

Chapter 1

Fəsil 1. Maşın Öyrənməsi Landşaftı

Çox yaxın keçmişdə, telefonunuzu götürüb evə gedəcəyinizi soruşsaydırınız, o, sizi görməzlilikdən gələrdi və insanlar əqli vəziyyətinizi şübhə altına alardılar. Lakin maşın öyrənməsi artıq elmi fantastika deyil: milyardlarla insan hər gün ondan istifadə edir. Əslində, bu, bəzi ixtisaslaşmış tətbiqlərdə, məsələn, optik xarakter tanıma (OCR) kimi, onilliklərdir mövcuddur. Gerçəkdən həyatları yaxşılaşdırın və yüz milyonlarla insanın həyatını dəyişdirən ilk maşın öyrənmə tətbiqi 1990-ci illərdə dünyani fəth etdi: spam filtri. Bu, tam olaraq özünüdərk edən bir robot deyil, amma texniki baxımdan maşın öyrənməsi olaraq qiymətləndirilir: o, o qədər yaxşı öyrənib ki, artıq çox nadir hallarda elektron poçtları spam olaraq işarəlemek lazım gelir. Ardınca yüzlərlə maşın öyrənməsi tətbiqi geldi və hazırda sizin müntəzəm istifadə etdiyiniz yüzlərlə məhsul və xüsusiyyətə sükutla güc verir: səsli əmr, avtomatik tərcümə, şəkil axtarışı, məhsul tövsiyələri və daha çox.

Maşın öyrənməsi haradan başlayır və harada bitir? Maşın nəyisə öyrənmək nə deməkdir? Əgər mən bütün Vikipediya məqalələrinin surətini yükləsəm, kompüterim həqiqətən bir şey öyrənmiş olurmu? Birdən daha ağıllı olarmı? Bu fəsildə maşın öyrənməsinin nə olduğunu və niyə ondan istifadə etmək istəyə biləcəyinizi aydınlaşdırmağa başlayacağam.

Sonra, maşın öyrənməsi qitəsini kəşf etməyə başlamazdan əvvəl xəritəyə baxacaq və əsas bölgələri və ən əlamətdar mənzərələri öyrənəcəyik: nəzarətli və nəzarətsiz öyrənmə və onların variantları, onlayn və kütləvi öyrənmə, nümunə əsaslı və model əsaslı öyrənmə. Sonra tipik bir maşın öyrənməsi layihəsinin iş axınına baxacaq, qarşılaşa biləcəyiniz əsas çətinlikləri müzakirə edəcəyik və bir maşın öyrənmə sistemini necə qiymətləndirib dəqiqləşdirəcəyimizi əhatə edəcəyik.

Bu fəsil hər bir məlumat mütəxəssisinin ürəkdən bilməli olduğu çox sayıda əsas anlayışı (və jargonları) təqdim edir. Bu, yüksək səviyyəli bir baxış olacaq (yalnız bu fəsildə kod yoxdur), sadədir, amma məqsədim, kitabın qalan hissəsinə keçməzdən əvvəl hər şeyin sizə tamamilə aydın olmasını təmin etməkdir. Buna görə də, bir fincan kofe götürün və başlayaq!

İPUCU

Əgər maşın öyrənməsinin əsasları ilə artıq tanışsınızsa, birbaşa Fəsil 2-yə keçmək istəyə bilərsiniz. Əgər əmin deyilsinizsə, fəsilin sonunda verilmiş bütün suallara cavab verməyə çalışın, sonra davam edin.

Maşın Öyrənməsi Nədir?

Maşın öyrənməsi, kompüterləri verilənlərdən öyrənə bilməsi üçün programlaşdırma elmi (və sənəti)dir.

Bir qədər daha ümumi bir tərif:

[Maşın öyrənməsi] kompüterlərə açıq şəkildə programlaşdırılmışdan öyrənmə qabiliyyəti verən öyrənmə sahəsidir.

—Arthur Samuel, 1959

Və mühəndislik yönümlü bir tərif:

Bir kompüter programı, bəzi tapşırıq T və bəzi performans ölçüsü P ilə əlaqədar olaraq E təcrübəsindən öyrənir, əgər T üzərindəki performansı, P tərəfindən ölçülən, E təcrübəsi ilə yaxşılaşırsa, o zaman bu programın öyrəndiyi deyilir.

—Tom Mitchell, 1997

Sizin spam filtriniz bir maşın öyrənməsi programıdır ki, istifadəçilər tərəfindən işarələnmiş spam emailləri və adı emaillər (spam olmayan, "ham" da adlanır) nümunələrini istifadə edərək spamı ayırd etməyi öyrənə bilər. Sistem öyrənmək üçün istifadə etdiyi nümunələrə təlim dəstəsi (training set) deyilir. Hər bir təlim nümunəsinə təlim nümunəsi (training instance) və ya nümunə (sample) deyilir. Maşın öyrənmə sisteminde öyrənən və proqnozlar verən hissəyə model deyilir. Neyron şəbəkələri və təsadüfi meşələr (random forests) modellərin nümunələridir.

Bu halda, tapşırıq T yeni emailləri spam olaraq işaretləmək, təcrübə E təlim verilənləridir, və performans ölçüsü P təyin edilməlidir; məsələn, düzgün təsnif edilmiş emaillərin nisbətini istifadə edə bilərsiniz. Bu xüsusi performans ölçüsü dəqiqlik (accuracy) adlanır və çox vaxt təsnifat tapşırıqlarında istifadə olunur.

Əgər sadəcə bütün Vikipediya məqalələrinin surətini yüklesəniz, kompüteriniz çox daha çox verilənlərə malik olur, amma o, heç bir tapşırıqda birdən yaxşılaşır. Bu, maşın öyrənməsi deyil.

Maşın Öyrənməsindən Niyə İstifadə Edirik?

Təsəvvür edin ki, spam filtrini ənənəvi programlaşdırma üsulları ilə necə yazardınız (Şəkil 1-1):

1. Əvvəlcə spamin nə cür olduğunu aşadırardınız. Bəlkə, bəzi sözlər və ifadələr (məsələn, "4U", "kredit kartı", "pulsuz" və "möhtəşəm") mövzu xəttində çox tez-tez istifadə olunduğunu görərsiniz. Ola bilər ki, göndəricinin adı, emailin mətni və emaillin digər hissələrində də bəzi başqa nümunələri fərqli edərsiniz.
2. Diqqət etdiyiniz hər bir nümunə üçün bir aşkarlama alqoritması yazardınız və programınız bu nümunələrdən bir neçə dənəsi aşkar edildikdə emailləri spam olaraq işaretləyərdi.
3. Programınızı sınadandan keçirər və 1 və 2-ci addımları təkrarlayardınız, ta ki program kifayət qədər yaxşı olub istifadəyə verilmək üçün hazır olana qədər.

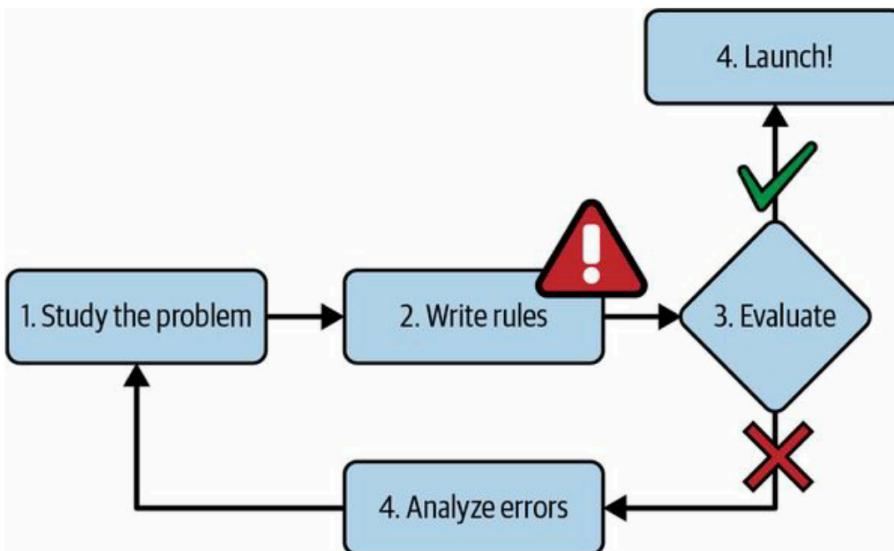


Figure 1-1. The traditional approach

Çünki problem çətin olduğu üçün programınız ehtimal ki, uzun və mürəkkəb qaydalar siyahısına çevriləcək — bu da onu idarə etməyi çətinləşdirəcək.

Buna qarşı, maşın öyrənməsi üsullarına əsaslanan bir spam filtrinin özü, spam nümunələri ilə ham nümunələri arasında sözlərin qeyri-adi tez-tez təkrarlanan nümunələrini aşkar edərək hansı sözlərin və ifadələrin spamın yaxşı proqnozlaşdırıcıları olduğunu avtomatik olaraq öyrənir (Şəkil 1-2). Program daha qısa, idarə edilməsi daha asan və çox güman ki, daha dəqiqdır.

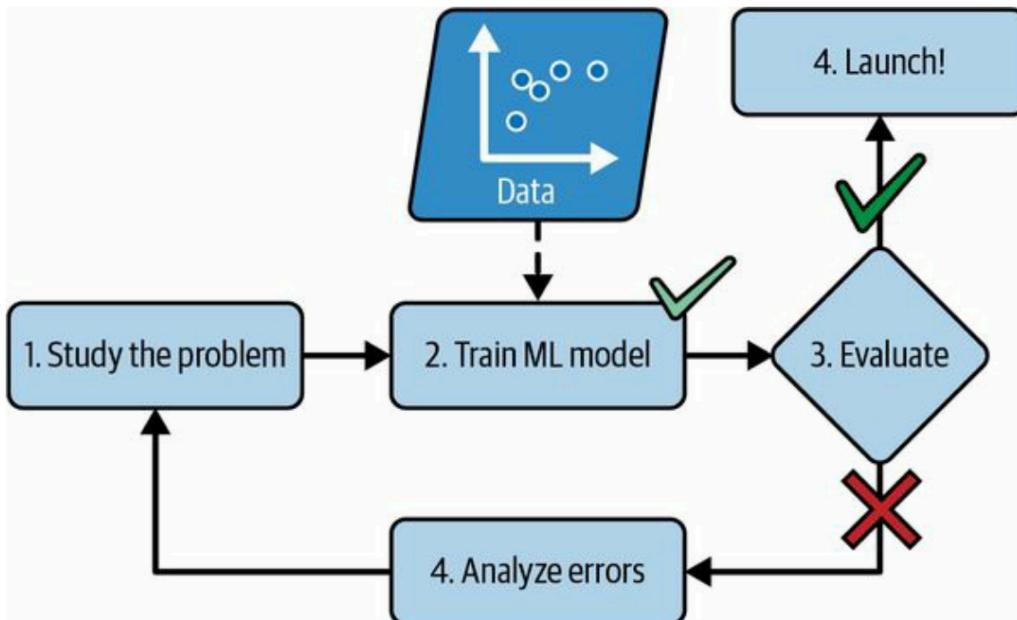


Figure 1-2. The machine learning approach

Əgər spam göndərənlər “4U” ifadəsinin bütün emaillərində bloklandığını görsələr, bəlkə də “For U” yazmağa başlaya bilərlər. Ənənəvi programlaşdırma üsullarına əsaslanan bir spam filtrinin “For U” emaillərini işaretləmək üçün yenilənməsi lazımlı olacaq. Əgər spam

göndərənlər spam filtrinizi daim aşib keçməyə davam etsələr, siz sonsuz olaraq yeni qaydalar yazmaq məcburiyyətində qalacaqsınız.

Buna qarşı, maşın öyrənməsi üsullarına əsaslanan bir spam filtrinin özü “For U”nun istifadəçilər tərəfindən işarələnmiş spamlarda qeyri-adi şəkildə tez-tez rast gəlinməyə başladığını avtomatik olaraq görür və sizin müdaxiləniz olmadan onları işarələməyə başlayır (Şəkil 1-3).

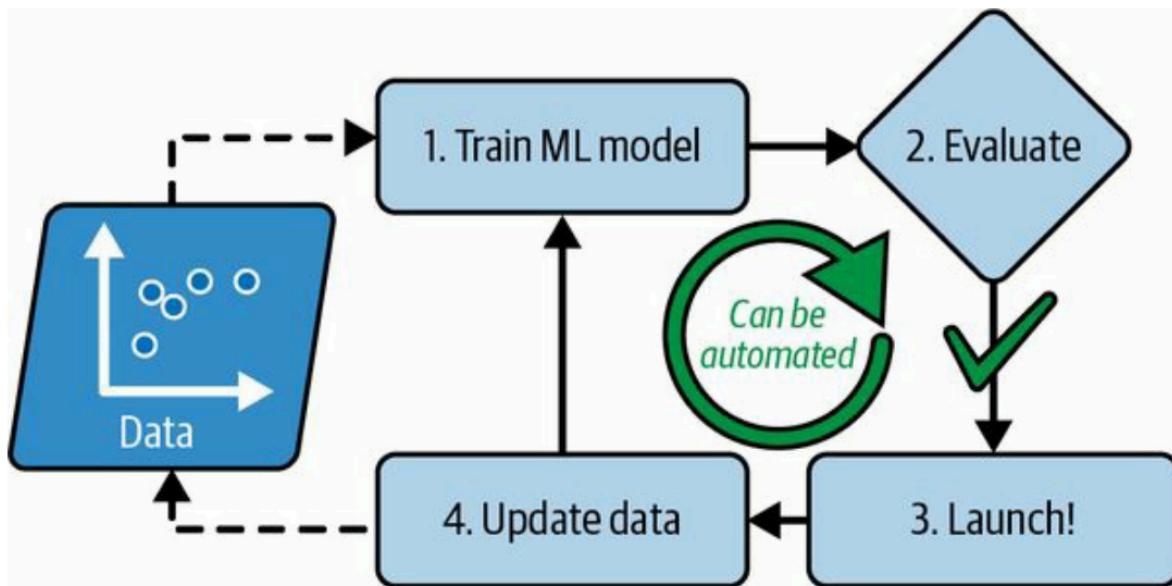


Figure 1-3. Automatically adapting to change

Maşın öyrənməsinin parlaq olduğu başqa bir sahə, ya ənənəvi yanaşmalar üçün çox mürəkkəb olan, ya da heç bir məlum alqoritması olmayan problemlərdir. Məsələn, səs tanıma məsələsini nəzərdən keçirin. Təsəvvür edin ki, sadə başlamaq istəyirsiniz və “bir” və “iki” sözlərini fərqləndirə bilən bir program yazmaq istəyirsiniz. Bəlkə də “iki” sözünün yüksək tonlu bir səslə (“T”) başladığını fərqli edərsiniz, buna görə də yüksək tonlu səs intensivliyini ölçən bir alqoritm yazıb onu birləşdirmək olar — amma aydın şəkildə bu üsul minlərlə sözün, milyonlarla çox fərqli insan tərəfindən, səs-küylü mühitlərdə və onlarla dildə səsləndirildiyi bir vəziyyətdə işləməyəcək. Ən yaxşı həll (bu gün üçün ən azı) hər bir söz üçün bir çox nümunə qeydləri verildikdə öz-özünə öyrənən bir alqoritm yazmaqdır.

Nəhayət, maşın öyrənməsi insanlara öyrənməkdə kömək edə bilər (Şəkil 1-4). ML modelləri öyrəndiklərini görmək üçün yoxlanıla bilər (baxmayaraq ki, bəzi modellər üçün bu çətin ola bilər). Məsələn, bir spam filtrinin kifayət qədər spamları öyrədildikdən sonra, onu asanlıqla yoxlayıb spamin ən yaxşı proqnozlaşdırıcıları olduğuna inandığı sözlər və söz birləşmələrinin siyahısını ortaya çıxara bilərsiniz. Bəzən bu, gözlənilməz əlaqələri və ya yeni tendensiyaları ortaya çıxaraq və bununla da problemi daha yaxşı başa düşməyə səbəb olacaq. Böyük miqdarda verilənlərə daxil olmaq və gizli nümunələri tapmaq məlumat mədənçiliyi (data mining) adlanır və maşın öyrənməsi bu sahədə mükəmməldir.

