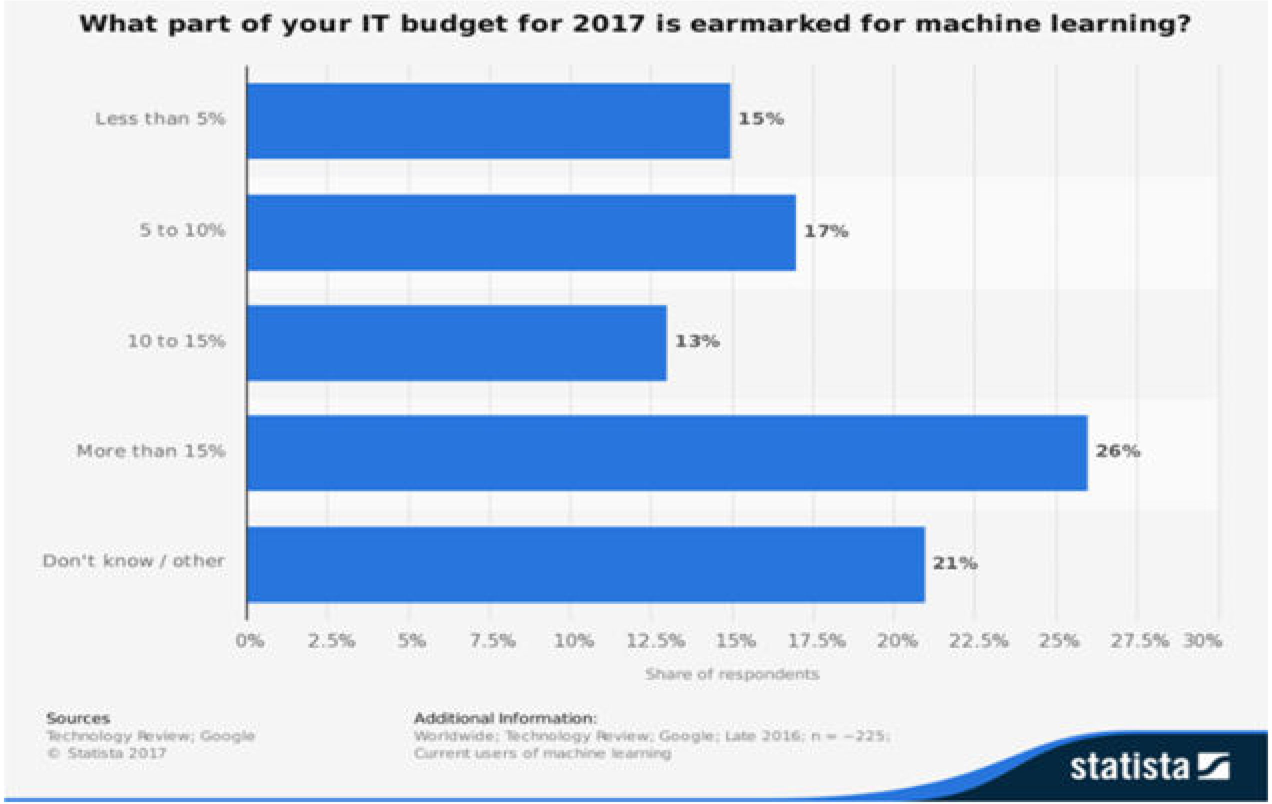
Machine Learning Introduction

James Connan

Why Machine Learning?

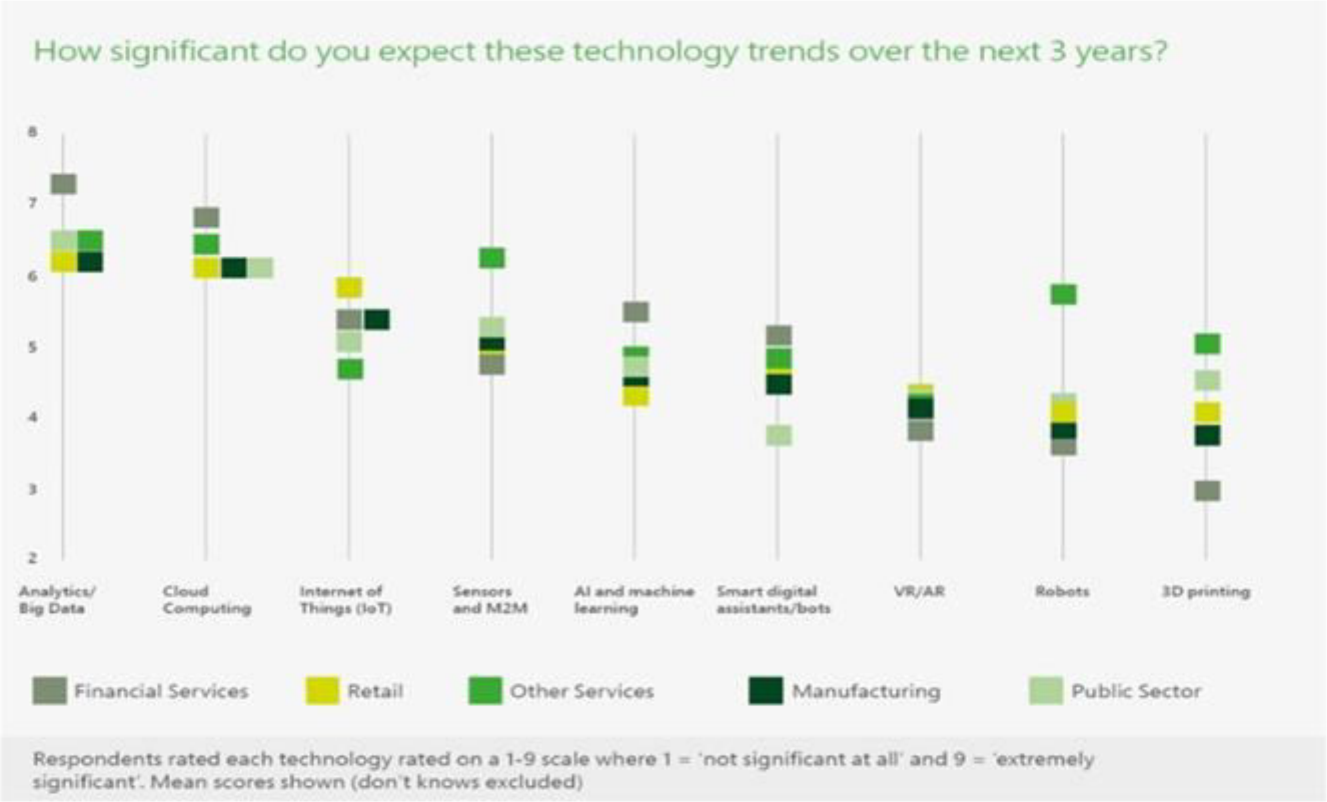


What part of your IT budget for 2017 is earmarked for machine learning?

Sources
Technology Review; Google
© Statista 2017

Additional Information:
Worldwide; Technology Review; Google; Late 2016; n = ~225;
Current users of machine learning

Is it relevant?



How significant do you expect these technology trends over the next 3 years?

Respondents rated each technology rated on a 1-9 scale where 1 = 'not significant at all' and 9 = 'extremely significant'. Mean scores shown (don't knows excluded)

What is Machine Learning?

