

See discussions, stats, and author profiles for this publication at: <https://www.researchgate.net/publication/363319688>

كتاب التعمق في التعلم العميق / الجزء الاول / الاساسيات والمقدمات

Book · September 2022

CITATIONS

0

READS

3,290

1 author:



Alaa Taima Albu-Salih
University of Al-Qadisyah

77 PUBLICATIONS 147 CITATIONS

SEE PROFILE

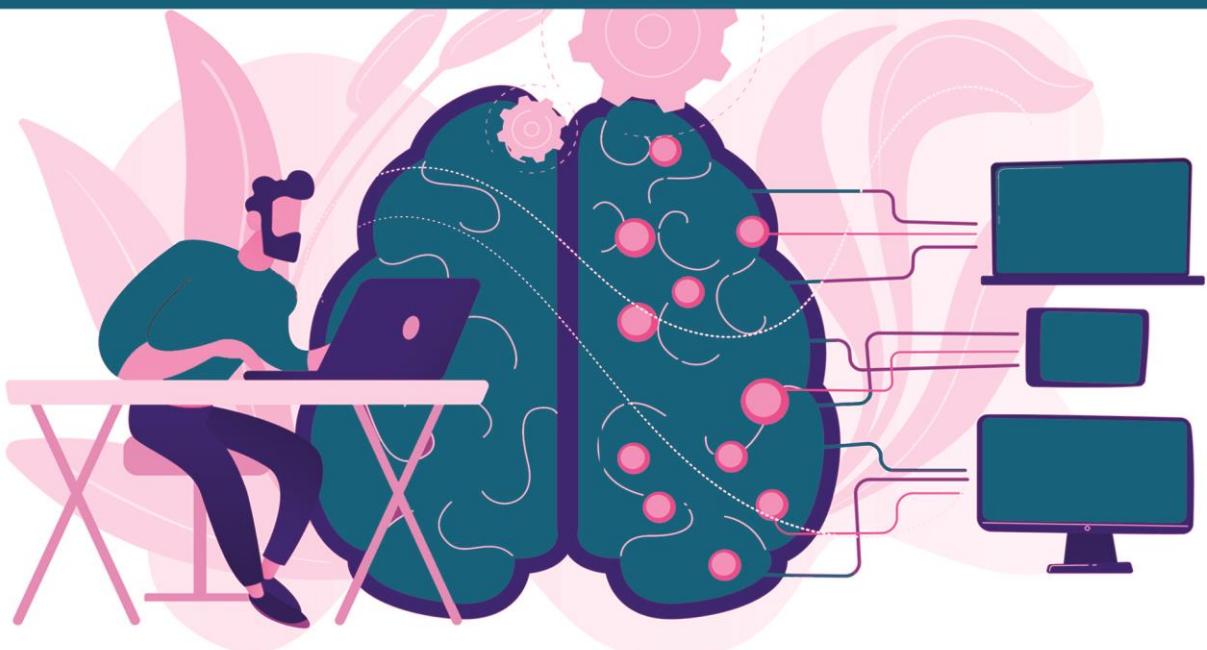
$2\arctg x - x = 0, I = (1, 10)$
 $\int_{-\frac{\pi}{2}}^{\frac{\pi}{2}} \sin^4 x \cdot \cos^3 x dx$
 $\cos^2 \alpha + \cos^2 \beta + \cos^2 \gamma = 1$
 $\frac{\partial z}{\partial x} = 2; \frac{\partial z}{\partial y} = 0 \quad \vec{z} = (F_x, F_y, F_z)$
 $\frac{x^2}{a^2} + \frac{y^2}{b^2} + \frac{z^2}{c^2} = 0$
 $\sin 2x = 2 \sin x \cdot \cos x$
 $|z| = \sqrt{a^2 + b^2}$
 $\delta(p_2) = \sqrt{0.16}$
 $a^2 + b^2 = c^2$
 $\alpha, \beta, \gamma \in C$
 $f(x) = 2^{-x} + 1, \epsilon = 0.005$
 $\lim_{x \rightarrow 0} \frac{e^{2x} - 1}{5x} = \frac{2}{5}$
 $bz + b\beta \neq 0, \beta \neq 0$
 $\lambda_1 = \sqrt{14}$
 $\lambda_2 = i\sqrt{14}$
 $\sin(x+y) = \sin x \cos y + \cos x \sin y$
 $y' = \frac{\sqrt{y}}{x+2}, y(0) = 1$
 $\cos \rho = \frac{(1,0), (\frac{1}{2}\sqrt{3}, \frac{1}{2})}{\sqrt{\frac{1}{12} + \frac{1}{48}}}$
 $\frac{\sin x}{x} \leq \frac{x}{x} = 1$
 $A + B + C = 8$
 $-3A - 7B + 2C = -10, 3$
 $-18A + 6B - 3C = 15$
 $n \rightarrow \infty \quad \frac{1}{3} h^2 + 2n-1$
 $\frac{a}{\sin \alpha} = \frac{b}{\sin \beta} = \frac{c}{\sin \gamma}$
 $y = \sqrt[3]{x+1}; x = \operatorname{tg} t$
 $x_t = \sqrt[3]{x_0 x_2}$
 $\cos 2x = \cos^2 x - \sin^2 x$
 $\sin^2 x + \cos^2 x = 1$
 $\int_R(x, y) \frac{\sqrt{a+b}}{c+d} dx$
 $\eta_1 = \lambda_1^2 - 3\lambda_1 + 1 + 0$
 $\frac{b^2}{a^2} = \frac{c}{c_1}, \frac{c_1}{c_2}$
 $\frac{a^2}{c_1} = \frac{c}{c_2}$

جامعة العلم والتكنولوجيا

الجزء الأول: الابيات والمقادمات

تأليف: أستون زانغ وآخرون

ترجمة: د. علاء طعيمة



بِهِ تَعَالَى

التفوق فـي التعلم العميـق

الأسباب والمقـدـرات

تألـيف:

آسـتون زـانـخ وـآخـرون

تـرـجمـة:

دـ. عـلـاء طـعـيمـة

مقدمة المترجم

على مدى السنوات القليلة الماضية، طور فريق من علماء أمازون كتاباً "Dive into Deep Learning" يكتسب شعبية بين الطلاب والمطورين الذين ينجذبون إلى مجال التعلم العميق المزدهر، وهو مجموعة فرعية من التعلم الآلي تركز على الشبكات العصبية الاصطناعية واسعة النطاق.

عند انتهاءي من قراءه هذا الكتاب، احببت ان اترجم هذا الكتاب وأشاركم هذه الترجمة لان هناك عدد من الأشياء الرائعة حول هذا الكتاب وأكثر ما يعجبني هو أنه يغطي كل مجالات التعلم العميق تقريباً مثلاً للمبتدئين هناك فصول مثل الشبكات العصبية والبيرسيپترون متعدد الطبقات والانحدار والتصنيف بالإضافة الى المفاهيم الأساسية من الجبر الخطي، وحساب التفاضل والتكامل، والاحتمال الى فصول متقدمة مثل الشبكات العصبية الالتفافية CNN، تم تضمين الشبكات العصبية المستكورة RNN والرؤبة الحاسوبية CV ومعالجة اللغات الطبيعية NLP أيضاً.

هذا كتاب تفاعلي مفتوح المصدر مقدم في شكل فريد يدمج النص والرياضيات وال코드، ويدعم الآن أطر برمجة PyTorchy TensorFlow وApache MXNet، والتي تمت صياغتها بالكامل من خلال Jupyter Notebook.

يمكن تقسيم الكتاب إلى ثلاثة أجزاء تقريباً، لقد قمنافي الوقت الحالي بترجمة الجزء الأول والذي يشمل الأساسيات والمقدمات وان شاء الله في المستقبل القريب سنتقوم بنشر الجزء الثاني والذي يشمل التقنيات الحديثة للتعلم العميق وبعده الجزء الثالث والذي يشمل مواضيع مثل الرؤبة الحاسوبية ومعالجة اللغات الطبيعية.

لقد اخترت كتاب "Dive into Deep Learning" لما رأيته من جودة هذا الكتاب، وللمنهجية التي اتبعها المؤلفون في ترتيبه وبساطة شرحه. لقد حاولت قدر المستطاع ان اخرج بترجمة ذات جودة عالية، ومع هذا يبقى عملاً بشعرياً يحتمل النقص، فإذا كان لديك أي ملاحظات حول هذا الكتاب، فلا تتردد بمراسلتنا عبر بريدينا الالكتروني alaa.taima@qu.edu.iq.

نأمل ان يساعد هذا الكتاب كل من يريد ان يدخل في مجال التعلم العميق ومساعدة القارئ العربي على تعلم هذا المجال. اسأل الله التوفيق في هذا العمل لأثراء المحتوى العربي الذي يفتقر أشد الافتقار إلى محتوى جيد ورصين في مجال الذكاء الاصطناعي وتعلم الآلة والتعلم العميق. ونرجو لك الاستمتاع مع التعلم العميق ولا تنسونا من صالح الدعاء.

المقدمة

قبل بضع سنوات فقط، لم يكن هناك جحافل من علماء التعلم العميق deep learning يطوروون منتجات وخدمات ذكية في الشركات الكبرى والشركات الناشئة. عندما دخلنا هذا المجال، لم يكن التعلم الآلي machine learning يتتصدر عناوين الصحف اليومية. لم يكن لدى والدينا أي فكرة عن ماهية التعلم الآلي، ناهيك عن سبب تفضيلنا له على مهنته في الطب أو القانون. كان التعلم الآلي تخصصاً أكاديمياً في السماء الزرقاء اقتصرت أهميته الصناعية على مجموعة ضيقة من تطبيقات العالم الحقيقي، بما في ذلك التعرف على الكلام speech recognition والرؤية الحاسوبية computer vision. علاوة على ذلك، تطلب العديد من هذه التطبيقات قدرًا كبيرًا من المعرفة بال مجال لدرجة أنه غالباً ما كان يُنظر إليها على أنها مناطق منفصلة تماماً كأن التعلم الآلي مكوناً صغيراً لها. في ذلك الوقت، كانت الشبكات العصبية neural networks – أسلاف أساليب التعلم العميق التي نركز عليها في هذا الكتاب – تعتبر بشكل عام عناها عليها الزمن.

في السنوات القليلة الماضية فقط، فاجأ التعلم العميق العالم، مما أدى إلى تقدم سريع في مجالات متنوعة مثل الرؤية الحاسوبية computer vision، ومعالجة اللغة الطبيعية natural language processing، والتعرف التلقائي على الكلام automatic speech recognition، والتعلم المعزز reinforcement learning، والمعلوماتية الطبية الحيوية biomedical informatics. علاوة على ذلك، أدى نجاح التعلم العميق في العديد من المهام ذات الأهمية العملية إلى تحفيز التطورات في التعلم الآلي النظري والإحصاءات. مع هذه التطورات في متناول اليد، يمكننا الآن بناء سيارات تقود نفسها باستقلالية أكثر من أي وقت مضى (واستقلالية أقل مما قد تعتقد بعض الشركات)، أنظمة الرد الذكية التي تقوم تلقائياً بصياغة معظم رسائل البريد الإلكتروني العادي، مما يساعد الناس معاً على الخروج من البريد الوارد الكبير بشكل قمعي، ووكلاء برمجيات يهيمنون على أفضل البشر في العالم في ألعاب الطاولة مثل Go، وهو إنجاز كان يُعتقد في السابق أنه يبعد عقوداً. بالفعل، تمارس هذه الأدوات تأثيرات واسعة النطاق على الصناعة والمجتمع، وتغيير طريقة صناعة الأفلام، وتشخيص الأمراض، وتلعب دوراً متزايدًا في العلوم الأساسية – من الفيزياء الفلكلورية إلى علم الأحياء.

حول هذا الكتاب

يمثل هذا الكتاب محاولتنا لجعل التعلم العميق سهل المنال، ويعمل على تبسيط المفاهيم والسباق والبرمجة.

متوسط واحد يجمع بين الكود والرياضيات و HTML

لكي تصل أي تقنية ح Osborne إلى تأثيرها الكامل، يجب أن تكون مفهومه جيداً وموثقة جيداً ومدعومة بأدوات ناضجة وجيدة الصيانة. يجب أن تكون الأفكار الرئيسية موجزة بوضوح، مما يقلل من وقت الإعداد اللازم لتحديث الممارسين الجدد. يجب أن تقوم المكتبات الناضجة بأتمتة المهام الشائعة، ويجب أن تسهل التعليمات البرمجية النموذجية على الممارسين تعديل التطبيقات الشائعة وتطبيقاتها وتوسيعها لتناسب احتياجاتهم. خذ تطبيقات الويب الديناميكية كمثال. على الرغم من وجود عدد كبير من الشركات، مثل Amazon، التي طورت تطبيقات ويب ناجحة تعتمد على قواعد البيانات في التسعينيات، فقد تم تحقيق إمكانات هذه التكنولوجيا لمساعدة رواد الأعمال المبدعين إلى درجة أكبر بكثير في السنوات العشر الماضية، ويرجع ذلك جزئياً إلى التطور من الأطر القوية والموثقة جيداً.

يمثل اختبار إمكانات التعلم العميق تحديات فريدة لأن أي تطبيق فردي يجمع بين مختلف التخصصات. يتطلب تطبيق التعلم العميق فهماً متزامناً (1) للدرواف طرح مشكلة بطريقة معينة؛ (2) الشكل الرياضي لنموذج معين. (3) خوارزميات التحسين لتلائم النماذج مع البيانات؛ (4) المبادئ الإحصائية التي تخبرنا متى يجب أن نتوقع أن نعمم نماذجنا على البيانات غير المرئية والطرق العملية للتتصديق على أنها، في الواقع، معممة؛ و (5) التقنيات الهندسية المطلوبة لتدريب النماذج بكفاءة، والتغلب على مخاطر الحوسبة الرقمية وتحقيق أقصى استفادة من الأجهزة المتاحة. يمثل تدريس كل من مهارات التفكير النقدي المطلوبة لصياغة المشكلات، والرياضيات لحلها، والأدوات البرمجية لتنفيذ هذه الحلول كلها في مكان واحد تحديات هائلة. هدفنا في هذا الكتاب هو تقديم مورد موحد لإطلاع الممارسين المختصين على السرعة.

عندما بدأنا مشروع الكتاب هذا، لم تكن هناك موارد (1) ظلت محدثة في نفس الوقت؛ (2) تغطية اتساع نطاق ممارسات التعلم الآلي الحديثة بعمق تقني كافٍ؛ و (3) عرض معشق للجودة التي يتوقعها الماء من كتاب مدرسي مع الكود النظيف القابل للتشغيل الذي يتوقعه الماء من برنامج تعليمي عملي. لقد وجدنا الكثير من أمثلة التعليمات البرمجية لكيفية استخدام إطار عمل تعليمي عميق معين (على سبيل المثال، كيفية إجراء الحوسبة الرقمية الأساسية باستخدام المصفوفات في TensorFlow) أو لتنفيذ تقنيات معينة (على سبيل المثال، مقتطفات التعليمات البرمجية لـ LeNet و AlexNet و ResNet وما إلى ذلك) المنشورة عبر العديد من منشورات المدونة ومستودعات GitHub. ومع ذلك، ركزت هذه الأمثلة عادةً على كيفية تفزيذ نهج معين، لكنها استبعدت مناقشة سبب اتخاذ قرارات خوارزمية معينة. بينما ظهرت بعض الموارد التفاعلية بشكل متقطع لمعالجة موضوع معين، على سبيل المثال، منشورات المدونة الجذابة المنشورة على موقع الويب Distill أو المدونات الشخصية، فإنها تقطي فقط موضوعات محددة في التعلم العميق، غالباً ما تفتقر إلى التعليمات البرمجية المرتبطة. من ناحية أخرى، بينما ظهرت العديد

من كتب التعلم العميق - على سبيل المثال، (Goodfellow et al., 2016)، والتي تقدم مسحًا شاملًا لأساسيات التعلم العميق - فإن هذه الموارد لا تقرن الأوصاف بإدراك المفاهيم في الكود. وأحياناً يترك القراء جاهلين بكيفية تتنفيذها. علاوة على ذلك، يتم إخفاء عدد كبير جدًا من الموارد خلف جدران حظر الاشتراك المدفوعة لمقدمي الدورات التدريبية التجارية.

شرعنا في إنشاء مورد يمكن (1) أن يكون متاحًا للجميع؛ (2) توفر عميقًا تقنيًا كافيًا لتوفير نقطة انطلاق على الطريق لتصبح بالفعل عالمًا تطبيقيًا للتعلم الآلي؛ (3) تضمّن التعليمات البرمجية القابلة للتشغيل، والتي توضح للقراء كيفية حل المشكلات عمليًا؛ (4) السماح بالتحديثات السريعة، سواء من جانبنا أو من قبل المجتمع community ككل؛ و (5) استكماله بمنتدى للمناقشة التفاعلية للتتفاصيل التقنية وللإجابة على الأسئلة.

كانت هذه الأهداف في كثير من الأحيان متعارضة. من الأفضل إدارة المعادلات والنظريات والاستشهادات ووضعها في LaTeX. أفضل وصف للكود في بايثون. وصفحات الويب أصلية بتنسيق HTML و JavaScript. علاوة على ذلك، نريد أن يكون المحتوى متاحًا سواء أكان رمزاً قابلاً للتنفيذ أو كتاباً مادياً أو ملف PDF قابل للتثبيت أو على الإنترنت كموقع ويب. لم يبدأ أي سير عمل مناسباً لهذه المطالب، لذلك قررنا تجميع مهامنا الخاصة (القسم 20.6). اتفقنا على GitHub لمشاركة المصدر وتسهيل مساهمات المجتمع؛ دفاتر جوبير Jupyter notebooks لخلط الكود والمعادلات والنص؛ Sphinx كمحرك تقديم؛ Discourse كمنصة مناقشة. في حين أن نظامنا ليس مثالياً، إلا أن هذه الخيارات تقدم حلًا وسطاً بين الاهتمامات المتنافسة. نعتقد أن كتاب Dive into Deep Learning قد يكون الكتاب الأول الذي يتم نشره باستخدام سير العمل المتكامل integrated workflow هنا.

التعلم عبر الممارسة

تقدم العديد من الكتب المدرسية المفاهيم على التوالي، وتقطع كل منها بتفصيل شامل. على سبيل المثال، يُعلم الكتاب المدرسي الممتاز لكرييس بيشوب Chris Bishop's excellent textbook (Bishop, 2006) كل موضوع بدقة شديدة للدرجة أن الوصول إلى فصل الانحدار الخططي يتطلب قدرًا غير ضئيل من العمل. بينما يحب الخبراء هذا الكتاب على وجه التحديد لشموله، بالنسبة للمبتدئين الحقيقيين، فإن هذه الخاصية تحد من فائدته كنص تمهيدي.

في هذا الكتاب، نقوم بتدريس معظم المفاهيم في الوقت المناسب. بمعنى آخر، سوف تتعلم المفاهيم في نفس اللحظة التي تكون فيها ضرورية لتحقيق بعض الأهداف العملية. بينما نأخذ بعض الوقت في البداية لتدريس المقدمات الأساسية، مثل الجبر الخططي linear algebra والاحتمالية probability، نريدك أن تتذوق الرضا بتدريب نموذجك الأول قبل القلق بشأن المزيد من المفاهيم الباطنية esoteric concepts.

بصرف النظر عن بعض النوتبوكس notebooks الأولية التي توفر دورة تدريبية مكثفة في الخلفية الرياضية الأساسية، يقدم كل فصل لاحق عدداً معقولاً من المفاهيم الجديدة ويوفر العديد من أمثلة العمل المستقلة، باستخداممجموعات بيانات حقيقية. قدم هذا تحدياً تنظيمياً. قد يتم بشكل منطقي تجميع بعض الطرز معّاً في دفتر ملاحظات واحد. وأفضل طريقة لتدريس بعض الأفكار هي تنفيذ عدة نماذج متتالية. من ناحية أخرى، هناك ميزة كبيرة للالتزام بسياسة مثل عملي واحد، نوتبوك واحد: هذا يجعل الأمر سهلاً قدر الإمكان بالنسبة لك لبدء مشاريع البحث الخاصة بك من خلال الاستفادة من التعليمات البرمجية الخاصة بنا. فقط انسخ النوتبوك وابدأ في تعديله.

طوال الوقت، تقوم بإدخال الكود القابل للتشغيل مع مواد الخلفية حسب الحاجة. بشكل عام، نخطئ في جانب إتاحة الأدوات قبل شرحها بالكامل (غالباً ما يتم ملء الخلفية لاحقاً). على سبيل المثال، قد نستخدم التدرج الاشتتقافي العشوائي stochastic gradient descent قبل شرح سبب فائدته أو تقديم حدس حول سبب نجاحه. يساعد هذافي إعطاء الممارسين الذخيرة اللازمة لحل المشكلات بسرعة، على حساب مطالبة القارئ بالثقة بنافي بعض القرارات التنظيمية.

يعلم هذا الكتاب مفاهيم التعلم العميق من الصفر. في بعض الأحيان، تتعقب في التفاصيل الدقيقة حول النماذج التي عادةً ما تكون مخفية عن المستخدمين بواسطة أطر التعلم العميق الحديثة. يظهر هذا بشكل خاص في البرامج التعليمية الأساسية، حيث نريد منك أن تفهم كل ما يحدث في طبقة أو مُحسنٍ معين. في هذه الحالات، غالباً ما نقدم نسختين من المثال: أحدهما ننفذ فيه كل شيء من البداية، ونعتمد فقط على دوال تشبه NumPy والتمايز التلقائي automatic differentiation، ومثال عملي أكثر، حيث نكتب كوداً موجزاً باستخدام واجهات برمجة التطبيقات عالية المستوى API high-level لـ أطر التعلم العميق. بعد شرح كيفية عمل بعض المكونات، نعتمد على واجهة برمجة التطبيقات عالية المستوى في البرامج التعليمية اللاحقة.

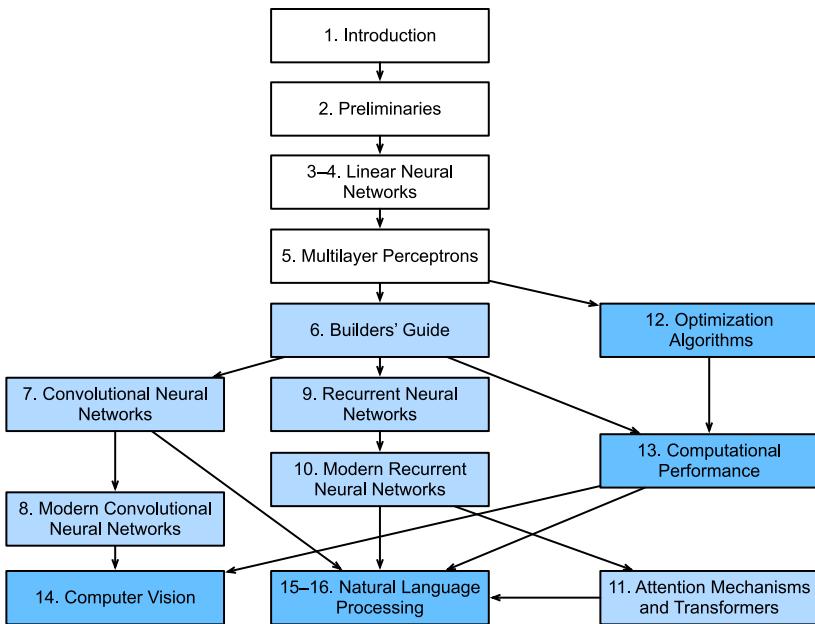
المحتوى والبنية

يمكن تقسيم الكتاب إلى ثلاثة أجزاء تقريرياً، مع التركيز على المقدمات وتقنيات التعلم العميق والموضوعات المتقدمة التي تركز على الأنظمة والتطبيقات الحقيقية (الشكل 1).

- **الجزء الأول: الأساسيات والمقدمات.** يقدم القسم الأول مقدمة للتعلم العميق. بعد ذلك، في القسم الثاني، نطلعك بسرعة على المتطلبات الأساسية الازمة للتعلم العميق العملي، مثل كيفية تخزين البيانات ومعالجتها، وكيفية تطبيق العمليات العددية المختلفة بناءً على المفاهيم الأساسية من الجبر الخطى، وحساب التفاضل والتكامل، والاحتمال. يعطي القسم الرابع والقسم الخامس المفاهيم والتقنيات الأساسية في التعلم العميق، بما في ذلك الانحدار regression والتصنیف classification؛ نماذج

خطية linear models: البيرسيptron متعددة الطبقات multilayer perceptron وفرط التجهيز overfitting والتنظيم regularization.

- الجزء الثاني: تقنيات التعلم العميق الحديثة. يصف القسم السادس المكونات الحاسبية الرئيسية لأنظمة التعلم العميق ويضع الأساس لتطبيقاتنا اللاحقة لنماذج أكثر تعقيداً. بعد ذلك، يقدم القسم السابع والقسم الثامن الشبكات العصبية التلاغيفية (CNNs)، وهي أدوات قوية تشكل العمود الفقري لمعظم أنظمة الرؤية الحاسوبية الحديثة. وبالمثل، يقدم القسم التاسع والقسم العاشر الشبكات العصبية المتكررة (RNNs)، وهي نماذج تستغل البنية المتسلسلة (على سبيل المثال، الزمنية) في البيانات وتستخدم بشكل شائع لمعالجة اللغة الطبيعية والتنبؤ بالسلسل الزمنية. في القسم الحادي عشر، قدمتنا فئة جديدة نسبياً من النماذج تعتمد على ما يسمى بالآيات الانتباه attention mechanisms التي حلّت محل معلم RNNs باعتبارها البنية المهمة لمعظم مهام معالجة اللغة الطبيعية. سُلطَّلَعَت هذه الأقسام على أقوى الأدوات والأدوات العامة المستخدمة على نطاق واسع من قبل ممارسي التعلم العميق.
- الجزء الثالث: قابلية التوسيع والكافأة والتطبيقات. في القسم الثاني عشر، نناقش العديد من خوارزميات التحسين الشائعة المستخدمة لتدريب نماذج التعلم العميق. بعد ذلك، في القسم الثالث عشر، ندرس العديد من العوامل الرئيسية التي تؤثر على الأداء الحسابي لكود التعلم العميق. بعد ذلك، في القسم الرابع عشر، نوضح التطبيقات الرئيسية للتعلم العميق في الرؤية الحاسوبية. أخيراً، في القسم الخامس عشر والقسم السادس عشر، نوضح كيفية التدريب المسبق لنماذج تمثيل اللغة وتطبيقاتها على مهام معالجة اللغة الطبيعية. هذا الجزء متاح على الإنترنت.



الكود

تتميز معظم أقسام هذا الكتاب بكتاب قابل للتنفيذ. نعتقد أن أفضل طريقة لتطوير بعض الديفيهيات هي التجربة والخطأ trial and error، وتعديل الكود بطرق صغيرة ومراقبة النتائج. من الناحية المثلية، قد تخبرنا النظرية الرياضية الأنيقة بدقة كيفية تعديل الكود الخاص بنا لتحقيق النتيجة المرجوة. ومع ذلك، يجب على ممارسي التعلم العميق اليوم أن يسيروا في كثير من الأحيان حيث لا توجد نظرية قوية توفر التوجيه. على الرغم من أفضل محاولاتنا، لا تزال التفسيرات الرسمية لفعالية التقنيات المختلفة غير متوفرة، لأن الرياضيات لتوصيف هذه النماذج يمكن أن تكون صعبة للغاية، لأن التفسير يعتمد على الأرجح على خصائص البيانات التي تفتقر حالياً إلى تعريفات واضحة، ولأن التحقيق الجاد حول هذه الموضوعات مؤخراً إلى مستوى عالٍ. نأمل أنه مع تقدم نظرية التعلم العميق، ستتوفر كل طبعة مستقبلية من هذا الكتاب رؤى تتتفوق على تلك المتاحة حالياً.

لتجنب التكرار غير الضروري، نقوم بتغليف بعض الدوال والفئات الأكثر استخداماً واستيراداً في حزمة `d2l`. طوال الوقت، نقوم بتمييز كتل التعليمات البرمجية (مثل الدوال أو الفئات أو مجموعة عبارات الاستيراد) باستخدام `#@save` للإشارة إلى أنه سيتم الوصول إليها لاحقاً عبر حزمة `d2l`. نقدم نظرة عامة مفصلة عن هذه الدوال والفئات في القسم 20.8. حزمة `d2l` خفيفة الوزن وتطلب فقط التبعيات التالية:

`#@save`

```

import collections
import hashlib
import inspect
import math
import os
import random
import re
import shutil
import sys
import tarfile
import time
import zipfile
from collections import defaultdict
import pandas as pd
import requests
from IPython import display
from matplotlib import pyplot as plt
from matplotlib_inline import backend_inline

```

d21 = sys.modules[__name__]

يعتمد معظم الكود في هذا الكتاب على TensorFlow ، وهو إطار مفتوح المصدر للتعلم العميق يتم اعتماده على نطاق واسع في الصناعة ويحظى بشعبية بين الباحثين. اجتازت جميع التعليمات البرمجية الموجودة في هذا الكتاب الاختبارات بموجب أحدث إصدار ثابت من TensorFlow . ومع ذلك، نظراً للتطور السريع في التعلم العميق، فقد لا تعمل بعض التعليمات البرمجية في النسخة المطبوعة بشكل صحيح في الإصدارات المستقبلية من TensorFlow . نحن نخطط لتحديث النسخة عبر الإنترنت. في حالة مواجهة أي مشاكل، يرجى الرجوع إلى التثبيت لتحديث التعليمات البرمجية وبيئة وقت التشغيل.

. إليك كيفية استيراد الوحدات النمطية من TensorFlow

```

#@save
import numpy as np
import tensorflow as tf

```

الجمهور المستهدف

هذا الكتاب مخصص للطلاب (الجامعيين أو الخريجين) والمهندسين والباحثين الذين يسعون إلى فهم قوي للتقنيات العملية للتعلم العميق. نظراً لأننا نشرح كل مفهوم من البداية، فلا يلزم وجود خلفية سابقة في التعلم العميق أو التعلم الآلي. يتطلب التفسير الكامل لأساليب التعلم العميق بعض الرياضيات والبرمجة، لكننا سنفترض فقط أنك تأتي ببعض الأساسيات، بما في ذلك

كميات متواضعة من الجبر الخطي وحساب التفاضل والتكامل والاحتمالات وبرمجة بايثون. فقط في حالة نسيان الأساسيات، يوفر الملحق تجدیداً لمعظم الرياضيات التي ستجدها في هذا الكتاب. في معظم الأحيان، سمنحك الحدس والأفكار الأولية على الدقة الرياضية. إذا كنت ترغب في توسيع هذه الأساس إلى ما هو أبعد من المتطلبات الأساسية لفهم كتابنا، فإننا نوصي بسعادة بعض الموارد الرائعة الأخرى: التحليل الخطي بواسطة Bela Bollobas (Bollobas, 1999) يغطي الجبر الخطي والتحليل الدالي بعمق كبير. توفر كافة الإحصائيات (Wasserman, 2013) مقدمة رائعة للإحصاءات. تعد كتب دورات جو بليرشتاين حول الاحتمالات والاستدلال جواهر تربوية. وإذا لم تكن قد استخدمنا بايثون من قبل، فقد ترغب في الاطلاع على برنامج بايثون التعليمي هذا . Python tutorial

الم المنتدى

مرتبطاً بهذا الكتاب، أطلقنا منتدى للمناقشة، يقع في discuss.d2l.ai. عندما يكون لديك أسئلة حول أي قسم من الكتاب، يمكنك العثور على ارتباط إلى صفحة المناقشة المرتبطة في نهاية كل نوتيتك.

شكر وتقدير

نحن مدینون لمئات المساهمین في كل من المسودتين الإنجليزية والصینية. لقد ساعدوا في تحسین المحتوى وقدموا ملاحظات قيمة. على وجه التحديد، نشكر كل مساهم في هذه المسودة الإنجليزية لجعلها أفضل للجميع. معرفات أو أسماء GitHub الخاصة بهم (بدون ترتيب معین): alxnorden, avinashsingit, bowen0701, brettkoonce, Chaitanya, Prakash Bapat, cryptonaut, Davide Fiocco, edgarroman, gkutiel, John Mitro, Liang Pu, Rahul Agarwal, Mohamed Ali Jamaoui, Michael (Stu) Stewart, Mike Müller, NRauschmayr, Prakhar Srivastav, sad-, sfermigier, Sheng Zha, sundeepteki, topecongiro, tpd1, vermicelli, Vishaal Kapoor, Vishwesh Ravi Shrimali, YaYaB, Yuhong Chen, Evgeniy Smirnov, lgov, Simon Corston-Oliver, Igor Dzreyev, Ha Nguyen, pmuens, Andrei Lukovenko, senorcinco, vfdev-5, dsweet, Mohammad Mahdi Rahimi, Abhishek Gupta, uwsd, DomKM, Lisa Oakley, Bowen Li, Aarush Ahuja, Prasanth Buddareddygari, brianhendee, mani2106, mtn, lkevinzc, caojilin, Lakshya, Fiete Lüer, Surbhi Vijayvargeeya, Muhyun Kim, dennismalmgren, adursun, Anirudh Dagar, liqingnz, Pedro Larroy, lgov, ati-ozgur, Jun Wu, Matthias Blume, Lin Yuan, geogunow, Josh Gardner,

Maximilian Böther, Rakib Islam, Leonard Lausen, Abhinav Upadhyay, rongruosong, Steve Sedlmeyer, Ruslan Baratov, Rafael Schlatter, liusy182, Giannis Pappas, ati-ozgur, qbaza, dchoi77, Adam Gerson, Phuc Le, Mark Atwood, christabella, vn09, Haibin Lin, jjangga0214, RichyChen, noelo, hansent, Giel Dops, dvincent1337, WhiteD3vil, Peter Kulits, codypenta, joseppinilla, ahmaurya, karolszk, heytitle, Peter Goetz, rigtorp, Tiep Vu, sfilip, mlxrd, Kale-ab Tessera, Sanjar Adilov, MatteoFerrara, hsneto, Katarzyna Biesialska, Gregory Bruss, Duy—Thanh Doan, paulaurel, graytowne, Duc Pham, sl7423, Jaedong Hwang, Yida Wang, cys4, clhm, Jean Kaddour, austinmw, trebeljahr, tbaums, Cuong V. Nguyen, pavelkomarov, vzlamal, NotAnotherSystem, J-Arun-Mani, jancio, eldarkurtic, the-great-shazbot, doctorcolossus, gducharme, cclauss, Daniel-Mietchen, hoonose, biagiom, abhinavsp0730, jonathanhrandall, ysraell, Nodar Okroshiashvili, UgurKap, Jiyang Kang, StevenJokes, Tomer Kaftan, liweiwp, netyster, ypandya, NishantTharani, heiligerl, SportsTHU, Hoa Nguyen, manuel-arno-korfmann-webentwicklung, aterzis-personal, nxby, Xiaoting He, Josiah Yoder, mathresearch, mzz2017, jroberayalas, iluu, ghejc, BSharmi, vkramdev, simonwardjones, LakshKD, TalNeoran, djliden, Nikhil95, Oren Barkan, guoweis, haozhu233, pratikhack, Yue Ying, tayfununal, steinsag, charleybeller, Andrew Lumsdaine, Jiekui Zhang, Deepak Pathak, Florian Donhauser, Tim Gates, Adriaan Tijsseling, Ron Medina, Gaurav Saha, Murat Semerci, Lei Mao, Levi McClenny, Joshua Broyde, jake221, jonbally, zyhazwraith, Brian Pulfer, Nick Tomasino, Lefan Zhang, Hongshen Yang, Vinney Cavallo, yuntai, Yuanxiang Zhu, amarazov, pasricha, Ben Greenawald, Shivam Upadhyay, Quanshangze Du, Biswajit Sahoo, Parthe Pandit, Ishan Kumar, HomunculusK, Lane Schwartz, varadgunjal, Jason Wiener, Armin Gholampoor, Shreshtha13, eigenarnav, Hyeonggyu Kim, EmilyOng, Bálint Mucsányi, Chase DuBois, Juntian Tao, Wenxiang Xu, Lifu Huang, filevich, quake2005, nils-werner, Yiming Li, Marsel Khisamutdinov, Francesco “Fuma” Fumagalli, Peilin Sun, Vincent Gurgul, qingfengtommy, Janmey Shukla, Mo Shan,

Kaan Sancak, regob, AlexSauer, Gopalakrishna Ramachandra, Tobias Uelwer, Chao Wang, Tian Cao, Nicolas Corthorn, akash5474, kxxt, zxydi1992, Jacob Britton, Shuangchi He, zhmou, krahets, Jie-Han Chen, Atishay Garg, Marcel Flygare, adtygan, Nik Vaessen, bolded, Louis Schlessinger, Balaji Varatharajan, atgctg, Kaixin Li, Victor Barbaros, Riccardo Musto, Elizabeth Ho, azimjonn, Guilherme Miotto, Alessandro Finamore, Joji Joseph, Anthony Biel, Sere1nz.

نشكر Swami Sivasubramanian و Amazon Web Services، وخاصة Andrew Jassy و Adam Selipsky DeSantis لدعمهم السخي في كتابة هذا الكتاب. لولا الوقت والموارد المتاحة والمناقشات مع الزملاء والتشجيع المستمر لما حدث هذا الكتاب.

الملخص

أحدث التعلم العميق ثورة في التعرف على الأنماط pattern recognition، حيث أدخل التكنولوجيا التي تعمل الآن على تشغيل مجموعة واسعة من التقنيات، في مجالات متنوعة مثل الرؤية الحاسوبية، ومعالجة اللغة الطبيعية، والتعرف التلقائي على الكلام. لتطبيق التعلم العميق بنجاح، يجب أن تفهم كيفية طرح مشكلة، والرياضيات الأساسية للتمذجة، والخوارزميات لتناسب النماذج الخاصة بك مع البيانات، والتقنيات الهندسية لتنفيذها جمیعاً. يقدم هذا الكتاب مورداً شاملاً، بما في ذلك النشر والأرقام والرياضيات والاكواد، كل ذلك في مكان واحد. لطرح (أو الإجابة) أسئلة تتعلق بهذا الكتاب، قم بزيارة منتدىنا على <https://discuss.d2l.ai/>. جميع النوتوكوس الخاصة بنا متاحة للتثبيت على موقع الويب [D2L.ai website](#) وعلى [GitHub](#).

التمارين

- قم بتسجيل حساب في منتدى المناقشة الخاص بهذا الكتاب discuss.d2l.ai.
- قم بتنزيل بايثون على جهاز الكمبيوتر الخاص بك.
- اتبع الروابط الموجودة في الجزء السفلي من قسم المنتدى، حيث ستتمكن من طلب المساعدة ومناقشة الكتاب والعثور على إجابات لأسئلتك من خلال إشراك المؤلفين والمجتمع الأوسع.

التثبيت

من أجل البدء والتشغيل، ستحتاج إلى بيئة لتشغيل بايثون Jupyter Notebook والمكتبات ذات الصلة والكود اللازم لتشغيل الكتاب نفسه.

Miniconda تثبيت

أبسط خيار لك هو تثبيت Miniconda. لاحظ أن إصدار.x Python مطلوب. يمكنك تخطي الخطوات التالية إذا كان جهازك مثبتاً بالفعل على .conda.

قم بزيارة موقع Miniconda على الويب وحدد الإصدار المناسب لنظامك بناءً على إصدار Python 3.x وبنية الجهاز. افترض أن إصدار بايثون الخاص بك هو 3.9 (الإصدار الذي تم اختباره). إذا كنت تستخدم macOS، فيمكنك تنزيل برنامج bash النصي الذي يحتوي اسمه على السلاسل "MacOSX"، وانتقل إلى موقع التنزيل، وقم بتنفيذ التثبيت على النحو التالي (معأخذ كمثال Intel Macs):

```
# The file name is subject to changes
```

```
sh Miniconda3-py39_4.12.0-MacOSX-x86_64.sh -b
```

يقوم مستخدم Linux بتنزيل الملف الذي يحتوي اسمه على السلاسل "Linux" وتنفيذ ما يلي في موقع التنزيل:

```
# The file name is subject to changes
```

```
sh Miniconda3-py39_4.12.0-Linux-x86_64.sh -b
```

بعد ذلك، قم بتهيئة shell حتى نتمكن من تشغيل conda مباشرةً.

```
~/miniconda3/bin/conda init
```

ثم أغلق وأعد فتح shell الحالي. يجب أن تكون قادراً على إنشاء بيئة جديدة على النحو التالي:

```
conda create --name d21 python=3.9 -y
```

الآن يمكننا تنشيط بيئة d21:

```
conda activate d21
```

تثبيت إطار عمل التعلم العميق وحزمة d21

قبل تثبيت أي إطار عمل للتعلم العميق، يرجى أولاً التتحقق مما إذا كان لديك وحدات معالجة رسومات GPU مناسبة على جهازك أم لا (وحدات معالجة الرسومات التي تشغل الشاشة على كمبيوتر محمول قياسي ليست ذات صلة بأغراضنا). على سبيل المثال، إذا كان جهاز الكمبيوتر الخاص بك يحتوي على وحدات معالجة رسومات NVIDIA CUDA وقام بتنصيب CUDA، فأنت جاهز تماماً. إذا كان جهازك لا يحتوي على أي وحدة معالجة رسومات، فلا داعي للقلق الآن. توفر وحدة المعالجة المركزية CPU الخاصة بك أكثر من قوة حسانية كافية لتصفح الفصول القليلة الأولى. فقط تذكر أنك سترغب في الوصول إلى وحدات معالجة الرسومات قبل تشغيل النماذج الأكبر.

يمكنك تثبيت TensorFlow باستخدام دعم وحدة المعالجة المركزية CPU أو وحدة معالجة الرسومات GPU على النحو التالي:

```
pip install tensorflow tensorflow-probability
```

خطوتنا التالية هي تثبيت حزمة d2l التي قمنا بتطويرها لتغليف الدوال والฟئات (الكلاسات) المستخدمة بشكل متكرر والموجودة في هذا الكتاب:

```
pip install d2l==1.0.0a1
```

تحميل وتشغيل الكود

بعد ذلك، ستحتاج إلى تنزيل النوتبوكس بحيث يمكنك تشغيل كل من كتل التعليمات البرمجية للكتاب. ما عليك سوى النقر فوق علامة التبويب "Notebooks" أعلى أي صفحة HTML على موقع [D2L.ai](https://d2l.ai) لتنزيل الرمز ثم فك ضغطه. بدلاً من ذلك، يمكنك جلب دفاتر الملاحظات من سطر الأوامر على النحو التالي:

```
mkdir d2l-en && cd d2l-en
curl https://d2l.ai/d2l-en.zip -o d2l-en.zip
unzip d2l-en.zip && rm d2l-en.zip
cd tensorflow
```

إذا لم يكن لديك برنامج unzip مثبتاً بالفعل، فقم أولاً بتشغيل [Jupyter Notebook](#) من خلال تشغيل:

```
jupyter notebook
```

في هذه المرحلة، يمكنك فتح <http://localhost:8888> (ربما تم فتحه تلقائياً بالفعل) في مستعرض الويب الخاص بك. ثم يمكنك تشغيل الكود لكل قسم من الكتاب. عندما تفتح نافذة سطر أوامر جديدة، ستحتاج إلى تنفيذ ([conda activate d2l](#)) لتنشيط بيئه وقت التشغيل قبل تشغيل نوتبوكس D2L ، أو تحديث الحزم الخاصة بك (إما إطار عمل التعلم العميق أو حزمة d2l). للخروج من البيئه، قم بتشغيل [.conda deactivate](#).

المحتويات

4	حول هذا الكتاب
5	متوسط واحد يجمع بين الكود والرياضيات و HTML
6	التعلم عبر الممارسة
7	المحتوى والبنية
9	ال코드
10	الجمهور المستهدف
11	المنتدى
11	شكر وتقدير
13	الملخص
13	التمارين
13	الثبيت
14	ثبيت Miniconda
14	ثبيت إطار عمل التعلم العميق وحزمة d2l
15	تحميل وتشغيل الكود
28	1. المقدمة
29	1.1 مثال محفز
32	1.2 المكونات الرئيسية
32	1.2.1 البيانات
34	1.2.2 النماذج
34	1.2.3 دوال الهدف
35	1.2.4 خوارزميات التحسين
36	1.3 أنواع مشاكل التعلم الآلي
36	1.3.1 التعلم الخاضع للإشراف Supervised Learning
37	1.3.1.1 الانحدار Regression
39	1.3.1.2 التصنيف classification
41	1.3.1.3 وضع العلامات Tagging
42	1.3.1.4 البحث Search

43	أنظمة التوصية Recommender Systems	1.3.1.5
44	التعلم المتسلسل Sequence Learning	1.3.1.6
46	Self-Supervised. التعلم غير الخاضع للإشراف الذاتي Unsupervised والإشراف الذاتي Self-Supervised	1.3.2
48	التعامل مع البيئة Interacting with an Environment	1.3.3
49	التعلم المعزز Reinforcement Learning	1.3.4
51	الجذور Roots	1.4
54	الطريق إلى التعلم العميق The Road to Deep Learning	1.5
58	قصص نجاح Success Stories	1.6
60	جوهر التعلم العميق The Essence of Deep Learning	1.7
62	الملخص	1.8
62	التمارين	1.9
65	2. الأساسيات Preliminaries	
65	معالجة البيانات Data Manipulation	2.1
65	البداية 2.1.1	
68	الفهرسة والتقطيع Indexing and Slicing	2.1.2
69	العمليات Operations	2.1.3
71	البث Broadcasting	2.1.4
72	حفظ الذاكرة Saving Memory	2.1.5
73	التحويل إلى كائنات بايثون الأخرى	2.1.6
74	الملخص	2.1.7
74	التمارين	2.1.8
74	معالجة البيانات Data Preprocessing	2.2
74	قراءة مجموعة البيانات Reading the Dataset	2.2.1
75	تحضير البيانات Data Preparation	2.2.2
76	التحويل إلى تنسيق Tensor	2.2.3
77	المناقشة	2.2.4
77	التمارين	2.2.5
78	الجبر الخطى Linear Algebra	2.3
78	الكميات القياسية Scalars	2.3.1

79	. المتجهات Vectors	2.3.2
80	. المصفوفات Matrices	2.3.3
81	. الموترات Tensors	2.3.4
82	. الخصائص الأساسية لحساب الموتر	2.3.5
83	. الاختزال Reduction	2.3.6
84	. مجموع عدم الاختزال Non-Reduction Sum	2.3.7
85	. الضرب النقطي Dot Products	2.3.8
86	. ضرب Matrix-Vector	2.3.9
87	. ضرب المصفوفة – المصفوفة Matrix-Matrix Multiplication	2.3.10
88	. المعيار Norms	2.3.11
90	. المناقشة	2.3.12
91	. التمارين	2.3.13
92	. التفاضل والتكامل Calculus	2.4
92	. المشتقات والتفاضل Derivatives and Differentiation	2.4.1
94	. أدوات الرسم Visualization Utilities	2.4.2
97	. المشتقات الجزئية والتدرجات Partial Derivatives and Gradients	2.4.3
98	. قاعدة السلسلة Chain Rule	2.4.4
98	. المناقشة	2.4.5
99	. التمارين	2.4.6
99	. التفاضل التلقائي Automatic Differentiation	2.5
100	. دالة بسيطة A Simple Function	2.5.1
101	. عكسيًا بالنسبة للمتغيرات غير العددية Backward for Non-Scalar	2.5.2
102	. فصل الحساب Detaching Computation	2.5.3
102	. التدرجات وتدفق التحكم في بيانات Gradients and Python Control Flow	2.5.4
103	. المناقشة	2.5.5
104	. التمارين	2.5.6
104	. الاحتمال والاحصاء Probability and Statistics	2.6
105	. مثال بسيط: رمي العملات المعدنية Tossing Coins	2.6.1

108	A More Formal Treatment 2.6.2
109	المتغيرات العشوائية 2.6.3
110	Multiple Random Variables 2.6.4
114	مثال 2.6.5
116	التوقعات 2.6.6
118	المناقشة 2.6.7
119	التمارين 2.6.8
120	Documentation 2.7
120	Functions and Classes in a Module 2.7.1
120	Specific Functions and Classes 2.7.2
124 .. Linear Neural Networks for Regression	3. الشبكات العصبية الخطية للانحدار
124	الانحدار الخطى 3.1
125	الأساسيات 3.1.1
126	دالة الخطأ 3.1.1.2
128	حل التحليلي 3.1.1.3
128	Minibatch Stochastic 3.1.1.4
128	Gradient Descent
131	Predictions 3.1.1.5
131	التوجيه من أجل السرعة 3.1.2
131	The Normal Distribution and Squared Loss 3.1.3
132	
134	الانحدار الخطى كشبكة عصبية 3.1.4
135	Biology 3.1.4.1
136	الملخص 3.1.5
137	التمارين 3.1.6
138	Object-Oriented Design for Implementation 3.2
139	الأدوات المساعدة 3.2.1
141	النماذج 3.2.2
143	بيانات 3.2.3

143	3.2.4 التدريب Training
144	3.2.5 الملخص
145	3.3 بيانات الانحدار التركيبية Synthetic Regression Data
145	3.3.1 إنشاء مجموعة البيانات Generating the Dataset
146	3.3.2 قراءة مجموعة البيانات Reading the Dataset
	3.3.3 التنفيذ المختصر لمحمل البيانات Concise Implementation of the Data
148	Loader
149	3.3.5 التمارين
	3.4 تنفيذ الانحدار الخطى من الصفر Linear Regression Implementation from scratch
150	Scratch
150	3.4.1 تعريف النموذج Defining the Model
151	3.4.2 تعريف دالة الخطأ Defining the Loss Function
151	3.4.3 تعريف خوارزمية التحسين Defining the Optimization Algorithm
152	3.4.4 التدريب Training
155	3.4.5 الملخص
155	3.4.6 التمارين
156	3.5 التنفيذ المختصر للانحدار الخطى Concise Implementation of Linear Regression
156	Linear Regression
157	3.5.1 تعريف النموذج Defining the Model
158	3.5.2 تعريف دالة الخطأ Defining the Loss Function
158	3.5.3 تعريف خوارزمية التحسين Defining the Optimization Algorithm
158	3.5.4 التدريب Training
159	3.5.5 الملخص
160	3.5.6 التمارين
160	3.6 التعميم Generalization
	3.6.1 خطأ في التدريب وخطأ في التعميم Training Error and Generalization
162	Error
163	3.6.1.1 تعقييد النموذج Model Complexity
164	3.6.2 الضبط الناقص Underfitting أو الضبط الزائد Overfitting
165	3.6.2.1 ملائمة منحنى متعدد الحدود Polynomial Curve Fitting

166	Dataset Size 3.6.2.2 حجم مجموعة البيانات
166	Model Selection 3.6.3 اختيار النموذج
167	Cross-Validation 3.6.3.1 التحقق المتبادل
167	 الملخص 3.6.4
168	التمارين 3.6.5
168	Weight Decay 3.7 اضمحلال الوزن
169	3.7.1 .المعايير وتناقص الوزن Norms and Weight Decay
171	3.7.2 .الانحدار الخطى عالي الأبعاد High-Dimensional Linear Regression
172	3.7.3 .التنفيذ من البداية Implementation from Scratch
172	3.7.3.1 .تحديد عقوبة ℓ_2 المعيارية Defining ℓ_2 Norm Penalty
172	3.7.3.2 .تعريف النموذج Defining the Model
173	3.7.3.3 .التدريب بدون التنظيم Training without Regularization
173	3.7.3.4 .استخدام تناقص الوزن Using Weight Decay
174	3.7.4 .التنفيذ المختصر Concise Implementation
175	 الملخص 3.7.5
176	التمارين 3.7.6
178	Linear Neural Networks for Classification 4. الشبكات العصبية الخطية للتصنيف
178	Softmax 4.1 انحدار
179	4.1.1 .التصنيف Classification
180	4.1.1.1 .النموذج الخطى Linear Model
181	4.1.1.2 .سوفت ماكس Softmax
182	4.1.1.3 .فيكتوريزاشن Vectorization
182	4.1.2 .دالة الخطأ Loss Function
183	4.1.2.1 .Log-Likelihood
184	4.1.2.2 .Softmax and Cross-Entropy Loss وخطأ الانتروبيا Softmax
185	4.1.3 .أساسيات نظرية المعلومات Information Theory Basics
185	4.1.3.1 .الإنتروبيا Entropy
185	4.1.3.2 .Surprisal
186	4.1.3.3 .إعادة النظر عبر الانتروبيا Cross-Entropy Revisited

186	4.1.4 الملخص والمناقشة
187	4.1.5 التمارين
188	4.2 مجموعة بيانات تصنیف الصور
189	4.2.1 تحميل مجموعة البيانات
190	4.2.2 قراءة الدفعات الصغيرة
191	4.2.3 التمثيل البياني
192	4.2.4 الملخص
192	4.2.5 التمارين
193	4.3 نموذج التصنیف الأساسي
193	4.3.1 فئة المصنف
194	4.3.2 الدقة
194	4.3.3 الملخص
195	4.3.4 تمارين
195	4.4 تنفيذ انحدار Softmax من الصفر
195	Scratch
195	4.4.1 سوفت ماكس
196	4.4.2 الموديل
197	4.4.3 الخطأ عبر الانتروپیا
198	4.4.4 التدريب
199	4.4.5 التنبؤ
199	4.4.6 الملخص
200	4.4.7 التمارين
200	4.5 التنفيذ المختصر لانحدار Softmax
201	4.5.1 تعريف النموذج
201	4.5.2 إعادة النظر في سوفت ماكس
202	4.5.3 التدريب
203	4.5.4 الملخص
203	4.5.5 التمارين
204	4.6 التعميم في التصنیف

205	4.6.1. مجموعة الاختبار
207	4.6.2. إعادة استخدام مجموعة الاختبار
209	4.6.3. نظرية التعلم الإحصائي
211	4.6.4. الملخص
212	4.6.5. التمارين
212	4.7. التحول البيئي والتوزيع
213	4.7.1. أنواع تحول التوزيع
214	4.7.1.1. التحول المتغير Covariate Shift
215	4.7.1.2. تحول التسمية Label Shift
215	4.7.1.3. تحول المفهوم Concept Shift
216	4.7.2. أمثلة على تحول التوزيع Examples of Distribution Shift
216	4.7.2.1. التشخيصات الطبية Medical Diagnostics
217	4.7.2.2. السيارات ذاتية القيادة Self-Driving Cars
217	4.7.2.3. التوزيعات غير الثابتة Nonstationary Distributions
218	4.7.2.4. المزيد من الحكايات More Anecdotes
218	4.7.3. تصحيح تحول التوزيع Correction of Distribution Shift
218	4.7.3.1. الخطأ التجريبية والخطأ Empirical Risk and Risk
219	4.7.3.2. تصحيح التحول المتغير Covariate Shift Correction
221	4.7.3.3. تصحيح تحول التسمية Label Shift Correction
222	4.7.3.4. تصحيح تحول المفهوم Concept Shift Correction
223	4.7.4. تصنیف مشاكل التعلم A Taxonomy of Learning Problems
223	4.7.4.1. التعلم الجماعي Batch Learning
223	4.7.4.2. التعلم الاولئي Online Learning
223	4.7.4.3. Bandits
224	4.7.4.4. وحدات التحكم Control
224	4.7.4.5. التعلم المعزز Reinforcement Learning
224	4.7.4.6. مراعاة البيئة Considering the Environment
225	4.7.5.الإنصاف والمساءلة والشفافية في التعلم الآلي Fairness, Accountability, and Transparency in Machine Learning
226	4.7.6. الملخص

226	4.7.7 التمارين
228	5. البيرسيبترون متعدد الطبقات
228	Multilayer Perceptrons
228	5.1 البيرسيبترون متعدد الطبقات
228	5.1.1. الطبقات المخفية
229	5.1.1.1. عيوب النماذج الخطية
230	5.1.1.2. دمج الطبقات المخفية
231	5.1.1.3. من الخطى إلى غير الخطى
232	5.1.1.4. المقربين العالميين
233	5.1.2. دوال التنشيط
233	5.1.2.1. ReLU. دالة
235	5.1.2.2. Sigmoid. دالة
237	5.1.2.3. Tanh. دالة
238	5.1.3. الملخص
238	5.1.4. التمارين
239	5.2 تنفيذ البيرسيبترون متعدد الطبقات
239	5.2.1. التنفيذ من البداية
239	5.2.1.1. Initializing Model Parameters
240	5.2.1.2. Model. النموذج
240	5.2.1.3. Training. التدريب
241	5.2.2. التنفيذ المختصر
241	5.2.2.1. Model. النموذج
241	5.2.2.2. Training. التدريب
242	5.2.3. الملخص
242	5.2.4. التمارين
243	5.3 الانبعاث الأمامي، Forward Propagation، والانبعاث الخلفي
243	Computational Graphs والرسوم البيانية الحسابية
243	5.3.1. Forward Propagation. الانبعاث الأمامي
244	5.3.2. Computational Graph of Forward. الرسم البياني الحسابي للانبعاث الأمامي
244	Propagation

246	5.3.4 تدريب الشبكات العصبية Training Neural Networks
247	5.3.5 الملخص
247	5.3.6 التمارين
248	5.4 الاستقرار العددي والتهيئة Numerical Stability and Initialization
248	5.4.1 تلاشي وانفجار التدرجات Vanishing and Exploding Gradients
249	5.4.1.1 تلاشي التدرجات Vanishing Gradients
250	5.4.1.2 انفجار التدرجات Exploding Gradients
251	5.4.1.3 كسر التماثل Breaking the Symmetry
251	5.4.2 تهيئة المعلمة Parameter Initialization
251	5.4.2.1 التهيئة الافتراضية Default Initialization
252	5.4.2.2 Xavier
253	5.4.2.3 Beyond
253	5.4.3 الملخص
254	5.4.4 التمارين
254	5.5 التعميم في التعلم العميق Generalization in Deep Learning
255	5.5.1 إعادة النظر في فرط التجهيز والتنظيم Revisiting Overfitting and Regularization
257	5.5.2 إلهام من غير البارامترية Inspiration from Nonparametrics
258	5.5.3 التوقف المبكر Early Stopping
259	5.5.4 طرق التنظيم الكلاسيكية للشبكات العميقه Classical Regularization
259	5.5.5 Methods for Deep Networks
259	5.5.5 الملخص
260	5.5.6 التمارين
260	5.6 الحذف العشوائي Dropout
261	5.6.1 الحذف العشوائي في الممارسة Dropout in Practice
262	5.6.2 التنفيذ من البداية Implementation from Scratch
263	5.6.2.1 تعريف النموذج Defining the Model
264	5.6.2.2 التدريب Training
264	5.6.3 التنفيذ المختصر Concise Implementation
266	5.6.4 الملخص

266	5.6.5 التمارين
266	5.7 توقع أسعار المنازل في كاجل Predicting House Prices on Kaggle
267	5.7.1 تنزيل البيانات Downloading Data
268	5.7.2 Kaggle
5.7.3 الوصول إلى مجموعة البيانات وقراءتها Accessing and Reading the Dataset	
268	
270	5.7.4 المعالجة المسبيقة للبيانات Data Preprocessing
272	5.7.5 قياس الخطأ Error Measure
273	5.7.6 K-Fold Cross Validation
274	5.7.7 اختيار النموذج Model Selection
275	5.7.8 تقديم التنبؤات على كاجل Submitting Predictions on Kaggle
276	5.7.9 الملخص
276	5.7.10 التمارين

المقدمة

1

1. المقدمة

حتى وقت قريب، تم ترميز كل برنامج كمبيوتر تقريباً قد تتفاعل معه في يوم عادي كمجموعة صارمة من القواعد التي تحدد بدقة كيف يجب أن يتصرف. لنفترض أننا أردنا كتابة تطبيق لإدارة منصة التجارة الإلكترونية e-commerce platform. بعد الاتفاق حول السبورة لبعض ساعات للتفكير في المشكلة، قد نستقر على الخطوط العريضة لحل عملي، على سبيل المثال: (1) يتفاعل المستخدمون مع التطبيق من خلال واجهة تعمل في متصفح الويب أو تطبيق الهاتف المحمول؛ (2) يتفاعل تطبيقنا مع محرك قاعدة بيانات من الدرجة التجارية لتبني حالة كل مستخدم والاحتفاظ بسجلات المعاملات التاريخية؛ و(3) في قلب تطبيقنا ، يوضح منطق العمل (يمكنك القول ، العقول brains) لتطبيقنا مجموعة من القواعد التي تحدد كل ظرف يمكن تصوره للإجراء المقابل الذي يجب أن يتخده برنامجاً.

لبناء عقول تطبيقنا، قد نقوم بتوسيع جميع الأحداث المشتركة التي يجب أن يتم التعامل معها برنامجاً. على سبيل المثال، عندما ينقر أحد العملاء لإضافة عنصر إلى سلة التسوق الخاصة به، يجب على برنامجاً إضافة إدخال إلى جدول قاعدة بيانات عربة التسوق، مع ربط معرف المستخدم بمعرف المنتج المطلوب. قد نحاول بعد ذلك مراجعة كل حالة ركينة محتملة، واختبار مدى ملاءمة قواعدها وإجراء أي تعديلات ضرورية. ماذا يحدث إذا بدأ المستخدم الشراء بسلة تسوق فارغة؟ في حين أن عددًا قليلاً من المطورين قد فهموا الأمر بشكل صحيح تماماً في المرة الأولى (قد يستغرق الأمر بعض الاختبارات التجريبية لحل الخلل)، في الغالب، يمكننا كتابة مثل هذه البرامج وإطلاقها بشقة قبل روئية عميل حقيقي. إن قدرتنا على تصميم الأنظمة الآلية التي تقود المنتجات والأنظمة العامة بيدويًا، غالباً في المواقف الجديدة، هي إنجاز معرفي رائع. وعندما تكون قادرًا على ابتكار حلول تعمل 100% في ذلك الوقت، فلا داعي للقلق بشأن التعلم الآلي.

لحسن الحظ بالنسبة للمجتمع المتنامي لعلماء التعلم الآلي ، فإن العديد من المهام التي نرغب في أتمتها لا تتحبني بسهولة إلى براعة الإنسان. تخيل أنك تتجمع حول السبورة البيضاء مع ذكى العقول التي تعرفها ، لكنك هذه المرة تعالج إحدى المشكلات التالية:

- اكتب برنامجاً يتبناً بطقس الغد بناءً على المعلومات الجغرافية وصور القمر الصناعي ونافذة تتبع الطقس السابق.
- اكتب برنامجاً يأخذ سؤالاً واقعياً ، معبراً عنه بنص حر ، والإجابة عليه بشكل صحيح.
- اكتب برنامجاً ، من خلال تقديم صورة ، يحدد جميع الأشخاص الذين تم تصويرهم فيه ويرسم الخطوط العريضة حول كل منهم.

- اكتب برنامجاً يقدم للمستخدمين منتجات من المحتمل أن يستمتعوا بها ولكن من غير المحتمل أن يواجهوها في سياق التصفح الطبيعي.

بالنسبة لهذه المشكلات ، سيواجه حتى نخبة المبرمجين صعوبة في ترميز الحلول من الصفر. يمكن أن تختلف الأسباب. في بعض الأحيان ، يتبع البرنامج الذي نبحث عنه نمطاً يتغير بمرور الوقت ، لذلك لا توجد إجابة صحيحة ثابتة! في مثل هذه الحالات ، يجب أن يتکيف أي حل ناجح برشاقة مع عالم متغير. في أوقات أخرى ، قد تكون العلاقة (على سبيل المثال بين وحدات البكسل والفئات المجردة abstract categories) معقدة للغاية ، وتتطلب آلاف أو ملايين الحسابات واتباع مبادئ غير معروفة. في حالة التعرف على الصور ، تكمن الخطوات الدقيقة المطلوبة لأداء المهمة خارج فهمنا الوعي ، على الرغم من أن عمليات الإدراك اللاواعي لدينا تنفذ المهمة دون عناء.

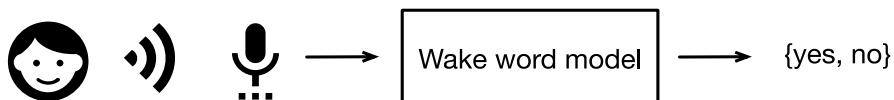
التعلم الآلي Machine learning هو دراسة الخوارزميات التي يمكن أن تتعلم من التجربة. نظراً لأن خوارزمية التعلم الآلي تجمع المزيد من الخبرة ، عادةً في شكل بيانات رصد أو تفاعلات مع بيئه ، فإن أدائها يتحسن. قارن هذا منصة التجارة الإلكترونية الحتمية الخاصة بنا ، والتي تتبع نفس منطق الأعمال ، بعض النظر عن مقدار الخبرة المتراكمة ، حتى يتعلم المطورون أنفسهم ويقررون أن الوقت قد حان لتحديث البرنامج. في هذا الكتاب ، سنعلمك أساسيات التعلم الآلي ، مع التركيز بشكل خاص على التعلم العميق ، ومجموعة قوية من التقنيات التي تقود الابتكارات في مجالات متنوعة مثل الرؤية الحاسوبية computer vision ، ومعالجة اللغة الطبيعية في مجالات مثل الرعاية الصحية healthcare ، natural language processing ، علم الجينوم genomics.

1.1 مثال محفز

قبل البدء في الكتابة ، كان على مؤلفي هذا الكتاب ، مثل الكثير من القوى العاملة ، أن يتناولوا الكافيين. قفزنا في السيارة وبدأنا في القيادة. باستخدام جهاز iPhone ، دعا أليكس "Siri" يا Blue Mu بـ "الاتجاهات إلى مقهى Bottle". عرض الهاتف بسرعة نسخ أمره. كما أدركنا أننا كنا نطلب الاتجاهات وأطلقنا تطبيق الخرائط (التطبيق) لتلبية طلبنا. بمجرد إطلاقه ، حدد تطبيق الخرائط عدداً من المسارات. بجانب كل مسار ، عرض الهاتف وقت عبور موقعه. بينما قمنا بتلفيق هذه القصة لتوفير الراحة التربوية ، فإنها توضح أنه في غضون بضع ثوانٍ فقط ، يمكن أن تتفاعل تفاعلاتنا اليومية مع الهاتف الذكي مع العديد من نماذج التعلم الآلي.

تخيل مجرد كتابة برنامج للرد على كلمة تنبية مثل "Alexa" و "OK Google" و "Hey Siri". جرب برمجتها في غرفة بمفردك باستخدام جهاز كمبيوتر ومحرر كود ، كما هو موضح

في الشكل 1.1.1. كيف تكتب مثل هذا البرنامج من المبادئ الأولى؟ فكر في الأمر ... المشكلة صعبة. كل ثانية ، سيجمع الميكروفون ما يقرب من 44000 عينة. كل عينة هي قياس لسعة الموجة الصوتية. ما القاعدة التي يمكن تعينها بشكل موثوق من مقتطف صوت خام إلى تنبؤات واثقة حول ما إذا كان المقتطف يحتوي على كلمة التنبيه wake؟ إذا كنت عالقاً ، فلا تقلق. نحن لا نعرف كيف نكتب مثل هذا البرنامج من الصفر أيضاً. لهذا السبب نستخدم التعلم الآلي.



الشكل 1.1.1 تحديد كلمة Wake

ها هي الحيلة. في كثير من الأحيان ، حتى عندما لا نعرف كيفية إخبار الكمبيوتر بشكل صريح بكيفية التعين من المدخلات إلى المخرجات ، فإننا مع ذلك قادرون على أداء العمل الفذ المعرف في بأنفسنا. بمعنى آخر ، حتى إذا كنت لا تعرف كيفية برمجة جهاز كمبيوتر للتعرف على كلمة "Alexa" ، فأنت نفسك قادر على التعرف عليها. مسلحين بهذه الإمكانيات ، يمكننا جمع مجموعة بيانات ضخمة تحتوي على أمثلة من المقتطفات الصوتية والتسميات المرتبطة بها ، مما يشير إلى المقتطفات التي تحتوي على كلمة التنبيه. في النهج السائد للتعلم الآلي ، لا نحاول تصميم نظام بشكل صريح للتعرف على كلمات التنبيه. بدلاً من ذلك ، نحدد برنامجاً مرئياً يتم تحديد سلوكه من خلال عدد من المعلمات. ثم نستخدم مجموعة البيانات لتحديد أفضل قيم المعلمات الممكنة ، أي تلك التي تعمل على تحسين أداء برنامجنا فيما يتعلق بمقاييس الأداء المختار.

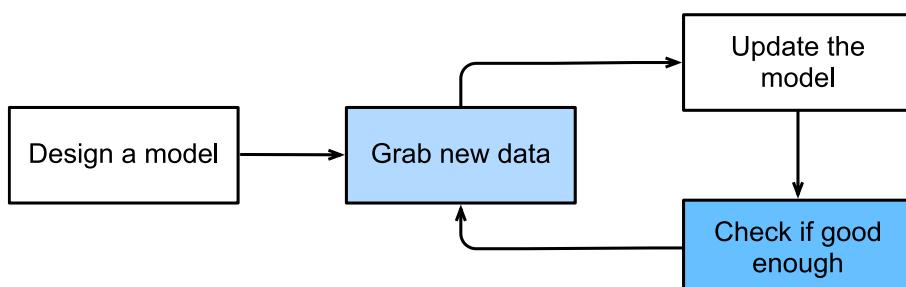
يمكنك التفكير في المعلمات parameters على أنها مقابض يمكننا تشغيلها ، واللاعب بسلوك البرنامج. بتحديد المعلمات ، نسمي البرنامج نموذجاً model. تسمى مجموعة جميع البرامج المميزة (تعينات المدخلات والمخرجات input-output mappings) التي يمكننا إنتاجها فقط من خلال معالجة المعلمات بمجموعة من النماذج. والبرنامج الفوقي الذي يستخدم مجموعة البيانات الخاصة بنا لاختيار المعلمات يسمى خوارزمية التعلم learning algorithm.

قبل أن نتمكن من المضي قدماً وإشراك خوارزمية التعلم ، يتعين علينا تحديد المشكلة بدقة ، وتحديد الطبيعة الدقيقة للمدخلات والمخرجات ، واختيار عائلة نموذجية مناسبة. في هذه الحالة ، يتلقى نموذجنا مقتطفاً من الصوت كمدخلات ، ويقوم النموذج بإنشاء تحديد من بين {yes,no} المخرجات. إذا سارت الأمور وفقاً للنقطة ، فعادة ما تكون تخمينات النموذج صحيحة فيما يتعلق بما إذا كان المقتطف يحتوي على كلمة wake.

إذا اخترنا مجموعة النماذج المناسبة ، فيجب أن يكون هناك إعداد واحد للمقابض بحيث يطلق النموذج "نعم" في كل مرة يسمع فيها كلمة "Alexa". نظرًا لأن الاختيار الدقيق لكلمة الاستيقاظ Wake عشوائي ، فربما نحتاج إلى عائلة نموذجية غنية بما يكفي ، من خلال إعداد آخر للمقابض ، يمكنها إطلاق "نعم" فقط عند سماع كلمة "مشمش". نتوقع أن تكون نفس العائلة النموذجية مناسبة للتعرف على "Alexa" وتقدير الممشمش "Apricot" لأنهما يبدوان ، بشكل حدسي ، مهمتين متشابهتين. ومع ذلك ، قد نحتاج إلى مجموعة مختلفة من النماذج تماماً إذا أردنا التعامل مع مدخلات أو مخرجات مختلفة اختلافاً جزرياً ، لنقل إذا أردنا التعيين من الصور إلى التسميات التوضيحية ، أو من الجمل الإنجليزية إلى الجمل الصينية.

كما قد تتخيل ، إذا قمنا بتعيين جميع المقابض بشكل عشوائي ، فمن غير المرجح أن يتعرف نموذجنا على "Alexa" أو "Apricot" أو أي كلمة إنجليزية أخرى. في التعلم الآلي ، التعلم هو العملية التي نكتشف من خلالها الإعداد الصحيح للمقابض التي تفرض السلوك المطلوب من نموذجنا. بمعنى آخر ، نقوم بتدريب نموذجنا بالبيانات. كما هو مبين في الشكل 1.1.2 ، عادة ما تبدو عملية التدريب كما يلي:

1. ابدأ بنموذج مهياً عشوائياً لا يمكنه فعل أي شيء مفید.
2. احصل على بعض البيانات الخاصة بك (على سبيل المثال ، المقاطفات الصوتية والتسميات {yes,no} المقابلة).
3. قم بتعديل المقابض لجعل النموذج يعمل بشكل أفضل كما تم تقييمه في تلك الأمثلة.
4. كرر الخطوتين 2 و 3 حتى يصبح النموذج رائعاً.



الشكل 1.1.2 عملية تدريب نموذجية.

للتلخيص ، بدلاً من ترميز أداة التعرف على كلمة التنبيه Wake ، نقوم بترميز برنامج يمكنه تعلم التعرف على كلمات التنبيه ، إذا تم تقديمها مع مجموعة بيانات كبيرة معنونة. يمكنك التفكير في هذا الفعل المتمثل في تحديد سلوك البرنامج من خلال تقديمها مع مجموعة بيانات dataset كبرمجة مع البيانات programming with data. وهذا يعني أنه يمكننا "برمجة" جهاز الكشف

عن الققطط من خلال تزويد نظام التعلم الآلي الخاص بنا بالعديد من الأمثلة على الققطط والكلاب. بهذه الطريقة سيعمل الكاشف في النهاية بإصدار رقم موجب كبير جداً إذا كان قطة ، ورقمًا سالبًا كبيرًا جداً إذا كان كلبًا ، وشيء أقرب إلى الصفر إذا لم يكن متأكدًا. هذا بالكاد يخدش سطح ما يمكن أن يفعله التعلم الآلي. التعلم العميق Deep learning، الذي سنشرحه بمزيد من التفصيل لاحقًا ، هو مجرد واحد من بين العديد من الطرق الشائعة لحل مشاكل التعلم الآلي.

1.2 المكونات الرئيسية

في مثال كلمة التنبيه wake، وصفنا مجموعة بيانات تتكون من مقاطف صوتية وتسميات ثنائية ، وقدمنا إحساساً مموجًا يدوياً لكيفية تدريب نموذج لتقرير الخرائط من المقاطف إلى التصنيفات. هذا النوع من المشاكل ، حيث نحاول التنبؤ بعلامة غير معروفة محددة بناءً على المدخلات المعروفة بمجموعة بيانات تتكون من أمثلة معروفة تسمياتها labels ، يسمى التعلم الخاضع للإشراف supervised learning. هذه مجرد واحدة من بين العديد من مشكلات التعلم الآلي. قبل أن نستكشف أصنافاً أخرى ، نود أن نلقي مزيدًا من الضوء على بعض المكونات الأساسية التي ستبعنا ، بغض النظر عن نوع مشكلة التعلم الآلي التي نتعامل معها:

1. البيانات data التي يمكننا التعلم منها.
2. النموذج model لكيفية تحويل البيانات.
3. دالة الهدف objective function تحدد مدى جودة (أو سوء) النموذج.
4. الخوارزمية algorithm لضبط معلمات النموذج لتحسين دالة الهدف.

1.2.1 Data. البيانات

قد يكون من نافلة القول أنه لا يمكنك القيام بعلم البيانات data science بدون بيانات. قد نفقد مئات الصفحات عند التفكير في ماهية البيانات بالضبط ، ولكن في الوقت الحالي ، سنركز على الخصائص الرئيسية لمجموعات البيانات التي سنهتم بها. بشكل عام ، نحن مهتمون بمجموعة من الأمثلة. من أجل العمل مع البيانات بشكل مفيد ، نحتاج عادةً إلى التوصل إلى تمثيل رقمي مناسب. يتكون كل مثال (أو نقطة بيانات data point ، مثل بيانات features (تسمى instance ، عينة sample) عادةً من مجموعة من السمات تسمى الميزات features (تسمى أحياناً المتغيرات المشتركة أو المدخلات) ، بناءً على النموذج الذي يجب أن يقوم بتبنّائه. في مشاكل التعلم الخاضعة للإشراف ، هدفنا هو التنبؤ بقيمة سمة خاصة ، تسمى التسمية label (أو الهدف target) ، والتي ليست جزءًا من مدخلات النموذج.

إذا كنا نعمل مع بيانات الصورة ، فقد يتكون كل مثال من صورة فردية (الميزات features) ورقم يشير إلى الفتة التي تنتمي إليها الصورة (التسمية label). سيتم تمثيل الصورة عدديًا على شكل ثلاث شبكات من القيم الرقمية التي تمثل سطوع الضوء الأحمر والأخضر والأزرق في كل

موقع بكسل. على سبيل المثال 200×200 ، قد تكون الصورة الملونة من $3 \times 200 \times 200 = 120000$ قيم عدديّة.

بدلاً من ذلك ، قد نعمل مع بيانات السجلات الصحية الإلكترونيّة ونتعامل مع مهمة التنبؤ باحتمالية بقاء مريض معين على قيد الحياة خلال الثلاثين يوماً القادمة. هنا ، قد تتكون ميزاتنا من مجموعة من السمات المتاحة بسهولة والقياسات المسجلة بشكل متكرر ، بما في ذلك العمر والعلامات الحيوية والأمراض المصاحبة والأدوية الحالى والإجراءات الحديثة. سيكون التسمية المتاحة label للتدريب عبارة عن قيمة ثنائية تشير إلى ما إذا كان كل مريض في البيانات التاريخية قد نجا خلال نافذة الثلاثين يوماً.

في مثل هذه الحالات، عندما يتميز كل مثال بنفس عدد السمات العددية، نقول إن المدخلات عبارة عن متجهات ذات طول ثابت ونطلق على الطول (الثابت) للمتجهات أبعاد البيانات dimensionality of the data. كما قد تخيل، يمكن أن تكون المدخلات ذات الطول الثابت مريحة، مما يمنحك تعقیداً أقل للقلق. ومع ذلك، لا يمكن بسهولة تمثيل جميع البيانات كمتجهات ثابتة الطول fixed-length vectors. بينما قد تتوقع أن تأتي الصور المجهريّة من معدات قياسية، لا يمكننا أن نتوقع أن تظهر الصور الملغومة من الإنترنّت جميعها بنفس الدقة أو الشكل. بالنسبة للصور، قد نفكّر في اقتصاصها جميّعاً إلى الحجم القياسي، لكن هذه الإستراتيجية تصلنا فقط حتى الآن. نحن نجازف بفقدان المعلومات في الأجزاء المقطعة. علاوة على ذلك، تقاوم البيانات النصية التمثيلات ذات الطول الثابت بشكل أكثر عناًداً. ضع في اعتبارك تقييمات العملاء المتبقية على موقع التجارة الإلكترونيّ مثل TripAdvisor وAmazon وIMDb. بعضها قصير: "يتنـ! it stinks!". يتوجّل آخرون للصفحات. تتمثل إحدى الميزات الرئيسيّة للتعلم العميق على الطرق التقليديّة في النعمة المقارنة التي يمكن للنماذج الحديثة من خلالها التعامل مع بيانات متفاوتة الطول varying-length data.

بشكل عام، كلما زادت البيانات المتوفرة لدينا، أصبحت مهمتها أسهل. عندما يكون لدينا المزيد من البيانات، يمكننا تدريب نماذج أكثر قوّة والاعتماد بشكل أقل على الافتراضات المسبقة. يعدّ تغيير النظام من البيانات الصغيرة (نسبياً) إلى البيانات الكبيرة مساهمًا رئيسيًا في نجاح التعلم العميق الحديث. لتوجيه هذه النقطة الرئيسيّة، لا تعمل العديد من النماذج الأكثر إثارة في التعلم العميق بدون مجموعات البيانات الكبيرة. يعمل البعض الآخر في نظام البيانات الصغيرة، لكنها ليست أفضل من الأساليب التقليدية.

أخيرًا، لا يكفي وجود الكثير من البيانات ومعالجتها بذكاء. نحن بحاجة إلى البيانات الصحيحة. إذا كانت البيانات مليئة بالأخطاء، أو إذا كانت الميزات المختارة لا تتبع بالكمية المستهدفة من الاهتمام، فإن التعلم سيفشل. يتم التقاط الموقف جيداً من خلال الكليشيهات:

القمامات في الداخل، والقمامات خارج garbage in, garbage out. علاوة على ذلك، فإن ضعف الأداء التنبئي ليس هو النتيجة المحتملة الوحيدة. في التطبيقات الحساسة للتعلم الآلي، مثل السياسة التنبؤية predictive policing، واستئناف الفحص resume screening، ونماذج المخاطر risk models المستخدمة للإقرارات lending، يجب أن تكون متقيظين بشكل خاص لعاقب البيانات المهملة data garbage. يحدث أحد أوضاع الفشل الشائعة في مجموعات البيانات حيث لا يتم تمثيل بعض مجموعات الأشخاص في بيانات التدريب. تخيل تطبيق نظام التعرف على سرطان الجلد في البرية لم يسبق له مثيل من قبل. يمكن أن يحدث الفشل أيضًا عندما لا تكون البيانات مجرد تمثيل ناقص لبعض المجموعات ولكنها تعكس التحيزات المجتمعية. على سبيل المثال، إذا تم استخدام قرارات التوظيف السابقة لتدريب نموذج تنبؤي سيتم استخدامه لفحص السير الذاتية، فيمكن لنماذج التعلم الآلي عن غير قصد التقاط المظالم التاريخية التلقائيًا automate historical injustices. لاحظ أن كل هذا يمكن أن يحدث دون أن يتآمر عالم البيانات بنشاط، أو حتى أن يكون على علم.

1.2.2 Models. النماذج

يتضمن معظم التعلم الآلي تحويل البيانات إلى حد ما. قد نرغب في بناء نظام يستوعب الصور ويتبأ بالابتسامة. بدلاً من ذلك، قد نرغب في استيعاب مجموعة من القراءاتأجهزة الاستشعار والتنبؤ بمدى طبيعية مقارنة القراءات الشاذة. حسب النموذج، نشير إلى الآلة الحسابية لاستيعاب البيانات من نوع واحد، وإخراج تنبؤات من نوع مختلف محتمل. على وجه الخصوص، نحن مهتمون بالنماذج الإحصائية التي يمكن تقديرها من البيانات. في حين أن النماذج البسيطة قادرة تماماً على معالجة المشكلات بشكل مناسب، فإن المشكلات التي تركز عليها في هذا الكتاب تزيد من حدود الطرق الكلاسيكية. يختلف التعلم العميق عن الأساليب الكلاسيكية بشكل أساسي من خلال مجموعة النماذج القوية التي يركز عليها. تتكون هذه النماذج من العديد من التحولات المتتالية للبيانات التي يتم ربطها بعضها البعض من أعلى إلى أسفل، وبالتالي يطلق عليها اسم التعلم العميق deep learning. في طريقنا لمناقشة النماذج العميق، سنناقش أيضاً بعض الأساليب التقليدية.

1.2.3 دوال الهدف Objective Functions

في وقت سابق ، قدمتنا التعلم الآلي باعتباره التعلم من التجربة. من خلال التعلم هنا ، فإننا نعني التحسين في بعض المهام بمرور الوقت. ولكن من سيقول ما الذي يشكل تحسينا؟ قد تتخيل أنه يمكننا اقتراح تحديث نموذجنا ، وقد يختلف بعض الأشخاص حول ما إذا كان التحديث المقترن بمثلك تحسيناً أم رفضاً.

من أجل تطوير نظام رياضي رسمي لآلات التعلم ، نحتاج إلى مقاييس رسمية لمدى جودة (أو سوء) نماذجنا. في التعلم الآلي والتحسين بشكل عام ، نسمي هذه دوال الهدف Objective Functions . حسب الاصطلاح ، نحدد عادةً دوال الهدف بحيث يكون الأقل أفضل. هذه مجرد اتفاقية. يمكنك أن تأخذ أي دالة أعلى هو أفضل ، وتحويلها إلى دالة جديدة متطابقة نوعياً ولكن الأقل أفضل من خلال قلب العلامة. لأن الأقل هو الأفضل ، تسمى هذه الدوال أحياناً دوال الخسارة أو الخطأ loss functions.

عند محاولة التنبؤ بالقيم العددية ، فإن دالة الخطأ الأكثر شيوعاً هي الخطأ التربيعي squared error ، أي مربع الفرق بين التنبؤ والهدف الحقيقي target ground truth . بالنسبة للتصنيف، فإن الهدف الأكثر شيوعاً هو تقليل معدل الخطأ ، أي جزء من الأمثلة التي لا تتفق توقعاتنا معها مع الحقيقة الأساسية. من السهل تحسين بعض الأهداف (على سبيل المثال، الخطأ التربيعي squared error) ، بينما يصعب تحسين أهداف أخرى (على سبيل المثال ، معدل الخطأ rate) بشكل مباشر ، بسبب عدم التفضيل أو المضارعات الأخرى. في هذه الحالات ، من الشائع تحسين هدف بديل surrogate objective .

أثناء التحسين ، نفك في الخطأ كدالة لمعلمات النموذج ، ونتعامل مع مجموعة بيانات التدريب باعتبارها ثابتة. نتعلم أفضل القيم لمعلمات نموذجنا من خلال تقليل الخطأ المتبدد على مجموعة تتكون من عدد من الأمثلة التي تم جمعها للتدريب. ومع ذلك ، فإن الأداء الجيد في بيانات التدريب لا يضمن أننا سنعمل بشكل جيد على البيانات غير المرئية. لذلك سنزيد عادةً تقسيم البيانات المتوفرة إلى قسمين: مجموعة بيانات التدريب training dataset (أو مجموعة التدريب training set)، من أجل معلمات نموذج التعلم ؛ ومجموعة بيانات الاختبار test dataset (أو مجموعة الاختبار test set)، والتي تم تعليقها للتقييم. في نهاية اليوم ، نبلغ عادةً عن أداء نماذجنا على كلا القسمين. يمكنك التفكير في الأداء التدريبي باعتباره مشابهاً للدرجات التي يحققها الطالب في امتحانات الممارسة المستخدمة للتحضير لامتحاننهائي حقيقي. حتى لو كانت النتائج مشجعة ، فهذا لا يضمن النجاح في الامتحان النهائي. خلال فترة الدراسة ، قد يبدأ الطالب في حفظ أسئلة الممارسة ، ويبدو أنه يتقن الموضوع ولكنه يتعثر عند مواجهة أسئلة لم يتم رؤيتها من قبل في الاختبار النهائي الفعلي. عندما يكون أداء النموذج جيداً في مجموعة التدريب ولكنه يفشل في التعميم على البيانات غير المرئية ، فإننا نقول إنه يعني من فرط التجهيز او التعلم overfitting مع بيانات التدريب.

1.2.4. خوارزميات التحسين Optimization Algorithms

بمجرد حصولنا على بعض مصادر البيانات والتمثيل ، ونموذج ، ودالة هدف محددة جيداً ، نحتاج إلى خوارزمية قادرة على البحث عن أفضل المعلمات الممكنة لتقليل دالة الخطأ. تعتمد

خوارزميات التحسين الشائعة للتعلم العميق على نهج يسمى التدرج الاستقافي gradient descent باختصار ، في كل خطوة ، تتحقق هذه الطريقة لترى ، لكل معلمة ، الطريقة التي ستتحرك بها خطأ مجموعة التدريب إذا قمت بتشويش هذه المعلمة بمقدار صغير فقط. ثم يقوم بتحديث المعلمة في الاتجاه الذي يقلل من الخطأ.

1.3 أنواع مشاكل التعلم الآلي

مشكلة كلمة الاستيقاظ Wake في مثانا التحفيزي هي مجرد واحدة من بين العديد من المشكلات التي يمكن أن يعالجها التعلم الآلي. لتحفيز القارئ بشكل أكبر وتزويدنا بعض اللغة المشتركة التي ستبعنا في جميع أنحاء الكتاب ، نقدم الآن نظرة عامة واسعة على مشهد تركيبات مشكلة التعلم الآلي.

1.3.1. التعلم الخاضع للإشراف Supervised Learning

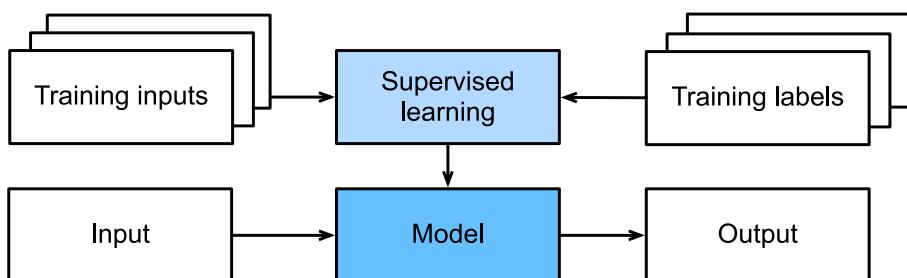
يصف التعلم الخاضع للإشراف المهام حيث يتم تزويدنا بمجموعة بيانات تحتوي على كل من الميزات features والتسميات labels ويتم تكليفنا بإنتاج نموذج للتنبؤ بالتسميات المعطاة لميزات الإدخال. يُطلق على كل زوج من السمات والتسمية (feature-label pair) مثلاً Example. في بعض الأحيان ، عندما يكون السياق واضحًا، قد نستخدم مصطلح الأمثلة للإشارة إلى مجموعة من المدخلات ، حتى عندما تكون التسميات المقابلة غير معروفة. يتم تشغيل الإشراف لأنه من أجل اختيار المعلمات ، فإننا (المشرفون) نوفر للنموذج مجموعة بيانات تتكون من أمثلة مصنفة. من الناحية الاحتمالية ، نحن مهتمون عادةً بتقدير الاحتمال الشرطي لسمات إدخال معينة. في حين أنه مجرد نموذج واحد من بين عدة نماذج في التعلم الآلي ، فإن التعلم الخاضع للإشراف يمثل غالبية التطبيقات الناجحة للتعلم الآلي في الصناعة. يرجع ذلك جزئياً إلى أنه يمكن وصف العديد من المهام المهمة بدقة على أنها تقدر احتمالية وجود شيء غير معروف في ضوء مجموعة معينة من البيانات المتاحة:

- توقع الإصابة بالسرطان مقابل ليس السرطان ، في ضوء صورة التصوير المقطعي بالكمبيوتر.
- توقع الترجمة الصحيحة باللغة الفرنسية ، مع إعطاء جملة بالإنجليزية.
- توقع سعر السهم الشهر المقبل بناءً على بيانات التقارير المالية لهذا الشهر.

بينما يتم التقاط جميع مشكلات التعلم الخاضع للإشراف من خلال الوصف البسيط "التنبؤ بالتسميات المقدمة لميزات الإدخال" ، يمكن أن يتخذ التعلم تحت الإشراف أشكالاً متعددة ويطلب الكثير من قرارات النماذج ، اعتماداً على (من بين اعتبارات أخرى) نوع وحجم وكمية المدخلات والمخرجات. على سبيل المثال ، نستخدم نماذج مختلفة لمعالجة متواليات أطوال عشوائية sequences of arbitrary lengths ومعالجة تمثيلات المتجهات ذات الطول الثابت

سوف نتعمق في العديد من هذه المشاكل خلال fixed-length vector representations هذا الكتاب.

بشكل غير رسمي ، تبدو عملية التعلم كما يلي. أولاً ، احصل على مجموعة كبيرة من الأمثلة التي تُعرف بها الميزات واختر منها مجموعة فرعية عشوائية ، واكتسب تسميات الحقيقة الأساسية لكل منها. في بعض الأحيان ، قد تكون هذه التسميات هي البيانات المتاحة التي تم جمعها بالفعل (على سبيل المثال ، هل مات مريض خلال العام التالي؟) وفي أحياناً أخرى قد تحتاج إلى توظيف شروح بشرية لتسمية البيانات ، (على سبيل المثال ، تعين الصور للفئات). تشكل هذه المدخلات والتسميات المقابلة معاً مجموعة التدريب. تقوم بتغذية مجموعة بيانات التدريب في خوارزمية تعلم خاصة للإشراف ، وهي دالة تأخذ مجموعة بيانات كمدخلات وتخرج دالة أخرى: النموذج الذي تم تعلمه. أخيراً ، يمكننا تغذية المدخلات غير المرئية سابقاً إلى النموذج الذي تم تعلمه ، باستخدام مخرجاته كتنبؤات للتسمية المقابلة. تم رسم العملية الكاملة في الشكل 1.



الشكل 1.3.1 التعلم الخاضع للإشراف.

Regression 1.3.1.1

ربما يكون الانحدار هو أبسط مهمة تعليمية خاضعة للإشراف لتلتف حولها. ضع في اعتبارك ، على سبيل المثال ، مجموعة من البيانات التي تم جمعها من قاعدة بيانات مبيعات المنازل. قد تقوم ببناء جدول ، حيث يتتوافق كل صف مع منزل مختلف ، وكل عمود يتتوافق مع بعض السمات ذات الصلة ، مثل المساحة المربعة للمنزل ، وعدد غرف النوم ، وعدد الحمامات ، وعدد الدقائق (المشي) إلى وسط المدينة. في مجموعة البيانات هذه ، سيكون كل منزل متزلاً محدداً ، وسيكون متوجه الميزة المقابل في صف واحد في الجدول. إذا كنت تعيش في نيويورك أو سان فرانسيسكو ، ولم تكن الرئيس التنفيذي لشركة Amazon أو Google أو Microsoft أو Facebook ، فإن (لقطات مربعة ، عدد غرف النوم ، عدد الحمامات ، مسافة سير) ميزة متوجه لمنزلك قد يبدو مثل: [600,1,1,60]. ومع ذلك ، إذا كنت تعيش في بيتسبرغ ، فقد يبدو

الأمر أشبه [3000,4,3,10]. تعد متجهات الميزات ذات الطول الثابت مثل هذه ضرورية لمعظم خوارزميات التعلم الآلي الكلاسيكية.

ما يجعل مشكلة ما هو الانحدار هو في الواقع شكل الهدف. قل أنك في السوق للحصول على منزل جديد. قد ترغب في تقدير القيمة السوقية العادلة للمنزل ، مع الأخذ في الاعتبار بعض الميزات مثل أعلاه. قد تكون البيانات هنا من قوائم المنازل التاريخية وقد تكون التسميات هي أسعار المبيعات المرصودة. عندما تأخذ التسميات قيمًا رقمية عشوائية (حتى ضمن فترة زمنية معينة) ، فإننا نسمى هذا مشكلة الانحدار. الهدف هو إنتاج نموذج تقارب تنبؤاته بشكل وثيق قيم التسمية الفعلية.

الكثير من المشاكل العملية توصف بسهولة بأنها مشاكل انحدار. يمكن اعتبار توقع التصنيف الذي سيخصصه المستخدم لفيلم ما مشكلة انحدار وإذا صممت خوارزمية رائعة لإنجاز هذا العمل الفذ في عام 2009 ، فربما تكون قد فازت بجائزة Netflix البالغة مليون دولار. توقع طول إقامة المرضى في المستشفى هو أيضًا مشكلة الانحدار. من القواعد الأساسية الجيدة أن أي كم؟ أو كم عددها؟ يجب أن تشير المشكلة إلى الانحدار ، على سبيل المثال:

- كم ساعة ستستغرق هذه الجراحة؟
- كم ستهطل الأمطار في هذه البلدة خلال الساعات الست القادمة؟

حتى لو لم تكن قد عملت مع التعلم الآلي من قبل ، فمن المحتمل أنك عملت من خلال مشكلة الانحدار بشكل غير رسمي. تخيل ، على سبيل المثال ، أنك قمت بإصلاح البالوعات الخاصة بك وأن السباق الخاص بك أمضى 3 ساعات في إزالة الأوساخ من أنابيب الصرف الصحي الخاصة بك. ثم أرسل لك فاتورة بقيمة 350 دولارًا. تخيل الآن أن صديقك استأجر نفس السباق لمدة ساعتين وأنه تلقى فاتورة بقيمة 250 دولارًا. إذا سألك شخص ما بعد ذلك عن المبلغ الذي تتوقعه في فاتورته القادمة لإزالة المواد غير المرغوب فيها ، فقد تضع بعض الافتراضات المعقولة ، مثل زيادة ساعات العمل التي تكلف المزيد من الدولارات. قد تفترض أيضًا أن هناك بعض الرسوم الأساسية وأن المقاول يتضمن رسومًا في الساعة. إذا كانت هذه الافتراضات صحيحة ، فالنظر إلى هذين المثالين من البيانات ، يمكنك بالفعل تحديد هيكل تسعير المقاول: 100 دولار للساعة بالإضافة إلى 50 دولارًا لظهور في منزلك. إذا اتبعت هذا كثيرًا ، فأنت بالفعل تفهم الفكرة رفيعة المستوى وراء الانحدار الخططي .linear regression

في هذه الحالة ، يمكننا إنتاج المعلمات التي تتطابق تماماً مع أسعار السباق. في بعض الأحيان لا يكون هذا ممكناً ، على سبيل المثال ، إذا كان بعض التباين مدينًا بعده عوامل إلى جانب المميزتين. في هذه الحالات ، سنحاول تعلم النماذج التي تقلل المسافة بين تنبؤاتنا والقيم المرصودة. في معظم فصولنا ، سنركز على تقليل دالة الخطأ التربيعية squared error loss

كما سنرى لاحقاً ، تتوافق هذه الخطأ مع الافتراض بأن بياناتنا قد تعرضت للتلف بسبب الضوضاء الغاويسية Gaussian noise .

