Project Phase2

Q1.

1-

We perform hypothesis on variable Radiant_win which indicates if Radiant team win the match or lose it. We take 100 random samples out of dataset in order that the effect size become practical significant.

H0: p=0.5

Ha: p>0.5

conditions:

1-Independence: random sample is taken and 100 < 10% of population (which is 4200)

2-Sample size/skew: 100*0.5=50 > 10

```
dota2 = fread("C:\\Users\\Hamed\\Desktop\\DOTA2\\DOTA2.csv")
radinat_win = dota2[!duplicated(dota2[,'match_id']),'radiant_win']$radiant_wi
n
n = 100
radiant_win=sample(radinat_win,size = n)
SE = sqrt(0.5*0.5/n)
p_hat = sum(radiant_win)/n
Z= (p_hat-0.5)/SE
pnorm(Z,lower.tail = FALSE)

## [1] 0.2118554
```

p-value is 0.21 and > 0.05 so we fail to reject H0 which means there's no strong evidence that Radiant teams win more than Dire teams.

2-

Effect size for default case equals to 0.54 - 0.5 = 0.04 and for alternative case is less than 0.04 for example for p = 0.6 equals to 0.54 - 0.6 = -0.06.

As we can see, effect sizes are not practically significant but they're statistically significant due to small value of standard errors and the fact that hypothesis test is done on proportion and proportions are limited to interval [0,1]

Q2.

1-

we consider 'win' and 'class' variable and test if there's dependency between heroes' classes and the number of matches they win. we take 300 random samples out of dataset and split it into two group of ‘AGI’ hero class and others. Then we compute the proportion of win and lose within each group.

```
df <- dota2[sample(nrow(dota2), 300),c('Win','Class')]
df_agi <- df[df$Class=='AGI',]
df_others <- df[df$Class!='AGI',]
n1 = nrow(df_agi)
n2 = nrow(df_others)
p1 = sum(df_agi$Win)/n1
p2 = sum(df_others$Win)/n2
```

Conditions:

1-Independence:

within groups: random sample is taken and 300 (also the number of samples in each group) < 10% of population (equal to 4200)

between groups: No reason to expect sampled class heroes to be dependent.

2-Sample size/skew: ‘AGI’ class 91*0.48 > 10 & 91*0.52 > 10

‘others’ class :209*0.507 > 10 & 209*0.493 > 10

```
SE = sqrt((p1*(1-p1)/n1)+(p2*(1-p2)/n2))
lb = (p1-p2) + qnorm(0.025)*SE
ub = (p1-p2) - qnorm(0.025)*SE
```

95% CI for difference of proportion = (-0.14 , 0.099)

2-

H0: hero class and match win are independent

Ha: hero class and match win are dependent

First we compute contingency table:

```
t <- table(df$Win,df$Class)
N=sum(t)
t

##
##      AGI INT STR
##   0   58  62  42
##   1   44  47  47
```

conditions:

independence: samples are taken randomly - 300 is less than 10% of population - each case only contributes to one cell in the table

Sample size: each cell has at least 5 expected cases.

```
t2 <- t
for (i in 1:2)
{
  for (j in 1:3)
  {
    t2[i,j] = round(sum(t[i,])*sum(t[,j])/N)
  }
}
t3 <-((t-t2)^2)/t2
pchisq(sum(t3),2,lower.tail = FALSE)

## [1] 0.3141582
```

p-value is 0.31 > 0.05 so we fail to reject H0 which means there’s no strong evidence that hero class and match wins are dependent.

Q3.

we test if Radiant team wins more than 50% of games or not. So we have:

H0: p = 0.5

Ha: p > 0.5

we do this by flipping a simulated fair coin 15 times and count how many heads come. we repeat the experiment for 1000 times and count how many samples are greater than estimated proportion.

```
radinat_win <- dota2[!duplicated(dota2[,'match_id']),'radiant_win']$radiant_w
in
radiant_win <- sample(radiant_win,size = 15)
p_hat = sum(radiant_win)/15
outcomes = c('Head','tail')
cnt = 0
for (i in 1:1000)
{
  d <- sample(outcomes,15,replace = TRUE)
  if (sum(d=='Head')/15 > p_hat)
  {
    cnt = cnt + 1
  }
}
cnt / 1000

## [1] 0.153
```

p-value is 0.153 > 0.05 so we fail to reject H0 which means there’s no strong evidence that Radiant team wins more than 50% of matches.

Q4.

awe choose hero class variable, and for bias selection we take random samples from hero which kills more than 10 in a match. first we test unbiased sample.

```
N = nrow(dota2)
ot = round((table(dota2$Class)*500)/N)
udf <- dota2[sample(nrow(dota2), 500),'Class']$Class
ut = table(udf)
ut
## AGI INT STR
## 162 168 170

ot

##
## AGI INT STR
## 154 182 164
```

Conditions:

independence:

samples are taken randomly - 500 is less than 10% of population - each case only contributes to one cell in the table.

sample size: Each cell has at least 5 expected cases.

```
t3 <-((ot-ut)^2)/ot
pchisq(sum(t3),2,lower.tail = FALSE)

## [1] 0.4248539
```

p-value is 0.42 > 0.05 so we fail to reject H0 which means there’s no strong evidence that hero class unbiased sampling distribution is different from original distribution.

now we test biased sample

```
b_df <- dota2[dota2$kills > 10,'Class']$Class
b_df<-sample(b_df, 500)
tbf = table(b_df)
tbf

## b_df
## AGI INT STR
## 226 135 139
```

conditions are met like before.

```
t3 <-((ot-tbf)^2)/ot
pchisq(sum(t3),2,lower.tail = FALSE)

## [1] 1.687247e-11
```

p-value is less than 0.05 so we reject H0 which means there's strong evidence that hero class sampling distribution which killed more than 10 in a match, is different from original distribution.

b-

we select team variable which indicates which team hero was in.

