

# How Machine Learning Can Help Game Design

Tiago Tex Pine

2024

# This Presentation

- How game data is usually tracked and shaped.
- Extremely brief introduction of ML (for game makers)
- How to apply Regression, Classification, Clustering, Dimensionality Reduction and LDA for game design.
- Reinforcement Learning for NPCs, economy simulation, game QA - and its many application challenges.
- LLM for NPC dialogue? Nowhere near as easy as it looks.

*What we won't cover:* Gen AI in game art or code. No product pitch.

# About Me:



- Product Owner: a mix of team leader, game designer and producer.
- Developing video games since 2005. Data scientist between 2017 and 2023.
- Worked on various components of game development across the years: engineering, UX, game design, business development, market research, economy, monetization, data science and team leadership.
- 30+ games, across various companies, genres, platforms, business models and team sizes - from 4 to 400 developers.
- Enthusiastic about combining the art of game design with the science of data.
- I also run lots of tabletop RPGs and Dungeons & Dragons in my spare time. :)
- I joined Prodigy **to create games that improve people's lives.**



# How is the Data of Game Activity

Common traits in any database tracking player activity.

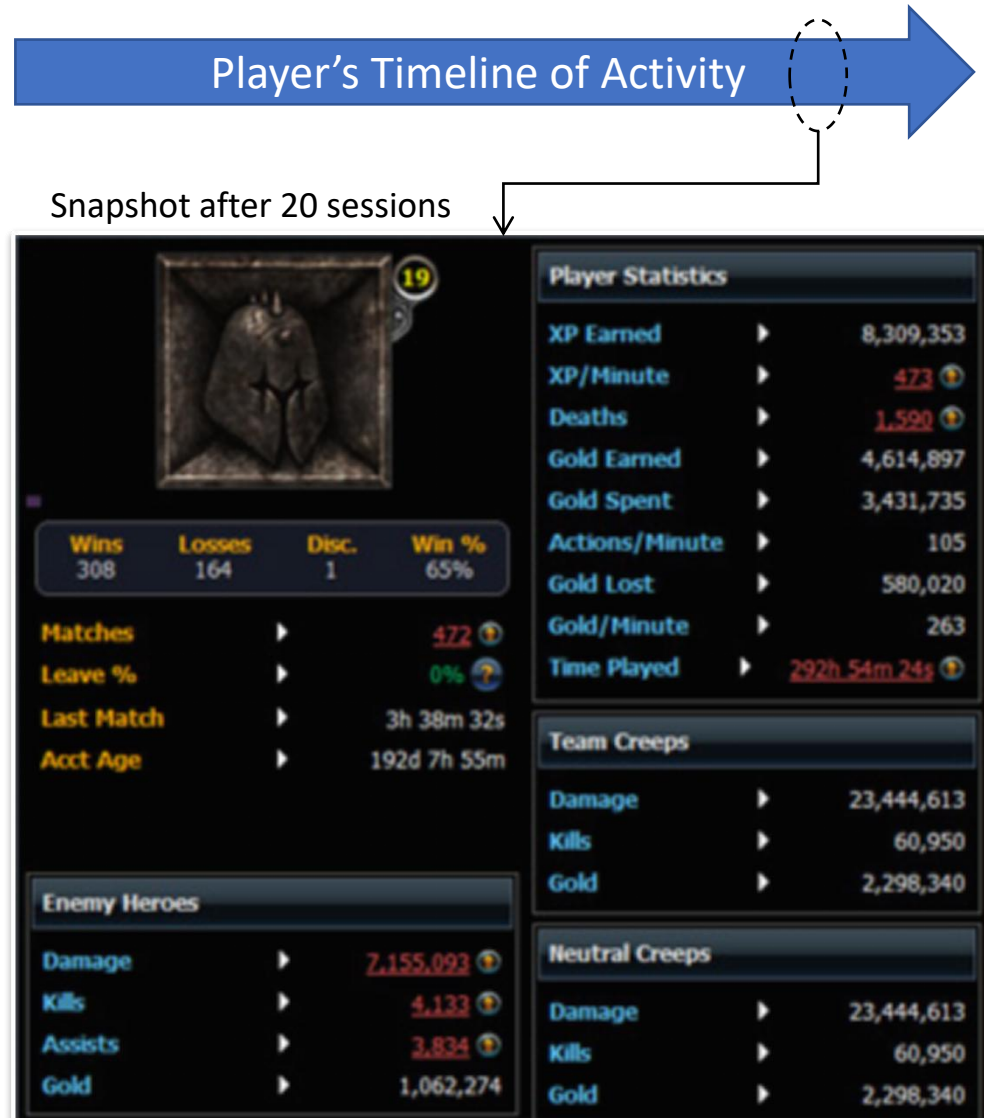
# Creation of data is “under control”

- The way the data is created is **under control of game developers**.
- Like a software or website, game programmers and data analysts can decide exactly what to track, in what format, in what moment of the flow of the code.
- Metadata is under control. *Very little* manual input from users (except voice and text chats).
- Few sources or only one source of data:
  - The game itself, 95% of all data;
    - Player Activities;
    - Server health monitoring;
  - Payment solution;
  - First-party auth (Google Play, iTunes, Steam, Epic);
  - Sometimes a CRM solution with emails, but it’s actually rare.
- Because of the items above, Game Data is usually **not very dirty** - or, at least, much less *compared* to other businesses.

```
{
    session_id: "e7925afa-d0fc-481a-9743-
    ecaa413da8c9",
    player_id: "e548af35-7cb6-4875-a205-
    f01adc9a9715",
    trigger_id: "75da07ac-5918-4f5e-93ec-
    fde79139bea2",
    event_name: "equip_weapon",
    event_id: 32,
    weapon_type: 2, # sword
    weapon_material: 8, # adamantine
    weapon_visuals: 3, # camouflage
    weapon_affixes: [
        {affix_name: "Slaying",
        affix_mod: 0.2},
        {affix_name: "of Guidance",
        affix_mod: 0.3, affix_mod2: 0.1}
    ]
}
```

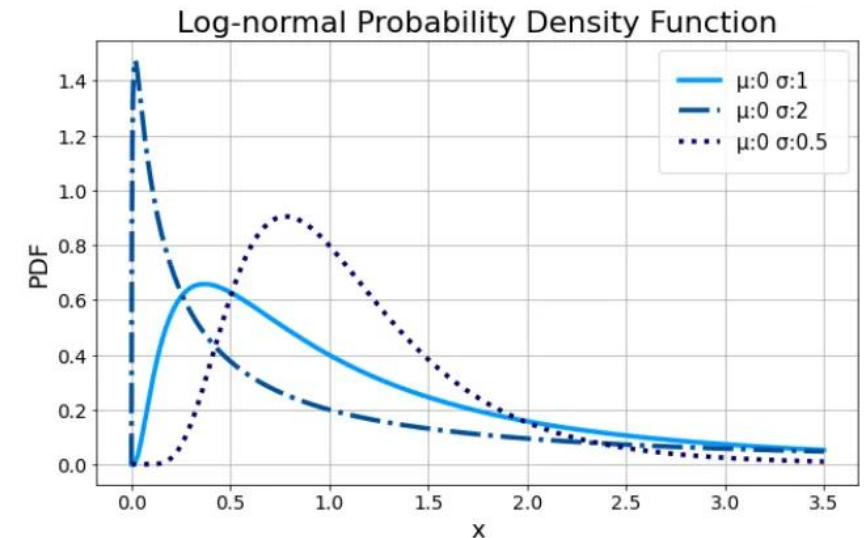
# User-level and Anonymous

- The raw, underlying data is on a user level, with each user choice and click over time.
- But almost always, game developers **do not know demographic information** of these users.
  - We don't know age, gender, location, income, nothing like that.
- Two reasons:
  - Companies that *can* have them are worried about **PII** and don't link account data to player IDs;
  - Companies that *cannot* have them are the *majority*!



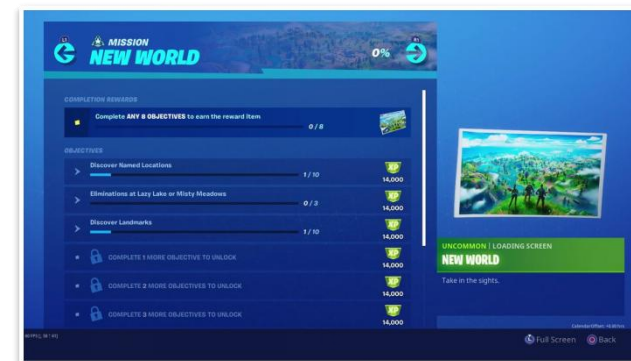
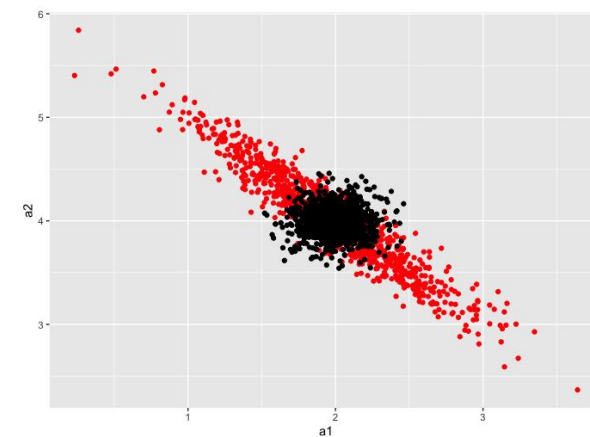
# Non-normal distributions

- Non-normal distributions are **common**.
  - Long-tails, Logarithmic and Log-normals
- Particularly when it comes to revenue, game progression and activities over time, **players are not equal**.
- In free-to-play games: ~20% of the entire user base is *most of your business*.
  - 5% will ever convert;
  - Half your total revenue depends on 0.5%;
- We need to be very careful on using math that assumes data is normally distributed, such as *student's t-tests* for AB tests.



# Multicollinearity and Confounding

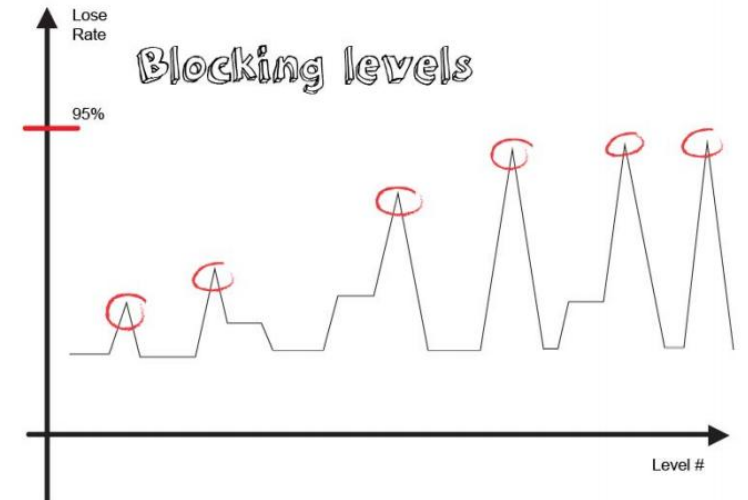
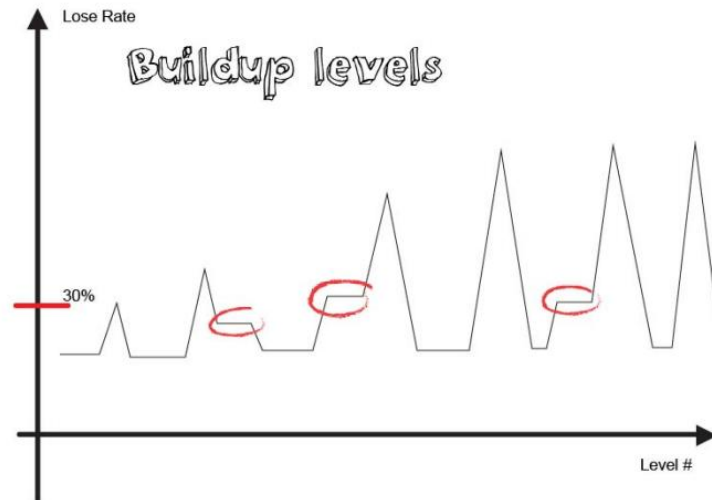
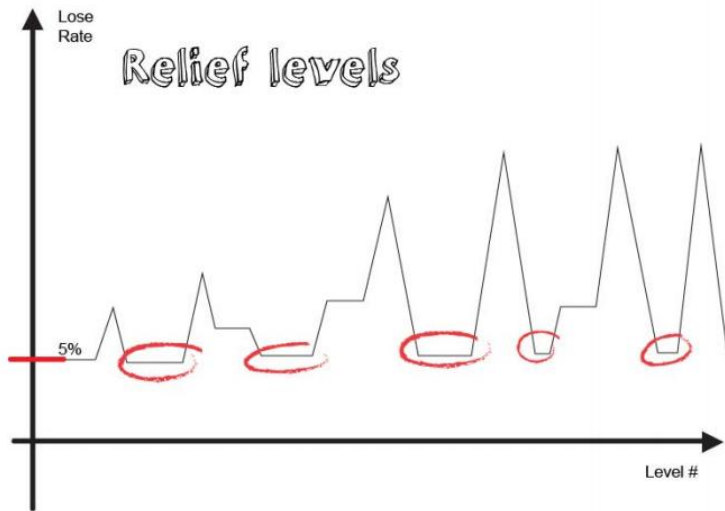
- Dimensions of player progression data can frequently have high but imperfect degrees of collinearity.
- For some few features, it may actually be a *perfect* collinearity.
- Every analysis and ML need to tune, or at least consider, that multiple dimensions may go hand in hand to each other - and sometimes *confounded* in a same “invisible” game mechanics.
- Examples of collinearity:
  - amount of HP vs. character level;
  - enemy level vs. campaign stage;
  - soft currency vs. playtime.
- Examples of confounded features:
  - character level vs. enemy level in an open world game that doesn't auto-scale difficulty;
  - ladder/arena tier vs. usage of a specific hero
  - progress in a Battle Pass vs. popularity of fire spells





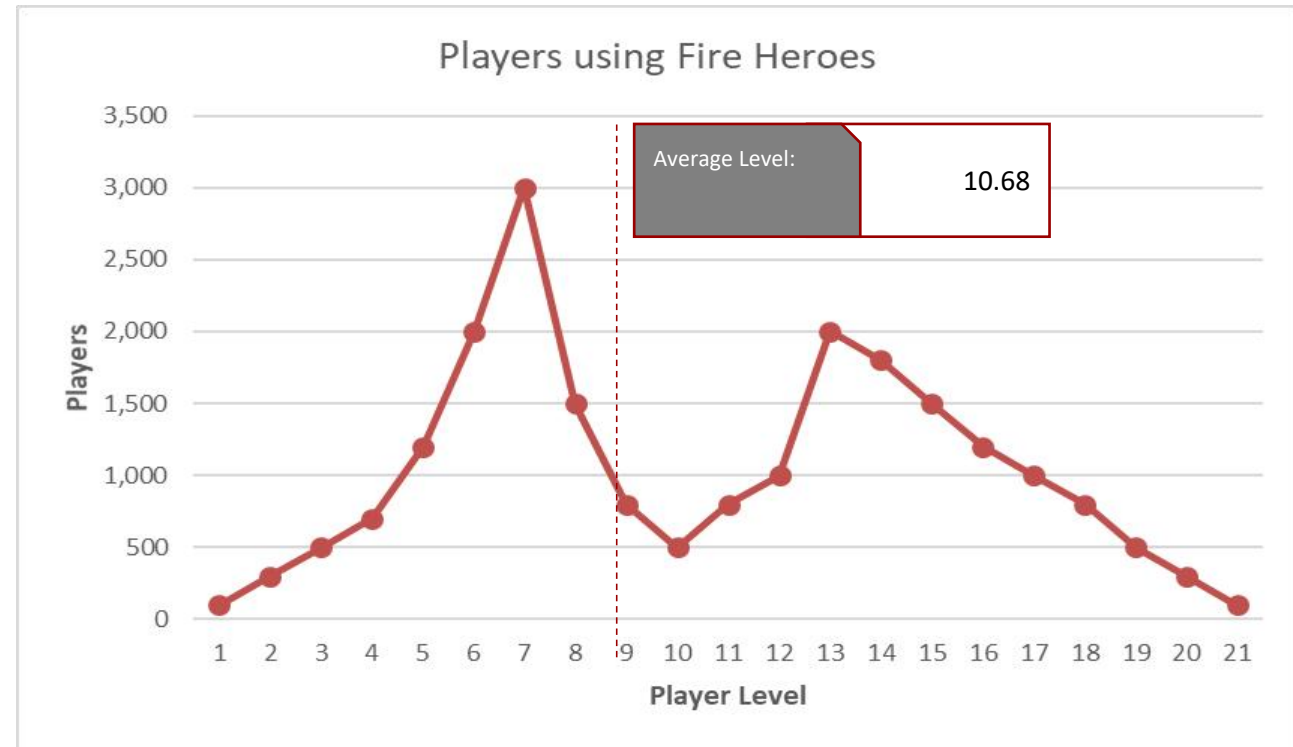
# Experience across player journey

- Every good game has a dynamic progression.
- What may be good in session 5, may not good in session 30.
- Player become more powerful, more skillful, economy changes.
- **Context matters a lot - and context is either tracked *or* designed.**



# Averages will lie

- Relying only on Averages might make you miss have *different populations* among your players.
- Thus, miss design and business opportunities.
- Do data exploration and scatter matrices when dealing with multiple dimensions.



- Averages would tell us Fire Heroes are more used around Level 10.
- But in fact, there's a problem where they are *less* used at this point.

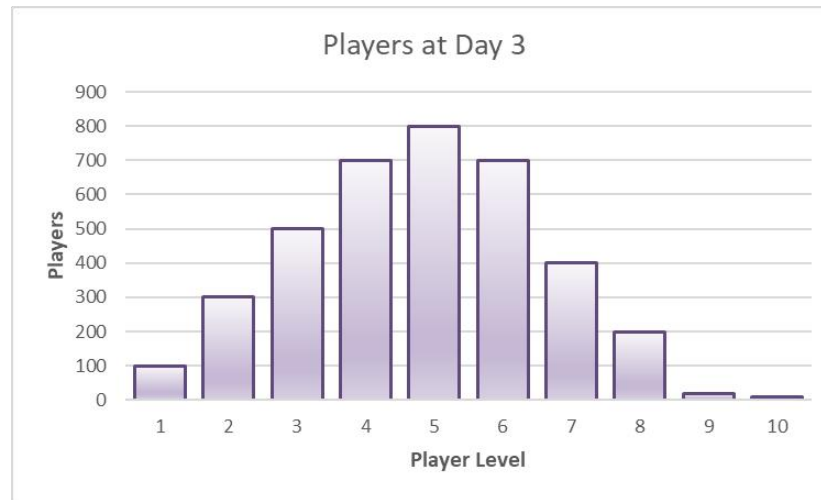
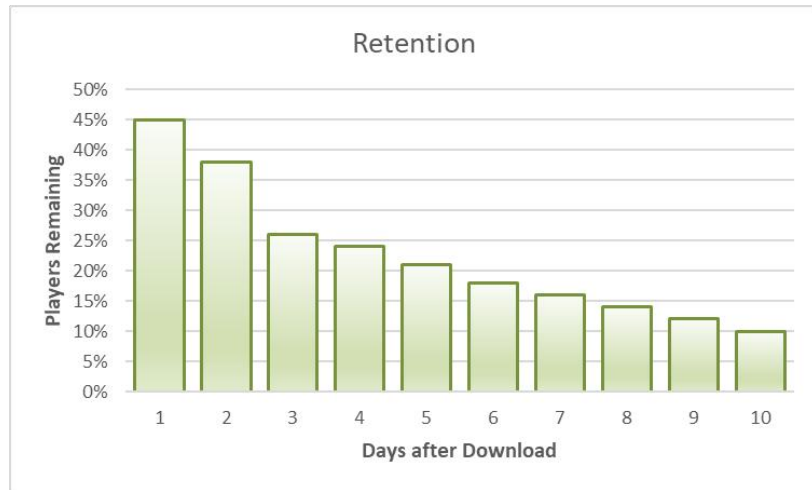
# Business KPIs don't really help game designers

- Things like ARPU, Conversion, ARPPU, and even Retention are good to know, generally.
- But for designers, they are *not* real insights. They are “*trivia data*”.
- If a designer during Live Ops is like a doctor trying to diagnose a patient:
  - Those KPIs are like thermometers.
  - But what doctors really like is to have blood tests and radiographies.



