5. Оценяване на модели в Machine Learning: Основни концепции

Проблеми: Training (Трениране), Testing (Тестване), Tuning (Настройка на параметри)

Прогнозиране на представянето:

- **Confidence limits (Граници на доверителния интервал)**
- **Holdout (Разделяне на данните), Cross-validation (Крос-валидация), Bootstrap (Бутстрап метод)**
- **Hyperparameter selection (Избор на хиперпараметри)**
- **Comparing machine learning schemes (Сравняване на ML методи)**
- **Predicting probabilities (Прогнозиране на вероятности)**
- **Cost-sensitive evaluation (Оценяване с отчитане на разходите)**
- **Evaluating numeric prediction (Оценка на числови прогнози)**
- **Minimum description length principle (Принцип на минималната дължина на описанието)**
- **Model selection using a validation set (Избор на модел чрез валидационен сет)**
- Evaluation (Оценка): Ключът към успеха - Колко добра е прогнозата на модела? Грешката върху обучителни данни **не е добър индикатор** за представяне върху нови данни. Ако беше така, **1-NN (1-Nearest Neighbor)** би бил оптималният модел, но това не е вярно.
- Прост подход при наличие на голямо количество (етикетирани) данни: Разделяне на данните на **training set (обучителен сет) и test set (тестов сет)**. Проблем: Обикновено наличните данни са **ограничени**; Трябва да се използват по-сложни методи за оценка.

Проблеми при оценяването:

- **Статистическа надеждност** на разликите в представянето (**тестове за значимост**).

- **Избор на метрика за оценка:**
  - **Accuracy (Точност)** – Брой правилно класифицирани примери.
  - **Вероятностни оценки** – Колко точни са вероятностите?
  - **Грешка в числовите прогнози**.
  - **Разходи, свързани с различни видове грешки** (пример: FP & FN в медицинска диагностика).
- В реални приложения често има **разходи**, свързани с грешки.

Training and Testing (Обучение и Тестване)

I. Основна метрика за класификационни задачи: Error rate (Честота на грешките) Успех – Прогнозираният клас е правилен. Грешка – Прогнозираният клас е грешен. Error rate – Пропорцията на грешките в целия dataset.

Resubstitution error (Грешка от повторно използване) - Изчислява се като грешката на модела върху **обучителните данни**. **Проблем:** Тази грешка е прекалено **оптимистична** и не отразява реалното представяне.

Тестване с независим dataset (Test set)

Test set (Тестов сет) = Данни, които не са използвани при създаването на класификатора. **Допускане:** Обучителните и тестовите данни са **репрезентативни** за реалния проблем.Проблем: Разлика в разпределенията. Пример: Класификаторът е обучен с данни от клиенти на град A. Ако го тестваме с данни от клиенти на град B, може да не работи добре.

Parameter Tuning (Настройка на параметри) - Важно е **тестовите данни да не се използват** за настройка на модела. Някои алгоритми работят на **два етапа**: 1/Изграждане на основната структура; 2/Оптимизиране на параметрите

Решение: Използване на три сета: Training set (Обучителен сет) – за обучение. Validation set (Валидационен сет) – за настройка на параметри. Test set (Тестов сет) – за окончателна оценка.

Използване на всички налични данни - След оценката **всички данни могат да бъдат използвани за крайния модел**. **Колкото по-голям е обучителният сет, толкова по-добър е моделът** (но ефектът намалява с увеличаването на данните). **Колкото по-голям е тестовият сет, толкова по-точна е оценката на грешката**. **Holdout метод:** Разделя оригиналния dataset на **обучителен** и **тестов** сет.

Дилема: Идеално е и двата сета да са големи, но това рядко е възможно!

Predicting Performance (Прогнозиране на представянето)

Пример: Изчислена грешка = **25%**. Колко близо е това до **истинската** грешка? **Зависи от размера на тестовите данни**.

