Cara kerja jaringan saraf tiruan

Sebuah JST biasanya melibatkan sejumlah besar prosesor yang beroperasi secara paralel dan diatur dalam tingkatan. Tingkat pertama menerima informasi input mentah -- analog dengan saraf optik dalam pemrosesan visual manusia. Setiap tingkat berturut-turut menerima output dari tingkat sebelumnya, bukan input mentah - dengan cara yang sama neuron lebih jauh dari saraf optik menerima sinyal dari mereka yang lebih dekat. Tingkat terakhir menghasilkan output dari sistem.

Setiap node pemrosesan memiliki lingkup kecil pengetahuannya sendiri, termasuk apa yang telah dilihatnya dan aturan apa pun yang awalnya diprogram atau dikembangkan untuk dirinya sendiri. Tier sangat saling berhubungan, yang berarti setiap node di tier n akan terhubung ke banyak node di tier n-1 -- inputnya -- dan di tier n+1, yang menyediakan data input untuk node tersebut. Mungkin ada satu atau beberapa node di lapisan output, dari mana jawaban yang dihasilkannya dapat dibaca.

Jaringan saraf tiruan terkenal karena adaptif, yang berarti mereka memodifikasi diri mereka sendiri saat mereka belajar dari pelatihan awal dan proses selanjutnya memberikan lebih banyak informasi tentang dunia. Model pembelajaran yang paling dasar berpusat pada pembobotan aliran input, yaitu bagaimana setiap node menimbang pentingnya data input dari masing-masing pendahulunya. Masukan yang berkontribusi untuk mendapatkan jawaban yang benar diberi bobot lebih tinggi.

Bagaimana jaringan saraf belajar

Biasanya, JST awalnya dilatih atau diberi masukan sejumlah besar data. Pelatihan terdiri dari memberikan input dan memberi tahu jaringan seperti apa output yang seharusnya. Misalnya, untuk membangun jaringan yang mengidentifikasi wajah aktor, pelatihan awal mungkin berupa serangkaian gambar, termasuk aktor, non-aktor, topeng, patung, dan wajah binatang. Setiap input disertai dengan identifikasi yang cocok, seperti nama aktor atau informasi "bukan aktor" atau "bukan manusia". Memberikan jawaban memungkinkan model menyesuaikan bobot internalnya untuk mempelajari cara melakukan tugasnya dengan lebih baik.

Misalnya, jika node David, Dianne, dan Dakota memberi tahu node Ernie bahwa gambar input saat ini adalah gambar Brad Pitt, tetapi node Durango mengatakan itu adalah Betty White, dan program pelatihan mengonfirmasi bahwa itu adalah Pitt, Ernie akan mengurangi bobot yang diberikannya. Masukan Durango dan peningkatan bobot yang diberikannya kepada David, Dianne, dan Dakota.

Dalam menentukan aturan dan membuat penentuan -- yaitu, keputusan setiap node tentang apa yang akan dikirim ke tingkat berikutnya berdasarkan masukan dari tingkat sebelumnya -- jaringan saraf menggunakan beberapa prinsip. Ini termasuk pelatihan berbasis gradien, logika fuzzy, algoritma genetika dan metode Bayesian. Mereka mungkin diberikan beberapa aturan dasar tentang hubungan objek dalam data yang dimodelkan.

Misalnya, sistem pengenalan wajah mungkin diinstruksikan, "Alis ada di atas mata," atau "Kumis ada di bawah hidung. Kumis ada di atas dan/atau di samping mulut." Aturan pramuat dapat membuat pelatihan lebih cepat dan membuat model lebih kuat lebih cepat. Tapi itu juga membangun asumsi tentang sifat masalah, yang mungkin terbukti tidak relevan dan tidak membantu atau tidak benar dan kontraproduktif, membuat keputusan tentang apa, jika ada, aturan untuk membangun sangat penting.