- Is an approach to digital transformation that adds new capability for computers to make computing processes more efficient, cost-effective, and reliable.
- It is a business-critical technology that makes decision-making a more data-driven affair.
- It grew out of work in Artificial Intelligence (AI) but means a lot of things.
    - Its field is vast and is expanding rapidly by being continually partitioned and sub-partitioned into different specialties and subspecialties of machine learning

Examples of Machine Learning

- Database Mining: large datasets from growth of automation/web.
  - E.g., Web click data, medical records, biology, engineering
- Application that can’t program by hand.
  - E.g., Autonomous helicopter, handwriting recognition, most of Natural Language Processing (NLP), Computer Vision.
- Self-customizing programs
  - E.g., Amazon, Netflix product recommendations
- Understanding human learning (brain, real AI).

Machine Learning Applications 1

- Customer services firms track customer happiness.
  - By analyzing user activity, smart machines can spot a potential account closure before it occurs. They can also track spending patterns and customer behavior to offer tailored financial advice.
- Reacting to market trends.
  - Another application of machine learning is market analysis. Smart machines can be trained to track trading volatility or manage wealth and assets on behalf of an investor. These algorithms can identify trends more efficiently than humans and react in real-time (reducing the impact of major financial events such as Brexit).
- Calculating risk.
  - Smart machines can analyze a large number of disparate datasets (credit scores, spending patterns, financial data etc.) to accurately assess risk in both insurance underwriting and loan assessments, tailoring them to a specific customer profile.

Machine Learning Applications 2

- Remaining competitive.
  - This example of machine learning is perhaps the most relatable to management level execs, giving firms a clinical edge in a fierce industry, by helping them remain innovative. With the right machine learning algorithms, companies can act quickly on business intelligence, increasing productivity and opening up new streams of revenue.
- Personalized health monitoring.
  - Smart watches and other wearable devices (IoT) have made health telemetry a reality. But machine learning is taking things one step further, allowing doctors and relatives to monitor the health of elderly family members. The more personal data these algorithms are fed, the better they understand a user's profile, enabling healthcare professionals to spot potential anomalies earlier on.

Machine Learning Applications 3

- Online recommendations.
    - Machine learning allows retailers to offer you personalized recommendations based on your previous purchases or activity.
- Better customer service and delivery systems.
    - In large companies where response time is limited by staff resources, machine learning can help ease some of the burden. Smart machines can decipher the intent and meaning behind emails and delivery notes to prioritise tasks and ensure sustained satisfaction.
- Tracking price changes.
    - The price of retail items tends to fluctuate over a certain period of time. Machine learning is helping ecommerce companies track patterns in these fluctuations and set their prices according to demand.

Machine Learning Defined

- 1959: Arthur Samuel stated:
  - “**Machine Learning is the field of study that gives computers the ability to learn without being explicitly programmed.**”
- 1997: Tom Mitchell stated:
  - “**A computer program is said to learn from experience E with respect to some task T and some performance measure P, if its performance on T, as measured by P, improves with experience E.**”
- So if you want your program to predict, for example, traffic patterns at a busy intersection (**task T**), you can run it through a machine learning algorithm with data about past traffic patterns (**experience E**) and, if it has successfully “learned”, it will then do better at predicting future traffic patterns (**performance measure P**).

Spam Filter

“A computer program is said to learn from experience **E** with respect to some task **T** and some performance measure **P**, if its performance on **T**, as measured by **P**, improves with experience **E**.”

- Suppose your email program watches which emails you do or do not mark as spam, and based on that learns how to better filter spam. What is the task **T** in this setting?
    - Classifying emails as spam or not spam:
    - Watching you label emails as spam or not spam:
    - The number (or fraction) of emails correctly classified as spam/not spam:
    - None of the above—this is not a machine learning problem.

Spam Filter

“A computer program is said to learn from experience **E** with respect to some task **T** and some performance measure **P**, if its performance on **T**, as measured by **P**, improves with experience **E**.”

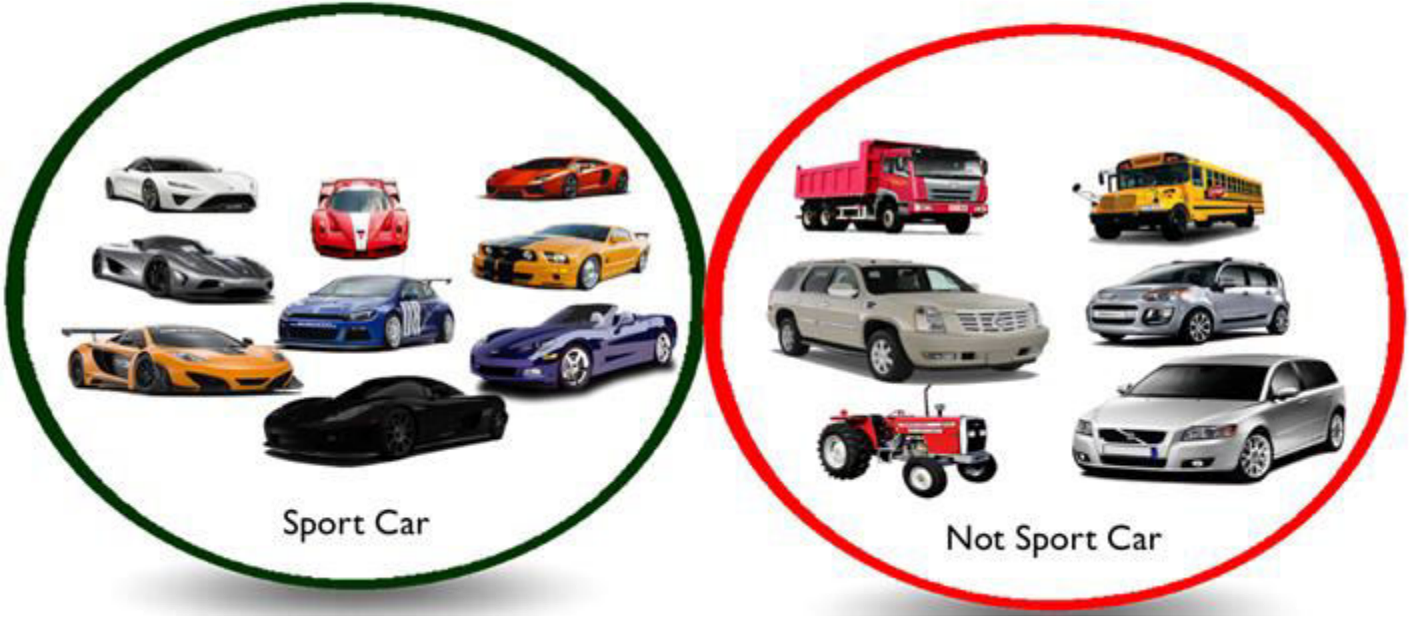
- Suppose your email program watches which emails you do or do not mark as spam, and based on that learns how to better filter spam. What is the task **T** in this setting?
  - ○ Classifying emails as spam or not spam: **T**
  - ○ Watching you label emails as spam or not spam: **E**
  - ○ The number (or fraction) of emails correctly classified as spam/not spam: **P**
  - ○ None of the above—this is not a machine learning problem.

Types of Machine Learning

- Supervised learning
  - The program is “trained” on a predefined set of “training examples”, which then facilitate its ability to reach an accurate conclusion when given new data. Two major subcategories are:
    - Regression machine learning systems: Systems where the value being predicted falls somewhere on a continuous spectrum. These systems help us with questions of “How much?” or “How many?”.
    - Classification machine learning systems: Systems where we seek a yes-or-no prediction, such as “Is this tumour cancerous?”, “Does this cookie meet our quality standards?”, and so on.
- Unsupervised learning
  - The program is given a bunch of data and must find patterns and relationships therein.
- Reinforcement
- Recommender systems.

