# Crunching the Numbers

## Analyzing data to find statistical patterns indicating unhealthy workplace practices in game development studios

by:

James Connor

November, 2019

Crunch is an issue that has continually plagued the game development crunch. The New York Times defines crunch as a "sudden spike in work hours, as many as 20 a day, that can last for days or weeks on end."[1] This oftentimes mandatory overtime can have disastrous effects on the physical and psychological health of game developers. Game Workers Unite, a movement campaigning for unionization of the game development industry, explains that game developers have suffered "stomach pains, memory loss, extreme anxiety, loss of family time, divorce, severe burnout, and more" at the expense of crunching on a game.

A number of solutions have been proposed for solving this issue, from behavioral changes within studios to legal protections for developers and everything in between. However attempts at exposing and solving these problems are impeded by the leaders and executives of development studios. In an article on Polygon, a website for gaming news, Colin Campbell explained that "[an employee's] boss will likely hurt [a developer's] career if they catch [them] talking to the press about abusive working conditions … Change comes by shining a light on these malpractices." The ideal scenario is to be able to identify studios that crunch without requiring employees to come forward and risk their jobs. The goal of this research was to identify the patterns that indicate the use of crunch in a company, solely based on online sources and mathematical data manipulation..

The data manipulation processes used were based on three key assumptions: one, that studios that put out many games rapidly in proportion to their employee size are more likely to employ crunch; two, that focusing on the development of certain genres, such as action games, indicate a higher chance of workplace malpractice; and three, that negative reviews of a company by employees indicate an unhealthy workplace. Confirming or refuting these assumptions would help in establishing patterns that indicate crunch and, as a result, were at the center of this research.

Using Wikipedia, Giantbomb - a site that collects information about games and game development studios - and the free web-based APIs[1] available through both sites, information regarding studios and how often they publish games was downloaded. The data, when analyzed, indicated possible support for the first assumption. Information downloaded from the Internet Games Database API was used to address the second assumption, regarding the genres of published games. The third assumption was addressed by utilizing data, including employee ratings and reviews, downloaded from Glassdoor, a website that provides information on various companies. To use these data points in making predictions, they had to be manipulated through the use of mathematical functions.

The first set of data contained two important pieces of information: the number of employees at a particular studio and, for each game that the studio published, the year they published it. The first step in making this data usable was to handle the list of years; after all, the input needs to be a single decimal number, not an indefinitely long list of values. Lining up these years on a graph in chronological order, there appeared to be  a curve that was exponential in nature, such as the one seen in Figure 1. This exponential property was observed in the graphs of

[1] API stands for "Application Programming Interface" and refers to a system that allows a program to gather information from another program, typically hosted online.

multiple game studios. As a result, it was decided that utilizing values from the exponential curve of best fit could help reduce this information to a single number.


*Figure 1 - A graph portraying the years that Rockstar Games published their games.*

Any exponential curve is in the form $y = A(r)^{Bx},\ r \neq 1.$ As $r$ increases, the exponential curve becomes steeper. The more games that are published in a specific year by a studio, the less steep the exponential curve becomes, and the graph becomes increasingly horizontal in nature. If more games produced per year indicates an increased possibility of a studio crunching and a less steep curve represents more games published per year, it naturally follows that a lower $r$ value points to an increased chance of the studio utilizing crunch. To find the input for the games published over time, which will be referred to as the *game over time* input for the remainder of this paper, the exponential curve of best fit had to be found for the graph of years that a studio published games

To calculate this best fit curve, the first step was to properly organize the values. To do this, the years were sorted into chronologically ascending order. Then each year was paired with the index that it had in the list. For instance, if the years {1996, 2017, 2003, 2008, 1998} were provided, they were first sorted chronologically into {1996, 1998, 2003, 2008, 2017} and then paired with their index in the list. So the first was paired with the number one, the second was paired with the number two, etc. This generated a series of points to calculate the exponential equation with.

| | | | | | |
|---|---|---|---|---|---|
| **x** | 1 | 2 | 3 | 4 | 5 |
| **y** | 1996 | 1998 | 2003 | 2008 | 2017 |

Before calculating a curve of best fit for the points, a line of best fit had to be constructed. A line of best fit, written in point-slope form, is formatted as: $y = mx + b$. The goal for a best fit function is to reduce the distance between the output of the function and the actual y-value. To evaluate how well the function fits a set of data, the error sum of squares (SSE) is calculated using the following formula.