Selanjutnya, asumsi yang dibuat orang ketika algoritma pelatihan menyebabkan jaringan saraf memperkuat bias. Kumpulan data bias merupakan tantangan berkelanjutan dalam sistem pelatihan yang menemukan jawaban sendiri dengan mengenali pola dalam data. Jika data memberi inputan algoritme tidak netral -- dan hampir tidak ada data -- mesin menyebarkan bias.

Jenis jaringan saraf

Jaringan saraf kadang-kadang dijelaskan dalam hal kedalamannya, termasuk berapa banyak lapisan yang mereka miliki antara input dan output, atau model yang disebut lapisan tersembunyi. Inilah sebabnya mengapa istilah jaringan saraf digunakan hampir secara sinonim dengan pembelajaran mendalam (*deep learning*). Mereka juga dapat digambarkan dengan jumlah node tersembunyi yang dimiliki model atau dalam hal berapa banyak input dan output yang dimiliki

setiap node. Variasi pada desain jaringan saraf klasik memungkinkan berbagai bentuk penyebaran informasi ke depan dan ke belakang di antara tingkatan.

Jenis khusus jaringan saraf tiruan meliputi:

Feed-forward neural networks: salah satu varian paling sederhana dari jaringan saraf. Mereka melewatkan informasi dalam satu arah, melalui berbagai node input, hingga sampai ke node output. Jaringan mungkin atau mungkin tidak memiliki lapisan simpul tersembunyi, membuat fungsinya lebih dapat ditafsirkan. Itu disiapkan untuk memproses sejumlah besar kebisingan. Jenis model komputasi JST ini digunakan dalam teknologi seperti pengenalan wajah dan visi komputer.

Recurrent neural networks: lebih kompleks. Mereka menyimpan output dari node pemrosesan dan memasukkan hasilnya kembali ke dalam model. Ini adalah bagaimana model dikatakan belajar memprediksi hasil dari suatu lapisan. Setiap node dalam model RNN bertindak sebagai sel memori, melanjutkan komputasi dan implementasi operasi. Jaringan saraf ini dimulai dengan propagasi depan yang sama dengan jaringan feed-forward, tetapi kemudian terus mengingat semua informasi yang diproses untuk digunakan kembali di masa mendatang. Jika prediksi jaringan salah, maka sistem belajar sendiri dan terus bekerja menuju prediksi yang benar selama backpropagation. Jenis ANN ini sering digunakan dalam konversi text-to-speech.

Convolutional neural networks: salah satu model paling populer yang digunakan saat ini. Model komputasi jaringan saraf ini menggunakan variasi perceptron multilayer dan berisi satu atau lebih lapisan konvolusi yang dapat terhubung seluruhnya atau digabungkan. Lapisan convolutional ini membuat peta fitur yang merekam wilayah gambar yang akhirnya dipecah menjadi persegi panjang dan dikirim untuk nonlinier. Model CNN sangat populer di bidang pengenalan gambar; itu telah digunakan di banyak aplikasi AI paling canggih, termasuk pengenalan wajah, digitalisasi teks, dan pemrosesan bahasa alami. Kegunaan lain termasuk deteksi parafrase, pemrosesan sinyal dan klasifikasi gambar.

Deconvolutional neural networks: memanfaatkan proses model CNN terbalik. Mereka bertujuan untuk menemukan fitur atau sinyal yang hilang yang mungkin awalnya dianggap tidak penting untuk tugas sistem CNN. Model jaringan ini dapat digunakan dalam sintesis dan analisis citra.

Modular neural networks: berisi beberapa jaringan saraf yang bekerja secara terpisah satu sama lain. Jaringan tidak berkomunikasi atau mengganggu aktivitas satu sama lain selama proses komputasi. Akibatnya, proses komputasi yang kompleks atau besar dapat dilakukan lebih efisien.

