یادگیری ماشین عملی با Scikit-Learn، Keras و TensorFlow

مفاهیم، ابزارها و تکنیک‌هایی برای ساخت سیستم‌های هوشمند

(یادگیری ماشین با پایتون)

مولف:

اورلین جرون

مترجمین:

دکتر جواد وحیدی – (عضو هیات علمی دانشگاه علم و صنعت ایران)

مهندس مژگان الیکائی آهاری

فن‌آوری نوین

سرشناسه	:	گرون، اورلین Géron, Aurélien
عنوان و نام پدیدآور	:	یادگیری ماشین عملی با Scikit- learn, keras و TensorFlow: مفاهیم، ابزارها و تکنیک‌هایی برای ساخت سیستم‌های هوشمند (یادگیری ماشین با پایتون)/ مولف اورلین جرون؛ مترجمین جواد وحیدی، مژگان الیکائی آهاری.
مشخصات نشر	:	بابل: فناوری نوین، ۱۴۰۰.
مشخصات ظاهری	:	۴۷۴ ص.: مصور، جدول، نمودار.
شابک	:	978-622-7393-61-3
وضعیت فهرست نویسی	:	فیپا
یادداشت	:	عنوان اصلی: Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow : ۲nd ed, ۲concepts, tools, and techniques to build intelligent systems, ۲۰۱۹.
عنوان دیگر	:	یادگیری ماشین با پایتون.
موضوع	:	تنسورفلو TensorFlow
موضوع	:	پایتون (زبان برنامه‌نویسی کامپیوتر) Python (Computer program language) فراگیری ماشینی Machine learning هوش مصنوعی Artificial intelligence
شناسه افزوده	:	وحیدی، جواد، ۱۳۴۸ –، مترجم
شناسه افزوده	:	Vahidi, Javad
شناسه افزوده	:	الیکائی آهاری، مژگان، ۱۳۶۸–، مترجم
رده بندی کنگره	:	QA۷۶/۷۳
رده بندی دیویی	:	۰۰۵/۱۳۳
شماره کتابشناسی ملی	:	۸۵۷۲۰۲۶
اطلاعات رکورد کتابشناسی	:	فیپا

فن‌آوری نوین www.novinpub.ir بابل، کد پستی ۴۷۱۶۷-۷۳۴۴۸ تلفن: ۰۱۱-۳۲۲۵۶۶۸۷

یادگیری ماشین عملی با Scikit-Learn، Keras وTensorFlowمفاهیم، ابزارها و تکنیک‌هایی برای ساخت سیستم‌های هوشمند(یادگیری ماشین با پایتون)

ترجمه: جواد وحیدی، مژگان الیکائی آهاری.

نوبت چاپ: چاپ اول

سال چاپ: پاییز ۱۴۰۰

شمارگان: ۲۰۰

قیمت: ۱۳۸۰۰۰ تومان

نام چاپخانه و صحافی: دفتر فنی سورنا

طراح جلد: مهندس الهام بیده

شابک چاپی: 978-622-7393-61-3

شابک الکترونیکی: 978-622-7393-62-0

نشانی ناشر: بابل، چهارراه نواب، کاظم‌بیگی، جنب مسجد منصور کاظم‌بیگی، طبقه اول

فروشگاه و پخش کتاب چاپی: تهران، تلفن ۶۶۴۰۰۲۲۰– ۶۶۴۰۰۱۴۴

تهران، خ اردیبهشت، نبش وحید نظری، پلاک ۱۴۲ تلفکس: ۶۶۴۰۰۲۲۰-۶۶۴۰۰۱۴۴

فهرست مطالب

فصل اول: چشم‌انداز یادگیری ماشین

۱-۱. یادگیری ماشین چیست؟..... ۱۵

۱-۲. چرا باید از یادگیری ماشینی استفاده کرد؟..... ۱۶

۱-۳. انواع سیستم‌های یادگیری ماشین..... ۱۹

۱-۳-۱. یادگیری با ناظر / بدون ناظر..... ۲۰

۱-۳-۲. یادگیری برخط و دسته‌ای..... ۲۷

۱-۳-۳. یادگیری مبتنی بر نمونه در مقابل مبتنی مدل ۳۰

۱-٤. چالش‌های اصلی یادگیری ماشین..... ۳٦

۱-٤-۱. اثربخشی نامعقول داده‌ها..... ۳٦

۱-٤-۲. داده‌های آموزشی غیر معرف..... ۳۷

۱-٤-۳. داده‌های کم کیفیت..... ۳۹

۱-٤-٤. ویژگی‌های نامربوط..... ۳۹

۱-٤-٥. بیش برازش داده‌های آموزش..... ٤۰

۱-٤-٦. کم برازش داده‌های آموزش..... ٤۲

۱-٤-۷. عقب‌نشینی..... ٤۲

۱-٥. تست و اعتبار سنجی ٤۳

۱-٥-۱. تنظیم ابر پارامترها و انتخاب مدل..... ٤۳

۱-٥-۲. عدم تطابق داده‌ها..... ٤٤

۱-٦. تمرینات..... ٤٦

فصل دوم: پروژه یادگیری ماشین End-to-End

۲-۱-۱. کار با داده‌های واقعی..... ٤۷

۲-۱-۲. به تصویر بزرگ نگاه کنید..... ٤۸

۲-۱-۳. مسئله را قاب‌بندی کنید..... ٤۸

۲-۱-٤. یک معیار عملکرد را انتخاب کنید..... ٥۰

۲-۱-٥. مفروضات را بررسی کنید..... ٥۳

۲-۲. اطلاعات را دریافت کنید..... ٥۳

۲-۲-۱. ایجاد محیطی جداگانه..... ٥٤

۲-۲-۲. بارگیری اطلاعات..... ٥۷

۲-۲-۳. نگاهی گذرا به ساختار داده بیندازید..... ٥۸

۲-۲-٤. یک مجموعه تست ایجاد کنید..... ٦۲

۲-۳. برای به دست آوردن بینش، داده‌ها را کشف و تجسم کنید..... ٦۷

۲-۳-۱. مصورسازی داده‌های جغرافیایی..... ٦۷

۲-۳-۲. جستجو برای همبستگی‌ها..... ۷۰

۲-۳-۳. تجربه کردن با ترکیبات ویژگی..... ۷۳

۲-٤. داده‌ها را برای الگوریتم‌های یادگیری ماشین آماده کنید ۷٤

۲-٤-۱. پاک‌سازی داده‌ها..... ۷٥

۲-٤-۲. جستجو برای همبستگی‌ه رسیدگی به متن و خصوصیات طبقه‌ای..... ۷۸

۲-٤-۳. مبدل‌های سفارشی..... ۸۰

۲-٤-٤. مقیاس‌گذاری ویژگی..... ۸۱

۲-٤-٥. تبدیل پایپلاین..... ۸۲

۲-٥. یک مدل را انتخاب و آموزش دهید..... ۸٤

۲-٥-۱. آموزش و ارزیابی روی مجموعه آموزش ۸٤

۲-٥-۲. ارزیابی بهتر با استفاده از اعتبارسنجی متقابل..... ۸٦

۲-٦. مدل خود را خوب تنظیم کنید..... ۸۸

۲-٦-۱. جستجوی شبکه‌ای..... ۸۸

۲-٦-۲. جستجوی تصادفی..... ۹۰

۲-٦-۳. روش‌های جمعی..... ۹۱

۲-٦-٤. بهترین مدل‌ها و خطاهای آن‌ها را تحلیل کنید ۹۱

۲-٦-٥. سیستم خود را در مجموعه آزمایش ارزیابی کنید ۹۲

۲-٦-٦. سیستم خود را راه‌اندازی، نظارت و نگهداری کنید ۹۳

۲-۷. تمرینات..... ۹٤

فصل سوم: طبقه‌بندی

۳-۱. MNIST..... ۹٥

۳-۲. آموزش کلاس باینری..... ۹۷

۳-۳. اندازه گیری عملکرد.....۹۸
۱-۳-۳. اجرای اعتبار سنجی متقابل.....۹۸
۲-۳-۳. ماتریس درهم ریختگی.....۱۰۰
۳-۳-۳. دقت و بازیابی.....۱۰۲
٤-۳-۳. مصالحه دقت / بازیابی.....۱۰۳
٥-۳-۳. منحنی ROC.....۱۰۷
٤-۳. طبقه بندی چند کلاسه.....۱۱۰
٥-۳. تجزیه و تحلیل خطا.....۱۱۳
٦-۳. طبقه بندی چند برچسبه.....۱۱۷
۷-۳. طبقه بندی چند خروجی.....۱۱۸
۸-۳. تمرینات.....۱۱۹

فصل چهارم: مدل‌های یادگیری

۱-٤. رگرسیون خطی ۱۲۲
۱-۱-٤. معادله نرمال.....۱۲۳
۲-۱-٤. پیچیدگی محاسباتی.....۱۲٦
۲-٤. گرادیان کاهشی.....۱۲٦
۱-۲-٤. گرادیان کاهشی Batch.....۱۳۰
۲-۲-٤. نرخ همگرایی.....۱۳۱
۳-۲-٤. گرادیان کاهشی تصادفی.....۱۳۱
٤-۲-٤. گرادیان کاهشی Mini-batch.....۱۳۲
۳-٤. رگرسیون چندجمله‌ای.....۱۳٦
٤-٤. منحنی‌های یادگیری.....۱۳۸
٥-٤. تعادل بین بایاس و واریانس.....۱٤۲
٦-٤. مدل‌های خطی منظم.....۱٤۲
۱-٦-٤. رگرسیون ستیغی.....۱٤۳
۲-٦-٤. رگرسیون لاسو.....۱٤٤
۳-٦-٤. نت الاستیک.....۱٤۷
٤-٦-٤. توقف زودهنگام.....۱٤۷
۷-٤. رگرسیون لجستیک.....۱٤۹
۱-۷-٤. تخمین احتمالات ۱٤۹
۲-۷-٤. تابع هزینه و آموزش.....۱٥۰
۳-۷-٤. مرزهای تصمیم‌گیری.....۱٥۱
٤-۷-٤. آنتروپی متقاطع.....۱٥٥
۸-٤. تمرینات.....۱٥۷

فصل پنجم: ماشین بردار پشتیبان

۱-٥. طبقه‌بندی خطی SVM.....۱٥۹
۱-۱-٥. طبقه‌بندی حاشیه نرم.....۱٦۰
۲-٥. طبقه‌بندی غیرخطی SVM.....۱٦۲
۱-۲-٥. هسته چندجمله‌ای.....۱٦۳
۲-۲-٥. اضافه کردن ویژگی‌های شباهت.....۱٦٤
۳-۲-٥. هسته تابع شعاعی پایه گوسین.....۱٦٥
٤-۲-٥. پیچیدگی محاسباتی.....۱٦۷
۳-٥. رگرسیون SVM.....۱٦۷
٤-٥. در زیر Hood.....۱٦۹
۱-٤-٥. تابع تصمیم‌گیری و پیش‌بینی‌ها.....۱۷۰
۲-٤-٥. هدف آموزش.....۱۷۱
۳-٤-٥. برنامه‌نویسی درجه دوم.....۱۷۲
٤-٤-٥. مساله دوگان.....۱۷۳
٥-٤-٥. SVM هسته‌دارشده.....۱۷٤
٦-٤-٥. تئوری مرسر.....۱۷٥

۷-٤-٥. اتلاف هینج ۱۷۷
۷-٥. تمرینات ۱۷۷
فصل ششم: درخت تصمیم‌گیری
۱-٦. آموزش و تجسم درخت تصمیم ۱۷۹
۲-٦. پیش‌بینی کردن ۱۸۰
۳-٦. تخمین احتمالات کلاس ۱۸۲
٤-٦. الگوریتم آموزش CART ۱۸۳
٥-٦. پیچیدگی محاسباتی ۱۸۳
٦-٦. ناخالصی جینی یا آنتروپی؟ ۱۸٤
۷-٦. تنظیم ابر پارامترها ۱۸٤
۸-٦. رگرسیون ۱۸٦
۹-٦. بی‌ثباتی ۱۸۸
۱۰-٦. تمرینات ۱۸۹
فصل هفتم: یادگیری جمعی و جنگل تصادفی
۱-۷. طبقه‌بندی کننده رأی‌گیری ۱۹۱
۲-۷. بگینگ و Pasting ۱۹٥
۱-۲-۷. بگینگ و Pasting در Scikit-Learn ۱۹٦
۲-۲-۷. محاسبات Out-of-Bag ۱۹۷
۳-۷. پچ‌های تصادفی و زیر فضاهای تصادفی ۱۹۸
٤-۷. جنگل‌های تصادفی ۱۹۹
۱-٤-۷. درختان اضافی ۲۰۰
۲-٤-۷. اهمیت ویژگی ۲۰۰
٥-۷. تقویت ۲۰۲
۱-٥-۷. آدابوست ۲۰۲
۲-٥-۷. گرادیان تقویتی ۲۰٥
٦-۷. پشته‌سازی ۲۱۰
فصل هشتم: کاهش ابعاد
۱-۸. مشقت بعد چندی ۲۱٥
۲-۸. رویکردهای اصلی کاهش ابعاد ۲۱٦
۱-۲-۸. نگاشت ۲۱۷
۲-۲-۸. یادگیری منیفولد ۲۱۸
۳-۸. تحلیل مؤلفه‌های اصلی ۲۲۰
۱-۳-۸. Preserving واریانس ۲۲۰
۲-۳-۸. مؤلفه‌های اصلی ۲۲۱
۳-۳-۸. نگاشت به بعد پایین‌تر تا d بعد ۲۲۲
٤-۳-۸. استفاده از Scikit-Learn ۲۲۳
٥-۳-۸. نرخ واریانس توضیح داده‌شده ۲۲۳
٦-۳-۸. انتخاب تعداد مناسب ابعاد ۲۲۳
۷-۳-۸. PCA برای فشرده‌سازی ۲۲٤
۸-۳-۸. PCA تصادفی ۲۲٥
۹-۳-۸. PCA افزایشی ۲۲٦
٤-۸. آنالیز مؤلفه‌های اصلی مبتنی بر هسته ۲۲۷
۱-٤-۸. انتخاب و تنظیم ابر پارامترهای هسته ۳۲۷
٥-۸. جاسازی خطی محلی ۲۳۰
٦-۸. سایر تکنیک‌های کاهش ابعاد ۲۳۲
۷-۸. تمرینات ۲۳۳
فصل نهم: تکنیک‌های یادگیری بدون نظارت
۱-۹. خوشه‌بندی ۲۳٦
۱-۱-۹. K-Means ۲۳۸
۲-۱-۹. محدودیت‌های K-Means ۲٤۷