1.3.1.2. classification

بينما تعتبر نماذج الانحدار رائعة لمعالجة كم عدد؟ الأسئلة ، الكثير من المشاكل لا تنحني بشكل مريح لهذا القالب. ضع في اعتبارك ، على سبيل المثال ، بنكاً يريد تطوير ميزة مسح الشيكات لتطبيقه على الهاتف المحمول. من الناحية المثلية ، يقوم العميل ببساطة بالتقاط صورة للشيك وسيقوم التطبيق تلقائياً بالتعرف على النص من الصورة. بافتراض أن لدينا بعض القدرة على تجزئة تصحيحات الصور المقابلة لكل حرف مكتوب بخط اليد ، فإن المهمة الأساسية المتبقية ستكون تحديد أي حرف من بين مجموعة معروفة يتم تصويره في كل تصحيح صورة. هذه الأنواع من أي واحد؟ تسمى المشاكل التصنيف classification وتحتاج إلى مجموعة من الأدوات مختلفة عن تلك المستخدمة في الانحدار ، على الرغم من أن العديد من التقنيات مستمرة.

في التصنيف ، نريد أن ينظر نموذجنا في الميزات ، على سبيل المثال ، قيم البكسل في صورة ما ، ثم يتبعها بالصنف category (تسمى أحياناً فئة class) من بين مجموعة منفصلة من الخيارات ، مثل ينتمي. بالنسبة للأرقام المكتوبة بخط اليد ، قد يكون لدينا عشر فئات ، مطابقة للأرقام من 0 إلى 9. أبسط شكل من أشكال التصنيف هو عندما يكون هناك فئتان فقط ، وهي مشكلة تسمى التصنيف الثنائي binary classification. على سبيل المثال ، يمكن أن تكون مجموعة البيانات الخاصة بنا من صور للحيوانات وقد تكون تسمياتنا هي الفئات {cat, dog}. بينما في الانحدار ، سعينا إلى معادل لإخراج قيمة عددية ، في التصنيف ، نبحث عن مصنف classifier ، يكون ناتجه هو تحديد الفئة المتوقعة.

لأسباب سوف ندخل فيها عندما يصبح الكتاب أكثر تقنية ، قد يكون من الصعب تحسين نموذج يمكنه فقط إخراج مهمة فئوية صعبة ، على سبيل المثال ، إما "قطة" أو "كلب". في هذه الحالات ، يكون من الأسهل عادةً التعديل عن نموذجنا بـ "الاحتمالات". بالنظر إلى ميزات أحد الأمثلة ، يقوم نموذجنا بتعيين احتمالية لكل فئة ممكنة. بالعودة إلى مثال تصنيف الحيوانات الخاص بنا حيث توجد الفئات ، قد يرى المصنف صورة ويخرج احتمالية أن الصورة قطة بقيمة 0.9. يمكننا تفسير هذا الرقم بالقول إن المصنف متأنق بنسبة 90٪ من أن الصورة تصور قطة. ينقل حجم الاحتمالية للفئة المتوقعة فكرة واحدة عن عدم اليقين uncertainty. إنها ليست الفكرة الوحيدة لعدم اليقين وستناقش الآخرين في فصول أكثر تقدماً.

عندما يكون لدينا أكثر من فئتين محتملتين ، فإننا نسمي المشكلة تصنيف متعدد الفئات multiclass classification. تشمل الأمثلة الشائعة التعرف على الأحرف المكتوبة بخط اليد

{ ... 0,1,2, ... 9, a, b, c, ... } . بينما هاجمنا مشاكل الانحدار من خلال محاولة تقليل دالة خسارة الخطأ التربيعية ، فإن دالة الخسارة الشائعة لمشاكل التصنيف تسمى الانتروبيا المتقاطعة – cross-entropy ، والتي يمكن إزالتها الغموض عن اسمها من خلال مقدمة لنظرية المعلومات في الفصول اللاحقة.

لاحظ أن الفئة الأكثر ترجيحاً ليست بالضرورة هي التي ستستخدمها لاتخاذ قرارك. افترض أنك وجدت فطراً جميلاً في الفناء الخلفي الخاص بك كما هو موضح في الشكل 1.3.2.



الشكل 1.3.2 Death cap – لا تأكل!

الآن ، افترض أنك قمت ببناء مصنف ودربيه على التنبؤ بما إذا كان الفطر ساماً بناءً على صورة فوتوغرافية. لنفترض أن مخرجات مصنف اكتشاف السموم لدينا تشير إلى أن احتمال احتواء الشكل 1.3.2 على فطر سام Death cap هو 0.2. بعبارة أخرى ، المصنف متأكد بنسبة 80٪ أن الفطر ليس ساماً Death cap. ومع ذلك ، يجب أن تكون أحمق لتناوله. وذلك لأن الفائدة المؤكدة من تناول عشاء لذيد لا تساوي 20٪ خطر الموت منه. وبعبارة أخرى ، فإن تأثير المخاطر غير المؤكدة يفوق المنفعة إلى حد بعيد. وبالتالي ، من أجل اتخاذ قرار بشأن تناول الفطر ، نحتاج إلى حساب عدم الانتظام المتوقع المرتبط بكل إجراء والذي يعتمد على كل من النتائج المحتملة والفوائد أو الأضرار المرتبطة بكل منها. في هذه الحالة ، قد يكون فقد الناتج عن تناول الفطر $0.2 \times 0 + 0.8 \times \infty$ ، في حين أن فقدان التخلص منه هو $= 1 \times 0.8$. كان حذرنا مبرراً: كما يخبرنا أي اختصاصي فطريات ، فإن الفطر في الشكل 1.3.2 هو في الواقع Death cap.

يمكن أن يصبح التصنيف أكثر تعقيداً من مجرد التصنيف الثنائي أو متعدد الفئات. على سبيل المثال ، هناك بعض متغيرات التصنيف التي تتناول الفئات المهيكلة بشكل هرمي. في مثل هذه الحالات ، ليست كل الأخطاء متساوية – إذا كان يجب علينا أن نخطئ ، فقد نفضل أن نخطئ

في التصنيف إلى فئة ذات صلة بدلاً من فئة بعيدة. عادة ، يشار إلى هذا التصنيف الهرمي . للإلهام، قد تفكّر في [Linnaeus](#) ، الذي نظم الحيوانات في تسلسل هرمي.

في حالة تصنيف الحيوانات، قد لا يكون من السريع جداً الخلط بين كلب بودل وشنورز ، لكن نموذجنا سيدفع غرامة كبيرة إذا خلط بين كلب بودل وديناصور. قد يعتمد التسلسل الهرمي المناسب على كيفية التخطيط لاستخدام النموذج. على سبيل المثال ، قد تكون الأفاعي الجرسية والثعابين ذات الأربطة قريبة من شجرة النشوء والتطور ، ولكن يمكن أن يكون الخلط بين أفعى الجرد وجرباب مميتاً.

1.3.1.3 وضع العلامات Tagging

تتلاع姆 بعض مشكلات التصنيف بدقة مع إعدادات التصنيف الثنائي أو متعدد الفئات. على سبيل المثال ، يمكننا تدريب مصنف ثنائي عادي لتمييز القطط عن الكلاب. نظراً للحالة الحالية للرؤيا الحاسوبية ، يمكننا القيام بذلك بسهولة باستخدام أدوات جاهزة. ومع ذلك ، بغض النظر عن مدى دقة نموذجنا ، فقد نجد أنفسنا في مأزق عندما يصادف المصنف صورة لموسيقي المدينة في برلين Town Musicians of Bremen ، وهي قصة خيالية ألمانية شهرية تضم أربعة حيوانات (الشكل 1.3.3).



الشكل 1.3.3: حمار ، كلب ، قطة ، وديك.

كما ترى ، تظهر الصورة قطة وديك وكلب وحمار مع بعض الأشجار في الخلفية. عندما نتوقع مواجهة مثل هذه الصور ، قد لا يكون التصنيف متعدد الفئات هو الصيغة الصحيحة للمشكلة. بدلاً من ذلك ، قد نرغب في منح النموذج خيار قول أن الصورة تصور قطة وكلباً وحماراً وديكًا.

تسمى مشكلة تعلم التنبؤ بالفئات التي لا تستبعد بعضها البعض بالتصنيف متعدد التسميات multi-label classification. أفضل وصف لمشاكل وضع العلامات التلقائي هو مشاكل التصنيف متعدد التصنيفات multi-label classification problems. فكر في العلامات التي قد يطبقها الأشخاص على المنشورات في مدونة تقنية ، على سبيل المثال ، "التعلم الآلي" ، "التكنولوجيا" ، "الأدوات" ، "لغات البرمجة" ، "Linux" ، "الحوسبة السحابية" ، "AWS". قد يتم تطبيق 5–10 علامات على المقالة النموذجية. عادةً ، ستعرض العلامات بعض بنية الارتباط correlation structure. من المرجح أن تشير المنشورات حول "الحوسبة السحابية" إلى "AWS" ومن المرجح أن تشير المشاركات حول "التعلم الآلي" إلى "وحدات معالجة الرسومات GPUs".

في بعض الأحيان ، تعتمد مشاكل وضع العلامات هذه علىمجموعات تصنيف هائلة. توظف المكتبة الوطنية للطلب العديد من المعلقين المحترفين الذين يربطون كل مقالة ليتم فهرستها في PubMed بمجموعة من العلامات المستمدة من الأنطولوجيا لعناوين الموضوعات الطبية (MeSH) ، وهي مجموعة من 28000 علامة تقريباً. يعد وضع علامات على المقالات بشكل صحيح أمراً مهماً لأنه يسمح للباحثين بإجراء مراجعات شاملة للأدبيات. هذه عملية تستغرق وقتاً طويلاً وعادةً ما يكون للمضيفين فترة تأخير لمدة عام واحد بين الأرشفة ووضع العلامات. يمكن أن يوفر التعلم الآلي علامات مؤقتة حتى يمكن الحصول على مراجعة يدوية مناسبة لكل مقالة. في الواقع ، لعدة سنوات ، استضافت منظمة BioASQ مسابقات لهذه المهمة.

1.3.1.4 Search

في مجال استرجاع المعلومات information retrieval ، غالباً ما نفرض تصنيفات علىمجموعات من العناصر. خذ بحث الويب على سبيل المثال. لا يتمثل الهدف في تحديد ما إذا كانت صفحة معينة ذات صلة باستعلام ما ، ولكن بدلاً من ذلك ، من بين مجموعة النتائج ذات الصلة التي يجب عرضها بشكل بارز لمستخدم معين. قد يكون أحد الحلول الممكنة هو تعين درجة لكل عنصر في المجموعة أولاً ثم استرداد العناصر ذات التصنيف الأعلى. كانت PageRank ، وهي الخلطة السرية الأصلية وراء محرك بحث Google ، مثلاً مبكراً على نظام التسجيل هذا. ومن الغريب أن الدرجات التي يوفرها نظام ترتيب الصفحات لا تعتمد على الاستعلام الفعلي. بدلاً من ذلك ، اعتمدوا على عامل تصفية بسيط للأهمية لتحديد مجموعة المرشحين ذوي الصلة ثم استخدموها نظام ترتيب الصفحات PageRank لتحديد أولويات الصفحات الأكثر موثوقية. في الوقت الحاضر ، تستخدم محركات البحث التعلم الآلي والنماذج

السلوكية behavioral models للحصول على درجات ذات صلة تعتمد على الاستعلام. هناك مؤتمرات أكاديمية كاملة مخصصة لهذا الموضوع.

1.3.1.5. Recommender Systems. أنظمة التوصية

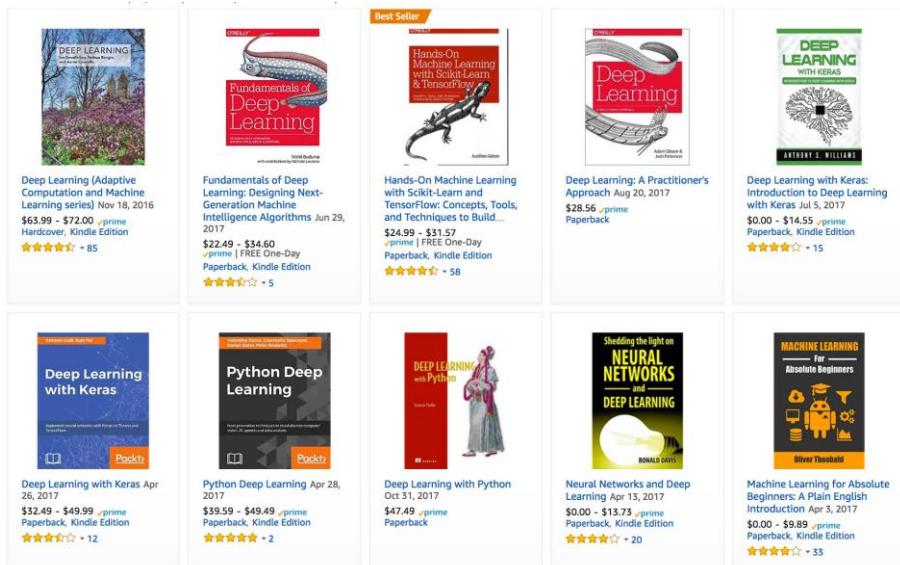
أنظمة التوصية Recommender Systems هي إعداد مشكلة آخر يتعلق بالبحث والتصنيف. تتشابه المشكلات بقدر ما يتمثل الهدف في عرض مجموعة من العناصر ذات الصلة للمستخدم. الاختلاف الرئيسي هو التركيز على التخصيص لمستخدمين محددين في سياق أنظمة التوصية. على سبيل المثال، بالنسبة لتوصيات الأفلام، قد تختلف صفحة النتائج لمحبي الخيال العلمي وصفحة النتائج لمتدوقي الأعمال الكوميدية ليتر سيلرز اختلافاً كبيراً. تظهر المشكلات مماثلة في إعدادات التوصية الأخرى، على سبيل المثال، لمنتجات البيع بالتجزئة والموسيقى والتوصية بالأخبار.

في بعض الحالات، يقدم العملاء ملاحظات صريحة، لإبلاغهم بمدى إعجابهم بمنتج معين (على سبيل المثال، تقييمات المنتج والمراجعات على Goodreads وIMDb وAmazon). في حالات أخرى، يقدمون تعليقات ضمنية، على سبيل المثال، عن طريق تحطيم العناوين في قائمة التشغيل، مما قد يشير إلى عدم الرضا، أو قد يشير فقط إلى أن الأغنية كانت غير مناسبة في السياق. في أبسط الصيغ، يتم تدريب هذه الأنظمة لتقدير بعض النقاط، مثل التصنيف النجمي المتوقع أو احتمال قيام مستخدم معين بشراء عنصر معين.

بالنظر إلى مثل هذا النموذج، لأي مستخدم معين، يمكننا استرداد مجموعة الكائنات ذات الدرجات الأكبر، والتي يمكن بعد ذلك التوصية بها للمستخدم. تعد أنظمة الإنتاج أكثر تقدماً بشكل كبير وتأخذ في الاعتبار نشاط المستخدم المفصل وخصائص العنصر عند حساب مثل هذه الدرجات. يعرض الشكل 1.3.4 كتب التعلم العميق التي أوصت بها أمازون بناءً على خوارزميات التخصيص التي تم ضبطها للتقطاف تفضيلات أستون Aston's preferences.

على الرغم من قيمتها الاقتصادية الهائلة، فإن أنظمة التوصية المبنية بسذاجة على النماذج التنبؤية تعاني من بعض العيوب المفاهيمية الخطيرة. للبدء، نلاحظ فقط التعليقات الخاضعة للرقابة: يفضل المستخدمون تقييم الأفلام التي يشعرون بقوتها تجاهها. على سبيل المثال، على مقياس مكون من خمس نقاط، قد تلاحظ أن العناصر تتلقى العديد من التقييمات ذات النجمة الواحدة والخمس نجوم ولكن هناك عدداً قليلاً من التقييمات الثلاث نجوم بشكل واضح. علاوة على ذلك، غالباً ما تكون عادات الشراء الحالية نتيجة لخوارزمية التوصية المعمول بها حالياً، لكن خوارزميات التعلم لا تأخذ دائماً هذه التفاصيل في الاعتبار. وبالتالي، من الممكن أن تتشكل حلقات التغذية الراجعة حيث يدفع نظام التوصية بشكل تفضيلي عنصراً يتم أخذة ليكون أفضل (بسبب عمليات الشراء الأكبر) وبالتالي يوصى به بشكل متكرر أكثر. العديد من هذه المشاكل

المتعلقة بكيفية التعامل مع الرقابة والحوافز وحلقات التغذية الراجعة هي أسئلة بحثية مفتوحة و مهمة.



الشكل 1.3.4 كتب التعلم العميق التي أوصت بها أمازون.

1.3.1.6. التعلم المتسلسل Sequence Learning

حتى الآن ، نظرنا في المشكلات حيث لدينا عدداً ثابتاً من المدخلات ونتخرج عدداً ثابتاً من المخرجات. على سبيل المثال ، أخذنا في الاعتبار توقع أسعار المنازل في ضوء مجموعة ثابتة من الميزات: المساحة بالقدم المربع ، وعدد غرف النوم ، وعدد الحمامات ، ووقت العبور إلى وسط المدينة. ناقشنا أيضاً التعيين من صورة (ذات بُعد ثابت) إلى الاحتمالات المتوقعة التي تنتهي إليها كل واحدة من بين عدد ثابت من الفئات والتنبؤ بتصنيفات النجوم المرتبطة بالمشتريات بناءً على معرف المستخدم ومعرف المنتج وحده. في هذه الحالات ، بمجرد تدريب نموذجنا ، بعد إدخال كل مثال اختبار في نموذجنا ، يتم نسيانه على الفور. افترضنا أن الملاحظات المتتالية كانت مستقلة وبالتالي لم تكن هناك حاجة للتمسك بهذا السياق.

ولكن كيف نتعامل مع مقاطع الفيديو video snippets ؟ في هذه الحالة ، قد يتكون كل مقطع snippet من عدد مختلف من الإطارات. وقد يكون تخميننا لما يحدث في كل إطار أقوى بكثير إذا أخذنا في الاعتبار الإطارات السابقة أو اللاحقة. الشيء نفسه ينطبق على اللغة. إحدى مشكلات التعلم العميق الشائعة هي الترجمة الآلية machine translation: مهمة استيعاب الجمل في بعض اللغات المصدر والتنبؤ بترجمتها بلغة أخرى.

تحدث هذه المشاكل أيضاً في الطب. قد نرغب في نموذج لمراقبة المرضى في وحدة العناية المركزية وإطلاق التنبؤات عندما يتجاوز خطر الوفاة خلال الـ 24 ساعة القادمة بعض العتبة. هنا، لن نتخلص من كل ما نعرفه عن تاريخ المريض كل ساعة ، ونقوم بالتنبؤات بناءً على أحدث القياسات فقط.

تعد هذه المشكلات من بين أكثر التطبيقات إثارة للتعلم الآلي وهي أمثلة على التعلم المتسلسل sequence learning. إنها تتطلب نموذجاً إما لاستيعاب تسلسلات من المدخلات أو لإصدار تسلسلات من المخرجات (أو كليهما). على وجه التحديد ، يأخذ التعلم من التسلسل إلى التسلسل sequence-to-sequence learning في اعتبار المشكلات التي تتكون فيها المدخلات والمخرجات من متواليات متغيرة الطول. تشمل الأمثلة الترجمة الآلية ونسخ الكلام إلى نص speech-to-text transcription. في حين أنه من المستحيل مراعاة جميع أنواع تحويلات التسلسل ، فإن الحالات الخاصة التالية تستحق الذكر.

العلامات والتحليل Tagging and Parsing. يتضمن هذا التعليق على تسلسل نصي بالسمات. هنا ، تتم محاذاة المدخلات والمخرجات ، أي أنها من نفس العدد وتحدث بترتيب مطابق. على سبيل المثال ، في وضع العلامات على جزء من الكلام part-of-speech (PoS) "Ent" ، تقوم بتعليق توضيحي لكل كلمة في جملة مع الجزء المقابل من الكلام ، أي "اسم" أو "كائن مباشر". بدلاً من ذلك ، قد نرغب في معرفة مجموعات الكلمات المجاورة التي تشير إلى كيانات محددة ، مثل الأشخاص أو الأماكن أو المؤسسات. في المثال الكرتوني البسيط أدناه، قد نرغب فقط في الإشارة ، لكل كلمة في الجملة ، إلى ما إذا كانت جزءاً من كيان مسمى (يُشار إليه باسم "Ent" tagged .)

Tom has dinner in Washington with Sally
Ent - - - Ent - Ent

التعرف التلقائي على الكلام Automatic Speech Recognition. مع التعرف على الكلام ، يكون تسلسل الإدخال عبارة عن تسجيل صوتي لمكبر الصوت (الشكل 1.3.5)، والإخراج عبارة عن نسخة نصية لما قاله المتحدث. التحدي هو أن هناك العديد من الإطارات الصوتية (يتمأخذ عينات الصوت عادةً عند 8 كيلو هرتز أو 16 كيلو هرتز) من النص ، أي أنه لا يوجد تطابق 1: 1 بين الصوت والنص ، حيث قد تتوافقآلاف العينات مع الكلمة منطقاً واحدة. هذه هي مشاكل التعلم من التسلسل إلى التسلسل ، حيث يكون الناتج أقصر بكثير من المدخلات.



الشكل 1.3.5 - في تسجيل صوتي.

النص إلى الكلام **Text to Speech**. هذا هو عكس التعرف التلقائي على الكلام. هنا، الإدخال عبارة عن نص والمخرج عبارة عن ملف صوتي. في هذه الحالة ، يكون الإخراج أطول بكثير من الإدخال. في حين أن البشر بارعون بشكل ملحوظ في التعرف على الكلام ، حتى من الصوت منخفض الجودة ، فإن جعل أجهزة الكمبيوتر تؤدي هذا العمل الفذ يمثل تحديًا هائلاً.

الترجمة الآلية Machine Translation. على عكس حالة التعرف على الكلام ، حيث تحدث المدخلات والمخرجات المقابلة بنفس الترتيب ، في الترجمة الآلية ، تشكل البيانات غير المحاذاة تحديًا جديداً. هنا يمكن أن يكون لسلسلة الإدخال والإخراج أطوال مختلفة ، وقد تظهر المناطق المقابلة من التسلسال المعنية في أوامر مختلفة. تأمل المثال التوضيحي التالي للميل الغريب للألمان لوضع الأفعال في نهاية الجمل:

German: Haben Sie sich schon dieses grossartige Lehrwerk angeschaut?

English: Did you already check out this excellent tutorial?

Wrong alignment: Did you yourself already this excellent tutorial looked-at?

تظهر العديد من المشكلات ذات الصلة في مهام التعلم الأخرى. على سبيل المثال ، تحديد الترتيب الذي يقرأ به المستخدم صفحة ويب هو مشكلة تحليل تخطيط ثنائي الأبعاد *drowsiness detection in Deep Learning: a review* أنواع التعقيبات الإضافية ، حيث يتطلب تحديد ما سيقال بعد ذلك مراعاة معرفة العالم الحقيقي والحالة السابقة للمحادثة عبر مسافات زمنية طويلة. هذه هي مجالات البحث النشطة.

1.3.2. التعلم غير الخاضع للإشراف **Unsupervised** والإشراف الذاتي-**Supervised**

ركزت الأمثلة السابقة على التعلم الخاضع للإشراف **supervised learning**، حيث تقوم بتغذية النموذج بمجموعة بيانات عملاقة تحتوي على كل من الميزات وقيم التسمية المقابلة. يمكنك التفكير في أن المتعلم الخاضع للإشراف لديه وظيفة متخصصة للغاية ورئيس ديكاتورى للغوية. يقف الرئيس فوق كتفه ويخبره بالضبط بما يجب القيام به في كل موقف حتى تتعلم التخطيط من المواقف إلى الإجراءات. يبدو أن العمل لدى مثل هذا الرئيس ضعيف جداً. من ناحية أخرى ، فإن إرضاء مثل هذا الرئيس أمر سهل للغاية. أنت فقط تعرف على النمط بأسرع ما يمكن وتقليل أفعالهم.

بالنظر إلى الموقف المعاكس ، قد يكون من المحبط العمل لدى رئيس ليس لديه فكرة عما يريدون منك القيام به. ومع ذلك ، إذا كنت تخطط لأن تكون عالم بيانات ، فمن الأفضل أن تعتاد عليها. قد يقوم المدير بتسليمك كمية هائلة من البيانات ويخبرك بالقيام ببعض علوم البيانات

باستخدامها! هذا يبدو غامضا لأنه كذلك. نحن نطلق على هذه الفئة من المشكلات اسم التعلم غير الخاضع للإشراف unsupervised learning، نوع وعدد الأسئلة التي يمكن أن نطرحها يقتصر على إبداعنا فقط. سوف نتناول تقنيات التعلم غير الخاضعة للإشراف في فصول لاحقة. لإثارة شهيتك في الوقت الحالي ، نصف بعض الأسئلة التالية التي قد تطرحها.

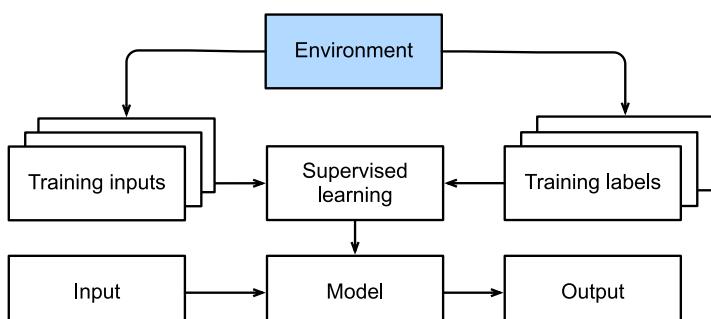
- هل يمكننا العثور على عدد صغير من النماذج الأولية التي تلخص البيانات بدقة؟ بالنظر إلى مجموعة من الصور ، هل يمكننا تجميعها في صور مناظر طبيعية وصور الكلاب والربيع والقطط وقم الجبال؟ وبالمثل ، في ضوء مجموعة من أنشطة تصفح المستخدمين، هل يمكننا تجميعها في مستخدمين لديهم سلوك مشابه؟ تُعرف هذه المشكلة عادةً باسم التجميع clustering.
- هل يمكننا العثور على عدد صغير من المعلمات التي تلتقط بدقة الخصائص ذات الصلة للبيانات؟ مسارات الكرة موصوفة جيداً حسب سرعة الكرة وقطرها وكتلتها. لقد طور الخياطون عدداً صغيراً من المعلمات التي تصف شكل جسم الإنسان بدقة إلى حد ما لغرض ملائمة الملابس. يشار إلى هذه المشاكل باسم تقدير الفضاء الجزيئي subspace estimation. إذا كان الاعتماد خطياً linear estimation ، فإنه يسمى تحليل المكون الأساسي Principal Component Analysis(PCA).
- هل هناك تمثيل للكائنات (منظم بشكل تعسفي arbitrarily structured) في الفضاء الإقليدي بحيث يمكن مطابقة الخصائص الرمزية بشكل جيد؟ يمكن استخدام هذا لوصف الكيانات وعلاقتها ، مثل "روما" - "إيطاليا" + "فرنسا" = "باريس".
- هل هناك وصف للأسباب الجذرية لكثير من البيانات التي نلاحظها؟ على سبيل المثال ، إذا كانت لدينا بيانات ديموغرافية حول أسعار المنازل والتلوث والجريمة والموقع والتعليم والرواتب ، فهل يمكننا اكتشاف كيفية ارتباطها بناءً على البيانات التجريبية empirical data؟ المجالات المعنية بالسببية causality والنماذج الرسومية الاحتمالية probabilistic graphical models تعالج مثل هذه الأسئلة.
- التطور الأخير المهم والمثير في التعلم غير الخاضع للإشراف هو ظهور النماذج التوليدية العميقه deep generative models. تقدر هذه النماذج كثافة البيانات، سواء بشكل صريح أو ضمني. بمجرد التدريب ، يمكننا استخدام نموذج توليد إما لتسجيل الأمثلة وفقاً لمدى احتمالية وجودها ، أو لأخذ عينات من الأمثلة التركيبة من التوزيع المكتسب. جاءت اختراعات التعلم العميق المبكرة في النماذج التوليدية مع اختراع المشفرات التلقائية المتعددة variational autoencoders (Kingma and Welling 2014) واستمرت في تطوير شبكات الخصومة

،Goodfellow et al. (2014) generative adversarial networks التوليدية (GANs). تشمل التطورات الحديثة تعبيع التدفقات normalizing flows ونماذج الانتشار diffusion models والنماذج القائمة على النقاط score-based models.

كان أحد التطورات الرئيسية في التعلم غير الخاضع للإشراف هو ظهور التعلم تحت الإشراف الذاتي self-supervised learning، وهي التقنيات التي تستفيد من بعض جوانب البيانات غير المصنفة ل توفير الإشراف. بالنسبة للنصوص، يمكننا تدريب النماذج على "ملء الفراغات fill in the blanks" من خلال التنبؤ بالكلمات المقترنة عشوائياً باستخدام الكلمات المحيطة بها (السياقات contexts) في مجموعات كبيرة دون بذل أي جهد في وضع العلامات (Devlin et al., 2018)! بالنسبة للصور، قد نقوم بتدريب النماذج لإخبار الموضع النسبي بين منطقتين تم اقتصاها من نفس الصورة (Doersch et al., 2015)، للتنبؤ بجزء مغلق من الصورة بناءً على الأجزاء المتبقية من الصورة، أو للتنبؤ بما إذا كان مثلاً هما إصدارات مضطربة من نفس الصورة الأساسية. غالباً ما تتعلم النماذج الخاضعة للإشراف الذاتي التمثيلات التي يتم الاستفادة منها لاحقاً عن طريق ضبط النماذج الناتجة في بعض المهام النهائية ذات الأهمية.

1.3.3. التعامل مع البيئة Interacting with an Environment

حتى الآن، لم نناقش من أين تأتي البيانات فعلياً، أو ما يحدث بالفعل عندما يولد نموذج التعلم الآلي مخرجات. وذلك لأن التعلم تحت الإشراف والتعلم غير الخاضع للإشراف لا يعالجان هذه القضية بطريقة معقّدة للغاية. في كلتا الحالتين، نحصل على كومة كبيرة من البيانات مقدماً، ثم نضع آلات التعرف على الأنماط الخاصة بنا في حالة حركة دون التفاعل مع البيئة مرة أخرى. نظراً لأن التعلم كله يحدث بعد فصل الخوارزمية عن البيئة، يُسمى هذا أحياناً التعلم الأولافلين offline learning. على سبيل المثال، يفترض التعلم الخاضع للإشراف نمط التفاعل البسيط الموضح في الشكل 1.3.6.



الشكل 1.3.6 جمع البيانات للتعلم الخاضع للإشراف من بيئه ما.

إن بساطة التعلم الافتراضي له سحره. الجانب الإيجابي هو أنه يمكننا القلق بشأن التعرف على الأنماط بشكل منفصل ، دون القلق بشأن المضاعفات الناشئة عن التفاعلات مع بيئه ديناميكية. لكن صياغة هذه المشكلة محدودة. إذا نشأت في قراءة روايات روبوت Asimov ، فقد تتخيل عمالء ذكاء اصطناعياً ليسوا فقط قادرين على التنبؤ ، ولكن أيضاً على اتخاذ الإجراءات في العالم. نريد أن نفكر في العوامل الذكية، وليس فقط النماذج التنبؤية. هذا يعني أننا بحاجة إلى التفكير في اختيار الإجراءات، وليس مجرد التنبؤ. على عكس مجرد التنبؤات ، تؤثر الإجراءات فعلياً على البيئة. إذا أردنا تدريب وكيل ذكي intelligent agent, يجب أن نحسب الطريقة التي قد تؤثر بها أفعاله على الملاحظات المستقبلية للوكيل agent.

إن التفكير في التفاعل مع البيئة يفتح مجموعة كاملة من أسئلة النمذجة الجديدة. ما يلي مجرد أمثلة قليلة.

- هل تتذكر البيئة ما فعلناه سابقاً؟
- هل تريد البيئة مساعدتنا، على سبيل المثال، مستخدم يقرأ النص في أداة التعرف على الكلام؟
- هل تريد البيئة التغلب علينا، على سبيل المثال، تغيير مرسل البريد الإلكتروني العشوائي لتفادي عوامل تصفيية البريد العشوائي؟
- هل البيئة لديها ديناميات متغيرة؟ على سبيل المثال، هل تشبه البيانات المستقبلية الماضي دائماً أم أن الأنماط تتغير بمرور الوقت، إما بشكل طبيعي أو استجابة لأدواتنا الآلية؟

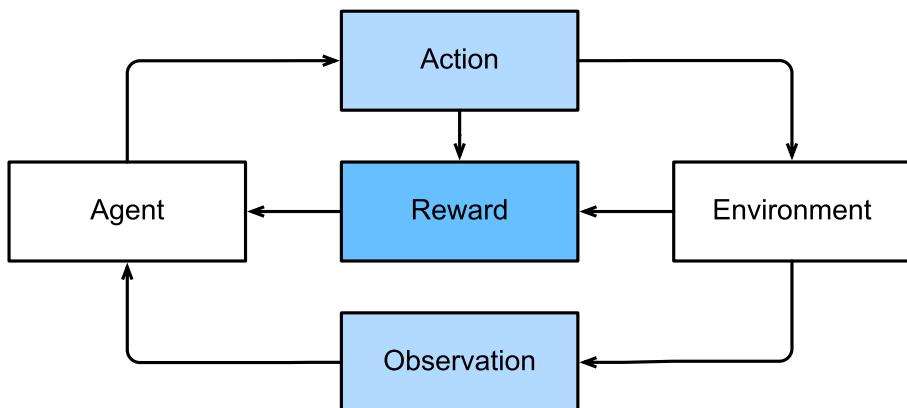
تشير هذه الأسئلة مشكلة تحول التوزيع distribution shift، حيث تختلف بيانات التدريب والاختبار. لقد واجه معظمنا هذه المشكلة عند إجراء الاختبارات التي كتبها محاضر ، بينما كان الواجب المنزلي مؤلفاً من قبل مساعديهم التدرسيين. بعد ذلك ، نصف بـ reinforcement learning يطبق ذلك تطبيقات للروبوتات وأنظمة الحوار وحتى تطوير الذكاء الاصطناعي (AI) لألعاب الفيديو. لقد ازدادت شعبية التعلم المعزز العميق Deep reinforcement learning ، الذي يطبق التعلم العميق لمشاكل التعلم المعزز. إن شبكة Q-network العميقه التي تغلبت على البشر في ألعاب Atari باستخدام المدخلات المرئية فقط (Mnih et al. 2015) ، وبرنامج

1.3.4. التعلم المعزز Reinforcement Learning

إذا كنت مهتماً باستخدام التعلم الآلي لتطوير عامل يتفاعل مع بيئه ويتخذ إجراءات، فمن المحتمل أن يتهمي بك الأمر بالتركيز على التعلم المعزز reinforcement learning. قد يشمل ذلك تطبيقات للروبوتات وأنظمة الحوار وحتى تطوير الذكاء الاصطناعي (AI) لألعاب الفيديو. لقد ازدادت شعبية التعلم المعزز العميق Deep reinforcement learning ، الذي يطبق التعلم العميق لمشاكل التعلم المعزز. إن شبكة Q-network العميقه التي تغلبت على البشر في ألعاب Atari باستخدام المدخلات المرئية فقط (Mnih et al. 2015) ، وبرنامج

AlphaGo الذي فاز عالي بطل العالم في لعبة اللوحة Go (Silver et al., 2016) هما مثالان بارزان.

يقدم التعلم المعزز بياناً عاماً جداً للمشكلة ، حيث يتفاعل الوكيل agent مع بيئه عبر سلسلة من الخطوات الزمنية steps time. في كل خطوة زمنية ، يتلقى الوكيل بعض الملاحظات observation من البيئة ويجب عليه اختيار إجراء action يتم نقلاً إلى البيئة عبر آلية ما (تسمى أحياناً المشغل actuator). أخيراً ، يتلقى الوكيل مكافأة reward من البيئة. هذه العملية موضحة في الشكل 1.3.7. ثم يتلقى الوكيل ملاحظة لاحقة subsequent observation، ويختار إجراءً لاحقاً ، وما إلى ذلك. يخضع سلوك عامل التعلم المعزز لسياسة policy. باختصار، السياسة هي مجرد دالة تقوم بالخطيط من ملاحظات البيئة إلى الإجراءات. الهدف من التعلم المعزز هو إنتاج سياسات جيدة.



الشكل 1.3.7 التفاعل بين التعلم المعزز والبيئة.

من الصعب المبالغة في عمومية إطار التعلم المعزز. على سبيل المثال ، يمكننا أن نلتقي بمشاكل التعلم الخاضع للإشراف على أنها مشاكل تعلم معزز. لنفترض أن لدينا مشكلة في التصنيف. يمكننا إنشاء عامل تعلم معزز بإجراء واحد يتوافق مع كل فئة. يمكننا بعد ذلك إنشاء بيئه تمنح مكافأة تساوي تماماً دالة الخطأ من مشكلة التعلم الأصلية الخاضعة للإشراف.

ومع ذلك ، يمكن أن يعالج التعلم المعزز أيضاً العديد من المشكلات التي لا يستطيع التعلم الخاضع للإشراف القيام بها. على سبيل المثال ، في التعلم الخاضع للإشراف، تتوقع دائمًا أن تكون مدخلات التدريب مرتبطة بالتسمية الصحيحة. لكن في التعلم المعزز، لا نفترض أنه بالنسبة لكل ملاحظة تخبرنا البيئة بالعمل الأمثل. بشكل عام ، نحن فقط نحصل على بعض المكافآت. علاوة على ذلك ، قد لا تخبرنا البيئة حتى عن الإجراءات التي أدت إلى المكافأة.

تأمل في لعبة الشطرنج. تأتي إشارة المكافأة الحقيقة الوحيدة في نهاية اللعبة عندما نفوز ، ونكتسب مكافأة ، على سبيل المثال، 1 ، أو عندما نخسر ، نحصل على مكافأة ، على سبيل المثال، -1. لذا يجب على المتعلمين المعززين Reinforcement learners التعامل مع مشكلة تعين الائتمان credit assignment problem: تحديد الإجراءات التي يجب إلقاء اللوم عليها في النتيجة. الشيء نفسه ينطبق على الموظف الذي حصل على ترقية في 11 أكتوبر. من المحتمل أن تعكس هذه الترقية عدداً كبيراً من الإجراءات المختارة جيداً خلال العام السابق. يتطلب الحصول على المزيد من الترقيات في المستقبل معرفة الإجراءات التي أدت إلى الترقية.

قد يضطر المتعلمون المعززون أيضاً إلى التعامل مع مشكلة الملاحظة الجزئية partial observability. أي أن الملاحظة الحالية قد لا تخبرك بكل شيء عن حاليك الحالية. لنفترض أن روبوت التنظيف وجد نفسه محاصراً في واحدة من العديد من الخزانات المماثلة في المنزل. قد يتطلب استنتاج الموقع الدقيق للروبوت النظر في ملاحظاته السابقة قبل دخول الخزانة.

أخيراً ، في أي نقطة معينة، قد يعرف المتعلمو التعزيز سياسة جيدة واحدة ، ولكن قد يكون هناك العديد من السياسات الأفضل الأخرى التي لم يجريها الوكيل مطلقاً. يجب على متعلم التعزيز أن يختار باستمرار ما إذا كان سيسعى أفضل استراتيجية معروفة (حالياً) كسياسة ، أو استكشاف مساحة الاستراتيجيات، مما قد يتخلّى عن بعض المكافآت قصيرة المدى في مقابل المعرفة.

مشكلة التعلم المعزز العامة هي بيئة عامة للغاية very general setting تؤثر الإجراءات على الملاحظات اللاحقة. يتم ملاحظة المكافآت فقط وفقاً للإجراءات المختارة. قد تتم ملاحظة البيئة بشكل كامل أو جزئي. قد يسأل الكثير من الباحثين عن تفسير كل هذا التعقيد دفعه واحدة. علاوة على ذلك، لا تظهر كل مشكلة عملية كل هذا التعقيد. نتيجة لذلك ، درس الباحثون عدداً من الحالات الخاصة لمشاكل التعلم المعزز.

عندما تتم ملاحظة البيئة بالكامل، فإننا نطلق على مشكلة التعلم المعزز عملية قرار ماركوف Markov decision process. عندما لا تعتمد الحالة على الإجراءات السابقة، فإننا نطلق على المشكلة اسم مشكلة قطاع الطريق السريع contextual bandit problem. عندما لا تكون هناك حالة ، فقط مجموعة من الإجراءات المتاحة مع مكافآت غير معروفة في البداية ، فإن هذه المشكلة هي مشكلة ماكينات الألعاب المتعددة الكلاسيكية classic multi-armed bandit problem.

1.4 الجذور Roots

لقد راجعنا للتو مجموعة فرعية صغيرة من المشكلات التي يمكن للتعلم الآلي معالجتها. بالنسبة لمجموعة متنوعة من المشكلات التعلم الآلي، يوفر التعلم العميق أدوات قوية لحلها. على

الرغم من أن العديد من أساليب التعلم العميق هي اختراعات حديثة ، فقد تمت دراسة الأفكار الأساسية وراء التعلم من البيانات لعدة قرون. في الواقع ، كان لدى البشر الرغبة في تحليل البيانات والتنبؤ بالنتائج المستقبلية لفترة طويلة والكثير من العلوم الطبيعية لها جذورها في ذلك. على سبيل المثال ، تمت تسمية توزيع برنولي Bernoulli distribution على اسم جاكوب برنولي (1655–1705)، واكتشف كارل فريدريش غاوس(1777–1855) التوزيع الغاوسي Gaussian distribution . اخترع، على سبيل المثال ، خوارزمية المربعات الأقل متوسطا least squares algorithm ، والتي لا تزال تُستخدم حتى اليوم لمشاكل لا حصر لها من حسابات التأمين إلى التشخيص الطبي. أدت هذه الأدوات إلى ظهور نهج تجريبي في العلوم الطبيعية – على سبيل المثال ، قانون أوم المتعلق بالتيار والجهد في المقاوم موصوفاً تماماً بواسطة نموذج خططي.

حتى في العصور الوسطى ، كان لدى علماء الرياضيات حدس شديد للتقديرات. على سبيل المثال ، يوضح كتاب الهندسة لجاكوب كوبيل (1460–1533) متوسط طول قدم الرجل البالغ 16 لتقدير متوسط طول القدم في السكان (الشكل 1.4.1).



شكل 1.4.1 تقدير طول القدم.

عندما خرجت مجموعة من الأفراد من الكنيسة ، طلب من 16 رجلاً بالغاً الوقوف في صف واحد وقياس أقدامهم. ثم تم قسمة مجموع هذه القياسات على 16 للحصول على تقدير لما يصل الآن إلى قدم واحدة. تم تحسين هذه "الخوارزمية" فيما بعد للتعامل مع الأقدام المشوهة. تم طرد الرجلين ذوي أقصر وأطول أقدام ، بمتوسط أكثر من الباقى فقط. هذا من بين الأمثلة المبكرة لتقدير المتوسط المقطوع trimmed mean estimate.

لقد انطلقت الإحصائيات حفأً من خلال جمع البيانات وتوافرها. ساهم أحد روادها ، رونالد فيشر (1890–1962) ، بشكل كبير في نظريتها وتطبيقاتها أيضًا في علم الوراثة. لا تزال العديد من خوارزمياته (مثل تحليل التمايز الخطي linear discriminant analysis) والصيغ (مثل مصفوفة معلومات فيشر Fisher information matrix) تحتل مكانة بارزة في أساس الإحصاء الحديث. حتى موارد البيانات الخاصة به كان لها تأثير دائم. لا تزال مجموعة بيانات Iris التي أصدرها فيشر في عام 1936 مستخدمة أحياناً لإثبات خوارزميات التعلم الآلي. كان فيشر أيضًا مؤيدًا لعلم تحسين النسل eugenics ، والذي يجب أن يذكرنا بأن الاستخدام المشكوك فيه أخلاقيًا لعلم البيانات له تاريخ طويل ودائم مثل استخدامه المنتج في الصناعة والعلوم الطبيعية.

جاء التأثير الثاني للتعلم الآلي من نظرية المعلومات لكلاود شانون (1916–2001) ونظرية الحساب عبر آلان تورينج (1912–1954). طرح تورينج السؤال "هل يمكن للألات أن تفكّر؟" في مقالته الشهيرة Computing Machinery and Intelligence (Turing, 1950) ما وصفه باختبار تورينج Turing test ، يمكن اعتبار الآلة ذكية intelligent إذا كان من الصعب على المقيم البشري التمييز بين الردود من الآلة والإنسان بناءً على التفاعلات النصية.

يمكن العثور على تأثير آخر في علم الأعصاب neuroscience وعلم النفس psychology . بعد كل شيء، من الواضح أن البشر يظهرون سلوكًا ذكيًا. تسائل العديد من العلماء عما إذا كان بإمكان المرء تفسير هذه القدرة وربما عكسها. أحد أقدم الخوارزميات المستوحاة من الناحية البيولوجية صاغها دونالد هب (1904–1985). في كتابه الرائد تنظيم السلوك The Organization of Behavior (Hebb and Hebb, 1949) ، افترض أن الخلايا العصبية تتعلم من خلال التعزيز الإيجابي. أصبح هذا معروفاً باسم قاعدة التعلم Hebbian. هذه الأفكار المستوحاة من أعمال لاحقة مثل خوارزمية تعلم الإدراك الحسي لروزنبلات Rosenblatt's perceptron learning التي تدعم التعلم العميق اليوم: تعزيز السلوك المرغوب وتقليل السلوك غير المرغوب فيه للحصول على إعدادات جيدة للمعلمات في الشبكة العصبية.

الإلهام البيولوجي Biological inspiration هو ما أعطى الشبكات العصبية اسمها. لأكثر من قرن (يعود تاريخها إلى نماذج ألكسندر باین ، 1873 وجیمس شیرینجتون ، 1890) ، حاول

الباحثون تجمعوا دوائر حسابية تشبه شبكات الخلايا العصبية المترابطة. بمرور الوقت ، أصبح تفسير علم الأحياء أقل حرافية ، لكن الاسم عالق. تكمن في جوهرها بعض المبادئ الأساسية التي يمكن العثور عليها في معظم الشبكات اليوم:

- غالباً ما يشار إلى تبديل وحدات المعالجة الخطية وغير الخطية باسم الطبقات layers.
- استخدام قاعدة السلسلة chain rule (المعروفة أيضاً باسم backpropagation) لضبط المعلومات في الشبكة بأكملها مرة واحدة.

بعد التقدم السريع الأولي، ضعفت الأبحاث في الشبكات العصبية من حوالي عام 1995 حتى عام 2005. وهذا يرجع أساساً إلى سببين. أولاً ، يعد تدريب شبكة ما مكلفاً للغاية من الناحية الحسابية. بينما كانت ذاكرة الوصول العشوائي وفيرة في نهاية القرن الماضي ، كانت القوة الحسابية نادرة. ثانياً ، كانتمجموعات البيانات صغيرة نسبياً. في الواقع ، كانت مجموعة بيانات Fisher's Iris من عام 1932 أداة شائعة لاختبار فاعلية الخوارزميات. اعتبرت مجموعة بيانات MNIST بأرقامها المكتوبة بخط اليد البالغ عددها 60000 صورة.

نظرًا لندرة البيانات والحسابات، أثبتت الأدوات الإحصائية القوية مثل طرق النواة kernel methods وأشجار القرار decision trees والنمذج الرسومية graphical models أنها متفوقة تجريبيًا في العديد من التطبيقات. علاوة على ذلك ، على عكس الشبكات العصبية neural networks ، لم يحتاجوا إلى أسابيع للتدريب وقدمو نتائج يمكن التنبؤ بها مع ضمانات نظرية قوية.

1.5 الطريق إلى التعلم العميق The Road to Deep Learning

تغير الكثير من هذا مع توفر كميات كبيرة من البيانات ، بسبب شبكة الويب العالمية ، وظهور الشركات التي تخدم مئات الملايين من المستخدمين عبر الإنترنت ، ونشر أجهزة استشعار رخيصة وعالية الجودة ، وتخزين بيانات رخيص (قانون Kryder)، والحساب الرخيص (قانون Moore). على وجه الخصوص، تم إحداث ثورة في مشهد الحساب في التعلم العميق من خلال التقدم في وحدات معالجة الرسومات GPU، والتي تم تصميمها في الأصل لألعاب الكمبيوتر. فجأة ، أصبحت الخوارزميات والنمذج التي بدأنا بذرها في الناحية الحسابية ذات صلة (والعكس صحيح). هذا هو أفضل توضيح في القسم 1.5.

Decade	Dataset	Memory	Floating point calculations per second
1970	100 (Iris)	1 KB	100 KF (Intel 8080)
1980	1 K (house prices in Boston)	100 KB	1 MF (Intel 80186)
1990	10 K (optical character recognition)	10 MB	10 MF (Intel 80486)
2000	10 M (web pages)	100 MB	1 GF (Intel Core)
2010	10 G (advertising)	1 GB	1 TF (Nvidia C2050)
2020	1 T (social network)	100 GB	1 PF (Nvidia DGX-2)

الجدول: مجموعة البيانات مقابل ذاكرة الكمبيوتر والقدرة الحسابية

لاحظ أن ذاكرة الوصول العشوائي لم تواكب نمو البيانات. في الوقت نفسه، فاقت الزيادات في القوة الحسابية النمو فيمجموعات البيانات. هذا يعني أن النماذج الإحصائية تحتاج إلى أن تصبح أكثر كفاءة في استخدام الذاكرة، وأن تكون حرفيًّا إنفاق المزيد من دورات الكمبيوتر لتحسين المعلومات، بسبب زيادة ميزانية الحساب. وبالتالي، انتقلت النقطة المثلالية في التعلم الآلي والإحصاءات من النماذج الخطية (المعممة) وطرق النواة إلى الشبكات العصبية العميقه. هذا أيضًا أحد الأسباب التي تجعل العديد من الدعائم الأساسية للتعلم العميق، مثل البيرسيبترون متعدد الطبقات McCulloch and Pitts) multilayer perceptron (1943، و الشبكات العصبية التلافيفية convolutional neural networks (LeCun et al. 1998)، والذاكرة طويلة قصيرة المدى long short-term memory (Hochreiter and Schmidhuber 1992)، وWatkins and Dayan 1997)، و Q-Learning (أعيد اكتشافهما" في العقد الماضي ، بعد أن ظل خامدًا نسبيًّا لفترة طويلة.

تم تشبيه التقدم الأخير في النماذج والتطبيقات والخوارزميات الإحصائية في بعض الأحيان بالانفجار الكمبيري Cambrian explosion: لحظة تقدم سريع في تطور الأنواع. في الواقع، ليست أحدث ما توصلت إليه التكنولوجيا مجرد نتيجة للموارد المتاحة، المطبقة على خوارزميات عمرها عقود. لاحظ أن القائمة أدناه بالكاد تخدش سطح الأفكار التي ساعدت الباحثين على تحقيق تقدم هائل خلال العقد الماضي.

- ساعدت الأساليب الجديدة للتحكم في السعة، مثل الحذف العشوائي dropout (Srivastava et al., 2014) في التخفيف من فرط التعلم overfitting. هنا، يتم حقن الضوضاء (Bishop, 1995) في جميع أنحاء الشبكة العصبية أثناء التدريب.
- حلت آليات الانتباه Attention mechanisms مشكلة ثانية ابتليت بها الإحصائيات لأكثر من قرن: كيفية زيادة الذاكرة وتعقيد النظام دون زيادة عدد

المعلمات القابلة للتعلم. وجد الباحثون حلاً أنيقاً باستخدام ما يمكن اعتباره هيكل مؤشر قابل للتعلم إلى تذكر تسلسل نصي كامل، على سبيل المثال، Bahdanau et al. (learnable pointer structure 2014). بدلاً من الاضطرار إلى تذكر تسلسل نصي كامل، على سبيل المثال، للترجمة الآلية في تمثيل ثابت الأبعاد، كان كل ما يلزم تخزينه هو مؤشر إلى الحالة الوسيطة لعملية الترجمة. سمح ذلك بزيادة الدقة بشكل كبير للتسلسلات الطويلة، حيث لم يعد النموذج بحاجة إلى تذكر التسلسل بالكامل قبل البدء في إنشاء تسلسل جديد. مبنية فقط على آليات الانتباه، أظهرت بنية المحولات transformer (Vaswani et al., 2017) نجاحاً مقنعًا في مجموعة واسعة من المجالات. على سبيل المثال، يمكن لمحول واحد تم اختباره مسبقاً على طرائق متنوعة مثل النصوص والصور وعزم الدوران joint torques تشغيل Atari والصور التوضيحية والدردشة والتحكم في الروبوت (Reed et al., 2022).

- سمحت التصاميم متعددة المراحل Multi-stage designs ، على سبيل المثال، عبر شبكات الذاكرة (Sukhbaatar et al. 2015) والمبرمج العصبي – المترجم neural programmer–interpreter (Reed and De Freitas 2015) للمصممين الإحصائيين لوصف الأساليب التكرارية للاستدلال. تسمح هذه الأدوات بتعديل الحالة الداخلية للشبكة العصبية العميقه بشكل متكرر، وبالتالي تنفيذ خطوات لاحقة في سلسلة من التفكير، على غرار الطريقة التي يمكن بها للمعالج تعديل الذاكرة لإجراء عملية حسابية.

- تطور رئيسي آخر كان اختراع شبكات الخصومة التوليدية Generative Adversarial Networks(GAN) (Goodfellow et al. 2014). تقليديا، ركزت الطرق الإحصائية لتقدير الكثافة والنماذج التوليدية على إيجاد توزيعات احتمالية مناسبة وخوارزميات (تقريبية غالباً) لأخذ العينات منها. نتيجة لذلك، كانت هذه الخوارزميات محدودة إلى حد كبير بسبب الافتقار إلى المرونة الكامنة في النماذج الإحصائية. كان الابتكار الحاسم في شبكات الخصومة التوليدية هو استبدال جهاز أخذ العينات بخوارزمية تعسفية بمعملات قابلة للتفاضل. ثم يتم تعديلها بطريقة تجعل المميز discriminator (اختبار من عيتيين فعلياً) لا يستطيع التمييز بين البيانات المزيفة والحقيقة. من خلال القدرة على استخدام الخوارزميات التعسفية لتوليد البيانات، فقد فتح تقدير الكثافة لمجموعة متنوعة من التقنيات. أمثلة على ركض الحمر الوحشية Zebras galloping (Zhu et al. 2017)، ووجوه المشاهير المزيفة (Karras et al. 2017) دليل على هذا التقدم. يمكن حتى لرسامي رسومات الشعار المبتكرة الهواة إنتاج صور واقعية استناداً إلى

الرسومات التخطيطية فقط التي تصف كيف يبدو تخطيط المشهد (Park et al., 2019).

- في كثير من الحالات، لا تكفي وحدة معالجة الرسومات الواحدة لمعالجة الكميات الكبيرة من البيانات الممتدة للتدريب. على مدى العقد الماضي، تحسنت القدرة على بناء خوارزميات تدريب متوازية وموزعة بشكل كبير. أحد التحديات الرئيسية في تصميم خوارزميات قابلة للتطوير هو أن العمود الفقري لتحسين التعلم العميق، التدرج الاستوائي العشوائي stochastic gradient descent، يعتمد على دفعات صغيرة minibatches نسبياً من البيانات المراد معالجتها. في الوقت نفسه، تحد الدفعات الصغيرة من كفاءة وحدات معالجة الرسومات. وبالتالي، فإن التدريب على 1024 وحدة معالجة رسومات بحجم صغير يبلغ 32 صورة على سبيل المثال لكل دفعه يصل إلى مجموعة مصغرة مجمعة من حوالي 32000 صورة. العمل الأخير، أولاًً بواسطة Li (2017)، وبعد ذلك بواسطة You et al (2017) و Jia et al (2018) دفع الحجم إلى 64000 ملاحظة، مما قلل من وقت التدريب لنموذج ResNet-50 على مجموعة بيانات ImageNet إلى أقل من 7 دقائق. للمقارنة – تم قياس أوقات التدريب في البداية بترتيب الأيام.

- ساهمت القدرة على موازاة الحساب أيضاً في التقدم في التعلم المعزز ، وقد أدى ذلك إلى تقدم كبير في أجهزة الكمبيوتر التي تحقق أداءً خارقاً في مهام مثل Go ، وألعاب Atari ، و Starcraft ، وفي محاكاة الفيزياء (على سبيل المثال ، باستخدام MuJoCo)، حيث توجد محاكيات البيئة متوفرة. انظر ، على سبيل المثال Silver et al, 2016) للحصول على وصف لكيفية تحقيق ذلك في AlphaGo. باختصار ، يعمل التعلم المعزز بشكل أفضل إذا توفر الكثير من مجموعات (الحالة state، الإجراء action ، المكافأة reward). المحاكاة توفر مثل هذا الطريق.

- لعبت أطر التعلم العميق دوراً مهماً في نشر الأفكار. يتكون الجيل الأول من الأطر مفتوحة المصدر لنماذج الشبكة العصبية من Caffe و Torch و Theano . تمت كتابة العديد من الأوراق الأساسية باستخدام هذه الأدوات. حتى الآن ، حل محلها TensorFlow (غالباً ما تستخدم عبر API Keras عالي المستوى) و CNTK و 2 و Caffe و Apache MXNet . يتكون الجيل الثالث من الأدوات مما يسمى بالأدوات الإلزامية imperative tools للتعلم العميق ، وهو اتجاه أشعله تشير على الأرجح، والذي استخدم صيغة مشابهة لـ Python NumPy لوصف النماذج. تم تبني هذه الفكرة من قبل كل من Gluon API و PyTorch و Jax و MXNet.

أدى تقسيم العمل بين الباحثين في النظام الذين يبنون أدوات أفضل والمصممين الإحصائيين لبناء شبكات عصبية أفضل إلى تبسيط الأمور إلى حد كبير. على سبيل المثال ، كان تدريب نموذج الانحدار اللوجستي الخطى linear logistic regression يمثل مشكلة واجبات منزلية غير بدائية ، تستحق أن تمنحك طلاب دكتوراه التعلم الآلي الجدد في جامعة كارنيجي ميلون في 2014. حتى الآن ، يمكن إنجاز هذه المهمة بأقل من 10 أسطر من التعليمات البرمجية ، مما يضعها بقوة في متناول المبرمجين.

1.6 قصص نجاح Success Stories

للذكاء الاصطناعي تاريخ طويل في تقديم النتائج التي يصعب تحقيقها بخلاف ذلك. على سبيل المثال ، تم نشر أنظمة فرز البريد باستخدام التعرف الضوئي على الأحرف منذ التسعينيات. هذا ، بعد كل شيء ، مصدر مجموعة بيانات MNIST الشهيرة للأرقام المكتوبة بخط اليد. الأمر نفسه ينطبق على قراءة الشيكولات للودائع المصرفية وتسجيل الجدارنة الائتمانية لمقدمي الطلبات. يتم فحص المعاملات المالية تلقائياً بحثاً عن الاحتيال. يشكل هذا العمود الفقري للعديد من أنظمة الدفع للتجارة الإلكترونية ، مثل PayPal و Stripe و AliPay و WeChat و Apple و Visa و MasterCard. كانت برامج الكمبيوتر للشطرنج تنافسية منذ عقود. يغذي التعلم الآلي البحث والتوصية والتخصيص والترتيب على الإنترنت. بمعنى آخر ، التعلم الآلي منتشر ، وإن كان غالباً ما يكون مخفياً عن الأنظار.

في الآونة الأخيرة فقط، أصبح الذكاء الاصطناعي في دائرة الضوء ، ويرجع ذلك في الغالب إلى حلول المشكلات التي كانت تعتبر مستعصية في السابق والتي تتعلق مباشرة بالمستهلكين. يُعزى العديد من هذه التطورات إلى التعلم العميق.

- يمكن للمساعدين الرقميين الأذكياء ، مثل Siri من Apple و Alexa و Google و مساعد Amazon الإجابة على الأسئلة المنطقية بدرجة معقولة من الدقة. يتضمن ذلك المهام البسيطة، مثل تشغيل مفاتيح الإضاءة ، والمهام الأكثر تعقيداً، مثل ترتيب مواعيد الحلاق وتقديم مربع حوار الدعم عبر الهاتف. من المحتمل أن تكون هذه هي العلامة الأكثر وضوحاً على أن الذكاء الاصطناعي يؤثر على حياتنا.

- أحد المكونات الرئيسية في المساعدين الرقميين هو القدرة على التعرف على الكلام بدقة. تدريجياً، زادت دقة هذه الأنظمة إلى حد تحقيق التكافؤ البشري لبعض التطبيقات (Xiong et al. 2018).

- كما قطع التعرف على الأشياء Object recognition شوطاً طويلاً. كان تقدير الكائن في صورة مهمة صعبة إلى حد ما في عام 2010. حقق باحثو معيار

Urbana-Champaign من NEC Labs ImageNet و جامعة إلينوي في Urbana-Champaign ، بمعدل خطأ أعلى 5٪ (Lin et al. 2010). بحلول عام 2017 ، انخفض معدل الخطأ هذا إلى 2.25٪ (Hu et al. 2018). وبالمثل ، تم تحقيق نتائج مذهلة في التعرف على الطيور وتشخيص سرطان الجلد.

- تُستخدم البراعة Prowess في الألعاب لتوفير عصا قياس للذكاء البشري. بدءًا من TD-Gammon ، أدى برنامج لعب الطاولة باستخدام التعلم المعزز للفرق الزمني والتقدم الحسابي والخوارزمي إلى خوارزميات لمجموعة واسعة من التطبيقات. على عكس لعبة الطاولة backgammon ، فإن لعبة الشطرنج بها مساحة أكثر تعقيدًا ومجموعة من الإجراءات. تغلب DeepBlue على Garry Kasparov باستخدام التوازي الهائل massive parallelism والأجهزة ذات الأغراض الخاصة والبحث الفعال من خلال شجرة اللعبة (Campbell et al. 2002). لا يزال Go أكثر صعوبة ، بسبب مساحة الحالة الضخمة. وصل إلى التكافؤ البشري في عام 2015 ، باستخدام التعلم العميق جنبًا إلى جنب معأخذ عينات شجرة مونت كارلو (Silver et al. 2016). كان التحدي في لعبة البوكر هو أن مساحة الحالة كبيرة ولا يتم ملاحظتها إلا جزئياً (لا نعرف بطاقات الخصم). تجاوز Libratus الأداء البشري في البوكر باستخدام استراتيجيات منظمة بكفاءة (Brown and Sandholm 2017).

- مؤشر آخر على التقدم في الذكاء الاصطناعي هو ظهور السيارات والشاحنات ذاتية القيادة self-driving cars and trucks. في حين أن الاستقلالية الكاملة ليست في متناول اليد تماماً ، فقد تم إحراز تقدم ممتاز في هذا الاتجاه ، مع منتجات شحن مثل Tesla و NVIDIA و Waymo التي تتيح الاستقلال الذاتي الجزئي على الأقل. ما يجعل الاستقلالية الكاملة صعبة للغاية هو أن القيادة المناسبة تتطلب القدرة على الإدراك والتفكير ودمج القواعد في النظام. في الوقت الحاضر ، يتم استخدام التعلم العميق بشكل أساسي في جانب رؤية الكمبيوتر لهذه المشاكل. يتم ضبطباقي بشكل كبير من قبل المهندسين.

هذا بالكاد يخدش السطح للتطبيقات المؤثرة للتعلم الآلي. على سبيل المثال ، تدين الروبوتات واللوجستيات وعلم الأحياء الحاسبي وفيزياء الجسيمات وعلم الفلك ببعض التطورات الحديثة الأكثر إثارة للإعجاب على الأقل في أجزاء من التعلم الآلي. وهكذا أصبح التعلم الآلي أداة منتشرة في كل مكان للمهندسين والعلماء.

في كثير من الأحيان ، أثيرت أسئلة حول نهاية العالم القادمة للذكاء الاصطناعي ومعقولية التفرد في مقالات غير تقنية عن الذكاء الاصطناعي. الخوف هو أن أنظمة التعلم الآلي ستتصبح

بطريقة ما واعية وتتخذ القرارات ، بشكل مستقل عن مبرمجيها الذين يؤثرون بشكل مباشر على حياة البشر. إلى حد ما ، يؤثر الذكاء الاصطناعي بالفعل على سبل عيش البشر بطرق مباشرة: يتم تقييم الجدارنة الائتمانية تلقائياً ، ويتنقل الطيارون الآليون في الغالب في المركبات ، واتخاذ قرارات بشأن منح بيانات إحصائية لاستخدام الكفالة كمدخلات. بشكل تافه ، يمكننا أن نطلب من Alexa تشغيل آلة القهوة.

لحسن الحظ ، نحن بعيدون عن نظام ذكاء اصطناعي واعي sentient AI system يمكنه اللتلاعب عمداً بمنشئيه البشريين. أولاًً ، يتم تصميم أنظمة الذكاء الاصطناعي وتدربيها ونشرها بطريقة محددة ووجهة نحو الهدف. في حين أن سلوكهم قد يعطي وهم الذكاء العام ، إلا أنه مزيج من القواعد والاستدلال والنمذج الإحصائية التي تكمّن وراء التصميم. ثانياً ، في الوقت الحالي ، لا توجد أدوات للذكاء العام الاصطناعي قادرة على تحسين نفسها ، والتفكير حول نفسها ، والقادرة على تعديل بنيتها الخاصة وتوسيعها وتحسينها أثناء محاولة حل المهام العامة.

من أكثر الأمور إلحاحاً كيفية استخدام الذكاء الاصطناعي في حياتنا اليومية. من المحتمل أن تتم أتمتها العديد من المهام الوضيعة التي ينجزها سائقي الشاحنات ومساعدو المتاجر. من المحتمل أن تقلل روبوتات المزرعة من تكلفة الزراعة العضوية ولكنها ستعمل أيضاً على أتمتها عمليات الحصاد. قد يكون لهذه المرحلة من الثورة الصناعية عواقب وخيمة على قطاعات واسعة من المجتمع، حيث أن سائقي الشاحنات ومساعدي المتاجر هم من أكثر الوظائف شيوعاً في العديد من البلدان. علاوة على ذلك ، يمكن أن تؤدي النماذج الإحصائية، عند تطبيقها دون عناء، إلى تحيز عرقي أو جنساني أو عمري وتشير مخاوف معقولة بشأن الإنصاف الإجرائي إذا كانت مؤتمته لقيادة القرارات اللاحقة. من المهم التأكد من استخدام هذه الخوارزميات بعناية، مع ما نعرفه اليوم ، فإن هذا يشير قلقنا أكثر إلحاحاً من قدرة الذكاء الخارق الخبيث على تدمير البشرية.

1.7 جوهر التعلم العميق The Essence of Deep Learning

حتى الآن ، تحدثنا عن التعلم الآلي على نطاق واسع. التعلم العميق هو مجموعة فرعية من التعلم الآلي تهتم بالنمذج القائمة على الشبكات العصبية متعددة الطبقات. إنه عميق deep بمعنى أن نماذجها تعلم طبقات layers كثيرة من التحولات. في حين أن هذا قد يبدو ضيقاً ، فقد أدى التعلم العميق إلى ظهور مجموعة مذهلة من النماذج والتقنيات وصياغات المشكلات والتطبيقات. تم تطوير العديد من الحدس لشرح فوائد العمق. يمكن القول أن التعلم الآلي يحتوي على العديد من طبقات الحساب، تكون الأولى من خطوات معالجة الميزات. ما يميز التعلم العميق هو أن العمليات التي تم تعلمها في كل طبقة من طبقات التمثيلات العديدة يتم تعلمها بشكل مشترك من البيانات.

المشاكل التي ناقشناها حتى الآن، مثل التعلم من إشارة الصوت الخام ، أو قيم البكسل الأولية للصور، أو التعيين بين الجمل ذات الأطوال التعسفية ونظيراتها في اللغات الأجنبية ، هي تلك التي يتفوق فيها التعلم العميق وتعثر الأساليب التقليدية. اتضح أن هذه النماذج متعددة الطبقات قادرة على معالجة البيانات الإدراكية منخفضة المستوى low-level perceptual data بطريقة لم تستطع الأدوات السابقة القيام بها. يمكن القول إن أهم عامل مشترك في أساليب التعلم العميق هو التدريب الشامل end-to-end training. بمعنى ، بدلاً من تجميع نظام يعتمد على المكونات التي يتم ضبطها بشكل فردي، يقوم المرء بناء النظام ثم ضبط أدائه بشكل مشترك. على سبيل المثال، استخدم العلماء في رؤية الكمبيوتر لفصل عملية هندسة الميزات feature engineering عن عملية بناء نماذج التعلم الآلي. ساد كاشف الحواف Canny (Canny 1987) ومستخرج ميزة SIFT من Lowe (Lowe, 2004) لأكثر من عقد من الزمن كخوارزميات لتعيين الصور في متجهات الميزات feature vectors. في الأيام الخوالي ، كان الجزء الأساسي من تطبيق التعلم الآلي على هذه المشكلات يتألف من التوصل إلى طرق مصممة يدوياً لتحويل البيانات إلى شكل قابل للنماذج الضحلة shallow models. لسوء الحظ ، لا يوجد سوى القليل جداً الذي يمكن للبشر تحقيقه بالبراعة مقارنةً بالتقنيات المتقدمة لملايين الخيارات التي يتم تنفيذها تلقائياً بواسطة خوارزمية. عندما تولى التعلم العميق زمام الأمور ، تم استبدال مستخلصات الميزات feature extractors هذه بفلاتر تم ضبطها تلقائياً automatically tuned filters ، مما أدى إلى الحصول على دقة فائقة.

وبالتالي ، فإن إحدى الميزات الرئيسية للتعلم العميق هي أنه لا يستبدل النماذج الضحلة في نهاية خطوط التعلم التقليدية فحسب ، بل يستبدل أيضاً عملية هندسة الميزات كثيفة العمالة labor-intensive. علاوة على ذلك ، من خلال استبدال الكثير من المعالجة المسبقة الخاصة بال المجال ، أزال التعلم العميق العديد من الحدود التي كانت تفصل سابقاً بين الرؤية الحاسوبية ، والتعرف على الكلام ، ومعالجة اللغة الطبيعية ، والمعلوماتية الطبية ، ومجالات التطبيق الأخرى ، مما يوفر مجموعة موحدة من الأدوات لمعالجة مشاكل متنوعة.

إلى جانب التدريب الشامل end-to-end training ، نشهد انتقالاً من الأوصاف الإحصائية البارامترية parametric إلى النماذج اللابارامترية nonparametric بالكامل. عندما تكون البيانات شحيحة ، يحتاج المرء إلى الاعتماد على افتراضات مبسطة حول الواقع من أجل الحصول على نماذج مفيدة. عندما تكون البيانات وفيرة ، يمكن استبدالها بنماذج غير بارامترية تناسب البيانات بشكل أفضل. إلى حد ما ، يعكس هذا التقدم الذي شهدته الفيزياء في منتصف القرن الماضي مع توافر أجهزة الكمبيوتر. بدلاً من حل التقديرات البارامترية لكيفية تصرف الإلكترونات يدوياً ، يمكن للمرء الآن اللجوء إلى المحاكاة العددية للمعادلات التفاضلية الجزئية المرتبطة بها. وقد أدى ذلك إلى نماذج أكثر دقة ، وإن كان ذلك في كثير من الأحيان على حساب قابلية الشرح.

هناك اختلاف آخر عن العمل السابق وهو قبول الحلول دون المثالية suboptimal ، والتعامل مع مشاكل التحسين غير الخطية غير المحدبة nonconvex nonlinear optimization ، والاستعداد لتجربة الأشياء قبل إثباتها. أدت هذه التجريبية empiricism المكتشفة حديثاً في التعامل مع المشكلات الإحصائية ، جنباً إلى جنب مع التدفق السريع للمواهب إلى تقدم سريع في الخوارزميات العملية ، وإن كان ذلك في كثير من الحالات على حساب تعديل وإعادة اختراع الأدوات التي كانت موجودة منذ عقود.

في النهاية، يفخر مجتمع التعلم العميق بنفسه لمشاركة الأدوات عبر الحدود الأكاديمية والشركات، وإطلاق العديد من المكتبات الممتازة، والنماذج الإحصائية ، والشبكات المدرية كمصدر مفتوح. وبهذه الروح ، فإن النوتبوك notebooks التي تشكل هذا الكتاب متاحة مجاناً للتوزيع والاستخدام. لقد عملنا بجد لخفض حواجز الوصول للجميع للتعرف على التعلم العميق ونأمل أن يستفيد قرائنا من ذلك.

1.8 الملخص

يدرس التعلم الآلي كيف يمكن لأنظمة الكمبيوتر الاستفادة من الخبرة (غالباً البيانات) لتحسين الأداء في مهام محددة. فهو يجمع بين الأفكار من الإحصائيات ، واستخراج البيانات ، والتحسين. في كثير من الأحيان ، يتم استخدامه كوسيلة لتنفيذ حلول الذكاء الاصطناعي. بصفته فئة من فئات التعلم الآلي ، يركز التعلم التمثيلي representational learning على كيفية العثور تلقائياً على الطريقة المناسبة لتمثيل البيانات. نظراً لأن التعلم التمثيلي متعدد المستويات من خلال تعلم العديد من طبقات التحولات، فإن التعلم العميق لا يحل محل النماذج الضحلة في نهاية خطوط أنابيب التعلم الآلي التقليدية فحسب، بل يستبدل أيضاً عملية هندسة الميزات كثيفة العمالة. تم إطلاق الكثير من التقدم الأخير في التعلم العميق من خلال وفرة البيانات الناشئة عن أجهزة الاستشعار الرخيصة والتطبيقات على نطاق الانترنت، والتقدم الكبير في الحساب ، في الغالب من خلال وحدات معالجة الرسومات. إلى جانب ذلك ، فإن توافر أطر عمل التعلم العميق الفعالة قد جعل تصميم وتنفيذ تحسين النظام بالكامل أسهل بشكل كبير ، وهو عنصر أساسي في الحصول على أداء عالي.

1.9 التمارين

- أي أجزاء من التعليمات البرمجية التي تكتبها حالياً يمكن "تعلمها" ، أي تحسينها بالتعلم وتحديد خيارات التصميم تلقائياً التي يتم إجراؤها في التعليمات البرمجية الخاصة بك؟ هل تتضمن التعليمات البرمجية الخاصة بك خيارات تصميم إرشادية؟ ما هي البيانات التي قد تحتاجها لتعلم السلوك المطلوب؟

2. ما هي المشاكل التي تواجهها والتي لديها العديد من الأمثلة على كيفية حلها ، ولكن لا توجد طريقة محددة لأتمتها؟ قد يكون هؤلاء مرشحين رئيسيين لاستخدام التعلم العميق.
3. وصف العلاقات بين الخوارزميات والبيانات والحساب. كيف تؤثر خصائص البيانات والموارد الحاسوبية الحالية المتاحة على ملاءمة الخوارزميات المختلفة؟
4. قم بتسمية بعض الإعدادات حيث لا يكون التدريب الشامل end-to-end هو الأسلوب الافتراضي حالياً ولكنه قد يكون كذلك.

الابيات

2

2. الأساسيات Preliminaries

لتحضير للغوص (للتعلم العميق ، ستحتاج إلى بعض مهارات البقاء: (1) تقنيات تخزين البيانات ومعالجتها ؛ (2) مكتبات لاستيعاب ومعالجة البيانات من مجموعة متنوعة من المصادر؛ (3) معرفة العمليات الجبرية الخطية الأساسية التي نطبقها على عناصر البيانات عالية الأبعاد ؛ (4) ما يكفي من حساب التفاضل والتكمال لتحديد الاتجاه الذي يضبط كل معلمة لتقليل دالة الخطأ ؛ (5) القدرة على حساب المشتقات تلقائياً بحيث يمكنك الكثير من حسابات التفاضل والتكمال التي تعلمتها للتو ؛ (6) بعض الطلاقة الأساسية في الاحتمالية ، لغتنا الأساسية للاستدلال في ظل عدم اليقين ؛ و (7) بعض القدرة على العثور على إجابات في الوثائق الرسمية عندما تتعرّض.

باختصار، يوفر هذا الفصل مقدمة سريعة للأساسيات التي ستحتاجها لمتابعة معظم المحتوى الفني في هذا الكتاب.

2.1. معالجة البيانات Data Manipulation

من أجل إنجاز أي شيء، تحتاج إلى طريقة ما لتخزين البيانات ومعالجتها. بشكل عام، هناك شيئاً مهماً نحتاج إلى القيام بهما بالبيانات: (1) الحصول عليها؛ و (2) معالجتها بمجرد دخولها إلى الكمبيوتر. لا جدوى من الحصول على البيانات دون طريقة ما لتخزينها، لذا فلنبدأ، دعنا ننسخ أيدينا باستخدام المصفوفات ذات الأبعاد $n - dimensional arrays$ ، والتي نسميها أيضاً الموترات tensors. إذا كنت تعرف بالفعل حزمة الحوسبة العلمية NumPy، فسيكون هذا أمراً سهلاً. بالنسبة لجميع أطر التعلم العميق الحديثة، فإن فئة الموتر (ndarray) في NumPy و TensorFlow و PyTorch و MXNet تشبه ndarray من NumPy، مع إضافة بعض الميزات القاتلة. أولاً، تدعم فئة الموتر التمايز التلقائي. ثانياً، يستفيد من وحدات معالجة الرسومات GPU لتسريع الحساب العددي، بينما يعمل NumPy فقط على وحدات المعالجة المركزية (CPU). يجعل هذه الخصائص الشبكات العصبية سهلة البرمجة وسريعة التشغيل.

2.1.1. البداية

للبدء، نستورد Tensorflow. للإيجاز، غالباً ما يقوم الممارسون بتعيين الاسم المستعار `tf`.

```
import tensorflow as tf
```

يمثل الموتر مجموعة (ربما متعددة الأبعاد) من القيم العددية. مع محور واحد، يسمى موتر متوجه vector. مع محوريين، يسمى الموتر مصفوفة matrix. باستخدام المحاور $2 < k$ ، نسقط الأسماء المتخصصة ونشير فقط إلى الكائن باعتباره موتر ترتيب (k^{th} – order tensor).

يوفر TensorFlow مجموعة متنوعة من الدوال لإنشاء موترات جديدة مسبقاً بالقيم. على سبيل المثال، من خلال استدعاء النطاق (n)، يمكننا إنشاء متوجه لقيم متباينة بشكل متساوٍ بدءاً من 0 (مضمن) وينتهي عند n (غير مضمن). بشكل افتراضي، يكون حجم الفاصل الزمني هو 1. ما لم يتم تحديد خلاف ذلك، يتم تخزين الموترات الجديدة في الذاكرة الرئيسية وتخصيصها للحسابات المعتمدة على وحدة المعالجة المركزية CPU.

```
x = tf.range(12, dtype=tf.float32)
```

```
x
```

```
<tf.Tensor: shape=(12,), dtype=float32, numpy=
array([ 0.,  1.,  2.,  3.,  4.,  5.,  6.,  7.,  8.,  9.,
       10., 11.],
      dtype=float32)>
```

كل من هذه القيم تسمى عنصر من موتر element of the tensor. يحتوي الموتر x على 12 عنصراً. يمكننا فحص العدد الإجمالي للعناصر في موتر عبر دالة الحجم.

```
tf.size(x)
```

يمكننا الوصول إلى شكل الموتر tensor's shape (الطول على طول كل محور) من خلال فحص سمة الشكل الخاصة به. نظراً لأننا نتعامل مع متوجه هنا، فإن الشكل shape يحتوي على عنصر واحد فقط وهو مطابق للحجم.

```
x.shape
```

```
TensorShape([12])
```

يمكننا تغيير شكل الموتر دون تغيير حجمه أو قيمه، عن طريق استدعاء إعادة الشكل reshape على سبيل المثال، يمكننا تحويل المتوجه x الذي شكله (12) إلى مصفوفة X بالشكل (3, 4). يحفظ هذا الموتر الجديد بجميع العناصر ولكنه يعيد تكوينها في مصفوفة. لاحظ أن عناصر المتوجه الخاصة بنا يتم وضعها في صف واحد في كل مرة وبالتالي $X[0] == X[3] == \dots == X[11]$.

```
X = tf.reshape(x, (3, 4))
```

```
X
```

```
<tf.Tensor: shape=(3, 4), dtype=float32, numpy=
array([[ 0.,  1.,  2.,  3.],
       [ 4.,  5.,  6.,  7.],
       [ 8.,  9., 10., 11.]], dtype=float32)>
```

لاحظ أن تحديد كل مكون من مكونات الشكل لإعادة شكله reshape أمر زائد. نظراً لأننا نعرف بالفعل حجم الموتر الخاص بنا، فيمكننا إيجاد مكون واحد من الشكل بالنظر إلىباقي. على سبيل المثال، بالنظر إلى موتر الحجم n والشكل المستهدف (w, h)، فإننا نعرف $w = n/h$. لاستنتاج مكون واحد للشكل تلقائياً، يمكننا وضع 1 - لمكون الشكل الذي يجب استنتاجه

تلقاءً. في حالتنا، بدلاً من استدعاء `x.reshape(3, 4)`، كان بإمكاننا استدعاء `x.reshape(3, -1)` أو `x.reshape(-1, 4)`.

غالباً ما يحتاج الممارسون إلى العمل مع الموترات المهيأة لاحتواء جميع الأصفار أو الآحاد. يمكننا بناء موتر مع ضبط جميع العناصر على الصفر وشكل `(2, 3, 4)` عبر دالة `zeros`.

```
tf.zeros((2, 3, 4))
<tf.Tensor: shape=(2, 3, 4), dtype=float32, numpy=
array([[[0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.]],
      [[0., 0., 0., 0.],
       [0., 0., 0., 0.],
       [0., 0., 0., 0.]]], dtype=float32)>
```

وبالمثل، يمكننا إنشاء موتر مع كل الآحاد باستدعاء `ones`.

```
tf.ones((2, 3, 4))
<tf.Tensor: shape=(2, 3, 4), dtype=float32, numpy=
array([[[1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.]],
      [[1., 1., 1., 1.],
       [1., 1., 1., 1.],
       [1., 1., 1., 1.]]], dtype=float32)>
```

غالباً ما نرغب فيأخذ عينة من كل عنصر بشكل عشوائي (وبشكل مستقل) من توزيع احتمالي معين. على سبيل المثال، غالباً ما تتم تهيئة معلمات الشبكات العصبية بشكل عشوائي. يُشغى المقتطف التالي موتراً بعناصر مستمدة من توزيع غاوسي قياسي (عادي) بمتوسط 0 وانحراف معياري 1.

```
tf.random.normal(shape=[3, 4])
<tf.Tensor: shape=(3, 4), dtype=float32, numpy=
array([[ 1.5391479 , -1.7407725 ,  0.44389075,
       0.8466314 ],
      [-0.486678 , -1.3573753 ,  0.09524103, -1.758265
     ],
      [ 0.7030494 ,  1.1621033 ,  0.98620087,
       1.6612175 ]], dtype=float32)>
```

أخيراً، يمكننا إنشاء الموترات من خلال توفير القيم الدقيقة لكل عنصر من خلال توفير (ربما تكون متداخلة) قائمة (قوائم) بايثون تحتوي على حرفية رقمية. هنا، نقوم ببناء مصفوفة بقائمة من القوائم، حيث تتوافق القائمة الخارجية مع المحور 0، والقائمة الداخلية مع المحور 1.

```
tf.constant([[2, 1, 4, 3], [1, 2, 3, 4], [4, 3, 2, 1]])
<tf.Tensor: shape=(3, 4), dtype=int32, numpy=
array([[2, 1, 4, 3],
       [1, 2, 3, 4],
       [4, 3, 2, 1]], dtype=int32)>
```

2.1.2 indexing and Slicing

كما هو الحال مع قوائم بايثون، يمكننا الوصول إلى عناصر الموتر عن طريق الفهرسة indexing (بدءاً من 0). للوصول إلى عنصر بناءً على موضعه بالنسبة إلى نهاية القائمة، يمكننا استخدام الفهرسة السلبية negative indexing. أخيراً، يمكننا الوصول إلى نطاقات كاملة من المؤشرات عبر التقطيع slicing (على سبيل المثال، $X[start:stop]$) ، حيث تتضمن القيمة المُعاددة الفهرس الأول (start) وليس الأخير (stop). أخيراً، عندما يتم تحديد فهرس واحد index فقط (أو شريحة slice) لموتّر ترتيب (k^{th} order tensor)، يتم تطبيقه على طول المحور 0. وهكذا، في الكود التالي، [-1] يحدد الصف الأخير و [1:3] يحدد الثاني والثالث صفوّف.

```
X[-1], X[1:3]
(<tf.Tensor: shape=(4,), dtype=float32, numpy=array([
 8., 9., 10., 11.], dtype=float32)>,
 <tf.Tensor: shape=(2, 4), dtype=float32, numpy=
 array([[ 4.,  5.,  6.,  7.],
        [ 8.,  9., 10., 11.]], dtype=float32)>)
```

الموترات TensorFlow **Tensors** غير قابلة للتغيير immutable ولا يمكن التخصيص لها. المتغيرات TensorFlow **Variables** هي عبارة عن حاويات قابلة للتغيير mutable للحالة تدعم التعينات assignments. ضع في اعتبارك أن التدرجات gradients في TensorFlow لا تتدفق للخلف من خلال المهام المتغيرة.

بالإضافة إلى تعين قيمة للمتغير بأكمله، يمكننا كتابة عناصر المتغير **Variable** عن طريق تحديد المؤشرات.

```
X_var = tf.Variable(X)
X_var[1, 2].assign(9)
X_var
<tf.Variable 'Variable:0' shape=(3, 4) dtype=float32,
numpy=
array([[ 0.,  1.,  2.,  3.],
```

```
[ 4.,  5.,  9.,  7.],
[ 8.,  9., 10., 11.], dtype=float32)>
```

إذا أردنا تعين عناصر متعددة بنفس القيمة، فإننا نطبق الفهرسة على الجانب الأيسر من عملية الإسناد. على سبيل المثال، يصل [:,2:] إلى الصنفين الأول والثاني، حيث: يأخذ جميع العناصر على طول المحور 1 (العمود). بينما ناقشنا فهرسة المصفوفات، فإن هذا يعمل أيضاً مع المتجهات والموترات التي تزيد عن بعدين.

```
X_var = tf.Variable(X)
X_var[:2, :].assign(tf.ones(X_var[:2,:].shape,
dtype=tf.float32) * 12)
X_var
```

```
<tf.Variable 'Variable:0' shape=(3, 4) dtype=float32,
numpy=
array([[12., 12., 12., 12.],
       [12., 12., 12., 12.],
       [ 8.,  9., 10., 11.]], dtype=float32)>
```

Operations 2.1.3

الآن بعد أن عرفنا كيفية بناء الموترات وكيفية القراءة من عناصرها والكتابة إليها، يمكننا البدء في معالجتها بعمليات رياضية مختلفة. من بين الأدوات الأكثرفائدة هي عمليات العناصر elementwise operations. هذه تطبق عملية scalar قياسية على كل عنصر من عناصر الموتر. بالنسبة للدوال التي تأخذ موترتين كمدخلات، تطبق العمليات الأولية بعض العوامل الثنائية القياسية على كل زوج من العناصر المقابلة. يمكننا إنشاء دالة عنصرية elementwise function من أي دالة تقوم بتعيينها من عددي إلى عددي.

في التدوين الرياضي mathematical notation، نشير إلى هذه العوامل العددية الأحادية (مع إدخال مدخل واحد) من خلال التوقع $\mathbb{R} \rightarrow \mathbb{R}$: f . هذا يعني فقط أن الدالة ترسم من أي رقم حقيقي إلى عدد حقيقي آخر. يمكن تطبيق معظم العوامل المعيارية بطريقة أساسية بما في ذلك المشغلين الأحاديين مثل e^x .

```
tf.exp(x)
<tf.Tensor: shape=(12,), dtype=float32, numpy=
array([1.000000e+00, 2.7182817e+00, 7.3890562e+00,
2.0085537e+01,
      5.4598148e+01, 1.4841316e+02, 4.0342877e+02,
1.0966332e+03,
      2.9809580e+03, 8.1030840e+03, 2.2026465e+04,
5.9874141e+04],
      dtype=float32)>
```

وبالمثل، فإننا نشير إلى العوامل العددية الثانية، والتي تقوم بتعيين أزواج من الأرقام الحقيقة إلى رقم حقيقي (واحد) عبر الترقيق $\mathbb{R} \rightarrow \mathbb{R}$. بالنظر إلى أي متجهين \mathbf{u} و \mathbf{v} من نفس الشكل، وعامل ثانئي f ، يمكننا إنتاج متجه $\mathbf{c} = F(\mathbf{u}, \mathbf{v})$ عن طريق تعين عناصر المتجهات $c_i \leftarrow f(u_i, v_i)$ لكل i ، وأين u_i, v_i thعن عناصر المتجهات \mathbf{u} ، \mathbf{c} . هنا، أنتجنا قيمة المتجه $F: \mathbb{R}^d, \mathbb{R}^d \rightarrow \mathbb{R}^d$ برفع الدالة العددية إلى عملية متجه عنصري. تم رفع جميع العوامل الحسابية القياسية الشائعة للجمع (+)، والطرح (-)، والضرب (*)، والقسمة (/)، والأنس (**)) إلى العمليات الأولية لموترات متطابقة الشكل ذات شكل عشوائي.

```
x = tf.constant([1.0, 2, 4, 8])
y = tf.constant([2.0, 2, 2, 2])
x + y, x - y, x * y, x / y, x ** y
(<tf.Tensor: shape=(4,), dtype=float32, numpy=array([
3., 4., 6., 10.], dtype=float32)>,
 <tf.Tensor: shape=(4,), dtype=float32, numpy=array([-1.,
0., 2., 6.], dtype=float32)>,
 <tf.Tensor: shape=(4,), dtype=float32, numpy=array([
2., 4., 8., 16.], dtype=float32)>,
 <tf.Tensor: shape=(4,), dtype=float32,
numpy=array([0.5, 1. , 2. , 4. ], dtype=float32)>,
 <tf.Tensor: shape=(4,), dtype=float32, numpy=array([
1., 4., 16., 64.], dtype=float32)>)
```

بالإضافة إلى الحسابات الأولية، يمكننا أيضًا إجراء عمليات الجبر الخطى linear algebra مثل حاصل الضرب النقطى dot products وضرب المصفوفات matrix multiplications. سنشرح هذه الأمور بالتفصيل قريباً في القسم 2.3.

يمكننا أيضًا ربط العديد من الموترات معًا، وتكليسها من طرف إلى طرف لتشكيل موتر أكبر. نحتاج فقط إلى تقديم قائمة بالمoterات وإخبار النظام بالمحور المراد ربطه. يوضح المثال أدناه ما يحدث عندما نجمع مصفوفتين على طول الصفوف (المحور 0) مقابل الأعمدة (المحور 1). يمكننا أن نرى أن طول المحور 0 للمخرج الأول (6) هو مجموع أطوال المحور 0 لموتر الإدخال (3 + 3)؛ بينما طول المحور 1 للمخرج الثاني (8) هو مجموع أطوال محور 1 لموتر الإدخال (4 + 4).

```
X = tf.reshape(tf.range(12, dtype=tf.float32), (3, 4))
Y = tf.constant([[2.0, 1, 4, 3], [1, 2, 3, 4], [4, 3, 2, 1]])
tf.concat([X, Y], axis=0), tf.concat([X, Y], axis=1)
(<tf.Tensor: shape=(6, 4), dtype=float32, numpy=
array([[ 0.,  1.,  2.,  3.],
```

```
[ 4.,  5.,  6.,  7.],
[ 8.,  9., 10., 11.],
[ 2.,  1.,  4.,  3.],
[ 1.,  2.,  3.,  4.],
[ 4.,  3.,  2.,  1.]], dtype=float32)>,
<tf.Tensor: shape=(3, 8), dtype=float32, numpy=
array([[ 0.,  1.,  2.,  3.,  2.,  1.,  4.,  3.],
       [ 4.,  5.,  6.,  7.,  1.,  2.,  3.,  4.],
       [ 8.,  9., 10., 11.,  4.,  3.,  2.,  1.]],  
dtype=float32)>
```

في بعض الأحيان، نريد بناء موتور ثالثي عبر البيانات المنطقية.خذ $Y == X$ كمثال. لكل موضع j ، إذا كانت $X[i, j]$ و $Y[j]$ متساوية ، فإن الإدخال المقابل في النتيجة يأخذ القيمة 1 ، وإلا فإنه يأخذ القيمة 0.

```
X == Y
<tf.Tensor: shape=(3, 4), dtype=bool, numpy=
array([[False, True, False, True],
       [False, False, False, False],
       [False, False, False, False]])>
```

يؤدي جمع كل العناصر في الموتر إلى إنتاج موتر بعنصر واحد فقط.

```
tf.reduce_sum(X)
<tf.Tensor: shape=(), dtype=float32, numpy=66.0>
```

Broadcasting 2.1.4

الآن، أنت تعرف كيفية إجراء عمليات ثنائية العناصر على موترين من نفس الشكل. في ظل ظروف معينة، حتى عندما تختلف الأشكال، لا يزال بإمكاننا إجراء عمليات ثنائية العناصر عن طريق استدعاء آلية البث Broadcasting. يعمل البث وفقاً للإجراء التالي المكون من خطوتين: (1) توسيع أحد المصفوفتين أو كليهما عن طريق نسخ العناصر على طول المحاور بطول 1 بحيث يكون للموترين نفس الشكل بعد هذا التحويل؛ (2) إجراء عملية عنصرية elementwise على المصفوفات الناتجة. operation

```
a = tf.reshape(tf.range(3), (3, 1))
b = tf.reshape(tf.range(2), (1, 2))
a, b
(<tf.Tensor: shape=(3, 1), dtype=int32, numpy=
array([[0],
       [1],
       [2]]], dtype=int32)>,
```

```
<tf.Tensor: shape=(1, 2), dtype=int32, numpy=array([[0, 1]], dtype=int32)>
```

بما أن a و b هما 1×3 و 2×1 مصفوفتان، على التوالي، فإن أحشاكا لهما لا تتطابق. ينتج عن البث مصفوفة أكبر من خلال تكرار المصفوفة a على طول الأعمدة والمصفوفة b على طول الصفوف قبل إضافتها إلى العناصر.

$a + b$

```
<tf.Tensor: shape=(3, 2), dtype=int32, numpy=
array([[0, 1],
       [1, 2],
       [2, 3]], dtype=int32)>
```

2.1.5 حفظ الذاكرة

يمكن أن تؤدي العمليات الجارية إلى تخصيص ذاكرة جديدة لنتائج المضيف. على سبيل المثال، إذا كتبنا $Y = X + Y$ ، فإننا نلغي الإشارة إلى الموتر الذي استخدمه Y للإشارة إليه وبدلًا من ذلك نشير إلى Y في الذاكرة المخصصة حديثًا. يمكننا توضيح هذه المشكلة باستخدام دالة `(id)` في بايثون ، والتي تعطينا العنوان الدقيق للكائن المشار إليه في الذاكرة. لاحظ أنه بعد تشغيل $Y = Y + X, id(Y) = Y$ ، يشير المعرف (Y) إلى موقع مختلف. هذا لأن بايثون أول من يقيم $X + Y$ ، ويخصص ذاكرة جديدة للنتيجة ثم يشير Y إلى هذا الموقع الجديد في الذاكرة.

```
before = id(Y)
Y = Y + X
id(Y) == before
```

False

قد يكون هذا غير مرغوب فيه لسببين. أولاً، لا نريد الالتفاف حول تخصيص الذاكرة دون داعٍ طوال الوقت. في التعلم الآلي، غالباً ما يكون لدينا مئات الميجابايت من المعلمات ونقوم بتحديثها جمِيعاً عدة مرات في الثانية. كلما كان ذلك ممكناً، نريد إجراء هذه التحديثات في مكانها الصحيح. ثانياً، قد نشير إلى نفس المعلمات من متغيرات متعددة. إذا لم نقم بالتحديث في مكانه الصحيح، فيجب أن تكون حريصين على تحديث كل هذه المراجع، لثلا نتسبب في تسرب للذاكرة أو نشير عن غير قصد إلى معلمات قديمة.

المتغيرات `Variables` عبارة عن حاويات قابلة للتغيير للحالة في TensorFlow. أنها توفر طريقة لتخزين معلمات النموذج الخاص بك. يمكننا إسناد نتيجة العملية إلى متغير `Variable` مع تعين `assign`. لتوضيح هذا المفهوم، قمنا بالكتابة فوق قيم `Variable Z` بعد تهيئته، باستخدام `zeros_like`، ليكون لها نفس شكل Y .

```
Z = tf.Variable(tf.zeros_like(Y))
print('id(Z):', id(Z))
```

```
Z.assign(X + Y)
print('id(Z):', id(Z))
id(Z): 140011833215008
id(Z): 140011833215008
```

حتى بعد تخزين الحالة باستمرار في متغير `Variable`، قد ترغب في تقليل استخدام الذاكرة بشكل أكبر عن طريق تجنب التخصيصات الزائدة لموترات ليست معلمات نموذجك. نظرًا لأن موترات TensorFlow غير قابلة للتغيير `immutable` ولا تتدفق التدرجات `gradients` من خلال التعينات الممتدة، لا يوفر TensorFlow طريقة واضحة لتشغيل عملية فردية في المكان.