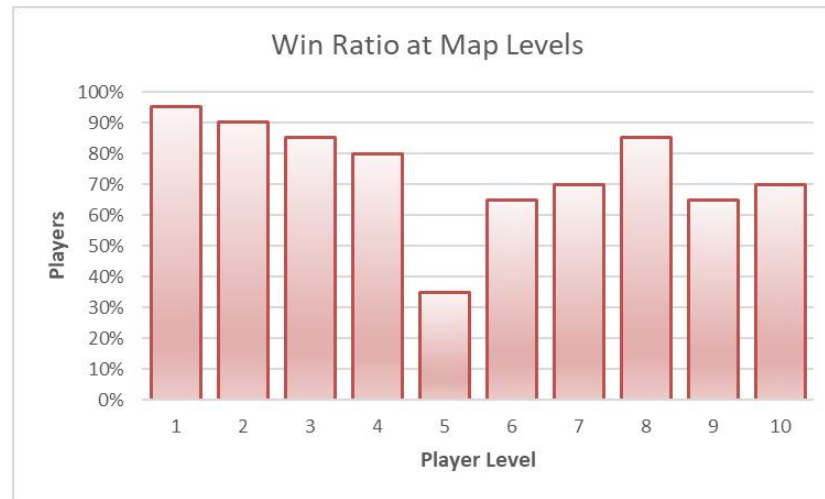
ARPPU  
Conversion K-factor  
ARPU CPI DAU  
ARPPAU LTV  
Retention





- Player Level tends to be 4-6 at day 3.
- What are they doing to drop?

- Retention points to a problem in day 3. But why?



- The win ratio of map level 5 is very *low*.
- We found a real insight that can help re-balancing.

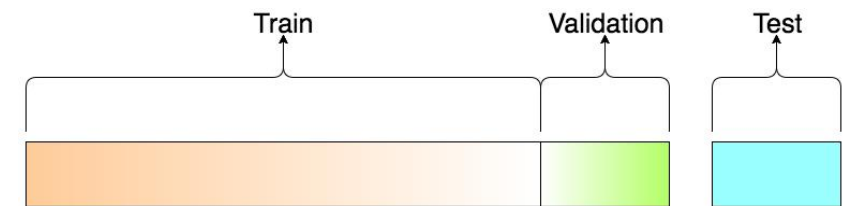
# Premises of ML for game design

Super brief, for game developers

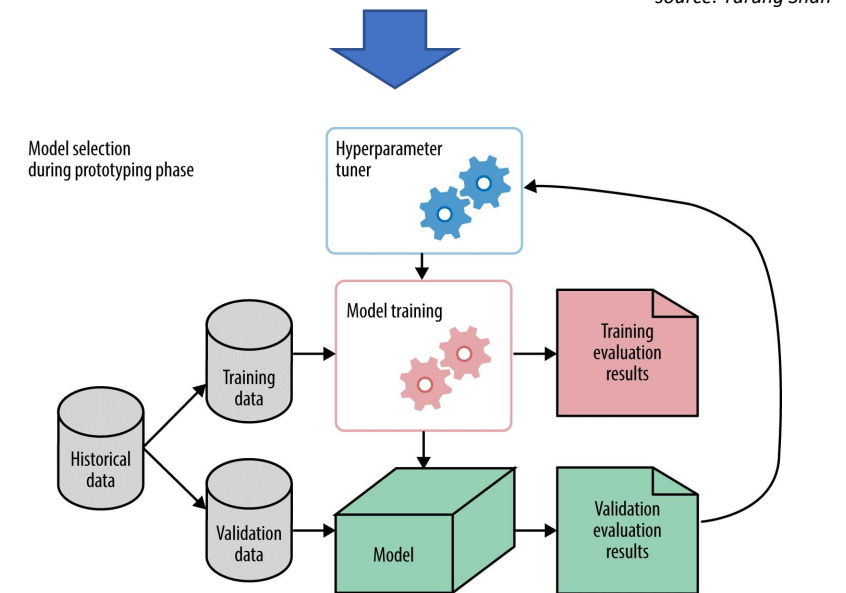
# Needs lots of players

(for the most part)

- Machine Learning = Statistical Learning.
- Hopefully enough to split your data in 3 subsets (70%/15%/15%) and still have > 3,000 data points (table rows) in the smaller ones.
  - Those are sets to, respectively, **train, test and validate**.
- Anything below 50 users is unusable.
- Data generated by developers, company workers and QAs are **NOT** good enough and should be considered extremely biased.

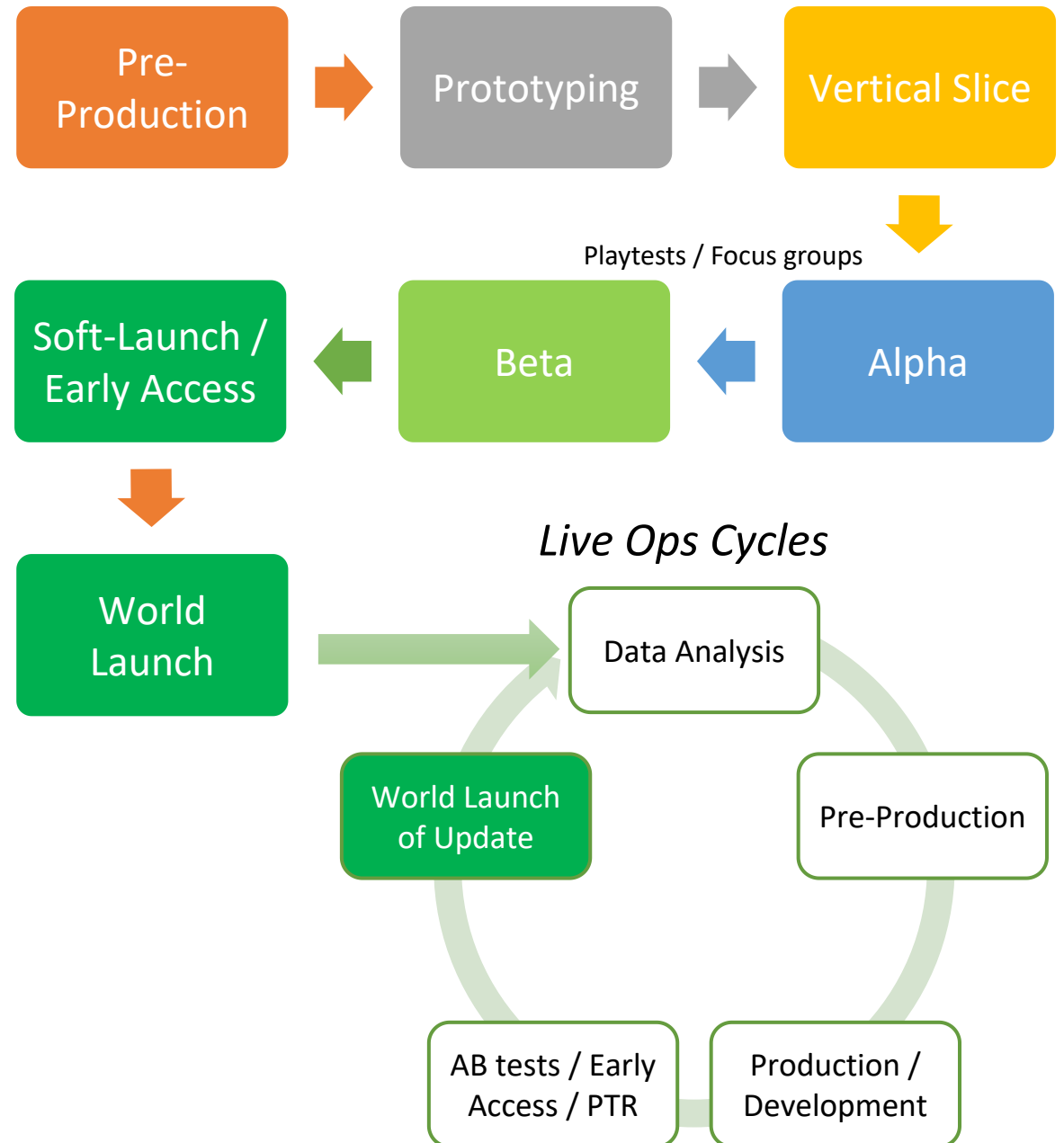


\*source: Tarang Shah



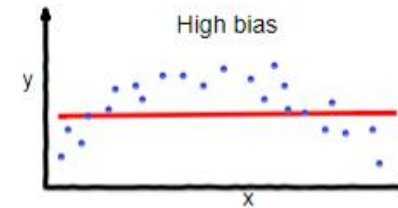
\*source: Alice Zheng

- Thus, ML mostly **can't be used in early stages of development** of a new game, where the only real player data is coming from playtests.
  - No, can't use the data from another game - usually they don't exist anyway.
  - Data from another game *will be biased to that game's mechanics and dynamics (metagame)*
  - *Exception:* some applications of Reinforcement Learning.
- However, once beta / pre-launch / soft-launch / live ops begin, then we can start unlocking ML.

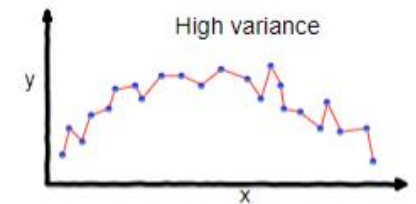


# Must use the output as a generalization.

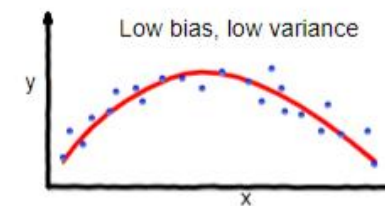
- The best ML models are the ones that are capable of generalizing. They are neither trying to be 100% precise nor too broad.
  - Bias vs. Variance tradeoff
  - Overfitting vs. Underfitting
- When properly tuned, they provide generalizations that are more robust statistics than Averages.
  - KPIs based on Averages are common and easy to explain
  - But *dangerous* simplifications.
  - "*When Bill Gates enter in a bar...*"



underfitting



overfitting



Good balance



# Do not look too much into specific users.

- Playtests and focus groups are **qualitative data** - also very important, but not usable in ML.
  - Game designers are used to this kind of data, from **playtests** during development.
- Models of statistical learning that make good generalizations *will* eventually misclassify very particular cases that not represented enough in the training data.



*"Machine Learning models are statistically impressive, but individually unreliable."*

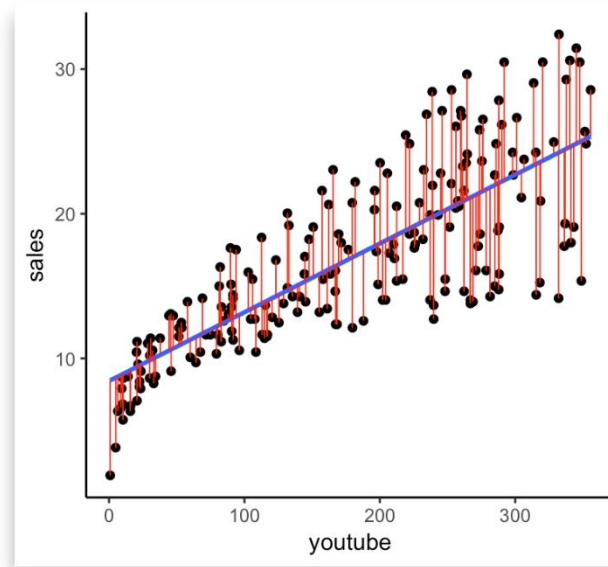
- John Launchbury, the Director of DARPA's Information Innovation Office

# ML to understand player behavior

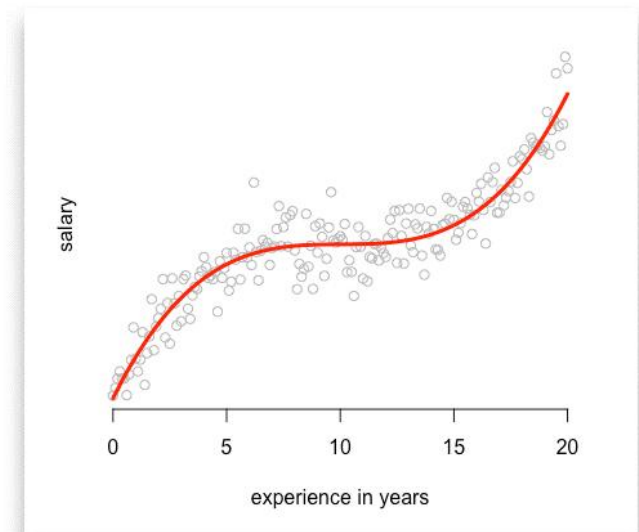
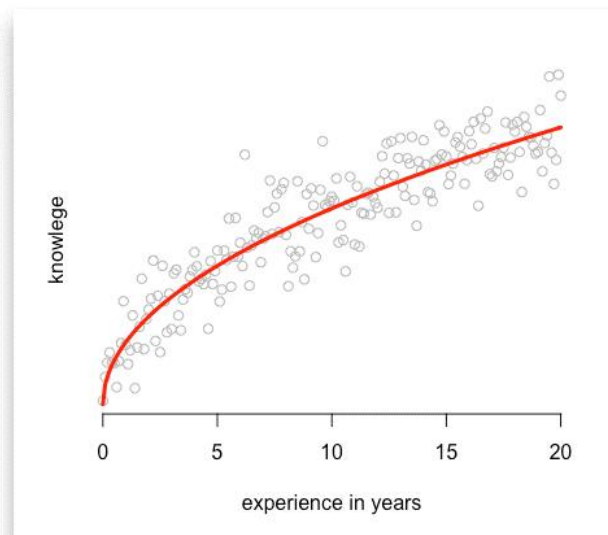
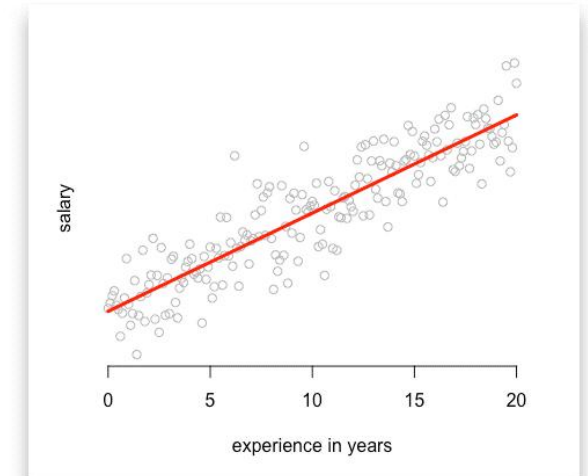
Decomposing multi-dimensional spaces into actionable directions.

# Regression

- Algorithms that try to predict a value based on a variable + historical data. The most common ones fit a line across 2 or more dimensions.
- "Least Squares" regression is the simplest one, but only effective when the data is relatively linear.
- Non-linear regressions are possible if your data "looks" like it could be fit in another function, such as polynomial or logarithmical.
  - But requires plotting of the distribution and careful study first.

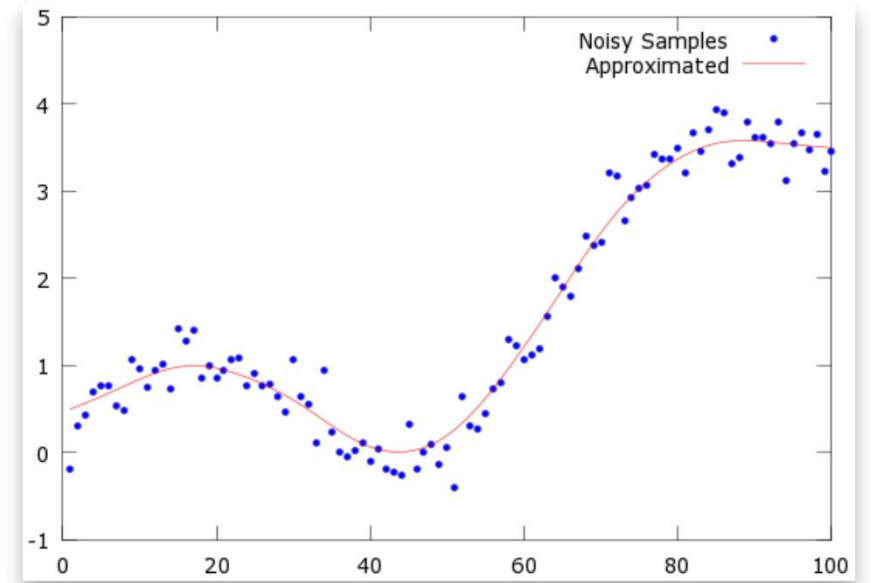


*\*source: Björn Hartmann*

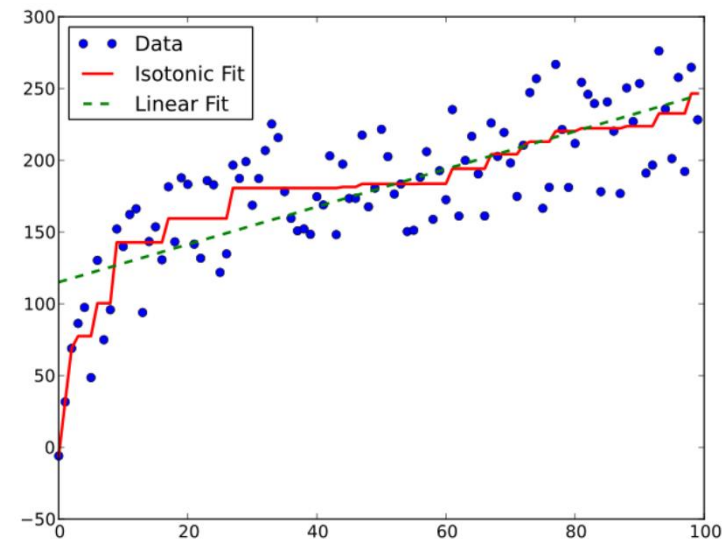


*\*source: Alboukadel Kassambara*

- But generally, **gameplay data** is much more messy and it's usually more efficient to regress on **non-parametric models**.
- The game, metagame and player habits are changing over time and "disturbing" the curves.
- More efficient models to help game design are non-parametric, such as **Isotonic** and **Gaussian Kernel** regressions.

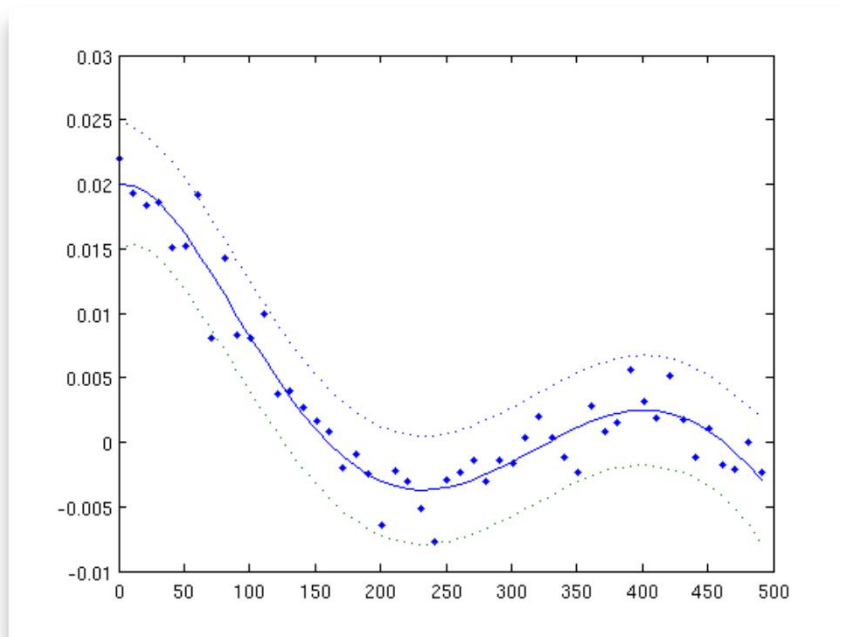


*\*source: Chris McCormick*

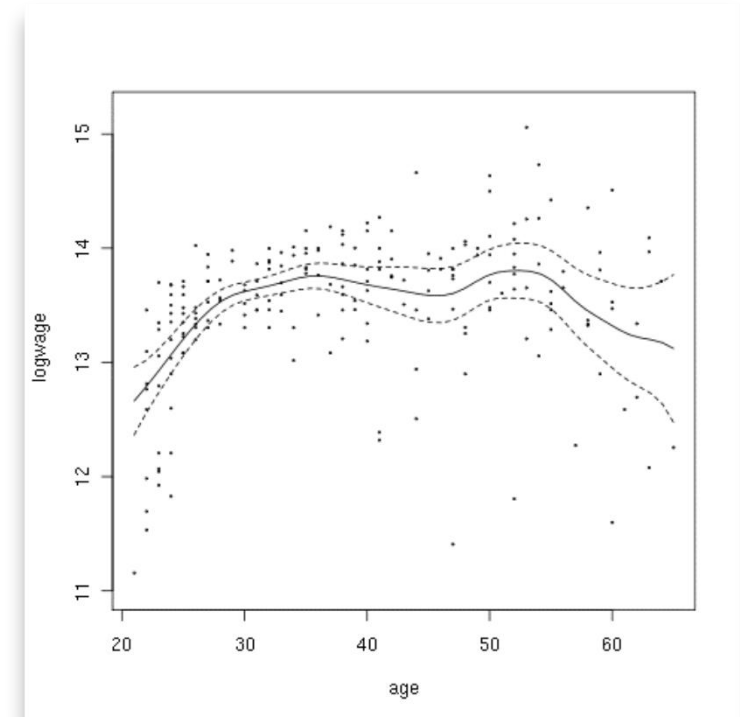


*\*source: Wikipedia*

- It's also useful to project variability bounds, or *confidence intervals*, in the predictions. So the expected variance is well known by designers.
- Example: "We expect players that play 10 hours to have earned 2554 bucks, varying between 4125 and 2145."