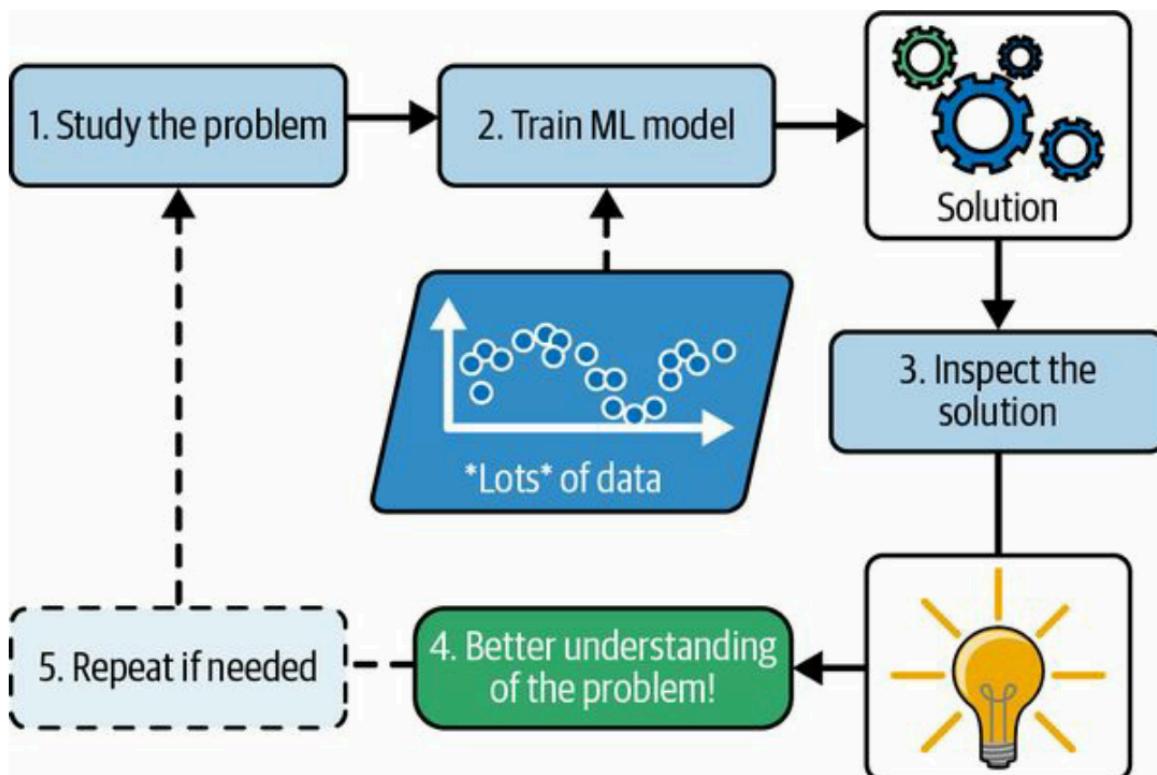


Figure 1-4. Machine learning can help humans learn

Xülasə etmək üçün, maşın öyrənməsi aşağıdakı vəziyyətlərdə əladır:

- Mövcud həllərin çoxlu təkmilləşdirmə və ya uzun qaydalar siyahısı tələb etdiyi problemlər (maşın öyrənmə modeli tez-tez kodu sadələşdirir və ənənəvi yanaşmadan daha yaxşı nəticə verə bilər)
- Ənənəvi yanaşma tətbiq edildikdə yaxşı həll olmayan mürəkkəb problemlər (maşın öyrənmə texnikaları bəlkə də həll tapa bilər)
- Dəyişkən mühitlər (maşın öyrənmə sistemi asanlıqla yeni məlumatlarla yenidən öyrədilə bilər və həmişə aktual saxlanılır)
- Mürəkkəb problemlər və böyük həcmli məlumatlar haqqında məlumat əldə etmək

Tətbiq Nümunələri

Maşın öyrənmə tapşırıqlarının bəzi konkret nümunələrinə və onlarla mübarizə aparmaq üçün istifadə edilən texnikalara nəzər salaq:

- İstehsal xəttində məhsulların şəkillərini təhlil edərək onları avtomatik olaraq təsnifləşdirmək
 - Bu, adətən konvolyusiya neyron şəbəkələri (CNN) və ya bəzən transformerlərdən istifadə edilərək həyata keçirilən şəkil təsnifləşdirməsidir.
- Beyin müayinələrində şışləri aşkarlamaq
 - Bu, hər bir pikselin təsnifləşdirildiyi semantik şəkil seqmentləşdirməsidir (şışlərin dəqiq yerini və formasını müəyyən etmək üçün), adətən CNN-lər və ya transformerlərdən istifadə olunur.
- Xəbər məqalələrini avtomatik olaraq təsnifləşdirmək

- Bu, təbii dil emalı (NLP), daha konkret desək, mətn təsnifləşdirməsidir; təkrarlanan neyron şəbəkələr (RNN) və CNN-lər istifadə edilə bilər, lakin transformerlər daha yaxşı işləyir.
- Diskussiya forumlarında təhqiredici şərhləri avtomatik olaraq işarələmək
- Bu da eyni NLP alətləri ilə mətn təsnifləşdirməsidir.
- Uzun sənədləri avtomatik olaraq xülasə etmək
- Bu, mətn xülasəsi adlanan NLP filialıdır və yenə də eyni alətlərdən istifadə olunur.
- Çatbot və ya şəxsi köməkçi yaratmaq
 - Bu, təbii dili anlama (NLU) və sual-cavab modulları daxil olmaqla bir çox NLP komponentlərini əhatə edir.
- Şirkətinizin gələn ilki gəlirini müxtəlif performans metrikleri əsasında proqnozlaşdırmaq
 - Bu, xətti regresiya və ya polinom regresiya modeli kimi istənilən regresiya modeli ilə həll edilə bilən regresiya tapşırığıdır.
- Tətbiqinizi səsli komandalara reaksiya vermək üçün hazırlamaq
 - Bu, uzun və mürəkkəb ardıcılıqları olan audio nümunələri emal etməyi tələb edən səs tanımıdır; bu, adətən RNN-lər, CNN-lər və ya transformerlər vasitəsilə işlənir.
- Kredit kartı fırıldaqlığını aşkar etmək
 - Bu, anomaliya aşkarlanmasıdır və təcrid meşələri, Qaus qarışiq modelləri və ya autoencoderlərdən istifadə etməklə həll edilə bilər.
- Müştəriləri alışlarına əsasən seqmentləşdirmək və hər seqment üçün fərqli marketing strategiyası hazırlamaq
 - Bu, k-means, DBSCAN və daha çox istifadə edərək həyata keçirilə bilən klasterləşdirmədir.
- Mürəkkəb, yüksək ölçülü verilənlər toplusunu aydın və anlayışlı bir diaqramda təqdim etmək
 - Bu, tez-tez ölçülərin azaldılması texnikaları istifadə edilən verilənlərin vizuallaşdırılmasıdır.
- Müştərinin keçmiş alışlarına əsaslanaraq onun maraqlana biləcəyi məhsulu tövsiyə etmək
 - Bu, tövsiyə sistemidir; bir yanaşma keçmiş alışları (və müştəri haqqında digər məlumatları) süni neyron şəbəkəsinə daxil etməkdir və o, növbəti alışın ehtimalını göstərir.
- Oyunda ağıllı bir bot yaratmaq
 - Bu, tez-tez agentləri (bot kimi) vaxt keçdikcə mükafatlarını maksimuma çatdırmaq üçün hərəkətləri seçmək üçün öyrədən maşın öyrənməsinin bir qolu olan möhkəmləndirici öyrənmə (RL) vasitəsilə həll edilir.

Bu siyahını daha da davam etdirmək olar, lakin ümid edirəm ki, maşın öyrənmənin həll edə biləcəyi və hər bir tapşırıq üçün istifadə ediləcək texnikaların növləri barədə sizə bir təsəvvür verir.

Maşın Öyrənməsi Sistemlərinin Növləri

Maşın öyrənmə sistemlərinin müxtəlif növləri mövcuddur və onları geniş kateqoriyalara bölmək faydalıdır. Bunu aşağıdakı meyarlara əsasən edə bilərik:

- Təlim zamanı nə qədər nəzarət edildiklərinə görə (nəzarətli, nəzarətsiz, yarımnəzarətli, özünü-nəzarətli və digərləri)
- Dərhal öyrənə bilib-bilmədiklərinə görə (onlayn və ya paket öyrənmə)
- Yeni verilən nöqtələri məlum verilən nöqtələrlə müqayisə edərək işləmələrinə, yoxsa təlim verilənlərindəki naxışları müəyyən edərək elmi yanaşmada olduğu kimi proqnozlaşdırıcı model yaratmalarına görə (nümunə əsaslı və ya model əsaslı öyrənmə)

Bu meyarlar bir-birini istisna etmir; onları istənilən şəkildə birləşdirə bilərsiniz. Məsələn, müasir bir spam filtri insan tərəfindən təmin edilən spam və normal e-poçt nümunələri üzərində öyrədilən dərin neyron şəbəkə modeli ilə dərhal öyrənə bilər; bu, onu onlayn, model əsaslı, nəzarətli öyrənmə sistemi edir.

Gəlin bu meyarların hər birinə daha yaxından baxaq:

Təlim Nəzarəti

Maşın öyrənməsi sistemləri təlim zamanı aldıqları nəzarətin miqdarına və növünə görə təsnif edilə bilər. Bir çox kateqoriya var, lakin əsas olanları müzakirə edəcəyik: nəzarətli öyrənmə, nəzarətsiz öyrənmə, özünü-nəzarətli öyrənmə, yarımnəzarətli öyrənmə və möhkəmləndirici öyrənmə.

Nəzarətli Öyrənmə

Nəzarətli öyrənmədə, algoritmə təqdim etdiyiniz təlim dəstинə istənilən həllər, yəni etiketlər daxil edilir.

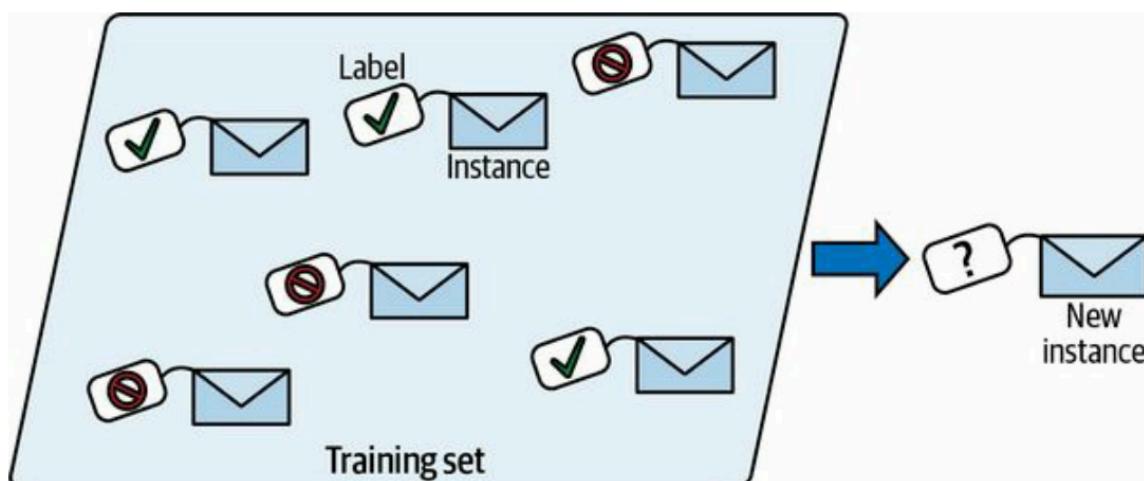


Figure 1-5. A labeled training set for spam classification (an example of supervised learning)

Tipik nəzarətli öyrənmə tapşırıqlarından biri təsnifatdır. Spam filtri bunun yaxşı bir nümunəsidir: o, bir çox nümunə e-poçtlarla (spam və ya normal) təlim edilir və yeni e-poçtları necə təsnif edəcəyini öyrənməlidir.

Başqa bir tipik tapşırıq isə verilən xüsusiyyətlərə (məsələn, yürüş, yaşı, markası və s.) əsasən avtomobilin qiyməti kimi bir hədəf ədədi dəyərini proqnozlaşdırmaqdır. Bu tip tapşırıq regressiya adlanır. Sistemi öyrətmək üçün ona avtomobillərin həm xüsusiyyətlərini, həm də hədəf dəyərlərini (yəni qiymətlərini) daxil edən bir çox nümunə verməlisiniz.

Qeyd edək ki, bəzi regressiya modelləri təsnifat üçün də istifadə edilə bilər və əksinə. Məsələn, logistik regressiya təsnifat üçün tez-tez istifadə olunur, çünkü o, müəyyən bir sinfə aid olma ehtimalına uyğun gələn bir dəyər çıxara bilər (məsələn, 20% spam olma ehtimalı).

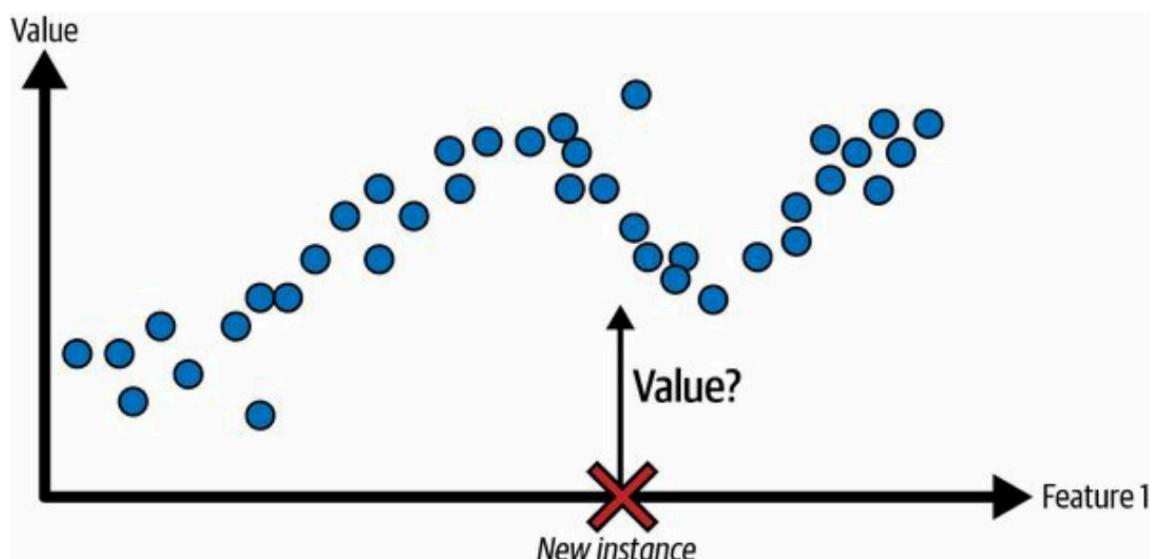


Figure 1-6. A regression problem: predict a value, given an input feature (there are usually multiple input features, and sometimes multiple output values)

Qeyd:

Supervised öyrənmədə “hədəf” və “etiket” sözləri ümumiyyətlə sinonim kimi qəbul edilir, lakin “hədəf” regressiya tapşırıqlarında, “etiket” isə təsnifat tapşırıqlarında daha çox istifadə olunur. Bundan başqa, xüsusiyyətlər bəzən “proqnozlaşdırıcı” və ya “atribut” adlanır. Bu terminlər fərdi nümunələrə (məsələn, “bu avtomobilin yürüş xüsusiyyəti 15,000-dir”) və ya bütün nümunələrə (məsələn, “yürüş xüsusiyyəti qiymətlə güclü əlaqəlidir”) istinad edə bilər.

Nəzarətsiz öyrənmə:

Nəzarətsiz öyrənmədə, təlim məlumatları etiketsiz olur (Şəkil 1-7). Sistem, müəllim olmadan öyrənməyə çalışır.

Məsələn, blogunuzun ziyarətçiləri haqqında çoxlu məlumatınız olduğunu təsəvvür edin. Bənzər ziyarətçilər qruplarını aşkar etmək üçün bir klasterləşdirmə alqoritmi işlətsəniz, heç bir mərhələdə alqoritmə hansı qrupla ziyarətçinin aid olduğunu bildirmirsəniz; bu əlaqələri özünüzün köməyiniz olmadan tapır. Məsələn, ziyarətçilərin 40%-nin komiks sevən və ümumiyyətlə dərsdən sonra blogunu oxuyan yeniyetmələr, 20%-nin isə elmi-fantastik janrını sevən və həftə sonları ziyarət edən böyükər olduğunu fərq edə bilər. Əgər iyerarxik klasterləşdirmə alqoritmi istifadə edirsənizsə, hər bir qrupu daha kiçik alt qruplara bölə bilər. Bu, hər qrupa uyğun yazılarınızı hədəfləməyinize kömək edə bilər.

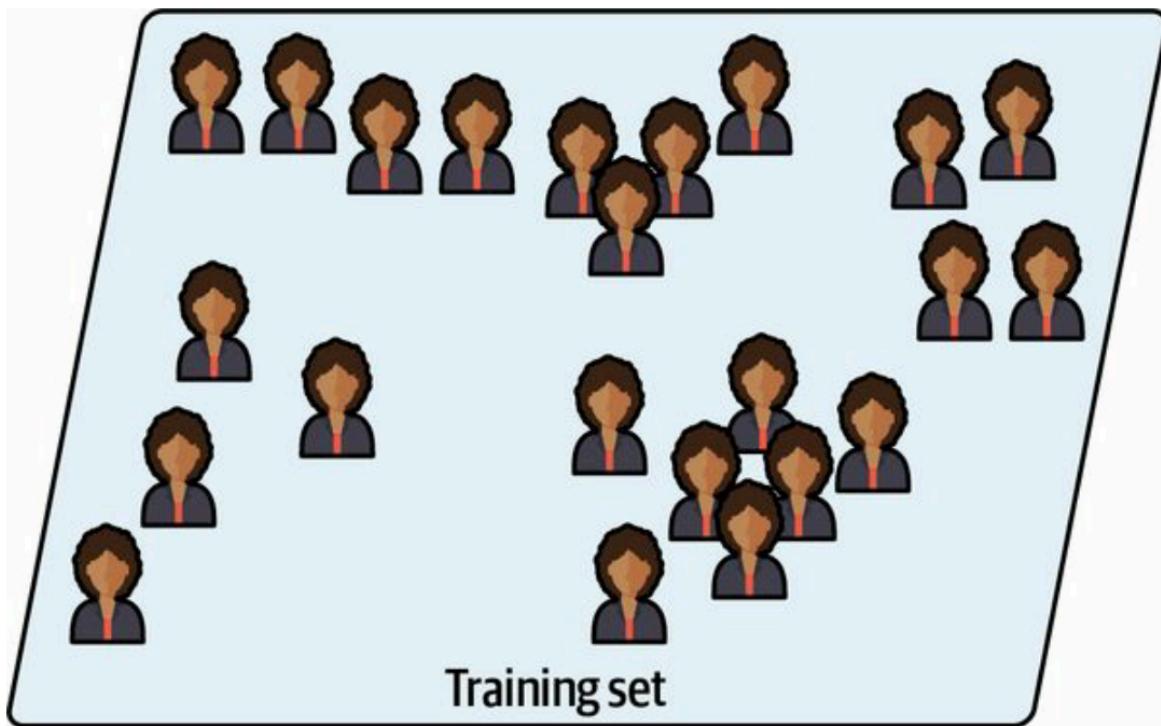


Figure 1-7. An unlabeled training set for unsupervised learning

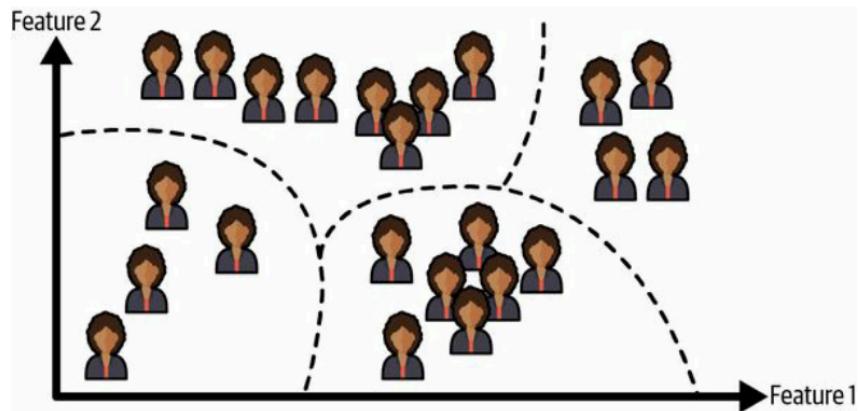


Figure 1-8. Clustering

Vizualizasiya alqoritmləri də nəzarətsiz öyrənmənin yaxşı nümunələridir: onlara çox miqdarda mürəkkəb və etiketsiz məlumat verirsınız və onlar məlumatınızı asanlıqla qrafik şəklində göstərə biləcəyiniz 2D və ya 3D təmsilini çıxarır (Şəkil 1-9). Bu alqoritmlər mümkün qədər çox strukturu qorumağa çalışır (məsələn, giriş məkanında ayrı olan klasterlərin vizualizasiya zamanı üst-üstə düşməməsinə çalışaraq) ki, məlumatın necə təşkil olunduğunu başa düşə biləsiniz və gözlənilməyən nümunələri müəyyən edə biləsiniz.

