

جزوه دوره

جامپ یادگیری ماشین Quera College

تهیه شده توسط علی فاضل نیا

دانشجوی علوم کامپیوتر

راههای ارتباط:

Email: alifazelniya.1384@gmail.com

Telegram: @Norbert_desu

GitHub: github.com/AliFazelniya

فهرست مطالب

ت	پیشگفتار
۱	۱ مقدمه
۱	۱ سلام!
۲	۲ قالب کالج
۳	۳.۱ سیستم امتیازات و دریافت گواهی اصلی
۴	۴.۱ یادگیری ماشین چیست؟
۱۱	۵.۱ چرا پایتون؟
۱۳	۶.۱ آماده‌سازی محیط کار
۱۳	۷.۱ اجرای نوت‌بوک‌ها
۱۳	۸.۱ گوگل کولب
۱۳	۹.۱ معرفی مجموعه‌داده
۱۴	۲ مدیریت پروژه
۱۴	۱.۲ اهداف فصل
۱۵	۲.۲ چرخه پروژه
۱۷	۳.۲ اولویت‌بندی
۲۰	۴.۲ سازماندهی تیم
۲۲	۵.۲ چرا پروژه‌ها شکست می‌خورند؟
۲۶	۳ آماده‌سازی داده
۲۶	۱.۳ اهداف فصل
۲۷	۲.۳ سوالاتی درباره داده
۳۱	۳.۳ چالش‌های داده
۳۵	۴.۳ ویژگی‌های مجموعه‌داده‌ی خوب
۳۶	۵.۳ تقسیم‌بندی مجموعه‌داده
۴۲	۶.۳ داده‌های پرت
۴۲	۷.۳ مقادیر گم‌شده
۴۳	۸.۳ مجموعه‌داده نامتوازن
۴۴	۴ مهندسی ویژگی
۴۴	۱.۴ اهمیت
۴۴	۲.۴ مشخصات ویژگی خوب

۱۴۴	۳.۴ ویژگی‌های دسته‌ای
۴۴	۴.۴ مقادیر گم شده در ویژگی‌های دسته‌ای
۴۴	۵.۴ ویژگی‌های تقویمی
۴۵	۶.۴ سنتز ویژگی
۴۵	۷.۴ تغییر مقیاس ویژگی
۴۵	۸.۴ نشت داده
۴۵	۹.۴ فوت و فن‌های مهندسی ویژگی
۴۵	۱۰.۴ کاهش ابعاد
۴۵	۱۱.۴ انتخاب ویژگی
۴۵	۱۲.۴ خط لوله
۴۶	۵ رگرسیون
۴۶	۱.۵ اهداف فصل
۴۶	۲.۵ مقدمه
۴۶	۳.۵ مدل چیست؟
۴۶	۴.۵ تخمین، تابع هزینه و بهینه‌سازی
۴۶	۵.۵ رگرسیون خطی
۴۷	۶.۵ ارزیابی
۴۷	۷.۵ رگرسیون چندجمله‌ای
۴۷	۸.۵ عمومیت
۴۷	۹.۵ رگولاریزیشن
۴۸	۶ دسته‌بندی
۴۸	۱.۶ مقدمه
۴۸	۲.۶ رگرسیون لجستیک
۴۸	۳.۶ ارزیابی - قسمت اول
۴۸	۴.۶ ارزیابی - قسمت دوم
۴۸	۵.۶ کراس ولیدیشن
۴۹	۶.۶ نزدیکترین- k همسایه
۴۹	۷.۶ بیز ساده‌لوحانه
۴۹	۸.۶ ماشین بردار پشتیبان
۴۹	۹.۶ هایپرپارامترها
۴۹	۱۰.۶ آشنایی با کتابخانه‌ی $H2M$
۴۹	۱۱.۶ درخت تصمیم
۴۹	۱۲.۶ فوت و فن درخت تصمیم
۵۰	۱۳.۶ بیش‌برازش درخت تصمیم
۵۱	۷ یادگیری تجمعی
۵۱	۱.۷ اهداف فصل
۵۱	۲.۷ مقدمه
۵۱	۳.۷ جنگل تصادفی

۵۱	الگوریتم AdaBoost	۴.۷
۵۱	الگوریتم Boosting Gradient	۵.۷
۵۲	الگوریتم XGboost	۶.۷
۵۲	روش Stacking	۷.۷
۵۳		پروژه اول	۸
۵۳	مقدمه	۱.۸
۵۳	یادداشت‌ها و راه حل	۲.۸
۵۴		شبکه عصبی	۹
۵۴	اهداف فصل	۱.۹
۵۴	پرسپترون	۲.۹
۵۴	آموزش پرسپترون	۳.۹
۵۴	پرسپترون چندلایه	۴.۹
۵۴	عمومیت	۵.۹
۵۵		یادگیری ناظارت‌نشده	۱۰
۵۵	مقدمه	۱.۱۰
۵۵	الگوریتم PCA	۲.۱۰
۵۵	الگوریتم t-SNE	۳.۱۰
۵۵	k-means خوشه‌بندی با	۴.۱۰
۵۵	k-modes خوشه‌بندی با	۵.۱۰
۵۶	k-prototype خوشه‌بندی با	۶.۱۰
۵۷		پروژه دوم	۱۱
۵۷	اهداف فصل	۱.۱۱
۵۷	تعابیری متن	۲.۱۱
۵۷	فاصله‌ی ویرایش	۳.۱۱
۵۷	RBO معیار شباهت	۴.۱۱
۵۸		بیشتر بدانید	۱۲
۵۸	نمونه‌کاهی با NearMiss	۱.۱۲
۵۸	نمونه‌افزایی با SMOTE	۲.۱۲
۵۸	درخت رگرسیون	۳.۱۲

پیشگفتار

اینجا هدف، دامنه، و نحوه استفاده از جزو را بنویس. (می‌توانی همین متن را بعداً دقیقاً با متن خودت جایگزین کنی).

فصل ۱

مقدمه

۱.۱ سلام!

سلام؛ ورود شما را به کالج «یادگیری ماشین ۲ | جامپ تکنیکال» خوشآمد می‌گوییم. از اینکه در کوئرا کالج افتخار میزبانی شما را داریم، به خود می‌بایلیم

«یادگیری ماشین ۲ | جامپ تکنیکال» کالج سوم از مسیر علم داده و یادگیری ماشین کوئراست که پس از «یادگیری ماشین ۰ | دروازه ورود» و «یادگیری ماشین ۱ | تحلیل داده با پایتون» طراحی شده است.

هدف ما از تدوین کالج این است که شما را به شکل اصولی و گامبهگام با الگوریتم‌های یادگیری ماشین آشنا کنیم؛ به طوری که در نهایت بتوانید الگوریتم‌ها را تحلیل کنید، نقاط ضعف و قوت آن‌ها را بشناسید و تشخیص دهید چگونه از آن‌ها برای حل مسائل دنیای واقعی کمک بگیرید! علاوه بر این که با الگوریتم‌های یادگیری ماشین آشنا می‌شویم، تکنیک‌هایی برای استفاده عملی از آن‌ها را نیز خواهید آموخت؛ به عبارت بهتر، «یادگیری ماشین ۲ | جامپ تکنیکال» آمیخته‌ای موزون از دانش علمی و مهارت عملی یادگیری ماشین کلاسیک است!

این کالج مناسب افرادی است که در حد متوسط با پایتون و کتابخانه‌های Numpy و Pandas آشنایی داشته باشند. داشتن دانش حداقلی از آمار احتمال و حسابان (مباحث مشتق) و جبر خطی (ماتریس‌ها) به شما در فرآگیری محتوای این کالج کمک می‌کند. به صورت کلی می‌توان گفت اگر با ریاضیات در حد مقطع دبیرستان آشنا باشید، به راحتی می‌توانید به ماجراجویی در این کالج بپردازید! اگر با پیشنبازهای کالج آشنا نیستید، می‌توانید با گذراندن سایر دوره‌های کوئرا کالج، مهارت‌های لازم را کسب کنید.

شما با داشتن پیش‌نیازهای کالج شروع می‌کنید و در انتهای پس از مطالعه‌ی درسنامه‌ها و حل تمرین‌ها، به «دانشمند داده» و «مهندس یادگیری ماشین» تبدیل می‌شوید که قطعاً راه طولانی اما شیرینی برای حرفه‌ای شدن پیش رو دارد!

البته برای حفظ تناسب کالج و افزایش احتمال یادگیری عمیق، از آموزش کامل یادگیری عمیق، که زیرمجموعه یادگیری ماشین است، پرهیز شده است؛ بلکه شبکه‌های عصبی مصنوعی و مقدمات یادگیری عمیق را در فصل «شبکه عصبی» آورده‌ایم. برای فرآگیری یادگیری عمیق، می‌توانید منتظر کالج‌های بعدی مسیر علم داده و یادگیری ماشین کوئرا کالج باشید

امیدواریم با ارائه‌ی آموزش باکیفیت، گامی مثبت در افزایش دانش و پیشرفت شما برداریم. سختی‌های

مسیر نه تنها ناامیدمان نمی‌کند بلکه توانمان را بیشتر و تصمیم‌مان را راسخ‌تر می‌کند. مسیر رسیدن به قله‌ای که آرزویش را داریم؛ جوانانی توانمند، پرتلاش و ایرانی پیشرفته...



۲.۱ قالب کالج

قالب کالج روند آموزشی کوئرا کالج

امروزه با فرآگیر شدن آموزش‌های آنلاین، قالب‌های متعددی برای یادگیری مفاهیم علوم کامپیوتر بصورت آنلاین پیاده‌سازی شده است. آموزش آنلاین این مفاهیم، فرصت برنامه‌نویسی به همراه یادگیری را فراهم می‌کند و همچنین انجام تمرین‌های واقعی و استفاده از کتابخانه‌ها و قالب‌های نزدیک به صنعت را تسهیل می‌کند. اما آموزش آنلاین چالش‌هایی نیز به همراه دارد؛ زیرا تعاملی که استاد در سر کلاس با دانشجو دارد و نظمی که جلسات کلاس به یادگیری دانشجو می‌دهد به سختی در قالب‌های آموزش آنلاین گنجانده می‌شود. در کوئرا کالج تلاش کردیم قالبی برای برطرف شدن نیاز تعاملی بودن، نظم، و همچنین عملی بودن آموزش آماده کنیم.

در کل، آموزش این دوره متشکل از چندین فصل می‌باشد، که هر فصل شامل تعدادی درسنامه، تمرین یا آزمون است.

درسنامه‌ها

پس از مطالعه‌ی هر درسنامه می‌توانید تیک "خواندم" آن را بزنید و پیش بروید. در پایین هر درسنامه بخش کامنت‌ها تعییه شده تا بتوانید پرسش‌ها و نظرات عمومی خود را درباره‌ی درسنامه مربوطه با ما و سایر

شرکت‌کنندگان دوره مطرح کنید.

تمرین‌ها

تمرین‌ها نیز همگی توسط سامانهٔ داوری خودکار Quera تصحیح شده، و پس از ارسال کد، توسط سیستم، نمره‌دهی می‌شود. داوری تمرین ممکن است شامل چندین تست مختلف باشد و سعی شده هنگامیکه در کسب نمره‌ی کامل یک تست دچار مشکل می‌شود با خورد مناسبی توسط سیستم خروجی داده شود.

در تمرین‌هایی که عملکرد مدل یادگیری ماشین شما سنجیده می‌شود یک حد آستانه (Threshold) تعريف شده و در صورتی‌که مدل شما عملکردی بهتر از آن حد داشته باشد تمرین با موفقیت گذرانده می‌شود. البته اگر عملکرد مدل شما از مقدار خواسته شده بهتر باشد امتیاز اضافه‌تری کسب خواهد کرد.

آزمون‌ها

در بعضی از فصل‌ها به منظور درک عمیق‌تر مباحث آموخته شده تعدادی آزمون چندگزینه‌ای در نظر گرفته شده است. پاسخ شما به آزمون‌ها نیز مشابه با تمرین‌ها توسط سامانهٔ داوری خودکار Quera انجام خواهد گرفت. هرچند که از نظر تعداد ارسال پاسخ محدودیتی نخواهید داشت، با این حال امتیاز آزمون‌ها در مقایسه با تمرین‌ها کمتر در نظر گرفته شده است.

پشتیبانی

ما گام به گام در طول مسیر این کالج همراه‌تان هستیم! در صورت وجود هرگونه پرسش یا ابهام دربارهٔ هر بخشی از این کالج می‌توانید از طریق پیغام خصوصی با پشتیبانان کالج در ارتباط باشید.

۳.۱ سیستم امتیازات و دریافت گواهی اصلی

سیستم امتیازات و دریافت گواهی اصلی

با حل هر تمرین یا آزمون، مقداری امتیاز به شما تعلق می‌گیرد. با این امتیاز می‌توانید راه حل تمرین‌های دوره را دریافت کنید. پس از پایان دوره، بر اساس امتیاز کسب شده توسط شما، گواهی صادر خواهد شد. پس سعی کنید از امتیازات خود به نحو مناسبی استفاده کنید! گذراندن فصل‌ها

دوره‌ی «یادگیری ماشین ۲ اجمالی تکنیکال» از چندین فصل تشکیل شده است. پیشنهاد ما این است که برنامه خود را مطابق با سرفصل دوره تنظیم کرده و مباحث را به ترتیب پیش ببرید. چیدمان محتوا، مورد تایید مهندسان و استادان برتر داخلی و خارجی است و پیش‌رفتن طبق آن، یادگیری شما را تضمین می‌کند! توضیحات دریافت گواهی

شما با حل هر یک از تمرین‌های دوره، مقداری امتیاز دریافت می‌کنید و می‌توانید با استفاده از این امتیازات، جواب تمرین‌ها را خریداری کنید. زمانی که دوره را به اتمام می‌رسانید، مقداری امتیاز برای شما باقی می‌ماند و بر حسب این مقدار به شما گواهی داده می‌شود.

اگر جواب هیچ تمرینی را خریداری نکنید و همه تمرین‌ها را حل کنید، مجموع امتیاز شما به ۳۲۰۰ می‌رسد.

در نهایت بر حسب امتیاز نهایی، یکی از چهار سطح زیر در گواهی گزارش می‌شود:

سطح Perfect برای نمرات بالای ۲۲۰۰

سطح Good Very برای نمرات کمتر از ۲۲۰۰ و بالای ۱۸۰۰

سطح Good برای نمرات کمتر از ۱۸۰۰ و بالای ۱۴۰۰

سطح Fair برای نمرات کمتر از ۱۴۰۰

برای هر فصل، یک آستانه تعریف شده است که شما برای دریافت گواهی، حتما باید بیشتر از آستانه، فصل را مطالعه کرده باشید. به عنوان مثال اگر فصلی دارای آستانه ۸۰ درصدی باشد، برای آنکه بتوانید گواهی را دریافت کنید، حداقل ۸۰ درصد آن فصل را باید مطالعه کنید. حتی اگر امتیازتان بیشتر از ۱۶۰۰ باشد ولی فصلی باشد که کمتر از آستانه مطالعه شده باشد، گواهی صادر نمی‌شود!

البته پس از پایان دوره شما می‌توانید با حل سوالات دیگر داخل فصل‌های دوره، امتیاز خود را افزایش داده و گواهی Perfect را دریافت کنید.

۴.۱ یادگیری ماشین چیست؟

یادگیری ماشین چیست؟

فرض کنید دهه پنجاه میلادی است و چیزی به نام یادگیری ماشین وجود ندارد. در آن دوره یکی از مهندسان IBM که نامش آقای آرتور ساموئل (Arthur Samuel) بود، برای اولین بار از عبارت یادگیری ماشین (Machine Learning) استفاده کرد و تعریف زیر را برای آن ارائه داد: «یادگیری ماشین زمینه‌ای از تحقیقات است که به کامپیوترها توانایی یادگیری بدون برنامه‌نویسی صحیح را می‌دهد.»

در دیدگاه آرتور ساموئل، یادگیری ماشین با برنامه‌نویسی صحیح تفاوت دارد. در برنامه‌نویسی ساده، ما باید الگوهای متفاوت را خودمان تشخیص داده و به صورت دستی برای آنها برنامه‌نویسی کنیم. اما در یادگیری ماشین، ما صرفاً مدلی را طراحی می‌کنیم که قادر به یادگیری و پیدا کردن خودکار الگوها از روی مجموعه داده (Dataset) است. در مواردی که مسئله‌ی مورد نظر پیچیده شود، پیدا کردن این الگوها اغلب برای انسان دشوار یا حتی غیر ممکن است. اما برای ماشین‌ها به دلیل قدرت پردازشی بالا و توانایی استفاده از الگوریتم‌ها، این کار بسیار ساده‌تر است. همین موضوع، دلیل اصلی شهرت یادگیری ماشین است. الگوریتم‌های یادگیری ماشین، مثل انسان به کمک تجربه یاد می‌گیرند. داده (Data) همان تجربه‌ای است که به عنوان ورودی به الگوریتم داده می‌شود.

آقای تام میشل (Tom Mitchell) در کتاب یادگیری ماشین خود، یادگیری ماشین را از دید مهندسی به این شکل تعریف کرده‌است: «اگر کارایی برنامه در انجام تکلیف TT که با معیار عملکرد PP ارزیابی می‌شود، با تجربه‌ی EE افزایش یابد، می‌گوییم که برنامه یاد گرفته است از تجربه‌ی EE با توجه به تکلیف TT و معیار عملکرد PP استفاده کند.» تکلیف (Task)

تکلیف در واقع همان مسئله‌ای است که ما انتظار داریم بتوانیم با یادگیری ماشین حل کنیم. برای مثال بانک را تصور کنید که می‌خواهد تصمیم بگیرد آیا به یک مشتری وام اختصاص بدهد یا خیر. انتخاب وام دادن یا ندادن به مشتری را تکلیف TT می‌گوییم. تجربه (Experience)

برای انجام فرآیند یادگیری که منجر به حل تکلیف TT می‌شود، نیازمند تعدادی نمونه (Sample) هستیم که اطلاعات مورد نیاز در مورد مسئله را به ما می‌دهند. برای مثال در مسئله‌ی وام دادن بانک، می‌توان از سابقه‌ی مشتریان پیشین و این که وام خود را پرداخت کرده‌اند یا خیر برای مجموعه داده یا نمونه‌ها استفاده

کرد. معیار عملکرد (Performance)

هر مدل یادگیری ماشینی که طراحی کنیم، همواره به طور قطعی و ۱۰۰ درصدی نتیجه‌ی درست و مناسبی را ارائه نمی‌دهد؛ بنابراین به معیاری برای بررسی و اندازه‌گیری عملکرد آن نیاز داریم تا در صورت عملکرد نامناسب بتوانیم با تغییر پارامترها به مدل بهتری دست یابیم. به این معیار، معیار عملکرد PP می‌گوییم. انواع یادگیری ماشین

یادگیری ماشین را می‌توان به طور کلی به سه دسته تقسیم کرد. البته بعضی از منابع تقسیم‌بندی‌های دیگری را نیز برای یادگیری ماشین تصور کرده‌اند، اما اکثر آن‌ها همین سه قسمت کلی را به عنوان زیرشاخه‌های یادگیری ماشین معرفی می‌کنند. در ادامه به تعریف هر کدام از این دسته‌ها می‌پردازیم. ۱. یادگیری نظارت شده Learning (Supervised

فرض کنید که کامپیوتر یک بچه است و ما ناظر (supervisor) به طور مثال پدر یا مادر او هستیم. ما می‌خواهیم به این کودک یاد بدیم که یک خروس چه شکلیست. برای این کار، ما تعدادی عکس که بعضی از آن‌ها عکس خروس و بعضی حیوانات دیگری هستند را به بچه نشان می‌دهیم. وقتی که عکس خروس را نشان می‌دهیم، جمله‌ی «این خروس است» را گفته و وقتی عکس‌هایی که خروس نیستند را نشان می‌دهیم، جمله‌ی «این خروس نیست» را می‌گوییم. به این ترتیب، بچه‌ی ما یاد خواهد گرفت که عکس‌های خروس را از غیر خروس تشخیص دهد. به این روش یادگیری، یادگیری نظارت شده Learning (Supervised می‌گوییم.

در این نوع از یادگیری، نمونه‌هایی که برای آموزش مدل استفاده می‌شوند، دارای برچسب (Label) هستند. به این معنی که مدل یادگیری ماشین با استفاده از داده‌هایی که از قبل برچسب مشخصی دارند («خروس بودن» و «خروس نبودن» در این مسئله)، الگوهای اساسی را تا زمانی که به عملکرد رضایت‌بخشی برای ما برسند، پیدا می‌کند.

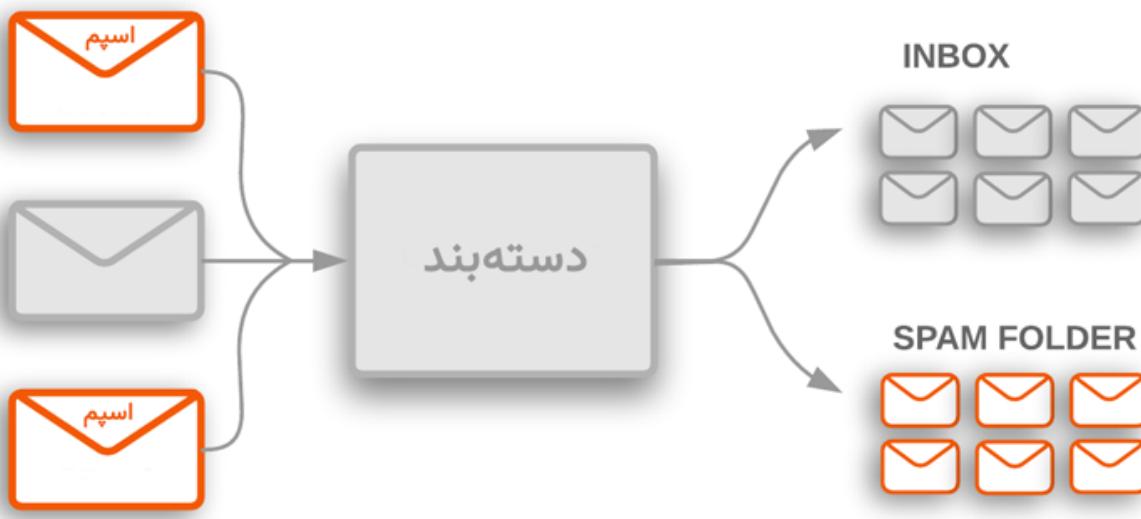
به عنوان مثال، جدول زیر را در نظر داشته باشید. این جدول اطلاعاتی از خانه‌های شهر پکن به ما می‌دهد.