\*source: MatLab

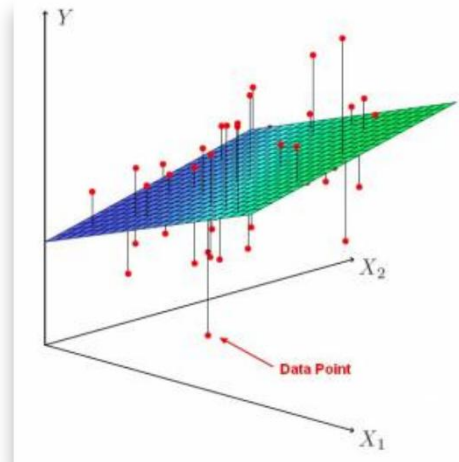


\*source: Wikipedia

- Things get "interesting" when using multiple dimensions, which is common.
- But if too many dimensions are being used, the **curse of dimensionality** kicks in and it's probably better to regress with Deep Learning.
  - DL can also do multi-output regressions, which are useful for economy balancing.

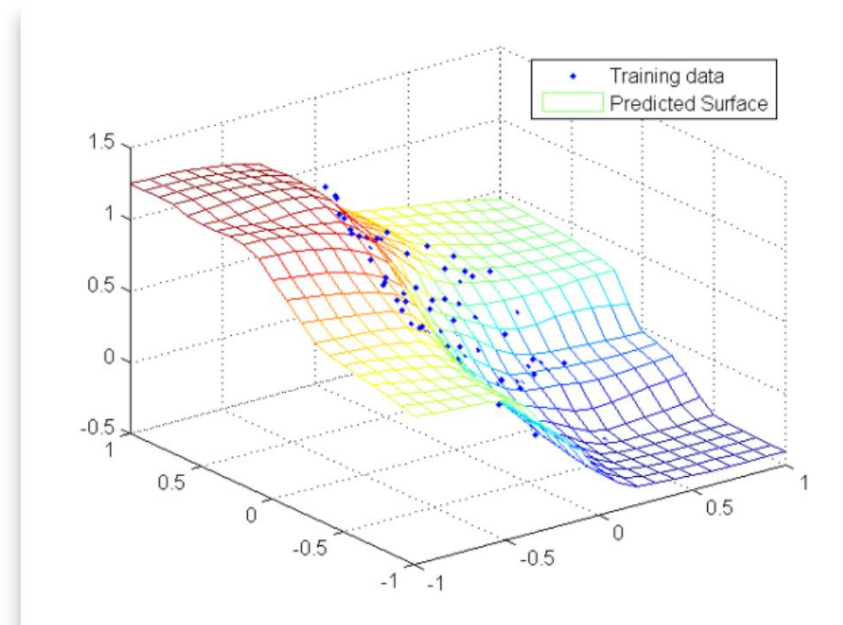
When the dimensionality increases, the volume of the space increases so fast that the *available data become sparse*. In order to obtain a reliable result, the *amount of data needed often grows exponentially* with the dimensionality.

Linear:



\*source: MatLab

Gaussian Kernel:



\*source: MatLab

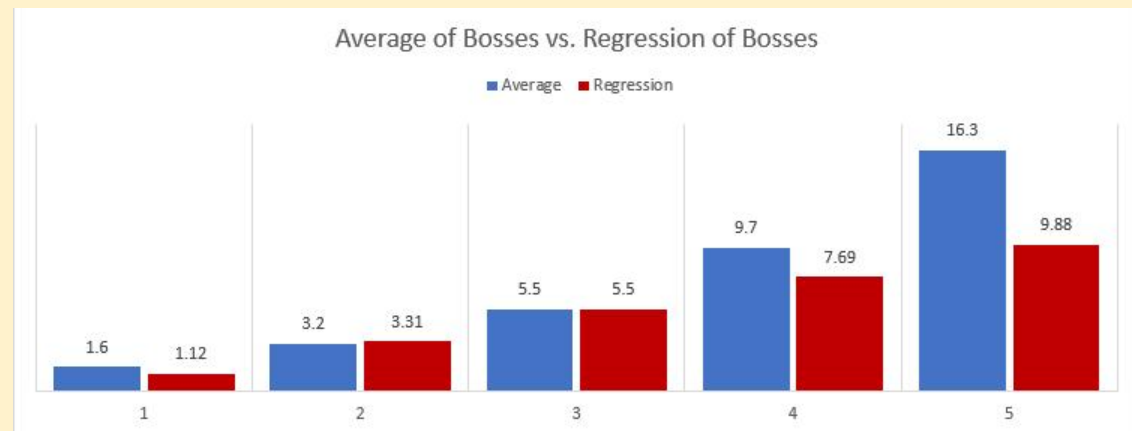
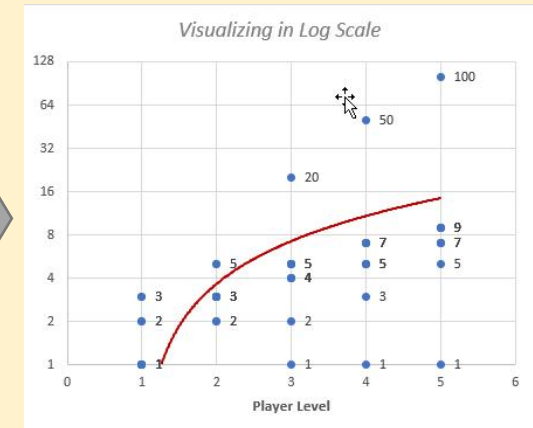
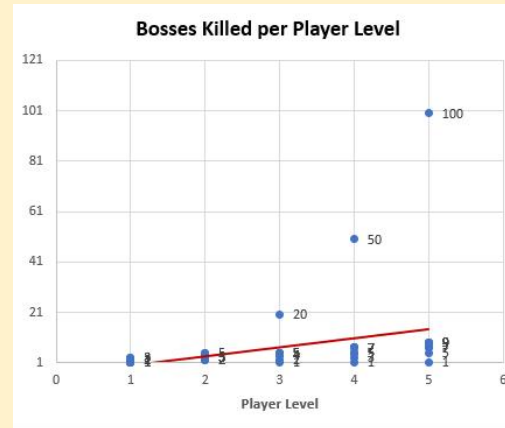


## How Regression help Game Designers

Projection of expected values across many different aspects of the player progression:

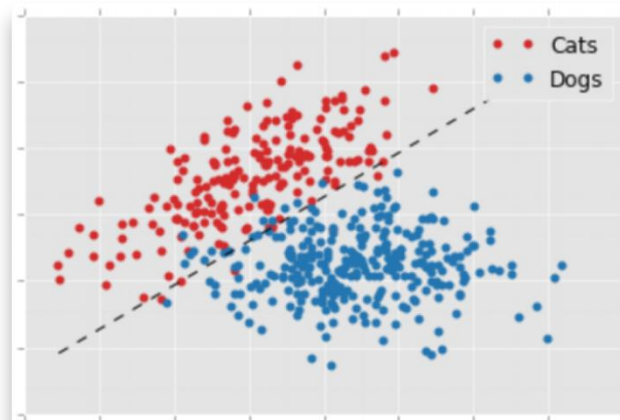
- Resource accumulation per game milestone.
- Time to reach game milestones.
- Time to accumulate high level items.
- Resource inflation over time.
- Rates of Tutorial completion.
- Participation on events / game modes.
- Combat efficiency over time and over player levels (for example, on win ratios)

Also, **Regression models could replace analysis based on averages**, as it is more robust against outliers and can use optimization techniques like Gradient Descent. Example with a *very simple* model (least squares):

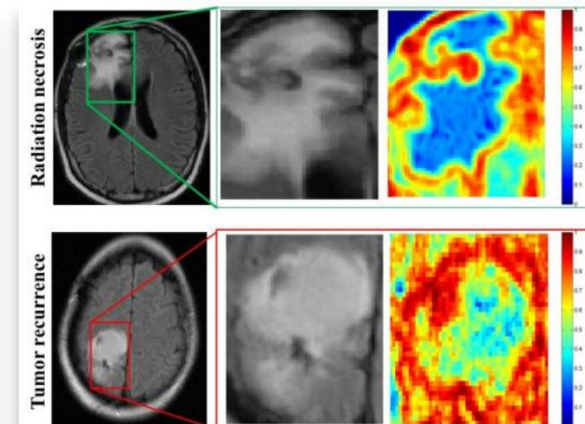


# Classification

- The goal of these models is to classify a data point into categories (or *labels*).
- The models work by discovering criteria that can somehow separate data points in a multi-dimensional space.
- Ideally, the best models are capable of creating complex boundaries to differentiate between *several classes*.
- Classification models are a very useful bunch, widely used in the most famous applications of Machine Learning across many industries.

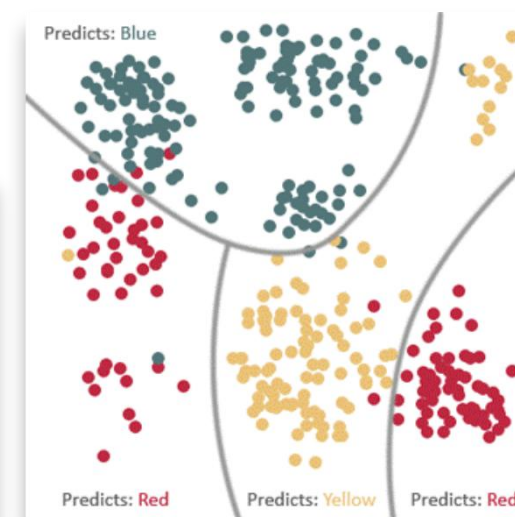


\*source: Hyunjik Kim



A similarity matrix table for movie recommendations. The columns represent movies: SHERLOCK, BRIDGES ON FIRE, AVENGERS: ENDGAME, JOURNALS, and WALKING DEAD. The rows represent users, indicated by avatars. The table shows similarity scores between users and movies. A hand cursor is pointing to the cell for the second user and 'BRIDGES ON FIRE'.

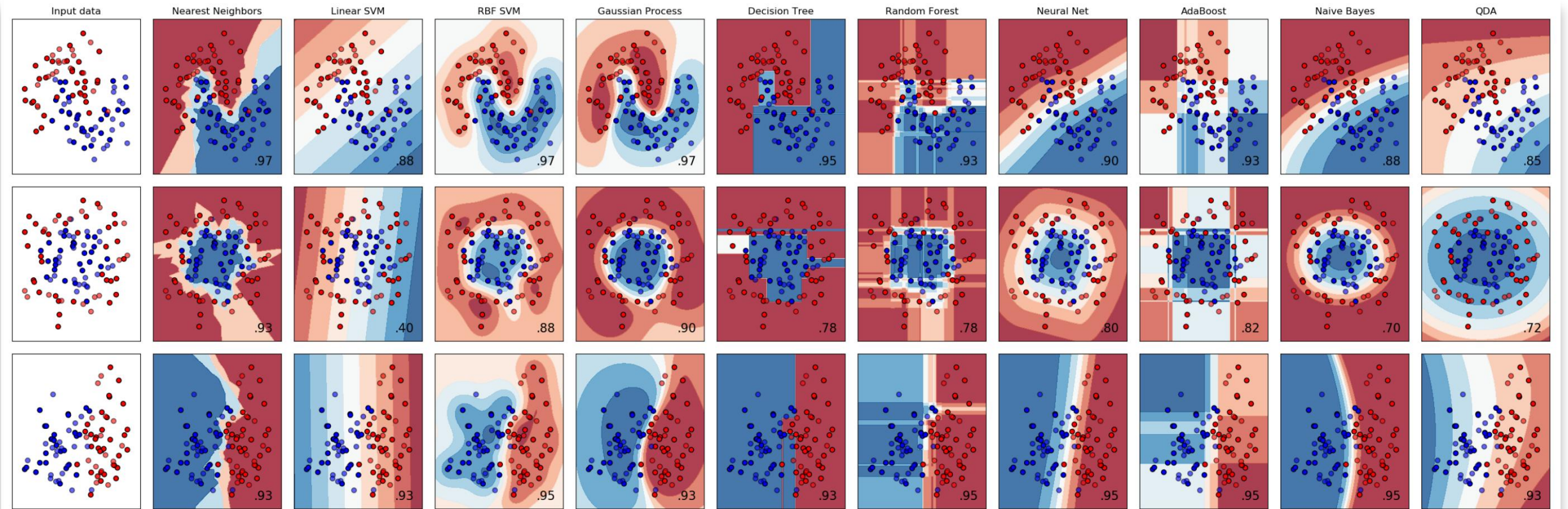
	SHERLOCK	BRIDGES ON FIRE	AVENGERS: ENDGAME	JOURNALS	WALKING DEAD
User 1	2			4	5
User 2	5		4		1
User 3			5		2
User 4		1		5	4
User 5			4		2
User 6	4	5		1	
sim(i,j)	-1	-1	0.86	1	NA



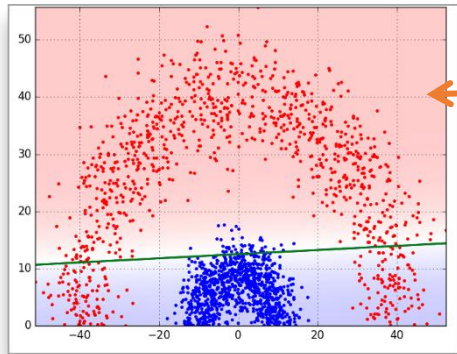
\*source: Fimarkets



- Depending on the classification algorithm, boundaries between classes can be of straight lines (hyperplanes, actually), smooth curves, rules-based lines, uncertain classification in lower probability zones, and anything in-between.



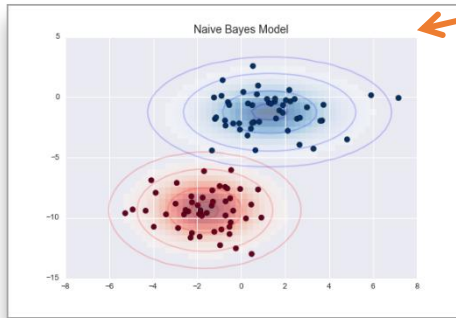
Classification models for **game data** must consider the density of the player population as they progress through the user journey.



\*source: mlxtend

### Logistic Regression

Assumes data to be highly correlated and linearly separable. Works best for few categories. Very fast to train and experiment with. **Early game data has potentially less dimensions to train in.**

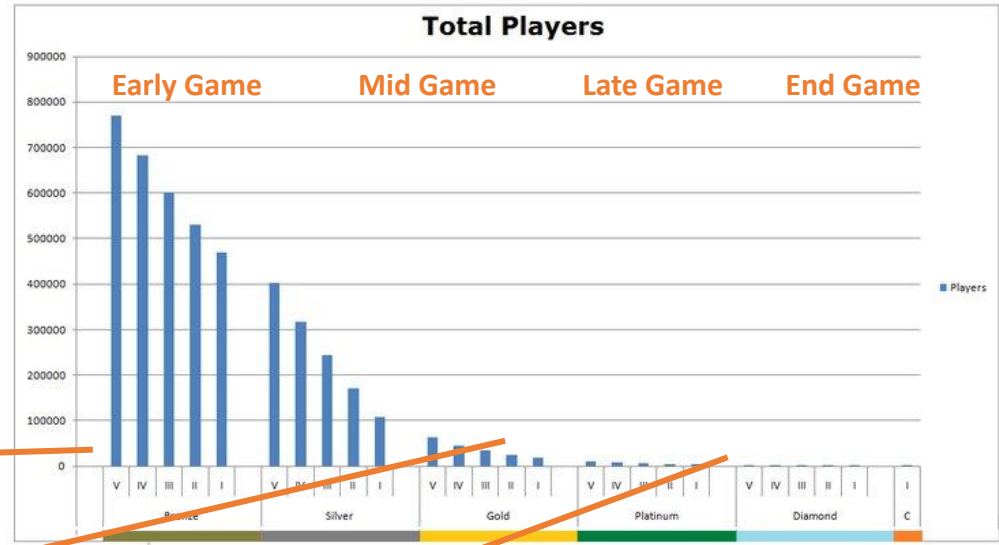


\*source: Karim O. Elish

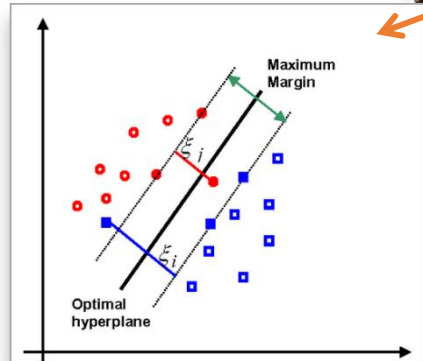
### Gaussian Naive Bayes

Learns the probability of a classification based on Bayes. Very fast to train and experiment with. May work well on relatively less training data. **Mid to late game has less players.**

Tier	Division	Players	Percentile
Bronze	V	770000	100
	IV	683380	88.75
	III	601639	78.13
	II	531362	69
	I	469530	60.97
Silver	V	403025	52.34
	IV	316536	41.1
	III	243229	31.58
	II	171626	22.28
	I	107825	14
Gold	V	62686	8.14
	IV	45090	5.85
	III	34601	4.49
	II	25333	3.29
	I	17601	2.28
Platinum	V	11095	1.44
	IV	7509	0.96
	III	5864	0.76
	II	4350	0.56
	I	2424	0.43
Diamond	V	2149	0.28
	IV	1358	0.18
	III	1076	0.14
	II	684	0.09
	I	462	0.06
Champion	I	50	0.006



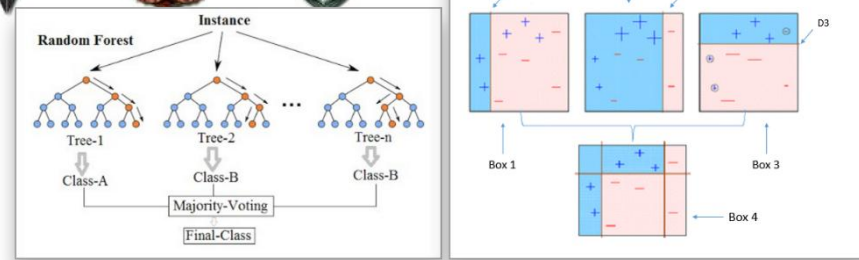
\*source: William Koehrsen, Manish Pathak



\*source: Ganapathi Pulipaka

### Support Vector Machines

Learns by trying to find boundaries that maximize the spatial distance from data points. Slow to train and should only be used when you *don't* have too much data (less than 100 k data points). **Late and end game has much less players, 1% to 5%**



### Decision Trees, Random Forests, "Boosted Forests" (XGBoost/AdaBoost/MART)

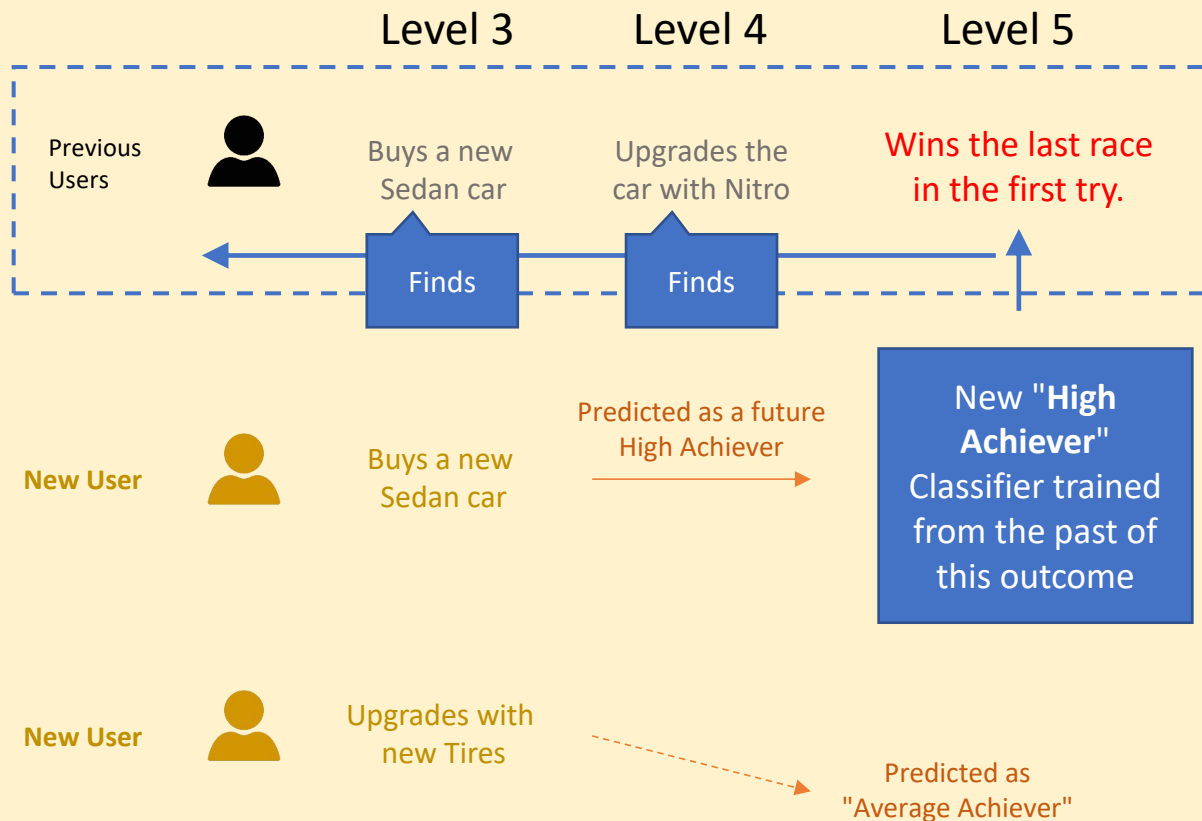
Learns by finding rules in the data that establish a if->else branching trees that classifies a data point based on its features. We can use **Entropy** to approximate the importance of features by information gain. "Boosted forests" can be more powerful but harder to tune.

