-07 0.00394 0.13231 2.3e-05  7.8e-05 0.00126  0.00082 0.12416
## ice      0.00525 0.24274 0.89649 0.01842  0.01576 0.13050  0.15913 0.42520
## normal   0.41823 0.29854 0.00733 0.68018  0.39888 0.57569  0.27097 0.92413
## poison   0.97382 0.19609 0.00846 0.87216  0.74205 0.36661  0.18231 0.77442
## psychic  0.20344 0.62773 0.04524 0.38577  0.23839 0.95626  0.67248 0.93169
```

```
## rock     0.00357 0.34796 0.58480 0.01987  0.01929 0.18207  0.22375 0.51324
## steel    4.6e-10 3.8e-05 0.00552 4.4e-08  5.1e-07 9.0e-06  2.8e-06 0.03308
## water    0.25393 0.40791 0.01255 0.50333  0.30027 0.73063  0.39767 0.98791
##          ghost   grass   ground  ice     normal  poison  psychic rock
## dark     -       -       -       -       -       -       -       -
## dragon   -       -       -       -       -       -       -       -
## electric -       -       -       -       -       -       -       -
## fairy    -       -       -       -       -       -       -       -
## fighting -       -       -       -       -       -       -       -
## fire     -       -       -       -       -       -       -       -
## flying   -       -       -       -       -       -       -       -
## ghost    -       -       -       -       -       -       -       -
## grass    0.12125 -       -       -       -       -       -       -
## ground   0.00485 7.5e-07 -       -       -       -       -       -
## ice      0.25698 0.00501 0.11561 -       -       -       -       -
## normal   0.30218 0.41541 5.4e-06 0.01846 -       -       -       -
## poison   0.19839 0.97967 3.2e-05 0.01693 0.57122 -       -       -
## psychic  0.62210 0.20044 0.00020 0.08009 0.53196 0.32775 -       -
## rock     0.36780 0.00328 0.02803 0.70532 0.01601 0.01898 0.10365 -
## steel    5.4e-05 3.5e-10 0.18679 0.00520 3.0e-09 8.5e-08 4.5e-07 0.00039
## water    0.40919 0.24887 1.1e-05 0.02946 0.73128 0.42233 0.72101 0.02832
##          steel
## dark     -
## dragon   -
## electric -
## fairy    -
## fighting -
## fire     -
## flying   -
## ghost    -
## grass    -
## ground   -
## ice      -
## normal   -
## poison   -
## psychic  -
## rock     -
## steel    -
## water    6.8e-09
##
## P value adjustment method: none
```