Bununla əlaqəli bir tapşırıq da ölçünün azaldılmasıdır və bu prosesdə məqsəd məlumatları çox az məlumat itkisi ilə sadələşdirməkdir. Bunu həyata keçirməyin bir yolu, bir neçə əlaqəli xüsusiyyəti birləşdirməkdir. Məsələn, bir avtomobilin yürüşü ilə yaşı arasında güclü bir əlaqə ola bilər, buna görə də ölçünün azaldılması alqoritmi onları avtomobilin köhnəlmə dərəcəsini təmsil edən bir xüsusiyyətdə birləşdirir. Bu prosesə "xüsusiyyət çıxarma" deyilir.

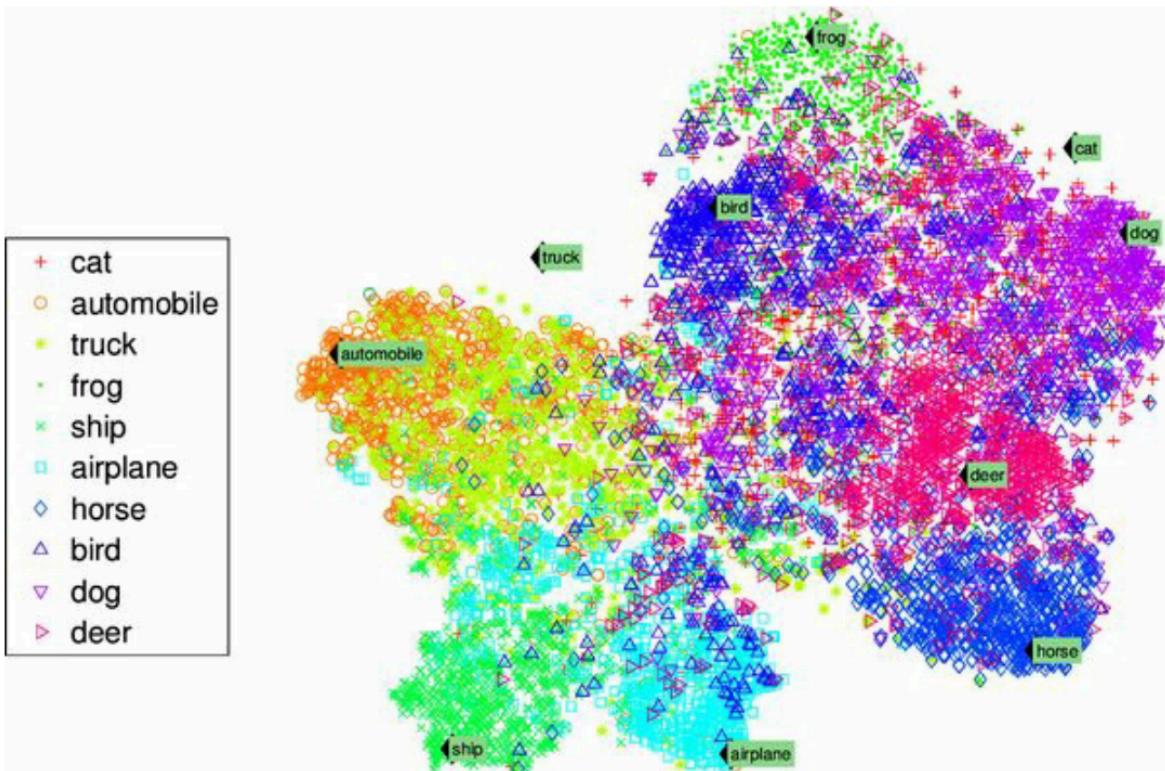


Figure 1-9. Example of a t-SNE visualization highlighting semantic clusters ²

Məsləhət

Məlumatları başqa bir maşın öyrənmə alqoritminə (məsələn, nəzarətli öyrənmə alqoritminə) verməzdən əvvəl ölçü azaldılması alqoritmi istifadə edərək təlim məlumatlarınızdakı ölçülərin sayını azaltmağa çalışmaq çox vaxt yaxşı bir fikir ola bilər. Bu, alqoritmin daha sürətli işləməsinə səbəb olacaq, məlumat daha az disk və yaddaş sahəsi tutacaq və bəzi hallarda nəticə daha yaxşı ola bilər.

Başqa bir mühüm nəzarətsiz tapşırıq anomaliya aşkarlanmasıdır. Məsələn, saxtakarlığın qarşısını almaq üçün qeyri-adi kredit kartı əməliyyatlarını aşkar etmək, istehsal qüsurlarını tutmaq və ya məlumat dəstini digər öyrənmə alqoritminə vermədən əvvəl avtomatik olaraq çıxıntıları çıxarmaq. Sistem təlim zamanı əsasən normal nümunələrlə tanış olur və onları tanımağı öyrənir; daha sonra yeni bir nümunə ilə qarşılaşdıqda, onun normal birinə bənzəyib-bənzəmədiyini və ya anomaliya olub-olmadığını müəyyən edə bilir (bax Şəkil 1-10). Çox bənzər bir tapşırıq isə yenilik aşkarlanmasıdır: bu, təlim dəstindəki bütün nümunələrdən fərqli görünən yeni nümunələri aşkar etməyə yönəlir. Bu, təlim dəstinin çox “təmiz” olmasını tələb edir, yəni alqoritmin aşkarlamasını istəmədiyiniz nümunələrdən azad olması lazımdır. Məsələn, əgər minlərlə it şəkiliniz varsa və onların 1%-i Çivava cinsinə aid olsa, yenilik aşkarlama alqoritmi yeni Çivava şəkillərini yenilik kimi qəbul etməməlidir. Digər tərəfdən, anomaliya aşkarlama alqoritmləri bu cins itləri digər itlərdən çox nadir və fərqli hesab edərək onları anomaliya kimi təsnif edə bilər (Çivava itlərinə hörmətsizlik olmasın).

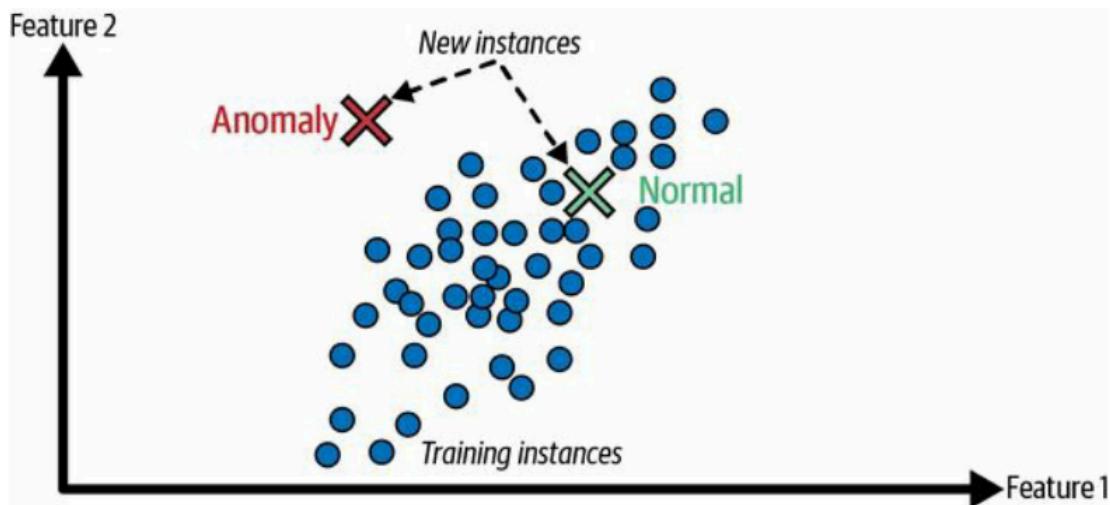


Figure 1-10. Anomaly detection

Nəhayət, digər bir ümumi nəzarətsiz təpşiriq assosiasiya qaydalarının öyrənilməsidir, burada məqsəd çoxlu məlumatı araşdırmaq və atributlar arasında maraqlı əlaqələri aşkar etməkdir. Məsələn, supermarketiniz olduğunu düşünək. Satış qeydlərinizdə bir assosiasiya qaydasını işlətmək, barbekü sousu və kartof çipsi alan insanların steyk də almağa meyilli olduğunu göstərə bilər. Buna görə, bu məhsulları bir-birinə yaxın yerləşdirmək istəyə bilərsiniz.

Yarım nəzarətli öyrənmə

Etiketləmə prosesi adətən çox vaxt tələb etdiyindən və maliyyətli olduğundan, çoxlu etiketlənməmiş nümunələrə və bir neçə etiketlənmiş nümunəyə sahib ola bilərsiniz. Bəzi alqoritmlər qismən etiketlənmiş məlumatlarla işləyə bilir. Bu, yarım nəzarətli öyrənmə adlanır (bax Şəkil 1-11).

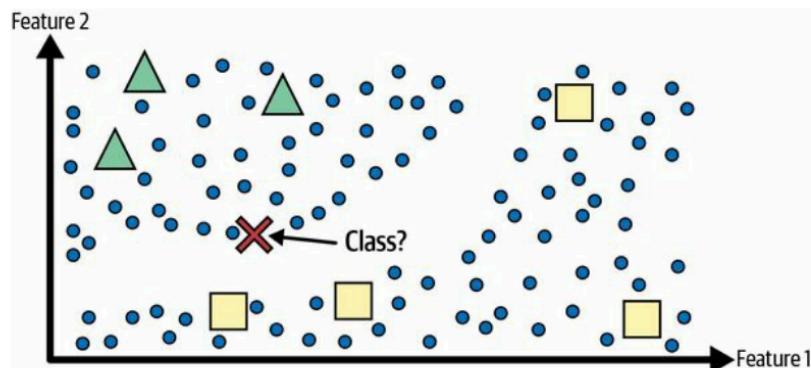


Figure 1-11. Semi-supervised learning with two classes (triangles and squares): the unlabeled examples (circles) help classify a new instance (the cross) into the triangle class rather than the square class, even though it is closer to the labeled squares

Bəzi foto-hostinq xidmətləri, məsələn Google Photos, bunun yaxşı nümunələridir. Bütün ailə fotolarınızı xidmətə yüklədikdən sonra, sistem avtomatik olaraq A şəxsin 1, 5 və 11-ci fotolarda, B şəxsin isə 2, 5 və 7-ci fotolarda görünüşünü tanıyır. Bu, alqoritmin nəzarətsiz hissəsidir (klasterləşdirmə). İndi isə sistemin ehtiyacı olan tək şey bu insanların kim olduğunu sizdən öyrənməkdir. Hər bir şəxs üçün sadəcə bir etiket əlavə edin və o, hər bir fotosəkildə hər kəsin adını müəyyən edə biləcək, bu isə fotoları axtarmaq üçün faydalıdır.

Əksər yarım nəzarətli öyrənmə alqoritmləri nəzarətsiz və nəzarətli alqoritmlərin kombinasiyalarıdır. Məsələn, bir klasterləşdirmə alqoritmi oxşar nümunələri birlikdə qruplaşdırmaq üçün istifadə edilə bilər və sonra hər bir etiketlənməmiş nümunə klasterindəki ən çox yayılmış etiketlə etiketlənə bilər. Bütün məlumat dəsti etiketləndikdən sonra istənilən nəzarətli öyrənmə alqoritmindən istifadə etmək mümkündür.

Özünü-nəzarətli öyrənmə

Maşın öyrənmənin digər yanaşması tamamilə etiketlənməmiş bir məlumat dəstəsindən tam etiketlənmiş bir məlumat dəsti yaratmağı nəzərdə tutur. Yenə də, bütün məlumat dəsti etiketləndikdən sonra istənilən nəzarətli öyrənmə alqoritmindən istifadə etmək mümkündür. Bu yanaşmaya özünü-nəzarətli öyrənmə deyilir.

Məsələn, əgər böyük bir etiketlənməmiş şəkil məlumat dəstəniz varsa, hər bir şəkilin kiçik bir hissəsini təsadüfi olaraq maskalaya və sonra modeli orijinal şəkili bərpa etməsi üçün öyrədə bilərsiniz (bax Şəkil 1-12). Təlim zamanı maskalanmış şəkillər modelin girişləri kimi, orijinal şəkillər isə etiketlər kimi istifadə olunur.

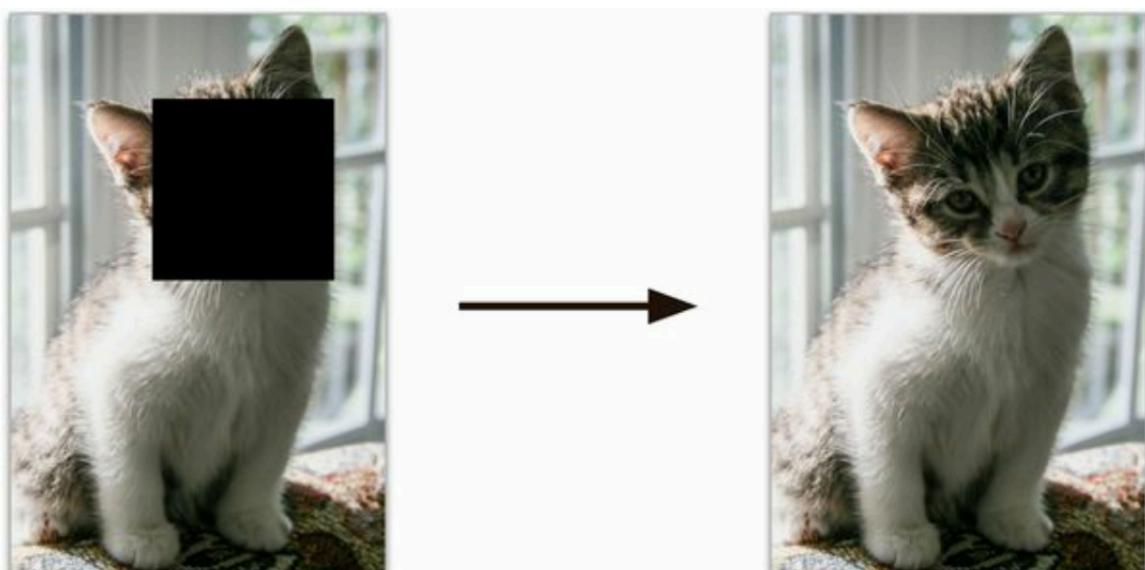


Figure 1-12. Self-supervised learning example: input (left) and target (right)

Nəticə olaraq yaranan model özlüyündə çox faydalı ola bilər—məsələn, zədələnmiş şəkilləri təmir etmək və ya şəkillərdən istənməyən obyektləri silmək üçün. Lakin, çox vaxt özünü-nəzarətli öyrənmə ilə təlim edilmiş model son məqsəd olmur. Adətən, modeli bir az fərqli bir tapşırıq üçün düzəltmək və incə tənzimləmək istəyirsiniz—gerçəkdən maraqlandığınız bir tapşırıq.

Məsələn, deyək ki, həqiqətən istədiyiniz şey ev heyvanlarının təsnifat modelidir: hər hansı bir ev heyvanının şəklini verdikdə, hansı növə aid olduğunu söyləməsi. Əgər sizdə etiketlənməmiş ev heyvanlarının şəkillərindən ibarət böyük bir məlumat dəsti varsa, özünü-nəzarətli öyrənmə istifadə edərək bir şəkil təmiri modeli təlim etməyə başlaya bilərsiniz. Bir dəfə yaxşı performans göstərdikdə, bu model müxtəlif ev heyvanlarının növlərini ayırd etməli olmalıdır: məsələn, üzünü maskalamış bir pişikin şəklini təmir edərkən,

it üzünü əlavə etməməlidir. Əgər modelin arxitekturası bunu dəstəkləyirsə (və əksər neyron şəbəkə arxitekturaları bunu edir), sonra modeli düzəldərək şəkil təmiri yerinə ev heyvanları növünü proqnozlaşdırması mümkün olur. Sonuncu addım modelin etiketli məlumat dəsti üzərində incə tənzimlənməsidir: model artıq pişiklərin, itlərin və digər ev heyvanlarının necə göründüyünü bilir, buna görə də bu addım yalnız modelin öyrəndiyi növlər ilə gözlənilən etiketlər arasındaki əlaqəni öyrənməsi üçün lazımdır.