۳-۱-۹. استفاده از خوشه‌بندی برای بخش‌بندی تصویر ۲٤۸
٤-۱-۹. استفاده از خوشه‌بندی برای پیش‌پردازش ۲۵۰
۵-۱-۹. استفاده از خوشه‌بندی برای یادگیری نیمه نظارت‌شده ۲۵۱
٦-۱-۹. DBSCAN ۲۵۵
۲-۹. مخلوط‌های گاوسی ۲۵۷
۱-۲-۹. تشخیص ناهنجاری با استفاده از مخلوط‌های گاوسی ۲٦۳
۲-۲-۹. انتخاب تعداد خوشه‌ها ۲٦۵
۳-۲-۹. مدل‌های مخلوط گاوسی بیزی ۲٦۸
٤-۲-۹. سایر الگوریتم‌های تشخیص ناهنجاری و تازگی ۳۷۱
فصل دهم: مقدمه‌ای بر شبکه‌های عصبی مصنوعی با Keras
۱-۱۰. از سلول‌های عصبی بیولوژیکی گرفته تا مصنوعی ۲۷۳
۱-۱-۱۰. نورون‌های بیولوژیکی ۲۷٤
۲-۱-۱۰. محاسبات منطقی با سلول‌های عصبی ۲۷۵
۳-۱-۱۰. پرسپترون ۲۷٦
٤-۱-۱۰. پرسپترون چند لایه و پس انتشار ۲۸۱
۵-۱-۱۰. رگرسیون MLPs ۲۸۳
٦-۱-۱۰. طبقه‌بندی MLPها ۲۸٤
۲-۱۰. پیاده‌سازی MLPها با Keras ۲۸٦
۱-۲-۱۰. نصب TensorFlow ۲ ۲۸٦
۲-۲-۱۰. ساخت یک طبقه‌بندی تصویر با استفاده از API پی‌درپی(متوالی) ۲۸۷
۳-۲-۱۰. ایجاد مدل با استفاده از Sequential API ۲۸۹
٤-۲-۱۰. ساخت MLP رگرسیون با استفاده از API متوالی ۲۹٦
۵-۲-۱۰. ساخت مدل‌های پیچیده با استفاده از API عملکردی ۲۹۷
٦-۲-۱۰. ساخت مدل‌های پویا با استفاده از Subclassing API ۳۰۲
۷-۲-۱۰. ذخیره و بازیابی یک مدل ۳۰۳
۸-۲-۱۰. استفاده از بازخوانی ۳۰٤
۹-۲-۱۰. تصویرسازی با استفاده از TensorBoard ۳۰٦
۳-۱۰. تنظیم خوب ابرپارامترهای شبکه عصبی ۳۰۸
۱-۳-۱۰. تعداد لایه‌های پنهان ۳۱۲
۲-۳-۱۰. تعداد نورون‌ها در هر لایه پنهان ۳۱۳
۳-۳-۱۰. نرخ یادگیری، اندازه دسته و سایر پارامترهای دیگر ۳۱٤
٤-۱۰. تمرینات ۳۱٦
فصل یازدهم: آموزش شبکه‌های عصبی عمیق
۱-۱۱. مشکل محوشدگی/انفجار گرادیان ۳۱۹
۱-۱-۱۱. مقداردهی اولیه HE و Glorot ۳۲۱
۲-۱-۱۱. توابع Nonsaturating Activation ۳۲۲
۳-۱-۱۱. نرمال‌سازی دسته‌ای ۳۲٦
٤-۱-۱۱. برش گرادیان ۳۳۱
۲-۱۱. استفاده مجدد از لایه‌های پیش آموزش دیده ۳۳۲
۱-۲-۱۱. یادگیری انتقال با Keras ۳۳۳
۲-۲-۱۱. پیش آموزش بدون نظارت ۳۳۵
۳-۲-۱۱. پیش آموزش روی یک وظیفه کمکی ۳۳٦
۳-۱۱. بهینه سازهای سریع‌تر ۳۳۷
۱-۳-۱۱. بهینه‌سازی ممنتوم ۳۳۷
۲-۳-۱۱. گرادیان شتاب‌یافته‌ی نستروف ۳۳۸
۳-۳-۱۱. AdaGrad ۳۳۹
٤-۳-۱۱. RMSProp ۳٤۱
۵-۳-۱۱. بهینه‌سازی Adam و Nadam ۳٤۱
٦-۳-۱۱. آموزش مدل‌های اسپارس ۳٤۳
۷-۳-۱۱. زمان‌بندی نرخ یادگیری ۳٤۳

۱۱-٤. جلوگیری از بیش برازش هنگام تنظیم پارامتر ۳٤۸
۱۱-٤-۱. تنظیم ℓ_2 و ℓ_1 ..... ۳٤۸
۱۱-٤-۲. Dropout ۳٤۹
۱۱-٤-۳. Dropout مونت کارلو ۳۵۱
۱۱-٤-٤. تنظیم‌کننده Max-Norm ۳۵۳
۱۱-۵. رهنمودهای خلاصه و عملی ۳۵٤
۱۱-٦. تمرینات ۳۵۵
فصل دوازدهم: مدل‌های سفارشی و آموزش با TensorFlow
۱۲-۱. گشت‌وگذار سریع در TensorFlow ۳۵۹
۱۲-۲. استفاده از TensorFlow مانند NumPy ۳٦۲
۱۲-۲-۱. تنسورها و عملیات ۳٦۲
۱۲-۲-۲. حسگرها و NumPy ۳٦٤
۱۲-۲-۳. تبدیل نوع ۳٦٤
۱۲-۲-٤. متغیرها ۳٦۵
۱۲-۲-۵. سایر ساختارهای داده ۳٦۵
۱۲-۳. سفارشی کردن مدل‌ها و الگوریتم‌های آموزش ۳٦۷
۱۲-۳-۱. توابع از دست دادن سفارشی ۳٦۷
۱۲-۳-۲. ذخیره و بارگیری مدل‌هایی که دارای اجزای سفارشی هستند ۳٦۸
۱۲-۳-۳. توابع فعال‌سازی سفارشی، مقدار اولیه دهنده، تنظیم‌کننده‌های پارامتر و محدودیت‌ها ۳۷۰
۱۲-۳-٤. معیارهای سفارشی ۳۷۱
۱۲-۳-۵. لایه‌های سفارشی ۳۷٤
۱۲-۳-٦. مدل‌های سفارشی ۳۷۷
۱۲-۳-۷. معیار و زیان بر اساس مدل داخلی ۳۷۹
۱۲-۳-۸. محاسبه گرادیان‌ها با استفاده از Autodiff ۳۸۱
۱۲-۳-۹. حلقه‌های آموزشی سفارشی ۳۸۵
۱۲-٤. توابع و نمودارهای TensorFlow ۳۸۸
۱۲-٤-۱. رونویسی و ردیابی ۳۸۹
۱۲-٤-۲. قوانین تابع TF ۳۹۱
فصل سیزدهم: بارگیری و پیش پردازش داده‌ها با TensorFlow
۱۳-۱. API داده ۳۹٤
۱۳-۱-۱. تبدیل زنجیره‌ای ۳۹۵
۱۳-۱-۲. برُ زدن داده ۳۹٦
۱۳-۱-۳. جایگذاری خطوط از چندین فایل ۳۹۸
۱۳-۱-٤. پیش پردازش داده ها ۳۹۹
۱۳-۱-۵. همه چیز را کنار هم قرار دهید ٤۰۰
۱۳-۱-٦. پیش واکشی ٤۰۱
۱۳-۱-۷. استفاده از مجموعه داده با tf.keras ٤۰۲
۱۳-۲. قالب TFRecord ٤۰٤
۱۳-۲-۱. فایل‌های فشرده TFRecord ٤۰۵
۱۳-۲-۲. مقدمه ای مختصر بر Protocol Buffers. ٤۰۵
۱۳-۲-۳. TensorFlow Protobufs ٤۰۷
۱۳-۲-٤. بارگیری و تجزیه نمونه‌ها ٤۰۸
۱۳-۲-۵. مدیریت لیست ها با استفاده از SequenceExample protobuf ٤۱۰
۱۳-۳. ویژگی‌های API ٤۱۱
۱۳-۳-۱. ویژگی‌های طبقه بندی شده ٤۱۲
۱۳-۳-۲. ویژگی‌های طبقه بندی متقاطع ٤۱۳
۱۳-۳-۳. رمزگذاری ویژگی‌های طبقه بندی با استفاده از بردارهای One-Hot ٤۱٤
۱۳-۳-٤. رمزگذاری ویژگی‌های دسته بندی با استفاده از جاسازی‌ها ٤۱٤
۱۳-۳-۵. جاسازی کلمه ٤۱۵
۱۳-۳-٦. استفاده از ستون‌های ویژگی برای تجزیه ٤۱۷

۷-۳-۱۳. استفاده از ستون‌های ویژگی در مدل..... ٤١٧
١٣-٤. تبدیل TF..... ٤١٩
١٣-٥. پروژه مجموعه داده‌های TensorFlow (TFDS)..... ٤٢١

فصل چهاردهم: عمیق کامپیوتری با استفاده از شبکه‌های عصبی پیچشی

١٤-١. معماری قشر بصری..... ٤٢٣
١٤-٢. لایه پیچشی ٤٢٤
١٤-٢-١. فیلترها ٤٢٦
١٤-٢-٢. پشته‌سازی چند ویژگی نقشه‌ها..... ٤٢٧
١٤-٢-٣. پیاده‌سازی TensorFlow..... ٤٢٩
١٤-٢-٤. الزامات حافظه..... ٤٣٢
١٤-٣. لایه pooling..... ٤٣٣
١٤-٣-١. پیاده‌سازی TensorFlow..... ٤٣٥
١٤-٤. معماری CNN..... ٤٣٧
١٤-٤-١. LeNet-٥..... ٤٣٩
١٤-٤-٢. AlexNet..... ٤٤٠
١٤-٤-٣. GoogLeNet ٤٤٣
١٤-٤-٤. VGGNet..... ٤٤٧
١٤-٤-٥. ResNet..... ٤٤٧
١٤-٤-٦. Xception..... ٤٥٠
١٤-٤-٧. SENet..... ٤٥٢
١٤-٥. پیاده‌سازی ResNet-٣٤ CNN با استفاده از Keras..... ٤٥٤
١٤-٦. مدل‌های از پیش آموزش‌دیده شده برای یادگیری انتقالی..... ٤٥٨
١٤-٨. تشخیص شیء ٤٦٠
١٤-٧. طبقه‌بندی و محلی سازی ٤٦٢
١٤-٨-١. شبکه‌های کاملاً پیچشی (FCN)..... ٤٦٤
١٤-٨-٢. شما فقط یک‌بار نگاه می‌کنید (YOLO)..... ٤٦٦
١٤-٨-٣. متوسط میانگین دقت (mAP) ٤٦٨
١٤-٩. تقسیم‌بندی معنایی..... ٤٦٩
١٤-١٠. تمرینات..... ٤٧٣

مقدمه

سونامی یادگیری ماشین

در سال ۲۰۰۶، جفری هینتون و همکارانش[1] مقاله‌ای منتشر کردند که نشان می‌داد چگونه آموزش یک شبکه عصبی عمیق قادر به تشخیص ارقام دست‌نویس با دقت پیشرفته (< ۹۸>) را است. آن‌ها این تکنیک را یادگیری عمیق نام‌گذاری کردند. در آن زمان آموزش شبکه عصبی عمیق غیرممکن تلقی می‌شد و اکثر محققان از دهه ۱۹۹۰ این ایده را کنار گذاشته بودند. نتایج این مقاله علاقه جامعه علمی را در مورد یادگیری عمیق دوباره زنده کرد و نشان داد که یادگیری عمیق نه‌تنها امکان‌پذیر است، بلکه می‌تواند دستاوردهای حیرت‌انگیزی داشته باشد که هیچ تکنیک یادگیری ماشین (ML) نمی‌تواند با آن‌ها مطابقت داشته باشد. این اشتیاق به‌زودی به بسیاری از زمینه‌های دیگر یادگیری ماشین گسترش یافت.

۱۰یادگیری ماشین صنعت را تسخیر کرده است و اکنون در قلب بسیاری از جادوهای محصولات با تکنولوژی قرار دارد. یادگیری ماشین نتایج جستجوی وب را رتبه‌بندی کرده، تشخیص گفتار تلفن هوشمند را تقویت و به شما ویدیو توصیه می‌کند.

یادگیری ماشین در پروژه‌های شما

آیا به یادگیری ماشین علاقه دارید و دوستدارید به این داستان بپیوندید؟

شاید شرکت شما دارای تعداد زیادی داده (گزارشات کاربر، داده‌های مالی، داده‌های تولید، داده‌های حسگر ماشین، آمار خط تلفن، گزارشات منابع انسانی و غیره) باشد و به‌احتمال‌زیاد اگر بدانید کجا باید نگاه کنید برخی مشکلات حل می‌شوند:

- مشتریان را تقسیم‌بندی کرده و بهترین استراتژی بازاریابی را برای هر گروه پیدا کنید.
- محصولات را برای هر مشتری بر اساس خرید مشتریان مشابه توصیه کنید.
- تشخیص دهید کدام تراکنش‌ها احتمالاً تقلبی هستند.
- پیش‌بینی درآمد سال آینده.
- و غیره

به هر دلیلی تصمیم گرفته‌اید یادگیری ماشین را بیاموزید و آن را در پروژه‌های خود پیاده‌سازی کنید.

هدف و رویکرد

این کتاب فرض می‌کند که شما تقریباً هیچ‌چیز درباره یادگیری ماشین نمی‌دانید. هدف آن ارائه مفاهیم، شهود و ابزارهایی است که برای اجرای برنامه‌های قابل یادگیری از داده‌ها نیاز دارید.

ما تکنیک‌های زیادی را پوشش می‌دهیم، از ساده‌ترین و متداول‌ترین (مانند رگرسیون خطی) تا برخی از تکنیک‌های یادگیری عمیق که به‌طور منظم در مسابقات برنده می‌شوند.

به‌جای اجرای نسخه‌های خودمان از هر الگوریتم، از چارچوب‌های آماده تولید پایتون استفاده می‌کنیم:

- استفاده از Scikit-Learn بسیار آسان است، بااین‌حال بسیاری از الگوریتم‌های یادگیری ماشین را

[1] http://www.cs.toronto.edu/~hinton/

به‌طور کارآمد پیاده‌سازی می‌کند، بنابراین یک نقطه ورود عالی برای یادگیری ماشین است.

- TensorFlow یک کتابخانه پیچیده‌تر برای محاسبه عددی توزیع‌شده است. این امکان را فراهم می‌کند که شبکه‌های عصبی بسیار بزرگ را با تقسیم محاسبات در صدها سرور چند GPU به‌طور کارآمد آموزش داده و اجرا کنیم. TensorFlow در Google ایجادشده و از بسیاری از برنامه‌های بزرگ یادگیری ماشین را پشتیبانی می‌کند. این منبع در نوامبر ۲۰۱۵ باز شد.
- Keras یک API یادگیری عمیق سطح بالا است که آموزش و اجرای شبکه‌های عصبی را بسیار ساده می‌کند. می‌تواند روی TensorFlow، Theano یا Micro soft Cognitive Toolkit(که قبلاً به‌عنوان CNTK شناخته می‌شد) اجرا شود. TensorFlow با پیاده‌سازی خود از این API، به نام tf.keras، پشتیبانی می‌کند که از برخی ویژگی‌های پیشرفته TensorFlow(به‌عنوان‌مثال برای بارگذاری کارآمد داده‌ها) پشتیبانی می‌کند.