```
.05/(1+2+49+49)
```

```
## [1] 0.0004950495
```

After conducting a 2 ANOVA tests we find that both weight and height were both significantly different in mean differences even with the adjusted p values. After that we used a post-hoc test with an adjusted p value, in order to determine which variables were different based on typing. I found that in the area of height Dragon had the most significant differences with almost each other type. Which makes sense since in the pokemon universe, dragons tended to be the legendary pokemon so therefore, they're usually big and tall. In the area of weight we found that steel and ground both had a lot of significant differences with the rest of the typing. As mentioned before, most legendaries are dragons and therefore the tallest and presummible the heaviest. However, based on this data it sees that the legendaries might be the tallest but are not the heaviest.

Randomization Test

Pokemon is a long standing franchise that is constantly getting new games every year. As more titles come out the developers of the game try to introduce new mechanics to enhance parts of the game. During the release of Pokemon X & Y, the developers introduced this new feature called "Mega evolution." Typically in pokemon games a pokemon has two evolutions, this new feature introduced an additional evolution in battle that would last the length of the battle. As a competetive player myself, this new feature was loved by the fans and brought the competitive scene to another level. A large discrepency about mega evolution was to not every pokemon recieved one. The relationship between the pokemon that have a mega evolution and the ones who do not are unknown.

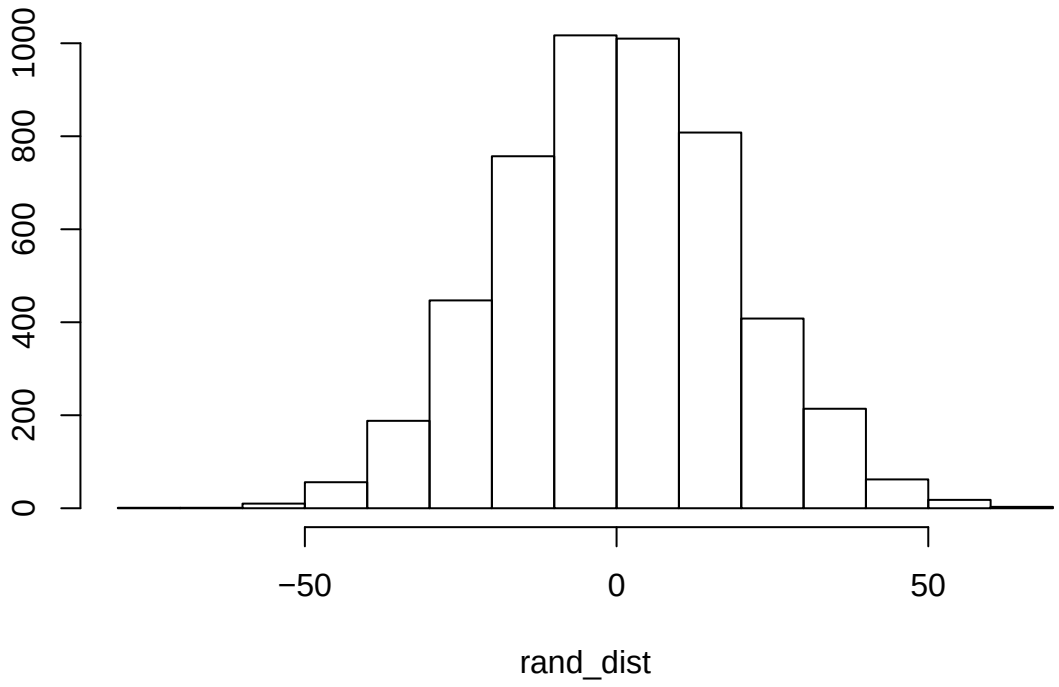
My initial guess is that the pokemon with the highest total base stats got that extra evolution. In order to test this, I will conduct a randomization test to see if there is any relationship between the variables.

```
pokemon%>%group_by(mega)%>%summarize(means=mean(base_total)) %>% summarize(`mean_diff:`=diff(means))
```

```
## `summarise()` ungrouping output (override with `.groups` argument)
```

```
## # A tibble: 1 x 1
##   `mean_diff:`
##          <dbl>
## 1         197.
```

```
rand_dist<-vector()
for(i in 1:5000){
new<-data.frame(time=sample(pokemon$base_total),condition=pokemon$mega)
rand_dist[i]<-mean(new[new$condition=="1",]$time)-
 mean(new[new$condition=="0",]$time)}