QEYD

Bir tapşırıqdan digərinə bilik ötürülməsi transfer öyrənmə adlanır və bu, bu gün maşın öyrənməsinin ən vacib texnikalarından biridir, xüsusilə də dərin neyron şəbəkələri (yəni, bir çox neyron qatlarından ibarət olan neyron şəbəkələri) istifadə edildikdə. Bunu II hissədə ətraflı müzakirə edəcəyik.

Bəzi insanlar özünü-nəzarətli öyrənməni nəzarətsiz öyrənmənin bir hissəsi hesab edirlər, çünki tamamilə etiketlənməmiş məlumat dəstləri ilə işləyir. Lakin özünü-nəzarətli öyrənmə təlim zamanı (yaradılmış) etiketlərdən istifadə edir, buna görə də bu baxımdan nəzarətli öyrənməyə daha yaxındır. "Nəzarətsiz öyrənmə" termini ümumiyyətlə klasterləşdirmə, ölçü azalması və ya anomaliya aşkarlanması kimi tapşırıqlarla işləyərkən istifadə olunur, halbuki özünü-nəzarətli öyrənmə nəzarətli öyrənmə ilə eyni tapşırıqlara fokuslanır: əsasən təsnifat və regressiya. Qısacısı, özünü-nəzarətli öyrənmə ən yaxşı şəkildə öz kateqoriyası olaraq qəbul edilməlidir.

Gücləndirmə öyrənməsi

Gücləndirmə öyrənməsi çox fərqli bir yanaşmadır. Bu öyrənmə sistemi, bu kontekstdə agent adlanır, ətraf mühiti müşahidə edə bilir, hərəkətləri seçə və icra edə bilər və qarşılığında mükafat (və ya mənfi mükafat şəklində cəza) alır (bax Şəkil 1-13). Sonra özünü öyrənməli, zamanla ən çox mükafatı əldə etmək üçün ən yaxşı strategiyani—politikası—öyrənməlidir. Bir siyaset agentin müəyyən bir vəziyyətdə hansı hərəkəti seçməli olduğunu müəyyən edir.

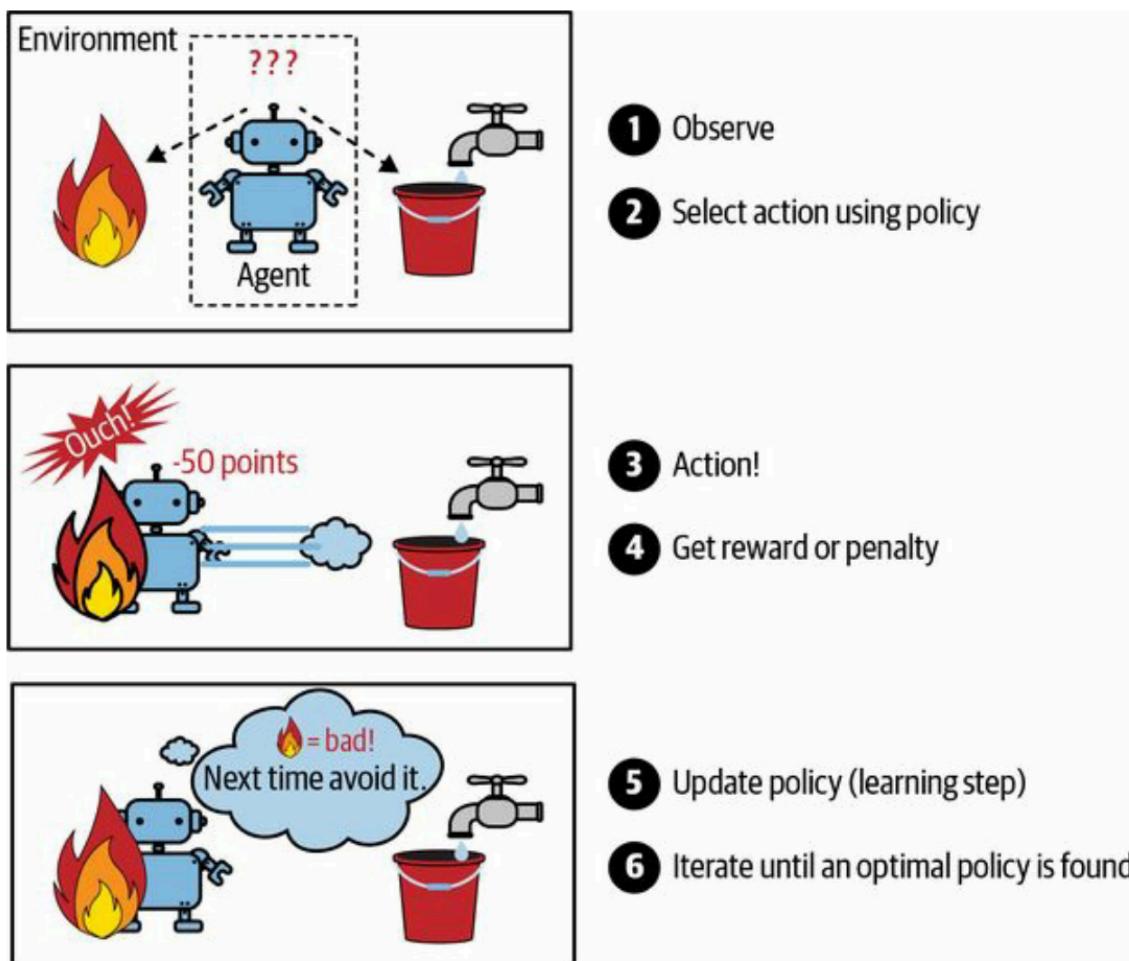


Figure 1-13. Reinforcement learning

Məsələn, bir çox robot gücləndirmə öyrənmə alqoritmalarını yürüməyi öyrənmək üçün tətbiq edir. DeepMind'ın AlphaGo programı da gücləndirmə öyrənməsinin yaxşı bir nümunəsidir: 2017-ci ilin may ayında Ke Jie adlı o dövrün dünya çempionu olan oyuncunu Go oyununda məğlub etməsi ilə manşetlərə çıxdı. O, qalibiyət siyasetini milyonlarla oyunu təhlil edərək və sonra özünə qarşı bir çox oyun oynayaraq öyrəndi. Diqqət yetirin ki, çempiona qarşı oyunlar zamanı öyrənmə söndürülmüşdü; AlphaGo yalnız öyrəndiyi siyaseti tətbiq edirdi. Növbəti bölmədə görəcəyiniz kimi, buna offline öyrənmə deyilir.

Batch Öyrənmə və Online Öyrənmə

Maşın öyrənmə sistemlərini sinifləndirmək üçün istifadə olunan başqa bir meyar, sistemin daxil olan verilənlər axınından tədricən öyrənə bilməsidir, yoxsa öyrənmə bütün verilənlərlə bir yerdə, birdən baş verirmi?

Batch Öyrənmə

Batch öyrənməsində, sistem tədricən öyrənməyi bacarmır: bütün mövcud verilənlərlə təlim edilməlidir. Bu, adətən çox vaxt və hesablaşma resursları tələb edir, buna görə də ümumiyyətlə offline həyata keçirilir. İlk olaraq sistem təlim edilir, sonra isə istehsalata buraxılır və artıq öyrənməyi dayandırır; sadəcə öyrəndiyi məlumatları tətbiq edir. Buna offline öyrənmə deyilir. Təəssüf ki, bir modelin performansı zamanla tədricən pisleşir, çünki dünya davamlı olaraq dəyişir, amma model dəyişməz qalır. Bu fenomendən tez-tez modelin

çürüməsi və ya verilənlərdə dəyişiklik adlanır. Həlli isə modelin müntəzəm olaraq yenilənmiş verilənlər üzərində təlim edilməsi və yenidən öyrədilməsidir. Nə qədər tez-tez bunun lazımlığı istifadə sahəsində asılıdır: əgər model pişik və it şəkillərini təsnif edirsə, performansı çox yavaş pisləşər, amma əgər model sürətlə dəyişən sistemlərlə məşğul olursa, məsələn maliyyə bazarlarında proqnozlar verməklə, o zaman performans sürətlə pisləşə bilər.

XƏBƏRDARLIQ

Hətta pişik və it şəkillərini təsnif etmək üçün öyrədilmiş bir model belə, müntəzəm olaraq yenidən öyrədilməlidir, çünki pişiklər və itlər gecə ərzində mutasiya etməz, lakin kameralar, şəkil formatları, aydınlıq, parlaqlıq və ölçü nisbətləri dəyişir. Bundan əlavə, insanlar növbəti il daha fərqli cinsləri bəyəndikləri üçün və ya ev heyvanlarını kiçik şlyapalarla geyindirməyi qərarlaşdırıb bilərlər—kim bilir?

Əgər siz batch öyrənmə sisteminin yeni verilənlər haqqında (məsələn, yeni bir spam növü) məlumatlı olmasını istəyirsinizsə, bütün məlumat dəstini (yalnız yeni verilənləri deyil, həm də köhnə verilənləri) istifadə edərək sistemin yeni bir versiyasını sıfırdan təlim etməlisiniz və sonra köhnə modeli yeni model ilə əvəz etməlisiniz. Şükürələr olsun ki, maşın öyrənmə sisteminin təlimi, qiymətləndirilməsi və işə salınması prosesi olduqca asanlıqla avtomatlaşdırıla bilər (Şəkil 1-3-də gördüyüümüz kimi), beləliklə hətta bir batch öyrənmə sistemi də dəyişikliklərə uyğunlaşa bilər. Yalnız verilənləri yeniləyin və sistemin yeni bir versiyasını sıfırdan təlim edin, nə qədər lazım olsa o qədər.

Bu həll sadədir və çox vaxtı yaxşı işləyir, lakin tam verilənlər dəsti ilə təlim çox saatlar çəkə bilər, buna görə də adətən yeni sistemi yalnız hər 24 saatda bir və ya hətta yalnız həftəlik təlim edərsiniz. Əgər sisteminiz sürətlə dəyişən verilənlərə uyğunlaşmalıdır (məsələn, səhmlərin qiymətini proqnozlaşdırmaq), onda daha reaktiv bir həllə ehtiyacınız olacaq.

Bundan əlavə, tam verilənlər dəsti ilə təlim çoxlu hesablama resursları tələb edir (CPU, yaddaş sahəsi, disk sahəsi, disk I/O, şəbəkə I/O və s.). Əgər çox verilənləriniz varsa və sisteminizi hər gün sıfırdan təlim etmək üçün avtomatlaşdırırsınızsa, bu çox baha başa gələ bilər. Əgər verilənlər çoxdursa, batch öyrənmə alqoritmasından istifadə etmək mümkün belə olmaya bilər.

Son olaraq, əgər sisteminizin müstəqil şəkildə öyrənə bilməsi lazımdırsa və onun məhdud resursları varsa (məsələn, smartfon tətbiqi və ya Marsda bir rover), o zaman böyük miqdarda təlim verilənlərini daşımaq və hər gün saatlarla təlim keçmək çox çətin ola bilər.

Bu hallarda daha yaxşı bir seçim incremental olaraq öyrənə bilən alqoritmalar istifadə etməkdir.

Online Öyrənmə

Online öyrənmədə, sistemi tədricən təlim edirsiniz, verilənlər nümunələrini ardıcıl olaraq, ya fərdi şəkildə, ya da kiçik qruplar (mini-batch-lər) şəklində təqdim edirsiniz. Hər bir öyrənmə addımı sürətli və ucuzdur, beləliklə sistem yeni verilənlərə anında uyğunlaşa bilər, gəldikcə öyrənə bilər (bax Şəkil 1-14).

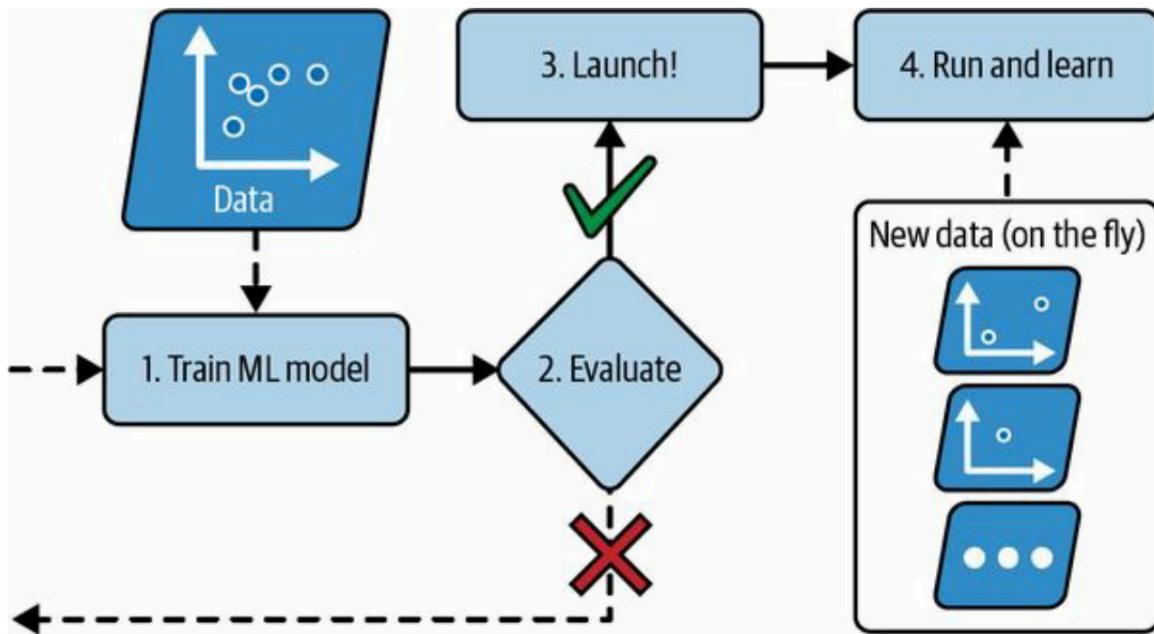


Figure 1-14. In online learning, a model is trained and launched into production, and then it keeps learning as new data comes in

Online öyrənmə, sistemlərin çox sürətli dəyişikliklərə uyğunlaşması lazımlığı hallarda faydalıdır (məsələn, səhmlər bazarında yeni naxışları aşkar etmək üçün). Həmçinin, məhdud hesablamalar resurslarına sahib olduğunuzda da yaxşı bir seçimdir; məsələn, əgər model mobil cihazda təlim edilirsə.

Bundan əlavə, online öyrənmə alqoritmaları, bir maşının əsas yaddaşına siğmayan böyük verilənlər dəstləri üzərində modelləri təlim etmək üçün də istifadə edilə bilər (buna out-of-core öyrənmə deyilir). Alqoritma verilənlərin bir hissəsini yükləyir, o verilənlər üzərində bir təlim addımı icra edir və prosesi bütün verilənlər üzərində təkrarlayır (bax Şəkil 1-15).

Bu üsul, verilənlər çox böyük olduqda və ya verilənlər ardıcıl şəkildə gəlirsə, sistemin davamlı olaraq öyrənə bilməsi üçün çox əlverişlidir. Online öyrənmə sistemləri verilənlər axınından real vaxtda yeni məlumatları qəbul edib təlim etməyə imkan verir, beləliklə, model dəyişən şəraitə tez uyğunlaşır.

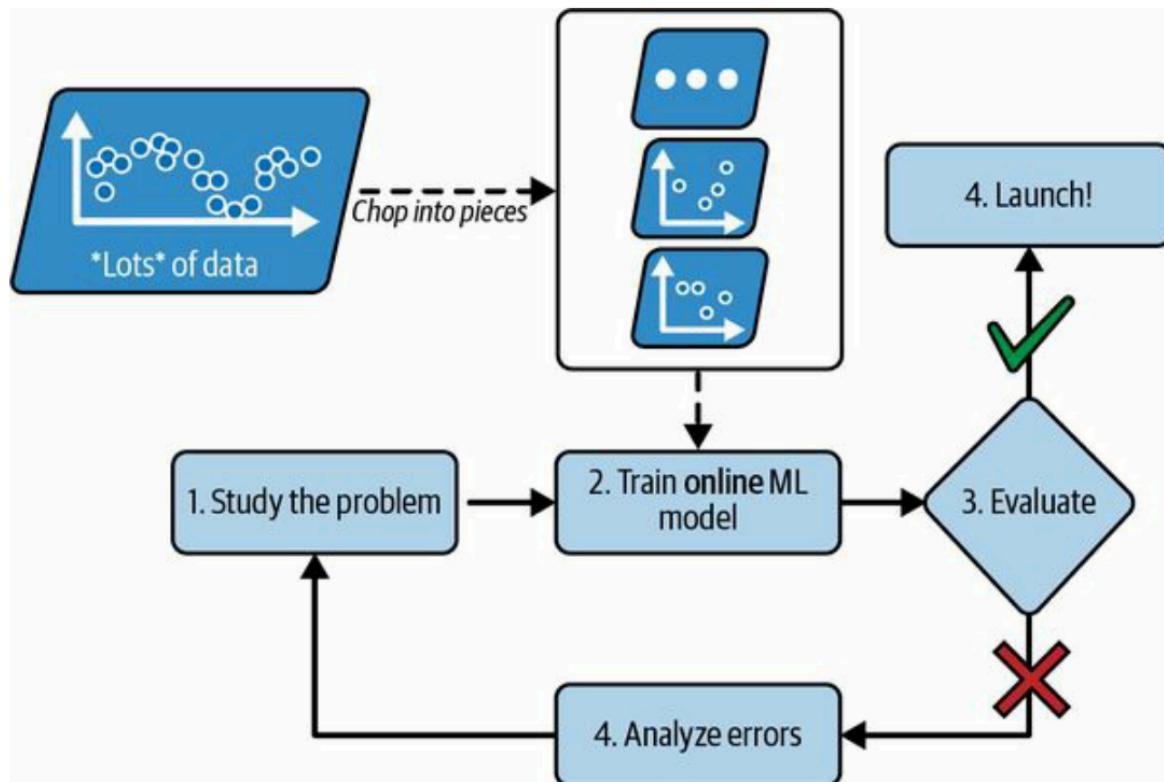


Figure 1-15. Using online learning to handle huge datasets

Online öyrənmə sistemlərinin mühüm parametrlərindən biri onların dəyişən verilənlərə nə qədər tez uyğunlaşması lazımlı olduğudur: buna öyrənmə sürəti deyilir. Əgər yüksək öyrənmə sürəti təyin etsəniz, sisteminiz yeni verilənlərə sürətlə uyğunlaşacaq, lakin eyni zamanda köhnə verilənləri tez unudacaq (və siz istəmirsiniz ki, spam filtri yalnız ona göstərilən ən son spam növlərini qeyd etsin). Əksinə, əgər aşağı öyrənmə sürəti təyin etsəniz, sistem daha çox inersiya göstərəcək; yəni daha yavaş öyrənəcək, amma yeni verilənlərdəki səs-küyə və ya təmsil etməyən verilənlər nöqtələrinin ardıcılıqlarına (outliers) daha az həssas olacaq.