Lng	Lat	constructionTime	elevator	Price
116.69	39.8706	2010	0	82762
116.406	39.9577	2005	1	82762
116.471	39.9011	2005	1	42473
116.337	39.8941	1985	0	41586
116.359	39.9321	1992	1	87161
116.353	40.0502	2012	1	69539
116.296	39.8484	1996	0	19176
116.336	39.9322	1980	0	80059
116.48	39.9115	1985	0	42761
116.576	39.8620	2001	0	21109

هر سطر این جدول، مربوط به یک خانه است و ستون‌های آن، اطلاعاتی از هر خانه را نشان می‌دهند. اگر بخواهیم به کمک یادگیری ماشین، قیمت هر متر مربع خانه‌ها را پیش‌بینی کنیم، ستون Price همان برچسبی است که مدل سعی می‌کند به کمک سایر ستون‌ها، آن را پیش‌بینی کند.

الگوریتم‌های یادگیری نظارت شده را می‌توان به دو بخش دسته‌بندی (Classification) و رگرسیون- (Regression) تقسیم کرد که در ادامه به معرفی بیشتر هر کدام می‌پردازیم. دسته‌بندی (Classification) در دسته‌بندی، هدف ما پیدا کردن دسته (Class) یا برچسب مناسب برای نمونه‌های بدون برچسب است.

برای این کار، ما مدل یادگیری ماشینی خود را با استفاده از نمونه‌های برچسب‌دار، آموزش می‌دهیم. بر اساس این آموزش، مدل ما یاد می‌گیرد که داده‌ها را به دسته‌های مختلف تقسیم کند. به عنوان مثال، دسته‌بندی ایمیل‌ها به دو دسته‌ی اسپم (Spam) و غیر اسپم را در نظر بگیرید. برای این کار، شما مجموعه‌داده‌ای شامل میلیون‌ها متن ایمیل، موضوع ایمیل و دیگر ویژگی‌هایی که ممکن است مهم باشند را جمع‌آوری می‌کنید. سپس، بر اساس اینکه هر ایمیل اسپم بوده است یا خیر، آن‌ها را برچسب می‌زنید. اکنون، با استفاده از یکی از الگوریتم‌های دسته‌بندی، مدلی را روی نمونه‌های برچسب‌دار، آموزش می‌دهید. مدل شما در نهایت می‌تواند یک ایمیل اسپم را از غیر اسپم تشخیص دهد.

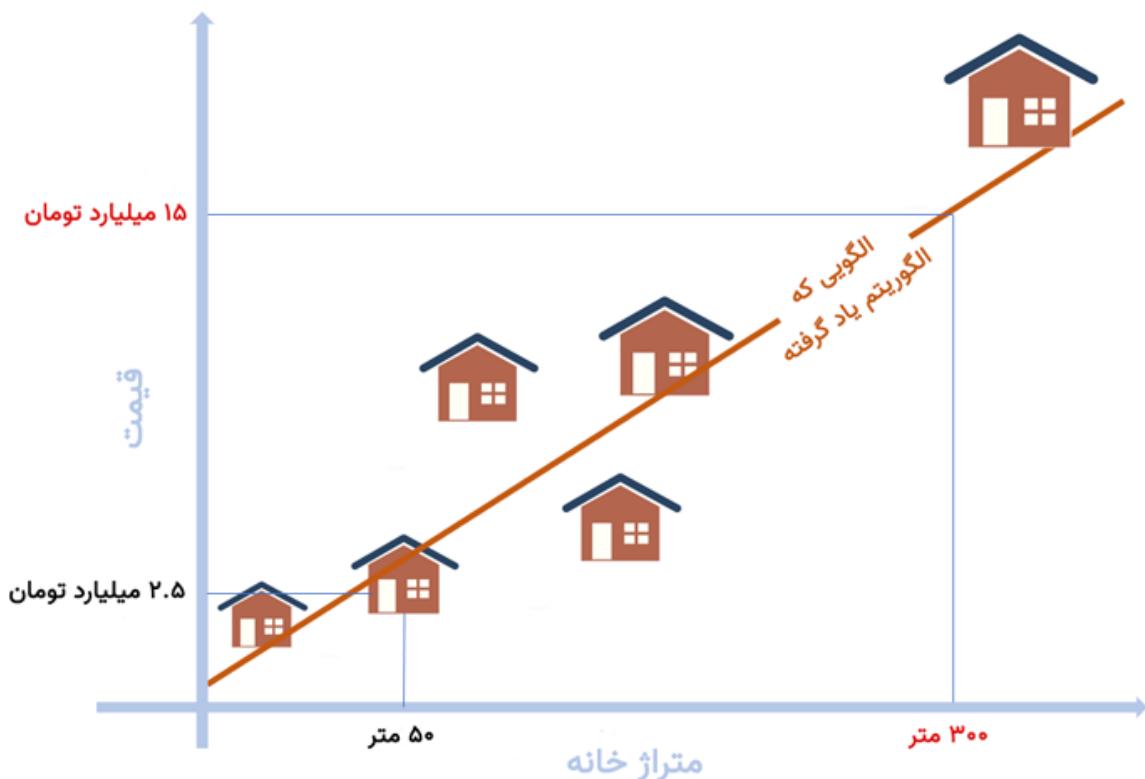


رگرسیون (Regression)

در رگرسیون، هدف ما تخمین مقدار یک ویژگی (این بار مقداری عددی/پیوسته) برای یک نمونه است. این الگوریتم‌ها برای پیش‌بینی روند بازار، قیمت خانه و دیگر برچسب‌های عددی به کار می‌روند.

به طور مثال، برای پیش‌بینی قیمت خانه، می‌توان از اطلاعات خانه‌های دیگر برای تخمین قیمت یک خانه استفاده کرد. ویژگی‌هایی مانند متراز، تعداد اتاق، داشتن یا نداشتن پارکینگ، داشتن یا نداشتن حیاط و دیگر ویژگی‌های تاثیرگذار بر قیمت یک خانه، می‌توانند به عنوان اطلاعات ورودی در نظر گرفته شوند.

در مدل زیر، فقط از متراز خانه‌ها برای ساخت مدل یادگیری ماشین استفاده شده است. هر نقطه یک خانه را نشان می‌دهد. برای مثال، خانه‌ی ۵۰ متری، ۵.۲ میلیارد تومان ارزش دارد. از نظر مدل ما، خانه‌ی ۳۰۰ که قیمت آن مشخص نیست، ۱۵ میلیارد تومان ارزش دارد.



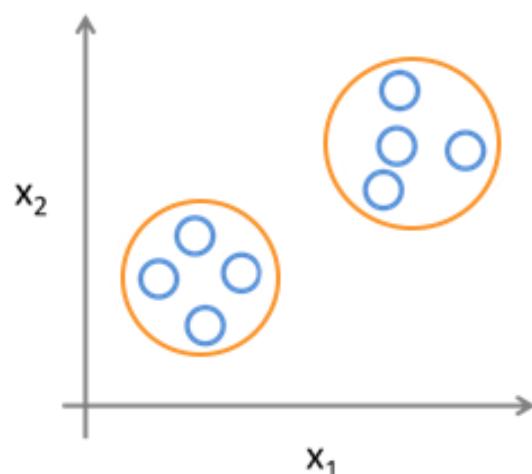
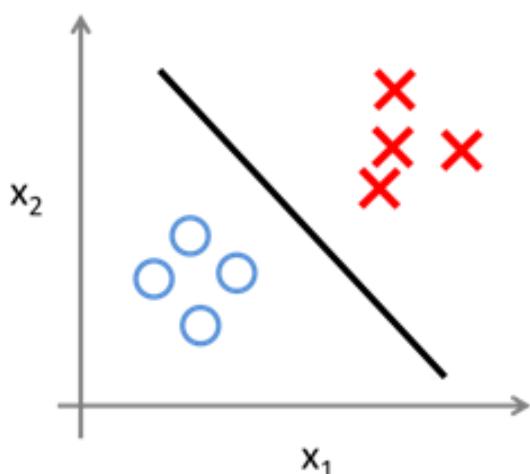
تفاوت دسته‌بندی و رگرسیون

همان‌طور که دیدید در هر دو رویکرد مدل یاد می‌گیرد تا برچسب داده‌ها را پیش‌بینی کند. تفاوت میان این دو، در نوع این برچسب است. در الگوریتم‌های دسته‌بندی برچسبی که می‌خواهیم پیش‌بینی کنیم به صورت متغیری دسته‌ای/گسسته است اما در الگوریتم‌های رگرسیون منغیر هدف از نوع عددی/پیوسته است.

۲. یادگیری ناظارت‌نشده (Unsupervised Learning)

تفاوت یادگیری ناظارت‌نشده (Unsupervised Learning) با یادگیری ناظارت‌شده، در نبودن برچسب‌ها است. به عبارت دیگر، هیچ ناظری (برچسب) به کامپیوتر نمی‌گوید که چه زمانی درست پیش‌بینی کرده و چه زمانی مرتكب خطا شده است. در این رویکرد یادگیری، مدل بهنهایی و بدون کمک برچسب‌هایی که در روش ناظارت‌شده دیدیم، باید الگوها را شناسایی کند.

برای مثال در تصویر پایین سمت راست، نقاط هیچ برچسبی ندارند (از لحاظ ظاهری تفاوتی ندارند) اما فاصله‌ی نقاط از هم است که آن‌ها را متمایز می‌کند و در دسته‌های مجزا قرار می‌دهد. از یادگیری ناظارت‌نشده برای تحلیل اکتشافی و خوشه‌بندی مجموعه‌داده استفاده می‌شود. توجه داشته باشید که بیشتر مجموعه‌داده‌های موجود، بدون برچسب هستند و به این دلیل، این روش‌ها بسیار کاربردی هستند.



در تصویر بالا سمت چپ دو نوع داده وجود دارد، داده‌هایی با علامت ضربدر قرمز و داده‌هایی با دایره‌های آبی. دایره‌ی آبی و ضربدر قرمز همان برجسب‌ها هستند. این تصویر در واقع همان مسئله‌ی دسته‌بندی (نوعی از یادگیری نظارت شده) را نشان می‌دهد و وظیفه‌ی مدل پیدا کردن خط سیاه رنگ است که به کمک آن بتوان دسته‌ی «دایره‌های آبی» را از دسته‌ی «ضربدرهای قرمز» تفکیک کرد. هر نقطه‌ای که سمت راست خط مشکی باشد، متعلق به دسته‌ی «ضربدرهای قرمز» است. هر نقطه‌ای هم که سمت چپ خط قرار بگیرد، متعلق به دسته‌ی «دایره‌های آبی» خواهد بود.

اما در تصویر سمت راست هیچ گونه برچسبی وجود ندارد. در این حالت ماشین تشخیص داده است که مجموعه‌داده‌ی ما، قابل تقسیم به دو دسته است: یکی پایین سمت چپ و دیگری بالا سمت راست صفحه. الگوریتم‌های یادگیری ماشین نظارت نشده، که در آن مدل بدون دخالت انسان و بدون برچسب، الگوهای پنهان را پیدا می‌کند، به سه دسته‌ی زیر تقسیم می‌شود:

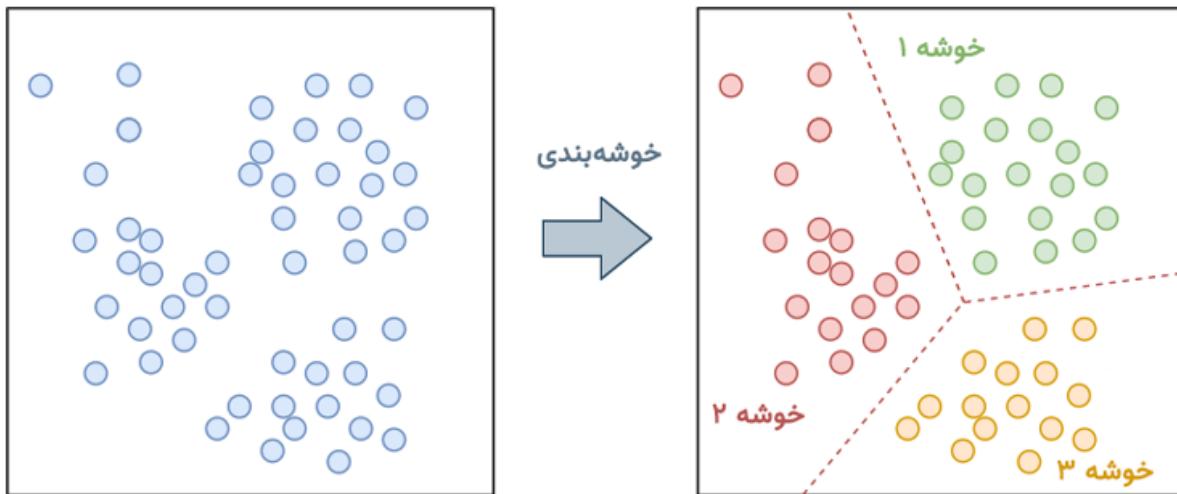
(Clustering)

کاهش ابعاد (Dimensionality Reduction)

استخراج قانون وابستگی (Association Rule Mining)

در این درسنامه، به معروف «خوشه‌بندی» و «کاهش ابعاد» می‌پردازیم و از توضیح «استخراج قانون وابستگی» به دلیل آن که خارج از حوزه‌ی این دوره است، صرف‌نظر می‌کنیم. خوشه‌بندی (Clustering) خوشه‌بندی

خوشه‌بندی به معنی دسته‌بندی خودکار مجموعه داده، به خوشه‌های همگن است؛ به‌گونه‌ای که نمونه‌های هر خوشه، ویژگی‌های یکسانی داشته باشند. اولین گام، انتخاب معیاری برای سنجش فاصله بین نمونه‌ها مانند فاصله اقلیدسی است که یکی از پرکاربردترین معیارهای سنجش فاصله است. به عنوان مثال، عکس پایین نمونه‌ای از خوشه‌بندی است که نمونه‌ها را بر اساس معیار فاصله اقلیدسی به سه دسته گروه‌بندی کرده است. اما لازم است بدانیم که معیارهای مناسب برای فاصله، تنها به فاصله اقلیدسی محدود نمی‌شوند.



کاهش ابعاد (Dimensionality Reduction)

در ساده‌ترین حالت، روش کاهش ابعاد یعنی کاهش دادن تعداد ویژگی‌هایی که از آن‌ها برای آموزش مدل یادگیری ماشین خود استفاده می‌کنیم. به طور مثال، کاهش دادن تعداد ستون‌های یک مجموعه‌داده‌ی جدولی، حالتی از کاهش ابعاد است.

سوالی که مطرح می‌شود این است که «چه نیازی به این کار داریم؟ چرا لازم است تعداد ستون‌های یک مجموعه داده جدولی که مثلًاً ۸۰ ستون دارد را کاهش دهیم؟ چرا به سادگی از تمام این ۸۰ ویژگی برای آموزش مدل خود استفاده نکنیم؟»

در درسنامه‌های فصل یادگیری نظارت‌نشده با معرفی تکنیک‌های مختلف استخراج ویژگی Feature Extraction که زیرشاخه‌ای از کاهش ابعاد شناخته می‌شود پاسخ این پرسش‌ها را بررسی خواهیم کرد. ۳. یادگیری تقویتی Reinforcement Learning

فرض کنید در حال انجام یک بازی معماهی به‌طور مثال بازی هزارتو هستید. هدف شما خارج شدن از هزارتو است و هر بار که قدمی در مسیر خارج شدن از هزارتو بردارید، پاداشی دریافت می‌کنید. همچنین زمانی که در مسیری گام بردارید که شما را به خارج از هزارتو هدایت نکند، از امتیاز شما کم می‌شود (مجازات می‌شود). در این بازی ممکن است تا زمانی که بتوانید از هزارتو خارج شوید به دفعات به بنبست برسید. زمانی که قدم‌های درستی بردارید، با گرفتن امتیاز متوجه خواهید شد که در مسیر درست قرار دارید و با سعی در ادامه‌ی این مسیر می‌توانید از هزارتو خارج شوید.

روندی که در مثال بالا در پیش گرفتید در واقع همان رویکرد یادگیری تقویتی Reinforcement Learning است. یادگیری تقویتی با ذهنیت آزمون و خطاكار می‌کند. عامل هوشمند Agent (Agent) طبق حالت جاری State، حرکتی Action را انجام می‌دهد و بر اساس آن حرکت بازخورد Reward (Reward) دریافت می‌کند؛ این بازخورد ممکن است مثبت یا منفی (پاداش یا تنبیه) باشد و متناسب با این بازخورد خطمنشی Policy خود را تغییر دهد.

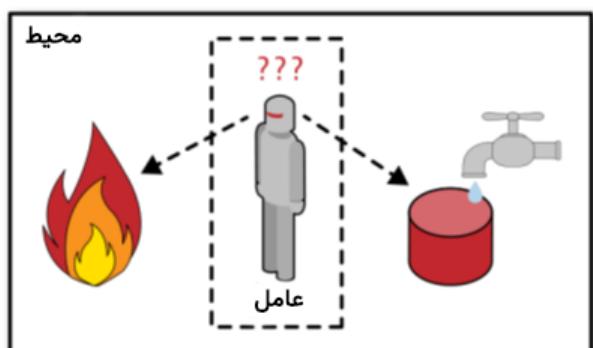
توضیح اضافه

در مثال هزارتو:

عامل هوشمند Agent (Agent) شما هستید که سعی می‌کنید از هزارتو خارج شوید. حالت جاری State (State) مختصات

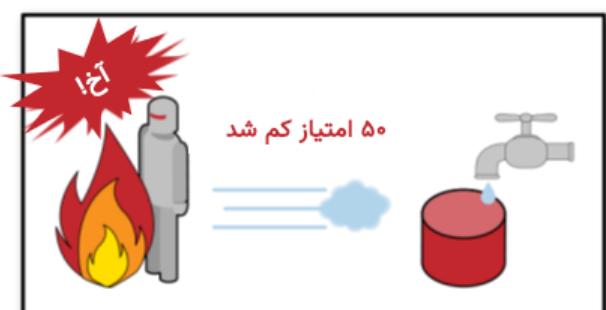
مکان فعلی شما در هزارتو و اطلاعات بیشتری در مورد محیطی که در آن به یادگیری می‌پردازید را نشان می‌دهد. حرکت یا عمل (Action) گامی است که در یک جهت بر می‌دارید. بازخورد (Reward) امتیاز مثبت یا منفی‌ای است که دریافت می‌کنید تا بفهمید آیا در مسیر درستی قرار دارید یا نه! خطمشی (Policy) مشخص می‌کند که در هر حالت چه عملی را انتخاب کنید تا بهترین پاداش را بگیرید.

شکل زیر نشان می‌دهد که یک ربات چگونه یاد می‌گیرد که به آتش نزدیک نشود. (برگرفته از کتاب Hands TensorFlow and Keras Scikit-Learn, with Learning Machine On) حالت به یادگیری انسان است. یک کودک برای آموختن چگونه راه رفتن، مدام تلاش می‌کند و پدر و مادرش با تشویق کردن سعی می‌کنند به او در این یادگیری کمک کنند. برای حرف زدن هم انسان‌ها فرآیند مشابهی را طی می‌کنند. یادگیری تقویتی برخلاف یادگیری نظارت‌شده وابسته به داده نیست، بلکه به واسطه‌ی تعامل با محیط می‌آموزد.



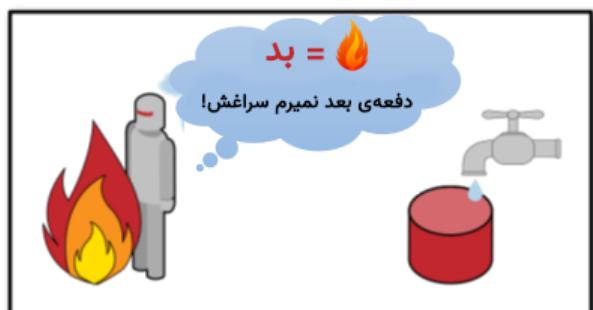
۱. مشاهده

۲. انتخاب حرکت با استفاده از خطمشی



۳. انجام حرکت!

۴. دریافت پاداش یا جزا



۵. به روزرسانی خطمشی (مرحله‌ی یادگیری)

۶. تکرار تا وقتی که خطمشی بهینه پیدا شود

این روش از یادگیری، نزدیک‌ترین حالت به یادگیری انسان است. یک کودک برای آموختن چگونه راه رفتن، مدام تلاش می‌کند و پدر و مادرش با تشویق کردن سعی می‌کنند به او در این یادگیری کمک کنند. برای حرف زدن هم انسان‌ها فرآیند مشابهی را طی می‌کنند. یادگیری تقویتی برخلاف یادگیری نظارت‌شده و یادگیری نظارت‌شده وابسته به داده نیست، بلکه به واسطه‌ی تعامل با محیط می‌آموزد.

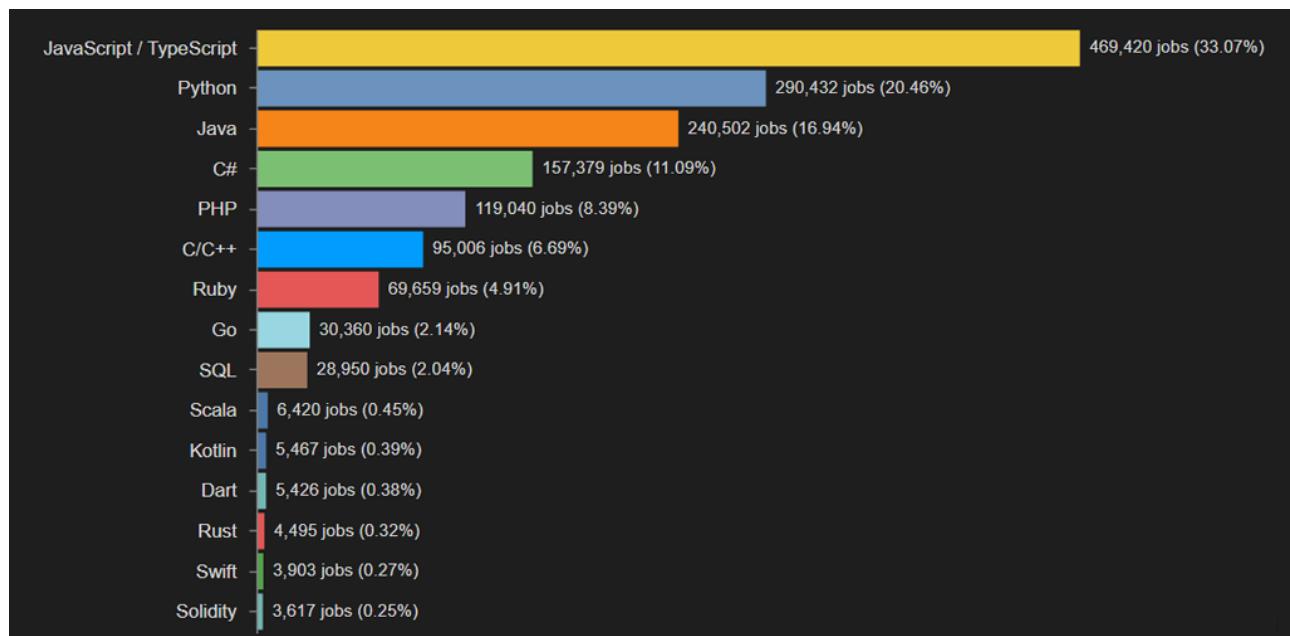
۵.۱ چرا پایتون؟

چرا پایتون؟

همان‌گونه که از اسم دوره انتظار می‌رود، زبان برنامه‌نویسی مورد استفاده‌ی ما، پایتون (Python) است. ممکن است تا به حال دوره‌های دیگر یا آگهی‌های شغلی زیادی دیده باشید که همگی از زبان برنامه‌نویسی پایتون استفاده می‌کردند. سوالی که اینجا مطرح می‌شود این است که دلیل محبوبیت این زبان چیست؟

به‌طور کلی، دلایل انتخاب پایتون در این دوره را می‌توان به چهار دسته تقسیم کرد. زبان غالب در صنعت در اکثر شرکت‌هایی که با داده سر و کار دارند، زبان برنامه‌نویسی پایتون در حال استفاده است. چون می‌خواهیم دوره‌ی یادگیری ماشین، کاربردی باشد، نیاز است زبانی را انتخاب کنیم که شرکت‌های زیادتری متقاضی آن هستند. البته زبان‌های برنامه‌نویسی دیگر مانند R MATLAB و Julia نیز در صنعت داده کاربرد خود را دارند، اما شاید بتوان گفت پایتون اولین انتخاب بسیاری از افراد است.