ومع ذلك، يوفر TensorFlow مصمم `tf.function` لاتفاق الحساب داخل الرسم البياني TensorFlow الذي يتم تجميعه وتحسينه قبل التشغيل. يتيح ذلك لـ TensorFlow تقليل `prune` القيم غير المستخدمة وإعادة استخدام التخصيصات السابقة التي لم تعد مطلوبة. هذا يقلل من حمل الذاكرة لحسابات TensorFlow.

```
@tf.function
def computation(X, Y):
    Z = tf.zeros_like(Y) # This unused value will be
    pruned out
    A = X + Y # Allocations will be reused when no
    Longer needed
    B = A + Y
    C = B + Y
    return C + Y

computation(X, Y)
<tf.Tensor: shape=(3, 4), dtype=float32, numpy=
array([[ 8.,  9., 26., 27.],
       [24., 33., 42., 51.],
       [56., 57., 58., 59.]], dtype=float32)>
```

2.1.6. التحويل إلى كائنات بايثون الأخرى

يعد التحويل إلى موثر (ndarray) NumPy أو العكس أمرًا سهلاً. النتيجة المحولة لا تشتراك في الذاكرة. هذا الإزعاج البسيط مهم جدًا في الواقع: عند إجراء عمليات على وحدة المعالجة المركزية CPU أو على وحدات معالجة الرسومات GPUs، فأنت لا تريد إيقاف الحساب، في انتظار معرفة ما إذا كانت حزمة NumPy من بايثون قد ترغب في القيام بشيء آخر بنفس جزء الذاكرة.

```
A = X.numpy()
B = tf.constant(A)
type(A), type(B)
```

```
(numpy.ndarray,
tensorflow.python.framework.ops.EagerTensor)
```

لتحويل موتر بحجم 1 إلى Python scalar، يمكننا استدعاء دالة العنصر `item` أو دوال بايثون المضمنة.

```
a = tf.constant([3.5]).numpy()
a, a.item(), float(a), int(a)
(array([3.5], dtype=float32), 3.5, 3.5, 3)
```

2.1.7 الملخص

- كلاس الموتر `tensor class` هي الواجهة الرئيسية لتخزين البيانات ومعالجتها في مكتبات التعلم العميق.
- توفر `Tensors` مجموعة متنوعة من الدوال بما في ذلك إجراءات البناء `construction routines`؛ الفهرسة `indexing` والتقطيع `slicing` العمليات `memory`-`broadcasting`؛ التخصيص الفعال للذاكرة-`assignment`؛ والتحويل من وإلى كائنات بايثون الأخرى.

2.1.8 التمارين

- قم بتشغيل الكود في هذا القسم. قم بتغيير العبارة الشرطية `Y == X < Y` إلى `Y > X` أو `Y > X` ، ثم انظر إلى نوع الموتر الذي يمكنك الحصول عليه.
- استبدل الموترين اللتين تعملان بواسطة عنصر في آلية الـ `broadcasting` ، على سبيل المثال ، موترات ثلاثة الأبعاد. هل النتيجة هي نفسها كما هو متوقع؟

2.2 معالجة البيانات

حتى الآن، كنا نعمل مع البيانات التركيبية التي وصلت في موترات جاهزة. ومع ذلك، لتطبيق التعلم العميق في البرية، يجب علينا استخراج البيانات الفوضوية `messy data` المخزنة بتنسيقات عشوائية، ومعالجتها مسبقاً لتناسب احتياجاتنا. لحسن الحظ، يمكن لمكتبة `pandas` القيام بالكثير من الرفع الثقيل. هذا القسم، على الرغم من عدم وجود بدليل عن برنامج تعليمي مناسب لـ `pandas` ، سيمنحك دورة مكثفة حول بعض الإجراءات الروتينية الأكثر شيوعاً.

2.2.1 قراءة مجموعة البيانات

ملفات القيم المفصولة بفواصل (CSV) موجودة في كل مكان لتخزين البيانات الجدولية (مثل جداول البيانات). هنا، يتوافق كل سطر مع سجل واحد ويكون من عدة حقول (مفصولة بفواصل comma-separated)، على سبيل المثال، "أليرت أينشتاين، 14 مارس 1879، أولم ، مدرسة الفنون التطبيقية الفيدرالية ، الإنجازات في مجال فيزياء الجاذبية". لتوسيع كيفية تحميل ملفات

مع CSV ./data/house_tiny.csv أدناه pandas نقوم بإنشاء ملف .. يمثل هذا الملف مجموعة بيانات للمنازل، حيث يتواافق كل صف مع منزل مميز ومتواافق الأعمدة مع عدد الغرف (NumRooms) ونوع السقف (RoofType) والسعر (Price).

```
import os
```

```
os.makedirs(os.path.join('..', 'data'), exist_ok=True)
data_file = os.path.join('..', 'data', 'house_tiny.csv')
with open(data_file, 'w') as f:
    f.write('''NumRooms,RoofType,Price
NA,NA,127500
2,NA,106000
4,Slate,178100
NA,NA,140000''')
```

دعنا الآن نستورد pandas ونحمل مجموعة البيانات مع .read_csv

```
import pandas as pd
```

```
data = pd.read_csv(data_file)
print(data)
```

	NumRooms	RoofType	Price
0	NaN	NaN	127500
1	2.0	NaN	106000
2	4.0	Slate	178100
3	NaN	NaN	140000

2.2.2. تحضير البيانات

في التعلم الخاضع للإشراف، نقوم بتدريب النماذج للتنبؤ بقيمة مستهدفة معينة، مع الأخذ في الاعتبار مجموعة معينة من قيم الإدخال. خطوتنا الأولى في معالجة مجموعة البيانات هي فصل الأعمدة المقابلة للإدخال input مقابل القيم المستهدفة target values. يمكننا تحديد الأعمدة integer-location based إما بالاسم أو عن طريق الفهرسة القائمة على الموقع الصحيح (iloc indexing).

ربما لاحظت أن pandas استبدلت جميع إدخالات CSV بقيمة NA خاصة (ليس رقمًا not a number). يمكن أن يحدث هذا أيضًا عندما يكون الإدخال فارغاً empty على سبيل المثال، "3,,270000". تسمى هذه القيم المفقودة missing values وهي "bugs" في علم البيانات، وهو تهديد دائم ستواجهه طوال حياتك المهنية. اعتماداً على السياق، يمكن معالجة القيم المفقودة إما عن طريق التضمين imputation أو الحذف deletion.

يُستبدل التضمين Imputation القيم المفقودة بتقديرات لقيمها بينما يتجاهل الحذف deletion ببساطة إما تلك الصنوف أو تلك الأعمدة التي تحتوي على قيم مفقودة.

فيما يلي بعض أساليب التضمين الشائعة. بالنسبة لحقول الإدخال الفئوية categorical input fields، يمكننا التعامل مع NaN كفئة. نظرًا لأن العمود RoofType يأخذ القيم Slate و NaN، يمكن للـ pandas تحويل هذا العمود إلى عمودين RoofType_Slate و RoofType_nan. سيحدد الصنف الذي يكون نوع زقاقه هو Slate قيمة RoofType_nan على 1 و 0 على التوالي. يحمل العكس لصنف RoofType به قيمة RoofType مفقودة.

```
inputs, targets = data.iloc[:, 0:2], data.iloc[:, 2]
inputs = pd.get_dummies(inputs, dummy_na=True)
print(inputs)
```

	NumRooms	RoofType_Slate	RoofType_nan
0	NaN	0	1
1	2.0	0	1
2	4.0	1	0
3	NaN	0	1

بالنسبة للقيم العددية المفقودة missing numerical values، يمثل أحد الأساليب الإرشادية heuristic الشائعة في استبدال إدخالات NaN بالقيمة المتوسطة للعمود المقابل.

```
inputs = inputs.fillna(inputs.mean())
print(inputs)
```

	NumRooms	RoofType_Slate	RoofType_nan
0	3.0	0	1
1	2.0	0	1
2	4.0	1	0
3	3.0	0	1

2.2.3 التحويل إلى تنسيق Tensor

الآن بعد أن أصبحت جميع الإدخالات في المدخلات والأهداف رقمية، يمكننا تحميلها في موتور (تذكرة) (2.1).

```
import tensorflow as tf
```

```
X, y = tf.constant(inputs.values),
tf.constant(targets.values)
X, y
(<tf.Tensor: shape=(4, 3), dtype=float64, numpy=
array([[3., 0., 1.],
```

```
[2., 0., 1.],  
[4., 1., 0.],  
[3., 0., 1.]]))>,  
<tf.Tensor: shape=(4,), dtype=int64,  
numpy=array([127500, 106000, 178100, 140000])>)
```

2.2.4 المناقشة

أنت تعرف الآن كيفية تقسيم أعمدة البيانات، وإسناد المتغيرات المفقودة، وتحميل بيانات Pandas إلى موترات. في القسم 5.7، سوف تكتسب المزيد من مهارات معالجة البيانات. في حين أن هذه الدورة التدريبية المكثفة تبقي الأمور بسيطة، يمكن أن تصبح معالجة البيانات صعبة. على سبيل المثال، بدلاً من الوصول إلى ملف CSV واحد، قد تنتشر مجموعة البيانات الخاصة بنا عبر ملفات متعددة مستخرجة من قاعدة بيانات علائقية. على سبيل المثال، في تطبيق التجارة الإلكترونية e-commerce، قد تكون عناوين العملاء موجودة في جدول وشراء البيانات في جدول آخر.علاوة على ذلك، يواجه الممارسون أنواعاً لا تعد ولا تحصى من البيانات تتجاوز الفئوية وال الرقمية. تتضمن أنواع البيانات الأخرى سلاسل نصية وصور وبيانات صوتية وسُحب النقاط point clouds. في كثير من الأحيان، تكون الأدوات المتقدمة والخوارزميات الفعالة مطلوبة لمنع معالجة البيانات من أن تصبح أكبر عنق زجاجي في خط أنابيب التعلم الآلي. ستظهر هذه المشاكل عندما نصل إلى الرؤية الحاسوبية ومعالجة اللغة الطبيعية. أخيراً، يجب أن نتبه إلى جودة البيانات quality data. غالباً ما تعاني مجموعات البيانات الواقعية من القيم المتطرفة outliers، والقياسات الخاطئة من أجهزة الاستشعار sensors، وأخطاء التسجيل، والتي يجب معالجتها قبل تغذية البيانات في أي نموذج. يمكن أن تساعدك أدوات تصوّر البيانات مثل seaborn أو matplotlib أو Bokeh على فحص البيانات يدوياً وتطوير حدس حول المشكلات التي قد تحتاج إلى معالجتها.

2.2.5 التمارين

- حاول تحميل مجموعات البيانات datasets، على سبيل المثال، Abalone من مستودع [UCI Machine Learning Repository](#) وفحص خصائصها. أي جزء منهن يحتوي على قيم مفقودة missing values؟ ما هو جزء المتغيرات العددية numerical أو الفئوية categorical أو النصية text؟
- جرب فهرسة و اختيار أعمدة البيانات بالاسم بدلاً من رقم العمود. تحتوي وثائق Pandas الخاصة بالفهرسة على مزيد من التفاصيل حول كيفية القيام بذلك.
- ما حجم مجموعة البيانات التي تعتقد أنه يمكنك تحميلها بهذه الطريقة؟ ما قد تكون القيود؟ تلميح: ضع في اعتبارك الوقت اللازم لقراءة البيانات والتّمثيل والمعالجة

وبصمة الذاكرة. جرب هذا على الكمبيوتر المحمول الخاص بك. ما الذي يتغير إذا جربته على الخادم؟

4. كيف ستتعامل مع البيانات التي تحتوي على عدد كبير جدًا من الفئات categories؟

ماذا لو كانت تسميات الفئات كلها فريدة unique؟ هل يجب عليك تضمين الأخير؟

5. ما هي بدائل الباندا Pandas التي يمكنك التفكير فيها؟ ماذا عن تحميل موترات NumPy من ملف؟ تحقق من Pillow ، مكتبة تصوير Python.

2.3 الجبر الخطى

في الوقت الحالي، يمكننا تحميلمجموعات البيانات إلى موترات ومعالجة هذه المواترات من خلال العمليات الحسابية الأساسية. لبدء بناء نماذج متطرفة، سنحتاج أيضًا إلى بعض الأدوات من الجبر الخطى linear algebra. يقدم هذا القسم مقدمة لطيفة لأهم المفاهيم، بدءًا من الحساب العددي scalar arithmetic والتكتيف ramping up وصولاً إلى ضرب المصفوفة matrix multiplication.

2.3.1 الكميات القياسية Scalars

ت تكون معظم الرياضيات اليومية من معالجة الأرقام واحدًا تلو الآخر. رسميًا، نسمي هذه القيم الكميات القياسية Scalars. على سبيل المثال، درجة الحرارة في بالو ألتون هي 72 درجات فهرنهايت معتدلة. إذا أردت تحويل درجة الحرارة إلى درجات سيليزية، فعليك تقييم التعبير $c = \frac{5}{9}(f - 32)$ ، وضبط f على 72. في هذه المعادلة، القيم 5 ، 9 و 32 هي كميات قياسية. المتغيرات c و f تمثل الكميات القياسية غير المعروفة unknown scalars.

نشير إلى الكميات القياسية scalars بأحرف عادية صغيرة (على سبيل المثال، x و y و z) ومساحة جميع المقاييس ذات القيمة الحقيقة (المستمرة) بواسطة \mathbb{R} . من أجل المنفعة، سوف نتخاطر التعريفات الصارمة للمساحات السابقة spaces. فقط تذكر أن التعبير $x \in \mathbb{R}$ هو طريقة رسمية للقول بأن هذا عدد حقيقي ذو قيمة x . يشير الرمز \in ("in") إلى العضوية في مجموعة. على سبيل المثال، $x, y \in \{0, 1\}$ يشير إلى أن المتغيرات التي يمكن أن تأخذ فقط قيمًا 0 أو 1 .

```
import tensorflow as tf
```

```
x = tf.constant(3.0)
```

```
y = tf.constant(2.0)
```

```
x + y, x * y, x / y, x**y
```

```
(<tf.Tensor: shape=(), dtype=float32, numpy=5.0>,
```

```
<tf.Tensor: shape=(), dtype=float32, numpy=6.0>,
```

```
<tf.Tensor: shape=(), dtype=float32, numpy=1.5>,
```

```
<tf.Tensor: shape=(), dtype=float32, numpy=9.0>
```

2.3.2. المتجهات Vectors

لأغراضنا، يمكنك التفكير في المتجهات Vectors على أنها مصفوفات ذات طول ثابت من الكميات القياسية scalars. كما هو الحال مع نظائرهم في الكود، نسمى هذه القيم عناصر المتجه elements of the vector (المرادفات تشمل الإدخالات والمكونات). عندما تمثل المتجهات أمثلة من مجموعات بيانات من العالم الحقيقي، فإن قيمها تحمل بعض الأهمية في العالم الحقيقي. على سبيل المثال، إذا كنا ندرب نموذجاً للتنبؤ بمخاطر التخلف عن سداد القرض loan defaulting، فقد نربط كل متقدم بمتجه يتطابق مكوناته مع كميات مثل دخله أو طول فترة التوظيف أو عدد حالات التخلف عن السداد السابقة. إذا كنا ندرس مخاطر النوبات القلبية heart attack risk، فقد يمثل كل متوجه مريضاً وقد تتوافق مكوناته مع أحد ثعالباته الحيوية ومستويات الكوليسترول ودقائق من التمارين في اليوم وما إلى ذلك. x و y و z .

يتم تنفيذ المتجهات كموترات ترتيب 1st-order tensors. بشكل عام، يمكن أن يكون لمثل هذه الموترات أطوال عشوائية، تخضع لقيود الذاكرة. تحذير: في بايثون، كما هو الحال في معظم لغات البرمجة، تبدأ مؤشرات المتجهات zero-based indexing عند 0، والمعروفة أيضاً باسم الفهرسة الصفرية vector indices one-based indexing في الجبر الخطي، تبدأ نصوص الجبر الخطي من 1 (الفهرسة على أساس واحد .(indexing

```
x = tf.range(3)
```

```
x
```

```
<tf.Tensor: shape=(3,), dtype=int32, numpy=array([0, 1, 2], dtype=int32)>
```

يمكنا الإشارة إلى عنصر متوجه باستخدام حرف منخفض subscript. على سبيل المثال، x_2 يشير إلى العنصر الثاني من x . نظرًا لأنّه قيمة قياسية scalar، فإننا لا نظهره بخط عريض. بشكل افتراضي، نحن نتخيل المتجهات عن طريق تكديس عناصرها عمودياً.

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix},$$

فيما يلي عناصر المتوجه x_n, \dots, x_1 . في وقت لاحق، سوف نميز بين متجهات الأعمدة column vectors ومتجهات الصفوف row vectors التي يتم تكديس عناصرها أفقياً. تذكر أننا نصل إلى عناصر الموتر عبر الفهرسة indexing.

```
x[2]
```

```
<tf.Tensor: shape=(), dtype=int32, numpy=2>
```

للإشارة إلى أن المتجه يحتوي على عناصر n ، نكتب رسمي، نطلق n على أبعاد المتجه dimensionality of the vector في الكود، هذا يتواافق مع طول الموتر، ويمكن الوصول إليه عبر دالة `len` المدمجة في بايثون.

`len(x)`

3

يمكننا أيضاً الوصول إلى الطول عبر سمة الشكل `shape`. الشكل عبارة عن مجموعة تشير إلى طول الموتر على طول كل محور. الموترات التي لها محور واحد فقط لها أشكال تحتوي على عنصر واحد فقط.

`x.shape`

`TensorShape([3])`

في كثير من الأحيان، يتم تحميل الكلمة "بعد dimension" بشكل زائد لتعني كلاً من عدد المحاور والطول على طول محور معين. لتجنب هذا الالتباس، نستخدم الترتيب `order` للإشارة إلى عدد المحاور والأبعاد dimensionality حسرياً للإشارة إلى عدد المكونات.

2.3.3. المصفوفات

كما أن الكميات القياسية scalars هي موترات الترتيب 0^{th} -order tensors والمتغيرات 1^{st} -order tensors هي vectors و 2^{nd} -order tensors هي matrices . نشير إلى المصفوفات بأحرف كبيرة غامقة (على سبيل المثال، \mathbf{X} ، \mathbf{Y} و \mathbf{Z})، ونمثلها في رمز بواسطة موترات بمحورين. يشير التعبير $\mathbf{A} \in \mathbb{R}^{m \times n}$ إلى أن المصفوفة \mathbf{A} تحتوي على قيم قياسية scalers حقيقة القيمة $m \times n$ ، مرتبة في شكل m صفوف و n أعمدة. عندما $m = n$ نقول إن المصفوفة مربعة square. بصرياً، يمكننا توضيح أي مصفوفة كجدول. للإشارة إلى عنصر فردي، نقوم بتدوين كل من فهارس الصفوف والأعمدة، على سبيل المثال، a_{ij} القيمة التي تنتهي إلى $\mathbf{A}'s i^{\text{th}}$ صف و j^{th} عمود:

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}.$$

في الكود، نحن نمثل مصفوفة $\mathbf{A} \in \mathbb{R}^{m \times n}$ بواسطة موتر الترتيب 2^{nd} -order tensors بالشكل (m, n) . يمكننا تحويل أي موتر بحجم مناسب $m \times n$ إلى مصفوفة $n \times m$ بتمرير الشكل المطلوب لإعادة تشكيله `reshape`:

```
A = tf.reshape(tf.range(6), (3, 2))
```

A

```
<tf.Tensor: shape=(3, 2), dtype=int32, numpy=
```

```
array([[0, 1],
       [2, 3],
       [4, 5]], dtype=int32)>
```

في بعض الأحيان، نريد قلب المحاور. عندما نتبادل صفوف وأعمدة المصفوفة، فإن النتيجة تسمى تبديلاها transpose. بشكل رسمي، نشير إلى تبديل المصفوفة A بواسطة A^T وإذا $A^T = A$ ، ثم $b_{ij} = a_{ji}$ لجميع i و j . وبالتالي، فإن تبديل المصفوفة $m \times n$ عبارة عن مصفوفة $:n \times m$

$$A^T = \begin{bmatrix} a_{11} & a_{21} & \dots & a_{m1} \\ a_{12} & a_{22} & \dots & a_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \dots & a_{mn} \end{bmatrix}.$$

في الكود، يمكننا الوصول إلى تبديل أي مصفوفة matrix's transpose على النحو التالي:

```
tf.transpose(A)
<tf.Tensor: shape=(2, 3), dtype=int32, numpy=
array([[0, 2, 4],
       [1, 3, 5]], dtype=int32)>
```

المصفوفات المتماثلة Symmetric matrices هي مجموعة فرعية من المصفوفات المرسدة square matrices التي تساوي التبادلات الخاصة بها: $A = A^T$. المصفوفة التالية متماثلة:

```
A = tf.constant([[1, 2, 3], [2, 0, 4], [3, 4, 5]])
A == tf.transpose(A)
<tf.Tensor: shape=(3, 3), dtype=bool, numpy=
array([[ True,  True,  True],
       [ True,  True,  True],
       [ True,  True,  True]])>
```

المصفوفات مفيدة في تمثيلمجموعات البيانات. عادةً ما تتوافق المصفوفات مع السجلات الفردية وتتوافق الأعمدة مع سمات مميزة.

2.3.4 Tensors

بينما يمكنك الذهاب بعيداً في رحلة التعلم الآلي الخاصة بك باستخدام الكميات القياسية Scalars والمتجهات vectors والمصفوفات Matrices فقط، فقد تحتاج في النهاية إلى العمل مع الموترات ذات الترتيب الأعلى higher-order tensors. تقدم لنا الموترات طريقة عامة لوصف الامتدادات لمصفوفات الترتيب n^{th} -order arrays . نحن نطلق على كلاسات البرامج من فئة الموتر "موترات tensors " على وجه التحديد لأنها يمكن أن تحتوي أيضاً على أعداد عشوائية من المحاور. في حين أنه قد يكون من المربك استخدام كلمة موتر لكل من الكائن الرياضي وإدراكه في الكود، يجب أن يكون معناها واضحاً من السياق. نشير إلى الموترات العامة بأحرف

كبيرة مع وجه خط خاص (على سبيل المثال, X, Y و Z) وأآلية الفهرسة الخاصة بهم (على سبيل المثال, x_{ijk} و $x_{1,2i-1,3j}$) تبع بشكل طبيعي من المصفوفات.

ستصبح الموترات أكثر أهمية عندما نبدأ العمل مع الصور. تصل كل صورة على هيئة موتر بترتيب 3rd-order tensor مع محاور مقابلة لارتفاع والعرض والقناة channel. في كل موقع مكاني، تراكم شدة كل لون (أحمر وأخضر وأزرق) على طول القناة. علاوة على ذلك، يتم تمثيل مجموعة من الصور في رمز بواسطة موتر ترتيب 4th-order tensor، حيث يتم فهرسة الصور المميزة على طول المحور الأول. يتم إنشاء الموترات ذات الترتيب الأعلى بشكل مشابه للمتجهات والمصفوفات، من خلال زيادة عدد مكونات الشكل.

```
tf.reshape(tf.range(24), (2, 3, 4))
<tf.Tensor: shape=(2, 3, 4), dtype=int32, numpy=
array([[[ 0,  1,  2,  3],
       [ 4,  5,  6,  7],
       [ 8,  9, 10, 11]],
      [[12, 13, 14, 15],
       [16, 17, 18, 19],
       [20, 21, 22, 23]]], dtype=int32)>
```

2.3.5. الخصائص الأساسية لحساب الموتر

الكميات القياسية والمتجهات والمصفوفات والموترات ذات الترتيب الأعلى جماعتها لها بعض الخصائص المفيدة. على سبيل المثال، تنتج عمليات elementwise مخرجات لها نفس شكل معاملاتها.

```
A = tf.reshape(tf.range(6, dtype=tf.float32), (2, 3))
B = A # No cloning of `A` to `B` by allocating new
      memory
A, A + B
(<tf.Tensor: shape=(2, 3), dtype=float32, numpy=
array([[0., 1., 2.],
       [3., 4., 5.]], dtype=float32)>,
 <tf.Tensor: shape=(2, 3), dtype=float32, numpy=
array([[ 0.,  2.,  4.],
       [ 6.,  8., 10.]], dtype=float32)>)
```

يسمى الضرب الأولى elementwise product لمصفوفتين ضرب Hadamard (يشار إليه \odot). أدناه، نوضح إدخالات منتج Hadamard لمصفوفتين $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m \times n}$

$$\mathbf{A} \odot \mathbf{B} = \begin{bmatrix} a_{11}b_{11} & a_{12}b_{12} & \dots & a_{1n}b_{1n} \\ a_{21}b_{21} & a_{22}b_{22} & \dots & a_{2n}b_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1}b_{m1} & a_{m2}b_{m2} & \dots & a_{mn}b_{mn} \end{bmatrix}.$$

A * B

```
<tf.Tensor: shape=(2, 3), dtype=float32, numpy=
array([[ 0.,  1.,  4.],
       [ 9., 16., 25.]], dtype=float32)>
```

إضافة أو مضاعفة قيمة قياسية scalar وموتر tensor ينتج عنه نتيجة بنفس شكل الموتر الأصلي. هنا ، يضاف كل عنصر من عناصر الموتر إلى (أو يضرب في) القيمة القياسية.

```
a = 2
X = tf.reshape(tf.range(24), (2, 3, 4))
a + X, (a * X).shape
```

```
(<tf.Tensor: shape=(2, 3, 4), dtype=int32, numpy=
array([[[ 2,  3,  4,  5],
       [ 6,  7,  8,  9],
       [10, 11, 12, 13]],
      [[14, 15, 16, 17],
       [18, 19, 20, 21],
       [22, 23, 24, 25]]], dtype=int32)>,
TensorShape([2, 3, 4]))
```

Reduction 2.3.6

في كثير من الأحيان، نرغب في حساب مجموع عناصر الموتر. للتعبير عن مجموع العناصر في متوجه x الطول n ، نكتب $\sum_{i=1}^n x_i$. هناك دالة بسيطة لها:

```
x = tf.range(3, dtype=tf.float32)
x, tf.reduce_sum(x)
```

```
(<tf.Tensor: shape=(3,), dtype=float32, numpy=array([0.,
1., 2.], dtype=float32)>,
<tf.Tensor: shape=(), dtype=float32, numpy=3.0>)
```

للتعبير عن المجاميع sums على عناصر الموترات ذات الشكل التعسفي arbitrary shape ، فإننا ببساطة نجمع جميع محاورها. على سبيل المثال، يمكن كتابة مجموع عناصر $n \times m$

$$\cdot \sum_{i=1}^m \sum_{j=1}^n a_{ij} \text{ كـ } \mathbf{A}$$

```
A.shape, tf.reduce_sum(A)
```

```
(TensorShape([2, 3]), <tf.Tensor: shape=(),
dtype=float32, numpy=15.0>)
```

بشكل افتراضي، يؤدي استدعاء دالة الجمع إلى تقليل الموتر على طول جميع محاوره، مما يؤدي في النهاية إلى إنتاج قيمة قياسية scalar. تتيح لنا مكتباتنا أيضًا تحديد المحاور التي يجب أن يتم تقليل الموتر على طولها. لتجميع جميع العناصر على طول الصفوف (المحور 0)، نحدد axis=0 في sum. نظرًا لأن مصفوفة الإدخال تقلل على طول المحور 0 لتوليد متوجه الإخراج، فإن هذا المحور مفقود من شكل المخرجات.

```
A.shape, tf.reduce_sum(A, axis=0).shape  
(TensorShape([2, 3]), TensorShape([3]))
```

سيؤدي تحديد axis=1 إلى تقليل بُعد العمود (المحور 1) عن طريق جمع عناصر جميع الأعمدة.

```
A.shape, tf.reduce_sum(A, axis=1).shape  
(TensorShape([2, 3]), TensorShape([2]))
```

إن اختزال مصفوفة على طول كل من الصفوف والأعمدة عن طريق الجمع يعادل تلخيص كل عناصر المصفوفة.

```
tf.reduce_sum(A, axis=[0, 1]), tf.reduce_sum(A) # Same  
as `tf.reduce_sum(A)`  
(<tf.Tensor: shape=(), dtype=float32, numpy=15.0>,  
<tf.Tensor: shape=(), dtype=float32, numpy=15.0>)
```

الكمية ذات الصلة هي المتوسط mean، وتسمى أيضًا average. نحسب المتوسط بقسمة المجموع على العدد الإجمالي للعناصر. نظرًا لأن حساب المتوسط شائع جدًا، فإنه يحصل على دالة مكتبة مخصصة تعمل بشكل مماثل للجمع sum.

```
tf.reduce_mean(A), tf.reduce_sum(A) / tf.size(A).numpy()  
(<tf.Tensor: shape=(), dtype=float32, numpy=2.5>,  
<tf.Tensor: shape=(), dtype=float32, numpy=2.5>)
```

وبالمثل، يمكن للدالة حساب المتوسط أن تقلل من موتر على طول محاور معينة.

```
tf.reduce_mean(A, axis=0), tf.reduce_sum(A, axis=0) /  
A.shape[0]  
(<tf.Tensor: shape=(3,), dtype=float32,  
numpy=array([1.5, 2.5, 3.5], dtype=float32)>,  
<tf.Tensor: shape=(3,), dtype=float32,  
numpy=array([1.5, 2.5, 3.5], dtype=float32)>)
```

2.3.7 مجموع عدم الاختزال Non-Reduction Sum

قد يكون من المفيد أحيانًا الاحتفاظ بعدد المحاور دون تغيير عند استدعاء الدالة لحساب المجموع أو المتوسط. هذا مهم عندما نريد استخدام آلية البث broadcast mechanism.

```
sum_A = tf.reduce_sum(A, axis=1, keepdims=True)
sum_A, sum_A.shape
```

```
<tf.Tensor: shape=(2, 1), dtype=float32, numpy=
array([[ 3.],
       [12.]], dtype=float32)>,
TensorShape([2, 1]))
```

على سبيل المثال، بما أن `sum_A` تحافظ على محوريها بعد جمع كل صف، يمكننا قسمة `A` على `sum_A` مع البث لإنشاء مصفوفة حيث يتم جمع كل صف إلى 1.

```
A / sum_A
```

```
<tf.Tensor: shape=(2, 3), dtype=float32, numpy=
array([[ 0.          ,  0.33333334,  0.66666667 ],
       [ 0.25        ,  0.33333334,  0.41666666]],
      dtype=float32)>
```

إذا أردنا حساب المجموع التراكمي `cumulative sum` لعناصر `A` على طول بعض المحاور، قلل `axis=0` (صف بصف)، يمكننا استدعاء دالة `cumsum`. حسب التصميم، لا تقلل هذه الدالة من موثر الإدخال على طول أي محور.

```
tf.cumsum(A, axis=0)
```

```
<tf.Tensor: shape=(2, 3), dtype=float32, numpy=
array([[ 0.,  1.,  2.],
       [ 3.,  5.,  7.]], dtype=float32)>
```

2.3.8. الضرب النقطي

حتى الآن، قمنا فقط بإجراء العمليات الأولية، والمجموع، والمتosطات. وإذا كان هذا هو كل ما يمكننا فعله، فلن يستحق الجبر الخطي قسماً خاصاً به. لحسن الحظ، هذا هو المكان الذي تصبح فيه الأشياء أكثر إثارة للاهتمام. يعد حاصل الضرب النقطي Dot Products من أهم العمليات الأساسية. بالنظر إلى متوجهين $\mathbf{x}, \mathbf{y} \in \mathbb{R}^d$ ، فإن حاصل الضرب النقطي $\mathbf{y}^T \mathbf{x}$ (أو $\langle \mathbf{y}, \mathbf{x} \rangle$) هو مجموع حاصل ضرب العناصر في نفس الموضع:

$$\mathbf{x}^T \mathbf{y} = \sum_{i=1}^d x_i y_i$$

```
y = tf.ones(3, dtype=tf.float32)
x, y, tf.tensordot(x, y, axes=1)
```

```
<tf.Tensor: shape=(3,), dtype=float32, numpy=array([ 0.,
       1.,  2.], dtype=float32)>,
<tf.Tensor: shape=(3,), dtype=float32, numpy=array([ 1.,
       1.,  1.], dtype=float32)>,
<tf.Tensor: shape=(), dtype=float32, numpy=3.0>)
```

بالتساوي، يمكننا حساب حاصل الضرب النقطي لمتجهي عن طريق إجراء عملية ضرب عنصري `:sum` elementwise multiplication متبوعة بمجموع

```
tf.reduce_sum(x * y)
```

```
<tf.Tensor: shape=(), dtype=float32, numpy=3.0>
```

يعتبر الضرب النقطي مفديفي مجموعة واسعة من السياقات. على سبيل المثال، بالنظر إلى مجموعة من القيم، يُشار إليها بالمتوجه $\mathbf{w} \in \mathbb{R}^n$ ومجموعة من الأوزان التي يُشار إليها في \mathbf{x} . يمكن التعبير عن مجموع الأوزان للقيمة وفقاً للأوزان \mathbf{w} على أنها حاصل الضرب النقطي $\mathbf{x}^T \mathbf{w}$. عندما تكون الأوزان غير سالية ومجموعها واحد، مثل $(\sum_{i=1}^n w_i = 1)$ ، يعبر الضرب النقطي عن مجموع الأوزان weighted average. بعد تسوية متوجهين بحيث يكون لهما طول الوحدة unit length، تعبير حاصل الضرب النقطي عن جيب التمام للزاوية بينهما. لاحظ في هذا القسم، سنقدم رسمياً فكرة الطول length هذه.

2.3.9 ضرب Matrix-Vector

الآن بعد أن عرفنا كيفية حساب حاصل الضرب النقطي، يمكننا البدء في فهم حاصل الضرب بين $m \times n$ مصفوفة \mathbf{A} ومتوجه الأبعاد \mathbf{x} . للبدء، نتخيل المصفوفة matrix بدلالة متوجهات الصفر : row vectors

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_1^T \\ \mathbf{a}_2^T \\ \vdots \\ \mathbf{a}_m^T \end{bmatrix},$$

حيث كل $\mathbf{a}_i^T \in \mathbb{R}^n$ هو متوجه صف يمثل i^{th} صف المصفوفة \mathbf{A} .

حاصل ضرب المصفوفة المتوجه \mathbf{Ax} هو ببساطة متوجه طول العمود m ، الذي يكون عنصره i^{th} هو حاصل الضرب النقطي $\mathbf{a}_i^T \mathbf{x}$:

$$\mathbf{Ax} = \begin{bmatrix} \mathbf{a}_1^T \\ \mathbf{a}_2^T \\ \vdots \\ \mathbf{a}_m^T \end{bmatrix} \mathbf{x} = \begin{bmatrix} \mathbf{a}_1^T \mathbf{x} \\ \mathbf{a}_2^T \mathbf{x} \\ \vdots \\ \mathbf{a}_m^T \mathbf{x} \end{bmatrix}.$$

يمكنا أن نفكري في الضرب بمصفوفة $\mathbf{A} \in \mathbb{R}^{m \times n}$ كتحويل يُسقط المتوجهات منه n إلى \mathbb{R}^m . هذه التحولات مفيدة بشكل ملحوظ. على سبيل المثال، يمكننا تمثيل عمليات التدوير على أنها عمليات ضرب بواسطة مصفوفات مربعة معينة. تصف حاصل ضرب Matrix-vector أيضًا الحساب الأساسي المتضمن في حساب مخرجات كل طبقة في الشبكة العصبية بالنظر إلى مخرجات الطبقة السابقة.

للتعبير عن حاصل الضرب -المتجه في الكود، نستخدم دالة matvec. لاحظ أن بعد العمود \mathbf{A} (طوله على طول المحور 1) يجب أن يكون هو نفسه بعد \mathbf{x} (طوله).

```
A.shape, x.shape, tf.linalg.matvec(A, x)
(TensorShape([2, 3]),
TensorShape([3]),
<tf.Tensor: shape=(3,), dtype=float32, numpy=array([5., 14.], dtype=float32)>)
```

2.3.10 ضرب المصفوفة - المصفوفة

إذا كانت قد حصلت على تعليق حاصل الضرب النقطي وضرب المصفوفة - المنتج، فيجب أن يكون ضرب المصفوفة - المصفوفة أمرًا سهلاً.

قل إن لدينا مصفوفتين $\mathbf{B} \in \mathbb{R}^{k \times m}$ و $\mathbf{A} \in \mathbb{R}^{n \times k}$

$$\mathbf{A} = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1k} \\ a_{21} & a_{22} & \cdots & a_{2k} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nk} \end{bmatrix}, \mathbf{B} = \begin{bmatrix} b_{11} & b_{12} & \cdots & b_{1m} \\ b_{21} & b_{22} & \cdots & b_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ b_{k1} & b_{k2} & \cdots & b_{km} \end{bmatrix}.$$

افتراض $\mathbf{a}_i^T \in \mathbb{R}^k$ تشير إلى متوجه الصفر i^{th} الذي يمثل صف المصفوفة \mathbf{A} ودعنا $\mathbf{b}_j \in \mathbb{R}^k$ تشير إلى متوجه العمود من j^{th} عمود المصفوفة \mathbf{B} :

$$\mathbf{A} = \begin{bmatrix} \mathbf{a}_1^T \\ \mathbf{a}_2^T \\ \vdots \\ \mathbf{a}_n^T \end{bmatrix}, \mathbf{B} = [\mathbf{b}_1 \quad \mathbf{b}_2 \quad \cdots \quad \mathbf{b}_m].$$

لإيجاد حاصل ضرب المصفوفة $\mathbf{C} \in \mathbb{R}^{n \times m}$ ، نقوم ببساطة بحساب كل عنصر c_{ij} على أنه حاصل الضرب النقطي بين i^{th} صف لـ \mathbf{A} وصف j^{th} لـ \mathbf{B} ، أي $\mathbf{a}_i^T \mathbf{b}_j$.

$$\mathbf{C} = \mathbf{AB} = \begin{bmatrix} \mathbf{a}_1^T \\ \mathbf{a}_2^T \\ \vdots \\ \mathbf{a}_n^T \end{bmatrix} [\mathbf{b}_1 \quad \mathbf{b}_2 \quad \cdots \quad \mathbf{b}_m] = \begin{bmatrix} \mathbf{a}_1^T \mathbf{b}_1 & \mathbf{a}_1^T \mathbf{b}_2 & \cdots & \mathbf{a}_1^T \mathbf{b}_m \\ \mathbf{a}_2^T \mathbf{b}_1 & \mathbf{a}_2^T \mathbf{b}_2 & \cdots & \mathbf{a}_2^T \mathbf{b}_m \\ \vdots & \vdots & \ddots & \vdots \\ \mathbf{a}_n^T \mathbf{b}_1 & \mathbf{a}_n^T \mathbf{b}_2 & \cdots & \mathbf{a}_n^T \mathbf{b}_m \end{bmatrix}.$$

يمكنا التفكير في ضرب المصفوفة - المصفوفة \mathbf{AB} على أنه حاصل ضرب مصفوفة متوجهية أو ضرب نقطي $m \times n$ وربط النتائج معًا لتشكيل مصفوفة $m \times n$. في المقتطف التالي، نقوم بضرب المصفوفة على \mathbf{A} و \mathbf{B} . هنا، \mathbf{A} عبارة عن مصفوفة تتكون من صفين و 3 أعمدة، و \mathbf{B} عبارة عن مصفوفة مكونة من 3 صفوف و 4 أعمدة. بعد الضرب، نحصل على مصفوفة من صفين و 4 أعمدة.

```
B = tf.ones((3, 4), tf.float32)
tf.matmul(A, B)
<tf.Tensor: shape=(3, 4), dtype=float32, numpy=
```

```
array([[ 3.,  3.,  3.],
       [12., 12., 12.]], dtype=float32)>
```

غالباً ما يتم تبسيط مصطلح ضرب المصفوفة-matrix المصفوفة إلى ضرب المصفوفة multiplication، ويجب عدم الخلط بينه وبين ضرب Hadamard.

2.3.11 Norms. المعيار

بعض العوامل الأكثر فائدة في الجبر الخطي هي المعيار Norm. بشكل غير رسمي، يخبرنا المعيار المتوجه عن حجمه. على سبيل المثال، يقيس المعيار ℓ_2 الطول (الإقليدي) للمتجه. هنا، نحن نستخدم مفهوم الحجم size الذي يتعلق بحجم مكونات المتوجه (وليس أبعاده).

المعيار هو دالة $\|\cdot\|$ تقوم بتعيين متوجه vector إلى رقمي scalar وتفيد بالخصائص الثلاث التالية:

- بالنظر إلى أي متوجه \mathbf{x} ، إذا قمنا بقياس (جميع عناصر) المتوجه بواسطة قيمة قياسية norms triangle inequality: فسيتم قياس معياره وفقاً لذلك:

$$\|\alpha \mathbf{x}\| = |\alpha| \|\mathbf{x}\|.$$

- لأي متوجهات \mathbf{x} و \mathbf{y} : $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$.

- معيار المتوجه norm of a vector غير سالب ويختفي فقط إذا كان المتوجه صفرًا: $\|\mathbf{x}\| > 0$ for all $\mathbf{x} \neq 0$.

العديد من الدوال هي معايير صالحة ومعايير مختلفة تقوم بتمييز مفاهيم مختلفة للحجم. المعيار الإقليدية Euclidean norm التي تعلمناها جميعاً في هندسة المدرسة الابتدائية عند حساب وتر المثلث الأيمن هي الجذر التربيعي لمجموع مربعات عناصر المتوجه. رسمياً، هذا يسمى المعيار ℓ_2 ويتم التعبير عنه كـ:

$$\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^n x_i^2}.$$

طريقة المعيار تحسب المعيار ℓ_2 .

```
u = tf.constant([3.0, -4.0])
tf.norm(u)
```

```
<tf.Tensor: shape=(), dtype=float32, numpy=5.0>
```

معيار ℓ_1 شائع أيضاً ويسمى المقياس المرتبط بمسافة مانهاتن Manhattan distance. بحكم التعريف، يجمع المعيار ℓ_1 القيم المطلقة لعناصر المتوجه:

$$\|\mathbf{x}\|_1 = \sum_{i=1}^n |x_i|.$$

بالمقارنة مع المعيار ℓ_2 ، فهي أقل حساسية للقيم المتطرفة outliers. لحساب المعيار ℓ_1 ، نقوم بتكوين القيمة المطلقة باستخدام عملية الجمع.

`tf.reduce_sum(tf.abs(u))`

`<tf.Tensor: shape=(), dtype=float32, numpy=7.0>`

كل من المعيار ℓ_2 والمعيار ℓ_1 هي حالات خاصة للمعایير الأكثر عمومية ℓ_p :

$$\|\mathbf{x}\|_p = \left(\sum_{i=1}^n |x_i|^p \right)^{\frac{1}{p}}.$$

في حالة المصفوفات، تكون الأمور أكثر تعقيداً. بعد كل شيء، يمكن النظر إلى المصفوفات كمجموعات من الإدخالات الفردية وكتائنات تعمل على المتجهات وتحولها إلى متجهات أخرى. على سبيل المثال، يمكننا أن نسأل إلى أي مدى يمكن أن يكون حاصل ضرب المصفوفة-mathttge $\mathbf{X}\mathbf{v}$ مرتبطة بـ \mathbf{v} . هذا الخط النظري يؤدي إلى معيار يسمى المعيار الطيفي spectral norm. في الوقت الحالي، نقدم معيار Frobenius، والذي يسهل حسابه ويتم تعريفه على أنه الجذر التربيعي لمجموع مربعات عناصر المصفوفة:

$$\|\mathbf{X}\|_F = \sqrt{\sum_{i=1}^m \sum_{j=1}^n x_{ij}^2}.$$

يتصرف معيار Frobenius كما لو كان معياراً ℓ_2 لمتجه على شكل مصفوفة. سيؤدي استدعاء الدالة التالية إلى حساب قاعدة Frobenius للمصفوفة.

`tf.norm(tf.ones((4, 9)))`

`<tf.Tensor: shape=(), dtype=float32, numpy=6.0>`

بينما لا نريد أن نتقدم كثيراً على أنفسنا، يمكننا زرع بعض الحدس بالفعل حول سبب فائدة هذه المفاهيم. في التعلم العميق، نحاول غالباً حل مشكلات التحسين optimization problems: تعظيم maximize الاحتمال المخصص للبيانات المرصودة observed data؛ تعظيم الإيرادات المرتبطة بنموذج التوصية؛ تقليل المسافة بين التنبؤات وأرصاد القيم الحقيقية minimize ground-truth؛ تقليل المسافة بين تمثيلات صور الشخص نفسه مع تعظيم المسافة بين تمثيلات الصور لأشخاص مختلفين. غالباً ما يتم التعبير عن هذه المسافات، التي تشكل أهداف خوارزميات التعلم العميق، كمعايير norms.

2.3.12. المناقشة

في هذا القسم، راجعنا جميع الجبر الخطي الذي ستحتاجه لفهم جزء كبير من التعلم العميق الحديث. هناك الكثير من الجبر الخطي والكثير منه مفید للتعلم الآلي. على سبيل المثال، يمكن أن تتحلل المصفوفات إلى عوامل factors، ويمكن أن تكشف هذه التحليلات عن بنية منخفضة الأبعاد فيمجموعات البيانات الواقعية. هناك حقول فرعية كاملة للتعلم الآلي تركز على استخدام تحليلات المصفوفة وعملياتها على الموترات عالية الترتيب لاكتشاف البنية فيمجموعات البيانات وحل مشاكل التنبؤ. لكن هذا الكتاب يركز على التعلم العميق. ونعتقد أنك ستكون أكثر ميلاً لتعلم المزيد من الرياضيات بمجرد أن تتسخ يديك عند تطبيق التعلم الآلي علىمجموعات بيانات حقيقية. لذلك بينما نحتفظ بالحق في تقديم المزيد من الرياضيات لاحقاً، فإننا نختتم هذا القسم هنا.

إذا كنت حريصاً على تعلم المزيد من الجبر الخطي، فهناك العديد من الكتب والموارد الممتازة عبر الإنترنت. للحصول على دورة مكثفة أكثر تقدماً ، ضع في اعتبارك التحقق ([Kolter, 2008](#), [Petersen et al., 2008](#), [Strang, 1993](#))

الخلاصة:

- القيم القياسية scalars والمتجهات vectors والمصفوفات matrices والموترات tensors هي الكائنات الرياضية الأساسية المستخدمة في الجبر الخطي ولها عدد من المحاور صفر واحد واثنان وعدد عشوائي من المحاور، على التوالي.
- يمكن تقطيع أو تقليل الموترات على طول محاور محددة من خلال الفهرسة، أو عمليات مثل المجموع والمتوسط، على التوالي.
- تسمى حاصل الضرب الأولى Elementwise products حاصل ضرب Hadamard. على التقىض من ذلك، فإن الضرب النقطي وحاصل ضرب المصفوفة-المتجه وضرب المصفوفة-المصفوفة ليست عمليات عنصرية elementwise operations وفي العموم تُرجع الكائنات التي لها أشكال مختلفة عن المعاملات.
- مقارنة بحاصل ضرب Hadamard، يستغرق ضرب المصفوفة-المصفوفة وقتاً أطول بكثير للحساب (الوقت المكعب بدلاً من الوقت التربيعي).
- تلتقط المعايير Norms المفاهيم المختلفة لحجم المتجه magnitude of a vector، ويتم تطبيقها بشكل شائع على الفرق بين متجهين لقياس المسافة بينهما.
- تتضمن معايير المتجه الشائعة المعيار ℓ_1 والمعيار ℓ_2 ، وتشمل معايير المصفوفة المشتركة المعايير الطيفية ومعايير Frobenius.

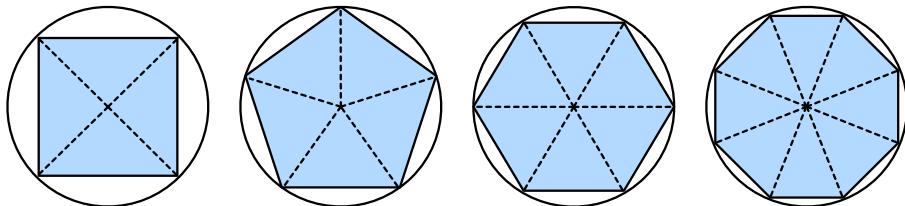
2.3.13 التمارين

1. اثبت أن دور transpose لمدور المصفوفة هو المصفوفة نفسها: $(\mathbf{A}^T)^T = \mathbf{A}$
2. بالنظر إلى مصفوفتين \mathbf{A} و \mathbf{B} ، اثبت ان $\text{sum}(\mathbf{A}^T + \mathbf{B}^T) = \text{sum}(\mathbf{B}^T + \mathbf{A}^T)$
3. بالنظر إلى أي مصفوفة مربعة \mathbf{A} ، هل $\mathbf{A} + \mathbf{A}^T$ دائمًا متماثل symmetric؟ هل يمكنك إثبات النتيجة باستخدام نتيجة التمرينين السابقين فقط؟
4. حددنا موتور \mathbf{X} للشكل (2, 3, 4) في هذا القسم. ما هو خرج $\text{len}(\mathbf{X})$ ؟ اكتب إجابتك دون تنفيذ أي كود، ثم تحقق من إجابتك باستخدام الكود.
5. بالنسبة للمotor \mathbf{X} ذي الشكل التعسفي، هل يتوافق $\text{len}(\mathbf{X})$ دائمًا مع طول محور معين من \mathbf{X} ؟ ما هذا المحور؟
6. قم بتنفيذ $\text{A} / \text{A.sum(axis=1)}$ وشاهد ما سيحدث. هل يمكنك تحليل السبب؟
7. عند السفر بين نقطتين في وسط مانهاتن، ما هي المسافة التي يتبعها عليك قطعها من حيث الإحداثيات، أي من حيث السبل والشوارع؟ هل يمكنك السفر قطرياً؟
8. ضع في اعتبارك موتور ذو شكل (4, 3, 2). ما هي أشكال مخرجات الجمع على طول المحور 0 و 1 و 2؟
9. قم بتغذية موتور ثلاثة محاور أو أكثر لدالة linalg.norm ورافق ناتجها. ماذا تحسب هذه الدالة لموترات الشكل التعسفي؟
10. حدد ثلاث مصفوفات كبيرة، على سبيل المثال $\mathbf{B} \in \mathbb{R}^{2^{10} \times 2^{16}}$ ، $\mathbf{A} \in \mathbb{R}^{2^{10} \times 2^{16}}$ و $\mathbf{C} \in \mathbb{R}^{2^5 \times 2^{14}}$ على سبيل المثال تم تهيئتها باستخدام متغيرات عشوائية غاويسية Gaussian random variables. تزيد حساب المنتج \mathbf{ABC} . هل هناك أي اختلاف في مساحة الذاكرة وسرعتها، اعتماداً على ما إذا كنت تقوم بحساب $(\mathbf{AB})\mathbf{C}$ أم $\mathbf{A}(\mathbf{BC})$. لماذا؟
11. حدد ثلاث مصفوفات كبيرة، على سبيل المثال $\mathbf{B} \in \mathbb{R}^{2^{10} \times 2^{16}}$ ، $\mathbf{A} \in \mathbb{R}^{2^{10} \times 2^{16}}$ و $\mathbf{C} \in \mathbb{R}^{2^5 \times 2^{16}}$ هل هناك أي اختلاف في السرعة حسب ما إذا كنت تقوم بحساب $\mathbf{AB}\mathbf{C}$ أم $\mathbf{AC}^T\mathbf{B}$ ؟ لماذا؟ ما الذي يتغير إذا قمت بتهيئة $\mathbf{C} = \mathbf{B}^T$ بدون استنساخ cloning memory؟ لماذا؟
12. حدد ثلاث مصفوفات، على سبيل المثال $\mathbf{A}, \mathbf{B}, \mathbf{C} \in \mathbb{R}^{100 \times 200}$. تشكيل موتور مع 3 محاور عن طريق تكديس $[\mathbf{A}, \mathbf{B}, \mathbf{C}]$. ما هي الأبعاد؟ اقطع Slice الإحداثي الثاني للمحور الثالث لاسترجاع \mathbf{B} . تحقق من صحة إجابتك.

2.4 التفاضل والتكامل Calculus

لفترة طويلة، ظلت كيفية حساب مساحة الدائرة لغزاً. بعد ذلك، جاء عالم الرياضيات اليوناني القديم أرخميدس بفكرة ذكية لتسجيل سلسلة من المضلعات مع زيادة أعداد الرؤوس داخل دائرة (الشكل 2.4.1). بالنسبة إلى المضلع ذي الرؤوس n ، نحصل على مثلثات n . يقترب ارتفاع كل مثلث من نصف القطر r لأننا نقسم الدائرة بشكل أكثر دقة. في نفس الوقت، تقترب قاعدتها $\frac{2\pi r}{n}$ ، لأن النسبة بين القوس والقاطع تقترب من 1 لعدد كبير من الرؤوس. وبالتالي، فإن مساحة المثلث تقترب

$$\text{مساحة المثلث} \approx r \cdot \frac{1}{2} \left(\frac{2\pi r}{n} \right) = \pi r^2.$$



الشكل 2.4.1 إيجاد مساحة الدائرة كإجراء نهائي.

يؤدي هذا الإجراء المحدد إلى حساب التفاضل والتكميل التفاضلي (القسم 19.5). يمكن أن يخبرنا الأول عن كيفية زيادة قيمة دالة أو إنقاذهما من خلال معالجة وسيطاتها. يكون هذا مفيداً لمشاكل التحسين التي نواجهها في التعلم العميق، حيث نقوم بتحديث معلماتنا بشكل متكرر لتقليل دالة الخطأ. يعالج التحسين كيفية ملاءمة (fit) نماذجنا لبيانات التدريب، وحساب التفاضل والتكميل calculus هو شرطه الأساسي. ومع ذلك، لا ننس أن هدفنا النهائي هو الأداء الجيد على البيانات غير المرئية من قبل unseen data. هذه المشكلة تسمى التعميم generalization وستكون محور التركيز الرئيسي للفصول الأخرى.

2.4.1 المشتقات والتفاضل Derivatives and Differentiation

بساطة، المشتق derivative هو معدل التغيير في دالة فيما يتعلق بالتغييرات في مدخلاتها arguments. يمكن للمشتقات أن تخبرنا عن مدى سرعة زيادة دالة الخطأ أو نقصانها إذا قمنا بزيادة أو تقليل كل معلمة بمقدار ضئيل للغاية. بشكل رسمي، بالنسبة للدوال $f: \mathbb{R} \rightarrow \mathbb{R}$ ، تلك الخريطة من القيم القياسية scalars إلى القيم القياسية scalars، يتم تعريف مشتق f عند نقطة x على أنه

$$f'(x) = \lim_{h \rightarrow 0} \frac{f(x + h) - f(x)}{h}.$$

يسمى هذا المصطلح الموجود على الجانب الأيمن بالغاية limit ويخبرنا بما يحدث لقيمة التعبير عندما يقترب المتغير المحدد من قيمة معينة. يخبرنا هذا الحد بما تقارب فيه النسبة بين الاضطراب h والتغيير في قيمة الدالة $f(x + h) - f(x)$ كلما قلصنا حجمها إلى الصفر.

عندما يكون $(x)' f'$ موجوداً، يُقال إنه قابل للتفاضل differentiable عند x ; وعندما $f'(x)$ يكون موجوداً لجميع x في مجموعة، على سبيل المثال، الفترة $[a, b]$ ، نقول أن f هنا قابل للتلفاضل في هذه المجموعة. ليست كل الدوال قابلة للتلفاضل، بما في ذلك العديد من الدوال التي نرغب في تحسينها، بما في ذلك الدقة والمنطقة الواقع تحت خاصية التشغيل المستقبلة (AUC). ومع ذلك، نظراً لأن حساب مشتق الخسارة يعد خطوة حاسمة في جميع الخوارزميات تقريرياً لتدريب الشبكات العصبية العميق، فإننا غالباً ما نقوم بتحسين بدليل قابل للتلفاضل بدلاً من ذلك.

يمكنا تفسير المشتق $(x)' f'$ على أنه معدل اللحظي instantaneous rate للتغير بالنسبة لـ $f(x) = 3x^2 - 4x$. دعونا نطور بعض الحدس بمثال. حدد x .

```
%matplotlib inline
import numpy as np
from matplotlib_inline import backend_inline
from d2l import tensorflow as d2l

def f(x):
    return 3 * x ** 2 - 4 * x
```

ضبط $\frac{f(x+h)-f(x)}{h}$ ، النهج 2 كنهاج 0. بينما تفتقر هذه التجربة إلى صرامة إثبات رياضي، سترى ذلك قريباً بالفعل $f'(1) = 2$.

```
for h in 10.0**np.arange(-1, -6, -1):
    print(f'h={h:.5f}, numerical limit={(f(1+h)-
f(1))/h:.5f}')
```

```
h=0.10000, numerical limit=2.30000
h=0.01000, numerical limit=2.03000
h=0.00100, numerical limit=2.00300
h=0.00010, numerical limit=2.00030
h=0.00001, numerical limit=2.00003
```

هناك العديد من الاصطلاحات الترميزية المكافئة للمشتقات. معطى $y = f(x)$ ، العبارات التالية متكافئة:

$$f'(x) = y' = \frac{dy}{dx} = \frac{df}{dx} = \frac{d}{dx} f(x) = Df(x) = D_x f(x),$$

حيث الرموز $\frac{d}{dx}$ و D هم عوامل التفاضل differentiation operators. أدناه، نقدم مشتقات بعض الدوال الشائعة:

$$\begin{aligned}\frac{d}{dx} C &= 0 && \text{for any constant } C \\ \frac{d}{dx} x^n &= nx^{n-1} && \text{for } n \neq 0 \\ \frac{d}{dx} e^x &= e^x \\ \frac{d}{dx} \ln x &= x^{-1}\end{aligned}$$

غالباً ما تكون الدوال المكونة من دوال قابلة للتفاضل قابلة للتفاضل. القواعد التالية مفيدة للعمل مع تراكيب أي دوال قابلة للتفاضل f و g و ثابت.

$$\begin{aligned}\frac{d}{dx} [Cf(x)] &= C \frac{d}{dx} f(x) && \text{Constant multiple rule} \\ \frac{d}{dx} [f(x) + g(x)] &= \frac{d}{dx} f(x) + \frac{d}{dx} g(x) && \text{Sum rule} \\ \frac{d}{dx} [f(x)g(x)] &= f(x) \frac{d}{dx} g(x) + g(x) \frac{d}{dx} f(x) && \text{Product rule} \\ \frac{d}{dx} \frac{f(x)}{g(x)} &= \frac{g(x) \frac{d}{dx} f(x) - f(x) \frac{d}{dx} g(x)}{g^2(x)} && \text{Quotient rule}\end{aligned}$$

باستخدام هذا، يمكننا تطبيق القواعد لإيجاد مشتقة $4x^2 - 3x$ عما:

$$\frac{d}{dx} [3x^2 - 4x] = 3 \frac{d}{dx} x^2 - 4 \frac{d}{dx} x = 6x - 4.$$

يُظهر التوصيل $x = 1$ أن المشتق 2 موجود بالفعل في هذا المكان. لاحظ أن المشتقات تخبرنا عن ميل slope الدالة في موقع معين.

2.4.2 أدوات الرسم Visualization Utilities

يمكننا رسم ميل الدوال باستخدام مكتبة matplotlib. نحن بحاجة إلى تحديد بعض الدوال. كما يشير اسمه، فإن matplotlib use_svg_display تخبر بـmatplotlib بـ`use_svg_display` بـ`matplotlib.use('svg')`. التعليق `#@save` هو مُعدلّ الرسومات بتنسيق SVG للحصول على صور أكثر وضوحاً. التعليق `#@d2l` هو مُعدلّ modifier خاص يسمح لنا بحفظ أي دالة أو فئة أو كتلة رمز أخرى في حزمة d2l حتى نتمكن من استدعاؤها لاحقاً دون تكرار الكود ، على سبيل المثال ، عبر `d2l.use_svg_display()`.

```
def use_svg_display(): #@save
    """Use the svg format to display a plot in
Jupyter."""
    backend_inline.set_matplotlib_formats('svg')
 بشكل ملائم، يمكننا تعين أحجام الشكل باستخدام set_figsize . نظراً لأن بيان الاستيراد من matplotlib import pyplot as plt حيث تم تميز plt .d2l.pyplot في حزمة d2l، يمكننا استدعاء via #@save
```

```
def set_figsize(figsize=(3.5, 2.5)): #@save
    """Set the figure size for matplotlib."""
    use_svg_display()
    d2l.pyplot.rcParams['figure.figsize'] = figsize
يمكن أن تربط الدالة set_axes المحاور بالخصائص، بما في ذلك التسميات
.scales والمقاييس ranges والنطاقات.
```

```
#@save
def set_axes(axes, xlabel, ylabel, xlim, ylim, xscale,
yscale, legend):
    """Set the axes for matplotlib."""
    axes.set_xlabel(xlabel), axes.set_ylabel(ylabel)
    axes.set_xscale(xscale), axes.set_yscale(yscale)
    axes.set_xlim(xlim),      axes.set_ylim(ylim)
    if legend:
        axes.legend(legend)
    axes.grid()
```

باستخدام هذه الدوال الثلاث، يمكننا تحديد دالة الرسم plot لترابك منحنيات متعددة. جزء كبير من الكود هنا هو مجرد ضمان تطابق أحجام وأشكال المدخلات.

```
#@save
def plot(X, Y=None, xlabel=None, ylabel=None, legend=[], 
xlim=None,
        ylim=None, xscale='linear', yscale='linear',
        fmts=('-', 'm--', 'g-.', 'r:'), figsize=(3.5,
2.5), axes=None):
    """Plot data points."""
def has_one_axis(X): # True if `X` (tensor or list)
has 1 axis
    return (hasattr(X, "ndim") and X.ndim == 1 or
isinstance(X, list)
            and not hasattr(X[0], "__len__"))
```

```

if has_one_axis(X): X = [X]
if Y is None:
    X, Y = [[]] * len(X), X
elif has_one_axis(Y):
    Y = [Y]
if len(X) != len(Y):
    X = X * len(Y)

set_figsize(figsize)
if axes is None: axes = d2l.plt.gca()
axes.cla()
for x, y, fmt in zip(X, Y, fmts):
    axes.plot(x,y,fmt) if len(x) else
axes.plot(y,fmt)
set_axes(axes, xlabel, ylabel, xlim, ylim, xscale,
yscale, legend)
الآن يمكننا رسم الدالة  $f(x) = 2x - 3$  وخط المماس  $u = 2x - 3$  عند  $x = 1$ , حيث المعامل هو ميل خط المماس

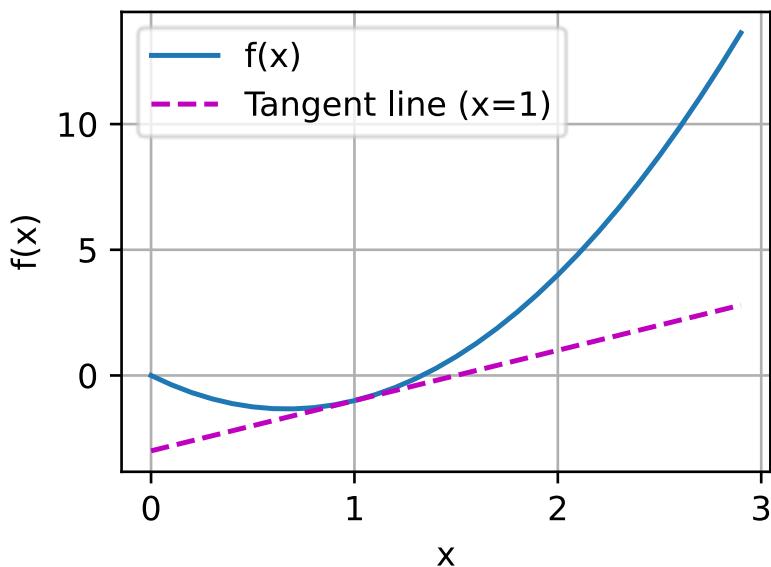
```

.slope of the tangent line

```

x = np.arange(0, 3, 0.1)
plot(x, [f(x), 2 * x - 3], 'x', 'f(x)', legend=['f(x)', 'Tangent line (x=1)'])

```



2.4.3. المشتقات الجزئية والتدرجات Partial Derivatives and Gradients

حتى الآن، كنا نفرق بين دوال متغير واحد فقط. في التعلم العميق، نحتاج أيضًا إلى العمل مع دوال العديد من المتغيرات. نقدم بإيجاز مفاهيم المشتق التي تنطبق على مثل هذه الدوال متعددة المتغيرات .multivariate functions

لتكون $y = f(x_1, x_2, \dots, x_n)$ دالة مع n من المتغيرات. المشتق الجزئي partial derivative y فيما يتعلق بمعامله i^{th} هو

$$\frac{\partial y}{\partial x_i} = \lim_{h \rightarrow 0} \frac{f(x_1, \dots, x_{i-1}, x_i + h, x_{i+1}, \dots, x_n) - f(x_1, \dots, x_i, \dots, x_n)}{h}.$$

لحساب $\frac{\partial y}{\partial x_i}$ ، يمكننا التعامل مع $x_1, x_2, \dots, x_{i-1}, x_{i+1}, \dots, x_n$ على أنها ثوابت ونحسب مشتقة y بالنسبة إلى x . تعد اصطلاحات الترميز التالية للمشتقات الجزئية شائعة وكلها تعني نفس الشيء:

$$\frac{\partial y}{\partial x_i} = \frac{\partial f}{\partial x_i} = \partial_{x_i} f = \partial_i f = f_{x_i} = f_i = D_i f = D_{x_i} f.$$

يمكننا ربط المشتقات الجزئية partial derivatives لدالة متعددة المتغيرات multivariate gradient of the function. لنفترض أن مدخلات الدالة $f: \mathbb{R}^n \rightarrow \mathbb{R}$ عبارة عن متوجه ذي أبعاد $\mathbf{x} = [x_1, x_2, \dots, x_n]^T$ وأن الناتج عبارة عن قيمة قياسية scalar. انحدار الدالة f بالنسبة إلى \mathbf{x} متوجه للمشتقات الجزئية:

$$\nabla_{\mathbf{x}} f(\mathbf{x}) = [\partial_{x_1} f(\mathbf{x}), \partial_{x_2} f(\mathbf{x}), \dots, \partial_{x_n} f(\mathbf{x})]^T.$$

عندما لا يكون هناك غموض $\nabla_{\mathbf{x}} f(\mathbf{x})$, no ambiguity ، عادة ما يتم استبداله بـ $\nabla f(\mathbf{x})$. القواعد التالية مفيدة للتمييز بين الدوال متعددة المتغيرات:

- لكل $\mathbf{A} \in \mathbb{R}^{m \times n}$ ما لدينا $\nabla_{\mathbf{x}} \mathbf{x}^T \mathbf{A} = \mathbf{A}^T \nabla_{\mathbf{x}} \mathbf{A} \mathbf{x} = \mathbf{A}^T$
- بالنسبة للمصفوفات المربعة $\mathbf{A} \in \mathbb{R}^{n \times n}$ ، لدينا ذلك $\nabla_{\mathbf{x}} \mathbf{x}^T \mathbf{A} \mathbf{x} = (\mathbf{A} + \mathbf{A}^T) \mathbf{x}$
- وعلى وجه الخصوص $\nabla_{\mathbf{x}} \|\mathbf{x}\|^2 = \nabla_{\mathbf{x}} \mathbf{x}^T \mathbf{x} = 2\mathbf{x}$

وبالمثل، لدينا $2\mathbf{X} = \nabla_{\mathbf{X}} \|\mathbf{X}\|_F^2$ لأي مصفوفة \mathbf{X} .

2.4.4 قاعدة السلسلة Chain Rule

في التعلم العميق، غالباً ما يصعب حساب التدرجات gradients ذات الاهتمام لأننا نعمل مع دوال متداخلة بعمق (دوال (دوال (...))). لحسن الحظ، تهتم قاعدة السلسلة Chain Rule بهذا الأمر. بالعودة إلى دوال متغير واحد، افترض أن $y = f(g(x))$ والدالة الأساسية $y = f(u)$ و $u = g(x)$ كلاهما قابل للتفاضل. تنص قاعدة السلسلة على ذلك

$$\frac{dy}{dx} = \frac{dy}{du} \frac{du}{dx}.$$

بالعودة إلى الدوال متعددة المتغيرات multivariate functions، افترض أن $y = f(\mathbf{u})$ لها متغيرات u_1, u_2, \dots, u_m ، حيث لكل $u_i = g_i(\mathbf{x})$ من \mathbf{x} منها متغيرات x_1, x_2, \dots, x_n ، أي. ثم تنص قاعدة السلسلة على ذلك

$$\frac{\partial y}{\partial x_i} = \frac{\partial y}{\partial u_1} \frac{\partial u_1}{\partial x_i} + \frac{\partial y}{\partial u_2} \frac{\partial u_2}{\partial x_i} + \dots + \frac{\partial y}{\partial u_m} \frac{\partial u_m}{\partial x_i} \text{ and thus } \nabla_{\mathbf{x}} y = \mathbf{A} \nabla_{\mathbf{u}} y,$$

حيث $\mathbf{A} \in \mathbb{R}^{n \times m}$ هي مصفوفة تحتوي على مشتق المتجه \mathbf{u} فيما يتعلق بالمتجه \mathbf{x} . وبالتالي، فإن تقييم التدرج يتطلب حساب ضرب المصفوفة - المتجه. هذا هو أحد الأسباب الرئيسية التي تجعل الجبر الخطي لبنة أساسية في بناء أنظمة التعلم العميق.

2.4.5 المناقشة

على الرغم من أننا قد خدشنا للتو سطح موضوع عميق، فقد تم التركيز بالفعل على عدد من المفاهيم: أولاً، يمكن تطبيق قواعد تكوين التفاضل rules for differentiation بلا تفكير، مما يمكننا من حساب التدرجات gradients تلقائياً. لا تتطلب هذه المهمة إبداعاً، وبالتالي يمكننا تركيز قرتنا المعرفية في مكان آخر. ثانياً، يتطلب حساب مشتقات الدوال ذات القيمة المتجهية vector-valued functions مضاعفة المصفوفات بينما نتبع الرسم البياني للتبعدية للمتغيرات من المخرجات إلى المدخلات. على وجه الخصوص، يتم اجتياز هذا الرسم البياني في اتجاه أمامي عندما نقوم بتقييم دالة وفي اتجاه عكسي عندما نحسب التدرجات. ستقدم الفصول اللاحقة بشكل رسمي الانتشار الخلفي backpropagation، وهو إجراء حسابي لتطبيق قاعدة السلسلة chain rule.

من وجهة نظر التحسين optimization، تسمح لنا التدرجات gradients بتحديد كيفية تحريك معلمات النموذج لتقليل الخطأ، وستطلب كل خطوة من خوارزميات التحسين المستخدمة في هذا الكتاب حساب التدرج.

2.4.6 التمارين

1. حتى الآن أخذنا قواعد المشتقات كأمر مسلم به. باستخدام التعريف والغايات اثبت خصائص $f(x) = c$ (i) $f(x) = x^n$ (ii) $f(x) = e^x$ (iii) $f(x) = \log x$ (iv).
2. على نفس المنوال، قم بإثبات قاعدة الضرب، المجموع، وحاصل القسمة من المبادئ الأولى.
3. إثبت أن قاعدة المضاعفات الثابتة constant multiple rule تتبع حالة خاصة لقاعدة حاصل الضرب.
4. احسب مشتق x^x .
5. ماذا يعني ذلك $f'(x) = 0$ بالنسبة لبعض x ? أعط مثالاً عن دالة f وموقع x قد يكون هذا مناسباً لهما.
6. ارسم الدالة $y = f(x) = x^3 - \frac{1}{x}$ وارسم خط المماس الخاص بها عند $x = 1$.
7. أوجد انحدار (تدرج) الدالة $f(x) = 3x_1^2 + 5e^{x_2}$.
8. هل يمكنك كتابة قاعدة السلسلة chain rule للحالة حيث $z = f(x, y, z)$ و $u = f(x, y, z)$ و $v = f(x, y, z)$ ؟
9. بالنظر إلى دالة $f(x)$ قابلة للعكس invertible، احسب مشتق معكوسها $(f^{-1}(x))'$. هنا لدينا ذلك $x = f(y)$ والعكس $y = f^{-1}(x)$. تلميح: استخدم هذه الخصائص في اشتراكك.

2.5 التفاضل التلقائي Automatic Differentiation

تدذكر من القسم 2.4 أن حساب المشتقات هو الخطوة الحاسمة في جميع خوارزميات التحسين التي سنسخدمها لتدريب الشبكات العميقية. في حين أن الحسابات واضحة و مباشرة، إلا أن حسابها يدوياً يمكن أن يكون مملاً وعرضة للخطأ، وتزداد هذه المشكلة فقط عندما تصبح نماذجنا أكثر تعقيداً.

لحسن الحظ، تعمل جميع أطر التعلم العميق الحديثة على إخراج هذا العمل من لوحاتنا من خلال تقديم التفاضل التلقائي Automatic Differentiation (غالباً ما يتم اختصاره إلى autograd). بينما تقوم بتمرير البيانات عبر كل دالة متتالية، يقوم إطار العمل بناء رسم بياني حسابي يتبع كيف تعتمد كل قيمة على الآخرين. لحساب المشتقات، تعمل حزم التفاضل التلقائي في الاتجاه المعاكس من خلال هذا الرسم البياني بتطبيق قاعدة السلسلة. الخوارزمية الحسابية لتطبيق قاعدة السلسلة بهذه الطريقة تسمى الانتشار الخلفي backpropagation.

بينما أصبحت مكتبات autograd مصدر قلق ساخن خلال العقد الماضي، إلا أنها تتمتع بتاريخ طويل. في الواقع، تعود أقدم الإشارات إلى autograd إلى أكثر من نصف قرن (Wengert، 1964). تعود الأفكار الأساسية الكامنة وراء الانتشار الخلفي backpropagation الحديثة إلى أطروحة دكتوراه من عام 1980 (Speelpenning، 1980) وتم تطويرها بشكل أكبر في أوائل الثمانينيات (Griewank، 1989). بينما أصبح الانتشار الخلفي هو الأسلوب الافتراضي لحساب التدرجات gradients، فإنه ليس الخيار الوحيد. على سبيل المثال، تستخدم لغة برمجة جوليا Julia (Julia et al., 2016) الانتشار الأمامي (Julia et al., 2016). قبل استكشاف الطرق، دعنا نتعمق أولًا في استخدام حزمة autograd.

2.5.1 دالة بسيطة A Simple Function

لنفترض أننا مهتمون بالتفاضل بين الدالة $y = 2x^T$ فيما يتعلق بمتجه العمود x . للبدء، نحدد قيمة أولية.

```
import tensorflow as tf
```

```
x = tf.range(4, dtype=tf.float32)
x
<tf.Tensor: shape=(4,), dtype=float32, numpy=array([0.,
1., 2., 3.], dtype=float32)>
```

قبل أن نحسب الانحدار بالنسبة لـ y ، نحتاج إلى مكان لتخزين x . بشكل عام، نتجنب تحصيص ذاكرة جديدة في كل مرة نأخذ فيها أحد المشتقات لأن التعلم العميق يتطلب مشتقات حوسية متتالية فيما يتعلق بنفس المعلومات آلاف أو ملايين المرات، وقد تخاطر بنفاد الذاكرة. لاحظ أن تدرج دالة ذات قيمة رقمية فيما يتعلق بالمتجه x لها قيمة متوجهة ولها نفس الشكل x .

```
x = tf.Variable(x)
```

نحسب الآن دالة x ونحسب النتيجة لـ y .

```
# Record all computations onto a tape
with tf.GradientTape() as t:
    y = 2 * tf.tensordot(x, x, axes=1)
```

y

```
<tf.Tensor: shape=(), dtype=float32, numpy=28.0>
```

يمكننا الآن حساب انحدار y بالنسبة إلى x باستدعاء دالة الانحدار gradient.

```
x_grad = t.gradient(y, x)
x_grad
```

```
<tf.Tensor: shape=(4,), dtype=float32, numpy=array([ 0.,
4., 8., 12.], dtype=float32)>
```

نحن نعلم بالفعل أن تدرج الدالة $y = 2x^T x$ إلى ي يجب أن يكون $4x$. يمكننا الآن التتحقق من تطابق حساب التدرج والنتيجة المتوقعة.

```
x_grad == 4 * x
<tf.Tensor: shape=(4,), dtype=bool, numpy=array([ True,
 True, True, True])>
```

دعونا الآن نحسب دالة أخرى لـ x ونأخذ انحدارها. لاحظ أن TensorFlow يعيد تعين المخزن المؤقت للتدرج كلما سجلنا تدرجاً جديداً.

```
with tf.GradientTape() as t:
    y = tf.reduce_sum(x)
t.gradient(y, x) # Overwritten by the newly calculated
gradient
<tf.Tensor: shape=(4,), dtype=float32, numpy=array([1.,
1., 1., 1.], dtype=float32)>
```

2.5.2. عكسياً بالنسبة للمتغيرات غير العددية - Backward for Non-Scalar Variables

عندما تكون y متتجهاً، فإن التفسير الأكثر طبيعية لمشتق y بالنسبة إلى المتتجه x هو مصفوفة تسمى Jacobian تحتوي على المشتقات الجزئية لكل مكون من y بالنسبة إلى كل مكون من مكونات x . وبالمثل، بالنسبة إلى y و x ذات الترتيب الأعلى، يمكن أن تكون نتيجة الاشتراق موترةً أعلى رتبة $.higher-order tensor$.

بينما يظهر Jacobian في بعض تقنيات التعلم الآلي المتقدمة ، فإننا نريد بشكل أكثر شيوعاً تلخيص تدرجات كل مكون من مكونات y فيما يتعلق بالمتتجه الكامل x ، مما يتوج عنه متتجه من نفس الشكل مثل x . على سبيل المثال ، غالباً ما يكون لدينا متتجه يمثل قيمة دالة الخطأ لدينا محسوبة بشكل منفصل لكل مجموعة من أمثلة التدريب. هنا، نريد فقط تلخيص التدرجات المحسوبة بشكل فردي لكل مثال.

بشكل افتراضي، يُرجع TensorFlow تدرج المجموع. بمعنى آخر، بدلاً من إرجاع $\partial_x \sum_i y_i$ ، فإنه يُرجع تدرج المجموع $\partial_x y$

```
with tf.GradientTape() as t:
    y = x * x
t.gradient(y, x) # Same as `y = tf.reduce_sum(x * x)`
<tf.Tensor: shape=(4,), dtype=float32, numpy=array([0.,
2., 4., 6.], dtype=float32)>
```

2.5.3. فصل الحساب Detaching Computation

في بعض الأحيان، نرغب في نقل بعض الحسابات خارج الرسم البياني الحسابي المسجل. على سبيل المثال، لنفترض أننا نستخدم المدخلات لإنشاء بعض المصطلحات الوسيطة المساعدة التي لا نريد حساب التدرج لها. في هذه الحالة، نحتاج إلى فصل detach الرسم البياني للتأثير الحسابي عن النتيجة النهائية. يوضح مثال اللعبة التالي هذا الأمر بشكل أوضح: افترض أن لدينا $y = x * z$ و $x * x = u$ لكتنا نريد التركيز على التأثير المباشر لـ x على z بدلاً من التأثير المترافق y عبر u . في هذه الحالة، يمكننا إنشاء متغير جديد u يأخذ نفس قيمة y ولكن مصدره (كيف تم إنشاؤه) قد تم محوه. وبالتالي، ليس لدى u أسلاف في الرسم البياني والتدرجات لا تتدفق عبر u إلى x . على سبيل المثال، سيؤدي أخذ التدرج لـ u إلى الحصول على النتيجة x ، (وليس كما كننا متوقعاً $x * x * x * 3$).

```
# Set `persistent=True` to preserve the compute graph.
# This lets us run `t.gradient` more than once
with tf.GradientTape(persistent=True) as t:
    y = x * x
    u = tf.stop_gradient(y)
    z = u * x

x_grad = t.gradient(z, x)
x_grad == u
<tf.Tensor: shape=(4,), dtype=bool, numpy=array([ True,
   True,  True,  True])>
```

2.5.4. التدرجات وتدفق التحكم في بایثون Gradients and Python Control Flow

لقد راجعنا حتى الآن الحالات التي تم فيها تحديد المسار من الإدخال إلى المخرجات بشكل جيد عبر دالة مثل $x * x * x = z$. تمنحنا البرمجة مزيداً من الحرية في كيفية حساب النتائج. على سبيل المثال، يمكننا جعلها تعتمد على المتغيرات المساعدة أو اختياريات الشرط على النتائج الوسيطة. تتمثل إحدى فوائد استخدام التفاضل التلقائي في أنه حتى إذا كان إنشاء الرسم البياني الحسابي لدالة ما يتطلب المرور عبر متاهة من تدفق التحكم control flow في بایثون (على سبيل المثال، الشرطية والحلقات واستدعاءات الدوال التعسفية)، فلا يزال بإمكاننا حساب التدرج للمتغير الناتج. لتوضيح ذلك، ضع في اعتبارك مقتطف الشفرة التالي حيث يعتمد عدد مرات تكرار حلقة while وتقييم جملة if على قيمة الإدخال a .

```
def f(a):
    b = a * 2
```

```

while tf.norm(b) < 1000:
    b = b * 2
if tf.reduce_sum(b) > 0:
    c = b
else:
    c = 100 * b
return c
<tf.Tensor: shape=(), dtype=float32, numpy=1024.0>

```

أدناه، نستدعي هذه الدالة، ونمرر قيمة عشوائية كمدخلات. نظرًا لأن الإدخال متغير عشوائي، فإننا لا نعرف الشكل الذي سيستخدمه الرسم البياني الحسابي computational graph. ومع ذلك، عندما ننفذ $f(a)$ على إدخال معين، فإننا ندرك رسمًا بيانيًا حسابيًا محدداً ويمكنا بعد ذلك الذهاب للخلف .backward

```

a = tf.Variable(tf.random.normal(shape=()))
with tf.GradientTape() as t:
    d = f(a)
d_grad = t.gradient(d, a)
d_grad
<tf.Tensor: shape=(), dtype=float32, numpy=1024.0>

```

على الرغم من أن دالتنا f مصممة قليلاً لأغراض توضيحية، إلا أن اعتمادها على المدخلات بسيط للغاية: إنها دالة خطية linear function بمقاييس محدد متعدد التعريف piecewise. على هذا التحو، فإن $a / f(a)$ عبارة عن متوجه للمدخلات الثابتة، علاوة على ذلك، تحتاج $a / f(a)$ إلى مطابقة تدرج $f(a)$ فيما يتعلق بـ a .

```

d_grad == d / a
<tf.Tensor: shape=(), dtype=bool, numpy=True>

```

تدفق التحكم الديناميكي Dynamic control flow شائع جدًا في التعلم العميق. على سبيل المثال، عند معالجة النص، يعتمد الرسم البياني الحسابي على طول المدخلات. في هذه الحالات، يصبح التفاضل التلقائي أمرًا حيوياً للنموذج الإحصائية لأنه من المستحيل حساب التدرج مسبقاً.

2.5.5 المناقشة

لقد اكتسبت الآن طعمًا لقوة التفاضل التلقائي. لقد كان تطوير المكتبات لحساب المستقات تلقائيًا وفعالًا بمثابة معزز كبير للإنتاجية لممارسي التعلم العميق، مما أتاح لهم التركيز على اهتمامات أعلى. علاوة على ذلك، يسمح لنا برنامج autograd بتصميم نماذج ضخمة تكون حسابات التدرج بالقلم والورق لها مضيعة للوقت. ومن المثير للاهتمام، أنه بينما نستخدم autograd لتحسين النماذج (بالمعنى الإحصائي)، فإن تحسين مكتبات autograd نفسها

(بالمعنى الحسابي) بعد موضوعاً غنياً ذا أهمية حيوية لمصممي إطار العمل. هنا، يتم الاستفادة من الأدوات من المجمعين ومعالجة الرسم البياني لحساب النتائج بالطريقة الأكثر ملاءمة وفعالية للذاكرة.

في الوقت الحالي، حاول أن تذكر هذه الأساسيات: (1) إرفاق التدرجات بتلك المتغيرات التي نرغب في المستويات فيما يتعلق بها؛ (2) تسجيل حساب القيمة المستهدفة؛ (3) تنفيذ دالة الانشار الخلفي backpropagation؛ و (4) الوصول إلى التدرج الناتج.

2.5.6 التمارين

- لماذا يكون حساب المشتق الثاني أكثر تكلفة بكثير من المشتق الأول؟
- بعد تشغيل دالة backpropagation، قم بتشغيلها على الفور مرة أخرى وشاهد ما سيحدث. لماذا؟
- في مثال تدفق التحكم control flow حيث نحسب مشتق d بالنسبة إلى a ، ماذا سيحدث إذا غيرنا المتغير a إلى متوجه عشوائي أو مصفوفة؟ في هذه المرحلة، لم تعد نتيجة الحساب $f(a)$ عدداً قياسياً. ماذا يحدث للنتيجة؟ كيف نحلل هذا؟
- لتكن $(x) = \sin(x)$. ارسم الرسم البياني لـ f ومشتقاته f' . لا تستغل حقيقة ذلك $(x) = \cos(x)$ بل استخدم التفاضل التلقائي للحصول على النتيجة.
- لتكن $x^{-1} \cdot \sin x + (\log x^2) \cdot f(x) = f(x)$. اكتب نتائج تتبع الرسم البياني للتبعية من x إلى $f(x)$.
- استخدم قاعدة السلسلة chain rule لحساب المشتق $\frac{df}{dx}$ من الدالة المذكورة أعلاه، مع وضع كل مصطلح على الرسم البياني للتبعية الذي قمت بإنشائه مسبقاً.
- بالنظر إلى الرسم البياني ونتائج المشتقون الوسيطة، لديك عدد من الخيارات عند حساب التدرج. قم بتقييم النتيجة بمجرد البدء من x إلى f ومرة واحدة من f التبع إلى الخلف x . يُعرف المسار من x إلى f بشكل عام باسم التفاضل الأمامي forward differentiation، بينما يُعرف المسار من f إلى x باسم التمايز الخلفي backward differentiation.
- متى قد ترغب في استخدام التفاضل الأمامي ومتي التفاضل الخلفي؟ تلميح: ضع في اعتبارك مقدار البيانات الوسيطة المطلوبة، والقدرة على موازنة الخطوات، وحجم المصفوفات والمتوجهات المعنية.

2.6 الاحتمال والاحصاء Probability and Statistics

بطريقة أو بأخرى، يدور التعلم الآلي حول عدم اليقين uncertainty. في التعلم الخاضع للإشراف، نريد أن نتنبأ بشيء غير معروف (الهدف target) بالنظر إلى شيء معروف (الميزات

(features). اعتماداً على هدفنا، قد نحاول توقع القيمة الأكثر احتمالاً للهدف. أو قد نتوقع القيمة بأقل مسافة متوقعة من الهدف. وأحياناً لا نرغب فقط في التنبؤ بقيمة معينة ولكن تحديد عدم اليقين لدينا quantify our uncertainty. على سبيل المثال، بالنظر إلى بعض الميزات التي تصف المريض، قد نرغب في معرفة مدى احتمالية تعرضه لأزمة قلبية في العام المقبل. في التعلم غير الخاضع للإشراف، غالباً ما نهتم بعدم اليقين. لتحديد ما إذا كانت مجموعة القياسات شاذة anomalous، من المفيد معرفة مدى احتمالية ملاحظة القيم في المجتمع محل الاهتمام. علاوة على ذلك، في التعلم المعزز reinforcement learning، نرغب في تطوير عوامل تعامل بذكاء في بيئات مختلفة. يتطلب هذا التفكير في كيفية توقع تغيير البيئة والمكافآت التي قد يتوقع المرء مواجهتها استجابة لكل من الإجراءات المتاحة.

الاحتمالية Probability هو المجال الرياضي المعنى بالاستدلال reasoning في ظل عدم اليقين. بالنظر إلى نموذج احتمالي لبعض العمليات، يمكننا التفكير في احتمال وقوع أحداث مختلفة. إن استخدام الاحتمالات لوصف تكرار الأحداث القابلة للتكرار (مثل رمي العملات المعدنية coin tosses) غير مثير للجدل إلى حد ما. في الواقع، يلتزم العلماء frequentist scholars بتفسير الاحتمال الذي ينطبق فقط على مثل هذه الأحداث القابلة للتكرار. على النقيض من ذلك، يستخدم علماء بايز Bayesian scholars لغة الاحتمال على نطاق أوسع لإضفاء الطابع الرسمي على تفكيرنا في ظل عدم اليقين. يتميز احتمالية بايز Bayesian probability بميزتين فريدتين: (1) تعيين درجات من الاعتقاد للأحداث غير القابلة للتكرار، على سبيل المثال، ما هو احتمال أن يكون القمر مصنوعاً من الجبن؟ و (2) الذاتية subjectivity – بينما يوفر احتمالية بايز قواعد لا لبس فيها لكيفية تحديث المرء لمعتقداته في ضوء أدلة جديدة، فإنه يسمح للأفراد المختلفين بالبدء بمعتقدات سابقة مختلفة. تساعدها الإحصائيات على التفكير بشكل عكسي، بدءاً من جمع البيانات وتنظيمها والتراجع عن الاستنتاجات التي قد نستخلصها حول العملية التي أنتجت البيانات. عندما نحلل مجموعة بيانات، ونبحث عن الأنماط التي نأمل أن تميز مجموعة أكبر من السكان، فإننا نستخدم التفكير الإحصائي. تم تخصيص معظم الدورات والتخصصات والأطروحات والمهن والإدارات والشركات والمؤسسات لدراسة الاحتمالات والإحصاءات. في حين أن هذا القسم يخوض السطح فقط، فإننا سنوفر الأساس الذي تحتاجه لبدء بناء النماذج.

2.6.1 مثال بسيط: رمي العملات المعدنية Tossing Coins

تخيل أننا نخطط لرمي عملة معدنية ونريد تحديد مدى احتمالية رؤية الرؤوس heads (مقابل tails). إذا كانت العمدة عادلة، فإن كلا النتيجتين (الرأس والذيل) متساويان في الاحتمال. علاوة على ذلك، إذا كنا نخطط لـ n لقاءات العملة، فيجب أن يتطابق الجزء الذي تتوقع رؤيته من الرؤوس تماماً مع الكسر المتوقع من ذيول. إحدى الطرق البديهية لرؤيه ذلك هي من خلال

التناسق symmetry: لكل نتيجة محتملة مع n_h رؤوس و($n - n_h$) ذيول، هناك نتيجة محتملة متساوية مع n_t رؤوس و n_h ذيول. لاحظ أن هذا ممكّن فقط إذا كانا متوقع في المتوسط رؤية $\frac{1}{2}$ الرميات تظهر على الرؤوس و $\frac{1}{2}$ تظهر على الذيول. بالطبع، إذا أجريت هذه التجربة عدّة مرات مع كل $n = 1000000$ رميات، فقد لا ترى تجربة في أي $n_h = n_t$ بالضبط.

رسمياً، تسمى الكمية $\frac{1}{2}$ بالاحتمالية probability وهي هنا تلتقط اليقين الذي ستظهر به أي رمية. تقوم الاحتمالات بتعيين درجات بين 0 و1 نتائج الاهتمام outcomes of interest، تسمى الأحداث events. هنا حدث الاهتمام بالرأس heads ونشير إلى الاحتمال المقابل $P(\text{heads})$. يشير احتمال 1 اليقين المطلقاً (تخيل عملية خدعة حيث كان كلا الجانبيين رؤوساً) واحتمال 0 يشير إلى استحالة (على سبيل المثال، إذا كان كلا الجانبيين ذيول). الترددات $\frac{n_t}{n}$ و $\frac{n_h}{n}$ ليست احتمالات بل إحصائيات statistics. الاحتمالات هي الكميات النظرية التي تقوم عليها عملية توليد البيانات. هنا، الاحتمال $\frac{1}{2}$ هو خاصية للعملة نفسها. على النقيض من ذلك، الإحصائيات هي كميات تجريبية يتم حسابها كدوال للبيانات المرصودة. تتشابك اهتماماتنا في الكميات الاحتمالية والإحصائية بشكل لا ينفصّم. غالباً ما نصمم إحصائيات خاصة تسمى المقدّرات estimators التي، في ضوء مجموعة بيانات، تنتج تقدّيرات لمعلمات النموذج مثل الاحتمالات. علاوة على ذلك، عندما تتحقّق تلك المقدّرات خاصية لطيفة تسمى الاتساق consistency، فإن تقدّيراتنا ستتقارب مع الاحتمال المقابل. بدورها، تخبرنا هذه الاحتمالات المستنيرة عن الخصائص الإحصائية المحتملة للبيانات من نفس المجتمع والتي قد نواجهها في المستقبل.

لنفترض أننا عثرنا على عملية حقيقة لم نكن نعرف حقيقة وجودها $P(\text{heads})$. للتحقق من هذه الكمية بالطرق الإحصائية، نحتاج إلى (1) جمع بعض البيانات؛ و (2) تصميم مقدر estimator. الحصول على البيانات هنا سهل؛ يمكننا رمي العملة عدّة مرات وتسجيل جميع النتائج. رسمياً، يُطلق على استخلاص الإنجازات من بعض العمليات العشوائية الأساسيةأخذ العينات sampling. كما قد تكون خمنت، أحد المقدّر الطبيعى natural estimator هو الكسر fraction بين عدد الرؤوس المرصودة observed heads بالعدد الإجمالي للرمي number of tosses.

```
%matplotlib inline
import random
import tensorflow as tf
from tensorflow_probability import distributions as tfd
from d2l import tensorflow as d2l
```

الآن، افترض أن العمالة كانت في الواقع عادلة، أي $P(\text{heads}) = 0.5$. لمحاكاة رميات عملة عادلة، يمكننا استدعاء أي مولد أرقام عشوائي. بعض الطرق السهلة لرسم عينات من حدث ذي احتمالية. على سبيل المثال، ينبع `random.random()` في `random` في بايثون أرقاماً في الفترة $[0,1]$ حيث يكون احتمال الكذب في أي فتره فرعية $[0,1] \subset [a,b]$ مساوياً $a - b$. وبالتالي يمكننا الخروج من 0 و 1 باحتمال 0.5 لكل منهما باختبار ما إذا كان الطفو المرتجل أكبر من 0.5

```
num_tosses = 100
heads = sum([random.random() > 0.5 for _ in range(100)])
tails = num_tosses - heads
print("heads, tails: ", [heads, tails])
<tf.Tensor: shape=(2,), dtype=float32, numpy=array([45., 55.], dtype=float32)>
```

هنا، على الرغم من أن عملتنا المحاكاة عادلة (قمنا بتعيين الاحتمالات $[0.5, 0.5]$ بأنفسنا)، قد لا تكون تعدادات الرأس والذيل متطابقة. ذلك لأننا رسمنا عددًا محدودًا فقط من العينات. إذا لم ننفذ المحاكاة بأنفسنا، ورأينا النتيجة فقط، فكيف لنا أن نعرف ما إذا كانت العمالة غير عادلة إلى حد ما أو إذا كان الانحراف المحتمل عنها $\frac{1}{2}$ مجرد قطعة أثرية من حجم العينة الصغير؟ دعونا نرى ما يحدث عندما نقوم بمحاكاة 10000 رمية.

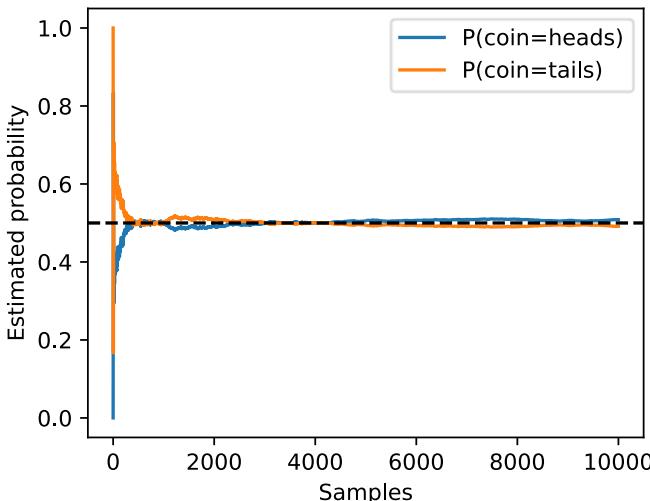
```
counts = tfd.Multinomial(10000, fair_probs).sample()
counts / 10000
<tf.Tensor: shape=(2,), dtype=float32,
numpy=array([0.5056, 0.4944], dtype=float32)>
```

بشكل عام، بالنسبة لمتوسطات الأحداث المتكررة (مثل رمي العملات المعدنية)، مع تزايد عدد التكرارات، نضمن أن تقارب تقديراتنا مع الاحتمالات الأساسية الحقيقية. يُطلق على الدليل الرياضي لهذه الظاهرة اسم قانون الأعداد الكبيرة law of large numbers وتخبرنا نظرية الحد المركزي central limit theorem أنه في العديد من المواقف، مع نمو حجم العينة n ، يجب أن تنخفض هذه الأخطاء بمعدل $(\frac{1}{\sqrt{n}})$. دعنا نحصل على مزيد من الحدس من خلال دراسة كيفية تطور تقديراتنا مع زيادة عدد الرميات من 1 إلى 10000.

```
counts = tfd.Multinomial(1, fair_probs).sample(10000)
cum_counts = tf.cumsum(counts, axis=0)
estimates = cum_counts / tf.reduce_sum(cum_counts,
axis=1, keepdims=True)
estimates = estimates.numpy()

d2l.set_figsize((4.5, 3.5))
```

```
d21=plt.plot(estimated[:, 0], label="P(coin=heads)")
d21=plt.plot(estimated[:, 1], label="P(coin=tails"))
d21=plt.axhline(y=0.5, color='black',
linestyle='dashed')
d21=plt.gca().set_xlabel('Samples')
d21=plt.gca().set_ylabel('Estimated probability')
d21=plt.legend();
```



يتافق كل منحنى صلب مع إحدى قيمتي العملة المعدنية ويعطي الاحتمالية المقدرة لعملة العملة هذه بعد كل مجموعة من التجارب. يعطي الخط الأسود المتقطع الاحتمال الأساسي الحقيقي. مع حصولنا على المزيد من البيانات من خلال إجراء المزيد من التجارب، تتقارب المنحنيات نحو الاحتمال الحقيقي. قد تبدأ بالفعل في رؤية شكل بعض الأسئلة الأكثر تقدماً التي تشغيل بالإحصائيين: ما مدى سرعة حدوث هذا التقارب convergence؟ إذا كنا قد اختبرنا بالفعل العديد من العملات المعدنية المصنعة في نفس المصنع، فكيف يمكننا دمج هذه المعلومات؟

2.6.2. معاملة رسمية أكثر

لقد وصلنا بالفعل إلى حد بعيد: طرح نموذج احتمالي، وإنشاء بيانات تركيبية، وتشغيل مقدر إحصائي، وتقييم التقارب بشكل تجريبي، والإبلاغ عن مقاييس الخطأ (التحقق من الانحراف deviation). ومع ذلك، للمضي قدماً، سنحتاج إلى أن نكون أكثر دقة.