**They "think" a lot like game designers but can go much deeper in sequences of conditions. May "just work".**



## How Classification help Game Designers

Useful to predict how a player will behave in the future based on behavior of other players in the past.



Once the classifier is trained and tuned, it outputs a list of the most relevant and correlated mechanics in the game that designers can adjust.

Something like:

Feature	Importance
Sedan_1_Bought_Level_3	0.73
Nitro_2_Level_4	0.64
Sedan_1_Bought_Level_2	0.52
Gas_Consumed	0.41
Events_Completed	0.38
Nitro_1_Level_4	0.04
Tire_2_Level_4	0.03



Certain types of classifications are particularly useful:

- Players who will **churn**.
- Players who will **convert**.
- Players who will participate in an upcoming event.
- Players who will play a game mode unlocked later in the game.
- Players who will engage in social modes like Guilds.



## How **Classification** help Live Ops

With the proper backend implemented, the classifier can be implemented as a real-time system to deliver custom content like gifts and in-game events.



Unique Race now available



Special Offer of a new Car



Share Free Fuel for 1 Day

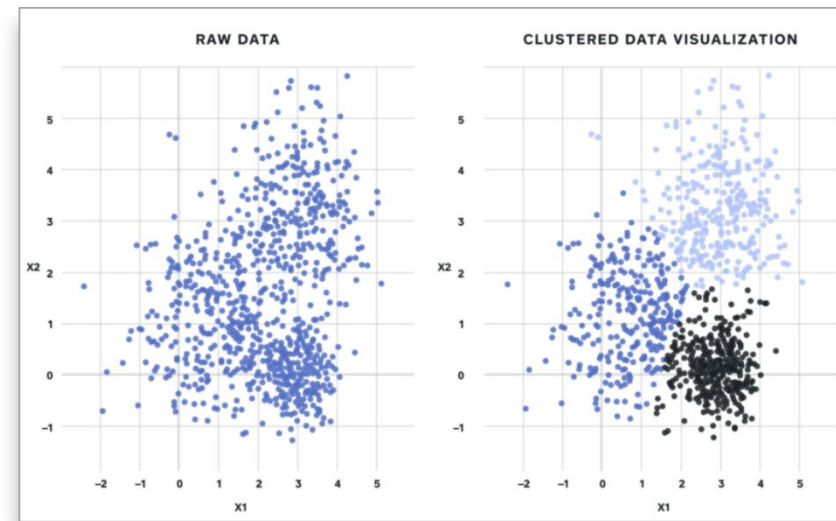
# Clustering

- The goal is to autonomously find clusters of data points based on its spatial proximity or feature similarity.
- Because they can "discover clusters by their own", they are *Unsupervised Learning*.
- Discovering clusters algorithmically eliminate the reliance on biased human judgement that may not be supported by data (like the **BARTLE taxonomy**)

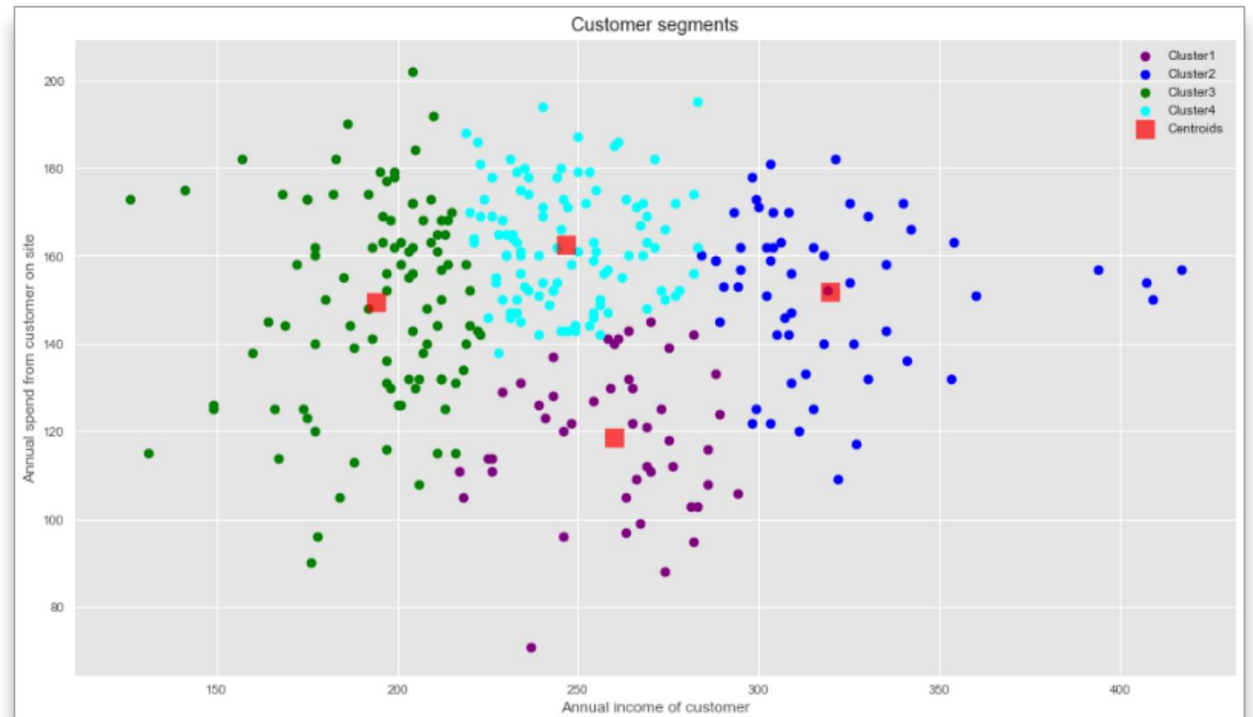
"Intuitive" Categorization		
<i>(depends who you ask, will be different...)</i>		
Group	Amount Spent	Income
1	Up to \$10	Low
2	Up to \$50	Low-Medium
3	Up to \$100	Medium-High
Fans	>\$100	Medium-High
Super Fans	>\$200	High
Whales	>\$500	High



...but...

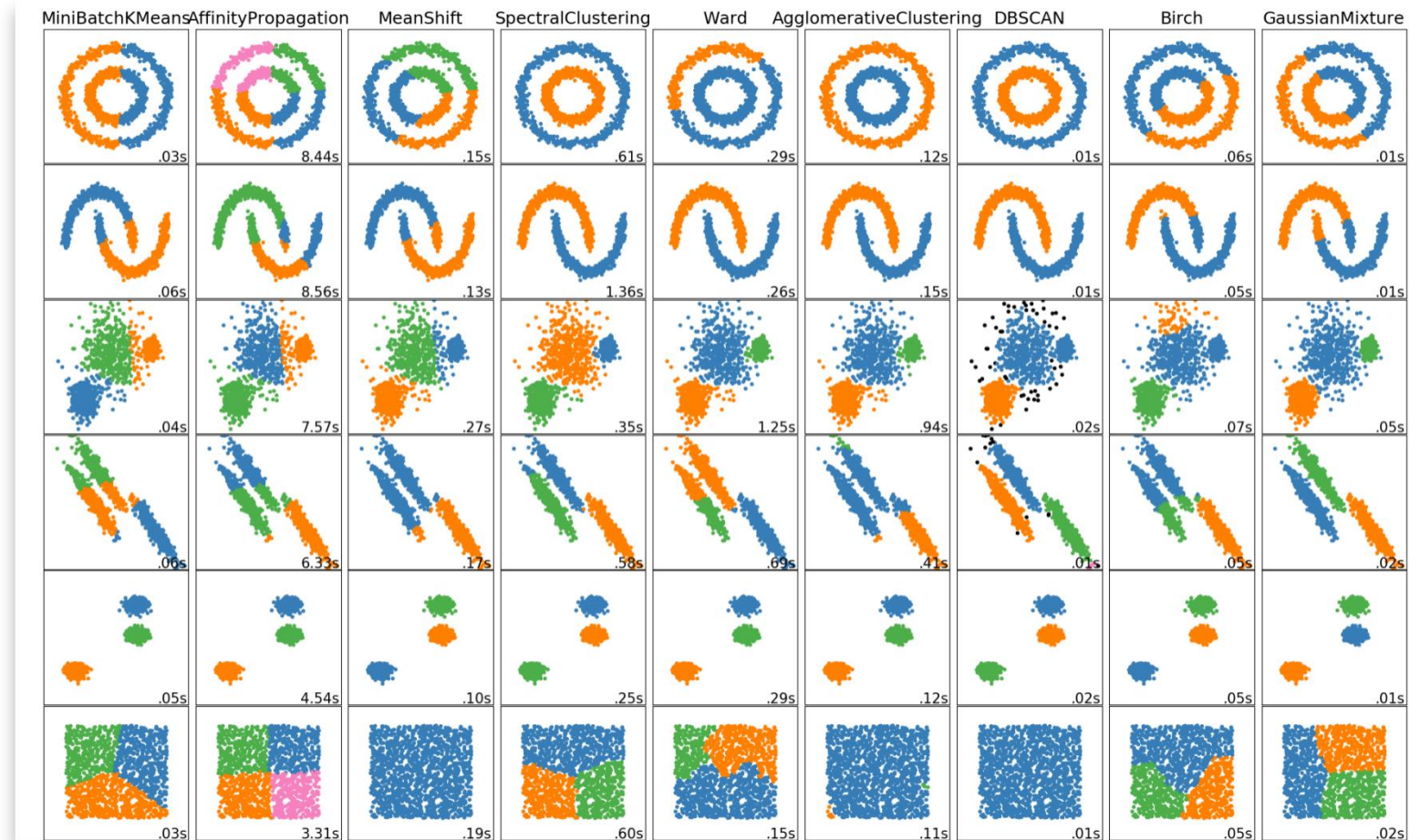


\*source: Inna Kaler



\*source: Sowmya Vivek

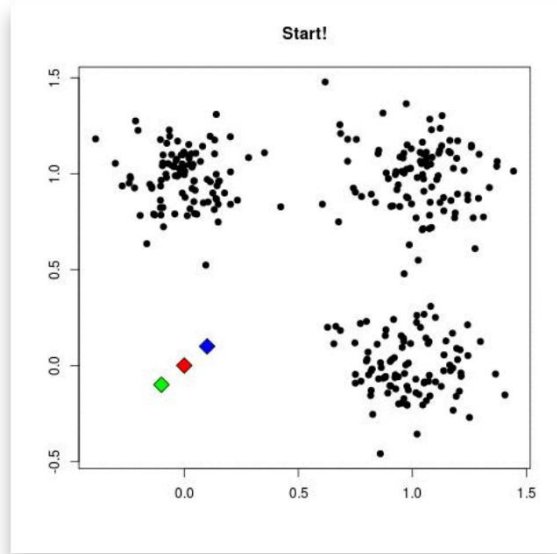
- Different algorithms employ different strategies to determine what is a "close point" in multidimensional space. The problem is harder than it seems in real-world messy datasets.



\*source: Scikit-Learn



Dimensionality may render Euclidean-based approaches less and less useful.

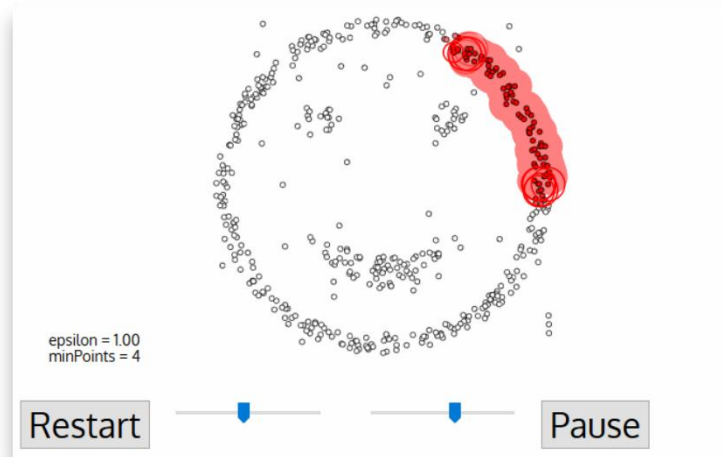


*\*source: Mubaris NK*

## k-Means

The most used. Fast and scalable. Creates clusters with distance of data points, moving the centroid until convergence. Needs to be told the number of clusters - which could be too arbitrary and needs the "elbow rule".

Data within a narrow progression bracket of the game (example: XP levels 10 to 15) may work as they are closer to hyperspheres, but game-wide, k-means clustering can suffer from misclassification of behaviors.

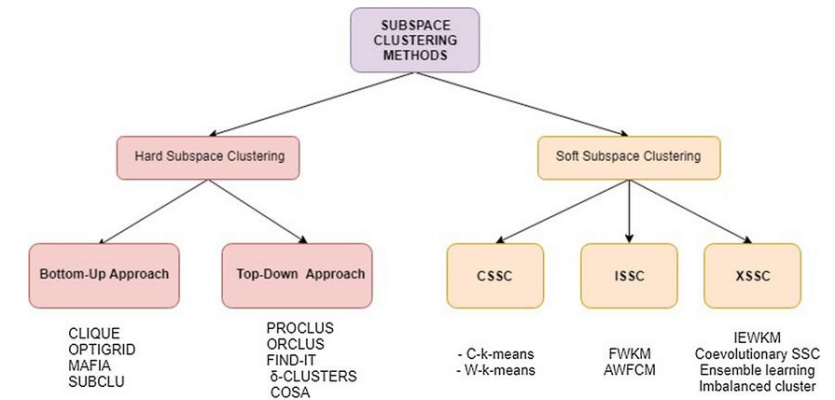


*\*source: George Seif*

## HDBSCAN

Density-based search of clusters, works by finding points that are close to a random non-visited starting point, and continuing to include other close points. Works great but is resource-intensive and vulnerable to clusters of different densities.

Produces good results but is difficult to productionalize due to resource utilization. Better for ad-hocs.



## SUBCLU

An implementation of DBSCAN for high dimensional spaces, which uses sub-space clustering to reduce the issues of the curse of dimensionality with Euclidean distances. Subspace clustering without assuming all of the clusters in a dataset are found in the same set of dimensions.

Lack of implementation in the most-known ML packages like SkLearn.





# How Clustering help Game Designers

## Development

	<h3>Killers</h3> <p><b>Defined by:</b> A focus on winning, rank, and direct peer-to-peer competition.</p> <p><b>Engaged by:</b> Leaderboards, Ranks</p>		<h3>Achievers</h3> <p><b>Defined by:</b> A focus on attaining status and achieving preset goals quickly and/or completely.</p> <p><b>Engaged by:</b> Achievements</p>
	<h3>Socialites</h3> <p><b>Defined by:</b> A focus on socializing and a drive to develop a network of friends and contacts.</p> <p><b>Engaged by:</b> Newsfeeds, Friends Lists, Chat</p>		<h3>Explorers</h3> <p><b>Defined by:</b> A focus on exploring and a drive to discover the unknown.</p> <p><b>Engaged by:</b> Obfuscated Achievements</p>



## Updates

Study made on the MMO game *Tera*.

TABLE 2: INTERPRETED BEHAVIORAL CLUSTERS FOR *TERA*, LEVEL 32 BIN ONLY, K-MEANS. %P = %PLAYERS IN BIN.

Title	%P	Characteristics
Elite	5.78	Highest scores for all features except Mining and Plants which are the lowest in the game.
Stragglers	39.4	Lowest scores for all features, including deaths from monsters.
Average Joes	12.7	Better scores across all categories than Low Performers, 4 <sup>th</sup> ranked overall
Dependables	18.6	Average scores across all categories, high number of friends, 3 <sup>rd</sup> ranked overall, 2 <sup>nd</sup> rank in monster kills
Worker I	15.9	Similar to the Average Joes, but high Mining and Plants, and loot value 3 <sup>rd</sup> ranked.
Worker II	7.6	Similar to The Dependables, but highest Mining and Plants value in the game. 2 <sup>nd</sup> ranked overall. Loot 2 <sup>nd</sup> ranked.

TABLE 3: INTERPRETED BEHAVIORAL CLUSTERS FOR *TERA*, LEVEL 32 BIN ONLY, SIMM. %P = %PLAYERS IN BIN

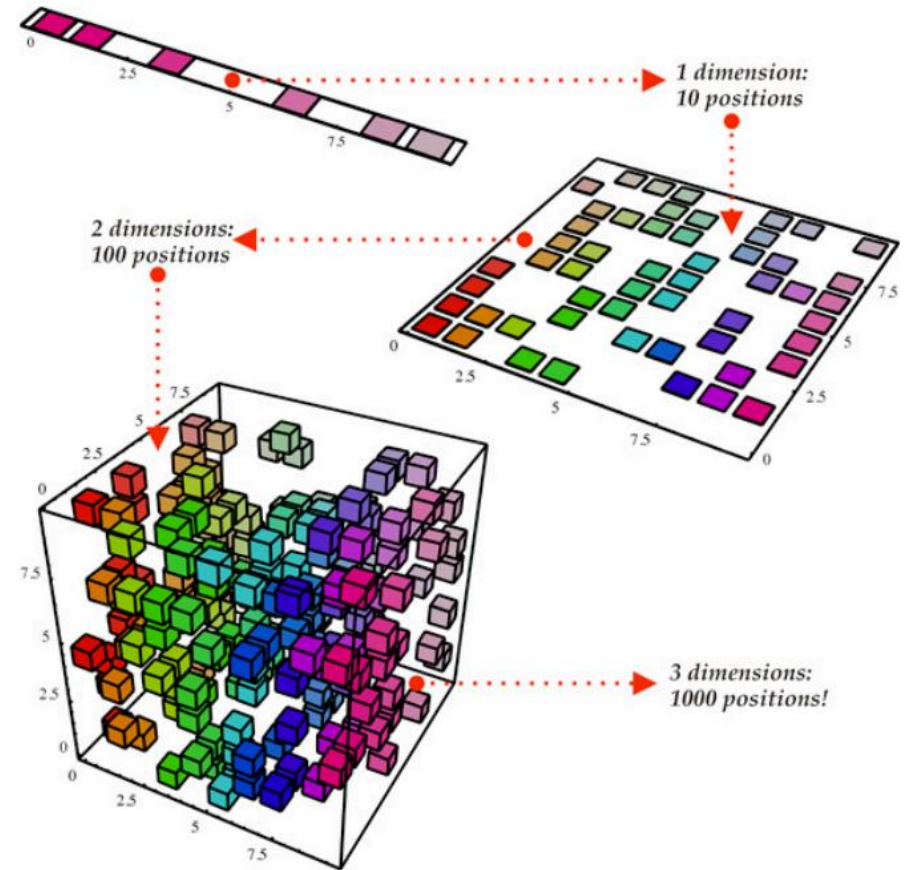
Title	%P	Characteristics
Elite	3.9	High scores overall, except for Mining/Plants and deaths from monsters. No auctions created.
Stragglers	7.6	Low scores overall, dies a lot from monsters
Planters	21.6	Middling scores, but high Plants skill
Miners	15.0	Middling scores, but high Mining skill
Auction Devils	1.1	Highest auction and achievement score. 2 <sup>nd</sup> ranked loot and kills scores. 2 <sup>nd</sup> ranked friends score, high mining score
Friendly Pros	50.8	Highest friends score, scores similar to Auction Devils apart from low auction score and 2 <sup>nd</sup> lowest loot score

- Clustering is useful to find groups of players from their behavior in the game. Frameworks of psychology like the Bartle Types are useful for conception of features, but once you launch data science can provide you how your players really behave.