XƏBƏRDARLIQ Out-of-core öyrənmə adətən offline (yəni canlı sistemdə deyil) edilir, buna görə də online öyrənmə qarşıq bir ad ola bilər. Bunu incremental öyrənmə kimi düşünün.

Online öyrənmə ilə bağlı böyük bir çətinlik ondan ibarətdir ki, əgər sistemə pis verilənlər daxil edilərsə, sistemin performansı düşə bilər, bəlkə də tez bir şəkildə (verilənlərin keyfiyyətinə və öyrənmə sürətinə bağlı olaraq). Əgər bu canlı bir sistemdirse, müştəriləriniz bunu hiss edəcək. Məsələn, pis verilənlər bir səhv (məsələn, robotda işləməyən bir sensor) və ya sistemin manipulyasiya edilməyə çalışılması (məsələn, axtarış nəticələrində yüksək sıralanmaq üçün axtarış mühərrikinə spam göndərmək) nəticəsində gələ bilər. Bu riski azaltmaq üçün sisteminizi diqqətlə izləməli və performansın azaldığını aşkarladığınız zaman öyrənməni dərhal söndürməli (və bəlkə də əvvəlki işləyən vəziyyətə qayitmalısınız). Həmçinin, daxil olan verilənləri izləmək və anormal verilənlərə reaksiya vermək istəyirsiniz; məsələn, anomaliya aşkarlama alqoritmasından istifadə edərək (Bölmə 9-a baxın).

Nümunə-Əsaslı və Model-Əsaslı Öyrənmə Maşın öyrənmə sistemlərini kateqoriyalara ayırmak üçün bir başqa üsul onların necə ümumiləşdirməsidir. Çox vaxt maşın öyrənmə tapşırıqları proqnozlar verməkdən ibarətdir. Bu o deməkdir ki, bir sıra təlim nümunələri

verildikdə, sistem heç vaxt görmədiyi nümunələr üçün yaxşı proqnozlar verməli və ya ümumiləşdirməlidir. Təlim verilənlərində yaxşı performans ölçüsünə sahib olmaq yaxşıdır, amma yetərli deyil; həqiqi məqsəd yeni nümunələrdə yaxşı performans göstərməkdir.

Ümumiləşdirmənin iki əsas yanaşması vardır: nümunə-əsaslı öyrənmə və model-əsaslı öyrənmə.

Nümunə-Əsaslı Öyrənmə

Bəlkə də ən sadə öyrənmə forması sadəcə olaraq yadda öyrənməkdir. Əgər spam filtri bu şəkildə yaradılsara, o zaman istifadəçilər tərəfindən artıq işarələnmiş e-poçtlara bənzər olan bütün e-poçtları qeyd edəcək—pis bir həll deyil, amma mütləq ən yaxşısı deyil.

Sadəcə olaraq, artıq tanınmış spam e-poçtlarına eyni olan e-poçtları qeyd etmək yerinə, spam filtriniz tanınmış spam e-poçtlarına çox bənzəyən e-poçtları da qeyd edə bilər. Bu, iki e-poçt arasındaki oxşarlığın ölçülülməsini tələb edir. İki e-poçt arasındaki (çox sadə) oxşarlıq ölçüsü, onların ortaq olan sözlərinin sayını saymaq ola bilər. Sistem, tanınmış spam e-poçtları ilə çox ortaq sözü olan bir e-poçtu spam kimi qeyd edərdi.

Bu, nümunə-əsaslı öyrənmə adlanır: sistem nümunələri yadda saxlayır, sonra oxşarlıq ölçüsündən istifadə edərək onları öyrənilmiş nümunələrə (və ya onların bir hissəsinə) müqayisə edərək yeni hallara ümumiləşdirir. Məsələn, Şəkil 1-16-da yeni nümunə üçbucaq kimi təsnif ediləcəkdir, çünki ən oxşar nümunələrin çoxu bu sinifə aiddir.

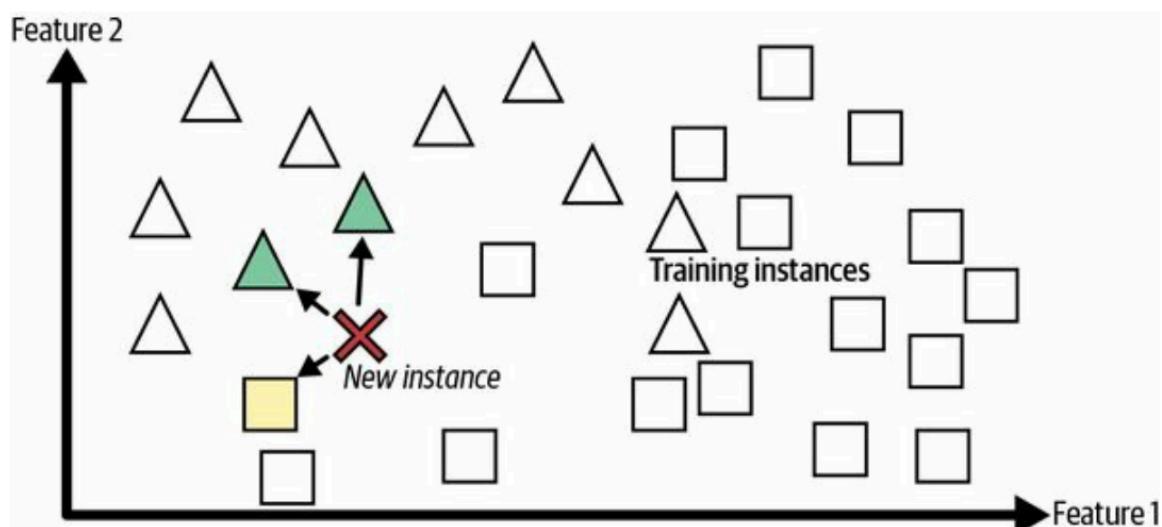


Figure 1-16. Instance-based learning

Model-Əsaslı Öyrənmə və Tipik Maşın Öyrənmə İşıqlandırması

Nümunələr toplusundan ümumiləşdirməyin başqa bir yolu, bu nümunələrin bir modelini qurmaq və sonra bu modeli istifadə edərək proqnozlar verməkdir. Buna model-əsaslı öyrənmə deyilir (Şəkil 1-17).

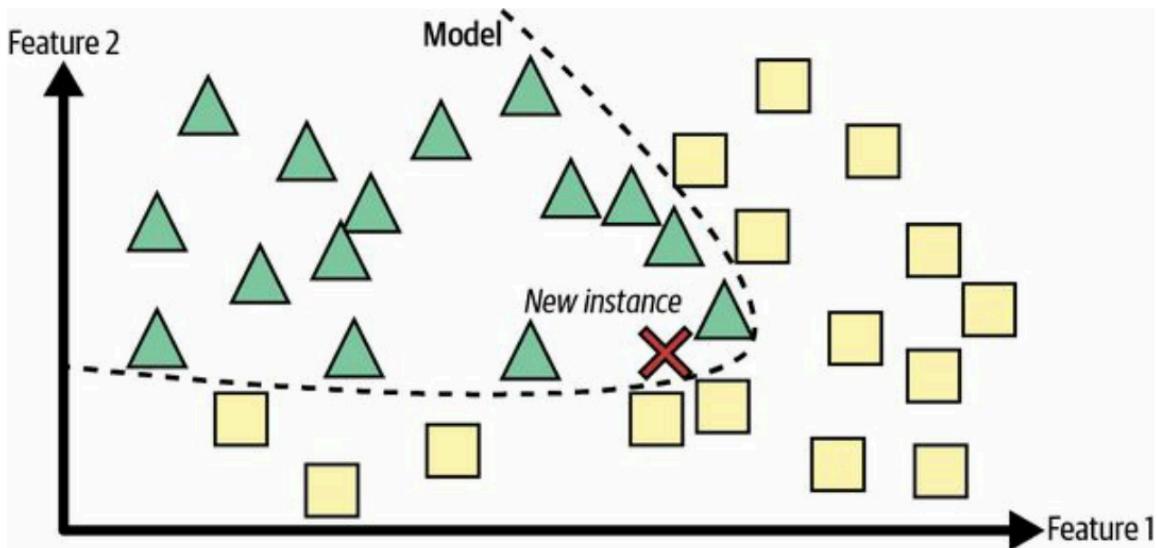


Figure 1-17. Model-based learning

Məsələn, əgər siz pulların insanları daha xoşbəxt edib-etmədiyini öyrənmək istəyirsinizsə, OECD-nin vəb saytından Better Life Index məlumatlarını və Ümumdünya Bankının adambaşına düşən Ümumi Daxili Məhsul (ÜDM) statistikalarını yükləyirsiniz. Sonra cədvəlləri birləşdirib adambaşına ÜDM-ə görə sıralayırsınız. Aşağıda 1-1 cədvəlində əldə etdiyiniz məlumatlardan bir parçası görünürsünüz:

Cədvəl 1-1. Pul insanları xoşbəxt edirmi?

Country	GDP per capita (USD)	Life satisfaction
Turkey	28,384	5.5
Hungary	31,008	5.6
France	42,026	6.5
United States	60,236	6.9
New Zealand	42,404	7.3
Australia	48,698	7.3
Denmark	55,938	7.6

Gəlin, bu ölkələr üçün məlumatları qrafikləşdirmək (Şəkil 1-18).

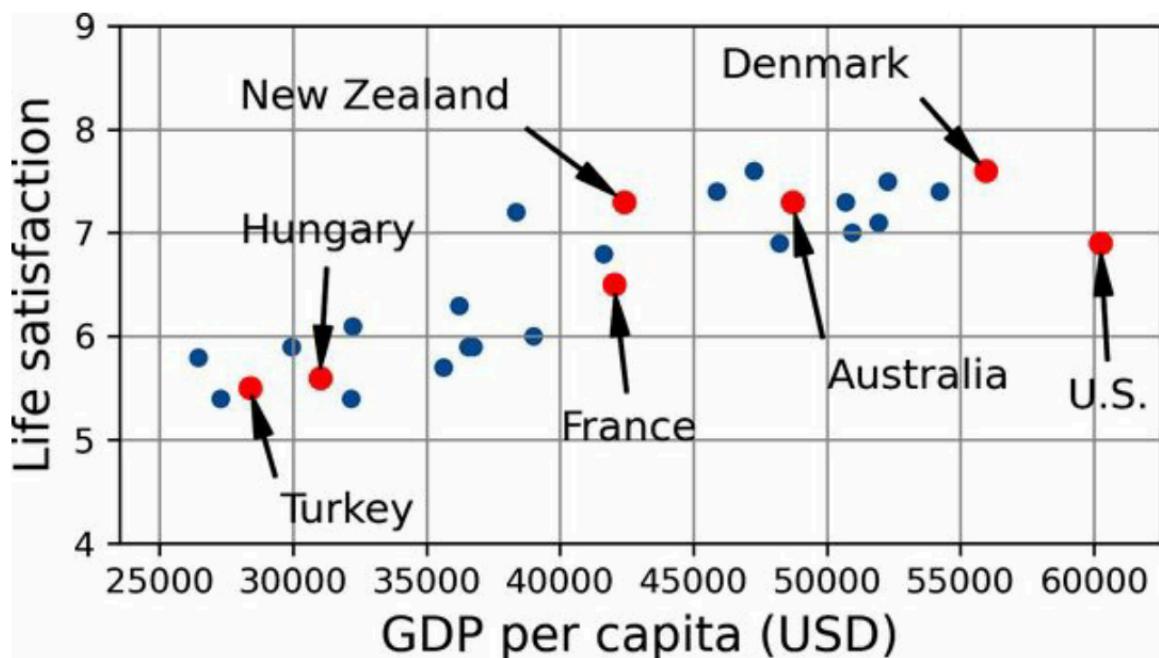


Figure 1-18. Do you see a trend here?

Burada bir tendensiya olduğu görünür! Verilənlər səsgüclüdür (yəni, qismən təsadüfi), amma görünür ki, həyat məmənuniyyəti ölkənin adambaşına düşən ÜDM-i artıraqca daha çox və ya az xətti şəkildə yüksəlir. Beləliklə, həyat məmənuniyyətini adambaşına düşən ÜDM-in xətti funksiyası kimi modelləşdirməyə qərar verirsiniz. Bu addım model seçimi adlanır: siz həyat məmənuniyyətini yalnız bir atributla, adambaşına düşən ÜDM ilə xətti model olaraq seçdiniz (Tənlik 1-1).

Tənlik 1-1. Sadə xətti model

$$\text{həyat_məmənuniyyəti} = \theta_0 + \theta_1 \times \text{ÜDM_adambaşına}$$

Bu modelin iki model parametri var, θ_0 və θ_1 . Bu parametrləri tənzimləyərək, modelinizi hər hansı bir xətti funksiyani təmsil edəcək şəkildə qura bilərsiniz, bu da Şəkil 1-19-da göstərilib.

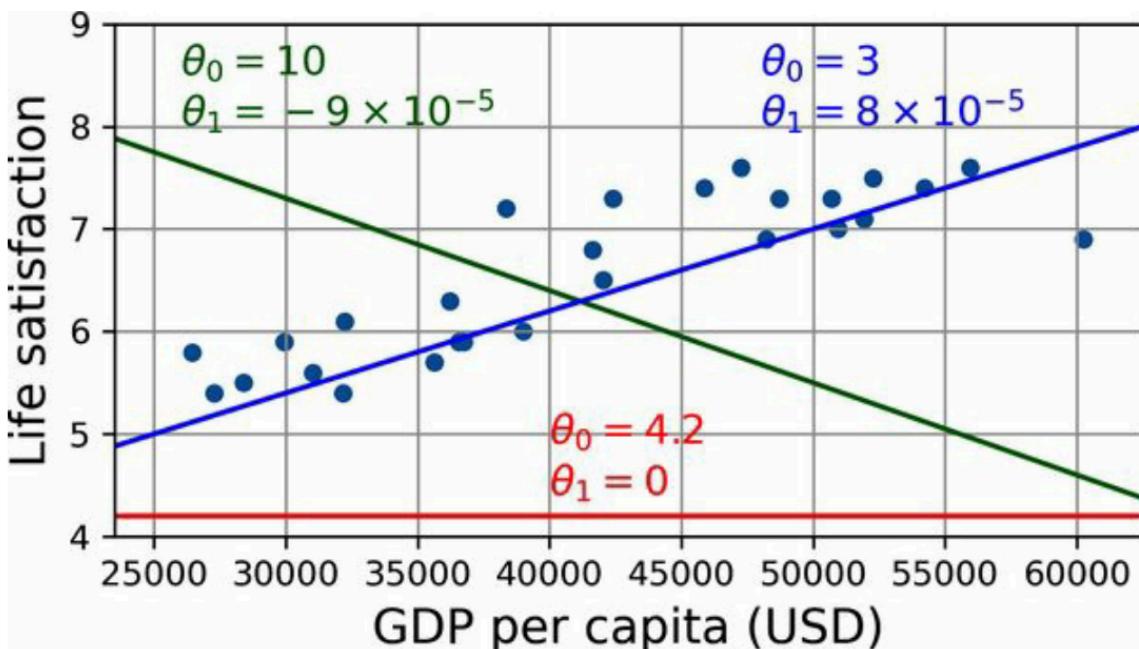


Figure 1-19. A few possible linear models

Modelinizi istifadə etməzdən əvvəl, θ_0 və θ_1 parametrlərinin dəyərlərini müəyyən etməlisiniz. Hansı dəyərlərin modelinizin ən yaxşı performansını göstərəcəyini necə bilə bilərsiniz? Bu suala cavab vermək üçün bir performans ölçüsü təyin etməlisiniz. Siz ya modelinizin nə qədər yaxşı olduğunu ölçən bir fayda funksiyası (və ya uyğunluq funksiyası) müəyyən edə bilərsiniz, ya da modelinizin nə qədər pis olduğunu ölçən bir xərc funksiyası təyin edə bilərsiniz. Xətti regressiya problemləri üçün insanlar adətən xətti modelin proqnozları ilə təlim nümunələri arasındaki məsafəni ölçən bir xərc funksiyası istifadə edirlər; məqsəd bu məsafəni minimuma endirməkdir.

Burada xətti regressiya alqoritması devreye girir: təlim nümunələrinizi ona verirsınız və o, xətti modelin verilənlərinizə ən yaxşı uyğunlaşmasını təmin edəcək parametrləri tapır. Bu addıma modelin təlimi deyilir. Bizim vəziyyətimizdə, alqoritm optimal parameter dəyərlərinin $\theta_0 = 3.75$ və $\theta_1 = 6.78 \times 10^{-6}$ olduğunu tapır.