{hist(rand_dist,main="",ylab=""); abline(v = -204.7756,col="blue")}
```




```
mean(rand_dist>204.7756)*2
```

```
## [1] 0
```

The Ho is that there is no diffrence in base total between the pokemon that have mega evolutions. The Ha is that there is a difference in base total between the pokemon taht have mega evolutions. The test statistic from this data is 204.7756. When we randomize the base stats the new F statistic is way beyond than the actual F stat. Therefore, concluding that there is in fact a difference in average base stats from the pokemon with mega evolution and the ones without it.

Linear Regression

The strength of any preticular pokemon is divided into five stats. Hp,Attack, Sp Attack, Defence, Sp Defence, and speed. Usually, the pokemon with the highest combined stats are very strong and used in competetive. So in an attempt to see any relationships to the base total I will be conducting a linear regression using weight and height as predictor variables.

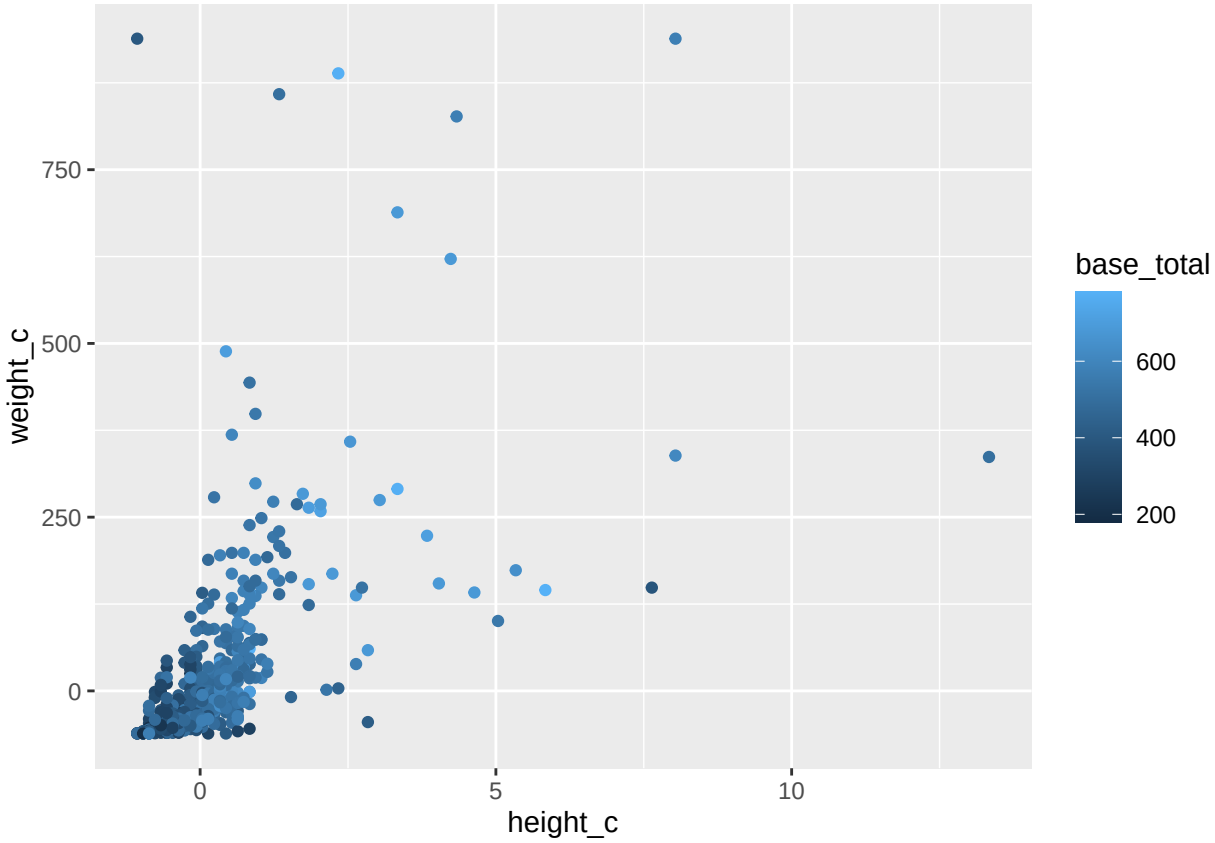
```
pokemon$weight_c<-pokemon$weight_kg-mean(pokemon$weight_kg,na.rm=T)
pokemon$height_c<-pokemon$height_m-mean(pokemon$height_m,na.rm=T)
fit1<-lm(base_total~weight_c*height_c, data=pokemon)
summary(fit1)
```

```
##
## Call:
## lm(formula = base_total ~ weight_c * height_c, data = pokemon)
##
## Residuals:
```

```
##     Min      1Q  Median      3Q     Max
## -488.33  -60.17   -4.13   56.29  317.11
##
## Coefficients:
##                    Estimate Std. Error t value Pr(>|t|)
## (Intercept)       439.12028    3.40915 128.806   <2e-16 ***
## weight_c            0.38554    0.04118   9.362   <2e-16 ***
## height_c           70.15796    4.47439  15.680   <2e-16 ***
## weight_c:height_c  -0.13995    0.01221 -11.466   <2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 91.88 on 777 degrees of freedom
##   (20 observations deleted due to missingness)
## Multiple R-squared:  0.4108, Adjusted R-squared:  0.4086
## F-statistic: 180.6 on 3 and 777 DF,  p-value: < 2.2e-16
```