\*source: Anders Drachen, Rafet Sifa, Christian Bauckhage and Christian Thurau

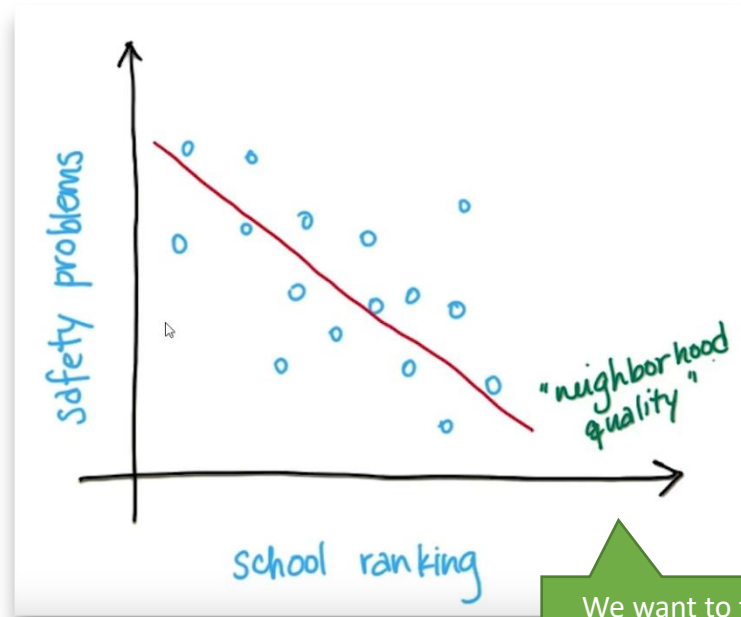
# Dimensionality Reduction

- A group of algorithms that can decompose multiple dimensions of data into fewer ones.
- The decomposed dimensions represent "parts" of data that were highly aligned with each other.
- The mathematical description of these new dimensions may seem clunky, but are interpretable by a data analyst.
- Very useful for metagame analysis.

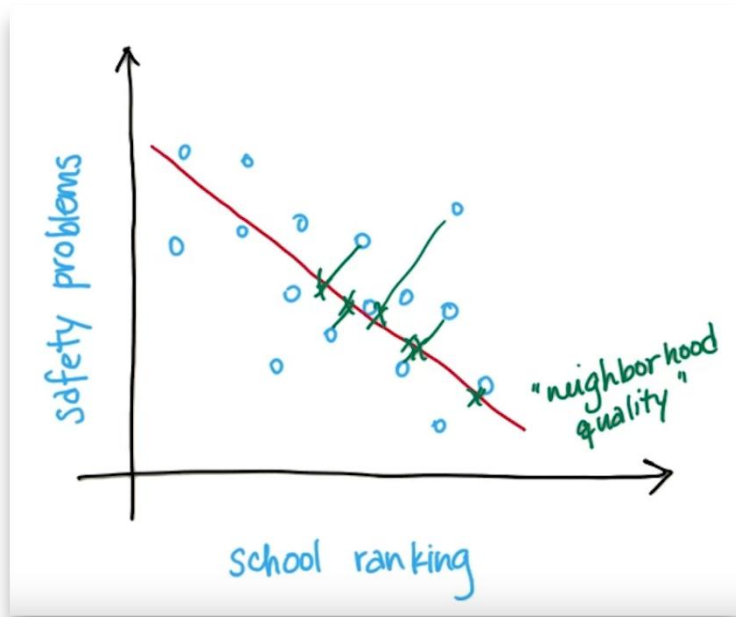


*\*source: BigSnarf blog*

- The most used method is the **Principal Component Analysis (PCA)**.
- It works by finding new coordinate systems in the data that can potentially describe latent features, and then projecting 2 or more dimensions into a single one, losing information in the process.
- For example:



We want to find this latent feature hidden in 2 dimensions.



- Also great to visualize multi-dimensional spaces in ways anyone can understand.
- Example: here's an analysis [made by Jose A. Dianes](#) on cases of Tuberculosis across 2 decades and all countries:

##		PC1	PC2
##	Afghanistan	-3.490274	0.973495650
##	Albania	2.929002	0.012141345
##	Algeria	2.719073	-0.184591877
##	American Samoa	3.437263	0.005609367
##	Andorra	3.173621	0.033839606
##	Angola	-4.695625	1.398306461

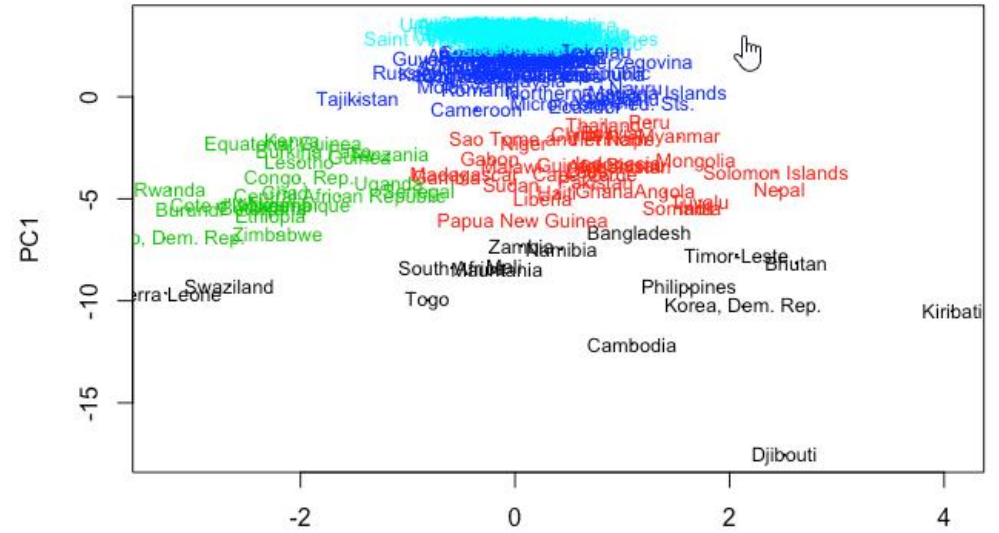
Compressing all years to 2  
Principal Components

year	1990	1991	1992	1993	1994	1995	1996	1997	1998	1999	2000	2001	2002
country													
Afghanistan	436	429	422	415	407	397	397	387	374	373	346	326	304
Albania	42	40	41	42	42	43	42	44	43	42	40	34	32
Algeria	45	44	44	43	43	42	43	44	45	46	48	49	50
American Samoa	42	14	4	18	17	22	0	25	12	8	8	6	5
Andorra	39	37	35	33	32	30	28	23	24	22	20	20	21

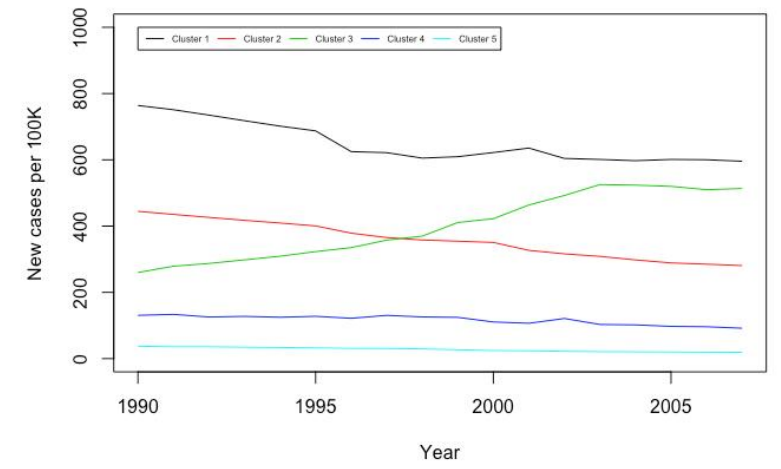


Plotting and  
Clustering

Existing TB cases per 100K distribution

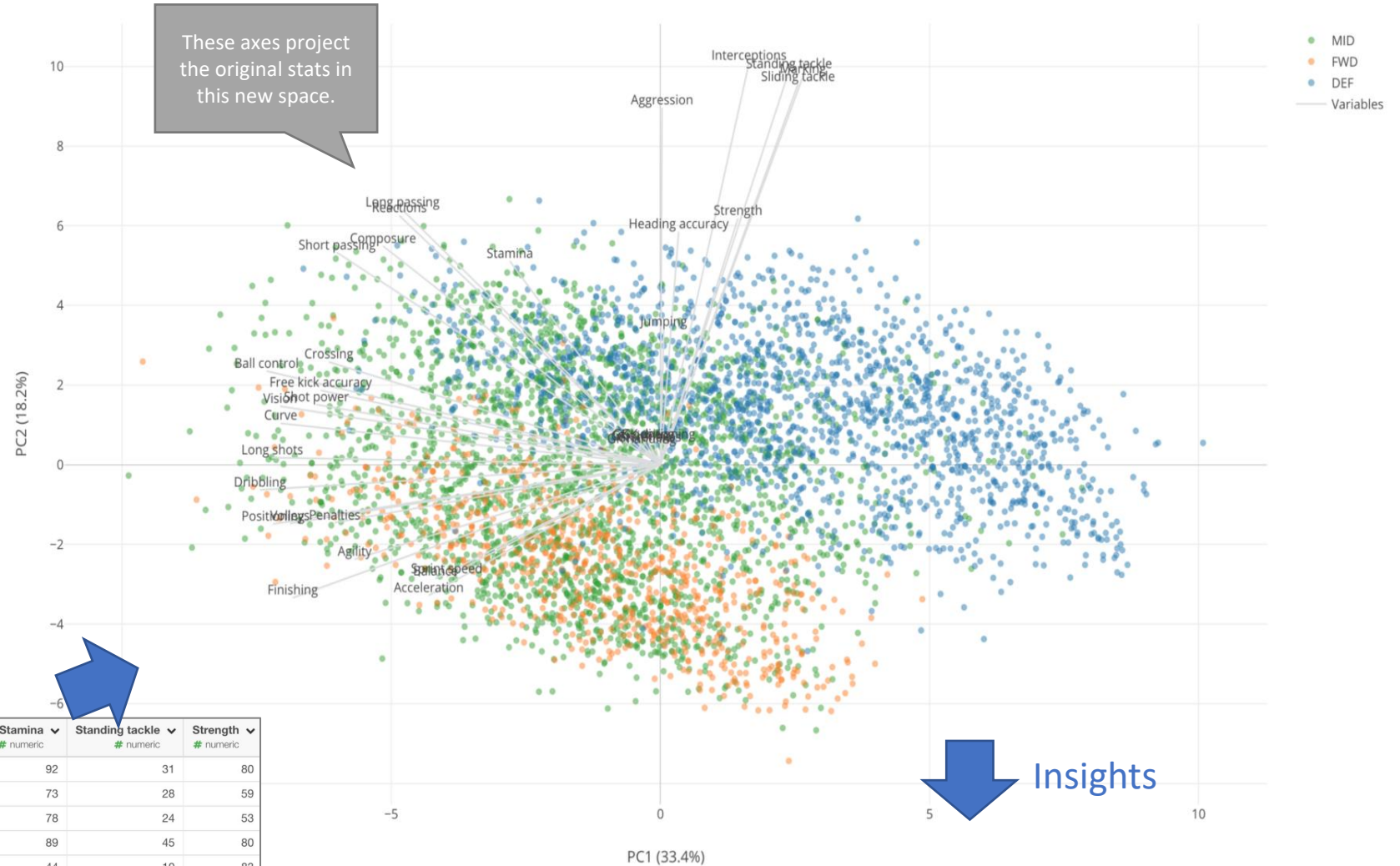


Analysis of what the  
clusters represent.



## How PCAs help Game Designers

- Like in the example before, where the clusters captured the trend over time for several countries at once, PCAs can be used to find latent information and trends in your data
- Here's an example decomposing 34 stats of soccer players in 2018 into only 2 dimensions, [made by Kan Nishida](#)



## Insights

1. The dimension PC2 is strongly associated to stats that help Defense, like Marking, Interception, Strength, Aggregation.

2. Mid Fields cluster more towards:

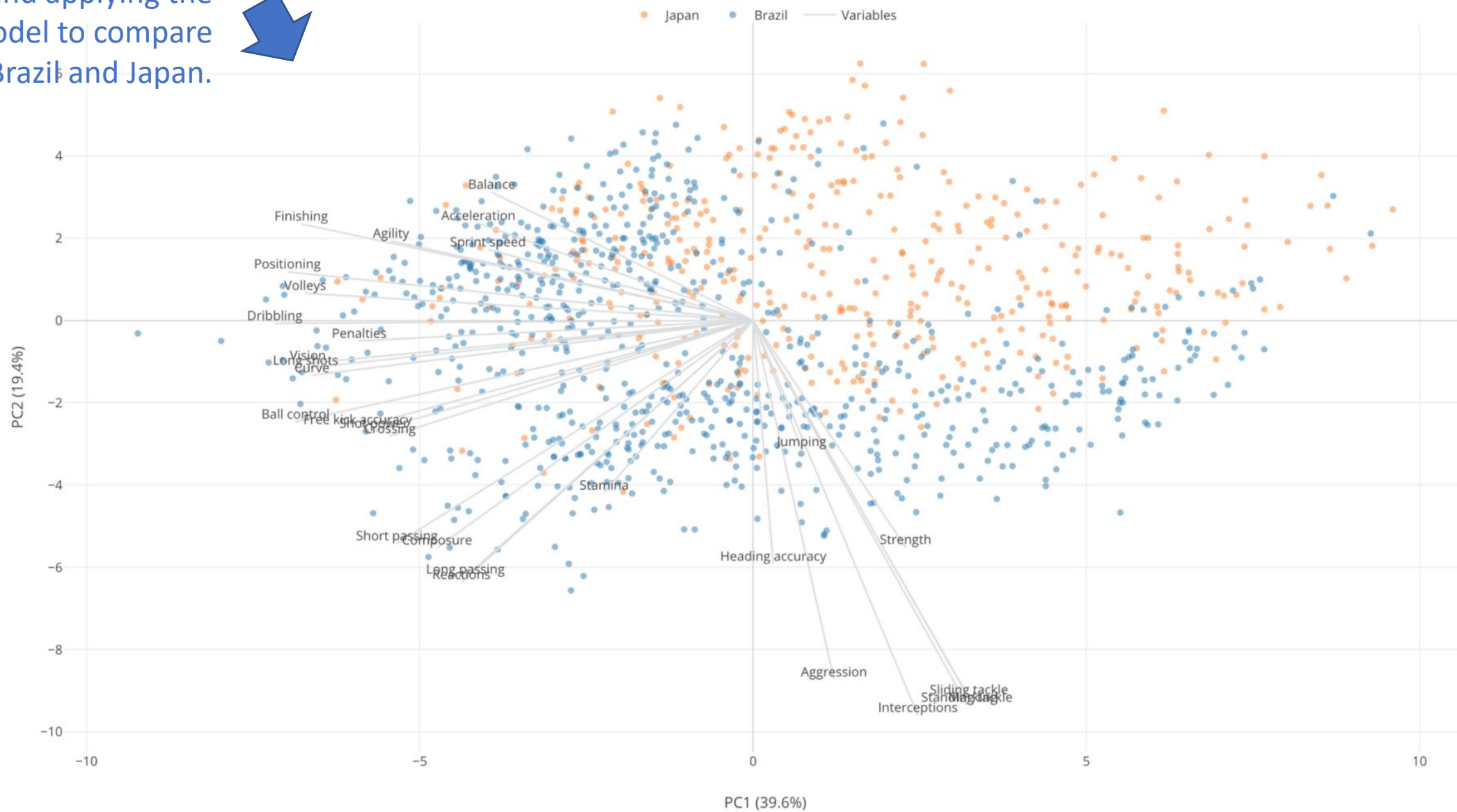
- Short Pass
- Stamina
- Crossing
- Free Kick Accuracy
- Vision
- Shot Power

3. Forward players share much of the same skill space in PC1 with Mid Fields, but cluster more towards:

- Agility
- Acceleration
- Sprint Speed
- Finishing.

Penalties	Positioning	Reactions	Short passing	Shot power	Sliding tackle	Sprint speed	Stamina	Standing tackle	Strength
# numeric	# numeric	# numeric	# numeric	# numeric	# numeric	# numeric	# numeric	# numeric	# numeric
85	95	96	83	94	23	91	92	31	80
74	93	95	88	85	26	87	73	28	59
81	90	88	81	80	33	90	78	24	53
85	92	93	83	87	38	77	89	45	80
47	12	85	55	25	11	61	44	10	83
81	91	91	83	88	19	83	79	42	84
40	12	88	50	31	13	58	40	21	64
86	85	85	86	79	22	87	79	27	65
73	79	86	90	87	69	52	77	82	74
70	92	88	75	88	18	80	72	22	85
68	52	85	78	79	91	77	84	89	81
77	84	88	90	85	40	75	87	51	73
27	13	81	32	36	16	52	38	18	70
77	86	87	81	84	35	84	85	39	72
80	79	88	92	73	73	71	82	80	58
76	86	87	86	91	52	95	76	55	80

... and applying the model to compare only Brazil and Japan.



