EXECUTIVE SUMMARY/ABSTRACT:

Since the creation of TCGs (Trading Card Games), players try to use data available to help them win in a complex environment. As you pay a price for your decisions in game, it`s interesting to know if this prices adjusted correctly for what you are paying and what would be the most “efficient deal” in this scenario. Here we made a linear model to access if a minion card’s mana cost in the game Hearthstone can be derived from the cards attributes Attack and Health. The model showed different weights according to the factors taken into account. Conclusions were that attack costs slightly more than health when you spend mana to summon a minion and that there are clear imbalances in certain minion costs.

INTRODUCTION:

In 2014, Hearthstone was released. It is a CCG (collectible card game) where two players starts with 30 life points. There are creatures, spells and weapons. In short, each of them do damage, recover life or make interactions.

Mechanics

The game is a virtual board game where players spend resources (mana) to play cards on the field. These cards have various features, all supposed to be correlated to the mana cost, where stronger cards (in the sense that they “beat” weaker ones) have a higher mana cost.

Picture 1: Minion cards (from https://hearthstone.gamepedia.com/Snowchugger and https://hearthstone.gamepedia.com/Chillwind_Yeti)

In picture 1 we have two minions. Every card of this type is composed of attack (yellow sword), health (red blood), mana cost (blue crystal), mechanics (text) class, race (in this case, a mech), class (blue border) and rarity (the blue jewel). Snowchugger is a more complex card than Chillwind Yeti as it is restricted to a class (i.e. only mage decks can have this card), has a race and a text. The Yeti is the simplest kind of minion in the game as it is almost a conversion of mana into values of attack and health on the board.

From previous trials to linearly model the cost, such an effort was not worth taking into account all these factors. We on this work will restrict to only three factors: cost, attack and health.

Objective:

Explore the relation between a cards mana cost and its attack and health attributes. To do this a linear model will be build which cost being explained by other variables that compose a card.

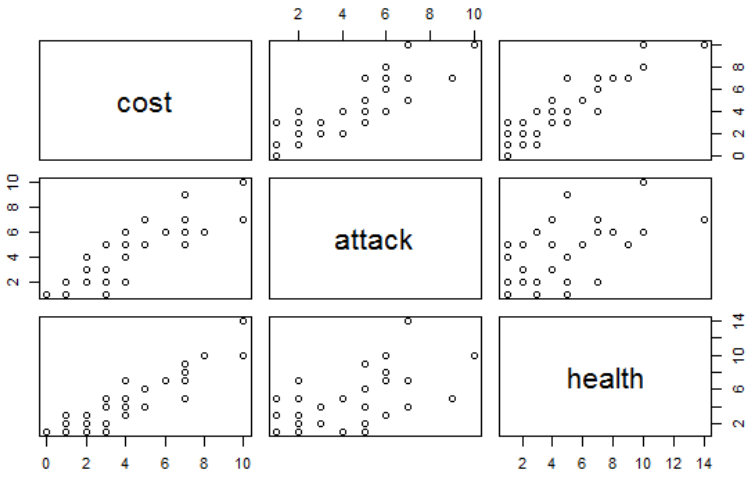
DATA:

The data come from a json file from the website https://hearthstonejson.com/docs/cards.html that contains data on all the cards released until today (6/1/18). This data was chosen for the satisfactory amount of cards it addresses. To treat the data the libraries rjags, rjson, coda and jsonlite were used. The database contains 1614 observations (cards) and 33 variables. Unfortunately, a very small subset of the data will be used here due to the constraints of the work length and for simplicity. The structure of the data used here is described below:

```
'data.frame':     29 obs. of  4 variables:
 $ cost  : int  7 3 5 7 4 3 4 2 1 2 ...
 $ attack: int  6 5 5 5 6 5 2 2 2 3 ...
 $ health: int  8 2 6 9 3 1 7 3 1 2 ...
```

MODEL:

As cards are assumed to be balanced based on its constituent attributes it`s interesting to check if this relation is linear and mana price can be derived from the cards values.



Picture 2: matrix of scatterplots of the factors.

It is almost clear that there is a linear relationship between the variables. Because of this intuition, we choose weakly informative priors. There is not too much data to work with too.