$$SSE = \sum_{i=1}^{n} [y_i - (mx_i + b)]^2$$

The goal is to reduce the total SSE as much as possible. Using the following formulas, one can calculate the ideal slope (*m*) and y-intercept (*b*) that would result in the lowest SSE.

$$m = \frac{\overline{y} \sum_{i=1}^{n} x_i^2) - \overline{x} (\sum_{i=1}^{n} x_i y_i)}{\sum_{i=1}^{n} x_i^2 - (n * \overline{x}^2)}$$

$$b = \frac{\sum_{i=1}^{n} x_i y_i) - (n * \overline{x} * \overline{y})}{\sum_{i=1}^{n} x_i^2 - (n * \overline{x}^2)}$$

Substituting values into these equations will create a line of best fit for the data provided. So, for example, the data shown in the table above resulted in a slope of 5.2 and a y-intercept of 1988.8. However, this only created a linear relationship and, as seen in Figure 1, the years seemed to represent an exponential relationship. The same formulas can be used to derive an exponential best fit curve. Referring to the previously mentioned form of an exponential curve, $y = A(r)^{Bx},\ r \neq 1,$ the equations for the slope and y-intercept of the line of best fit can be applied to a curve, if it's in the form of a line. This line can be constructed by substituting any value for *r* and taking the log base *r* of the curve - in this case, 10 was used for *r*:

$$y = A * 10^{Bx}$$

$$\log(y) = \log(A) + Bx$$

Solving for the line of best fit through the points whose y-coordinates are the logarithm of the previous y-coordinates produced values of 0.0009955 for *m* and 3.299 for *b*. These values were then translated back into the original exponential equation.

$$\log(y) = \log(A) + Bx$$

$$m = B$$

$$b = \log(A) \qquad A = 10^{b}$$

$$y = 10^{b} * 10^{mx} = 10^{mx+b}$$

The purpose of finding the equation for the exponential curve of best fit was to address the “steepness” of the graph of games published by a development studio. If the curve was steeper, the associated game studio was less likely to have employed crunch. Because $10^{BX}$ is the number that represents a curve’s steepness, the rest of the equation was disregarded. $10^{Bx} = 10^{B^x}$ , so the constant that indicates steepness is $10^{B}$. As this constant increases, so too does the graph’s rate of change. However, using this constant alone as an indication meant that a small team of five people putting out one game every year - a difficult task in most cases - held the same weight as a major AAA[2] studio of two-thousand people doing the same. For this reason, the constant was multiplied by the number of employees at the game studio. For small indie teams, this increased the score by a smaller amount than the larger increase that defined bigger studios. This number, defined as the employee count multiplied by the curve’s rate of change, is called the *game over time score*. As this score approaches 0, it points to a larger and larger probability that a studio uses crunch.

Once the *game over time score* was established, it was time to look into the second assumption: that a studio focusing more on the development of certain genres, such as action games, indicates a higher chance of employing crunch. This assumption was made after it was observed that many of the news reports detailing particular games that employed crunch seemed to consistently be reporting on action games, particularly those made by AAA studios. To investigate whether this assumption held any ground, a list was compiled of the top twenty-five games of each of seven major game genres, as determined by *IGN*, *Gameranx*, and *Rock Paper*

[2] AAA (or “triple-A”) studios refer to larger game development companies that typically have thousands of employees and larger budgets than their independent counterparts.

*Shotgun*. The seven genres utilized were action, adventure, puzzle, RPG[3], simulator, sports, and strategy games. For each game on the lists, the game and the studio that created it was manually researched using online news sources and Glassdoor to determine whether crunch was used during the game's creation. Upon completion of this research phase, the approximate statistical probability of crunch being used based on a game's genre was determined, as seen in Figure 2. These numbers represent the ratio between the number of games/studios in each genre found to utilize crunch over 25, the total number of games analyzed for each genre.

| | A | AB |
|---|---|---|
| 1 | Action | 0.48 |
| 2 | Adventure | 0.20 |
| 3 | Role Playing | 0.40 |
| 4 | Simulation | 0.08 |
| 5 | Strategy | 0.32 |
| 6 | Sports | 0.40 |
| 7 | Puzzle | 0.12 |

*Figure 2 - A list of game genres showing the approximate probability of crunch being used.*