طبق گزارش تهیه شده توسط DevJobsScanner در سال ۲۰۲۲ که با تحلیل بیش از ۷ میلیون آگهی شغلی توسعه‌دهندگان به دست آمده، زبان پایتون توانسته رتبه‌ی دوم را کسب کند که خود نشان از اهمیت آن در صنعت دارد.



کتابخانه‌های متن‌باز فراوان

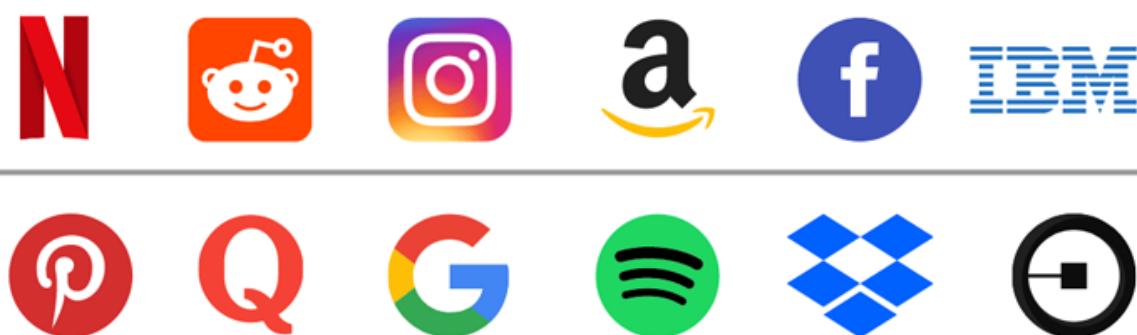
در پایتون کتابخانه‌های زیادی برای بارگذاری و دستکاری داده، مدل‌سازی، فعالیت‌های آماری، پردازش تصویر و پردازش متن وجود دارند. هر کدام از این کتابخانه‌ها، توابع زیادی در اختیار توسعه‌دهندگان قرار می‌دهند تا بتوانند به راحتی و در حالت بهینه، برنامه خود را توسعه دهند. به عنوان مثال Numpy برای کار با آرایه‌ها و اعمال ریاضی، pandas برای ذخیره‌سازی داده‌های جدولی و دستکاری داده، OpenCV برای کار با تصاویر و scikit-learn برای یادگیری ماشین در پایتون مورد استفاده قرار می‌گیرند.



تعامل با کد و داده

در علوم داده و یادگیری ماشین، تعامل با کد و توسعه تعاملی بسیار مرسوم و معمول است. در پایتون با یک تغییر جزئی در کد می‌توانیم از برنامه اجرا بگیریم، نتیجه را مشاهده و تحلیل کنیم و مجددًا کد را تغییر دهیم. این چرخه آنقدر تکرار می‌شود تا به نتیجه‌ی مطلوب برسیم. با کمک ترمینال یا ابزارهایی مانند جوپیتر، تعامل با کد در پایتون بسیار راحت خواهد بود. کاربردهای دیگر پایتون

پایتون یک زبان برنامه‌نویسی همه‌منظوره است. بنابراین کتابخانه‌های قدرتمندی برای ساخت واسط کاربری، کار با وب و دیگر کاربردها دارد. تصور کنید تنها با یادگیری یک زبان برنامه‌نویسی، می‌توانید یادگیری ماشین انجام داده و مدل را به کمک چارچوب‌هایی مانند جنگو، در وب قابل دسترس کرده و یا برای آن، واسط کاربری طراحی کنید! برخی از کمپانی‌های شناخته‌شده‌ای که از زبان پایتون در سرویس‌هایشان استفاده می‌کنند در تصویر زیر آمده است.



این دلایل تنها بخشی از علل هستند که باعث می‌شوند اکثر دانشمندان داده یا مهندسان یادگیری ماشین، پایتون را به عنوان زبان اصلی خود انتخاب کنند.

۶.۱ آماده‌سازی محیط کار

آماده‌سازی محیط کار

اگر تصمیم دارید تا بتوانید کدها را بر روی سیستم خود اجرا کنید این درسنامه راهاندازی محیط مورد نیاز به کمک شما می‌آید. با آنکه راهاندازی محیط توسعه (Development Environment) می‌تواند با توجه به مهارت‌ها و سلیقه‌های هر فرد به شکل متفاوتی صورت گیرد اما ما قصد داریم یک راه آسان و اصولی را پیش پای شما بگذاریم. در این درسنامه ابتدا با نحوه نصب پایتون و conda آشنا خواهید شد، سپس یک محیط مجازی خواهید ساخت و پکیج‌های پیش‌نیاز این دوره را بر روی محیط ساخته‌شده نصب خواهید کرد. در درسنامه‌های بعد نیز با نحوه راهاندازی یک محیط کدنویسی مناسب و حرفه‌ای آشنا خواهید شد.

۱. نصب پایتون

نسخه‌ی پیشنهادی ما برای این دوره، پایتون ۱۲.۳ است.

۷.۱ اجرای نوتبوک‌ها

[متن شما]

۸.۱ گوگل کولب

[متن شما]

۹.۱ معرفی مجموعه‌داده

[متن شما]

فصل ۲

مدیریت پروژه

۱.۲ اهداف فصل

اهداف فصل



فرض کنید تصمیم گرفته‌اید با دوستان تان یک سفر دسته‌جمعی آخر هفته داشته باشید. همه هیجان‌زده‌اند، اما هیچ‌کس دقیق نمی‌داند کجا می‌روید، کی حرکت می‌کنید، چه کسی ماشین می‌آورد و اصلاً چه چیزی باید با خود بیاورید! تنها چیزی که مشخص است، «هیجان سفر» است. صبح جمعه، عده‌ای خواب مانده‌اند، یکی یادش رفته بنزین بزند، و آن یکی وسط راه متوجه می‌شود مدارک ماشینش را جا گذاشته است. در آخر این سفر بهجای خوش‌گذرانی تبدیل می‌شود به یک خاطره‌ی تلخ و خنده‌دار.

مدیریت پروژه هم درست همین طور است. اگر بدون برنامه ریزی، تعیین اولویت‌ها، تقسیم وظایف و فهمیدن دلایل شکست‌های قبلی وارد پروژه‌ای شویم، احتمال اینکه پروژه ما هم به سرنوشت آن سفر دچار شود، بسیار زیاد است.

در این فصل یاد می‌گیریم که چطور مثل یک رهبر حرفه‌ای پروژه را از ابتدا تا انتها مدیریت کنیم. در ادامه، خلاصه‌ای از مطالعی که در این فصل راجع به آن بحث می‌شود آورده شده است.

آشنایی با مفهوم چرخه پروژه و مراحل اصلی آن
بررسی روش‌های اولویت‌بندی وظایف و منابع در پروژه‌ها
آشنایی با اصول سازماندهی و مدیریت تیم پروژه
شناخت دلایل رایج شکست پروژه‌ها و راهکارهای پیشگیری از آن‌ها

۲.۲ چرخه پروژه

چرخه پروژه

هر پروژه‌ای (چه یادگیری ماشین باشد یا نباشد) برای پاسخ دادن به یک یا چند نیاز تعریف می‌شود. برای پاسخ‌دهی به این نیازها، هدفی مشخص می‌شود و برای دستیابی به آن هدف، پروژه شروع می‌شود. پس می‌توان اولین گام در هر پروژه‌ای را تعریف هدف آن در نظر گرفت.

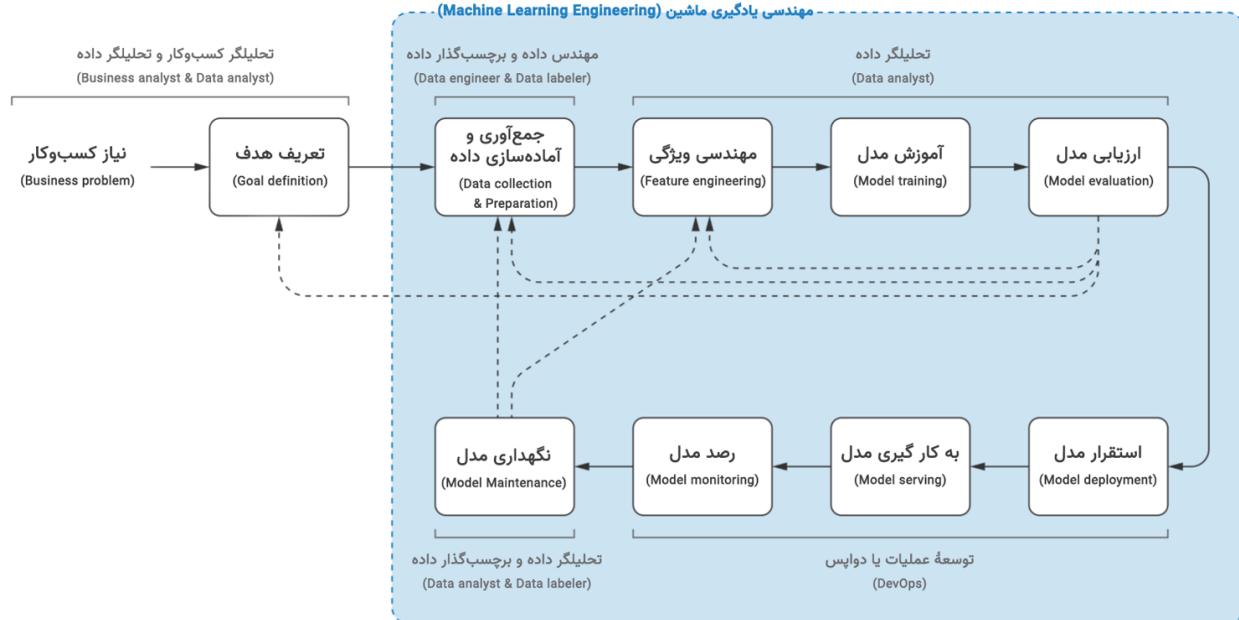
پروژه‌های یادگیری ماشین به خاطر یک نیاز کسب‌وکار شروع می‌شوند و این نیاز، هدف پروژه را تعریف می‌کند. متناسب با این هدف، تیم سازماندهی می‌شود و کار پروژه شروع می‌شود.

هدف تعریف شده پروژه، بایستی دارای ویژگی‌های زیر باشد:

میزان پیشرفت پروژه و معیار اتمام آن مشخص باشد. ورودی و خروجی مسئله واضح باشند.

توجه داشته باشید که لزوماً هدف یک پروژه‌ی یادگیری ماشین، همان هدف کسب‌وکار نیست. بلکه هدف کلی را به چند هدف کوچک‌تر می‌شکنیم و برای رسیدن به هر هدف پروژه‌ی مناسبی تعریف می‌کنیم. به عنوان مثال، کوئرا می‌خواهد رضایت کاربران از سامانه‌ی استخدام برنامه‌نویسان را افزایش دهد. یکی از پروژه‌های یادگیری ماشین که در رسیدن به این هدف به کوئرا کمک می‌کند، پیشنهاد آگهی‌های شغلی بر اساس رزومه به توسعه‌دهندگان است. یک پروژه دیگر، پیشنهاد توسعه‌دهنده‌ها با رزومه‌ی بهتر به شرکت‌هاست. چندین پروژه دیگر مانند دو مثالی که آورده شدند، تعریف و پیاده‌سازی شده‌اند تا هدف کسب‌وکار یعنی «افزایش رضایت کاربران» برآورده شوند.

به صورت کلی، یک پروژه‌ی یادگیری ماشین از ۹ گام زیر، تشکیل می‌شود. شکل زیر، مراحل مختلف یک پروژه یادگیری ماشین را نشان می‌دهد:



در این دوره، ما مراحل مختلف انجام یک پروژه را با هم می‌آموزیم. هنگام انجام تمرین‌ها و پروژه فرض می‌کنیم که مرحله‌ی اول یعنی تعیین هدف، از پیش مشخص شده و در اختیارمان قرار گرفته است. لیست زیر به صورت خلاصه، هر گام را توضیح می‌دهد. توضیحات مفصل‌تر را در هر فصل و درسنامه‌های آن مطالعه خواهید کرد. جمع‌آوری و پاک‌سازی (آماده‌سازی) داده‌ها

مهم‌ترین رکن در یادگیری ماشین، داده (data) است. در این مرحله، داده‌ی مورد نیاز مسئله را از منابع موجود، استخراج می‌کنیم. داده‌های خام ممکن است نیاز به پاک‌سازی داشته باشند؛ به عنوان مثال، جنس ستون‌ها نیاز به تغییر داشته باشند یا مقادیر گم شده، بایستی که پر شوند. مهندسی ویژگی

پس از پاک‌سازی داده، داده‌ی خام به مجموعه‌ای تبدیل شده که قادر خطا و مقادیر ناموجود است. اما همین کافی نیست. می‌توانیم با داشتن دانش زمینه‌ای از کسب‌وکار، مشاهدات و آزمون‌های آماری به مهندسی ویژگی (Feature Engineering) پرداخته ستون‌هایی را حذف یا ستون‌های جدیدی تولید کنیم تا مجموعه‌داده برای مرحله‌ی مدل‌سازی آماده شود. مدل‌سازی

با انتخاب الگوریتم‌های مناسب، می‌توانیم یک مدل (Model) یادگیری ماشین برای محاسبه‌ی رابطه بین متغیرهای مستقل (ویژگی‌ها) و متغیر وابسته (هدف) بسازیم. ارزیابی عملکرد مدل

وقت آن رسیده تا عملکرد مدل، مورد ارزیابی (Evaluation) قرار بگیرد. انتخاب معیار ارزیابی مناسب، مهم‌ترین تصمیمی است که بایستی در این مرحله گرفته شود. اگر که مدل نتواند امتیاز قابل قبولی از این معیار کسب کند، مهندس یادگیری ماشین بایستی با شناسایی علت، به دنبال راهکاری برای ساختن مدلی با عملکرد بهتر باشد. استقرار و به کار گیری مدل

پس از آن که مدل ساختیم که حداقل عملکرد قابل قبول از لحاظ معیار ارزیابی را دارا بود، باید آن را برای استفاده‌ی عملی، مستقر (Deploy) کرد. منظور از مستقر کردن، قرار گرفتن در بستری است که کاربران بتوانند از آن استفاده کنند. در نهایت، باید درخواست‌های کاربران توسط کانالی مانند واسطه کاربری، دریافت و به ورودی مورد استفاده مدل تبدیل شوند تا خروجی آن، برای کاربر برگردانده شود. رصد و نگهداری

منظور از رصد، پیگیری عملکرد مدل در طول زمان می‌باشد تا مطمئن شویم که مدل همچنان معتبر و دارای عملکرد قابل قبولی است. در صورتی که مدل، حداقل عملکرد قابل قبول را نداشته باشد، نیاز است که اصلاحاتی

مانند آموزش روی نمونه‌های جدید، روی مدل صورت بگیرد؛ به مجموعه کارهایی که به منظور حفظ کیفیت مدل انجام می‌شود، نگهداری (Maintenance) می‌گوییم.

۳.۲ اولویت‌بندی

اولویت‌بندی

هنگامی که به عنوان مهندس یادگیری ماشین، در یک شرکت یا سازمان مشغول به کار می‌شوید، احتمالاً ایده‌های مختلفی جهت استفاده از قدرت یادگیری ماشین، با شما به اشتراک گذاشته می‌شود و شما باستی که آنها را بر اساس یک اولویت‌بندی، انجام دهید. اما این اولویت‌بندی چگونه انجام می‌شود؟ این امر، می‌تواند تبدیل به یک تصمیم سلیقه‌ای شود.

در این درسنامه چارچوبی را معرفی خواهیم کرد که به کمک آن بتوانید به صورت ساختارمند، در اینباره تصمیم بگیرید. جهت اولویت‌بندی نیاز است برای هر ایده، دو مورد «ارزش» (Value) و «امکان‌پذیری» (Feasibility) آن را حساب کنید. در ادامه به معرفی کامل هرکدام از این اصطلاحات می‌پردازیم. ارزش (Value) استفاده از یادگیری ماشین در یک پروژه، هنگامی دارای «ارزش» زیادی است که: ۱) یادگیری ماشین بتواند جایگزین بخش پیچیده‌ای از پروژه شود.

به عنوان مثال، بخش پیچیده‌ای از یک سامانه‌ی موجود، می‌تواند مبتنی بر قواعد یا به اصطلاح Rule-based if-else باشد. در نتیجه، شما با قطعه کدی روبرو می‌شوید که دارای تعداد زیادی دستور مانند به صورت تودرتو، پیچیده و با استثنایات مختلفی هستند. نگهداری و توسعه‌ی چنین کدی در طول زمان می‌تواند دشوار، زمانبر و مستعد خطا باشد. همچنین این بخش سیستم، می‌تواند به قسمتی تبدیل شود که مهندسین نرمافزار از آن فراری باشند. آیا می‌شود که این دستورات، به جای آن که برنامه‌نویسی شوند، یاد گرفته شوند؟ (اگر جواب این سوال، بله است؛ استفاده از یادگیری ماشین، می‌تواند ارزشمند باشد.) ۲) دستیابی به پیش‌بینی‌های ارزان ولی احتمالاً با کمی خطأ، دارای مزیت باشد.

به عنوان مثال، در سیستمی که تعداد زیادی درخواست (request) از کاربران دریافت می‌کند، فرض کنید که تعداد زیادی از این درخواست‌ها، «آسان» هستند و می‌توانند توسط اتوماسیون به سرعت انجام شوند. در نتیجه، فقط لازم است که درخواست‌های باقی‌مانده که «سخت» طبقه‌بندی می‌شوند، به صورت دستی مورد بررسی قرار بگیرند.

یک سیستم مبتنی بر یادگیری ماشین که درخواست‌های «آسان» را از «سخت» تشخیص می‌دهد، می‌تواند زمان زیادی را از وقت نیروی انسانی صرفه‌جویی کند. زیرا که نیروی انسانی فقط لازم است روی درخواست‌هایی که نیاز به دخالت دستی دارد، تمرکز کند. همچنین اگر که درخواست سختی به اشتباه آسان پیش‌بینی شود، اتوماسیون در پردازش آنها، دچار خطأ می‌شود. در نتیجه، چنین درخواست‌هایی می‌توانند توسط نیروی انسانی مورد ارزیابی مجدد قرار بگیرد. از طرف دیگر، اگر که یک نیروی انسانی، به اشتباه درخواست آسان دریافت کند، هیچ مشکلی وجود ندارد. چون او می‌تواند تشخیص دهد که این درخواست آسان است و آن را به اتوماسیون جهت پردازش ارسال کند. ۳) آن ایده، بتواند ارزش مالی قابل توجهی برای سازمان بیاورد.

ارزش مالی می‌تواند به صورت کاهش هزینه‌ها (حقوق نیروی انسانی) و افزایش درآمد (میزان فروش) مورد حساب قرار گیرد. همچنین مواردی مانند افزایش رضایت مشتریان نیز می‌توانند به عنوان ارزش مالی لحاظ گردند. امکان‌پذیری (Feasibility)

امکان‌پذیری یک پروژه یادگیری ماشین، می‌تواند توسط ۳ عامل زیر، مشخص شود: ۱) سختی مسئله آیا یک الگوریتم پیاده‌سازی شده و یا یک کتابخانه‌ی نرم‌افزاری برای حل این مسئله موجود است؟ اگر بله، تا حد خوبی مسئله آسان می‌شود. آیا نیاز به زیرساخت‌های محاسباتی خاصی برای ساختن مدل و یا استفاده از آن در عمل (production) است؟ اگر خیر، مسئله تا حد خوبی آسان می‌شود.

همچنین برای تخمين سختی یک مسئله، تعدادی مجھول وجود دارد که اگر در پروژه‌های مشابه کار کرده باشید و یا نتایج مشابهی را مطالعه کرده باشید، می‌توانید آن‌ها را حدس بزنید. این مجھولات عبارتند از:

آیا حداقل عملکرد مطلوب در عمل قابل دستیابی است؟ برای دسترسی به عملکرد مورد نظر، چه میزان داده مورد نیاز است؟ چه ویژگی‌هایی و چندتا از آن‌ها لازم است تا بتوانیم یک مدل قابل انتکای یادگیری ماشین برای استفاده در عمل بسازیم؟ الگوریتم استفاده شده برای آموزش مدل باید چقدر بزرگ باشد؟ آموزش مدل و زمان پیش‌بینی کردن آن، چقدر طول می‌کشد؟ چقدر زمان و چند مرحله آموزش لازم است تا که مدل شما به حداقل دقت مطلوب برسد؟

ساده‌سازی مسئله‌ی اصلی نیز یک روش حدس زدن است. برای مثال، فرض کنید که در مجموعه‌دادهی خانه‌های شهر پکن، شما می‌خواهید قیمت یک سری خانه را پیش‌بینی کنید. به منظور ساده‌سازی این مسئله، می‌توانید ابتدا قیمت‌های خانه‌ها را به گروه‌هایی مانند گران، متوسط و ارزان دسته‌بندی کنید. بدین صورت، صورت مسئله‌ی شما به دسته‌بندی قیمت خانه‌ها تبدیل می‌شود و این مسئله حتی برای یک انسان نیز ساده‌تر است. در نتیجه، مسئله‌ی ساده‌تری برای حل توسط یادگیری ماشین داریم.

در صورتی‌که یادگیری ماشین بتواند این مسئله را حل کند، می‌توان امید داشت که بتواند مسئله‌ی اصلی که سخت‌تر است را نیز حل کند. همچنین اطلاعاتی که در هنگام حل مسئله‌ی ساده‌تر کسب می‌کنیم به ما کمک می‌کند که تخمين بهتری از حل مسئله‌ی اصلی پیدا کنیم. توجه داشته باشید که نمی‌توانید به سادگی فقط با یک ضرب، تخمين مورد نظر را روی مسئله‌ی اصلی کسب کنید ولی معمولاً افزایش تعداد گروه‌های مورد نیاز برای پیش‌بینی در یک مسئله، نیازمند میزان بیشتری نمونه برای مجموعه‌دادهی آموزش خواهد بود و این رابطه خطی نیست!