The models and implementations are described below:

Linear non hierarchical

$$y_i \mid p_{mibirici}, \omega_{pmibirici}, \sigma \ \sim^{ind} \mathrm{N}(\omega \ , \sigma)$$

$$\omega \ = \ \mathrm{b}_1 + \mathrm{b}_2 * \mathrm{attack} + \mathrm{b}_3 * \mathrm{health}$$

$$\alpha \sim \mathrm{N}(0,1)$$

$$\sigma \sim \mathrm{G}(1/2, 1)$$

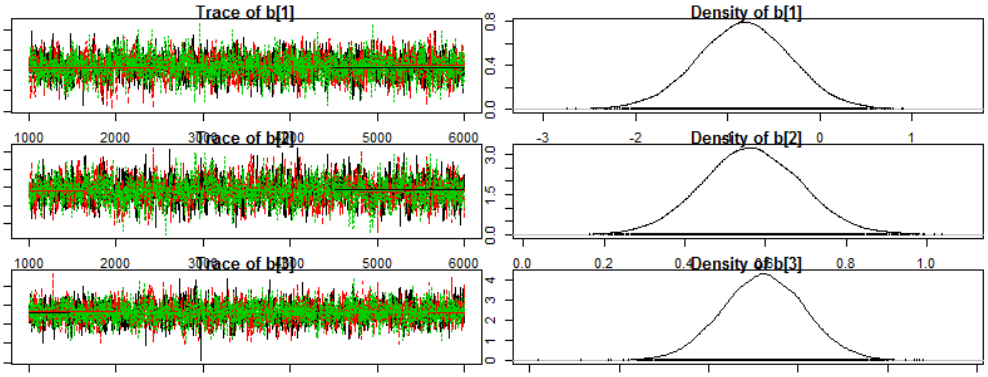
```
#string ----------------------------------
---

mod_string = " model {
  for (i in 1:length(y)) {
    y[i] ~ dnorm(mu[i], prec)
    mu[i] = b[1] + b[2]*attack[i] +
b[3]*health[i]
  }
  for (j in 1:3) {
  b[j] ~ dnorm(0.0, 1.0)
  }

  prec ~ dgamma(1/2.0, 1)

} "

set.seed(666)
data_jags = list(y=hs2$cost,
                 attack=hs2$attack,
                 health=hs2$health,
                 rarity=hs2$rarity)


params = c("b")
```

```
        mod =
jags.model(textConnection(mod_string),
data=data_jags, n.chains=3)



        #Run-----------------------------------


        mod =
jags.model(textConnection(mod_string),
data=data_jags, n.chains=3)

        update(mod, 2e3)

        mod_sim = coda.samples(model=mod,

variable.names=params,
                               n.iter=8000)

        mod_csim = as.mcmc(do.call(rbind,
mod_sim))



        ## convergence diagnostics---------------
-----

        par(mar=c(1,1,1,1))
        plot(mod_sim)

        gelman.diag(mod_sim)
        coda::gelman.plot(mod_sim)

        autocorr.diag(mod_sim)
```

```
        autocorr.plot(mod_sim)

        effectiveSize(mod_sim)


        set.seed(62)
        post3 = mh(n=n, ybar=ybar, n_iter=500,
mu_init=10.0, cand_sd=0.3)
        coda::traceplot(as.mcmc(post3$mu))

        summary(mod_csim)


        #Check resids-------------------------

        attack <- sample(0:10, 29, replace =
TRUE)
        health <- sample(0:10, 29, replace =
TRUE)
        inits <- data.frame(attack, health)

        pm_params = colMeans(mod_csim)
        out <- pm_params[1] +
pm_params[2]*hs2$attack +
pm_params[3]*hs2$health
        diff <- out - hs2$cost

        par(mar=c(2,2,2,2))
        summary(diff)
        plot(diff)
        boxplot(diff)
        hist(diff)
        qqnorm(diff)
```

RESULTS

Diagnostics



Picture 3: Convergence

Autocorrelation

```
         b[1]        b[2]         b[3]
Lag 0   1.00000000 1.00000000 1.000000000
Lag 1   0.76431910 0.83117976 0.776551043
Lag 5   0.21829569 0.33324990 0.237349470
Lag 10 0.02270085 0.07073793 0.008023041
Lag 50 0.01554860 0.03622810 0.028424552
```