H0: heroes class and team are independent

Ha: heroes class and team are dependent

First we compute contingency table:

```r
df1 <- dota2[sample(nrow(dota2), 500),]
t <- table(df1$team,df1$Class)
N=sum(t)
t

##
##      AGI INT STR
##    D  81  90  76
##    R  75  92  86
```

conditions:

independence: samples are taken randomly - 500 is less than 10% of population - each case only contributes to one cell in the table

sample size: each cell has at least 5 expected cases.

```r
t2 <- t
for (i in 1:2)
{
  for (j in 1:3)
  {
    t2[i,j] = round(sum(t[i,])*sum(t[,j])/N)
  }
}
t3 <-((t-t2)^2)/t2
pchisq(sum(t3),2,lower.tail = FALSE)

## [1] 0.6684973
```

p-value is 0.66 > 0.05 so we fail to reject H0 which means there's no strong evidence that hero class and their team are dependent.

Q5.

The question is "Is there a difference between times players kill when they win a match and when they lose it?" hypothesis test is like below (we take 500 random samples):

H0: $\mu_{win} - \mu_{lose} = 0$

Ha: μ_{win} - μ_{lose} ǂ 0

conditions:

1-Independence:

within groups: random sample is taken and 500 (also the number of samples in each group) < 10% of population (which is 4200)

between groups: no reason to expect that players' kills in two groups are dependent.

2-Sample size/skew: sample size (500) is more than 30.

```
df <- dota2[sample(nrow(dota2), 500),c('Win','kills')]
wd <- df[df$Win==1,]$kills
ld <- df[df$Win==0,]$kills
SE = sqrt(var(wd)/length(wd) + var(ld)/length(ld))
t = (mean(wd)-mean(ld))/SE
degree = min(c(length(wd)-1,length(ld)-1))
2*pt(t,df=degree,lower.tail = FALSE)

## [1] 2.566857e-08
```

p-value is nearly zero and < 0.05 so we reject H0 which means there is a strong evidence that there exists a difference between times players kill when they win a match and when they lose it.

Q6.

we choose gold spent by a player as a response variable and match duration, kills, level, win, team as explanatory variables.

a-

I guess win status, level, kills and duration are significant predictors of response variable so I mention my reasons for each one in the below.

duration: as match lasts longer players spend more gold.

level: As the level of a player gets higher it earns more gold.

kills: when players kill a hero they earn specific amount of gold depending on the situation.

win status: winners probably have spent more gold than losers.

b-

backward elimination using adjusted R^2:

Full:

```
sampled_dota <- dota2[sample(nrow(dota2), 500),]
L = lm(gold_spent~duration+kills+Win+team+level,data = sampled_dota)
```

```
s = summary(L)
s$adj.r.squared

## [1] 0.7835877
```

Step1:

```
L = lm(gold_spent~duration+kills+Win+team,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.6828358

L = lm(gold_spent~duration+kills+Win+level,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7838889

L = lm(gold_spent~duration+kills+team+level,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7703324

L = lm(gold_spent~duration+Win+team+level,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7615328

L = lm(gold_spent~kills+Win+team+level,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7817888
```

we remove ‘team’ variable from set of variables.

step2:

```
L = lm(gold_spent~duration+kills+Win,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.6797202

L = lm(gold_spent~duration+kills+level,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7707236
```

```r
L = lm(gold_spent~duration+Win+level,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7619923

L = lm(gold_spent~kills+Win+level,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7839111
```

we remove ‘duration’ from set of variables.

step3:

```r
L = lm(gold_spent~kills+Win,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.5071386

L = lm(gold_spent~kills+level,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7709621

L = lm(gold_spent~Win+level,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7624233
```

adjusted R^2 wasn’t increased so we select ‘win’, ‘level’, ‘kills’.

forward selection:

step1:

```r
L = lm(gold_spent~duration,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.2920707

L = lm(gold_spent~kills,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.487099
```

```
L = lm(gold_spent~level,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7431572

L = lm(gold_spent~Win,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.1314128

L = lm(gold_spent~team,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] -0.001565527
```

so we add 'level' to the set of variables

step2:

```
L = lm(gold_spent~level+duration,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7453719

L = lm(gold_spent~level+kills,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7709621

L = lm(gold_spent~level+Win,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7624233

L = lm(gold_spent~level+team,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7426487
```

so we add 'kills' to the set of variables

step3:

```
L = lm(gold_spent~level+kills+duration,data = sampled_dota)
s = summary(L)
s$adj.r.squared
```

```
## [1] 0.7707236

L = lm(gold_spent~level+kills+Win,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7822111

L = lm(gold_spent~level+kills+team,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7705365
```

so we add ‘win’ to the set of variables

step4:

```
L = lm(gold_spent~level+kills+Win+duration,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7820889

L = lm(gold_spent~level+kills+Win+team,data = sampled_dota)
s = summary(L)
s$adj.r.squared

## [1] 0.7817888
```

adjusted R^2 is not increased so variables are level+kills+Win.

my impression was almost true except that ‘duration’ variable has a little negative effect on predicting gold spent by a player which was unexpected.