این کتاب از یک رویکرد عملی استفاده می‌کند و درک شهودی از یادگیری ماشین را از طریق مثال‌های عملی و کمی تئوری افزایش می‌دهد. توصیه می‌کنیم نمونه‌های کد موجود در اینترنت را به‌عنوان نوتبوک Jupyter در https://github.com/ageron/handson-ml۲ بررسی کنید.

پیش‌نیازها

این کتاب فرض می‌کند که شما برخی از تجربه برنامه‌نویسی پایتون را دارید و با کتابخانه‌های اصلی علمی پایتون، به‌ویژه NumPy، Pandas و Matplotlib آشنا هستید.

همچنین، اگر به موارد زیر توجه دارید، باید از ریاضیات سطح دانشگاه نیز درک خوبی داشته باشید (حساب، جبر خطی، احتمالات و آمار).

اگر هنوز پایتون را نمی‌شناسید، http://learnpython.org/ مکان عالی برای شروع است. آموزش رسمی در python.org نیز بسیار خوب است.

اگر هرگز از Jupyter استفاده نکرده‌اید، فصل ۲ شما را در مورد نصب و اصول اولیه راهنمایی می‌کند.

اگر با کتابخانه‌های علمی پایتون آشنایی ندارید، کتاب‌های ارائه‌شده Jupyter note include شامل چند آموزش است. همچنین یک آموزش ریاضی سریع برای جبر خطی وجود دارد.

نقشه راه

این کتاب در دو بخش تنظیم‌شده است. **قسمت اول، مبانی یادگیری ماشین، موضوعات زیر را پوشش می‌دهد:**

- یادگیری ماشین چیست؟ سعی می‌کند چه مشکلاتی را حل کند؟ مقوله‌های اصلی و مفاهیم اساسی سیستم‌های یادگیری ماشین چیست؟
- مراحل اصلی در یک پروژه معمول یادگیری ماشین.
- یادگیری با متناسب‌سازی مدل با داده‌ها.

- بهینه‌سازی عملکرد هزینه.
- رسیدگی، تمیز کردن و آماده‌سازی داده‌ها.
- انتخاب و مهندسی ویژگی‌ها.
- انتخاب مدل و تنظیم پارامترها با استفاده از اعتبار متقابل.
- چالش‌های اصلی یادگیری ماشین، به‌ویژه بیش برازش و کم برازش (معامله(موازنه) بایاس/ واریانس).
- کاهش ابعاد داده‌های آموزش برای مبارزه با نکبت ابعاد.
- سایر تکنیک‌های یادگیری بدون نظارت، ازجمله خوشه‌بندی، برآورد چگالی و تشخیص ناهنجاری.
- رایج‌ترین الگوریتم‌های یادگیری: رگرسیون خطی و چندجمله‌ای، رگرسیون لجستیک، k-Nearest Neighbers، ماشین‌های بردار پشتیبان، درختان تصمیم‌گیری، جنگل‌های تصادفی و روش‌های Ensemble.

بخش دوم، شبکه‌های عصبی و یادگیری عمیق، موضوعات زیر را پوشش می‌دهد:

- شبکه‌های عصبی چیست؟ به چه دردی می‌خورند؟
- ایجاد و آموزش شبکه‌های عصبی با استفاده از TensorFlow و Keras
- مهم‌ترین معماری‌های شبکه عصبی: شبکه‌های عصبی پیش‌خور، شبکه‌های پیچشی، شبکه‌های بازگشتی، شبکه‌های حافظه طولانی کوتاه – مدت (LSTM)، خود رمزگذار و شبکه‌های مولد تخاصمی (GAN)
- تکنیک‌های آموزش شبکه‌های عصبی عمیق.
- مقیاس بندی شبکه‌های عصبی برای مجموعه داده‌های بزرگ.
- استراتژی‌های یادگیری با یادگیری تقویتی.
- کنترل عدم قطعیت با یادگیری عمیق بیزین.

قسمت اول بیشتر بر اساس Scikit-Learn و در قسمت دوم از TensorFlow و Keras استفاده‌شده است.

درحالی‌که یادگیری عمیق بدون شک یکی از مهیج‌ترین زمینه‌های یادگیری ماشین است، ابتدا باید بر اصول اولیه تسلط داشته باشید. علاوه بر این اکثر مسائل را می‌توان به‌خوبی با استفاده از تکنیک‌های ساده‌تر مانند جنگل‌های تصادفی و روش‌های گروهی (در بخش اول موردبحث قرار گرفت) حل کرد. یادگیری عمیق برای مسائل پیچیده‌ای مانند تشخیص تصویر، تشخیص گفتار یا پردازش زبان طبیعی مناسب است. (به‌شرط داشتن اطلاعات کافی ، قدرت محاسبه و صبر)

سایر منابع

وب‌سایت‌های جالبی در مورد یادگیری ماشین وجود دارد که راهنمای کاربر .Scikit-Learn را دارند. همچنین ممکن است از Dataquest، که آموزش‌های تعاملی بسیار خوبی ارائه می‌دهد و وبلاگ‌های ML مانند

موارد ذکرشده در Quora لذت ببرید. درنهایت وب‌سایت Deep Learning فهرست خوبی از منابع برای کسب اطلاعات بیشتر دارد.

البته کتاب‌های مقدماتی دیگری نیز در مورد یادگیری ماشین وجود دارد، به‌ویژه:

- « Data Science from Scratch » نوشته Joel Grus که اصول اساسی یادگیری ماشین را ارائه می‌دهد و برخی از الگوریتم‌های اصلی را در پایتون پیاده‌سازی می‌کند.
- «Machine Learning: An Algorithmic Perspective » نوشته استفان مارسلند، دیدگاه الگوریتمی (چاپمن و هال). این کتاب یک مقدمه عالی برای یادگیری ماشین است که طیف وسیعی از موضوعات را به‌طور عمیق با مثال‌های کد در پایتون (همچنین از ابتدا، اما با استفاده از NumPy پوشش می‌دهد.
- «Python Machine Learning » نوشته سباستین راشکا یک مقدمه عالی برای یادگیری ماشین است و از کتابخانه‌های منبع باز پایتون۲ Pylearn و Theanoاستفاده می‌کند.
- « *Deep Learning with Python* » نوشته فرانسوا شولت یک کتاب بسیار کاربردی که طیف وسیعی از موضوعات را به‌روشنی و مختصر پوشش می‌دهد؛ همان‌طور که از نویسنده کتابخانه عالی Keras انتظار دارید. این نمونه‌های کد را بر نظریه ریاضی ترجیح می‌دهد.
- « *Learning fromData*» نوشته Yaser S. Abu-Mostafa, Malik Magdon-Ismail, وHsuan-Tien Lin رویکرد نسبتاً تئوریک به ML است، بینش عمیقی را ارائه می‌دهد به‌ویژه در مورد معامله بایاس/ واریانس (به فصل ۴ مراجعه کنید).
- «*Artificial Intelligence: A Modern Approach, ۳rd Edition*» نوشته Stuart Russell و Peter Norvig کتابی عالی است که تعداد زیادی از موضوعات ازجمله یادگیری ماشین را پوشش می‌دهد. به چشم‌انداز ML کمک می‌کند.

سرانجام یک راه عالی برای یادگیری این است که به وب‌سایت‌های رقابت ML مانند Kaggle.com بپیوندید، این به شما امکان می‌دهد با کمک و بینش برخی از بهترین متخصصان ML در خارج، مهارت‌های خود را در مورد مسائل دنیای واقعی تمرین کنید.

نمونه‌های کد

مطالب تکمیلی (نمونه کد، تمرینات و غیره) برای بارگیری در آدرس زیر https://github.com/ageron/handson-ml۲ موجود و بیش‌تر از نوت بوک‌های Jupyter تشکیل‌شده است.

برخی از مثال‌های کد موجود در کتاب برخی از بخش‌های تکراری یا جزئیاتی را که آشکار یا غیر مرتبط با یادگیری ماشین است، کنار می‌گذارند. این امر تمرکز بر قسمت‌های مهم کد را حفظ می‌کند و باعث صرفه‌جویی در فضا برای پوشش موضوعات بیشتر می‌شود. بااین‌حال، اگر می‌خواهید نمونه کد کامل همه آن‌ها در نوت بوک‌های Jupyter موجود است.

توجه داشته باشید که وقتی نمونه‌های کد برخی خروجی‌ها را نشان می‌دهند، این نمونه کدها با دستورات پایتون (>>> و ...) ، مانند پوسته پایتون، نشان داده می‌شود تا کد را از خروجی‌ها به‌وضوح تشخیص دهد. به‌عنوان‌مثال، این کد تابع ()square را تعریف می‌کند، سپس مربع ۳ را محاسبه و نمایش می‌دهد:

```
>>> def square(x):
...     return x ** ۲
...
>>> result = square(۳)
>>>
result۹
```

وقتی کد چیزی را نمایش نمی‌دهد، از دستورات استفاده نمی‌شود. بااین‌حال گاهی اوقات ممکن است نتیجه به‌صورت یک نظر مانند این نشان داده شود:

```
def square(x):
    return x ** ۲

result = square(۳)  # result is ۹
```

استفاده از نمونه‌های کد

این کتاب به شما کمک می‌کند تا کار خود را انجام دهید. به‌طورکلی ، می‌توانید از کدهای ارائه‌شده در این کتاب می‌توان در برنامه‌ها و اسناد خود استفاده کنید. گنجاندن مقدار قابل‌توجهی کد نمونه از این کتاب در اسناد محصول شما نیاز به مجوز دارد.

نحوه تماس با ما

ما یک صفحه وب برای این کتاب داریم که در آن خطاها، نمونه‌ها و موارد دیگر را ذکر می‌کنیم اطلاعات می‌توانید به این صفحه در آدرس http://bit.ly/hands-on-machine-learning- with-scikit-learn-and- tensorflow یا https://homl.info/oreilly دسترسی پیدا کنید.

برای نظر دادن یا پرسیدن سؤالات فنی در مورد این کتاب ایمیل به bookques- tions@oreilly.com ایمیل بزنید.

برای اطلاعات بیشتر در مورد کتاب‌ها، دوره‌ها، کنفرانس‌ها و اخبار، به وب‌سایت http://www.oreilly.com مراجعه کنید.

ما را در فیس بوک پیدا کنید: http://facebook.com/oreilly

ما را در توییتر دنبال کنید: http://twitter.com/oreillymedia

ما را در YouTube تماشا کنید: http://www.youtube.com/oreillymedia

تقدیر و تشکر

پیام‌های زیادی از خوانندگان دریافت کردم بسیاری از آن‌ها سوا لاتی مطرح کردند، برخی با مهربانی به اشتباهات اشاره کردند و بیشتر کلمات دلگرم‌کننده‌ای برای من ارسال کردند. از همه این خوانندگان به خاطر حمایت فوق‌العاده‌شان قدردانی می‌کنم. لطفاً در صورت مشاهده اشکال در مثال‌های کد یا در صورت مشاهده

خطا در متن، از ارسال نظرات خود دریغ نکنید. برخی از خوانندگان همچنین در مورد نحوه کمک این کتاب به آن‌ها برای یافتن اولین شغل خود یا نحوه کمک به آن‌ها برای حل یک مشکل مشخص که در آن کار می‌کردند توضیح دادند؛ من چنین بازخوردی را فوق‌العاده انگیزه‌بخش می‌دانم. اگر این کتاب را مفید می‌دانید خوشحال می‌شوم نظرات خود را با من به اشتراک بگذارید.

من همچنین از همه افراد تشکر می‌کنم که کتاب من را با این دقت مرور کردند. به‌طور خاص، از François Chollet تشکر کنم که همه فصل‌ها را بر اساس Keras & TensorFlow مرور کرده و بازخورد عمیق و مفیدی به من داد. ازآنجاکه Keras یکی از موارد اصلی اضافه‌شده به این نسخه است، بررسی نویسنده آن کتاب ارزشمند بود. خواندن کتاب Deep Learning with Python را توصیه می‌کنم: [۲]این کتاب مختصر ، وضوح و عمق خود کتابخانه Keras را دارد. همچنین از Ankur Patel تشکر می‌کنم، که هر فصل از این نسخه دوم را مرور کرد و بازخورد خوبی به من داد.

با تشکر فراوان از Haesun Park که هنگام نوشتن ترجمه کره‌ای چاپ اول این کتاب بازخورد بسیار خوبی به من داد. او همچنین نوت بوک‌های Jupyter را به کره‌ای ترجمه کرد.

با تشکر فراوان از کارکنان O'Reilly به‌ویژه نیکول تاچ، که بازخورد روشنی به من دادند، همیشه شاد، دلگرم‌کننده و مفید بوده است. از مایکل کرونین نیز تشکر می‌کنم که در شروع این ویرایش دوم بسیار صبور بود. با تشکر از ماری بوگورو، بن لوریکا، مایک لوکیدز و لورل روما که به این پروژه اعتقاد داشتند و در تعریف دامنه آن به من کمک کردند. از مت هکر و همه تیم اطلس برای پاسخگویی به همه سؤالات فنی من در مورد قالب‌بندی، asciidoc و LaTeX تشکر می‌کنم و از راشل موناغن، نیک آدامز و همه تیم سازنده برای بررسی نهایی و صدها تصحیح آن‌ها تشکر می‌کنم.

همچنین می‌خواهم از همکاران سابقم در Google به‌ویژه تیم طبقه‌بندی ویدیوهای YouTube تشکر کنم که به من در مورد یادگیری ماشین بسیار آموزش دادند.

[۲] "Deep Learning with Python," François Chollet (2017).

فصل ۱ چشم‌انداز یادگیری ماشین

وقتی بیش‌تر مردم کلمه **یادگیری ماشین** (ML) را می‌شنوند؛ یک ربات یا قهرمان را به خاطر می‌آورند. اما یادگیری ماشین به آینده‌ای فانتزی مربوط نیست. در حقیقت، دهه‌هاست که در برخی از برنامه‌های تخصصی مانند **نویسه‌خوان نوری**[1] وجود دارد. اما اولین برنامه یادگیری ماشین که جریان اصلی پیدا کرد و زندگی صدها میلیون نفر را بهبود بخشید و در دهه ۱۹۹۰ جهان را تسخیر کرد فیلتر اسپم بود. به دنبال آن صدها برنامه یادگیری ماشین وجود داشت که اکنون بی‌سروصدا به محصولات و ویژگی‌هایی که به‌طور منظم از آن‌ها استفاده می‌کنید (مانند توصیه‌های بهتر جست‌وجوی صوتی) اضافه شد.

یادگیری ماشین از کجا شروع می‌شود و به کجا ختم می‌شود؟ و یادگیری برای ماشین به چه معنی است؟ اگر نسخه‌ای از ویکی‌پدیا را بارگیری کنم، آیا کامپیوتر من واقعاً چیزی آموخته است؟ آیا ناگهان هوشمندتر می‌شود؟ در این فصل یادگیری ماشین را برای شما روشن‌تر خواهیم کرد.

قبل از این‌که به بررسی یادگیری ماشین بپردازیم، مناطق اصلی و برجسته آن را مرور می‌کنیم: یادگیری باناظر در مقابل بدون ناظر، یادگیری برخط در مقابل دسته‌ای، یادگیری مبتنی بر نمونه در مقابل مبتنی بر مدل. سپس به گردش کار یک پروژه معمولی یادگیری ماشین نگاهی خواهیم انداخت، در مورد چالش‌های اصلی که ممکن است با آن روبرو شوید بحث کرده و نحوه ارزیابی و تنظیم دقیق یک سیستم یادگیری ماشین را بیان خواهیم کرد.