Kelebihan jaringan syaraf tiruan antara lain:

- ✓ Kemampuan pemrosesan paralel berarti jaringan dapat melakukan lebih dari satu pekerjaan pada satu waktu.
- ✓ Informasi disimpan di seluruh jaringan, bukan hanya database.
- ✓ Kemampuan untuk belajar dan memodelkan hubungan yang kompleks dan nonlinier membantu memodelkan hubungan kehidupan nyata antara input dan output.
- ✓ Toleransi kesalahan berarti kerusakan satu atau lebih sel JST tidak akan menghentikan pembangkitan output.
- ✓ Korupsi bertahap berarti jaringan akan perlahan menurun seiring waktu, alih-alih masalah menghancurkan jaringan secara instan.
- ✓ Kemampuan untuk menghasilkan output dengan pengetahuan yang tidak lengkap dengan hilangnya kinerja didasarkan pada betapa pentingnya informasi yang hilang itu.
- ✓ Tidak ada batasan yang ditempatkan pada variabel input, seperti bagaimana mereka harus didistribusikan.

- ✓ Pembelajaran mesin berarti JST dapat belajar dari peristiwa dan membuat keputusan berdasarkan pengamatan.
- ✓ Kemampuan untuk mempelajari hubungan tersembunyi dalam data tanpa memerintahkan hubungan tetap apa pun berarti JST dapat memodelkan data yang sangat fluktuatif dan varians non-konstan dengan lebih baik.
- ✓ Kemampuan untuk menggeneralisasi dan menyimpulkan hubungan yang tidak terlihat pada data yang tidak terlihat berarti JST dapat memprediksi output dari data yang tidak terlihat.

Kekurangan jaringan saraf tiruan antara lain:

- ➢ Kurangnya aturan untuk menentukan struktur jaringan yang tepat berarti arsitektur jaringan saraf tiruan yang tepat hanya dapat ditemukan melalui trial and error dan pengalaman.
- ➢ Persyaratan prosesor dengan kemampuan pemrosesan paralel membuat jaringan saraf bergantung pada perangkat keras.
- ➢ Jaringan bekerja dengan informasi numerik, oleh karena itu semua masalah harus diterjemahkan ke dalam nilai numerik sebelum dapat disajikan ke JST.
- ➢ Kurangnya penjelasan di balik solusi probing adalah salah satu kelemahan terbesar dalam JST.
- ➢ Ketidakmampuan untuk menjelaskan mengapa atau bagaimana di balik solusi menghasilkan kurangnya kepercayaan pada jaringan.

Aplikasi jaringan saraf tiruan

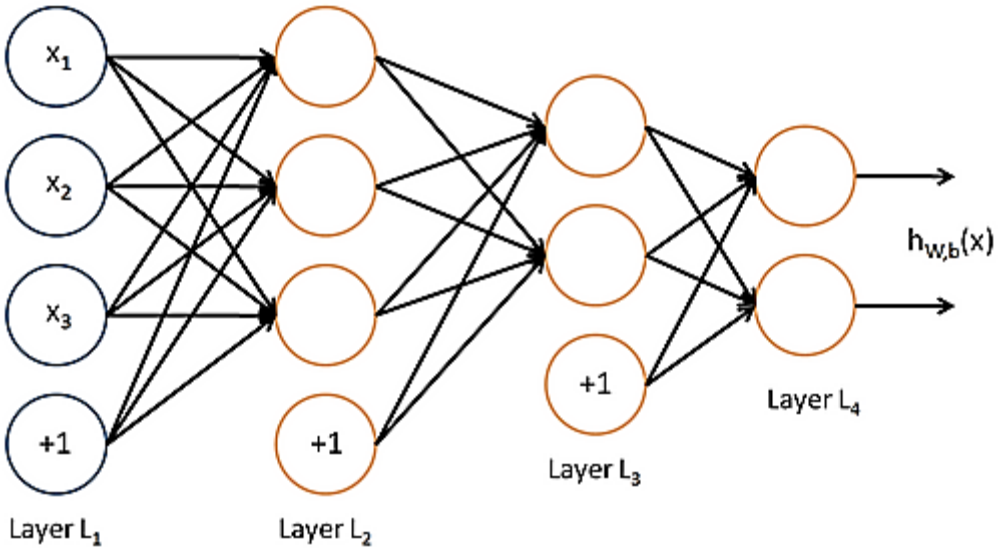
Pengenalan gambar adalah salah satu area pertama di mana jaringan saraf berhasil diterapkan, tetapi penggunaan teknologi telah berkembang ke lebih banyak area, termasuk:

- ➢ **Chatbot**
- ➢ **Pemrosesan bahasa alami, terjemahan, dan generasi Bahasa [Natural language processing, translation and language generation]**
- ➢ **Prediksi pasar saham [Stock market prediction]**
- ➢ **Perencanaan dan pengoptimalan rute pengemudi pengiriman [Delivery driver route planning and optimization]**
- ➢ **Penemuan dan pengembangan obat [Drug discovery and development]**

Berdasarkan alam, jaringan saraf adalah representasi yang biasa kita buat dari otak: neuron saling berhubungan dengan neuron lain yang membentuk jaringan. Sebuah informasi sederhana melintas di banyak dari mereka sebelum menjadi hal yang nyata, seperti "gerakkan tangan untuk mengambil pensil ini".

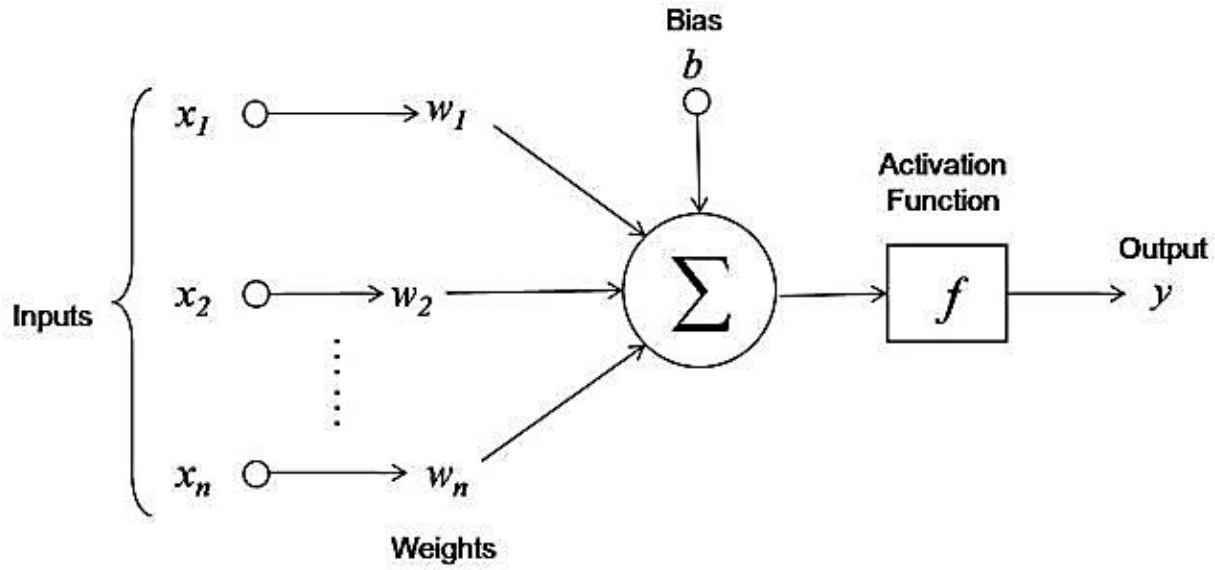
Pengoperasian jaringan saraf lengkap sangat mudah: satu memasukkan variabel sebagai input (misalnya gambar jika jaringan saraf seharusnya memberi tahu apa yang ada di gambar), dan setelah beberapa perhitungan, output dikembalikan (mengikuti contoh pertama, memberikan gambar kucing harus mengembalikan kata "kucing").

Perlu diketahui bahwa jaringan syaraf tiruan biasanya diletakkan pada kolom, sehingga neuron kolom n hanya dapat dihubungkan dengan neuron dari kolom n-1 dan n+1. Ada beberapa jenis jaringan yang menggunakan arsitektur berbeda, tetapi kami akan fokus pada yang paling sederhana untuk saat ini.