Supervised Learning: Training

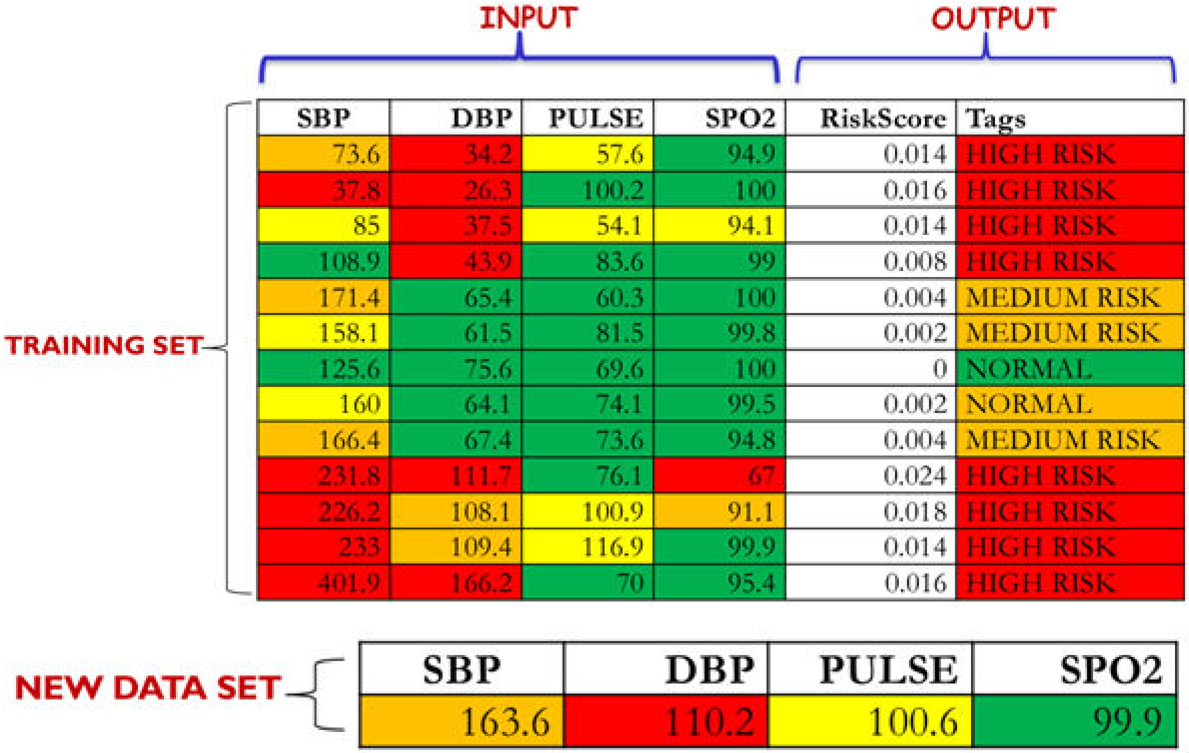
- Automatic learning and feature selection.
  - Supply training data (+ve and –ve classes).



Supervised Learning: Classification

Classification: Sport Car

Supervised Learning: Health


INPUT
OUTPUT
TRAINING SET
NEW DATA SET

SBP	DBP	PULSE	SPO2	RiskScore	Tags
73.6	34.2	57.6	94.9	0.014	HIGH RISK
37.8	26.3	100.2	100	0.016	HIGH RISK
85	37.5	54.1	94.1	0.014	HIGH RISK
108.9	43.9	83.6	99	0.008	HIGH RISK
171.4	65.4	60.3	100	0.004	MEDIUM RISK
158.1	61.5	81.5	99.8	0.002	MEDIUM RISK
125.6	75.6	69.6	100	0	NORMAL
160	64.1	74.1	99.5	0.002	NORMAL
166.4	67.4	73.6	94.8	0.004	MEDIUM RISK
231.8	111.7	76.1	67	0.024	HIGH RISK
226.2	108.1	100.9	91.1	0.018	HIGH RISK
233	109.4	116.9	99.9	0.014	HIGH RISK
401.9	166.2	70	95.4	0.016	HIGH RISK

SBP	DBP	PULSE	SPO2
163.6	110.2	100.6	99.9

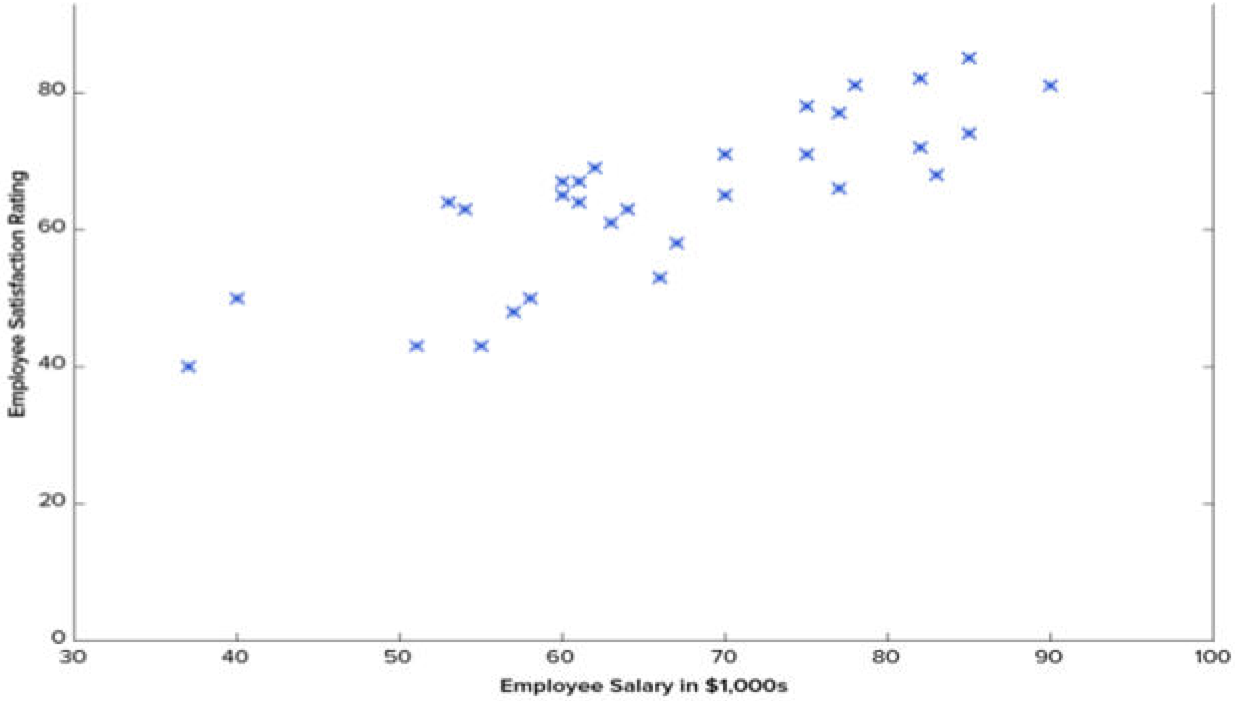
The Supervised Learning Process

- The goal is to develop a finely tuned predictor function ***h(x)***.
  - Sometimes called the “hypothesis”.
- The “Learning” process consists of using sophisticated mathematical algorithms to optimize the function ***h(x)*** so that
  - given input data ***x*** about a certain domain (say, the size of a house house)
  - it will accurately predict some interesting value ***h(x)*** (say, market price for said house).
- In practice, *x* almost always represents multiple data points.
  - A housing price predictor might take not only square-footage (x_1) but also number of bedrooms (x_2), number of bathrooms (x_3), number of floors (x_4), year built (x_5), zip code(x_6), and so forth.
- Determining which inputs to use is an important part of ML design.
- The predictor can be of the form $h(x) = \theta_0 + \theta_1 x$ where θ_0 and θ_1 are constant values.