همچنین، شما می‌توانید با استفاده از تعاریف موجود در داده‌ها، مسئله اصلی را به مسائل کوچک‌تر و ساده‌تر تبدیل کنید. فرض کنید که در مجموعه‌دادهی خانه‌های شهر پکن، به جای آن‌که یک مدل برای قیمت‌گذاری تمامی املاک مناطق مختلف داشته باشید، برای هر منطقه، یک مدل مجزا بسازید چون که به صورت طبیعی انتظار می‌رود که مثلاً بازار املاک شمال شهر با بازار املاک جنوب شهر، رفتار متفاوتی داشته باشند. ۲) هزینه دسترسی به داده

به دست آوردن مجموعه‌دادهی مناسب و در مقدار مناسب می‌تواند بسیار هزینه‌بر باشد. به خصوص وقتی‌که آن داده‌ها، نیاز به برچسب زدن دستی داشته باشند. در نتیجه، نیاز است که ما به سوالات زیر در مورد هزینه‌ی دسترسی به داده‌ها پاسخ دهیم:

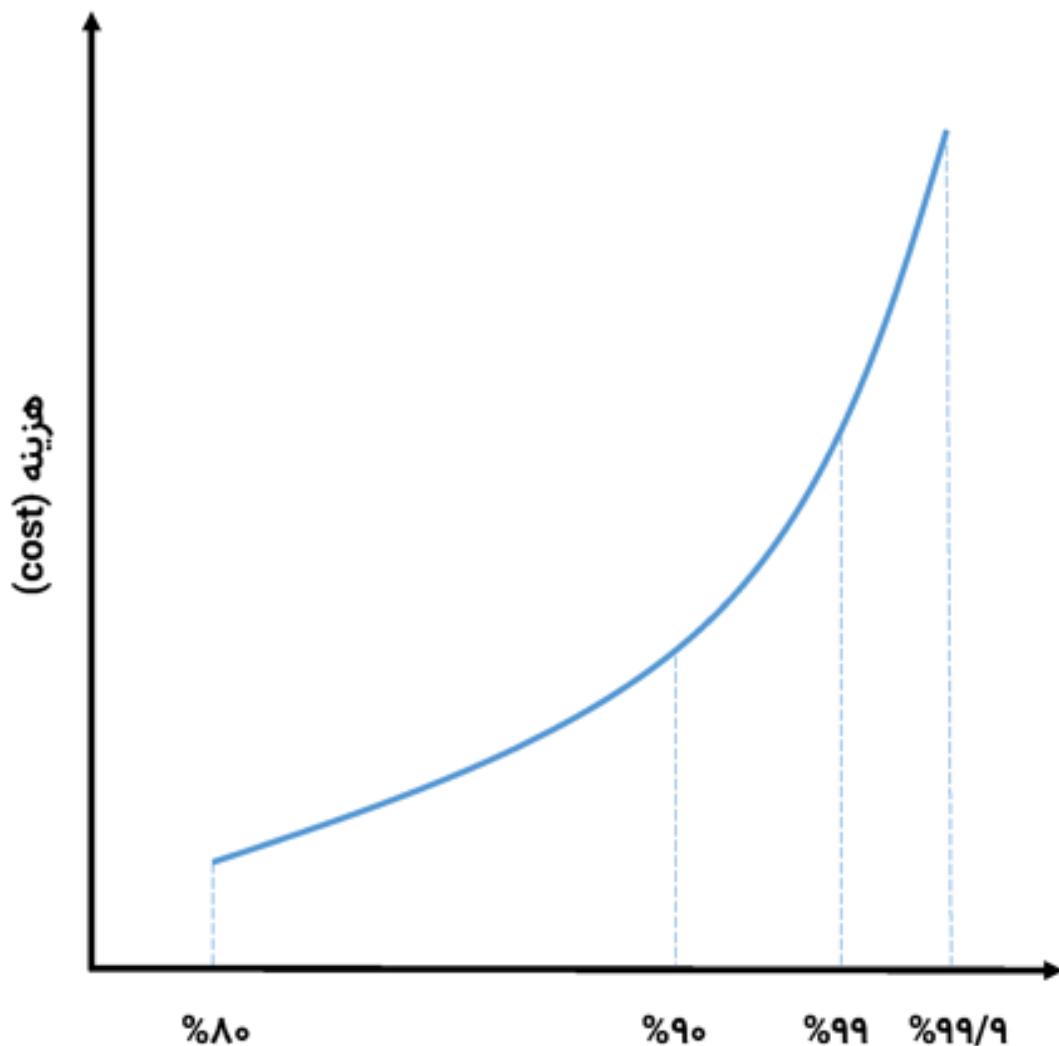
آیا می‌توان مجموعه‌دادهی مورد نیاز را به صورت خودکار تولید کرد؟ اگر بله، دسترسی به داده‌ها تا حد زیادی، «садه» می‌شود. هزینه‌ی برچسب‌زدن دستی به مجموعه‌داده‌ای بدون برچسب چقدر است؟ چه میزان داده مورد نیاز است؟ (ممکن است این عدد را از قبل از شروع پروژه دانست، اما می‌توان بر اساس نتایج مشابه منتشر شده و یا تجربیات قبلی، آن را تخمين زد.)

(۳) حداقل عملکرد مطلوب

داشتن یک سیستم یادگیری ماشین با عملکرد بالا، می‌تواند به معنی نیاز به داده‌های بیشتر یا استفاده

از مدل‌های پیچیده‌ای باشد که از منظر سخت‌افزاری یا طراحی هزینه‌ی بالایی دارند. به همین دلیل، در محاسبه‌ی امکان‌پذیری، بایستی که حداقل عملکرد مطلوب سیستم یادگیری ماشین را لحاظ کنیم.

طبق تصویر زیر، می‌توان گفت که هزینه‌ی یک پروژه یادگیری ماشین با عملکرد مطلوب آن، یک رابطه‌ی نمایی دارد. به عبارت دیگر، هرچه که حداقل عملکرد مطلوب بیشتری نیاز داشته باشیم، امکان‌پذیری پیاده‌سازی چنین سیستمی، کاهش پیدا کرده و سخت‌تر می‌شود.



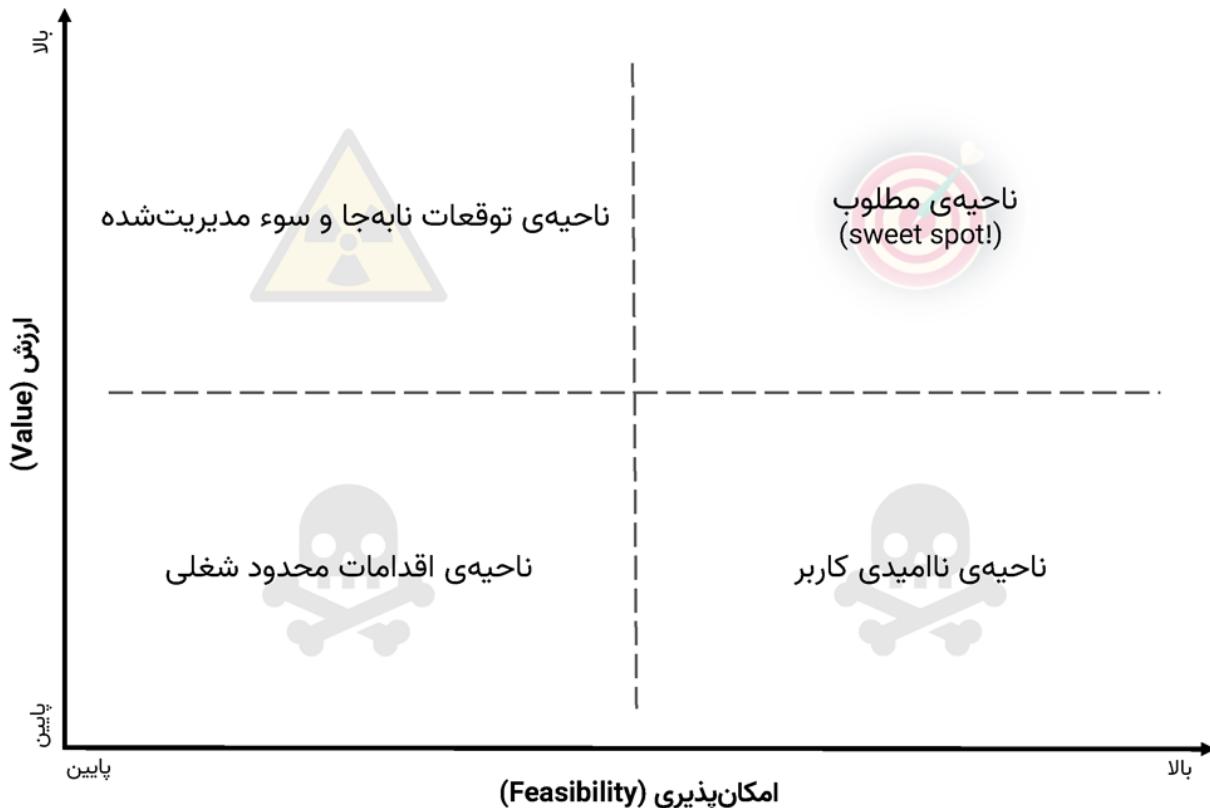
همچنین باید در نظر داشته باشیم که پیشرفت یک پروژه‌ی یادگیری ماشین، غیرخطی است. در هفته‌های ابتدایی، سرعت افزایش عملکرد مدل بالا است ولی بعد از آن، بهبود عملکرد خیلی با کندی سرعت می‌گیرد. به همین دلیل، شما بایستی که قبل از شروع پروژه، این مورد را در نظر داشته باشید و بتوانید انتظارات مدیر محصول را مدیریت کنید.

با توجه به این که یک سیستم یادگیری ماشین در عمل مورد استفاده قرار خواهد گرفت، اگر که دارای عملکرد پایینی باشد می‌تواند ما را دچار ضرر و زیان کند. بنابراین بایستی که عملکرد مطلوب را بر اساس نکات زیر محاسبه کنیم:

هر پیش‌بینی اشتباه چقدر هزینه‌بر است؟ پایین‌ترین سطح عملکرد که زیر آن، پیش‌بینی‌های مدل غیرکاربردی می‌شوند، چقدر است؟

اولویت‌بندی

بعد از آن که برای هر ایده، ارزش و امکان‌پذیری آن را حساب کردید. ایده‌ها را همانند شکل زیر روی نمودار دو بعدی قرار دهید:



در این نمودار، محور افقی، نشان‌دهنده میزان امکان‌پذیری پروژه است و محور عمودی، میزان ارزش آن را نشان می‌دهد. بعد از این که ایده‌های مختلف به صورت نسبی از هم‌دیگر در این نمودار دو بعدی قرار داده شدند، اولویت شروع با ایده‌هایی هست که هم ارزش بالایی برای شرکت/سازمان بیاورند و هم امکان‌پذیری بیشتری از بقیه داشته باشند.

۴.۲ سازماندهی تیم

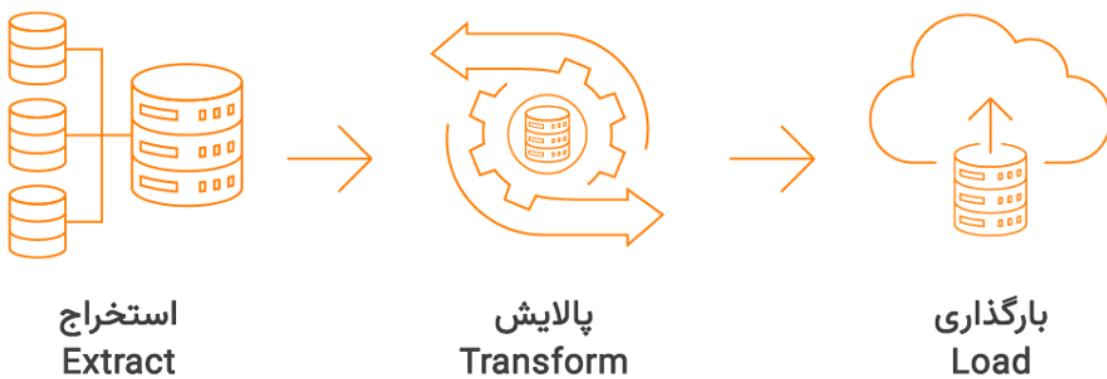
سازماندهی تیم

دو دیدگاه متفاوت در سازماندهی تیم‌های یادگیری ماشین وجود دارد. در دیدگاه نخست تیم یادگیری ماشین می‌تواند متشكل از متخصصین با مهارت‌های مختلف (مثلًا تحلیلگر داده، مهندس نرم‌افزار و غیره) باشد که در کنار هم‌دیگر کار می‌کنند و صرفاً زبان هم‌دیگر را می‌فهمند. در دیدگاه دوم تمام افراد تیم باید ترکیبی از مهارت‌های مختلف را دارا باشند. معمولاً در شرکت‌های بزرگی همچون گوگل که افرادی با تخصص‌های از مخصوص جذب می‌کنند دیدگاه نخست در پیش گرفته می‌شود و دیدگاه دوم بیشتر مورد استفاده‌ی شرکت‌های کوچک و استارت‌آپ‌ها قرار می‌گیرد. در ادامه به شرح کامل‌تر هر کدام از این رویکردها خواهیم پرداخت. (۱) دیدگاه تخصص‌محور

در این دیدگاه، یک تیم یادگیری ماشین از افرادی تشکیل می‌شود که هر کدام وظیفه‌ی مشخصی دارند و فقط در وظیفه‌ی خود بسیار ماهر هستند. این تیم بسته به اندازه و حجم کار یک شرکت، می‌تواند از مهندسان نرم‌افزار، تحلیلگران داده، مهندسان داده، دانشمندان داده و مهندسان یادگیری ماشین، تشکیل شود.

به عنوان مثال، تیم یادگیری ماشین در یک کارگزاری قیمت مسکن را تصور کنید که در حال توسعه پروژه‌ای جهت تخمین قیمت خانه‌هاست. برای این پروژه، مهندسان داده، داده‌های خام را از پایگاه‌های داده استخراج کرده و تغییراتی روی آن اعمال می‌کنند تا آماده‌ی استفاده برای تحلیلگران داده شود.

فرآیند ETL چیست؟



در سازمان‌ها جهت جمع‌آوری، ترکیب و سنتز داده‌های خام از منابع مختلف معمولاً از فرآیندی به نام ETL استفاده می‌شود که هر حرف آن کوتاه‌شده کلمات زیر است:

استخراج (Extract): در ابتدا داده‌ها از منابع مختلفی استخراج و جمع‌آوری می‌شوند.

پالایش (Transform): در این مرحله داده‌ها مطابق با کسبوکار مربوطه به روش‌های مختلفی (همچون پاکسازی داده، مرتب‌سازی، فیلتر، حذف داده‌های تکراری و غیره) پالایش می‌شوند.

بارگذاری (Load): در نهایت داده‌های پالایش‌شده در یک منبع داده ذخیره می‌شوند.

مهندسان داده همواره سعی می‌کنند فرآیند ETL را خودکار کنند تا دسترسی کسانی که از داده استفاده می‌کنند به آن راحت‌تر انجام شود. تحلیلگران داده، آمار و حقایقی از مولفه‌های مختلف خانه‌های موجود در پایگاه‌داده را استخراج می‌کنند و با ارائه‌ی نمودارها و گزارش‌ها، به تیم توسعه‌ی کسبوکار و تیم محصول در گرفتن تصمیمات داده‌محور کمک می‌کنند. دانشمندان داده و مهندسان یادگیری ماشین، مسئول پاکسازی داده، مهندسی ویژگی و مدل‌سازی هستند. آن‌ها مدل‌های توسعه‌یافته را در زیرساخت مناسب مستقر، رصد و نگهداری می‌کنند. مهندسان نرم‌افزار، برنامه‌ای مانند یک وبسایت، اپلیکیشن موبایل یا دسکتاپ، طراحی و پیاده‌سازی می‌کنند تا که از مدل مستقرشده استفاده و به کاربر نهایی، خدمت ارائه کند. ۲) دیدگاه تیم چابک (Agile)

در این دیدگاه از تعداد کمی از افراد که هر یک توانایی انجام تخصص‌های مختلفی را دارند، برای داشتن یک تیم چابک استفاده می‌شود. برای مثال افراد چنین تیمی می‌توانند هم کار مهندسی داده، هم کار مهندسی یادگیری ماشین و هم کار مهندسی نرم‌افزار را انجام دهند. در نتیجه، برای افراد محدودیتی در انجام کارها وجود ندارد، اما دستیابی به این هدف نیازمند عضویت افرادی با چند تخصص در تیم است. درست است که تیم، چابک و کوچک می‌شود، اما این به خاطر این است که این افراد حرفه‌ای‌تر هستند و می‌توانند چند

تخصص مختلف را با هم انجام دهنند. بنابراین، تشکیل چنین تیمی هم سخت‌تر است و هم هزینه‌ی آن به ازای هر نفر، بیشتر از افراد تیم تشکیل‌شده در دیدگاه اول می‌شود.

هر دو دیدگاه مزایا و معایبی دارند. برخی در طرفداری از دیدگاه اول می‌گویند، هر کس باید در جایی که هست، بهترین باشد. مثلًاً یک دانشمند داده و مهندس یادگیری ماشین بتواند به بهترین نحو ممکن مدل‌سازی کند و مدلی توسعه دهد که به بیشترین عملکرد ممکن دست پیدا کند. عده‌ای دیگر در طرفداری از دیدگاه دوم این‌گونه استدلال می‌کنند که معمولاً توسعه‌دهندگان مربوط به داده، اصول مهندسی را رعایت نمی‌کنند. به عنوان مثال، تمیز و ساختاریافته کد نمی‌نویسند. بیشتر به فکر بهتر کردن عملکرد مدل هستند و امکان استقرار مدل در محیط محصول (Production Environment) را در نظر نمی‌گیرند. بنابراین هنگام ترکیب کردن کد افراد با تخصص‌های داده محور با کد مهندسی شده و تمیز، هزینه‌ی زیادی مصرف می‌شود و گاهی حتی نیاز است تمام یا بخشی از کدها توسط مهندسان نرم‌افزار، بازنویسی شوند.

اگر تمايل دارید در مورد موقعیت‌های شغلی مرتبط با داده بیشتر مطالعه کنید، این درسنامه از دوره دروازه ورود به یادگیری ماشین، منبع خوبی است. همچنین اگر به دنبال شغل در این حوزه هستید، حتماً به قسمت کاریابی کوئرا بروید و برای شغل‌های موجود این حوزه، اقدام بکنید.

۵.۲ چرا پروژه‌ها شکست می‌خورند؟

چرا پروژه‌ها شکست می‌خورند؟



طبق برآوردهای مختلف بین سالهای ۲۰۱۷ تا ۲۰۲۰ میلادی، از ۷۴ تا ۸۷ درصد از پروژه‌های یادگیری ماشین شکست می‌خورند یا به مرحله‌ی استفاده عملی توسط یک شرکت و سازمان نمی‌رسند. دلایل مختلفی برای شکست این پروژه‌ها وجود دارد و در این درسنامه، به مهم‌ترین آن‌ها اشاره خواهیم کرد. شاید اطلاع داشتن از این دلایل، بتواند مانع شکست پروژه یادگیری ماشین شما بشود. (۱) کمبود نیروی انسانی با تجربه

مهندسی یادگیری ماشین، هنوز رشته‌ی نسبتاً جدیدی است و روش استانداردی برای آموزش آن وجود ندارد. بیشتر سازمان‌ها نمی‌دانند چگونه در این حوزه، متخصص استخدام کنند. همچنین، اکثر نیروی انسانی موجود در بازار کار، افرادی هستند که فقط یک یا چند دوره آنلاین گذرانده‌اند و تجربه‌ی عملی قابل توجهی ندارند. در نتیجه، بخش قابل توجهی از نیروی کار موجود، فقط دارای متخصص سطحی و تجربه‌ی کار با داده‌های مورد استفاده در کلاس‌های درس هستند. لازم به ذکر است که بسیاری از این افراد، تجربه‌ی کار بر روی تمامی مراحل یک چرخهٔ پروژهٔ یادگیری ماشین را نیز ندارند.^{۲)} عدم پشتیبانی توسط مدیران ارشد

معمولًاً مهندسین یادگیری ماشین و نرم‌افزار دارای اهداف، انگیزه‌ها و معیارهای موفقیت متفاوتی هستند. آن‌ها همچنین بسیار متفاوت عمل می‌کنند. در یک سازمان چاپک، مهندسین نرم‌افزار از متدولوژی اسکرام (Scrum) استفاده کرده و وظایف خود را در اسپرینت‌هایی با خروجی‌های قابل تعریف و عدم قطعیت کم، جلو می‌برند. در حالی که، مهندسین یادگیری ماشین به علت ذات این حوزه که با اکتشاف و آزمایش‌های متعدد همراه است، با عدم قطعیت بیشتری، مواجه هستند. بسیاری از این آزمایش‌ها، به یک نتیجه‌ی قابل تحويل ختم نمی‌شود و یا بعد از پایان وظایف تعریف شده، دقت مورد نظر سیستم، بهبود مورد نظر را پیدا نمی‌کند و نیاز به بازتعریف کارها و انجام مراحل جدید می‌باشد. در نتیجه، این موارد می‌توانند توسط افراد خارج از تیم، اتلاف وقت تلقی شوند.

این موارد را در کنار این حقیقت قرار بدھید که در اکثر شرکت‌ها، مدیران رتبه بالای آن، دارای تجربه کار فنی و مهندسی نیستند. آن‌ها نمی‌دانند که هوش مصنوعی چگونه کار می‌کند و یا درک سطحی یا بیش از حد خوش‌بینانه‌ای دارند و فکر می‌کنند که با منابع فنی و انسانی، هوش مصنوعی بتوانند در کوتاه مدت هر مشکلی را حل کند. هنگامی‌که پیشرفت سریع توسط تیم یادگیری ماشین اتفاق نمی‌افتد، آن‌ها به راحتی، اعضای تیم را سرزنش می‌کنند یا علاقه‌ی خود را به هوش مصنوعی به عنوان یک ابزار بی‌اثر با نتایج پیش‌بینی نشده و نامشخص به‌طور کامل از دست می‌دهند.