Summaries

```
Mean       SD  Naive SE Time-series SE
b[1] -0.8248 0.51090 0.0041715       0.010709
b[2]  0.5663 0.12527 0.0010229       0.003080
b[3]  0.5227 0.09356 0.0007639       0.001974
```

Just for a simple comparison, an usual least squares fit was made.

```
lm(formula = hs2$cost ~ +hs2$attack + hs2$health)

Residuals:
     Min       1Q   Median       3Q      Max
-0.46225 -0.23181  0.04139  0.13357  0.65595

Coefficients:
             Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.83963    0.09760  -8.603 4.41e-09 ***
hs2$attack   0.56847    0.02418  23.510  < 2e-16 ***
hs2$health   0.52304    0.01820  28.745  < 2e-16 ***
```

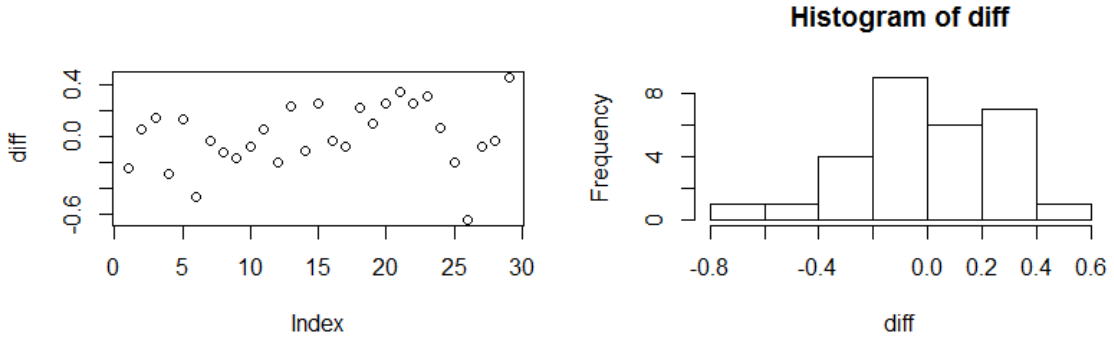
```
---
Signif. codes:  0 ‘***’ 0.001 ‘**’ 0.01 ‘*’ 0.05 ‘.’ 0.1 ‘ ’ 1

Residual standard error: 0.2591 on 26 degrees of freedom
Multiple R-squared:  0.9923,  Adjusted R-squared:  0.9917
F-statistic:  1669 on 2 and 26 DF,  p-value: < 2.2e-16
```

Error checking

To check the errors we modeled the same cards with the paramaters of the model. Code bellow:

```
set.seed(666)

pm_params = colMeans(mod_csim)
out <- pm_params[1] + pm_params[2]*hs2$attack + pm_params[3]*hs2$health
diff <- out - hs2$cost

summary(diff)
par(mfrow=c(1,2))
plot(diff)
hist(diff)
```

```
Min.        1st Qu.     Median       Mean    3rd Qu.       Max.
-0.645731 -0.125368 -0.033815  0.002894  0.228215  0.457097
```



Picture 4: Error checking

CONCLUSIONS

Convergence and correlation were OK. Predictive performance was not bad either.

It is clear the relation of attack and defense, a card should have a little more health than attack to cost for its attack. For instance, the model correctly predicts the mana cost of the minions in picture 1: Chillwind Yeti predicted cost is 4.0539. Interestingly Snowchugger’s cost is 1.8759. This was expected as pertaining to a class (should have negative weight), a tribe (positive) and having a text (positive) explains the difference of 0.1241 mana. This is very low for what it does (it is considered a good card) making it possibly very mana efficient. Correlations on win rates and mana cost could be an idea for future analyses.

Finally, we can conclude that attack and health indeed have an average value to contribute to the mana cost. Still history of Hearthstone decks shows that just mana efficiency shouldn’t tell us too much about the outcome of a game as it`s the direct result of comparison of cards and doesn’t account for correlations. More complex models are suggested to future works.