Q7.

a-

I think the best model is: goldspent ~ level + kills + deaths + assists + last_hits + hero_damage + tower_damage+win

```
L = lm(gold_spent~level + kills + deaths + assists + last_hits + hero_damage
+ tower_damage+ Win,data = sampled_dota)
s = summary(L)
s

##
## Call:
## lm(formula = gold_spent ~ level + kills + deaths + assists +
##     last_hits + hero_damage + tower_damage + Win, data = sampled_dota)
##
## Residuals:
```

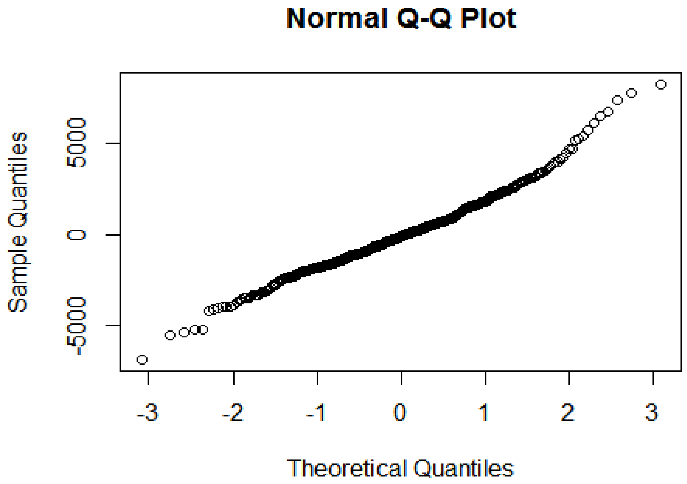
```
##     Min      1Q  Median      3Q     Max 
## -6842.0 -1324.1  -103.5  1107.4  8247.7 
## 
## Coefficients:
##                Estimate Std. Error t value Pr(>|t|)    
## (Intercept)  -1.159e+03  5.615e+02  -2.065   0.0395 *  
## level         5.085e+02  4.752e+01  10.702  < 2e-16 ***
## kills         2.282e+02  3.977e+01   5.738 1.68e-08 ***
## deaths       -3.871e+01  3.032e+01  -1.277   0.2023    
## assists       3.303e+01  2.105e+01   1.569   0.1173    
## last_hits     2.634e+01  1.828e+00  14.410  < 2e-16 ***
## hero_damage  -9.040e-03  3.345e-02  -0.270   0.7871    
## tower_damage  1.177e-01  7.616e-02   1.546   0.1229    
## Win           1.372e+03  2.622e+02   5.233 2.47e-07 ***
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 2047 on 491 degrees of freedom
## Multiple R-squared:  0.8773, Adjusted R-squared:  0.8753 
## F-statistic: 438.8 on 8 and 491 DF,  p-value: < 2.2e-16
```

b-

As I am familiar to dota2 game and the dataset, the factors which can help prediciting gold spent by players, are highly related to the individual performance of players. These factors are player kills, deaths (negative impact), assists, last hits and damages. The main reason behind this, is that by increasing each of above variables, players will be rewarded certian amount of gold (except for death which gold will be reduced). Also 'win' status can impact response variable as it shows team performance.

c-

```
qqnorm(L$residuals)
```

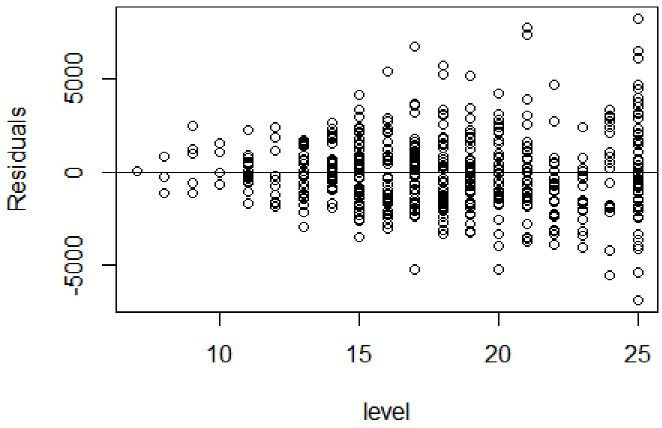


As points have nearly formed a line,residuals follow a normal distribution.

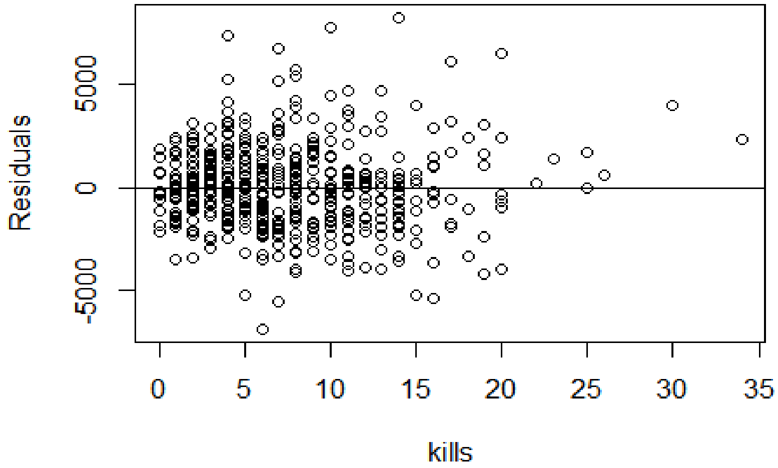
d-

linearity:

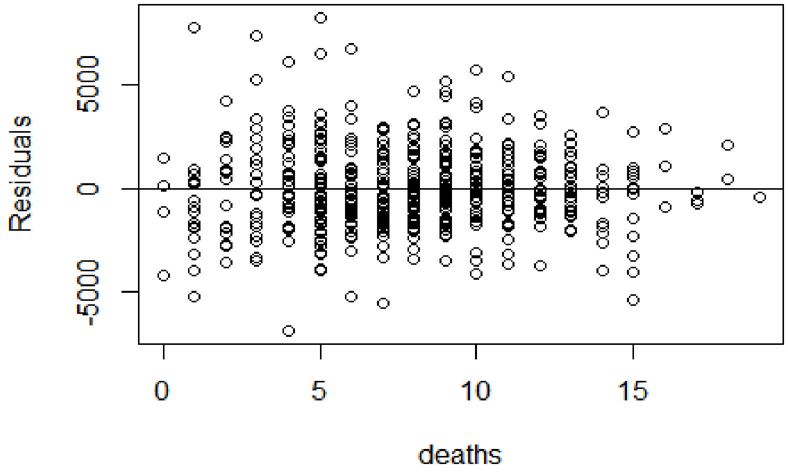
```
plot(sampled_dota$level,L$residuals, ylab="Residuals", xlab="level")
abline(0, 0)
```



