Základní praktiky strojového učení

Pokyny k online schůzce

1. Používejte nejlépe desktopovou aplikaci Teams.

2. Pokud nekladete dotaz nebo se neúčastníte diskuze, mějte prosím vypnutý mikrofon.

3. Pokud jen trochu můžete, mějte puštěnou kameru. Vás výraz tváře pomůže vyučujícímu :)

4. Jak můžete položit dotaz:
   a. Zapněte si mikrofon a rovnou se zeptejte.
      nebo:
   b. Napište dotaz do chatu. nebo:
   c. Použijte tlačítko zvednout ruku.
      (Po vyvolání ruku sundejte)

Agenda

- pracovní postup projektu strojového učení
- definice problému
- zdroje datasetů
- prozkoumání a vizualizace dat
- příprava dat pro algoritmy ML
- výběr a trénování modelu
- vyladění modelu
- prezentace výsledků

Pracovní postup projektu strojového učení

1. Definice problému
2. Získání datasetu
3. Výběr metriky úspěchu a stanovení způsobu evaluace
4. Vizualizace a prozkoumání datasetu
5. Příprava dat pro algoritmy strojového učení
6. Výběr a trénování modelu
7. Vyladění modelu
8. Prezentace výsledků
9. Spuštění, monitorování a údržba systému

Kde vzít datasety?

- populární repozitáře datasetů
  - Kaggle
  - UC Irvine Machine Learning Repository
  - Open Data AWS datasets
  - Datahub.io
- meta portály (katalogy repozitářů datasetů)
  - Dataportals.org
  - OpenDataMonitor.eu
  - Quandl.com
- ostatní zdroje
  - seznam ML datasetů na wikipedii
  - diskuze o datasetech na Quora.com
  - subreddit o datasetech
  - sklearn.datasets, keras.datasets
  - wikipedia jako celek
  - Národní katalog otevřených dat (NKOD)
  - Common Crawl

Definice problému

- jaká budou vstupní data?
  - co se snažíme predikovat?
    - naučit se něco predikovat lze jen tehdy, pokud máme trénovací data
  - dostupnost dat bývá omezujícím faktorem
- jaký typ problému řešíme?
  - binární klasifikace, regrese, klasifikace do více tříd s více označenými třídami, shlukování, generování, ...
  - identifikace typu problému vede k výběru architektury modelu, ztrátové funkce atd.
- hypotézy, na kterých stavíme
  - výstupy lze predikovat ze vstupů
  - dostupná data jsou dostatečně informativní, aby z nich bylo možné získat vztahy mezi vstupy a výstupy

Definice problému: Predikce cen nemovitostí

- cíl: predikovat cenu nemovitosti v Kalifornii na základě jejích vlastností
  - využije se jako vstup při rozhodování v investiční společnosti
- vstupní data
  - California Housing Prices ze StatLib repozitáře
    - data nejsou za jednotlivé nemovitosti, ale za okrsky (600-3000 obyvatel)
- typ problému
  - supervised (známe cílové hodnoty - průměrné ceny nemovitostí v okrscích)
  - regrese - chceme predikovat číselnou hodnotu (cenu)
    - vícenásobná regrese - máme k dispozici řadu příznaků (features), ze kterých budeme predikovat výsledek
    - jednorozměrná regrese - chceme predikovat jednu hodnotu
  - batch learning - máme celý dataset najednou a nepotřebujeme řešit rychlé aktualizace

Výběr metriky úspěchu

- chceme-li něco řídit, musíme to sledovat
- metrika by měla být sladěna s cíli vyšší úrovně v rámci celého businessu

- možné metriky
  - problémy s vyváženými třídami - správnost (accuracy) a oblast pod křivkou ROC (ROC AUC)
  - problémy s nevyváženými třídami - přesnost (precision) a úplnost (recall)
  - uspořádávací (ranking) problémy, klasifikace s více označeními tříd - průměrná správnost
  - regresní úlohy - RMSE (Root Mean Square Error) = odmocnina ze střední kvadratické chyby (MSE)
  - vlastní metriky

Výběr metriky úspěchu: Predikce cen nemovitostí

- zvolíme RMSE
- říká nám, jak moc velké chyby se systém typicky při predických dopouští
  - velké chyby mají větší význam

$$\mathrm{RMSE}(\mathbf{X}, h) = \sqrt{\frac{1}{m}\sum_{i=1}^{m}\left(h\left(\mathbf{x}^{(i)}\right) - y^{(i)}\right)^2}$$

- *m* = počet instancí v datasetu, např. 2000
- $\mathbf{x}^{(i)}$ = vektor příznaků (bez labelu) pro i-tou instanci, např.: $\mathbf{x}^{(1)} = \begin{pmatrix} -118.29 \\ 33.91 \\ 1,416 \\ 38,372 \end{pmatrix}$
- $y^{(i)}$ = label (cílová hodnota) i-té instance, např. $y^{(1)} = 156400$