عند التعامل مع العشوائية randomness، فإننا نشير إلى مجموعة النتائج المحتملة \mathcal{S} ونطلق عليها مساحة العينة sample space أو مساحة النتيجة outcome space. هنا، كل عنصر هو

نتيجة ممكنة مميزة في حالة درجة عملة واحدة، $S = \{heads, tails\}$. لنردد واحد = $\{1, 2, 3, 4, 5, 6\}$. عند قلب عملتين، يكون لدينا أربع نتائج محتملة:

$$\{(heads, heads), (heads, tails), (tails, heads), (tails, tails)\}$$

الأحداث هيمجموعات فرعية من مساحة العينة. على سبيل المثال، حدث "ظهور أول رمية عملة على الرأس" يتوافق مع المجموعة $\{(heads, heads)\}$. كلما كانت نتيجة التجربة العشوائية تتحقق $A, z \in A$ حدث ذلك. بالنسبة إلى لفة واحدة من النرد، يمكننا تحديد الأحداث "رؤى 5" ($A = \{5\}$) و "رؤى رقم فردي" ($B = \{1, 3, 5\}$). في هذه الحالة، إذا جاء النرد 5، فستقول أن كلاهما A و B حدث. من ناحية أخرى، إذا $z = 3$ ، إذن A لم يحدث بل B حدث.

تقوم دالة الاحتمال بتعيين الأحداث على قيم حقيقية $[0, 1] \rightarrow P: A \subseteq S \rightarrow P(A)$. يفي احتمال وقوع حدث A في مساحة العينة المحددة S بالخصائص التالية:

- احتمال حدوث أي حدث A هو رقم حقيقي غير سالب، أي $P(A) \geq 0$:
- احتمال مساحة العينة بأكمتها هو 1 ، أي $P(S) = 1$:
- بالنسبة لأي تسلسل قابل للعد من الأحداث ... A_1, A_2, \dots التي تكون متنافية $(A_i \cap A_j = \emptyset)$ mutually exclusive فإن احتمال حدوث أي منها يساوي مجموع احتمالاتها الفردية، أي $P(\bigcup_{i=1}^{\infty} A_i) = \sum_{i=1}^{\infty} P(A_i)$.

يمكن تطبيق بديهيات نظرية الاحتمالات، التي اقترحها كولموغوروف (1933)، لاشتقاق عدد من النتائج المهمة بسرعة. على سبيل المثال، يترتب على ذلك على الفور أن احتمال حدوث أي حدث A أو مكمل حدوثه A' يحدثان في وقت واحد هو $P(A \cap A') = 0$. بشكل غير رسمي، يخبرنا هذا أن الأحداث المستحيلة ليس لها أي احتمال لحدوثها.

2.6.3. المتغيرات العشوائية Random Variables

عندما تحدثنا عن أحداث مثل ظهور احتمالات ظهور النرد أو ظهور أول عملة معدنية، كنا نستدعي فكرة المتغير العشوائي random variable. بشكل رسمي، المتغيرات العشوائية عبارة عن تعيينات من مساحة عينة أساسية إلى مجموعة من (ربما العديد) من القيم. قد تتساءل كيف يختلف المتغير العشوائي عن مساحة العينة، لأن كلاهما عبارة عن مجموعة من النتائج. الأهم من ذلك، يمكن أن تكون المتغيرات العشوائية أكثر خشونة much coarser من مساحة العينة الأولية. يمكننا تحديد متغير عشوائي ثانئي مثل "أكبر من 0.5" حتى عندما تكون مساحة العينة الأساسية لانهائية، على سبيل المثال، المقطع الخطي بين 0 و 1. بالإضافة إلى ذلك، يمكن للمتغيرات العشوائية المتعددة أن تشتراك في نفس مساحة العينة الأساسية. على سبيل المثال، "ما

إذا كان إنذار منزلي ينطلق" و "ما إذا كان منزلي قد تعرض للسطو" كلاهما متغيرين عشوائيين يشتراكان في مساحة عينة أساسية. وبالتالي، فإن معرفة القيمة المأخوذة بواسطه متغير عشوائي واحد يمكن أن يخبرنا شيئاً عن القيمة المحتملة لمتغير عشوائي آخر. مع العلم أن جهاز الإنذار قد انطلق، قد نشك في احتمال تعرض المنزل للسطو.

تتوافق كل قيمة مأخوذة بواسطه متغير عشوائي مع مجموعة فرعية من مساحة العينة الأساسية. وبالتالي فإن الحدوث الذي يأخذ فيه المتغير العشوائي X قيمة v ، يُرمز إليه بـ $P(X = v)$ ، هو حدث ويشير إلى احتماله. في بعض الأحيان، يمكن أن يصبح هذا الترميز غير مؤكداً، ويمكننا إساءة استخدام التدوين عندما يكون السياق واضحًا. على سبيل المثال، قد نستخدم $P(X)$ للإشارة على نطاق واسع إلى توزيع X ، أي الدالة التي تخبرنا بالاحتمال الذي X يأخذ أي قيمة معينة. في أحيان أخرى نكتب تعبيرات مثل $P(X, Y) = P(X)P(Y)$ ، كاختصار للتعبير عن عبارة صحيحة لجميع القيم التي يمكن أن تأخذها المتغيرات العشوائية X و Y ، أي لكل i, j ما تحمله $(j = i)P(X = i)P(Y = j) = P(X = i \text{ and } Y = j)$. في أوقات أخرى، نسيء استخدام التدوين عن طريق الكتابة عندما يكون المتغير العشوائي واضحًا من السياق. نظرًا لأن الحدث في نظرية الاحتمالات هو مجموعة من النتائج من مساحة العينة، يمكننا تحديد نطاق من القيم لمتغير عشوائي ليأخذها. على سبيل المثال، يدل $P(1 \leq X \leq 3)$ على احتمال وقوع الحدث $\{1 \leq X \leq 3\}$.

لاحظ أن هناك فرقاً طفيفاً بين المتغيرات العشوائية المقطعة discrete random variables، مثل تقليب عملة معدنية أو رمي نرد، والمتغيرات المستمرة continuous، مثل وزن وطول الشخص المأخوذ عشوائياً من السكان في هذه الحالة نادرًا ما نهتم حقًا بالطول الدقيق لشخص ما. علاوة على ذلك، إذا أخذنا قياسات دقيقة كافية، لن يكون لديك نفس الارتفاع لهما نفس الارتفاع بالضبط. في الواقع، مع قياسات دقيقة كافية، لنقل بين 1.79 و 1.81 متراً. في هذه عند الاستيقاظ وعندما تنام. ليس هناك فائدة تذكر في السؤال عن الاحتمال الدقيق أن يبلغ طول شخص ما إذا كان ارتفاع شخص ما يقع في فترة زمنية معينة، لنقل بين 1.79 و 1.81 متراً. في هذه تحديد ما إذا كان ارتفاع شخص ما يقع في فترة زمنية معينة، لنقل بين 1.79 و 1.81 متراً. في هذه الحالات، نتعامل مع كثافات الاحتمالات. لا يوجد احتمال لارتفاع 1.80 متراً بالضبط، لكن كثافة غير صفرية nonzero density. لإخراج الاحتمال المخصص لفترة ما، يجب أن نأخذ جزءاً لا يتجزأ من الكثافة على تلك الفترة.

2.6.4 متغيرات عشوائية متعددة Multiple Random Variables

ربما لاحظت أنه لا يمكننا حتى تجاوز القسم الأخير دون الإدلاء بعبارات تتضمن تفاعلات بين متغيرات عشوائية متعددة (استدعاء $P(X, Y) = P(X)P(Y)$). يهتم معظم التعلم الآلي بمثل

هذه العلاقات. هنا، ستكون مساحة العينة هي الفئة المستهدفة، كما يقول العملاء الذين يتعاملون مع شركة، أو صور فوتوغرافية على الإنترنت، أو بروتينات معروفة لعلماء الأحياء. يمثل كل متغير عشوائي القيمة (غير المعروفة) لسمة مختلفة. عندما نقوم بأخذ عينة من فرد من السكان، نلاحظ تحقيق كل من المتغيرات العشوائية. نظراً لأن القيم المأخوذة بواسطة المتغيرات العشوائية تتوافق معمجموعات فرعية من مساحة العينة التي يمكن أن تكون متداخلة أو متداخلة جزئياً أو منفصلة تماماً، فإن معرفة القيمة المأخوذة بواسطة متغير عشوائي واحد يمكن أن تجعلنا نقوم بتحديث معتقداتنا حول القيم المحتملة لمتغير عشوائي آخر. إذا دخل المريض إلى المستشفى ولاحظنا أنه يعاني من صعوبة في التنفس فقد حاسة الشم، فإننا نعتقد أنه من المرجح أن يكون مصاباً بـ COVID-19 أكثر مما قد يكون عليه الحال إذا لم يكن لديه مشكلة في التنفس وكان عادياً تماماً. حاسة الشم.

عند العمل مع متغيرات عشوائية متعددة، يمكننا إنشاء أحاديث تتوافق مع كل مجموعة من القيم التي يمكن أن تأخذها المتغيرات بشكل مشترك. تسمى دالة الاحتمال التي تعين الاحتمالات لكل من هذه المجموعات (على سبيل المثال $a = A = b$ و $b = B$) دالة الاحتمال المشترك joint probability function وترجع ببساطة الاحتمال المعين لتقاطع المجموعات الفرعية المقابلة من مساحة العينة. الاحتمال المشترك المخصص للحدث حيث يتم الإشارة إلى المتغيرات العشوائية A و B القيم a و b ، على التوالي، حيث تشير $P(A = a, B = b)$ والفاصلة إلى $P(A = a, B = b) \leq$ "and". لاحظ أنه بالنسبة لأية قيم a و b ، فهي تحمل ذلك $P(A = a, B = b) \leq P(A = a)$ ، ومنذ حدوث $A = a$ ، $B = b$ يجب أن يحدث $B = b$ يجب أن يحدث أيضاً. ومن المثير للاهتمام أن الاحتمال المشترك يخبرنا بكل ما يمكننا معرفته عن هذه المتغيرات العشوائية بالمعنى الاحتمالي، ويمكن استخدامه لاشتقاق العديد من الكميات المفيدة الأخرى، بما في ذلك استعادة التوزيعات الفردية $P(A)$ و $P(B)$. لاسترداد $P(A = a)$ ، نجمع ببساطة $P(A = a, B = v)$ جميع القيم v التي يمكن أن تأخذها المتغير العشوائي B :

$$P(A = a) = \sum_v P(A = a, B = v)$$

النسبة $\frac{P(A=a,B=b)}{P(A=a)}$ تبين أنها مهمة للغاية. يطلق عليه الاحتمال الشرطي conditional probability، ويُشار إليه بالرمز " $|$ ". يخبرنا الاحتمال الجديد المرتبط بالحدث $b = B$ ، بمجرد أن نشتراك في حقيقة حدوثه $a = A$. يمكننا أن نفك في هذا الاحتمال الشرطي على أنه يقيد الانتباه فقط إلى المجموعة الفرعية من مساحة العينة المرتبطة بـ $a = A$ ثم إعادة التسوية renormalizing بحيث تصل جميع الاحتمالات إلى 1. الاحتمالات الشرطية هي في الواقع احتمالات، وبالتالي نحترم جميع البديهيات axioms، طالما أنت شرط جميع المصطلحات في نفس الحدث وبالتالي قصر الانتباه على نفس مساحة العينة. على سبيل المثال، بالنسبة للأحداث

المنفصلة B و B' لدينا ذلك $P(B' | A = a)$

باستخدام تعريف الاحتمالات الشرطية conditional probabilities، يمكننا اشتقاق النتيجة الشهيرة المسماة نظرية بايز Bayes' theorem. من خلال البناء، لدينا ذلك $P(A, B) = P(A | B)P(B)$ و $P(A | B) = P(A | B)P(B) / P(B | A)P(A)$. الجمع بين كلا المعادلين يتبع وبالتالي $P(B | A)P(A) = P(A | B)P(B)$

$$P(A | B) = \frac{P(B | A)P(A)}{P(B)}.$$

هذه المعادلة البسيطة لها آثار عميقة لأنها تسمح لنا بعكس ترتيب الشرط. إذا عرفنا كيف نقدر $P(B | A)$ ، ثم يمكننا أن نقدر $P(A | B)$. غالباً ما نجد أنه من الأسهل تقدير مصطلح واحد بشكل مباشر ولكن ليس الآخر ويمكن أن تتفق نظرية بايز Bayes' theorem هنا. على سبيل المثال، إذا عرفنا مدى انتشار أعراض مرض معين، والانتشار العام للمرض والأعراض، على التوالي، يمكننا تحديد مدى احتماليةإصابة شخص ما بالمرض بناءً على أعراضه. في بعض الحالات، قد لا يكون لدينا وصول مباشر لـ $P(B)$ ، مثل انتشار الأعراض. في هذه الحالة، تكون النسخة المبسطة من نظرية بايز مفيدة:

$$P(A | B) \propto P(B | A)P(A).$$

بما أننا نعلم أن $P(A | B) \propto P(B | A)P(A)$ يجب تسويته إلى 1 ، أي $\sum_a P(A = a | B) = 1$ يمكننا استخدامه لحساب:

$$P(A | B) = \frac{P(B | A)P(A)}{\sum_b P(B = b | A)P(A)}.$$

في إحصائيات بايز Bayesian statistics، نعتقد أن المراقب observer يمتلك بعض المعتقدات السابقة (الذاتية subjective) حول معقولة الفرضيات المتاحة المشفرة في السابق $P(H)$ ، ودالة احتمالية توضح مدى احتمالية ملاحظة أي قيمة للأدلة التي تم جمعها لكل من الفرضيات في الفتة $P(E | H)$. ثم يتم تفسير نظرية بايز على أنها تخبرنا بكيفية تحديث السابقة $P(H | E)$ initial prior في ضوء الأدلة المتاحة E لإنتاج معتقدات لاحقة $P(H | E)$. بشكل غير رسمي، يمكن ذكر ذلك على أنه "لاحقاً يساوي احتمالية الأوقات السابقة prior times likelihood على الدليل evidence". الآن، نظراً لأن الدليل $P(E)$ هو نفسه بالنسبة لجميع الفرضيات، يمكننا التخلص ببساطة من تسوية الفرضيات.

لاحظ أن $P(A = a | B) = \sum_a P(A = a, B)$ يسمح لنا أيضًا بالتهبيش marginalize على المتغيرات العشوائية. وهذا يعني أنه يمكننا إسقاط المتغيرات من التوزيع المشترك مثل $P(A, B)$. بعد كل شيء، لدينا ذلك

$$\sum_a P(A = a, B) = P(B) \sum_a P(A = a | B) = P(B).$$

الاستقلال Independence هو مفهوم آخر مهم بشكل أساسي يشكل العمود الفقري للعديد من الأفكار المهمة في الإحصاء. باختصار، يكون متغيرين مستقلين إذا كان التكيف conditioning على قيمة A لا يسبب أي تغيير في توزيع الاحتمالات المرتبط بـ B والعكس صحيح. بشكل رسمي أكثر، الاستقلال، والمشار إليه $A \perp B$ ، يتطلب ذلك $P(A | B) = P(A)$ ، وبالتالي، ذلك $P(A, B) = P(A | B)P(B) = P(A)P(B)$. غالباً ما يكون الاستقلال افتراضًا مناسباً. على سبيل المثال، إذا كان المتغير العشوائي A يمثل النتيجة من رمي عملة عادلة ويمثل المتغير العشوائي B النتيجة من رمي عملة أخرى، فإن معرفة ما إذا كان المتغير العشوائي A لا يجب أن يؤثر على احتمالية B ظهور الرأس.

يكون الاستقلال مفيداً بشكل خاص عندما يكون بين السحبوبات draws المتتالية لبياناتنا من بعض التوزيعات الأساسية (مما يسمح لنا بعمل استنتاجات إحصائية قوية) أو عندما يكون بين المتغيرات المختلفة في بياناتنا، مما يسمح لنا بالعمل مع نماذج أبسط ترميز بنية الاستقلال هذه. من ناحية أخرى، غالباً ما يكون تقدير التبعيات بين المتغيرات العشوائية هو الهدف الأساسي للتعلم. نحن نهتم بتقدير احتمالية المرض نظراً للأعراض على وجه التحديد لأننا نعتقد أن الأمراض والأعراض ليست مستقلة.

لاحظ أنه نظراً لأن الاحتمالات الشرطية هي احتمالات مناسبة، فإن مفاهيم الاستقلال independence والاعتماد dependence تتطابق عليها أيضاً. متغيرين عشوائيين A و B مستقلان بشكل مشروط مع الأخذ في الاعتبار متغير ثالث C إذا وفقط إذا $= P(A, B | C) = P(A | C)P(B | C)$. ومن المثير للاهتمام، أن متغيرين يمكن أن يكونا مستقلين بشكل عام ولكنهما يعتمدان عند التكيف على متغير ثالث. يحدث هذا غالباً عندما يتوافق المتغيران العشوائيان A و B مع أسباب متغير ثالث C . على سبيل المثال، قد تكون العظام المكسورة وسرطان الرئة مستقلين في عموم السكان، ولكن إذا شرطنا وجودنا في المستشفى، فقد نجد أن العظام المكسورة مرتبطة سلباً بسرطان الرئة. وذلك لأن العظم المكسور يفسر سبب وجود شخص مافي المستشفى وبالتالي يقلل من احتمالية إصابته بسرطان الرئة.

وعلى العكس من ذلك، يمكن أن يصبح متغيرين عشوائيين مستقلين عند التكيف على متغير ثالث. يحدث هذا غالباً عندما يكون لحدثين غير مرتبطين سبيلاً مشتركاً. يرتبط حجم الحذاء

ومستوى القراءة ارتباطاً وثيقاً بين طلاب المدارس الابتدائية، لكن هذا الارتباط يختفي إذا شرطنا في العمر.

2.6.5 مثال

دعونا نختبر مهاراتنا. افترض أن الطبيب يقوم بإجراء اختبار فيروس نقص المناعة البشرية HIV على المريض. هذا الاختبار دقيق إلى حد ما ولا يفشل إلا مع احتمال 1%. إذا كان المريض يتمتع بصحة جيدة ولكنه يبلغ عن إصابته بالمرض. علاوة على ذلك، فإنه لا يفشل أبداً في اكتشاف فيروس نقص المناعة البشرية إذا كان المريض مصاباً به بالفعل. نستخدم $D_1 \in \{0,1\}$ للإشارة إلى التشخيص (0 إذا كان سلبياً و1 إذا كان إيجابياً) و $H \in \{0,1\}$ للإشارة إلى حالة فيروس نقص المناعة البشرية.

Conditional probability	$H = 1$	$H = 0$
$P(D_1 = 1 H)$	1	0.01
$P(D_1 = 0 H)$	0	0.99

لاحظ أن مجاميع الأعمدة كلها 1 (لكن مجاميع الصفوف ليست كذلك)، لأنها احتمالات مشروطة. دعونا نحسب احتمال إصابة المريض بفيروس نقص المناعة البشرية إذا كانت نتيجة الاختبار إيجابية، أي $P(H = 1 | D_1 = 1)$. حديدياً، سيعتمد هذا على مدى انتشار المرض، لأنه يؤثر على عدد الإنذارات الكاذبة. افترض أن السكان يتمتعون بصحة جيدة إلى حد ما، على سبيل المثال، $P(H = 1) = 0.0015$. لتطبيق نظرية بايز، نحتاج إلى تطبيق التهميش marginalization

$$\begin{aligned} P(D_1 = 1) &= P(D_1 = 1, H = 0) + P(D_1 = 1, H = 1) \\ &= P(D_1 = 1 | H = 0)P(H = 0) + P(D_1 = 1 | H = 1)P(H = 1) \\ &= 0.011485. \end{aligned}$$

هذا يقودنا إلى

$$P(H = 1 | D_1 = 1) = \frac{P(D_1 = 1 | H = 1)P(H = 1)}{P(D_1 = 1)} = 0.1306.$$

بمعنى آخر، هناك احتمال بنسبة 13.06%. فقط أن يكون المريض مصاباً بالفعل بفيروس نقص المناعة البشرية، على الرغم من استخدام اختبار دقيق للغاية. كما نرى، يمكن أن يكون الاحتمال مخالفًا للحدس. ماذا يفعل المريض عند تلقي مثل هذه الأخبار المرعبة؟ من المحتمل أن يطلب

المريض من الطبيب إجراء اختبار آخر لتوضيح الأمر. الاختبار الثاني له خصائص مختلفة وهو ليس بنفس جودة الاختبار الأول.

Conditional probability	$H = 1$	$H = 0$
$P(D_2 = 1 H)$	0.98	0.03
$P(D_2 = 0 H)$	0.02	0.97

لسوء الحظ، الاختبار الثاني يأتي إيجابياً أيضاً. دعونا نحسب الاحتمالات المطلوبة لاستدعاء نظرية بايز بافتراض الاستقلال الشرطي conditional independence:

$$P(D_1 = 1, D_2 = 1 | H = 0) = P(D_1 = 1 | H = 0)P(D_2 = 1 | H = 0) = 0.0003,$$

$$P(D_1 = 1, D_2 = 1 | H = 1) = P(D_1 = 1 | H = 1)P(D_2 = 1 | H = 1) = 0.98.$$

يمكنا الآن تطبيق التهميش marginalization للحصول على احتمالية أن كلا الاختبارين يأتي بنتائج إيجابية:

$$P(D_1 = 1, D_2 = 1) = P(D_1 = 1, D_2 = 1, H = 0) + P(D_1 = 1, D_2 = 1, H = 1)$$

$$= P(D_1 = 1, D_2 = 1 | H = 0)P(H = 0) + P(D_1 = 1, D_2 = 1 | H = 1)P(H = 1)$$

$$= 0.00176955.$$

أخيراً، فإن احتمال إصابة المريض بفيروس نقص المناعة البشرية مع إعطاء كلا الاختبارين إيجابياً

$$P(H = 1 | D_1 = 1, D_2 = 1) = \frac{P(D_1 = 1, D_2 = 1 | H = 1)P(H = 1)}{P(D_1 = 1, D_2 = 1)}$$

$$= 0.8307.$$

أي أن الاختبار الثاني سمح لنا باكتساب ثقة أعلى بكثير من أن ليس كل شيء على ما يرام. على الرغم من أن الاختبار الثاني أقل دقة بكثير من الاختبار الأول، إلا أنه لا يزال يحسن تقديرنا بشكل كبير. كان افتراض أن كلا الاختبارين مستقلين عن بعضهما البعض مشووطاً أمراً حاسماً لقدرتنا على إنشاء تقدير أكثر دقة. خذ الحالة القصوى حيث نجري نفس الاختبار مرتين. في هذه الحالة، نتوقع نفس النتيجة في كلتا المرتين، وبالتالي لا يتم اكتساب رؤية إضافية من إجراء نفس الاختبار مرة أخرى. ربما لاحظ القارئ الذي أن التشخيص تصرف كمصنف يختبر على مرأى من الجميع حيث تزيد قدرتنا على تقرير ما إذا كان المريض يتمتع بصحة جيدة مع حصولنا على المزيد من الميزات (نتائج الاختبار).

2.6.6 التوقعات Expectations

في كثير من الأحيان، لا يتطلب اتخاذ القرارات مجرد النظر إلى الاحتمالات المخصصة للأحداث الفردية، بل تجمعها معاً في مجاميع مفيدة يمكن أن تزودنا بالإرشادات. على سبيل المثال، عندما تأخذ المتغيرات العشوائية قيمًا عددياً مستمرة، فإننا غالباً ما نهتم بمعرفة القيمة التي نتوقعها في المتوسط. تسمى هذه الكمية رسمياً بالتوقع expectation. إذا كنا نقوم باستثمارات، فقد تكون الكمية الأولى من الفائدة هي العائد الذي يمكن أن نتوقعه، بمتوسط جميع النتائج المحتملة (والترجح حسب الاحتمالات المناسبة). على سبيل المثال، لنفترض أنه مع وجود احتمال بنسبة 50٪، قد يفشل الاستثمار تماماً، مع احتمال 40٪ أنه قد يوفر عائدًا $\times 2$ ، ومع احتمال 10٪ قد يوفر عائد $\times 10$. لحساب العائد المتوقع، نقوم بتجميع الكل العوائد، وضرب كل منها في احتمال حدوثها. هذا ينتج التوقع $1.8 = 10 \cdot 0.1 + 0.4 \cdot 2 + 0.5 \cdot 0$. ومن ثم فإن العائد المتوقع هو 1.8.

بشكل عام، يتم تعريف توقع (أو متوسط) المتغير العشوائي X على أنه

$$E[X] = E_{x \sim P}[x] = \sum_x xP(X=x).$$

وبالمثل، بالنسبة للكثافات التي نحصل عليها $E[X] = \int xdp(x)$. في بعض الأحيان نحن مهتمون بالقيمة المتوقعة لبعض دوال x . يمكننا حساب هذه التوقعات على أنها

$$E_{x \sim P}[f(x)] = \sum_x f(x)P(x) \text{ and } E_{x \sim P}[f(x)] = \int f(x)p(x)dx$$

للاحتمالات والكثافات المقطعة، على التوالي. بالعودة إلى مثال الاستثمار أعلاه، قد تكون المنفعة (السعادة) المرتبطة بالعائد. لاحظ الاقتصاديون السلوكيون منذ فترة طويلة أن الناس يربطون بين عدم الكفاءة وخسارة الأموال أكثر من المنفعة المكتسبة من كسب دولار واحد مقارنة بخط الأساس. علاوة على ذلك، تميل قيمة النقود إلى أن تكون دون خطية sublinear. يمكن لامتلاك 100 ألف دولار مقابل صفر دولار أن يحدث فرقاً بين دفع الإيجار وتناول الطعام الجيد والتتمتع بالرعاية الصحية الجيدة مقابل المعاناة من التشرد. من ناحية أخرى، فإن المكاسب الناتجة عن امتلاك 200 ألف مقابل 100 ألف أقل دراماتيكية. مثل هذا الاستدلال يحفل الكليشيهات القائلة بأن "منفعة المال لوغاريتمية". the utility of money is logarithmic .

إذا كانت المنفعة المرتبطة بخسارة إجمالية هي -1 ، وكانت المرافق المرتبطة بعائدات 1 و 2 و 10 هي 1 و 2 و 4 على التوالي، فإن السعادة المتوقعة للاستثمار ستكون $+0.4 \cdot (-1) + 0.5 \cdot 0 + 0.1 \cdot 2 + 0.4 \cdot 1$.

$0.1 \cdot 4 = 0.7$ (خسارة فائدة متوقعة قدرها 30%). إذا كانت هذه هي بالفعل دالة المفعة utility function الخاصة بك، فقد يكون من الأفضل لك الاحتفاظ بالمال في البنك.

بالنسبة للقرارات المالية، قد نرغب أيضًا في قياس مدى خطورة risky الاستثمار. هنا، لا نهتم فقط بالقيمة المتوقعة ولكن إلى أي مدى تمثل القيم الفعلية إلى الاختلاف بالنسبة إلى هذه القيمة. لاحظ أنه لا يمكننا أن نأخذ فقط توقع الفرق بين القيم الفعلية والمتوسطة. وذلك لأن توقع الاختلاف هو اختلاف التوقعات، وبالتالي $E[X - E[X]] = E[X] - E[E[X]] = 0$. ومع ذلك، يمكننا أن ننظر إلى توقع أي دالة غير سلبية لهذا الاختلاف. يتم حساب التباين في المتغير العشوائي من خلال النظر إلى القيمة المتوقعة للانحرافات التربيعية squared deviations:

$$\text{Var}[X] = E[(X - E[X])^2] = E[X^2] - E[X]^2.$$

هنا تأتي المساواة من خلال توسيع $(X - E[X])^2 = X^2 - 2XE[X] + E[X]^2$ وأخذ التوقعات لكل مصطلح. الجذر التربيعي للتباين variance هو كمية مفيدة أخرى تسمى الانحراف المعياري standard deviation. بينما ينقل التباين والانحراف المعياري نفس المعلومات (يمكن حساب أي منها من الآخر)، فإن الانحراف المعياري له خاصية لطيفة التي يتم التعبير عنها في نفس الوحدات مثل الكمية الأصلية التي يمثلها المتغير العشوائي.

أخيرًا، يتم تعريف تباين دالة المتغير العشوائي بشكل مشابه

$$\text{Var}_{x \sim P}[f(x)] = E_{x \sim P}[f^2(x)] - E_{x \sim P}[f(x)]^2.$$

بالعودة إلى مثالنا الاستثماري، يمكننا الآن حساب تباين الاستثمار. أعطيت $0.5 \cdot 0 + 0.4 \cdot 2^2 = 8.36 - 1.8^2 + 10^2 = 2.0$. لجميع المقاصد والأغراض، هذا استثمار محفوف بالمخاطر. لاحظ أنه من خلال الاستصلاح الرياضي، غالباً ما يشار إلى المتوسط والتباين ك μ و σ^2 . هذا شائع بشكل خاص عندما نستخدمه لتحديد توزيع غاوسي.

بنفس الطريقة التي قدمنا بها التوقعات والتباين للمتغيرات العشوائية العددية، يمكننا القيام بذلك للمتغيرات ذات القيمة المتجهة ones vector-valued ones. التوقعات سهلة، حيث يمكننا تطبيقها بطريقة أساسية. على سبيل المثال، $\mu \stackrel{\text{def}}{=} E_{\mathbf{x} \sim P}[\mathbf{x}]$ لديه إحداثيات $[x_i] = \mu_i = E_{\mathbf{x} \sim P}[x_i]$. التغيرات Covariances أكثر تعقيداً. نقوم بحل المشكلة بأخذ توقعات الناتج الخارجي للفرق بين المتغيرات العشوائية ومتواسطها.

$$\Sigma \stackrel{\text{def}}{=} \text{Cov}_{\mathbf{x} \sim P}[\mathbf{x}] = E_{\mathbf{x} \sim P}[(\mathbf{x} - \mu)(\mathbf{x} - \mu)^T].$$

يشار إلى هذه المصفوفة بمصفوفة التغيرات covariance matrix. طريقة سهلة لمعرفة تأثيرها هي النظر في بعض المتجهات \mathbf{v} بنفس حجم \mathbf{x} . إنه يتبع هذا

$$\mathbf{v}^\top \boldsymbol{\Sigma} \mathbf{v} = E_{\mathbf{x} \sim P} [\mathbf{v}^\top (\mathbf{x} - \boldsymbol{\mu})(\mathbf{x} - \boldsymbol{\mu})^\top \mathbf{v}] = \text{Var}_{\mathbf{x} \sim P} [\mathbf{v}^\top \mathbf{x}].$$

على هذا النحو، $\boldsymbol{\Sigma}$ يسمح لنا بحساب التباين لأي دالة خطية لـ \mathbf{x} من خلال ضرب مصفوفة بسيط. تخربنا العناصر خارج القطر عن مدى ارتباط الإحداثيات: القيمة 0 تعني عدم وجود ارتباط no correlation، حيث تعني القيمة الإيجابية الأكبر أنها أكثر ارتباطاً strongly correlated.

2.6.7 المناقشة

في التعلم الآلي، هناك العديد من الأشياء التي يجب عدم التأكيد بشأنها! يمكن أن تكون غير متأكدين من قيمة التسمية في ضوء المدخلات. يمكن أن تكون غير متأكدين من القيمة المقدرة للمعلمة. يمكننا حتى أن نكون غير متأكدين مما إذا كانت البيانات التي تصل إلى النشر هي حتى من نفس التوزيع مثل بيانات التدريب.

من خلال عدم اليقين المتكرر aleatoric uncertainty، نشير إلى أن عدم اليقين المتأصل في المشكلة، وبسبب العشوائية الحقيقية التي لا تحسّبها المتغيرات المرصودة. من خلال عدم اليقين المعرفي epistemic uncertainty، نشير إلى عدم اليقين بشأن معلمات النموذج، وهو نوع عدم اليقين الذي نأمل في تقليله من خلال جمع المزيد من البيانات. قد يكون لدينا حالة من عدم اليقين المعرفي فيما يتعلق باحتمالية ظهور عملة ما، ولكن حتى بمجرد أن نعرف هذا الاحتمال، فإننا نشك في عدم يقين دائم بشأن نتيجة أي رمية مستقبلية. بعض النظر عن المدة التي شاهد فيها شخصاً ما يرمي عملة عادلة، فلن تكون أبداً أكثر أو أقل من 50٪ على يقين من أن القرعة التالية ستظهر رأساً. تدين هذه المصطلحات إلى الأديبات في النمذجة الميكانيكية mechanical modeling (انظر على سبيل المثال، Der Kiureghian and Ditlevsen (2009) لمراجعة هذا الجانب من عدم اليقين الكمي uncertainty quantification). من الجدير بالذكر أن هذه المصطلحات تشكل إساعة طفيفة للغة. يشير المصطلح المعرفي إلى أي شيء يتعلق بالمعرفة knowledge، وبالتالي بالمعنى الفلسفى، فإن كل عدم اليقين هو معرفي epistemic.

لقد رأينا أنأخذ العينات للبيانات من توزيع احتمالي غير معروف يمكن أن يزودنا بمعلومات يمكن استخدامها لتقدير معلمات توزيع توليد البيانات. ومع ذلك، فإن المعدل الذي يمكن به هذا يمكن أن يكون بطبيعاً للغاية. في مثالنا لرمي العملات (والعديد من الأمثلة الأخرى) لا يمكننا أن نفعل أفضل من تصسيم المقدرات التي تتقارب بمعدل $\sqrt{n}/1$ حيث n حجم العينة (على سبيل المثال، عدد الرميات). هذا يعني أنه بالانتقال من 10 إلى 1000 ملاحظة (عادة ما تكون مهمة قابلة للتحقيق للغاية)، نرى انخفاضاً بمقدار عشرة أضعاف في عدم اليقين uncertainty في حين أن الملاحظات الألف التالية تساعد قليلاً نسبياً، حيث تقدم فقط تقليلاً بمقدار 1.41 مرة. هذه ميزة دائمة للتعلم الآلي: في حين أن هناك غالباً مكاسب سهلة، فإنها تتطلب قدرًا كبيراً جداً من البيانات، وغالباً ما تكون معها قدرًا هائلاً من الحسابات لتحقيق المزيد من المكاسب.

للحصول على مراجعة تجريبية لهذه الحقيقة لنماذج اللغة واسعة النطاق، انظر Revels et al (2016).

كما شحدنا لغتنا وأدواتنا للنمذجة الإحصائية. في هذه العملية، تعلمنا عن الاحتمالات الشرطية وحول واحدة من أهم المعادلات في الإحصاء – نظرية بايز. إنها أداة فعالة لفصل المعلومات التي تنقلها البيانات من خلال مصطلح الاحتمالية $P(B | A)$ الذي يتناول مدى مطابقة الملاحظات مع اختيار المعلومات A ، والاحتمال السابق $P(A)$ الذي يحكم مدى معقولة اختيار معين A كان في المقام الأول. على وجه الخصوص، رأينا كيف يمكن تطبيق هذه القاعدة لتعيين الاحتمالات للتشخيص، بناءً على فعالية الاختبار وانتشار المرض نفسه (أي سابقاً).

أخيراً، قدمنا مجموعة أولى من الأسئلة غير التافهة حول تأثير توزيع احتمالي محدد، وهي التوقعات expectations والبيانات variances. في حين أن هناك أكثر من مجرد توقعات خطية وتربيعية لتوزيع الاحتمالات، فإن هذين الأمرين يوفران بالفعل قدرًا جيدًا من المعرفة حول السلوك المحتمل للتوزيع. على سبيل المثال، تنص عدم المساواة Chebyshev's inequality على $P(|X - \mu| \geq k\sigma) \leq 1/k^2$ حيث μ هو التوقع، σ^2 هو تباين التوزيع، $k > 1$ وهو معيار ثقة من اختيارنا. يخبرنا أن السحب من التوزيع يمكنه في احتمال 50٪ على الأقل خلال $[\sqrt{2}\sigma, -\sqrt{2}\sigma]$ فترة تتمحور حول التوقع.

2.6.8 التمارين

- أعط مثلاً حيث يمكن أن تؤدي مراقبة المزيد من البيانات إلى تقليل مقدار عدم اليقين بشأن النتيجة إلى مستوى منخفض بشكل تعسفي.
- أعط مثلاً حيث ستؤدي مراقبة المزيد من البيانات إلى تقليل مقدار عدم اليقين فقط إلى حد ما ثم لا أكثر. اشرح سبب حدوث ذلك وأين تتوقع حدوث هذه النقطة.
- لقد أثبتنا تجريبياً التقارب مع متوسط رمي عملة معدنية. احسب التباين في تقدير الاحتمال الذي نراه بعد سحب n من العينات.
- كيف يتاسب التباين مع عدد الملاحظات number of observations؟
- استخدم عدم المساواة في Chebyshev للحد من الانحراف عن التوقع.
- كيف تتصل بنظرية الحد المركزي central limit theorem؟

2.7 Documentation

بينما لا يمكننا تقديم كل دالة وفته TensorFlow فردية (وقد تصبح المعلومات قديمة بسرعة)، توفر وثائق API والبرامج التعليمية والأمثلة الإضافية مثل هذا التوثيق documentation. يقدم TensorFlow API هذا القسم بعض الإرشادات حول كيفية استكشاف دواله.

2.7.1 Functions and Classes in a Module

من أجل معرفة الدوال والفئات التي يمكن استدعاؤها في وحدة نمطية، نستدعي الدالة `dir`. على سبيل المثال، يمكننا الاستعلام عن جميع الخصائص في الوحدة لتوليد أرقام عشوائية:

```
['Algorithm', 'Generator', '__builtins__', '__cached__',
 '__doc__', '__file__', '__loader__', '__name__',
 '__package__', '__path__', '__spec__', '__sys',
 'all_candidate_sampler', 'categorical',
 'create_rng_state', 'experimental',
 'fixed_unigram_candidate_sampler', 'gamma',
 'get_global_generator',
 'learned_unigram_candidate_sampler',
 'log_uniform_candidate_sampler', 'normal', 'poisson',
 'set_global_generator', 'set_seed', 'shuffle',
 'stateless_binomial', 'stateless_categorical',
 'stateless_gamma', 'stateless_normal',
 'stateless_parameterized_truncated_normal',
 'stateless_poisson', 'stateless_truncated_normal',
 'stateless_uniform', 'truncated_normal', 'uniform',
 'uniform_candidate_sampler']
```

بشكل عام، يمكننا تجاهل الدوال التي تبدأ وتنتهي بـ `_` (كائنات خاصة في بايثون) أو الدوال التي تبدأ بـ `_` (عادةً دوال داخلية). استناداً إلى أسماء الدوال أو السمات المتبقية، قد نخاطر بتخمين أن هذه الوحدة تقدم طرقاً مختلفة لتوليد أرقام عشوائية، بما في ذلكأخذ العينات من التوزيع المنتظم (`uniform`) والتوزيع الطبيعي (`normal`) والتوزيع متعدد الحدود (`multinomial`).

2.7.2 Specific Functions and Classes

لمزيد من الإرشادات المحددة حول كيفية استخدام دالة أو فئة معينة، يمكننا استدعاء دالة `help`. كمثال، دعنا نستكشف تعليمات الاستخدام لدالة موترات `ones`.

```

Help on function ones in module
tensorflow.python.ops.array_ops:

ones(shape, dtype=tf.float32, name=None)
    Creates a tensor with all elements set to one
(1) .

    See also tf.ones_like, tf.zeros, tf.fill,
tf.eye.

    This operation returns a tensor of type dtype
with shape shape and
    all elements set to one.

>>> tf.ones([3, 4], tf.int32)
<tf.Tensor: shape=(3, 4), dtype=int32, numpy=
array([[1, 1, 1, 1],
       [1, 1, 1, 1],
       [1, 1, 1, 1]], dtype=int32)>

Args:
    shape: A list of integers, a tuple of
integers, or
        a 1-D Tensor of type int32.
    dtype: Optional DType of an element in the
resulting Tensor. Default is
        tf.float32.
    name: Optional string. A name for the
operation.

Returns:
    A Tensor with all elements set to one (1).

```

من الوثائق documentation، يمكننا أن نرى أن الدالة one تنشئ موتراً جديداً بالشكل المحدد وتضبط جميع العناصر على القيمة 1. كلما أمكن، يجب عليك إجراء اختبار سريع لتأكيد التفسير الخاص بك:

```

tf.ones(4)
<tf.Tensor: shape=(4,), dtype=float32, numpy=array([1.,
1., 1., 1.], dtype=float32)>

```

في نوتبوك Jupyter، يمكننا استخدام `? لعرض المستند في نافذة أخرى. على سبيل المثال، help(list) سينشئ محتوى مطابقاً تقريباً لـ (list)، ويعرضه في نافذة متصفح جديدة.`

بالإضافة إلى ذلك، إذا استخدمنا علامتي استفهام، مثل `?list`، فسيتم أيضًا عرض كود بايثون الذي ينفذ الدالة.

يوفر التوثيق الرسمي الكثير من الأوصاف والأمثلة التي تتجاوز هذا الكتاب. ينصب تركيزنا على تغطية حالات الاستخدام المهمة التي ستتيح لك البدء بسرعة في المشكلات العملية، بدلاً من اكتمال التغطية. نشجعك أيضًا على دراسة الكود المصدري للمكتبات لمشاهدة أمثلة على تطبيقات عالية الجودة لکود الإنتاج. من خلال القيام بذلك، ستصبح مهندسًا أفضل بالإضافة إلى أن تصبح عالماً أفضل.

الثباتات العربية الخطية للاندبار

٣

3. الشبكات العصبية الخطية للانحدار

Networks for Regression

قبل أن نقلق بشأن جعل شبكتنا العصبية عميقه deep، سيكون من المفيد تفتيذ بعض الشبكات العصبية الضحلة shallow neural networks، والتي تتصل مدخلاتها مباشرة بالمخرجات. سيبثت هذا أهميته لعدة أسباب. أولاً، بدلًا من تشتت الانتباه بسبب البنى المعقدة، يمكننا التركيز على أساسيات تدريب الشبكة العصبية، بما في ذلك تحديد معلمات طبقة الإخراج، ومعالجة البيانات، وتحديد دالة الخطأ، وتدرير النموذج. ثانياً، تكون هذه الفتة من الشبكات الضحلة لتشمل مجموعة من النماذج الخطية، والتي تشمل العديد من الطرق الكلاسيكية للتنبؤ الإحصائي، بما في ذلك الانحدار الخطى linear regression وانحدار Softmax. يعد فهم هذه الأدوات الكلاسيكية أمراً محورياً لأنها تُستخدم على نطاق واسع في العديد من السياقات وستحتاج غالباً إلى استخدامها كخطوط أساسية عند تبرير استخدام عمارات مختلفه. سيركز هذا الفصل بشكل ضيق على الانحدار الخطى وسيعمل الفصل التالي على توسيع مجموعة النماذج لدينا من خلال تطوير شبكات عصبية خطية من أجل التصنيف classification.

3.1. الانحدار الخطى

تظهر مشاكل الانحدار Regression problems كلما أردنا توقع قيمة عدديه. تشمل الأمثلة الشائعة التنبؤ بالأسعار (للمنازل، والمخزون، وما إلى ذلك)، والتنبؤ بطول الإقامة (للمرضى في المستشفى)، والتنبؤ بالطلب (لمبيعات التجزئة)، من بين أمور أخرى لا حصر لها. ليست كل مشكلة تنبؤ مشكلة انحدار كلاسيكية. في وقت لاحق، سوف نقدم مشاكل التصنيف classification problems، حيث الهدف هو التنبؤ بالعضوية بين مجموعة من الفئات.

كمثال جار، لنفترض أننا نرغب في تقدير أسعار المنازل (بالدولار) بناءً على مساحتها (بالأقدام المربعة) والعمر (بالسنوات). لتطوير نموذج للتنبؤ بأسعار المنازل، نحتاج إلى الحصول على بيانات تتكون من المبيعات، بما في ذلك سعر البيع والمساحة والعمر لكل منزل. في مصطلحات التعلم الآلي، تسمى مجموعة البيانات dataset مجموعة بيانات التدريب training dataset أو مجموعة التدريب training set، ويسمى كل صف (يحتوي على البيانات المقابلة لعملية بيع واحدة) بمثال (أو نقطة بيانات data point ، مثل instance ، عينة sample). الشيء الذي نحاول توقعه (السعر) يسمى التسمية label (أو الهدف target). تسمى المتغيرات (العمر والمنطقة) التي تستند إليها التنبؤات الميزات (أو المتغيرات المشتركة covariates).

Basics 3.1.1

قد يكون الانحدار الخطى Linear regression هو الأسط وألأكثري شيوعاً بين الأدوات القياسية لمعالجة مشاكل الانحدار. يعود تاريخ الانحدار الخطى إلى فجر القرن التاسع عشر Gauss، 1809، Legendre، 1805، ويتدفق الانحدار الخطى من بعض الافتراضات البسيطة. أولاً، نفترض أن العلاقة بين الميزات x والهدف y خطية تقريباً، أي أنه يمكن التعبير عن المتوسط الشرطي $E[Y | X = \mathbf{x}]$ كمجموع وزني للسمات \mathbf{x} . يسمح هذا الإعداد بأن القيمة المستهدفة قد لا تزال تتحرف عن قيمتها المتوقعة بسبب ضوضاء المراقبة observation noise. بعد ذلك، يمكننا فرض افتراض أن أي ضوضاء من هذا القبيل حسن التصرف، بعد توزيع غاوسي. عادة، سنستخدم n للإشارة إلى عدد الأمثلة في مجموعة البيانات الخاصة بنا. نحن نستخدم النصوص الغوقة superscripts لتعداد العينات والأهداف، والنصوص التحتية subscripts لفهرسة الإحداثيات. بشكل ملموس، $\mathbf{x}^{(i)}$ يشير إلى العينة i th و $x_j^{(i)}$ يدل على إحداثياتها j th.

Model 3.1.1.1

يوجدي قلب كل حل نموذج يصف كيف يمكن تحويل الميزات إلى تقدير للهدف. يعني افتراض الخطية أنه يمكن التعبير عن القيمة المتوقعة للهدف target (السعر) كمجموع اوزان للسمات (المنطقة وال عمر) features:

$$\text{price} = w_{\text{area}} \cdot \text{area} + w_{\text{age}} \cdot \text{age} + b. \quad 3.1.1$$

هنا w_{area} و w_{age} تسمى الأوزان weights، وتسمى b التحيز bias (أو الإزاحة offset أو التقاطع intercept). تحدد الأوزان تأثير كل ميزة على تنبؤاتنا. يحدد التحيز قيمة التقدير عندما تكون جميع الميزات صفرية. على الرغم من أننا لن نرى أبداً أي منازل مبنية حديثاً بمساحة صفر بالضبط، إلا أننا ما زلنا بحاجة إلى التحيز لأنه يسمح لنا بالتعبير عن جميع الدوال الخطية لميزاتنا (مقابل تقييدنا بالخطوط التي تمر عبر الأصل). بالمعنى الدقيق للكلمة، عبارة عن تحويل أفيني لميزات الإدخال، والتي تتميز بتحويل خطى للميزات عبر مجموع الأوزان، جنباً إلى جنب مع الترجمة عبر التحيز الإضافي. بالنظر إلى مجموعة البيانات، فإن هدفنا هو اختيار الأوزان w والانحياز b ، في المتوسط، يجعل توقعات نموذجنا تتناسب مع الأسعار الحقيقية التي لوحظت في البيانات بأكبر قدر ممكن.

في التخصصات التي يكون من الشائع فيها التركيز علىمجموعات البيانات مع بعض الميزات فقط، والتعبير الواضح عن النماذج طويلة الشكل، كما في (3.1.1)، أمر شائع. في التعلم الآلي، نعمل عادةً معمجموعات البيانات عالية الأبعاد high-dimensional datasets، حيث يكون من الملائم استخدام تدوين الجبر الخطى المدمج. عندما تكون مدخلاتنا من d ميزات، يمكننا

تعيين فهرس لكل منها (بين 1 و d) والتعبير عن توقعاتنا \hat{y} (بشكل عام، يشير رمز "hat" إلى تقدير estimate)

$$\hat{y} = w_1x_1 + \cdots + w_dx_d + b. \quad 3.1.2$$

بتجميع كل الميزات في متوجه $\mathbf{x} \in \mathbb{R}^d$ وجميع الأوزان في متوجه $\mathbf{w} \in \mathbb{R}^d$ ، يمكننا التعبير عن نموذجنا بشكل مضغوط عبر الضرب النقطي بين \mathbf{w} و \mathbf{x} :

$$\hat{y} = \mathbf{w}^\top \mathbf{x} + b. \quad 3.1.3$$

في (3.1.3)، يتافق المتوجه \mathbf{x} مع ميزات مثال واحد. غالباً ما نجد أنه من الملائم الرجوع إلى ميزات مجموعة البيانات الكاملة للأمثلة n عبر مصفوفة التصميم $\mathbf{X} \in \mathbb{R}^{n \times d}$. هنا، \mathbf{X} يحتوي على صف واحد لكل مثال وعمود واحد لكل ميزة. لمجموعة من الميزات \mathbf{X} والتنبؤات $\hat{\mathbf{y}} \in \mathbb{R}^n$ يمكن التعبير عنها عبر حاصل ضرب المصفوفة-matrix–vector product المتوجه:

$$\hat{\mathbf{y}} = \mathbf{X}\mathbf{w} + b, \quad 3.1.4$$

حيث يتم تطبيق البث broadcasting (القسم 2.1.4) أثناء عملية الجمع. بالنظر إلى ميزات مجموعة بيانات التدريب \mathbf{X} والتسبييات المقابلة (المعروفه) \mathbf{y} ، فإن الهدف من الانحدار الخطى هو العثور على متوجه الوزن \mathbf{w} ومصطلح التحيز b الذي يعطى ميزات لمثال بيانات جديد مأخذ من نفس التوزيع مثل \mathbf{X} ، فإن تسمية المثال الجديد سوف (في التوقع) يتوقع بأقل خطأ.

حتى إذا كنا نعتقد أن أفضل نموذج للتنبؤ ب y معطى \mathbf{x} هو خطى، فإننا لا نتوقع العثور على مجموعة بيانات حقيقة من الأمثلة n حيث $y^{(i)} = \mathbf{w}^\top \mathbf{x}^{(i)} + b$ لكل $i \leq n$. على سبيل المثال، أيا كانت الأدوات التي نستخدمها لمراقبة الميزات \mathbf{X} والتسبييات \mathbf{y} قد تعاني من قدر ضئيل من أخطاء القياس measurement error. وبالتالي، حتى عندما تكون واثقين من أن العلاقة الأساسية خطية، فستقوم بتضمين مصطلح ضوضاء noise لتفسير مثل هذه الأخطاء.

قبل أن نتمكن من البحث عن أفضل المعلمات parameters (أو معلمات النموذج model)، سنحتاج إلى شيئين آخرين: (1) مقياس جودة لبعض النماذج المحددة؛ و (2) إجراء لتحديث النموذج لتحسين جودته.

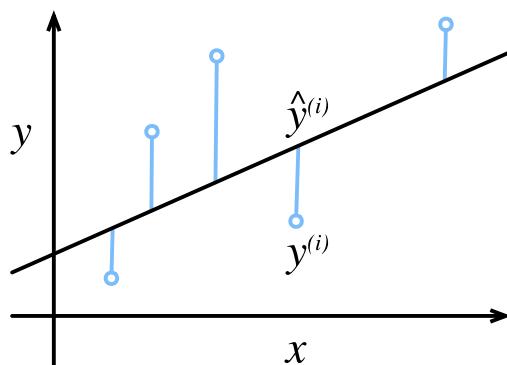
3.1.1.2 دالة الخطأ Loss Function

بطبيعة الحال، يتطلب ملاءمة fitting نموذجنا للبيانات أن نتفق على بعض مقاييس الملاءمة Loss functions (أو، على نحو مكافئ، عدم اللياقة unfitness). تحدد دوال الخطأ fitness

المسافة بين القيمة الحقيقة $y^{(i)}$ والمتوقعة $\hat{y}^{(i)}$ للهدف. ستكون الخسارة (الخطأ) عادةً رقمًا غير سالب حيث تكون القيمة الأصغر أفضل وتحمّل التنبؤات الكاملة خسارة قدرها 0. بالنسبة لمشاكل الانحدار، فإن دالة الخسارة الأكثر شيوعاً هي الخطأ التربيعي squared error عندما يكون توقعنا للحصول على مثال i هو $\hat{y}^{(i)}$ والتسمية الحقيقة المقابلة هي $y^{(i)}$ ، الخطأ التربيعي مُعطى بواسطة:

$$l^{(i)}(\mathbf{w}, b) = \frac{1}{2} (\hat{y}^{(i)} - y^{(i)})^2. \quad 3.1.5$$

لا يحدث الثابت $\frac{1}{2}$ فرقاً حقيقياً ولكنه يثبت أنه ملائم من الناحية المعيارية، لأنه يلغى عندما نأخذ مشتق الخطأ. نظراً لأن مجموعة بيانات التدريب تُمنح لنا، وبالتالي فهي خارجة عن سيطرتنا، فإن الخطأ التجاري ليس سوى دالة لمعلمات النموذج. أدناه، تخيل ملائمة نموذج الانحدار الخططي في مشكلة ذات مدخلات أحادية البعد (الشكل 3.1.1).



الشكل 3.1.1 ملائمة نموذج الانحدار الخططي لبيانات أحادية البعد.

لاحظ أن الفروق الكبيرة بين التقديرات $\hat{y}^{(i)}$ والأهداف $y^{(i)}$ تؤدي إلى مساهمات أكبر في الخسارة، بسبب الشكل التربيعي quadratic form للخسارة (يمكن أن يكون هذا سيفاً مزدوج الحافة). بينما يشجع النموذج على تجنب الأخطاء الكبيرة، فإنه يمكن أن يؤدي أيضاً إلى حساسية مفرطة للبيانات الشاذة anomalous data. لقياس جودة نموذج في مجموعة البيانات الكاملة لـ n من الأمثلة، نقوم ببساطة بمتوسط (أو بشكل مكافئ، جمع) الخسائر في مجموعة التدريب:

$$L(\mathbf{w}, b) = \frac{1}{n} \sum_{i=1}^n l^{(i)}(\mathbf{w}, b) = \frac{1}{n} \sum_{i=1}^n \frac{1}{2} (\mathbf{w}^\top \mathbf{x}^{(i)} + b - y^{(i)})^2.$$

عند تدريب النموذج، نزيد العثور على المعلمات (\mathbf{w}^*, b^*) التي تقلل الخسارة الإجمالية عبر جميع أمثلة التدريب:

$$\mathbf{w}^*, b^* = \underset{\mathbf{w}, b}{\operatorname{argmin}} L(\mathbf{w}, b).$$

3.1.1.3. الحل التحليلي Analytic Solution

على عكس معظم النماذج التي سنغطيها، يقدم لنا الانحدار الخطى مشكلة تحسين سهلة بشكل مدهش. على وجه الخصوص، يمكننا العثور على المعلمات المثلثى (كما تم تقييمها في بيانات التدريب) بشكل تحليلي من خلال تطبيق صيغة بسيطة على النحو التالي. أولاً، يمكننا تضمين التحيز b في المعلمة \mathbf{w} من خلال إلحاق عمود بمصفوفة التصميم المكونة من جميع العناصر. إذن مشكلة التنبؤ لدينا هي تقليل $\|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2$. طالما أن مصفوفة التصميم \mathbf{X} لها رتبة كاملة (لا توجد ميزة تعتمد خطياً على العناصر الأخرى)، عندئذ ستكون هناك نقطة حرجة واحدة فقط على سطح الخسارة وتوافق مع الحد الأدنى من الخسارة على المجال بأكمله. أخذ مشتق الخسارة فيما يتعلق بـ \mathbf{w} يجعله يساوي صفر عوائد:

$$\partial_{\mathbf{w}} \|\mathbf{y} - \mathbf{X}\mathbf{w}\|^2 = 2\mathbf{X}^T(\mathbf{X}\mathbf{w} - \mathbf{y}) = 0 \text{ and hence } \mathbf{X}^T\mathbf{y} = \mathbf{X}^T\mathbf{X}\mathbf{w}.$$

يوفر لنا حل \mathbf{w} الحل الأمثل لمشكلة التحسين optimization problem. لاحظ أن هذا الحل

$$\mathbf{w}^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$$

ستكون فريدة فقط عندما تكون المصفوفة $\mathbf{X}^T\mathbf{X}$ قابلة للعكس invertible، أي عندما تكون أعمدة مصفوفة التصميم مستقلة خطياً (Golub and Van Loan, 1996).

في حين أن المشكلات البسيطة مثل الانحدار الخطى قد تقبل حلولاً تحليلية، يجب ألا تعتاد على مثل هذا الحظ الجيد. على الرغم من أن الحلول التحليلية تسمح بتحليل رياضي لطيف، إلا أن متطلبات الحل التحليلي مقيدة للغاية بحيث تستبعد تقريباً جميع الجوانب المثيرة للتعلم العميق.

3.1.1.4. الانحدار التدرجي العشوائي ذو الدفعات الصغيرة Minibatch Stochastic Gradient Descent

لحسن الحظ، حتى في الحالات التي لا يمكننا فيها حل النماذج بشكل تحليلي، لا يزال بإمكاننا تدريب النماذج بفعالية في الممارسة العملية.علاوة على ذلك، بالنسبة للعديد من المهام، يتبيّن أن تلك النماذج التي يصعب تحسينها أفضل بكثير لدرجة أن اكتشاف كيفية تدريبها ينتهي به الأمر يستحق العناء.

تتمثل التقنية الرئيسية لتحسين أي نموذج تعلم عميق تقريباً، والتي سوف ندعوك إليها خلال هذا الكتاب، في تقليل الخطأ بشكل متكرر عن طريق تحديد المعلمات في الاتجاه الذي يقلل بشكل تدريجي من دالة الخسارة. هذه الخوارزمية تسمى التدرج الاشتتقافي (الانحدار الاشتتقافي) gradient descent.

يتمثل التطبيق الأكثر سذاجة للتدرج الاشتتقافي فيأخذ مشتق دالة الخسارة، وهو متوسط الخسائر المحسوبة في كل مثال فردي في مجموعة البيانات. من الناحية العملية، يمكن أن يكون هذا بطيئاً للغاية: يجب أن نمرر مجموعة البيانات بأكملها قبل إجراء تحديث واحد، حتى لو كانت خطوات التحديث قوية جداً (Liu and Nocedal, 1989). والأسوأ من ذلك، إذا كان هناك الكثير من التكرار في بيانات التدريب، فإن فائدة التحديث الكامل تكون أقل.

الطرف الآخر هو النظري في مثال واحد فقط في كل مرة واتخاذ خطوات التحديث بناءً على ملاحظة واحدة في كل مرة. يمكن أن تكون الخوارزمية الناتجة، التدرج الاشتتقافي العشوائي stochastic gradient descent (SGD) استراتيجية فعالة (Bottou, 2010)، حتى بالنسبة لمجموعات البيانات الكبيرة. لسوء الحظ، فإن SGD لها عيوب، حسابية وإحصائية. تنشأ مشكلة واحدة منحقيقة أن المعالجات تتضمن وتضيف الأرقام أسرع بكثير مما هي عليه في نقل البيانات من الذاكرة الرئيسية إلى ذاكرة التخزين المؤقت للمعالج. الأمر الذي يصل إلى مرتبة المقدار order of magnitude هو أكثر كفاءة لإجراء عملية ضرب المصفوفة – المتوجه مقارنة بعدد مماثل من عمليات ضرب المتوجه – المتوجه. هذا يعني أن معالجة عينة واحدة في كل مرة قد تستغرق وقتاً أطول بكثير مقارنة بالدفعة الكاملة. المشكلة الثانية هي أن بعض الطبقات، مثل تسوية الدفعات batch normalization (سيتم وصفها في القسم 8.5)، تعمل بشكل جيد فقط عندما يكون لدينا وصول إلى أكثر من ملاحظة واحدة في كل مرة.

الحل لكلا المشكلتين هو اختيار استراتيجية وسيطة: بدلاً منأخذ دفعة كاملة أو عينة واحدة فقط في كل مرة، نأخذ دفعة صغيرة minibatch من الملاحظات (Li et al., 2014). يعتمد الاختيار المحدد لحجم الدفعة الصغيرة المذكور على العديد من العوامل، مثل مقدار الذاكرة وعدد المعجلات واختيار الطبقات وإجمالي حجم مجموعة البيانات. على الرغم من كل ذلك، فإن الرقم بين 32 و256، ويفضل أن يكون مضاعفاً لقوة كبيرة، يعد بداية جيدة. هذا يقودنا إلى الانحدار التدريجي العشوائي ذو الدفعات الصغيرة minibatch stochastic gradient descent.

في أبسط أشكالها، في كل تكرار t ، نقوم أولاً بأخذ عينة عشوائية من الدفعات الصغيرة \mathcal{B}_t يتكون من عدد ثابت $|\mathcal{B}|$ من أمثلة التدريب. نقوم بعد ذلك بحساب مشتق (الدرج او الانحدار gradient) لمتوسط الخسارة على minibatch فيما يتعلق بمعلمات النموذج. أخيراً، نضرب

التدريج في قيمة موجبة صغيرة محددة مسبقاً η ، تسمى معدل التعلم learning rate ، ونطرح المصطلح الناتج من قيم المعلمة الحالية. يمكننا التعبير عن التحديث على النحو التالي:

$$(\mathbf{w}, b) \leftarrow (\mathbf{w}, b) - \frac{\eta}{|\mathcal{B}|} \sum_{i \in \mathcal{B}_t} \partial_{(\mathbf{w}, b)} l^{(i)}(\mathbf{w}, b).$$

باختصار، يتم تنفيذ minibatch SGD على النحو التالي: (1) تهيئة قيم معلمات النموذج، عادةً بشكل عشوائي؛ (2) أخذ عينات متكررة من الدفعات الصغيرة العشوائية من البيانات، وتحديث المعلمات في اتجاه التدرج السالب. بالنسبة للخسائر التربيعية والتحولات المرتبطة، فإن هذا له توسيع مغلق الشكل:

$$\begin{aligned} \mathbf{w} &\leftarrow \mathbf{w} - \frac{\eta}{|\mathcal{B}|} \sum_{i \in \mathcal{B}_t} \partial_{\mathbf{w}} l^{(i)}(\mathbf{w}, b) &= \mathbf{w} - \frac{\eta}{|\mathcal{B}|} \sum_{i \in \mathcal{B}_t} \mathbf{x}^{(i)} (\mathbf{w}^T \mathbf{x}^{(i)} + b - y^{(i)}) \\ b &\leftarrow b - \frac{\eta}{|\mathcal{B}|} \sum_{i \in \mathcal{B}_t} \partial_b l^{(i)}(\mathbf{w}, b) &= b - \frac{\eta}{|\mathcal{B}|} \sum_{i \in \mathcal{B}_t} (\mathbf{w}^T \mathbf{x}^{(i)} + b - y^{(i)}). \end{aligned}$$

نظرًا لأننا نختار دفعات صغيرة \mathcal{B} ، فنحن بحاجة إلى التسوية normalize من خلال حجمه $|\mathcal{B}|$. في كثير من الأحيان يتم تحديد حجم الدفعات الصغيرة ومعدل التعلم من قبل المستخدم. تسمى هذه المعلمات القابلة للضبط التي لم يتم تحديثها في حلقة التدريب بالمعلمات الفائقة hyperparameters. يمكن ضبطها تلقائياً من خلال عدد من التقنيات، مثل تحسين بايزري Bayesian optimization (Frazier, 2018) في النهاية، يتم تقييم جودة الحل عادةً على مجموعة بيانات تحقق منفصلة validation dataset (أو مجموعة التحقق من الصحة validation set).

بعد التدريب على عدد محدد مسبقاً من التكرارات iterations (أو حتى يتم استيفاء معيار إيقاف آخر)، نسجل معلمات النموذج المقدرة، المشار إليها $\hat{\mathbf{w}}, \hat{b}$. لاحظ أنه حتى لو كانت دالتنا خطية حقاً وبدون ضوابط، فلن تكون هذه المعلمات هي الحد الأدنى الدقيق للخسارة، أو حتى حتمية. على الرغم من أن الخوارزمية تتقارب ببطء نحو المصغرات minimizers، إلا أنها لا تستطيع تحقيقها بالضبط في عدد محدود من الخطوات. علاوة على ذلك، يتم اختيار الدفعات الصغيرة \mathcal{B} المستخدمة لتحديث المعلمات عشوائياً. هذا يكسر الحتمية determinism.

يحدث الانحدار الخططي ليكون مشكلة تعليمية بحد أدنى عالمي (كلما كانت \mathbf{X} ذات رتبة كاملة، أو ما يعادلها، كلما كان $\mathbf{X}^T \mathbf{X}$ قابلاً للعكس). ومع ذلك، فإن الأسطح الخاسرة للشبكات العميقية تحتوي على العديد من نقاط السرج saddle points والحد الأدنى minima. لحسن الحظ، لا نهتم عادةً بإيجاد مجموعة محددة من المعلمات ولكن فقط أي مجموعة من المعلمات تؤدي إلى تنبؤات دقيقة (وبالتالي خطأ منخفض). من الناحية العملية، نادرًا ما يكافح ممارسو التعلم العميق للعثور على معايير تقلل من الخطأ مجموعات التدريب Frankle and Carbin (2018).

(Izmailov et al., 2018). المهمة الأكثر صعوبة هي العثور على المعلومات التي تؤدي إلى تنبؤات دقيقة حول البيانات غير المرئية من قبل unseen data، وهو تحد يسمى التعميم generalization. نعود إلى هذه المواضيع في جميع أنحاء الكتاب.

3.1.1.5. Predictions التنبؤات

بالنظر إلى النموذج $\hat{y} = \hat{w}^T x + \hat{b}$ ، يمكننا الآن عمل تنبؤات لمثال جديد، على سبيل المثال، للتنبؤ بسعر بيع منزل لم يسبق رؤيته في ضوء مساحته وعمره. اعتاد ممارسو التعلم العميق على استدعاء inference مرحلة التنبؤ ولكن هذا نوع من التسمية الخاطئة misnomer – يشير الاستدلال على نطاق واسع إلى أي استنتاج تم التوصل إليه على أساس الأدلة، بما في ذلك قيم المعلومات والتسمية المحتملة لممثل غير مرئي. إذا كان هناك أي شيء، في كثير من الأحيان يشير الاستدلال في الأدبيات الإحصائية إلى استدلال المعلومات وهذا التحميل الزائد للمصطلحات يخلق ارتباكاً غير ضروري عندما يتحدث ممارسو التعلم العميق إلى الإحصائيين. فيما يلي سوف نتمسك بالتنبؤ كلما أمكن ذلك.

3.1.2. التوجيه من أجل السرعة Vectorization for Speed

عند تدريب نماذجنا، نريد عادةً معالجةمجموعات صغيرة كاملاً من الأمثلة في وقت واحد. يتطلب القيام بذلك بكفاءة أن نقوم بتوجيه vectorize الحسابات والاستفادة من مكتبات الجبر الخطية السريعة بدلاً من كتابة الحلقات المكلفة في بايثون.

```
%matplotlib inline
import math
import time
import numpy as np
import tensorflow as tf
from d2l import tensorflow as d2l
```

لتوضيح سبب أهمية هذا كثيراً، يمكننا التفكير في طريقتين لإضافة المتجهات. للبدء، نقوم بإنشاء مثيل لمتجهين كل منهما 10,000 بعد يحتويان على جميع المتجهات. في إحدى الطرق، نقوم بعمل حلقة فوق المتجهات باستخدام Python for-loop. في الطريقة الأخرى، نعتمد على استدعاء واحد لـ`+`.

```
n = 10000
a = tf.ones(n)
b = tf.ones(n)
for i in range(n):
    a[i] += b[i]
```

الآن يمكننا قياس أعباء العمل. أولاً، نضيفهم، إحداثي واحد في كل مرة، باستخدام حلقة for-loop.

```
c = tf.Variable(tf.zeros(n))
t = time.time()
for i in range(n):
    c[i].assign(a[i] + b[i])
f'{time.time() - t:.5f} sec'
'9.45401 sec'
```

بدلاً من ذلك، نعتمد على العامل $+ \mu$ المعاد تحميله لحساب مجموع العناصر.

```
t = time.time()
d = a + b
f'{time.time() - t:.5f} sec'
'0.00031 sec'
```

الطريقة الثانية أسرع بشكل كبير من الطريقة الأولى. غالباً ما ينبع عن التعليمات البرمجية الموجهة Vectorizing code تسريع من حيث الحجم. علاوة على ذلك، نقوم بدفع المزيد من الرياضيات إلى المكتبة دون الحاجة إلى كتابة أكبر عدد من العمليات الحسابية بأنفسنا، مما يقلل من احتمالية حدوث أخطاء ويزيّد من إمكانية نقل الشفرة.

3.1.3. التوزيع الطبيعي ومربيع الخطأ Squared Loss

لقد قدمنا حتى الآن حافراً وظيفياً إلى حد ما لهدف الخطأ التربيعي: تعيد المعلمات المثلثي التوقع الشرطي $E[Y | X]$ عندما يكون النمط الأساسي خطياً حقاً، وتعين الخطأ عقوبات كبيرة للقيم المتطرفة outliers. يمكننا أيضاً توفير دافع رسمي أكثر لهدف الخطأ التربيعي من خلال وضع افتراضات احتمالية حول توزيع الضوابط.

تم اختيار الانحدار الخطى في مطلع القرن التاسع عشر. على الرغم من أنه قد نوقش منذ فترة طويلة ما إذا كان Gauss أو Legendre قد فكر في الفكرة لأول مرة، إلا أن Gauss هو الذي اكتشف أيضاً التوزيع الطبيعي normal distribution (يسمى أيضاً Gaussian). اتضح أن التوزيع الطبيعي والانحدار الخطى مع الخطأ التربيعي يشتراكان في اتصال أعمق من النسب الشائعة.

للبدء، تذكر أنه يتم إعطاء التوزيع الطبيعي بمتوسط μ وتبان σ^2 (الانحراف المعياري σ)

$$p(x) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(x - \mu)^2\right).$$

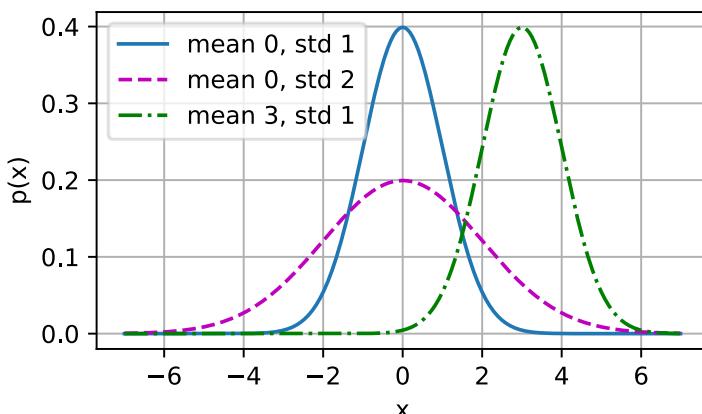
نحدد أدناه دالة لحساب التوزيع الطبيعي.

```
def normal(x, mu, sigma):
    p = 1 / math.sqrt(2 * math.pi * sigma**2)
    return p * np.exp(-0.5 * (x - mu)**2 / sigma**2)
```

يمكنا الآن رسم التوزيعات الطبيعية.

```
# Use numpy again for visualization
x = np.arange(-7, 7, 0.01)

# Mean and standard deviation pairs
params = [(0, 1), (0, 2), (3, 1)]
d2l.plot(x, [normal(x, mu, sigma) for mu, sigma in params], xlabel='x',
          ylabel='p(x)', figsize=(4.5, 2.5),
          legend=[f'mean {mu}, std {sigma}' for mu, sigma in params])
```



لاحظ أن تغيير المتوسط يتواافق مع التحول على طول المحور x ، وزيادة التباين يؤدي إلى انتشار التوزيع، مما يقلل من ذروته.

تتمثل إحدى طرق تحفيز الانحدار الخطى بخطأ مربع في افتراض أن الملاحظات تنشأ من القياسات الصادقة، حيث يتم توزيع الضوابط عادةً على النحو التالي:

$$y = \mathbf{w}^T \mathbf{x} + b + \epsilon \text{ where } \epsilon \sim \mathcal{N}(0, \sigma^2).$$

وبالتالي، يمكننا الآن كتابة احتمالية likelihood رؤية y معين لـ \mathbf{x} عبر

$$P(y | \mathbf{x}) = \frac{1}{\sqrt{2\pi\sigma^2}} \exp\left(-\frac{1}{2\sigma^2}(y - \mathbf{w}^T \mathbf{x} - b)^2\right).$$

على هذا النحو، فإن الاحتمالية عوامل likelihood factorizes. وفقاً لمبدأ الحد الأقصى من الاحتمالية principle of maximum likelihood، فإن أفضل قيم المعلمات \mathbf{w} و b هي تلك التي تزيد من احتمالية مجموعة البيانات بأكملها:

$$P(\mathbf{y} | \mathbf{X}) = \prod_{i=1}^n p(y^{(i)} | \mathbf{x}^{(i)}).$$

تبعد المساواة equality منذ أن تم رسم جميع الأزواج $(\mathbf{x}^{(i)}, y^{(i)})$ بشكل مستقل عن بعضها البعض. تسمى التقديرات المختارة وفقاً لمبدأ الاحتمالية القصوى بمقدرات الاحتمالية القصوى maximum likelihood estimators. في حين أن تكبير ناتج العديد من الدوال الأساسية قد يبدو صعباً، يمكننا تبسيط الأشياء بشكل كبير، دون تغيير الهدف، من خلال تعظيم لوغاريتم الاحتمال بدلاً من ذلك. لأسباب تاريخية، غالباً ما يتم التعبير عن التحسينات على أنها تصغير minimization بدلاً من تكبير maximization. لذلك، بدون تغيير أي شيء، يمكننا تقليل احتمالية السجل السلبية negative log-likelihood، والتي يمكننا التعبير عنها على التحو التالي:

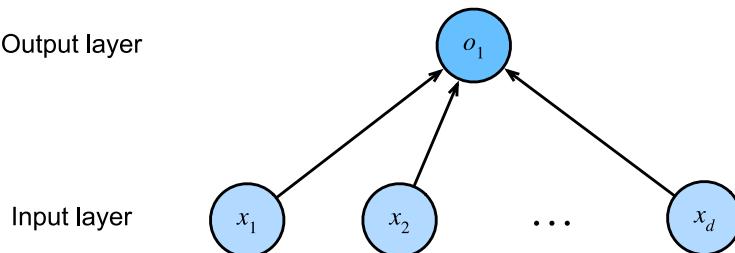
$$-\log P(\mathbf{y} | \mathbf{X}) = \sum_{i=1}^n \frac{1}{2} \log (2\pi\sigma^2) + \frac{1}{2\sigma^2} (y^{(i)} - \mathbf{w}^\top \mathbf{x}^{(i)} - b)^2.$$

إذا افترضنا أن σ تم إصلاحه، فيمكننا تجاهل المصطلح الأول، لأنه لا يعتمد على \mathbf{w} أو b . المصطلح الثاني مطابق لخطأ الخطأ التربيعي التي تم تقديمها سابقاً، باستثناء ثابت الضرب $\frac{1}{\sigma^2}$. لحسن الحظ، الحل لا يعتمد على أي منهما. ويترتب على ذلك أن التقليل من متوسط الخطأ التربيعي يعادل الحد الأقصى لتقدير الاحتمالية لنموذج خطى في ظل افتراض الضوابط الغاويسية المضافة.

3.1.4 الانحدار الخطى كشبكة عصبية Network

في حين أن النماذج الخطية ليست غنية بما يكفى للتعبير عن العديد من الشبكات العصبية المعقدة التي سنقدمها في هذا الكتاب، فإن الشبكات العصبية neural networks غنية بما يكفى لتضمين النماذج الخطية كشبكات عصبية يتم فيها تمثيل كل ميزة بواسطة خلية عصبية، وكلها متصلة مباشرة إلى الإخراج.

الشكل 3.1.2 يصور الانحدار الخطى كشبكة عصبية. يبرز الرسم التخطيطي نمط الاتصال مثل كيفية توصيل كل مدخل بالمخرجات، ولكن ليس القيم المحددة المأخوذة بواسطة الأوزان أو التحيزات.

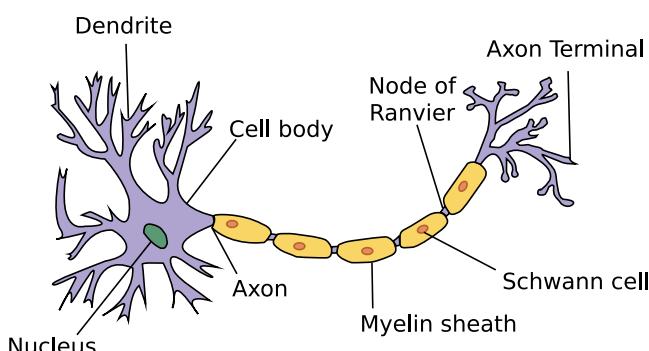


الشكل 3.1.2 الانحدار الخطي هو شبكة عصبية أحادية الطبقة.

المدخلات x_d, \dots, x_1 . نشير إلى عدد المدخلات number of inputs أو أبعاد الميزة feature dimensionality في طبقة الإدخال. ناتج الشبكة هو o_1 . نظراً لأننا نحاول فقط التنبؤ بقيمة عدديّة واحدة، فلدينا فقط خلية عصبية ناتجة واحدة. لاحظ أن جميع قيم الإدخال معطاة. هناك فقط خلية عصبية واحدة محسوبة. باختصار، يمكننا التفكير في الانحدار الخطي كشبكة عصبية أحادية الطبقة متصلة بالكامل single-layer fully connected neural network. سنواجه شبكات ذات طبقات أكثر بكثير في الفصول المستقبلية.

Biology 3.1.4.1

نظراً لأن الانحدار الخطي يسبّع علم الأعصاب الحسابي computational neuroscience، فقد يدوّن وصف الانحدار الخطي من حيث الشبكات العصبية مفارقة تاريخية. ومع ذلك، فقد كانت مكاناً طبيعياً للبدء عندما بدأ علماء علم الإنترن트 وعلماء الفسيولوجيا العصبية وارين ماكولوتش والتر بيتس في تطوير نماذج من الخلايا العصبية الاصطناعية. ضع في اعتبارك الصورة الكرتونية للخلايا العصبية البيولوجية في الشكل 3.1.3، والتي تتكون من التشعبات (أطراف الإدخال)، والنواة nucleus (وحدة المعالجة المركزية CPU)، والمotor العصبي axon (سلك الإخراج)، والمحطّات الطرفية axon terminals (أطراف الإخراج)، مما يتّبع التوصيلات بالخلايا العصبية الأخرى عبر المشابك synapses.



الشكل 3.1.3 الخلية العصبية (العصبون) الحقيقي.

يتم تلقي المعلومات x الواردة من الخلايا العصبية الأخرى (أو أجهزة الاستشعار البيئية environmental sensors) في التشعبات dendrites. على وجه الخصوص، يتم اخذ اوزان weighted هذه المعلومات من خلال الأوزان المشبكية w_i (synaptic weights)، وتحديد تأثير المدخلات، على سبيل المثال، التنشيط أو التشطير عبر المت俊ج $w_i x_i$. يتم تجميع المدخلات الموزونة الواردة من مصادر متعددة في النواة كمجموع اوزان $y = \sum_i x_i w_i + b$ ، وربما تخضع لبعض عمليات المعالجة اللاحقة غير الخطية عبر $(y)^{\sigma}$. ثم يتم إرسال هذه المعلومات عبر المحور العصبي axon إلى المحاور الطرفية axon terminals، حيث تصل إلى وجهتها على سبيل المثال، المشغل actuator مثل العضلات muscle) أو يتم تغذيتها في خلية عصبية أخرى عبر التشعبات dendrites.

من المؤكد أن الفكرة رفيعة المستوى القائلة بإمكانية دمج العديد من هذه الوحدات مع الاتصال الصحيح وخوارزمية التعلم الصحيحة، لإنتاج سلوك أكثر تعقيداً وإثارة للاهتمام من أي خلية عصبية واحدة وحدها يمكن أن تدين بدراسة لأنظمة العصبية البيولوجية الحقيقية. في الوقت نفسه، تستمد معظم الأبحاث في التعلم العميق الإلهام من مصدر أوسع بكثير. نستدعي ستิوارت راسل وبيتير نورفيج (راسل ونورفيج، 2016) اللذين أشاروا إلى أنه على الرغم من أن الطائرات ربما كانت مستوحاة من الطيور، إلا أن علم الطيور لم يكن المحرك الأساسي لابتكار الطيران لعدة قرون. وبالمثل، يأتي الإلهام في التعلم العميق هذه الأيام على قدم المساواة أو أكبر من الرياضيات واللغويات وعلم النفس والإحصاء وعلوم الكمبيوتر والعديد من المجالات الأخرى.

3.1.5 الملخص

في هذا القسم، قدمنا الانحدار الخطي التقليدي، حيث يتم اختيار معلمات الدالة الخطية لتقليل الخسارة التربيعية في مجموعة التدريب. لقد حفزنا أيضاً اختيار الهدف هذا من خلال بعض الاعتبارات العملية ومن خلال تفسير الانحدار الخطي باعتباره تقديرًا أقصى لاحتمالية في ظل افتراض الخطية والمضواب الغاوسي. بعد مناقشة كل من الاعتبارات الحسابية والارتباطات بالإحصاءات، أظهرنا كيف يمكن التعبير عن هذه النماذج الخطية كشبكات عصبية بسيطة حيث يتم توصيل المدخلات مباشرة بالمخرجات (المخرجات). على الرغم من أنها ستنتقل قريباً إلى النماذج الخطية السابقة تماماً، إلا أنها كافية لتقديم معظم المكونات التي تتطلبها جميع نماذجنا: النماذج البارامترية parametric forms، والأهداف القابلة للتفاضل differentiable objectives ، والتحسين عبر التدرج الاشتتقافي العشوائي المصغر ، وفي النهاية ، التقييم على البيانات غير المرئية سابقاً.

3.1.6 التمارين

- افتراض أن لدينا بعض البيانات $x_1, \dots, x_n \in \mathbb{R}$. هدفنا هو إيجاد ثابت b يتم تضييقه إلى أدنى حد. ابحث عن حل تحليلي لقيمة المثلث $|x_i - b|^2$.
- كيف ترتبط هذه المشكلة وحلها بالتوزيع الطبيعي؟
- ماذا لو غيرنا الخسارة من $(x_i - b)^2$ إلى $|\sum_i x_i - b|$? هل يمكنك العثور على الحل الأمثل لـ b ؟
- إثبت أن الدوال الأفينية التي يمكن التعبير عنها بواسطة $b + \mathbf{x}^T \mathbf{w}$ تعادل الدوال الخطية في $(\mathbf{x}, 1)$.
- افتراض أنك تريد إيجاد دوال تربعية لـ \mathbf{x} , أي $f(\mathbf{x}) = b + \sum_i w_i x_i + \sum_{i \leq j} w_{ij} x_i x_j$. كيف يمكنك صياغة هذافي شبكة عميقة؟
- تذكر أن أحد شروط حل مشكلة الانحدار الخطى هو أن مصفوفة التصميم $\mathbf{X}^T \mathbf{X}$ لها رتبة كاملة full rank.
- ماذا يحدث إذا لم يكن الأمر كذلك؟
- كيف يمكنك اصلاحها؟ ماذا يحدث إذا أضفت قدرًا صغيرًا من الضوضاء الغاويسية المستقلة ذات التنسيق الإحداثي إلى جميع إدخالات \mathbf{X} ؟
- ما هي القيمة المتوقعة لمصفوفة التصميم $\mathbf{X}^T \mathbf{X}$ في هذه الحالة؟
- ماذا يحدث مع التدرج الاستقافي العشوائي SGD عندما لا يكون له رتبة كاملة؟
- افتراض أن نموذج الضوضاء الذي يحكم الضوضاء المضافة ϵ هو التوزيع الأسوي. هذا هو، $p(\epsilon) = \frac{1}{2} \exp(-|\epsilon|)$
- اكتب الاحتمالية السلبية لسجل البيانات تحت النموذج negative log-likelihood $-\log P(\mathbf{y} | \mathbf{X})$.
- هل يمكنك إيجاد حل الشكل المغلق؟
- اقتراح خوارزمية التدرج الاستقافي العشوائي مع الدفعات الصغيرة لحل هذه المشكلة. ما الذي يمكن أن يحدث بشكل خاطئ (تلخيص: ماذا يحدث بالقرب من النقطة الثابتة بينما نستمر في تحديث المعلمات)? أيمكنك إصلاح هذا؟
- افتراض أننا نريد تصميم شبكة عصبية ذات طبقتين من خلال تكوين طبقتين خطيتين. أي أن إخراج الطبقة الأولى يصبح مدخلات الطبقة الثانية. لماذا مثل هذا التكوين الساذج لا يعمل؟
- ماذا يحدث إذا كنت تريدين استخدام الانحدار لتقدير واقعي لأسعار المنازل أو أسعار الأسهم؟

1. أظهر أن افتراض الضوابط الغاويسية المضافة غير مناسب. تلميح: هل يمكننا الحصول على أسعار سلبية؟ ماذا عن التقلبات fluctuations ؟
2. لماذا يكون الانحدار إلى لوغاريم السعر أفضل بكثير، أي $y = \log \text{ price}$ ؟
3. ما الذي يجب أن تقلق بشأنه عند التعامل مع pennystock ، أي الأسهم ذات الأسعار المنخفضة جداً؟ تلميح: هل يمكنك التداول بجميع الأسعار الممكنة؟ لماذا هذه مشكلة أكبر للأسهم الرخيصة؟
4. لمزيد من المعلومات، راجع نموذج Black-Scholes الشهير لتسعير الخيارات ([Black and Scholes, 1973](#)).
8. لنفترض أننا نريد استخدام الانحدار لتقدير عدد التفاح المباع في محل بقالة.

 1. ما هي المشاكل مع نموذج الضوابط الغاويسية المضافة؟ تلميح: أنت تبيع التفاح وليس الزيت.
 2. يلتفت توزيع بواسون Poisson distribution التوزيعات على العد. أعطيت من قبل $p(k | \lambda) = \frac{\lambda^k e^{-\lambda}}{k!}$ هي دالة المعدل و k هي عدد الأحداث التي تراها. إثبات أن λ هي القيمة المتوقعة للعد k .
 3. صمم دالة خطأ مرتبطة بتوزيع بواسون.
 4. صمم دالة خطأ لتقدير λ بدلاً من ذلك.

3.2 التصميم الكينوني للتنفيذ Implementation

في مقدمتنا للانحدار الخطي، مررنا عبر مكونات مختلفة بما في ذلك البيانات والنماذج ودالة الخطأ وخوارزمية التحسين. في الواقع، يعد الانحدار الخطي أحد أبسط نماذج التعلم الآلي. ومع ذلك، فإن التدريب عليه يستخدم العديد من نفس المكونات التي تتطلبها النماذج الأخرى في هذا الكتاب. لذلك، قبل الغوص في تفاصيل التنفيذ، من المفيد تصميم بعض واجهات برمجة التطبيقات المستخدمة في هذا الكتاب. عند التعامل مع المكونات في التعلم العميق على أنها كائنات، يمكننا البدء بتحديد فئات لهذه الكائنات وتفاعلاتها. سيؤدي هذا التصميم الموجه للكائنات للتنفيذ إلى تبسيط العرض التقديمي بشكل كبير وقد ترغبه في استخدامه في مشاريعك.

مستوحاة من مكتبات مفتوحة المصدر مثل PyTorch Lightning، على مستوى عالٍ، نرغب في الحصول على ثلات فئات: (1) تحتوي الوحدة النمطية على نماذج وخصائص وطرق تحسين؛ (2) توفر `DataModule` أدوات تحميل بيانات للتدريب والتحقق من الصحة؛ (3) يتم الجمع بين كلا الفتتين باستخدام فئة `Trainer`، مما يسمح لنا بتدريب النماذج على مجموعة متنوعة من منصات الأجهزة. تتكيف معظم التعليمات البرمجية في هذا الكتاب مع الوحدة النمطية

ووحدة البيانات `DataModule` فقط عندما ستنظر إلى فئة المدرب `Trainer`. ناقش وحدات معالجة الرسومات GPU ووحدات المعالجة المركزية CPU والتدريب الموازي وخوارزميات التحسين.

```
import time
import numpy as np
import tensorflow as tf
from d2l import torch as d2l
```

3.2.1. الأدوات المساعدة Utilities

نحتاج إلى بعض الأدوات المساعدة Utilities لتبسيط البرمجة الموجهة للكائنات-`object` في نوتبوكس Jupyter oriented programming. يتمثل أحد التحديات في أن تعريفات الفئات تميل إلى أن تكون كتل طويلة إلى حد ما من التعليمات البرمجية. تتطلب قابلية قراءة النوتبوك `Notebook` أجزاء قصيرة من التعليمات البرمجية، تخللها تفسيرات، وهو مطلب لا يتوافق مع أسلوب البرمجة الشائع في مكتبات بايثون. تتيح لنا دالة الأداة المساعدة الأولى تسجيل الدوال كطرق في الكلاس بعد إنشاء الكلاس. في الواقع، يمكننا القيام بذلك حتى بعد إنشاء نسخ من الكلاس! يسمح لنا بتقسيم تنفيذ كلاس إلى كتل تعليمات برمجية متعددة.

```
def add_to_class(Class):    #@save
    def wrapper(obj):
        setattr(Class, obj.__name__, obj)
    return wrapper
```

دعونا نلقي نظرة سريعة على كيفية استخدامه. نحن نخطط لتنفيذ كلاس A مع طريقة `do`. بدلاً من وجود رمز لكل من A والقيام به في نفس كتلة التعليمات البرمجية، يمكننا أولاً إعلان الكلاس .`a` (instance) A وإنشاء مثيل

```
class A:
    def __init__(self):
        self.b = 1
```

بعد ذلك نحدد الطريقة `do` كما نفعل عادة ، ولكن ليس في نطاق الكلاس A. بدلاً من ذلك ، نقوم بتزويين هذه الطريقة عن طريق طريق `add_to_class` بالفئة A ك وسيطة لها. عند القيام بذلك، تكون الطريقة قادرة على الوصول إلى متغيرات الأعضاء في A كما نتوقع إذا تم تعريفها كجزء من تعريف A. دعونا نرى ما يحدث عندما نستدعيها للمثيل a.

```
@add_to_class(A)
def do(self):
```

```
print('Class attribute "b" is', self.b)
```

```
a.do()
```

```
Class attribute "b" is 1
```

الثاني هو كلاس utility التي تحفظ جميع الوسائل في طريقة كلاس `__init__` كسمات للكلasse. هذا يسمح لنا بتوسيع توقع استدعاء المنشئ constructor ضمنياً دون الحاجة إلى تعليمات برمجية إضافية.

```
class HyperParameters: #@save
    def save_hyperparameters(self, ignore=[]):
        raise NotImplemented
```

نؤجل تنفيذه إلى القسم 20.7. لاستخدامها، نحدد صنفنا الذي يرث من `__init__` في طريقة `save_hyperparameters` ويستدعى `HyperParameters`

```
# Call the fully implemented HyperParameters class saved
# in d2l
class B(d2l.HyperParameters):
    def __init__(self, a, b, c):
        self.save_hyperparameters(ignore=['c'])
        print('self.a =', self.a, 'self.b =', self.b)
        print('There is no self.c =', not hasattr(self,
        'c'))
```

```
b = B(a=1, b=2, c=3)
```

```
self.a = 1 self.b = 2
```

```
There is no self.c = True
```

الأداة الأخيرة تسمح لنا برسم تقدم التجربة بشكل تفاعلي أثناء استمرارها. احتراماً للوحدة الأكثر قوة (والمعقدة)، نطلق عليها اسم `ProgressBoard`. تم تأجيل التنفيذ إلى القسم 20.7. في الوقت الحالي، دعنا نراه في العمل.

ترسم دالة `draw` نقطة (`y`, `x`) في الشكل ، مع تحديد `label` في وسيلة الإيضاح `legend`. يعمل الخيار `every_n` على تمهيد الخط من خلال إظهار النقاط $n/1$ في الشكل فقط. يتم حساب متوسط قيمها من نقاط الجوار n في الشكل الأصلي.

```
class ProgressBoard(d2l.HyperParameters): #@save
    """Plot data points in animation."""
    def __init__(self, xlabel=None, ylabel=None,
    xlim=None,
                ylim=None, xscale='linear',
    yscale='linear',
```

```

ls=['-', '--', '-.', ':'],
colors=['C0', 'C1', 'C2', 'C3'],
fig=None, axes=None, figsize=(3.5,
2.5), display=True):
    self.save_hyperparameters()

def draw(self, x, y, label, every_n=1):
    raise NotImplementedError

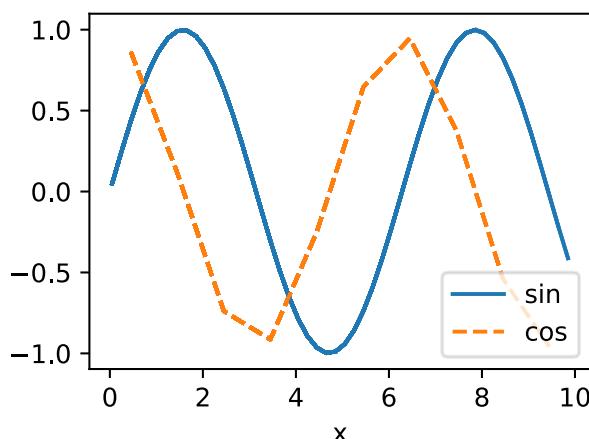
```

في المثال التالي، نرسم الجيب \sin وجيب التمام \cos بنعومة مختلفة. إذا قمت بتشغيل كتلة التعليمات البرمجية هذه، فسترى الخطوط تنمو في الرسوم المتحركة.

```

board = d2l.ProgressBar('x')
for x in np.arange(0, 10, 0.1):
    board.draw(x, np.sin(x), 'sin', every_n=2)
    board.draw(x, np.cos(x), 'cos', every_n=10)

```



3.2.2 النماذج Models

كلاس الوحدة النمطية `Module` هي الكلاس الأساسي لجميع النماذج التي سنتفذهها. كحد أدنى نحتاج إلى تحديد ثلاثة طرق. تقوم طريقة `__init__` بتخزين المعلمات القابلة للتعلم، وتقبل طريقة `training_step` مجموعة بيانات لإرجاع قيمة الخطأ، وتعيد طريقة `configure_optimizers` طريقة التحسين، أو قائمة بها، تُستخدم لتحديث المعلمات القابلة للتعلم. اختيارياً يمكننا تحديد `validation_step` للإبلاغ عن تدابير التقييم. في بعض الأحيان نضع الكود لحساب الإخراج في طريقة `forward` منفصلة لجعله أكثر قابلية لإعادة الاستخدام.

```

class Module(tf.keras.Model, d2l.HyperParameters):
    #@save

```

```
def __init__(self, plot_train_per_epoch=2,
plot_valid_per_epoch=1):
    super().__init__()
    self.save_hyperparameters()
    self.board = ProgressBoard()
    self.training = None

def loss(self, y_hat, y):
    raise NotImplementedError

def forward(self, X):
    assert hasattr(self, 'net'), 'Neural network is
defined'
    return self.net(X)

def call(self, X, *args, **kwargs):
    if kwargs and "training" in kwargs:
        self.training = kwargs['training']
    return self.forward(X, *args)

def plot(self, key, value, train):
    """Plot a point in animation."""
    assert hasattr(self, 'trainer'), 'Trainer is not
initialized'
    self.board.xlabel = 'epoch'
    if train:
        x = self.trainer.train_batch_idx / \
            self.trainer.num_train_batches
        n = self.trainer.num_train_batches / \
            self.plot_train_per_epoch
    else:
        x = self.trainer.epoch + 1
        n = self.trainer.num_val_batches / \
            self.plot_valid_per_epoch
    self.board.draw(x, value.numpy(), (
        'train_' if train else 'val_') + key,
every_n=int(n))

def training_step(self, batch):
    l = self.loss(*batch[:-1]), batch[-1])
    self.plot('loss', l, train=True)
    return l
```

```
def validation_step(self, batch):
    l = self.loss(*batch[:-1]), batch[-1])
    self.plot('loss', l, train=False)

def configure_optimizers(self):
    raise NotImplementedError
```

قد تلاحظ أن `Module` هي فئة فرعية من `tf.keras.Model` ، وهي الفئة الأساسية للشبكات العصبية في TensorFlow. يوفر ميزات ملائمة للتعامل مع الشبكات العصبية. على سبيل المثال، تستدعي طريقة `call` في طريقة `__call__` المضمنة. هنا نعيد توجيهه إلى دالة `forward` ، وحفظ وسائطها كسمة فئة. نقوم بذلك لجعل الكود الخاص بنا أكثر تشابهًا مع تطبيقات إطار العمل الأخرى.