These numbers indicate that action games are the most likely type of game to be crunched on, while simulation games are the least. Using this information, the genres were then sorted into a list, organized from most likely to least likely to be crunched on. This resulted in the following order: action, sports, role playing, strategy, adventure, puzzle, and simulation. However, even if one genre is attributed to each game that a studio published, it still wouldn't result in the single input needed. So the numbers had to be averaged in such a way as to reduce them to a single number while still being representative of the statistical probability and the

[3] RPG is short for "Role-Playing Game", a genre of game in which characters create and control a fictional character in an environment or world with lots of background, story, or lore.

number of published games in each genre. For instance, five action games had to have a different result than 5 simulation games, which had to have a different result than three simulation games. Finally, it would be beneficial if there were an upper and lower bounds for the outputted value, so that a statistical prediction model could find reasonable patterns within the data. The need for varying values indicated that the use of a number line would be best. However, the need for a minimum and maximum value suggest that using a circle or cyclical manipulator for the data would be the best approach. As a result, the genres were interpreted through the use of the unit circle.

Since there were seven genres, the unit circle was divided into seven slices, each with a $\frac{2\pi}{7}$ radian arc representing one of the genres. The genres with a higher probability of indicating crunch were placed closer to 0 radians.. So action was given an angle of 0 radians, while simulation was given an angle of $\frac{12\pi}{7}$ radians, as demonstrated in Figure 3.

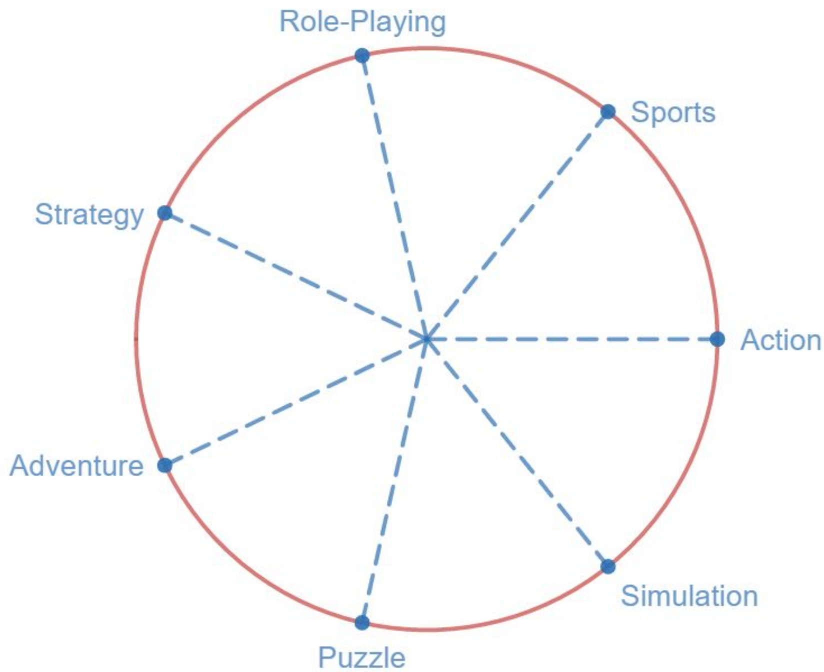


*Figure 3 - A unit circle divided into the seven major game genres.*

Then the genre of each game published by a company was analyzed to find an average angle, which would provide an "average genre". For instance, the game studio Rocksteady Studios had published three action games, two sports games, one role-playing game, and one simulation game. So the points on the unit circle for each of these genres were averaged until a single point was found within the unit circle. The action genre is at the angle 0 radians, so it lies on the unit circle at (1, 0). The sports genre, being at angle $\frac{2\pi}{7}$ radians, lies at the point (0.624, 0.782). Finally, the role-playing and simulation genres lie at (-0.223, 0.975) and (0.623, -0.781) respectively. However, it wasn't enough to just take genre into account; the number of games under each genre had to be accounted for as well. So for Rocksteady Studios, since there were three action games, the point (1, 0) was multiplied by 3, resulting in the point (3, 0). Similarly, the sports genre was multiplied by 2, to create the point (1.247, 1.564). Once this was done for all of the genres, the points could be averaged. So Rocksteady Studios's average point was calculated as shown below:

$$x = \frac{1}{7}(3 * \cos(0) + 2 * \cos(\frac{2\pi}{7}) + 1 * \cos(\frac{4\pi}{7}) + 1 * \cos(\frac{12\pi}{7})) \approx 0.664$$

$$y = \frac{1}{7}(3 * \sin(0) + 2 * \sin(\frac{2\pi}{7}) + 1 * \sin(\frac{4\pi}{7}) + 1 * \sin(\frac{12\pi}{7})) \approx 0.251$$