این مشکلات را در کنار ناتوانی یک مهندس یادگیری ماشین در مکاتبه‌ی نتایج و چالش‌های خود با مدیریت قرار دھید. دلیل این امر این است که این دو گروه، واگان مشترکی ندارند و از لحاظ فنی دارای سطوح بسیار متفاوتی هستند. حتی موفقیتی که بد ارائه شود، می‌تواند به عنوان یک شکست توسط مدیران تلقی شود. به همین دلیل است که در سازمان‌های موفق در این حوزه، مدیران سطح بالا و مسئول هوش مصنوعی، اغلب دارای سابقه‌ی فنی یا علمی متناسب هستند.^{۳)} نبود زیرساخت داده

تیم یادگیری ماشین، با داده سر و کار دارند و کیفیت داده برای موفقیت یک پروژهٔ یادگیری ماشین بسیار مهم است. به همین دلیل است که اعضای این تیم بایستی بتوانند به‌سادگی به وسیله‌ی زیرساخت داده‌ی یک شرکت/سازمان به داده‌های باکیفیت دسترسی داشته باشند. در عین حال، بایستی این اطمینان حاصل شود که در هنگام استفاده از مدل آموزش‌دیده برای پیش‌بینی روی نمونه‌های جدید، آن نمونه‌ها نیز دارای کیفیت مشابه باشند.

متاسفانه در اغلب موارد و به دلیل عدم وجود یک تیم مهندسی داده (Data Engineering)، این زیرساخت داده در یک شرکت و یا سازمان وجود ندارد. در نتیجه، تیم یادگیری ماشین مجبور می‌شود با استفاده از اسکریپت‌های مختلف موقعت، مجموعه‌داده‌ی مربوط به آموزش هر پروژه را به‌دست آورند. این امر باعث افزایش پیچیدگی برای دسترسی به داده‌های با کیفیت در پروژه‌های جدید و همچنین سختی نگهداری کد و تکرار نتایج می‌شود.^{۴)} چالش برچسب‌گذاری داده

اکثر پروژه‌های یادگیری ماشین، از نوع یادگیری نظارت‌شده هستند و تیم یادگیری ماشین از مجموعه‌داده‌ی دارای برچسب برای آن‌ها استفاده می‌کنند. این مجموعه‌داده‌ها معمولاً سفارشی هستند، بنابراین برچسب‌زدن به‌طور خاص برای هر پروژه انجام می‌شود. بر اساس برخی گزارش‌ها تا سال ۲۰۱۹، حدود ۷۶ درصد تیم‌ها،

خود، مجموعه‌داده‌های مورد نیاز را برچسب‌گذاری می‌کردند و فقط ۶۳ درصد از آن‌ها، فرآیند برچسب‌گذاری را خودکار کرده‌اند.

این امر منجر به صرف زمان قابل توجهی توسط اعضای ماهر این تیم‌ها، بر روی برچسب‌زن و توسعه‌ی ابزار برچسب‌گذاری می‌شود و این یک چالش بزرگ برای اجرای موثر یک پروژه یادگیری ماشین است. به همین دلیل است که برخی از شرکت‌ها، فرآیند برچسب‌گذاری داده را به شرکت‌های دیگر، بروونسپاری می‌کنند. با این حال، این تیم‌ها باقیست که کیفیت داده‌های برچسب‌گذاری شده دریافتی را نیز تایید کنند تا از کیفیت پایین و یا اشتباہ بودن آن‌ها جلوگیری کنند.

به منظور حفظ کیفیت و ثبات، بعضی از شرکت‌ها و سازمان‌ها اقدام به آموزش دادن و استخدام افراد برچسب‌زن (افراد داخل یا از شرکت بیرونی) کرده‌اند و این امر، به نوبه خود می‌تواند باعث کندی پیشرفت پروژه‌های یادگیری ماشین بشود. در نظر داشته باشید که بر اساس گزارشات مشابه، شرکت‌هایی که برچسب‌گذاری داده‌ها را بروونسپاری می‌کنند، به احتمال زیاد و خوبی پروژه‌های یادگیری ماشین خود را به مرحله‌ی استفاده عملی می‌رسانند.^۵ عدم همکاری

مجموعه‌داده‌ی مورد نیاز برای یک پروژه یادگیری ماشین، اغلب در دپارتمان‌های مختلف یک شرکت/سازمان و تحت مالکیت و محدودیت‌های امنیتی آن‌ها هستند. همچنین، این داده‌ها می‌توانند هریک دارای قالب مختلفی باشند. در نظر داشته باشید که افراد مسئول داده در دپارتمان‌های مختلف، می‌توانند هریک دارای اهداف و اولویت‌های خود باشند. عدم اعتماد/شناخت و همکاری بین این افراد، می‌تواند باعث اصطکاک، کندی و یا توقف طولانی‌مدت پروژه شوند.

علاوه بر این، معمولاً دپارتمان‌های مختلف، بودجه‌های خاص خود را دارند و شاید علاقه‌ای به صرف آن بودجه برای کمک به دپارتمان دیگر نکنند. این موارد را در کنار عدم همکاری‌هایی که بین افراد مختلف یک تیم یادگیری ماشین می‌تواند رخ دهد، قرار دهید.^۶ پروژه‌های امکان‌نایابی

به‌خاطر هزینه نیروی انسانی و زیرساخت داده، بسیاری از پروژه‌های یادگیری ماشین دارای هزینه‌ی بالایی برای شرکت/سازمان هستند. به همین دلیل و برای جبران سرمایه‌گذاری، ممکن است که شرکت/سازمان، اهداف بسیار بلندپروازانه‌ای مانند تغییر کامل سازمان یا محصول یا وعده‌ی غیرواقعی بازگشت سرمایه بدنهند.

در نتیجه، پروژه‌های بسیار بزرگ که شامل همکاری بین تیم‌های متعدد، دپارتمان‌ها و شرکت‌های بیرونی می‌شود، فشار زیادی را روی نیروی انسانی وارد می‌کنند. چنین پروژه‌های بیش از حد بلندپروازانه‌ای، ممکن است ماهها یا حتی سال‌ها به طول انجامد. در طول این مدت، ممکن است که مدیران ارشد علاقه‌ی خود به این پروژه را از دست داده و به تمام حوزه‌ی یادگیری ماشین بدین شوند و این حوزه از لیست پروژه‌های با اولویت‌های بالا خارج شود.

در نظر داشته باشید که اگر پروژه‌ی یادگیری ماشین خیلی طول بکشد تا تکمیل شود، ممکن است که دیر وارد بازار شود و در عمل به اهداف خود دست پیدا نکند. به همین دلیل است که پیشنهاد می‌کنیم در ابتدا، روی پروژه‌های امکان‌نایابی از لحاظ فنی که شامل همکاری‌های ساده بین تیم‌ها و دپارتمان‌ها هستند، تمرکز کنید تا که یک هدف تجاری ساده را پاسخ دهید.^۷ عدم هماهنگی بین تیم‌های فنی و تجاری

بسیاری از پروژه‌های یادگیری ماشین بدون درک روشن تیم فنی از اهداف تجاری آن، شروع می‌شوند. تیم یادگیری ماشین معمولاً، مسئله را به یک سوال دسته‌بندی یا رگرسیون تقلیل می‌دهند که باید دقت بالایی (خطای کمی) برای یک هدف فنی دیگر کسب کند. بدون تعامل با تیم تجاری برای کسب آگاهی از اهداف تجاری مانند افزایش نرخ کلیک یا حفظ کاربر که آن‌ها برای این پروژه تصور کرده‌اند، خروجی کار توسط تیم فنی، احتمالاً مورد قبول تیم تجاری قرار نمی‌گیرد. در چنین شرایطی، پروژه‌ها به دلیل صرف زمان و منابع به

پایان می‌رسند ولی تیم تجاری، از نتیجه‌ی کار راضی نیست.

فصل ۳

آماده‌سازی داده

۱.۳ اهداف فصل

اهداف فصل



فرض کنید، بعد از مطالعه‌ی طولانی دوره‌ی یادگیری ماشین قصد دارید خوراکی‌ای نوش جان کنید. شما هوس پیتزا کرده‌اید، بنابراین دست به کار می‌شوید ولی به دلیل خستگی مرحله آماده‌سازی مواد اولیه را نادیده می‌گیرید. فلفل دلمه‌ای را بدون خرد کردن و شتسن بر روی خمیر قرار می‌دهید و همچنین بدون اندازه‌گیری مشخص و به مقدار نامعلوم به خمیر خود از هر نوع سسی که دارید اضافه می‌کنید و همچنین

بدون توجه با تاریخ مصرف، سوسيسي که فاسد شده است را به ترکیب بالا اضافه می‌کنید. آیا این پیتزا قابل خوردن است؟ خیر

به همین صورت وقتی از داده‌های نامناسب برای تحلیل یا یادگیری ماشین استفاده می‌کنیم، نمی‌توان انتظار داشت که بدون توجه به داده‌های ورودی نتایج کارآمدی به دست بیاوریم؛ به همین دلیل، لازم است پیش از انجام تحلیل‌های مختلف بر روی یک مجموعه داده، ابتدا یک سری پیش‌پردازش‌هایی روی داده‌ها انجام دهیم و به اصطلاح داده‌ها را برای تحلیل خود آماده کنیم.

در این فصل به معرفی پراهمیت‌ترین روش‌های موجود برای آماده‌سازی داده‌ها می‌پردازیم و همچنین مقداری با مفهوم خود داده آشنا می‌شویم.

خلاصه‌ای از مطالب این فصل:

آشنایی با نکات قابل توجه قبل از جمع‌آوری و استفاده‌ی داده آشنایی با ویژگی‌هایی برای سنجش کیفیت داده و نحوه‌ی تشخیص کیفیت مجموعه داده‌ی درک چالش‌های داده (شامل سوگیری و نویز و ...) آشنایی با تقسیم‌بندی داده‌ها (آموزش، اعتبارسنجی و آزمایش) آشنایی با داده‌های پرت و نحوه‌ی مدیریت آن‌ها آشنایی با روش‌های شناسایی داده‌های گمشده و جایگزینی یا مدیریت آن‌ها بررسی چگونگی برخورد با مجموعه داده‌هایی با توزیع نامتقارن

۲.۳ سوالاتی درباره داده

سوالاتی درباره داده

شما یک مسئله‌ی یادگیری ماشین دارید و هدف، ورودی و خروجی آن مشخص است. حال، می‌خواهید شروع به جمع‌آوری مجموعه داده‌ی موردنیاز برای آن کنید. صبر کنید، سوالاتی هست که قبل از انجام این کار، بایستی به آن‌ها پاسخ دهید: آیا می‌توان به داده‌ی موردنظر دسترسی پیدا کرد؟

آیا داده‌ای که به آن نیاز دارید، هم‌اکنون وجود دارد؟ اگر جواب مثبت است، آیا داده در دسترس است؟ (از نظر فیزیکی، اخلاقی، مالی و دیگر موضوعات) اگر باید داده را خریداری کرد یا از مجموعه داده‌ی شخص دیگری استفاده کرد، به این فکر کرده‌اید که چگونه باید از این اطلاعات استفاده کرد و نحوه‌ی اشتراک داده به چه شکل است؟ آیا باید توافق جدیدی با تولیدکننده‌ی اصلی داده صورت بگیرد؟

اگر داده در دسترس است، آیا باید قوانین کپیرایت را رعایت کرد؟ اگر مالکیت داده به شکل اشتراکی باشد، تکلیف ما چیست؟ اگر داده حساس است (حریم خصوصی افراد، شرکت‌ها یا سازمان‌های دولتی در خطر باشد)، چگونه بایستی با آن رفتار کرد؟ آیا می‌توانیم این اطلاعات را برای استفاده‌ی بلندمدت نزد خود نگه داریم؟

آیا این داده قرار است در کنار مدل به اشتراک گذاشته شود؟ اگر جواب مثبت است، آیا باید توافقنامه جدیدی با صاحبان داده نوشته شود؟ آیا در تحلیل‌ها و اشتراک داده، اطلاعات هویتی مانند نام و آدرس افراد را باید حذف کرد؟

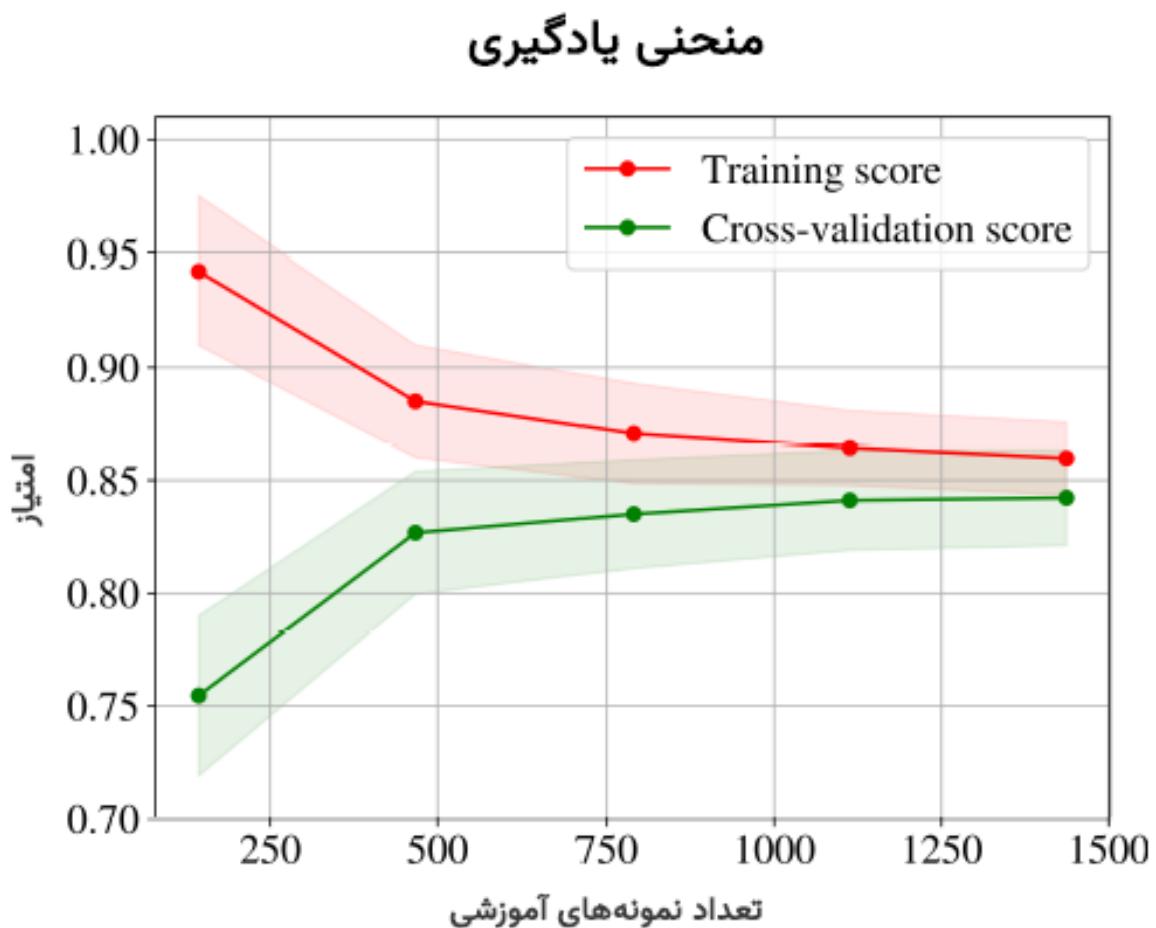
حتی اگر داده کاملاً در دسترس شما باشد، قبل از شروع کار حتماً به سوالات بالا پاسخ دهید و در صورت نیاز، با تیم حقوقی شرکت خود، درباره‌ی این موضوع گفتگو کنید تا مطمئن شوید در آینده هیچ‌گونه مشکلی ایجاد نشود. آیا حجم داده کافی است؟

احتمالاً می‌دانید که در مسائل یادگیری ماشین، معمولاً مشخص نیست که چه مقدار داده لازم است، مخصوصاً وقتی که حداقل عملکرد موردنیاز سخت‌گیرانه انتخاب شده باشد.

اگر در مورد کافی بودن داده مطمئن نیستید، به این فکر کنید که داده‌ی جدید با چه سرعتی تولید می‌شود. برای بعضی پروژه‌ها، می‌توان با داده‌ی موجود شروع به کار کرد و تا وقتی که مشغول حل مسائل مرتبط با مهندسی ویژگی، مدل‌سازی و دیگر مسائل هستید، داده‌ی جدید به تدریج وارد کار شود. داده‌ی جدید می‌تواند نتیجه‌ی طبیعی بعضی فرآیندها باشد یا توسط متخصصان برچسب‌گذاری، آماده و به شما برسد.

تخمین بزنید که پروژه برای کامل شدن به چقدر زمان نیاز دارد. آیا تا زمان اتمام پروژه، داده‌ی کافی جمع‌آوری می‌شود؟ برای پاسخ به این سوال باید از تجربیات کار روی پروژه‌های مشابه یا گزارش‌های ثبت‌شده استفاده کنید.

یک روش عملی برای یافتن مقدار داده‌ی کافی، رسم نمودار منحنی‌های یادگیری است. برای این کار، امتیاز آموزش و امتیاز اعتبارسنجی را به ازای تعداد نمونه‌های آموزشی رسم کنید. یک نمونه از این نوع نمودار، در شکل زیر آمده است:



امتیاز آموزش و امتیاز اعتبارسنجی چیست؟

منظور از امتیاز آموزش (Training Score)؛ عملکرد مدل یادگیری ماشین شما بر روی نمونه‌هاییست که مدل طبق آن‌ها آموزش دیده و منظور از امتیاز اعتبارسنجی (Validation Score)؛ عملکرد مدل بر روی

نمونه‌هایی است که کنار گذاشته شده و در هنگام آموزش دیده نشده‌اند. از آنجا که این نمونه‌ها در آزمایش‌های شما مکرراً مورد استفاده قرار می‌گیرد نام اعتبارسنجی بر روی آن گذاشته شده است. برای محاسبه‌ی عملکرد مدل، معیارهای ارزیابی گوناگونی در دسترس شماست که در فصل‌های بعدی با آن‌ها آشنا خواهید شد.

درباره‌ی انواع نمونه‌ها نیز در درسنامه‌ی « تقسیم‌بندی مجموعه‌داده » به تفصیل بحث خواهد شد.

با توجه به شکل مشخص می‌شود کارایی مدل از یک نقطه به بعد بهتر نمی‌شود و افزایش داده‌های آموزشی دیگر تاثیر خاصی ندارد.

اگر با افزایش داده، کارایی مدل بهتر نشد، می‌توان نتیجه گرفت که نیازی به اضافه کردن داده نیست. البته عدم بهبود کارایی مدل می‌تواند دلایل دیگری مانند موارد زیر داشته باشد:

ویژگی‌هایی که از آن‌ها برای آموزش مدل استفاده می‌شوند، قدرت پیش‌بینی کافی ندارند. مدل انتخاب شده ساده است و عملکرد مناسبی ندارد.

در فصل‌های بعدی، با روش‌هایی برای حل هر یک از این موارد، آشنا خواهید شد.

بعضی متخصصان از قواعدی که بر اساس تجربه به دست آمده جهت تخمین تعداد نمونه‌های موردنیاز استفاده می‌کنند. بعضی از پرکاربردترین آن‌ها عبارتند از:

حداقل ۱۰ برابر تعداد ویژگی‌ها حداقل ۱۰۰ یا ۱۰۰۰ برابر تعداد دسته‌ها حداقل ۱۰ برابر تعداد پارامترهای قابل آموزش

توجه داشته باشید که داشتن تعداد نمونه‌های بالا و حجم زیاد داده‌ها، به این معنی نیست که بایستی از تمام آن‌ها استفاده کنید. یک زیرمجموعه‌ی کوچک از مجموعه‌داده می‌تواند نتایج خوبی در عمل داشته باشد و فرآیند آموزش مدل سریع‌تر انجام شود. هنگام نمونه‌برداری از مجموعه‌داده باید توجه کنید که نمونه‌های انتخابی شما، به خوبی ویژگی‌های کل داده‌ها را نشان دهد. به عنوان مثال، اگر مجموعه‌دادهی شما در یک بازه‌ی ۵ ساله جمع‌آوری شده باشد، آیا داده‌های ۵ سال قبل، همچنان معتبر هستند؟ یا بایستی به داده‌های فقط ۱ یا ۲ سال اخیر، اکتفا کنید؟ آیا داده قابل استفاده است؟

کیفیت داده یکی از مهم‌ترین عوامل تاثیرگذار بر کارایی مدل را آموزش دهید که با گرفتن نام افراد، جنسیت آن‌ها را مشخص کند. شما یک مجموعه‌داده شامل نام و جنسیت افراد به مدل خود می‌دهید، اما می‌بینید که دقت مدل بسیار پایین است. به نظر شما چه اتفاقی افتاده است؟

شاید داده‌های موجود در مجموعه‌داده‌ای که از آن استفاده کرده‌اید واقعی نبوده و خروجی یک مدل آماری بوده است. در این صورت، مدل شما در بهترین حالت می‌تواند به کارایی آن مدل آماری برسد.

همچنین باید بررسی کنیم که آیا داده با فرمت مناسب برای الگوریتم‌های یادگیری ماشین ذخیره شده است یا خیر. در فرمت مناسب، هر سطر، یک نمونه و هر ستون، یک ویژگی برای آن نمونه‌ها می‌باشد. ممکن است برخی مقادیر ویژگی‌ها ناموجود باشد یا به بیان دیگر داده دارای مقادیر گم‌شده Values (Missing) باشد. روش‌هایی برای پر کردن مقادیر گم‌شده وجود دارد که در درسنامه‌های بعدی با آن‌ها آشنا خواهیم شد. مسئله دیگری که هنگام کار با مجموعه‌داده‌ها پیش می‌آید، وجود سطرهای تکراری است. معمولاً سطرهای تکراری را باید حذف کرد مگر اینکه خودمان با هدفی خاص (برای مثال، متوازن کردن مجموعه‌دادهی غیرمتوازن) بعضی سطرهای را تکرار کرده باشیم. با تکنیک‌های متوازن کردن داده نیز در فصل بعد آشنا می‌شویم.

داده‌ی تاریخ مصرف گذشته یا به روزرسانی نشده هم می‌تواند باعث کارا نبودن مدل شود. فرض کنید می‌خواهید مدلی بسازید که خراب بودن یک پرینتر را تشخیص دهد. برای این کار باید اطلاعات وضعیت پرینتر در حالتی که درست کار می‌کند و وضعیتی که مشکل دارد را به مدل بدهید. اگر اطلاعات شما مربوط به نسل‌های قبلی

پرینتر باشد، و مدل‌های جدید پرینتر ارتقا یافته باشند، مدل شما قادر به تشخیص پرینترهای ایراددار نخواهد بود.