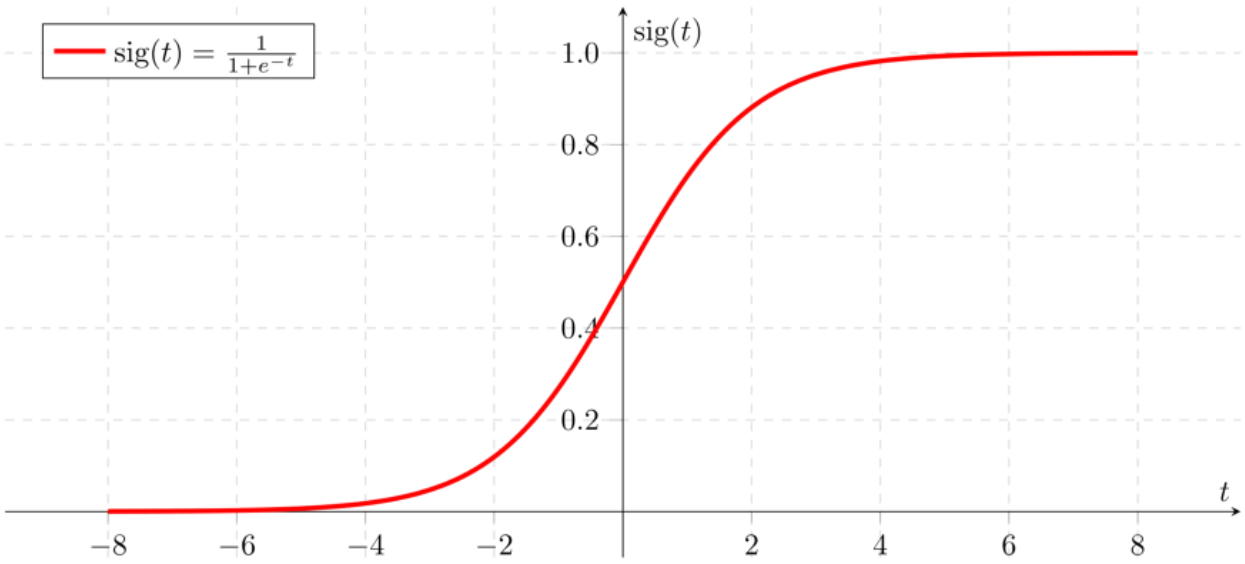
Jaringan saraf biasanya dapat dibaca dari kiri ke kanan. Di sini, lapisan pertama adalah lapisan di mana input dimasukkan. Ada 2 lapisan internal (disebut lapisan tersembunyi) yang melakukan perhitungan, dan satu lapisan terakhir yang berisi semua kemungkinan keluaran. Jangan repot-repot dengan "+1" di bagian bawah setiap kolom. Itu adalah sesuatu yang disebut "bias" dan kita akan membicarakannya nanti.

Apa yang dilakukan neuron:



Pertama, ia menjumlahkan nilai setiap neuron dari kolom sebelumnya yang terhubung dengannya. Pada Gambar 2 terdapat 3 input (x1, x2, x3) yang masuk ke neuron, sehingga 3 neuron kolom sebelumnya terhubung dengan neuron kita.
Nilai ini dikalikan, sebelum ditambahkan, dengan variabel lain yang disebut “berat” (w1, w2, w3) yang menentukan hubungan antara dua neuron. Setiap sambungan neuron memiliki bobotnya masing-masing, dan hanya nilai-nilai itulah yang akan dimodifikasi selama proses pembelajaran.

Selain itu, nilai bias dapat ditambahkan ke nilai total yang dihitung. Ini bukan nilai yang berasal dari neuron tertentu dan dipilih sebelum fase pembelajaran, tetapi dapat berguna untuk jaringan.
Setelah semua penjumlahan itu, neuron akhirnya menerapkan fungsi yang disebut "fungsi aktivasi" ke nilai yang diperoleh.