XƏBƏRDARLIQ

Qarışılıqlı yaranan bir şəkildə, "model" sözü model növünə (məsələn, xətti regressiya), tam təyin olunmuş model arxitekturasına (məsələn, bir giriş və bir çıxışla xətti regressiya) və ya proqnozlar üçün istifadə edilməyə hazır olan son təlim olunmuş modelə (məsələn, $\theta_0 = 3.75$ və $\theta_1 = 6.78 \times 10^{-6}$ ilə bir giriş və bir çıxışlı xətti regressiya) aid ola bilər. Model seçimi model növünü seçmək və onun arxitekturasını tam təyin etməkdən ibarətdir. Modelin təlimi isə alqoritmani işlədərək modelin parametrlərini tapmaq və onu təlim verilənləri ilə ən yaxşı uyğunlaşdırmaqdan ibarətdir, və ümid edilir ki, yeni verilənlər üzərində yaxşı proqnozlar verəcək.

İndi model təlim verilənlərinə mümkün qədər yaxın uyğunlaşır (xətti model üçün), bunu Şəkil 1-20-də görə bilərsiniz.

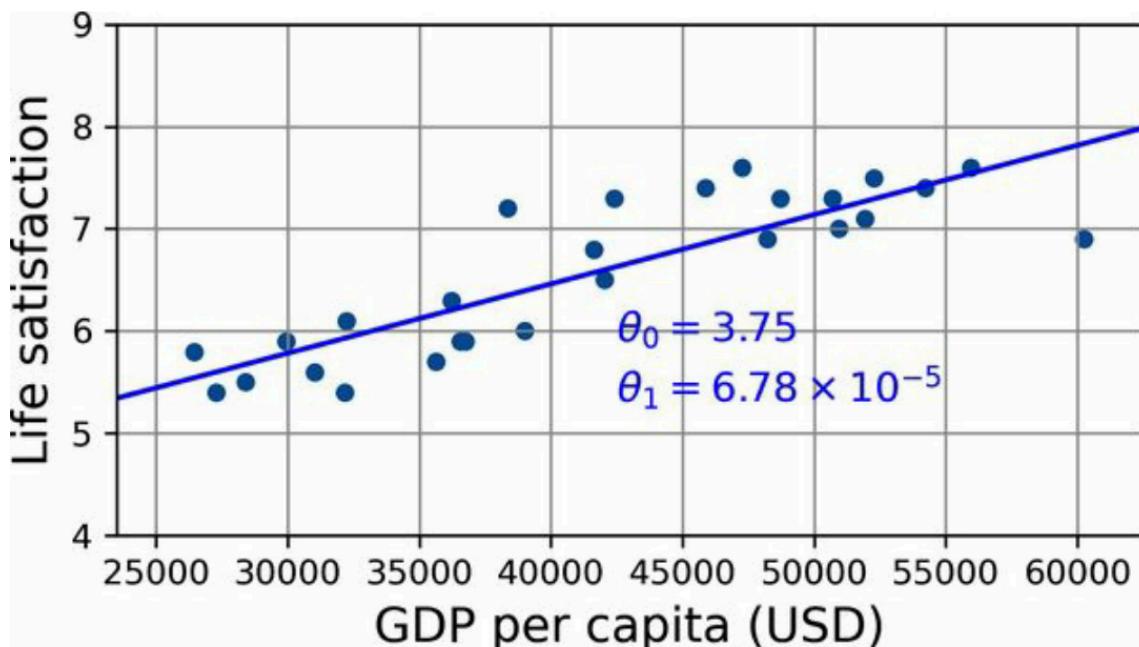


Figure 1-20. The linear model that fits the training data best

Nəhayət, modelinizi işlədib proqnozlar verməyə hazırlısanız. Məsələn, deyək ki, Kiprlilərin nə qədər xoşbəxt olduğunu bilmək istəyirsiniz, amma OECD məlumatları bunun cavabını vermir. Şükürlər olsun ki, modelinizdən istifadə edərək yaxşı bir proqnoz verə bilərsiniz: Kiprin ÜDM per capita-nı tapırsınız, 37,655 dollar olduğunu görürsünüz və sonra modelinizi tətbiq edərək həyat məmənuniyyətinin təxminən $3.75 + 37,655 \times 6.78 \times 10^{-6} = 6.30$ olacağını tapırsınız.

Sizdən əvvəlki nümunəni görmək üçün, Example 1-1-də verilən Python kodu, verilənləri yükleyir, girişləri (X) və etiketləri (y) ayıır, vizuallaşdırma üçün bir səpələnmə qrafiki yaradır, sonra xətti modeli təlim edir və proqnoz verir.

Example 1-1. Scikit-Learn istifadə edərək xətti modelin təlimi və işlədilməsi

```
import matplotlib.pyplot as plt
import numpy as np
import pandas as pd
from sklearn.linear_model import LinearRegression

# Verilənləri yükleyib hazırlayın
data_root = "https://github.com/ageron/data/raw/main/"
lifesat = pd.read_csv(data_root + "lifesat/lifesat.csv")
X = lifesat[["GDP per capita (USD)"]].values
y = lifesat[["Life satisfaction"]].values

# Verilənləri vizuallaşdırın
lifesat.plot(kind='scatter', grid=True,
             x="GDP per capita (USD)", y="Life satisfaction")
```

```

plt.axis([23_500, 62_500, 4, 9])
plt.show()

# Xətti modeli seçin
model = LinearRegression()

# Modeli təlim edin
model.fit(X, y)

# Kipr üçün proqnoz verin
X_new = [[37_655.2]] # 2020-ci ildə Kiprin ÜDM per capita
print(model.predict(X_new)) # nəticə: [[6.30165767]]

```

Qeyd:

Əgər yerinə instansiya əsaslı öyrənmə alqoritmasından istifadə etmiş olsaydınız, İsrailin Kiprə ən yaxın ÜDM per capita-ya (38,341 dollar) sahib olduğunu tapardınız və OECD məlumatları bize İsraililərin həyat məmənnuniyyətinin 7.2 olduğunu deyir. Beləliklə, Kipr üçün həyat məmənnuniyyəti proqnozunu 7.2 olaraq verərdiniz. Bir az daha uzağa baxsanız, Litva və Sloveniya, hər ikisi də 5.9 həyat məmənnuniyyətinə malikdir. Bu üç dəyərin ortalamasını alsanız, 6.33 alırsınız ki, bu da model əsaslı proqnozunuza çox yaxındır. Bu sadə alqoritma k-nəarest qonşular rəqressiyası (bu nümunədə, $k = 3$) adlanır.

Əvvəlki kodda xətti rəqressiya modelini k-nəarest qonşular rəqressiyası ilə əvəz etmək o qədər də çətin deyil, sadəcə bu sətirləri dəyişdirməlisiniz:

```

from sklearn.linear_model import LinearRegression
model = LinearRegression()

```

yerinə:

```

from sklearn.neighbors import KNeighborsRegressor
model = KNeighborsRegressor(n_neighbors=3)

```

Əgər hər şey yaxşı gedərsə, modeliniz yaxşı proqnozlar verəcəkdir. Əgər yox, o zaman daha çox atribut (işsizlik nisbəti, sağlamlıq, hava çirkiliyi və s.) əlavə etməli, daha çox və ya daha keyfiyyətli təlim verilənləri əldə etməli və ya daha güclü bir model seçməli ola bilərsiniz (məsələn, polinomial rəqressiya modeli).

Xülasə:

- Verilənləri öyrəndiniz.
- Bir model seçdiniz.
- Onu təlim verilənləri üzərində təlim etdiniz (yəni, öyrənmə alqoritması xərc funksiyasını minimuma endirəcək model parametrlərini tapdı).
- Nəhayət, modelinizi yeni hallar üzrə proqnoz vermək üçün tətbiq etdiniz (bu, nəticə çıxarma adlanır), ümid edirsiniz ki, bu model yaxşı ümumiləşəcək.

Bu, tipik bir maşın öyrənməsi layihəsinin necə göründüyüdür. 2-ci fəsildə bu prosesi başdan sona qədər bir layihə üzərində tətbiq edərək birbaşa təcrübə edəcəksiniz.

Biz çox şey öyrəndik: indi maşın öyrənməsinin nə olduğunu, niyə faydalı olduğunu, ən çox istifadə olunan maşın öyrənmə sistemləri kateqoriyalarını və tipik bir layihə iş axışının necə olduğunu bilirsiniz. İndi isə öyrənmədə nələrin səhv gedə biləcəyini və dəqiq proqnozlar verməyinizi mane ola biləcəyini araşdırıq.

Maşın Öyrənməsinin Əsas Çətinlikləri

Qısa desək, əsas vəzifəniz model seçmək və onu bəzi verilənlər üzərində təlim etmək olduğu üçün, səhv gedə biləcək iki əsas şey var: "pis model" və "pis verilənlər". Gəlin, əvvəlcə pis verilənlər nümunələrinə baxaq.

Təlim Verilənlərinin Kifayət Etməməsi

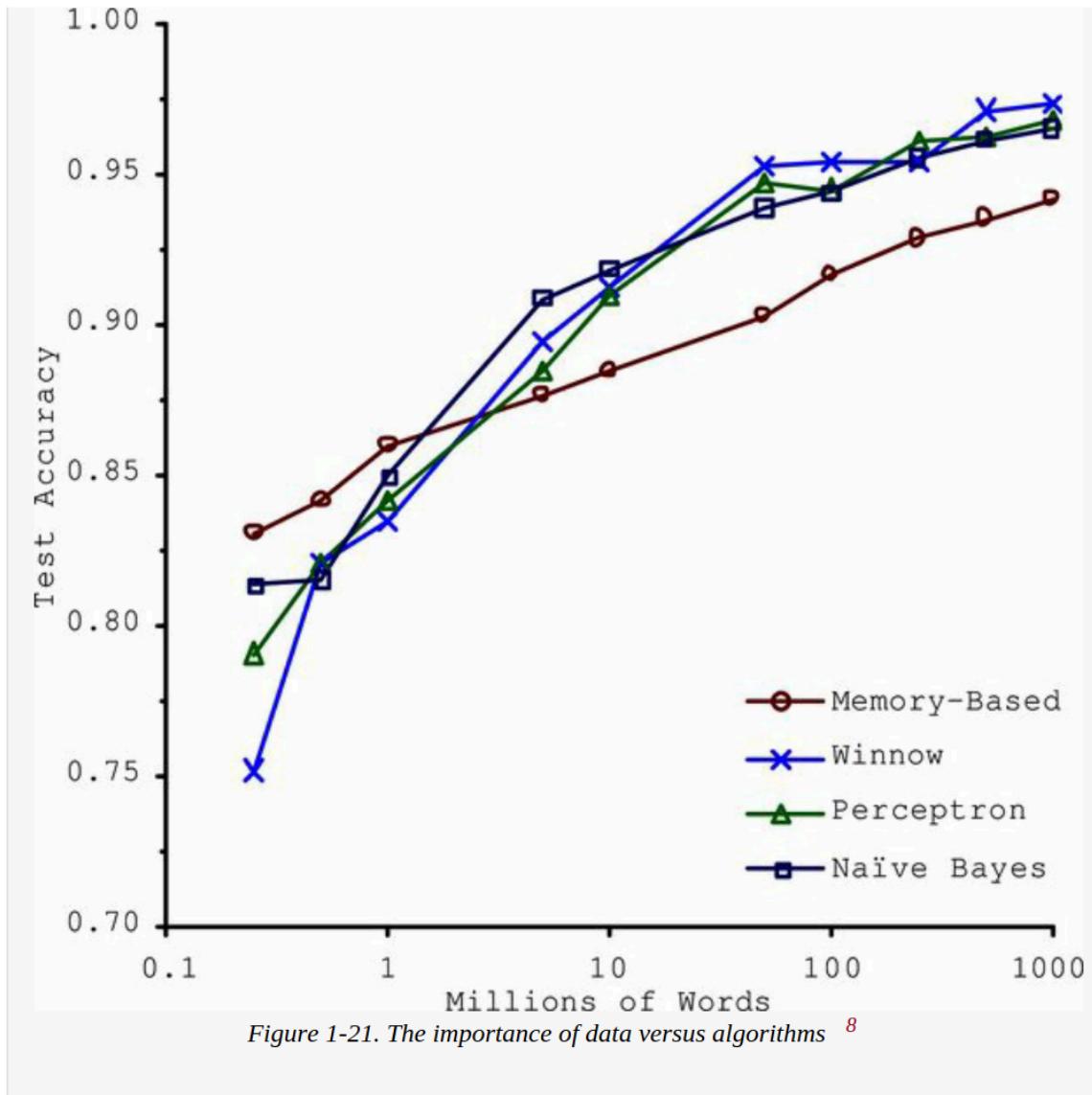
Bir uşaq alma nə olduğunu öyrənmək üçün sadəcə siz alma işarəsi edib "alma" deməyiniz yetərlidir (bəlkə də bu proseduru bir neçə dəfə təkrarlamaq lazımdır). İndi uşaq hər cür rəngdə və formada olan almanın tanıya bilir. Dahi. Maşın öyrənməsi hələ bu səviyyədə deyil; əksər maşın öyrənmə alqoritmalarının düzgün işləməsi üçün çoxlu verilənlər tələb olunur. Hətta çox sadə problemlər üçün də adətən minlərlə nümunəyə ehtiyacınız olacaq, və şəkil və ya səs tanıma kimi mürəkkəb problemlər üçün milyonlarla nümunəyə ehtiyac ola bilər (mövcud bir modelin bəzi hissələrini təkrar istifadə edə bilməsəniz).

VERİLƏNLƏRİN MƏNTİQSİZ EFFEKTİVLİYİ

2001-ci ildə Microsoft tədqiqatçıları Michele Banko və Eric Brill tərəfindən nəşr olunan məşhur bir məqalədə, çox fərqli maşın öyrənmə alqoritmalarının, o cümlədən olduqca sadə olanların, kifayət qədər verilənlər verildikdə təbii dilin qeyri-müəyyənliyini aradan qaldırmaqdə demək olar ki, eyni dərəcədə yaxşı nəticə verdiyini göstərdilər (bunu Şəkil 1-21-də görə bilərsiniz).

Müəlliflərin dediklərinə görə, "bu nəticələr göstərir ki, biz alqoritm inkişafına sərf edilən vaxt və pul ilə korpus inkişafına sərf edilən vaxt və pulu yenidən nəzərdən keçirməliyik".

Verilənlərin mürəkkəb problemlər üçün alqoritmlərdən daha çox əhəmiyyət kəsb etdiyi fikri, 2009-cu ildə Peter Norvig və digərləri tərəfindən "Verilənlərin Mətləbə Gəlməyən Effektivliyi" adlı məqalədə daha da populyarlaşdırıldı. Lakin qeyd etmək lazımdır ki, kiçik və orta ölçülü verilənlər hələ də çox yaygındır və əlavə təlim verilənləri əldə etmək hər zaman asan və ucuz deyil — buna görə də alqoritmləri hələ tərk etməyin.



Nümayəndə Olmayan Təlim Verilənləri

Yaxşı ümumiləşdirmək üçün təlim verilənlərinizin, ümumiləşdirmək istədiyiniz yeni hallar üçün nümayəndəlik etməsi çox vacibdir. Bu, istər nümunə əsaslı öyrənmə, istərsə də model əsaslı öyrənmə istifadə etməyinizdən asılı olmayaraq doğrudur.

Məsələn, əvvəlki linak modelini təlim etdiyiniz ölkələr dəstisi tam nümayəndəçi deyildi; bu, \$23,500-dən aşağı və ya \$62,500-dən yuxarı olan heç bir ölkəni əhatə etmirdi. Şəkil 1-22-də bu cür ölkələri əlavə etdikdə verilənlərin necə göründüyünü görə bilərsiniz.

Əgər bu verilənlər üzərində bir xətti model təlim etsəniz, nəticə möhkəm xətt olacaq, əvvəlki model isə nöqtəli xəttlə təmsil olunur. Görə biləcəyiniz kimi, yalnız bir neçə əskik ölkə əlavə etmək modelin nəticəsini əhəmiyyətli dərəcədə dəyişdirir, eyni zamanda bu sadə xətti modelin heç vaxtı yaxşı işləməyəcəyini göstərir. Görünür ki, çox zəngin ölkələr, orta dərəcədə

zəngin ölkələrdən daha xoşbəxt deyillər (əslində, onlar bir az daha az xoşbəxt görünür!), əksinə, bəzi kasib ölkələr bir çox zəngin ölkələrdən daha xoşbəxt görünür.

Nümayəndə olmayan təlim dəstindən istifadə edərək, xüsusilə çox kasib və çox zəngin ölkələr üçün doğru proqnozlar verməyəcək bir model təlim etdiniz.

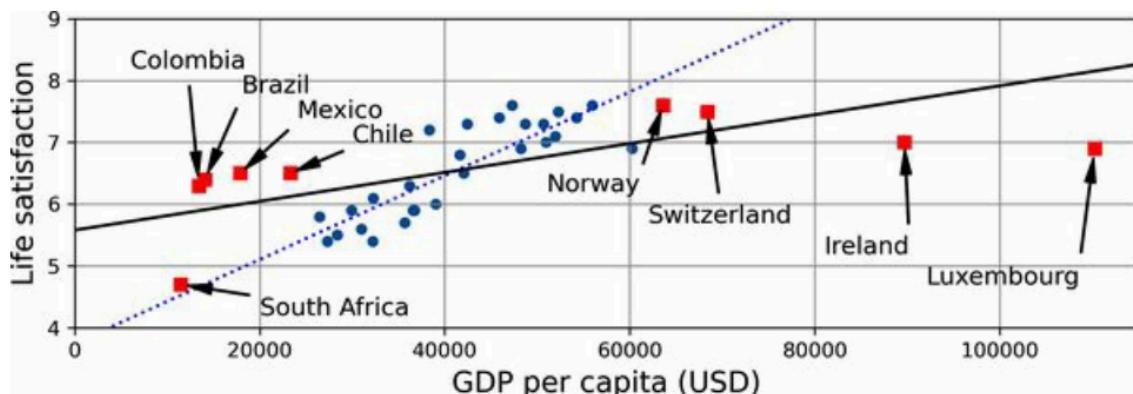


Figure 1-22. A more representative training sample

Təlim Dəstinin Nümayəndə Olması Vacibdir

Ümumiləşdirmək istədiyiniz hallar üçün təlim dəstinizin nümayəndəçi olması çox vacibdir. Bu, eşidildiyi qədər asan deyil: əgər nümunə çox kiçikdirse, sizdə nümunə səhvi olacaq (yəni, şans nəticəsində nümayəndə olmayan verilənlər), lakin hətta çox böyük nümunələr də səhv ola bilər, əgər nümunə götürmə metodu səhvdirse. Buna nümunə önyarğısı (sampling bias) deyilir.