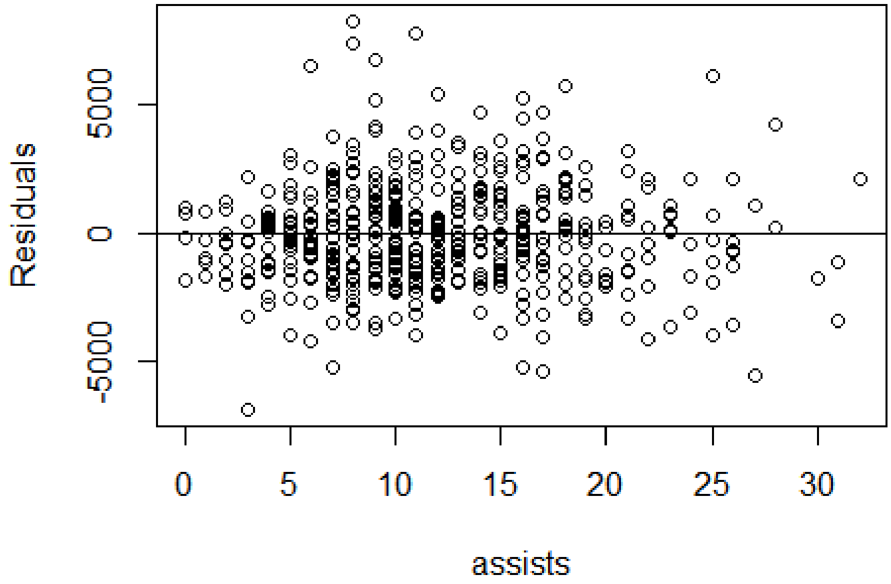
```
plot(sampled_dota$kills,L$residuals, ylab="Residuals", xlab="kills")
abline(0, 0)
```



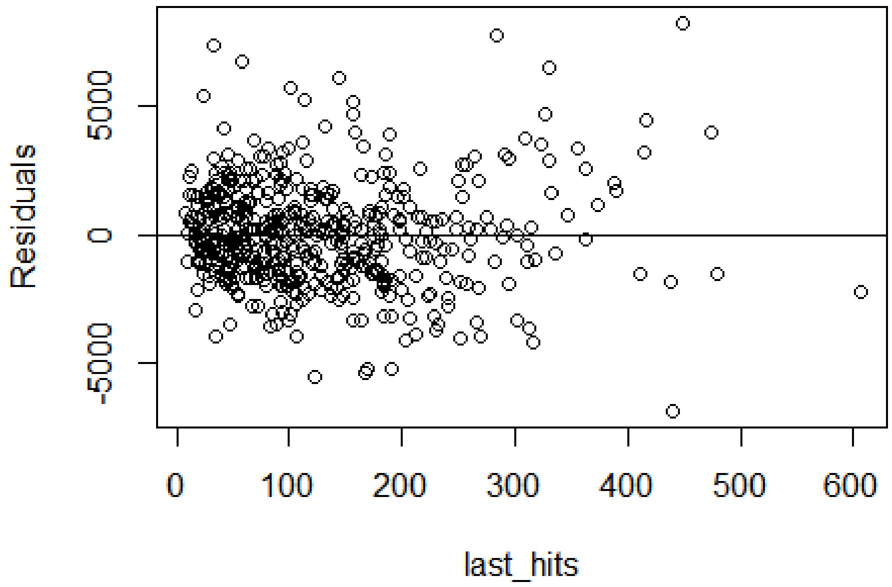
```
plot(sampled_dota$deaths,L$residuals, ylab="Residuals", xlab="deaths")
abline(0, 0)
```



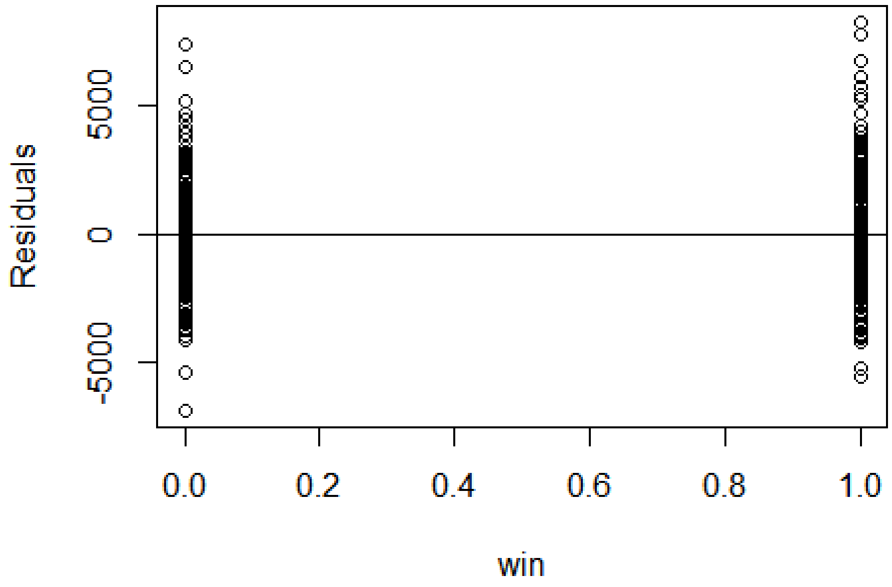
```
plot(sampled_dota$assists,L$residuals, ylab="Residuals", xlab="assists")
abline(0, 0)
```