Výběr míry úspěchu: Predikce cen nemovitostí

- **X** = matice příznaků všech instancí (příkladů) v datasetu
  - každá instance má svůj řádek
  - každý příznak má svůj sloupec
- *h* = predikční funkce (hypotéza)
  - $\hat{y}^{(i)} = h(\mathbf{x}^{(i)})$

$$\mathbf{X} = \begin{pmatrix} \left(\mathbf{x}^{(1)}\right)^T \\ \left(\mathbf{x}^{(2)}\right)^T \\ \vdots \\ \left(\mathbf{x}^{(1999)}\right)^T \\ \left(\mathbf{x}^{(2000)}\right)^T \end{pmatrix} = \begin{pmatrix} -118.29 & 33.91 & 1,416 & 38,372 \\ \vdots & \vdots & \vdots & \vdots \end{pmatrix}$$

- místo RMSE můžeme alternativně použít MAE (Mean Absolute Error)
  - vhodné pokud je v datasetu hodně odlehlých hodnot, které by příliš zvyšovaly RMSE

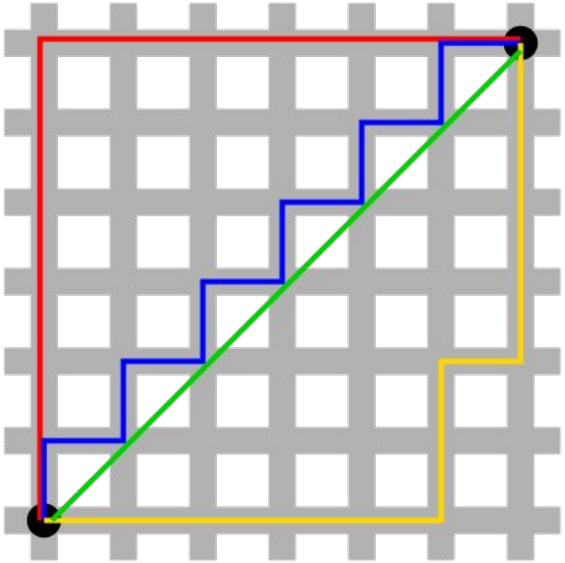
$$\mathrm{MAE}(\mathbf{X}, h) = \frac{1}{m} \sum_{i=1}^{m} \left| h\left(\mathbf{x}^{(i)}\right) - y^{(i)} \right|$$

Určení vzdálenosti mezi dvěma vektory

- RMSE i MAE v podstatě měří vzdálenost dvou vektorů
  - vektor predikcí
  - vektor cílových hodnot

- RMSE (odmocnina ze součtu čtverců) = Euklidovská norma, L_2
- MAE (součet absolutních hodnot) = Manhattanská norma (metrika), L_1

- čím vyšší index normy, tím více se metrika zaměřuje na velké hodnoty a zanedbává malé
  - RMSE je tedy citlivější na odlehlé hodnoty než MAE