در این فصل بسیاری از مفاهیم اساسی (و اصطلاحات اصولی) معرفی‌شده است که هر دانشمند داده باید آن‌ها را بداند. این نمای کلی از سطح بالا خواهد بود .

۱–۱. یادگیری ماشین چیست؟

یادگیری ماشین علم (و هنر) برنامه‌نویسی کامپیوتر است تا بتواند از داده‌ها بیاموزد.

در اینجا یک تعریف کلی‌تری وجود دارد:

[یادگیری ماشین] رشته تحصیلی است که به کامپیوترها توانایی یادگیری را بدون برنامه‌ریزی صریح می‌دهد.

– آرتور ساموئل ۱۹۵۹

تعریف مهندسی گرا:

به برنامه رایانه‌ای گفته‌شده که از تجربه E با توجه به برخی از وظایف T و برخی معیارهای عملکرد p، اگر عملکرد آن روی T، همان‌طور که توسط P اندازه‌گیری می‌شود، با تجربه E بهبود می‌یابد.

گفته می‌شود که یک برنامه رایانه‌ای با توجه به برخی از وظایف T و برخی از معیارهای عملکرد P؛ از تجربه E می‌آموزد اگر عملکرد آن روی T، همان‌طور که توسط P اندازه‌گیری شده، با تجربه E بهبود یابد.

– تام میچل، ۱۹۹۷

[1] Optical Character Recognition (OCR)

به‌عنوان‌مثال، فیلتر اسپم شما یک برنامه یادگیری ماشین است که می‌تواند با پرچم گذاری اسپم و نمونه‌هایی از ایمیل‌های معمولی (غیر اسپم که به آن‌ها ham نیز گفته می‌شود) یاد بگیرد. نمونه‌هایی که سیستم برای یادگیری آن‌ها استفاده می‌کند؛ مجموعه آموزشی نامیده می‌شود. به هر مثال آموزش یک نمونه آموزش (یا نمونه) گفته می‌شود. در این حالت، وظیفه T پرچم گذاری اسپم یا هرزنامه‌ها برای ایمیل‌های جدید است، تجربه E داده‌های آموزش است و اندازه‌گیری عملکرد P باید تعریف شود. به‌عنوان‌مثال، می‌توانید از نسبت ایمیل‌های طبقه‌بندی‌شده صحیح استفاده کنید. این معیار عملکرد خاص دقت نامیده می‌شود و اغلب در کارهای طبقه‌بندی مورداستفاده قرار می‌گیرد. اگر یک نسخه از ویکی‌پدیا را بارگیری کنید، رایانه شما داده‌های در اختیار خواهد داشت اما به‌طور ناگهانی در یک کار بهتر نخواهد شد بنابراین اینجا یادگیری ماشین نخواهیم داشت.

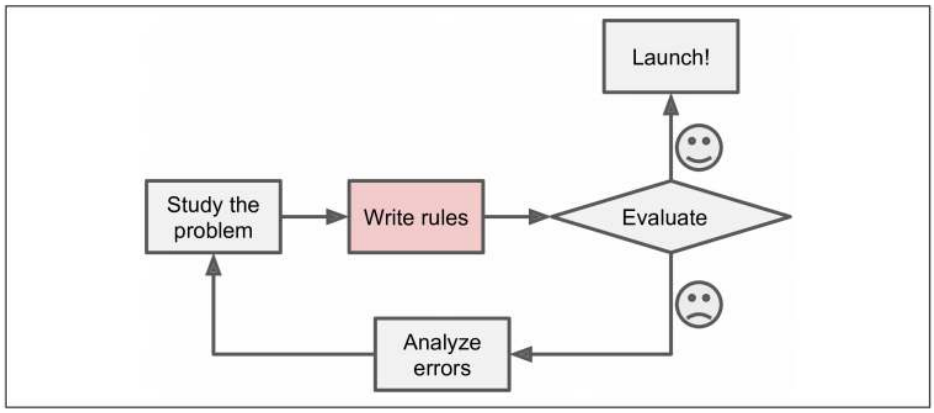
۲-۱. چرا باید از یادگیری ماشینی استفاده کرد؟

در نظر بگیرید که چگونه می‌توانید فیلتر اسپم را با استفاده از تکنیک‌های سنتی برنامه‌نویسی بنویسید (شکل ۱-۱):

۱. ابتدا ببینیم هرزنامه چگونه است. ممکن است متوجه شوید که برخی از کلمات یا عبارات (4U، "کارت اعتباری"، "رایگان" و "شگفت‌انگیز") در این موضوع زیاد مطرح می‌شوند. همچنین ممکن است چند الگو دیگر در نام فرستنده، متن ایمیل و غیره مشاهده کنید.

۲. برای هر یک از الگوهایی که یافتید، یک الگوریتم تشخیص می‌نویسید و اگر تعدادی از این الگوها شناسایی شود، برنامه شما ایمیل‌ها را به‌عنوان اسپم پرچم گذاری می‌کند.

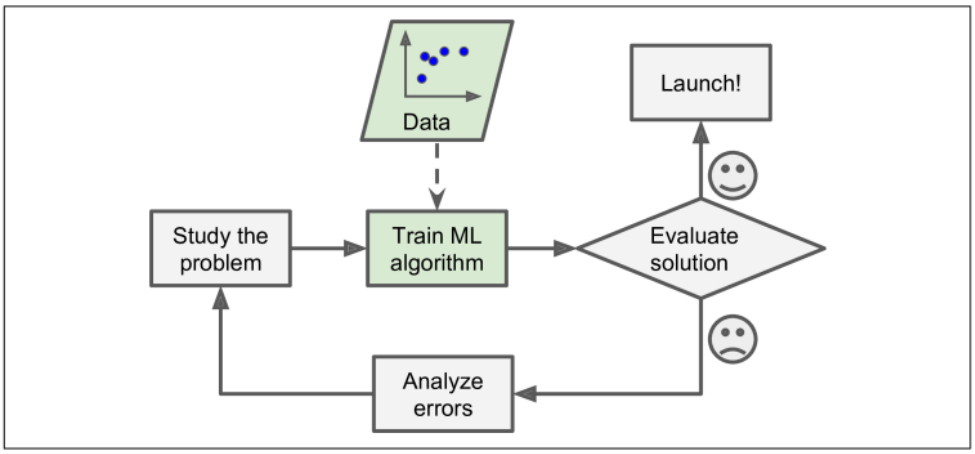
۳. برنامه خود را تست و مراحل ۱ و ۲ را تکرار کنید تا به‌اندازه کافی خوب شود.



شکل ۱-۱. رویکرد سنتی.

ازآنجاکه مساله بدیهی نیست؛ برنامه شما به لیست طولانی از قوانین پیچیده تبدیل می‌شود که حفظ آن بسیار دشوار است.

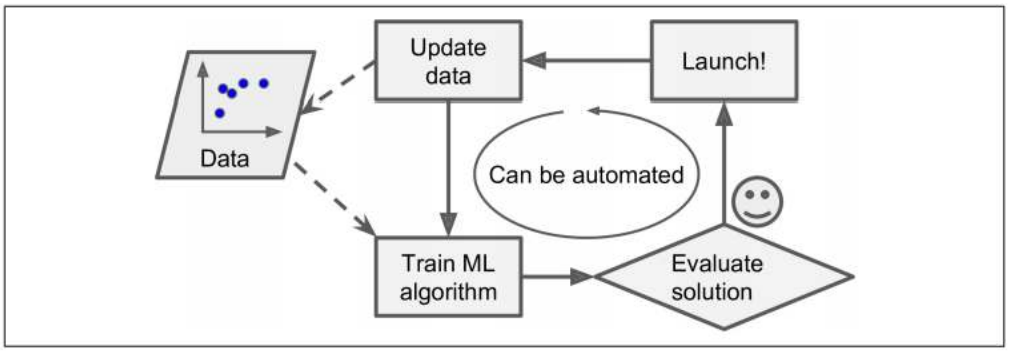
در مقابل، یک فیلتر هرزنامه مبتنی بر تکنیک‌های یادگیری ماشین با شناسایی الگوهای مکرر غیرمعمول کلمات در نمونه‌های هرزنامه در مقایسه با نمونه‌های غیر اسپم، به‌طور خودکار می‌آموزد کدام کلمات و عبارات پیش‌بینی کننده خوبی برای هرزنامه هستند (شکل ۲-۱). برنامه بسیار کوتاه‌تر، نگهداری آسان‌تر و به‌احتمال‌زیاد دقیق‌تر است.



شکل ۲-۱. رویکرد یادگیری ماشین.

علاوه بر این، اگر هرزنامه نویسان متوجه شوند که همه ایمیل‌های آن‌ها حاوی «4U» مسدود شده است از « for U»استفاده خواهند کرد. برای پرچم گذاری ایمیل‌های حاوی این نوع از نوشتن For U، یک فیلتر هرزنامه با استفاده از تکنیک‌های سنتی برنامه‌نویسی باید به‌روز شود. اگر اسپمرها همچنان در اطراف فیلتر هرزنامه در حال تلاش هستند، باید به نوشتن قوانین جدید برای همیشه ادامه دهید.

در مقابل یک فیلتر هرزنامه مبتنی بر تکنیک‌های یادگیری ماشین به‌طور خودکار متوجه می‌شود که این عبارت توسط کاربر پرچم اسپم گذاری شده است و به‌طور غیرمعمولی تکرار می‌شود بنابراین بدون دخالت شما شروع به پرچم گذاری آن‌ها می‌کند (شکل ۳-۱).

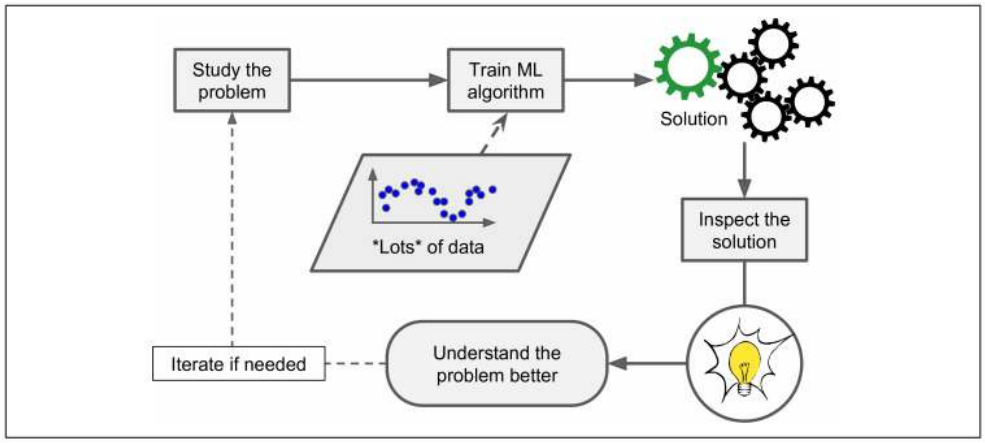


شکل ۳-۱. به‌طور خودکار با تغییر سازگار می‌شود.

حوزه دیگری که یادگیری ماشین در آن می‌درخشد، مسائلی است که یا برای رویکردهای سنتی بسیار پیچیده هستند یا الگوریتم شناخته‌شده‌ای ندارند. به‌عنوان‌مثال، تشخیص گفتار را در نظر بگیرید: برنامه‌ای بنویسید که توانایی تشخیص کلمات one و two را داشته باشد. ممکن است متوجه شوید که کلمه two با صدای بلند T شروع می‌شود، بنابراین می‌توانید یک الگوریتم را برای اندازه‌گیری شدت صدا کد نویسی کنید و از آن برای تشخیص one و two استفاده کنید. بدیهی است که این روش برای هزاران کلمه‌ای که میلیون‌ها انسان بسیار متفاوت در محیط‌های پرسروصدا و ده‌ها زبان بیان می‌کنند مناسب نخواهد بود. بهترین راه‌حل نوشتن الگوریتمی است که به‌تنهایی و با ضبط مثال‌های زیاد برای هر کلمه یاد بگیرد.

سرانجام، یادگیری ماشین می‌تواند به انسان کمک کند تا یاد بگیرد (شکل ۴-۱): الگوریتم‌های ML را می‌توان بررسی کرد تا ببینند چه چیزهایی آموخته‌اند (اگرچه برای برخی از الگوریتم‌ها این ممکن است مشکل باشد). به‌عنوان‌مثال، هنگامی‌که فیلتر اسپم در مورد هرزنامه کافی آموزش داده شد، می‌توان به‌راحتی آن را بررسی کند تا لیستی از کلمات و ترکیب کلماتی را که معتقد است بهترین پیش‌بینی کننده هرزنامه هستند را نشان دهد.

گاهی اوقات همبستگی‌های نامشخص یا روندهای جدید را نشان داده و درنتیجه منجر به درک بهتر مسئله می‌شود. استفاده از تکنیک‌های ML برای جست‌وجوی مقدار زیادی داده می‌تواند به کشف الگوهایی کمک کند که سریعاً آشکار نیستند که به این داده‌کاوی می‌گویند.



شکل ۴-۱. یادگیری ماشینی می‌تواند به انسان در یادگیری کمک کند.

به‌طور خلاصه، یادگیری ماشین برای موارد زیر مناسب است:

- مسائلی که راه‌حل‌های موجود برای آن‌ها نیاز به تنظیم دستی یا لیست طولانی قوانین زیادی دارد: یک الگوریتم یادگیری ماشین اغلب می‌تواند کد را ساده کند و عملکرد بهتری داشته باشد.
- مسائل پیچیده‌ای که با استفاده از یک رویکرد سنتی هیچ راه‌حل خوبی برای آن‌ها وجود ندارد: بهترین روش‌های یادگیری ماشین می‌توانند راه‌حلی پیدا کنند.
- نوسان محیط: سیستم یادگیری ماشین می‌تواند با داده‌های جدید سازگار شود.
- دریافت بینش درباره مسائل پیچیده و مقدار زیاد داده.

۳-۱. انواع سیستم‌های یادگیری ماشین

انواع مختلفی از سیستم‌های یادگیری ماشین وجود دارد که طبقه‌بندی آن‌ها در دسته‌های گسترده بر اساس:

- این‌که آیا آن‌ها با نظارت انسانی (یادگیری باناظر، بدون ناظر، نیمه نظارت‌شده و تقویتی) آموزش‌دیده‌اند یا نه
- این‌که آیا آن‌ها می‌توانند به‌صورت تدریجی در حین کار یاد بگیرند یا نه (برخط در مقابل یادگیری دسته‌ای)
- این‌که آیا آن‌ها با مقایسه ساده نقاط داده جدید با نقاط داده شناخته‌شده عمل می‌کنند؛ یا به‌جای آن الگوهایی را در داده‌های آموزش شناسایی و یک مدل پیش‌بینی می‌سازند، دقیقاً مانند دانشمندان (یادگیری مبتنی بر نمونه در مقابل مدل)

این معیارها منحصربه‌فرد نیستند. شما می‌توانید آن‌ها را به هر روشی که دوست دارید ترکیب کنید. به‌عنوان‌مثال، یک فیلتر پیشرفته اسپم ممکن است با استفاده از یک مدل شبکه عصبی عمیق آموزش‌دیده با استفاده از نمونه‌هایی

از هرزنامه و غیر اسپم، در حین کار بیاموزد. این امر آن را به یک سیستم یادگیری برخط یا آنلاین و مبتنی بر مدل تبدیل می‌کند.