3.2.3 بيانات Data

فئة `DataModule` هي الفئة الأساسية للبيانات. في كثير من الأحيان يتم استخدام طريقة `__init__` لإعداد البيانات. يتضمن ذلك التنزيل والمعالجة المسبقة إذا لزم الأمر. يُرجع أداة تحميل البيانات `train_dataloader` مجموعة بيانات التدريب. أداة تحميل البيانات هي منشئ `generator` (بايثون) ينتج دفعة بيانات في كل مرة يتم استخدامها. يتم بعد ذلك إدخال هذه الدفعتين في طريقة `training_step` للوحدة النمطية `Module` لحساب الخطأ. يوجد اختياري لإرجاع أداة تحميل مجموعة بيانات التحقق. يتصرف بنفس الطريقة، باستثناء أنه ينبع دفعات بيانات لأسلوب `validation_step` في `Module`.

```
class DataModule(d2l.HyperParameters):    #@save
    def __init__(self, root='../../data'):
        self.save_hyperparameters()

    def get_dataloader(self, train):
        raise NotImplementedError

    def train_dataloader(self):
        return self.get_dataloader(train=True)

    def val_dataloader(self):
        return self.get_dataloader(train=False)
```

3.2.4 التدريب Training

يقوم فئة `Trainer` بتدريب المعلمات القابلة للتعلم في فئة `Module` مع البيانات المحددة في `DataModule`. الطريقة الرئيسية `fit` ، والتي تقبل وسيطين: `model` ، مثيل للوحدة النمطية `data` ، والبيانات `Module` ، مثيل `DataModule`. ثم يتكرر على مدار مجموعة البيانات

عدد مرات لتدريب النموذج. كما في السابق، سوف نوجل تنفيذ هذه الدالة إلى فصول لاحقة.

```
class Trainer(d2l.HyperParameters): #@save
    def __init__(self, max_epochs, num_gpus=0,
    gradient_clip_val=0):
        self.save_hyperparameters()
        assert num_gpus == 0, 'No GPU support yet'

    def prepare_data(self, data):
        self.train_dataloader = data.train_dataloader()
        self.val_dataloader = data.val_dataloader()
        self.num_train_batches =
len(self.train_dataloader)
        self.num_val_batches = (len(self.val_dataloader)
                               if self.val_dataloader
is not None else 0)

    def prepare_model(self, model):
        model.trainer = self
        model.board.xlim = [0, self.max_epochs]
        self.model = model

    def fit(self, model, data):
        self.prepare_data(data)
        self.prepare_model(model)
        self.optim = model.configure_optimizers()
        self.epoch = 0
        self.train_batch_idx = 0
        self.val_batch_idx = 0
        for self.epoch in range(self.max_epochs):
            self.fit_epoch()

    def fit_epoch(self):
        raise NotImplementedError
```

3.2.5 الملخص

لتسلیط الضوء على التصمیم الموجه للكائنات لتنفيذ التعلم العميق في المستقبل، توضح الفئات المذکورة أعلاه فقط كيف تخزن كائنتها البيانات وتفاعل مع بعضها البعض. سنتعمّر في إثراء تطبيقات هذه الفئات، مثل `@add_to_class`، في بقية الكتاب. علاوة على ذلك، يتم حفظ هذه الفئات المنفذة بالكامل في مكتبة [d2l library](#) (`d2l` library)، وهي مجموعة أدوات خفيفة الوزن

تجعل النمذجة المنظمة للتعلم العميق أمرًا سهلاً. على وجه الخصوص، فإنه يسهل إعادة استخدام العديد من المكونات بين المشاريع دون تغيير الكثير على الإطلاق. على سبيل المثال، يمكننا استبدال المحسن `optimizer` فقط، والنموذج `model` فقط، ومجموعة البيانات `dataset` فقط، وما إلى ذلك؛ هذه الدرجة من النمطية `modularity` تؤتي ثمارها في جميع أنحاء الكتاب من حيث الإيجاز والبساطة (وهذا هو سبب إضافتها) ويمكنها أن تفعل الشيء نفسه لمشاريعك الخاصة.

3.2.6. التمارين

- حدد موقع عمليات التنفيذ الكاملة للفئات المذكورة أعلاه المحفوظة في مكتبة `d2l`.
نوصي بشدة أن تنظر إلى التنفيذ بالتفصيل بمجرد أن تكتسب مزيداً من الإلمام بنمذجة التعلم العميق.
- قم بإزالة عبارة `save_hyperparameters` في الفئة `B`. هل لا يزال بإمكانك طباعة `self.b` و `self.a`؟ اختياري: إذا كنت قد تعمقت في التنفيذ الكامل لفئة `HyperParameters` ، فهل يمكنك توضيح السبب؟

3.3. بيانات الانحدار التركيبية Synthetic Regression Data

يدور التعلم الآلي حول استخراج المعلومات من البيانات. لذلك قد تتساءل، ما الذي يمكن أن نتعلمه من البيانات التركيبية `synthetic data`؟ على الرغم من أنها قد لا نهتم بشكل جوهري بالأنيمات التي قمنا بتخزينها في نموذج إنشاء البيانات الاصطناعية، إلا أن مجموعات البيانات هذه مفيدة للأغراض التعليمية، مما يساعدنا على تقييم خصائص خوارزميات التعلم الخاصة بنا والتأكد من أن تطبيقاتنا تعمل كما هو متوقع. على سبيل المثال، إذا أنشأنا بيانات تُعرف بالمعلمات الصحيحة لها مسبقاً `priori`، فيمكننا التتحقق من أن نموذجنا يمكنه في الواقع استعادتها.

```
%matplotlib inline
import random
import tensorflow as tf
from d2l import tensorflow as d2l
```

3.3.1. إنشاء مجموعة البيانات Generating the Dataset

في هذا المثال، سنعمل الأبعاد المنخفضة `low-dimensional` للإيجاز. يُشَيَّع مقتطف الشفرة التالي 1000 مثال بميزات ثنائية الأبعاد مستمدة من التوزيع العادي القياسي. تتتمى مصفوفة التصميم \mathbf{X} الناتجة إلى $\mathbb{R}^{1000 \times 2}$. نقوم بإنشاء كل تسمية من خلال تطبيق دالة خطية للقيم الحقيقية `ground truth` ، وأفسدتها عن طريق الضوضاء المضافة ϵ ، مرسومة بشكل مستقل ومتطابق لكل مثال:

$$\mathbf{y} = \mathbf{X}\mathbf{w} + b + \epsilon.$$

للسهولة، نفترض أن ذلك ϵ مستمد من التوزيع الطبيعي بمتوسط $0 = \mu$ وانحراف معياري $\sigma = 0.01$. لاحظ أنه بالنسبة للتصميم الموجه للكائنات، نضيف الكود إلى طريقة `__init__` لفئة `d2l.DataModule` (تم تقديمها في القسم 3.2.3). من الممارسات الجيدة السماح بتعيين أي معلمات فائقة إضافية. نحقق ذلك باستخدام `.save_hyperparameters()`. س يتم تحديد حجم الدفعة `batch_size` لاحقاً.

```
class SyntheticRegressionData(d2l.DataModule): #@save
    def __init__(self, w, b, noise=0.01, num_train=1000,
                 num_val=1000,
                 batch_size=32):
        super().__init__()
        self.save_hyperparameters()
        n = num_train + num_val
        self.X = tf.random.normal((n, w.shape[0]))
        noise = tf.random.normal((n, 1)) * noise
        self.y = tf.matmul(self.X, tf.reshape(w, (-1,
                                                1))) + b + noise
```

أدناه، قمنا بتعيين المعلمات الحقيقة على $\mathbf{w} = [2, -3.4]^T$ و $b = 4.2$. لاحقاً، يمكننا التتحقق من معلماتنا المقدرة مقابل قيم الحقيقة الأساسية هذه.

```
data = SyntheticRegressionData(w=tf.constant([2, -3.4]),
                                b=4.2)
```

يتكون كل صفي في `features` من متوجه \mathbb{R}^2 وكل صفي في `labels` عبارة عن قيمة قياسية `scalar`. دعونا نلقي نظرة على الإدخال الأول.

```
print('features:', data.X[0], '\nlabel:', data.y[0])
features: tf.Tensor([-0.44379362  0.23580837],
shape=(2,), dtype=float32)
label: tf.Tensor([2.506998], shape=(1,), dtype=float32)
```

3.3.2 قراءة مجموعة البيانات

غالباً ما تتطلب نماذج التعلم الآلي للتدريب تمريرات متعددة على مجموعة بيانات `Dataset` مع الحصول على دفعة صغيرة `minibatch` واحدة من الأمثلة في كل مرة. ثم يتم استخدام هذه البيانات لتحديث النموذج. لتوضيح كيفية عمل ذلك، نقوم بتنفيذ دالة `get_dataloader`، وتسجيلها كطريقة في فئة `SyntheticRegressionData` عبر `add_to_class` (المقدمة في القسم 3.2.1). يأخذ حجم الدفعة، مصفوفة الميزات `matrix of features`، ومتوجه `a vector of labels` على هذا التسميات `batch_size`.

النحو، يتكون كل minibatch من مجموعة من الميزات والتسميات. لاحظ أننا بحاجة إلى أن نضع في اعتبارنا ما إذا كنا في وضع التدريب training أو التحقق من الصحة validation: في السابق، سترغب في قراءة البيانات بترتيب عشوائي، في حين أن القدرة على قراءة البيانات بترتيب محدد مسبقاً قد تكون مهمة لأغراض التصحيح.

```
@d2l.add_to_class(SyntheticRegressionData)
def get_dataloader(self, train):
    if train:
        indices = list(range(0, self.num_train))
        # The examples are read in random order
        random.shuffle(indices)
    else:
        indices = list(range(self.num_train,
self.num_train+self.num_val))
    for i in range(0, len(indices), self.batch_size):
        j = tf.constant(indices[i : i+self.batch_size])
        yield tf.gather(self.X, j), tf.gather(self.y, j)
```

لبناء بعض الحدس، دعنا نفحص الدفعـة الأولى من البيانات. توفر لنا كل دفعـة صغيرة من الميزات حجمها وأبعـاد ميزات الإدخـال. وبالـمثـل، سيكون لمجموعـة التسمـيات الصغـيرة المـاخصـة بـنا شـكـل .batch_size مـطـابـقـ تم تحـديـده بـواسـطـة .batch_size

```
X, y = next(iter(data.train_dataloader()))
print('X shape:', X.shape, '\ny shape:', y.shape)
X shape: (32, 2)
y shape: (32, 1)
```

بينما يبدو غير ضار innocuous فإن استدعاء iter(data.train_dataloader()) يوضح قوـة تصـمـيم باـيثـون المـوجـه لـلكـائـنـاتـ. لـاحـظـ أـنـاـ أـضـفـنـاـ طـرـيـقـةـ إـلـىـ فـتـةـ SyntheticRegressionDataـ بـعـدـ إـنـشـاءـ كـائـنـ الـبـيـانـاتـ. وـعـذـلـكـ،ـ فـإـنـ الـكـائـنـ يـسـتـفـيدـ مـنـ إـضـافـةـ الدـالـةـ بـأـثـرـ رـجـعـيـ إـلـىـ الـفـتـةـ.

خلال التكرار نحصل على دفعـات صـغـيرـةـ مـمـيـزةـ حتـىـ يتمـ استـنـفـادـ مـجـمـوعـةـ الـبـيـانـاتـ بالـكـامـلـ (ـجـربـ هـذـاـ).ـ فـيـ حـينـ أـنـ التـكـرـارـ الـذـيـ تمـ تـنـفـيـذـهـ أـعـلاـهـ مـفـيدـ لـلـأـغـرـاضـ الـتـعـلـيمـيـةـ،ـ إـلـاـ أـنـهـ غـيرـ فـعالـ فـيـ الـطـرـقـ الـتـيـ قدـ تـسـبـبـ لـنـاـ مـشـاكـلـ فـيـ مـشـاكـلـ حـقـيقـيـةـ.ـ عـلـىـ سـبـيلـ الـمـثـالـ،ـ يـتـطـلـبـ الـأـمـرـ أـنـ نـقـومـ بـتـحـمـيلـ جـمـيعـ الـبـيـانـاتـ الـمـوـجـودـةـ فـيـ الـذـاـكـرـةـ وـأـنـ نـقـومـ بـالـكـثـيرـ مـنـ الـوصـولـ الـعـشـوـائـيـ إـلـىـ الـذـاـكـرـةـ.ـ تـعـدـ أـجـهـزـةـ التـكـرـارـ الـمـضـمـنـةـ bu~lt-in iteratorsـ الـتـيـ يـتـمـ تـنـفـيـذـهـاـ فـيـ إـطـارـ عـمـلـ الـتـعـلـمـ الـعـمـيقـ أـكـثـرـ كـفـاعـةـ إـلـىـ حدـ كـبـيرـ وـيـمـكـنـهـ الـتـعـاملـ مـعـ مـصـادـرـ مـثـلـ الـبـيـانـاتـ الـمـخـزـنـةـ فـيـ الـمـلـفـاتـ،ـ وـالـبـيـانـاتـ

المستلمة عبر التدفق، والبيانات التي يتم إنشاؤها أو معالجتها أثناء التنقل. بعد ذلك، دعونا نحاول تنفيذ نفس الدالة باستخدام التكرارات المضمنة.

3.3.3. التنفيذ المختصر لمحمل البيانات

the Data Loader

بدلاً من كتابة المكرر الخاص بنا، يمكننا استدعاء واجهة برمجة التطبيقات API الموجودة في إطار عمل لتحميل البيانات. كما كان من قبل، نحتاج إلى مجموعة بيانات بالميزات X والتسميات y. علاوة على ذلك ، قمنا بتعيين `batch_size` في أداة تحميل البيانات المضمنة `tf.data.Dataset`. وندعها تهتم بأمثلة الخلط `shuffling examples` `built-in data loader`.

```
@d2l.add_to_class(d2l.DataModule) #@save
def get_tensorloader(self, tensors, train,
indices=slice(0, None)):
    tensors = tuple(a[indices] for a in tensors)
    shuffle_buffer = tensors[0].shape[0] if train else 1
    return
tf.data.Dataset.from_tensor_slices(tensors).shuffle(
    buffer_size=shuffle_buffer).batch(self.batch_size)

@d2l.add_to_class(SyntheticRegressionData) #@save
def get_dataloader(self, train):
    i = slice(0, self.num_train) if train else
    slice(self.num_train, None)
    return self.get_tensorloader((self.X, self.y),
train, i)
```

تعمل أداة تحميل البيانات الجديدة تماماً مثل السابقة، باستثناء أنها أكثر كفاءة ولديها بعض الدوال الإضافية.

```
X, y = next(iter(data.train_dataloader()))
print('X shape:', X.shape, '\ny shape:', y.shape)
X shape: (32, 2)
y shape: (32, 1)
```

على سبيل المثال، أداة تحميل البيانات التي توفرها واجهة برمجة التطبيقات API الخاصة بإطار العمل تدعم طريقة `len` المضمنة ، حتى نتمكن من الاستعلام عن طولها ، أي عدد الدفعات `.number of batches`.

```
len(data.train_dataloader())
```

32

3.3.4. الملخص

تُعد برامج تحميل البيانات Data loaders طريقة ملائمة لاستخراج عملية تحميل البيانات ومعالجتها. وبهذه الطريقة، فإن خوارزمية التعلم الآلي نفسها قادرة على معالجة العديد من أنواع ومصادر البيانات المختلفة دون الحاجة إلى تعديل. أحد الأشياء الرائعة حول برنامج تحميل البيانات هو أنه يمكن تكوينها. على سبيل المثال، قد نقوم بتحميل الصور ومن ثم يكون لدينا مرشح ما بعد المعالجة يقوم بقصها أو تعديلها بطريقة أخرى. على هذا النحو، يمكن استخدام برامج تحميل البيانات لوصف خط أنابيب معالجة البيانات بالكامل.

بالنسبة للنموذج نفسه، فإن النموذج الخطي ثنائي الأبعاد هو نموذج بسيط كما قد نواجهه. يتبع لنا اختبار دقة نماذج الانحدار دون القلق بشأن عدم وجود كميات كافية من البيانات أو نظام معادلات غير محدد بشكل كافٍ. سنستخدم هذا بشكل جيد في القسم التالي.

3.3.5. التمارين

1. ماذا سيحدث إذا كان عدد الأمثلة لا يمكن تقسيمه على حجم الدفعه. كيف يمكن تغيير هذا السلوك من خلال تحديد وسيلة مختلفة باستخدام واجهة برمجة التطبيقات الخاصة API بإطار العمل Framework؟

2. ماذا لو أردنا إنشاء مجموعة بيانات ضخمة، حيث يكون حجم متوجه المعلمة W وعدد الأمثلة $num_examples$ عدداً كبيراً؟

1. ماذا يحدث إذا لم نتمكن من الاحتفاظ بجميع البيانات في الذاكرة؟
 2. كيف يمكنك تبديل البيانات إذا تم الاحتفاظ بالبيانات على القرص؟ مهمتك هي تصميم خوارزمية فعالة لا تتطلب الكثير من القراءة أو الكتابة العشوائية. تلميح: مولدات التقليل العشوائية الزائفة pseudorandom permutation تسمح لك بتصميم التعديل دون الحاجة إلى تخزين جدول permutation table بشكل صريح (Naor and Reingold, 1999).

3. قم بتنفيذ منشئ البيانات data generator الذي ينتج بيانات جديدة بسرعة، في كل مرة يتم استدعاء المكرر.

4. كيف يمكنك تصميم منشئ بيانات عشوائي random data generator يقوم بإنشاء نفس البيانات في كل مرة يتم استدعاؤها؟

3.4. تنفيذ الانحدار الخطى من الصفر Implementation from Scratch

نحن الآن جاهزون للعمل من خلال التنفيذ الكامل للانحدار الخطى. في هذا القسم، سنقوم بتنفيذ الطريقة بأكملها من البداية، بما في ذلك (1) النموذج model؛ (2) دالة الخطأ loss function؛ (3) مُحسن التدرج الاستقaci العشوائى المصغر minibatch stochastic gradient descent optimizer؛ و (4) دالة التدريب training function التي تجمع كل هذه القطع معًا. أخيرًا، سنقوم بتشغيل منشى البيانات التركيبة الخاص بنا من القسم 3.3 ونطبق نموذجنا على مجموعة البيانات الناتجة. بينما يمكن لأطر التعلم العميق الحديثة أتمتة كل هذا العمل تقريبًا، فإن تنفيذ الأشياء من البداية هو الطريقة الوحيدة للتتأكد من أنك تعرف حقًا ما تفعله. علاوة على ذلك، عندما يحين وقت تخصيص النماذج، وتحديد طبقاتنا الخاصة أو دوال الخطأ، فإن فهم كيفية عمل الأشياء تحت الغطاء سيكون مفيدًا. في هذا القسم، سوف نعتمد فقط على الموررات والتفاضل التلقائي. في وقت لاحق، سنقدم تطبيقًا أكثر إيجازًا، مستفيدًا من أجراس وصفارات أطر التعلم العميق مع الاحتفاظ بهيكل ما يلي أدناه.

```
%matplotlib inline
import tensorflow as tf
from d2l import tensorflow as d2l
```

3.4.1. تعريف النموذج Defining the Model

قبل أن نبدأ في تحسين معلمات نموذجنا عن طريق SGD، نحتاج إلى بعض المعلمات parameters في المقام الأول. فيما يلي نقوم بتهيئة الأوزان عن طريق سحب أرقام عشوائية من التوزيع الطبيعي بمتوسط 0 وانحراف معياري 0.01 غالبًا ما يعمل الرقم السحري 0.01 جيدًا في الممارسة، ولكن يمكنك تحديد قيمة مختلفة من خلال وسيطة sigma. علاوة على ذلك، قمنا بتعيين الانحياز إلى 0. لاحظ أنه بالنسبة للتصميم الموجه للكائنات، نضيف الكود إلى طريقة `__init__` لفئة فرعية من وحدة `d2l.Module` (المقدمة في القسم 3.2.2).

```
class LinearRegressionScratch(d2l.Module): #@save
    def __init__(self, num_inputs, lr, sigma=0.01):
        super().__init__()
        self.save_hyperparameters()
        w = tf.random.normal((num_inputs, 1), mean=0,
                             stddev=0.01)
        b = tf.zeros(1)
        self.w = tf.Variable(w, trainable=True)
        self.b = tf.Variable(b, trainable=True)
```

بعد ذلك، يجب علينا تحديد نموذجنا، وربط مدخلاته ومعلماته بمخرجاته. بالنسبة لنموذجنا الخطى، نأخذ ببساطة منتج متوجه المصفوفة لميزات الإدخال \mathbf{X} وأوزان النموذج \mathbf{w} ، ونضيف الإزاحة(offset) b إلى كل مثال. $\mathbf{X}\mathbf{w} + b$ هو متوجه و b هو قيمة قياسية scalar. نظرًا لأن مكون (انظر القسم 2.1.4)، عندما نضيف متوجهًا وقيمة قياسية، تُضاف القيمة القياسية إلى كل مكون من مكونات المتوجه. يتم تسجيل دالة forward الناتجة كطريقة في فئة add_to_class عبر LinearRegressionScratch (تم تقديمها في القسم 3.2.1).

```
@d2l.add_to_class(LinearRegressionScratch) #@save
def forward(self, X):
    """The linear regression model."""
    return tf.matmul(X, self.w) + self.b
```

3.4.2. تعريف دالة الخطأ Defining the Loss Function

نظرًا لأن تحديث نموذجنا يتطلب أخذ التدرج gradient لدالة الخطأ loss function ، فيجب علينا تحديد دالة الخطأ أولًا. هنا نستخدم دالة الخطأ التربيعية squared loss function (3.1.5). في التنفيذ، نحتاج إلى تحويل القيمة الحقيقة y إلى شكل القيمة المتوقعة y_{hat} . النتيجة التي تم إرجاعها بواسطة الدالة التالية سيكون لها أيضًا نفس شكل y_{hat} . نعيد أيضًا قيمة الخطأ المتوسطة بين جميع الأمثلة minibatch.

```
@d2l.add_to_class(LinearRegressionScratch) #@save
def loss(self, y_hat, y):
    l = (y_hat - tf.reshape(y, y_hat.shape)) ** 2 / 2
    return tf.reduce_mean(l)
```

3.4.3. تعريف خوارزمية التحسين Algorithm

كما نوقش في القسم 3.1، فإن الانحدار الخطى له حل مغلق الشكل closed-form solution . ومع ذلك، فإن هدفنا هنا هو توضيح كيفية تدريب شبكات عصبية أكثر عمومية، وهذا يتطلب أن نعلمك كيفية استخدام SGD minibatch . ومن ثم سنتنهز هذه الفرصة لتقديم مثالك العملى الأول لـ SGD. في كل خطوة، باستخدام minibatch مأخذ عشوائياً من مجموعة البيانات الخاصة بنا، نقدر تدرج الخطأ فيما يتعلق بالمعلمات. بعد ذلك، تقوم بتحديث المعلمات في الاتجاه الذي قد يقلل من الخطأ.

ال코드 التالي يطبق التحديث، بالنظر إلى مجموعة من المعلمات، معدل التعلم α . نظرًا لأن خسارتنا يتم حسابها كمتوسط على minibatch ، لا نحتاج إلى ضبط معدل التعلم مقابل حجم الدفعه. في فصول لاحقة سنبحث في كيفية تعديل معدلات التعلم للدفعات الصغيرة minibatch

الكبيرة جداً عند ظهورها في التعلم الموزع على نطاق واسع. في الوقت الحالي، يمكننا تجاهل هذه التبعية.

نحدد فئة SGD الخاصة بنا، وهي فئة فرعية من d2l.HyperParameters (تم تقديمها في القسم 3.2.1)، للحصول على واجهة برمجة تطبيقات API مماثلة لمُحسن SGD المدمج. نقوم بتحديث المعلمات في طريقة apply_gradients. يقبل قائمة من المعلمات وأزواج التدرج.

```
class SGD(d2l.HyperParameters): #@save
    def __init__(self, lr):
        """Minibatch stochastic gradient descent."""
        self.save_hyperparameters()

    def apply_gradients(self, grads_and_vars):
        for grad, param in grads_and_vars:
            param.assign_sub(self.lr * grad)
.SGD بعد ذلك طريقة configure_optimizers ، والتي ترجع مثيلاً لفئة SGD
@d2l.add_to_class(LinearRegressionScratch) #@save
def configure_optimizers(self):
    return SGD(self.lr)
```

3.4.4. Training

الآن بعد أن أصبح لدينا جميع الأجزاء في مكانها (المعلمات، دالة الخسارة (الخط)، النموذج، والمُحسن)، نحن جاهزون لتنفيذ حلقة التدريب الرئيسية. من الأهمية بممكان أن تفهم هذا الرمز جيداً لأنك ستستخدم حلقات تدريب مماثلة لكل نموذج تعليم عميق آخر مغطى في هذا الكتاب. في كل فترة epoch، نكرر مجموعة بيانات التدريب بأكملها، ونمرر مرة واحدة في كل مثال (بافتراض أن عدد الأمثلة قابل للقسمة على حجم الدفعه). في كل تكرار، نحصل على دفعه صغيرة من أمثلة التدريب، ونحسب خسارتها من خلال طريقة training_step للنموذج. بعد ذلك، نحسب التدرجات فيما يتعلق بكل معلمة. أخيراً، سنقوم باستدعاء خوارزمية التحسين لتحديث معلمات النموذج. باختصار، سنقوم بتنفيذ الحلقة التالية:

- تهيئة المعلمات (\mathbf{w}, b)
- كرر حتى الانتهاء
- حساب التدرج $\mathbf{g} \leftarrow \partial_{(\mathbf{w}, b)} \frac{1}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} l(\mathbf{x}^{(i)}, y^{(i)}, \mathbf{w}, b)$
- تحديث المعلمات $(\mathbf{w}, b) \leftarrow (\mathbf{w}, b) - \eta \mathbf{g}$

تذكّر أن مجموعة بيانات الانحدار التركيبية synthetic regression dataset أنشأناها في القسم 3.3 لا توفر مجموعة بيانات للتحقّق validation dataset. ومع ذلك، في معظم الحالات، سنستخدم مجموعة بيانات تتحقّق لقياس جودة نموذجنا. هنا نمرر أدلة تحميل بيانات التحقّق مرة واحدة في كل فترة لقياس أداء النموذج. بعد تصميمنا الموجّه للકائنات، يتم تسجيل دالتي `fit_epoch` و `prepare_batch` كطريقتين من فئة `d2l.Trainer` (مقدمة في القسم 3.2.4).

```
@d2l.add_to_class(d2l.Trainer) #@save
def prepare_batch(self, batch):
    return batch

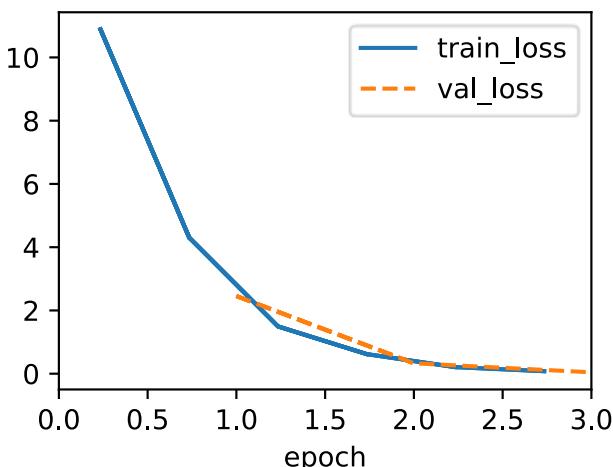
@d2l.add_to_class(d2l.Trainer) #@save
def fit_epoch(self):
    self.model.training = True
    for batch in self.train_dataloader:
        with tf.GradientTape() as tape:
            loss =
    self.model.training_step(self.prepare_batch(batch))
    grads = tape.gradient(loss,
    self.model.trainable_variables)
    if self.gradient_clip_val > 0:
        grads =
    self.clip_gradients(self.gradient_clip_val, grads)
    self.optim.apply_gradients(zip(grads,
    self.model.trainable_variables))
    self.train_batch_idx += 1
    if self.val_dataloader is None:
        return
    self.model.training = False
    for batch in self.val_dataloader:

    self.model.validation_step(self.prepare_batch(batch))
    self.val_batch_idx += 1
```

نحن جاهزون تقريباً لتدريب النموذج، لكننا نحتاج أولاً إلى بعض البيانات للتدريب عليها. هنا نستخدم فئة `SyntheticRegressionData` ونمرر بعض المعلومات الحقيقة-ground truth `lr=0.03` وتعيين `max_epochs=3`. لاحظ أنه بشكل عام، كل من عدد الفترات ومعدل التعلم عبارة عن معلمات فائقة hyperparameters. بشكل عام، يعد إعداد المعلمات الفائقة أمراً صعباً وسنرغب عادةً في استخدام تقسيم ثلاثي الاتجاهات 3-way split، ومجموعة واحدة للتدريب، وثانية لقسم

المعلمات الفائقة، والثالثة مخصصة للتقييم النهائي. نحن نتجاهل هذه التفاصيل في الوقت الحالي ولكننا سنراجعها لاحقاً.

```
model = LinearRegressionScratch(2, lr=0.03)
data = d2l.SyntheticRegressionData(w=tf.constant([2, -3.4]), b=4.2)
trainer = d2l.Trainer(max_epochs=3)
trainer.fit(model, data)
```



نظرًا لأننا قمنا بتجمیع مجموعة البيانات بأنفسنا، فإننا نعرف بالضبط ما هي المعلمات الحقيقة. وبالتالي، يمكننا تقييم نجاحنا في التدريب من خلال مقارنة المعلمات الحقيقة بتلك التي تعلمناها من خلال حلقة التدريب الخاصة بنا. في الواقع، تبيّن أنهم قريبون جدًا من بعضهم البعض.

```
print(f'error in estimating w: {data.w - tf.reshape(model.w, data.w.shape)}')
print(f'error in estimating b: {data.b - model.b}')
error in estimating w: [ 0.06905794 -0.15879321]
error in estimating b: [0.22819376]
```

لا ينبغي لنا أن نأخذ القدرة على استعادة المعلمات الحقيقة كأمر مسلم به. بشكل عام، بالنسبة للنمذجة العميق، لا توجد حلول فريدة للمعلمات، وحتى بالنسبة للنمذجة الخطية، يكون استرداد المعلمات بالضبط ممكناً فقط عندما لا تعتمد أي ميزة خطياً على الميزات الأخرى. ومع ذلك، في التعلم الآلي، غالباً ما نكون أقل اهتماماً باستعادة المعلمات الأساسية الحقيقة، وأكثر اهتماماً بالمعلمات التي تؤدي إلى تنبؤ عالي الدقة (Vapnik, 1992). لحسن الحظ، حتى في مشكلات التحسين الصعبة، غالباً ما يجد التدرج الاستقرائي العشوائي حلولاً جيدة بشكل ملحوظ، ويرجع

ذلك جزئياً إلى حقيقة أنه، بالنسبة للشبكات العميقه، توجد العديد من تكوينات المعلمات التي تؤدي إلى تنبؤ عالي الدقة.

3.4.5 الملخص

في هذا القسم، اخذنا خطوة مهمة نحو تصميم أنظمة التعلم العميق من خلال تنفيذ نموذج شبكة عصبية يعمل بكامل طاقته وحلقة تدريب في هذه العملية، قمنا ببناء أداة تحميل البيانات، ونموذج، ودالة الخطأ، وإجراء التحسين، وأداة الرسم والمراقبة. لقد فعلنا ذلك من خلال تكوين كائن بايثون يحتوي على جميع المكونات ذات الصلة لتدريب النموذج. على الرغم من أن هذا ليس تطبيقاً احترافياً بعد، إلا أنه يعمل بشكل مثالي ويمكن أن يساعدك رمز مثل هذا بالفعل في حل المشكلات الصغيرة بسرعة. في الأقسام التالية، سنرى كيفية القيام بذلك بشكل أكثر إيجازاً (تجنب الكود المعياري avoiding boilerplate code) وبخفاقة أكبر (استخدام وحدات معالجة الرسومات GPUs الخاصة بنا إلى أقصى إمكاناتها).

3.4.6 التمارين

1. ماذا سيحدث إذا قمنا بتهيئة الأوزان إلى الصفر. هل ستظل الخوارزمية تعمل؟ ماذا لو قمنا بتهيئة المعلمات مع التباين 1,000 بدلاً من 0.01 ؟

2. افترض أنك جورج سيمون أوتم تحاول ابتكر نموذج للمقاومات التي تربط الجهد والتيار. هل يمكنك استخدام التفاضل التلقائي automatic differentiation لمعرفة معلمات نموذجك؟

3. هل يمكنك استخدام قانون بلانك Planck's Law لتحديد درجة حرارة جسم ما باستخدام كثافة الطاقة الطيفية؟ كمراجع، فإن الكثافة الطيفية B للإشعاع المنبعث من

$$\text{جسم أسود هي } 1 - \left(\exp \frac{hc}{\lambda kT} \right)^{-1} \cdot \frac{2hc^2}{\lambda^5} \cdot \lambda \text{ هو الطول الموجي، } T \text{ هو درجة الحرارة، } c \text{ سرعة الضوء، } h \text{ كمية بلانك، } k \text{ ثابت بولتزمان. تقسيس الطاقة لأطوال موجية مختلفة وتحتاج الآن لمواهنة منحنى الكثافة الطيفية مع قانون بلانك.}$$

4. ما هي المشاكل التي قد تواجهها إذا أردت حساب المشتقات الثانية للخطأ؟ كيف تصلحهم؟

5. لماذا طريقة إعادة التشكيل reshape مطلوبة في دالة الخطأ loss function ؟

6. جرب استخدام معدلات تعلم مختلفة لمعرفة مدى سرعة انخفاض قيمة دالة الخطأ. هل يمكنك تقليل الخطأ عن طريق زيادة عدد فترات التدريب number of epochs of training ؟

7. إذا كان عدد الأمثلة لا يمكن تقسيمه على حجم الدفعه، فماذا يحدث ل data_iter في نهاية الفترة epoch ؟

8. حاول تنفيذ دالة خطأ مختلفة، مثل خطأ القيمة المطلقة absolute value loss .

$$(y_hat - d21.reshape(y, y_hat.shape)).abs().sum()$$
1. تحقق مما يحدث للبيانات العادية.
2. تتحقق مما إذا كان هناك اختلاف في السلوك إذا قمت بتشویش بعض إدخالات y مثل $y_5 = 10,000$.
3. هل يمكنك التفكير في حل رخيص للجمع بين أفضل جوانب الخسارة التربيعية وخسارة القيمة المطلقة؟ تلميح: كيف يمكنك تجنب قيم التدرج الكبيرة حقًا؟
9. لماذا نحتاج إلى تعديل مجموعة البيانات؟ هل يمكنك تصميم حالة تؤدي فيها مجموعة بيانات ضارة إلى كسر خوارزمية التحسين بطريقة أخرى؟

3.5 التنفيذ المختصر للانحدار الخطى Concise Implementation of Linear Regression

شهد التعلم العميق انفجاراً كامبيئياً Cambrian explosion من نوع ما خلال العقد الماضي. إن العدد الهائل من التقنيات والتطبيقات والخوارزميات يفوق إلى حد بعيد التقدم المحرز في العقود السابقة. ويرجع ذلك إلى مجموعة مصادفة من عدة عوامل، أحدها هو الأدوات المجانية القوية التي يقدمها عدد من أطر التعلم العميق مفتوحة المصدر. يمكن القول إن Theano (Jia et al., 2012) و Caffe (Bergstra et al., 2010) و DistBelief (Dean et al., 2012) يمثل الجيل الأول من هذه النماذج التي وجدت اعتماداً واسع النطاق. على عكس الأعمال السابقة (الأساسية) مثل Bottou (Simulateur Neuristique) Cun (Le Cun and Le Cun, 1988)، والتي قدمت تجربة برمجة تشبه Lisp ، توفر الأطر الحديثة تمييزاً تلقائياً وملاءمة بايثون. تسمح لنا هذه الأطر بأتمتة العمل المتكرر لتنفيذ خوارزميات التعلم القائم على التدرج وتصميمه.

في القسم 3.4، اعتمدنا فقط على (1) الموررات لتخزين البيانات والجبر الخطى؛ و (2) التفاضل التلقائي لحساب التدرجات. في الممارسة العملية، نظراً لأن مكررات البيانات ودوال الخطأ والمحسنات وطبقات الشبكة العصبية شائعة جداً، فإن المكتبات الحديثة تنفذ هذه المكونات لنا أيضاً. في هذا القسم، سنوضح لك كيفية تنفيذ نموذج الانحدار الخطى من القسم 3.4 بايجاز باستخدام واجهات برمجة التطبيقات API عالية المستوى لأطر التعلم العميق.

```
import numpy as np
import tensorflow as tf
from d2l import tensorflow as d2l
```

3.5.1. تعريف النموذج Defining the Model

عندما قمنا بتنفيذ الانحدار الخطى من البداية في القسم 3.4، قمنا بتعريف معلمات نموذجنا بوضوح وقمنا بترميز الحسابات لإن الحاجة مخرجات باستخدام عمليات الجبر الخطى الأساسية. يجب أن تعرف كيف تفعل هذا. ولكن بمجرد أن تصبح نماذجك أكثر تعقيداً، وبمجرد أن تضطر إلى القيام بذلك كل يوم تقريباً، ستكون سعيداً بالمساعدة. الوضع مشابه لترميز مدونتك الخاصة من البداية. يعد القيام بذلك مرة أو مرتين أمراً مفيداً ومفيدةً، لكنك ستكون مطمور ويب رديلاً إذا قضيت شهراً في إعادة اختراع العجلة.

بالنسبة للعمليات القياسية، يمكننا استخدام الطبقات المحددة مسبقاً لإطار العمل، والتي تسمح لنا بالتركيز على الطبقات المستخدمة لبناء النموذج بدلاً من القلق بشأن تنفيذها. أذكر معمارية شبكة أحادية الطبقة على النحو الموصوف في الشكل 2.1.3. تسمى الطبقة متصلة بالكامل fully connected، نظراً لأن كل مدخل من مدخلاتها متصل بكل من مخرجاتها عن طريق ضرب المصفوفة_المتجه $\text{matrix-vector multiplication}$.

في Keras، يتم تحديد الطبقة المتصلة بالكامل في فئة `Dense`. نظراً لأننا نريد فقط إنشاء ناتج قيمة قياسية واحدة `Scalar`، فقد قمنا بتعيين هذا الرقم على 1. وتجدر الإشارة إلى أنه، للسهولة، لا يتطلب منا Keras تحديد شكل الإدخال لكل طبقة. لا تحتاج إلى إخبار Keras بعدد المدخلات التي تدخل في هذه الطبقة الخطية. عندما نحاول لأول مرة تمرير البيانات من خلال نموذجنا، على سبيل المثال، عندما ننفذ `(X) net` لاحقاً، ستستنتج Keras تلقائياً عدد المدخلات لكل طبقة. سنصف كيف يعمل هذا بمزيد من التفصيل لاحقاً.

```
class LinearRegression(d2l.Module): #@save
    def __init__(self, lr):
        super().__init__()
        self.save_hyperparameters()
        initializer =
            tf.initializers.RandomNormal(stddev=0.01)
        self.net = tf.keras.layers.Dense(1,
kernel_initializer=initializer)
```

في طريقة `forward`، نستدعي دالة `call` المضمنة للطبقات المحددة مسبقاً لحساب المخرجات.

```
@d2l.add_to_class(LinearRegression) #@save
def forward(self, X):
    """The linear regression model."""
    return self.net(X)
```

3.5.2. تعريف دالة الخطأ Defining the Loss Function

تحسب فئة `MeanSquaredError` متوسط الخطأ التربيعي (بدون $1/2$ العامل في (3.1.5)). بشكل افتراضي، تقوم بإرجاع متوسط الخطأ على الأمثلة.

```
@d2l.add_to_class(LinearRegression) #@save
def loss(self, y_hat, y):
    fn = tf.keras.losses.MeanSquaredError()
    return fn(y, y_hat)
```

3.5.3. تعريف خوارزمية التحسين Defining the Optimization Algorithm

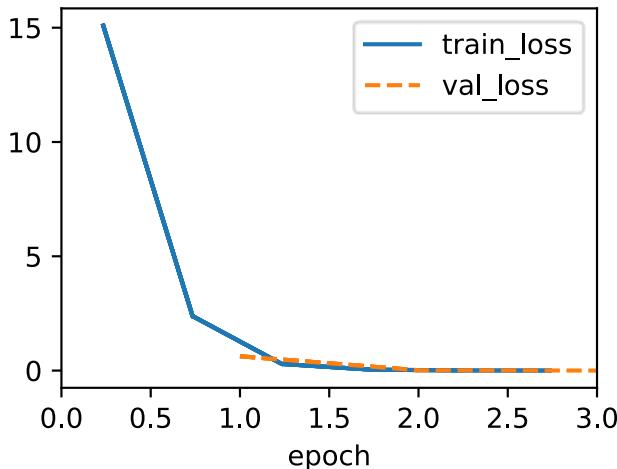
Minibatch SGD هي أداة قياسية لتحسين الشبكات العصبية ، وبالتالي تدعمها Keras جنباً إلى جنب مع عدد من الاختلافات في هذه الخوارزمية في وحدة `optimizers`.

```
@d2l.add_to_class(LinearRegression) #@save
def configure_optimizers(self):
    return tf.keras.optimizers.SGD(self.lr)
```

3.5.4. التدريب Training

ربما لاحظت أن التعبير عن نموذجنا من خلال واجهات برمجة التطبيقات API عالية المستوى لإطار عمل التعلم العميق يتطلب عدداً أقل من أسطر التعليمات البرمجية. لم يكن علينا تخصيص المعلمات بشكل فردي أو تحديد دالة الخطأ أو تنفيذ SGD minibatch. بمجرد أن نبدأ العمل مع نماذج أكثر تعقيداً، فإن مزايا واجهة برمجة التطبيقات عالية المستوى ستنمو بشكل كبير. الآن بعد أن أصبح لدينا جميع الأجزاء الأساسية في مكانها الصحيح، فإن حلقة التدريب نفسها هي نفسها التي طبقناها من البداية. لذلك نحن فقط نستدعي طريقة `fit` (المقدمة في القسم 3.2.4)، والتي تعتمد على تنفيذ طريقة `fit_epoch` في القسم 3.4 ، لتدريب نموذجنا.

```
model = LinearRegression(lr=0.03)
data = d2l.SyntheticRegressionData(w=tf.constant([2, -3.4]), b=4.2)
trainer = d2l.Trainer(max_epochs=3)
trainer.fit(model, data)
```



أدنى، نقارن معلمات النموذج التي تم تعلمها من خلال التدريب على البيانات المحدودة والمعلمات الفعلية التي أنشأت مجموعة البيانات الخاصة بنا. للوصول إلى المعلمات، نصل إلى أوزان وانحياز الطبقة التي نحتاجها. كما هو الحال في تنفيذنا من البداية، لاحظ أن معلماتنا المقدرة قريبة من نظيراتها الحقيقية.

```
@d2l.add_to_class(LinearRegression) #@save
def get_w_b(self):
    return (self.get_weights()[0],
    self.get_weights()[1])

w, b = model.get_w_b()
print(f'error in estimating w: {data.w - tf.reshape(w,
data.w.shape)}')
print(f'error in estimating b: {data.b - b}')
error in estimating w: [ 0.00565076 -0.0058403 ]
error in estimating b: [0.01354933]
```

3.5.5 الملخص

يحتوي هذا القسم على أول تطبيق لشبكة عميقة (في هذا الكتاب) للاستفادة من وسائل الراحة التي توفرها إطار التعلم العميق الحديثة، مثل Tensorflow ، PyTorch ، JAX ، Gluon ، استخدمنا الإعدادات الافتراضية لإطار العمل لتحميل البيانات، وتحديد الطبقة، ودالة الخطأ، والمحسن، وحلقة التدريب. عندما يوفر إطار العمل جميع الميزات الضرورية، فمن الجيد عموماً استخدامها، نظراً لأن تطبيقات المكتبة لهذه المكونات تمثل إلى تحسين الأداء بشكل كبير واختبارها بشكل صحيح من أجل الموثوقية. في نفس الوقت، حاول ألا تنسى أنه يمكن تنفيذ هذه

الوحدات مباشرة. هذا مهم بشكل خاص للباحثين الطموحين الذين يرغبون في العيش على حافة النزف لتطوير النموذج، حيث ستبتكر مكونات جديدة لا يمكن أن توجد في أي مكتبة حالياً.

في TensorFlow، توفر وحدة `data` أدوات لمعالجة البيانات، وتحدد وحدة `keras` عدداً كبيراً من طبقات الشبكة العصبية ودوال الخطأ الشائعة. علاوة على ذلك، توفر وحدة `initializers` طرقاً مختلفة لتهيئة معلمة النموذج. يتم استنتاج الأبعاد والتخصيص للشبكات تلقائياً (ولكن احرص على عدم محاولة الوصول إلى المعلمات قبل تهيئتها).

3.5.6. التمارين

1. كيف ستحتاج إلى تغيير معدل التعلم إذا قمت باستبدال الخطأ الإجمالي على minibatch بمتوسط فوق الخطأ؟

2. راجع وثائق إطار العمل لمعرفة دوال الخطأ المتوفرة. على وجه الخصوص، استبدل الخطأ التربيعي بدالة الخطأ القوية لهوبير Huber's robust loss function. أي، استخدم دالة الخطأ:

$$l(y, y') = \begin{cases} |y - y'| - \frac{\sigma}{2} & \text{if } |y - y'| > \sigma \\ \frac{1}{2\sigma}(y - y')^2 & \text{otherwise} \end{cases}$$

3. كيف يمكنك الوصول إلى التدرج لأوزان النموذج؟

4. كيف يتغير الحل إذا قمت بتغيير معدل التعلم وعدد الفترات؟ هل تستمر في التحسن؟

5. كيف يتغير الحل عندما تقوم بتغيير كمية البيانات التي يتم إنشاؤها؟

1. ارسم خطأ التقدير $L(\hat{w}, \hat{b}, b)$ كدالة لمقدار البيانات. تلميح: قم بزيادة كمية البيانات لوغاريتmic وليس خطياً، أي $5, 10, 20, \dots, 10000$ بدلاً من $1000, 2000, \dots, 10000$.

2. لماذا الاقتراح في التلميح مناسب؟

3.6. التعوييم Generalization

ضع في اعتبارك أن اثنين من طلاب الكلية يستعدان بجد للامتحان النهائي. بشكل عام، سيتألف هذا الإعداد من ممارسة واختبار قدرتهم من خلال إجراء الاختبارات التي أجريت في السنوات السابقة. ومع ذلك، فإن الأداء الجيد في الاختبارات السابقة لا يضمن تفوقهم عندما يكون الأمر مهماً. على سبيل المثال، تخيل طالباً، تدعى إيليفنتين إيلي Ellie Elephantine، تألف إعدادها بالكامل من حفظ إجابات أسئلة امتحان السنوات السابقة. حتى لو كانت إيلي تتمتع بذاكرة فبل، وبالتالي تمكنت تماماً من تذكر إجابة أي سؤال سبق رؤيته، فقد تتجمد مع ذلك عندما تواجه

سؤالاً جديداً (لم يسبق رؤيته). بالمقارنة، تخيل أن تلميذة أخرى، إيرين الاستقرائية Inductive Irene، لديها مهارات حفظ ضعيفة نسبياً، ولكنها موهوبة في التقاط الأنماط. لاحظ أنه إذا كان الاختبار يتألف حقاً من أسئلة معاد تدويرها من عام سابق، فستتفوق إيلي بسهولة على إيرين. حتى لو أسفرت أنماط إيرين المستنبطة عن تنبؤات دقيقة بنسبة 90٪، فلا يمكنها أبداً منافسة استدعاء إيلي بنسبة 100٪. ومع ذلك، حتى لو كان الاختبار يتكون بالكامل من أسئلة جديدة، فقد تحافظ إيرين على متوسطها البالغ 90٪.

كعلماء في التعلم الآلي، هدفنا هو اكتشاف الأنماط discover patterns. ولكن كيف يمكننا التأكد من أنها اكتشفنا حقاً نمطاً عاماً ولم نحفظ بياناتنا ببساطة؟ في معظم الأحيان، تكون تنبؤاتنا مفيدة فقط إذا اكتشفنا نموذجنا مثل هذا النمط. لا نريد توقع أسعار الأسهم في الأمس، ولكن أسعار الغد. لا نحتاج إلى التعرف على الأمراض التي تم تشخيصها بالفعل للمرضى الذين تمت رؤيتهم سابقاً، ولكن بالأحرى الأمراض التي لم يتم تشخيصها من قبل في المرضى الذين لم يتم رؤيتهم من قبل. هذه المشكلة – كيفية اكتشاف الأنماط التي تعم how to discover patterns that generalize جميع الإحصائيات. قد نعتبر هذه المشكلة مجرد شريحة واحدة من سؤال أعظم بكثير يبتلع العلم كله: متى يكون لدينا ما يبرر قيامنا بالقفزة من ملاحظات معينة إلى عبارات أكثر عمومية (Popper, 2005)؟

في الحياة الواقعية، يجب علينا ملائمة fit النماذج باستخدام مجموعة محدودة من البيانات. تختلف المقاييس النموذجية لتلك البيانات بشكل كبير عبر المجالات. بالنسبة للعديد من المشكلات الطبية المهمة، لا يمكننا الوصول إلا إلى بضعة آلاف من نقاط البيانات. عند دراسة الأمراض النادرة، قد تكون محظوظين للوصول إلى المئات. على النقيض من ذلك، تحتوي أكبرمجموعات البيانات العامة التي تتكون من صور فوتوغرافية مصنفة (على سبيل المثال، ImageNet (Deng et al., 2009)، على ملايين الصور. وبعضمجموعات الصور غير المسماة unlabeled image Flickr YFC100M يمكن أن تكون أكبر، حيث تحتوي على أكثر من 100 مليون صورة (Thomee et al., 2016). ومع ذلك، حتى في هذا النطاق الأقصى، يظل عدد نقاط البيانات المتاحة صغيراً بشكل لا نهائي مقارنة بمساحة جميع الصور الممكنة بدقة 1 ميجابكسل. عندما نعمل مع عينات محدودة، يجب أن نضع في اعتبارنا المخاطر التي قد نلائمهَا مع بيانات التدريب الخاصة بنا، فقط لنكتشف أننا فشلنا في اكتشاف نمط قابل للتمثيم.

تسمى ظاهرة الاقتراب من بيانات التدريب الخاصة بنا أكثر من التوزيع الأساسي بفرط التجهيز او التعلم overfitting، وغالباً ما تسمى تقنيات مكافحة فرط التعلم بأساليب التنظيم

regularization. بينما لا يوجد بديل لمقدمة مناسبة لنظرية التعلم الإحصائي (انظر Vapnik (1998), Boucheron et al. (2005)). سمنحك الحدس الكافي فقط للبدء. سوف نعيid النظري التعميم في العديد من الفصول في جميع أنحاء الكتاب، ونستكشف كل مما هو معروف عن المبادئ الكامنة وراء التعميم في النماذج المختلفة، وكذلك التقنيات الاستدلالية التي تم العثور عليها (تجريبياً) لتحقيق تعميم أفضل للمهام ذات الأهمية العملية.

3.6.1 خطأ التدريب وخطأ التعميم Training Error and Generalization Error

في إعداد التعلم القياسي الخاضع للإشراف، نفترض أن بيانات التدريب وبيانات الاختبار مستمدة بشكل مستقل عن التوزيعات المتطابقة. Independently from Identical Distributions. وهذا ما يسمى عادة افتراض IID. في حين أن هذا الافتراض قوي، فمن الجدير بالذكر أنه في غياب أي افتراض من هذا القبيل سنكون ميتين في الماء. لماذا يجب أن نعتقد أن بيانات التدريب المأخوذة من التوزيع $P(X, Y)$ يجب أن تخبرنا بكيفية عمل تنبؤات بشأن بيانات الاختبار الناتجة عن توزيع مختلف $Q(X, Y)$? إن تحقيق مثل هذه القفزات يتطلب افتراضات قوية حول كيفية ارتباط P و Q . ستناقش لاحقاً بعض الافتراضات التي تسمح بالتغييرات في التوزيع ولكننا نحتاج أولاً إلى فهم حالة IID حيث $P(\cdot) = Q(\cdot)$.

بادئ ذي بدء، نحتاج إلى التفريق بين خطأ التدريب R_{emp} ، وهو إحصاء محسوب على مجموعة بيانات التدريب، وخطأ التعميم R ، وهو توقع يتم إجراؤه فيما يتعلق بالتوزيع الأساسي. يمكن التفكير في خطأ التعميم كما تراه إذا قمت بتطبيق النموذج الخاص بك على دفق لا نهائي infinite stream من أمثلة البيانات الإضافية المستمدة من نفس توزيع البيانات الأساسي. رسمياً، يتم التعبير عن خطأ التدريب كمجموع (بنفس الترميز في القسم 3.1):

$$R_{\text{emp}}[\mathbf{X}, \mathbf{y}, f] = \frac{1}{n} \sum_{i=1}^n l(\mathbf{x}^{(i)}, y^{(i)}, f(\mathbf{x}^{(i)})),$$

بينما يتم التعبير عن خطأ التعميم كجزء لا يتجزأ:

$$R[p, f] = E_{(\mathbf{x}, y) \sim P}[l(\mathbf{x}, y, f(\mathbf{x}))] = \int \int l(\mathbf{x}, y, f(\mathbf{x})) p(\mathbf{x}, y) d\mathbf{x} dy.$$

من الناحية الإشكالية، لا يمكننا أبداً حساب خطأ التعميم تماماً. لا أحد يخبرنا أبداً عن الشكل الدقيق لدالة الكثافة $p(\mathbf{x}, y)$. علاوة على ذلك، لا يمكننا أخذ عينة من دفق لا نهائي من نقاط البيانات. وبالتالي، من الناحية العملية، يجب علينا تقدير خطأ التعميم من خلال تطبيق نموذجنا على مجموعة اختبار مستقلة تتكون من اختيار عشوائي \mathbf{X} للأمثلة والتسميات \mathbf{y} التي تم حجبها

من مجموعة التدريب الخاصة بنا. يتكون هذا من تطبيق نفس الصيغة المستخدمة لحساب خطأ التدريب التجاري ولكن على مجموعة اختبار \mathbf{y}' .

بشكل حاسم، عندما نقوم بتقدير المصنف الخاص بنا على مجموعة الاختبار، فإننا نعمل مع مصنف ثابت fixed classifier (لا يعتمد على عينة مجموعة الاختبار)، وبالتالي فإن تقدير الخطأ هو ببساطة مشكلة تقدير المتوسط problem of mean estimation. ومع ذلك لا يمكن قول الشيء نفسه بالنسبة لمجموعة التدريب. لاحظ أن النموذج الذي انتهينا إليه يعتمد بشكل صريح على اختيار مجموعة التدريب، وبالتالي فإن خطأ التدريب سيكون بشكل عام تقديرًا متحيزًا للخطأ الحقيقي على السكان الأساسيين. السؤال المركزي للتعميم هو متى يجب أن تتوقع أن يكون خطأ التدريب لدينا قريباً من الخطأ السكاني population error (وبالتالي خطأ التعميم).

3.6.1.1 Model Complexity

في النظرية الكلاسيكية، عندما يكون لدينا نماذج بسيطة وبيانات وفيرة، تميل خطاء التدريب والتعميم إلى التقارب. ومع ذلك، عندما نعمل مع نماذج أكثر تعقيداً و / أو أمثلة أقل، نتوقع أن ينخفض خطأ التدريب بينما تنمو فجوة التعميم. هذا يجب أن لا يكون مفاجئاً. تخيل أن فئة النموذج معبرة جداً للدرجة أنه بالنسبة لأي مجموعة بيانات n من الأمثلة، يمكننا العثور على مجموعة من المعلمات التي يمكن أن تتلاءم تماماً مع التسميات التعسفية، حتى لو تم تعينها عشوائياً. في هذه الحالة، حتى لو تناسبنا بيانات التدريب الخاصة بنا تماماً، كيف يمكننا استنتاج أي شيء عن خطأ التعميم؟ لكل ما نعرفه، قد لا يكون خطأ التعميم أفضل من التخمين العشوائي.

بشكل عام، في حالة عدم وجود أي قيود على فئة النموذج لدينا، لا يمكننا الاستنتاج بناءً على ملائمة بيانات التدريب وحدها أن نموذجنا قد اكتشف أي نمط قابل للتعميم (Vapnik et al., 1994). من ناحية أخرى، إذا كانت فئة النموذج الخاصة بنا غير قادرة على ملائمة تسميات عشوائية، فلا بد أنها اكتشفت نمطاً. استمدت الأفكار النظرية التعليمية حول تعقيد النموذج بعض الإلهام من أفكار كارل بوبير، فيلسوف العلم المؤثر، الذي صاغ معيار القابلية للتزميف criterion of falsifiability. وفقاً لبوبير، النظرية التي يمكن أن تشرح أي وجميع الملاحظات ليست نظرية علمية على الإطلاق! بعد كل شيء، ماذا قال لنا عن العالم إذا لم يستبعد أي احتمال؟ باختصار، ما نريده هو فرضية لا يمكن أن تنكسر أي ملاحظات قد نتصورها ومع ذلك تصادف أنها متوافقة مع تلك الملاحظات التي نقوم بها في الواقع.

الآن ما يشكل بدقة فكرة مناسبة لتعقيد النموذج هو مسألة معقدة. في كثير من الأحيان، تكون النماذج التي تحتوي على المزيد من المعلمات قادرة على ملائمة عدد أكبر من التسميات المعينة بشكل تعسفي. ومع ذلك، هذا ليس صحيحاً بالضرورة. على سبيل المثال، تعمل طرق التوازن

kernel methods تعقيدها بوسائل أخرى (Scholkopf and Smola, 2002). أحد المفاهيم التي غالباً ما تكون مفيدة عن التعقيد هو نطاق القيم التي يمكن أن تخذلها المعلمات. هنا، سيكون النموذج الذي يُسمح لمعلماته بأخذ قيم عشوائية أكثر تعقيداً. سعید النظر في هذه الفكرة في القسم التالي، عندما نقدم تناقض او اضمحلال الوزن weight decay، وهو أول أسلوب عملي لتتنظيم الأمور. والجدير بالذكر أنه قد يكون من الصعب مقارنة التعقيد بين أعضاء فئات النماذج المختلفة إلى حد كبير (على سبيل المثال، أشجار القرار decision trees مقابل الشبكات العصبية neural networks).

في هذه المرحلة، يجب أن نؤكّد على نقطة مهمة أخرى سعید النظر فيها عند إدخال الشبكات العصبية العميقية. عندما يكون النموذج قادرًا على تركيب تسميات عشوائية، فإن خطأ التدريب المنخفض لا يعني بالضرورة خطأ التعميم منخفض. ومع ذلك، فإنه لا يعني بالضرورة خطأ التعميم العالي أيضًا! كل ما يمكننا قوله بشدة هو أن خطأ التدريب المنخفض وحده لا يكفي للتصديق على خطأ التعميم المنخفض. تبيّن أن الشبكات العصبية العميقية هي مجرد نماذج من هذا القبيل: في بينما يتم تعميمها جيدًا في الممارسة العملية، فهي قوية جدًا بحيث لا تسمح لنا باستنتاج الكثير على أساس خطأ التدريب وحده. في هذه الحالات، يجب أن نعتمد بشكل أكبر على بياناتنا المؤجلة للمصادقة على التعميم بعد وقوع الحقيقة. يسمى الخطأ في بيانات الانتظار، أي مجموعة التتحقق من الصحة validation set، خطأ التتحقق من الصحة validation error.

3.6.2. الضبط الناقص Underfitting أو الضبط الزائد Overfitting؟

عندما نقارن خطأ التدريب والتحقق من الصحة، نريد أن نضع في اعتبارنا حالتين شائعتين. أولاً، نريد أن ننتبه للحالات التي يكون فيها خطأ التدريب وخطأ التتحقق من الصحة كبيرًا ولكن هناك فجوة صغيرة بينهما. إذا كان النموذج غير قادر على تقليل خطأ التدريب، فقد يعني ذلك أن نموذجنا بسيط للغاية (أي غير معبر بشكل كافٍ insufficiently expressive) لالتقاط النمط الذي نحاول نمذجته. علاوة على ذلك، نظرًا لأن فجوة التعميم ($R_{\text{emp}} - R$) بين خطأ التدريب والتعميم لدينا صغيرة، فلدينا سبب للاعتقاد بأنه يمكننا التخلص من نموذج أكثر تعقيدًا. تُعرف هذه الظاهرة باسم الضبط الناقص underfitting.

من ناحية أخرى، كما ناقشنا أعلاه، نريد أن ننتبه للحالات التي يكون فيها خطأ التدريب لدينا أقل بكثير من خطأ التتحقق من الصحة، مما يشير إلى الضبط الزائد (فرط التجهيز) overfitting الشديد. لاحظ أن فرط التجهيز ليس دائمًا أمرًا سيئًا. في التعلم العميق على وجه الخصوص، غالباً ما يكون أداء أفضل النماذج التنبؤية أفضل بكثير على بيانات التدريب مقارنةً ببيانات الرافضة holdout data. في النهاية، نحن نهتم عادةً بتوجيه خطأ التعميم إلى الأسفل، ولا نهتم إلا بالفجوة

يقدر ما تصبح عقبة في طريق تحقيق هذه الغاية. لاحظ أنه إذا كان خطأ التدريب صفرًا، فإن فجوة التعميم تساوي تماماً خطأ التعميم ولا يمكننا إحراز تقدم إلا من خلال تقليل الفجوة.

3.6.2.1 ملائمة منحنى متعدد الحدود

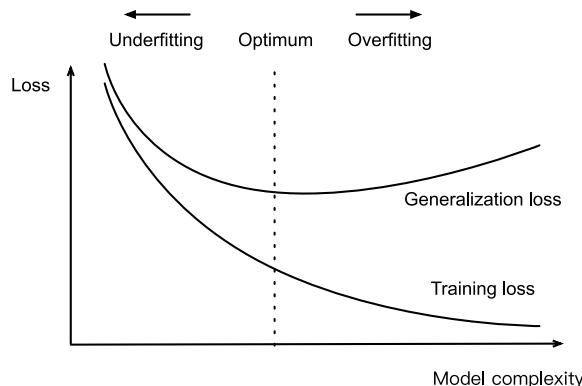
لتوضيح بعض الحدس الكلاسيكي حول الضبط الزائد وتعقيد النموذج، ضع في اعتبارك ما يلي: بالنظر إلى بيانات التدريب التي تتكون من ميزة واحدة x وعلامة ذات قيمة حقيقية مقابلة y ، نحاول العثور على متعدد الحدود للدرجة d

$$\hat{y} = \sum_{i=0}^d x^i w_i$$

لتقدير التسمية y . هذه مجرد مشكلة انحدار خططي حيث يتم إعطاء ميزاتنا من خلال قوى x ، وتعطى أوزان النموذج من قبل w_i ، ويتم إعطاء التحيز w_0 من $1 = x^0$ الحين لكل x . نظرًا لأن هذه مجرد مشكلة انحدار خططي، فيمكننا استخدام الخطأ التربيعي كدالة للخطأ.

تعتبر دالة كثيرة الحدود ذات الترتيب الأعلى higher-order polynomial function تعقيدًا من دالة كثيرة الحدود ذات الترتيب الأدنى lower-order polynomial function، نظرًا لأن كثير الحدود ذو الترتيب الأعلى يحتوي على معلمات أكثر ونطاق اختيار دالة النموذج أوسع. عند إصلاح مجموعة بيانات التدريب، يجب أن تتحقق الدوال متعددة الحدود ذات الترتيب الأعلى دائمًا خطأ تدريب أقل (في أسوأ الأحوال، متساوٍ بالنسبة إلى كثيرات الحدود من الدرجة الأدنى). في الواقع، كلما كان لكل مثال بيانات قيمة مميزة لـ x ، فإن دالة متعددة الحدود بدرجة متساوية لعدد أمثلة البيانات يمكن أن تلامس fit مجموعة التدريب تماماً. نحن نتخيل العلاقة بين درجة متعددة الحدود (تعقيد النموذج) والضبط الناقص مقابل الضبط الزائد في الشكل

3.6.1



الشكل 3.6.1 تأثير تعقيد النموذج على الضبط الناقص والزائد

3.6.2.2 Dataset Size مجموعة البيانات حجم

كما يشير الحد أعلى بالفعل، هناك اعتبار كبير آخر يجب أخذها في الاعتبار وهو حجم مجموعة البيانات dataset size. عند إصلاح نموذجنا، كلما قل عدد العينات التي لدينا في مجموعة بيانات التدريب، زادت احتمالية (والأكثر خطورة) أن نواجه فرط التجهيز. مع زيادة كمية بيانات التدريب، ينخفض خطأ التعميم عادةً. علاوة على ذلك، بشكل عام، المزيد من البيانات لا يضر أبداً. بالنسبة للمهمة الثابتة وتوزيع البيانات، يجب ألا يزيد تعقيد النموذج بسرعة أكبر من كمية البيانات. بالنظر إلى المزيد من البيانات، قد نحاول ملاءمة نموذج أكثر تعقيداً. في غياب البيانات الكافية، قد يكون من الصعب التغلب على النماذج الأليفة. بالنسبة للعديد من المهام، يتضمن التعلم العميق فقط على النماذج الخطية عندما توفر عدةآلاف من أمثلة التدريب. يرجع النجاح الحالي للتعلم العميق جزئياً إلى وفرةمجموعات البيانات الضخمة الناشئة عن شركات الإنترنت والتخزين الرخيص والأجهزة المتصلة والرقمنة الواسعة للاقتصاد.

3.6.3 Model Selection اختيار النموذج

عادة، نختار نموذجنا النهائي، فقط بعد تقييم نماذج متعددة تختلف بطرق مختلفة (بني مختلفة، أهداف تدريب، ميزات مختارة، معالجة البيانات المسبقة، معدلات التعلم، إلخ). الاختيار من بين العديد من النماذج يسمى على نحو مناسب اختيار النموذج model selection.

من حيث المبدأ، لا ينبغي أن نلمس مجموعة الاختبار الخاصة بنا إلا بعد أن نختار جميع معلماتنا الفائقة. إذا استخدمنا بيانات الاختبار في عملية اختيار النموذج، فهناك خطر أننا قد نفترط overfit في بيانات الاختبار. عندها سنكون في مشكلة خطيرة. إذا قمنا بزيادة بيانات التدريب الخاصة بنا، فهناك دائمًا تقييم لبيانات الاختبار لإيقاعنا صادقين. ولكن إذا تجاوزنا بيانات الاختبار، فكيف لنا أن نعرف ذلك؟ انظر al Ong et al (2005) على سبيل المثال كيف يمكن أن يؤدي ذلك إلى نتائج سخيفة حتى بالنسبة للنماذج التي يمكن التحكم فيها بإحكام في التعقيد.

وبالتالي، لا ينبغي لنا أبداً الاعتماد على بيانات الاختبار لاختيار النموذج. ومع ذلك لا يمكننا الاعتماد فقط على بيانات التدريب لاختيار النموذج إما لأننا لا نستطيع تقدير خطأ التعميم على نفس البيانات التي نستخدمها لتدريب النموذج.

في التطبيقات العملية، تصبح الصورة أكثر تعكيراً. في حين أنها من الناحية المثالية لن نلمس بيانات الاختبار إلا مرة واحدة، لتقييم أفضل نموذج أو لمقارنة عدد صغير من النماذج مع بعضها البعض، نادرًا ما يتم تجاهل بيانات الاختبار الواقعية بعد استخدام واحد فقط. نادرًا ما يمكننا تحمل مجموعة اختبار جديدة لكل جولة من التجارب. في الواقع، يمكن أن يكون لإعادة تدوير البيانات المعيارية لعقود تأثير كبير على تطوير الخوارزميات، على سبيل المثال، لتصنيف الصور image classification والتعرف البصري على الأحرف optical character recognition.

تتمثل الممارسة الشائعة لمعالجة مشكلة التدريب على مجموعة الاختبار في تقسيم بياناتنا بثلاث طرق، بما في ذلك مجموعة التحقق من الصحة بالإضافة إلى مجموعات بيانات التدريب والاختبار. والنتيجة هي ممارسة غامضة حيث تكون الحدود بين التتحقق من الصحة وبيانات الاختبار غامضة بشكل مثير للقلق. ما لم ينص صراحة على خلاف ذلك، في التجارب في هذا الكتاب، نحن نعمل حفاظاً على ما ينبغي أن يُطلق عليه حفاظاً على بيانات التدريب وبيانات التتحقق من الصحة، بدونمجموعات اختبار حقيقية. لذلك، فإن الدقة المبلغ عنها هي كل تجربة للكتاب هي في الحقيقة دقة التتحقق وليس دقة مجموعة اختبار حقيقية.

3.6.3.1. Cross-Validation

عندما تكون بيانات التدريب نادرة، فقد لا نتمكن حتى من تحمل بيانات كافية لتشكيل مجموعة تتحقق مناسبة. أحد الحلول الشائعة لهذه المشكلة هو استخدام التتحقق المتبادل k-fold Cross-Validation. هنا، يتم تقسيم بيانات التدريب الأصلية إلى K مجموعات فرعية غير متداخلة. ثم يتم تنفيذ تدريب النموذج والتتحقق من الصحة K مرات، في كل مرة يتم فيها التدريب على مجموعات فرعية والتتحقق من صحة $1 - K$ مجموعة فرعية مختلفة (المجموعة التي لم يتم استخدامها للتدريب في تلك الجولة). أخيراً، يتم تقدير أخطاء التدريب والتتحقق من الصحة من خلال حساب متوسط النتائج من K التجارب.

3.6.4. الملخص

استكشف هذا القسم بعض أساسيات التعميم في التعلم الآلي. تصبح بعض هذه الأفكار معقدة وغير بدائيّة عندما نصل إلى نماذج أعمق، فهناك، تكون النماذج قادرة على الضبط الزائد overfitting للبيانات بشكل سيء، ويمكن أن تكون مفاهيم التعقيد ذات الصلة ضمنية وغير بدائيّة (على سبيل المثال، البني الأكبر مع المزيد من المعلومات المعممة بشكل أفضل). ترك لك بعض القواعد الأساسية:

1. استخدام مجموعات التتحقق من الصحة (أو التتحقق المتبادل k-fold Cross-validation) لاختيار النموذج؛
2. غالباً ما تتطلب النماذج الأكثر تعقيداً المزيد من البيانات؛
3. تتضمن مفاهيم التعقيد ذات الصلة كلاً من عدد المعلومات ونطاق القيم التي يُسمح لها بأخذها؛
4. مع الحفاظ على جميع الأمور الأخرى متساوية، تؤدي البيانات المتزايدة دائمًا تقريرياً إلى تعميم أفضل؛

5. كل هذا الحديث عن التعميم مبني على افتراض IID. إذا قمنا بتخفيف هذا الافتراض، والسماح للتوزيعات بالانتقال بين فترات التدريب والاختبار، فلا يمكننا قول أي شيء عن التعميم في غياب افتراض آخر (ربما أكثر اعتدالاً).

3.6.5 التمارين

1. متى يمكنك حل مشكلة الانحدار متعدد الحدود polynomial regression بالضبط؟

2. أعط ما لا يقل عن خمسة أمثلة حيث المتغيرات العشوائية التابعة تجعل معالجة المشكلة على أنها بيانات IID غير مستحسنة.

3. هل يمكن أن تتوقع ألا ترى أي خطأ في التدريب؟ ما هي الظروف التي سترى فيها خطأ التعميم الصغرى؟

4. لماذا يعد التحقق من الصحة المتبادل ذو الطيات k-fold Cross-Validation مكلفاً جداً للحساب؟

5. لماذا يكون تقدير خطأ التتحقق المتبادل ذو أضعاف متحازة؟

6. يتم تعريف VC على أنه الحد الأقصى لعدد النقاط التي يمكن تصنيفها باستخدام تسميات عشوائية $\{\pm 1\}$ بواسطة دالة لفثة من الدوال. لماذا قد لا تكون هذه فكرة جيدة لقياس مدى تعقيد فئة الدوال؟ تلميح: ماذا عن حجم الدوال؟

7. يمنحك مديرك مجموعة بيانات صعبة لا تعمل فيها الخوارزمية الحالية بشكل جيد. كيف تبرر له أنك بحاجة إلى مزيد من البيانات؟ تلميح: لا يمكنك زيادة البيانات ولكن يمكنك تقليلها.

3.7 اضمحلال الوزن Weight Decay

الآن بعد أن وصفنا مشكلة الضبط الزائد overfitting، يمكنك تقديم تقنية التنظيم regularization الأولى لدينا. تذكر أنه يمكننا دائمًا التخفيف من فرط التجهيز (الضبط الزائد) من خلال جمع المزيد من بيانات التدريب. ومع ذلك، قد يكون ذلك مكلفاً أو يستغرق وقتاً طويلاً أو خارج عن سيطرتنا تماماً، مما يجعله مستحيلاً على المدى القصير. في الوقت الحالي، يمكنك أن نفترض أن لدينا بالفعل قدرًا كبيرًا من البيانات عالية الجودة بقدر ما تسمح به مواردنا ونركز على الأدوات الموجودة تحت تصرفنا حتى عندما يتمأخذ مجموعة البيانات على أنها أمر مفروغ منه.

تذكرة أنه في مثال الانحدار متعدد الحدود polynomial regression الخاص بنا (القسم 3.6.2.1)، يمكنك تحديد سعة نموذجنا عن طريق تعديل درجة كثير الحدود المتفوقة. في الواقع، يعد الحد من عدد الميزات أسلوبًا شائعاً للتخفيف من الضبط الزائد. ومع ذلك، فإن مجرد

إهمال الميزات جانبياً يمكن أن يكون أداة حادة للغاية. بالتمسك بمثال الانحدار متعدد الحدود، يضع في اعتبارك ما يمكن أن يحدث مع المدخلات عالية الأبعاد $\text{high-dimensional input}$. تسمى الامتدادات الطبيعية لكثيرات الحدود للبيانات متعددة المتغيرات بـ احادية الحد monomials ، والتي هي ببساطة نتاج قوى المتغيرات. درجة المتغيرات أحادية الحد هي مجموع القوى. فمثلاً، x_2^2 و x_3^5 كلاهما أحادي الحد من الدرجة الثالثة.

لاحظ أن عدد الحدود مع الدرجة d يتضخم بسرعة كلما زاد حجم d . بالنظر إلى المتغيرات k ، يكون عدد أحadiات الحد من الدرجة d (أي k متعدد الخيوط d) هو $\binom{k-1+d}{k-1}$. حتى التغييرات الصغيرة في الدرجة، نقل من 2 إلى 3 ، تزيد بشكل كبير من تعقيد نموذجنا. وبالتالي نحتاج غالباً إلى أداة أكثر دقة لتعديل تعقيد الدالة.

3.7.1 المعايير وتناقص الوزن

بدلاً من التلاعب المباشر بعدد المعلمات، يعمل تناقص الوزن عن طريق تقييد القيم التي يمكن أن تأخذها المعلمات. أكثر شيوعاً يسمى التنظيم (ℓ_2 regularization) خارج دوائر التعلم العميق عند تحسينه عن طريق التدرج الاستقافي العشوائي المصغر minibatch SGD قد يكون تناقص الوزن هو الأسلوب الأكثر استخداماً على نطاق واسع لتنظيم نماذج التعلم الآلي البارامترية. يتم تحفيز هذه التقنية من خلال الحدس الأساسي الذي مفاده أن الدالة $f = 0$ (تعين القيمة 0 لجميع المدخلات) من بين جميع الدوال f هي أبسطها بمعنى ما، وأنه يمكننا قياس مدى تعقيد الدالة من خلال مسافة معلماتها من الصفر. ولكن ما مدى دقة قياس المسافة بين الدالة والصفر؟ لا توجد إجابة واحدة صحيحة. في الواقع، تم تخصيص فروع كاملة للرياضيات، بما في ذلك أجزاء من التحليل الدالي ونظرية فضاءات باناخ Banach spaces لمعالجة مثل هذه القضايا.

قد يكون أحد التفسيرات البسيطة هو قياس مدى تعقيد دالة خطية $\mathbf{w}^T \mathbf{x} = f(\mathbf{x})$ من خلال معيار norm معين لمتجه وزنها، على سبيل المثال $\|\mathbf{w}\|^2$. تذكر أنها قدمنا المعيار ℓ_2 والمعيار ℓ_1 ، وهما حالات خاصة للمعيار الأكثر عمومية ℓ_p في القسم 3.11. الطريقة الأكثر شيوعاً لضمان متجه وزن صغير هي إضافة معياره كمصطلاح جزائي لمشكلة تقليل الخطأ. وهكذا نستبدل هدفنا الأصلي، وهو تقليل خطأ التنبؤ prediction loss على تسميات التدريب training labels، بهدف جديد، وتقليل مجموع خطأ التنبؤ sum of the prediction loss ومدة العقوبة penalty term. الآن، إذا نما متجه الوزن بشكل كبير جداً، فقد تركز خوارزمية التعلم الخاصة بنا على تقليل معيار الوزن $\|\mathbf{w}\|^2$ إلى الحد الأدنى مقابل تقليل خطأ التدريب. هذا هو بالضبط ما نريده. لتوضيح الأشياء في الكود، نعيد إحياء مثالنا السابق من القسم 3.1 للانحدار الخطى. هناك، خطأنا:

$$L(\mathbf{w}, b) = \frac{1}{n} \sum_{i=1}^n \frac{1}{2} (\mathbf{w}^\top \mathbf{x}^{(i)} + b - y^{(i)})^2.$$

تذكر أن $\mathbf{x}^{(i)}$ هي الميزات، $y^{(i)}$ هي التسمية لأي مثال بيانات i ، (\mathbf{w}, b) وهي معلمات الوزن والتحيز، على التوالي. لمعاقبة حجم متوجه الوزن weight vector $\|\mathbf{w}\|^2$ بطريقة ما إلى دالة الخطأ، ولكن كيف يجب أن يقايس النموذج الخطأ القياسي لهذه العقوبة المضافة الجديدة؟ في الممارسة العملية، نميز هذه المقايسة من خلال ثابت التنظيم λ ، وهو معلمة فائقة غير سلبية نلائمه باستخدام بيانات التحقق من الصحة validation data :

$$L(\mathbf{w}, b) + \frac{\lambda}{2} \|\mathbf{w}\|^2.$$

بالنسبة $\lambda > 0$ ، نستعيد دالة الخسارة الأصلية الخاصة بنا. $\lambda < 0$ ، نحن نقيد حجم $\|\mathbf{w}\|$. نقسم على 2 حسب الاصطلاح: عندما نأخذ مشتق دالة تربيعية، نلغى 2 ونلغي 1/2 ، مما يضمن أن تعبير التحديث يبدو لطيفاً وبسيطاً. قد يتساءل القارئ الذكي عن سبب تعاملنا مع المعيار التربيعي squared norm وليس المعيار القياسي standard norm (أي المسافة الإقليدية). نحن نفعل هذا للراحة الحسابية. بتربع المعيار ℓ_2 ، نزيل الجذر التربيعي، ونترك مجموع مربعات كل مكون من متوجه الوزن. هذا يجعل من السهل حساب مشتق العقوبة: مجموع المشتقات يساوي مشتق المجموع.

علاوة على ذلك، قد تسأل لماذا تعامل مع المعيار ℓ_2 في المقام الأول وليس، على سبيل المثال، المعيار ℓ_1 . في الواقع، الخيارات الأخرى صالحة وشائعة في جميع الإحصاءات. في حين أن النماذج الخطية المنتظمة ℓ_2 تشكل خوارزمية انحدار ريدج ridge regression algorithm ، فإن الانحدار الخطى المنتظم ℓ_1 هو طريقة أساسية مماثلة في الإحصاء، والمعروفة باسم انحدار لاسو (lasso regression). أحد أسباب العمل مع المعيار ℓ_1 هو أنه يفرض عقوبة كبيرة على المكونات الكبيرة لمتجه الوزن. يؤدي هذا إلى تحيز خوارزمية التعلم الخاصة بنا نحو النماذج التي توزع الوزن بالتساوي عبر عدد أكبر من الميزات. في الممارسة العملية، قد يجعلهم ذلك أكثر قوية في مواجهة خطأ القياس في متغير واحد. على النقيض من ذلك، تؤدي العقوبات ℓ_1 إلى نماذج تركز الأوزان على مجموعة صغيرة من الميزات عن طريق إزالة الأوزان الأخرى إلى الصفر. يمكننا هذا طريقة فعالة لاختيار الميزة، والتي قد تكون مرغوبة لأسباب أخرى. على سبيل المثال، إذا كان نموذجنا يعتمد فقط على بعض الميزات، فقد لا تحتاج إلى جمع البيانات أو تخزينها أو نقلها للميزات الأخرى (التي تم إسقاطها dropped).

باستخدام نفس الترميز في (3.1.11)، تتبع تحديثات التدرج الاستقaci العشوائي المصغر للانحدار المنتظم ℓ_2 :

$$\mathbf{w} \leftarrow (1 - \eta\lambda)\mathbf{w} - \frac{\eta}{|\mathcal{B}|} \sum_{i \in \mathcal{B}} \mathbf{x}^{(i)} (\mathbf{w}^\top \mathbf{x}^{(i)} + b - y^{(i)}).$$

كما في السابق، نقوم بتحديث \mathbf{w} بناءً على المقدار الذي يختلف به تقديرنا عن الملاحظة. ومع ذلك، فإننا نقوم أيضاً بتقليل حجم \mathbf{w} باتجاه الصفر. هذا هو السبب في أن الطريقة تسمى أحياناً "تضاؤل او تناقص الوزن weight decay" : نظراً لمصطلح العقوبة وحده، فإن خوارزمية التحسين الخاصة بنا تقص الوزن في كل خطوة من خطوات التدريب. على عكس اختيار الميزة feature selection، يوفر لنا تناقص الوزن آلية مستمرة لضبط تعقيد الدالة. توافق القيم الأصغر λ لأقل تقييداً لـ \mathbf{w} ، في حين أن القيم الأكبر λ تقييد \mathbf{w} أكثر وضوحاً. ما إذا كنا نقوم بتضمين عقوبة التحيز b^2 المقابلة يمكن أن تختلف عبر التطبيقات، وقد تختلف عبر طبقات الشبكة العصبية. في كثير من الأحيان، لا نقوم بتظام مصطلح التحيز. إلى جانب ذلك، على الرغم من أن التنظيم قد لا يكون مكافأً لتناقص الوزن بالنسبة لخوارزميات التحسين الأخرى، إلا أن فكرة التنظيم من خلال تقليل حجم الأوزان لا تزال صحيحة.

3.7.2 الانحدار الخطي عالي الأبعاد Regression

يمكننا توضيح فوائد تناقص الوزن من خلال مثال تركيبي بسيط.

```
%matplotlib inline
import tensorflow as tf
from d2l import tensorflow as d2l
```

أولاً، نقوم بإنشاء بعض البيانات كما في السابق:

$$y = 0.05 + \sum_{i=1}^d 0.01x_i + \epsilon \text{ where } \epsilon \sim \mathcal{N}(0, 0.01^2).$$

في مجموعة البيانات التركيبية synthetic dataset هذه، يتم إعطاء التسمية الخاصة بنا من خلال دالة خطية أساسية لمدخلاتنا، تالفة بسبب الضوضاء الغاويسية مع متوسط الصفر والانحراف المعياري 0.01. لأغراض توضيحية، يمكننا أن نجعل تأثيرات الضبط الزائد واضحة، من خلال زيادة أبعاد مشكلتنا لـ $d = 200$ والعمل مع مجموعة تدريب صغيرة مع 20 مثلاً فقط.

```
class Data(d2l.DataModule):
    def __init__(self, num_train, num_val, num_inputs,
batch_size):
        self.save_hyperparameters()
        n = num_train + num_val
```

```

self.X = tf.random.normal((n, num_inputs))
noise = tf.random.normal((n, 1)) * 0.01
w, b = tf.ones((num_inputs, 1)) * 0.01, 0.05
self.y = tf.matmul(self.X, w) + b + noise

def get_dataloader(self, train):
    i = slice(0, self.num_train) if train else
slice(self.num_train, None)
    return self.get_tensorloader([self.X, self.y],
train, i)

```

3.7.3. التنفيذ من البداية

الآن، دعونا نحاول تطبيق تناقص الوزن من الصفر. نظرًا لأن التدرج الاشتقافي العشوائي المصغر SGD هو المحسن الخاص بنا ، فنحن نحتاج فقط إلى إضافة عقوبة تربيعية ℓ_2 إلى دالة الخطأ الأصلية.

3.7.3.1. تحديد عقوبة ℓ_2 المعيارية

ربما تكون الطريقة الأكثر ملاءمة لتنفيذ هذه العقوبة هي تسوية كل الشروط الموجودة وتلخيصها.

```

def l2_penalty(w):
    return tf.reduce_sum(w**2) / 2

```

3.7.3.2. Defining the Model

في النموذج النهائي، لم يتغير الانحدار الخطى والخطأ التربيعي منذ القسم 3.4، لذلك سنقوم فقط بتعريف فئة فرعية من `d2l.LinearRegressionScratch`. التغيير الوحيد هنا هو أن خطأنا تشمل الآن مصطلح العقوبة.

```

class WeightDecayScratch(d2l.LinearRegressionScratch):
    def __init__(self, num_inputs, lambd, lr,
sigma=0.01):
        super().__init__(num_inputs, lr, sigma)
        self.save_hyperparameters()

```

```

    def loss(self, y_hat, y):
        return super().loss(y_hat, y) + self.lambd *
l2_penalty(self.w)

```

يناسب الكود التالي نموذجنا في مجموعة التدريب مع 20 مثالًا ويقيمه على مجموعة التحقق من الصحة مع 100 مثال.

```

data = Data(num_train=20, num_val=100, num_inputs=200,
batch_size=5)

```

```

trainer = d2l.Trainer(max_epochs=10)

def train_scratch(lambd):
    model = WeightDecayScratch(num_inputs=200,
    lambd=lambd, lr=0.01)
    model.board.yscale='log'
    trainer.fit(model, data)
    print('L2 norm of w:', float(l2_penalty(model.w)))

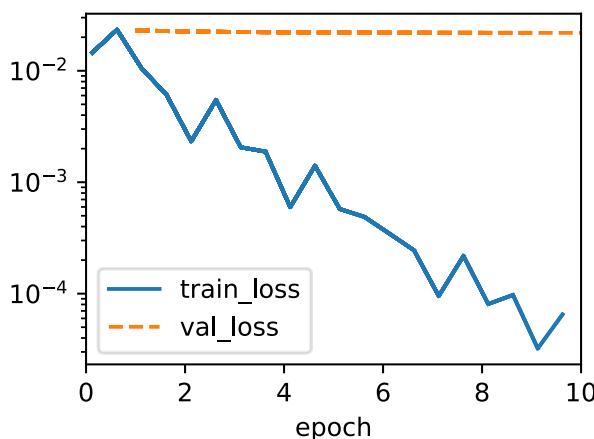
```

3.7.3.3. التدريب بدون التنظيم

نقوم الآن بتشغيل هذا الكود مع $\lambda = 0$ ، مما يؤدي إلى تعطيل تناقص الوزن weight decay. لاحظ أننا نفترض في التجهيز بشكل سيء، مما يقلل من خطأ التدريب ولكن ليس خطأ التحقق من الصحة – وهي حالة كتابية من الضبط الزائد overfitting.

train_scratch(0)

L2 norm of w: 0.010603310540318489

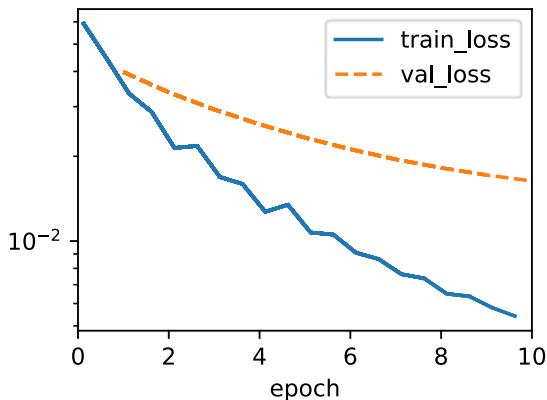


3.7.3.4. استخدام تناقص الوزن Using Weight Decay

أدناه، نجري مع تناقص كبير في الوزن. لاحظ أن خطأ التدريب يزداد ولكن خطأ التتحقق من الصحة يتناقص. هذا هو بالضبط التأثير الذي نتوقعه من التنظيم.

train_scratch(3)

L2 norm of w: 0.0014634879771620035



Concise Implementation 3.7.4

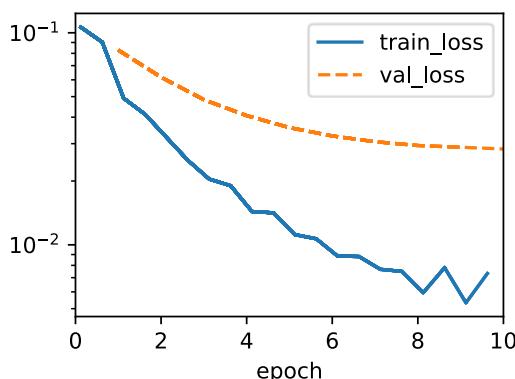
نظرًا لأن تناقص الوزن موجود في كل مكان في تحسين الشبكة العصبية، فإن إطار التعلم العميق يجعله مناسًّا بشكل خاص، حيث يدمج تناقص الوزن في خوارزمية التحسين نفسها لسهولة الاستخدام مع أي دالة خطأ.علاوة على ذلك، يخدم هذا التكامل فائدة حسابية، مما يسمح لحيل التنفيذ بإضافة تناقص الوزن إلى الخوارزمية، دون أي نفقات حسابية إضافية. نظرًا لأن جزء تناقص الوزن في التحديث يعتمد فقط على القيمة الحالية لكل معلمة، يجب أن يلمس المحسن كل معلمة مرة واحدة على أي حال.

في الكود التالي، أنشأنا منظم ℓ_2 باستخدام المعامل التشبعي لتناقص الوزن ونطبقه على أوزان الطبقة من خلال وسيلة `.kernel_regularizer`.

```
class WeightDecay(d2l.LinearRegression):
    def __init__(self, wd, lr):
        super().__init__(lr)
        self.save_hyperparameters()
        self.net = tf.keras.layers.Dense(
            1,
            kernel_regularizer=tf.keras.regularizers.l2(wd),
            kernel_initializer=tf.keras.initializers.RandomNormal(0,
            0.01))
    def loss(self, y_hat, y):
        return super().loss(y_hat, y) + self.net.losses
```

تبعد الحبكة مشابهة لتلك التي حدثت عندما نفذنا عملية تناقص الوزن من نقطة الصفر. ومع ذلك، يعمل هذا الإصدار بشكل أسرع وأسهل في التنفيذ، ويصبح الفوائد أكثر وضوحاً عندما تعالج مشاكل أكبر ويصبح هذا العمل روتينياً بشكل أكبر.

```
model = WeightDecay(wd=3, lr=0.01)
model.board.yscale='log'
trainer.fit(model, data)
print('L2 norm of w:',
float(12_penalty(model.get_w_b()[0])))
```



حتى الآن، تطرقنا فقط إلى فكرة واحدة عما يشكل دالة خطية بسيطة. علاوة على ذلك، ما الذي يشكل دالة غير خطية بسيطة يمكن أن يكون سؤالاً أكثر تعقيداً. على سبيل المثال، يسمح الخطيّة في سياق غير خطّي. لسوء الحظ، تميل الخوارزميات المستندة إلى RKHS إلى التوسيع بشكل سبيئ في البيانات الكبيرة عالية الأبعاد. في هذا الكتاب، غالباً ما نعتمد الاستدلال المشترك حيث يتم تطبيق انحلال (تناقص) الوزن على جميع طبقات الشبكة العميقه.

3.7.5 الملخص

- التّنظيم Regularization هو طريقة شائعة للتعامل مع فرط التجهيز overfitting. تضيف تقيّبات التّنظيم الكلاسيكية مصطلحًا جزائياً penalty term لدالة الخطأ (عند التدريب) لتقليل تعقيد النموذج الذي تم تعلمه.
- أحد الخيارات المحددة للحفاظ على النموذج بسيطاً هو استخدام عقوبة ℓ_2 . هذا يؤدي إلى تناقص الوزن في خطوات التحديث لخوارزمية التدرج الستقائي العشوائي المصغر.
- يتم توفير دالة تناقص الوزن في المُحسّنون من إطار التعلم العميق.

- يمكن أن يكون للمجموعات المختلفة من المعلمات سلوكيات تحديث مختلفة في نفس حلقة التدريب.

3.7.6. التمارين

- جرب قيمة λ في مسألة التقدير في هذا القسم. ارسم دقة التدريب ودقة التحقق من الصحة كدالة. ماذا تلاحظ؟
- استخدم مجموعة التحقق validation set للعثور على القيمة المثلثي لـ λ . هل هي حقا القيمة المثلثي؟ هل هذا مهم؟
- كيف ستبدو معادلات التحديث إذا استخدمنا $|w_i|_1$ بدلاً من $\|w\|^2$ كعقوبة اختيار (تنظيم ℓ_1)؟
- نحن نعلم $w^T w = \|w\|^2$. هل يمكنك العثور على معادلة مماثلة للمصفوفات (راجع معيار Frobenius في القسم 2.3.11)؟
- راجع العلاقة بين خطأ التدريب training error وخطأ التععمق generalization بالإضافة إلى تناقص الوزن weight decay، وزيادة التدريب error training، واستخدام نموذج التعقيد المناسب، ما هي الطرق الأخرى التي يمكنك التفكير بها للتعامل مع فرط التجهيز؟
- في إحصائيات بايز Bayesian statistics، نستخدم ناتج سابق واحتمالية الوصول إلى لاحقة عبر $P(w | x) \propto P(x | w)P(w)$. كيف يمكنك تحديد $P(w)$ مع التنظيم regularization؟

الثباتات العربية الخطية للتحنيف

4

4. الشبكات العصبية الخطية للتصنيف Linear Neural Networks for Classification

الآن بعد أن عملت من خلال جميع الآليات، فأنت جاهز لتطبيق هذه المهارات على أنواع أوسع من المهام. حتى عندما تتجه نحو التصنيف Classification، تظل معظم أعمال السباكة كما هي: تحميل البيانات، وتمريرها عبر النموذج، وتوليد المخرجات، وحساب الخطأ، وأخذ التدرجات فيما يتعلق بالأوزان، وتحديث النموذج. ومع ذلك، فإن الشكل الدقيق للأهداف، ومعلمات طبقة المخرجات، واختيار دالة الخطأ سوف تتكيّف لتلائم إعداد التصنيف.

4.1 انحدار Softmax

في القسم 3.1، قدمنا الانحدار الخطى، والعمل من خلال عمليات التنفيذ من البداية في القسم 3.4 ومرة أخرى باستخدام واجهات برمجة التطبيقات API عالية المستوى لإطار عمل التعلم العميق في القسم 3.5 للقيام بالرفع الثقيل.

الانحدار Regression هو المطرقة التي نصل إليها عندما نريد أن نجيب إلى أي مدى؟ أو كم عددها؟ أسئلة. إذا كنت ترغب في التنبؤ بعدد الدولارات (السعر) التي س يتم بيع منزل بها، أو عدد مرات الفوز التي قد يحققها فريق البيسبول، أو عدد الأيام التي سيبقى فيها المريض في المستشفى قبل الخروج من المستشفى، فمن المحتمل أنك تبحث عن نموذج الانحدار. ومع ذلك، حتى في نماذج الانحدار، هناك فروق مهمة. على سبيل المثال، لن يكون سعر المنزل سالباً أبداً وقد تكون التغيرات في الغالب مرتبطة بسعره الأساسي. على هذا النحو، قد يكون التراجع عن لوغاریتم السعر أكثر فاعلية. وبالمثل، فإن عدد الأيام التي يقضيها المريض في المستشفى هو متغير عشوائي منفصل غير سلبي. على هذا النحو، قد لا تكون المربعات الأقل دلالة نهجاً مثالياً أيضاً. يأتي هذا النوع من النماذج من وقت إلى حدث time-to-event modeling مع مجموعة من المضاعفات الأخرى التي يتم التعامل معها في حقل فرعى متخصص يسمى نمذجة البقاء survival modeling.

وجهة النظر هنا ليست إرباكاً ولكن فقط لإعلامك بأن هناك الكثير من التقدير أكثر من مجرد تقليل الأخطاء التربيعية. وعلى نطاق أوسع، هناك الكثير للتعلم الخاضع للإشراف أكثر من الانحدار. في هذا القسم نركز على مشاكل التصنيف classification problems حيث نضع جانباًكم؟ الأسئلة وبدلاً من ذلك التركيز على أي فئة؟ أسئلة.