Fungsi aktivasi yang disebut biasanya berfungsi untuk mengubah nilai total yang dihitung sebelumnya ke angka antara 0 dan 1 (dilakukan misalnya oleh fungsi sigmoid yang ditunjukkan oleh Gambar 3). Fungsi lain ada dan dapat mengubah batas fungsi kita, tetapi tetap memiliki tujuan yang sama untuk membatasi nilai.

Itu saja yang dilakukan neuron! Ambil semua nilai dari neuron yang terhubung dikalikan dengan bobotnya masing-masing, tambahkan, dan terapkan fungsi aktivasi. Kemudian, neuron siap mengirim nilai barunya ke neuron lain.

Setelah setiap neuron kolom melakukannya, jaringan saraf lolos ke kolom berikutnya. Pada akhirnya, nilai-nilai terakhir yang diperoleh harus menjadi salah satu yang dapat digunakan untuk menentukan output yang diinginkan.

Sekarang setelah kita memahami apa yang dilakukan neuron, kita mungkin dapat membuat jaringan apa pun yang kita inginkan. Namun, ada operasi lain yang harus diterapkan untuk membuat jaringan saraf belajar.

Bagaimana jaringan saraf belajar?

Ya, membuat variabel dan membuatnya berinteraksi satu sama lain memang bagus, tetapi itu tidak cukup untuk membuat seluruh jaringan saraf belajar dengan sendirinya. Kita perlu menyiapkan banyak data untuk diberikan ke jaringan kita. Data tersebut termasuk input dan output yang diharapkan dari jaringan saraf.

Mari kita lihat bagaimana proses pembelajaran bekerja:

Pertama-tama, ingat bahwa ketika input diberikan ke jaringan saraf, ia mengembalikan output. Pada percobaan pertama, ia tidak bisa mendapatkan output yang tepat dengan sendirinya (kecuali dengan keberuntungan) dan itulah sebabnya, selama fase pembelajaran, setiap input datang dengan labelnya, menjelaskan output apa yang seharusnya ditebak oleh jaringan saraf. Jika pilihannya bagus, parameter aktual disimpan dan input berikutnya diberikan. Namun, jika output yang diperoleh tidak sesuai dengan label, bobot diubah. Itulah satu-satunya variabel yang dapat diubah selama fase pembelajaran. Proses ini dapat dibayangkan sebagai beberapa tombol, yang berubah menjadi kemungkinan yang berbeda setiap kali input tidak ditebak dengan benar.

Untuk menentukan bobot mana yang lebih baik untuk dimodifikasi, proses tertentu, yang disebut “backpropagation” dilakukan. Kami tidak akan berlama-lama tentang itu, karena jaringan saraf yang akan kami bangun tidak menggunakan proses yang tepat ini, tetapi terdiri dari kembali ke jaringan saraf dan memeriksa setiap koneksi untuk memeriksa bagaimana output akan berperilaku sesuai dengan perubahan pada beratnya.

Akhirnya, ada parameter terakhir yang harus diketahui untuk dapat mengontrol cara jaringan saraf belajar: "laju pembelajaran". Namanya mengatakan itu semua, nilai baru ini menentukan kecepatan apa yang akan dipelajari jaringan saraf, atau lebih khusus lagi bagaimana ia akan mengubah bobot, sedikit demi sedikit atau dengan langkah yang lebih besar. 1 umumnya merupakan nilai yang baik untuk parameter tersebut.

Perceptron

Oke kita sudah tahu dasar-dasarnya, mari kita cek tentang neural network yang akan kita buat. Yang dijelaskan di sini disebut Perceptron dan merupakan jaringan saraf pertama yang pernah dibuat. Terdiri dari 2 neuron pada kolom input dan 1 neuron pada kolom output. Konfigurasi ini memungkinkan untuk membuat classifier sederhana untuk membedakan 2 grup. Untuk lebih memahami kemungkinan dan batasannya, mari kita lihat contoh singkat (yang tidak terlalu menarik kecuali untuk dipahami):

Katakanlah Anda ingin jaringan saraf Anda dapat mengembalikan output sesuai dengan aturan "inklusif atau".

jika A benar dan B benar, maka A atau B benar.
jika A benar dan B salah, maka A atau B benar.
jika A salah dan B benar, maka A atau B benar.
jika A salah dan B salah, maka A atau B salah.