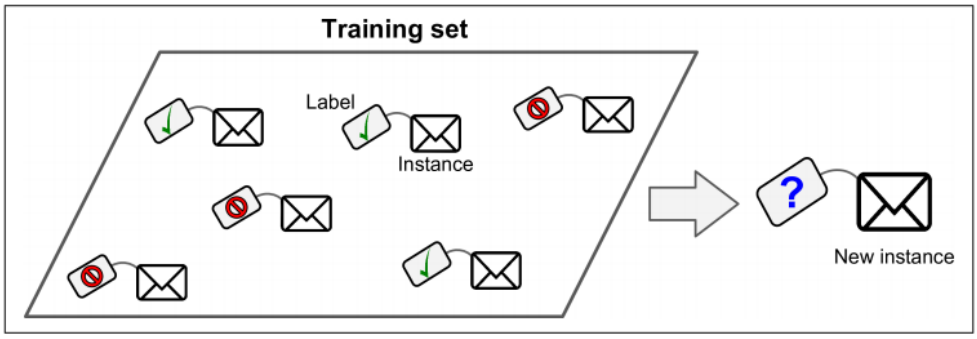
حال هرکدام از این معیارها را کمی دقیق‌تر بررسی می‌کنیم.

۱–۳–۱. یادگیری با ناظر / بدون ناظر

سیستم‌های یادگیری ماشین را می‌توان با توجه به میزان و نوع نظارتی که در حین آموزش دریافت می‌کنند، طبقه‌بندی کرد. چهار دسته عمده وجود دارد: یادگیری باناظر، بدون ناظر، نیمه نظارت‌شده و تقویتی.

یادگیری باناظر یا تحت نظارت

در یادگیری تحت نظارت داده‌های آموزشی که به الگوریتم می‌دهید شامل جواب‌های موردنظر به نام برچسب (شکل ۵–۱) است.



شکل ۵–۱. مجموعه آموزشی دارای برچسب برای یادگیری تحت نظارت (به‌عنوان‌مثال، طبقه‌بندی هرزنامه).

یک کار معمول یادگیری تحت نظارت، طبقه‌بندی است. فیلتر هرزنامه مثال خوبی برای این مورد است: با بسیاری از نامه‌های الکترونیکی همراه با کلاس آن‌ها (هرزنامه یا غیر آن) آموزش داده می‌شود و باید نحوه دسته‌بندی ایمیل‌های جدید را بیاموزد.

یک کار معمول دیگر پیش‌بینی مقدار عددی هدف مانند قیمت یک ماشین با توجه به مجموعه‌ای از ویژگی‌ها (مسافت پیموده شده، سن، مارک و غیره) به نام پیش‌بینی کننده‌ها است. به این نوع مسائل رگرسیون گفته می‌شود (شکل ۶–۱)

برای آموزش سیستم، باید مثال‌های زیادی از اتومبیل‌ها را به آن بدهید: ازجمله پیش‌بینی کننده‌ها و برچسب‌های آن‌ها (یعنی قیمت آن‌ها).

شکل ۶-۱. رگرسیون.

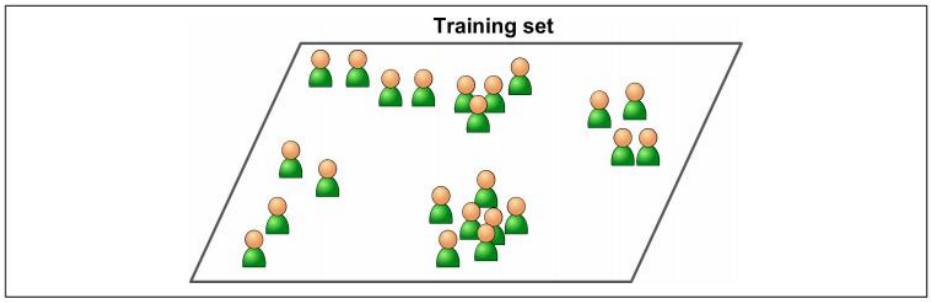
توجه داشته باشید که برخی از الگوریتم‌های رگرسیون را می‌توان برای طبقه‌بندی نیز استفاده کرد و بالعکس. به‌عنوان‌مثال، از رگرسیون لجستیک معمولاً برای طبقه‌بندی استفاده می‌شود، زیرا می‌تواند مقداری را متناسب با احتمال تعلق به یک کلاس معین (مثلاً ۲۰درصد احتمال هرزنامه بودن) تولید کند.

در اینجا برخی از مهم‌ترین الگوریتم‌های یادگیری تحت نظارت (که در این کتاب آورده شده است) ذکر می‌کنیم:

- k نزدیک‌ترین همسایه
- رگرسیون خطی
- رگرسیون لجستیک
- ماشین بردار پشتیبان (SVM)
- درختان تصمیم‌گیری و جنگل‌های تصادفی
- شبکه‌های عصبی

یادگیری بدون نظارت

در یادگیری بدون نظارت داده‌های آموزش بدون برچسب هستند (شکل ۷-۱). این سیستم سعی می‌کند بدون معلم یاد بگیرد.



شکل ۷-۱. یک مجموعه آموزشی بدون برچسب برای یادگیری بدون نظارت.

برخی از مهم‌ترین الگوریتم‌های یادگیری بدون نظارت به‌صورت زیر هستند (در فصل ۸ و ۹ آورده شده است):

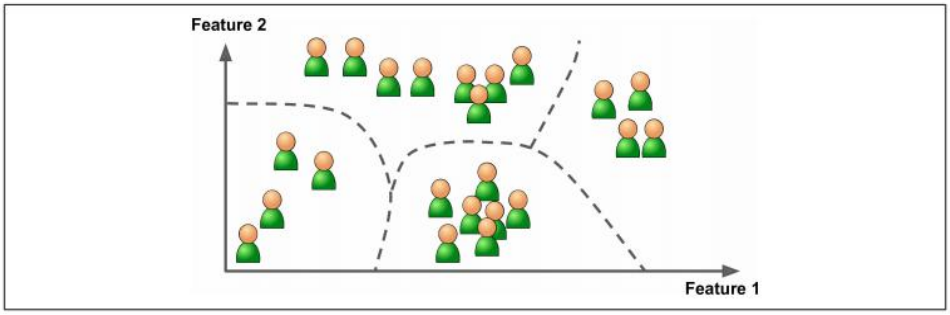
- خوشه‌بندی
  1. میانگین-k
  2. DBSCAN
- تجزیه‌وتحلیل خوشه‌بندی سلسله مراتبی (HCA)
- تشخیص ناهنجاری و تازگی
  1. SVM یک -کلاسه
  2. جنگل ایزوله
- مصورسازی یا کاهش ابعاد
  1. تحلیل مؤلفه‌های اصلی (PCA)
  2. Kernel PCA
  3. Locally-Linear Embedding
  4. t-distributed Stochastic Neighbor Embedding (t-SNE)
- یادگیری **قانون وابستگی**[۲]
  1. **آپریوری**[۳]
  2. Eclat

به‌عنوان‌مثال شما اطلاعات زیادی در مورد بازدیدکنندگان وبلاگ خود دارید. ممکن است بخواهید با استفاده از یک الگوریتم خوشه‌بندی گروه‌هایی از بازدیدکنندگان مشابه را شناسایی کنید (شکل ۸-۱). هیچ‌وقت به الگوریتم نمی‌گویید که یک بازدیدکننده از کدام گروه است: او این ارتباطات را بدون کمک شما پیدا می‌کند. به‌عنوان‌مثال، ممکن است متوجه شود که ۴۰درصد بازدیدکنندگان شما مرد و عاشق کتاب‌های طنز هستند و عصرها از وبلاگ

---

[۲] association rule learning
[۳] Apriori

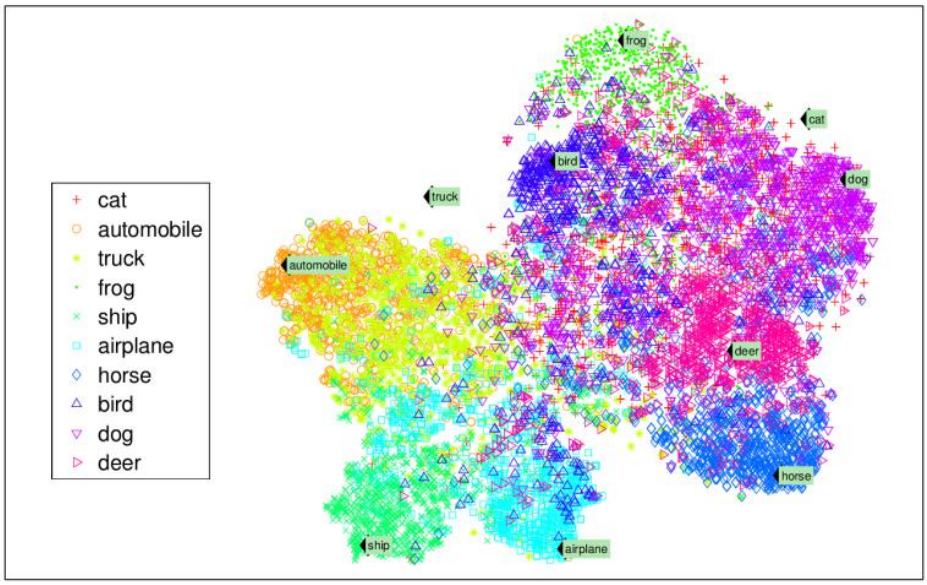
بازدید می‌کنند؛ درحالی‌که ۲۰درصد جوانان علاقه‌مند به موارد علمی-تخیلی هستند که در آخر هفته‌ها بازدید می‌کنند. اگر از الگوریتم خوشه‌بندی سلسله مراتبی استفاده می‌کنید، ممکن است هر گروه را به گروه‌های کوچک‌تر تقسیم کند. این شما کمک می‌کند تا پست‌های خود را برای هر گروه هدف قرار دهید.



شکل ۸-۱. خوشه‌بندی.

الگوریتم‌های مصورسازی[۴] نیز نمونه‌های خوبی از یادگیری بدون نظارت هستند: شما داده‌های پیچیده و بدون برچسب زیادی را به آن‌ها می‌دهید و آن‌ها نمایشی ۲ یا ۳ بعدی از داده‌های شما را ترسیم می‌کنند (شکل ۹-۱). این الگوریتم‌ها سعی می‌کنند ساختار را حفظ کنند. به‌عنوان‌مثال تلاش می‌کنند خوشه‌های جداگانه در فضای ورودی در تصویر تجزیه نشوند)، بنابراین می‌توانید نحوه سازمان‌دهی داده‌ها را بفهمید و شاید الگوهای نامشخص را شناسایی کنید.

[۴] Visualization algorithms



شکل ۹-۱. نمونه‌ای از مصورسازی t-SNE که خوشه‌های معنایی را برجسته می‌کند.

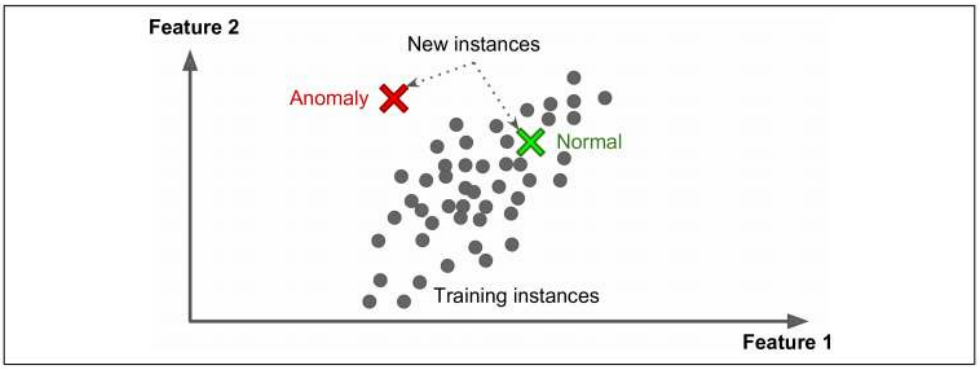
در کاهش ابعاد هدف ساده‌سازی داده‌ها بدون از دست دادن اطلاعات بیش‌ازحد است. یکی از راه‌های انجام این کار ادغام چندین ویژگی همبسته در یک ویژگی است. به‌عنوان‌مثال، مسافت پیموده شده اتومبیل ممکن است با سن آن بسیار مرتبط باشد، بنابراین الگوریتم کاهش ابعاد آن‌ها را در یک ویژگی ادغام می‌کند که نمایانگر فرسودگی ماشین است. به این عمل **استخراج ویژگی**[۵] گفته می‌شود.

یکی دیگر از کارهای مهم بدون نظارت، تشخیص ناهنجاری است.(به‌عنوان‌مثال، شناسایی تراکنش‌های غیرمعمول کارت اعتباری برای جلوگیری از تقلب قبل از استفاده از آن در الگوریتم یادگیری دیگر)

در هنگام آموزش به این سیستم بیش‌تر نمونه‌های نرمال را یاد می‌گیرد و آن‌ها را تشخیص می‌دهد پس با دیدن نمونه جدیدی تشخیص خواهد داد که نرمال یا احتمالاً ناهنجاری است (شکل ۱۰-۱ را ببینید). یک وظیفه بسیار مشابه، تشخیص تازگی است: تفاوت در این است که الگوریتم‌های تشخیص تازگی انتظار دارند که فقط داده‌های عادی را در حین آموزش ببینند، درحالی‌که الگوریتم‌های تشخیص ناهنجاری معمولاً مقاوم‌تر هستند، آن‌ها اغلب حتی با درصد کمی از داده‌های خارج از مجموعه آموزش، می‌توانند عملکرد خوبی داشته باشند.

---

[۵] feature extraction



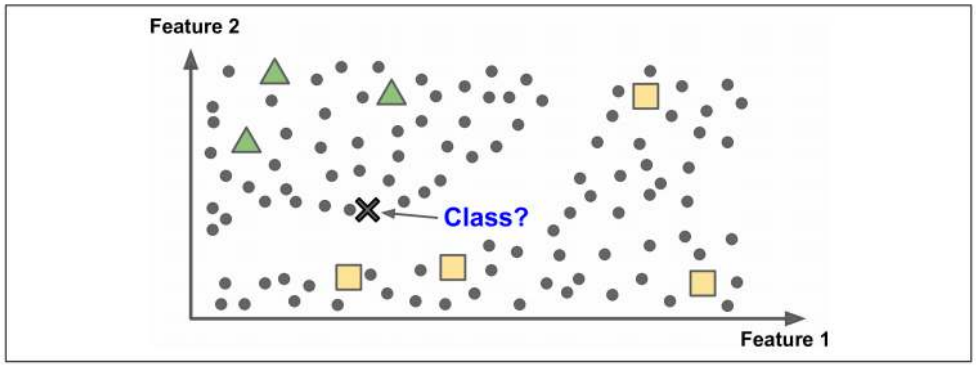
شکل ۱۰-۱. تشخیص ناهنجاری.