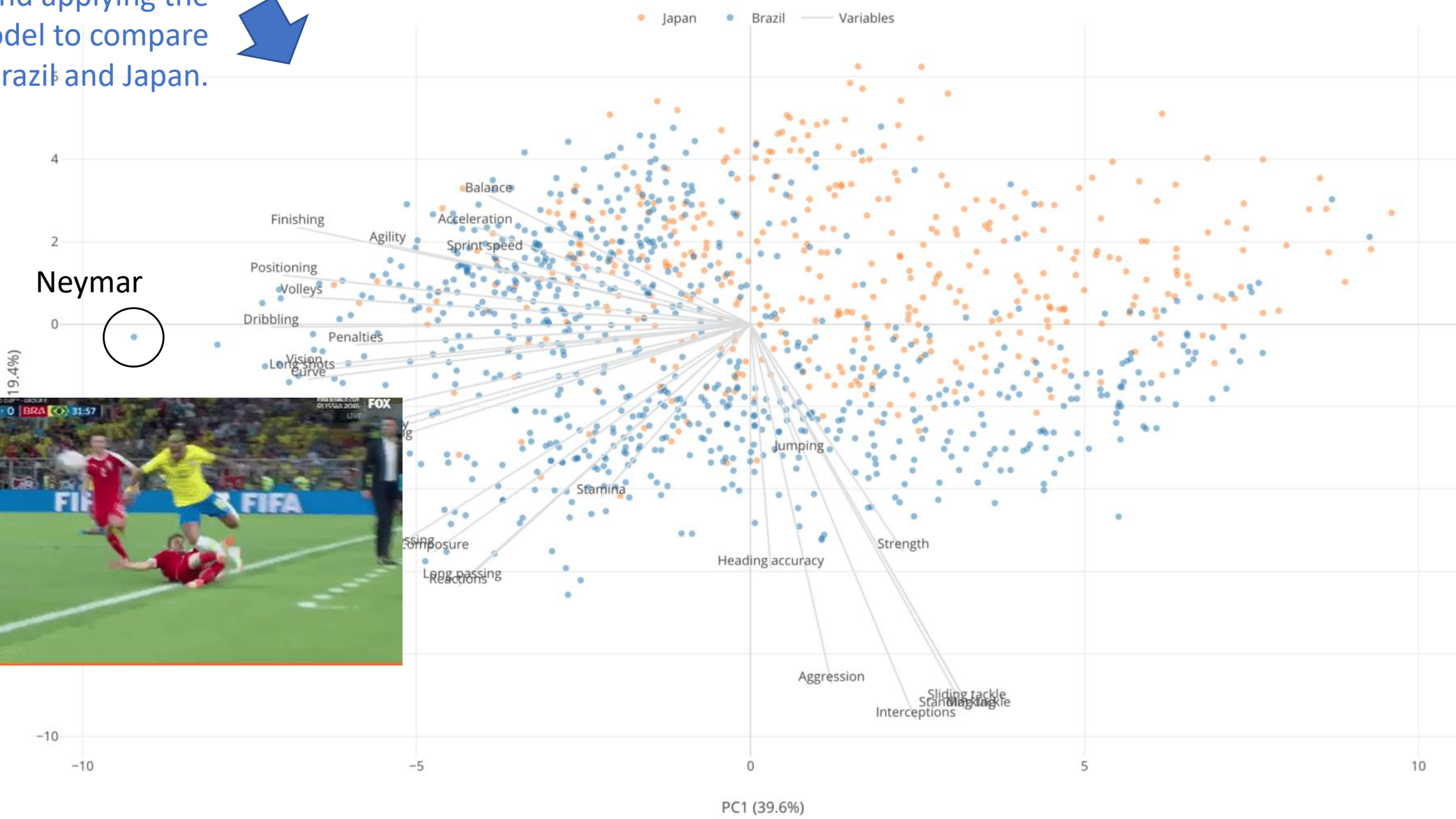
... and applying the model to compare only Brazil and Japan.



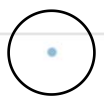
Neymar



... and applying the model to compare only Brazil and Japan.



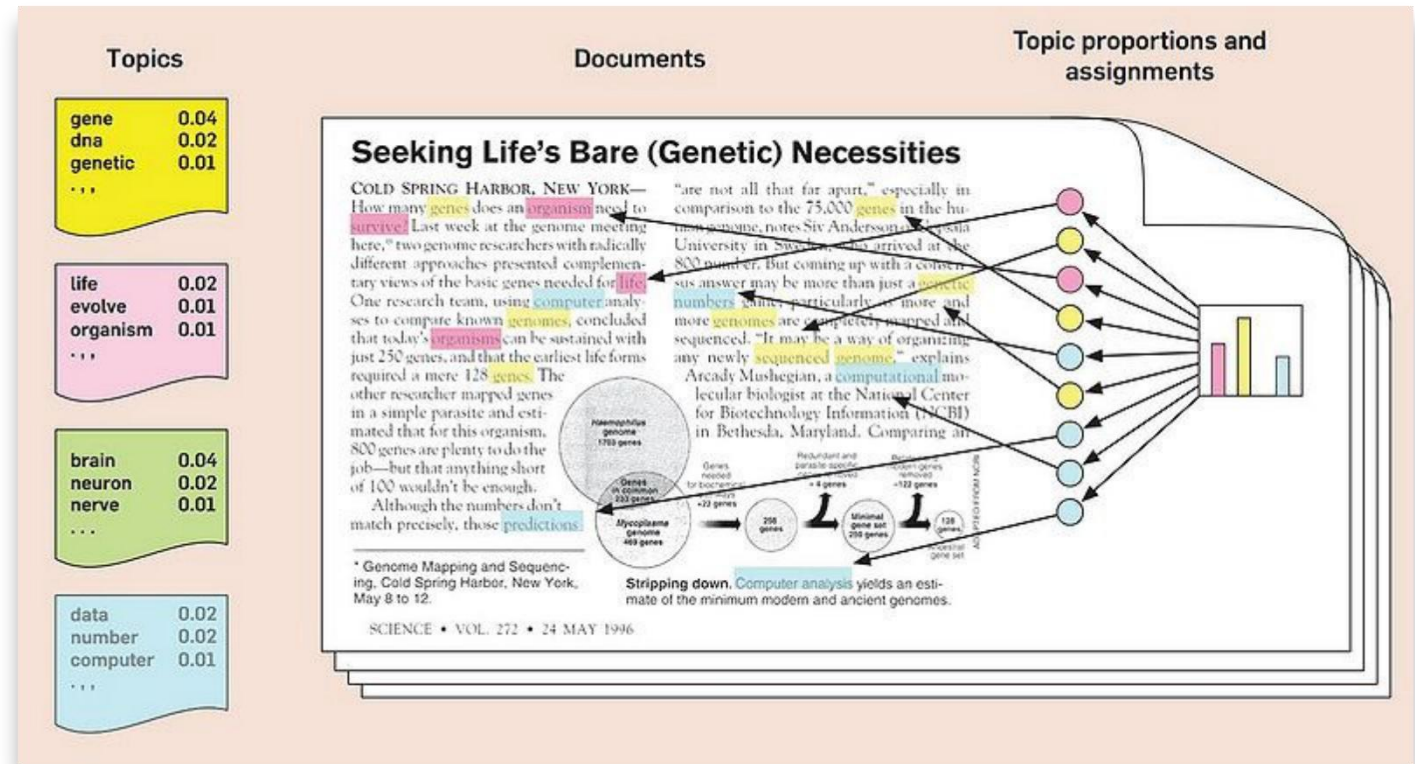
Neymar





# Latent Dirichlet Allocation (LDA)

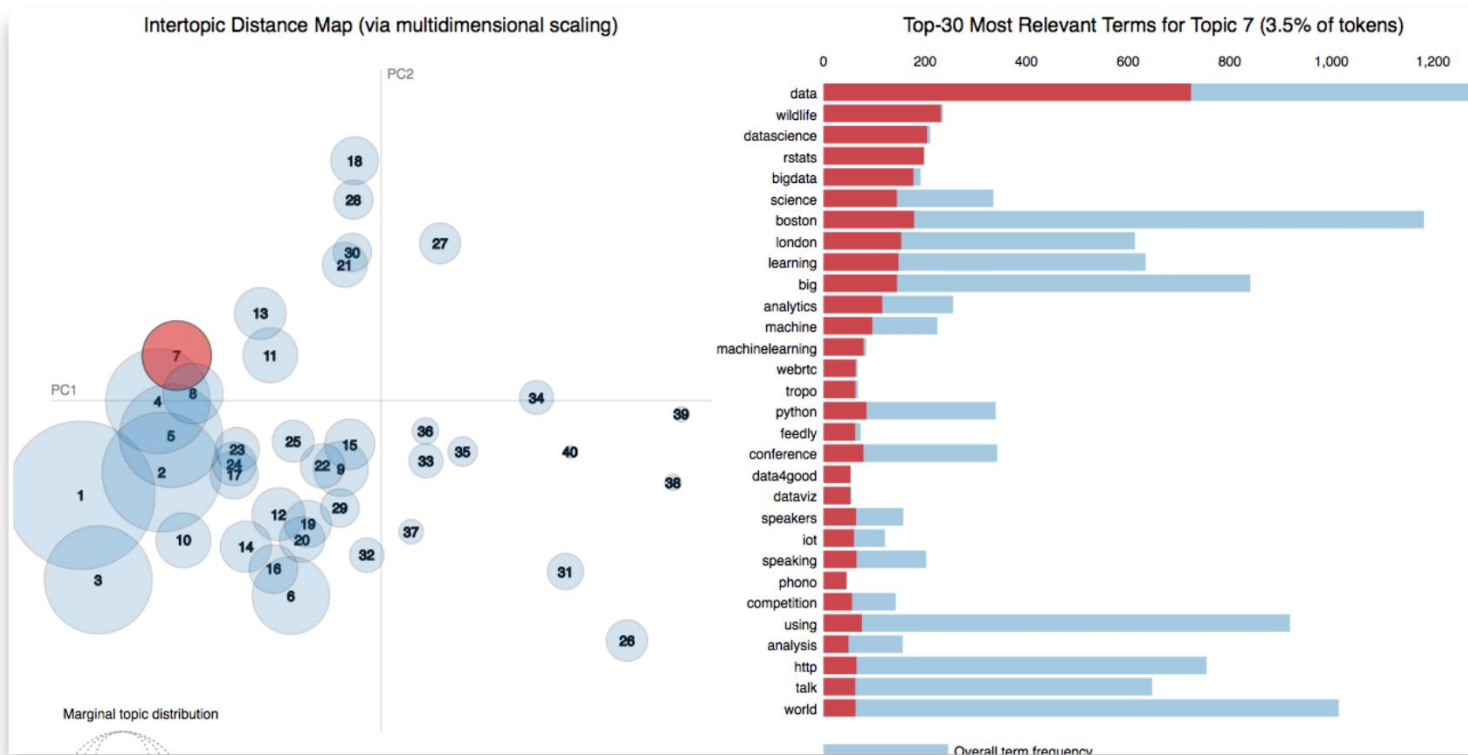
- A *semi-supervised* learning algorithm, used to predict topics and classify data points based on the repetition of a certain content across those data points.
- It's a "soft" classification method, in the sense that a single document can belong to 2 or more topic.
- Hard to train and tune, but can be work for analysis of multi-dimensional strategies.



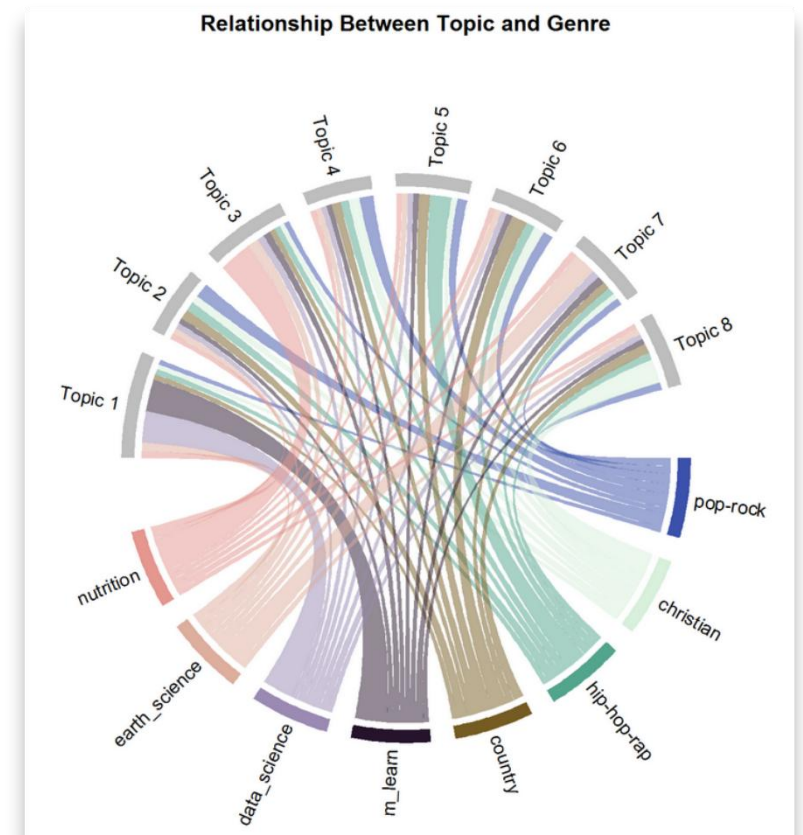
\*source: BigSnarf blog

- LDA Topic Modelling is used to discover broader topics of documents of any type, (include scientific data, like genetic measurements) and relationships between such topics.
- But was also used for content recommendation and analysis of overlapping communities in social networks.

### LDA on Twitter messages



\*source: Alexis Perrier



\*source: Debbie Liske



## How LDA Topic Modelling help Game Designers

- By revealing and structuring metagame patterns and strategies that players are using.
- Here's an [analysis from Hlynur Davíð Hlynsson](#) using LDA to discover and relate deck strategies in Magic: The Gathering.

Since LDA is a generative model, we can also make this model create new decks!

The model discover Archetypes of decks and the probability that a certain card will be present in each deck Archetype.

Archetype 27	Archetype 26
0.054 Noble Hierarch	0.053 Urza's Mine
0.053 Inkmoth Nexus	0.053 Expedition Map
0.053 Glistener Elf	0.053 Urza's Tower
0.052 Vines of Vastwood	0.053 Urza's Power Plant
0.052 Mutagenic Growth	0.042 Walking Ballista
0.051 Might of Old Kroa	0.038 Thought-Knot Seer
0.049 Blighted Agent	0.036 Karn Liberated
0.043 Blossoming Defense	0.036 Relic of Progenitus
0.041 Become Immense	0.034 Chalice of the Void
0.040 Groundswell	0.034 Matter Reshaper
0.036 Windswept Heath	0.034 Eldrazi Temple
0.030 Verdant Catacombs	0.034 Reality Smasher
0.029 Spell Pierce	0.028 Ghost Quarter
0.027 Misty Rainforest	0.025 Warping Wail
0.026 Breeding Pool	0.021 Dismember
0.026 Forest	0.021 Wurmcoil Engine

Affinity for AI	
4 Blinkmoth Nexus	2 Master of Etherium
4 Darksteel Citadel	3 Spire of Industry
4 Mox Opal	1 Memnite
4 Signal Pest	3 Ghirapur Aether Grid
4 Inkmoth Nexus	2 Ancient Grudge
4 Arcbound Ravager	3 Glimmervoid
4 Cranial Plating	2 Grafdigger's Cage
4 Ornithopter	2 Spell Pierce
4 Springleaf Drum	1 Whipflare
4 Vault Skirge	1 Rest in Peace
4 Steel Overseer	2 Relic of Progenitus
4 Etched Champion	1 Vandalblast
3 Galvanic Blast	1 Gemstone Caverns

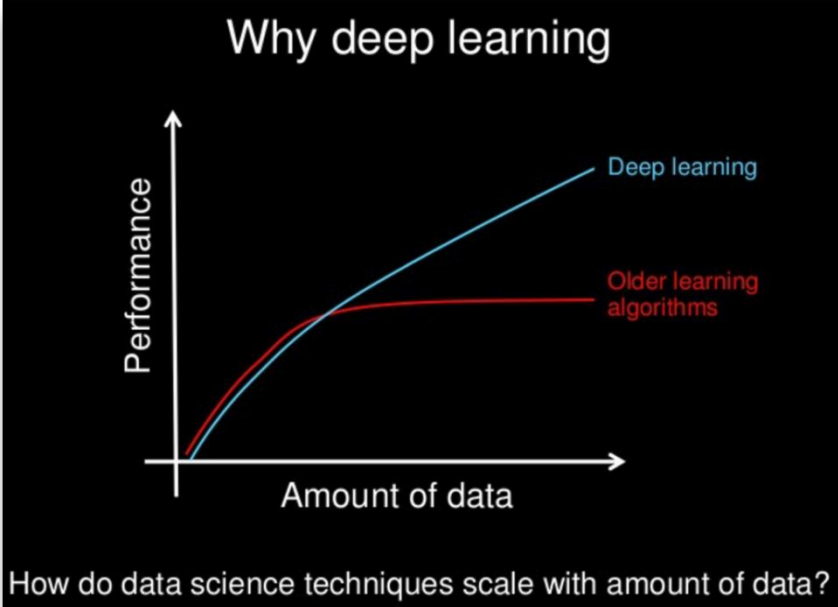
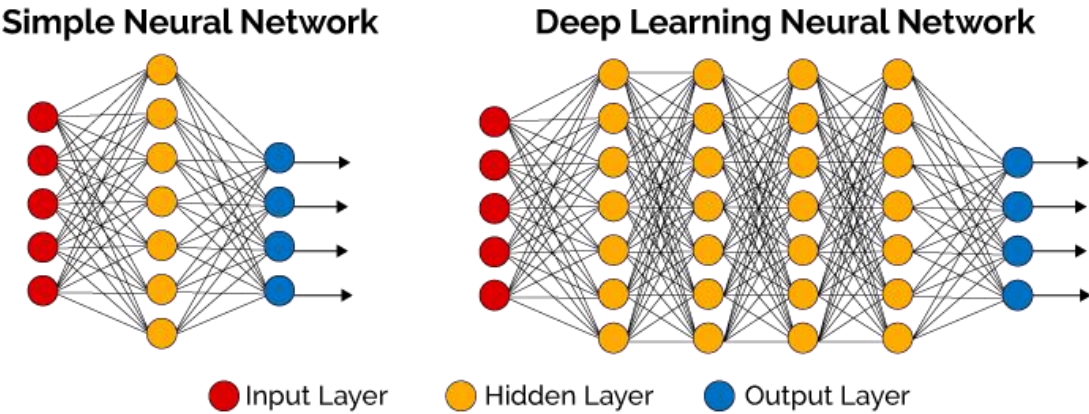
VisualDecklist.com

# Reinforcement Learning

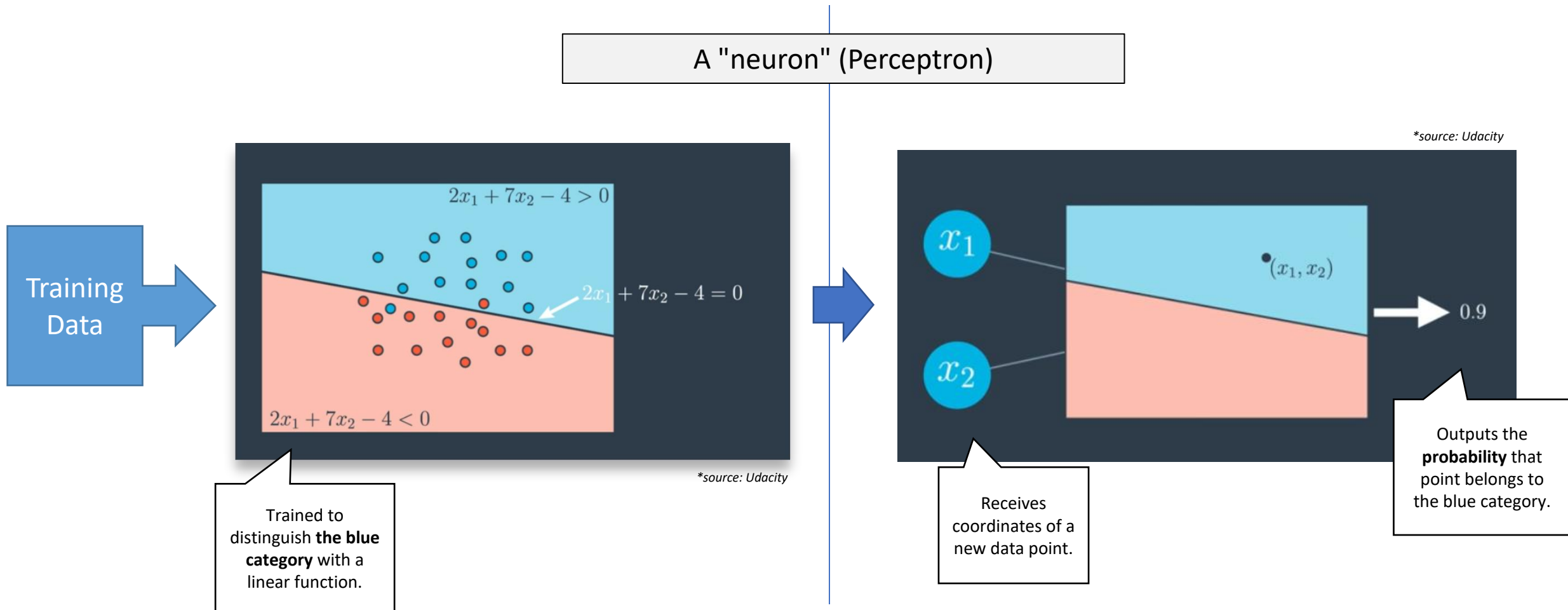
RL, Deep RL, and Deep Q-Networks can help gamedev.