Finally, the score was calculated by finding the angle to this point on the unit circle. This was calculated by taking the inverse tangent of the y-coordinate over the x-coordinate:

$$s = \arctan(\frac{0.251}{0.664}) \approx 0.361$$

This represented the *genre score* for Rocksteady Studios, and another input for the prediction model. These formulas were then generalized, where $g_k$ represents the number of games in each of the seven genres.

$$x = \frac{1}{7} * \sum_{k=0}^{6} g_k * \cos(\frac{2\pi}{7} * k)$$

$$y = \frac{1}{7} * \sum_{k=0}^{6} g_k * \sin(\frac{2\pi}{7} * k)$$

$$s = \arctan(\frac{y}{x})$$

With these equations, the *genre score* could be calculated for any game studio based on a list of the studio's published games.

The final set of data was in reality two scores, but from a single online source. They were simpler to calculate than the others because they relied on basic ratios. These ratios were based on the assumption that negative reviews of a company by employees indicate unhealthy workplace practices. So reviews of a game studio by employees were gathered from Glassdoor, a website that provides information about companies to job searchers. Then the average rating of a company was found. This is known as the *review score*. However, there was one other piece of information that could be gathered from the data, and that represents the second score that could be derived from this online source. Each review on Glassdoor has a section where employees can detail cons or negative aspects of working at a company. By searching the cons of each review for keywords such as “crunch” or “overtime”, the model could create an even better picture of whether a studio crunches. So the ratio of reviews containing those keywords to total reviews was supplied as a fourth input, known as the *cons score*. Once the four inputs were calculated they could be fed into a logistic regression model, a form of statistical modelling that classifies a set of inputs into a singular probability between 0.0 and 1.0.

The idea of a regression model is that a base function is manipulated to fit a set of data as accurately as possible. In this case, that data would be the four calculated scores as inputs and a binary value of zero or one as the output. An output of one represents a studio that employs crunch and an output of zero represents one that doesn’t. Since the outputs are not continuous analog numbers, this is known as a classification model. The trick to creating the model is to use logistic regression, which creates a model using linear regression and feeds it into a sigmoid function. The linear regression model creates weights that are calculated by supplying a list of inputs and their known outputs, helping to train the model. This results in an equation for the resultant line as seen below, where $w_k$ is a different weight or slope of the line.

$$y = w_0x_0 + w_1x_1 + ... + w_kx_k + w_{k+1}$$

In this case, the line has five weights, with one associated for each input and an extra weight representing the y-intercept of the function. The output of this line was then fed into a sigmoid function, which was used because it produces outputs confined to the range [0, 1]. A sigmoid function is simply a line of the equation $y = \frac{1}{1+e^{-x}}$ , as graphed below in Figure 4.