```
ggplot(pokemon,aes(height_c,weight_c,color=base_total))+geom_point()
```

```
## Warning: Removed 20 rows containing missing values (geom_point).
```




```
bptest(fit1)
```

```
##
##  studentized Breusch-Pagan test
##
## data:  fit1
```

```
## BP = 52.35, df = 3, p-value = 2.522e-11
```

```
coeftest(fit1, vcov = vcovHC(fit1))[,1:2]
```

```
##                       Estimate  Std. Error
## (Intercept)       439.1202779  5.00740750
## weight_c            0.3855366  0.16002030
## height_c           70.1579553 14.08326825
## weight_c:height_c  -0.1399472  0.05622109
```

```
fit2<-lm(base_total~weight_c+height_c, data=pokemon)

anova(fit1,fit2,test="LRT")
```

```
## Analysis of Variance Table
##
## Model 1: base_total ~ weight_c * height_c
## Model 2: base_total ~ weight_c + height_c
##   Res.Df     RSS Df Sum of Sq  Pr(>Chi)
## 1    777 6558828
## 2    778 7668631 -1  -1109803 < 2.2e-16 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

439.120 is the predicted value of base_total when weight and height = 0, 0.385 is the slope for weight on base_total while holding height constant,70.157 is the slope for height on base_total while holding weight constant. After conducting a Heteroskedasticity Robust Standard Error test we find that there was no change from the orginal data. Therefore, the SE do not need to be adjusted and heteroskedasticity is assumed. After creating am model without the interaction and comparing the two we conclude that we can reject the null hypothesis. The linear regression without the interaction is better.

Bootstrapped

```
samp_distn<-replicate(5000, {
 boot_dat<-pokemon[sample(nrow(pokemon),replace=TRUE),]
 fit3<-lm(base_total~weight_c*height_c,data=boot_dat)
 coef(fit3)
})

samp_distn%>%t%>%as.data.frame%>%summarize_all(sd)
```

```
##   (Intercept)  weight_c height_c weight_c:height_c
## 1     4.26838 0.1271068 11.64737        0.04191056
```

There seems to be a big difference of SE from our original model to the bootstrapped model. It seems as if all the variables SE decreased.