- هل ينتمي هذا البريد الإلكتروني إلى مجلد البريد العشوائي أو صندوق البريد الوارد؟
- هل من المرجح أن يقوم هذا العميل بالتسجيل أو عدم الاشتراك في خدمة الاشتراك؟
- هل تصور هذه الصورة حماراً أو كلباً أو قطة أو ديكاً؟

- ما هو الفيلم الذي من المرجح أن يشاهدته أستون بعد ذلك؟
- أي قسم من الكتاب سوف تقرأه بعد ذلك؟

بالعامية، يُطلق ممارسو التعلم الآلي كلمة التصنيف classification لوصف مشكلتين مختلفتين تماماً: (1) تلك التي نهتم فيها فقط بالخصائص الصعبة للأمثلة للفئات categories (الفئات classes)؛ و(2) تلك التي نرغب في إجراء تحصيقات بسيطة، أي لتقدير احتمالية تطبيق كل فئة. يميل التمييز إلى التعتمد جزئياً، لأنه في كثير من الأحيان، حتى عندما نهتم فقط بالمهام الصعبة، ما زلنا نستخدم النماذج التي تقوم بمهام بسيطة.

أكثر من ذلك، هناك حالات يكون فيها أكثر من تسمية label واحدة صحيحة. على سبيل المثال، قد تغطي مقالة إخبارية في نفس الوقت موضوعات الترفيه والأعمال ورحلات الفضاء، ولكن لا تغطي موضوعات الطلب أو الرياضة. وبالتالي، فإن تصنيفها إلى إحدى الفئات المذكورة أعلاه بمفردها لن يكون مفيداً للغاية. تُعرف هذه المشكلة عموماً باسم التصنيف متعدد التسميات multi-label classification. انظر Tsoumakas and Katakis (2007) للحصول على نظرة عامة و Huang et al (2015) لخوارزمية فعالة عند وضع علامات tagging على الصور.

Classification 4.1.1

لتبليل أقدامنا، لنبدأ بمشكلة بسيطة في تصنیف الصور image classification. هنا، يتكون كل إدخال من 2×2 صورة ذات تدرج رمادي. يمكننا تمثيل كل قيمة بكسل باستخدام عدد قياسي واحد، مما يعطينا أربع ميزات x_4, x_3, x_2, x_1 . علاوة على ذلك، لنفترض أن كل صورة تتبع إلى واحدة من بين الفئات "قطة" و "دجاج" و "كلب".

بعد ذلك، علينا أن نختار كيفية تمثيل التسميات labels. لدينا خيارات واضحان. ربما يكون الدافع الأكثر طبيعية هو اختيار $y \in \{1,2,3\}$ ، حيث تمثل الأعداد الصحيحة $\{\text{dog,cat,chicken}\}$ على التوالي. هذه طريقة رائعة لتخزين مثل هذه المعلومات على جهاز الكمبيوتر. إذا كان للفئات بعض الترتيب الطبيعي فيما بينها، على سبيل المثال إذا كنا نحاول التنبؤ ب $\{\text{baby,toddler,adolescent,young adult,adult,geriatric}\}$ ، فقد يكون من المنطقي اعتبار ذلك مشكلة انحدار ترتيبية والاحتفاظ بالتسميات بهذا التنسيق. انظر Moon et al (2010) للحصول على نظرة عامة على أنواع مختلفة من دوال فقدان الترتيب و Beutel et al (2014) لنهج بايزي Bayesian approach الذي يعالج الاستجابات بأكثر من وضع.

بشكل عام، لا تأتي مشاكل التصنيف مع الترتيب الطبيعي بين الفئات classes: categorical data. ابتكر الإحصائيون منذ فترة طويلة طريقة بسيطة لتمثيل البيانات الفئوية categorical data: التشفير الواحد الساخن one-hot encoding. الترميز الواحد الساخن هو متوجه يحتوي على

العديد من المكونات مثل الفئات الموجودة لدينا. يتم تعين المكون المقابل لفئة مثل معين على 1 ويتم تعين جميع المكونات الأخرى على 0. في حالتنا، سيكون التسمية y متوجهًا ثلاثي الأبعاد، مع $(1,0,0)$ المقابل لـ "قطة" و $(0,1,0)$ "دجاجة" و $(0,0,1)$ "كلب":

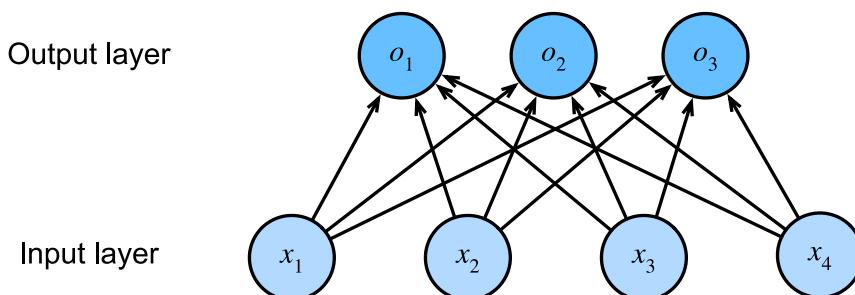
$$y \in \{(1,0,0), (0,1,0), (0,0,1)\}.$$

4.1.1.1. النموذج الخطي Linear Model

لتقدير الاحتمالات الشرطية المرتبطة بجميع الفئات الممكنة، نحتاج إلى نموذج بمحركات متعددة، واحدة لكل فئة. لمعالجة التصنيف بالنماذج الخطية، سنحتاج إلى العديد من الدوال الأفينية affine functions مثل عدد المحركات. بالمعنى الدقيق للكلمة، نحتاج فقط إلى واحد أقل، لأن الفئة الأخيرة يجب أن تكون الفرق بين 1 ومجموع الفئات الأخرى ولكن لأسباب التناقض symmetry نستخدم معلمات زائدة عن الحاجة. كل ناتج يتواافق مع الدالة الأفينية الخاصة به. في حالتنا، نظرًا لأن لدينا 4 ميزات و3 فئات إخراج محتملة، نحتاج إلى 12 قيمة قياسية Scalar لتمثيل الأوزان (w مع الرموز المنخفضة subscripts)، و3 مقاييس لتمثيل التحيزات (b مع الرموز المنخفضة subscripts). هذه العوائد:

$$\begin{aligned} o_1 &= x_1 w_{11} + x_2 w_{12} + x_3 w_{13} + x_4 w_{14} + b_1, \\ o_2 &= x_1 w_{21} + x_2 w_{22} + x_3 w_{23} + x_4 w_{24} + b_2, \\ o_3 &= x_1 w_{31} + x_2 w_{32} + x_3 w_{33} + x_4 w_{34} + b_3. \end{aligned}$$

يظهر مخطط الشبكة العصبية المقابل في الشكل 4.1.1. كما هو الحال في الانحدار الخطى، نستخدم شبكة عصبية أحادية الطبقه single-layer neural network. ونظرًا لأن حساب كل ناتج o_1, o_2, o_3 ، ويعتمد على جميع المدخلات، x_1, x_2, x_3 و x_4 ، يمكن أيضًا وصف طبقه fully connected layer .المخرجات بأنها طبقه متصلة بالكامل



الشكل 4.1.1 انحدار Softmax عبارة عن شبكة عصبية أحادية الطبقه.

للحصول على تدوين أكثر إيجازاً، نستخدم المتجهات والمصفوفات: $\mathbf{o} = \mathbf{Wx} + \mathbf{b}$ وهي أكثر ملاءمة للرياضيات والرموز. لاحظ أننا جمعنا كل أوزاناني مصفوفة 4×3 وجميع التحيزات في متوجه $\mathbf{b} \in \mathbb{R}^3$.

4.1.1.2 Softmax

بافتراض دالة خطأ مناسبة، يمكننا أن نحاول، مباشرة، تقليل الاختلاف بين \mathbf{o} والتسميات \mathbf{y} . بينما اتضح أن معالجة التصنيف على أنه مشكلة انحدار ذات قيمة متوجهية vector-valued regression تعمل بشكل جيد مدهش، إلا أنها مع ذلك تفتقر إلى الطرق التالية:

- ليس هناك ما يضمن أن مجموع المخرجات o_i يصل إلى 1 بالطريقة التي تتوقع أن تتصرف بها الاحتمالات.
- ليس هناك ما يضمن أن الناتج o_i هي حتى غير سالبة، حتى لو كانت مخرجاتها تصل إلى 1، أو أنها لا تتجاوز 1.

كلا الجانبين يجعل مشكلة التقدير صعبة الحل والحل هش للغاية للقيم المتطرفة outliers. على سبيل المثال، إذا افترضنا أن هناك تبعية خطية linear dependency موجبة بين عدد غرف النوم واحتمال قيام شخص ما بشراء منزل، فقد يتجاوز الاحتمال 1 عندما يتعلق الأمر بشراء قصر! على هذا النحو، نحن بحاجة إلى آلية "squish" للمخرجات.

هناك العديد من الطرق التي يمكننا من خلالها تحقيق هذا الهدف. على سبيل المثال، يمكننا أن نفترض أن المخرجات \mathbf{o} هي إصدارات تالفة من \mathbf{y} ، حيث يحدث التلف عن طريق إضافة ضوضاء مستمدة من التوزيع الطبيعي. بمعنى آخر، $\mathbf{y} = \mathbf{o} + \epsilon$ حيث $(0, \sigma^2) \sim \mathcal{N}(\epsilon_i)$. هذا هو ما يسمى بنموذج الاختبار probit model، الذي قدمه Fechner لأول مرة (1860). في حين أنها جذابة، فإنها لا تعمل بشكل جيد أو تؤدي إلى مشكلة تحسين لطيفة بشكل خاص، عند مقارنتها بـ softmax.

هناك طريقة أخرى لتحقيق هذا الهدف (ولضممان اللاسلبية nonnegativity) وهي استخدام دالة أسيّة $P(y_i = o_i) \propto \exp(o_i)$. هذا بالفعل يفي بمتطلبات زيادة احتمال الطبقة الشرطية مع زيادة o_i ، وهو رتب monotonic ، وجميع الاحتمالات غير سالبة. يمكننا بعد ذلك تحويل هذه القيم بحيث يتم جمعها إلى 1 بقسمة كل منها على مجموعها. هذه العملية تسمى التسوية normalization: يمنحك وضع هاتين القطعتين معًا دالة softmax.

$$\hat{\mathbf{y}} = \text{softmax}(\mathbf{o}) \text{ where } \hat{y}_i = \frac{\exp(o_i)}{\sum_j \exp(o_j)}.$$

لاحظ أن أكبر إحدائي لـ $\mathbf{0}$ يتوافق مع الفئة الأكثر احتمالاً وفقاً لـ $\hat{\mathbf{y}}$. علاوة على ذلك، نظرًا لأن عملية softmax تحافظ على الترتيب بين وسيطاتها، فإننا لا نحتاج إلى حساب softmax لتحديد الفئة التي تم تخصيص أعلى احتمالية لها.

$$\underset{j}{\operatorname{argmax}} \hat{\mathbf{y}}_j = \underset{j}{\operatorname{argmax}} o_j.$$

تعود فكرة softmax إلى جيبيس Gibbs، الذي قام بتكييف الأفكار من الفيزياء (Gibbs، 1902). منذ زمن بعيد، استخدم بولتزمان، والد الديناميكا الحرارية الحديثة، هذه الحيلة لنجدجة التوزيع على حالات الطاقة في جزيئات الغاز. على وجه الخصوص، اكتشف أن انتشار حالة من الطاقة في مجموعة ديناميكية حرارية، مثل الجزيئات في الغاز، يتاسب مع $\exp(-E/kT)$. هنا، طاقة الحالة، T هي درجة الحرارة، و k ثابت بولتزمان. عندما يتحدث الإحصائيون عن زيادة أو خفض "درجة حرارة" نظام إحصائي، فإنهم يشيرون إلى التغيير لصالح حالات الطاقة المنخفضة أو الأعلى. باتباع فكرة جيبيس، الطاقة تساوي الخطأ. تستخدم النماذج القائمة على الطاقة (Ranzato et al., 2007) وجهاً النظر هذه عند وصف المشكلات في التعلم العميق.

4.1.1.3. Vectorization

لتحسين الكفاءة الحسابية، نقوم بتوجيه الحسابات في الدفعات الصغيرة من البيانات. افترض أننا حصلنا على دفعات صغيرة $\mathbf{X} \in \mathbb{R}^{n \times d}$ من n الميزات ذات الأبعاد (عدد المدخلات) d . علاوة على ذلك، افترض أن لدينا q فئات في المخرجات. ثم الأوزان تحقق $\mathbf{W} \in \mathbb{R}^{d \times q}$ والتحيز $\mathbf{b} \in \mathbb{R}^{1 \times q}$.

$$\begin{aligned}\mathbf{0} &= \mathbf{X}\mathbf{W} + \mathbf{b}, \\ \hat{\mathbf{Y}} &= \text{softmax}(\mathbf{0}).\end{aligned}$$

يؤدي هذا إلى تسريع العملية السائد في ضرب المصفوفة–المصفوفة $\mathbf{X}\mathbf{W}$. علاوة على ذلك، نظرًا لأن كل صفي في \mathbf{X} يمثل مثلاً للبيانات، يمكن حساب عملية softmax نفسها في اتجاه الصف rowwise: لكل صف من $\mathbf{0}$ ، ثم قم بتسويتها بالمجموع. لاحظ، مع ذلك، أنه يجب توخي الحذر لتجنب الأس وأخذ اللوغاريتمات ذات الأعداد الكبيرة، لأن هذا يمكن أن يسبب تجاوزاً رقمياً أو تدنياً. تعني أطر التعلم العميق بهذا الأمر تلقائياً.

4.1.2. دالة الخطأ

الآن بعد أن أصبح لدينا تعين من الميزات \mathbf{x} إلى الاحتمالات $\hat{\mathbf{y}}$ ، نحن بحاجة إلى طريقة لتحسين دقة هذا التعين. سوف نعتمد على تقدير الاحتمال الأقصى maximum likelihood

estimation، وهو نفس المفهوم الذي واجهناه عند تقديم تبرير احتمالي لمتوسط خسارة الخطأ التربيعي في القسم 3.1.3.

Log-Likelihood 4.1.2.1

تعطينا دالة softmax متوجهاً $\hat{\mathbf{y}}$ ، والتي يمكننا تفسيرها على أنها احتمالات مشروطة (مقدمة) لكل فئة، مع الأخذ في الاعتبار أي مدخلات \mathbf{x} ، مثل $(\mathbf{x} | \hat{\mathbf{y}}_1 = P(y = \text{cat} | \mathbf{x}))$. فيما يلي نفترض أنه بالنسبة لمجموعة البيانات ذات الميزات \mathbf{X} ، يتم تمثيل التسميات \mathbf{Y} باستخدام متوجه تسمية ترميز واحد ساخن one-hot encoding. يمكننا مقارنة التقديرات بالواقع من خلال التحقق من مدى احتمالية أن تكون الفئات الفعلية وفقاً لنموذجنا، مع مراعاة الميزات:

$$P(\mathbf{Y} | \mathbf{X}) = \prod_{i=1}^n P(\mathbf{y}^{(i)} | \mathbf{x}^{(i)}).$$

يُسمح لنا باستخدام التحليل إلى عوامل factorization لأننا نفترض أن كل تسمية مرسومة بشكل مستقل عن التوزيع $P(\mathbf{y} | \mathbf{x}^{(i)})$ الخاص بها. نظرًا لأن تعظيم منتج المصطلحات أمر غير ملائم، فإننا نأخذ اللوغاريتم السالب للحصول على المشكلة المكافئة لتقليل log-likelihood:

$$-\log P(\mathbf{Y} | \mathbf{X}) = \sum_{i=1}^n -\log P(\mathbf{y}^{(i)} | \mathbf{x}^{(i)}) = \sum_{i=1}^n l(\mathbf{y}^{(i)}, \hat{\mathbf{y}}^{(i)}),$$

أين لأي زوج من التسمية \mathbf{y} والتنبؤ $\hat{\mathbf{y}}$ بالنموذج على الفئات q ، فإن دالة الخطأ l هي

$$l(\mathbf{y}, \hat{\mathbf{y}}) = -\sum_{j=1}^q y_j \log \hat{y}_j.$$

لأسباب تم توضيحيها لاحقًا، فإن دالة الخطأ (4.1.8) تسمى عادةً خطأ الانتروبيا المتقطعة cross-entropy loss. نظرًا لأن \mathbf{y} متوجه واحد ساخن للطول q ، فإن المجموع على جميع الإحداثيات j يختفي لجميع المصطلحات باستثناء مصطلح واحد. لاحظ أن الخطأ $l(\mathbf{y}, \hat{\mathbf{y}})$ يحدها من الأسفل بواسطة 0 حيث \hat{y}_j هو متوجه احتمالي: لا يوجد إدخال واحد أكبر من 1، وبالتالي لا يمكن أن يكون اللوغاريتم السالب أقل من 0؛ فقط إذا $0 = l(\mathbf{y}, \hat{\mathbf{y}})$ توقعنا التسمية الفعلية على وجه اليقين certainty. لا يمكن أن يحدث هذا أبدًا لأن إعداد محدود للأوزان لأن أخذ إخراج softmax نحو 1 يتطلب أخذ المدخلات المقابلة o_i إلى ما لا نهاية (أو جميع المخرجات o_j الأخرى لـ $j \neq i$ إلى اللانهاية السالبة). حتى إذا كان بإمكان نموذجنا تعين

احتمال إخراج 0، فإن أي خطأ يرتكب عند تعين مثل هذه الثقة العالية سيؤدي إلى خطأ لا نهائي $(-\log 0 = \infty)$.

Softmax and Cross-Entropy Loss 4.1.2.2 وخطأ الانتروبيا

نظرًا لأن دالة softmax وخطأ الانتروبيا المقابلة شائعة جدًا، فمن الجدير فهم كيفية حسابها بشكل أفضل قليلاً. إدخال (4.1.3) في تعريف الخطأ (4.1.8) وباستخدام تعريف softmax نحصل عليه:

$$\begin{aligned} l(\mathbf{y}, \hat{\mathbf{y}}) &= -\sum_{j=1}^q y_j \log \frac{\exp(o_j)}{\sum_{k=1}^q \exp(o_k)} \\ &= \sum_{j=1}^q y_j \log \sum_{k=1}^q \exp(o_k) - \sum_{j=1}^q y_j o_j \\ &= \log \sum_{k=1}^q \exp(o_k) - \sum_{j=1}^q y_j o_j. \end{aligned}$$

لفهم ما يجري بشكل أفضل قليلاً، ضع في الاعتبار المشتق فيما يتعلق بأي لوغاريتم o_j . نحن نحصل

$$\partial_{o_j} l(\mathbf{y}, \hat{\mathbf{y}}) = \frac{\exp(o_j)}{\sum_{k=1}^q \exp(o_k)} - y_j = \text{softmax}(\mathbf{o})_j - y_j.$$

بمعنى آخر، المشتق derivative هو الفرق بين الاحتمال المحدد بواسطة نموذجنا، كما يعبر عنه عملية SoftMax، وما حدث بالفعل، على النحو المعبر عنه بالعناصر الموجودة في متوجه التسمية الساخنة الواحدة. بهذا المعنى، فهو مشابه جدًا لمارأينا في الانحدار، حيث كان التدرج هو الفرق بين الملاحظة y والتقدير \hat{y} . هذه ليست صدفة. في أي نموذج عائلي أسي، يتم إعطاء تدرجات احتمالية السجل من خلال هذا المصطلح على وجه التحديد. هذه الحقيقة تجعل حوسبة التدرجات سهلة في الممارسة.

الآن ضع في اعتبارك الحالة التي نلاحظ فيها ليس فقط نتيجة واحدة ولكن توزيعًا كاملاً على النتائج. يمكننا استخدام نفس التمثيل السابق للتسمية \mathbf{y} . الاختلاف الوحيد هو أنه بدلاً من المتوجه الذي يحتوي على مدخلات ثنائية فقط، على سبيل المثال (0,0,1)، لدينا الآن متوجه احتمالي عام، على سبيل المثال (0.1,0.2,0.7). الرياضيات التي استخدمناها سابقاً لتحديد الخطأ (4.1.8) لا تزال تعمل بشكل جيد، فقط أن التفسير أكثر عمومية. إنها القيمة المتوقعة للخطأ cross-entropy loss وهو توزيع على التسميات. يسمى هذه الخطأ بخطأ الانتروبيا المتقاطعة

واحد من أكثر الأخطاء استخداماً لمشاكل التصنيف. يمكننا إزالة الغموض عن الاسم من خلال تقديم أساسيات نظرية المعلومات فقط. باختصار، إنه يقيس عدد البتات لترميز ما نراه y بالنسبة لما نتوقع حدوثه \hat{y} . نقدم شرحاً أساسياً جداً فيما يلي. لمزيد من التفاصيل حول نظرية المعلومات انظر (MacKay and Mac Kay, 1999) أو (Cover and Thomas, 2003).

4.1.3 أساسيات نظرية المعلومات

تستخدم العديد من أوراق التعلم العميق الحدس والمصطلحات من نظرية المعلومات information theory لفهمها، تحتاج إلى لغة مشتركة. هذا هو دليل البقاء على قيد الحياة. تعامل نظرية المعلومات مع مشكلة ترميز المعلومات وفك تشفيرها ونقلها ومعالجتها (المعروفة أيضاً باسم البيانات data).

4.1.3.1 Entropy

الفكرة المركزية في نظرية المعلومات هي تحديد كمية المعلومات الواردة في البيانات. هذا يضع حداً لقدرنا على ضغط البيانات. بالنسبة للتوزيع P ، يتم تعريف الانتروبيا على النحو التالي:

$$H[P] = \sum_j -P(j)\log P(j).$$

تنص إحدى النظريات الأساسية لنظرية المعلومات على أنه من أجل ترميز البيانات المستمدة عشوائياً من التوزيع P ، تحتاج على الأقل $H[P]$ إلى "nats" لتشفيتها (Shannon, 1948). إذا كنت تتساءل ما هو "nat" فهو مكافئ للبت ولكن عند استخدام رمز مع الأساس e بدلاً من واحد مع الأساس 2. وبالتالي، فإن nat واحد هو $1.44 \approx \frac{1}{\log(2)}$ بت.

4.1.3.2 Surprisal

قد تسأعل ما علاقة الضغط بالتبؤ. تخيل أن لدينا دفقة من البيانات التي نريد ضغطها. إذا كان من السهل علينا دائمًا توقع الرمز المميز التالي، فمن السهل ضغط هذه البيانات. خذ المثال حيث يأخذ كل رمز مميز في الدفق نفس القيمة دائمًا. هنا دفق بيانات ممل للغاية! وهو ليس مملاً فحسب، بل يسهل توقعه أيضًا. نظرًا لأنهما دائمًا متماثلان، فلا يتغير علينا نقل أي معلومات لتوصيل محتويات الدفق. سهل التنبؤ، سهل الضغط.

ومع ذلك، إذا لم نتمكن من التنبؤ بكل حدث تماماً، فقد نتفاجأ أحيانًا. تكون دهشتنا أكبر عندما خصصنا حدثاً احتمالية أقل. استقر كلويد شانون على $-\log \frac{1}{P(j)}$ لتقدير مفاجأة المرء في مراقبة حدث ما ز بعد أن منحه احتمالاً (j). إن الانتروبيا المحددة في (4.1.11) هي

إذن المفاجأة المتوقعة عندما يقوم أحدهم بتعيين الاحتمالات الصحيحة التي تتطابق بالفعل مع عملية توليد البيانات.

4.1.3.3 إعادة النظر عبر الانتروبيا Cross-Entropy Revisited

لذا، إذا كان الانتروبيا هو مستوى المفاجأة الذي يختبره شخص يعرف الاحتمال الحقيقي ، فقد تتساءل ، ما هو الانتروبيا المتقاطعة cross-entropy؟ إن الانتروبيا المتقاطعة من P إلى Q ، المشار إليها ($H(P, Q)$ ، هي المفاجأة المتوقعة لمراقب ذي احتمالات ذاتية عند رؤية البيانات التي تم إنشاؤها بالفعل وفقاً للاحتمالات P . هذا معطى من قبل -
$$H(P, Q) \stackrel{\text{def}}{=} \sum_j P(j) \log Q(j)$$
. يتم تحقيق أدنى قدر ممكن من الانتروبيا عندما $Q = P$. في هذه الحالة، فإن الانتروبيا المتقاطعة P إلى Q هي $H(P, P) = H(P)$.

باختصار، يمكننا التفكير في هدف التصنيف عبر الانتروبيا بطريقتين: (1) تعظيم احتمالية البيانات المرصودة؛ و (2) لتقليل مفاجأتنا (وبالتالي عدد البتات) المطلوبة لتوصيل التسميات labels.

4.1.4 الملخص والمناقشة

في هذا القسم، واجهنا أول دالة خطأ غير بدائية، مما يسمح لنا بتحسين مساحات الإخراج المنفصلة. كان المفتاح في تصميمه هو أننا اتخذنا نهجاً احتمالياً، حيث تعاملنا مع الفئات المنفصلة على أنها حالات سحب من توزيع احتمالي. أكثر جانبي، واجهنا softmax، وهي دالة تنشيط مرية تحول مخرجات طبقة الشبكة العصبية العادي إلى توزيعات احتمالية منفصلة صالحة. لقد رأينا أن مشتق خطأ الانتروبيا المتقاطعة عندما يقتربن بـ softmax يتصرف بشكل مشابه جداً لمشتق الخطأ التربيعي، أي عن طريقأخذ الفرق بين السلوك المتوقع والتنبؤ به. وبينما كنا قادرين فقط على خدش السطح ذاته، واجهنا روابط مثيرة للفيزياء الإحصائية ونظرية المعلومات.

في حين أن هذا كافٍ ليأخذك في طريقك، ونأمل أن يكون كافياً لإثارة شهيتك، إلا أننا بالتأكيد نغوص بعمق هنا. من بين أمور أخرى، تخططنا الاعتبارات الحسابية. على وجه التحديد، بالنسبة لأي طبقة متصلة بالكامل بها d مدخلات و q مخرجات، المعلمات والتكلفة الحسابية هي $\mathcal{O}(dq)$ ، والتي يمكن أن تكون باهظة في الممارسة العملية. لحسن الحظ، يمكن تقليل تكلفة تحويل المدخلات d إلى مخرجات q من خلال التقرير approximation والضغط compression. على سبيل المثال، يستخدم Deep Fried Convnets (Yang et al., 2015) مجموعة من التباديل وتحويلات فورييه والقياس لتقليل التكلفة من الترميز إلى اللوغاريتمية الخطية. تعمل تقنيات مماثلة من أجل تقرير مصفوفة هيكلية أكثر تقدماً (Sindhwani et al., 2015). أخيراً، يمكننا استخدام تحليلات تشبه Quaternion لتقليل التكلفة إلى $\mathcal{O}(\frac{dq}{n})$ ، مرة أخرى،

إذاً كنا على استعداد لمقاييس قدر ضئيل من الدقة بتكلفة الحساب والتخزين (Zhang et al., 2021) بناءً على عامل الضغط n . هذا مجال نشط للبحث. ما يجعل الأمر صعباً هو أننا لا نسعى بالضرورة إلى التمثيل الأكثر إحكاماً أو أصغر عدد من عمليات الفاصلة العائمة floating point ولكن بالأحرى للحل الذي يمكن تفديه بكفاءة أكبر على وحدات معالجة الرسومات الحديّة GPUs.

4.1.5. التمارين

1. يمكننا استكشاف العلاقة بين العائلات الأُسيّة و softmax بعمق أكبر.
 1. احسب المشتق الثاني لخط الانتروبيا \hat{y} , y من أجل softmax.
 2. احسب تباين التوزيع الذي قدمه softmax (\mathbf{o}) وأظهر أنه يطابق المشتق الثاني المحسوب أعلاه.
2. افترض أن لدينا ثلاثة فئات تحدث باحتمالية متساوية، أي متوجه الاحتمال هو $(\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$
 1. ما هي المشكلة إذا حاولنا تصميم كود ثنائي لها؟
 2. هل يمكنك تصميم كود أفضل؟ تلميح: ماذا يحدث إذا حاولنا ترميز ملاحظتين مستقلتين؟ ماذا لو قمنا بترميز n ملاحظات بشكل مشترك؟
3. عند إرسال إشارات ترميز عبر سلك مادي، لا يستخدم المهندسون دائمًا رموزًا ثنائية. على سبيل المثال، يستخدم PAM-3 ثلاثة مستويات للإشارة $\{0, 1, 2\}$ بدلاً من مستويين $\{0, 1\}$. كم عدد الوحدات الثلاثية ternary units التي تحتاجها لإرسال عدد صحيح في النطاق $\{0, \dots, 7\}$ ؟ لماذا قد تكون هذه فكرة أفضل من حيث الإلكترونيات؟
 4. يستخدم نموذج برادلي تيري Bradley-Terry model نموذجاً لوجيستياً logistic model للتقطّع التفضيلات. لكي يختار المستخدم بين التفاح والبرتقال، يفترض المرء أن الدرجات $o_{apple} > o_{orange}$. متطلباتنا هي أن الدرجات الأكبر يجب أن تؤدي إلى احتمالية أعلى في اختيار العنصر المرتبط وأن العنصر الحاصل على أكبر درجة هو العنصر الأكثر احتمالية لاختياره (Bradley and Terry, 1952).
1. إثبت أن softmax يلبي هذا المطلب.
 2. ماذا يحدث إذا كنت تريد السماح بخيار افتراضي لا يختار فيه التفاح ولا البرتقال؟ تلميح: الآن لدى المستخدم 3 اختيارات.
5. تشتق Softmax اسمها من التعين التالي:

$$\text{RealSoftMax}(a, b) = \log(\exp(a) + \exp(b))$$
 1. اثبت $\text{RealSoftMax}(a, b) > \max(a, b)$

2. ما مدى صغر حجم الفرق بين الدالتين؟ تلميح: بدون فقدان التعميم يمكنك ضبط $a \geq b$ و $b = 0$.
3. إثبت أن $\lambda^{-1} \text{RealSoftMax}(\lambda a, \lambda b) > 0$ بشرط $\lambda > 0$.
4. أظهر ذلك $\lambda \rightarrow \infty$ لـ $\text{RealSoftMax}(\lambda a, \lambda b) \rightarrow \max(a, b)$ لدينا.
5. كيف تبدو soft-min ؟
6. قم بتوسيع هذا إلى أكثر من رقمين.
6. الدالة $x_i \stackrel{\text{def}}{=} \log \sum_i \exp$ يشار إليها أحياناً أيضاً باسم log-partition function.
1. إثبت أن الدالة محدبة convex. تلميح: للقيام بذلك، استخدم حقيقة أن المشتق الأول يرقي إلى الاحتمالات من دالة softmax وأظهر أن المشتق الثاني هو التباين variance.
2. أظهر أن g هي الترجمة ثابتة invariant، أي $g(\mathbf{x} + b) = g(\mathbf{x})$.
3. ماذا يحدث إذا كانت بعض الإحداثيات كبيرة جداً؟ ماذا يحدث إذا كانوا جميعاً صغاراً جداً؟
4. أظهر أنه إذا اخترنا $b = \max_i x_i$ سنتهي بتطبيق مستقر عدديًّا.
7. افترض أن لدينا بعض التوزيع الاحتمالي P . لنفترض أننا اخترنا توزيعاً آخر Q باستخدام $Q(i) \propto P(i)^\alpha$.
1. أي اختيار لـ α يتوافق مع مضاعفة درجة الحرارة؟ أي خيار يتوافق مع النصف؟
2. ماذا يحدث إذا تركنا درجة الحرارة تتقارب converge إلى 0؟
3. ماذا يحدث إذا تركنا درجة الحرارة تتقارب إلى ∞ ؟

4.2 مجموعة بيانات تصنيف الصور The Image Classification Dataset

إحدىمجموعات البيانات Dataset المستخدمة على نطاق واسع لتصنيف الصور هي مجموعة بيانات MNIST، (LeCun et al، 1998) للأرقام المكتوبة بخط اليد. في وقت إصداره في التسعينيات، شكل تحدياً هائلاً لمعظم خوارزميات التعلم الآلي، ويتألف من 60.000 صورة بدقة بكسل (بالإضافة إلى مجموعة بيانات اختبار من 10000 صورة). لوضع الأمور في نصابها الصحيح، في ذلك الوقت، تم اعتبار 5 Sun SPARCStation بسعة 64 ميجابايت من ذاكرة الوصول العشوائي الهائلة و5 MFLOPs من أحدث المعدات للتعلم الآلي في مختبرات AT&T Bell في عام 1995. كان تحقيق دقة عالية في التعرف على الأرقام بمثابة المكون الرئيسي في أتمتة فرز الرسائل لـ USPS في التسعينيات. الشبكات العميقة مثل LeNet-5 (LeCun et al، 1995) والات المتوجهات الداعمة support vector machines (SVM) مع الثوابt مع variances

tangent distance classifiers (Schölkopf et al. 1996) ، ومصنفات المسافة المماسية (Simard et al. 1998) كلها سمحت بالوصول إلى معدلات خطأ أقل من 1%.

لأكثر من عقد من الزمان، عملت MNIST كنقطة مرجعية لمقارنة خوارزميات التعلم الآلي. على الرغم من أنه كان يعمل بشكل جيد كمجموعة بيانات معيارية، إلا أن النماذج البسيطة وفقاً لمعايير اليوم تحقق دقة تصنيف تزيد عن 95٪، مما يجعلها غير مناسبة للتمييز بين النماذج الأقوى والنماذج الأضعف. أكثر من ذلك، تسمح مجموعة البيانات بمستويات عالية جداً من الدقة، لا تُرى عادةً في العديد من مشكلات التصنيف. هذا التطوير الخوارزمي المنحرف نحو عائلات معينة من الخوارزميات التي يمكن أن تستفيد منمجموعات البيانات النظيفة clean datasets، مثل أساليب المجموعة النشطة وخوارزميات المجموعة النشطة التي تسعى للحد من الحدود. اليوم، تعمل MNIST ب بشابة فحوصات سلامة أكثر من كونها معياراً. تشكل شبكة ImageNet كبيرة جداً بالنسبة للعديد من الأمثلة والرسوم التوضيحية في هذا الكتاب، حيث سيستغرق التدريب وقتاً طويلاً لجعل الأمثلة تفاعلية. كبديل، سنركز مناقشتنا في الأقسام القادمة على مجموعة بيانات Fashion-MNIST المتباينة نوعياً، ولكن الأصغر بكثير (Xiao et al. 2017)، والتي تم إصدارها في عام 2017. وهي تحتوي على صور لعشر فئات من الملابس بدقة 28×28 بكسل.

```
%matplotlib inline
import time
import tensorflow as tf
from d2l import tensorflow as d2l

d2l.use_svg_display()
```

4.2.1. تحميل مجموعة البيانات Loading the Dataset

نظرًا لأنها مجموعة بيانات مستخدمة بشكل متكرر، فإن جميع الأطر الرئيسية توفر إصدارات معالجة مسبقاً منها. يمكننا تنزيل مجموعة بيانات Fashion-MNIST وقراءتها في الذاكرة باستخدام دوال إطار العمل المضمنة.

```
class FashionMNIST(d2l.DataModule): #@save
    def __init__(self, batch_size=64, resize=(28, 28)):
        super().__init__()
        self.save_hyperparameters()
        self.train, self.val =
tf.keras.datasets.fashion_mnist.load_data()
```

يتكون Fashion-MNIST من صور من 10 فئات، كل منها ممثلة بـ 6000 صورة في مجموعة بيانات التدريب training dataset و1000 في مجموعة بيانات الاختبار test dataset. تُستخدم مجموعة بيانات الاختبار لتقييم أداء النموذج (لا يجب استخدامها للتدريب). وبالتالي، تحتوي مجموعة التدريب ومجموعة الاختبار على 60.000 و10000 صورة على التوالي.

```
data = FashionMNIST(resize=(32, 32))
len(data.train[0]), len(data.val[0])
```

الصور ذات تدرج رمادي وترقيتها إلى 32×32 بكسل في الدقة أعلى. هذا مشابه لمجموعة بيانات MNIST الأصلية التي تتكون من صور (ثنائية) بالأبيض والأسود. لاحظ، مع ذلك، أن معظم بيانات الصور الحديثة تحتوي على 3 قنوات (أحمر، أخضر، أزرق) وصور فائقة الطيف hyperspectral images يمكن أن تحتوي على أكثر من 100 قناة (مستشعر HyMap به 126 قناة). حسب الاصطلاح، نقوم بتخزين الصورة كموتر $w \times h \times c$ ، حيث يكون c عدد قنوات اللون، h هو الارتفاع و w هو العرض.

```
data.train[0][0].shape
(28, 28)
```

فئات Fashion-MNIST لها أسماء مفهومة من قبل الإنسان. يتم تحويل دالة الملاحة convenience function التالية بين التسميات الرقمية وأسمائها.

```
@d2l.add_to_class(FashionMNIST) #@save
def text_labels(self, indices):
    """Return text labels."""
    labels = ['t-shirt', 'trouser', 'pullover', 'dress',
    'coat',
              'sandal', 'shirt', 'sneaker', 'bag',
    'ankle boot']
    return [labels[int(i)] for i in indices]
```

4.2.2. قراءة الدفعات الصغيرة

لجعل حياتنا أسهل عند القراءة منمجموعات التدريب والاختبار، نستخدم مكرر البيانات المدمج built-in data iterator بدلاً من إنشاء واحد من البداية. تذكر أنه في كل تكرار، يقرأ مكرر البيانات دفعه صغيرة من البيانات ذات الحجم batch_size. نقوم أيضًا بترتيب الأمثلة عشوائيًا لمكرر بيانات التدريب.

```
@d2l.add_to_class(FashionMNIST) #@save
def get_dataloader(self, train):
    data = self.train if train else self.val
    process = lambda X, y: (tf.expand_dims(X, axis=3) /
255,
```

```

    tf.cast(y, dtype='int32'))
    resize_fn = lambda X, y:
(tf.image.resize_with_pad(X, *self.resize), y)
    shuffle_buf = len(data[0]) if train else 1
    return
tf.data.Dataset.from_tensor_slices(process(*data)).batch(
(

```

self.batch_size).map(resize_fn).shuffle(shuffle_buf)
 لمعرفة كيفية عمل ذلك، دعنا نحمل مجموعة صغيرة minibatch من الصور عن طريق استدعاء طريقة `train_dataloader` المضافة حديثاً. يحتوي على 64 صورة.

```

X, y = next(iter(data.train_dataloader()))
print(X.shape, X.dtype, y.shape, y.dtype)
(64, 32, 32, 1) <dtype: 'float32'> (64,) <dtype:
'int32'>

```

دعونا نلقي نظرة على الوقت المستغرق لقراءة الصور. على الرغم من أنه محمول مدمج-`built-in` loader، إلا أنه ليس سريعاً للغاية. ومع ذلك، يعد هذا كافياً لأن معالجة الصور باستخدام شبكة عميقة تستغرق وقتاً أطول قليلاً. ومن ثم، فمن الجيد بما يكفي ألا يكون تدريب الشبكة مقيداً بإدخال معلومات.

```

tic = time.time()
for X, y in data.train_dataloader():
    continue
f'{time.time() - tic:.2f} sec'
'0.82 sec'

```

4.2.3 التمثيل البياني

سنستخدم مجموعة بيانات Fashion-MNIST كثيراً. يمكن استخدام دالة ملائمة `show_images` لتمثيل البياني للصور والتسميات المرتبطة بها. تفاصيل تنفيذه مؤجلة إلى الملحق.

```

def show_images(imgs, num_rows, num_cols, titles=None,
scale=1.5): #@save
    """Plot a list of images."""
    raise NotImplementedError

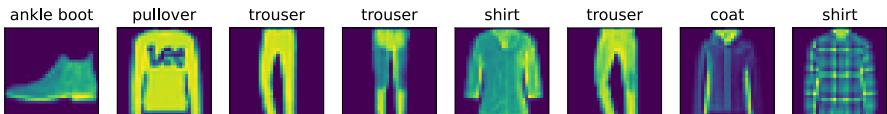
```

دعونا نستخدمها بشكل جيد. بشكل عام، من الجيد تمثيل وفحص البيانات التي تتدرّب عليها. إن البشر بارعون جداً في اكتشاف الجوانب غير المعتادة، وبالتالي، فإن التمثيل البياني بمثابة حماية

إضافية ضد الأخطاء في تصميم التجارب. فيما يلي الصور والتسميات المقابلة لها (في النص) للأمثلة القليلة الأولى في مجموعة بيانات التدريب.

```
@d2l.add_to_class(FashionMNIST) #@save
def visualize(self, batch, nrows=1, ncols=8, labels=[]):
    X, y = batch
    if not labels:
        labels = self.text_labels(y)
    d2l.show_images(tf.squeeze(X), nrows, ncols,
titles=labels)

batch = next(iter(data.val_dataloader()))
data.visualize(batch)
```



نحن الآن جاهزون للعمل مع مجموعة بيانات Fashion-MNIST في الأقسام التالية.

4.2.4. الملخص

لدينا الآن مجموعة بيانات أكثر واقعية لاستخدامها في التصنيف. Fashion-MNIST هي مجموعة بيانات لتصنيف الملابس تتكون من صور تمثل 10 فئات. سنستخدم مجموعة البيانات هذه في الأقسام والفصول اللاحقة لتقدير تصميمات الشبكات المختلفة، من نموذج خططي بسيط إلى شبكات متقدمة. كما نفعل عادةً مع الصور، نقرأها كموتر للشكل (حجم الدفع، عدد القنوات، الارتفاع، العرض). في الوقت الحالي، لدينا قناة واحدة فقط لأن الصور ذات تدرج رمادي (يستخدم التمثيل البياني أعلاه لوحدة ألوان زائفة لتحسين الرؤية).

أخيرًا، تعد أجهزة تكرار البيانات data iterators مكوناً رئيسياً للأداء الفعال. على سبيل المثال، قد نستخدم وحدات معالجة الرسومات GPU لإلغاء ضغط الصورة بكفاءة، أو تحويل ترميز الفيديو، أو غير ذلك من عمليات المعالجة المسبقة. كلما كان ذلك ممكناً، يجب أن تعتمد على مكررات البيانات التي تم تنفيذها جيداً والتي تستغل الحوسبة عالية الأداء لتجنب إبطاء حلقة التدريب الخاصة بك.

4.2.5. التمارين

- هل تقليل حجم الدفع `batch_size` (على سبيل المثال، إلى 1) يؤثر على أداء القراءة؟

2. أداء مكرر البيانات data iterator مهم. هل تعتقد أن التنفيذ الحالي سريع بما فيه الكفاية؟ استكشف الخيارات المختلفة لتحسينها. استخدم ملف تعريف النظام لمعرفة مكان الاختناقفات bottlenecks.

3. تحقق من وثائق API على الإنترن特 الخاصة بإطار العمل. ما هي مجموعات البيانات الأخرى المتوفرة؟

4.3 نموذج التصنيف الأساسي The Base Classification Model

ربما لاحظت أن عمليات التنفيذ من البداية implementations from scratch والتنفيذ المختصر concise implementation باستخدام دالة إطار العمل كانت متشابهة تماماً في حالة الانحدار. وينطبق الشيء نفسه على التصنيف. نظراً لأن العديد من النماذج في هذا الكتاب تعامل مع التصنيف، فمن الجدير إضافة بعض الدوال لدعم هذا الإعداد على وجه التحديد. يوفر هذا القسم فئة أساسية لنماذج التصنيف لتبسيط الكود المستقبلي.

```
import tensorflow as tf
from IPython import display
from d2l import tensorflow as d2l
```

4.3.1 فئة المصنف Classifer Class

نحدد فئة المصنف Classifier أدناه. في validation_step تقوم بالإبلاغ عن كل من قيمة الخطأ ودقة التصنيف classification accuracy على دفعه التتحقق من الصحة validation. نحن نرسم تحديثاً لكل عدد من الدفعات num_val_batches. هذا لهفائدة توليد batch. متسط الخطأ والدقة على بيانات التتحقق بأكملها. هذه الأرقام المتوسطة ليست صحيحة تماماً إذا كانت الدفعه الأخيرة تحتوي على أمثلة أقل، لكننا نتجاهل هذا الاختلاف الطفيف للحفاظ على بساطة الرمز.

```
class Classifier(d2l.Module): #@save
    def validation_step(self, batch):
        Y_hat = self(*batch[:-1])
        self.plot('loss', self.loss(Y_hat, batch[-1]),
train=False)
        self.plot('acc', self.accuracy(Y_hat, batch[-1]),
train=False)
```

بشكل افتراضي، نستخدم محسن التدرج الاستقائي العشوائي SGD، يعمل على الدفعات الصغيرة minibatches، تماماً كما فعلنا في سياق الانحدار الخطى.

```
@d2l.add_to_class(d2l.Module) #@save
def configure_optimizers(self):
    return tf.keras.optimizers.SGD(self.lr)
```

4.3.2. الدقة Accuracy

بالنظر إلى التوزيع الاحتمالي المتوقع y_{hat} ، فإننا عادةً ما نختار الفئة ذات أعلى احتمالية متوقعة عندما يتغير علينا إخراج تنبؤ صعب. في الواقع، تتطلب العديد من التطبيقات أن تقوم بالاختيار. على سبيل المثال، يجب أن يصنف Gmail بريداً إلكترونياً إلى "أساسي" أو "اجتماعي" أو "تحديثات" أو "منتديات" أو "بريد عشوائي". قد يقدر الاحتمالات داخلياً، ولكن في نهاية اليوم، يجب عليه اختيار واحد من بين الفئات.

عندما تتوافق التنبؤات مع فئة التسمية y ، فإنها صحيحة. دقة التصنيف هي جزء من كل التوقعات الصحيحة. على الرغم من أنه قد يكون من الصعب تحسين الدقة بشكل مباشر (لا يمكن التمييز بينها)، إلا أنه غالباً ما يكون مقياس الأداء الذي نهتم به كثيراً. غالباً ما تكون الكمية ذات الصلة في المعايير. على هذا النحو، سنقوم دائمًا بالإبلاغ عن ذلك عند تدريب المصنفات.

يتم حساب الدقة على النحو التالي. أولاً، إذا كانت y_{hat} عبارة عن مصفوفة، فإننا نفترض أن بعد الثاني يخزن درجات التنبؤ لكل فئة. نستخدم `argmax` للحصول على الفئة المتوقعة بواسطة الفهرس لأكبر إدخال في كل صف. ثم نقارن الفئة المتنبأ بها مع الحقيقة الأساسية y بشكل عنصري `elementwise`. نظرًا لأن عامل المساواة == حساس لأنواع البيانات، فإننا نقوم بتحويل نوع بيانات y_{hat} لمطابقة نوع بيانات y . والنتيجة هي موتر يحتوي على إدخالات من 0 (خطأ) و 1 (صواب). أخذ المجموع ينتج عنه عدد التنبؤات الصحيحة.

```
@d2l.add_to_class(Classifier) #@save
def accuracy(self, Y_hat, Y, averaged=True):
    """Compute the number of correct predictions."""
    Y_hat = tf.reshape(Y_hat, (-1, Y_hat.shape[-1]))
    preds = tf.cast(tf.argmax(Y_hat, axis=1), Y.dtype)
    compare = tf.cast(preds == tf.reshape(Y, -1),
                      tf.float32)
    return tf.reduce_mean(compare) if averaged else
compare
```

4.3.3. الملخص

التصنيف مشكلة شائعة بما فيه الكفاية لدرجة أنه يضمن دوال الملائمة convenience functions الخاصة به. من الأهمية بمكان في التصنيف دقة المصنف. لاحظ أنه بينما نهتم غالباً بالدقة، فإننا ندرب المصنفين لتحسين مجموعة متنوعة من الأهداف الأخرى لأسباب إحصائية وحسابية. ومع ذلك، بغض النظر عن دالة الخطأ التي تم تقليلها أثناء التدريب، فمن المفيد أن يكون لديك طريقة ملائمة لتقييم دقة المصنف تجريبيًا.

4.3.4. تمارين

1. قم بالإشارة إلى خطأ التحقق من الصحة L_v ، وليكن L_v^q تقديرها السريع والقذر محسوباً بواسطة دالة الخطأ المترافق المتوسطي هذا القسم. أخيراً، قم بالإشارة بواسطة الخطأ آخر l_v^b minibatch. قم بالتعبير عن L_v من حيث L_v^q و l_v^b وأحجام العينة و minibatch.

2. أظهر أن التقدير السريع والقذر L_v^q غير متحيز unbiased. هذا هو، أظهر ذلك $E[L_v] = E[L_v^q]$. لماذا لا تزال ترغب في استخدام L_v بدلاً من ذلك؟

3. بالنظر إلى خطأ التصنيف متعدد الطبقات multiclass classification loss $p(y | x)$ التي تدل على عقوبة التقدير y' عندما نرى y والاحتمالية المعطاة $p(y | x)$ ، قم بصياغة القاعدة للاختيار الأمثل \hat{y} . تلميح: عبر عن الخطأ المتوقع باستخدام $p(y | x)$ و l .

4.4 تنفيذ انحدار Softmax من الصفر Implementation from Scratch

نظراً لأن انحدار softmax أساسياً جداً، فنحن نعتقد أنه يجب عليك معرفة كيفية تنفيذه بنفسك. هنا، نقتصر على تحديد الجوانب الخاصة ب softmax للنموذج وإعادة استخدام المكونات الأخرى من قسم الانحدار الخطي، بما في ذلك حلقة التدريب.

```
import tensorflow as tf
from d2l import tensorflow as d2l
```

4.4.1 سوفت ماكس The Softmax

لنبدأ بالجزء الأكثر أهمية: التخطيط من القيم القياسية Scalars إلى الاحتمالات probabilities. لتجديد المعلومات، تذكر عملية عامل المجموع على طول أبعاد محددة في موتر، كما تمت مناقشتها في القسم 2.3.6 والقسم 2.3.7. بالنظر إلى المصفوفة X ، يمكننا جمع كل العناصر (افتراضياً) أو فقط العناصر الموجودة في نفس المحور. يتبع لنا متغير المحور حساب مجاميع الصحف والأعمدة:

```
X = tf.constant([[1.0, 2.0, 3.0], [4.0, 5.0, 6.0]])
tf.reduce_sum(X, 0, keepdims=True), tf.reduce_sum(X, 1,
keepdims=True)
```

```
<tf.Tensor: shape=(1, 3), dtype=float32,
numpy=array([[5., 7., 9.]], dtype=float32)>,
<tf.Tensor: shape=(2, 1), dtype=float32, numpy=
array([[ 6.],
       [15.]], dtype=float32)>
```

يتطلب حساب softmax ثلاثة خطوات: (1) الأصل لكل مصطلح؛ (2) مجموع كل صفات لحساب ثابت التسوية لكل مثال؛ (3) قسمة كل صفات على ثابت التسوية الخاص بها ، مع التأكد من أن النتيجة تصل إلى 1.

$$\text{softmax}(\mathbf{X})_{ij} = \frac{\exp(\mathbf{X}_{ij})}{\sum_k \exp(\mathbf{X}_{ik})}.$$

يسمى (لوغاريتم) المقام بدالة القسم (اللوغاريتم) partition function (\log). تم تقديمها في الفيزياء الإحصائية statistical physics لتلخيص جميع الحالات الممكنة في مجموعة الديناميكا الحرارية. التنفيذ مباشر:

```
def softmax(X):
    X_exp = tf.exp(X)
    partition = tf.reduce_sum(X_exp, 1, keepdims=True)
    return X_exp / partition # The broadcasting
mechanism is applied here
```

لأي إدخال X ، نحول كل عنصر إلى رقم غير سالب. يلخص كل صفات ما يصل إلى 1، كما هو مطلوب للاحتمال. تحذير: الكود أعلاه ليس قوياً ضد المدخلات الكبيرة جداً أو الصغيرة جداً. في حين أن هذا كافٍ لتوضيح ما يحدث، يجب ألا تستخدم هذا الرمز حرفيًا لأي غرض جاد. تحتوي أطر التعلم العميق على مثل هذه الحماية المضمنة وسنستخدم softmax المدمج في المستقبل.

```
X = tf.random.uniform((2, 5))
X_prob = softmax(X)
X_prob, tf.reduce_sum(X_prob, 1)
<tf.Tensor: shape=(2, 5), dtype=float32, numpy=
array([[0.15478596, 0.14451225, 0.31821206, 0.1521694 ,
0.23032041],
       [0.21168488, 0.29741856, 0.15408915, 0.14877847,
0.18802889]],
      dtype=float32)>,
<tf.Tensor: shape=(2,), dtype=float32,
numpy=array([1.0000001, 0.9999994], dtype=float32)>)
```

The Model 4.4.2

لدينا الآن كل ما نحتاجه لتنفيذ نموذج انحدار softmax. كما في مثال الانحدار الخطى الخاص بنا، سيتم تمثيل كل مثال instance بواسطة متوجه بطول ثابت. نظرًا لأن البيانات الأولية هنا تتكون من صور 28×28 بكسل، فإننا نقوم بتسوية كل صورة، ومعاملتها كمتوجهات بطول 784.

في فصول لاحقة، سنقدم الشبكات العصبية التلaffيفية convolutional neural networks والتي تستغل البنية المكانية بطريقة أكثر إرضاءً.

في انحدار softmax، يجب أن يكون عدد المخرجات من شبكة مساوياً لعدد الفئات. نظراً لأن مجموعة البيانات الخاصة بنا تتكون من 10 فئات، فإن بعد إخراج شبكة هو 10. وبالتالي، تشكل أوزاننا مصفوفة 10×784 بالإضافة إلى متوجه صاف 10×1 أبعاد للتحيزات. كما هو الحال مع الانحدار الخطي، نقوم بتهيئه الأوزان W بضوابط غاوسي. تتم تهيئه التحيزات كأصفار.

```
class SoftmaxRegressionScratch(d2l.Classifier):
    def __init__(self, num_inputs, num_outputs, lr,
sigma=0.01):
        super().__init__()
        self.save_hyperparameters()
        self.W = tf.random.normal((num_inputs,
num_outputs), 0, sigma)
        self.b = tf.zeros(num_outputs)
        self.W = tf.Variable(self.W)
        self.b = tf.Variable(self.b)
```

يحدد الكود أدناه كيف تقوم الشبكة بتعيين كل إدخال إلى أحد المخرجات. لاحظ أننا نقوم بتسوية كل صورة 28×28 بكسل في الدفعـة إلى متوجه باستخدام إعادة التشكيل reshape قبل تمرير البيانات عبر نموذجنا.

```
@d2l.add_to_class(SoftmaxRegressionScratch)
def forward(self, X):
    return softmax(tf.matmul(tf.reshape(
        X, (-1, self.W.shape[0])), self.W) + self.b)
```

4.4.3. الخطأ عبر الانترóبيا

بعد ذلك نحتاج إلى تنفيذ دالة الخطأ عبر الانترóبيا (المقدمة في القسم 4.1.2). قد تكون هذه هي دالة الخطأ الأكثر شيوعاً في كل التعلم العميق. في الوقت الحالي، فإن تطبيقات التعلم العميق تلقى بسهولة مشاكل التصنيف التي تفوق بكثير تلك التي يتم التعامل معها بشكل أفضل على أنها مشاكل انحدار.

تذكر أن الانترóبيا المتقطعة Cross-Entropy تأخذ احتمالية اللوغاريتم السلبية للاحتمال المتوقع المخصص للتسمية الحقيقة. من أجل الكفاءة، نتجنب بايثون for-loops ونستخدم الغهرسة indexing بدلاً من ذلك. على وجه الخصوص، يتيح لنا التشغيل الأحادي الساخن تحديد مصطلحات المطابقة في y بتحديد مصطلحات المطابقة في \hat{y} .

لرؤية هذا عملياً، قمنا بإنشاء عينة بيانات `y_hat` مع مثالين للاحتمالات المتوقعة عبر 3 فئات والتسميات المقابلة لها `y`. التسميات الصحيحة هي 1 و 2 على التوالي. باستخدام `y_hat` كمؤشرات للاحتمالات في `y_hat`، يمكننا اختيار الحدود بكفاءة.

```
y_hat = tf.constant([[0.1, 0.3, 0.6], [0.3, 0.2, 0.5]])
y = tf.constant([0, 2])
tf.boolean_mask(y_hat, tf.one_hot(y, depth=y_hat.shape[-1]))
<tf.Tensor: shape=(2,), dtype=float32, numpy=array([0.1, 0.5], dtype=float32)>
```

يمكنا الآن تنفيذ دالة الخطأ عبر الانتروبيا من خلال حساب المتوسط على لوغاريتمات الاحتمالات المحددة.

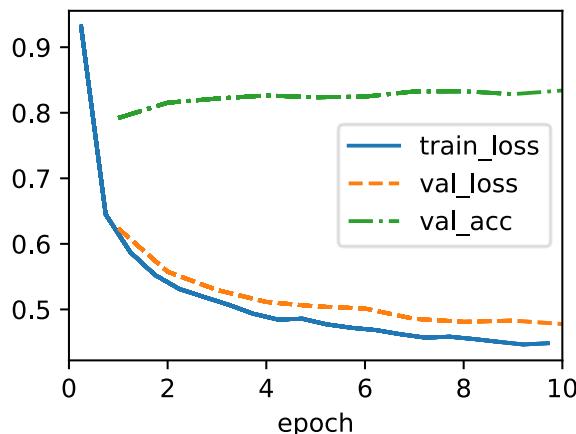
```
def cross_entropy(y_hat, y):
    return - tf.reduce_mean(tf.math.log(tf.boolean_mask(
        y_hat, tf.one_hot(y, depth=y_hat.shape[-1]))))

cross_entropy(y_hat, y)
<tf.Tensor: shape=(), dtype=float32, numpy=1.4978662>
@d2l.add_to_class(SoftmaxRegressionScratch)
def loss(self, y_hat, y):
    return cross_entropy(y_hat, y)
```

4.4.4 Training

نعيد استخدام طريقة الملاعمة المحددة في القسم 3.4 لتدريب النموذج على 10 فترات. لاحظ أن كلاً من عدد الفترات (`max_epochs`)، وحجم الدفعات الصغيرة (`batch_size`)، ومعدل التعلم (`lr`) هي معلمات فائقة قابلة للتعديل `adjustable hyperparameters`. هذا يعني أنه على الرغم من عدم تعلم هذه القيم خلال حلقة التدريب الأولى لدينا، إلا أنها لا تزال تؤثر على أداء نموذجنا، والبوت مقابل التدريب وأداء التعميم. من الناحية العملية، سترغب في اختيار هذه القيم بناءً على تقسيم التحقق من صحة البيانات ثم تقييم نموذجك النهائي في نهاية المطاف في قسم الاختبار. كما تمت مناقشته في القسم 3.6.3، سوف نتعامل مع بيانات اختبار-`Fashion-MNIST` باعتبارها مجموعة التحقق من الصحة، وبالتالي الإبلاغ عن خطأ التتحقق من الصحة ودقة التتحقق من الصحة على هذا التقسيم.

```
data = d2l.FashionMNIST(batch_size=256)
model = SoftmaxRegressionScratch(num_inputs=784,
    num_outputs=10, lr=0.1)
trainer = d2l.Trainer(max_epochs=10)
trainer.fit(model, data)
```



4.4.5. التنبؤ Prediction

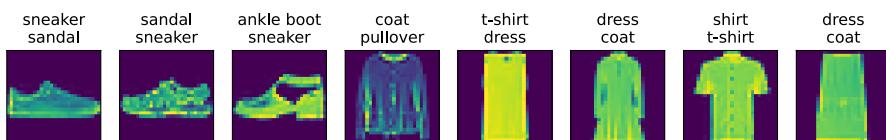
الآن بعد اكتمال التدريب، أصبح نموذجنا جاهزاً لتصنيف بعض الصور.

```
X, y = next(iter(data.val_dataloader()))
preds = tf.argmax(model(X), axis=1)
preds.shape
```

TensorShape([256])

نحن مهتمون أكثر بالصور التي نسميها label بشكل غير صحيح. نتخيلها من خلال مقارنة تسمياتها الفعلية (السطر الأول من إخراج النص) مع التنبؤات من النموذج (السطر الثاني من إخراج النص)

```
wrong = tf.cast(preds, y.dtype) != y
X, y, preds = X[wrong], y[wrong], preds[wrong]
labels = [a+'\n'+b for a, b in zip(
    data.text_labels(y), data.text_labels(preds))]
data.visualize([X, y], labels=labels)
```



4.4.6. الملخص Summary

لقد بدأنا الآن في اكتساب بعض الخبرة في حل مشاكل الانحدار الخطي والتصنيف. بواسطته، وصلنا إلى ما يمكن القول إنه حالة فن النمذجة الإحصائية في السبعينيات والستينيات من القرن

الماضي. في القسم التالي، سنوضح لك كيفية الاستفادة من أطر التعلم العميق لتنفيذ هذا النموذج بشكل أكثر كفاءة.

4.4.7 التمارين

في هذا القسم، قمنا بتنفيذ دالة softmax بشكل مباشر بناءً على التعريف الرياضي لعملية softmax. كما نوقش في القسم 4.1، يمكن أن يتسبب هذافي عدم استقرار رقمي numerical instabilities.

1. اختبر ما إذا كان softmax لا يزال يعمل بشكل صحيح إذا كانت قيمة الإدخال ≥ 100 ؟

2. اختبر ما إذا كان softmax لا يزال يعمل بشكل صحيح إذا كان أكبر المدخلات أصغر من 100؟

3. قم بتنفيذ الإصلاح بالنظر إلى القيمة المتعلقة بأكبر إدخال في الوسيطة.

2. تتنفيذ دالة softmax cross_entropy تتبع تعريف دالة الخطأ عبر الانتروبيا

$$\sum_i y_i \log \hat{y}_i$$

1. جربه في مثال الكود أعلاه.

2. لماذا تعتقد أنه يعمل بشكل أبطأ؟

3. هل يجب عليك استخدامه؟ في أي الحالات يكون له معنى؟

4. ما الذي تحتاج إلى تخفيض الحذر منه؟ تلميح: ضع في اعتبارك مجال اللوغاريتم.

3. هل من الجيد دائمًا إرجاع التسمية الأكثر احتمالاً؟ على سبيل المثال، هل ستفعل هذا من أجل التشخيص الطبي؟ كيف ستحاول معالجة هذا؟

4. افترض أننا نريد استخدام انحدار softmax للتنبؤ بالكلمة التالية بناءً على بعض الميزات. ما هي بعض المشاكل التي قد تنشأ من المفردات الكبيرة؟

5. جرب المعلمات الفائقة hyperparameters للشفرة أعلاه. خاصه:

1. ارسم كيف يتغير خطأ التتحقق أثناء تغيير معدل التعلم.

2. هل يتغير التتحقق من الصحة وخطأ التدريب كلما قمت بتغيير حجم الدفعات الصغيرة؟ إلى أي مدى يجب أن تذهب قبل أن ترى تأثيراً كبيراً أم صغيراً؟

4.5 التنفيذ المختصر لأنحدار Softmax

مثلاً سهلت أطر التعلم العميق عالية المستوى تنفيذ الانحدار الخطى (انظر القسم 3.5)، فهي مناسبة أيضاً هنا.

```
import tensorflow as tf
from d2l import tensorflow as d2l
```

4.5.1. تعريف النموذج Defining the Model

كما في القسم 3.5، نقوم ببناء الطبقة المتصلة بالكامل باستخدام الطبقة المضمنة built-in layer. ثم تستدعي طريقة `forward` المضمنة `call` كلما احتجنا إلى تطبيق الشبكة على بعض المدخلات.

```
class SoftmaxRegression(d2l.Classifier):
    def __init__(self, num_outputs, lr):
        super().__init__()
        self.save_hyperparameters()
        self.net = tf.keras.models.Sequential()
        self.net.add(tf.keras.layers.Flatten())
        self.net.add(tf.keras.layers.Dense(num_outputs))

    def forward(self, X):
        return self.net(X)
```

4.5.2. إعادة النظر في سوفت ماكس Softmax Revisited

في القسم 4.4، قمنا بحساب ناتج نموذجنا وطبقنا خطأ الانتروبيا المتقاطعة. في حين أن هذا معقول رياضيًّا تماماً، إلا أنه محفوف بالمخاطر من الناحية الحسابية، بسبب underflow و overflow في الأس.

تذكر أن دالة softmax تحسب الاحتمالات عبر $\hat{y}_j = \frac{\exp(o_j)}{\sum_k \exp(o_k)}$. إذا كان بعضها كبيراً جداً، أي إيجابي جداً، فقد يكون أكبر من أكبر رقم يمكننا الحصول عليه لأنواع بيانات معينة. هنا يسمى الفاصل overflow. وبالمثل، إذا كانت جميع المعاملات arguments سلبية للغاية، فستحصل على underflow. على سبيل المثال، تغطي أرقام الفاصل العائمة ذات الدقة الواحدة تقريباً نطاقاً إلى 10^{-38} . على هذا النحو، إذا كان الحد الأكبر في 0 يقع خارج الفترة الزمنية $[90, 90]$ ، فلن تكون النتيجة مستقرة. حل هذه المشكلة هو طرح $\bar{o} \stackrel{\text{def}}{=} \max_k o_k$ من كافة الإدخالات:

$$\hat{y}_j = \frac{\exp o_j}{\sum_k \exp o_k} = \frac{\exp (o_j - \bar{o}) \exp \bar{o}}{\sum_k \exp (o_k - \bar{o}) \exp \bar{o}} = \frac{\exp (o_j - \bar{o})}{\sum_k \exp (o_k - \bar{o})}.$$

من خلال البناء نعرف $0 \leq \bar{o} - o_j$ لـ j . على هذا النحو، بالنسبة لمشكلة تصنيف فئة q ، يتم احتواء المقام في الفاصل الزمني $[1, q]$. علاوة على ذلك، لا يتجاوز البسط أبداً 1 ، وبالتالي يمنع التدفق العددي numerical overflow. يحدث التدفق العددي فقط عندما $\exp(o_j - \bar{o})$ تُقييم عدديًّا كـ 0. ومع ذلك، على بعد خطوات قليلة على الطريق، قد نجد أنفسنا في مأزق عندما نريد حساب $\log \hat{y}_j$ كما $\log 0$. على وجه الخصوص، في الانشار الخلفي

قد نجد أنفسنا في مواجهة شائنة من نتائج NaN المخيفة (ليس رقمًا).(Not a Number).

لحسن الحظ، يتم إنقاذنا من حقيقة أنه على الرغم من أننا نحسب الدوال الأسية، فإننا نعترف في النهايةأخذ \log (عند حساب خطأ الانتروبيا المتقطعة). من خلال الجمع بين softmax و cross-entropy، يمكننا الهروب من مشكلات الاستقرار العددي تماماً. نملك:

$$\log \hat{y}_j = \log \frac{\exp(o_j - \bar{o})}{\sum_k \exp(o_k - \bar{o})} = o_j - \bar{o} - \log \sum_k \exp(o_k - \bar{o}).$$

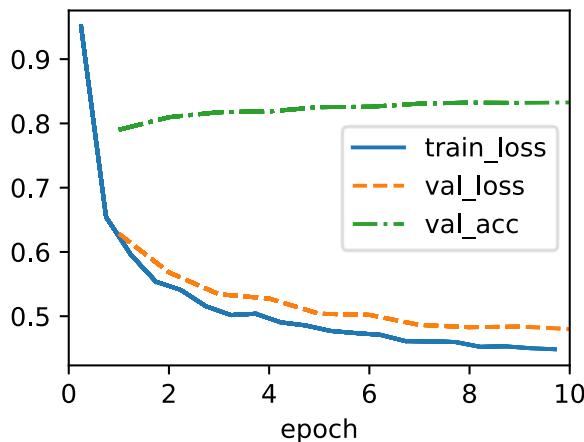
هذا يتتجنب كلاً من الفائض overflow والفيضان underflow. سرغب في الاحتفاظ بدالة softmax التقليدية في متناول اليدي حالة رغبتنا في تقدير احتمالات الإخراج من خلال نموذجنا. ولكن بدلاً من تمرير احتمالات softmax إلى دالة الخسارة الجديدة لدينا، نقوم فقط بتمرير السجلات logits وحساب softmax وسجله مرة واحدة داخل دالة فقدان الانتروبيا، والتي تقوم بأشياء ذكية مثل "خدعة" LogSumExp trick .

```
@d2l.add_to_class(d2l.Classifier) #@save
def loss(self, Y_hat, Y, averaged=True):
    Y_hat = tf.reshape(Y_hat, (-1, Y_hat.shape[-1]))
    Y = tf.reshape(Y, (-1,))
    fn =
    tf.keras.losses.SparseCategoricalCrossentropy(from_logits=True)
    return fn(Y, Y_hat)
```

4.5.3 Training.

بعد ذلك نقوم بتدريب نموذجنا. كما كان من قبل، نستخدم صور Fashion-MNIST، تسطح إلى 784 متوجهًا للميزات ذات الأبعاد flattened.

```
data = d2l.FashionMNIST(batch_size=256)
model = SoftmaxRegression(num_outputs=10, lr=0.1)
trainer = d2l.Trainer(max_epochs=10)
trainer.fit(model, data)
```



كما في السابق، تقارب هذه الخوارزمية مع حل يحقق دقة لائقة، وإن كان هذا الوقت يحتوي على عدد أقل من سطور التعليمات البرمجية عن ذي قبل.

4.5.4. الملخص

تعد واجهات برمجة التطبيقات APIs عالية المستوى ملائمة للغاية في إخفاء الجوانب التي يحتمل أن تكون خطيرة عن مستخدميها، مثل الاستقرار العددي numerical stability. علاوة على ذلك، فهي تسمح للمستخدمين بتصميم النماذج بأي جاز مع عدد قليل جداً من أسطر التعليمات البرمجية. هذا هو بمثابة نعمة ونقطة. الفائدة الواضحة هي أنه يجعل الأشياء سهلة الوصول، حتى للمهندسين الذين لم يأخذوا فئة واحدة من الإحصائيات في حياتهم (في الواقع، هذا هو أحد الجماهير المستهدفة للكتاب). لكن إخفاء الحواف الحادة يأتي أيضاً مع ثمن: عامل مشيط لإضافة مكونات جديدة ومختلفة بمفردك، نظراً لوجود ذاكرة عضلية قليلة للقيام بذلك. علاوة على ذلك، فإنه يجعل من الصعب إصلاح الأشياء كلما فشلت الحشوة الحامية protective padding لإطار العمل في تعطية جميع حالات الزاوية بالكامل. مرة أخرى، هذا يرجع إلى عدم الإلمام.

على هذا النحو، نحن بشدة على مراجعة كل من العظام المجردة والإصدارات الأنيقة للعديد من التطبيقات التالية. بينما نؤكد على سهولة الفهم، فإن التطبيقات عادة ما تكون عالية الأداء (التاليف convolutions هي الاستثناء الكبير هنا). نعتمد السماح لك بالبناء عليها عندما تختبر شيئاً جديداً لا يمكن لأي إطار عمل أن يمنحك إياه.

4.5.5. التمارين

1. يستخدم التعلم العميق العديد من تنسيقات الأرقام المختلفة، بما في ذلك الدقة المزدوجة لـ FP64 (نادرًا جدًا)، الدقة الفردية FP32، BFLOAT16 (جيدة للتمثيلات المضغوطة)، FP16 (غير مستقر جدًا)، TF32 (تنسيق جديد من

1. INT8، و NVIDIA احسب أصغر وأكبر وسيلة للدالة الأسيّة التي لا تؤدي overflow numerical underflow أو .
2. INT8 هو تنسيق محدود للغاية بأرقام غير صفرية من 1 إلى 255. كيف يمكن توسيع نطاقها الديناميكي دون استخدام المزيد من البิตات؟ هل الضرب والجمع القياسيان مازالاً يعملان؟
3. زيادة عدد فترات التدريب number of epochs for training. لماذا قد تختفي دقة التحقق بعد فترة؟ كيف يمكننا إصلاح هذا؟
4. ماذا يحدث كلما زادت معدل التعلم؟ قارن منحنين الخطأ للعديد من معدلات التعلم. أيهما يعمل بشكل أفضل؟ متى؟

4.6 التععمق في التصنيف Classification

حتى الآن، ركزنا على كيفية معالجة مشاكل التصنيف متعدد الطبقات multiclass classification problems من خلال تدريب الشبكات العصبية (الخطية) ذات المخرجات المتعددة ودوال softmax. عند تفسير مخرجات نموذجنا على أنها تنبؤات احتمالية، قمنا بتحفيز واستفاق دالة خطأ الانتروبيا المتقاطعة cross-entropy loss function، والتي تحسب احتمالية اللوغاريتم السلبي negative log likelihood التي يخصصها نموذجنا (لمجموعة ثابتة من المعلمات) للتسميات الفعلية. وأخيراً، نضع هذه الأدوات موضع التنفيذ من خلال fitting نموذجنا لمجموعة التدريب. ومع ذلك، كما هو الحال دائمًا، فإن هدفنا هو تعلم الأنماط العامة general patterns، كما تم تقييمها تجريبياً على بيانات غير مرئية من قبل unseen data (مجموعة الاختبار). الدقة العالية في مجموعة التدريب لا تعني شيئاً. عندما تكون كل من مدخلاتنا فريدة من نوعها (وهذا ينطبق بالفعل على معظممجموعات البيانات عالية الأبعاد)، يمكننا تحقيق دقة مثالية في مجموعة التدريب بمجرد حفظ مجموعة البيانات في فترة التدريب الأولى، ثم البحث عن التسمية كلما رأينا صورة جديدة. ومع ذلك، فإن حفظ التسميات الدقيقة المرتبطة بأمثلة التدريب الدقيقة لا يخبرنا بكيفية تصنيف الأمثلة الجديدة. في غياب المزيد من الإرشادات، قد نضطر إلى الرجوع إلى التخمين العشوائي كلما واجهنا أمثلة جديدة.

يتطلب عدد من الأسئلة الملحة اهتماماً فوريّاً:

1. كم عدد أمثلة الاختبار التي تحتاجها لتقدير دقة المصنفات الخاصة بنا على السكان الأساسيين؟
2. ماذا يحدث إذا وصلنا تقدير النماذج في نفس الاختبار بشكل متكرر؟
3. لماذا يجب أن نتوقع أن ملاعمة نماذجنا الخطية لمجموعة التدريب يجب أن تكون أفضل من مخطط الحفظ الساذج لدينا؟

بينما قدم القسم 3.6 أساسيات الضبط الزائد overfitting والتععمق generalization في سياق الانحدار الخطى، فإن هذا الفصل سوف يتعمق قليلاً، حيث يقدم بعض الأفكار التأسيسية لنظرية

التعلم الإحصائي. اتضح أنه يمكننا في كثير من الأحيان ضمان التعميم بشكل مسبق: بالنسبة للعديد من النماذج، ولأي حد أعلى مرغوب فيه على فجوة التعميم ϵ ، يمكننا في كثير من الأحيان تحديد عدد معين من العينات n المطلوبة مثل إذا كانت مجموعة التدريب لدينا تحتوي على عينات n على الأقل، ثم يقع خطأ التجربة في نطاق ϵ للخطأ الحقيقي، لأي توزيع لتوليد البيانات data generating distribution. لسوء الحظ، اتضح أيضًا أنه بينما توفر هذه الأنواع من الضمانات مجروبة عميقه من البناءات الفكرية، إلا أنها ذات فائدة عملية محدودة لممارسي التعلم العميق. باختصار، تشير هذه الضمانات إلى أن ضمان تعميم الشبكات العصبية العميق بشكل مسبق يتطلب عددًا سخيفاً من الأمثلة (ربما تريليونات أو أكثر)، حتى عندما نجد أنه في المهام التي نهتم بها تلك الشبكات العصبية العميقه عادةً ما يتم تعميمها جيدًا بشكل ملحوظ مع بعيد. عدد أقل من الأمثلة (بالآلاف). وبالتالي، غالباً ما يتخلّى ممارسو التعلم العميق عن الضمانات المسبقة تماماً، وبدلًا من ذلك يستخدمون الأساليب على أساس أنهم قد تعمموا جيدًا على مشكلات مماثلة في الماضي، ويصادقون على التعميم بعد ذلك من خلال التقييمات التجريبية. عندما نصل إلى القسم 5، سنعيد النظر في التعميم ونقدم مقدمة خفيفة للأدبيات العلمية الواسعة التي نشأت في محاولات لشرح سبب تعميم الشبكات العصبية العميقه في الممارسة.

4.6.1. مجموعة الاختبار The Test Set

نظرًا لأننا بدأنا بالفعل في الاعتماد علىمجموعات الاختبار كأسلوب معياري ذهبي لتقييم خطأ التعميم، فلنبدأ من خلال مناقشة خصائص تقديرات الخطأ هذه. دعنا نركز على مصنف ثابت f ، دون القلق بشأن كيفية الحصول عليه. علاوة على ذلك، افترض أننا نمتلك مجموعة بيانات جديدة من الأمثلة $\{(\mathbf{x}^{(i)}, y^{(i)})\}_{i=1}^n$ التي لم تستخدم لتدريب المصنف f . الخطأ التجاري لمصنفنا f على \mathcal{D} هو ببساطة جزء من الحالات التي لا يتفق فيها التنبؤ $(\mathbf{x}^{(i)}, f(\mathbf{x}^{(i)}))$ مع التسمية الحقيقية $y^{(i)}$ ، ويعطى بالتعبير التالي:

$$\epsilon_{\mathcal{D}}(f) = \frac{1}{n} \sum_{i=1}^n \mathbf{1}(f(\mathbf{x}^{(i)}) \neq y^{(i)}).$$

على التفاصيل من ذلك، فإن الخطأ السكاني population error هو الجزء المتوقع من الأمثلة في المجموعة الأساسية (بعض التوزيعات $P(X, Y)$ التي تميز بدالة كثافة الاحتمال $p(x, y)$ التي لا يتفق المصنف لدينا مع التسمية الحقيقية true label :

$$\epsilon(f) = E_{(\mathbf{x}, y) \sim P} \mathbf{1}(f(\mathbf{x}) \neq y) = \int \int \mathbf{1}(f(\mathbf{x}) \neq y) p(\mathbf{x}, y) d\mathbf{x} dy.$$

بينما $\epsilon(f)$ هي الكمية التي نهتم بها بالفعل، لا يمكننا ملاحظتها مباشرة، تماماً كما لا يمكننا أن نلاحظ بشكل مباشر متوسط الطول في عدد كبير من السكان دون قياس كل شخص. يمكننا فقط

تقدير هذه الكمية بناءً على العينات. نظرًا لأن مجموعة الاختبار \mathcal{D} الخاصة بنا ممثلة إحصائيًا للسكان الأساسيين، فيمكننا عرض $(f)_{\mathcal{D}}$ كمقدار إحصائي للخطأ السكاني $(f)_{\epsilon}$. علاوة على ذلك، نظرًا لأن كمية الاهتمام $(f)_{\epsilon}$ لدينا هي توقع (للمتغير العشوائي $(Y \neq f(X))$) والمقدار المقابل $(f)_{\mathcal{D}}$ هو متوسط العينة، فإن تقدير خطأ السكان هو ببساطة المشكلة الكلاسيكية لتقدير المتوسط، والتي قد تذكرها من القسم 2.6.

هناك نتيجة كلاسيكية مهمة من نظرية الاحتمالات تسمى نظرية الحد المركزي central limit theorem تضمن أنه كلما امتلكنا n عينات عشوائية مأخوذة من أي توزيع بمتوسط μ وانحراف معياري σ ، حيث يقترب عدد العينات n من اللانهاية، فإن متوسط العينة $\bar{\mu}$ يميل تقريرًا نحو التوزيع الطبيعي المتمحور حول الوسط الحقيقي والانحراف المعياري σ/\sqrt{n} . بالفعل، هنا يخبرنا بشيء مهم: مع تزايد عدد الأمثلة، يجب أن يقترب خطأ الاختبار $(f)_{\mathcal{D}}$ لدينا من الخطأ الحقيقي $(f)_{\epsilon}$ بمعدل (σ/\sqrt{n}) . وبالتالي، لتقدير خطأ الاختبار لدينا مرتين بدقة، يجب أن نجمع مجموعة اختبار أكبر بعشرة أضعاف. لتقليل خطأ الاختبار لدينا بعامل مائة، يجب أن نجمع مجموعة اختبار أكبر بآلاف مرة. بشكل عام، غالباً ما يكون هذا المعدل (σ/\sqrt{n}) هو أفضل ما يمكن أن نأمله في الإحصائيات.

الآن بعد أن عرفنا شيئاً عن المعدل المقارب asymptotic rate الذي يتقارب عنده خطأ الاختبار $(f)_{\mathcal{D}}$ لدينا مع الخطأ الحقيقي $(f)_{\epsilon}$ ، يمكننا تكبير بعض التفاصيل المهمة. تذكر أن المتغير العشوائي محل الاهتمام $(Y \neq f(X))$ يمكن أن يأخذ القيم 0 و 1 فقط، وبالتالي فهو متغير برنولي العشوائي Bernoulli random variable، يتميز بمعامل يشير إلى احتمال أن يأخذ قيمة 1. هنا، 1 يعني أن المصنف الخاص بنا قد ارتكب خطأ، وبالتالي فإن معلمة متغيرنا العشوائي هي في الواقع معدل الخطأ الحقيقي $(f)_{\epsilon}$. يعتمد تباين σ^2 برنولي على معاملته (هنا $(f)_{\epsilon}$) وفقاً للتعبير $((f)_{\epsilon} - 1)^2$. بينما $(f)_{\epsilon}$ غير معروف في البداية، نعلم أنه لا يمكن أن يكون أكبر من 1. يكشف القليل من الاستقصاء عن هذه الدالة أن تبايننا يكون أعلى عندما يكون معدل الخطأ الحقيقي قريباً من 0 ويمكن أن يكون أقل بكثير عندما يكون قريباً من 0 أو قريباً من 1. يخبرنا هذا أن الانحراف المعياري المقارب لتقديرنا $(f)_{\mathcal{D}}$ للخطأ $(f)_{\epsilon}$ (على اختيار عينات الاختبار) لا يمكن أن يكون أكبر من $\sqrt{0.25/n}$.

إذا تجاهلناحقيقة أن هذا المعدل يميز السلوك حيث يقترب حجم مجموعة الاختبار من اللانهاية وليس عندما نمتلك عينات محدودة، فهذا يخبرنا أنه إذا أردنا أن يقترب خطأ الاختبار $(f)_{\mathcal{D}}$ الخاص بنا من الخطأ السكاني $(f)_{\epsilon}$ بحيث يتواافق الانحراف المعياري مع فترة زمنية قدرها ± 0.01 ، ثم يجب أن نجمع ما يقرب من 2500 عينة. إذا أردنا احتواء انحرافيين معياريين في هذا النطاق وبالتالي يكون 95% من ذلك $\epsilon_{\mathcal{D}}(f) \in \epsilon(f) \pm 0.01$ ، فسنحتاج إلى 10000 عينة!

اتضح أن هذا هو حجم مجموعات الاختبار للعديد من المعايير الشائعة في التعلم الآلي. قد تتدشّس عندما تكتشف أنه يتم نشر الآلاف من أوراق التعلم العميق التطبيقي كل عام مما يجعل قدرًا كبيراً من التحسينات في معدل الخطأ 0.01 أو أقل. بالطبع، عندما تكون معدلات الخطأ أقرب بكثير لـ 0 ، فإن تحسين 0.01 يمكن أن يكون بالفعل مشكلة كبيرة.

إحدى السمات المزعجة لتحليلنا حتى الآن هي أنه يخبرنا فقط عن التقارب asymptotics، أي كيف تتطور العلاقة بين $\epsilon_{\mathcal{D}}$ و ϵ تتطور كحجم العينة وتذهب إلى ما لا نهاية. لحسن الحظ، نظرًا لأن متغيرنا العشوائي محدود، يمكننا الحصول على حدود عينة محدودة صالحة من خلال تطبيق متباعدة طبقاً لـ (Hoeffding 1963):

$$P(\epsilon_{\mathcal{D}}(f) - \epsilon(f) \geq t) < \exp(-2nt^2).$$

إن حل أصغر حجم لمجموعة بيانات من شأنه أن يسمح لنا بالاستنتاج بثقة 95٪ أن المسافة t بين تقديرنا $\epsilon_{\mathcal{D}}(f)$ ومعدل الخطأ الحقيقي $\epsilon(f)$ لا تتجاوز 0.01 ، ستتجد أن الأمثلة 15000 تقريرًا مطلوبة مقارنة بالأمثلة 10000 التي اقترحها التحليل المقارب أعلاه. إذا تعمقت في الإحصائيات ستتجد أن هذا الاتجاه ثابت بشكل عام. عادةً ما تكون الضمانات التي يتم الاحتفاظ بها حتى في العينات المحدودة أكثر تحفظاً قليلاً. لاحظ أنه في مخطط الأشياء، هذه الأرقام ليست متباعدة كثيراً، مما يعكس الفائدة العامة للتحليل المقارب لإعطائنا أرقام الملعب حتى لو لم يكن هناك ضمانات يمكننا تقديمها إلى المحكمة.

4.6.2 إعادة استخدام مجموعة الاختبار Test Set Reuse

بمعنى ما، أنت الآن جاهز للنجاح في إجراء بحث تجريبي للتعلم الآلي. تم تطوير جميع النماذج العملية تقريرًا والتحقق من صحتها بناءً على أداء مجموعة الاختبار وأنت الآن خير في مجموعة الاختبار. بالنسبة لأي مصنف ثابت f ، فأنت تعلم تقييم خطأ الاختبار $\epsilon_{\mathcal{D}}(f)$ الخاص به، ومعرفة ما يمكن (وما لا يمكن) قوله بشأن الخطأ السكاني $\epsilon(f)$ الخاص به.

فلنفترض أنك تأخذ هذه المعرفة وتستعد لتدريب نموذجك الأول f_1 . بمعرفة مدى الثقة التي تحتاجها لتكون على ثقة من أداء معدل الخطأ الخاص بالمصنف، يمكنك تطبيق تحليلنا أعلاه لتحديد عدد مناسب من الأمثلة لتخفيضها لمجموعة الاختبار. علاوة على ذلك، لنفترض أنك أخذت الدروس من القسم 3.6 إلى القلب وتأكدت من الحفاظ على قدسيّة مجموعة الاختبار من خلال إجراء جميع تحليلك الأولى، وضبط المعلمة الفائقة، وحتى الاختيار من بين العديد من هيكل النماذج المتنافسة على مجموعة التحقق من الصحة. أخيراً تقوم بتقييم النموذج f_1 الخاص بك على مجموعة الاختبار والإبلاغ عن تقدير غير متحيز unbiased estimate لخطأ السكان مع فاصل ثقة مرتبط.

حتى الآن يبدو أن كل شيء يسير على ما يرام. ومع ذلك، في تلك الليلة، تستيقظي في الثالثة صباحاً بفكرة رائعة لنهج جديد للنموذج. في اليوم التالي، تقوم بترميز نموذجك الجديد f_2 ، وضبط المعلمات الفائقة الخاصة به على مجموعة التحقق من الصحة validation set، ولا تجعل نموذجك الجديد يعمل فحسب، بل يبدو أن معدل الخطأ فيه أقل بكثير من معدل الخطأ للنموذج الأول f_1 . ومع ذلك، فإن إثارة الاكتشاف تتلاشى فجأة بينما تستعد للتقدير النهائي. ليس لديك مجموعة اختبار!

على الرغم من أن مجموعة الاختبار الأصلية D لا تزال موجودة على الخادم الخاص بك، فإنك تواجه الآن مشكلتين هائلتين. أولاً، عندما جمعت مجموعة الاختبار الخاصة بك، قمت بتحديد مستوى الدقة المطلوب على افتراض أنك كنت تقيم مصنفاً واحداً f . ومع ذلك، إذا دخلت في مجال تقييم المصنفات المتعددة f_k, f_1, \dots, f_l في نفس مجموعة الاختبار، فيجب أن تفكري في مشكلة الاكتشاف الخاطئ. من قبل، ربما كنت متاكداً بنسبة 95% من أنه $f \in \epsilon_D(f) \pm 0.01$ بالنسبة لمصنف واحد f ، وبالتالي فإن احتمال وجود نتيجة مضللة كان 5%. فقط. مع وجود k من المصنفات في المزيج، قد يكون من الصعب ضمان عدم وجود حتى واحد منهم يكون أداء مجموعة الاختبار مضللاً. مع وجود 20 مصنفاً قيد الدراسة، قد لا يكون لديك أي سلطة على الإطلاق لاستبعاد احتمال حصول واحد منهم على الأقل على درجة مضللة. تتعلق هذه المشكلة باختبار الفرضيات المتعددة multiple hypothesis testing، والتي على الرغم من الأدبيات الكثيرة في الإحصاء، لا تزال مشكلة مستمرة تعصف بالبحث العلمي.

إذا لم يكن ذلك كافياً للقلق، فهناك سبب خاص لعدم الثقة في النتائج التي تحصل عليها في التقييمات اللاحقة. تذكر أن تحليلنا لأداء مجموعة الاختبار استند إلى افتراض أن المصنف قد تم اختياره بدون أي اتصال بمجموعة الاختبار، وبالتالي يمكننا عرض مجموعة الاختبار على أنها مستمدة عشوائياً من السكان الأساسيين. هنا، لا تختر دولال متعددة فحسب، بل تم اختيار الدالة اللاحقة f_2 بعد ملاحظة أداء مجموعة الاختبار f_1 . بمجرد تسرب المعلومات من مجموعة الاختبار إلى مصمم النماذج، لا يمكن أن تكون مجموعة اختبار حقيقة مرة أخرى بالمعنى الدقيق للكلمة. تسمى هذه المشكلة فرط التجهيز التكيفي adaptive overfitting، وقد ظهرت مؤخراً كموضوع يثير اهتماماً شديداً لمنظري التعلم والإحصائيين (Dwork et al., 2015). لحسن الحظ، في حين أنه من الممكن تسريب جميع المعلومات من مجموعة الانتظار holdout set، والسيناريوهات النظرية للأسوأ قائمة، فقد تكون هذه التحليلات متحفظة للغاية. من الناحية العملية، احرص على إنشاءمجموعات اختبار حقيقة، واستشرها بشكل غير متكرر قدر الإمكان، ولمراجعة اختبار الفرضيات المتعددة عند الإبلاغ عن فترات الثقة، ولزيادة يقظتك بشكل أكثر قوة عندما تكون المخاطر عالية وحجم مجموعة البيانات الخاصة بك صغيراً. عند تشغيل سلسلة من التحديات المعيارية، غالباً ما يكون من الممارسات الجيدة الاحتفاظ بالعديد منمجموعات

الاختبار بحيث يمكن بعد كل جولة تخفيف مجموعة الاختبار القديمة إلى مجموعة التحقق من الصحة.

4.6.3 نظرية التعلم الإحصائي Statistical Learning Theory

في الحال، مجموعات الاختبار هي كل ما لدينا حقاً، ومع ذلك تبدو هذه الحقيقة غير مرضية بشكل غريب. أولاً، نادرًا ما نمتلك مجموعة اختبار حقيقية - ما لم نكن نحن من أنشأ مجموعة البيانات، فمن المحتمل أن شخصاً آخر قد قام بالفعل بتقييم المصنف الخاص به على "مجموعة الاختبار" المزعومة. وحتى عندما نحصل على الدرجات الأولى، سرعان ما نجد أنفسنا محبطين، ونتمنى أن نتمكن من تقييم محاولاتنا اللاحقة في النمذجة دون الشعور بالضيق بأننا لا نستطيع الوثوق بأرقامنا. علاوة على ذلك، حتى مجموعة الاختبار الحقيقية يمكنها فقط أن تخبرنا لاحقاً عمما إذا كان المصنف قد تعمم generalized في الواقع على السكان، وليس ما إذا كان لدينا أي سبب لتوقع مقدماً أنه ينبغي تعميمه.

مع وضع هذه الهواجس في الاعتبار، قد تكون الآن مستعداً بشكل كافٍ لرؤية جاذبية نظرية التعلم الإحصائي statistical learning، الحقل الفرعي الرياضي للتعلم الآلي الذي يهدف ممارسوه إلى توضيح المبادئ الأساسية التي تشرح لماذا / متى يمكن للنمذجة المدربة على البيانات التجريبية التعميم على بيانات غير مرئية unseen data. كان أحد الأهداف الأساسية لعدة عقود من الباحثين في التعلم الإحصائي هو تقييد فجوة التعميم generalization gap، فيما يتعلق بخصائص فئة النموذج، وعدد العينات في مجموعة البيانات.

يهدف منظرو التعلم Learning theorists إلى ربط الفرق بين الخطأ التجاريبي ($f_{\text{S}} - f_{\text{U}}$) للمصنف المتعلّم f ، سواء تم تدريبه أو تقييمه على مجموعة التدريب S ، والخطأ الحقيقي لنفس المصنف على السكان الأساسيين. قد يبدو هذا مشابهاً لمشكلة التقييم التي تناولناها للتو ولكن هناك فرق كبير. من قبل، تم إصلاح المصنف وكنا بحاجة إلى مجموعة بيانات فقط لأغراض التقييم. وبالفعل، فإن أي مصنف ثابت f لا يعمم: خطأ في مجموعة بيانات (غير مرئية سابقاً) هو تقدير غير متحيز لخطأ السكان. ولكن ماذا يمكننا أن نقول عندما يتم تدريب المصنف وتقييمه على نفس مجموعة البيانات؟ هل يمكننا أن نكون على ثقة من أن خطأ التدريب سيكون قريباً من خطأ الاختبار؟

افتراض أنه يجب اختيار المصنف f الذي تعلمناه من بين مجموعة دوال محددة مسبقاً F . تذكر من مناقشتنا لمجموعات الاختبار أنه في حين أنه من السهل تقدير خطأ مصنف واحد، فإن الأشياء تصبح صعبة عندما نبدأ في التفكير فيمجموعات المصنفات. حتى إذا كان الخطأ التجاريبي لأي مصنف واحد (ثابت) قريباً من خطأ الحقيقي مع احتمال كبير، بمجرد أن نفك في مجموعة من المصنفات، نحتاج إلى القلق بشأن احتمال أن يتلقى مصنف واحد فقط في المجموعة خطأ سياماً

في التقدير خطأ. القلق هو أنه إذا تلقى مصنف واحد فقط في مجموعتنا خطأً منخفضاً ملائماً، فقد نختاره وبالتالي نقلل بشكل كبير من الخطأ السكاني. علاوة على ذلك، حتى بالنسبة للنموذج الخطية، نظراً لتقييم معلماتها باستمرار، فإننا نختار عادةً من بين فئة لا حصر لها من الدوال $(|\mathcal{F}| = \infty)$.

يتمثل أحد الحلول الطموحة للمشكلة في تطوير أدوات تحليلية لإثبات التقارب المنتظم uniform convergence، أي أنه مع وجود احتمال كبير، فإن معدل الخطأ التجاريي لكل مصنف في الفئة \mathcal{F} سوف يتقارب في نفس الوقت مع معدل الخطأ الحقيقي. بعبارة أخرى، نسعي إلى مبدأ نظري من شأنه أن يسمح لنا أن نذكر أنه مع وجود احتمال على الأقل $\delta - 1$ (بالنسبة للبعض الصغير δ)، لن يتم تقدير معدل خطأ المصنف $(f) \epsilon$ (من بين جميع المصنفات في الفئة \mathcal{F}) بأكثر من مقدار ضئيل α . من الواضح أنه لا يمكننا إصدار مثل هذه العبارات لجميع فئات النموذج \mathcal{F} . تذكر فئة آلات الحفظ التي تحقق دائماً خطأً تجريبياً ولكنها لا تتفوق أبداً على التخمين العشوائي على المجتمع الأساسي.

يعني أن طبقة الحفظ مرنة للغاية. لا يمكن لنتيجة تقارب موحدة كهذه أن تصمد. من ناحية أخرى، المصنف الثابت عديم الفائدة – فهو معمم تماماً، لكنه لا يناسب بيانات التدريب ولا بيانات الاختبار. وهكذا تم تأطير السؤال المركزي للتعلم تاريخياً كمقاييسة بين فئات نموذج أكثر مرونة (تبين أعلى) تتلاءم بشكل أفضل مع بيانات التدريب ولكنها تنطوي على مخاطر أكبر من اللازم، مقابل فئات نموذجية أكثر صرامة (تحيز أعلى) تتعمم جيداً ولكنها تنطوي على مخاطر غير مناسبة. كان السؤال المركزي في نظرية التعلم هو تطوير التحليل الرياضي المناسب لتحديد مكان وجود نموذج على طول هذا الطيف، وتقدير الصيغات المرتبطة به.