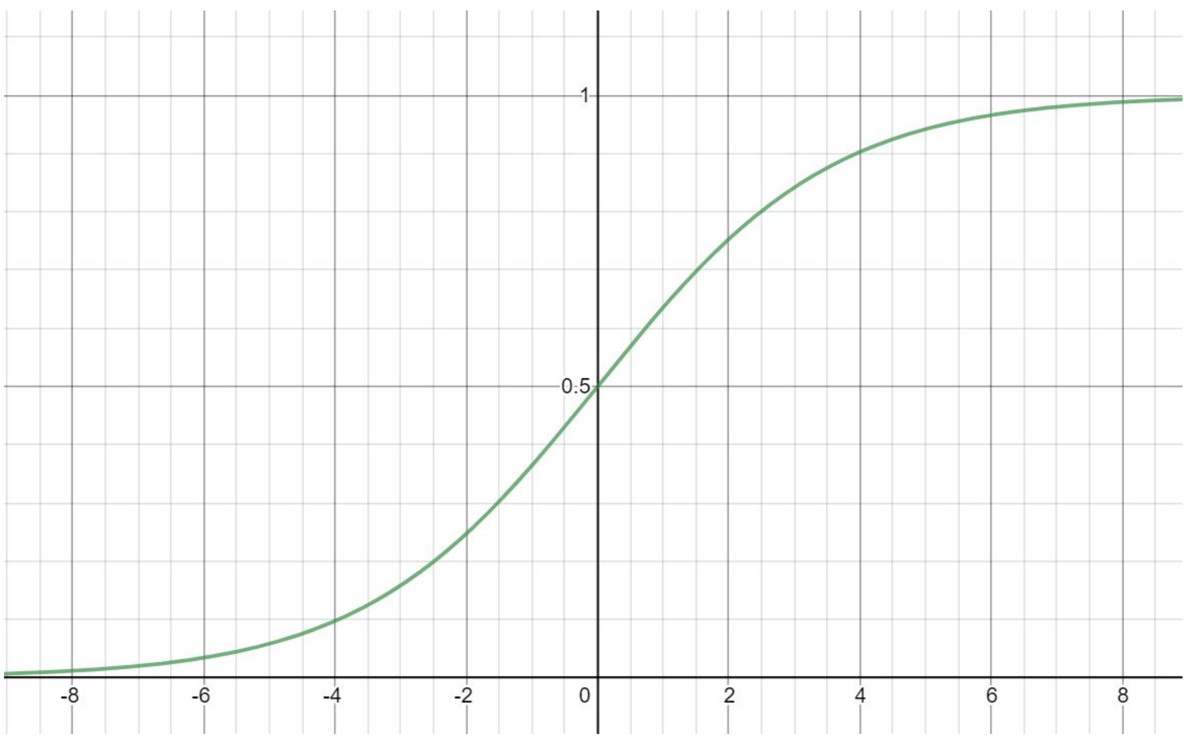


*Figure 4 - A sigmoid function.*

This entire process sounds great in theory, but how did it hold up with the actual data?

The first step was to create a data set, or a list of game studios that are known for certain to either use or not use crunch. These studios acted as the training data for the model, providing the known inputs and outputs necessary to calculate the model's weights. The studios used for training are shown below.

**Known to crunch**

Rockstar Games, Treyarch, Epic Games, Electronic Arts, Telltale Games, BioWare, NetherRealm Studios, CD Projekt, Capcom, Techland, Sir-Tech, Wargaming

**Known to not crunch**

IO Interactive, Relentless Software, Schell Games, Playground Games, 4A Games, Avalanche Studios, Bandai Namco, Milestone S.r.l.

This was based on manual research of each company, which involved looking into journalistic and employee-based reports of workplace practices within the studios. Once the training data was established, it was also necessary to establish test data. Test data is used to see how a model compares to actual results, but excludes the studios utilized in the training data. This is to avoid overfitting the model, a term that describes when a model is fit to be extremely accurate on a particular set of data, but becomes inaccurate on any previously unseen inputs. The studios used in the test data set are listed below.

**Known to crunch**

Bungie, Platinum Games, FromSoftware, Naughty Dog, Konami, Revolution Software, Funcom, Activision, Origin Systems, Looking Glass Studios, Blizzard Entertainment, Tose, Eagle Dynamics, Tripwire Interactive, Relic Entertainment, Firaxis Games, Take-Two Interactive, Neversoft, 5th Cell

**Known to not crunch**

Respawn Entertainment, Square Enix, Monolith Productions, Insomniac Games, Id Software, Rocksteady Studios, Machine Games, Warthog Games, SIE Santa Monica Studio, Fullbright, LucasArts, Wadjet Eye Games, Matrix Software, Black Isle Studios, Atlus, Dovetail Games, Razorworks, Gaijin Entertainment, Introversion Software, Klei Entertainment, Paradox Development Studio, Midway Games, Sony Interactive Entertainment, Nintendo, PopCap Games, Zachtronics, Croteam, Fireproof Games

Once these lists had been established, the inputs were gathered from the online databases using a custom written program.[4] The calculated inputs and outputs for the training data are shown below in Figure 5.

[4] The program is available at https://www.github.com/JamesEConnor/GameStudioScorer