The Supervised Learning Process

- Our goal is to find the perfect values of $\boldsymbol{\theta_0}$ and $\boldsymbol{\theta_1}$ to make our predictor work as well as possible. Optimizing the predictor ***h(x)*** is done using **training examples**.
- For each training example, we have an input value **x_train**, for which a corresponding output, **y**, is known in advance.
- For each example, we find the difference between the known, correct value **y**, and our predicted value ***h(x_train)***.
- With enough training examples, these differences give us a useful way to measure the error of ***h(x)***. We can then tweak ***h(x)*** by tweaking the values of $\boldsymbol{\theta_0}$ and $\boldsymbol{\theta_1}$ to reduce the error.
- This process is repeated over and over until the system has converged on the best values for $\boldsymbol{\theta_0}$ and $\boldsymbol{\theta_1}$. In this way, the predictor becomes trained, and is ready to do some real-world predicting.

Regression Example (Supervised)



Regression Example (Supervised)

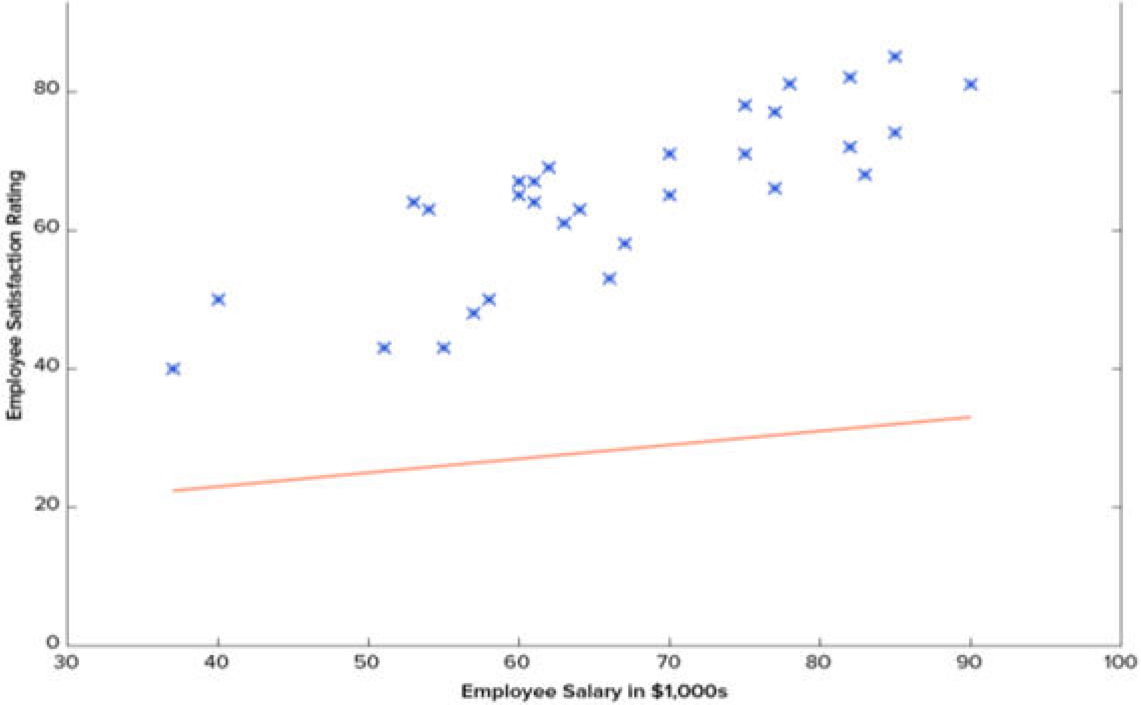
- First, notice that the data is a little noisy: while we can see that there is pattern to it (i.e. employee satisfaction tends to go up as salary goes up), it does not all fit neatly on a straight line. This will always be the case with real-world data (and we absolutely want to train our machine using real-world data!).
- So then how can we train a machine to perfectly predict an employee's level of satisfaction?
  - The answer, of course, is that we can't. The goal of ML is never to make "perfect" guesses, because ML deals in domains where there is no such thing. The goal is to make guesses that are good enough to be useful.
- **The goal of ML is never to make "perfect" guesses, because ML deals in domains where there is no such thing. The goal is to make guesses that are good enough to be useful.**

Regression Example (Supervised)

- Machine Learning builds heavily on statistics.
  - For example, when we train our machine to learn, we have to give it a statistically significant random sample as training data. If the training set is not random, we run the risk of the machine learning patterns that aren't actually there.
  - If the training set is too small , we won't learn enough and may even reach inaccurate conclusions.
    - For example, attempting to predict company-wide satisfaction patterns based on data from upper management alone would likely be error-prone.
- First we have to initialize our predictor ***h(x)*** with some reasonable values of $\boldsymbol{\theta_0}$ and $\boldsymbol{\theta_1}$.Now our predictor looks like this when placed over our training set:
  - $h(x) = 12.00 + 0.20x$

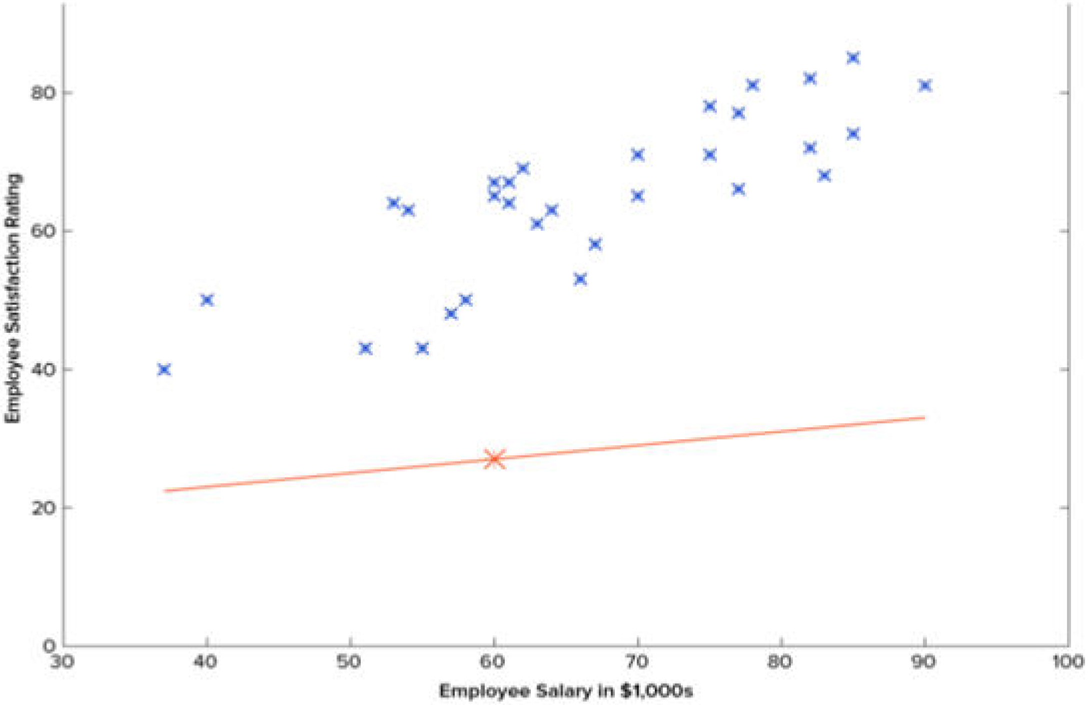
Regression Example (Supervised)