در نهایت، ممکن است مجموعه داده ناکامل باشد و شامل همه نمونه‌هایی که در عمل دیده می‌شوند، نباشد. برای مثال، ممکن است مجموعه داده تصاویر حیوانات، فقط شامل حیوانات یک ناحیه‌ی جغرافیایی خاص باشد یا تمام عکس‌ها در یک فصل خاص گرفته شده باشند. آیا داده قابل فهم است؟

اگر که مجموعه داده برای شما قابل فهم نباشد، می‌تواند باعث بروز خطا در فرآیند مدل‌سازی شود. به عنوان مثال شرایطی را در نظر بگیرید که متغیر هدف، با استفاده از یکی از ویژگی‌های مجموعه داده به دست می‌آید. یعنی متغیر هدف به کمک یکی از ویژگی‌ها کاملاً قابل محاسبه است ولی چون شما بر داده‌ها تسلط ندارید، متوجه این موضوع نمی‌شوید. فرض کنید بر روی یک پروژه‌ی تخمین قیمت خانه با استفاده از داده‌های واقعی مشغول به کار هستید. مدل شما با دریافت اطلاعات خانه از قبیل متراث، تعداد اتاق خواب و نوع منطقه باید قیمت آن خانه را تخمین بزند. اطلاعات معاملات اخیر را به شما داده‌اند و شما بدون معطی ستون قیمت را حذف می‌کنید و بقیه اطلاعات را به مدل خود می‌دهید. مدل آموزش می‌بیند و به شکلی باورنکردنی خوب عمل می‌کند. یعنی با دقت ۱۰۰ درصد قیمت خانه‌ها را تخمین می‌زند. مدل را به مشتری تحویل می‌دهید و مشتری مدل را به کار می‌گیرد اما مدل در عمل خیلی بد عمل می‌کند. به نظر شما چه اتفاقی افتاده است؟

چیزی که اتفاق افتاده نشت اطلاعات (Data Leakage) یا به بیان دقیق‌تر نشت ویژگی هدف نام دارد. پس از بررسی دقیق مجموعه داده متوجه می‌شوید که یکی از ویژگی‌ها، حق کمیسیون بنگاه معاملات املاک است. مدل، یاد گرفته که حق کمیسیون رابطه‌ی مستقیمی با قیمت خانه دارد و با ضرب یک ثابت در حق کمیسیون، قیمت خانه به دست می‌آید. حق کمیسیون فقط در داده‌ی آموزش وجود دارد و در عمل، قبل از انجام معامله خانه، حق کمیسیون مشخص نیست. این باعث شده تا مدل شما روی داده‌ی آموزشی خوب عمل کند، اما در عمل بسیار ضعیف کار کند. در فصل بعد با مفهوم نشت داده دقیق‌تر آشنا خواهیم شد. آیا داده قابل اعتماد است؟

قابل اعتماد بودن داده با توجه به رویه‌ای که برای جمع‌آوری داده طی شده، مشخص می‌شود. فرض کنید داده توسط افرادی برچسب خورده که راضی به این کار نبوده‌اند. آیا به برچسب‌ها اعتماد می‌کنید؟ در مواردی داده توسط سنسورها جمع‌آوری می‌شود. تا چه اندازه به دقت سنسورها اعتماد دارید؟

قابل اعتماد بودن داده‌ها به تعویق و غیرمستقیم بودن ذاتی برچسب‌ها هم بستگی دارد. اگر بین زمانی که داده تولید می‌شود و زمانی که برچسب داده مشخص می‌شود، فاصله‌ی زیادی وجود داشته باشد، می‌گوییم که برچسب تعویق دارد. با مثالی برچسب تعویق‌دار را نشان دهیم.

فرض کنید می‌خواهیم رفتار مشتریان را بررسی کنیم و برای هر مشتری مشخص کنیم که در آینده (فرضاً ۶ ماه بعد) مشتری، خرید از ما را قطع کرده یا هنوز مشتری ماست. اطلاعاتی که از مشتری داریم مربوط به رفتار فعلی اوست، اما برچسب (مشتری مانده یا رفته) در آینده مشخص می‌شود. ممکن است از الان تا رسیدن به آن تاریخ، اتفاقات مهمی رخ دهد که در تصمیم‌گیری مشتری برای ماندن یا رفتن تاثیرگذار باشند. پس برچسب تعویق‌دار باعث غیرقابل اعتماد شدن داده می‌شود.

حالا باید مسئله‌ی برچسب غیرمستقیم را بررسی کنیم. فرض کنید این بار می‌خواهیم جذابیت صفحات اینستاگرام برای کاربران را تخمین بزنیم. یعنی مدلی بسازیم که با گرفتن اطلاعات کاربر و صفحه‌ی اینستاگرام، مشخص کند که کاربر از آن صفحه خوشش آمده است یا خیر. یک برچسب مستقیم علاقه‌ی کاربر را نشان می‌دهد اما برچسب غیرمستقیم، نشان می‌دهد که کاربر مقداری از آن صفحه خوشش آمده است. برای مثال، اگر کاربر بر روی لایک کلیک کرده باشد، می‌گوییم که از صفحه خوشش آمده ولی اگر بر روی لینک

صفحه کلیک کرده باشد، می‌گوییم که از آن صفحه مقداری خوشش آمده است. کاربر ممکن است اشتباهی بر روی لینک کلیک کرده باشد یا تبلیغات او را فریب داده باشد. می‌بینید که در این موارد داده‌های ما برای تشخیص میزان علاقه خیلی قابل اعتماد نیستند.

مسئله‌ی دیگری که باعث غیر قابل اعتماد شدن داده می‌شود، حلقه‌ی بازخورد (loop) است. حلقه‌ی بازخورد ویژگی سیستم‌هایی است که داده‌ی لازم برای آموزش مدل، از خود مدل به دست آمده باشد. فرض کنید یک سیستم پیشنهاددهنده ساخته‌اید که با استفاده از فیلم‌هایی که کاربران دیده‌اند، به آنها فیلم جدید پیشنهاد می‌دهد. این سیستم فیلمی را برای پیشنهاد به کاربران انتخاب می‌کند و به طیف وسیعی از کاربران پیشنهاد می‌دهد. بسیاری از کاربران بر روی آن فیلم کلیک می‌کنند. سیستم اگر بر اساس کلیک بر روی لینک آموزش دیده باشد، فرض می‌کند که کاربران آن فیلم را دوست دارند و همان فیلم را به کاربران بیشتری پیشنهاد می‌دهد. این گونه می‌شود که سیستم داخل حلقه‌ی بازخورد می‌افتد.

۳.۳ چالش‌های داده

چالش‌های داده

به‌طور خلاصه، در یادگیری ماشین وظیفه‌ی اصلی شما انتخاب یک الگوریتم یادگیری و آموزش آن بر روی مجموعه‌داده‌ای است که در اختیار دارید. اما خبر بد این است که ممکن است مواردی در این حین اشتباه پیش برود و با چالش‌های مختلفی مواجه شوید. یکی از این موارد که ممکن است کار را خراب کند، داده‌ی بد است.

برای اینکه به یک کودک یاد بدهیم سبب چیست، کافی است به سبب اشاره کنیم و بگوییم «سبب» (احتمالاً) مجبور می‌شویم این عمل را چندین بار انجام دهیم) و تبریک! هم‌اکنون کودک توانسته سبب‌ها را در انواع رنگ‌ها و شکل‌ها تشخیص دهد. اما یادگیری ماشین به این اندازه باهوش نیست! برای اینکه اکثر الگوریتم‌های یادگیری ماشین به درستی کار کنند، به تعداد زیادی از نمونه‌ها نیاز دارند. حتی برای مسائل بسیار ساده معمولاً به هزاران نمونه نیاز داریم. برای مشکلات پیچیده‌تر مانند پردازش تصویر یا گفتار ممکن است به میلیون‌ها نمونه نیاز داشته باشیم. برچسب زدن نمونه‌ها

همان‌گونه که پیش از این هم مطالعه کردید در مسائل یادگیری نظارت شده، به مجموعه‌داده‌های برچسب‌دار (labeled dataset) نیاز داریم. اما در عمل تعداد مجموعه‌داده‌های بدون برچسب (unlabeled dataset) بسیار بیشتر از مجموعه‌داده‌های دارای برچسب است. بنابراین با توجه به داده‌هایی که در اختیارمان گذاشته می‌شود، ممکن است مجبور شویم در قدم اول، آن‌ها را برچسب بزنیم. درصد قابل توجهی (طبق برخی تحقیقات حدود ۸۰ درصد) از زمان یک پروژه‌ی یادگیری ماشین صرف جمع‌آوری، سازمان‌دهی و برچسب‌گذاری داده‌های آن می‌شود. با توجه به این‌که معمولاً تیم‌ها نمی‌خواهند چنین زمان و انرژی‌ای را صرف آماده‌سازی داده‌ها کنند، به سراغ استفاده از مجموعه‌داده‌ای آماده و ساختاریافته برای آموزش و استقرار مدل می‌روند.

فرض کنید هدف ما این است که متوجه شویم انواع مختلف کسب‌وکارها در یک شهر خاص در کدام مناطق قرار گرفته‌اند. بهترین راه حل، خرید چنین مجموعه‌داده‌ای از یک سازمان دولتی می‌باشد. اما دلایل مختلفی وجود دارد که این کار را ناممکن می‌سازد. به‌طور مثال، ممکن است پایگاه‌داده‌ی سازمان کامل نباشد یا بسیار قدیمی باشد. چه کنیم؟ ناگهان به فکر شما خطور می‌کند که اتومبیل‌های مجهز به دوربین را به خیابان‌های یک شهر بفرستید و آن‌ها از تمام مغازه‌های خیابان‌ها عکس‌برداری کنند!

احتمالاً حدس زده‌اید که چنین کاری هزینه‌ی بالایی داشته و همچنین صرفاً جمع‌آوری عکس از مغازه‌های یک شهر کافی نیست. ما نیاز داریم که نوع کسب‌وکار هر مغازه را نیز مشخص کنیم. در واقع نیاز داریم که مجموعه‌داده‌ی خود را برچسب بزنیم. به‌طور مثال، کافی‌شاپ، رستوران، داروخانه، باشگاه ورزشی، جایگاه‌های سوخت‌گیری. با توجه به این که این کار به صورت دستی صورت می‌گیرد، پرداخت هزینه به افراد بابت برچسب‌گذاری تمامی این داده‌ها ممکن است گران تمام شود.

برای کاهش هزینه‌های این فرآیند می‌توان کارهای خلاقانه‌ای انجام داد. به‌طور مثال نرم‌افزاری طراحی کرد که نام تمام کسب‌وکارها در آن نوشته شده باشد و در هر مرحله یک عکس به فرد برچسب‌زننده نشان داده شود. در اینجا آن شخص می‌تواند با موس بر روی کسب‌وکار منطبق با عکس کلیک کند و برچسب آن عکس زده شود. حتی می‌توانیم کار خود را بهبود دهیم و ابتدا تعداد کمی از داده‌ها را به شکل دستی برچسب بزنیم و آن داده‌ها را داده‌های «آموزش» در نظر گرفته و مدل را طبق آن‌ها آموزش دهیم. کامل‌اً مشخص است که این مدل، مدل قدرتمندی نیست زیرا داده‌هایی که با آن آموزش دیده است، بسیار کم بوده‌اند. منتها می‌توانیم داده‌های بدون برچسب باقی‌مانده را با استفاده از این مدل برچسب بزنیم و از فردی که وظیفه‌ی برچسب زدن به شکل دستی را دارد بپرسیم آیا این داده با برچسبی که زده شده مطابقت دارد یا خیر. بنابراین گزینه‌های انتخابی از تمامی کسب‌وکارها به «بله» یا «خیر» کاهش یافته است و فرد برچسب‌زننده می‌تواند در صورتی که برچسب اشتباه زده شده باشد، گزینه‌ی خیر را انتخاب کند و برچسب صحیح را به شکل دستی انتخاب و وارد کند. کیفیت داده

کیفیت مجموعه‌داده یکی از عوامل موثر بر عملکرد مدل می‌باشد و شامل دو جزء اصلی زیر است:

کیفیت مجموعه‌داده خام کیفیت برچسب‌گذاری

از برخی مشکلات رایج در مجموعه‌داده خام می‌توان به موارد زیر اشاره کرد که برخی از آن‌ها در این درسنامه و دو مورد آخر نیز در درسنامه‌های بعدی به صورت کامل شرح داده خواهند شد.

نویز بایاس قدرت پیش‌بینی پایین قدیمی شدن داده‌های پرت نشت داده

نویز (Noise)

نویز به اختلالات تصادفی، خطاهای ناخواسته یا تغییرات غیرمعناداری در داده‌ها گفته می‌شود که باعث می‌شوند مقدار مشاهده شده با مقدار واقعی پدیده تفاوت داشته باشد. این اختلالات ممکن است ناشی از خطای اندازه‌گیری، محدودیت ابزار ثبت داده، شرایط محیطی یا فرآیندهای تصادفی باشند و در انواع داده‌ها مانند داده‌های عددی، تصویری، متنی و صوتی مشاهده شوند. نویز معمولاً اطلاعات مفید جدیدی به داده اضافه نمی‌کند و صرفاً باعث کاهش کیفیت نمایش پدیده واقعی می‌شود، هرچند در برخی موارد می‌توان با استفاده از الگوریتم‌های حذف نویز یا بازسازی داده، اثر آن را تا حدی کاهش داد. توجه به این نکته ضروری است که نویز با داده‌های پرت یا داده‌های ناقص یکسان نیست؛ داده‌های پرت مقادیری هستند که به‌طور معناداری از توزیع غالب فاصله دارند و لزوماً اشتباه یا نویزی محسوب نمی‌شوند، اما مانند نویز می‌توانند بر عملکرد مدل و شاخص‌های آماری اثر قابل توجهی بگذارند. بایاس یا سوگیری (Bias)

بایاس دارای انواع مختلفی است که هر یک به تفصیل، در ادامه توضیح داده شده‌اند: ۱. بایاس انتخابی (Selection Bias)

زمانی اتفاق می‌افتد که مجموعه‌داده به گونه‌ای انتخاب شود که منعکس‌کننده‌ی توزیع مجموعه‌داده‌های دنیای واقعی نباشد. این اتفاق می‌تواند به‌دلیل راحتی دسترسی به برخی از منابع داده و یا مقرن به‌صرفه بودن آن‌ها باشد. برای مثال، فرض کنید یک نویسنده می‌خواهد نظر خوانندگان بر روی کتاب جدیدش را بداند، این نویسنده چندین فصل اولیه را برای خوانندگان قبلی کتاب‌هایش ارسال می‌کند و به احتمال زیاد،

چون خوانندگان قبلی از قلم این نویسنده خوششان می‌آید، کتاب جدیدش را هم دوست خواهند داشت. نکته‌ی اصلی اینجاست که این اطلاعات، نکات زیادی در مورد نظر یک خواننده عمومی به کتاب جدید این نویسنده به همراه ندارند. ۲. بایاس خودانتخابی (Self-Selection Bias)

در این مورد، داده‌ها معمولاً از منابعی دریافت می‌شوند که به صورت داوطلبانه ارائه می‌گردند. بیشتر مجموعه‌داده‌های نظرسنجی‌ها دارای چنین بایاسی هستند. برای مثال، می‌خواهید مدلی را آموزش دهید که رفتار کارآفرینان موفق را پیش‌بینی کند. شما تصمیم می‌گیرید ابتدا از کارآفرینان بپرسید که آیا آن‌ها موفق هستند یا خیر. سپس فقط داده‌های به دست آمده از کسانی را که خود را موفق اعلام کرده‌اند نگه می‌دارید. مشکل اینجاست که به احتمال زیاد، کارآفرینان واقعاً موفق، زمانی برای پاسخ به سؤالات شما ندارند! در حالی که کسانی که ادعا می‌کنند خود موفق هستند ممکن است در این مورد اشتباہ کنند. به مثال نظرسنجی برگردیم، کاربران ناراضی بیشتر تمایل به ارائه رتبه‌های میان‌رده به جای ضعیف و خیلی ضعیف دارند و همین باعث می‌شود مجموعه‌داده به جای سوگیری به سمت ضعیف و خیلی ضعیف، به سمت میان‌رده سوگیری کنند. ۳. بایاس متغیر حذف شده (Omitted Variable Bias)

زمانی اتفاق می‌افتد که مجموعه‌داده، ویژگی لازم برای پیش‌بینی دقیق را نداشته باشد. برای مثال، فرض کنید که در حال کار بر روی یک مدل پیش‌بینی هستید و می‌خواهید بدانید که آیا مشتری اشتراک خود را ظرف شش ماه آینده لغو می‌کند؟ یک مدل را آموزش می‌دهید و در همان لحظه این مدل به اندازه کافی دقیق است. با این حال، چندین هفته پس از استقرار، نتایج بسیار غیرمنتظره‌ای را مشاهده می‌کنید. با کاهش عملکرد مدل مواجه شده و بعد از بررسی‌ها متوجه می‌شوید که یک رقیب جدید اکنون خدمات بسیار مشابهی را با قیمت بسیار کمتر ارائه می‌دهد. این ویژگی در ابتدا برای مدل شما در دسترس نبود، بنابراین اطلاعات مهمی برای پیش‌بینی دقیق وجود نداشت. ۴. بایاس حمایت مالی (Sponsorship Funding or Bias)

فرض کنید یک شرکت قصد دارد با استفاده از یک مدل، شرکت‌های موفق در زمینه‌ی هوش مصنوعی در ایران را تا ۱۰ سال آینده پیش‌بینی کند و دو شرکت هوش مصنوعی اعلام کرده‌اند که حاضرند اسپانسر این پروژه بشوند. شاید آن‌ها انتظار داشته باشند در ازای اسپانسر شدن، مجموعه‌داده‌ای که از این دو شرکت به مدل خود می‌دهید، با واقعیت تفاوت داشته باشد تا در دست آورده‌های خود، اغراق کنند. چنین کاری می‌تواند باعث شود که عملکرد مدل شما، کمتر از حد مطلوب بشود. ۵. بایاس نمونه‌گیری (Sampling Bias)

این وظیفه‌ی دانشمندان داده است که مطمئن شوند نمونه‌هایی که با آن مدل می‌سازند با محیطی که قرار است مدل در آن مستقر شود، مطابقت داشته باشند. این نوع بایاس زمانی رخ می‌دهد که توزیع مجموعه‌داده‌ی مورداستفاده برای آموزش، توزیع مجموعه‌داده‌ی ورودی که مدل در هنگام استفاده عملی دریافت می‌کند را منعکس نکند. برای مثال در مجموعه‌داده‌ی خانه‌های پکن، می‌خواهیم تعداد اتاق خواب‌های هر خانه را با توجه به مشخصاتش، پیش‌بینی کنیم. ممکن است که نمونه‌برداری‌های ما از مجموعه‌داده، عمومیت نداشته باشند و بیشتر شامل خانه‌های بزرگ با اتاق خواب‌های زیاد باشند. در هنگام استقرار و استفاده از مدل، متوجه می‌شویم خطای مدل در پیش‌بینی تعداد اتاق خواب‌های خانه‌های کوچک‌تر و با اتاق خواب‌هایی کمتر، بالا رفته است. مجموعه‌داده‌ی خانه‌های پکن

این مجموعه‌داده شامل مشخصات و قیمت خانه‌های شهر پکن در چندین سال است و معمولاً از آن جهت پیش‌بینی قیمت خانه‌ها طبق مشخصات آن‌ها استفاده می‌شود. اگر کالج «تحلیل داده با پایتون» کوئرا را گذرانده باشید به خوبی با این مجموعه‌داده آشنا هستید. نسخه‌ی اصلاح شده این مجموعه‌داده را می‌توانید از این لینک دریافت کنید. در این نسخه، مقادیر عددی ستون‌ها با اسمی متناظر آن‌ها جایگزین شده تا درک بهتری از مجموعه‌داده داشته باشیم. ۶. بایاس‌های تعصی (Stereotype or Prejudice Bias)

این مورد اغلب در مجموعه‌داده‌های به دست آمده از منابع تاریخی (کتاب‌ها، آرشیو عکس و ...) و فعالیت‌های آنلاین (رسانه‌های اجتماعی، انجمن‌های آنلاین و نظرات به نشریات آنلاین) مشاهده می‌شود. فرض کنید که قصد تحلیل احساس کامنت‌های افراد بر روی یک پست تاریخی را داریم. ممکن است گروه‌های مختلف مردمی با توجه به دانسته‌ها و تعصبات خود، نظرات مختلفی بر روی این پست داده باشند. اگر ما صرفاً گروه‌های خاصی را برای تحلیل استفاده کنیم، سوگیری در مجموعه‌داده به وجود آمده است. ۷. بایاس تحریف مقدار سیستماتیک Bias (Distortion Value (Systematic

نوعی بایاس است که معمولاً با اندازه‌گیری یا مشاهدات دستگاه اتفاق می‌افتد و باعث می‌شود مدل یادگیری ماشین پس از استقرار، پیش‌بینی‌های ضعیفی انجام دهد. برای مثال، مجموعه‌داده‌ی آموزشی با استفاده از دوربینی جمع‌آوری می‌شود که رنگ‌های سفید را متمایل به زرد نشان می‌دهد. پس از آموزش و استقرار مدل، مهندسان تصمیم به استفاده از دوربین بسیار با کیفیت‌تر می‌گیرند که رنگ‌های سفید را کاملاً سفید نشان می‌دهد و آن‌ها عکس‌های تولیدی با این دوربین را به مدل می‌دهند. این قضیه در پیش‌بینی مدل تاثیر گذاشته و باعث ناکارآمدی مدل می‌شود. ۸. بایاس آزمونگر Experimenter Bias)

این بایاس هم اغلب زمانی اتفاق می‌افتد که نمونه‌ها، حاصل نتیجه‌ی یک نظرسنجی باشند. معمولاً هر نظرسنجی شامل چندین سوال است که شکل و ترتیب آن‌ها می‌توانند به صورت قابل توجهی بر نوع پاسخ‌ها تاثیر بگذارند. برای مثال، فرض کنید قصد داریم داده‌هایی را جمع‌آوری کنیم که نشان می‌دهد افراد از چه نوع پیتزایی خوش‌شان می‌آید. گزینه‌های سوال عبارتند از: پیرونی، مخصوص، مارگاریتا و مخلوط. ممکن است علایق شخصی که در نظرسنجی شرکت می‌کند در بین گزینه‌ها نباشد. پس بهتر است در این مورد، گزینه «سایر» نیز قرار داده شود تا مجموعه‌داده‌ای واقعی‌تر جمع‌آوری کنیم. به عنوان مثالی دیگر، می‌خواهیم متوجه شویم در مناطق مختلف یک کشور خاص، آیا زباله‌های بازیافتی از زباله‌های غیربازیافتی تفکیک می‌شوند یا خیر. نحوه و لحن پرسش در این مورد می‌تواند در پاسخ‌های افراد تاثیر بگذارد. برای مثال، پرسش «آیا زباله‌های بازیافتی و غیربازیافتی خود را تفکیک می‌کنید؟» می‌تواند باعث کسب پاسخ‌های صادقانه‌تری نسبت به پرسش «آیا از تفکیک کردن زباله‌های بازیافتی و غیربازیافتی طفره می‌روید؟» شود. ۹. بایاس برچسب‌گذاری Labeling Bias)