Logistic Regression Model

One of the things that Pokemon prides itself in is the ability to create a new world that is seemly similar to reality but contains these creatures. In the natural world, animals that are larger and weight alot tend to be apex organisms that have a lot of power. I want to use linear regression in order to see if those same rules apply to Pokemon.

```r
odds<-function(x)x/(1-x)
logit<-function(x)log(odds(x))

fit4<-glm(mega~base_total+type1, data=pokemon, family=binomial)
summary(fit4)
```

```
##
## Call:
## glm(formula = mega ~ base_total + type1, family = binomial, data = pokemon)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.16035  -0.23084  -0.06928  -0.01926   2.80125
##
## Coefficients:
##                 Estimate Std. Error z value Pr(>|z|)
## (Intercept)   -1.334e+01  1.670e+00  -7.989 1.36e-15 ***
## base_total     2.184e-02  2.911e-03   7.502 6.27e-14 ***
## type1dark     -2.876e-01  9.444e-01  -0.305   0.7607
## type1dragon   -1.600e+00  9.461e-01  -1.691   0.0909 .
## type1electric -7.840e-01  9.694e-01  -0.809   0.4186
## type1fairy    -1.795e+01  2.000e+03  -0.009   0.9928
## type1fighting  3.775e-01  1.025e+00   0.368   0.7128
## type1fire     -1.074e+00  9.020e-01  -1.191   0.2336
## type1flying   -1.785e+01  5.462e+03  -0.003   0.9974
## type1ghost    -3.263e-01  1.082e+00  -0.301   0.7631
## type1grass    -1.281e+00  9.639e-01  -1.329   0.1837
## type1ground   -1.900e+01  1.380e+03  -0.014   0.9890
## type1ice      -1.716e+01  1.961e+03  -0.009   0.9930
## type1normal   -9.699e-01  8.694e-01  -1.116   0.2646
## type1poison   -1.607e+01  1.708e+03  -0.009   0.9925
## type1psychic  -1.858e+00  9.142e-01  -2.033   0.0421 *
## type1rock     -9.239e-01  9.713e-01  -0.951   0.3415
## type1steel    -1.043e+00  9.769e-01  -1.068   0.2856
## type1water    -1.243e+00  8.208e-01  -1.514   0.1301
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 335.17  on 800  degrees of freedom
## Residual deviance: 190.88  on 782  degrees of freedom
## AIC: 228.88
##
## Number of Fisher Scoring iterations: 18
```

```r
exp(coeftest(fit4)) %>% round(3)
```

```
##
## z test of coefficients:
##
##               Estimate Std. Error  z value Pr(>|z|)
## (Intercept)      0.000      5.314    0.000    1.000
## base_total       1.022      1.003 1812.350    1.000
```

```
## type1dark        0.750      2.571    0.737    2.140
## type1dragon      0.202      2.576    0.184    1.095
## type1electric    0.457      2.636    0.445    1.520
## type1fairy       0.000        Inf    0.991    2.699
## type1fighting    1.459      2.788    1.445    2.040
## type1fire        0.341      2.465    0.304    1.263
## type1flying      0.000        Inf    0.997    2.711
## type1ghost       0.722      2.952    0.740    2.145
## type1grass       0.278      2.622    0.265    1.202
## type1ground      0.000        Inf    0.986    2.689
## type1ice         0.000        Inf    0.991    2.699
## type1normal      0.379      2.385    0.328    1.303
## type1poison      0.000        Inf    0.991    2.698
## type1psychic     0.156      2.495    0.131    1.043
## type1rock        0.397      2.641    0.386    1.407
## type1steel       0.352      2.656    0.344    1.331
## type1water       0.289      2.272    0.220    1.139
```