- $\theta_0 = 12.00$
- $\theta_1 = 0.20$
- $h(x) = 12.00 + 0.20x$



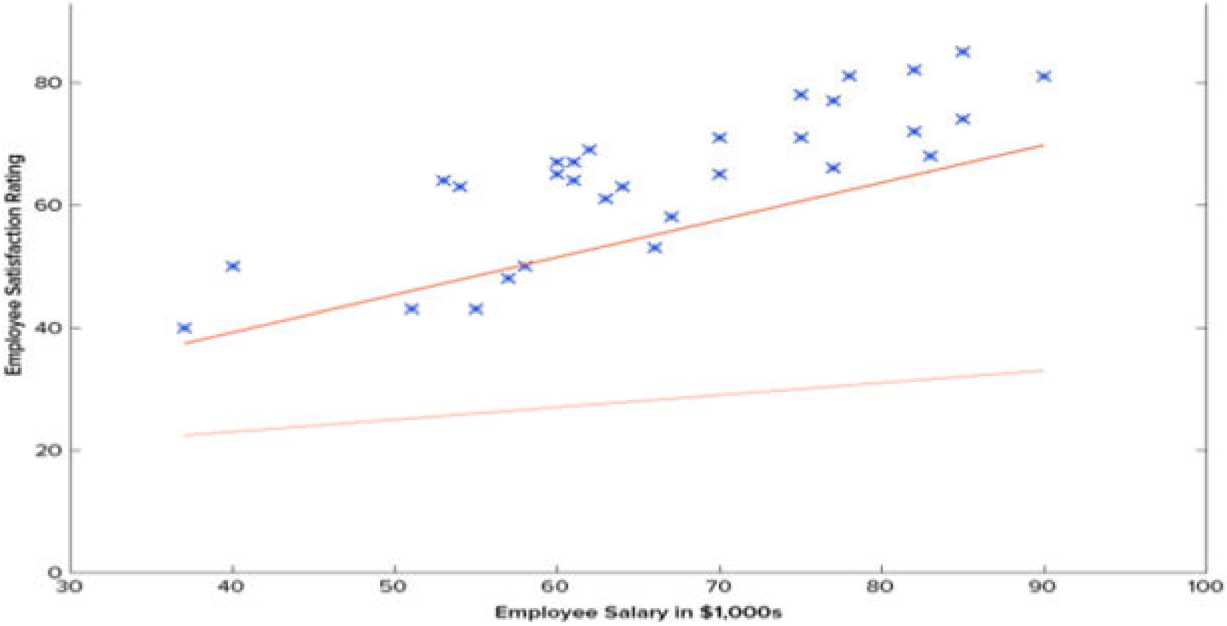
Regression Example (Supervised)

- $\theta_0 = 12.00$
- $\theta_1 = 0.20$
- $h(x) = 12.00 + 0.20x$
- $h(60k) = 24$



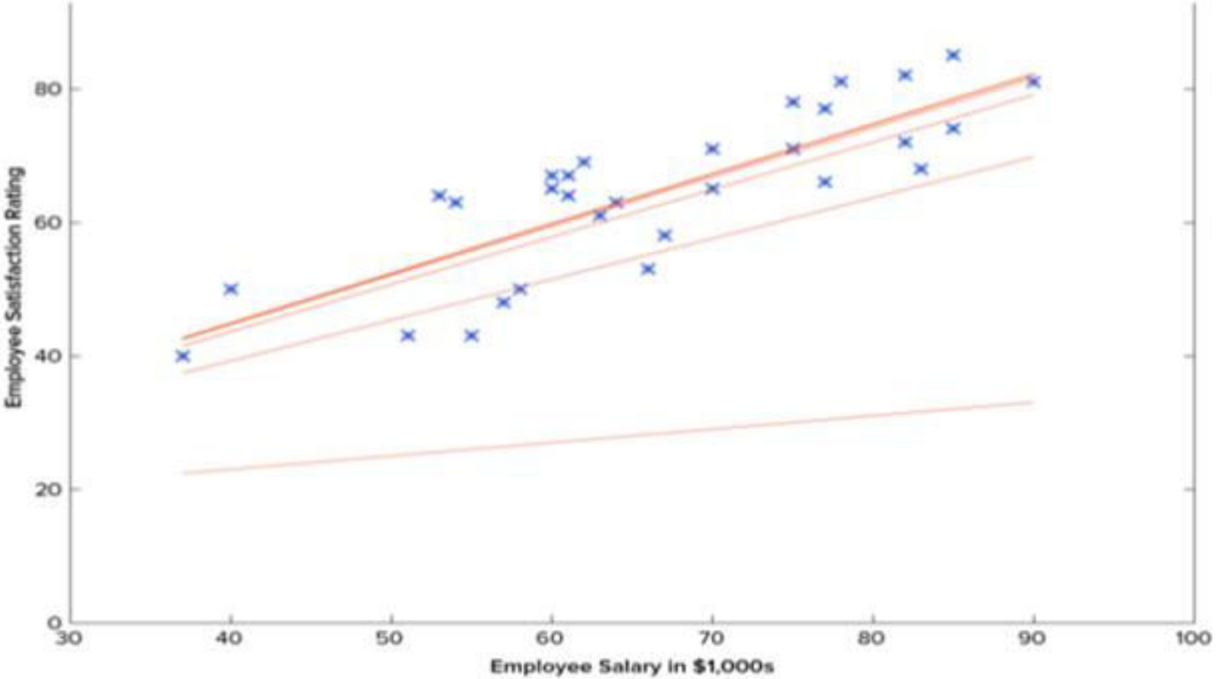
Regression Example (Supervised)

- *Gradient Descent*
- *$\theta_0 = 13.12$*
- *$\theta_1 = 0.61$*
- ***$h(x) = 13.12 + 0.61x$***