این نوع از بایاس، بیشتر به تفاوت عقاید و رفتار افراد مربوط می‌شود. برای مثال، قصد داریم مدل دسته‌بندی طراحی کنیم که به ما بگوید آیا یک فرد با توجه به سابقه شغلی، کاری و تحصیلی خود، فرد موفقی بوده است یا خیر؟ ممکن است گروهی از برچسب‌زنندگان، افراد پولدار ولی تحصیل‌نکرده را افرادی موفق تلقی کنند. در گروه مقابل، ممکن است افراد برچسب‌زننده اعتقاد داشته باشند که افراد تحصیل‌کرده ولی با درآمد متوسط، افراد موفق‌تری هستند. به همین دلیل بایستی که یک دستورالعمل که همه گروه‌های برچسب‌زننده روی آن اتفاق نظر دارند، نوشته شود تا این نوع بایاس در برچسب‌گذاری داده‌ها جلوگیری کند. قدرت پیش‌بینی پایین

نمونه‌هایی که به عنوان مجموعه‌ی آموزشی به مدل داده می‌شوند، باید حاوی اطلاعات کافی باشند تا مدل بتواند با استفاده از آن‌ها، عملیات یادگیری را به خوبی انجام دهد. برای مثال، یک شرکت بیمه به ما سفارش یک مدل یادگیری ماشین داده است و قصد دارد تمرکز تبلیغاتی خود را بر روی مشتریانی بگذارد که احتمال خرید بیمه توسط آن‌ها بیشتر از سایر مشتریان است. بنابراین ما قصد داریم مدلی بسازیم که پیش‌بینی کند آیا یک شخص، برای خود بیمه‌ی عمر خریداری می‌کند یا خیر؟ مجموعه‌داده‌ای که در اختیار ما قرار می‌گیرد، شامل ویژگی‌ها (ستون‌های) زیر هستند:

نام نام خانوادگی کد ملی شماره تلفن همراه وضعیت تأهل تعداد فرزند درآمد
در فرآیند پیش‌بینی خرید بیمه توسط یک شخص، ویژگی‌هایی مانند نام، نام خانوادگی، کد ملی و شماره

تلفن‌همراه تاثیری ندارند و صرفاً باعث پایین آوردن قدرت پیش‌بینی مدل می‌شوند. برای مثال، ممکن است مجموعه‌داده در یک مسئلهٔ خاص، گروه‌های خاص را در بر نگیرد و مدل پس از استقرار، بر روی گروه‌های خاص که مانندش را پیش از آن ندیده است، احتمالاً ضعیف عمل کند. قدیمی شدن

هنگامی که مدل ساخته شد و در مرحله‌ی استقرار قرار گرفت، اگر همه چیز خوب پیش برود، مدل ما برای مدتی به خوبی عمل می‌کند. اما پس از استقرار مدل، روش‌هایی نیز برای نظارت بر کیفیت آن در نظر خواهیم گرفت. مثلاً هنگامی که عملکرد مدل با مشکل مواجه شود، مجموعه‌ی آموزش جدیدی برای تنظیم عملکرد مدل اضافه می‌شود. سپس مدل دوباره آموزش داده شده و آماده‌ی استفاده می‌شود. برای مثال، تصور کنید مدل ما پیش‌بینی می‌کند آیا کاربر محتوای خاصی را در یک وب‌سایت دوست دارد یا خیر. با گذشت زمان و به دلیل مواردی مانند تغییر سن و یا تغییر دیدگاه افراد، رفتار برخی از کاربران ممکن است شروع به تغییر کند. برای مثال، یک کاربر تا سه سال پیش موسیقی سنتی گوش نمی‌داده، اما اکنون این سبک در مجموعه‌ی سبک‌های مورد علاقه‌ی موسیقی‌ای شخص قرار گرفته است. نمونه‌هایی که در گذشته به مجموعه‌داده‌ی آموزش اضافه شده‌اند، دیگر نمایانگر درستی از رفتار آن کاربر نیستند. پس به جای آن که به مدل کمک کنند، به عملکرد آن آسیب می‌رسانند. در این شرایط، بایستی که مجموعه‌داده‌ی آموزش به روزرسانی شود تا مدل بتواند به درستی آموزش ببیند.

در درسنامه‌ی داده‌های پرت، با چالشی به همین نام و در فصل «مهندسی ویژگی» نیز با چالش نشست داده، بیشتر آشنا خواهید شد.

۴.۳ ویژگی‌های مجموعه‌داده‌ی خوب

ویژگی‌های مجموعه‌داده‌ی خوب

در درسنامه‌ی قبل سوالاتی که قبل از جمع‌آوری مجموعه‌داده باید به آن‌ها پاسخ بدھیم را بررسی کردیم. حال در این درسنامه به سراغ بررسی ویژگی‌های یک مجموعه‌داده‌ی خوب می‌رویم. «مجموعه‌داده‌ی خوب حاوی اطلاعات است»

مجموعه‌داده‌ی مناسب اطلاعات کافی برای ساخت مدل را دارد. فرض کنید می‌خواهید مدل بسازید که مشخص کند که کاربر، کالای خاصی را خریداری می‌کند یا خیر. برای این کار هم باید اطلاعات کالا را داشته باشید و هم اطلاعات کالاهایی که کاربر قبلًا خریداری کرده است، اما اگر فقط اطلاعات آن کالای خاص و اطلاعات نام و موقعیت کاربر را داشته باشید، مدل شما برای تمام افراد یک منطقه، پیش‌بینی یکسانی خواهد داشت. اگر نمونه‌های آموزشی زیادی داشته باشید، شاید مدل بتواند جنسیت و قومیت را هم از روی نام افراد تشخیص دهد و برای مرد/زن/... و مناطق مختلف، پیش‌بینی‌های مختلفی داشته باشد. اما باز برای هر فرد، پیش‌بینی منحصر به فردی نخواهد داشت. «مجموعه‌داده‌ی خوب همه‌ی برچسب‌ها را پوشش می‌دهد»

مجموعه‌داده‌ی خوب باید چیزی که می‌خواهید مدل یاد بگیرد را به خوبی پوشش دهد. فرض کنید می‌خواهید مدل بسازید که صفحات اینترنتی را به عنوان ورودی دریافت کند و موضوع هر صفحه را مشخص کند. اگر تعداد موضوعات ممکن، ۱۰۰۰ تا باشد، آن‌گاه مجموعه‌داده‌ی ما بایستی که از هر یک از موضوعات، به اندازی کافی نمونه داشته باشد. در این صورت است که مدل می‌تواند موضوعات مختلف را از هم تفکیک کند. اگر در مجموعه‌ی آموزشی، برای یک موضوع خاص، نمونه‌ای وجود نداشته باشد و یا تعداد آن‌ها کم باشد، مدل قادر نخواهد بود رابطه‌ای بین ویژگی‌های نمونه‌ی ورودی و موضوع آن صفحه را شناسایی کند تا از آن رابطه،

برای موضوع‌بندی استفاده کند. «مجموعه‌دادهٔ خوب و رودهای واقعی را نشان می‌دهد»

مجموعه‌دادهٔ خوب باید شامل نمونه‌هایی باشد که مدل در عمل و در مسائل واقعی با آن‌ها روبرو می‌شود. فرض کنید که هدف شما، ساخت مدلی است که نوع یک ماشین موجود در عکس را شناسایی کند. اگر تصاویر شما در ساعت‌ها کاری گرفته شده باشند، در این صورت اکثر تصاویر در طول روز تهیه شده‌اند. اما وقتی که می‌خواهید در عمل از مدل استفاده کنید، این احتمال وجود دارد که تصاویری که در شب هم گرفته شده‌اند، به عنوان ورودی داده شوند. چون که مدل، این تصاویر را قبلًا ندیده است، در نتیجه روی آن‌ها به خوبی عمل نخواهد کرد. «مجموعه‌دادهٔ خوب با یاس ندارد»

مجموعه‌دادهٔ خوب سوگیری یا یاس ندارد. این ویژگی شاید مانند ویژگی قبل، به نظر برسد اما تفاوت‌هایی با آن دارد. یاس می‌تواند هم در داده‌های آموزشی و هم در داده‌های واقعی که در عمل دیده می‌شوند، وجود داشته باشد. منابع مختلف یاس را در درسنامه‌ی «چالش‌های داده» بررسی کردیم. مثلًاً، رابط کاربری هم می‌تواند یکی دیگر از منابع یاس باشد. فرض کنید که می‌خواهید محبوبیت مقالات را در یک سایت خبری تخمین بزنید. با توجه به این‌که معمولاً اخبار سیاسی در بالای سایت قرار می‌گیرند، اگر مدل شما از تعداد کلیک بر روی لینک به عنوان معیاری برای محبوبیت استفاده کند، محبوبیت اخبار سیاسی را بالا پیش‌بینی می‌کند، در حالی‌که ممکن است در واقعیت اخبار پایین سایت محبوبیت بیشتری داشته باشند. «مجموعه‌دادهٔ خوب نتیجه یک حلقه بازخورد نیست»

مجموعه‌دادهٔ خوب نتیجه‌ی یک خروجی از خود مدل نیست. در درسنامه‌ی «سؤالاتی درباره داده» با مفهوم حلقه بازخورد آشنا شدیم. برای مثال، نمی‌توان از خروجی یک مدل که جنسیت افراد را مشخص می‌کند، برای برچسب‌گذاری روی داده‌های جدید استفاده کرد. به عنوان مثالی دیگر، فرض کنید مدل شما ایمیل‌های مهم کاربر را برای او برجسته‌تر نشان می‌دهد. شما نمی‌توانید تعداد کلیک بر روی این ایمیل‌ها را به عنوان یک ویژگی نشان‌دهنده‌ی اهمیت آن ایمیل در نظر بگیرید. ممکن است کاربر به خاطر برجسته شدن آن ایمیل، روی آن کلیک کرده باشد. «مجموعه‌دادهٔ خوب دارای برچسب‌های ثابت است»

بی‌ثباتی یک مجموعه‌داده می‌تواند به دلایل زیر باشد:

مردم با ذهنیت‌های مختلف شروع به برچسب‌گذاری می‌کنند. حتی اگر همه‌ی آن‌ها از یک رویه استفاده کنند، تفسیر افراد از آن، ممکن است متفاوت باشد. تعریف برخی کلاس‌ها یا برچسب‌ها در طول زمان تغییر می‌کند. تفسیر غلط رفتار کاربر: فرض کنید که در یک سایت پخش فیلم، سیستم پیشنهادهندۀ، فیلمی را به یک کاربر پیشنهاد می‌دهد و کاربر آن را نادیده بگیرد. سیستم پیشنهادهندۀ، برچسب منفی بر روی فیلم می‌زند در حالی‌که شاید کاربر، فیلم را هم خیلی هم دوست داشته اما چون قبلًا فیلم را دیده، پیشنهاد را نادیده گرفته است.

«مجموعه‌دادهٔ خوب به اندازه کافی بزرگ است»

همان‌گونه که در درسنامه‌ی «سؤالاتی درباره داده»، گفته شد؛ بایستی که تعداد نمونه مورد نیاز برای داشتن یک مدل با حداقل کارایی موردنیاز، به روش‌هایی که توضیح داده شد، تخمین زده شود و مطمئن شوید که مجموعه‌داده، آن شرایط را داشته باشد.

۵.۴. تقسیم‌بندی مجموعه‌داده

ما در یادگیری ماشین با استفاده از مجموعه‌داده و الگوریتم‌های موجود، مدلی می‌سازیم که عملکرد قابل قبول برای مسئله‌ی مورد بحث داشته باشد. منطقی است هرچه تعداد داده‌هایی که برای آموزش مدل استفاده می‌کنیم بیشتر باشد (در صورتی که داده‌ها مناسب و بدون ایراد باشند) مدل نیز می‌تواند بیشتر از اشتباهات خود یاد بگیرد و پیش‌بینی‌های بهتری تولید کند. اما آیا باید از کل مجموعه‌داده‌ی در دسترس خود جهت آموزش مدل استفاده کنیم؟ پس چگونه مدل را بسنجیم و از عملکرد قابل قبول آن اطمینان کسب کنیم؟ به این نکته توجه داشته باشید که هدف ما این است که مدل در عمل و برای نمونه‌هایی که تاکنون ندیده است از عملکرد خوبی برخوردار باشد. بنابراین ارزیابی عملکرد آن برای نمونه‌هایی که طبق آن‌ها آموزش دیده کارآمد نخواهد بود.

بگذارید با یک سناریو پیش برویم. فرض کنید استاد درس هوش مصنوعی هستید و می‌خواهید به دانشجویان خود، مطالبی را بیاموزید. در طول ترم با ارائه‌ی مطالب و تدریس از روی کتاب و جزو، محتوای درسی را به دانشجویان منتقل می‌کنید. سپس با دادن تمرین و تکلیف، از آن‌ها می‌خواهید که مثال‌هایی را حل کرده و یادگیری خود را کامل کنند. اگر که تمرین‌ها را اشتباه حل کنند، به جزو مراجعه می‌کنند تا دوباره مطالب درسی را مطالعه و ایرادات خود را برطرف کنند.

در پایان ترم، برای اینکه بفهمید کدام دانشجوها واقعاً درس را یاد گرفته‌اند، آزمون پایان‌term برگزار می‌کنید و از آنان می‌خواهید به تعدادی سوال پاسخ بدهند. اگر به جای طرح سوالات جدید، فقط سوالات جزو یا حتی تمرین‌ها را به آن‌ها بدهید، آیا می‌توانید تشخیص دهید چه کسانی واقعاً درس را یاد گرفته‌اند؟ قطعاً نه!

در یادگیری ماشین هم به همین شکل عمل می‌کنیم. یعنی ما بخشی از مجموعه‌داده را به مدل می‌دهیم تا یاد بگیرد (کتاب و جزو)، بخشی دیگر از مجموعه‌داده را به مدل می‌دهیم تا از یادگیری آن مطمئن شویم و اگر نیاز بود، مرحله‌ی آموزش را تکرار کنیم تا مدل به خوبی یاد بگیرد (تمرین و تکلیف). بخش دیگری از مجموعه‌داده را که مدل اصلاً ندیده به عنوان داده‌های آزمون استفاده می‌کنیم (سوالات امتحانی) و مقدار خطایی که مدل در پاسخگویی با داده‌های آزمون دارد را به عنوان خطای نهایی یا نمره‌ی نهایی مدل در نظر می‌گیریم.

بنابراین مجموعه‌داده‌ی خام اولیه را باید به سه قسمت تقسیم کنیم:



مجموعه‌ی آموزش (Train Set): بخشی از مجموعه‌داده‌ی اصلی که مدل بر اساس آن یاد می‌گیرد. مجموعه‌ی اعتبارسنجی (Validation Set): بخشی از مجموعه‌داده‌ی اصلی که در فرآیند آموزش به کار نمی‌رود و فقط در فرآیند بهبود عملکرد مدل از آن استفاده می‌کنیم. مجموعه‌ی آزمون (Test Set): بخشی از مجموعه‌داده‌ی اصلی که عملکرد مدل بر اساس آن آزموده و به عنوان دقت نهایی گزارش می‌شود. از این داده‌ها نباید در هنگام ساخت مدل استفاده شود زیرا که این داده‌ها حکم نمونه‌های دنیای واقعی را دارند.

نکته

ممکن است در بسیاری از منابع مشاهده کنید که مجموعه‌داده را صرفاً به دو بخش تقسیم می‌کنند. یعنی

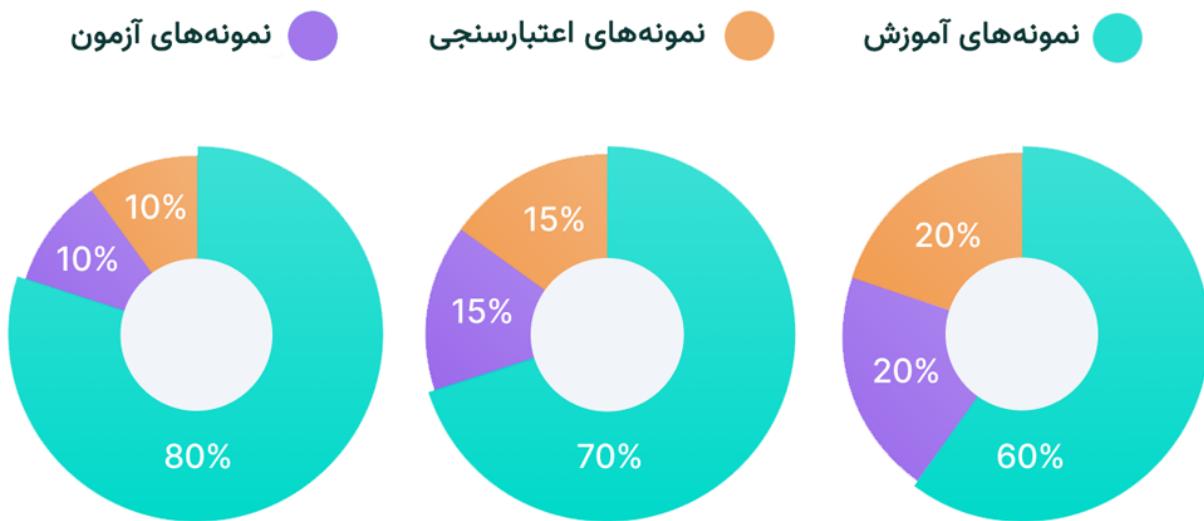
بخشی را به عنوان مجموعه‌ی آموزش جهت یادگیری مدل استفاده می‌کنند و بخشی از مجموعه‌داده را نیز جهت بررسی عملکرد مدل کنار می‌گذارند. شما نیز ممکن است در بسیاری از موقع و به دلایل مختلف مثلاً کمبود داده‌ها همین کار را انجام دهید و از مجموعه‌ی آزمون صرف‌نظر کنید. با این حال آن‌ها اشتباها نمونه‌های کنار گذاشته شده که جهت بررسی و بهبود عملکرد مدل مورد استفاده قرار می‌گیرند را مجموعه‌ی آزمون (Test Set) می‌نامند، در صورتی‌که طبق تعریف علمی این نمونه‌ها همان مجموعه‌ی اعتبارسنجی (Validation Set) هستند که معرفی کردیم. البته جهت انتخاب نمونه‌های اعتبارسنجی روش‌های گوناگونی وجود دارد که در فصل‌های مدل‌سازی و در درسنامه‌ی «کراس‌ولیدیشن» با آن‌ها آشنا خواهید شد.

برای تقسیم‌بندی مجموعه‌داده اصولی وجود دارند که در لیست زیر، آن‌ها را معرفی و توضیح می‌دهیم:

اولویت تقسیم‌بندی

تقسیم‌بندی باید روی مجموعه‌داده‌ی خام انجام شود و پیش از آن نباید هیچ‌گونه پیش‌پردازشی صورت بگیرد تا از نشت داده جلوگیری شود. در ادامه‌ی درسنامه بیشتر با این مفهوم آشنا خواهیم شد. تعداد نمونه‌های هر مجموعه

یک پرسش که همین ابتدا پیش می‌آید این است که چند درصد از مجموعه‌داده‌ی اصلی را به عنوان مجموعه‌ی آموزش، اعتبارسنجی یا آزمون در نظر بگیریم؟ هیچ درصد مشخص وجود ندارد و به عوامل متعددی همچون مسئله‌ی شما، مدل یادگیری ماشین انتخابی و همچنین خود مجموعه‌داده بستگی دارد. با این حال برخی از درصدهایی که معمولاً انتخاب می‌شود در شکل زیر نمایش داده شده است.



البته توجه داشته باشید که اگر مجموعه‌داده‌ی در دسترس شما بزرگ باشد و مثلاً شامل یک میلیون نمونه باشد، شاید در نظر گرفتن ۱۰ درصد نمونه‌ها برای اعتبارسنجی و ۱۰ درصد دیگر برای آزمون بسیار زیاد باشد زیرا که در ۱۰ درصد چنین مجموعه‌داده‌ای یعنی صد هزار نمونه! به‌طور کلی اگر حدود هزار تا پنجاه هزار نمونه داشتید در نظر گرفتن ۸۰ درصد داده‌ها برای آموزش انتخاب مناسبی خواهد بود. هرچه تعداد نمونه‌ها بیشتر شود می‌توانید از درصد مجموعه‌های اعتبارسنجی و آزمون کاسته و به درصد مجموعه‌ی آموزش اضافه کنید. مثلاً اگر یک میلیون نمونه در اختیار داشتید حتی ممکن است ۹۹ درصدشان را برای آموزش در نظر بگیرید. اما اگر مجموعه‌داده‌ی شما کوچک‌تر از این‌ها باشد بهتر است خودتان عوامل مختلف را تحلیل کرده و تصمیم بگیرید. تقسیم‌بندی تصادفی (Shuffling)

در هنگام تقسیم‌بندی این سوال پیش می‌آید که کدام بخش از مجموعه‌ی داده‌ی اصلی را باید به عنوان

مجموعه‌ی آموزش، اعتبارسنجی یا آزمون در نظر بگیریم. مثلًا X نمونه‌ی اول را به عنوان مجموعه‌ی آموزشی، Y نمونه‌ی بعدی را به عنوان مجموعه‌ی اعتبارسنجی و Z نمونه‌ی باقیمانده بعد از آن را به عنوان مجموعه‌ی آزمون در نظر بگیریم؟

به مثال خود باز گردیم. فرض کنید شما به عنوان استاد ۱۰۰ پرسش در اختیار دارید که به ترتیب ۸۰ پرسش اول از مبحث A و ۲۰ پرسش باقیمانده از مبحث B باشد. در این صورت اگر قصد داشته باشید تقسیم‌بندی را به ترتیب انجام دهید و مثلًا ۶۰ نمونه‌ی اول را به عنوان مجموعه‌ی آموزشی، ۲۰ نمونه‌ی بعدی را به عنوان مجموعه‌ی اعتبارسنجی و ۲۰ نمونه‌ی آخر را مجموعه‌ی آزمون در نظر بگیرید بسیار بی‌رحمانه‌ست! زیرا که ۶۰ پرسش در کلاس خود به شاگردان آموخته‌اید که همگی از مبحث A بوده و ۲۰ تمرینی هم که به آن‌ها دادید باز همگی از مبحث A هستند، اما ناگهان در امتحان از مبحث B سوال داده‌اید. مشخص است که آن شاگرد (مدل) عملکرد خوبی از خود نشان نخواهد داد.

اما اگر به‌سادگی پیش از تقسیم‌بندی یک‌بار مجموعه‌داده را بُر می‌زدید (Shuffling) و ترتیب نمونه‌ها را تصادفًا به هم می‌ریختید دیگر با چنین چالشی مواجه نمی‌شید.