في سلسلة من الأوراق البحثية الأساسية، وسع فابنيك وشيرفونينكيس نظرية تقارب الترددات النسبية إلى فئات دوال أكثر عمومية (Vapnik ، 1964 ، Vapnik and Chervonenkis 1971 ، Vapnik and Chervonenkis 1968 ، and Chervonenkis 1991 ، Vapnik and Chervonenkis 1981 ، Chervonenkis 1974 ، Chervonenkis 1974). أحد المساهمات الرئيسية لهذا النوع من العمل هو بعد-Chervonenkis (VC)، والذي يقيس (فكرة واحدة عن) مدى تعقيد (المرونة) لفئة النموذج. علاوة على ذلك، فإن إحدى نتائجهما الرئيسية تحديد الفرق بين الخطأ التجاريي وخطأ السكان كدالة بعد VC وعدد العينات:

$$P(R[p, f] - R_{\text{emp}}[\mathbf{X}, \mathbf{Y}, f] < \alpha) \geq 1 - \delta \text{ for } \alpha \geq c\sqrt{(VC - \log \delta)/n}.$$

$\delta > 0$ هو احتمال انتهاء الحد، و α هو الحد الأعلى لفجوة التعميم، و n هو حجم مجموعة البيانات. أخيراً، $c > 0$ هو ثابت يعتمد فقط على حجم الخسارة التي يمكن تكبدها. قد يكون أحد استخدامات الحد هو توصيل القيم المرغوبة لـ δ و α وتحديد عدد العينات المراد جمعها. يحدد بعد VC أكبر عدد من نقاط البيانات التي يمكننا تعين أي تصنيف عشوائي (ثنائي) لها ولكل منها تجد نموذجاً ملائماً للفئة يتافق مع هذا التصنيف. على سبيل المثال، النماذج الخطية على المدخلات ذات الأبعاد لها أبعاد VC . من السهل ملاحظة أن الخط يمكنه تعين أي تصنيف ممكن لثلاث نقاط في بعدين، ولكن ليس لأربع. لسوء الحظ، تميل النظرية إلى التشاؤم المفرط بالنسبة للنماذج الأكثر تعقيداً والحصول على هذا الضمان يتطلب عادةً أمثلة أكثر بكثير مما هو مطلوب بالفعل لتحقيق معدل الخطأ المطلوب. لاحظ أيضاً أن تحديد فئة النموذج و δ . معدل الخطأ لدينا يتخلل مرة أخرى مع المعدل المعتاد $(1/\sqrt{n})$. يبدو من غير المحتمل أن نتمكن من القيام بعمل أفضل من حيث. ومع ذلك، نظراً لأننا نغير فئة النموذج، يمكن أن يقدم VC صورة متشائمة لفجوة التعميم.

4.6.4 الملخص

الطريقة الأكثر مباشرة لتقدير النموذج هي الرجوع إلى مجموعة اختبار تتكون من بيانات غير مرئية من قبل. توفر تقييمات مجموعة الاختبار تقديرًا غير متحيز للخطأ الحقيقي وتتقارب مع المعدل المطلوب $(1/\sqrt{n})$ مع نمو مجموعة الاختبار. يمكننا تقديم فترات ثقة تقريرية بناءً على توزيعات مقاربة دقيقة أو فترات ثقة محلاًدة صالحة للعينة بناءً على ضمانات عينة محدودة (أكثُر تحفظاً). إن تقييم مجموعة الاختبارات في الواقع هو حجر الأساس لأبحاث التعلم الآلي الحديثة. ومع ذلك، نادرًا ما تكونمجموعات الاختبار مجموعات اختبار حقيقة (ويستخدمها باحثون متعددون مارأً وتكراراً). بمجرد استخدام نفس مجموعة الاختبار لتقدير نماذج متعددة، قد يكون من الصعب التحكم في الاكتشاف الخاطئ. هذا يمكن أن يسبب مشاكل ضخمة من الناحية النظرية. من الناحية العملية، تعتمد أهمية المشكلة على حجم مجموعات الانتظار المعنية وما إذا كانت تستخدم فقط لاختيار المعلومات الفائقة أو ما إذا كانت تقوم بتسريب المعلومات بشكل مباشر أكثر. ومع ذلك، فمن الممارسات الجيدة تنظيم مجموعات اختبار حقيقة (أو متعددة) وأن تكون متحفظاً قدر الإمكان بشأن عدد مرات استخدامها.

على أمل تقديم حل أكثر إرضاءً، طور منظرو التعلم الإحصائي طرقاً لضمان التقارب المنتظم على فئة النموذج. إذا اقترب خطأ تجريبي لكل نموذج بالفعل مع خطأه الحقيقي في وقت واحد، فعندها لنا الحرية في اختيار النموذج الأفضل أداءً، وتقليل خطأ التدريب، مع العلم أنه سيؤدي أيضاً أداءً جيداً بالمثل على بيانات الانتظار. بشكل حاسم، يجب أن تعتمد أي من هذه النتائج على بعض خصائص فئة النموذج. قدم فلايديمير فابنيك وأليكسبي تشيرنوفينكيس بعد VC وقدمو نتائج تقارب موحدة تتطبق على جميع الطرز في فئة VC . إن خطاء التدريب لجميع

النماذج في الفصل مضمونة (في نفس الوقت) لتكون قريبة من أخطائهم الحقيقة، ومضمونة لتقترب من معدالتها ($1/\sqrt{n}$). بعد الاكتشاف الثوري لبعد VC، تم اقتراح العديد من تدابير التعقيد البديلة، كل منها يسهل ضمان التعميم المماثل analogous generalization. انظر Boucheron et al (2005) لمناقشة تفصيلية للعديد من الطرق المتقدمة لقياس تعقيد الدالة. لسوء الحظ، بينما أصبحت مقاييس التعقيد هذه أدوات مفيدة على نطاق واسع في النظرية الإحصائية، فقد تبين أنها عاجزة (كما تم تطبيقها بشكل مباشر) لشرح سبب تعميم الشبكات العصبية العميقه. غالباً ما تحتوي الشبكات العصبية العميقه على ملايين المعلمات (أو أكثر)، ويمكنها بسهولة تعين تسميات عشوائية لمجموعات كبيرة من النقاط. ومع ذلك، فإنها تعم جيداً على المشكلات العملية، ومن المدهش أنها غالباً ما تعم بمثلك أفضل، عندما تكون أكبر وأعمق، على الرغم من تكبدها أبعاداً أكبر لـ VC. في الفصل التالي، سوف نعيد النظر في التعميم في سياق التعلم العميق.

4.6.5 التمارين

- إذا كنا نرغب في تقدير الخطأ لمودج ثابت f مع 0.0001 واحتمال أكبر من 99.9% . فكم عدد العينات التي تحتاجها؟
- افرض أن شخصاً آخر يمتلك مجموعة اختبار معنونة \mathcal{D} ولا يوفر سوى المدخلات (الميزات) غير المسممة unlabeled. افترض الآن أنه لا يمكنك الوصول إلى تسميات مجموعة الاختبار إلا عن طريق تشغيل نموذج f (لا توجد قيود مفروضة على فئة النموذج) على كل من المدخلات غير المسممة وتلقي الخطأ المقابل ($f_{\mathcal{D}}$). كم عدد النماذج التي تحتاج إلى تقييمها قبل ترتيب مجموعة الاختبار بأكملها وبالتالي قد يبدو أنها تحتوي على خطأ 0 ، بغض النظر عن خطأ الحقيقى؟
- ما هو بُعد VC لفئة كثيرات الحدود من الدرجة الخامسة $?5^{\text{th}}\text{-order polynomials}$
- ما هو بُعد VC للمستويات المحاذية للمحور axis-aligned rectangles في البيانات ثنائية الأبعاد؟

4.7 التحول البيئي والتوزيع

في الأقسام السابقة، عملنا من خلال عدد من التطبيقات العملية للتعلم الآلي، ونلائم النماذج لمجموعة متنوعة منمجموعات البيانات. ومع ذلك، لم نتوقف أبداً عن التفكير في مصدر البيانات في المقام الأول أو ما نخطط لفعله في النهاية بمحركات نماذجنا. في كثير من الأحيان، يندفع مطورو التعلم الآلي الذين يمتلكون البيانات لتطوير نماذج دون التوقف مؤقتاً للنظر في هذه المشكلات الأساسية.

يمكن إرجاع العديد من عمليات نشر التعلم الآلي الفاشلة إلى هذا النمط. في بعض الأحيان، يبدو أن النماذج تعمل بشكل رائع كما تم قياسها من خلال دقة مجموعة الاختبار ولكنها تفشل بشكل كارثي في الشرع عندما يتغير توزيع البيانات فجأة. بشكل أكثر مكرراً، في بعض الأحيان، يمكن أن يكون نشر نموذج ما هو العامل المحفز الذي يزعج توزيع البيانات. لنفترض، على سبيل المثال، أننا دربنا نموذجاً للتنبؤ بمن سيسلد مقابل التخلف عن سداد القرض، ووجدنا أن اختيار مقدم الطلب للأحدية كان مرتبطة بمخاطر التخلف عن السداد (تشير أوكسفورد إلى السداد، وتشير الأحدية الرياضية إلى التقصير). قد نميل بعد ذلك إلى منح قروض لجميع المتقدمين الذين يرتدون أحذية أوكسفورد ورفض ارتداء جميع المتقدمين للأحدية الرياضية.

في هذه الحالة، قد يكون لفقرة غير مدروسة من التعرف على الأنماط إلى اتخاذ القرار وفشلنا في التفكير النقدي في البيئة عوّاقب وخيمة. بالنسبة للمبتدئين، بمجرد أن بدأنا في اتخاذ القرارات بناءً على الأحدية، سيتعرّف العملاء على سلوكهم ويفسرونها. قبل مضي وقت طويٍ، كان جميع المتقدمين يرتدون أحذية أوكسفورد، دون أي تحسن متزامن في الجدارة الائتمانية. يستغرق الأمر دقيقة لاستيعاب هذا لأن مشكلات مماثلة تكثّف العديد من تطبيقات التعلم الآلي: من خلال تقديم قراراتنا المستندة إلى النموذج للبيئة، قد نكسر النموذج.

بينما لا يمكننا إعطاء هذه الموضوعات معالجة كاملة في قسم واحد، فإننا نهدف هنا إلى الكشف عن بعض الاهتمامات المشتركة، وتحفيز التفكير النقدي المطلوب لاكتشاف هذه المواقف مبكراً، وتحقيق الضرب، واستخدام التعلم الآلي بمسؤولية. بعض الحلول بسيطة (أسأل عن البيانات "الصحيحة")، وبعضها صعب تقنياً (تنفيذ نظام التعلم المعزز reinforcement learning system)، والبعض الآخر يتطلب أن نتخطى مجال التنبؤ الإحصائي تماماً ونكافح الأسئلة الفلسفية الصعبة المتعلقة بالأخلاقيات. تطبيق الخوارزميات.

4.7.1 أنواع تحول التوزيع Types of Distribution Shift

للبدء، نتمسك بإعداد التنبؤ السلبي مع الأخذ في الاعتبار الطرق المختلفة التي قد تتغير بها توزيعات البيانات وما يمكن فعله لإيقاذ أداء النموذج. في أحد الإعدادات الكلاسيكية، نفترض أنه تمأخذ عينات من بيانات التدريب الخاصة بنا من بعض التوزيعات $p_S(x, y)$ ولكن بيانات الاختبار الخاصة بنا ستتألف من أمثلة غير مسمة مأخوذة من توزيع مختلف $p_T(x, y)$. بالفعل، يجب أن نواجه حقيقة واقعية. في غياب أي افتراضات حول كيفية ارتباط p_S و p_T ببعضها البعض، فإن تعلم مصنف قوي أمر مستحيل.

النظري في مشكلة التصنيف الثنائي binary classification problem، حيث نرغب في التمييز بين الكلاب والقطط. إذا كان التوزيع يمكن أن يتغير بطرق عشوائية، فإن إعدادنا يسمح بالحالة المرضية التي يظل فيها التوزيع على المدخلات ثابتة: $(x) = p_T(x) = p_S(x)$ ولكن جميع التسميات

مقلوبة: $p_S(y | x) = 1 - p_T(y | x)$. بعبارة أخرى، إذا استطاع الله أن يقرر فجأة أن كل "القطط" أصبحت الآن كلاباً وأن ما نطلق عليه سابقاً "كلاب" أصبح الآن قططاً – دون أي تغيير في توزيع المدخلات (x) ، فلا يمكننا تمييز هذا الإعداد عن واحد التي لم يتغير فيها التوزيع على الإطلاق.

لحسن الحظ، في ظل بعض الافتراضات المقيدة حول الطرق التي قد تتغير بها بياناتنا في المستقبل، يمكن للخوارزميات المبدئية أن تكتشف التحول shift وأحياناً تكيف بشكل سريع، مما يحسن دقة المصنف الأصلي.

4.7.1.1. التحول المتغير Covariate Shift

من بين فئات تحول التوزيع shift، قد يكون التحول المتغير covariate shift، هو الأكثر دراسة على نطاق واسع. هنا، نفترض أنه بينما قد يتغير توزيع المدخلات بمرور الوقت، فإن دالة وضع العلامات (التسميات)، أي التوزيع الشرطي $(x | y) P$ لا تتغير. يسمى الإحصائيون لهذا التحول المتغير covariate shift لأن المشكلة تنشأ بسبب تحول في توزيع المتغيرات المشتركة (الميزات). بينما يمكننا أحياناً التفكير في تحول التوزيع دون التذرع بالسببية، نلاحظ أن التحول المتغير هو الافتراض الطبيعي الذي يجب استدعاه في الإعدادات التي نعتقد فيها أن x تسبب y .

ضع في اعتبارك التحدي المتمثل في التمييز بين القطط والكلاب. قد تكون بيانات التدريب الخاصة بنا من صور من النوع الموضح في الشكل 4.7.1.



الشكل 4.7.1 بيانات التدريب للتمييز بين القطط والكلاب.

يطلب منافٍ وقت الاختبار تصنيف الصور في الشكل 4.7.2.



الشكل 4.7.2 بيانات الاختبار للتمييز بين القطط والكلاب.

ت تكون مجموعة التدريب من صور photos، بينما تحتوي مجموعة الاختبار على رسوم متحركة cartoons فقط. يمكن أن يؤدي التدريب على مجموعة بيانات ذات خصائص مختلفة إلى حد كبير عن مجموعة الاختبار إلى حدوث مشكلة في غياب خطة متماسكة لكيفية التكيف مع المجال الجديد.

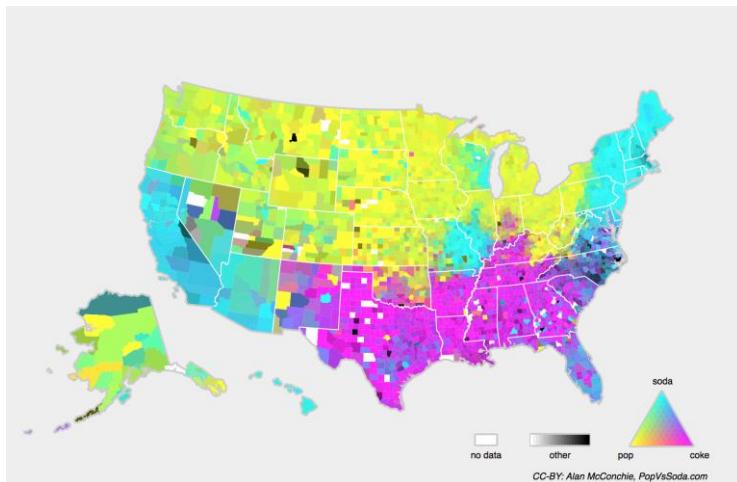
4.7.1.2 تحول التسمية Label Shift

يصف تحول التسمية Label Shift المشكلة العكسية converse problem. هنا، نفترض أن التسمية الهاشمية (y) P يمكن أن تتغير ولكن التوزيع الشرطي للفئة ($y | x$) P يظل ثابتاً عبر المجالات. إن تحول التسمية هو افتراض معقول يجب القيام به عندما نعتقد أن هذه y تسبب x . على سبيل المثال، قد نرغب في التنبؤ بالتشخيصات في ضوء أعراضها (أو غيرها من المظاهر)، حتى مع تغير الانتشار النسبي للتشخيصات بمرور الوقت. تحول التسمية هو الافتراض المناسب هنا لأن الأمراض تسبب الأعراض في بعض الحالات المتدهورة، يمكن أن يحدث تحول التسمية وافتراضات التحول المتغير في وقت واحد. على سبيل المثال، عندما تكون التسمية حتمية، سيتم استيفاء افتراض التحول المتغير، حتى عندما تكون y تسبب x . ومن المثير للاهتمام، في هذه الحالات، أنه غالباً ما يكون من المفيد العمل بالطرق التي تتدفق من افتراض تحول التسمية. وذلك لأن هذه الأساليب تمثل إلى التعامل مع الكائنات التي تبدو وكأنها تسميات (غالباً ما تكون منخفضة الأبعاد)، على عكس الكائنات التي تشبه المدخلات، والتي تمثل إلى أن تكون عالية الأبعاد في التعلم العميق.

4.7.1.3 تحول المفهوم Concept Shift

قد نواجه أيضاً مشكلة تحول Concept Shift المفهوم ذات الصلة، والتي تنشأ عندما تغير تعريفات التسميات ذاتها. هذا يبدو غريباً – القطة قطة، أليس كذلك؟ ومع ذلك، تخضع الفئات الأخرى للتغييرات في الاستخدام بمرور الوقت. تخضع المعايير التشخيصية للمرض العقلي، والسميات العصرية، والوظائف، لقدر كبير من تغيير المفهوم. اتضح أننا إذا تجولنا في جميع أنحاء الولايات المتحدة، وقمنا بتغيير مصدر بياناتنا حسب المنطقة الجغرافية، فسنجد تحولاً كبيراً في المفهوم فيما يتعلق بتوزيع أسماء المشروبات الغازية كما هو موضح في الشكل 4.7.3.

إذا أردنا بناء نظام ترجمة آلية machine translation system، فقد يختلف التوزيع $| y$ P حسب موقعنا. قد يكون من الصعب اكتشاف هذه المشكلة. قد نأمل في استغلال المعرفة التي لا تحدث إلا بشكل تدريجي إما بالمعنى الزمني أو الجغرافي.



الشكل 4.7.3 تحول المفهوم في أسماء المشروبات الغازية في الولايات المتحدة.

4.7.2 أمثلة على تحول التوزيع

قبل الخوض في الشكليات والخوارزميات، يمكننا مناقشة بعض المواقف الملهمة التي قد لا يكون فيها المتغير المشترك أو تحول المفهوم واضحًا.

4.7.2.1 Medical Diagnostics

تخيل أنك تريد تصميم خوارزمية لاكتشاف السرطان. تقوم بجمع البيانات من الأشخاص الأصحاء والمرضى وتقوم بتدريب الخوارزمية الخاصة بك. إنه يعمل بشكل جيد، مما يمنحك دقة عالية وتخالص إلى أنك مستعد لمهنة ناجحة في التشخيص الطبي. ليس بهذه السرعة.

قد تختلف التوزيعات التي أدت إلى بيانات التدريب وتلك التي ستواجهها في البرية اختلافاً كبيراً. حدث هذا لشركة ناشئة مؤسفة عمل معها بعضاً (المؤلفون) منذ سنوات. كانوا يطورون اختباراً للدم لمرض يصيب في الغالب كبار السن من الرجال ويأملون في دراسته باستخدام عينات الدم التي جمعوها من المرضى. ومع ذلك، فإن الحصول على عينات دم من الرجال الأصحاء أكثر صعوبة بكثير من المرضى الموجودين بالفعل في النظام. للتعويض، طلبت الشركة الناشئة التبع بالدم من الطلاب في الحرم الجامعي لتكون بمثابة ضوابط صحية في تطوير اختباراتهم. ثم سألوا عمما إذا كان بإمكاننا مساعدتهم في بناء مصنف لاكتشاف المرض.

كما أوضحنا لهم، سيكون من السهل بالفعل التمييز بين الأفواج الصحية والمريضة بدقة شبه كاملة. ومع ذلك، هذا لأن الأشخاص الخاضعين للاختبار مختلفون في العمر ومستويات الهرمونات والنشاط البدني والنظام الغذائي واستهلاك الكحول والعديد من العوامل الأخرى غير المرتبطة

بالمرض. لم يكن هذا هو الحال مع المرضى الحقيقيين. نظرًا لإجراءات أخذ العينات الخاصة بهم، يمكننا أن نتوقع مواجهة تحول متغير شديد. علاوة على ذلك، من غير المحتمل أن تكون هذه الحالة قابلة للتصحيح بالطرق التقليدية. باختصار، لقد أهدروا مبلغًا كبيرًا من المال.

4.7.2.2 السيارات ذاتية القيادة

لنفترض أن شركة ما أرادت الاستفادة من التعلم الآلي لتطوير سيارات ذاتية القيادة. أحد المكونات الرئيسية هنا هو جهاز الكشف roadside detector على جانب الطريق. نظرًا لأن الحصول على البيانات المشروحة الحقيقية باهظ التكلفة، فقد كانت لديهم فكرة (ذكية ومشكوك فيها) لاستخدام البيانات التركيبية synthetic data من محرك عرض الألعاب كبيانات تدريب إضافية. لقد نجح هذا الأمر جيدًا في "بيانات الاختبار" المستمدة من محرك العرض. للأسف، داخل سيارة حقيقة كانت كارثة. كما اتضح، تم تقديم جانب الطريق بنسيج بسيط للغاية. والأهم من ذلك، أن كل جوانب الطريق قد تم تقديمها بنفس الملمس وتعرف كاشف جانب الطريق على هذه "الميزة" بسرعة كبيرة.

حدث شيء مشابه للجيش الأمريكي عندما حاولوا اكتشاف الدبابات في الغابة لأول مرة. التقاطوا صورًا جوية للغابة بدون دبابات، ثم قادوا الدبابات إلى الغابة والتقطوا مجموعة أخرى من الصور. يبدو أن المصنف يعمل بشكل مثالى. لسوء الحظ، فقد تعلمت فقط كيفية التمييز بين الأشجار ذات الظلال من الأشجار التي ليس لها ظلال – تم التقاط المجموعة الأولى من الصور في الصباح الباكر، والمجموعة الثانية عند الظهر.

4.7.2.3 التوزيعات غير الثابتة Nonstationary Distributions

ينشأ موقف أكثر دقة عندما يتغير التوزيع بيئيًّا (يُعرف أيضًا بالتوزيع غير الثابت) ولا يتم تحديث النموذج بشكل كافٍ. فيما يلي بعض الحالات النموذجية.

- نقوم بتدريب نموذج إعلان حسابي ثم نفشل في تحديه بشكل متكرر (على سبيل المثال، ننسى أن ندمج جهازًا جديداً غامضًا يسمى iPad تم إطلاقه للتو).
- نحن نبني عامل تصفية البريد العشوائي spam filter. إنه يعمل جيدًا في اكتشاف جميع الرسائل غير المرغوب فيها التي رأيناها حتى الآن. ولكن بعد ذلك، يستيقظ مرسلي البريد العشوائي ويصنعون رسائل جديدة تبدو مختلفة عن أي شيء رأيناه من قبل.
- نحن نبني نظام توصية المنتج product recommendation system. إنه يعمل طوال فصل الشتاء ولكنه يستمر بعد ذلك في التوصية بقيعات سانتا لفترة طويلة بعد عيد الميلاد.

4.7.2.4. المزيد من الحكايات More Anecdotes

- نبني جهاز كشف الوجه face detector. يعمل بشكل جيد على جميع المعايير. لسوء الحظ، فشل في بيانات الاختبار – الأمثلة المخالفة هي لقطات قربة حيث يملأ الوجه الصورة بأكملها (لم تكن مثل هذه البيانات موجودة في مجموعة التدريب).
- نحن نبني محرك بحث على شبكة الإنترن特 لسوق الولايات المتحدة ونريد نشره في المملكة المتحدة.
- نقوم بتدريب مصنف الصور عن طريق تجميع مجموعة بيانات كبيرة حيث يتم تمثيل كل مجموعة من مجموعة كبيرة من الفئات بالتساوي في مجموعة البيانات، لنقل 1000 فئة، ممثلة بـ 1000 صورة لكل منها. ثم ننشر النظام في العالم الحقيقي، حيث يكون التوزيع الفعلي للتصنيفات للصور الفوتوغرافية غير منتظم بالتأكيد.

4.7.3. تصحيح تحول التوزيع Correction of Distribution Shift

كما ناقشنا، هناك العديد من الحالات التي يختلف فيها توزيع التدريب والاختبار ($P(\mathbf{x}, y)$). في بعض الحالات، تكون محوظتين وتعمل النماذج على الرغم من تغيير المتغير أو التسمية أو المفهوم. في حالات أخرى، يمكننا أن نفعل ما هو أفضل من خلال استخدام استراتيجيات قائمة على المبادئ للتعامل مع هذا التحول. ينمو الجزء المتبقい من هذا القسم بدرجة أكبر من الناحية الفنية. يمكن للقارئ الذي نفذ صبره المتتابعة إلى القسم التالي لأن هذه المادة ليست شرطاً أساسياً للمفاهيم اللاحقة.

4.7.3.1. الخطر التجريبية والخطر Empirical Risk and Risk

دعنا نفكّر أولاً فيما يحدث بالضبط أثناء تدريب النموذج: نحن نكرر الميزات والتصنيفات المرتبطة ببيانات التدريب $\{(\mathbf{x}_n, y_n), (\mathbf{x}_1, y_1), \dots\}$ ونحدّث معلمات النموذج f بعد كل دفعات صغيرة، من أجل التبسيط، لا نفكّر في التنظيم regularization، لذلك فإننا نقلل إلى حد كبير الخطأ في التدريب:

$$\underset{f}{\text{minimize}} \frac{1}{n} \sum_{i=1}^n l(f(\mathbf{x}_i), y_i),$$

حيث l هي دالة الخطأ التي تقيس "مدى سوء" التنبؤ (\mathbf{x}_i, f) الذي يُعطي التسمية المرتبطة به y_i . يسمى الإحصائيون المصطلح في (4.7.1) الخطر التجريبية empirical risk. الخطر التجريبي هو متوسط الخطأ على بيانات التدريب لتقرير الخطأ، وهو توقع الخسارة على مجموعة البيانات بأكملها المستمدّة من توزيعها الحقيقي $p(\mathbf{x}, y)$:

$$E_p(\mathbf{x}, y)[l(f(\mathbf{x}), y)] = \int \int l(f(\mathbf{x}), y)p(\mathbf{x}, y)d\mathbf{x}dy.$$

ومع ذلك، من الناحية العملية، لا يمكننا عادةً الحصول على مجموعة البيانات بالكامل. وبالتالي، فإن تقليل الخطأ التجاري empirical risk minimization، والذي يقلل من المخاطر التجريبية في (4.7.1)، هو استراتيجية عملية للتعلم الآلي، على أمل التقرير من تقليل المخاطر.

4.7.3.2 تصحيح التحول المتغير Covariate Shift Correction

افترض أننا نريد تقدير بعض التبعية $(\mathbf{x} | y) P$ التي قمنا بتسمية البيانات (\mathbf{x}_i, y_i) الخاصة بها. لسوء الحظ، يتم استخلاص الملاحظات \mathbf{x}_i من توزيع بعض المصادر بدلاً من التوزيع المستهدف. لحسن الحظ، فإن افتراض التبعية يعني أن التوزيع المشروط لا يتغير: $= (\mathbf{x} | y) P = (\mathbf{x} | y) q(\mathbf{x})$. إذا كان توزيع المصدر $(\mathbf{x}) q$ "خطأً"، فيمكننا تصحيح ذلك باستخدام الهوية البسيطة التالية في المخاطرة:

$$\int \int l(f(\mathbf{x}), y) p(y | \mathbf{x}) p(\mathbf{x}) d\mathbf{x} dy = \int \int l(f(\mathbf{x}), y) q(y | \mathbf{x}) q(\mathbf{x}) \frac{p(\mathbf{x})}{q(\mathbf{x})} d\mathbf{x} dy.$$

عبارة أخرى، نحتاج إلى إعادة وزن reweigh كل مثال من البيانات بنسبة احتمالية استخلاصه من التوزيع الصحيح إلى التوزيع الخطأ:

$$\beta_i \stackrel{\text{def}}{=} \frac{p(\mathbf{x}_i)}{q(\mathbf{x}_i)}.$$

بتوصيل الوزن β_i لكل مثال بيانات (\mathbf{x}_i, y_i) يمكننا تدريب نموذجنا باستخدام تقليل المخاطر التجريبية الموزونة weighted empirical risk minimization:

$$\underset{f}{\text{minimize}} \frac{1}{n} \sum_{i=1}^n \beta_i l(f(\mathbf{x}_i), y_i).$$

للأسف، لا نعرف هذه النسبة، لذا قبل أن نتمكن من فعل أي شيء مفيد نحتاج إلى تقديرها. توفر العديد من الطرق، بما في ذلك بعض الأساليب النظرية المشغلة التي تحاول إعادة معالجة عامل التوقع مباشرة باستخدام الحد الأدنى من المعايير أو الحد الأقصى لمبدأ الانتروبيا. لاحظ أنه لأي نهج من هذا القبيل، نحتاج إلى عينات مأخوذة من كلا التوزيعين - "صحيح" p ، على سبيل المثال، من خلال الوصول إلى بيانات الاختبار، وتلك المستخدمة لإنشاء مجموعة التدريب q (الأخير متاح بشكل ضئيل). لاحظ مع ذلك، أننا نحتاج فقط إلى ميزات $(\mathbf{x}) p \sim \mathbf{x}$; لا نحتاج للوصول إلى التسميات $(y) p \sim y$.

في هذه الحالة، يوجد نهج فعال للغاية سيعطي نتائج جيدة تقريرًا مثل الأصل: الانحدار اللوجستي logistic regression، وهو حالة خاصة من انحدار softmax (انظر القسم 4.1) للتصنيف الثنائي. هذا هو كل ما هو مطلوب لحساب نسب الاحتمال المقدرة. نتعلم المصنف للتمييز بين

البيانات المستمدة من (\mathbf{x}, p) والبيانات المستمدة من (\mathbf{x}, q) . إذا كان من المستحيل التمييز بين التوزيعين، فهذا يعني أن الحالات المرتبطة من المرجح أن تأتي من أحد التوزيعين. من ناحية أخرى، فإن أي حالات يمكن تمييزها بشكل جيد يجب أن يكون لها وزن زائد أو ناقص بشكل كبير وفقاً لذلك.

من أجل التبسيط، افترض أن لدينا عدداً متساوياً من المثلثات من كلا التوزيعين (\mathbf{x}, p) و (\mathbf{x}, q) على التوالي. قم الآن بالإشارة إلى التسميات z إلى 1 للبيانات المستمدة من p للبيانات المستمدة من q . ثم يتم إعطاء الاحتمال في مجموعة بيانات مختلطة بواسطة

$$P(z = 1 | \mathbf{x}) = \frac{p(\mathbf{x})}{p(\mathbf{x}) + q(\mathbf{x})} \text{ and hence } \frac{P(z = 1 | \mathbf{x})}{P(z = -1 | \mathbf{x})} = \frac{p(\mathbf{x})}{q(\mathbf{x})}.$$

وبالتالي، إذا استخدمنا نهج الانحدار اللوجستي، حيث $h(\mathbf{x}) = \ln(p(\mathbf{x}) / q(\mathbf{x}))$ هي دالة ذات معلمات، يتبع ذلك

$$\beta_i = \frac{1/(1 + \exp(-h(\mathbf{x}_i)))}{\exp(-h(\mathbf{x}_i))/(1 + \exp(-h(\mathbf{x}_i)))} = \exp(h(\mathbf{x}_i)).$$

نتيجة لذلك، نحتاج إلى حل مشكلتين: الأولى للتمييز بين البيانات المستمدة من كلا التوزيعين، ثم مشكلة تقليل المخاطر التجريبية الموزونة في (4.7.5) حيث نزن المصطلحات حسب β_i .

الآن نحن جاهزون لوصف خوارزمية التصحيح. افترض أن لدينا مجموعة تدريب $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)\}$ ومجموعة اختبار غير مسمة $\{\mathbf{u}_1, \dots, \mathbf{u}_m\}$. بالنسبة إلى تحول المتغير shift covariate، نفترض أن يتم استخلاص \mathbf{x}_i لكل من $i \leq n$ بغض توزيعات المصدر و \mathbf{u}_i لكل $i \leq m$ مستمدون من التوزيع المستهدف. فيما يلي خوارزمية نموذجية لتصحيح التحول المتغير:

1. إنشاء مجموعة تدريب التصنيف الثنائي:

$$\{(\mathbf{x}_1, -1), (\mathbf{x}_2, -1), (\mathbf{u}_1, 1), \dots, (\mathbf{u}_m, 1)\}$$

2. تدريب مصنف ثئابي باستخدام الانحدار اللوجستي للحصول على دالة h .

3. وزن بيانات التدريب باستخدام $\beta_i = \exp(h(\mathbf{x}_i))$ أو أفضل $\beta_i = \min(\exp(h(\mathbf{x}_i)), c)$ لبعض ثابت c .

4. استخدام الأوزان β_i للتدريب على $\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_n, y_n)\}$ في (4.7.5).

لاحظ أن الخوارزمية أعلاه تعتمد على افتراض حاسم. لكي يعمل هذا المخطط، نحتاج إلى أن يكون لكل مثال بيانات في الهدف (على سبيل المثال، وقت الاختبار) احتمالية غير صفرية

لحدوثها في وقت التدريب. إذا وجدنا نقطة حيث $0 > p(\mathbf{x}) - q(\mathbf{x})$ ، فإن وزن الأهمية المقابل يجب أن يكون لانهائي.

4.7.3.3 تصحيح تحول التسمية Label Shift Correction

افرض أنتا نتعامل مع مهمة تصنيف مع الفئات k . باستخدام نفس الترميز في القسم 4.7.3.2، q و p هما توزيع المصدر (على سبيل المثال، وقت التدريب training time) والتوزيع المستهدف (على سبيل المثال، وقت الاختبار test time)، على التوالي. افترض أن توزيع التسميات يتغير بمرور الوقت: $p(y) \neq q(y)$ ، لكن التوزيع الشرطي للفئة يظل كما هو: $q(\mathbf{x} | y) = p(\mathbf{x} | y)$. إذا كان توزيع المصدر $q(y)$ "خطاً" ، فيمكننا تصحيح ذلك وفقاً للهوية التالية في الخطوة كما هو محدد في (4.7.2):

$$\int \int l(f(\mathbf{x}), y) p(\mathbf{x} | y) p(y) d\mathbf{x} dy = \int \int l(f(\mathbf{x}), y) q(\mathbf{x} | y) q(y) \frac{p(y)}{q(y)} d\mathbf{x} dy.$$

هنا، سوف تتوافق أوزان الأهمية الخاصة بنا مع نسب احتمالية التسمية

$$\beta_i \stackrel{\text{def}}{=} \frac{p(y_i)}{q(y_i)}.$$

أحد الأشياء اللطيفة حول تحول التسمية label shift هو أنه إذا كان لدينا نموذج جيد بشكل معقول لتوزيع المصدر، فيمكننا الحصول على تقديرات متسقة لهذه الأوزان دون الحاجة إلى التعامل مع البعد المحيط. في التعلم العميق، تميل المدخلات إلى أن تكون كائنات عالية الأبعاد مثل الصور، في حين أن التسميات غالباً ما تكون كائنات أبسط مثل الفئات.

لتقدير توزيع التسمية المستهدف، نأخذ أولاً المصنف الجيد الجاهز لدينا (عادةً ما يتم تدريبه على بيانات التدريب) ونحسب مصفوفة الارتكاك confusion matrix الخاصة به باستخدام مجموعة التحقق من الصحة (أيضاً من توزيع التدريب). مصفوفة الارتكاك (confusion matrix) هي مجرد مصفوفة، حيث يتوافق كل عمود مع فئة التسمية (الحقيقة الأساسية) \mathbf{C} ، ويتوافق كل صف مع الفئة المتوقعة لنموذجنا. تمثل قيمة كل خلية c_{ij} جزءاً من إجمالي التوقعات في مجموعة التتحقق حيث كان التسمية الحقيقية كانت i وتوقع نموذجنا j .

الآن، لا يمكننا حساب مصفوفة الارتكاك على البيانات الهدف مباشرةً، لأننا لا نرى تسميات الأمثلة التي نراها في البرية، إلا إذا استثمنا في خط أنابيب معقد للتعليقات التوضيحية في الوقت الفعلي. ومع ذلك، ما يمكننا القيام به هو متوسط جميع تنبؤاتنا في وقت الاختبار معًا، مما يؤدي إلى متوسط مخرجات النموذج $\hat{y} \in \mathbb{R}^k$ الذي i^{th} عنصر $(\hat{y}_i)^{\mu}$ هو جزء من التوقعات الإجمالية في مجموعة الاختبار حيث توقع نموذجنا.

اتضح أنه في ظل بعض الظروف المعتدلة – إذا كان المصنف لدينا دقيقاً بشكل معقول في المقام الأول، وإذا كانت البيانات المستهدفة تحتوي فقط على الفئات التي رأيناها من قبل، وإذا كان افتراض تحول التسمية ثابتاً في المقام الأول (أقوى افتراض هنا)، ثم يمكننا تقدير توزيع تسمية مجموعة الاختبار عن طريق حل نظام خطى بسيط

$$\mathbf{C}p(\mathbf{y}) = \hat{\mu}(\mathbf{y}),$$

لأنه كتقدير $\hat{\mu}(y_i) = c_{ij}p(y_j)$ يحمل لكل $i \leq k$ ، حيث (y_j) هو p عنصر متوجه توزيع التسمية ذي k^{th} أبعاد (\mathbf{y}) . إذا كان المصنف لدينا دقيقاً بما يكفي لتدلي به، فستكون مصفوفة الارتباط \mathbf{C} قابلة للعكس، وسنحصل على حل $\hat{\mu}(\mathbf{y}) = \mathbf{C}^{-1}\mathbf{p}(\mathbf{y})$

نظرًا لأننا نلاحظ التسميات على بيانات المصدر، فمن السهل تقدير التوزيع $q(y)$. ثم بالنسبة لأي مثال تدريبي i مع التسمية y_i ، يمكننا أن نأخذ النسبة المقدرة لدينا $(y_i)/q(y_i)$ لحساب الوزن β_i ، ونعرض هذافي تقليل المخاطر التجريبية الموزونة في (4.7.5).

4.7.3.4 تصحيح تحول المفهوم Concept Shift Correction

من الصعب إصلاح تحول المفهوم بطريقة مبدئية. على سبيل المثال، في حالة تغيرت فيها المشكلة فجأة من تمييز القطط عن الكلاب إلى تمييز بين الحيوانات البيضاء والسوداء، سيكون من غير المعقول افتراض أنه يمكننا القيام بعمل أفضل بكثير من مجرد جمع تسميات جديدة والتدريب من نقطة الصفر. لحسن الحظ، في الممارسة العملية، مثل هذه التحولات المتطرفة نادرة. بدلاً من ذلك، ما يحدث عادةً هو أن المهمة تتغير ببطء. لجعل الأمور أكثر واقعية، إليك بعض الأمثلة:

- في الإعلانات الحاسوبية، يتم إطلاق منتجات جديدة، وتتصبح المنتجات القديمة أقل شعبية. هذا يعني أن التوزيع على الإعلانات وشعبيتها يتغيران تدريجياً وأي متغير بنسبة النقر إلى الظهور يجب أن يتغير تدريجياً معه.
- تتدحرج عدسات كاميرات المرور تدريجياً بسبب التآكل البيئي، مما يؤثر على جودة الصورة بشكل تدريجي.
- يتغير محتوى الأخبار تدريجياً (على سبيل المثال، تظل معظم الأخبار دون تغيير ولكن تظهر قصص جديدة).

في مثل هذه الحالات، يمكننا استخدام نفس النهج الذي استخدمناه لشبكات التدريب لجعلها تتكيف مع التغيير في البيانات. بمعنى آخر، نستخدم أوزان الشبكة الحالية ونقوم ببساطة ببعض خطوات التحديث باستخدام البيانات الجديدة بدلاً من التدريب من البداية.

4.7.4. تصنیف مشاکل التعلم A Taxonomy of Learning Problems

مسلحين بالمعرفة حول كيفية التعامل مع التغييرات في التوزيعات، يمكننا الآن النظري بعض الجوانب الأخرى لصياغة مشكلة التعلم الآلي.

4.7.4.1 Batch Learning

في التعلم الجماعي Batch Learning، لدينا إمكانية الوصول إلى ميزات التدريب والتسميات $\{(x_1, y_1), \dots, (x_n, y_n)\}$ ، والتي نستخدمها لتدريب نموذج $f(x)$. في وقت لاحق، قمنا بنشر هذا النموذج لتسجيل بيانات جديدة (x, y) مستمدة من نفس التوزيع. هذا هو الافتراض الافتراضي لأي من المشاكل التي ناقشها هنا. على سبيل المثال، قد نقوم بتدريب جهاز الكشف عن القحط بناءً على الكثير من صور القحط والكلاب. بمجرد أن نقوم بتدريبه، نقوم بسحبه كجزء من نظام الرؤية الحاسوبية الذكي الذي يسمح للقطط بالدخول. ثم يتم تثبيته في منزل العميل ولا يتم تحديده مرة أخرى (باستثناء الظروف القصوى).

4.7.4.2 Online Learning

تخيل الآن أن البيانات (x_i, y_i) تصل عينة واحدة في كل مرة. بشكل أكثر تحديداً، افترض أنتا نلاحظ x أولاً، ثم تحتاج إلى التوصل إلى تقدير $f(x)$ و فقط بمجرد قيامنا بذلك، نلاحظه ونلتقي مكافأة أو نتحمل خسارة، نظراً لقرارنا. تقع العديد من المشاكل الحقيقية في هذه الفئة. على سبيل المثال، تحتاج إلى توقع سعر سهم الغد، وهذا يسمح لنا بالتداول بناءً على هذا التقدير وفي نهاية اليوم نكتشف ما إذا كان تقديرنا يسمح لنا بتحقيق ربح. بعبارة أخرى، في التعلم الالكتروني Online Learning، لدينا الدورة التالية حيث نعمل باستمرار على تحسين نموذجنا في ضوء الملاحظات الجديدة.

model $f_t \rightarrow$ data $x_t \rightarrow$ estimate $f_t(x_t) \rightarrow$ observation $y_t \rightarrow$ loss $l(y_t, f_t(x_t)) \rightarrow$ model f_{t+1}

Bandits 4.7.4.3

Bandits هي حالة خاصة للمشكلة المذكورة أعلاه. بينما في معظم مشاكل التعلم لدينا دالة f ذات معلمات باستمرار حيث نريد معرفة معلماتها (على سبيل المثال، شبكة عميق)، في مشكلة Bandits لدينا فقط عدد محدود من الأسلحة التي يمكننا سحبها، أي عدد محدود من الإجراءات يمكننا أخذها. ليس من المستغرب جداً أنه يمكن الحصول على ضمادات نظرية أقوى من حيث الأمثلية لهذه المشكلة الأسطو. ندرجها بشكل أساسي لأن هذه المشكلة غالباً ما يتم التعامل معها (بشكل مربك) كما لو كانت بيئه تعليمية مميزة.

4.7.4.4. وحدات التحكم Control

في كثير من الحالات تتذكر البيئة ما فعلناه. ليس بالضرورة بطريقة عدائية ولكنها ستتذكر فقط وستعتمد الاستجابة على ما حدث من قبل. على سبيل المثال، ستلاحظ وحدة التحكم في غالية القهوة درجات حرارة مختلفة اعتماداً على ما إذا كانت تسخن الغلاية مسبقاً. تعد خوارزميات وحدة التحكم PID (تناسبي تكاملي تفاضلي proportional-integral-derivative) خياراً شائعاً هناك. وبالمثل، سيعتمد سلوك المستخدم على موقع الأخبار على ما أظهرناه له سابقاً (على سبيل المثال، سيقرأ معظم الأخبار مرة واحدة فقط). تشكل العديد من هذه الخوارزميات نموذجاً للبيئة التي تعمل فيها بحيث يجعل قراراتها تبدو أقل عشوائية. في الآونة الأخيرة، تم أيضاً استخدام نظرية التحكم (على سبيل المثال، متغيرات PID) لضبط المعلمات الفائقة تلقائياً لتحقيق جودة أفضل لفك التشابك وإعادة البناء، وتحسين تنوع النص الذي تم إنشاؤه وجودة إعادة بناء الصور التي تم إنشاؤها (Shao et al., 2020).

4.7.4.5. التعلم المعزز Reinforcement Learning

في الحالة العامة لبيئة ذات ذاكرة، قد نواجه مواقف تحاول فيها البيئة التعاون معنا (ألعاب تعاونية cooperative games)، على وجه الخصوص للألعاب ذات المجموع غير الصافي non-zero-sum games، أو حالات أخرى حيث ستحاول البيئة الفوز. الشطرنج أو Go أو Backgammon أو StarCraft هي بعض الحالات في التعلم المعزز Reinforcement Learning. وبالمثل، قد نرغب في بناء وحدة تحكم جيدة للسيارات ذاتية القيادة. من المحتمل أن تستجيب السيارات الأخرى لأسلوب قيادة السيارة المستقلة بطرق غير بدائية، على سبيل المثال، محاولة تجنبها ومحاولة التسبب في وقوع حادث ومحاولة التعاون معها.

4.7.4.6. مراعاة البيئة Considering the Environment

أحد الفروق الرئيسية بين المواقف المختلفة المذكورة أعلاه هو أن نفس الإستراتيجية التي ربما نجحت طوال الوقت في حالة البيئة الثابتة، قد لا تعمل طوال الوقت عندما تكون البيئة قادرة على التكيف. على سبيل المثال، من المرجح أن تختفي فرصة المراجحة arbitrage opportunity التي اكتشفها المتداول بمجرد أن يبدأ في استغلالها. تحدد السرعة والطريقة التي تتغير بها البيئة إلى حد كبير نوع الخوارزميات التي يمكننا استخدامها. على سبيل المثال، إذا علمتنا أن الأشياء قد تتغير ببطء فقط، فيمكننا إجبار أي تقدير على التغيير ببطء فقط أيضاً. إذا علمينا أن البيئة قد تتغير على الفور، ولكن بشكل نادر جداً، فيمكننا السماح بذلك. هذه الأنواع من المعرفة ضرورية لعالم البيانات الطموح للتعامل مع تحول المفهوم، أي عندما تكون المشكلة التي يحاول حلها مع مرور الوقت.

4.7.5 الإنصاف والمساءلة والشفافية في التعلم الآلي Accountability, and Transparency in Machine Learning

أخيرًا، من المهم أن تذكر أنه عند نشر أنظمة التعلم الآلي، فأنت لا تقوم فقط بتحسين نموذج تنبؤي – فأنت تقدم عادةً أداة سُتستخدم (جزئياً أو كلياً) لأتمتة القرارات. يمكن أن تؤثر هذه الأنظمة التقنية على حياة الأفراد الخاضعين للقرارات الناتجة. لا تشير القفزة من التفكير في التنبؤات إلى القرارات أسلأة فنية جديدة فحسب، بل تثير أيضاً عدداً كبيراً من الأسئلة الأخلاقية التي يجب مراعاتها بعناية. إذا كنا ننشر نظاماً تشخيصياً طيباً، فنحن بحاجة إلى معرفة الفئات السكانية التي قد يعمل بها وأيها قد لا يعمل. قد يؤدي التغاضي عن المخاطر المتوقعة على رفاهية مجموعة سكانية فرعية إلى تقديم رعاية أدنى.علاوة على ذلك، بمجرد التفكير في أنظمة صنع القرار، يجب أن نتراجع ونعيد النظر في كيفية تقييمنا لتقنيتنا. من بين النتائج الأخرى لهذا التغيير في النطاق، سنجد أن الدقة نادراً ما تكون المقاييس الصحيح. على سبيل المثال، عند ترجمة التنبؤات إلى أفعال، غالباً ما نرحب في مراعاة حساسية التكالفة المحتملة للخطأ بطرق مختلفة. إذا كان من الممكن النظر إلى إحدى طرق سوء تصنيف صورة ما على أنها خفة عرقية، في حين أن التصنيف الخاطئ إلى فئة مختلفة سيكون غير ضار، فقد نرحب في تعديل عتباتنا وفقاً لذلك، مع مراعاة القيم المجتمعية في تصميم بروتوكول اتخاذ القرار. نريد أيضاً توخي الحذر بشأن الكيفية التي يمكن أن تؤدي بها أنظمة التنبؤ إلى حلقات التغذية الراجعة. على سبيل المثال، ضع في اعتبارك أنظمة الشرطة التنبؤية predictive policing systems التي تخصل ضباط الدوريات في المناطق التي ترتفع فيها معدلات الجريمة المتوقعة. من السهل أن ترى كيف يمكن أن يظهر نمط مقلق:

1. الأحياء التي بها جرائم أكثر تحصل على المزيد من الدوريات.
2. وبالتالي، تم اكتشاف المزيد من الجرائم في هذه الأحياء، وإدخال بيانات التدريب المتاحة للتكرار في المستقبل.
3. يتعرض النموذج لمزيد من الإيجابيات، ويتوقع المزيد من الجرائم في هذه الأحياء.
4. في التكرار التالي، يستهدف النموذج المحدث الحي نفسه بشكل أكبر مما يؤدي إلى اكتشاف المزيد من الجرائم، وما إلى ذلك.

في كثير من الأحيان، لا يتم احتساب الآليات المختلفة التي يتم من خلالها اقتراح تنبؤات النموذج ببيانات التدريب الخاصة به في عملية النمذجة. يمكن أن يؤدي هذا إلى ما يسميه الباحثون حلقات التغذية الراجعة الجامحة runaway feedback loops. بالإضافة إلى ذلك، نريد توخي الحذر بشأن ما إذا كنا نعالج المشكلة الصحيحة في المقام الأول. تلعب الخوارزميات التنبؤية الآن دوراً كبيراً في التوسيط في نشر المعلومات. هل يجب أن يتم تحديد الأخبار التي يواجهها الفرد من خلال

مجموعة صفحات Facebook التي أعجبتهم؟ هذه مجرد أمثلة قليلة من بين العديد من المعضلات الأخلاقية الملحة التي قد تواجهها في مهنة تعلم الآلة.

4.7.6. الملخص

- في كثير من الحالات لا تأتي مجموعات التدريب والاختبار من نفس التوزيع. وهذا ما يسمى تحول التوزيع distribution shift.
- الخطر risk هو توقع الخسارة على مجموع البيانات المستمدة من توزيعها الحقيقي. ومع ذلك، عادة ما تكون هذه المجموعة بأكملها غير متوفرة. الخطر التجاري Empirical risk هو متوسط الخسارة على بيانات التدريب لتقرير المخاطر. في الممارسة العملية، نقوم بتقليل المخاطر التجريبية empirical risk minimization وفقاً للافتراضات المقابلة، يمكن اكتشاف تحول المتغير covariate shift وتحول التسمية label shift وتصحيحهما في وقت الاختبار. يمكن أن يصبح الفشل في تفسير هذا التحيز مشكلة في وقت الاختبار.
- في بعض الحالات، قد تذكر البيئة الإجراءات الآلية وتستجيب بطرق مفاجئة. يجب أن نأخذ في الحسبان هذا الاحتمال عند بناء النماذج والاستمرار في مراقبة الأنظمة الحية، والافتتاح على احتمال أن تصبح نماذجنا والبيئة متشابكة بطرق غير متوقعة.

4.7.7. التمارين

1. ماذا يمكن أن يحدث عندما نغير سلوك محرك البحث؟ ماذا يمكن أن يفعل المستخدمون؟ ماذا عن المعلنين؟
2. نفذ كاشف التحول المتغير covariate shift detector. تلميح: بناء مصنف.
3. استخدم مصحح تحول المتغير covariate shift corrector.
4. إلى جانب تحول التوزيع distribution shift، ما الذي يمكن أن يؤثر أيضاً على كيفية تقرير المخاطر التجريبية للمخاطر؟

البير بيترون متعدد الطبقات

5

5. البيرسيبترون متعدد الطبقات Perceptrons

في هذا الفصل، سوف نقدم لك أول شبكة عميقه حقاً. تُعرف أبسط الشبكات العميقه باسم البيرسيبترون متعدد الطبقات Multilayer Perceptrons، وهي تتكون من طبقات متعددة من الخلايا العصبية، كل منها مرتبطة تماماً بتلك الموجودة في الطبقة أدناه (التي تتلقى منها المدخلات) وت تلك الموجودة أعلى (والتي بدورها تؤثر عليها). على الرغم من أن التمايز التلقائي يسفل بشكل كبير تنفيذ خوارزميات التعلم العميق، إلا أنها ستنعمق في كيفية حساب هذه التدرجات gradients في الشبكات العميقه. بعد ذلك سنكون مستعدين لمناقشة القضايا المتعلقة بالاستقرار العددي وتهيئة المعلمات التي تعتبر أساسية لتدريب الشبكات العميقه بنجاح. عندما نقوم بتدريب مثل هذه النماذج عالية السعة، فإننا نخاطر بالضبط الزائد overfitting. وبالتالي، سنعيد النظر في التنظيم regularization والتعميم generalization للشبكات العميقه. طوال الوقت، نهدف إلى منحك فهماً قوياً ليس فقط للمفاهيم ولكن أيضاً لممارسة استخدام الشبكات العميقه. في نهاية هذا الفصل، نطبق ما قدمناه حتى الآن على حالة حقيقية: التنبؤ بأسعار المنزل. نرسل الأمور المتعلقة بالأداء الحسابي وقابلية التوسيع وكفاءة نماذجنا إلى الفصول اللاحقة.

5.1 البيرسيبترون متعدد الطبقات Multilayer Perceptrons

في القسم 4، قدمنا انحدار softmax (القسم 4.1)، وتنفيذ الخوارزمية من البداية (القسم 4.4) واستخدام واجهات برمجة التطبيقات APIs عالية المستوى (القسم 4.5). سمح لنا ذلك بتدريب المصنفات القادرة على التعرف على 10 فئات من الملابس من الصور منخفضة الدقة. على طول الطريق، تعلمنا كيفية تبديل البيانات data wrangle، وإجبار مخرجاتنا على توزيع احتمالي صالح، وتطبيق دالة خطأً مناسبة، وتقليلها فيما يتعلق بمعلمات نموذجنا. الآن وقد أتقنا هذه الميكانيكيات في سياق النماذج الخطية البسيطة، يمكننا إطلاق استكشافنا للشبكات العصبية العميقه، فئة النماذج الغنية نسبياً التي يهتم بها هذا الكتاب بشكل أساسي.

5.1.1. الطبقات المخفية Hidden Layers

وصفتنا التحولات الأفينية affine transformations في القسم 3.1.1.1 كتحولات خطية مع تحيز إضافي. للبدء، تذكر بنية النموذج المقابلة لمثال انحدار softmax الموضح في الشكل 4.1.1. يقوم هذا النموذج بتعيين المدخلات مباشرة إلى المخرجات عبر تحويل أفيني واحد، متبعاً بعملية softmax. إذا كانت تسمياتنا مرتبطة حقاً ببيانات الإدخال عن طريق تحويل أفيني بسيط، فسيكون هذا النهج كافياً. ومع ذلك، فإن الخطية (في التحولات الأفينية) هي افتراض قوي.

5.1.1.1. عيوب النماذج الخطية Limitations of Linear Models

على سبيل المثال، تشير الخطية *linearity* إلى الافتراض الأضعف للرتبة weaker assumption of monotonicity، أي أن أي زيادة في ميزتنا يجب أن تتسبب دائمًا في زيادة ناتج نموذجنا (إذا كان الوزن المقابل موجّهاً)، أو يتسبب دائمًا في انخفاض ناتج نموذجنا (إذا كان الوزن المقابل سلبي). في بعض الأحيان يكون هذا منطقياً. على سبيل المثال، إذا كاننا نحاول التنبؤ بما إذا كان الفرد سوف يسدد قرضاً، فقد نفترض بشكل معقول أن جميع الأشياء الأخرى متساوية، فمن المرجح دائمًا أن يسدد مقدم الطلب ذو الدخل المرتفع أكثر من الشخص ذي الدخل المنخفض. في حين أن هذه العلاقة رتيبة، من المحتمل ألا تكون مرتبطة خطياً باحتمال السداد. من المحتمل أن توافق الزيادة في الدخل من 0 دولار إلى 50000 دولار مع زيادة أكبر في احتمالية السداد مقارنة بالزيادة من 1 مليون دولار إلى 1.05 مليون دولار. قد تكون إحدى طرق التعامل مع هذا هي المعالجة اللاحقة لنتائجنا بحيث تصبح الخطية أكثر منطقية، باستخدام الخريطة اللوجستية logistic map (وبالتالي لوغاريتيم احتمالية النتيجة).

لاحظ أنه يمكننا بسهولة التوصل إلى أمثلة تنتهك الرتبة monotonicity. قل على سبيل المثال أننا نريد التنبؤ بالصحة كدالة في درجة حرارة الجسم. بالنسبة للأفراد الذين تزيد درجة حرارة الجسم عن 37 درجة مئوية (98.6 درجة فهرنهايت)، تشير درجات الحرارة المرتفعة إلى مخاطر أكبر. ومع ذلك، بالنسبة للأفراد الذين تقل درجة حرارة أجسامهم عن 37 درجة مئوية، فإن درجات الحرارة المنخفضة تشير إلى مخاطر أكبر! مرة أخرى، قد نحل المشكلة ببعض المعالجة المسبيقة الذكية، مثل استخدام المسافة من 37 درجة مئوية كميزة.

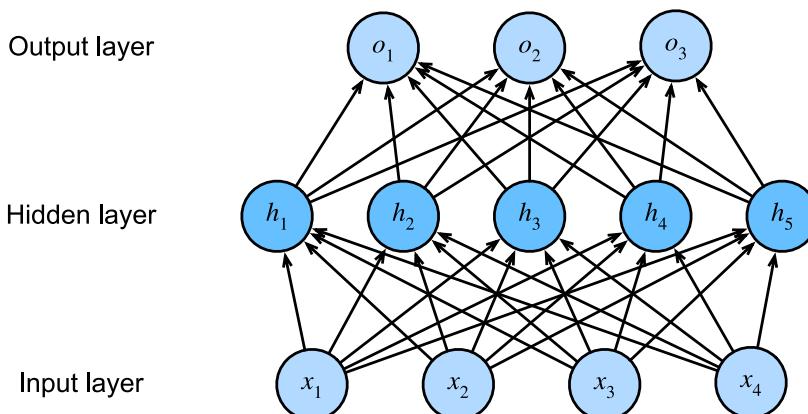
ولكن ماذا عن تصنيف صور القطط والكلاب؟ هل يجب أن تؤدي زيادة كثافة البكسل في الموقع (13، 17) دائمًا إلى زيادة (أو تقليل دائمًا) احتمالية أن الصورة تصور كلبًا؟ يتوافق الاعتماد على نموذج خططي مع الافتراض الضمني بأن المطلب الوحيد للتمييز بين القطط والكلاب هو تعيين سطوع وحدات البكسل الفردية. هذا النهج محكم عليه بالفشل في عالم يحافظ فيه عكس الصورة على الفئة.

ومع ذلك، على الرغم من العبث الواضح للخطية هنا، مقارنة بأمثلتنا السابقة، فمن غير الواضح أنه يمكننا معالجة المشكلة بإصلاح بسيط للمعالجة المسبيقة. وذلك لأن أهمية أي بكسل تعتمد بطرق معقدة على سياقها (قيم وحدات البكسل المحيطة). في حين أنه قد يكون هناك تمثيل لبياناتنا يأخذ في الاعتبار التفاعلات ذات الصلة بين ميزاتنا، وعلى رأسها سيكون النموذج الخططي مناسباً، فنحن ببساطة لا نعرف كيفية حسابه يدوياً. مع الشبكات العصبية العميقية، استخدمنا بيانات المراقبة لتعلم بشكل مشترك كلا من التمثيل عبر الطبقات المخفية والمتبوع الخططي الذي يعمل على هذا التمثيل.

تمت دراسة مشكلة اللاخطية nonlinearity هذه لمدة قرن على الأقل (Fisher، 1928). على سبيل المثال، تستخدم أشجار القرار decision trees في أبسط أشكالها سلسلة من القرارات الثنائية لاتخاذ قرار بشأن عضوية الفئة (Quinlan، 2014). وبالمثل، تم استخدام طرق التوازن kernel methods لعقود عديدة لنمذجة التبعيات غير الخطية (Aronszajn، 1950). وقد وجد هذا طريقه، على سبيل المثال، في نماذج الخيوط اللاملمعية nonparametric spline وطرق التوازن Wahba (1990) وطرق التوازن Schölkopf and Smola (2002). إنه أيضًا شيء يحله الدماغ بشكل طبيعي تماماً. بعد كل شيء، تتغذى الخلايا العصبية في الخلايا العصبية الأخرى التي بدورها تغذي الخلايا العصبية الأخرى مرة أخرى (Cajal and Azoulay، 1894). وبالتالي لدينا سلسلة من التحولات البسيطة نسبياً.

5.1.1.2 دمج الطبقات المخفية Incorporating Hidden Layers

يمكنا التغلب على قيود النماذج الخطية من خلال دمج طبقة مخفية واحدة أو أكثر. أسهل طريقة للقيام بذلك هي تكديس العديد من الطبقات المتصلة بالكامل فوق بعضها البعض. تتغذى كل طبقة في الطبقة التي فوقها، حتى نتاج النواج. يمكننا أن نفك في الطبقات الأولى $L - 1$ على أنها تمثيلنا والطبقة الأخيرة على أنها متتبع خطى. يُطلق على هذه البنية عادةً اسم بيرسيبترون متعدد الطبقات Multilayer perceptron، وغالباً ما يتم اختصاره باسم MLP (الشكل 5.1.1).



الشكل 5.1.1 MLP مع طبقة مخفية من 5 وحدات مخفية.

يحتوي MLP هذا على 4 مدخلات و3 مخرجات، وتحتوي طبقة المخفية على 5 وحدات مخفية. نظراً لأن طبقة الإدخال input layer لا تتضمن أي حسابات، فإن إنتاج مخرجات بهذه الشبكة يتطلب تنفيذ الحسابات لكل من الطبقات المخفية والمخرجات؛ وبالتالي، فإن عدد

الطبقات في MLP هذا هو 2. لاحظ أن كلا الطبقتين متصلتان بالكامل fully connected. يؤثر كل إدخال على كل خلية عصبية في الطبقة المخفية hidden layer، ويؤثر كل منها بدوره على كل خلية عصبية في طبقة الإخراج output layer. للأسف، لم ننتهي بعد.

5.1.1.3 من الخطى إلى غير الخطى From Linear to Nonlinear

كما كان من قبل، نشير بواسطة المصفوفة $\mathbf{X} \in \mathbb{R}^{n \times d}$ إلى دفعه صغيرة من n من الأمثلة حيث يحتوي كل مثال على d مدخلات (ميزات). بالنسبة إلى MLP ذات الطبقة المخفية الواحدة والتي تحتوي طبقتها المخفية على h وحدات مخفية hidden units، فإننا نشير إلى مخرجات الطبقة المخفية $\mathbf{H} \in \mathbb{R}^{n \times h}$ ، والتي تمثل تمثيلات مخفية hidden representations لأن كلا من الطبقات المخفية والمخرجة متصلتان تماماً fully connected، فلدينا أوزان وتحيزات طبقة مخفية $\mathbf{W}^{(1)} \in \mathbb{R}^{d \times h}$ و $\mathbf{b}^{(1)} \in \mathbb{R}^{1 \times h}$ على التوالي وأوزان وتحيزات طبقة المخرجات $\mathbf{W}^{(2)} \in \mathbb{R}^{h \times q}$ و $\mathbf{b}^{(2)} \in \mathbb{R}^{1 \times q}$ على التوالي. هذا يسمح لنا بحساب مخرجات MLP ذات الطبقة الواحدة المخفية على النحو التالي:

$$\begin{aligned}\mathbf{H} &= \mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)}, \\ \mathbf{O} &= \mathbf{H}\mathbf{W}^{(2)} + \mathbf{b}^{(2)}.\end{aligned}$$

لاحظ أنه بعد إضافة الطبقة المخفية، يتطلب نموذجنا الآن تتبع مجموعات إضافية من المعلمات وتحديتها. إذن ما الذي ربحنا في المقابل؟ قد تتفاجأ عندما تكتشف - في النموذج المحدد أعلاه - أننا لا نكسب شيئاً مقابل مشاكلنا! السبب واضح. يتم إعطاء الوحدات المخفية أعلاه بواسطة دالة أفينية للمدخلات، والمخرجات (pre-softmax) هي مجرد دالة أفينية للوحدات المخفية. علاوة على ذلك، كان نموذجنا الخطي قادرًا بالفعل على تمثيل أي دالة أفيني.

لرؤية هذا بشكل رسمي، يمكننا فقط طي الطبقة المخفية في التعريف أعلاه، مما ينتج عنه نموذج طبقة واحدة مكافئ مع معلمات $\mathbf{b} = \mathbf{b}^{(1)}\mathbf{W}^{(2)} + \mathbf{b}^{(2)}$ و $\mathbf{W} = \mathbf{W}^{(1)}\mathbf{W}^{(2)}$.

$$\mathbf{O} = (\mathbf{X}\mathbf{W}^{(1)} + \mathbf{b}^{(1)})\mathbf{W}^{(2)} + \mathbf{b}^{(2)} = \mathbf{X}\mathbf{W}^{(1)}\mathbf{W}^{(2)} + \mathbf{b}^{(1)}\mathbf{W}^{(2)} + \mathbf{b}^{(2)} = \mathbf{X}\mathbf{W} + \mathbf{b}.$$

من أجل إدراك إمكانات البني متعددة الطبقات، نحتاج إلى عنصر رئيسي آخر: دالة تنشيط غير خطية σ (nonlinear activation function) يتم تطبيقها على كل وحدة مخفية بعد التحويل affine transformation. على سبيل المثال، الخيار الشائع هو دالة التنشيطReLU الأفيني (Rectified Linear Unit) $\sigma(x) = \max(0, x)$ (Nair and Hinton, 2010) تعمل على عناصرها من حيث العناصر element-wise. تسمى مخرجات دوال التنشيط activations عمليات التنشيط activations. بشكل عام، مع وجود دوال التنشيط في مكانها الصحيح، لم يعد من الممكن طي MLP الخاص بنا في نموذج خطي:

$$\begin{aligned}\mathbf{H} &= \sigma(\mathbf{XW}^{(1)} + \mathbf{b}^{(1)}), \\ \mathbf{O} &= \mathbf{HW}^{(2)} + \mathbf{b}^{(2)}.\end{aligned}$$

نظرًا لأن كل صفي في \mathbf{X} يتافق مع مثال في الدفعات الصغيرة minibatch، مع بعض إساعة استخدام الترميز، فإننا نحدد اللاحظية σ لتطبيقها على مدخلاته بطريقة row-wise، أي مثال row-wise واحد في كل مرة. لاحظ أننا استخدمنا نفس الترميز لـ softmax عندما أشرنا إلى القسم 4.1.3. في كثير من الأحيان، لا تتطابق دوال التشبيط التي نستخدمها فقط على الصيوف ولكن من حيث العناصر. هذا يعني أنه بعد حساب الجزء الخطبي من الطبقة، يمكننا حساب كل تشبيط دون النظر إلى القيم التي اتخذتها الوحدات المخفية الأخرى.

لبناء MLPs أكثر عمومية، يمكننا الاستمرار في تكديس هذه الطبقات المخفية، على سبيل المثال $(\mathbf{b}^{(1)} + \mathbf{b}^{(1)}\mathbf{W}^{(1)} + \mathbf{b}^{(2)} = \sigma_1(\mathbf{H}^{(1)}\mathbf{W}^{(2)} + \mathbf{b}^{(2)}) = \sigma_2(\mathbf{H}^{(1)}\mathbf{W}^{(2)} + \mathbf{b}^{(2)})$ إداتها فوق الأخرى، مما يؤدي إلى إنتاج المزيد من النماذج التعبيرية.

5.1.1.4. المقربين العالميين Universal Approximators

نحن نعلم أن الدماغ قادر على تحليل إحصائي متتطور للغاية. على هذا النحو، يجدر بنا أن نسأل، إلى أي مدى يمكن أن تكون قوة الشبكة العميقية. تمت الإجابة على هذا السؤال عدة مرات، على سبيل المثال، في Cybenko (1989) في سياق MLPs، وفي Micchelli (1984) في سياق إعادة إنتاج مساحات kernel Hilbert بطريقة يمكن اعتبارها شبكات دالة أساس شعاعي (RBF) بطبقة واحدة مخفية. تشير هذه (والنتائج ذات الصلة) إلى أنه حتى مع وجود شبكة ذات طبقة واحدة مخفية، مع توفير عدد كافٍ من العقد (ربما بشكل سخيف)، ومجموعة الأوزان الصحيحة، يمكننا نمذجة أي دالة في الواقع، تعلم هذه الدالة هو الجزء الصعب. قد تعتقد أن شبكتك العصبية تشبه إلى حد ما لغة البرمجة سي. اللغة، مثل أي لغة حديثة أخرى، قادرة على التعبير عن أي برنامج محosب. لكن في الواقع، فإن الخروج ببرنامج يلبي المواصفات الخاصة بك هو الجزء الصعب.

علاوة على ذلك، لمجرد أن شبكة الطبقة الواحدة المخفية يمكن أن تتعلم أي دالة لا يعني أنه يجب عليك محاولة حل جميع مشاكلك مع شبكات الطبقة المفردة المخفية. في الواقع، تعتبر طرق النواة kernel methods في هذه الحالة أكثر فاعلية، لأنها قادرة على حل المشكلة تماماً حتى في المساحات ذات الأبعاد اللانهائية (Schölkopf, Kimeldorf and Wahba, 1971). في الواقع، يمكننا تقريب العديد من الدوال بشكل أكثر إحكاماً باستخدام شبكات أعمق (مقابل شبكات أوسع) (Simonyan and Zisserman, 2014). سنتطرق إلى حجاج أكثر صرامة في الفصول اللاحقة.

5.1.2 دوال التنشيط Activation Functions

تحدد دوال التنشيط Activation functions ما إذا كان يجب تنشيط الخلية العصبية أم لا عن طريق حساب مجموع الأوزان وإضافة المزيد من التحيز معه. هم مشغلون للفاضل لتحويل إشارات الإدخال إلى مخرجات، بينما يضيف معظمهم اللاخطية. نظرًا لأن دوال التنشيط أساسية للتعلم العميق، فلنستعرض بعض دوال التنشيط الشائعة.

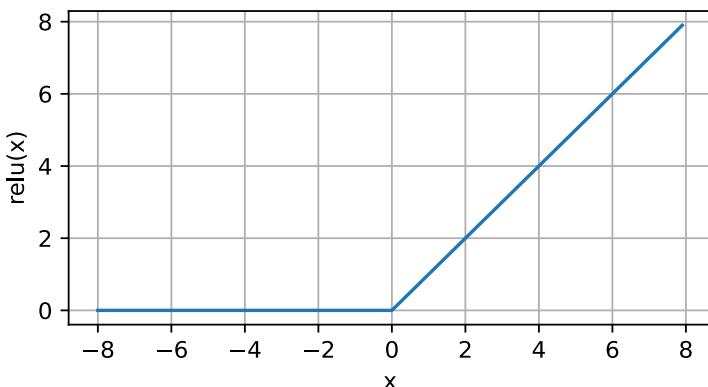
```
%matplotlib inline
import tensorflow as tf
from d2l import tensorflow as d2l
```

5.1.2.1 دالة ReLU

ال الخيار الأكثر شيوعًا، نظرًا لبساطة التنفيذ وأدائه الجيد في مجموعة متنوعة من المهام التنبؤية، هو الوحدة الخطية المصححة (ReLU)، (ReLU، 2010). يوفر ReLU تحويل غير خطية بسيطة للغاية. بالنظر لعنصر ما x ، يتم تعريف الدالة على أنها الحد الأقصى لهذا العنصر و 0:

$$\text{ReLU}(x) = \max(x, 0).$$

بشكل غير رسمي، تحتفظ دالة ReLU بالعناصر الإيجابية فقط وتتجاهل جميع العناصر السلبية عن طريق ضبط التنشيطات المقابلة على 0. لاكتساب بعض الحدس، يمكننا رسم الدالة. كما ترى، فإن دالة التنشيط خطية متعددة التعريف activation function is piecewise linear



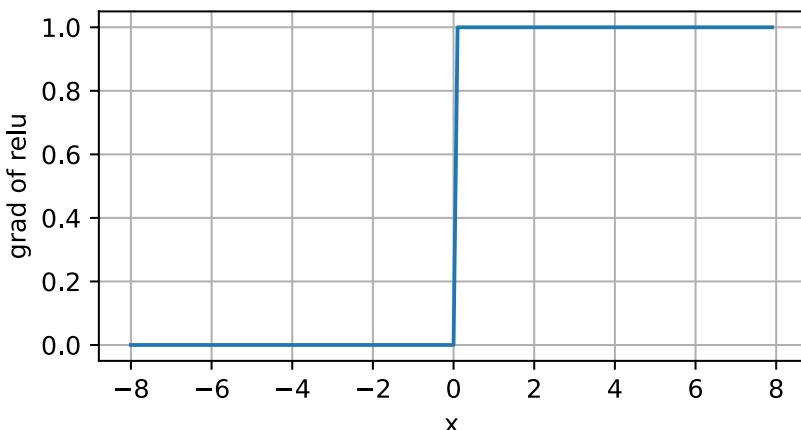
عندما يكون الإدخال سالبًا، يكون مشتق الدالة ReLU هو 0، وعندما يكون الإدخال موجبًا، يكون مشتق دالة ReLU هو 1. لاحظ أن دالة ReLU غير قابلة للاشتراق not differentiable عندما يأخذ الإدخال قيمة تساوي بالضبط 0. في هذه الحالات، نستخدم مشتق الجانب الأيسر افتراضيًّا ونقول إن المشتق يساوي 0 عندما يكون الإدخال 0. يمكننا التخلص من هذا لأن المدخلات قد لا تكون أبدًا صفرًا (قد يقول علماء الرياضيات أنه لا يمكن تمييزها

في مجموعة من قياس الصفر). هناك قول مأثور قديم مفاده أنه إذا كانت الظروف الحدودية الدقيقة مهمة، فربما تقوم بعمل رياضيات (حقيقية)، وليس هندسة.

"There is an old adage that if subtle boundary conditions matter, we are probably doing (*real*) mathematics, not engineering"

قد تنطبق هذه الحكمة التقليدية هنا، أو على الأقل، حقيقة أننا لا نقوم بإجراء تحسين مقيد (Rockafellar ، 1970 ، 1965 ، Mangasarian أدناه. نرسم مشتق دالة ReLU الموضحة

```
with tf.GradientTape() as t:
    y = tf.nn.relu(x)
d2l.plot(x.numpy(), t.gradient(y, x).numpy(), 'x', 'grad of relu',
          figsize=(5, 2.5))
```



سبب استخدام ReLU هو أن مشتقاته حسنة التصرف بشكل خاص: إما أنها تتلاشى أو تترك المدخل تمر. هذا يجعل التحسين يتصرف بشكل أفضل ويخفف من المشكلة الموثقة جيداً المتمثلة في اختفاء التدرجات vanishing gradients التي ابنتها الإصدارات السابقة من الشبكات العصبية (المزيد حول هذا لاحقاً).

لاحظ أن هناك العديد من المتغيرات لدالة ReLU، بما في ذلك دالة ReLU ذات المعلمات (parameterized ReLU) (He et al. 2015). يضيف هذا الاختلاف مصطلحاً خطياً إلى ReLU، لذلك لا تزال بعض المعلومات تصل، حتى عندما تكون الوسيطة سلبية:

$$\text{pReLU}(x) = \max(0, x) + \alpha \min(0, x).$$

Sigmoid دالة 5.1.2.2

تعمل دالة Sigmoid على تحويل مدخلاتها، والتي تكمن القيم في المجال، إلى مخرجات تقع على الفاصل الزمني (0، 1). لهذا السبب، غالباً ما يطلق على squashing دالة سحق Sigmoid function: إنه يسحق أي إدخال في النطاق (-inf, inf) إلى بعض القيمة في النطاق (0 ، 1):

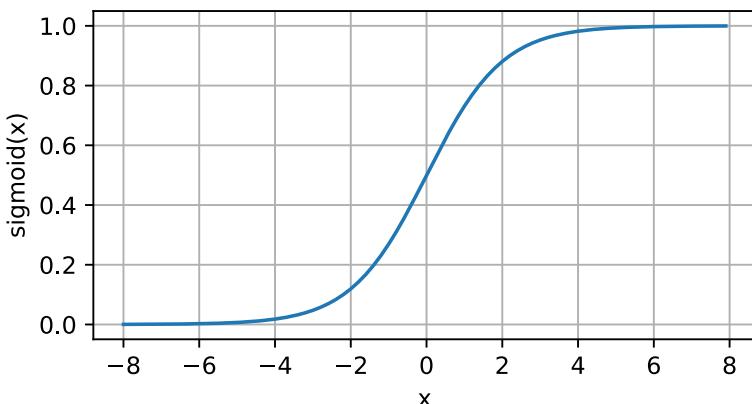
$$\text{sigmoid}(x) = \frac{1}{1 + \exp(-x)}.$$

في أقدم الشبكات العصبية، كان العلماء مهتمين بنمذجة الخلايا العصبية البيولوجية التي إما تطلق fire أو لا تطلق not fire. وهكذا ركز رواد هذا المجال، بالعودة إلى McCulloch و Pitts، على وحدات العتبة thresholding units (McCulloch and Pitts، 1943). يأخذ تنشيط العتبة القيمة 0 عندما يكون إدخاله أقل من بعض العتبة والقيمة 1 عندما يتجاوز الإدخال الحد.

عندما تحول الانتباه إلى التعلم القائم على التدرج gradient based learning، كانت دالة Sigmoid اختياراً طبيعياً لأنها تقريب سلس وقابل للتفاضل لوحدة العتبة. لا تزال Sigmoid تستخدم على نطاق واسع كدوال تشيشطي وحدات الإخراج، عندما نريد تفسير المخرجات على أنها احتمالات لمشاكل التصنيف الثنائي: يمكنك التفكير في Sigmoid كحالة خاصة من softmax. ومع ذلك، تم استبدال Sigmoid في الغالب بReLU الأبسط والأكثر سهولة في التدريب لمعظم الاستخدامات في الطبقات المخفية. يتعلق الكثير من هذا بحقيقة أن Sigmoid يفرض تحديات على التحسين (LeCun et al.، 1998) نظراً لأن تدرجها يتلاشى بسبب المدخلات الإيجابية والسلبية الكبيرة. هذا يمكن أن يؤدي إلى الهضاب plateaus التي يصعب الهروب منها. ومع ذلك، فإن Sigmoid مهمة في الفصول اللاحقة (على سبيل المثال، القسم 10.1 حول الشبكات العصبية المتكررة، سنصف البنى التي تستفيد من وحدات Sigmoid للتحكم في تدفق المعلومات عبر الوقت).

أدنى، نرسم دالة Sigmoid. لاحظ أنه عندما يكون الإدخال قريباً من 0، تقترب دالة Sigmoid من التحويل الخططي linear transformation.

```
y = tf.nn.sigmoid(x)
d2l.plot(x.numpy(), y.numpy(), 'x', 'sigmoid(x)',  
figsize=(5, 2.5))
```

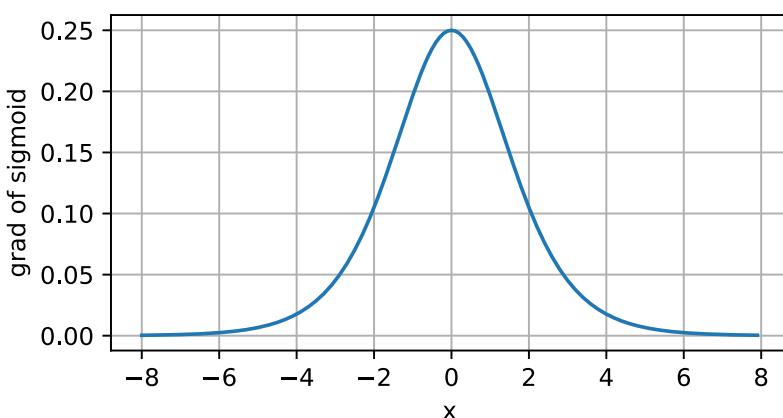


يتم الحصول على مشتق دالة Sigmoid بالمعادلة التالية:

$$\frac{d}{dx} \text{sigmoid}(x) = \frac{\exp(-x)}{(1 + \exp(-x))^2} = \text{sigmoid}(x)(1 - \text{sigmoid}(x)).$$

تم رسم مشتق دالة Sigmoid أدناه. لاحظ أنه عندما يكون الإدخال 0، فإن مشتق دالة Sigmoid يصل إلى 0.25 كحد أقصى. عندما ينحرف المدخل عن 0 في أي اتجاه، يقترب المشتق من 0.

```
with tf.GradientTape() as t:
    y = tf.nn.sigmoid(x)
d2l.plot(x.numpy(), t.gradient(y, x).numpy(), 'x', 'grad
of sigmoid',
          figsize=(5, 2.5))
```



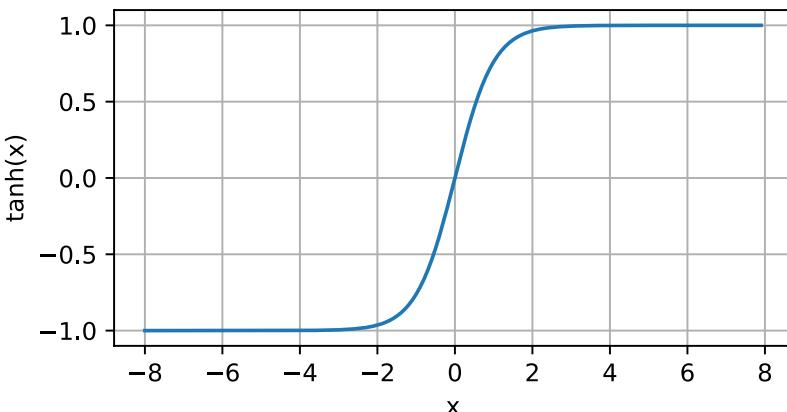
Tanh . 5.1.2.3

مثل الدالة sigmoid، تقوم دالة \tanh (الظل الزائد hyperbolic tangent) أيضاً بسحق مدخلاتها، وتحوילها إلى عناصر في الفترة بين -1 و 1 :

$$\tanh(x) = \frac{1 - \exp(-2x)}{1 + \exp(-2x)}.$$

نرسم دالة \tanh أدناه. لاحظ أنه عندما يقترب الإدخال من الصفر، تقترب الدالة \tanh من التحويل الخطى. على الرغم من أن شكل الدالة مشابه لشكل دالة Sigmoid، إلا أن دالة \tanh ظهرت تنازلاً نقطياً حول أصل نظام الإحداثيات (1992، Kalman and Kwasny).

```
y = tf.nn.tanh(x)
d2l.plot(x.numpy(), y.numpy(), 'x', 'tanh(x)',
figsize=(5, 2.5))
```

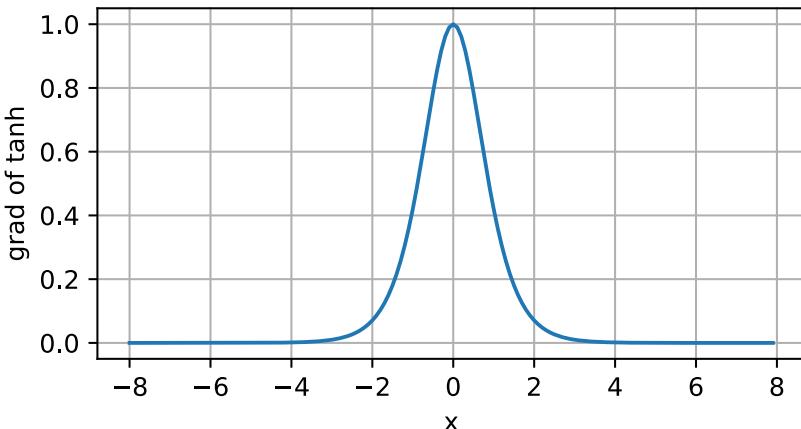


مشتق التابع \tanh هو:

$$\frac{d}{dx} \tanh(x) = 1 - \tanh^2(x).$$

تم رسمه أدناه. عندما يقترب المدخل من الصفر، يقترب مشتق الدالة \tanh من 1 كحد أقصى. وكما رأينا مع دالة Sigmoid، حيث يتحرك الإدخال بعيداً عن الصفر في أي من الاتجاهين، يقترب مشتق الدالة \tanh من الصفر.

```
with tf.GradientTape() as t:
    y = tf.nn.tanh(x)
d2l.plot(x.numpy(), t.gradient(y, x).numpy(), 'x', 'grad
of tanh',
        figsize=(5, 2.5))
```



5.1.3 الملخص

نحن نعرف الآن كيفية دمج اللاحظية لبناء بنية شبكة عصبية معبرة متعددة الطبقات. كملاحظة جانبية، فإن معرفتك تضلع بالفعل في قيادة مجموعة أدوات مماثلة للممارس حوالي عام 1990. في بعض النواحي، لديك ميزة على أي شخص يعمل في التسعينيات، لأنك يمكنك الاستفادة من إطار التعلم العميق القوية مفتوحة المصدر لبناء النماذج بسرعة، باستخدام بضعة أسطر فقط من التعليمات البرمجية. في السابق، كان تدريب هذه الشبكات يتطلب من الباحثين ترميز الطبقات والمشتقات بشكل صريح في C أو Fortran أو حتى Lisp (في حالةReLU).
فائدة ثانوية هي أنReLU أكثر قابلية للتحسين بشكل ملحوظ منSigmoid أو دالةtanh. يمكن للمرء أن يجادل في أن هذا كان أحد الابتكارات الرئيسية التي ساعدت على عودة التعلم العميق خلال العقد الماضي. لاحظ، مع ذلك، أن البحث في دوال التنشيط لم يتوقف. على سبيل المثال، يمكن أن تؤدي دالة تنشيطSwish $\sigma(x) = x \text{sigmoid}(\beta x)$ كما هو مقترن إلى دقة أفضل في كثير من الحالات. (Ramachandran et al., 2017).

5.1.4 التمارين

1. أظهر أن إضافة طبقات إلى شبكة عميق خطية linear deep network، أي شبكة بدون اللاحظية لا يمكنها أبداً زيادة القوة التعبيرية للشبكة. أعط مثالاً حيث يقلل ذلك بنشاط.
2. احسب مشتق من دالة التنشيطpReLU.
3. احسب مشتق من دالة التنشيط $x \text{sigmoid}(\beta x)$ Swish
4. أظهر أن MLP باستخدام ReLU فقط (أو pReLU) يبني دالة خطية مستمرة متعددة التعريفات continuous piecewise linear function.
5. و \tanh و Sigmoid متشابهان جداً.

1. اظهر $\tanh(x) + 1 = 2\text{sigmoid}(2x)$
2. إثبت أن فئات الدوال المحددة بواسطة كلا اللاخطيين متطابقة. تلميح: الطبقات الافقية affine layers لها مصطلحات متحيزة أيضاً.
6. افترض أن لدينا خاصية غير خطية تنطبق على الدفعات الصغيرة واحد في كل مرة، مثل تسوية الدفعات batch normalization (Ioffe and Szegedy, 2015). ما أنواع المشاكل التي تتوقع أن يسببها هذا؟
7. قدم مثالاً حيث تختلف التدرجات لدالة التنشيط Sigmoid.

5.2 تنفيذ البيرسيبترون متعدد الطبقات Multilayer Perceptron

البيرسيبترون متعدد الطبقات (MLPs) ليس أكثر تعقيداً في التنفيذ من النماذج الخطية البسيطة. يتمثل الاختلاف المفاهيمي الرئيسي في أنها نجمت طبقات متعددة الآن.

```
import tensorflow as tf
from d2l import tensorflow as d2l
```

5.2.1 التنفيذ من البداية Implementation from Scratch

لنبدأ مرة أخرى بتنفيذ مثل هذه الشبكة من البداية.

5.2.1.1 تهيئة معلمات النموذج Initializing Model Parameters

تذكر أن Fashion-MNIST يحتوي على 10 فئات، وأن كل صورة تتكون من شبكة $28 \times 28 = 784$ من قيم البكسل الرمادية. كما كان من قبل، سوف نتجاهل البنية المكانية بين وحدات البكسل في الوقت الحالي، لذلك يمكننا التفكير في هذا على أنه مجموعة بيانات تصنيف تحتوي على 784 ميزة إدخال و10 فئات. للبدء، سنقوم بتطبيق MLP بطبقة مخفية واحدة و256 وحدة مخفية. يمكن ضبط كل من عدد الطبقات وعرضها (تعتبر معلمات فائقة). عادة، نختار عرض الطبقة لتكون قابلة للقسمة على قوى أكبر من 2. وهذا فعال من الناحية الحسابية نظراً للطريقة التي يتم بها تخصيص الذاكرة ومعالجهافي الأجهزة.

مرة أخرى، سنقوم بتمثيل معلماتنا بعدة موترات tensors. لاحظ أنه بالنسبة لكل طبقة، يجب أن نتبع مصفوفة وزن weight matrix واحدة ومتوجه تحيز bias vector واحد. كما هو الحال دائمًا، نخصص ذاكرة لدرجات الخطأ gradients of the loss فيما يتعلق بهذه المعلمات.

```
class MLPScratch(d2l.Classifier):
    def __init__(self, num_inputs, num_outputs, num_hiddens,
                 lr, sigma=0.01):
        super().__init__()
        self.save_hyperparameters()
        self.W1 = tf.Variable(
            tf.random.normal((num_inputs, num_hiddens)) *
            sigma)
```

```

self.b1 = tf.Variable(tf.zeros(num_hiddens))
self.W2 = tf.Variable(
    tf.random.normal((num_hiddens, num_outputs)) *
sigma)
self.b2 = tf.Variable(tf.zeros(num_outputs))

```

Model 5.2.1.2

للتأكد من أننا نعرف كيف يعمل كل شيء، سنقوم بتنفيذ تنشيط ReLU بأنفسنا بدلاً من استدعاء دالة relu المدمجة مباشرةً.

```

def relu(X):
    return tf.math.maximum(X, 0)

```

نظرًا لأننا نتجاهل البنية المكانية، فنحن نعيد تشكيل كل صورة ثنائية الأبعاد إلى متوجه مسطح بطول عدد المدخلات num_inputs. أخيرًا، نطبق نموذجنا ببعضة أسطر من التعليمات البرمجية. نظرًا لأننا نستخدم إطار عمل autograd المدمج، فهذا كل ما يتطلب الأمر.

```

@d2l.add_to_class(MLPScratch)
def forward(self, X):
    X = tf.reshape(X, (-1, self.num_inputs))
    H = relu(tf.matmul(X, self.W1) + self.b1)
    return tf.matmul(H, self.W2) + self.b2

```

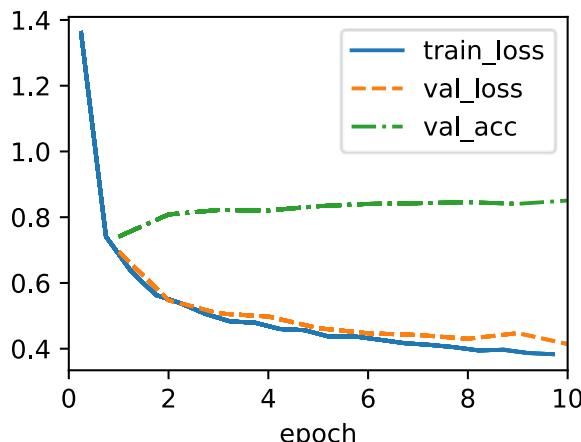
Training 5.2.1.3

لحسن الحظ، فإن حلقة التدريب لـ MLPs هي نفسها تماماً لـ انحدار softmax. نحدد النموذج والبيانات والمدرب وأخيراً نستدعي الدالة fit في النموذج والبيانات.

```

model = MLPScratch(num_inputs=784, num_outputs=10,
                    num_hiddens=256, lr=0.1)
data = d2l.FashionMNIST(batch_size=256)
trainer = d2l.Trainer(max_epochs=10)
trainer.fit(model, data)

```



5.2.2. Concise Implementation

كما قد توقع، من خلال الاعتماد على واجهات برمجة التطبيقات APIs عالية المستوى، يمكننا تنفيذ MLPs بشكل أكثر دقة.

5.2.2.1. النموذج Model

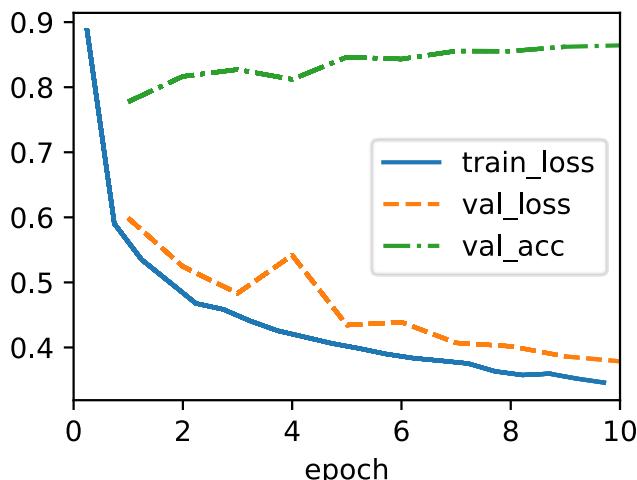
بالمقارنة مع تفاصيلنا المختصرة لتطبيق انحدار softmax (القسم 4.5)، فإن الاختلاف الوحيد هو أننا نضيف طبقتين متصلتين تماماً fully connected حيث أضفنا سابقاً واحدة فقط. الأولى هي الطبقة المخفية hidden layer، والثانية هي الطبقة الناتجة output layer.

```
class MLP(d2l.Classifier):
    def __init__(self, num_outputs, num_hiddens, lr):
        super().__init__()
        self.save_hyperparameters()
        self.net = tf.keras.models.Sequential([
            tf.keras.layers.Flatten(),
            tf.keras.layers.Dense(num_hiddens,
activation='relu'),
            tf.keras.layers.Dense(num_outputs)])
```

5.2.2.2. التدريب Training

حلقة التدريب هي نفسها تماماً عندما طبقنا انحدار softmax. يمكننا هذه النقطة من فصل الأمور المتعلقة بهندسة النموذج عن الاعتبارات المعمادة.

```
model = MLP(num_outputs=10, num_hiddens=256, lr=0.1)
trainer.fit(model, data)
```



5.2.3 الملخص

الآن بعد أن أصبح لدينا مزيد من الممارسة في تصميم الشبكات العميقة، فإن الخطوة من طبقة واحدة إلى طبقات متعددة من الشبكات العميقة لم تعد تشكل تحدياً كبيراً بعد الآن. على وجه الخصوص، يمكننا إعادة استخدام خوارزمية التدريب ومحمل البيانات. لاحظ، على الرغم من ذلك، أن تنفيذ MLPs من الصفر مع ذلك فوضوي: تسمية معلمات النموذج وتبعها تجعل من الصعب توسيع النماذج. على سبيل المثال، تخيل أنك ترغب في إدراج طبقة أخرى بين الطبقتين 42 و 43. قد تكون هذه الآن طبقة 42b، إلا إذا كانا على استعداد لإجراء إعادة تسمية متسلسلة. علاوة على ذلك، إذا قمنا بتنفيذ الشبكة من البداية، فسيكون من الصعب جداً على إطار العمل إجراء تحسينات مفيدة في الأداء.

مع ذلك ، فقد وصلت الآن إلى أحد ما توصلت إليه التكنولوجيا في أواخر الثمانينيات عندما كانت الشبكات العميقة المتصلة بالكامل هي الطريقة المفضلة لنمذجة الشبكة العصبية. ستكون خطوتنا المفاهيمية التالية هي النظر في الصور. قبل القيام بذلك، نحتاج إلى مراجعة عدد من الأساسيات الإحصائية والتفاصيل حول كيفية حساب النماذج بكفاءة.

5.2.4 التمارين

1. قم بتغيير عدد الوحدات المخفية ورسم كيف يؤثر عدها على دقة النموذج. ما هي أفضل قيمة لهذا المعامل الفائق؟

2. حاول إضافة طبقة مخفية لترى كيف تؤثر على النتائج.

3. لماذا يعتبر إدخال طبقة مخفية ذات خلية عصبية واحدة فكرة سيئة؟ ما الخطأ الذي يمكن أن يحدث؟

4. كيف تغير معدل التعلم يغير نتائجك؟ مع إصلاح جميع المعلمات الأخرى، ما هو معدل التعلم الذي يمكنك أفضل النتائج؟ كيف يرتبط هذا بعدد الفترات؟

5. دعنا نقوم بالتحسين عبر جميع المعلمات الفائقة معًا، أي معدل التعلم وعدد الفترات وعدد الطبقات المخفية وعدد الوحدات المخفية لكل طبقة.

1. ما هي أفضل نتيجة يمكنك الحصول عليها من خلال تحسينها جمیعاً؟

2. لماذا يعتبر التعامل مع العديد من المعلمات الفائقة أكثر صعوبة؟

3. صيغ إستراتيجية فعالة للتحسين عبر معايير متعددة بشكل مشترك.

6. قارن بين سرعة إطار العمل والتنفيذ من الصفر لمشكلة صعبة. كيف تتغير مع تعقيد الشبكة؟

7. قم بقياس سرعة عمليات ضرب مصفوفة الموتر للحصول على مصفوفات جيدة المحاذاة well-aligned وغير المحاذاة misaligned. على سبيل المثال، اختر

المصفوفات ذات الأبعاد 1024x1025 و 1026x1028 و 1029x1032.

1. كيف يتغير هذا بين وحدات معالجة الرسومات GPUs ووحدات المعالجة المركزية CPUs؟

2. حدد عرض ناقل الذاكرة memory bus width لوحدة المعالجة المركزية CPU ووحدة معالجة الرسومات GPU.
8. جرب دوال التنشيط المختلفة. أيهما أفضل؟
9. هل هناك فرق بين التهيئة للوزن للشبكة؟ هل يهم؟

5.3 الانتشار الأمامي Forward Propagation، والانتشار الخلفي Backward Propagation الحسابية والرسوم البيانية Computational Graphs

حتى الآن، قمنا بتدريب نماذجنا باستخدام التدرج الاشتقافي العشوائي المصغر minibatch SGD. ومع ذلك، عندما قمنا بتطبيق الخوارزمية، كنا قلقين فقط بشأن الحسابات التي ينطوي عليها الانتشار الأمامي Forward Propagation من خلال النموذج. عندما حان الوقت لحساب التدرجات، استدعينا للتو دالة الانتشار الخلفي backpropagation التي يوفرها إطار عمل التعلم العميق.