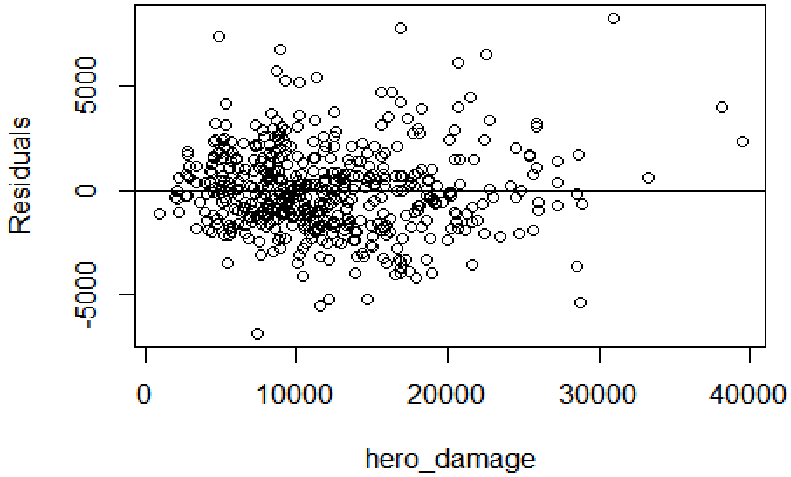
```
plot(sampled_dota$last_hits,L$residuals, ylab="Residuals", xlab="last_hits")
abline(0, 0)
```



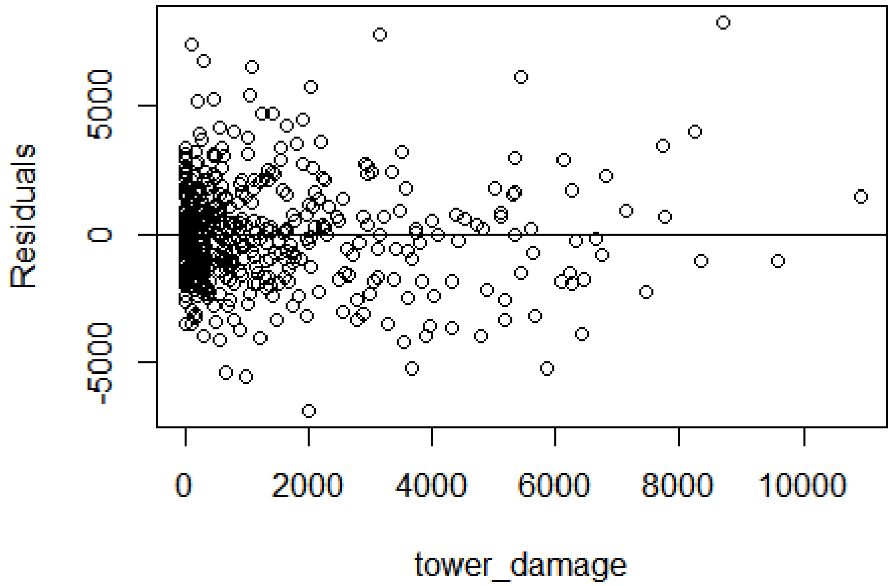
```
plot(sampled_dota$Win,L$residuals, ylab="Residuals", xlab="win")
abline(0, 0)
```



```
plot(sampled_dota$hero_damage,L$residuals, ylab="Residuals", xlab="hero_damag
e")
abline(0, 0)
```



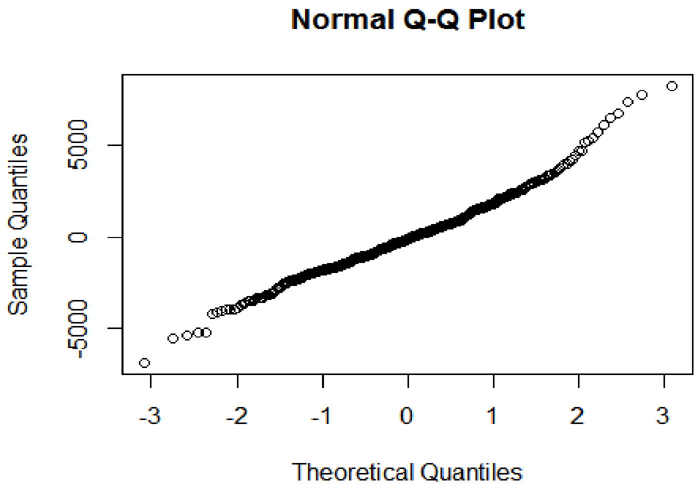
```
plot(sampled_dota$tower_damage,L$residuals, ylab="Residuals", xlab="tower_dam
age")
abline(0, 0)
```



As it's shown in above figures, scatter plots are random around the 0.