یکی دیگر از انواع یادگیری بدون نظارت، یادگیری قانون وابستگی ارتباط است که در آن هدف کاوش در مقدار زیادی داده و کشف روابط جالب بین ویژگی‌ها است. به‌عنوان‌مثال، تصور کنید که سوپرمارکت دارید. اجرای یک قانون وابستگی در سیاهه‌های مربوط به فروش ممکن است نشان دهد که افرادی که سس کچاپ و چیپس خریداری می‌کنند تمایل به خرید استیک نیز دارند. بنابراین، ممکن است این موارد را نزدیک به یکدیگر قرار دهید.

یادگیری نیمه نظارتی

برخی از الگوریتم‌ها می‌توانند با داده‌های آموزش دارای برچسب جزئی، (معمولاً بسیاری از داده‌های بدون برچسب و کمی داده‌های دارای برچسب)سروکار داشته باشند. این امر یادگیری نیمه نظارت‌شده (شکل ۱۱-۱) نامیده می‌شود.

برخی از خدمات اشتراک‌گذاری عکس، مانند Google Photos، نمونه‌های خوبی در این زمینه هستند. هنگامی‌که همه عکس‌های خانوادگی خود را در سرویس بارگذاری کردید، به‌طور خودکار تشخیص می‌دهد که همان شخص A در عکس‌های ۱، ۵ و ۱۱ نشان وجود دارد، درحالی‌که شخص B در عکس‌های ۲، ۵ و ۷ نشان داده‌شده است. این قسمت بدون نظارت الگوریتم (خوشه‌بندی) است. اکنون تنها یک برچسب به ازای هر شخص آن‌ها را در هر عکسی متمایز می‌کند که برای جست‌وجوی عکس‌ها مفید است.



شکل ۱۱-۱. یادگیری نیمه نظارتی.

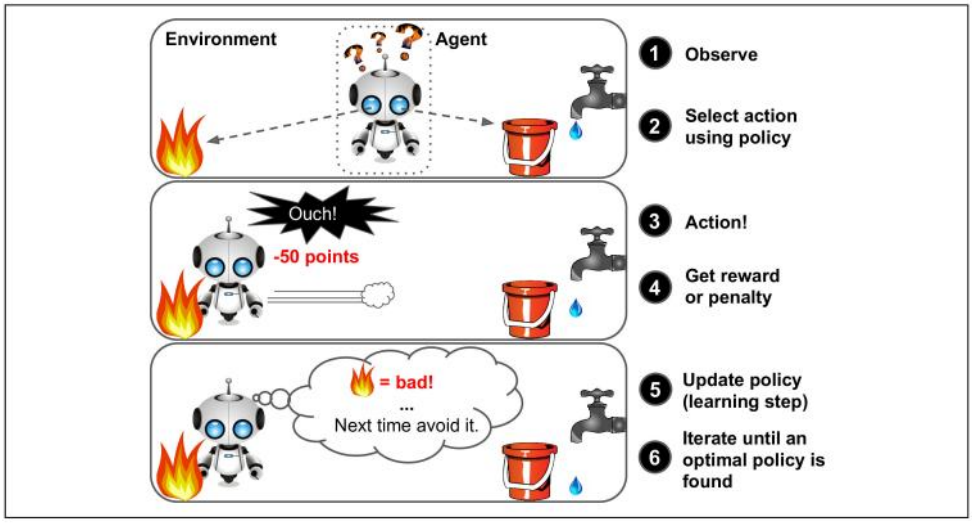
بیش‌تر الگوریتم‌های یادگیری نیمه نظارت ترکیبی از الگوریتم‌های بدون نظارت و نظارت‌شده هستند. به‌عنوان‌مثال، شبکه‌های **باور عمیق**[۶] بر اساس مؤلفه‌های بدون نظارت موسوم به **ماشین بولتزمن محدودشده**[۷] هستند که روی‌هم چیده شده‌اند. RBMها به ترتیب و بدون نظارت تحت آموزش قرار می‌گیرند و سپس کل سیستم با استفاده از تکنیک‌های یادگیری نظارت‌شده تنظیم می‌شود.

یادگیری تقویتی

سیستم یادگیری (که در اینجا عامل نامیده می‌شود) محیط را مشاهده و اقداماتی را انتخاب و انجام می‌دهد و در ازای آن پاداش یا مجازات‌هایی به‌صورت پاداش منفی، دریافت می‌کند(مانند شکل ۱۲-۱). سپس باید به‌خودی‌خود بیاموزد که بهترین استراتژی، به نام سیاست، برای دریافت بیش‌ترین پاداش در طول زمان چیست. خط‌مشی مشخص می‌کند که عامل در شرایطی مشخص چه اقدامی را باید انتخاب کند.

---

[۶] Deep belief network
[۷] restricted Boltzmann machines(RBM)



شکل ۱۲-۱. یادگیری تقویتی.

به‌عنوان‌مثال، بسیاری از ربات‌ها الگوریتم‌های یادگیری تقویتی برای یادگیری راه رفتن استفاده می‌کنند. برنامه AlphaGo DeepMind همچنین نمونه خوبی از آموزش تقویتی است که در ماه مه ۲۰۱۷ هنگامی‌که قهرمان جهان Ke Jie را در بازی Go شکست داد، خبرساز شد. این الگوریتم سیاست برنده شدن خود را با تجزیه‌وتحلیل میلیون‌ها بازی و سپس انجام بسیاری از بازی‌ها علیه خود یاد گرفت.

۲-۳-۱. یادگیری برخط و دسته‌ای[۸]

معیار دیگری که برای طبقه‌بندی سیستم‌های یادگیری ماشین مورداستفاده قرار می‌گیرد این است که آیا سیستم می‌تواند به‌تدریج از جریان داده‌های ورودی یاد بگیرد یا خیر؟.

یادگیری دسته‌ای

در یادگیری دسته‌ای، سیستم قادر به یادگیری تدریجی نیست: باید با استفاده از تمام داده‌های موجود آموزش داده شود. این کار معمولاً به زمان و محاسبات زیادی نیاز دارد، بنابراین به‌صورت آفلاین انجام می‌شود. ابتدا سیستم آموزش داده می‌شود و سپس از آن بدون یادگیری دیگری استفاده می‌شود به همین علت به آن یادگیری آفلاین می‌گویند.

[۸] Batch and Online Learning

اگر می‌خواهید یک سیستم یادگیری دسته‌ای در مورد داده‌های جدید (مانند نوع جدید هرزنامه) اطلاعاتی کسب کند؛ باید نسخه جدید سیستم را از ابتدا روی مجموعه داده کامل (نه‌فقط داده‌های جدید، بلکه داده‌های قدیمی) نیز آموزش دهید. سپس سیستم قدیمی را متوقف کرده و آن را با سیستم جدید جایگزین کنید.

خوشبختانه، کل فرآیند آموزش، ارزیابی و راه‌اندازی سیستم یادگیری ماشین می‌تواند به‌راحتی اتوماتیک شود (همان‌طور که در شکل ۳-۱ نشان داده‌شده است)، بنابراین حتی یک سیستم یادگیری دسته‌ای می‌تواند با تغییر سازگار شود. به‌سادگی داده‌ها را به‌روز کنید و در هر زمان که لازم است نسخه جدید سیستم را از ابتدا آموزش دهید.

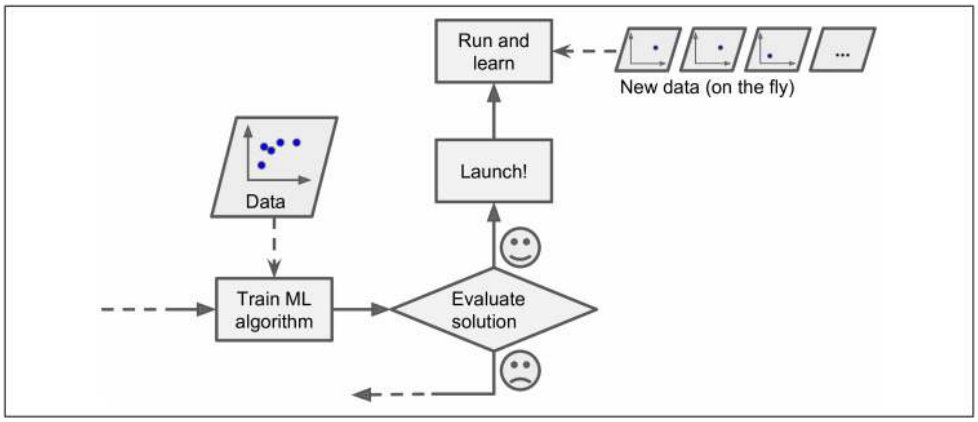
این راه‌حل ساده است و غالباً خوب عمل می‌کند؛ اما آموزش با استفاده از مجموعه کامل داده‌ها می‌تواند ساعت‌ها طول بکشد، بنابراین شما معمولاً فقط هر ۲۴ ساعت یا حتی هر هفته یک سیستم جدید آموزش می‌دهید. اگر سیستم شما نیاز به انطباق با داده‌هایی دارد که به‌سرعت در حال تغییر هستند (به‌عنوان‌مثال، برای پیش‌بینی قیمت سهام) به یک راه‌حل واکنشی‌تر نیاز دارید.

همچنین، آموزش در مورد مجموعه کامل داده‌ها به منابع محاسباتی زیادی احتیاج دارد (CPU، فضای حافظه، فضای دیسک، I / O دیسک، I / O شبکه و غیره). اگر داده‌های زیادی داشته باشید و هر روز آموزش از ابتدای سیستم خود را انجام دهید، درنهایت هزینه زیادی برای شما خواهد داشت. اگر مقدار داده‌ها زیاد باشد، حتی ممکن است استفاده از یک دسته غیرممکن باشد.

اگر سیستم شما نیاز به یادگیری مستقل داشته و منابع محدودی داشته باشد (به‌عنوان‌مثال، برنامه تلفن‌های هوشمند یا مریخ‌نورد در مریخ) باید داده‌های آموزشی زیادی را با خود حمل کنید و منابع زیادی را برای آموزش ساعت‌ها در هر زمان به خود اختصاص دهید. خوشبختانه گزینه بهتر در همه این موارد استفاده از الگوریتم‌هایی است که توانایی یادگیری تدریجی را دارند.

یادگیری برخط (آنلاین)

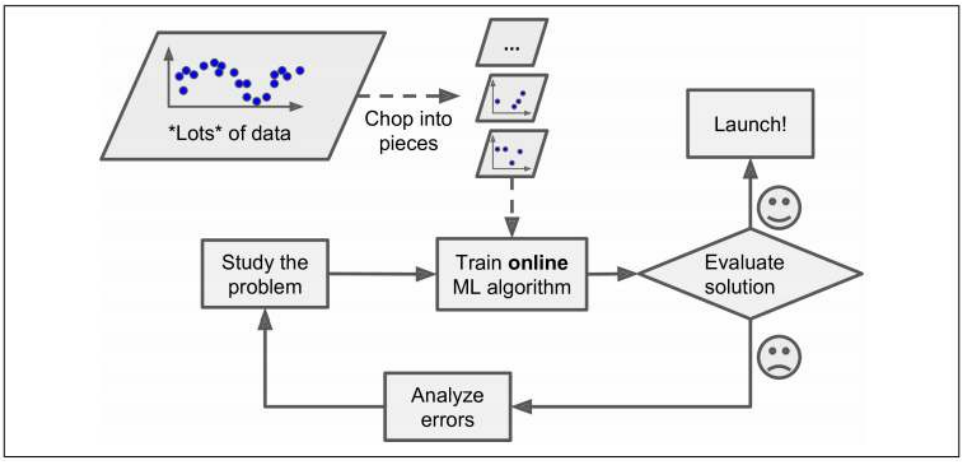
در یادگیری آنلاین سیستم را با تغذیه متوالی نمونه‌های داده، به‌صورت جداگانه یا توسط گروه‌های کوچکی به نام «mini-batches» به‌تدریج آموزش می‌دهند. هر مرحله یادگیری سریع و ارزان است، بنابراین سیستم می‌تواند هنگام رسیدن اطلاعات جدید، یاد بگیرد (شکل ۱۳-۱ را ببینید).



شکل ۱۳-۱. یادگیری آنلاین.

یادگیری آنلاین برای سیستم‌هایی که داده‌ها را به‌صورت جریان مداوم (به‌عنوان‌مثال قیمت سهام) دریافت و برای تغییر سریع یا خودمختار نیاز به سازگاری دارند، عالی است. اگر منابع محاسباتی محدودی داشته باشید نیز گزینه خوبی است: هنگامی‌که سیستم یادگیری آنلاین در مورد نمونه‌های داده جدید اطلاعاتی کسب کرد، دیگر نیازی به آن‌ها نیست؛ بنابراین می‌توانید آن‌ها را کنار بگذارید. این روش در فضای زیادی صرفه‌جویی می‌کند.

همچنین می‌توان از الگوریتم‌های یادگیری آنلاین برای آموزش سیستم‌ها در مجموعه داده‌های عظیمی استفاده کرد که نمی‌توانند در حافظه اصلی یک ماشین قرار بگیرند (این کار یادگیری خارج از هسته نامیده می‌شود). الگوریتم بخشی از داده‌ها را بارگیری می‌کند، یک مرحله آموزش روی آن داده‌ها اجرا و روند را تکرار می‌کند تا زمانی که روی همه داده‌ها اجرا شود (شکل ۱۴-۱ را ببینید).



شکل ۱۴-۱. استفاده از آموزش آنلاین برای مدیریت مجموعه کلان داده.

یکی از پارامترهای مهم سیستم‌های یادگیری آنلاین، سرعت سازگاری آن‌ها با تغییر داده‌هاست: به آن نرخ یادگیری می‌گویند. اگر نرخ یادگیری بالایی تعیین کنید، سیستم شما به‌سرعت با داده‌های جدید سازگار شده و همچنین به‌سرعت داده‌های قدیمی را فراموش می‌کند.

برعکس، اگر نرخ یادگیری پایینی تعیین کنید، اینرسی سیستم بیش‌تر خواهد شد. یعنی آهسته‌تر خواهد آموخت؛ اما همچنین نسبت به نویز در داده‌های جدید یا توالی نقاط داده‌های پرت حساسیت کم‌تری خواهد داشت.

یک چالش بزرگ در مورد یادگیری آنلاین این است که اگر داده‌های بد به سیستم وارد شوند، عملکرد سیستم به‌تدریج کاهش می‌یابد. برای کاهش این خطر، باید در صورت مشاهده افت عملکرد، سیستم خود را از نزدیک تحت نظر داشته باشید و بی‌درنگ یادگیری را قطع کنید (و احتمالاً به حالت قبلی بازگردانید). همچنین ممکن است بخواهید داده‌های ورودی را کنترل کرده و نسبت به داده‌های غیرعادی (مثلاً با استفاده از الگوریتم تشخیص ناهنجاری) واکنش نشان دهید.

