Importance of Character Statistics in the Mobile Game "Fallout Shelter"

Block, Max Rindstedt, Aron

Summer 2015

Contents

1 Introduction

1.1 Introduction and Game Mechanics

Fallout Shelter is a free-to-play mobile simulation video game developed by Bethesda Game Studios. It was released as a teaser for the fourth installment in the Fallout series scheduled for late 2015.

In Fallout Shelter, players build, extend, and manage their own **vault** as an overseer in a post-apocalyptic world after a nuclear war. The players rescue **dwellers** from the wasteland and assign them to resource generating structures within the vault. Each dweller has a set of seven SPECIAL stats: Strength, Perception, Endurance, Charisma, Intelligence, Agility, and Luck. Each character's SPECIAL affects their skill at generating different forms of resources. The statistics of a dweller can be increased by training.

Dwellers can also level up and be given new armor and weapons, which are useful when sent out into the wasteland to scout for additional (potentially better) gear and bottle caps, which are being used as currency.

Every minute-mark, a scouting dweller come across an enemy. The dweller can either (automatically) run from the enemy or try to fight it. In the first case the dweller can either be successful in running away or have to face the opponent. If fighing the dweller either loses and suffers damage or wins and takes caps and/or gear. When a dweller reaches zero health it dies. A dead dweller can be revived for a fee.

1.2 Aim

The aim of this report is to investigate the roles of the different stats when calculating the "success" of a dweller sent to the wasteland. We are only going to count the number of pieces of gear, and amount of bottle caps even though one could also examine experience points per time or use some other metric. Furthermore, we are going to measure a dwellers *success* in terms of number of items per hour and number of caps per hour, respectively.

We have two points of interest:

1. We want to be able to find the dweller with the highest success rate from a list of dwellers using SPECIAL stats and weapon damage. Remember that success is measured in number of items or caps per hour. Accuracy is not very important since this optimization is more of a qualitative problem; we want to be able to compare two characters and we need the model to deduce if one is significantly better than the other.

2. We want to be able to estimate the survival time of this dweller in to avoid having to pay the revival fee. Here, accuracy is of higher importance.

1.3 Assumptions and Simplifications

- Even though all gear are not created equal and some pieces are strictly better than others, or not comparable (weapons versus armor), we are going to disregard this and just count the number of found pieces.
- There is no way to see how close a character is to leveling up, so we are not going to log this.
- Characters finding strictly better gear[1] than currently equipped will change into the new piece. In order to keep logging feasible we are equipping reasonable gear to begin with and will sometimes assume no gear has been changed during the scouting.
- Even though the SPECIAL stats are discrete from $\mathbb{N}^+$, we will perform regression as though they were elements in $\mathbb{R}^+$.[2]

1.4 Data Collection Method

The data was collected in two datasets. For the first dataset we sent 31 dwellers[3] into the wasteland logging the number of found items as well as the dweller's SPECIAL, current damage, level and time. Each dweller was logged up to 17 times, depending on for how long it lived.

For the second dataset, we sent 117 dwellers into the wasteland. By adding a day to the internal clock of the iPhone the game ran on, we were sure that every dweller had died[4]. The SPECIAL stats, inital and final level, inital and final damage, time until death, and number of bottle caps found was logged.

2 Cleaning and Quantization

In our case we had no missing or nonsense data. Furthermore, all entries came from well-ordered sets. This step is thus trivial.

3 Importance of Features

Tables 1 through 3 contains the results when calling `summary` on the respective linear models. The following encoding for p-values are used:

p-value	**Encoding**
$0 < p \leq .001$	***
$.001 < p \leq .01$	**
$.01 < p \leq .05$	*
$.05 < p \leq .1$	.
$.1 < p \leq 1$	*blank*

[1] Giving a higher boost for each boosted stat.

[2] Here, superscript + means positive.

[3] Each dweller had full health and 25 **stimpaks** (for replenishing health) and 15 **radaways** (for removing radiation damage which is not affected by stimpaks).

[4] The game calculates its state depending on the running machine's internal clock, rather than using some server's time.

Table 1: Number of items per hour

	Estimate Std.	Error	t value	Pr(>|t|)	
(Intercept)	1.290960	0.771921	1.672	0.0959	.
level	-0.021353	0.017178	-1.243	0.2152	
dmg	-0.060882	0.057463	-1.059	0.2906	
s	-0.003986	0.055532	-0.072	0.9428	
p	-0.079242	0.095350	-0.831	0.4069	
e	0.031539	0.070731	0.446	0.6561	
c	-0.057148	0.091382	-0.625	0.5324	
i	0.324587	0.335993	0.966	0.3351	
a	0.214665	0.133610	1.607	0.1096	
l	0.042945	0.073570	0.584	0.5600	

Table 2: Average number of caps per hour

	Estimate	Std. Error	t value	Pr(>|t|)	
(Intercept)	83.81787937	7.20483461	11.63356	<2e-162	***
s	-0.21649106	0.49987023	-0.43309	0.665802	
p	0.61211724	0.67181840	0.91113	0.364235	
e	-0.88781188	0.53114306	-1.67151	0.097489	.
c	-0.18453188	0.59459417	-0.31035	0.756888	
i	0.19836365	0.56461809	0.35132	0.726024	
a	-0.04815488	0.54725246	-0.08799	0.930043	
l	13.48815469	0.50791765	26.55579	<2e-16	***
start.level	-0.68088744	0.16014426	-4.25171	0.00004495	***
start.damage	-1.56703535	0.79511334	-1.97083	0.051278	.
level.increase	-3.44085855	1.74133402	-1.97599	0.050683	.
death.damage	0.30906873	0.34846885	0.88693	0.377067	

Table 3: Survival time

	Estimate Std.	Error	t value	Pr(>|t|)	
(Intercept)	-0.62884167069	0.349672057960	-1.7983755246	0.074911	.
s	0.05346141178	0.022121911003	2.4166724012	0.017339	*
p	0.02862342249	0.029679649482	0.9644124169	0.336994	
e	0.18123434236	0.023695467837	7.6484812878	9e-11	***
c	0.01608604571	0.026402914404	0.6092526555	0.543637	
i	0.05055419848	0.025758219832	1.9626433352	0.052260	.
a	0.02581622298	0.024245608897	1.0647793211	0.289351	
l	-0.26496311935	0.044575915531	-5.9440869849	3e-7	***
start.level	0.11302287177	0.009129632057	12.3797838804	<2e-16	***
start.damage	0.14667002834	0.035506071927	4.1308435539	7e-4	***
level.increase	0.55390718215	0.085695065057	6.4636998849	3e-8	***
caps	0.00310366533	0.000495898471	6.2586709017	8e-8	***
death.damage	0.01143465929	0.015985561359	0.7153117136	0.475960	

3.1 Items per hour

In the case of number of items per hour only the intercept had a somewhat significant value. We deduced that all dwellers have the same chance of finding items.

3.2 Caps per hour

The only three significant predictors were the intercept, Luck, and the starting level.

3.3 Survival time

Significant predictors were `e`, `l`, `start.level`,`start.damage`, `level.increase`, `caps`. However, since we only were interested in predictors known before sending the dweller out we discarded all but Endurance, Luck, and the level and damage when starting.

3.4 Irrelevance of PCA

Since all of the selected features in the respective models were known to be independent, PCA would not have helped.

4 Model Types

5 Model Performances

6 Selected Model