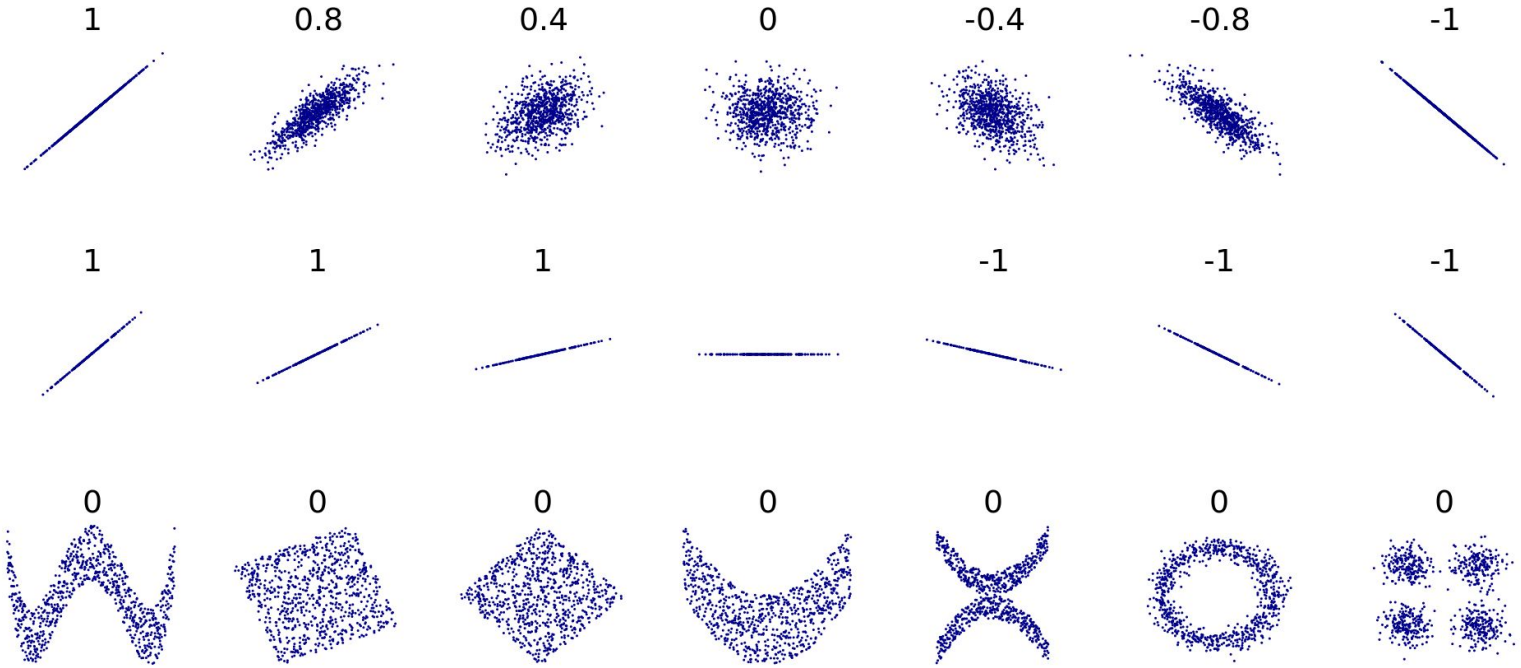


Zdroj: https://cs.wikipedia.org/wiki/Manhattansk%C3%A1_metrika

Vizualizace a prozkoumání datasetu

- viz Jupyter Notebook
- datasets/02/housing.csv
- každý řádek jeden okrsek
- atributy (příznaky): zem. délka, zem. šířka, průměrné stáří, celkový počet místností, celkový počet pokojů, počet obyvatel, počet domácností, průměrný příjem, průměrná cena nemovitosti, vzdálenost od oceánu

Korelační koeficient



Zdroj: https://en.wikipedia.org/wiki/Correlation_and_dependence

Příprava dat pro algoritmy strojového učení

- viz Jupyter Notebook

Převod textových a kategoriálních atributů na čísla

- varianta A: OrdinalEncoder
  - převedeme na čísla pomocí OrdinalEncoder. Každá kategorie má pak své číslo (0, 1, 2, ...).
  - nevýhodou je, že algoritmy ML pak budou předpokládat, že kategorie, které k sobě mají blízko číselně, mají k sobě blízko i v reálu.
  - někdy to lze použít (např. hodnocení “špatný”, “průměrný”, “dobrý”, “výborný”)
    - v našem případě to však smysl nedává
- varianta B: One-Hot Encoding
  - binární atribut pro každou kategorii
  - v našem případě máme 5 kategorií, takže vznikne 5 nových atributů.
    - jeden atribut bude vždy nabývat hodnoty 1 zatímco ostatní atributy hodnoty 0
  - použijeme OneHotEncoder
- varianta C: převod na nízkodimenzionální vektor (embedding)
  - embedding se naučí reprezentovat daná textová/kategoriální data pomocí nízkodimenzionálního vektoru
  - využitelné např. pro slova v souvislém textu nebo pro reprezentaci fotografie obličeje (vrátíme se k tomu později)

Normalizace a standardizace příznaků

- min-max normalizace
  - hodnoty jsou posunuty a škálovány tak, aby výsledné hodnoty byly v intervalu 0 až 1
  - od hodnot se odečte minimální hodnota a výsledek se vydělí rozdílem max - min
  - transformer MinMaxScaler

- standardizace
  - od hodnot je odečtena střední hodnota a výsledek je dělen směrodatnou odchylkou
  - na rozdíl od min-max normalizace tak výsledné hodnoty neleží v pevně daném intervalu, což může být pro některé algoritmy problém (např. neuronové sítě typicky očekávají hodnoty 0-1)
  - standardizace je naopak mnohem méně ovlivněna odlehlými hodnotami

Spuštění, monitorování a údržba systému

- napojení na datové zdroje
- monitorování kvality predikcí
  - obecně vyžaduje lidskou práci, protože je potřeba k predikcím přiřazovat skutečnost
- monitorování kvality vstupních dat
  - zejména klíčové pro online learning systémy
- automatizace procesu trénovaní modelu
  - jinak je problém udržet model aktuální a s dobrými výsledky
  - v případě online learning systémů dělat pravidelné zálohy

Zdroje

- Coelho, L. P.; Richert, W. (2013) Building machine learning systems with Python. Birmingham: Packt Publishing. ISBN 978-1-78216-140-0.

- Géron, A. (2019) Hands-On Machine Learning with Scikit-Learn, Keras, and TensorFlow, 2nd Edition. O'Reilly Media, Inc. ISBN 9781492032649.

- Chollet, F. (2019) Deep Learning v jazyku Python. Knihovny Keras, TensorFlow. Grada Publishing, a.s. ISBN 978-80-247-3100-1.

- Segaran, T. (2007) Programming collective intelligence: building smart web 2.0 applications. Beijing: O'Reilly Media. ISBN 0-596-52932-5.