۳-۳-۱. یادگیری مبتنی بر نمونه در مقابل مبتنی مدل

روش دیگر برای دسته‌بندی سیستم‌های یادگیری ماشین، نحوه تعمیم آن‌هاست.

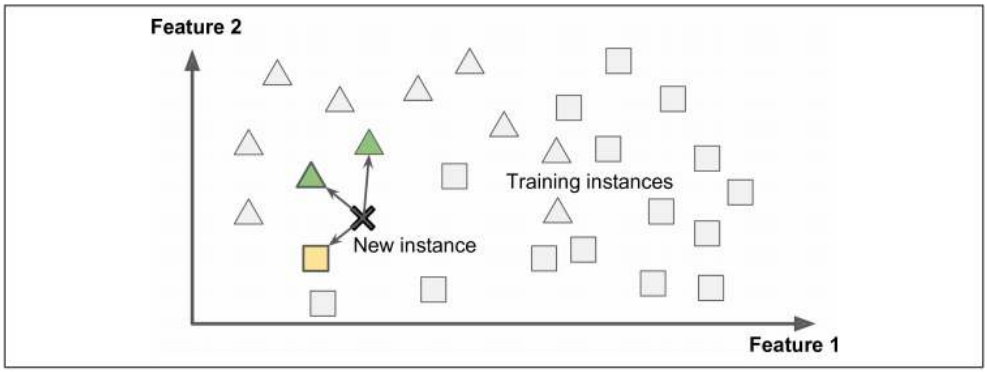
بیش‌تر کار یادگیری ماشین در مورد پیش‌بینی است. این بدان معنی است که با توجه به تعدادی مثال آموزشی، سیستم باید بتواند نمونه‌هایی را که قبلاً هرگز ندیده تعمیم دهد. داشتن معیار عملکرد مناسب روی داده‌های آموزش خوب است؛ اما کافی نیست. هدف واقعی این است که در موارد جدید عملکرد خوبی داشته باشیم. دو رویکرد اصلی یادگیری مبتنی بر نمونه و یادگیری مبتنی بر مدل برای تعمیم وجود دارد.

یادگیری مبتنی بر نمونه

احتمالاً پیش‌پاافتاده‌ترین شکل یادگیری صرفاً حفظ کردن است. اگر بخواهید از این طریق فیلتر اسپم ایجاد کنید، فقط ایمیل‌های یکسان با ایمیل‌هایی که قبلاً توسط کاربران پرچم گذاری شده؛ پرچم گذاری می‌کند.

فیلتر اسپم به‌جای فقط پرچم گذاری ایمیل‌هایی که با ایمیل‌های هرزنامه شناخته‌شده یکسان هستند، می‌تواند به‌گونه‌ای برنامه‌ریزی شود که ایمیل‌هایی را که بسیار شبیه نامه‌های اسپم شناخته‌شده هستند نیز پرچم گذاری کند. این مستلزم اندازه‌گیری شباهت بین دو ایمیل است. یک معیار شباهت بین دو ایمیل می‌تواند شمارش تعداد کلمات آن‌ها باشد. اگر کلمات زیادی با یک نامه الکترونیکی هرزنامه شناخته شود، این سیستم ایمیل را به‌عنوان اسپم علامت‌گذاری می‌کند.

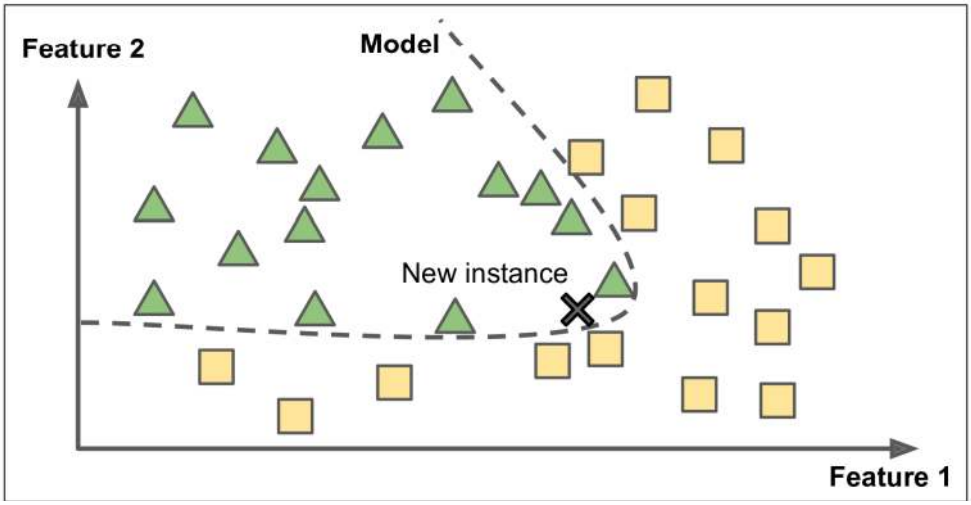
در یادگیری مبتنی بر نمونه سیستم نمونه‌ها را حفظ می‌کند، سپس با مقایسه آن‌ها با مثال‌های یاد گرفته‌شده به موارد جدید تعمیم می‌یابد. به‌عنوان‌مثال، در شکل ۱۵-۱ نمونه جدید به‌عنوان یک مثلث طبقه‌بندی می‌شود زیرا اکثر مشابه‌ترین نمونه‌ها به آن کلاس تعلق دارند.



شکل ۱۵-۱. یادگیری مبتنی بر نمونه.

یادگیری مبتنی بر مدل

روش دیگر برای تعمیم مجموعه‌ای از مثال‌ها، ساخت مدلی از آن‌ها است، سپس برای پیش‌بینی از آن مدل استفاده می‌شود. این را یادگیری مبتنی بر مدل می‌نامند (شکل ۱۶-۱).



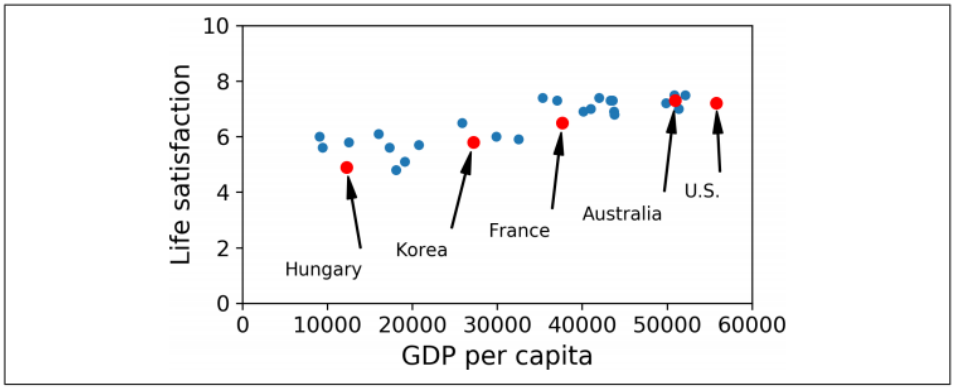
شکل ۱۶-۱. یادگیری مبتنی بر مدل

به‌عنوان‌مثال می‌خواهید بدانید که آیا پول مردم را خوشحال می‌کند، بنابراین داده‌های شاخص زندگی بهتر را از وب‌سایت OECD و همچنین آمار مربوط به تولید ناخالص داخلی سرانه از وب‌سایت صندوق بین‌المللی پول بارگیری می‌کنید. سپس جداول را به هم پیوست داده و بر اساس سرانه تولید ناخالص داخلی مرتب کنید. جدول ۱-۱ گزیده‌ای از آنچه دریافت می‌کنید را نشان می‌دهد.

جدول ۱-۱: آیا پول باعث خوشحالی مردم می‌شود؟

Country	GDP per capita (USD)	Life satisfaction
Hungary	12,240	4.9
Korea	27,195	5.8
France	37,675	6.5
Australia	50,962	7.3
United States	55,805	7.2

حال داده‌ها را برای چند کشور تصادفی رسم می‌کنیم (شکل ۱۷-۱).



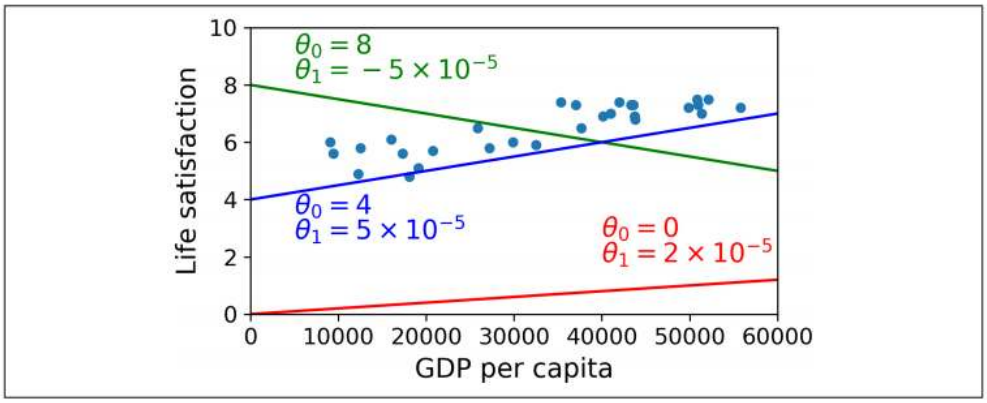
شکل ۱۷-۱. آیا روندی در اینجا می‌بینید؟

به نظر می‌رسد در اینجا روندی وجود دارد! اگرچه داده‌ها نویز دار هستند (یعنی تا حدی تصادفی)، اما به نظر می‌رسد با افزایش سرانه تولید ناخالص داخلی کشور، رضایت از زندگی کم‌وبیش به‌صورت خطی افزایش میابد. بنابراین تصمیم می‌گیرید رضایت از زندگی را به‌عنوان یک تابع خطی از تولید ناخالص داخلی سرانه مدل کنید.

این مرحله انتخاب مدل نامیده می‌شود: شما فقط با یک ویژگی، تولید ناخالص داخلی سرانه، یک مدل خطی از رضایت زندگی را انتخاب کرده‌اید (معادله ۱-۱).

$$life_satisfaction = \theta_0 + \theta_1 \times GDP_per_capita$$

این مدل دارای دو پارامتر مدل θ۰ و θ۱ است. با دست‌کاری این پارامترها، مدل خود را نشان‌دهنده هر عملکرد خطی می‌کنید، همان‌طور که در شکل ۱۸-۱ نشان داده‌شده است.

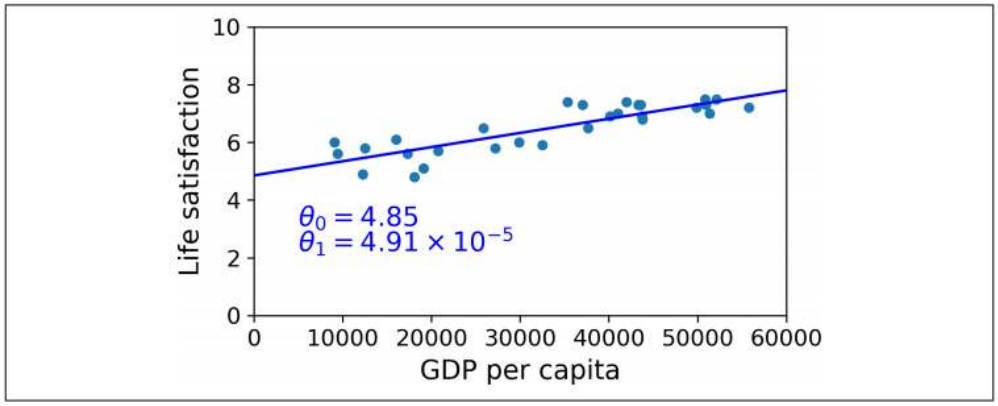


شکل ۱۸-۱. چند مدل خطی ممکن.

قبل از این‌که بتوانید از مدل خود استفاده کنید، باید مقادیر پارامتر θ۰ و θ۱ را تعریف کنید حال چگونه می‌توانید بفهمید کدام مقادیر باعث عملکرد بهتر مدل شما می‌شود؟ برای پاسخ به این سؤال، باید معیار عملکرد را مشخص کنید. یک تابع سودمندی (یا تابع تناسب) تعریف کنید که میزان خوب بودن مدل شما را اندازه‌گیری کند، یا می‌توانید یک تابع هزینه را تعریف کنید که بد بودن آن را اندازه‌گیری کند. برای مسائل رگرسیون خطی، مردم معمولاً از یک تابع هزینه استفاده می‌کنند که فاصله بین پیش‌بینی‌های مدل خطی و نمونه‌های آموزش را اندازه‌گیری می‌کند. هدف این است که این فاصله را به حداقل برسانید.

این همان‌جایی است که الگوریتم خطی رگرسیون ارائه می‌شود: شما نمونه‌های آموزشی خود را به آن خورانده و این الگوریتم پارامترهایی پیدا می‌کند که مدل خطی را به بهترین وجه با داده‌های شما سازگار می‌کند. به این روند آموزش مدل می‌گویند. الگوریتم ما دریافت که مقادیر بهینه پارامتر $\theta_0 = 4.85$ و $\theta_1 = 4.91 \times 10^{-5}$ هستند.

همان‌طور که در شکل ۱۹-۱ می‌بینید، این مدل تا حد ممکن متناسب با داده‌های آموزش است (برای یک مدل خطی).



شکل ۱۹-۱. مدل خطی که با داده‌های آموزش متناسب است.

اکنون آماده اجرای مدل برای پیش‌بینی هستید. به‌عنوان‌مثال می‌خواهید بدانید مردم قبرس چقدر خوشحال هستند و داده‌های سازمان همکاری و توسعه اقتصادی پاسخی برای آن ندارد. می‌توانید از مدل خود برای پیش‌بینی خوب استفاده کنید: سرانه تولید ناخالص داخلی قبرس را جست‌وجو می‌کنید، ۲۲۵۸۷ دلار پیدا می‌کنید و سپس مدل خود را اعمال و متوجه می‌شوید که رضایت از زندگی احتمالاً در حدود $4.85 + 22{,}587 \times 4.91 \times 10^{-5} = 5.96$.

مثال ۱-۱ کد پایتونی را نشان می‌دهد که داده‌ها را بارگیری، آن‌ها را آماده، طرح پراکنده‌ای برای مصورسازی ایجاد و سپس مدل خطی را آموزش داده و پیش‌بینی می‌کند.