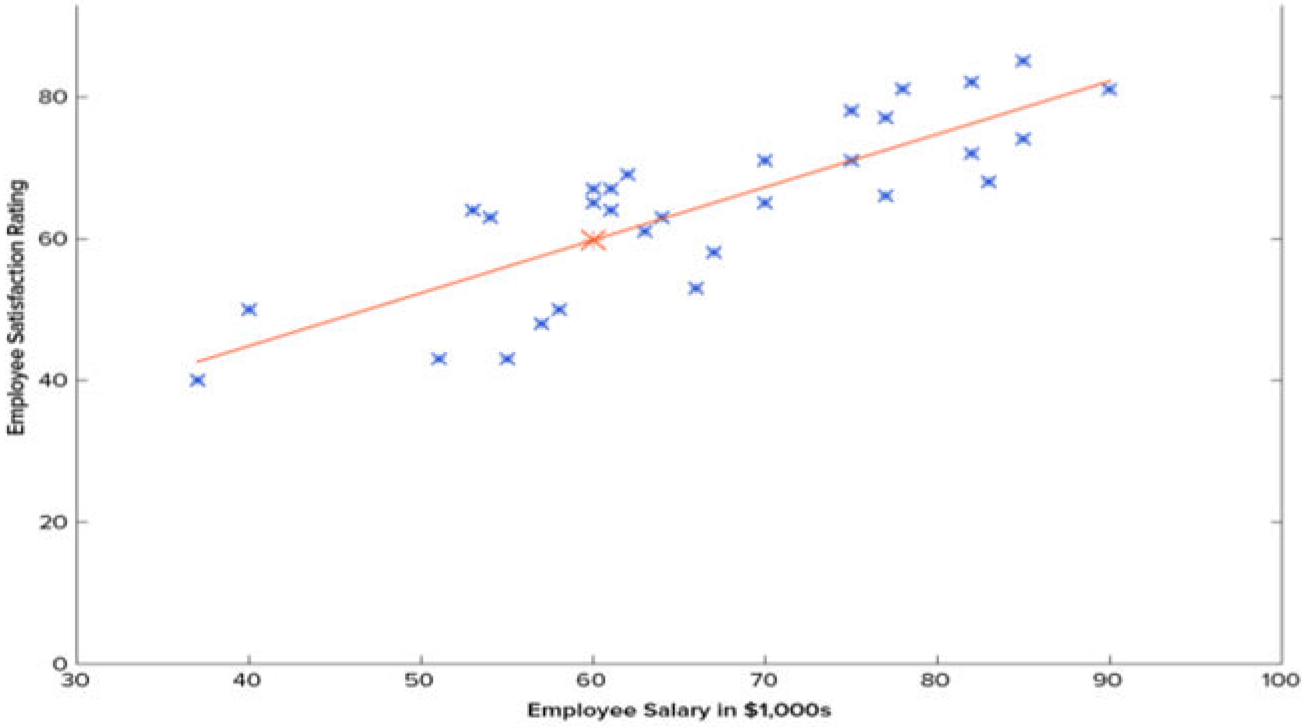
Regression Example (Supervised)

- *Gradient Descent*
- *Converges*
    - ○ *1500 iterations*
- $\boldsymbol{\theta_0}$ = *15.54*
- $\boldsymbol{\theta_1}$ = *0.75*
- ***h(x) = 15.45 + 0.75x***



Regression Example (Supervised)

- *h(60k) = 60*



Regression Example (Supervised)

- This example is technically a simple problem of univariate linear regression, which in reality can be solved by deriving a simple normal equation and skipping this “tuning” process altogether. However, consider a predictor that looks like this:
  - $h(x_1,x_2,x_3,x_4) = \theta_0 + \theta_1 x_1 + \theta_2 x_2^2 + \theta_3 x_3 x_4 + \theta_4 x_1^3 x_2^2 + \theta_5 x_2 x_3^4 x_4^2$
- This function takes input in four dimensions and has a variety of polynomial terms. Deriving a normal equation for this function is a significant challenge.
- Many modern machine learning problems take thousands or even millions of dimensions of data to build predictions using hundreds of coefficients.
  - Predicting how an organism’s genome will be expressed, or what the climate will be like in fifty years, are examples of such modern complex problems.

Regression Example (Supervised)

- Many ML problems take thousands or even millions of dimensions of data to build predictions using hundreds of coefficients. Fortunately, the iterative approach taken by ML systems is much more resilient in the face of such complexity.
- Instead of using brute force, a machine learning system “feels its way” to the answer. For big problems, this works much better. While this doesn’t mean that ML can solve all arbitrarily complex problems (it can’t), it does make for an incredibly flexible and powerful tool.

Gradient Descent (Supervised)

- As suggested earlier, machine learning algorithms rely on the gradient descent method to minimize wrongness in order to converge the learning process to the optimal predictor.
- Let’s take a closer look at how this iterative process works. In the above example, how do we make sure that $\boldsymbol{\theta_0}$ and $\boldsymbol{\theta_1}$ are getting better at each step and not worse ?
- The answer lies in our “measurement of wrongness” (**error**) alluded to previously, along with a little calculus.

Gradient Descent (Supervised)

- The wrongness measure is known as the **cost function** (a.k.a., loss function, error), ***J(θ)***
    - The input represents all of the coefficients we are using in our predictor. So in our case, is really the pair $\boldsymbol{\theta_0}$ and $\boldsymbol{\theta_1}$. $\boldsymbol{J(\theta_0,\theta_1)}$ gives us a mathematical measurement of how wrong our predictor is when it uses the given values of $\boldsymbol{\theta_0}$ and $\boldsymbol{\theta_1}$.
- The choice of the cost function is another important piece of an ML program. In different contexts, being “wrong” can mean very different things. In our employee satisfaction example, the well-established standard is the linear least square function:

$$J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h(x_{t,i}) - y)^2$$

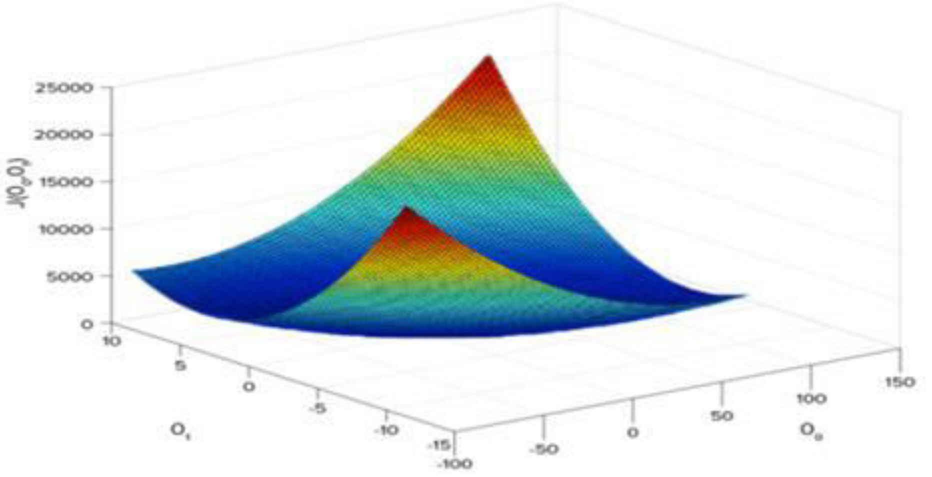
Gradient Descent (Supervised)

- In our employee satisfaction example, the well-established standard is the linear least square function: $J(\theta_0, \theta_1) = \frac{1}{2m} \sum_{i=1}^{m} (h(x_{t,i}) - y)^2$
- With least squares, the penalty for a bad guess goes up quadratically with the difference between the guess and the correct answer, so it acts as a very “strict” measurement of wrongness. The cost function computes an average penalty over all of the training examples.
- So now we see that our goal is to find ***θ*₀** and ***θ*₁** for our predictor ***h(x)*** such that our cost function is as small as possible. We call on the power of calculus to accomplish this.

Gradient Descent (Supervised)

Consider the following plot of a cost function $J(\theta_0, \theta_1) = \frac{1}{2m}\sum_{i=1}^{m}(h(x_{t,i}) - y)^2$ for some particular ML problem where we can see the cost associated with different values of $\boldsymbol{\theta_0}$ and $\boldsymbol{\theta_1}$.