NÜMUNƏ ÖNYARĞISI NÜMUNƏLƏRİ

Bəlkə də ən məşhur nümunə önyarğısı 1936-cı ildəki ABŞ prezident seçkilərində baş verdi, burada Landon Rooseveltə qarşı rəqabət etdi: "Literary Digest" çox böyük bir anket keçirdi və təxminən 10 milyon insana məktub göndərdi. 2.4 milyon cavab aldı və yüksək inamla Landonun 57% səs alacağını proqnozlaşdırıldı. Lakin Roosevelt 62% səsle qalib gəldi. Xətanın səbəbi Literary Digest-in nümunə götürmə metodunda idi:

Birinci, anketləri göndərmək üçün ünvanları əldə etmək məqsədilə Literary Digest telefon kataloqları, jurnallara abunə olanların siyahıları, klub üzvlüyü siyahıları və s. istifadə etdi. Bu siyahıların hamısı daha zəngin insanları üstün tuturdu, çünkü zəngin insanlar daha çox Respublikaçı (yəni Landon) olmağa meyllidirlər.

İkinci, anketə cavab verən insanların sayı 25%-dən az idi. Bu da yenə nümunə önyarğısı yaradırdı, çünkü çox maraqlanmayan insanlar, Literary Digest-i bəyənməyənlər və digər əhəmiyyətli qruplar cəlb edilə bilərdi. Bu xüsusi növ nümunə önyarğısına cavabsızlıq önyarğısı (nonresponse bias) deyilir.

Başqa bir nümunə: deyək ki, funk musiqi kliplərini tanıyan bir sistem qurmaq isteyirsiniz. Təlim dəstiniz qurmağın bir yolu "funk music"i YouTube-da axtarmaq və nəticədə yaranan videoları istifadə etməkdir. Lakin bu, YouTube-un axtarış mühərrikinin bütün funk musiqi

klipləri üçün nümayəndəçi bir dəst alacağını düşünür. Əslində, axtarış nəticələri ehtimal ki, məşhur sənətçiləri üstün tutacaq (və əgər Braziliyadansınızsa, çoxlu "funk carioca" klipləri əldə edəcəksiniz ki, bunlar James Brown-un musiqisinə heç bənzəmir). Digər tərəfdən, böyük bir təlim dəsti necə əldə edilə bilər?

Pis Keyfiyyətli Verilənlər

Aydın şəkildə, əgər təlim verilənləriniz səhvvlərlə, kənar qiymətlərlə və səs-küylə doludursa (məsələn, pis keyfiyyətli ölçmələr səbəbindən), bu, sistemin əsas nümunələri tapmasını çətinləşdirəcək, beləliklə sisteminizin yaxşı performans göstərmə ehtimalı azalacaq. Təlim verilənlərinizi təmizləmək üçün vaxt sərf etmək çox dəyərli ola bilər. Həqiqət budur ki, əksər verilənlər elmi mütəxəssisləri çox vaxtlarının əhəmiyyətli bir hissəsini məhz buna sərf edirlər. Aşağıda təlim verilənlərini təmizləmək üçün bir neçə nümunə var:

- Əgər bəzi nümunələr aydın şəkildə kənar qiymətlərdirdən, sadəcə onları atmaq və ya səhvvləri əl ilə düzəltmək faydalı ola bilər.
- Əgər bəzi nümunələr bəzi xüsusiyyətlərdən məhrumdursa (məsələn, müştərilərinizin 5%-i yaşlarını göstərməyib), siz bu xüsusiyyəti tamamilə gözardı etmək, bu nümunələri gözardı etmək, itkin qiymətləri doldurmaq (məsələn, orta yaşı) və ya bir modeli xüsusiyyətlə, digərini isə bu xüsusiyyət olmadan təlim etmək qərarını verməlisiniz.

Əhəmiyyətsiz Xüsusiyyətlər

Atalar sözündə deyildiyi kimi: "Atılan zibil, çıxan zibil". Sizin sisteminiz yalnız təlim verilənləri müvafiq xüsusiyyətlərlə dolu olduğu halda öyrənə bilər və çox sayda əhəmiyyətsiz xüsusiyyətlərdən uzaq olmalıdır. Maşın öyrənməsi layihəsinin uğurunun kritik bir hissəsi, təlim üçün yaxşı bir xüsusiyyət dəstinin seçilməsidir. Bu proses, **xüsusiyyət mühəndisliyi** adlanır və aşağıdakı addımları əhatə edir:

- **Xüsusiyyət seçimi** (mövcud xüsusiyyətlər arasında ən faydalı olanları seçmək)
- **Xüsusiyyət çıxarılması** (mövcud xüsusiyyətləri birləşdirərək daha faydalı bir xüsusiyyət yaratmaq — daha əvvəl gördükümüz kimi, ölçülərin azaldılması alqoritmləri buna kömək edə bilər)
- **Yeni xüsusiyyətlər yaratmaq** (yeni verilənlər toplayaraq yeni xüsusiyyətlər əlavə etmək)

İndi pis verilənlərin bir çox nümunəsinə baxdıqdan sonra, gəlin pis alqoritmlərin bir neçə nümunəsinə nəzər salaq.

Təlim Verilənlərinə Həddindən Artıq Uyğunlaşma (Overfitting)

Deyək ki, xarici bir ölkəyə səfər edirsiniz və taksi sürücüsü sizi aldadır. Sizi çox yandırsa, bu ölkədəki bütün taksi sürücülərinin oğru olduğunu söyləmək istəyə bilərsiniz. Çox ümmükləşdirmək, insanların tez-tez etdiyi bir şeydir və təəssüf ki, maşınlar da eyni tələyə düşə bilər, əgər biz diqqətli olmasaq. Maşın öyrənməsində buna **həddindən artıq uyğunlaşma** (overfitting) deyilir: bu, modelin təlim verilənləri üzərində yaxşı işləməsi, amma ümmükləşdirmədə zəif performans göstərməsi deməkdir.

Şəkil 1-23-də yüksək dərəcəli polinomial həyat məmənuniyyəti modelinin bir nümunəsi göstərilir, bu model təlim verilənlərinə güclü şəkildə uyğunlaşır. Hətta sadə xətti modeldən təlim verilənləri üzərində çox daha yaxşı performans göstərsə də, siz bu modelin proqnozlarına həqiqətən inanardınızmı?

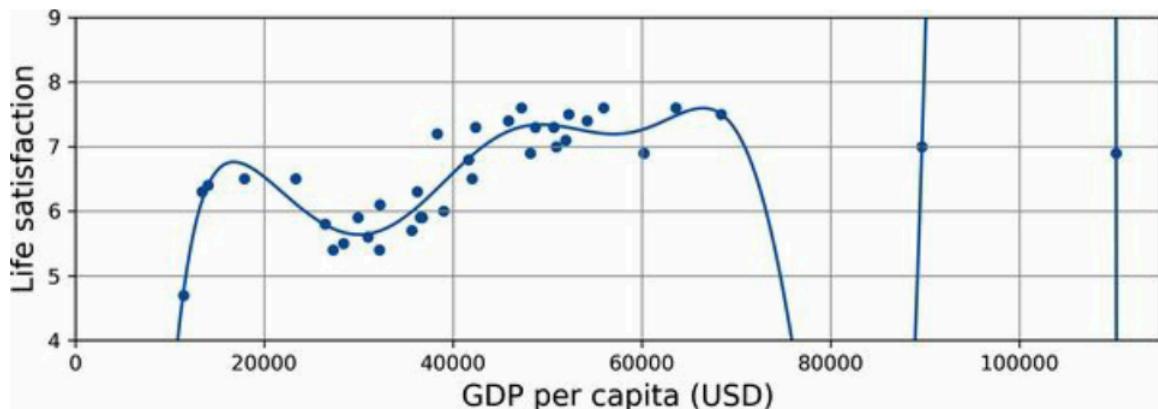


Figure 1-23. Overfitting the training data

Çox mürəkkəb modellər (məsələn, dərin neyron şəbəkələri) verilənlərdəki incə nümunələri aşkar edə bilirlər, amma təlim verilənləri çox səs-küylü olduqda və ya çox kiçik olduqda, bu, nümunə səs-küyü gətirə bilər və model verilənlərdəki səs-küydən nümunələri aşkar etməyə meyilli ola bilər (taksi sürücüsü nümunəsi kimi). Aydın olan budur ki, bu nümunələr yeni hallar üçün ümumiləşdirilməyəcək. Məsələn, deyək ki, həyat məmənuniyyəti modelinizə daha çox atribut verirsınız, o cümlədən qeyri-məlumat verən atributlar, məsələn, ölkənin adı. Bu halda, mürəkkəb bir model belə nümunələri aşkar edə bilər: məsələn, təlim verilənlərində adı "w" hərfi ilə bitən bütün ölkələrin həyat məmənuniyyəti 7-dən böyükdür: Yeni Zelandiya (7.3), Norveç (7.6), İsveç (7.3) və İsveçrə (7.5).

Neçə dərəcədə əminsiniz ki, "w" və həyat məmənuniyyəti qaydası Ruanda və ya Zimbabve üçün ümumiləşir? Aydın məsələdir ki, bu nümunə təlim verilənlərində sadəcə təsadüfən baş vermişdir, amma modelin səs-küydəki nümunələrin real olub-olmamasını ayırd etməyə heç bir yolu yoxdur.

XƏBƏRDARLIQ

Həddindən artıq uyğunlaşma (overfitting), modelin təlim verilənlərinin miqdarına və səs-küylü olmasına nisbətən çox mürəkkəb olduğu zaman baş verir. Burada bir neçə mümkün həll yolu təqdim edilir:

- Modeli sadələşdirmək:** Daha az parametrlı bir model seçin (məsələn, yüksək dərəcəli polinomial model yerinə xətti model), təlim verilənlərindəki atributların sayını azaldın və ya modeli məhdudlaşdırın.
- Daha çox təlim veriləni toplamaq.**
- Təlim verilənlərindəki səs-küyləri azaltmaq:** Məsələn, verilənlərdəki səhvələri düzəldin və üçün çıxanları silin.

Modeli məhdudlaşdırmaq və onu sadələşdirmək, həmçinin həddindən artıq uyğunlaşma riskini azaltmaq, **regulyarlaşdırma** adlanır. Məsələn, əvvəlki xətti modelimizdə iki parametr

var, $\theta_0\backslash\thetaeta_000$ və $\theta_1\backslash\thetaeta_101$. Bu modelə təlim verilənlərə uyğunlaşmaq üçün iki dərəcə sərbəstlik verir: həm hündürlüyü ($\theta_0\backslash\thetaeta_000$), həm də yamacı ($\theta_1\backslash\thetaeta_101$) dəyişdirə bilərik. Amma əgər $\theta_0=0\backslash\thetaeta_0 = 000=0$ olarsa, alqoritmin yalnız bir sərbəstliyi olacaq və verilənlərə düzgün uyğunlaşmaq çox çətin olacaq: yalnız xətti yuxarı və ya aşağıya doğru hərəkət etdirə bilər, bu halda model sadəcə ortalama ətrafında olacaq. Bu çox sadə bir model olar! Əgər biz $\theta_1\backslash\thetaeta_101$ -i dəyişməyə icazə versək, amma kiçik qalmasını məcbur etsək, alqoritm bir neçə sərbəstlik dərəcəsi olan daha sadə bir model yaradacaq. Bu model daha sadə olacaq, amma yalnız bir sərbəstlik dərəcəsi olan modeldən daha mürekkeb olacaq. Siz təlim verilənlərinə mükəmməl uyğunlaşma ilə modelin sadə olmasını təmin edəcək düzgün balansı tapmaq istəyirsiniz.

Şəkil 1-24-də üç model göstərilir:

- Dotted xətt əvvəlki modelin təlim verilənlərində (dairələr ilə təmsil olunan ölkələr) uyğunlaşdırıldığı vəziyyəti göstərir.
- Sərt xətt isə bütün ölkələrlə (dairələr və kvadratlar ilə) təlim verilənlərində olan ikinci modeli göstərir.
- Xətti xətləri olan model isə əvvəlki model ilə eyni verilənlərlə amma regulyarlaşdırma məhdudlaşdırması tətbiq edilərək təlim verilənləri ilə uyğunlaşdırılır.

Görürsünüz ki, **regulyarlaşdırma** modelin yamacını kiçik etməyə məcbur edib: bu model, təlim verilənlərindəki (dairələr) ilk model qədər yaxşı uyğunlaşmasa da, yeni nümunələrə (kvadratlar) daha yaxşı ümumiləşir.

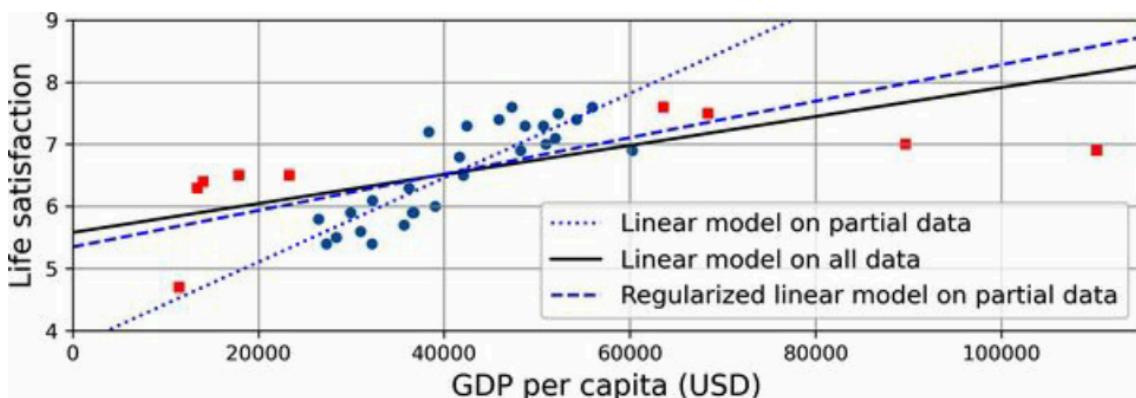


Figure 1-24. Regularization reduces the risk of overfitting

Öyrənmə zamanı tətbiq ediləcək **regulyarlaşdırma** miqdarı bir **hiperparametr** ilə idarə oluna bilər. Hiperparametr, öyrənmə alqoritminin (modelin deyil) parametridir. Buna görə də, o, öyrənmə alqoritminin özündən təsirlənmir; təlimdən əvvəl təyin edilməli və təlim müddətində sabit qalmalıdır. Əgər regulyarlaşdırma hiperparametrini çox böyük bir dəyərə təyin etsəniz, demək olar ki, düz bir model (yamac sıfır yaxın) əldə edərsiniz; öyrənmə alqoritmi təlim verilənlərinə uyğunlaşmayacaq, amma yaxşı bir həlli tapma ehtimalı azalacaq. Hiperparametrləri tənzimləmək, maşın öyrənmə sistemini qurmağın mühüm bir hissəsidir (növbəti fəsildə ətraflı bir nümunə görə bilərsiniz).

Təlim verilənlərində az uyğunlaşma (underfitting)

Az uyğunlaşma, çox sadə bir modelin verilənlərin əsas strukturunu öyrənməyə qadir olmadığı zaman baş verir. Məsələn, həyat məmənnuniyyəti üçün xətti model az uyğunlaşmağa meyillidir; reallıq sadəcə modeldən daha mürəkkəbdır, buna görə də onun proqnozları təlim nümunələrində belə səhv olacaq.

Bu problemi həll etmək üçün əsas seçimlər:

1. Daha güclü bir model seçmək, daha çox parametrlı.
2. Öyrənmə alqoritminə daha yaxşı xüsusiyyətlər təqdim etmək (xüsusiyyət mühəndisliyi).
3. Model üzərindəki məhdudiyyətləri azaltmaq (məsələn, regulyarlaşdırma hiperparametrini azaltmaq).

Addım geri atmaq

İndi maşın öyrənmə haqqında çox şey öyrəndiniz. Lakin biz çoxlu anlayışları keçdik, buna görə də bir az çəşqin hiss edə bilərsiniz, buna görə də gəlin, geri çəkilək və böyük şəkilə baxaq:

Maşın öyrənmə, maşınların verilənlərdən öyrənərk müəyyən bir vəzifəni daha yaxşı yerinə yetirməsini təmin etməkdir, əvəzinə qaydaları açıq şəkildə kodlaşdırmaq əvəzinə.

Maşın öyrənmə sistemlərinin bir çox fərqli növü var: nəzarət olunan və ya nəzarət olunmayan, kütləvi və ya onlayn, nümunə-əsaslı və ya model-əsaslı.

Bir maşın öyrənmə layihəsində verilənləri təlim verilənlərində toplayırsınız və təlim verilənlərini öyrənmə alqoritminə təqdim edirsiz. Əgər alqoritm model-əsaslıdırsa, modelə uyğunlaşdırmaq üçün bəzi parametrləri tənzimləyir (yəni, yalnız təlim verilənlərində yaxşı proqnozlar vermək üçün), və sonra ümid edirəm ki, yeni hallarda da yaxşı proqnozlar verə biləcək. Əgər alqoritm nümunə-əsaslırsa, o sadəcə nümunələri əzbərləyir və öyrənilmiş nümunələrlə müqayisə edərək yeni nümunələrə ümumiləşdirir.

Sisteminiz yaxşı işləməyəcək əgər təlim verilənləriniz çox kiçikdirsə, və ya verilənlər təmsilçi deyilsə, səs-küylüdürsə, və ya alakasız xüsusiyyətlərlə çirklənibsə (garbage in, garbage out). Nəhayət, modeliniz nə çox sadə olmalı (bu halda az uyğunlaşacaq), nə də çox mürəkkəb olmalıdır (bu halda çox uyğunlaşacaq).