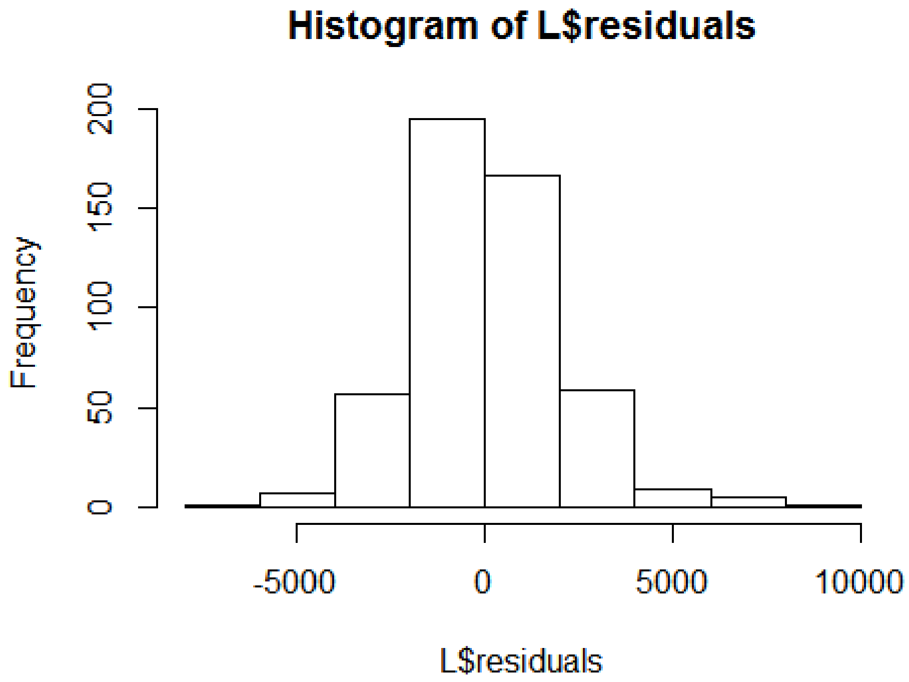
nearly normal residuals:

```
qqnorm(L$residuals)
```



As explained before points nearly form a line.

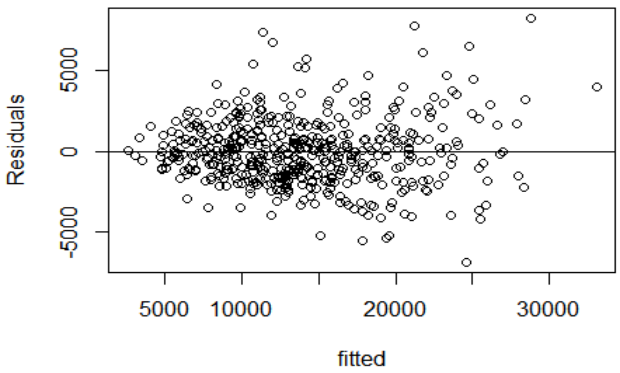
```
hist(L$residuals)
```



residual distribution is nearly normal.

3.constant variability

```r
plot(L$fitted.values,L$residuals, ylab="Residuals", xlab="fitted")
abline(0, 0)
```



Residuals are randomly scattered in a band with a nearly constant width around 0.

e-

intercept: if we assume that a player lost a match and all other variables are assumed to be zero then gold spent by a player would be -1159 (player may lose gold)

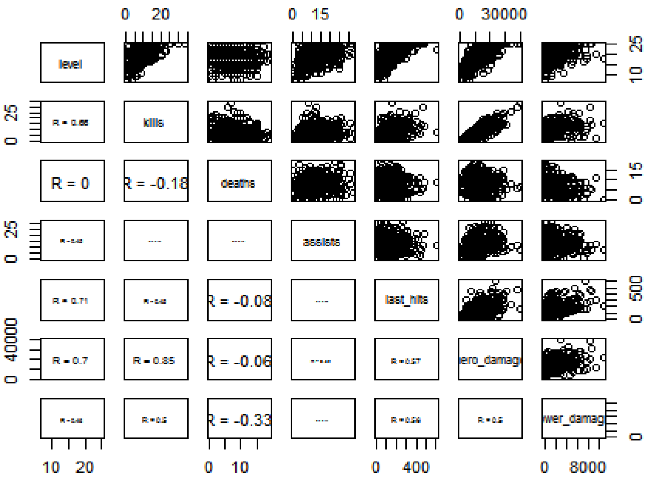
coefficient: for nemerical values interpretation is the same (assuming the coefficint is x):

all other variables held constant, if respective explanatory variable is increased by a unit, the gold_Spent will be increased by x (in case deaths will be deacreased)

for ‘win’ variable: all other variables held constant if a player win status changes form lose to win, gold spent by him/her is increased by 1372.06

f-

```r
panel.cor <- function(x, y){
    usr <- par("usr"); on.exit(par(usr))
    par(usr = c(0, 1, 0, 1))
    r <- round(cor(x, y), digits=2)
    txt <- paste0("R = ", r)
    cex.cor <- 0.8/strwidth(txt)
    text(0.5, 0.5, txt, cex = cex.cor * r)
}
# Create the plots
pairs(sampled_dota[,c('level','kills','deaths','assists','last_hits','hero_da
mage','tower_damage')],
      lower.panel = panel.cor)
```



It can be realized from above figure that hero damage and kills, hero damage and level, last hits and level have strong correlation.

g-

‘level’ varaible plays significant role in predicing gold spent.

i)

H0: b1=0

H1: b1≠0

```
SE = 47.52
t = (508.5 - 0)/SE
pt(t,df=500-2,lower.tail = FALSE)

## [1] 2e-16
```

p-value is nearly zero & < 0.05 so we reject H0 which means there’s strong evidence that ‘level’ is a significant predictor of response variable.

ii)

```
lb = 508.5  + qt(0.05,df=500-2)*SE
ub = 508.5  - qt(0.05,df=500-2)*SE
```

90% CI for b1 = (430.1909 , 586.8091)

```
N = 500
smp_size = floor(0.2*N)
val_ind <- sample(N, size = smp_size)
Ind = c(val_ind,sample(seq_len(N)[-val_ind] , size = 400))
mse = 0
for (i in 1:5)
```

```
{
  x <- seq(100*(i-1)+1:100*i)
  val_ind = Ind[x]
  train_ind = Ind[-x]
  train <- sampled_dota[train_ind,]
  L = lm(gold_spent~level + kills + deaths + assists + last_hits + hero_damag
e +tower_damage+ Win,data = train)
  val <- sampled_dota[val_ind]
  pr= predict.lm(L ,val,type="response")
  mse = mse + sum((val$gold_spent - pr)^2)
}
sqrt(mse/N)

## [1] 2294.56
```

this value means that in average each prediction made by this linear model will have error of 2294 unit.

i-

Adjusted R squared is equal to 0.88% so 88% of response variable is explained by the model.

j-

considering the fact that model is restricted to linear model, I think response variable has been predicted well, as it’s been shown by p-values, adjuste R^2 and f-statistics.

k-

No. in previous question I chose 5 rational variables for predicting spent gold but here here I select all variables that can have positive impact on predicting response variable so as it can be infered by adjusted R^2 this model is a better predictor than previous one.