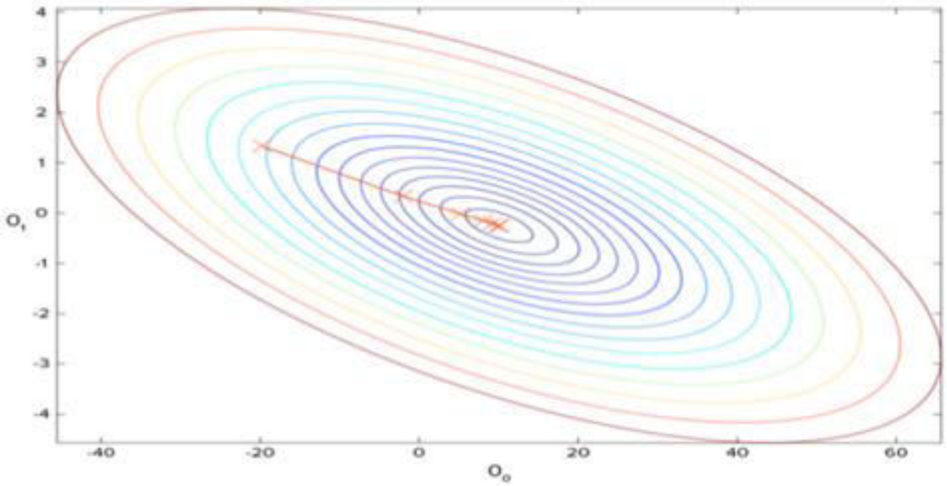
The bottom of the bowl represents the lowest cost our predictor can give us based on the given training data. The goal is to “roll down the hill”, and find $\boldsymbol{\theta_0}$ and $\boldsymbol{\theta_1}$.



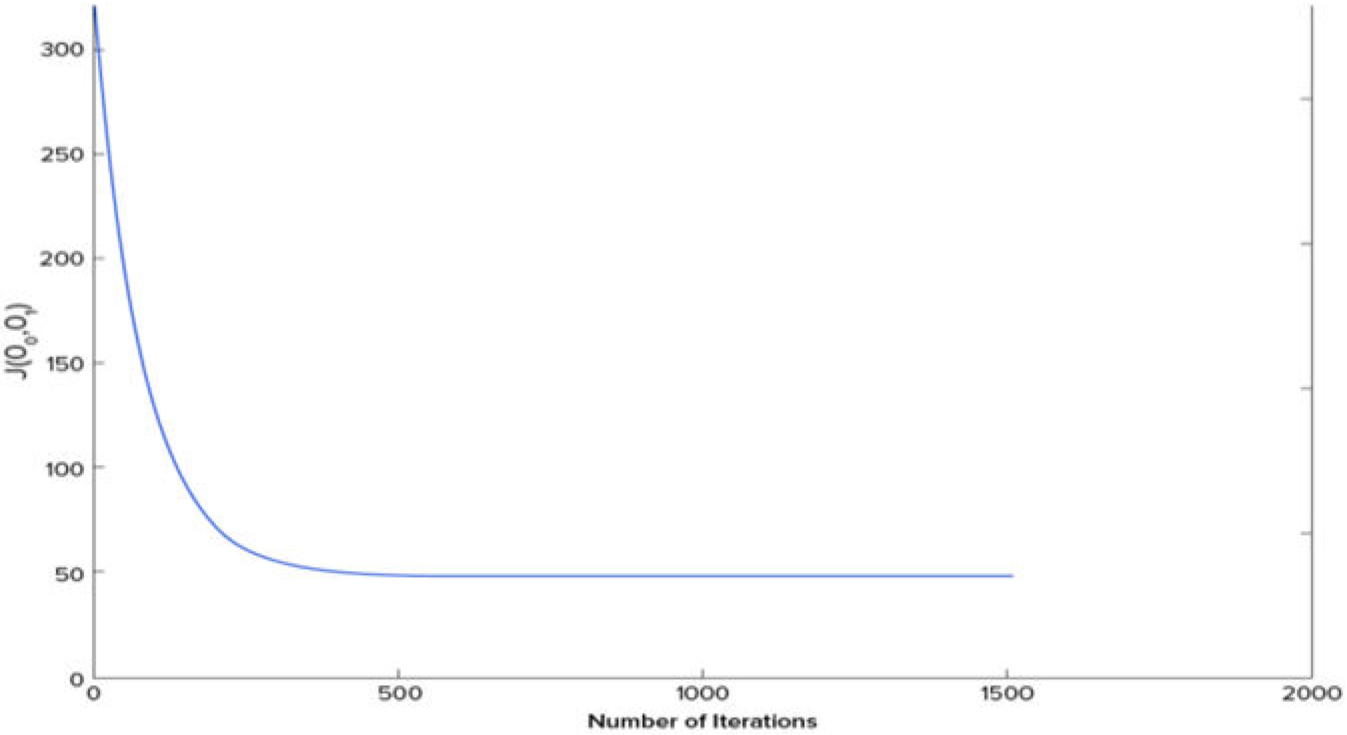
Gradient Descent (Supervised)

- Calculus is used to implement the gradient descent. The idea is to take the gradient of $J(\theta_0,\theta_1)$, which is the pair of derivatives of $J(\theta_0,\theta_1)$, one over θ_0 and the other over θ_1.

- The gradient descent is the process of alternating between calculating the current gradient, and updating θ from the results.

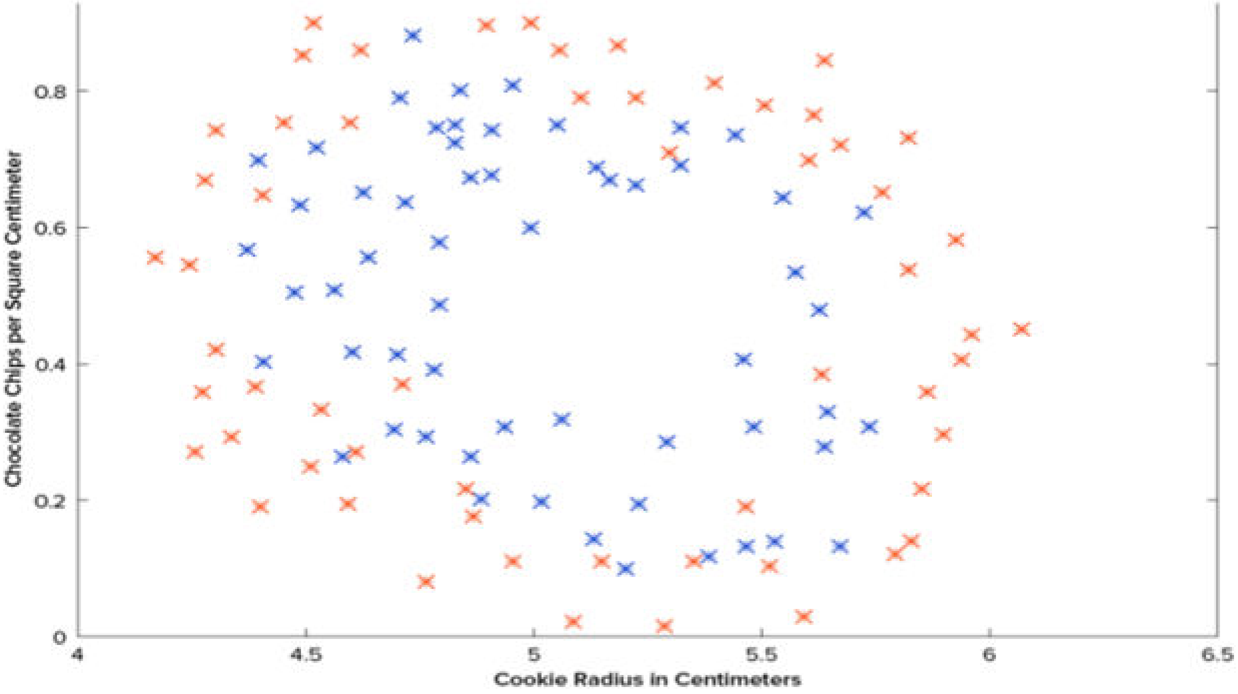


Gradient Descent (Supervised)



Classification Example (Supervised)

We have be given the results of a cookie quality testing study, where the training examples have all been labelled as either “good cookie” (y = 1) in **blue** or “bad cookie” (y = 0) in **red**.