| *Studio Name* | *Game over Time Score* | *Genres Score* | *Review Score* | *Cons Score* | *Expected output E(x)* |
|---|---|---|---|---|---|
| Rockstar Games | 2000.748 | $1.04381 * 10^{-8}$ | 0.67 | 0.30 | 1.0 |
| Treyarch | 620.3948 | $3.739365 * 10^{-8}$ | 0.745 | 0.425 | 1.0 |
| Epic Games | 1000.606 | 0.0 | 0.69 | 0.125 | 1.0 |
| Electronic Arts | 9302.457 | $1.33156 * 10^{-8}$ | 0.695 | 0.025 | 1.0 |
| Telltale Games | 100.0183 | $1.049958 * 10^{-9}$ | 0.55 | 0.20 | 1.0 |
| Bioware | 800.7363 | $2.475505 * 10^{-8}$ | 0.68 | 0.10 | 1.0 |
| NetherRealm Studios | 100.0915 | $1.86 * 10^{-8}$ | 0.722 | 0.222 | 1.0 |
| Capcom | 2832.313 | $-4.792948 * 10^{-8}$ | 0.470 | 0.125 | 1.0 |
| Techland | 300.1788 | $2.105936 * 10^{-8}$ | 0.686 | 0.143 | 1.0 |
| Sir-Tech | 100.1365 | $2.721198 * 10^{-8}$ | 0.893 | 0.067 | 1.0 |
| Wargaming | 100.1481 | $2.312938 * 10^{-8}$ | 0.67 | 0.0 | 1.0 |
| IO Interactive | 170.2753 | $1.86 * 10^{-8}$ | 0.814 | 0.071 | 0.0 |
| Relentless Software | 30.0136 | $7.95 * 10^{-9}$ | 0.814 | 0.071 | 0.0 |
| Playground Games | 200.32 | $4.17 * 10^{-9}$ | 0.718 | 0.0 | 0.0 |
| 4A Games | 2022.077 | $6.96 * 10^{-9}$ | 0.769 | 0.154 | 0.0 |
| Avalanche Studios | 320.4084 | $2.89 * 10^{-8}$ | 0.683 | 0.083 | 0.0 |
| Bandai Namco | 100.007 | $-1.13 * 10^{-7}$ | 0.57 | 0.0 | 0.0 |
| Milestone S.r.l. | 200.1008 | $-1.03 * 10^{-8}$ | 0.526 | 0.0 | 0.0 |

*Figure 5 - The inputs and corresponding outputs for the training data.*

Running logistic regression with these values produced a model with the following weights:

$$w_0 =- 3.71357002525046 \times 10^{-8}$$

$$w_1 = 7.21174331832094 \times 10^{-7}$$

$$w_2 =- 3.23401282259892$$

$$w_3 = 16.9924949222297$$

$$w_4 = 1.07926488584655$$

Some important statistical information could now be obtained from the model. The first is the correlation between the various independent variables or inputs, demonstrated by a measurement known as an odds ratio. The odds ratio represents how much of an impact a specific weight has on the overall output. As an odds ratio approaches 0.0 or $\infty$, its associated weight has a larger impact on the output of the model. As an odds ratio approaches 1.0, it has less of an impact on the model's output. To gain a better picture from the odds ratios, one can calculate the 95% confidence interval for the odds ratios. This provides two numbers that the odds ratios are most likely in between. The wider this interval, the less sure one can be of the true odds ratio. The following confidence intervals were calculated for the model.

$$OR_0 = [0.999998601016136,\ 1.00000132471432]$$

$$OR_1 = [0.999997388410105,\ 1.00000405395019]$$

$$OR_2 = [1.02226897236851E-06,\ 1518.4726501918]$$

$$OR_3 = [0.810502926670688,\ 709151408131148]$$

As one might notice, the confidence intervals for the *game over time score* and *genres score* have small ranges that are close to one, while the confidence intervals for the *review score* and *cons score* have larger ranges. This indicates that the first two inputs most likely have a smaller impact on whether a game studio employs crunch, while the last two likely have a much larger impact. Therefore, reports from employees, such as online reviews, represent the best indicator of unhealthy workplace practices being employed in a game development studio.

Now that statistical patterns have been established, it's important to analyze the actual accuracy of the model. The first measurement of this is the ratio between correct predictions and total predictions. This measurement of accuracy is simple enough to calculate from the test data listed earlier.

$$a = \frac{correct}{correct + incorrect} = \frac{38}{38+20} = \frac{38}{51} = 0.7451$$

The model had an overall accuracy of 0.7451. The model, however, also provides probabilities, not just predictions. The probabilities closer to the extrema of 0.0 and 1.0 represent a higher confidence in those predictions. By looking only within this high-confidence interval of [0.0, 0.2] and [0.8, 1.0], a high-confidence accuracy was calculated as shown.

$$a_h = \frac{correct\ high\ confidence}{correct + incorrect\ high\ confidence} = \frac{14}{14+3} = \frac{14}{17} = 0.8235$$

So the model was confidently correct with three of every four higher-confidence predictions. Another way to demonstrate the accuracy of the model was to see how well the best-fit function conforms to the data. This could be seen in both the root mean square error (RMSE) and in the R-squared value of the model.