Son bir mühüm mövzu var: bir model öyrəndikdən sonra, sadəcə “ümid” etmək istəmirsiniz ki, o, yeni hallara ümumiləşəcək. Onu qiymətləndirmək və lazımdıraq düzəliş etmək istəyirsiniz. Gəlin bunu necə edəcəyimizi görək.

Test etmək və təsdiqləmək

Bir modelin yeni hallara necə yaxşı ümumiləşəcəyini bilmək üçün yeganə yol onu yeni hallarda sınamaqdır. Bunu etmək üçün bir yol, modelinizi istehsalata çıxarmaq və onun necə performans göstərdiyini izləməkdir. Bu yaxşı işləyir, amma əgər modeliniz çox pisdirse, istifadəçiləriniz şikayət edəcək – bu, ən yaxşı fikir deyil.

Daha yaxşı bir seçim, verilənlərinizi iki hissəyə bölməkdir: təlim verilənləri və test verilənləri. Bu adlardan da başa düşüləcəyi kimi, modelinizi təlim verilənlərində öyrənirsiniz və test verilənlərində sınayırıınız. Yeni hallardakı səhv dərcəsi **ümumiləşdirmə səhvi** (və ya nümunə xaricindəki səhv) adlanır və modelinizi test verilənlərində qiymətləndirməklə, bu səhv haqqında bir təxmin əldə edirsiniz. Bu dəyər sizə modelinizin əvvəllər heç görmədiyi nümunələrdə necə işləyəcəyini bildirir.

Əgər təlim səhvi aşağıdır (yəni, modeliniz təlim verilənlərində çox az səhv edir), amma ümumiləşdirmə səhvi yüksəkdirsə, bu, modelinizin təlim verilənlərinə uyğunlaşdığını göstərir.

İpucu

Verilənlərin 80%-ni təlim üçün istifadə etmək və 20%-ni test etmək adı bir təcrübədir. Lakin bu, verilənlər dəstəsinin ölçüsündən asılıdır: əgər 10 milyon nümunə varsa, 1%-i saxlayaraq test verilənləri dəstəsi 100,000 nümunə olacaq, ki bu da ümumiləşdirmə səhvini yaxşı qiymətləndirmək üçün çox güman ki, yetərli olacaq.

Hiperparametr Tənzimləməsi və Model Seçimi

Bir modeli qiymətləndirmək olduqca sadədir: sadəcə test dəstini istifadə edin. Lakin, iki növ model arasında (məsələn, xətti model və ya polinomial model) seçim edərkən necə qərar verirsiniz? Bir seçim, hər iki modeli öyrətmək və test dəsti ilə onların necə ümumiləşdiyini müqayisə etməkdir.

İndi düşünək ki, xətti model daha yaxşı ümumiləşdirir, amma **overfitting**-i qarşısını almaq üçün bəzi regulyarlaşdırma tətbiq etmək istəyirsiniz. Suallar yaranır: regulyarlaşdırma hiperparametrinin dəyərini necə seçirsiniz? Bir seçim, 100 fərqli model öyrətmək və bu hiperparametr üçün 100 fərqli dəyər istifadə etməkdir. Təxmin edin ki, ən yaxşı hiperparametr dəyərini tapdınız ki, bu da ən aşağı ümumiləşdirmə səhvi ilə model yaradır — məsələn, yalnız 5% səhv. Bu modeli istehsalata çıxarırsınız, amma təəssüf ki, gözlədiyinizdən daha pis işləyir və 15% səhv verir. Nə baş verdi?

Problem ondadır ki, siz ümumiləşdirmə səhvini test dəstində bir neçə dəfə ölçünüz və modeli və hiperparametrləri həmin dəst üçün ən yaxşı model yaratmaq üçün uyğunlaşdırırdınız. Bu, modelin yeni verilənlər üzərində eyni şəkildə işləməyəcəyini göstərir.

Bu problemi həll etmək üçün ümumi bir həll yolu **holdout təsdiqləmə** adlanır (Şəkil 1-25): sadəcə təlim dəstinin bir hissəsini saxlayıb bir neçə namizəd modeli qiymətləndirir və ən yaxşısını seçirsiniz. Yeni saxlanılan dəst **təstiq dəsti** (və ya inkişaf dəsti, və ya dev dəsti) adlanır. Daha spesifik olaraq, siz müxtəlif hiperparametrlərə bir neçə modeli azaldılmış təlim dəstində (yəni, tam təlim dəstindən təstiq dəstini çıxararaq) öyrədirsiniz və ən yaxşı nəticə verən modeli təstiq dəstində seçirsiniz. Bu **holdout təsdiqləmə** prosesi başa çatdıqdan sonra, ən yaxşı modeli tam təlim dəstində (təstiq dəstini də daxil edərək) öyrədirsiniz və bu sizə son modeli təqdim edir. Nəhayət, bu son modeli test dəstində qiymətləndirirsiniz və ümumiləşdirmə səhvi haqqında bir təxmin əldə edirsiniz.

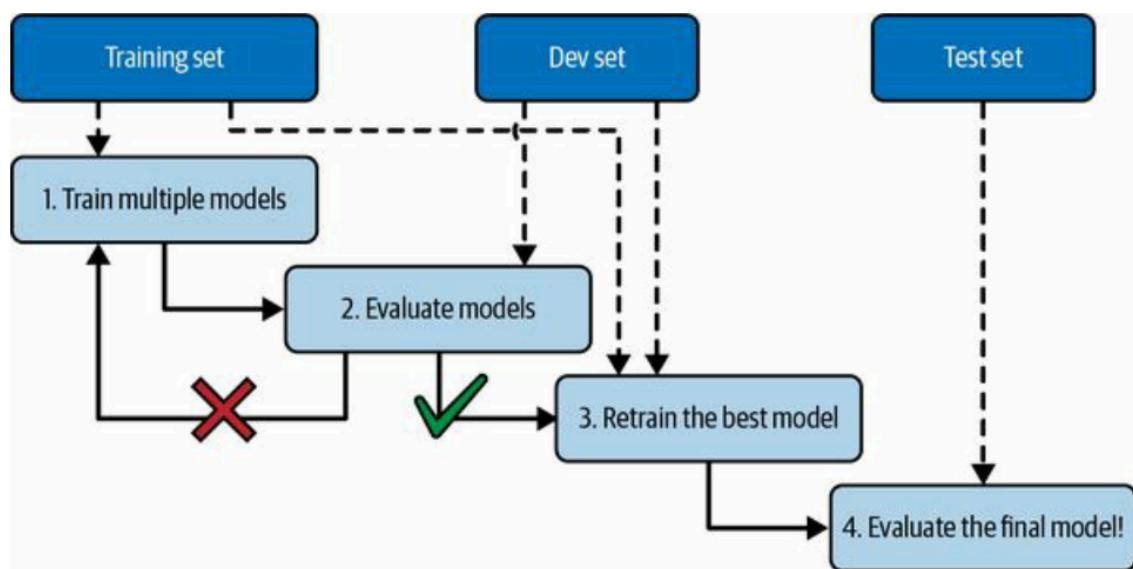


Figure 1-25. Model selection using holdout validation

Məlumat Uyğunsuzluğu

Bəzi hallarda, təlim üçün çox miqdarda məlumat əldə etmək asandır, amma bu məlumatlar istehsalatda istifadə ediləcək məlumatlarla tam uyğun olmaya bilər. Məsələn, deyək ki, bir mobil tətbiq yaratmaq istəyirsiniz ki, bu tətbiq güllərin şəkillərini çəksin və avtomatik olaraq növlərini təyin etsin. İnternetdən milyonlarla güllərin şəkilini asanlıqla yükləyə bilərsiniz, amma bunlar tətbiqin mobil cihazda çəkəcəyi şəkillərlə tam uyğun olmayıcaq. Bəlkə də yalnız 1,000 təmsilçi şəkiliniz var (yəni, tətbiq vasitəsilə çəkilənlər).

Bu halda, xatırlanmalı olan ən vacib qayda odur ki, həm təstiq dəsti, həm də test dəsti istehsalatda istifadə etməyi gözlədiyiniz verilənləri mümkün qədər yaxşı təmsil etməlidir. Buna görə də, bunlar yalnız təmsilçi şəkillərdən ibarət olmalıdır: onları qarışdırıb, yarısını təstiq dəstinə, yarısını isə test dəstinə yerləşdirə bilərsiniz (heç bir təkrarlanan və ya yaxın təkrarlanan şəkillərin hər iki dəstdə olmamasına diqqət yetirin). Əgər modelinizi vəb şəkilləri üzərində təlim etdikdən sonra, modelin təstiq dəstindəki performansının qənaətbəxş olmadığını müşahidə etsəniz, bunun səbəbi modelin təlim dəstinə overfit olub-olmaması və ya sadəcə vəb şəkilləri ilə mobil tətbiq şəkilləri arasındakı uyğunsuzluqdur.

Bir həll yolu, təlim şəkillərinizdən bəzilərini saxlayaraq, Andrew Ng-in "train-dev" dəsti adlandırdığı başqa bir dəst yaratmaqdır (Şəkil 1-26). Model təlim edildikdən sonra (təlim dəstində, train-dev dəstində deyil), onu train-dev dəstində qiymətləndirə bilərsiniz. Əgər modelin performansı pisdirsə, deməli, model təlim dəstinə overfit olub və modelinizi sadələşdirməli və ya regulyarlaşdırılmalıdır, daha çox təlim məlumatı əldə etməli və təlim məlumatlarını təmizləməlisiniz. Amma əgər train-dev dəstində yaxşı performans göstərisə, onda modelinizi dev dəstində qiymətləndirə bilərsiniz. Əgər burada performans pisdirsə, o zaman problem məlumat uyğunsuzluğundan gəlir. Bu problemi, vəb şəkillərini mobil tətbiq tərəfindən çəkilən şəkillərə bənzətmək üçün əvvəlcədən işləyərək həll etməyə cəhd edə bilərsiniz və sonra modeli yenidən təlim etdirə bilərsiniz. Bir dəfə modeliniz həm train-dev

dəstində, həm də dev dəstində yaxşı performans göstərdikdən sonra, onu son dəfə test dəstində qiymətləndirə bilərsiniz ki, istehsalatda necə performans göstərəcəyi barədə təxmin əldə edə biləsiniz.

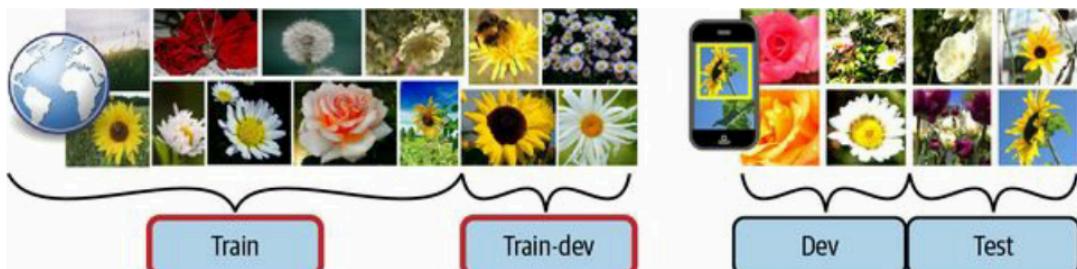


Figure 1-26. When real data is scarce (right), you may use similar abundant data (left) for training and hold out some of it in a train-dev set to evaluate overfitting; the real data is then used to evaluate data mismatch (dev set) and to evaluate the final model's performance (test set)

PULSUZ LUNCH TEOREMİ

Bir model, məlumatın sadələşdirilmiş bir təmsilidir. Sadələşdirmələr, yeni nümunələrə ümumiləşdirilə bilməyəcək artıq təfərruatları atmaq məqsədini güdürlər. Müəyyən bir model növünü seçərkən, siz məlumat haqqında qeyri-şəffaf fərziyyələr edirsınız. Məsələn, əgər bir xətti model seçirsinizsə, bu zaman məlumatın əsasən xətti olduğunu və nümunələr ilə düz xətt arasındakı məsafənin sadəcə səs-küy olduğunu və təhlükəsiz şəkildə gözardı edilə biləcəyini fərziyyə edirsınız.

1996-cı ildə David Wolpert məşhur bir məqaləsində göstərdi ki, əgər məlumat haqqında heç bir fərziyyə etməsəniz, onda bir modelin digərindən üstün olduğunu seçməyə heç bir səbəb yoxdur. Bu, Pulsuz Lunch (NFL) teoremi adlanır. Bəzi verilənlər dəstləri üçün ən yaxşı model xətti model, bəzilərində isə neyron şəbəkəsidir. Həmişə daha yaxşı işləyəcək bir model yoxdur (bundan dolayı teoremin adı). Hansı modelin ən yaxşı olduğunu əminliklə bilmək üçün onları hamisini qiymətləndirmək lazımdır. Lakin bu mümkün olmadığından, praktikada məlumat haqqında bəzi məqbul fərziyyələr edirsiniz və yalnız bir neçə məqbul modeli qiymətləndirirsınız. Məsələn, sadə tapşırıqlar üçün müxtəlif regulyarlaşdırma səviyyələri ilə xətti modelləri qiymətləndirə bilərsiniz, və daha mürəkkəb bir problem üçün müxtəlif neyron şəbəkələri qiymətləndirə bilərsiniz.

Təlimlər

Bu fəsildə biz maşın öyrənməsinin ən vacib anlayışlarını əhatə etdik. Növbəti fəsillərdə daha dərinə enəcək və daha çox kod yazacaqıq, amma etməzdən əvvəl aşağıdakı suallara cavab verə bildiyinizə əmin olun:

1. Maşın öyrənməsini necə tərif edərdiniz?
2. Onun parlaq olduğu dörd tətbiq sahəsini sadalaya bilərsinizmi?
3. Etiketli təlim dəsti nədir?
4. Ən çox rastlanan iki nəzarət olunan tapşırıq hansılardır?
5. Dörd ümumi nəzarətsiz tapşırıq adlandırma bilərsinizmi?

6. Robotu müxtəlif bilinməyən ərazilərdə gəzməyə imkan vermək üçün hansı növ alqoritm istifadə edərdiniz?
7. Müşterilərinizi müxtəlif qruplara ayırmaq üçün hansı növ alqoritm istifadə edərdiniz?
8. Spam aşkarlanması problemini nəzarət olunan öyrənmə problemi kimi, yoxsa nəzarətsiz öyrənmə problemi kimi çərçivələndirərdiniz?
9. Onlayn öyrənmə sistemi nədir?
10. Kənar öyrənmə nədir?
11. Hangi növ alqoritm proqnozlar vermək üçün oxşarlıq ölçüsünə əsaslanır?
12. Modelin parametri ilə modelin hipo-parametri arasındaki fərq nədir?
13. Model əsaslı alqoritmalar nə axtarır? Uğur qazanmaq üçün ən çox istifadə etdikləri strategiya nədir? Proqnozları necə edirlər?
14. Maşın öyrənməsində əsas dörd çətinliyi sadalaya bilərsinizmi?
15. Əgər modeliniz təlim verilənlərində çox yaxşı performans göstərisə, amma yeni nümunələrə ümumileşdirərkən pis nəticələr göstərisə, nə baş verir? Üç mümkün həlli adlandırma bilərsinizmi?
16. Test dəsti nədir və niyə istifadə etmək istəyərdiniz?
17. Validasiyanın məqsədi nədir?
18. Train-dev dəsti nədir, nə vaxt buna ehtiyacınız var və onu necə istifadə edirsınız?
19. Test dəsti istifadə edərək hiperparametrləri tənzimləsəniz nə səhv ola bilər?

Bu təlimlərin həlləri bu fəsilin notbukunun sonunda, <https://homl.info/colab3> ünvanında mövcuddur.

Fun fact: bu qəribə səslənən ad statistikada bir anlayışdır və Francis Galton tərəfindən təqdim edilib. O, uzun boylu insanların uşaqlarının adətən valideynlərindən qısa olduğunu araşdırarkən bu termini ortaya çıxarmışdır. Uşaqlar daha qısa olduqları üçün buna regression to the mean (orta qiymətə geri dönmə) adını vermişdir. Bu ad daha sonra dəyişənlər arasındaki əlaqələri təhlil etmək üçün istifadə etdiyi metodlara tətbiq olunmuşdur.

2. Notice how animals are rather well separated from vehicles and how horses are close to deer but far from birds. Şəkil, Richard Socher və digərlərindən “Zero-Shot Learning Through Cross-Modal Transfer”, Proceedings of the 26th International Conference on Neural Information Processing Systems 1 (2013): 935–943 məqaləsindən icazə ilə təkrarlanıb.
3. Bu, sistemin mükəmməl işlədiyi hallardır. Praktikada, çox vaxt hər bir insan üçün bir neçə klaster yaradır və bəzən oxşar görünən iki insanı qarışdırır, buna görə hər insan üçün bir neçə etiket təmin etməli və bəzi klasterləri əl ilə təmizləməli ola bilərsiniz.
4. Ümumi qaydaya görə, Yunan hərfi θ (theta) model parametrlərini təmsil etmək üçün tez-tez istifadə olunur.
5. Əgər hələ bütün kodu başa düşmürsünüzsə, narahat olmayın; Scikit-Learn-u növbəti fəsillərdə təqdim edəcəyəm.
6. Məsələn, kontekstdən asılı olaraq "to", "two" və ya "too" yazmaq lazımlı olub-olmadığını bilmək.

7. Peter Norvig və digərləri, "The Unreasonable Effectiveness of Data", IEEE Intelligent Systems 24, no. 2 (2009): 8–12.
8. Şəkil, Michele Banko və Eric Brill tərəfindən "Scaling to Very Very Large Corpora for Natural Language Disambiguation", Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics (2001): 26–33 məqaləsindən icazə ilə təkrarlanıb.
9. David Wolpert, "The Lack of A Priori Distinctions Between Learning Algorithms", Neural Computation 8, no. 7 (1996): 1341–1390.