# Deep Learning

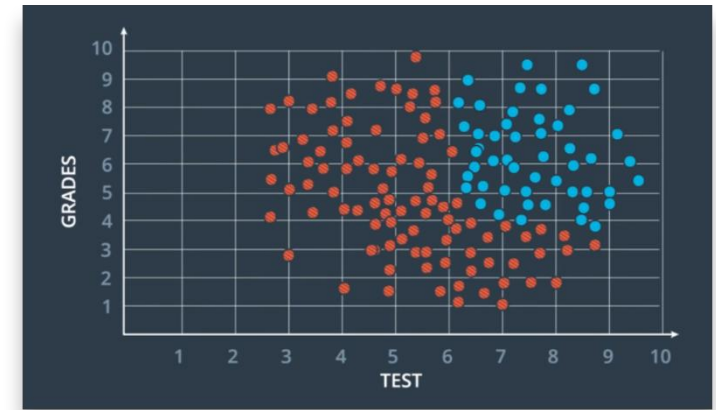
- A class of ML models composed of stacked Neural Networks.
- **Very flexible and very powerful**, and *continue to improve over time* as the amount of data increases.
- **But costly**: hard to tune, time-consuming, require GPUs, take a *lot* of processing time to train.
- Also, nearly impossible to know why and how a deep learning model makes decisions.



- First, let's understand what a Neural Network does. It's an architecture (usually used as a Classifier) that combines the separation made by various "neurons".

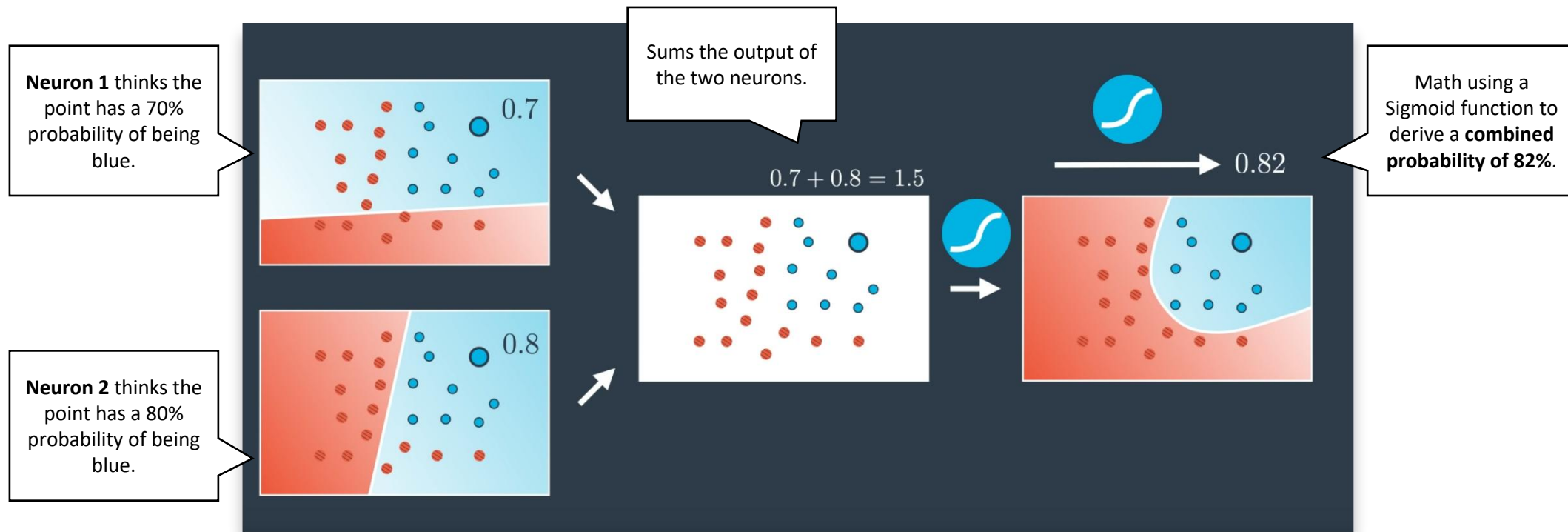


- However, in the real world, very few datasets are separable by one line. So then, like in the example below, we create a network of 2 neurons to create a more complex boundary.



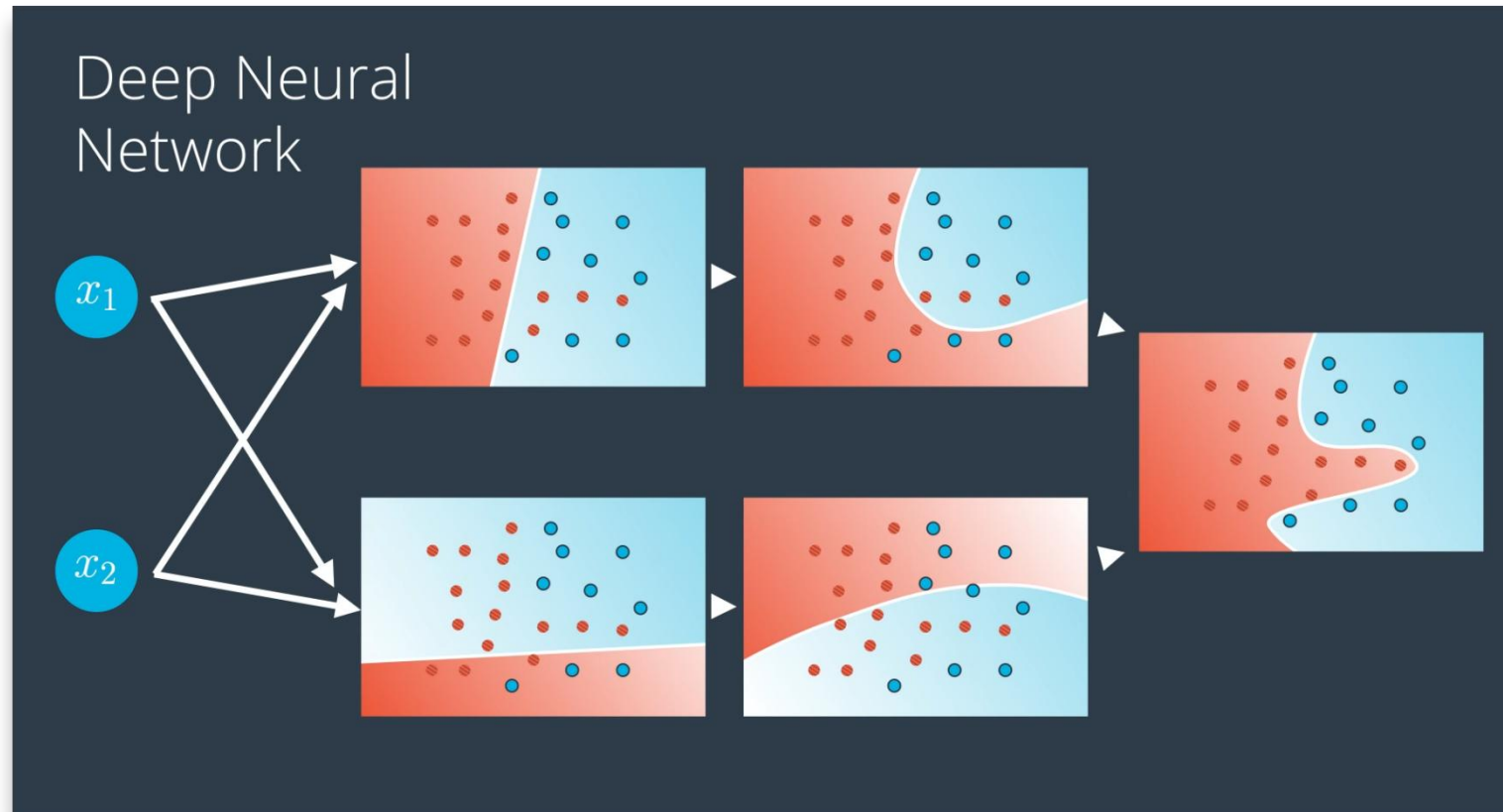
*\*source: Udacity*

### A Neural Network



*\*source: Udacity*

- Finally, for really complicated classifications, we can stack more layers of neurons that together create a very sophisticated classifier. The non-linear model of a layer is further expanded into more complex non-linear boundaries. **That's a deep neural network.**



*\*source: Udacity*

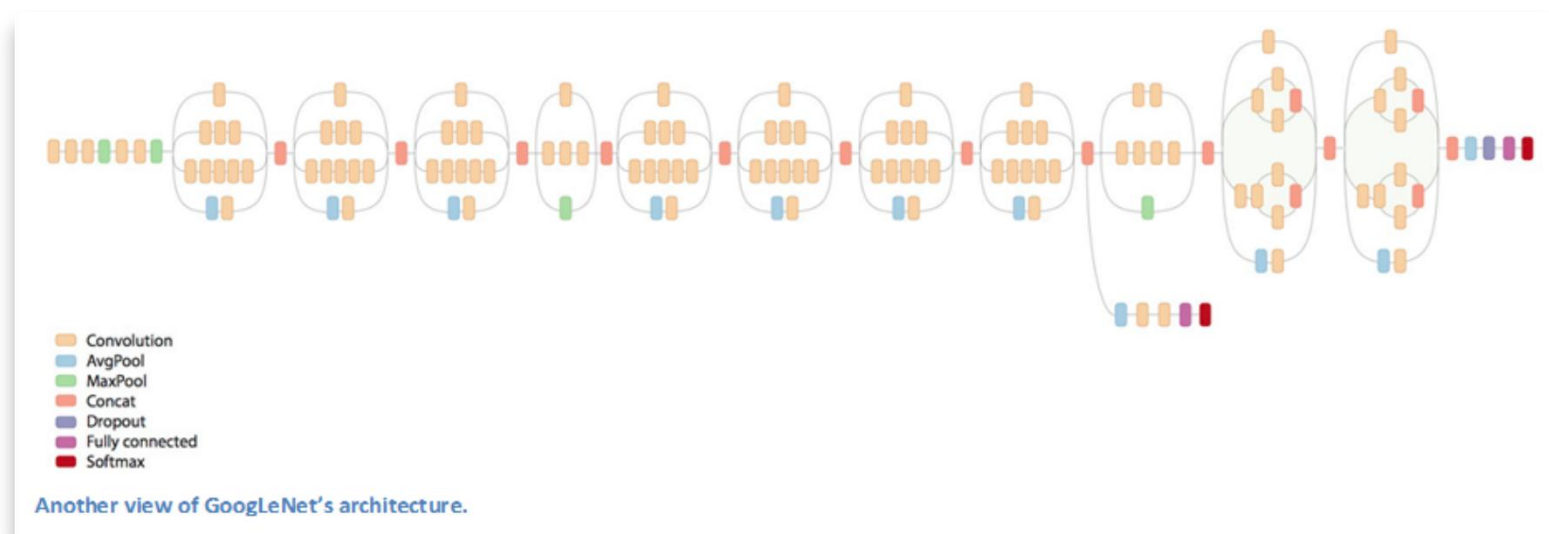


- The ability to stack more layers and experiment with different architectures when connecting neurons is what makes deep neural networks so flexible. And it can become really sophisticated.
- For example, Google's Inception models:



bell pepper, 95.5%

\*source: MathWorks



\*source: Google

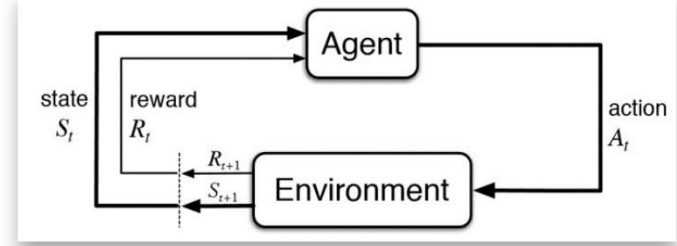


Using Inception\_V3 CNN  
 Prediction: Probability  
 bison: 93.83%  
 ox: 0.13%  
 ibex, Capra ibex: 0.05%  
 rhinoceros beetle: 0.05%  
 brown bear, bruin, Ursus arctos: 0.03%  
 ladybug, ladybeetle, lady beetle, ladybird, ladybird beetle: 0.03%  
 Leonberg: 0.03%  
 promontory, headland, head, foreland: 0.03%  
 cougar, puma, catamount, mountain lion, painter, panther, Felis concolor: 0.03%  
 wool, woolen, woollen: 0.03%

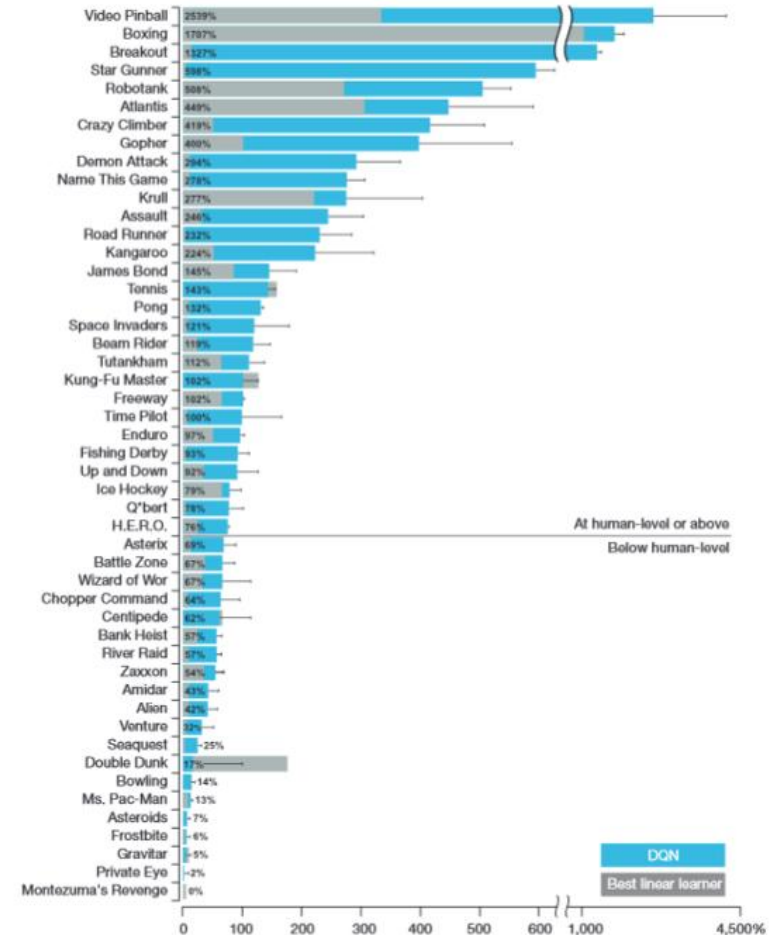
\*source: William Koehrsen

# Learning by Trial and Error

- Reinforcement Learning is the one type of Machine Learning that, *in principle*, doesn't require loads of data, but... doesn't do the same things either. :)
- The goal of Reinforcement models is to *learn how to execute tasks from scratch*, with minimal human intervention.
- It's a model based on Markov Decision Processes that learns on trial-and-error across thousands (millions?) of attempts, being guided by reward functions across many tries.
- RL agents that learn to play games by themselves. A team of *Mnih et al. 2015* trained agents in several different Atari games, frequently achieving super-human performance.

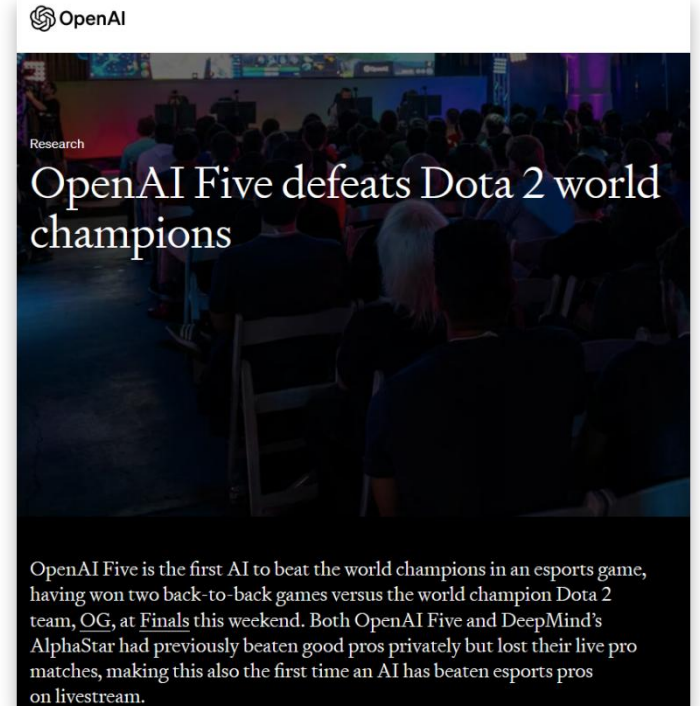


\*source: Shweta Bhatt



**Figure 3 | Comparison of the DQN agent with the best reinforcement learning methods<sup>19</sup> in the literature.** The performance of DQN is normalized with respect to a professional human games tester (that is, 100% level) and random play (that is, 0% level). Note that the normalized performance of DQN, expressed as a percentage, is calculated as:  $100 \times (\text{DQN score} - \text{random play score}) / (\text{human score} - \text{random play score})$ . It can be seen that DQN outperforms competing methods (also see Extended Data Table 2) in almost all the games, and performs at a level that is broadly comparable with or superior to a professional human games tester (that is, operationalized as a level of 75% or above) in the majority of games. Audio output was disabled for both human players and agents. Error bars indicate s.d. across the 30 evaluation episodes, starting with different initial conditions.

- But playing simple mechanics *is not good enough*.
- Most modern game have some level of **RPG and Strategy**, two genres of video games particularly challenging for RL:
  - High-dimensional state space;
  - Resource management with cascading consequences;
  - Control of many simultaneous features or units;
  - Planning long-term progression and “end-game”;
  - Time-locked features in free-to-play games;
  - Shifting metagame as games update often via Live Ops.
- Multi-agent RL (MARL), Hierarchical RL (HRL), Deep LSTM and proximal policy optimization (PPO) have been applied for more efficient learning of sub-tasks and long-term planning.
- OpenAI Five mastered Dota 2.
- [AlphaStar mastered Starcraft 2](#).



# RL in the real world of game development

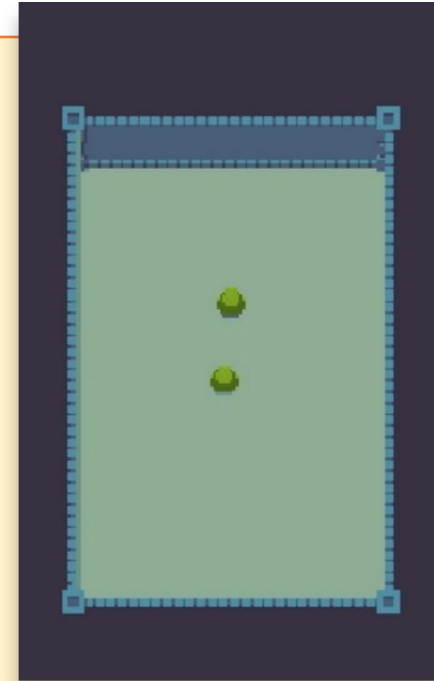
- *Low application or RL in game development:*
  - Deep Learning is **very costly**, both in terms of compute and data science talent.
  - On the talent front, the games industry has to compete with much higher-paying industries like Finance!
- Hard to tune. Game development requires a level of control in every feature.
- Reserchers of Deep RL are not usually trying to solve the industries' problems for ***developers***, just to benchmark their creation against the human skills of ***players***.
  - OpenAI Five and AlphaStar are impressive but useless for game *developers*.
- But can their science be used in tools to create and balance *new* games?
  - Games in development without any players?
  - Games with long-term economies that scales into *years* of gameplay?



## How Reinforcement Learning help Game Designers

Reinforcement learning models open a world of possibilities *in conjunction with **Deep Neural Networks:***

- Enemy AI that is *trained instead of coded*.
- QA bots that play the engine 24/7 and help test the work of developers and document bugs.
- Play the metagame and test long-term progression design and economy, *help balance the game economy and progression*. Predict resource inflation.
- Narrative agents without pre-scripted dialogues, that react to the player's choices more organically.
- Matchmaking for PvP games.