Q8.

a-

we select win status variable as a response variable. So in order to predict win variable we change dataset to indicate team information rather than heroes’ information so simply we aggregate heroes’ information:

```
team_dota<-aggregate(cbind(kills, level,deaths,denies) ~ match_id+team+Win, d
ata = dota2[,c('match_id','team','level','kills','deaths','denies','Win')], s
um)
sampled_team = team_dota[sample(nrow(team_dota),size = 500),]
gl = glm(Win~team+level+kills+deaths+denies,data=sampled_team,family = binomi
al)
summary(gl)

##
## Call:
```

```
## glm(formula = Win ~ team + level + kills + deaths + denies, family = binom
ial,
##     data = sampled_team)
##
## Deviance Residuals:
##      Min        1Q    Median        3Q       Max
## -2.56603  -0.25202  -0.03105   0.31641   2.97046
##
## Coefficients:
##               Estimate Std. Error z value Pr(>|z|)
## (Intercept) -2.5218109  1.2819082  -1.967   0.0492 **
## teamR        0.3806669  0.3318851   1.147   0.2514
## level        0.0617185  0.0214129   2.882  0.00395 *
## kills        0.1533130  0.0246683   6.215 5.13e-10 ***
## deaths      -0.2119728  0.0210683 -10.061  < 2e-16 ***
## denies       0.0002021  0.0129208   0.016   0.9875
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## (Dispersion parameter for binomial family taken to be 1)
##
##     Null deviance: 693.15  on 499  degrees of freedom
## Residual deviance: 220.48  on 494  degrees of freedom
## AIC: 258.21
##
## Number of Fisher Scoring iterations: 6
```

b-

coefficient: for numerical values interpretation is the same (assuming coefficient is x): all other variables held constant, if respective explanatory variable is increased by unit, log of odds ratio will be x.

interpretation for 'team' variable: all other variables held constant if team changes from Radinat to Dire log of odds ratio will be 0.38.

assuming team is Dire and all other variables are set to zero then log of odds will be -2.52 which is a interpretaion of intercept.

c-

exponential of coefficients are odds ratios so we have:

```
SE = 0.0202902
b = 0.0502892
lb = b + qnorm(0.01)*SE
ub = b - qnorm(0.01)*SE
print(sprintf("98%% log of odds ratio CI for 'level' variable = (%f , %f)",ex
p(lb),exp(ub)))
```

```
## [1] "98% log of odds ratio CI for 'level' variable = (1.003092 , 1.102402)
"
```

```
SE = 0.0246683
b = 0.1533130
```

```
lb = b + qnorm(0.01)*SE
ub = b - qnorm(0.01)*SE
print(sprintf("98%% log of odds ratio CI for 'kills' variable = (%f , %f)",ex
p(lb),exp(ub)))
```

```
## [1] "98% log of odds ratio CI for 'kills' variable = (1.100678 , 1.234542)
"
```

```
SE = 0.0210683
b = -0.2119728
lb = b + qnorm(0.01)*SE
ub = b - qnorm(0.01)*SE
print(sprintf("98%% log of odds ratio CI for 'deaths' variable = (%f , %f)",e
xp(lb),exp(ub)))
```

```
## [1] "98% log of odds ratio CI for 'deaths' variable = (0.770292 , 0.849625
)"
```

```
SE = 0.0129208
b = 0.0002021
lb = b + qnorm(0.01)*SE
ub = b - qnorm(0.01)*SE
print(sprintf("98%% log of odds ratio CI for 'denies' variable = (%f , %f)",e
xp(lb),exp(ub)))
```

```
## [1] "98% log of odds ratio CI for 'denies' variable = (0.970585 , 1.030723
)"
```

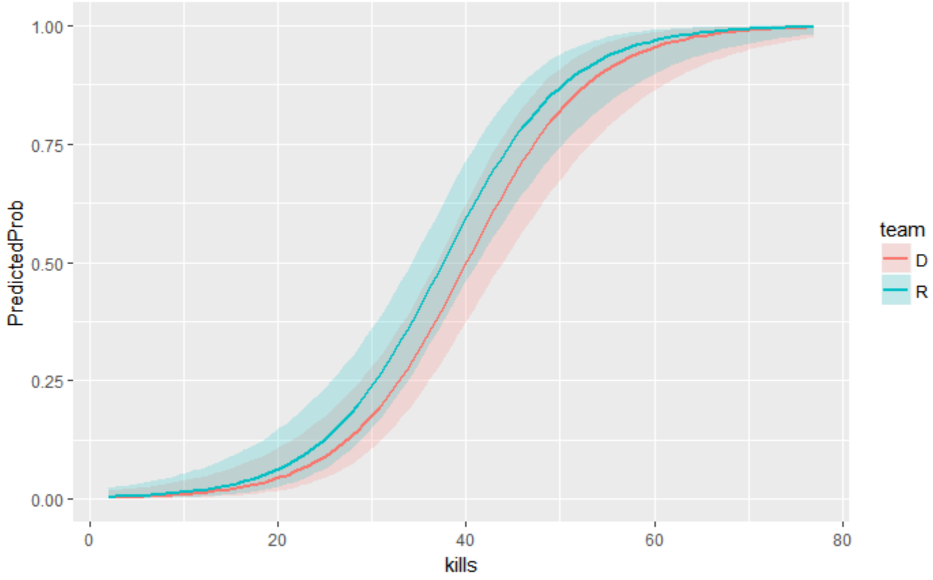
```
SE = 0.3318851
b = 0.3806669
lb = b + qnorm(0.01)*SE
ub = b - qnorm(0.01)*SE
print(sprintf("98%% log of odds ratio CI for 'team' variable = (%f , %f)",exp
(lb),exp(ub)))
```

```
## [1] "98% log of odds ratio CI for 'team' variable = (0.676101 , 3.166881)"
```