The RMSE represents the total amount that each output is off by from the line of best fit. For instance, if there are three points (1, 2), (2, 2), and (3, 4), the line of best fit is $y = x$. To solve for the RMSE of that line of best fit and the accompanying data, the square root of the average square error is taken, as seen below, where $y$ is the output of the line of best fit and $g$ is the output of each point.

$$RMSE = \sqrt{\overline{(y-g)^2}}$$

$$RMSE = \sqrt{\tfrac{(1-2)^2+(2-2)^2+(3-4)^2}{3}}$$

$$RMSE = \sqrt{\tfrac{(-1)^2+(0)^2+(-1)^2}{3}}$$

$$RMSE = \sqrt{\tfrac{1+0+1}{3}}$$

$$RMSE = \sqrt{\tfrac{2}{3}}$$

$$RMSE \approx 0.8165$$

The lower the root square mean error is for a model, the better it conforms to the provided training data. The RSME for the logistic regression model on the test data was approximately 0.05816. The other metric used to observe how well the model conforms to the data is the $R^2$ (or coefficient of determination) of the model. The coefficient of determination is formally defined as “the proportion of the variance in the dependent variable that is predictable from the independent variable(s)”. It represents how close a fitted function is to some data points. It’s calculated by taking the ratio between the residual sum of squares ($SS_{res}$) over the total sum of squares ($SS_{tot}$), and subtracting it from 1.0. The closer the value is to 1.0, the better the fit is for the data.

$$SS_{tot} = \sum_i (y_i - \overline{y})^2$$

$$SS_{res} = \sum_i (y_i - f_i)^2$$

$$R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$$

For this model, the $R^2$ value was calculated to be 0.3066. This may seem low, but when making predictions involving human behavior and psychology, $R^2$ values lower than 50% are normal and expected.

Looking at these measurements, there are certain conclusions that can be drawn about the relationships between the downloaded information and indications of whether a studio crunches. Looking first at the model’s fit, it seemed to be relatively close to the data. It had a small root mean square error and, while the explained variance represented by $R^2$ is low, overall it made a good fit. With an accuracy of about 75%, and a high confidence accuracy of about 82%, the model did a relatively good job of predicting whether a studio employs crunch.

The bigger and more significant question is: what patterns in the data are indicated by the model’s weights? Returning to the odds ratios for a moment, the most likely indicator of crunch and unhealthy workplace practices are those of reviews and reports from employees. The number of games published over time and the genres of those games don’t have a statistically significant effect on whether a studio is likely to have employed crunch. Having said that, examining the small weights of those two inputs indicates support for a few of the assumptions originally made at the beginning of this paper. While it may not have much of a noticeable effect, a lower *game over time score* does seem to indicate a higher likelihood of unhealthy workplace practices. The weights for the *review score* and *cons score* show that as the rating of a studio decreases and the use of words like “overtime” in reviews increases, the likelihood of crunch again increases. The only assumption that was not supported by the weights was the *genres score*, which had a positive weight, as opposed to the expected negative value that would show support for the original assumption regarding genres.

While it is possible to predict whether certain studios employ crunch, the accuracy of this particular model is not 100%. Having said that, mathematics has proven that there are certain patterns exhibited by studios that indicate a use of crunch. Although predicting its use is as much a question of psychology as it is of pure data, crunch and unhealthy workplace practices can be identified and predicted using data manipulation and logistic regression. Math has the power to help people better understand the indicators and patterns of crunch in game development studios, and therefore has the power to support and assist game developers across the United States and around the world.

Works Cited

Campbell, Colin. "Game development crunch can be changed by bringing abuses to light." *Polygon*, Vox Media, 29 Apr. 2019, www.polygon.com/2019/4/29/18518403/ game-development-crunch-unions-reporting. Accessed 15 Oct. 2019.

Game Workers Unite. "FAQ." *Game Workers Unite*, www.gameworkersunite.org/faq. Accessed 24 Nov. 2019.

Schreier, Jason. "Video Games Are Destroying the People Who Make Them." *The New York Times*, 25 Oct. 2017, www.nytimes.com/2017/10/25/opinion/work-culture-video-games-crunch.html. Accessed 14 Oct. 2019. Editorial.