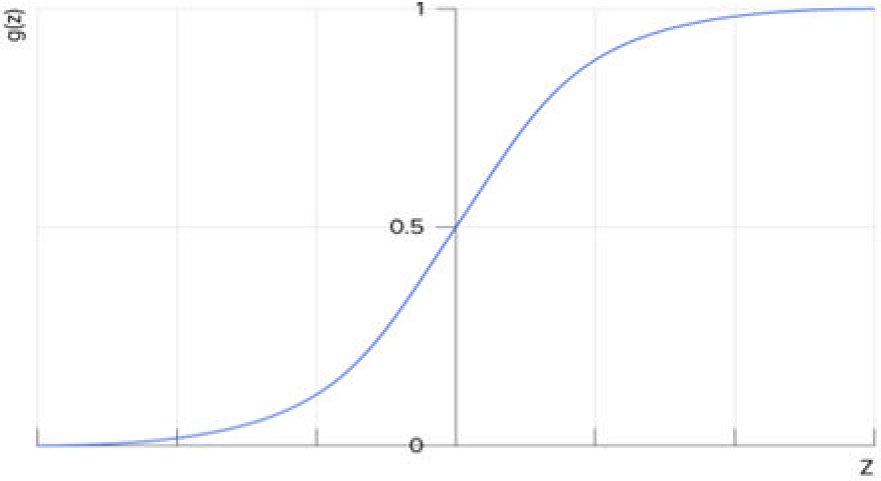


Classification Example (Supervised)

- In classification, a regression predictor is not very useful. What we usually want is a predictor that makes a guess somewhere between **0** and **1**.
- In a cookie quality classifier, a prediction of **1** would represent a very confident guess that the cookie is perfect and utterly mouthwatering.
- A prediction of **0** represents high confidence that the cookie is an embarrassment to the cookie industry.
- Values falling within this range represent less confidence, so we might design our system such that prediction of 0.6 means “Man, that’s a tough call, but I’m gonna go with yes, you can sell that cookie,” while a value exactly in the middle, at 0.5, might represent complete uncertainty. This isn’t always how confidence is distributed in a classifier but it’s a very common design and works for purposes of our illustration.

Classification Example (Supervised)

- It turns out there's a nice function that captures this behavior well. It's called the sigmoid function, ***g(z)***, and it looks something like this:
  - ***h(x) = g(z)***
- ***z*** is some representation of our inputs and coefficients, such as: $\boldsymbol{z = \theta_0 + \theta_1 x}$ so that our predictor becomes:
  - $\boldsymbol{h(x) = g(\theta_0 + \theta_1 x)}$
- Notice that the sigmoid function transforms our output into the range between **0** and **1**.



Classification Example (Supervised)

- The logic behind the design of the cost function is also different in classification. When designing the classification cost function, the following question is asked:
  - “what does it mean for a guess to be wrong?”
- The answer is found in a very good rule of thumb: if the correct guess was **0** and we guessed **1**, then we were completely and utterly wrong, and vice-versa. And since we can’t be more wrong than absolutely wrong, the penalty in this case is enormous.
- Alternatively if the correct guess was **0** and we guessed **0**, our cost function should not add any cost for each time this happens. If the guess was right, but we weren’t completely confident (e.g. y = **1**, but h(x) = **0.8**), this should come with a small cost.
- If our guess was wrong but we weren’t completely confident (e.g. y = **1** but h(x) = **0.3**), this should come with some significant cost, but not as much as if we were completely wrong.
- This behaviour above is captured by the **log function**, such that:

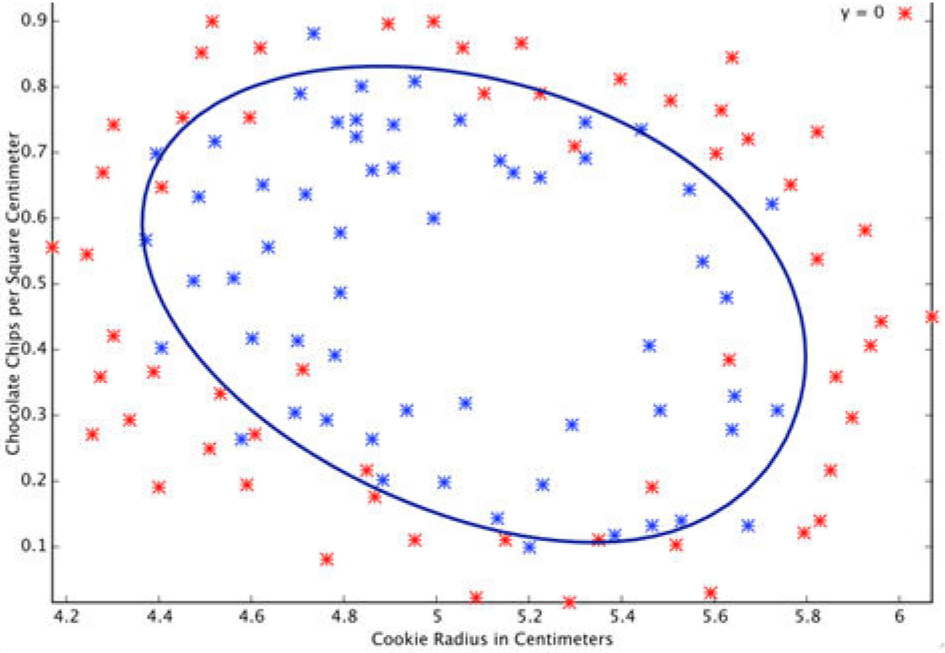
$$cost = \begin{cases} -log(h(x)) & ify = 1 \\ -log(1-h(x)) & ify = 0 \end{cases}$$

Classification Example (Supervised)

- Again, the cost function ***J(θ)*** gives us the average cost over all of our training examples.
- Note that tough the predictor ***h(x)*** and the cost function differ between regression and classification, the gradient descent still works fine.
- A classification predictor can be visualized by drawing the **boundary line**; i.e., the barrier where the prediction changes from a “yes” (a prediction greater than 0.5) to a “no” (a prediction less than 0.5).

Classification Example (Supervised)

A well-designed cookie data system can generate a classification boundary that looks as follows:



Unsupervised ML Algorithms

- Unsupervised machine learning is typically tasked with finding relationships within data.
  - There are no training examples used in the unsupervised machine learning process: the system is given a data set and tasked with finding patterns and correlations therein.
    - A good example is identifying close-knit groups of friends in social network data.
  - Some of the unsupervised machine learning algorithms include
    - clustering algorithms such as K-means and
    - dimensionality reduction algorithms such as principal component analysis.

Common ML Algorithms

- Linear Regression
- Logistic regression
- Neural network
- Support vector machine
- k-means Clustering
- Hidden Markov Models
- Decision tree
- Random forest
- Naive Bayes
- k-Nearest Neighbor
- Adaboost