Jika Anda mengganti "benar" dengan 1 dan "salah" dengan 0 dan menempatkan 4 kemungkinan sebagai titik dengan koordinat pada sebuah rencana, maka Anda menyadari bahwa dua kelompok terakhir "salah" dan "benar" dapat dipisahkan oleh a garis tunggal. Inilah yang dapat dilakukan oleh Perceptron.

Di sisi lain, jika kita memeriksa kasus "eksklusif atau" (di mana kasus "benar atau benar" (poin (1,1)) salah), maka kita dapat melihat bahwa garis sederhana tidak dapat memisahkan dua kelompok, dan Perceptron tidak mampu menangani masalah ini. Jadi, Perceptron memang bukan jaringan saraf yang sangat efisien, tetapi mudah dibuat dan mungkin masih berguna sebagai pengklasifikasi.

Pikirkan masalah regresi linier yang memiliki konsep fungsi kerugian. Sebuah jaringan saraf mengasah pada jawaban yang benar untuk suatu masalah dengan meminimalkan fungsi kerugian. Misalkan kita memiliki persamaan linier sederhana ini: y = mx + b. Ini memprediksi beberapa nilai y yang diberikan nilai x.

Model prediktif tidak selalu 100% benar. Ukuran seberapa salah itu adalah kerugiannya. Tujuan dari pembelajaran mesin itu untuk mengambil satu set pelatihan untuk meminimalkan fungsi kerugian. Itu benar dengan regresi linier, jaringan saraf, dan algoritma ML lainnya.
Sebagai contoh, misalkan m = 2, x = 3, dan b = 2. Maka nilai prediksi kita dari y = 2 * 3 + 2 = 8. Tetapi nilai pengamatan aktual kita adalah 10. Jadi kerugiannya adalah 10 – 8 = 2.

Perceptron adalah objek yang mengambil input biner dan mengeluarkan output biner. Ini menggunakan jumlah tertimbang dan ambang batas untuk memutuskan apakah hasilnya harus ya (1) atau tidak (0).
Misalnya, Anda ingin pergi ke Prancis tetapi hanya jika:
x1 -> Tiket pesawat kurang dari $1.000.
x2 -> Pacar atau pacar Anda bisa pergi dengan Anda.
Anda mewakili keputusan ini dengan vektor sederhana dari kemungkinan input ini:
(1,0), (0,1), (1,1), dan (0,0).

Dalam kasus pertama (1,0) tiketnya > 1.000 dan pacar Anda tidak bisa pergi bersama Anda.
Anda memberi bobot pada masing-masing dari dua perhitungan ini. Misalnya, jika Anda memiliki anggaran dan biaya penting, berikan bobot w1=4. Dan apakah pasangan Anda bisa pergi atau tidak, itu tidak terlalu penting. Jadi beri bobot w2=3.

Jadi Anda memiliki fungsi ini untuk Pergi ke Prancis:
(x1 * w1) + (x2 * w2) = (x1 * 4) + (x2 * 3) > beberapa threshold/ambang, b, katakanlah, 4.
Kami memindahkan b ke sisi lain dan menulis:
Jika (x1 * 4) + (x2 * 3) -4 > 0 lalu Pergi ke Prancis (yaitu, perceptron mengatakan 1)-
Kemudian masukkan vektor ke dalam persamaan. Jelas jika tiketnya > $1.000 dan jika pacar Anda tidak bisa pergi (0,0) maka Anda tidak akan melakukan perjalanan, karena

(0 * 3) + (0 * 4) – 4 jelas < 0.

Jika tiketnya murah tetapi Anda pergi sendiri, maka pergilah:

(1 * 4) + (0 * 3) – 4 = 0 yang tidak lebih besar dari 0.