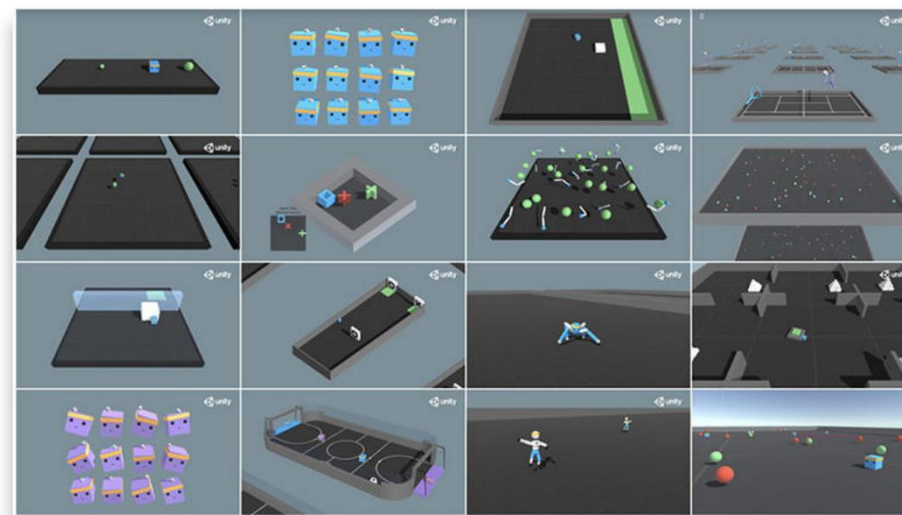
\*source: Unity



\*source: Unity

# RL for Enemy AI

- **Unity 3D** made a concerted effort to enable RL for game developers.
- Common example: enemies in an FPS (first person shooter)
- Some developers used it. Most *ditched* it.
  - Because it's very hard to tune
  - Requires a lot of time to get right.
- Didn't really help to design a game that is *more fun*
- ***The AI is not there to win.***
  - The reward functions for the RL training must factor more subjective requirements into their math: is the player engaged and entertained?
  - Is the player happy? Game designer must test with players and be able to **tune parameters**.
- Can it be used? Yes.
- Does it make game development easier or more efficient? **No.**



# RL for QA

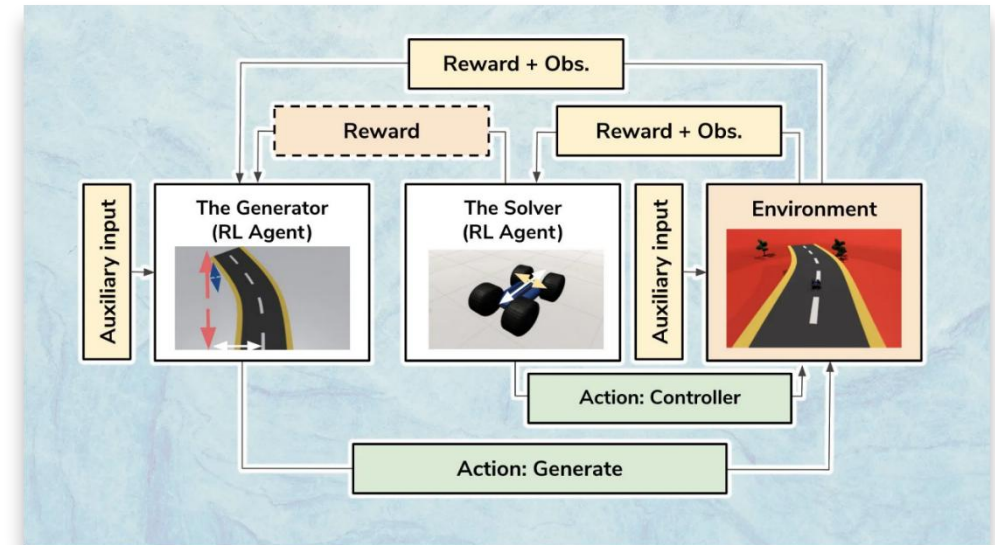
- Professional QA have to play the game over and over again, press the same button litteray hundreds of time per day.
- Can a RL model be created to test games and code in development and find bugs?
  - Find glitches;
  - Corner cases and soft-locks;
  - Character gets stuck;
  - Crashes.
- Each type and sub-type of issue can demand a very different RL training and reward functions.
- How can it *document* what it “sees” in the screen?
  - It needs to be able to describe *reproduction steps*.
- **The AI is *not there to win*.**
  - Reward functions and math should reward finding issues.



SEED // SEARCH FOR EXTRAORDINARY EXPERIENCES DIVISION  
seed.ea.com

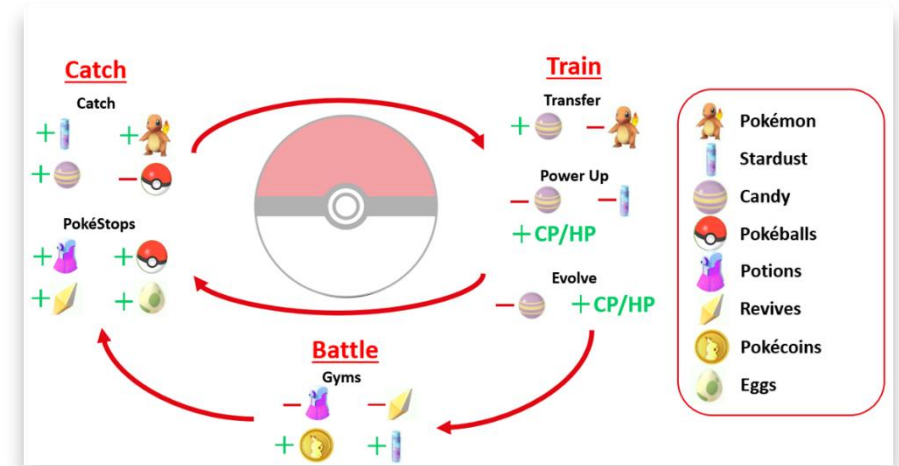
## Augmenting Automated Game Testing with Deep Reinforcement Learning

Joakim Bergdahl, Camilo Gordillo, Konrad Tollmar and Linus Gisslen  
SEED – Electronic Arts

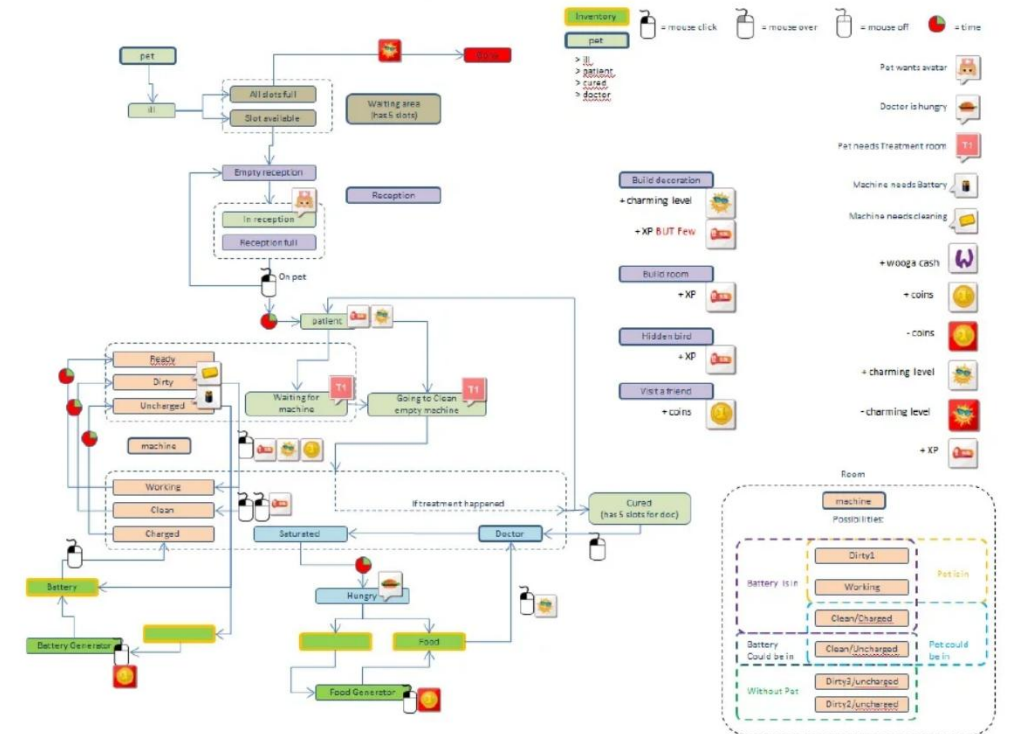


# RL for Economy and Balancing

- Games are systems of loops that repeat on top of each other.
  - Loops often involve the exchange of resources and currencies.
- Lots of modern games have long-term progression and permanent states of an account (“save-games”).
- Economic systems can be expressed as constructs that don’t necessarily need the AI to play the “core game”,
- This could accelerate training times enough that RL networks can be re-trained every time game designers are tuning economic values (example: “how much Unobtainium does it cost to craft a new artifact?”).



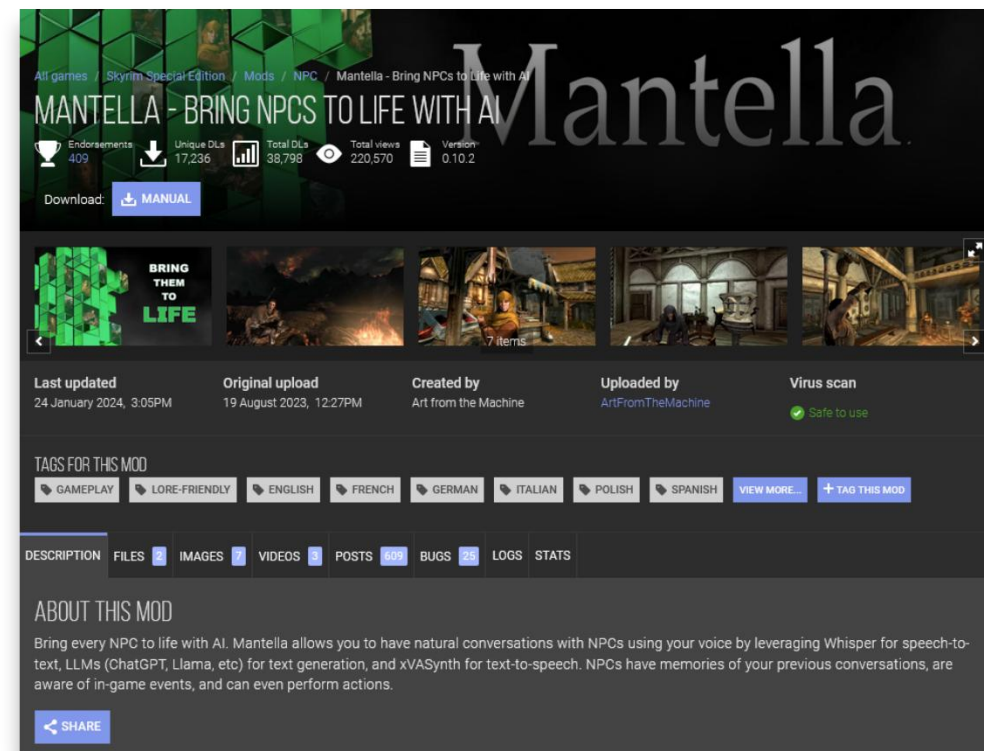
## The game core loop



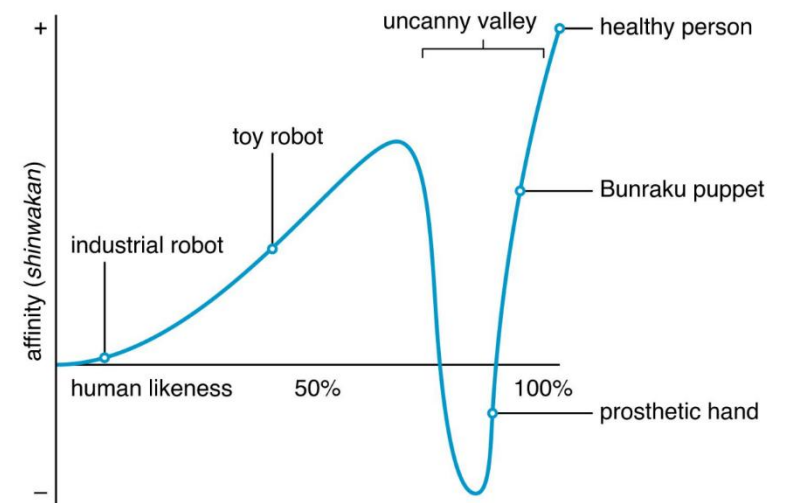


# RL for NPC dialogues

- The main problem is one of *immersion*. A second main problem is of a *legal and PR* nature.
- NPCs have limited knowledge of the world. But each NPC should know a bit more or less.
  - Does the **peasant** know about Dyrax the Red Mage in the crystal tower? He knows the place is scary and people who go there die.
  - Does the **travelling merchant** know about Dyrax the Red Mage? She knows the mage is known to buy black root and spices.
  - Does the **mayor** know about Dyrax the Red Mage in the crystal tower? He in fact has a corrupt deal with the mage to not investigate his crimes, and will lie.
- NPCs must never *break character*. It's easy to "game" current LLMs
- PR nightmare: game companies don't have nearly enough resources like Bug Tech to test for edge cases of bad speech.
- Can it be done? Yes, but don't underestimate the "uncanny valley".



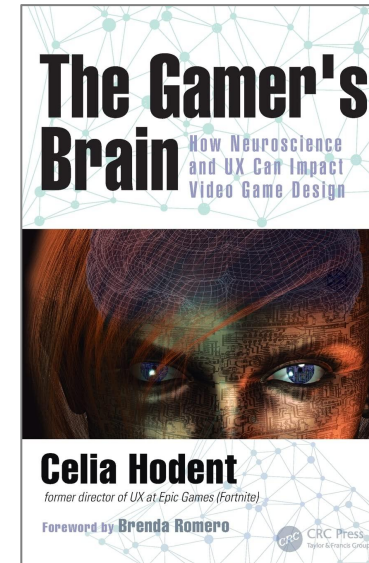
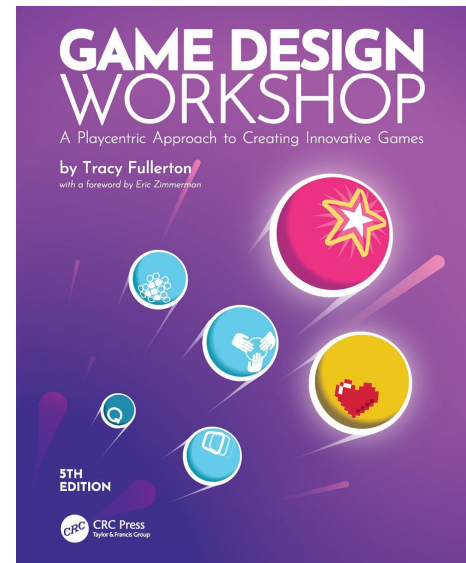
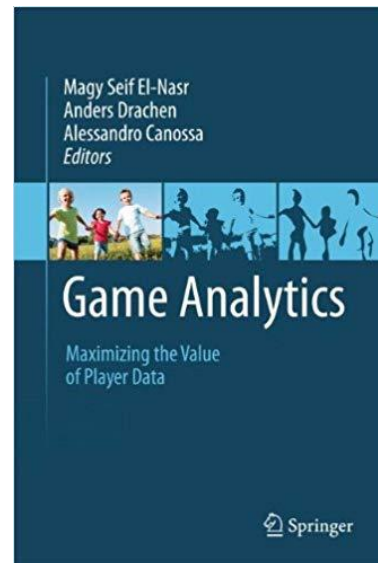
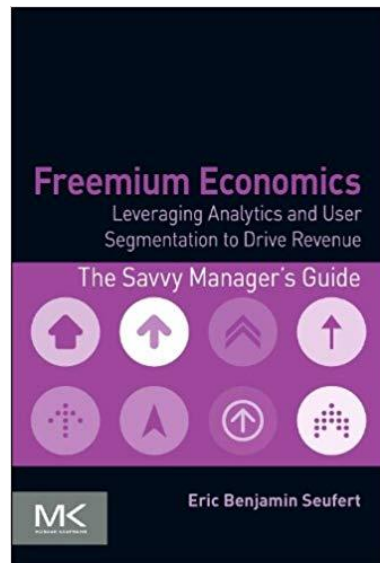
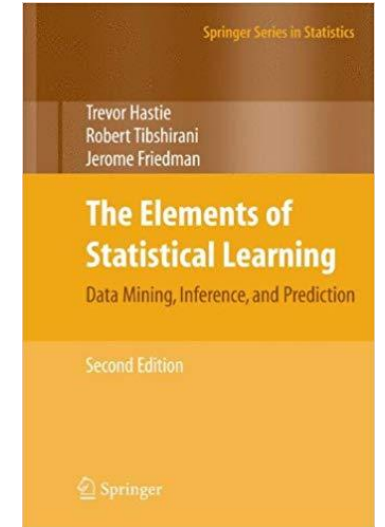
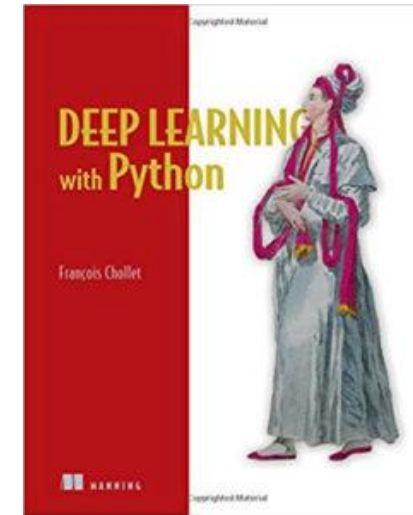
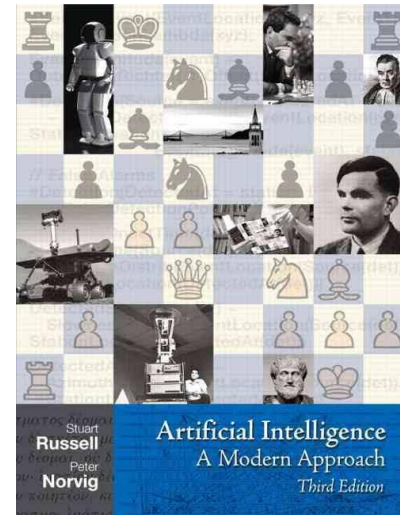
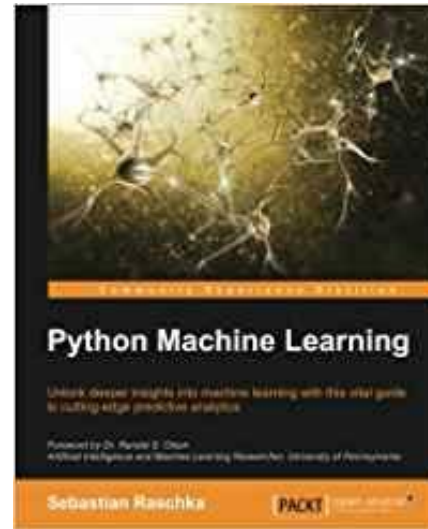
The screenshot shows the Steam Workshop page for the mod "Mantella - Bring NPCs to Life with AI". The page features a header with the mod title and a large "Mantella" logo. Below the header, there are statistics for the mod, including the number of endorsements (409), unique downloads (17,236), total downloads (38,798), total views (220,570), and version (0.10.2). A "Download" button is visible. The page also includes a "Last updated" date (24 January 2024, 3:05PM), "Original upload" date (19 August 2023, 12:27PM), "Created by" (Art from the Machine), and "Uploaded by" (ArtFromTheMachine). A "Virus scan" section indicates the mod is "Safe to use". The "TAGS FOR THIS MOD" section lists various categories like "GAMEPLAY", "LORE-FRIENDLY", "ENGLISH", "FRENCH", "GERMAN", "ITALIAN", "POLISH", and "SPANISH". The "DESCRIPTION" tab is selected, showing the "ABOUT THIS MOD" section, which describes the mod's functionality: "Bring every NPC to life with AI. Mantella allows you to have natural conversations with NPCs using your voice by leveraging Whisper for speech-to-text, LLMs (ChatGPT, Llama, etc) for text generation, and xVASynth for text-to-speech. NPCs have memories of your previous conversations, are aware of in-game events, and can even perform actions." A "SHARE" button is also present.



Source: Masahiro Mori, "The Uncanny Valley," *IEEE Robotics & Automation Magazine*, 19(2):98-100 (June 6, 2012).







Start here!

<http://www.r2d3.us/>



Thank you!  
Questions?  
[tex@texpine.com](mailto:tex@texpine.com)  
[texpine.com](http://texpine.com)

# Appendix

-  Icon Game by Orin zuu from the Noun Project
-  Icon Lighthouse by Nikita Kozin from the Noun Project
-  Icon User by Gagana from the Noun Project
-  Icon Car on Sale from all-free-download.com / BSGStudio
-  Icon Flag from all-free-download.com / Vector Graphics
-  Icon Flag from all-free-download.com / BSGStudio
- All materials sourced on Udacity are subject of Creative Commons Attribution-NonCommercial- NoDerivs 3.0 License, located at <http://creativecommons.org/licenses/by-nc-nd/4.0>