با این حال توجه داشته باشید که در برخی از مجموعه‌داده‌ها ممکن است ترتیب معناداری بین نمونه‌ها وجود داشته باشد، که در این صورت باید از بُر زدن مجموعه خودداری کنید. این نکته خصوصاً در مجموعه‌داده‌های سری‌های زمانی Series که نمونه‌ها به ترتیب زمان چیده می‌شوند برقرار است. مثلًا در تخمین قیمت سهام بورس، اگر پیش از تقسیم‌بندی، بُر بزنیم، روند صعودی یا نزولی بازار را به هم می‌ریزیم. به عنوان مثال دیگر اگر قصد پیش‌بینی قیمت بیت‌کوین را داشته باشیم و مجموعه‌داده‌ی ما قیمت هفتگی این ارز در چند سال اخیر را شامل شود، به هم زدن این ترتیب امری منطقی نخواهد بود. طبقه‌بندی Stratification

در بحث قبلی، این چالش که برخی از نمونه‌های هم‌نوع (مثلًا هم‌برچسب) فقط در یکی از مجموعه‌ها حاضر شوند را بررسی کردیم. اما چگونه می‌توانیم اطمینان کسب کنیم که از هر نوع نمونه‌ای در تمام مجموعه‌ها داده داشته باشیم؟ اینجاست که تکنیک طبقه‌بندی Stratification به کمک مان می‌آید.

فرض کنید که ۱۰ نمونه داریم که هرکدام طبق تصویر زیر دارای برچسب ۰ یا ۱ است. از طرف دیگر قصد داریم که ۸۰ درصد داده‌ها (۸ نمونه) را به عنوان مجموعه‌ی آموزش و ۲۰ درصد باقیمانده (۲ نمونه) را به عنوان مجموعه‌ی اعتبارسنجی تفکیک کنیم.

A	B	C	D	E	F	G	H	I	J
0	0	1	1	0	1	0	0	1	1

ابتدا نسبت هر برچسب به تعداد کل نمونه‌ها را محاسبه می‌کنیم. در مثال ما هر برچسب ۵ بار تکرار شده و مجموعاً ۱۰ نمونه داریم، پس فرکانس هر برچسب ۵.۰ یا ۵۰ درصد است. بنابراین اگر بخواهیم نسبت‌ها در طول تقسیم‌بندی ما نیز رعایت شود در مجموعه‌ی آموزشی باید ۸ ضرب در ۵.۰ یعنی ۴ نمونه از هر کلاس داشته باشیم. در مجموعه‌ی اعتبارسنجی نیز باید ۲ ضرب در ۵.۰ یعنی ۱ نمونه از هر کلاس وجود داشته باشد.

A	B	E	G	H
0	0	0	0	0

C	D	F	I	J
1	1	1	1	1

اکنون کافیست مثل تصویر بالا نمونه‌های هر برچسب را گرفته و ۴ نمونه از آن را به صورت تصادفی انتخاب کرده و به مجموعه‌ی آموزش اضافه کنیم و ۱ نمونه را نیز به مجموعه‌ی اعتبارسنجی اضافه کنیم.

C	F	E	I	A	B	J	G
1	1	0	1	0	0	1	0

D	H
1	0

توزیع یکسان

توزیع مجموعه‌های اعتبارسنجی و آزمون، بایستی یکسان باشند، چون می‌خواهیم که مجموعه‌ی اعتبارسنجی، نماینده‌ای از مجموعه‌ی آزمون که قرار است مدل در هنگام استفاده عملی با آن‌ها روبرو شود، باشند. اگر که توزیع مجموعه‌ی اعتبارسنجی و آزمون متفاوت باشند، نمی‌توان انتظار داشت که عملکرد مدل روی مجموعه‌ی آزمون، نشان‌دهنده‌ی عملکرد آن در هنگام استفاده عملی باشند. به عنوان یک مثال شهودی‌تر، یک استاد نباید سوالات تمرین و امتحانش خیلی زیاد با هم فرق داشته باشند (مگر آن‌که قصد آزار داشته باشد). در غیر این صورت، توزیع نمرات تمرین و امتحان با هم کاملاً متفاوت خواهند بود و این نشان می‌دهد که تمارین، آمادگی کافی را در دانشجویان برای امتحان ایجاد نکرده و بایستی که مورد بازبینی قرار بگیرند. جلوگیری از نشت داده

فرض کنید در مجموعه‌دادهی خانه‌های شهر پکن، اطلاعات معامله‌ی یک خانه، دو بار تکرار شده باشد و دفعه‌ی دوم انتشار مربوط به یک آگهی است که چند ماه بعد از اولین آگهی منتشر شده است. یعنی در عرض چند ماه، دوباره خانه با قیمت متفاوتی به فروش رسیده باشد. موقع تقسیم‌بندی باید مراقب باشیم که هر دو نمونه‌ی این خانه، در یکی از مجموعه‌ها قرار بگیرد (یا هر دو در آموزش، یا در اعتبارسنجی یا در آزمون). اگر این دو نمونه حین تقسیم‌بندی داده‌ها از هم جدا شوند، اصطلاحاً «نشت داده» رخ می‌دهد. فرض کنید بدون عمل به این اصل، تقسیم‌بندی صورت بگیرد. نمونه‌ی اول در مجموعه‌ی آموزش و نمونه‌ی دوم در مجموعه‌ی آزمون قرار بگیرد. حال مدل از نمونه‌ی اول یاد می‌گیرد. موقعی که از مجموعه‌ی آزمون استفاده می‌کنیم، مدل متوجه می‌شود که این خانه را قبل‌آیده دیده است، بنابراین با توجه به خانه، قیمت‌گذاری می‌کند، نه با توجه به ویژگی‌ها. در نتیجه در صورت نشت داده، مدل به خوبی آموزش نخواهد دید و نمی‌تواند در عمل به درستی عمل کند. البته ما اطمینان می‌دهیم که این اتفاق در مجموعه‌دادهی خانه‌های پکن اتفاق نیوفتداده است؛ یعنی هر سطر از این مجموعه داده، به یک خانه‌ی منحصر به‌فرد اشاره می‌کند. پیش‌پردازش قبل از تقسیم‌بندی یا بعد از آن؟

پیش‌پردازش داده‌ها بسته به نوع آن‌ها می‌تواند قبل یا بعد از تقسیم‌بندی داده‌ها به مجموعه‌های آموزش، اعتبارسنجی و آزمون انجام شود. نکته‌ی مهم در این موضوع جلوگیری از نشت اطلاعات (Information Leakage) از مجموعه آزمون یا اعتبارسنجی به مجموعه آموزش است.

به طور کلی می‌توان مراحل پیش‌پردازش را به دو دسته تقسیم کرد: ۱) پیش‌پردازش‌هایی که مستقل از داده و بدون یادگیری پارامتر انجام می‌شوند

این مراحل معمولاً می‌توانند قبل از تقسیم‌بندی داده‌ها انجام شوند، زیرا در آن‌ها از اطلاعات آماری یا ساختاری کل داده استفاده نمی‌شود و تنها تغییرات ثابت و عمومی اعمال می‌گردد.

نمونه‌ها:

حذف ستون‌های غیرضروری و از پیش مشخص (مانند شناسه‌ها)

اصلاح نوع داده‌ها (مثلًا تبدیل رشته به عدد)

حذف رکوردهای کاملاً نامعتبر (مانند داده‌هایی که برچسب ندارند)

۲) پیش‌پردازش‌هایی که وابسته به داده هستند و از داده، پارامتر استخراج می‌کنند

این مراحل باید پس از تقسیم‌بندی داده‌ها انجام شوند و تنها روی مجموعه آموزش (Train) تنظیم (Fit) شوند. سپس پارامترهای به دست آمده باید برای اعمال همان تبدیل روی مجموعه‌های اعتبارسنجی و آزمون استفاده شود. این کار از نشت اطلاعات جلوگیری کرده و باعث می‌شود ارزیابی مدل واقع‌بینانه‌تر باشد.

نمونه‌ها:

نرمال‌سازی و استانداردسازی (محاسبه میانگین و انحراف معیار)

پر کردن داده‌های مفقود (Imputation)

انتخاب ویژگی (Feature Selection)

کاهش بعد مانند PCA

حذف داده‌های پرت در صورتی که معیار تشخیص پرت از داده استخراج شود

در نتیجه، هر مرحله‌ای که در آن از اطلاعات آماری یا الگوهای موجود در داده برای تنظیم پارامتر استفاده شود، باید تنها با داده‌های آموزشی انجام شود تا از ایجاد خطای ارزیابی و نشت اطلاعات جلوگیری گردد. پیاده‌سازی در learn scikit

به کمک کتابخانه learn scikit می‌توانید به راحتی مجموعه‌داده‌های خود را تقسیم‌بندی کرده و از تکنیک‌های رایجی نظیر بُر زدن یا طبقه‌بندی نیز استفاده کنید.

فرض کنید که ویژگی‌های نمونه‌ها را در متغیر X و خروجی‌ها (مثل برچسب‌ها) را در متغیر y ذخیره کرده باشید. آن‌گاه کافیست ازتابع split test train استفاده کنید تا این مجموعه را با درصد دلخواه شما به دو بخش تقسیم کند. آرگومان size test را می‌توانید معادل درصد نمونه‌های آزمون (float) یا تعداد آن‌ها (int) تنظیم کنید. همچنانی اگر قصد بُر زدن مجموعه را دارید می‌توانید آرگومان shuffle را True کنید. اگر هم می‌خواهید از تکنیک طبقه‌بندی استفاده کنید می‌توانید آرگومان stratify را معادل لیست خروجی‌ها یعنی y تنظیم کنید.

```
1 from sklearn.model_selection import train_test_split
2 X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2,
                                                       shuffle=True)
```

توجه داشته باشید که با کد بالا، مجموعه‌داده به دو بخش مجموعه‌های آموزش و آزمون تقسیم شد. اگر

قصد تولید مجموعه‌ی اعتبارسنجی هم دارید می‌توانید مثل کد قبل عمل کرده و این بار X و y train را به دو مجموعه‌ی آموزش و اعتبارسنجی تقسیم کنید.

```
1 X_train, X_val, y_train, y_val = train_test_split(X_train, y_train, test_size  
=0.25, shuffle=True)
```

توجه داشته باشید اگر قصد دارید هر بار که این کد را اجرا می‌کنید عملیات انتخاب تصادفی شما به شکل مشابه عمل کند و همواره خروجی یکسانی تولید شود می‌توانید آرگومان state random را نیز معادل یک عدد دلخواه قرار دهید.

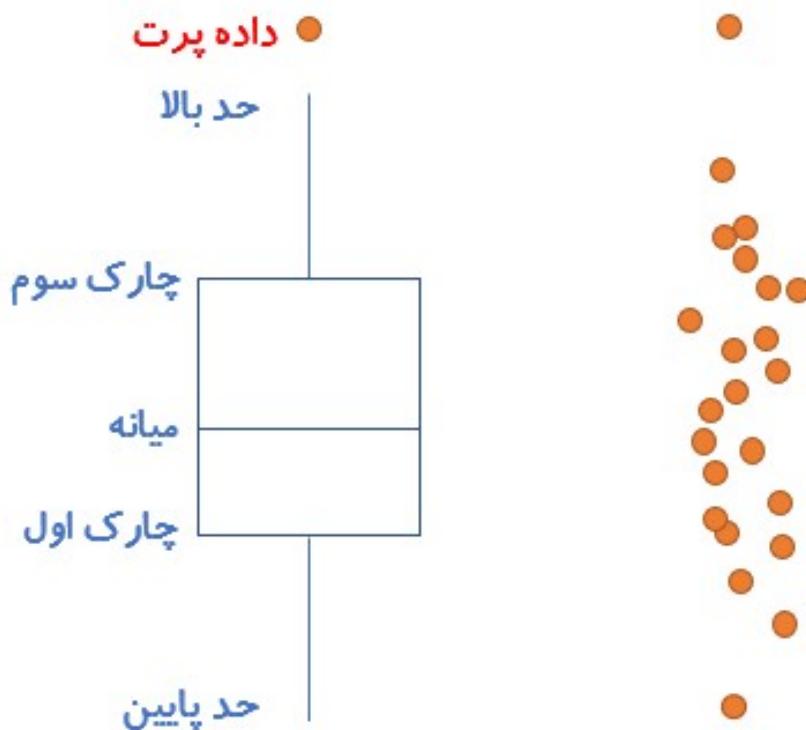
۶.۳ داده‌های پرت

داده‌ی پرت، (Outlier) نمونه‌ای است که با اکثر نمونه‌های مجموعه‌داده متفاوت به نظر می‌رسد. مجموعه‌داده‌ی شما می‌تواند به دو علت زیر شامل داده‌ی پرت باشد. ۱. اشتباه در فرآیند جمع‌آوری و ثبت داده:

این احتمال وجود دارد که در فرآیند جمع‌آوری داده، مقادیری به اشتباه توسط نیروی انسانی (خطای تایپی) و یا سامانه‌ی کامپیوتری در فایل مجموعه‌داده ثبت شده باشند. به عنوان مثال، فرض کنید که در مجموعه‌داده خانه‌های شهر پکن، کاربری که در حال ثبت «قیمت هر متر مربع یک خانه» بوده است، به اشتباه قیمت را به همراه یک صفر اضافه ثبت کند! ۲. توزیع واقعی داده‌ها:

اگر که داده‌های پرت، به خاطر «اشتباه در فرآیند جمع‌آوری و ثبت داده» نباشند، به این معنیست که توزیع واقعی داده‌ها به این شکل است. به مثال مجموعه‌داده خانه‌های شهر پکن برگردیم، شاید تعدادی خانه‌ی لوکس و اشرافی در مناطق اعیان‌نشین شهر پکن وجود داشته باشند. طبیعتاً انتظار می‌رود که «قیمت هر متر مربع یک خانه» آن‌ها، با بقیه خانه‌ها تفاوت چشم‌گیری داشته باشد. شناسایی داده‌های پرت

ابتدا، بایستی اقدام به شناسایی داده‌های پرت در مجموعه‌داده خود کنیم. به عنوان مثال، در مجموعه‌داده خانه‌های شهر پکن، به ستون «قیمت هر متر مربع یک خانه» که ستون هدف ماست، نگاه کنیم. روش‌های مختلفی برای شناسایی داده‌های پرت در این ستون وجود دارد که البته فعلاً به خاطر نیاز به پیش‌زمینه‌های مدل‌سازی، این روش‌ها مورد بحث قرار نمی‌گیرند. یکی از ساده‌ترین روش‌های شناسایی داده‌های پرت که نیاز به پیش‌زمینه‌ی مدل‌سازی ندارد، استفاده از روش، «نمودار جعبه‌ای» است که یکی از روش‌های قدیمی برای نمایش توزیع داده و شناسایی داده‌های پرت به شمار می‌رود. احتمالاً نحوه‌ی رسم دستی آن را از دوران تحصیل خود به خاطر دارید. برای رسم این نمودار به چارک اول و سوم و میانه داده نیاز است:

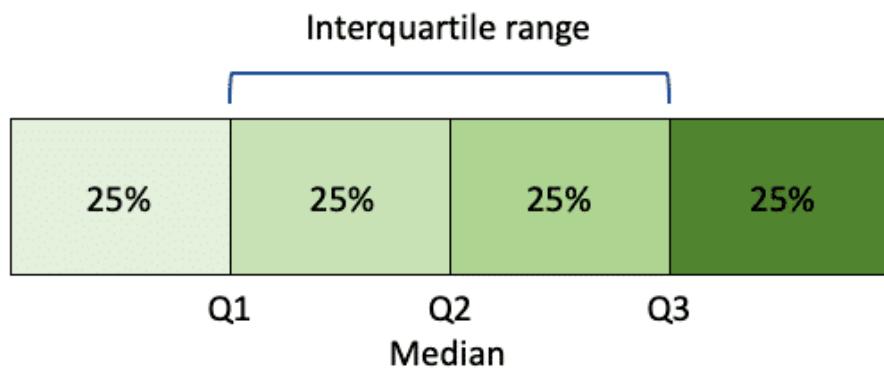


چندک‌ها و چارک‌ها

لازم است در اینجا توضیح مختصری درباره چندک‌ها بدهیم. فرض کنید مقادیری از یک کمیت را به عنوان داده در اختیار داریم. اگر این مقادیر را در یک آرایه‌ی مرتب داشته باشیم، می‌توانیم با برش آرایه، آن را به تعدادی قطعه با اندازه‌ی برابر تقسیم کنیم. به مقادیر متناظر با این برش‌ها چندک (quantile) می‌گوییم که متناسب با تعداد قطعه‌ها نام‌های مختلفی دارد. مثلاً اگر تعداد قطعه‌ها ۱۰ باشد، به مقادیر برش دهک (decile) می‌گوییم (احتمالاً دهک‌های درآمدی به گوش‌تان خورده است!). یا اگر تعداد قطعه‌ها برابر با ۴ باشد، سه برش داریم که به ترتیب چارک اول، چارک دوم و چارک سوم کوچکتر و یک چهارم آن‌ها از آن بزرگ‌ترند. به که از تعریف بر می‌آید، سه چهارم مقادیرمان از چارک سوم کوچکتر و یک چهارم آن‌ها از آن بزرگ‌ترند. به چارک دوم که همان مقدار میانی آرایه است، میانه (median) هم می‌گوییم. علاوه بر این percentile نیز به معنی صدک است. به عنوان مثال صدک ۷۵م همان برشیست که ۷۵ درصد مقادیر از آن کوچک‌ترند و ۲۵ درصد مقادیر از آن بزرگ‌ترند. پس صدک ۷۵م همان چارک سوم است. این نکته را هم باید یادآور شویم که اگر برش روی یک خانه آرایه بیفتند برابر با مقدار آن خانه است اما اگر بین دو خانه بیفتند برابر با میانگین آن‌ها در نظر گرفته می‌شود. برای مثال میانه‌ی آرایه $[10, 7, 1]$ برابر با ۷ است و میانه‌ی آرایه $[10, 10, 3, 4, 1]$ برابر با ۵.۳ است.

IQR معیار

اگر دوره‌ی «تحلیل داده با پایتون» کوئرا را گذرانده باشید حتماً با این معیار آشنا هستید. معیار IQR (in-range terquartile) فاصله معمولاً در عددی ضرب شده تا فاصله‌ی داده‌های قابل قبول از چارک‌ها را مشخص کند. به طورکلی می‌توانیم این عدد را معادل ۱.۵ فرض کنیم.



با استفاده از این روش، چارکهای اول (Q_1)، دوم یا میانه (Q_2) و سوم (Q_3) «قیمت هر متر مربع خانه» (ستون price)، محاسبه می‌شوند. سپس، فاصله‌ی بین $3Q_1$ و $3Q_3$ محاسبه می‌گردند ($IQR = Q_3 - Q_1$). با توجه به این که داده‌های پرت، نمونه‌هایی هستند که از نظر عددی با بقیه نمونه‌ها فاصله دارند، همانند کد زیر، نمونه‌هایی که «قیمت هر متر مربع خانه» آنها، کوچک‌تر از $Q_1 - 1.5 * IQR$ و یا بزرگ‌تر از $Q_3 + 1.5 * IQR$ است، به عنوان داده پرت، در نظر گرفته می‌شوند.

۷.۳ مقادیر گم شده

[متن شما]

۸.۳ مجموعه داده نامتوازن

[متن شما]

فصل ۴

مهندسی ویژگی

۱.۴ اهمیت

[متن شما]

۲.۴ مشخصات ویژگی خوب

[متن شما]

۳.۴ ویژگی‌های دسته‌ای

[متن شما]

۴.۴ مقادیر گم‌شده در ویژگی‌های دسته‌ای

[متن شما]

۵.۴ ویژگی‌های تقویمی

[متن شما]

۶.۴ سنتز ویژگی

[متن شما]

۷.۴ تغییر مقیاس ویژگی

[متن شما]

۸.۴ نشت داده

[متن شما]

۹.۴ فوت و فن‌های مهندسی ویژگی

[متن شما]

۱۰.۴ کاهش ابعاد

[متن شما]

۱۱.۴ انتخاب ویژگی

[متن شما]

۱۲.۴ خط لوله

[متن شما]

فصل ۵

رگرسیون

۱.۵ اهداف فصل

[متن شما]

۲.۵ مقدمه

[متن شما]

۳.۵ مدل چیست؟

[متن شما]

۴.۵ تخمین، تابع هزینه و بهینه‌سازی

[متن شما]

۵.۵ رگرسیون خطی

[متن شما]

۶.۵ ارزیابی

[متن شما]

۷.۵ رگرسیون چندجمله‌ای

[متن شما]

۸.۵ عمومیت

[متن شما]

۹.۵ رگولاریزیشن

[متن شما]

فصل ٦

دسته‌بندی

١.٦ مقدمه

[متن شما]

٢.٦ رگرسیون لجستیک

[متن شما]

٣.٦ ارزیابی - قسمت اول

[متن شما]

٤.٦ ارزیابی - قسمت دوم

[متن شما]

٥.٦ کراس ولیدیشن

[متن شما]

۶.۶ نزدیکترین-k همسایه

[متن شما]

۷.۶ بیز ساده‌لوحانه

[متن شما]

۸.۶ ماشین بردار پشتیبان

[متن شما]

۹.۶ هایپرپارامترها

[متن شما]

۱۰.۶ آشنایی با کتابخانه‌ی OH

[متن شما]

۱۱.۶ درخت تصمیم

[متن شما]

۱۲.۶ فوت و فن درخت تصمیم

[متن شما]

۱۳.۶ بیشبرازش درخت تصمیم

[متن شما]

فصل ٧

یادگیری تجمعی

١.٧ اهداف فصل

[متن شما]

٢.٧ مقدمه

[متن شما]

٣.٧ جنگل تصادفی

[متن شما]

٤.٧ الگوریتم AdaBoost

[متن شما]

٥.٧ الگوریتم Boosting Gradient

[متن شما]

٦.٧ الگوريتم XGboost

[متن شما]

٧.٧ روش Stacking

[متن شما]

فصل ۸

پروژه اول

۱.۸ مقدمه

[متن شما]

۲.۸ یادداشت‌ها و راه حل

[متن شما]

فصل ۹

شبکه عصبی

۱.۹ اهداف فصل

[متن شما]

۲.۹ پرسپترون

[متن شما]

۳.۹ آموزش پرسپترون

[متن شما]

۴.۹ پرسپترون چندلایه

[متن شما]

۵.۹ عمومیت

[متن شما]

فصل ۱۰

یادگیری ناظارت‌نشده

۱.۱۰ مقدمه

[متن شما]

۲.۱۰ الگوریتم PCA

[متن شما]

۳.۱۰ الگوریتم t-SNE

[متن شما]

۴.۱۰ خوشه‌بندی با k-means

[متن شما]

۵.۱۰ خوشه‌بندی با k-modes

[متن شما]

۶.۱۰ خوشبندی با k-prototype

[متن شما]

فصل ۱۱

پروژه دوم

۱.۱۱ اهداف فصل

[متن شما]

۲.۱۱ تعبیهی متن

[متن شما]

۳.۱۱ فاصله‌ی ویرایش

[متن شما]

۴.۱۱ معیار شباهت RBO

[متن شما]

فصل ۱۲

بیشتر بدانید

۱.۱۲ نمونه‌کاهی با NearMiss

[متن شما]

۲.۱۲ نمونه‌افزایی با SMOTE

[متن شما]

۳.۱۲ درخت رگرسیون

[متن شما]