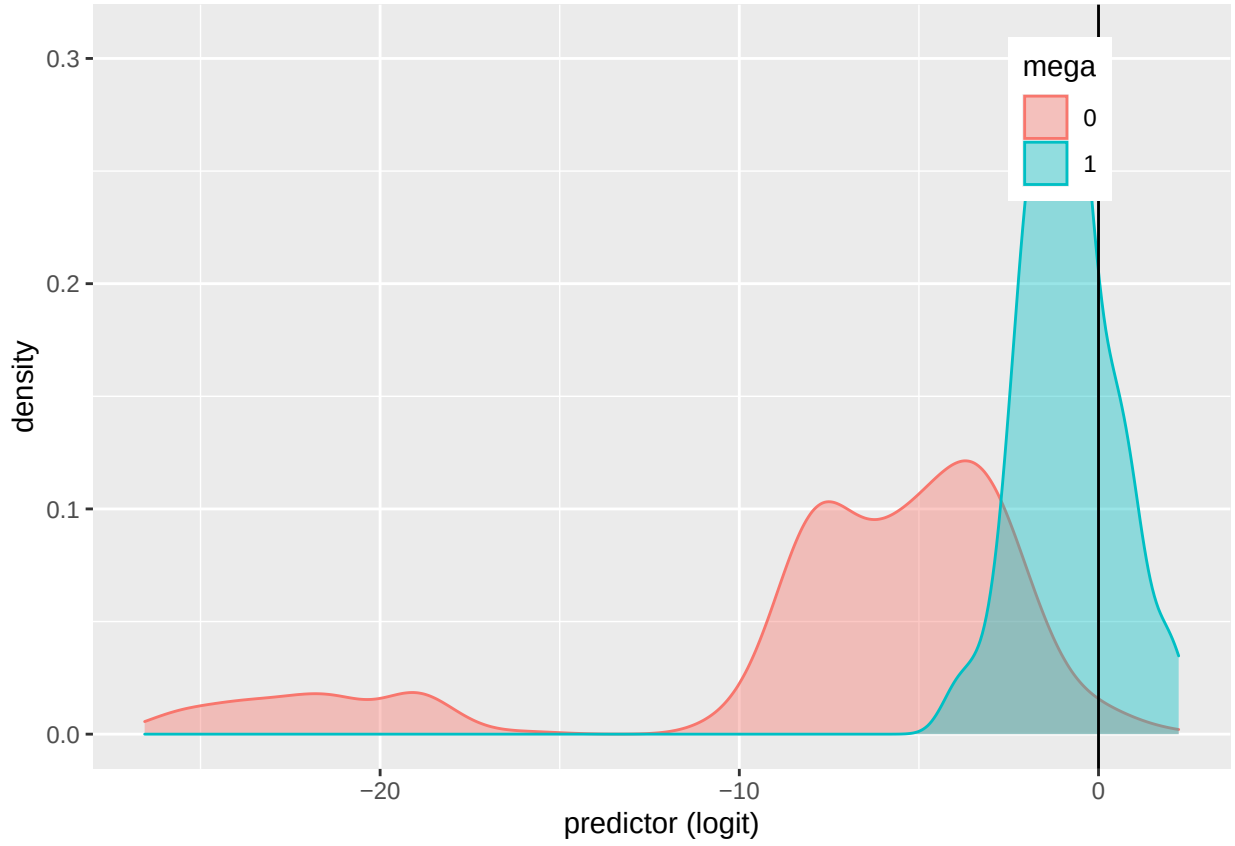
After running the linear regression model, I found that base total and type "psychic" are good variables for predicting mega evolutions. For every 1 unit increase in base total, odds of being a mega increase by a factor of 1.022 and for every 1 unit increase of type psychic, odds of being a mega decrease by a factor of 0.156.

```
pokemon$prob<-predict(fit4,type="response")

table(predict=as.numeric(pokemon$prob>.5),truth=pokemon$mega)%>%addmargins
```

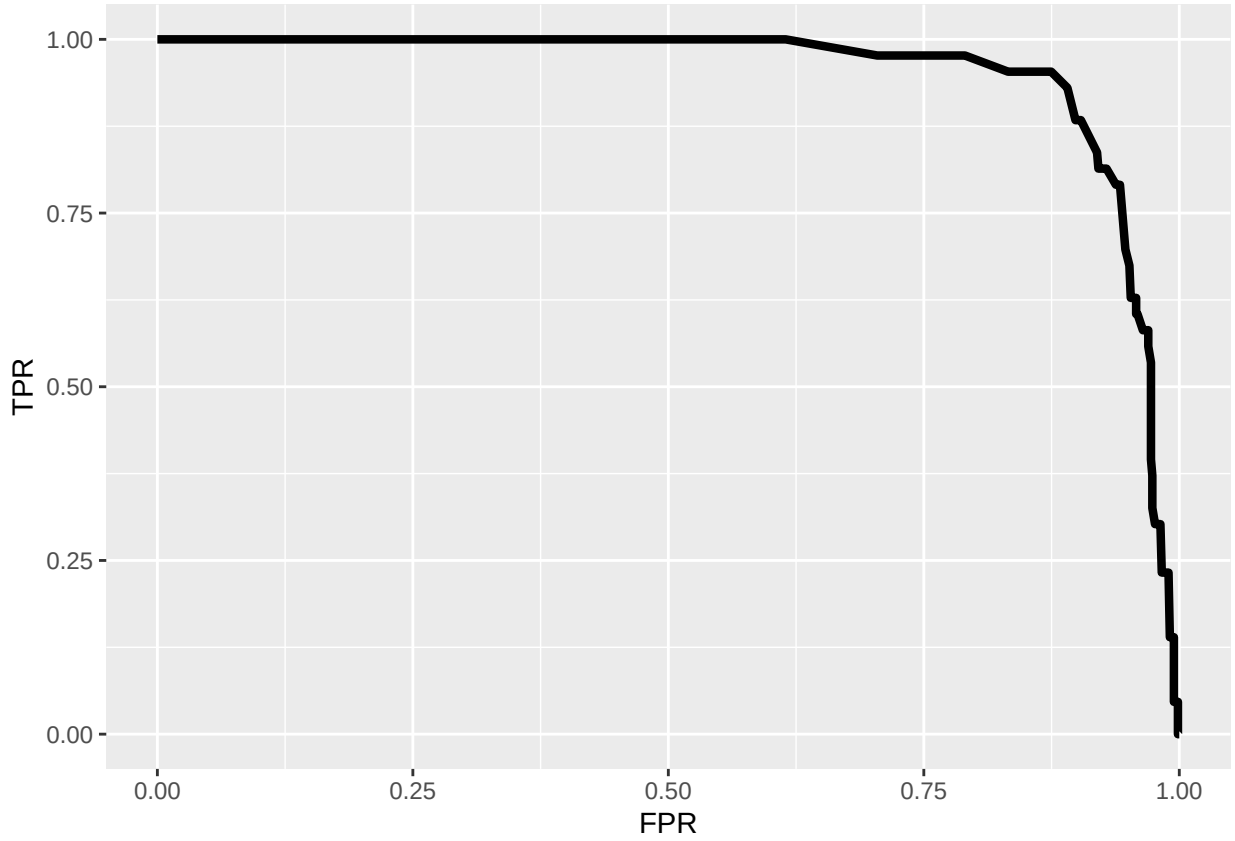
```
##        truth
## predict   0   1 Sum
##     0   747  33 780
##     1    11  10  21
##     Sum 758  43 801
```