يعمل الحساب التلقائي للتدرجات (التمايز التلقائي automatic differentiation) على تبسيط تطبيق خوارزميات التعلم العميق بشكل كبير. قبل التفاضل التلقائي، كانت التغييرات الصغيرة على النماذج المعقدة تتطلب إعادة حساب المشتقات المعقدة يدوياً. في كثير من الأحيان، كان من المثير للدهشة أن الأوراق الأكاديمية كان عليها تحصيص العديد من الصفحات لاشتقاق قواعد التحديث. بينما يجب أن نستمر في الاعتماد على التفاضل التلقائي حتى نتمكن من التركيز على الأجزاء المثيرة للاهتمام، يجب أن تعرف كيف يتم حساب هذه التدرجات gradients إذا كنت تريده تجاوز الفهم الضحل للتعلم العميق.

في هذا القسم، نلقي نظرة عميقة على تفاصيل الانتشار الخلفي backward propagation (يُطلق عليه أكثر شيوعاً backpropagation). لنقل بعض البصيرة لكل من التقنيات وتطبيقاتها، نعتمد على بعض الرياضيات الأساسية والرسوم البيانية الحسابية computational graphs. للبدء، نركز عرضاً على طبقة MLP ذات طبقة واحدة مخفية مع تناقص الوزن weight decay (التنظيم ℓ_2 ، على أن يتم وصفه في الفصول اللاحقة).

5.3.1 Forward Propagation

يشير الانتشار الأمامي Forward Propagation (أو التمرين الأمامي) إلى حساب وتخزين المتغيرات الوسيطة (بما في ذلك المخرجات) للشبكة العصبية بالترتيب من طبقة الإدخال إلى طبقة الإخراج. نحن نعمل الآن خطوة بخطوة من خلال آليات الشبكة العصبية بطبقة مخفية واحدة. قد يبدو هذا مملاً، لكن بالكلمات الأبدية للمبدع الفنان جيمس براون، يجب أن "تدفع الشمن لتكون الرئيس".

من أجل البساطة، دعنا نفترض أن مثال الإدخال هو $x \in \mathbb{R}^d$ وأن الطبقة المخفية لا تتضمن مصطلح التحيز. هنا المتغير الوسيط هو:

$$z = W^{(1)}x,$$

حيث $\mathbf{z} \in \mathbb{R}^h$ هي معلمة وزن الطبقة المخفية. بعد تشغيل المتغير الوسيط $\mathbf{W}^{(1)} \in \mathbb{R}^{h \times d}$ من خلال دالة التنشيط ϕ , نحصل على متوجه التنشيط المخفى للطول h ,

$$\mathbf{h} = \phi(\mathbf{z}).$$

ناتج الطبقة المخفية \mathbf{h} هو أيضاً متغير وسيط. بافتراض أن معلمات الطبقة المخرجة تمتلك وزناً فقط $\mathbf{W}^{(2)} \in \mathbb{R}^{q \times h}$, فيمكننا الحصول على متغير طبقة الإخراج مع متوجه الطول q :

$$\mathbf{o} = \mathbf{W}^{(2)}\mathbf{h}.$$

بافتراض أن دالة الخطأ l ومثال التسمية y , يمكننا بعد ذلك حساب مصطلح الخطأ لمثال بيانات واحد,

$$L = l(\mathbf{o}, y).$$

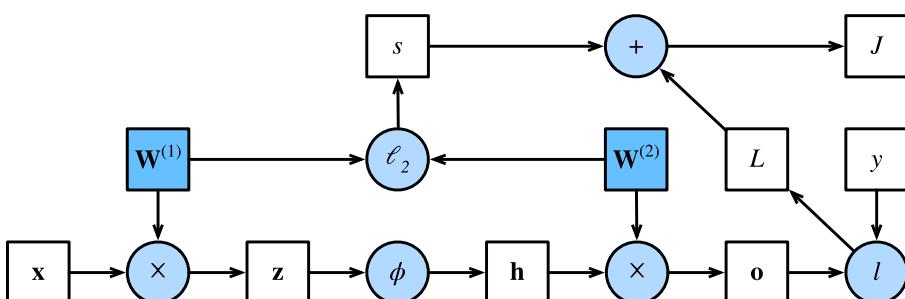
وفقاً لتعريف التنظيم ℓ_2 الذي سنقدمه لاحقاً، نظراً للمعلمة الفائقة، فإن مصطلح التنظيم هو

$$J = L + s.$$

نشير إلى J كدالة الهدف objective function في المناقشة التالية.

5.3.2 الرسم البياني الحسابي للانتشار الأمامي Computational Graph of Forward Propagation

يساعدنا رسم الرسوم البيانية الحسابية computational graphs على تصور تبعيات المشغلين والمتغيرات في الحساب. يحتوي الشكل 5.3.1 على الرسم البياني المرتبط بالشبكة البسيطة الموضحة أعلاه، حيث تشير المربعات إلى المتغيرات variables وتشير الدوائر إلى العوامل operators. الزاوية اليسرى السفلية تشير إلى الإدخال والزاوية اليمنى العلوية هي الإخراج. لاحظ أن اتجاهات الأسماء (التي توضح تدفق البيانات data flow) هي في المقام الأول إلى اليمين والصعود.



الشكل 5.3.1 رسم بياني حسابي للانتشار الأمامي.

يشير الانشار الخلفي Backpropagation إلى طريقة حساب التدرج gradient لمعلمات الشبكة العصبية. باختصار، تعبّر الطريقة الشبكة بترتيب عكسي، من المخرجات إلى طبقة الإدخال، وفقاً لقاعدة السلسلة chain rule من حساب التفاضل والتكمال. تقوم الخوارزمية بتخزين أي متغيرات وسيطة (مشتقات جزئية) مطلوبة أثناء حساب التدرج فيما يتعلق ببعض المعلمات. افترض أن لدينا دوال $Z = g(Y)$ و $Y = f(X)$ ، أن المدخلات والمخرجات X, Y, Z عبارة عن موتر لأشكال عشوائية. باستخدام قاعدة السلسلة، يمكننا حساب مشتقة Z بالنسبة إلى X بواسطة:

$$\frac{\partial Z}{\partial X} = \text{prod}\left(\frac{\partial Z}{\partial Y}, \frac{\partial Y}{\partial X}\right).$$

هنا نستخدم عامل التشغيل prod لمضاعفة وسيطاته بعد تنفيذ العمليات الضرورية، مثل التحويل وتبدل مواضع الإدخال. بالنسبة إلى المتجهات، هذا واضح ومبادر: إنه ببساطة ضرب مصفوفة—مصفوفة. بالنسبة للموترات ذات الأبعاد الأعلى، نستخدم النظير counterpart المناسب. يخفى عامل التشغيل prod كل الرموز العلوية.

تذكر أن معلمات الشبكة البسيطة ذات الطبقة المخفية، والتي يوجد رسمها البياني الحسابي في الشكل 5.3.1، هي $\mathbf{W}^{(1)}$ و $\mathbf{W}^{(2)}$. الهدف من الانشار الخلفي backpropagation هو حساب التدرجات $\frac{\partial J}{\partial \mathbf{W}^{(1)}}$ و $\frac{\partial J}{\partial \mathbf{W}^{(2)}}$. لتحقيق ذلك، نطبق قاعدة السلسلة ونحسب، بدورنا، التدرج لكل متغير وسيط ومعلمة. يتم عكس ترتيب الحسابات بالنسبة إلى تلك التي يتم إجراؤها في الانشار الأمامي، نظراً لأننا نحتاج إلى البدء بنتيجة الرسم البياني الحسابي والعمل في طريقنا نحو المعلمات. الخطوة الأولى هي حساب تدرجات دالة الهدف $s = J + L$ فيما يتعلق بمصطلح الخطأ L ومصطلح التنظيم s .

$$\frac{\partial J}{\partial L} = 1 \text{ and } \frac{\partial J}{\partial s} = 1.$$

بعد ذلك، نحسب التدرج لدالة الهدف فيما يتعلق بمتغير طبقة المخرجات \mathbf{o} وفقاً لقاعدة السلسلة:

$$\frac{\partial J}{\partial \mathbf{o}} = \text{prod}\left(\frac{\partial J}{\partial L}, \frac{\partial L}{\partial \mathbf{o}}\right) = \frac{\partial L}{\partial \mathbf{o}} \in \mathbb{R}^q.$$

بعد ذلك، نحسب تدرجات مصطلح التنظيم فيما يتعلق بكل من المعلمتين:

$$\frac{\partial s}{\partial \mathbf{W}^{(1)}} = \lambda \mathbf{W}^{(1)} \text{ and } \frac{\partial s}{\partial \mathbf{W}^{(2)}} = \lambda \mathbf{W}^{(2)}.$$

الآن نحن قادرون على حساب التدرج $\frac{\partial J}{\partial \mathbf{W}^{(2)}} \in \mathbb{R}^{q \times h}$ لمعلمات النموذج الأقرب إلى طبقة الإخراج. ينتج عن استخدام قاعدة السلسلة:

$$\frac{\partial J}{\partial \mathbf{W}^{(2)}} = \text{prod}\left(\frac{\partial J}{\partial \mathbf{o}}, \frac{\partial \mathbf{o}}{\partial \mathbf{W}^{(2)}}\right) + \text{prod}\left(\frac{\partial J}{\partial s}, \frac{\partial s}{\partial \mathbf{W}^{(2)}}\right) = \frac{\partial J}{\partial \mathbf{o}} \mathbf{h}^T + \lambda \mathbf{W}^{(2)}.$$

للحصول على التدرج فيما يتعلق $\mathbf{W}^{(1)}$ ، نحتاج إلى مواصلة backpropagation على طول طبقة الإخراج إلى الطبقة المخفية. يتم إعطاء التدرج فيما يتعلق بإخراج الطبقة المخفية بواسطة $\frac{\partial J}{\partial \mathbf{h}} \in \mathbb{R}^h$

$$\frac{\partial J}{\partial \mathbf{h}} = \text{prod}\left(\frac{\partial J}{\partial \mathbf{o}}, \frac{\partial \mathbf{o}}{\partial \mathbf{h}}\right) = \mathbf{W}^{(2)\top} \frac{\partial J}{\partial \mathbf{o}}.$$

نظرًا لأن دالة التنشيط ϕ تتطابق على العناصر، فإن حساب التدرج $\frac{\partial J}{\partial \mathbf{z}} \in \mathbb{R}^h$ للمتغير الوسيط \mathbf{z} يتطلب أن نستخدم عامل الضرب العنصري elementwise multiplication operator، والذي نشير إليه بـ \odot :

$$\frac{\partial J}{\partial \mathbf{z}} = \text{prod}\left(\frac{\partial J}{\partial \mathbf{h}}, \frac{\partial \mathbf{h}}{\partial \mathbf{z}}\right) = \frac{\partial J}{\partial \mathbf{h}} \odot \phi'(\mathbf{z}).$$

أخيرًا، يمكننا الحصول على التدرج $\frac{\partial J}{\partial \mathbf{W}^{(1)}} \in \mathbb{R}^{h \times d}$ لمعلمات النموذج الأقرب إلى طبقة الإدخال. وفقًا لقاعدة السلسلة، نحصل على

$$\frac{\partial J}{\partial \mathbf{W}^{(1)}} = \text{prod}\left(\frac{\partial J}{\partial \mathbf{z}}, \frac{\partial \mathbf{z}}{\partial \mathbf{W}^{(1)}}\right) + \text{prod}\left(\frac{\partial J}{\partial s}, \frac{\partial s}{\partial \mathbf{W}^{(1)}}\right) = \frac{\partial J}{\partial \mathbf{z}} \mathbf{x}^T + \lambda \mathbf{W}^{(1)}.$$

5.3.4. تدريب الشبكات العصبية Training Neural Networks

عند تدريب الشبكات العصبية، يعتمد الانتشار الأمامي والخلفي على بعضهما البعض. على وجه الخصوص، من أجل الانتشار الأمامي، نجتاز الرسم البياني الحسابي في اتجاه التبعيات ونحسب جميع المتغيرات على مسارها. ثم يتم استخدام هذه من أجل الانتشار الخلفي حيث يتم عكس ترتيب الحساب على الرسم البياني.

خذ الشبكة البسيطة المذكورة أعلاه كمثال للتوضيح. من ناحية أخرى، يعتمد حساب مصطلح التنظيم (5.3.5) أثناء الانتشار الأمامي على القيم الحالية لمعلمات النموذج $\mathbf{W}^{(1)}$ و $\mathbf{W}^{(2)}$. يتم تقديمها بواسطة خوارزمية التحسين وفقًا للانتشار الخلفي في التكرار الأخير. من ناحية أخرى، يعتمد حساب التدرج للمعامل (5.3.11) أثناء الانتشار الخلفي على القيمة الحالية لمخرجات الطبقة المخفية \mathbf{h} ، والتي يتم تقديمها عن طريق الانتشار الأمامي.

لذلك عند تدريب الشبكات العصبية، بعد تهيئة معلمات النموذج، نتبادل الانتشار الأمامي مع الانتشار الخلفي، وتحديث معلمات النموذج باستخدام التدرجات الممعطاة عن طريق الانتشار الخلفي. لاحظ أن الانتشار الخلفي يعيد استخدام القيم الوسيطة المخزنة من الانتشار الأمامي لتجنب الحسابات المكررة. إحدى العوائق هي أننا بحاجة إلى الاحتفاظ بالقيم الوسيطة حتى يكتمل الانتشار الخلفي. هذا أيضًا أحد الأسباب التي يجعل التدريب يتطلب ذاكرة أكبر بكثير من التوقع البسيط. إلى جانب ذلك، يتناسب حجم هذه القيم الوسيطة تقريبًا مع عدد طبقات الشبكة وحجم الدفعية. وبالتالي، فإن تدريب شبكات أعمق باستخدام أحجام دفعات أكبر يؤدي بسهولة أكبر إلى نفاد أخطاء الذاكرة.

5.3.5 الملخص

- يحسب الانتشار الأمامي للمتغيرات الوسيطة ويخرنها بالتسلسل داخل الرسم البياني الحاسبي المحدد بواسطة الشبكة العصبية. ينطلق من المدخلات إلى طبقة الإخراج.
- يقوم الانتشار الخلفي بحساب وتخزين تدرجات المتغيرات والمعلمات الوسيطة بشكل تسلسلي داخل الشبكة العصبية بالترتيب المعكوس.
- عند تدريب نماذج التعلم العميق، فإن الانتشار الأمامي والانتشار الخلفي مترابطان.
- يتطلب التدريب ذاكرة أكبر بكثير من التوقع.

5.3.6 التمارين

1. افترض أن مدخلات \mathbf{X} بعض الدوال العددية f عبارة عن مصفوفات $m \times n$. ما هي أبعاد التدرج f بالنسبة لـ \mathbf{X} ؟
2. أضف تحيزًا إلى الطبقة المخفية من النموذج الموضح في هذا القسم (لا تحتاج إلى تضمين التحيز في مصطلح التنظيم).
 1. ارسم الرسم البياني الحاسبي المطابق.
 2. اشتق معادلات الانتشار الأمامية والخلفية.
3. احسب بصمة الذاكرة memory footprint للتدريب والتنبؤ في النموذج الموضح في هذا القسم.
4. افترض أنك تريد حساب المشتقات الثانية. ماذا يحدث للرسم البياني الحاسبي؟ كم من الوقت تتوقع أن تستغرق العملية الحسابية؟
5. افترض أن الرسم البياني الحاسبي كبير جدًا بالنسبة لوحدة معالجة الرسومات GPU الخاصة بك.
 1. هل يمكنك تقسيمها على أكثر من وحدة معالجة رسومات GPU؟
 2. ما هي مزايا وعيوب التدريب على minibatch أصغر؟

5.4. الاستقرار العددي والتهيئة Initialization

حتى الآن، كل نموذج قمنا بتطبيقه يتطلب أن نقوم بتهيئة معلماته وفقاً لبعض التوزيعات المحددة مسبقاً. حتى الآن، أخذنا مخطط التهيئة كأمر مسلم به، مع إخفاء تفاصيل كيفية اتخاذ هذه الخيارات. قد يكون لديك انطباع بأن هذه الخيارات ليست مهمة بشكل خاص. على العكس من ذلك، يلعب اختيار مخطط التهيئة دوراً مهماً في تعلم الشبكة العصبية، ويمكن أن يكون حاسماً للحفاظ على الاستقرار العددي. علاوة على ذلك، يمكن ربط هذه الاختيارات بطرق مثيرة باختيار دالة التشغيل اللاخطي. يمكن أن تحدد الدالة التي نختارها وكيف نهيئ المعلمات مدى سرعة تقارب خوارزمية التحسين الخاصة بنا. قد تؤدي الخيارات السيئة هنا إلى مواجهة تدرجات متدرجة أو متلاشية أثناء التدريب. في هذا القسم، نتعمق في هذه الموضوعات بمزيد من التفاصيل ونناقش بعض الاستدلالات المفيدة التي ستتجدد مفيدة طوال حياتك المهنية في التعلم العميق.

5.4.1. تلاشي وانفجار التدرجات Vanishing and Exploding Gradients

ضع في اعتبارك شبكة عميق ذات طبقات L ومدخلات \mathbf{x} ومخروجات \mathbf{o} . مع تحديد كل طبقة l من خلال تحويل f_l يتم تحديد معلماته بواسطة الأوزان $(\mathbf{W}^{(l)}, \mathbf{b}^{(l)})$ ، والتي يكون ناتج الطبقة المخفية $(\mathbf{h}^{(l)} = \mathbf{h}^{(0)}, \dots, \mathbf{h}^{(L-1)})$ ، يمكن التعبير عن شبكتنا على النحو التالي:

$$\mathbf{h}^{(l)} = f_l(\mathbf{h}^{(l-1)}) \text{ and thus } \mathbf{o} = f_L \circ \dots \circ f_1(\mathbf{x}).$$

إذا كانت جميع مخرجات ومدخلات الطبقة المخفية عبارة عن متوجهات، فيمكننا كتابة التدرج \mathbf{o} فيما يتعلق بأي مجموعة من المعلمات $(\mathbf{W}^{(l)}, \mathbf{b}^{(l)})$ على النحو التالي:

$$\partial_{\mathbf{W}^{(l)}} \mathbf{o} = \partial_{\mathbf{h}^{(L-1)}} \mathbf{h}^{(L)} \cdot \dots \cdot \partial_{\mathbf{h}^{(l)}} \mathbf{h}^{(l+1)} \partial_{\mathbf{W}^{(l)}} \mathbf{h}^{(l)}.$$

$$\mathbf{M}^{(L)} \stackrel{\text{def}}{=} \quad \mathbf{M}^{(l+1)} \stackrel{\text{def}}{=} \quad \mathbf{v}^{(l)} \stackrel{\text{def}}{=}$$

معنى آخر، هذا التدرج هو ناتج ضرب $l - L$ مصفوفات $\mathbf{M}^{(L+1)} \cdot \dots \cdot \mathbf{M}^{(l)}$ ومتوجه التدرج $\mathbf{v}^{(l)}$. وبالتالي نحن عرضة لنفس مشاكل التدفق العددي numerical underflow الذي غالباً ما يظهر عند ضرب العديد من الاحتمالات معاً. عند التعامل مع الاحتمالات، فإن الحيلة الشائعة هي التبديل إلى مساحة اللوغاريتم log-space، أي تحويل الضغط من الجزء العشري إلى أس التمثيل العددي. لسوء الحظ، فإن مشكلتنا أعلىأ أكثر خطورة: في البداية قد تحتوي المصفوفات $\mathbf{M}^{(l)}$ على مجموعة متنوعة من القيم الذاتية eigenvalues. قد تكون صغيرة أو كبيرة، وقد يكون منتجها كبيراً جداً أو صغيراً جداً.

تتجاوز المخاطر التي تشكلها التدرجات غير المستقرة التمثيل العددي. كما تهدد التدرجات ذات الحجم غير المتوقع أيضاً استقرار خوارزميات التحسين الخاصة بنا. قد نواجه تحديات

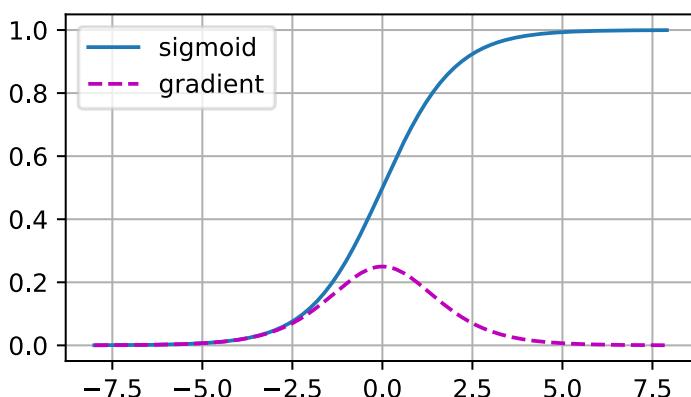
للمعلمات إما (1) كبيرة جدًا، مما يؤدي إلى تدمير نموذجنا (مشكلة التدرج exploding gradient)، أو (2) صغيرة للغاية (مشكلة التدرج المتباطئ vanishing gradient)، مما يجعل التعلم مستحيلاً لأن المعلمات بالكاد تتحرك في كل تحديث.

5.4.1.1 Vanishing Gradients تلاشي التدرجات

أحد الأسباب المتكررة لمشكلة تلاشي التدرج vanishing gradient هو اختيار دالة التنشيط التي يتم إلهاقها بعد العمليات الخطية لكل طبقة. تاريخياً، كانت دالة sigmoid $\frac{1}{1 + \exp(-x)}$ (المقدمة في القسم 5.1) شائعة لأنها تشبه دالة العتبة thresholding function، نظراً لأن الشبكات العصبية الاصطناعية المبكرة كانت مستوفحة من الشبكات العصبية البيولوجية، فإن فكرة الخلايا العصبية التي تطلق fire إما بشكل كامل أو لا تطلق not fire (مثل الخلايا العصبية البيولوجية) تبدو جذابة. دعونا نلقي نظرة فاحصة على دالة sigmoid لنرى لماذا يمكن أن يتسبب في تلاشي التدرجات.

```
%matplotlib inline
import tensorflow as tf
from d2l import tensorflow as d2l

x = tf.Variable(tf.range(-8.0, 8.0, 0.1))
with tf.GradientTape() as t:
    y = tf.nn.sigmoid(x)
d2l.plot(x.numpy(), [y.numpy(), t.gradient(y, x).numpy()],
         legend=['sigmoid', 'gradient'], figsize=(4.5, 2.5))
```



كما ترون، يتلاشى تدرج sigmoid عندما تكون مدخلاته كبيرة وعندما تكون صغيرة. علاوة على ذلك، عند الانتشار الخلفي عبر العديد من الطبقات، ما لم نكن في منطقة معتدلة ، حيث تقترب مدخلات العديد من sigmoid من الصفر ، فقد تختفي تدرجات المنتج الكلى. عندما تفتخر شبكتنا بالعديد من الطبقات، ما لم نتوخى الحذر، فمن المحتمل أن يتم قطع التدرج في طبقة ما. في الواقع، كانت هذه المشكلة تصيب التدريب الشبكي العميق. وبالتالي، فإن ReLU، التي هي أكثر استقراراً (ولكنها أقل قبولاً من الناحية العصبية)، ظهرت كخيار افتراضي للممارسين.

5.4.1.2. انفجار التدرجات Exploding Gradients

المشكلة المعاكسة، عندما تتفجر التدرجات، يمكن أن تكون محيرة بالمثل. لتوضيح هذا بشكل أفضل قليلاً، نرسم 100 مصفوفة عشوائية غاويسية ونضربها ببعض المصفوفة الأولية. بالنسبة للمقياس الذي اخترناه (اختيار التباين $1 = \sigma^2$)، ينفجر منتج المصفوفة. عندما يحدث هذا بسبب تهيئ شبكة عميق، فليس لدينا فرصة للحصول على محسن هبوط التدرج gradient descent optimizer ليتقارب.

```
M = tf.random.normal((4, 4))
print('a single matrix \n', M)
for i in range(100):
    M = tf.matmul(M, tf.random.normal((4, 4)))

print('after multiplying 100 matrices\n', M.numpy())
a single matrix
tf.Tensor(
[[ -3.7870526e-01 -8.8691898e-02 -1.1780064e+00
 4.2226687e-01]
 [ 1.5102199e+00  2.2903053e-01 -9.1348571e-01 -
 2.0801425e-01]
 [ -1.3337844e-03  8.2335420e-02 -2.4707975e+00 -
 1.1889901e+00]
 [ 7.2899163e-01 -7.2341427e-02  9.2103463e-01 -
 1.6827035e-01]], shape=(4, 4), dtype=float32)
after multiplying 100 matrices
[[ -1.4607350e+24 -5.4140549e+23 -1.7785702e+23
 1.0161147e+24]
 [ 2.3161050e+24  8.5843912e+23  2.8200557e+23 -
 1.6111261e+24]
 [ -1.0695384e+24 -3.9641278e+23 -1.3022544e+23
 7.4399098e+23]]
```

$$[\begin{array}{c} 1.6060195e+24 \\ 5.9525373e+23 \\ 1.9554657e+23 \\ - \\ 1.1171773e+24 \end{array}]$$

Breaking the Symmetry 5.4.1.3

مشكلة أخرى في تصميم الشبكة العصبية هي التنااظر او التماثل symmetry المتأصل في معاملاتهم. افترض أن لدينا MLP بسيطاً بطبقة مخفية واحدة ووحدتين. في هذه الحالة، يمكننا تبديل أوزان $W^{(1)}$ الطبقة الأولى وكذلك تبديل أوزان طبقة المخرجات للحصول على الدالة نفسها. لا يوجد شيء يميز يفرق بين الوحدة المخفية الأولى والوحدة المخفية الثانية. بعبارة أخرى، لدينا تنااظر تبادلي بين الوحدات المخفية لكل طبقة.

هذا أكثر من مجرد مصدر إزعاج نظري. ضع في اعتبارك MLP ذات الطبقة المخفية الواحدة المذكورة أعلاه مع وحدتين مخفيتين. للتوضيح، افترض أن طبقة المخرجات تحول الوحدتين المخفيتين إلى وحدة إخراج واحدة فقط. تخيل ما سيحدث إذا قمنا بتهيئة جميع معلمات الطبقة المخفية كـ $c = W^{(1)}$ بالنسبة لبعض الشواطئ. في هذه الحالة، أثناء الانتشار الأمامي، تأخذ إما الوحدة المخفية نفس المدخلات والمعلمات، وتنتج نفس التشخيص، الذي يتم تغذيته إلى وحدة الإخراج. أثناء الانتشار الخلفي، فإن التمييز بين وحدة الإخراج فيما يتعلق بالمعلمات $W^{(1)}$ يعطي التدرج الذي تأخذ جميع عناصره نفس القيمة. وهكذا، بعد التكرار القائم على التدرج (على سبيل المثال، التدرج الاشتقافي العشوائي المصغر)، لا تزال جميع العناصر تأخذ نفس القيمة. لن تكسر مثل هذه التكرارات التنااظر من تقاء نفسها وقد لا نتمكن أبداً من إدراك القوة التعبيرية للشبكة. تتصرف الطبقة المخفية كما لو كانت تحتوي على وحدة واحدة فقط. لاحظ أنه في حين أن التدرج الاشتقافي العشوائي المصغر لن يكسر هذا التنااظر، فإن تسوية التسرب dropout regularization (التي سيتم تقديمها لاحقاً) ستفعل!

Parameter Initialization 5.4.2

إحدى طرق معالجة – أو على الأقل التخفيف – المشكلات التي أثيرت أعلاه هي من خلال التهيئة الدقيقة initialization careful. كما سنرى لاحقاً، يمكن أن تؤدي الرعاية الإضافية أثناء التحسين والتنظيم المناسب إلى زيادة تعزيز الاستقرار.

Default Initialization 5.4.2.1

في الأقسام السابقة، على سبيل المثال، في القسم 3.5، استخدمنا التوزيع الطبيعي normal distribution لتهيئة قيم الأوزان الخاصة بنا. إذا لم نحدد طريقة التهيئة، فسيستخدم إطار العمل طريقة تهيئة عشوائية افتراضية، والتي غالباً ما تعمل جيداً في الممارسة العملية لأحجام المشكلات المعتدلة.

Xavier. 5.4.2.2

دعنا نلقي نظرة على توزيع مقياس الناتج o_i لبعض الطبقات المتصلة بالكامل بدون الالخطية. مع n_{in} المدخلات x_j والأوزان w_{ij} المرتبطة بها لهذه الطبقة، يتم إعطاء الإخراج بواسطة

$$o_i = \sum_{j=1}^{n_{\text{in}}} w_{ij} x_j.$$

يتم رسم جميع الأوزان w_{ij} بشكل مستقل عن نفس التوزيع. علاوة على ذلك، لنفترض أن هذا التوزيع ليس له أي متوسط وتبابين σ^2 . لاحظ أن هذا لا يعني أن التوزيع يجب أن يكون غاوسيًا، فقط أن المتوسط والتباين يجب أن يكونا موجودين. في الوقت الحالي، لنفترض أن مدخلات الطبقة j لها أيضًا متوسط صفر وتبابين γ^2 وأنها مستقلة عن w_{ij} ومستقلة عن بعضها البعض. في هذه الحالة، يمكننا حساب المتوسط والتباين o_i على النحو التالي:

$$\begin{aligned} E[o_i] &= \sum_{j=1}^{n_{\text{in}}} E[w_{ij} x_j] \\ &= \sum_{j=1}^{n_{\text{in}}} E[w_{ij}] E[x_j] \\ &= 0, \\ \text{Var}[o_i] &= E[o_i^2] - (E[o_i])^2 \\ &= \sum_{j=1}^{n_{\text{in}}} E[w_{ij}^2 x_j^2] - 0 \\ &= \sum_{j=1}^{n_{\text{in}}} E[w_{ij}^2] E[x_j^2] \\ &= n_{\text{in}} \sigma^2 \gamma^2. \end{aligned}$$

طريقة واحدة للحفاظ على التباين ثابتاً هي تعين $1 = n_{\text{in}} \sigma^2$. الآن النظري للانتشار الخلقي. هناك نواجه مشكلة مماثلة، وإن كان يتم نشر التدرجات من الطبقات الأقرب إلى الإخراج. باستخدام نفس المنطق الخاص بالانتشار الأمامي، نرى أن تباين التدرجات يمكن أن ينفجر ما لم يكن $1 = n_{\text{out}} \sigma^2$ حيث n_{out} . هذا يترکنافي مأزق: لا يمكننا تلبية كلا الشرطين في وقت واحد. بدلاً من ذلك، نحاول ببساطة أن نتحقق

$$\frac{1}{2} (n_{\text{in}} + n_{\text{out}}) \sigma^2 = 1 \text{ or equivalently } \sigma = \sqrt{\frac{2}{n_{\text{in}} + n_{\text{out}}}}.$$

هذا هو السبب الكامن وراء تهيئة Xavier الحالية والمفيدة عملياً، والتي سميت على اسم المؤلف الأول لمنشئها (Xavier, Glorot and Bengio, 2010). نموذجياً، عينات التهيئة Xavier الأوزان من توزيع غاوسي بمتوسط صفر وتباعن $\sigma^2 = \frac{2}{n_{in} + n_{out}}$. يمكننا أيضاً تكيف حدس Xavier لاختيار التباين عندأخذ عينات من الأوزان من توزيع منتظم. لاحظ أن التوزيع المنتظم $U(-a, a)$ له تباين $\frac{a^2}{3}$. يؤدي توصيل $\frac{a^2}{3}$ في حالتنا على σ^2 إلى اقتراح التهيئة وفقاً لـ

$$U\left(-\sqrt{\frac{6}{n_{in} + n_{out}}}, \sqrt{\frac{6}{n_{in} + n_{out}}}\right).$$

على الرغم من أن افتراض عدم وجود اللاخطية في التفكير الرياضي أعلاه يمكن انتهائه بسهولة في الشبكات العصبية، إلا أن طريقة تهيئة Xavier تعمل بشكل جيد في الممارسة العملية.

Beyond 5.4.2.3 وراء

المنطق أعلاه بالكاد يخدش سطح الأساليب الحديثة لتهيئة المعلمة parameter initialization. غالباً ما ينفذ إطار التعلم العميق أكثر من اثنين عشرة عملية استكشاف مختلفة. علاوة على ذلك، لا تزال تهيئة المعلمة منطقة ساخنة للبحث الأساسي في التعلم العميق. من بين هذه الاستدلالات المتخصصة للمعلمات المقيدة (المشتركة) والدقة الفائقة - super-resolution ونماذج التسلسل sequence models وغيرها من المواقف. على سبيل المثال، أظهر إمكانية تدريب 10000 طبقة من الشبكات العصبية بدون حيل معمارية Xiao et al (2018).

إذا كان الموضوع يثير اهتمامك، فإننا نقترح التعمق في عروض هذه الوحدة، وقراءة الأوراق التي اقترحت وحلل كل إرشادية، ثم استكشاف أحدث المنشورات حول هذا الموضوع. ربما سوف تتعرّأ أو تخترع فكرة ذكية وتساهم في تنفيذ أطر التعلم العميق.

5.4.3 الملخص

- يعد تلاشي وانفجار التدرجات من المشكلات الشائعة في الشبكات العميقه. مطلوب عناية كبيرة في تهيئة المعلمات لضمان أن تظل التدرجات والمعلمات خاضعة للرقابة بشكل جيد.

- هناك حاجة إلى أساليب التهيئة الاستدلالي الحدسية Initialization heuristics للتأكد من أن التدرجات الأولية ليست كبيرة جداً ولا صغيرة جداً.
- تعمل دوال تنشيط ReLU على تخفيف مشكلة تلاشي التدرج. هذا يمكن أن يسرع من التقارب.

• التهيئة العشوائية Random initialization هي المفتاح لضمان كسر التمازج symmetry قبل التحسين.

• تشير تهيئة Xavier إلى أنه، لكل طبقة، لا يتأثر التباين في أي ناتج بعدد المدخلات، ولا يتأثر تباين أي تدرج بعدد المخرجات.

5.4.4. التمارين

1. هل يمكنك تصميم حالات أخرى قد تعرض فيها الشبكة العصبية تنازلاً يتطلب كسرًا إلى جانب تنازلي التقليل permutation symmetry breaking؟ MLP

2. هل يمكننا تهيئة جميع معلمات الوزن في الانحدار الخطى أوفي انحدار softmax إلى نفس القيمة؟

3. ابحث عن الحدود التحليلية للقيم الذاتية لحاصل ضرب مصفوفتين. ماذا يخبرك هذا عن ضمان تكيف التدرجات بشكل جيد؟

4. إذا علمنا أن بعض المصطلحات تتبع diverge، فهل يمكننا إصلاح هذا بعد الحقائق؟ انظر إلى الورقة حول مقياس المعدل التكيفي للطبقات للإلهام You et al. (2017).

5.5 التععمق في التعلم العميق Generalization in Deep Learning

في القسم 3 والقسم 4، عالجنا مشاكل الانحدار والتصنيف من خلال ملاءمة fitting النماذج الخطية لبيانات التدريب. في كلتا الحالتين، قدمنا خوارزميات عملية للعثور على المعلمات التي زادت من احتمالية تسميات التدريب المرصودة. وبعد ذلك، قرب نهاية كل فصل، تذكرنا أن ملائمة بيانات التدريب كانت مجرد هدف وسيط. كان سعينا الحقيقي طوال الوقت هو اكتشاف الأنماط العامة التي يمكننا على أساسها إجراء تنبؤات دقيقة حتى على الأمثلة الجديدة المستمدة من نفس المجموعة الأساسية. باحثو التعلم الآلي هم مستهلكون لخوارزميات التحسين. في بعض الأحيان، يجب علينا تطوير خوارزميات تحسين جديدة. ولكن في نهاية المطاف، فإن التحسين هو مجرد وسيلة لتحقيق غاية. يعد التعلم الآلي في جوهره نظاماً إحصائياً ونرحب في تحسين خطأ التدريب فقط بقدر ما يؤدي بعض المبادئ الإحصائية (المعروفة أو غير المعروفة) إلى التععمق الناتج عن مجموعة التدريب.

على الجانب المشرق، اتضح أن الشبكات العصبية العميقية المدربة عن طريق التدرج الاستقرائي العشوائي SGD تعم جيداً بشكل ملحوظ عبر مشاكل التنبؤ التي لا تعد ولا تحصى، والتي تشمل الرؤية الحاسوبية computer vision؛ معالجة اللغة الطبيعية natural language processing؛ بيئات السلاسل الزمنية time series data؛ أنظمة التوصية recommender systems؛ السجلات الصحية الإلكترونية electronic health records؛ طي البروتين systems

Protein folding؛ تقريب دالة القيمة في ألعاب الفيديو وألعاب الطاولة؛ ومجالات أخرى لا حصر لها. على الجانب السلبي، إذا كنت تبحث عن حساب مباشر إما لقصة التحسين (لماذا يمكننا ملاءمتها لبيانات التدريب) أو قصة التعميم (لماذا تعمم النماذج الناتجة على أمثلة غير مرئية)، إذن قد ترغби في سكب شاي لنفسك. في حين أن إجراءاتنا لتحسين النماذج الخطية والخصائص الإحصائية للحلول موصوفة جيداً من خلال مجموعة نظرية شاملة، فإن فهمنا للتعلم العميق لا يزال يشبه الغرب المتواهش على كلا الجبهتين.

تطور نظرية وممارسة التعلم العميق بسرعة على كلا الجبهتين، حيث يتبنى المنظرون استراتيجيات جديدة لشرح ما يحدث، حتى مع استمرار الممارسين في الابتكار بوتيرة مذهلة، وبناء ترسانات من الاستدلال لتدريب الشبكات العميقه ومجموعة من البديهيات والمعارف الشعية التي تقدم إرشادات لتحديد التقنيات التي يجب تطبيقها في أي موقف.

لفترة طويلة، لم نقرأ في الوقت الحالي هو أن نظرية التعلم العميق قد أنتجت خطوط هجوم واحدة ونتائج رائعة متفرقة، لكنها لا تزال بعيدة كل البعد عن تفسير شامل لكل من (1) سبب قدرتنا على تحسين الشبكات العصبية و (2) كيف تمكنت النماذج التي تم تعلمها من خلال SGD من التعميم جيداً، حتى في المهام عالية الأبعاد. ومع ذلك، من الناحية العملية، (1) نادرًا ما يمثل مشكلة (يمكننا دائمًا العثور على المعلمات التي تناسب جميع بيانات التدريب لدينا) وبالتالي فإن فهم التعميم هو المشكلة الأكبر. من ناحية أخرى، حتى في غياب الراحة التي توفرها النظرية العلمية المتماسكة، فقد طور الممارسون مجموعة كبيرة من التقنيات التي قد تساعدك على إنتاج نماذج تعمم جيداً في الممارسة. في حين أنه لا يمكن لأي ملخص بلغ أن ينصف موضوع التعميم الكبير في التعلم العميق، وبينما لا تزال الحالة العامة للبحث بعيدة عن الحل، نأمل، في هذا القسم، أن نقدم نظرة عامة واسعة عن حالة البحث والممارسة.

5.5.1 إعادة النظر في فرط التجهيز والتنظيم Revisiting Overfitting and Regularization

تذكر أن نهجنا في تدريب نماذج التعلم الآلي يتكون عادةً من مرحلتين: (1) ملاءمة fit بيانات التدريب؛ و (2) تقدير خطأ التعميم generalization error (الخطأ الحقيقي على السكان الأساسيين) من خلال تقييم النموذج على بيانات الانتظار. يُطلق على الفرق بين ملاءمتنا لبيانات التدريب ومدى ملاءمتنا لبيانات الاختبار فجوة التعميم generalization gap وعندما تكون فجوة التعميم كبيرة، نقول إن نماذجنا تتلاءم مع بيانات التدريب. في الحالات القصوى من فرط التجهيز overfitting، قد نلائم بيانات التدريب تماماً، حتى عندما يظل خطأ الاختبار كبيراً. ومن وجهة النظر الكلاسيكية، فإن التفسير هو أن نماذجنا معقدة للغاية، وتتطلب إما تقليل عدد

الميزات، أو عدد المعلمات غير الصفرية التي تم تعلمها، أو حجم المعلمات كما تم تحديدها كمياً. تذكر مخطط تعقيد النموذج مقابل الخطأ (الشكل 3.6.1) من القسم 3.6.

لكن التعلم العميق يعقد هذه الصورة بطرق غير بدائية. أولاً، بالنسبة إلى مشاكل التصنيف، عادةً ما تكون نماذجنا معيبة بما يكفي لتناسب تماماً كل مثال تدريسي، حتى في مجموعات البيانات التي تتكون من الملايين (Zhang et al., 2021). في الصورة الكلاسيكية، قد نعتقد أن هذا الإعداد يقع في أقصى اليمين المتطرف لمحور تعقيد النموذج، وأن أي تحسينات في خطأ التعميم يجب أن تأتي عن طريق التنظيم regularization، إما عن طريق تقليل تعقيد فئة النموذج، أو عن طريق تطبيق عقوبة penalty، تعقيد بشدة مجموعة القيم التي قد تتخذها معلماتنا. ولكن هذا هو المكان الذي تبدأ فيه الأمور في أن تصبح غريبة.

الغريب، بالنسبة للعديد من مهام التعلم العميق (على سبيل المثال، التعرف على الصور وتصنيف النص) نختار عادةً من بين النماذجية، والتي يمكن أن تتحقق جميعها خطأ تدريب منخفض بشكل تعسفي (وخطأ تدريب صفرى). نظراً لأن جميع النماذج قيد الدراسة لا تتحقق أي خطأ في التدريب، فإن السبيل الوحيد لتحقيق المزيد من المكاسب هو تقليل الضبط الزائد (فرط التجهيز). والأغرب من ذلك، أنه على الرغم من ملاءمة بيانات التدريب بشكل مثالي، يمكننا في الواقع تقليل خطأ التعميم بشكل أكبر عن طريق جعل النموذج أكثر تعبيراً، على سبيل المثال، إضافة طبقات أو عقد أو تدريب لعدد أكبر من الفترات epochs. الغريب حتى الآن، أن النمط الذي يربط فجوة التعميم بتعقيد النموذج (كما تم التقاطه، على سبيل المثال، في عمق أو عرض الشبكات) يمكن أن يكون غير رتيب non-monotonic، مع تعقيد أكبر يضر في البداية ولكنه يساعد لاحقاً فيما يسمى نمط "النسب المزدوج double-descent" (Nakkiran et al., 2021). وبالتالي فإن ممارس التعلم العميق يمتلك مجموعة من الحيل، والتي يبدو أن بعضها يقييد النموذج بطريقة ما والبعض الآخر يجعله يبدو أكثر تعبيراً، وكلها، بمعنى ما، يتطلبها لتخفيض فرط التجهيز.

ومما يزيد الأمور تعقيداً، في حين أن الضمانات التي توفرها نظرية التعلم الكلاسيكية يمكن أن تكون متحفظة حتى بالنسبة للنماذج الكلاسيكية، فإنها تبدو عاجزة عن تفسير سبب تعميم الشبكات العصبية العميق في المقام الأول. نظراً لأن الشبكات العصبية العميق قادرة على تركيب تسميات عشوائية حتى لمجموعات البيانات الكبيرة، وعلى الرغم من استخدام طرق مألوفة مثل التنظيم، فإن حدود التعميم التقليدية القائمة على التعقيد، على سبيل المثال، تلك القائمة على بعد VC أو تعقيد Rademacher لفئة فرضية لا يمكنها تفسير السبب تعميم الشبكات العصبية.

5.5.2 إلهام من غير البارامترية Inspiration from Nonparametrics

عند الاقتراب من التعلم العميق لأول مرة، من المعمري اعتبارها نماذج بارامترية. بعد كل شيء، النماذج لديها الملائين من المعلمات. عندما نقوم بتحديث النماذج، نقوم بتحديث معلماتها. عندما نحفظ النماذج، نكتب معلماتها على القرص. ومع ذلك، فإن الرياضيات وعلوم الكمبيوتر مليئة بالتغييرات غير البديهية في المنظور، وتبدو التشابهات المفاجئة مشاكل مختلفة على ما يبدو. بينما تحتوي الشبكات العصبية بوضوح على معلمات، من بعض النواحي، قد يكون من المفيد التفكير فيها على أنها تتصرف مثل النماذج اللامعلمية (غير البارامترية). إذن ما الذي يجعل نموذجاً غير ملحي على وجه التحديد؟ بينما يغطي الاسم مجموعة متنوعة من الأساليب، فإن أحد الموضوعات الشائعة هو أن الأساليب اللامعلمية تميل إلى أن يكون لها مستوى من التعقيد ينمو مع زيادة كمية البيانات المتاحة.

ربما يكون أبسط مثال على نموذج غير ملحي هو خوارزمية الجار الأقرب k -nearest neighbor (سنعطي المزيد من النماذج اللامعلمية لاحقاً، كمافي القسم 11.2). هنا، في وقت التدريب، يحفظ المتعلم ببساطة مجموعة البيانات. ثم، في وقت التنبؤ، عندما يواجه المتعلم نقطة جديدة \mathbf{x} ، يبحث عن أقرب الجيران (k من النقاط \mathbf{x}_i) تقلل من بعض المسافة ($d(\mathbf{x}, \mathbf{x}_i)$). عندما $k = 1$ ، تسمى هذه الخوارزمية 1-nearest neighbors. وستتحقق الخوارزمية دائمًا خطأ تدريب قدره صفر. لكن هذا لا يعني أن الخوارزمية لن تعم في الواقع، اتضح أنه في ظل بعض الظروف المعتدلة، تكون خوارزمية الجار الأقرب متسقة (تقرب في النهاية إلى المتبني الأمثل).

لاحظ أن أحد الجيران الأقرب يتطلب أن نحدد بعض دوال المسافة d ، أو بشكل مكافئ، أن نحدد بعض دوال الأساس ذات القيمة المتجهة (\mathbf{x}) ϕ لتمييز بيانتنا. لأي اختيار لمقياس المسافة، سنتحقق خطأ تدريب ونصل في النهاية إلى متبني مثالي، لكن مقاييس المسافة المختلفة d تشفّر تحيزات استقرائية مختلفة وبكمية محدودة من البيانات المتاحة ستتّبع تنبؤات مختلفة. تمثل الاختيارات المختلفة لمقياس المسافة d افتراضات مختلفة حول الأنماط الأساسية وسيعتمد أداء المتبنيين المختلفين على مدى توافق الافتراضات مع البيانات المرصودة.

بمعنى ما، نظرًا لأن الشبكات العصبية مفرطة في المعلمات، وتمتلك العديد من المعلمات أكثر مما هو مطلوب لملاءمة بيانات التدريب، فإنها تميل إلى استيفاء بيانات التدريب (لتناسبها تمامًا) وبالتالي تتصرف، في بعض النواحي، مثل النماذج اللامعلمية nonparametric models. أثبتت الأبحاث النظرية الحديثة ارتباطًا عميقًا بين الشبكات العصبية الكبيرة والطرق اللامعلمية، ولا سيما طرق النواة kernel methods. على وجه الخصوص، أظهر (Jacot et al., 2018) أنه في الحد الأقصى، مع نمو البيرسيترون متعدد الطبقات مع أوزان مُهيأة عشوائيًا بشكل لا نهائي، تصبح مكافئة لطرق النواة (اللامبارامترية) لاختيار معين للدالة النواة kernel (بشكل أساسى، دالة

المسافة)، والتي يسمونها نواة المماس العصبية neural tangent kernel. في حين أن نماذج نواة المماس العصبية الحالية قد لا تفسر بشكل كامل سلوك الشبكات العميقة الحديثة، فإن نجاحها كأداة تحليلية يؤكّد فائدة النمذجة اللامعليمية لفهم سلوك الشبكات العميقة ذات المعلومات المفرطة.

5.5.3 التوقف المبكر Early Stopping

في حين أن الشبكات العصبية العميق قادرة على ملائمة fitting تسميات عشوائية، حتى عندما يتم تعين التسميات بشكل غير صحيح أو عشوائي (Zhang et al., 2021)، فإن هذه القدرة تظهر فقط عبر العديد من تكرارات التدريب. كشف خط عمل جديد (Rolnick et al., 2017) أنه في إعداد ضوابط التسمية، تميل الشبكات العصبية إلى احتواء البيانات المصنفة بشكل نظيف أو لاً وبعد ذلك فقط لاستيفاء البيانات ذات التسمية الخاطئة mislabeled data.علاوة على ذلك، فقد ثبت أن هذه الظاهرة تُترجم مباشرة إلى ضمان على التعميم: فكلما كان النموذج مناسباً للبيانات المصنفة بشكل واضح ولكن ليس الأمثلة المعرونة عشوائياً المدرجة في مجموعة التدريب، فقد تم تعميمه في الواقع (Garg et al., 2021).

تساعد هذه النتائج معافي تحفيز التوقف المبكر early stopping، وهي تقنية كلاسيكية لتنظيم الشبكات العصبية العميق. هنا، بدلاً من تقييد قيم الأوزان بشكل مباشر، يقييد المرء عدد فترات التدريب. الطريقة الأكثر شيوعاً لتحديد معايير التوقف هي مراقبة خطأ التحقق من الصحة خلال التدريب (عادةً عن طريق التتحقق مرة واحدة بعد كل فترة) وقطع التدريب عندما لا ينخفض خطأ التتحقق بأكثر من مقدار صغير لبعض الفترات. هذا يسمى أحياناً معايير الصبر patience criteria. إلى جانب إمكانية أن يؤدي إلى تعميم أفضل، في وضع التسميات الصاخبة noisy labels، هناك فائدة أخرى للتوقف المبكر وهي الوقت الذي يتم توفيره. بمجرد استيفاء معايير الصبر، يمكن للمرء إنهاء التدريب. بالنسبة للنماذج الكبيرة التي قد تتطلب أيامًا من التدريب في وقت واحد عبر 8 وحدات معالجة رسومات أو أكثر، يمكن أن يوفر التوقف المبكر الذي يتم ضبطه جيداً على الباحثين أيامًا من الوقت ويمكن أن يوفر على أصحاب العمل عدة آلاف من الدولارات.

والجدير بالذكر أنه في حالة عدم وجود ضوابط على التسمية وتكون مجموعات البيانات قابلة للتحقيق (يمكن فصل الفئات حقاً، على سبيل المثال، التمييز بين القطط والكلاب)، فإن التوقف المبكر لا يؤدي إلى تحسينات كبيرة في التعميم. من ناحية أخرى، عندما يكون هناك ضوابط في التسمية، أو تباين جوهري في التسمية (على سبيل المثال، التنبؤ بالوفيات بين المرضى)، فإن التوقف المبكر أمر بالغ الأهمية. عادة ما تكون نماذج التدريب حتى تستكمل البيانات المزعجة فكرة سيئة.

5.5.4 طرق التنظيم الكلاسيكية للشبكات العميقه

Classical Regularization Methods for Deep Networks

في القسم 3، وصفنا العديد من تقنيات التنظيم الكلاسيكية لتقيد تعقيد نماذجنا. على وجه الخصوص، قدم القسم 3.7 طريقة تسمى انحلال الوزن weight decay، والتي تتكون من إضافة مصطلح تنظيم إلى دالة الخطأ لمعاقبة القيم الكبيرة للأوزان. اعتماداً على معيار الوزن الذي يتم المعاقبة عليه، تعرف هذه التقنية إما باسم تنظيم ridge (للمقوعة ℓ_2) أو تنظيم lasso (للمقوعة ℓ_1). في التحليل الكلاسيكي لهؤلاء المنظمين، يُنظر إليهم على أنهما يقيدون القيم التي يمكن أن تأخذها الأوزان بشكل كافٍ لمنع النموذج من ملاعة التسميات العشوائية.

في تطبيقات التعلم العميق، يظل انحلال الوزن أداة شائعة. ومع ذلك، فقد لاحظ الباحثون أن نقاط القوة النموذجية للتنظيم غير كافية لمنع الشبكات من استيفاء البيانات (Zhang et al., 2021). وبالتالي فإن الفوائد إذا فُسرت على أنها تنظيم قد تكون منطقية فقط بالاقتران مع معايير التوقف المبكر. في غياب التوقف المبكر، من الممكن أن تؤدي هذه الأساليب إلى تعميم أفضل، ليس لأنها تقيد بشكل هادف قوة الشبكة العصبية ولكن لأنها تقوم بطريقة ما بتشغير التحيزات الاستقرائية inductive biases التي تتوافق بشكل أفضل مع الأنماط الموجودة في مجموعات بيانات الاهتمامات. وهكذا، يظل المنظمون الكلاسيكيون شائعين في تطبيقات التعلم العميق، حتى لو كان الأساس المنطقي النظري لفعاليتهم مختلفاً جذرياً.

والجدير بالذكر أن باحثي التعلم العميق قد اعتمدوا أيضاً على التقنيات التي تم تعميمها لأول مرة في سياقات التنظيم الكلاسيكية، مثل إضافة الضوضاء إلى مدخلات النموذج. في القسم التالي، سنقدم تقنية الحذف العشوائي (التسرب) dropout الشهيرة (التي ابتكرها سريفاستافا وآخرون (2014)), والتي أصبحت الداعمة الأساسية للتعلم العميق، حتى معبقاء الأساس النظري لفعاليتها غامضاً بالمثل.

5.5.5 الملخص

على عكس النماذج الخطية الكلاسيكية، التي تميل إلى أن تحتوي على معلمات أقل من الأمثلة، تمثل الشبكات العميقية إلى الإفراط في تحديد المعلمات، وفي معظم المهام تكون قادرة على ملاعة مجموعة التدريب بشكل مثالى. يتحدى نظام الاستيفاء interpolation regime هذا العديد من البديهيات الشديدة السرعة. من الناحية الوظيفية، تبدو الشبكات العصبية مثل النماذج البارامترية. لكن التفكير فيها كنماذج غير معلمية يمكن أن يكون أحياناً مصدرًا أكثر موثوقية للحدس. نظرًا لأنه غالباً ما تكون جميع الشبكات العميقية قيد الدراسة قادرة على ملاعة جميع تسميات التدريب، يجب أن تأتي جميع المكافئات تقريرياً عن طريق التخفيف من فرط التجهيز (سد فجوة التعميم). ومن المفارقات أن التدخلات التي تقلل فجوة التعميم تظهر أحياناً أنها تزيد

من تعقيد النموذج وفي أوقات أخرى يبدو أنها تقلل من التعقيد. ومع ذلك، نادرًا ما تقلل هذه الأساليب التعقيد بما يكفي للنظرية الكلاسيكية لشرح تعميم الشبكات العميق، ولماذا تؤدي بعض الخيارات إلى تحسين التعميم يظل في الغالب سؤالًا مفتوحًا كبيرًا على الرغم من الجهد المتضادرة للعديد من الباحثين اللامعين.

5.5.6 التمارين

1. بأي معنى تفشل المقايس التقليدية القائمة على التعقيد في تفسير تعميم الشبكات العصبية العميق؟
2. لماذا يمكن اعتبار التوقف المبكر تقنية تنظيم regularization technique؟
3. كيف يحدد الباحثون عادةً معايير التوقف stopping criteria؟
4. ما هو العامل المهم الذي يبدو أنه يميز الحالات عندما يؤدي التوقف المبكر إلى تحسينات كبيرة في التعميم generalization؟
5. بعد التعميم، صفات أخرى للتوقف المبكر early stopping.

5.6 الحذف العشوائي Dropout

دعونا نفكري بإيجاز فيما نتوقعه من نموذج تنبؤي جيد good predictive model. نريد لها أن تعمل بشكل جيد على البيانات غير المرئية unseen data. تقترح نظرية التعميم الكلاسيكية أنه لسد الفجوة بين أداء التدريب والاختبار، يجب أن نهدف إلى نموذج بسيط. يمكن أن تأتي البساطة في شكل عدد صغير من الأبعاد. اكتشفنا ذلك عند مناقشة دوال الأساس الأحادي monomial basis functions للنماذج الخطية في القسم 3.6. بالإضافة إلى ذلك، كما رأينا عند مناقشة تناقص الوزن (التنظيم ℓ_2) في القسم 3.7، فإن المعيار norm (العکسی) للمعلمات يمثل أيضًا مقاييسًا مفيدةً للبساطة. مفهوم آخر مفيد للبساطة هو النعومة smoothness، أي أن الدالة لا ينبغي أن تكون حساسة للتغييرات الصغيرة في مدخلاتها. على سبيل المثال، عندما نصف الصور، تتوقع أن تكون إضافة بعض الضوضاء العشوائية إلى وحدات البكسل غير ضارة في الغالب.

في عام 1995، صاغ كريستوفر بيتشوب هذه الفكرة عندما أثبت أن التدريب باستخدام ضوضاء الإدخال يعادل تنظيم تيخونوف Tikhonov regularization (Bishop, 1995). رسم هذا العمل ارتباطًا رياضيًّا واضحًا بين شرط أن تكون الدالة سلسة smooth (وبالتالي بسيطة)، ومتطلبات أن تكون مرنة resilient للاضطرابات في المدخلات.

ثم، في عام 2014، سريفاستافا وأخرون. (Srivastava et al., 2014) فكرت ذكية عن كيفية تطبيق فكرة بيتشوب على الطبقات الداخلية للشبكة أيضًا. تتضمن فكريتهم، المسماة الحذف العشوائي (التسرب) dropout، حقن الضوضاء أثناء حساب كل طبقة داخلية أثناء الانتشار الأمامي، وقد أصبحت تقنية قياسية لتدريب الشبكات العصبية. تسمى هذه الطريقة ب dropout.

لأننا حرفيًا نتخلى عن بعض الخلايا العصبية أثناء التدريب. خلال التدريب، في كل تكرار، يتكون الحذف العشوائي القياسي من تصفية جزء من العقدفي كل طبقة قبل حساب الطبقة التالية.

لكي نكون واضحين، نحن نفرض روايتنا الخاصة بالارتباط ببساطة. تقدم الورقة الأصلية حول dropout حدساً من خلال تشبثه بمدهش للتکاثر الجنسي. يجادل المؤلفون بأن فرط التجهيز للشبكة العصبية يتميز بحالة تعتمد فيها كل طبقة على نمط معين من التنشيطات في الطبقة السابقة، مما يطلق على هذا الشرط التكيف المشترك co-adaptation condition. وهم يزعمون أن dropout يؤدي إلى تفكك التكيف المشترك تماماً كما يقال إن التکاثر الجنسي يفكك الجينات المتکيفية معًا. في حين أن شرح هذه النظرية مطروح للنقاش بالتأكيد، فقد أثبتت أسلوب dropout نفسه أنه ثابت، ويتم تفزيذه من dropout في معظم مكتبات التعلم العميق.

التحدي الرئيسي هو كيفية ضخ هذه الموضوعات. تمثل إحدى الأفكار في حقن الموضوعات بطريقة غير منحازة بحيث تكون القيمة المتوقعة لكل طبقة - أثناء تثبيت الطبقات الأخرى - متساوية للقيمة التي كانت ستأخذها في غياب الموضوعات. في عمل بيسوب، أضاف موضوعات غاوسيّة إلى المدخلات إلى نموذج خطّي. في كل تكرار تدريسي، أضاف موضوعات مأخوذة من توزيع بمتوسط صفر $\mathcal{N}(0, \sigma^2)$ إلى المدخلات x ، مما أسفر عن نقطة مضطربة $x' = x + \epsilon$. في توقع $E[x'] = x$

في تنظيم dropout القياسي، يقوم أحد الأصفار بإزالة بعض أجزاء العقدفي كل طبقة ثم يقوم بإزالة الحواف من كل طبقة عن طريق التسويه بواسطة جزء العقد التي تم الاحتفاظ بها (غير المسقطة أو المحذوفة not dropped out). بمعنى آخر، مع احتمال الحذف العشوائي p ، يتم استبدال كل تنشيط وسيط h بمتغير عشوائي h' على النحو التالي:

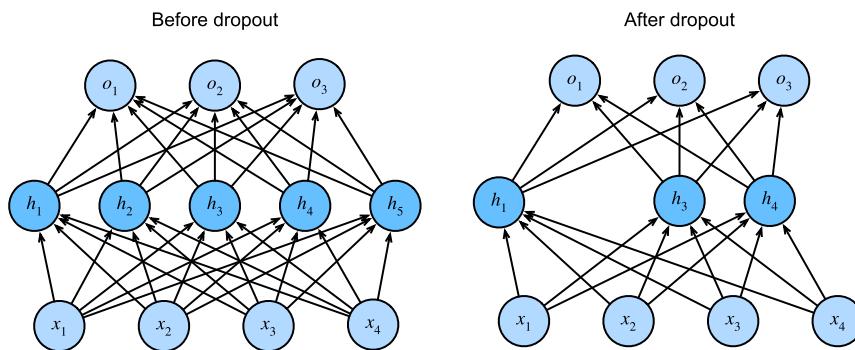
$$h' = \begin{cases} h & \text{with probability } p \\ 0 & \text{otherwise} \end{cases}$$

حسب التصميم، يظل التوقع دون تغيير، أي $E[h'] = h$

5.6.1. الحذف العشوائي في الممارسة Dropout in Practice

تدذكر MLP بطبقة مخفية و5 وحدات مخفية في الشكل 5.1.1. عندما نطبق dropout على طبقة مخفية، مع استبعاد كل وحدة مخفية باحتمالية، يمكن عرض النتيجة كشبكة تحتوي فقط على مجموعة فرعية من الخلايا العصبية الأصلية في الشكل 5.6.1 h_2 و h_5 تم إزالتها. وبالتالي، فإن حساب النواتج لم يعد يعتمد على h_2 أو h_5 والدرج الخاص بكل منها يختفي

أيضاً عند إجراء الانتشار الخلفي، بهذه الطريقة، لا يمكن أن يعتمد حساب طبقة المخرجات بشكل مفرط على أي عنصر واحد من h_1, \dots, h_5 .



شكل 1.5.6.1 MLP قبل وبعد dropout.

عادة، نقوم بتعطيل dropout في وقت الاختبار. بالنظر إلى نموذج مدرب ومثال جديد، فإننا لا نتجاهل أي عقد وبالتالي لا نحتاج إلى التسوية normalization. ومع ذلك، هناك بعض الاستثناءات: يستخدم بعض الباحثين dropout في وقت الاختبار كإرشاد لتقدير عدم اليقين uncertainty في تنبؤات الشبكة العصبية: إذا اتفقت التوقعات عبر العديد من أقنعة dropout المختلفة، فقد نقول إن الشبكة أكثر ثقة.

5.6.2 التنفيذ من البداية

لتنفيذ دالة dropout لطبقة واحدة، يجب علينا سحب أكبر عدد ممكن من العينات من متغير برنولي العشوائي (ثنائي) حيث تحتوي الطبقة الخاصة بنا على أبعاد، حيث يأخذ المتغير العشوائي قيمة 1 (احفظ بها) مع الاحتمال $p - 1$ و 0 (إسقاط dropout) مع الاحتمال p . تمثل إحدى الطرق السهلة لتنفيذ ذلك في سحب عينات أولًا من التوزيع المستنبط $[0,1]^U$. ثم يمكننا الاحفاظ بتلك العقد التي تكون العينة المقابلة لها أكبر من p ، وإسقاط الباقي.

في الكود التالي، نقوم بتنفيذ دالة `dropout_layer` التي تسقط العناصر في إدخال الموتر X مع احتمال `dropout`، مع إعادة قياس الباقي كما هو موضح أعلاه: قسمة الناجين على $1.0 - \text{dropout}$.

```
import tensorflow as tf
from d2l import tensorflow as d2l
```

```
def dropout_layer(X, dropout):
    assert 0 <= dropout <= 1
```

```

if dropout == 1: return tf.zeros_like(X)
mask = tf.random.uniform(
    shape=tf.shape(X), minval=0, maxval=1) < 1 -
dropout
return tf.cast(mask, dtype=tf.float32) * X / (1.0 -
dropout)

```

يمكنا اختبار دالة `dropout_layer` على بعض الأمثلة. في سطور الكود التالية، نقوم بتمرير الإدخال `X` الخاص بنا من خلال عملية التسرب `dropout`، مع الاحتمالات 0 و 0.5 و 1 على التوالي.

```

X = tf.reshape(tf.range(16, dtype=tf.float32), (2, 8))
print('dropout_p = 0:', dropout_layer(X, 0))
print('dropout_p = 0.5:', dropout_layer(X, 0.5))
print('dropout_p = 1:', dropout_layer(X, 1))
dropout_p = 0: tf.Tensor(
[[ 0.  1.  2.  3.  4.  5.  6.  7.]
 [ 8.  9. 10. 11. 12. 13. 14. 15.]], shape=(2, 8),
dtype=float32)
dropout_p = 0.5: tf.Tensor(
[[ 0.  0.  6.  0.  0. 12.  0.]
 [16. 18. 20.  0.  0.  0.  0.]], shape=(2, 8),
dtype=float32)
dropout_p = 1: tf.Tensor(
[[0. 0. 0. 0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0. 0. 0. 0.]], shape=(2, 8),
dtype=float32)

```

5.6.2.1 Defining the Model

يطبق النموذج أدناه `dropout` على إخراج كل طبقة مخفية (باتباع دالة التنشيط). يمكننا تعين احتمالات `dropout` لكل طبقة على حدة. يتمثل الاتجاه الشائع في تعين احتمالية `dropout` فقط أثناء التدريب. أقل بالقرب من طبقة الإدخال. نحن نضمن أن `dropout` نشط فقط أثناء التدريب.

```

class DropoutMLPScratch(d2l.Classifier):
    def __init__(self, num_outputs, num_hiddens_1,
num_hiddens_2,
                    dropout_1, dropout_2, lr):
        super().__init__()
        self.save_hyperparameters()
        self.lin1 = tf.keras.layers.Dense(num_hiddens_1,
activation='relu')

```

```

    self.lin2 = tf.keras.layers.Dense(num_hiddens_2,
activation='relu')
    self.lin3 = tf.keras.layers.Dense(num_outputs)

def forward(self, X):
    H1 = self.lin1(tf.reshape(X, (X.shape[0], -1)))
    if self.training:
        H1 = dropout_layer(H1, self.dropout_1)
    H2 = self.lin2(H1)
    if self.training:
        H2 = dropout_layer(H2, self.dropout_2)
    return self.lin3(H2)

```

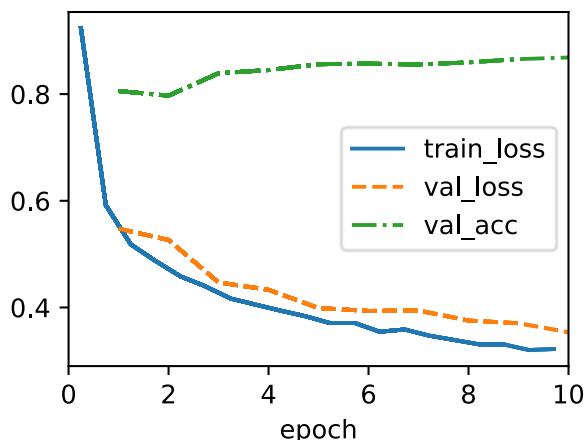
5.6.2.2 Training التدريب

ما يلي مشابه لتدريب MLPs الموصوف سابقاً.

```

hparams = {'num_outputs':10, 'num_hiddens_1':256,
'num_hiddens_2':256,
'dropout_1':0.5, 'dropout_2':0.5, 'lr':0.1}
model = DropoutMLPScratch(**hparams)
data = d2l.FashionMNIST(batch_size=256)
trainer = d2l.Trainer(max_epochs=10)
trainer.fit(model, data)

```



5.6.3 Concise Implementation المختصر

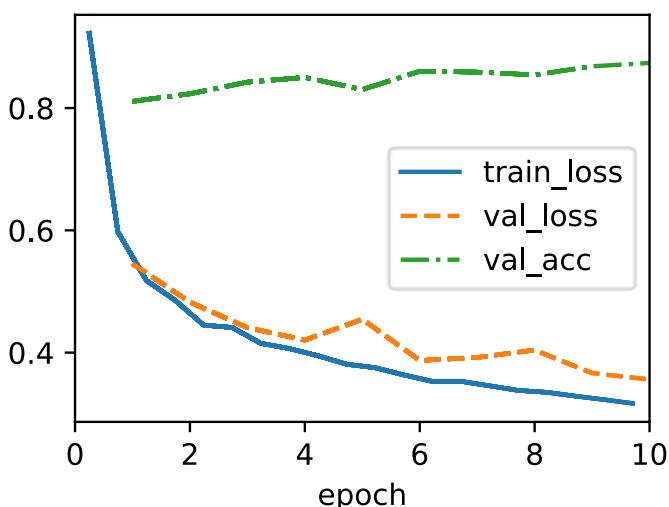
باستخدام واجهات برمجة التطبيقات API عالية المستوى، كل ما نحتاج إلى القيام به هو إضافة طبقة Dropout بعد كل طبقة متصلة بالكامل، لتمرير احتمال Dropout باعتباره الوسيطة الوحيدة لمنشئها. أثناء التدريب، ستسقط طبقة Dropout بشكل عشوائي مخرجات الطبقة السابقة (أو بشكل مكافئ، المدخلات إلى الطبقة اللاحقة) وفقاً لاحتمال Dropout المحدد.

عندما لا تكون في وضع التدريب، تقوم طبقة Dropout ببساطة بتمرير البيانات من خلالها أثناء الاختبار.

```
class DropoutMLP(d2l.Classifier):
    def __init__(self, num_outputs, num_hiddens_1,
num_hiddens_2,
                    dropout_1, dropout_2, lr):
        super().__init__()
        self.save_hyperparameters()
        self.net = tf.keras.models.Sequential([
            tf.keras.layers.Flatten(),
            tf.keras.layers.Dense(num_hiddens_1,
activation=tf.nn.relu),
            tf.keras.layers.Dropout(dropout_1),
            tf.keras.layers.Dense(num_hiddens_2,
activation=tf.nn.relu),
            tf.keras.layers.Dropout(dropout_2),
            tf.keras.layers.Dense(num_outputs)])
```

بعد ذلك، نقوم بتدريب النموذج.

```
model = DropoutMLP(**hparams)
trainer.fit(model, data)
```



5.6.4 الملخص

- إلى جانب التحكم في عدد الأبعاد وحجم متجه الوزن weight vector، فإن Dropout هو أداة أخرى لتجنب الضبط الزائد overfitting. غالباً ما يتم استخدامها بشكل مشترك.
- يستبدل Dropout التنشيط h بمتغير عشوائي ذي قيمة متوقعة h .
- يستخدم Dropout فقط أثناء التدريب.

5.6.5 التمارين

- ماذا يحدث إذا قمت بتغيير احتمالات Dropout للطبقتين الأولى والثانية؟ على وجه الخصوص، ماذا يحدث إذا قمت بتبديل الطبقات لكلا الطبقتين؟ صمم تجربة للإجابة على هذه الأسئلة، ووصف نتائجك من الناحية الكمية، ولخص النتائج النوعية.
- قم بزيادة عدد الفترات number of epochs ومقارنة النتائج التي تم الحصول عليها عند استخدام Dropout مع تلك التي تم الحصول عليها عند عدم استخدامها.
- ما هو تباين التنشيطات في كل طبقة مخفية عندما يتم تطبيق Dropout ولا يتم تطبيقه؟ ارسم مخططًا لإظهار كيفية تطور هذه الكمية بمرور الوقت لكلا النموذجين.
- لماذا لا يتم استخدام Dropout عادةً في وقت الاختبار test time؟
- باستخدام النموذج في هذا القسم كمثال، قارن بين تأثيرات استخدام Dropout وتناقص الوزن weight decay. ماذا يحدث عند استخدام Dropout وتتناقص الوزن في نفس الوقت؟ هل النتائج مضافة؟ هل هناك عوائد متناقصة (أو أسوأ)؟ هل يلغون بعضهم البعض؟
- ماذا يحدث إذا طبقنا Dropout على الأوزان الفردية لمصفوفة الوزن بدلاً من التنشيطات؟
- اختر تكنية أخرى لحقن ضوضاء عشوائية في كل طبقة تختلف عن تكنية Dropout القياسية. هل يمكنك تطوير طريقة تتفوق في الأداء على مجموعة بيانات Fashion-MNIST (للهندسة المعمارية الثابتة)؟

5.7 توقع أسعار المنازل في Kaggle

الآن بعد أن قدمنا بعض الأدوات الأساسية لبناء وتدريب شبكات عميقة وتنظيمها باستخدام تقنيات بما في ذلك تناقص الوزن dropout، نحن على استعداد لوضع كل هذه المعرفة موضع التنفيذ من خلال المشاركة في مسابقة Kaggle . تعتبر مسابقة التنبؤ بأسعار المنزل مكاناً رائعاً للبدء. البيانات عامة إلى حد ما ولا تظهر بنية غريبة قد تتطلب نماذج متخصصة (مثل الصوت

أو الفيديو). مجموعة البيانات هذه، التي جمعها Bart de Cock في عام 2011، (De Cock، 2011)، تغطي أسعار المنازل في Ames، IA من الفترة 2006–2010. إنه أكبر بكثير من مجموعة بيانات الإسكان الشهير في بوسطن Boston housing dataset لهاريسون وروبنفيلد (1978)، ويضم المزيد من الأمثلة والمزيد من الميزات.

في هذا القسم، سترشدك إلى تفاصيل المعالجة المسبقة للبيانات وتصميم النموذج واختيار المعلمة الفائقة. نأمل أن تكتسب من خلال نهج عملٍ بعض البديهيات التي ستوجهك في حياتك المهنية كعالم بيانات.

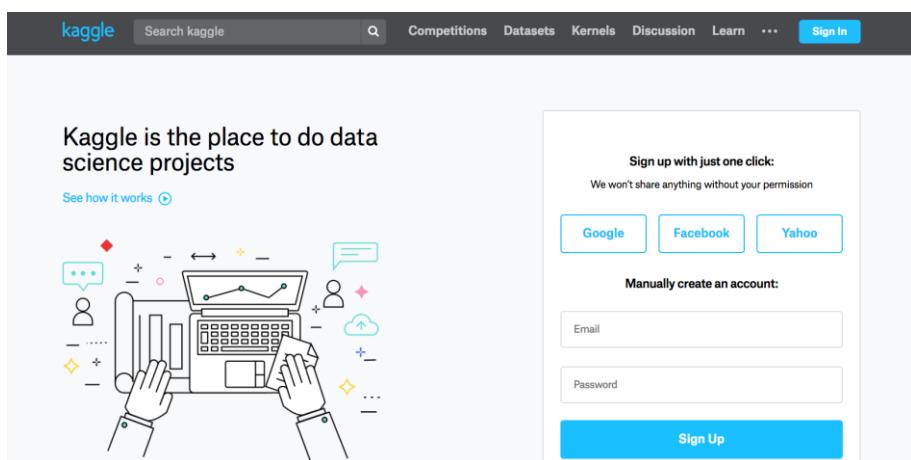
5.7.1 تنزيل البيانات

خلال الكتاب، سنقوم بتدريب واختبار النماذج على مجموعات البيانات المختلفة التي تم تنزيلها. هنا، نقوم بتنفيذ دالتين مفیدتين لتنزيل الملفات واستخراج ملفات zip أو tar. مرة أخرى، نرجئ تفزيدها إلى القسم 20.7.

```
def download(url, folder, sha1_hash=None):
    """Download a file to folder and return the Local
    filepath."""

def extract(filename, folder):
    """Extract a zip/tar file into folder."""

```



الشكل 5.7.1 موقع Kaggle الإلكتروني.

Kaggle 5.7.2

Kaggle هي منصة شهيرة تستضيف مسابقات التعلم الآلي. تركز كل مسابقة على مجموعة بيانات والعديد منها برعاية أصحاب المصلحة الذين يقدمون جوائز للحلول الفائزة. تساعد المنصة المستخدمين على التفاعل عبر المنتديات والاكواود المشتركة، مما يعزز التعاون والمنافسة. في حين أن مطارة المتتصرين غالباً ما تخرج عن نطاق السيطرة، مع تركيز الباحثين على خطوات المعالجة المسبقة بدلاً من طرح الأسئلة الأساسية، هناك أيضاً قيمة هائلة في موضوعية النظام الأساسي الذي يسهل المقارنات الكمية المباشرة بين الأساليب المتنافسة بالإضافة إلى مشاركة الكود بحيث يمكن للجميع أن يتلعلموا ما نجح وما لم ينجح. إذا كنت ترغب في المشاركة في مسابقة Kaggle، فستحتاج أولاً إلى التسجيل للحصول على حساب (انظر الشكل 5.7.1).

في صفحة مسابقة التنبؤ بسعر المنزل، كما هو موضح في الشكل 5.7.2، يمكنك العثور على مجموعة البيانات (ضمن علامة التبويب "بيانات")، وإرسال التنبؤات، والاطلاع على الترتيب الخاص بك، عنوان URL موجود هنا:

<https://www.kaggle.com/c/house-prices-advanced-regression-techniques>

شكل 5.7.2 صفحة مسابقة التنبؤ بسعر المنزل.

5.7.3 Accessing and Reading the Dataset

لاحظ أن بيانات المسابقة مقسمة إلى مجموعات تدريب training وختبار test. يتضمن كل سجل قيمة ممتلكات المنزل والسمات مثل نوع الشارع street type وسنة البناء year of

و نوع السقف construction و حالة الطابق السفلي basement condition إلى ذلك. تتكون الميزات من أنواع بيانات مختلفة. على سبيل المثال، يتم تمثيل سنة البناء بعدد صحيح، و نوع السقف بالخصائص الفئوية المنفصلة، والميزات الأخرى بأرقام الفاصلة العائمة. وهنا حيث يعقد الواقع الأشياء: بالنسبة لبعض الأمثلة، بعض البيانات مفقودة تماماً مع وضع علامة على القيمة المفقودة ببساطة على أنها "na". سعر كل منزل مشمول في مجموعة التدريب فقط (إنها منافسة بعد كل شيء). سنرغب في تقسيم مجموعة التدريب لإنشاء مجموعة التتحقق من الصحة validation set، لكننا فقط نقوم بتقييم نماذجنا على مجموعة الاختبار الرسمية بعد تحميل التنبؤات إلى Kaggle. تحتوي علامة التبويب "البيانات" في علامة تبويب المنافسة في الشكل 5.7.2 على روابط لتنزيل البيانات.

```
%matplotlib inline
import numpy as np
import pandas as pd
import tensorflow as tf
from d2l import tensorflow as d2l
```

للبدء، سنقرأ البيانات ونعالجها باستخدام pandas، والتي قدمناه في القسم 2.2. للسهولة، يمكننا تنزيل مجموعة بيانات الإسكان Kaggle وتخزينها مؤقتاً. إذا كان الملف المطابق لمجموعة البيانات هذه موجوداً بالفعل في دليل ذاكرة التخزين المؤقت وكان SHA-1 يتطابق مع sha1_hash، فسيستخدم الكود الخاص بنا الملف المخزن مؤقتاً لتجنب انسداد الانترنت بالتنزيلات الزائدة عن الحاجة.

```
class KaggleHouse(d2l.DataModule):
    def __init__(self, batch_size, train=None,
val=None):
        super().__init__()
        self.save_hyperparameters()
        if self.train is None:
            self.raw_train = pd.read_csv(d2l.download(
                d2l.DATA_URL +
                'kaggle_house_pred_train.csv', self.root,
                sha1_hash='585e9cc93e70b39160e7921475f9bcd7d31219ce'))
            self.raw_val = pd.read_csv(d2l.download(
                d2l.DATA_URL +
                'kaggle_house_pred_test.csv', self.root,
                sha1_hash='fa19780a7b011d9b009e8bff8e99922a8ee2eb90'))
```

تتضمن مجموعة بيانات التدريب 1460 مثلاً و80 ميزة، بينما تحتوي بيانات التحقق من الصحة على 1459 مثلاً و80 ميزة.

```
data = KaggleHouse(batch_size=64)
print(data.raw_train.shape)
print(data.raw_val.shape)
Downloading ../data/kaggle_house_pred_train.csv from
http://d2l-data.s3-
accelerate.amazonaws.com/kaggle_house_pred_train.csv...
Downloading ../data/kaggle_house_pred_test.csv from
http://d2l-data.s3-
accelerate.amazonaws.com/kaggle_house_pred_test.csv...
(1460, 81)
(1459, 80)
```

5.7.4 المعالجة المسبقة للبيانات Data Preprocessing

دعنا نلقي نظرة على الميزات الأربع الأولى والأخيرة بالإضافة إلى التصنيف (سعر البيع SalePrice) من الأمثلة الأربع الأولى.

```
print(data.raw_train.iloc[:4, [0, 1, 2, 3, -3, -2, -1]])
```

	Id	MSSubClass	MSZoning	LotFrontage	SaleType	SaleCondition	SalePrice
0	1	60	RL	65.0	WD	Normal	208500
1	2	20	RL	80.0	WD	Normal	181500
2	3	60	RL	68.0	WD	Normal	223500
3	4	70	RL	60.0	WD	Abnorml	140000

يمكنا أن نرى أنه في كل مثال، الميزة الأولى هي المعرف ID. هذا يساعد النموذج على تحديد كل مثال تدريب. في حين أن هذا مناسب، إلا أنه لا يحمل أي معلومات لأغراض التنبؤ. ومن ثم، سنقوم بإزالته من مجموعة البيانات قبل إدخال البيانات في النموذج. إلى جانب ذلك، بالنظر إلى مجموعة متنوعة من أنواع البيانات، سنحتاج إلى معالجة البيانات مسبقاً قبل أن نبدأ في التنبؤ .modeling

لنبدأ بالسمات العددية numerical features. أولاً، نطبق الاستدلال heuristic، مع استبدال جميع القيم المفقودة missing values بمتوسط الميزة المقابلة. بعد ذلك، لوضع جميع الميزات على مقاييس مشتركة، نقوم بتوحيد البيانات عن طريق إعادة قياس الميزات إلى صفر متوسط variance وتبالن الوحدة unit mean:

$$x \leftarrow \frac{x - \mu}{\sigma},$$

حيث μ و σ تشير إلى المتوسط والانحراف المعياري، على التوالي. للتحقق من أن هذا يحول بالفعل ميزتنا (المتغير) بحيث لا يحتوي متوسط صفر وتبين وحدة، لاحظ ذلك $= E[\frac{x-\mu}{\sigma}] = \frac{\mu-\mu}{\sigma} = 0$ وذلك $\sigma^2 = E[(x - \mu)^2] = (\sigma^2 + \mu^2) - 2\mu^2 + \mu^2 = 0$. بشكل حديسي، نقوم بتوحيد البيانات لسبعين. أولاً، ثبت أنه مناسب للتحسين. ثانياً، نظرًا لأننا لا نعرف مسبقًا أي الميزات ستكون ذات صلة، لا نريد معاقبة المعاملات المخصصة لميزة واحدة أكثر من أي ميزة أخرى.

بعد ذلك نتعامل مع القيم المتقطعة discrete values. يتضمن ذلك ميزات مثل "MSZoning". نقوم باستبدالها بترميز واحد ساخن one-hot encoding بنفس الطريقة التي قمنا بها سابقًا بتحويل التسميات متعددة الفئات إلى متجهات (انظر القسم 4.1.1). على سبيل المثال، تفترض "MSZoning" القيمتين "RL" و "RM". بإسقاط ميزة "MSZoning" ، يتم إنشاء ميزتين جديدين للمؤشر "MSZoning_RL" و "MSZoning_RM" بقيم إما 0 أو 1. وفقًا للترميز الواحد الساخن، إذا كانت القيمة الأصلية "MSZoning_RM" تساوي 1 و "MSZoning_RL" هي "RM" ، القيمة "MSZoning_RL" تساوي 0. حزمة pandas تقوم بذلك تلقائيًا لنا.

```
@d21.add_to_class(KaggleHouse)
def preprocess(self):
    # Remove the ID and Label columns
    label = 'SalePrice'
    features = pd.concat(
        (self.raw_train.drop(columns=['Id', label]),
         self.raw_val.drop(columns=['Id'])))
    # Standardize numerical columns
    numeric_features = features.dtypes[features.dtypes != 'object'].index
    features[numeric_features] =
    features[numeric_features].apply(
        lambda x: (x - x.mean()) / (x.std()))
    # Replace NAN numerical features by 0
    features[numeric_features] =
    features[numeric_features].fillna(0)
    # Replace discrete features by one-hot encoding.
    features = pd.get_dummies(features, dummy_na=True)
    # Save preprocessed features
    self.train =
    features[:self.raw_train.shape[0]].copy()
    self.train[label] = self.raw_train[label]
```

`self.val = features[self.raw_train.shape[0]:].copy()`
يمكنك أن ترى أن هذا التحويل يزيد عدد الميزات من 79 إلى 331 (باستثناء أعمدة المعرف والتسمية ID).

```
data.preprocess()
data.train.shape
(1460, 332)
```

5.7.5. قياس الخطأ Error Measure

للبدء، سنقوم بتدريب نموذج خطى بخسارة مربعة squared loss. ليس من المستغرب أن نموذجنا الخطى لن يؤدى إلى إرسال فائز بالمنافسة ولكنه يوفر فحصاً للعقل لمعرفة ما إذا كانت هناك معلومات مفيدة في البيانات. إذا لم نتمكن من القيام بما هو أفضل من التخمين العشوائى هنا، فقد تكون هناك فرصة جيدة لوجود خطأ معالجة البيانات. وإذا نجحت الأشياء، فسيكون النموذج الخطى بمثابة خط أساس يمنحك بعض الحدس حول مدى قرب النموذج البسيط من أفضل النماذج المبلغ عنها، مما يمنحك إحساساً بمدى المكاسب التي يجب أن تتوقعها من النماذج الأكثر رواجاً.

مع أسعار المساكن، كما هو الحال مع أسعار الأسهم، نهتم بالكميات النسبية أكثر من الكميات المطلقة. وبالتالي فإننا نميل إلى الاهتمام أكثر بالخطأ النسبي $\frac{\hat{y} - y}{y}$ من الخطأ المطلق $\hat{y} - y$. على سبيل المثال، إذا تم إيقاف توقعنا بمقدار 100,000 دولار أمريكي عند تقدير سعر منزل في ريف أوهايو، حيث تبلغ قيمة المنزل النموذجي 125,000 دولار أمريكي، فمن المحتمل أننا نقوم بعمل رهيب. من ناحية أخرى، إذا أخطأنا بهذا المبلغ في لوس أنطوس هيلز، كاليفورنيا، فقد يمثل هذا تنبؤاً دقيقاً بشكل مذهل (هناك، يتجاوز متوسط سعر المنزل 4 ملايين دولار أمريكي).

تتمثل إحدى طرق معالجة هذه المشكلة في قياس التناقض في لوغاريتmic تقييمات الأسعار في الواقع. يعد هذا أيضاً مقياس الخطأ الرسمي الذي تستخدمه المسابقة لتقييم جودة الطلبات المقدمة.

بعد كل شيء، قيمة صغيرة $\delta \leq \log y - \log \hat{y}$ تترجم إلى $e^\delta \leq \frac{\hat{y}}{y}$. يؤدي هذا إلى الخطأ الجذر التربيعي root-mean-squared-error التالي بين لوغاريتmic السعر المتوقع predicted price ولوغاريتmic سعر التسمية label price:

$$\sqrt{\frac{1}{n} \sum_{i=1}^n (\log y_i - \log \hat{y}_i)^2}.$$

```
@d2l.add_to_class(KaggleHouse)
def get_dataloader(self, train):
```

```

label = 'SalePrice'
data = self.train if train else self.val
if label not in data: return
get_tensor = lambda x: tf.constant(x.values,
dtype=tf.float32)
# Logarithm of prices
tensors = (get_tensor(data.drop(columns=[label])), 
# X

tf.reshape(tf.math.log(get_tensor(data[label])), (-1, 
1))) # Y
return self.get_tensorloader(tensors, train)

```

K-Fold Cross Validation .5.7.6

قد تذكر أنتا قدمنا التحقق المتبادل في القسم 3.6.3، حيث ناقشنا كيفية التعامل مع اختيار النموذج. سنسخدم هذا بشكل جيد لتحديد تصميم النموذج وضبط المعلمات الفائقة. نحتاج أولاً إلى دالة تُرجع i^{th} طي البيانات في إجراء تحقق متقطع. يتم المضي قدمًا عن طريق تقطيع المقطع كبيانات تحقق وإعادة البالى كبيانات تدريب. لاحظ أن هذه ليست الطريقة الأكثر فاعلية للتعامل مع البيانات وسنفعل بالتأكيد شيئاً أكثر ذكاءً إذا كانت مجموعة البيانات الخاصة بنا أكبر بكثير. لكن هذا التعقيد الإضافي قد يؤدي إلى تشويش الكود الخاص بنا دون داع لذلك يمكننا حذفها بأمان هنا بسبب بساطة مشكلتنا.

```

def k_fold_data(data, k):
    rets = []
    fold_size = data.train.shape[0] // k
    for j in range(k):
        idx = range(j * fold_size, (j+1) * fold_size)
        rets.append(KaggleHouse(data.batch_size,
data.train.drop(index=idx),
data.train.loc[idx]))
    return rets

```

يتم إرجاع متوسط خطأ التتحقق من الصحة عندما نتدرب K مرات في التتحقق المتبادل K-Fold.

```

def k_fold(trainer, data, k, lr):
    val_loss, models = [], []
    for i, data_fold in enumerate(k_fold_data(data, k)):
        model = d2l.LinearRegression(lr)
        model.board.yscale='log'
        if i != 0: model.board.display = False
        trainer.fit(model, data_fold)

```

```

val_loss.append(float(model.board.data['val_loss'][-
1].y))
models.append(model)
print(f'average validation log mse =
{sum(val_loss)/len(val_loss)}')
return models

```

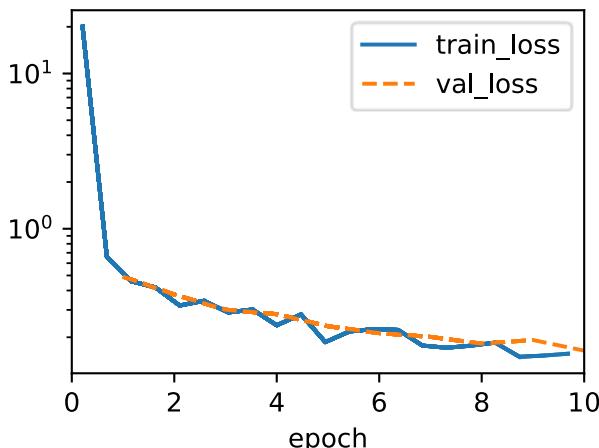
5.7.7 اختيار النموذج

في هذا المثال، نختار مجموعة غير مضبوطة من المعلمات الفائقة ونتركها للقارئ لتحسين النموذج. قد يستغرق العثور على خيار جيد وقتاً، اعتماداً على عدد المتغيرات التي يحسنها الممرء. مع وجود مجموعة بيانات كبيرة بما يكفي، والأنواع العادية من المعلمات الفائقة، يميل التحقق المتبادل K-Fold إلى أن يكون مناً بشكل معقول ضد الاختبارات المتعددة. ومع ذلك، إذا جربنا عدداً كبيراً بشكل غير معقول من الخيارات، فقد نحالينا الحظ ونجد أن أداء التتحقق من الصحة لم يعد يمثل الخطأ الحقيقي.

```

trainer = d2l.Trainer(max_epochs=10)
models = k_fold(trainer, data, k=5, lr=0.01)
average validation log mse = 0.1782047653198242

```

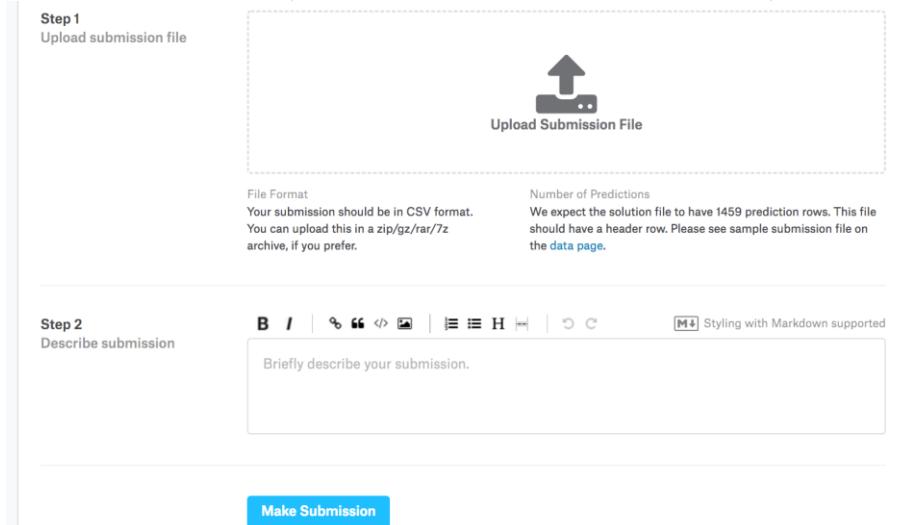


لاحظ أنه في بعض الأحيان يمكن أن يكون عدد أخطاء التدريب لمجموعة من المعلمات الفائقة منخفضاً جداً، حتى مع ارتفاع عدد الأخطاء في التتحقق المتبادل K-Fold بشكل كبير. هذا يشير إلى أنها نفرط في التجهيز overfitting. خلال التدريب، سترغب في مراقبة كل الرقمان. قد يشير التجاوز الأقل إلى أن بياناتنا يمكن أن تدعمنا نموذجاً أكثر قوة. قد يوحي فرط التجهيز الهائل بأنه يمكننا الاستفادة من خلال دمج تقنيات التنظيم regularization.

5.7.8 تقديم التنبؤات على Kaggle

الآن بعد أن عرفنا ما يجب أن يكون عليه الاختيار الجيد للمعلمات الفائقة، يمكننا حساب متوسط التنبؤات في الاختبار الذي تم تعينه بواسطة جميع النماذج. سيؤدي حفظ التنبؤات في ملف csv إلى تبسيط تحميل النتائج إلى Kaggle. سيسعى الكود التالي ملفاً يسمى `submission.csv`.

```
preds = [model(tf.constant(data.val.values,
                           dtype=tf.float32))
         for model in models]
# Taking exponentiation of predictions in the Logarithm
scale
ensemble_preds = tf.reduce_mean(tf.exp(tf.concat(preds,
                                             1)), 1)
submission = pd.DataFrame({'Id':data.raw_val.Id,
                           'SalePrice':ensemble_preds.numpy()})
submission.to_csv('submission.csv', index=False)
```



الشكل 5.7.3 تقديم البيانات إلى Kaggle

بعد ذلك، كما هو موضح في الشكل 5.7.3، يمكننا تقديم تنبؤاتنا على Kaggle ومعرفة كيفية مقارنتها بأسعار المنازل الفعلية (التسميات labels) في مجموعة الاختبار. الخطوات بسيطة للغاية:

- قم بتسجيل الدخول إلى موقع Kaggle الإلكتروني وقم بزيارة صفحة مسابقة التنبؤ بأسعار المنازل.

- انقر فوق الزر "إرسال التوقعات Submit Predictions" أو "التقديم المتأخر" (حتى كتابة هذه السطور، يوجد الزر على اليمين).
- انقر فوق الزر "تحميل ملف التقديم Upload Submission File" في المربع المتقطع أسفل الصفحة وحدد ملف التنبؤ الذي ترغب في تحميله.
- انقر فوق الزر "تقديم Make Submission" في أسفل الصفحة لعرض النتائج الخاصة بك.

5.7.9 الملخص

- غالباً ما تحتوي البيانات الحقيقية على مزيج من أنواع البيانات المختلفة وتحتاج إلى معالجة مسبقة.
- إعادة قياس Rescaling البيانات ذات القيمة الحقيقية إلى متوسط الصفر وتبالن الوحيدة يعد افتراضياً جيداً. لذلك يتم استبدال القيم المفقودة بمتوسطها.
- يتيح لنا تحويل الميزات الفئوية categorical features إلى ميزات مؤشر أن نتعامل معها على أنها متوجهات واحدة ساخنة.
- يمكننا استخدام التحقق المتبادل K-Fold لتحديد النموذج وضبط المعلمات الفائقة.
- اللوغاريتمات مفيدة للأخطاء النسبية.

5.7.10 التمارين

1. أرسل توقعاتك لهذا القسم إلى Kaggle. ما مدى جودة توقعاتك؟
2. هل من الجيد دائمًا استبدال القيم المفقودة بمتوسطها؟ تلميح: هل يمكنك بناء موقف لا تكون فيه القيم مفقودة بشكل عشوائي؟
3. قم بتحسين النتيجة على Kaggle عن طريق ضبط المعلمات الفائقة من خلال التحقق المتبادل K-Fold.
4. قم بتحسين النتيجة من خلال تحسين النموذج (على سبيل المثال، الطبقات وتناقص الوزن dropout).
5. ماذا يحدث إذا لم نقم بتوحيد standardize السمات العددية المستمرة مثل ما فعلناه في هذا القسم؟

Dive into Deep Learning

Basics and Preliminaries

**ASTON ZHANG, ZACHARY C. LIPTON, MU LI,
AND ALEXANDER J. SMOLA**

**Translated Into Arabic by
Dr. Alaa Taima**