d-

```
library(ggplot2)
M = max(sampled_team$kills)
m = min(sampled_team$kills)
newdata <- with(sampled_team, data.frame(kills = rep(seq(from = m, to = M, le
ngth.out = 100),2), level = mean(level),deaths = mean(deaths),denies = mean(d
enies), team = rep(c('R','D'), each = 100)))
newdata1 <- cbind(newdata, predict(gl, newdata = newdata, type = "link" , se=
TRUE))
newdata1 <- within(newdata1, {
    PredictedProb <- plogis(fit)
    LL <- plogis(fit - (1.96 * se.fit))
    UL <- plogis(fit + (1.96 * se.fit))
})
```

```
ggplot(newdata1, aes(x = kills, y = PredictedProb)) + geom_ribbon(aes(ymin =
LL,
    ymax = UL, fill = team), alpha = 0.2) + geom_line(aes(colour = team),
    size = 1)
```



e)

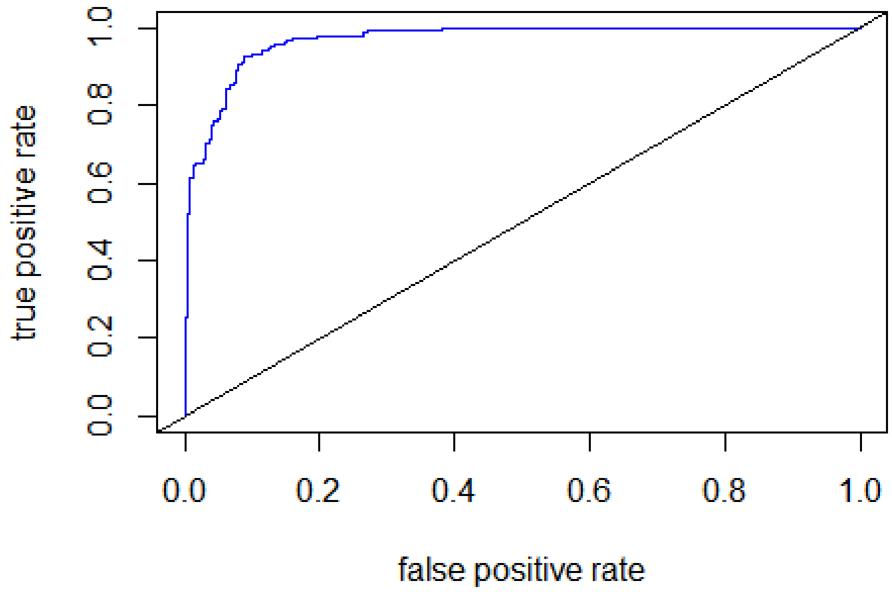
```
fity_ypos <- gl$fitted[sampled_team$Win == 1]
fity_yneg <- gl$fitted[sampled_team$Win == 0]

sort_fity <- sort(gl$fitted.values)

sens <- 0
spec_c <- 0

for (i in length(sort_fity):1){
    sens <- c(sens, mean(fity_ypos >= sort_fity[i]))
    spec_c <- c(spec_c, mean(fity_yneg >= sort_fity[i]))

}
plot(spec_c, sens, xlim = c(0, 1), ylim = c(0, 1), type = "l",
     xlab = "false positive rate", ylab = "true positive rate", col = 'blue')
abline(0, 1, col= "black")
```



```
npoints <- length(sens)
area <- sum(0.5 * (sens[-1] + sens[-npoints]) * (spec_c[-1] -
        spec_c[-npoints]))
area

## [1] 0.969136
```

AUC is 0.969 and it's close to one so the model is a good predictor of response variable. Because as AUC approaches one, the model's ture positive gets higher while it's having low false positive that means it can predict response variable well.

Q9.

I'm going to mention some valuable stories , dota2 dataset telling us in the below:

1) average gold spent by different hero types are the same

2) distribution of hero's class who kill more than 10 is different from orginal distribution of hero's class.

3)heroes who win a match kill more than heros who lose it.

4)gold spent by players can be predicted well by their individual performances like: kills , deaths and assists.

5)hero damages and kills, hero damages and its level, last_hits and level are highly correlated.

6)Radiant teams have been more successful than Dire teams but this difference is not statistically significant.

7)heroes' individual performances are also good predictors of their team's victory probability.

All of the above facts are answers to interesting research questions that has been accomplished in this project.

As the supposed dataset is about game statistics, it can't really solve a serious world problem, but it may help spectators bet on winner teams more easily!!!

Surely further analysis can be done on this dataset because I didn't use some unrelated special variables in this project but more analysis can be done on these variables too. Also there are lots of questions which hasn't been answered yet like: "Are hero types playing important role on winning the match?", "Does players' role effect their victory?" or "What hero type costs less gold?". As I was familiar to this game i felt good analyzing the dataset and for me it was a joyful project.