For our confusion matrix, I found that sensitivity was 10/43,specificity was 747/758, and precision was 10/21.

```
pokemon$logit<-predict(fit4,type="link")

pokemon%>%ggplot()+geom_density(aes(logit,color=mega,fill=mega), alpha=.4)+
  theme(legend.position=c(.85,.85))+geom_vline(xintercept=0)+xlab("predictor (logit)")
```




```
pokemon -> data
sens<-function(p,data=data, y=mega) mean(data[data$mega==1,]$prob>p)
spec<-function(p,data=data, y=mega) mean(data[data$mega==0,]$prob<p)

TPR<-sapply(seq(0,1,.01),sens,data)
FPR<-sapply(seq(0,1,.01),spec,data)

ROC1<-data.frame(TPR,FPR,cutoff=seq(0,1,.01))

ROC1%>%ggplot(aes(FPR,TPR))+geom_path(size=1.5)
```




```
widths<-diff(ROC1$FPR) #horizontal distances
heights<-(ROC1$TPR[-1]+ROC1$TPR[-length(ROC1$TPR)])/2 #avg heights
AUC<-sum(heights*widths) #sum up areas of trapezoids
AUC
```

```
## [1] 0.9516936
```

```
data -> pokemon
set.seed(1234)
k=10
data1<-pokemon[sample(nrow(pokemon)),] #randomly order rows
folds<-cut(seq(1:nrow(pokemon)),breaks=k,labels=F) #create folds
diags<-NULL
for(i in 1:k){
 ## Create training and test sets
 train<-data1[folds!=i,]
 test<-data1[folds==i,]
 truth<-test$mega
 ## Train model on training set
 fit<-glm(mega~base_total,data=pokemon,family="binomial")
 probs<-predict(fit,newdata = test,type="response")
 ## Test model on test set (save all k results)
 diags<-rbind(diags,class_diag(probs,truth))
}

apply(diags,2,mean)
```

```
##       acc      sens      spec       ppv       auc
## 0.9363272 0.2233333 0.9775986 0.4200000 0.9393252
```

The original AUC from the ROC curve is .9516936. After conducting a k-fold CV, the new values are acc= 0.9363580,sens= 0.2550000, spec= 0.9775398, auc= 0.9398991. As you can tell the AUC in the K-fold had decreased a bit compared to the orginal. Although there was a decrease, both are great models for pokemon getting mega evolutions.

“‘