مثال۱-۱:آموزش و اجرای یک مدل خطی با استفاده از Scikit-Learn

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

# Load the data
oecd_bli = pd.read_csv("oecd_bli_2015.csv", thousands=',')
gdp_per_capita = pd.read_csv("gdp_per_capita.csv",thousands=',',delimiter='\t',
encoding='latin1', na_values="n/a")

# Prepare the data
country_stats = prepare_country_stats(oecd_bli, gdp_per_capita) X =
np.c_[country_stats["GDP per capita"]]
y = np.c_[country_stats["Life satisfaction"]]

# Visualize the data
country_stats.plot(kind='scatter', x="GDP per capita", y='Life
satisfaction') plt.show()
```

```
# Select a linear model
model = sklearn.linear_model.LinearRegression()

# Train the model
model.fit(X, y)

# Make a prediction for Cyprus
X_new = [[22587]] # Cyprus' GDP per capita
print(model.predict(X_new))   # outputs [[ 5.96242338]]
```

اگر به‌جای آن از الگوریتم یادگیری مبتنی بر نمونه استفاده کرده باشید، متوجه می‌شوید که اسلوونی نزدیک‌ترین سرانه تولید ناخالص داخلی به سرانه قبرس (20732 دلار) را دارد و ازآنجاکه داده‌های OECD به ما می‌گوید رضایت از زندگی اسلوونی‌ها 5.7 است، شما رضایت از زندگی ۵٫۷ برای قبرس را پیش‌بینی می‌کنید. اگر به دو کشور بعدی نزدیک نگاه کنید، پرتغال و اسپانیا را به ترتیب با رضایت 5.1 و 6.5 خواهید یافت. با میانگین این سه مقدار، 5.77 به دست می‌آید که تقریباً به پیش‌بینی مبتنی بر مدل نزدیک است. به این الگوریتم ساده رگرسیون k **نزدیک‌ترین همسایه**[۹] گفته می‌شود(در این مثال k = 3)

جایگزینی مدل رگرسیون خطی با رگرسیون kنزدیک‌ترین همسایه در کد قبلی تنها جایگزینی دو خط کد زیر است:

```
import sklearn.linear_model
model = sklearn.linear_model.LinearRegression()
with these two:
import sklearn.neighbors
model = sklearn.neighbors.KNeighborsRegressor(n_neighbors=3)
```

اگر همه‌چیز خوب پیش برود، مدل شما پیش‌بینی‌های خوبی انجام می‌دهد. در غیر این صورت، لازم است که از ویژگی‌های بیش‌تری (میزان اشتغال، سلامت، آلودگی هوا و غیره) استفاده، داده‌های آموزشی با کیفیت بیش‌تر یا بهتر انتخاب کرده یا شاید مدل قدرتمندتری (مثلاً مدل رگرسیون چندجمله‌ای) انتخاب کنید.

به‌طور خلاصه:

- داده‌ها را مطالعه کردید.
- مدلی را انتخاب کردید.
- با استفاده از داده‌ها آموزش دادید (به‌عنوان‌مثال، الگوریتم یادگیری مقادیر پارامتر مدلی را جست‌وجو می‌کند که تابع هزینه را به حداقل می‌رساند).

[۹] k-Nearest Neighbors regression

- سرانجام، برای پیش‌بینی موارد جدید از این مدل استفاده کردید (استنتاج) و امیدواریم که این مدل به‌خوبی تعمیم یابد.

تاکنون زمینه‌های زیادی در مورد یادگیری ماشین را تحت پوشش قرار دادیم؛ اکنون می‌دانید یادگیری ماشین واقعاً در مورد چه چیزی است، چرا مفید است، انواع متداول سیستم‌های آن چیست و گردش کار پروژه به‌طورمعمول چگونه است. حال بیایید ببینیم چه چیزی در یادگیری اشتباه رخ‌داده و از پیش‌بینی دقیق جلوگیری می‌کند.

۴–۱. چالش‌های اصلی یادگیری ماشین

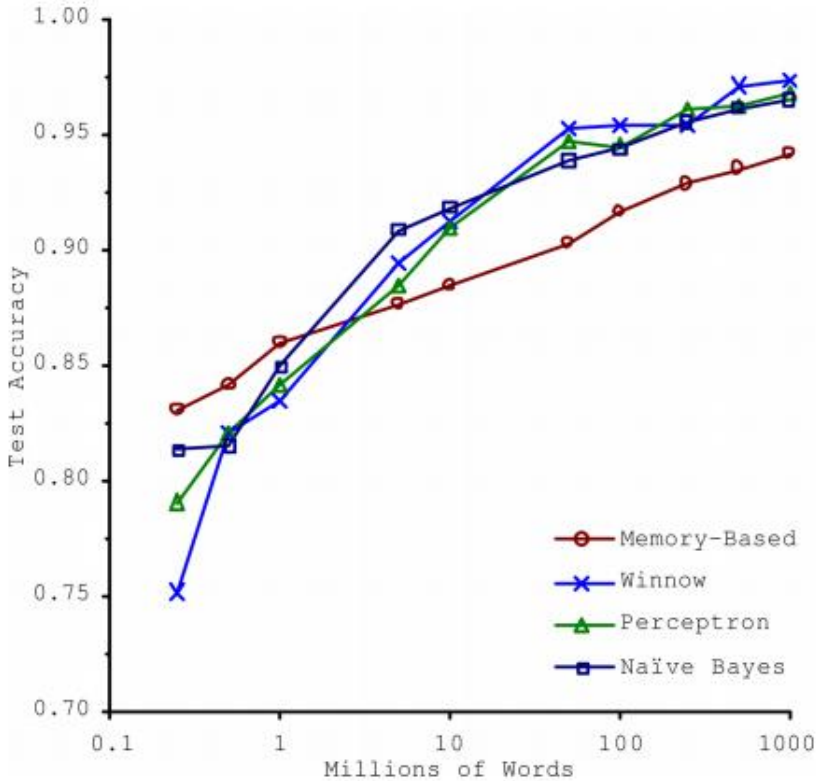
به‌طور خلاصه، ازآنجاکه وظیفه اصلی شما انتخاب یک الگوریتم یادگیری و آموزش آن روی برخی داده‌ها است دو موردی که می‌تواند اشتباه ایجاد کند الگوریتم و داده بد است. بیایید با نمونه‌هایی از داده‌های بد شروع کنیم.

مقدار کافی اطلاعات آموزشی یک کودک نوپا برای این‌که بفهمد سیب چیست، کافی است شما به یک سیب اشاره‌کرده و بگویید سیب (احتمالاً این روش را چند بار تکرار کنید). اکنون کودک قادر است سیب‌ها را در انواع رنگ‌ها و اشکال تشخیص دهد. اما برای کارکرد صحیح اکثر الگوریتم‌های یادگیری ماشین داده‌های زیادی لازم است. حتی برای مسائل بسیار ساده شما معمولاً به هزاران مثال نیاز دارید و برای مسائل پیچیده مانند تشخیص تصویر یا گفتار ممکن است به میلیون‌ها مثال نیاز داشته باشید (مگر این‌که بتوانید از قسمت‌های مدل موجود دوباره استفاده کنید).

۴–۱–۱. اثربخشی نامعقول داده‌ها[۱۰]

محققان مایکروسافت، میشل بانکو و اریک بریل در مقاله مشهوری که در سال ۲۰۰۱ منتشر شد؛ نشان دادند که الگوریتم‌های بسیار متفاوت یادگیری ماشین، در مورد یک مسئله پیچیده ابهام‌زدایی از زبان طبیعی در صورت داشتن داده‌های کافی (شکل ۲۰–۱) تقریباً یکسان عمل می‌کنند.

[۱۰] The Unreasonable Effectiveness of Data



شکل ۲۰-۱. اهمیت داده در مقابل الگوریتم.

این نتایج نشان می‌دهد که ما ممکن است بخواهیم در مورد هزینه صرف وقت و پول برای توسعه الگوریتم، در مقابل هزینه آن برای توسعه پیکره تجدیدنظر کنیم.

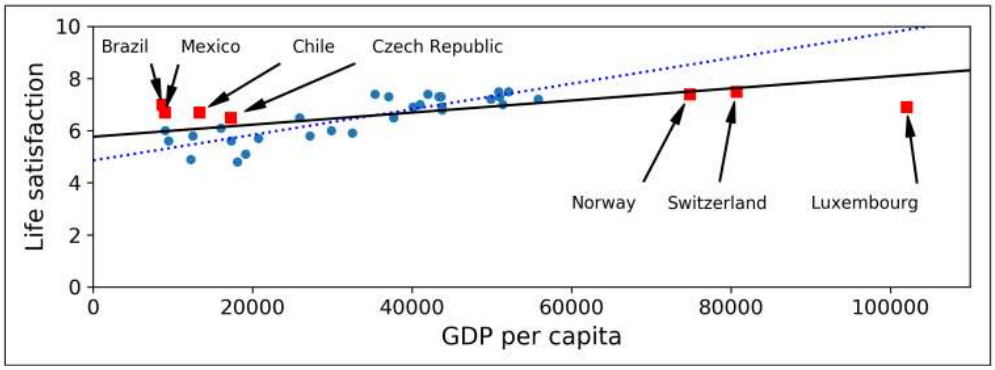
این ایده که داده‌ها برای مسائل پیچیده بیش از الگوریتم‌ها اهمیت دارند، توسط پیتر نورویگ و همکاران در مقاله‌ای تحت عنوان اثربخشی غیرمعقول داده‌ها در سال ۲۰۰۹ منتشرشده است.

بااین‌حال، باید توجه داشت که مجموعه داده‌های کوچک و متوسط هنوز هم بسیار رایج هستند و دریافت داده‌های آموزش اضافی همیشه آسان یا ارزان نیست، بنابراین فعلاً نباید الگوریتم‌ها را کنار گذاشت.

۲-۴-۱. داده‌های آموزشی غیر معرف

به‌منظور تعمیم خوب، بسیار مهم است که داده‌های آموزش شما نمایانگر موارد جدیدی باشد که به آن‌ها تعمیم می‌دهید. این‌که آیا شما از یادگیری مبتنی بر نمونه یا مبتنی بر مدل استفاده می‌کنید؛ درست است.

به‌عنوان‌مثال، مجموعه کشورهایی که قبلاً برای آموزش مدل خطی استفاده کردیم کاملاً معرف نبود و کشورهای کمی ازدست‌رفته بودند. شکل ۲۱-۱ نشان می‌دهد که داده‌ها هنگام اضافه کردن کشورهای گمشده به چه صورت است.



شکل ۲۱-۱. یک نمونه آموزش معرف‌تر.

اگر یک مدل خطی را روی این داده‌ها آموزش دهید، خط پُری را به دست می‌آورید، درحالی‌که مدل قدیمی با خط نقطه‌ای نشان داده می‌شود. همان‌طور که می‌بینید نه‌تنها افزودن چند کشور گمشده به‌طور قابل‌توجهی مدل را تغییر می‌دهد؛ بلکه روشن می‌کند که چنین مدل خطی ساده احتمالاً هرگز به‌خوبی عمل نخواهد کرد. به نظر می‌رسد که کشورهای بسیار ثروتمند شادتر از کشورهای ثروتمند متوسط نیستند و برعکس برخی از کشورهای فقیر از بسیاری از کشورهای ثروتمند خوشحال‌تر به نظر می‌رسند.

با استفاده از مجموعه آموزش غیرمعرفتی، الگویی را آموزش دادیم که بعید است پیش‌بینی دقیق داشته باشد؛ مخصوصاً برای کشورهای بسیار فقیر و بسیار ثروتمند.

استفاده از یک مجموعه آموزشی نمایانگر مواردی تعمیم دهنده بسیار مهم است. این اغلب دشوارتر از آن است که به نظر می‌رسد: اگر نمونه خیلی کوچک باشد، شما نویز نمونه‌برداری خواهید داشت؛ اما اگر روش نمونه‌برداری ناقص باشد، حتی نمونه‌های بسیار بزرگ نیز می‌توانند غیر معرف باشند. به این اتفاق بایاس نمونه‌گیری گفته می‌شود.

مثال معروف بایاس نمونه‌گیری

شاید مشهورترین نمونه بایاس نمونه‌گیری در جریان انتخابات ریاست جمهوری ایالات‌متحده در سال ۱۹۳۶ اتفاق افتاد، که لاندون را در برابر روزولت قرار داد: Literary Digest یک نظرسنجی بسیار بزرگ انجام داد و برای حدود ۱۰ میلیون نفر نامه ارسال کرد. این نظرسنجی 2.4 میلیون پاسخ دریافت کرد و با اطمینان بالا پیش‌بینی کرد که لاندون ۵۷درصد آرا را به خود اختصاص دهد بااین‌وجود روزولت با ۶۲درصد آرا پیروز شد. این نقص در Literary Digest بود:

- ابتدا، برای به دست آوردن آدرس‌ها جهت ارسال نظرسنجی‌ها از دایرکتوری‌های تلفنی، لیست مشترکین مجله‌ها، لیست‌های عضویت در باشگاه‌ها و موارد مشابه استفاده کرد و همه این لیست‌ها به نفع افراد ثروتمندی است که بیش‌تر به جمهوری خواهان(لاندون) رأی می‌دهند.
- دوم، کم‌تر از ۲۵درصد از افرادی که نظرسنجی را دریافت کردند پاسخ داده‌اند. این هم با رد کردن افرادی که زیاد به سیاست اهمیت نمی‌دهند، افرادی که Literary Digestرا دوست ندارند و سایر گروه‌های اصلی نشان‌دهنده نوعی بایاس نمونه‌گیری است. این نوع خاصی از بایاس نمونه‌گیری است که **بایاس بدون پاسخ**[۱۱] می‌نامند.

۳-۴-۱. داده‌های کم کیفیت

اگر داده‌های آموزش شما مملو از خطا، داده‌های پرت و اشکال باشد (به‌عنوان‌مثال، به دلیل آمار ناکافی)، تعیین منبع برای سیستم شما دشوار خواهد بود. این الگو می‌تواند باعث ازکارافتادن سیستم شود. معمولاً ارزش صرف وقت پاک‌سازی داده‌های آموزش را دارد.

حقیقت این است که اکثر دانشمندان وقت خود را برای انجام این کار صرف می‌کنند. برای مثال:

- اگر در بعضی از نمونه‌ها به‌طور واضح پرت باشند؛ می‌توانید آن‌ها را نادیده گرفته یا سعی کنید آن را به‌صورت دستی حل کنید.
- اگر برخی از نمونه‌ها ویژگی‌هایی را از دست دادند (به‌عنوان‌مثال ۵درصد از مشتریان سن خود را مشخص نمی‌کنند) باید تصمیم بگیرید که آیا می‌خواهید این ویژگی را نادیده بگیرید و مقدار گمشده را وارد کنید، یا مدلی با این ویژگی‌ها و یک مدل بدون آن‌ها را آموزش دهید.

۴-۴-۱. ویژگی‌های نامربوط

سیستم شما فقط در صورتی قادر به یادگیری خواهد بود که داده‌های آموزش شامل ویژگی‌های مربوط و ویژگی‌های کم‌تر بی‌ربط باشد. بخش مهمی از موفقیت یک پروژه یادگیری ماشین، ارائه مجموعه‌ای خوب از ویژگی‌ها برای آموزش است. این فرایند که مهندسی ویژگی نامیده می‌شود شامل موارد زیر است:

- انتخاب ویژگی‌ها: انتخاب مفیدترین ویژگی‌ها در بین ویژگی‌های موجود برای آموزش.
- استخراج ویژگی: ترکیب ویژگی‌های موجود برای تولید یک ویژگی مفیدتر (همان‌طور که قبلاً دیدیم، الگوریتم‌های کاهش بعد می‌توانند کمک کنند).
- ایجاد ویژگی‌های جدید با جمع‌آوری داده‌های جدید.

اکنون‌که نمونه‌های زیادی از داده‌های بد را بررسی کردیم، به چند نمونه از الگوریتم‌های بد نگاهی خواهیم انداخت.

---

[۱۱] nonresponse bias