Статистически модел: Прогнозирането на грешка е **като хвърляне на монета** (успех или грешка). Това е **Bernoulli процес** (последователност от независими събития). **Статистическата теория** ни дава **доверителни интервали** за истинската грешка.

Confidence Intervals (Доверителни интервали) - Можем да кажем, че **p** (истинската стойност) лежи в **определен интервал** с **определена сигурност**. Пример:**S = 750 успеха от N = 1000 теста → p ≈ 75%**

С **80% сигурност** истинската стойност p е в интервала **[73.2%, 76.7%]**.

За малки N (напр. 10) → Доверителните интервали не са надеждни!

Mean and Variance (Средно и дисперсия)

Средна стойност на успехите **f = S/N**.

- За големи **N**, разпределението следва **Нормално разпределение**.
- **Доверителните граници** се изчисляват чрез **z-стойности**.

Примерни изчисления за различни размери на N:

- **N = 1000** → Надежден доверителен интервал.
- **N = 100** → По-широк, но все още приемлив.
- **N = 10** → Резултатите трябва да се приемат с **резерви**.

Holdout Estimation (Задържащо оценяване) - Какво да правим, ако имаме само един dataset? Holdout методът отделя определено количество от данните за **тестване**, а останалите се използват за **обучение** (след разбъркване). Обикновено: **1/3 за тестване**, останалите за обучение. **Проблем**: извадките може да не са **представителни**. Пример: даден **клас** може да липсва в тестовите данни. **Подобрена версия**: използва **стратификация (stratification) - всяки клас** е представен с приблизително **еднаква пропорция** в двата подмножествени набора.

Repeated Holdout Method (Повторяемо задържащо оценяване) - Holdout оценката става **по-надеждна**, ако процесът се **повтаря** с различни подизвадки. При всяка итерация: Определен процент от данните се избира **случайно** за обучение (**със стратификация**). Изчисляват се **грешките**, след което се **усредняват** резултатите от всички итерации. **Проблем**: тестовите набори могат да **се припокриват**. **Може ли това да бъде избегнато?**

Cross-Validation (Крос-валидация)

-K-fold Cross-Validation (K-кратна крос-валидация) Разделяне на данните на **K подмножествени набора** с **равни размери**. Всеки поднабор се използва **веднъж за тестване**, останалите **за обучение**. **K различни обучения** -> усредняване на грешките.

-Stratified K-fold Cross-Validation (Стратифицирана K-кратна крос-валидация)- Подмножествата са **стратифицирани** -> намалява **вариацията** на оценката. **Предимство:** По-надеждна оценка в сравнение с **holdout метода**. **Недостатък**: **Изчислително скъп** при големи набори от данни.

-Leave-One-Out Cross-Validation (LOO-CV) – Изключване на една проба - **Специален случай на K-fold CV**, където **K = броя на тренировъчните примери**. **Всеки пример** се използва **веднъж за тестване**, останалите – за обучение. **Предимства**: **Максимално използване на данните**; Не зависи от **случайното разделяне** на данните. **Недостатъци**: **Много изчислително скъпо**; **Невъзможна стратификация** (само един тестов пример).

The Bootstrap (Бутстрап метод) - Използва **вземане на проби с връщане (sampling with replacement)**.

-Процедура: N примера се избират **N пъти с връщане** -> формира се нов тренировъчен набор. **Останалите неизбрани примери** се използват за **тестване**.

0.632 Bootstrap: Всеки пример има **1 – (1/N) шанс да не бъде избран**. Това означава, че **~63.2% от данните** влизат в **тренировъчния** набор.

Предимство: Подходящ за **много малки набори от данни**. **Недостатък**: Може да **подцени грешката** при модели, които просто запомнят данните (**overfitting**).

Hyperparameter Selection (Избор на хиперпараметри)

Хиперпараметър: **параметър**, който се настройва **преди** обучението на модела. Пример: **k** в **k-Nearest Neighbors**.

Важно правило: **Не използваме тестовите данни** за избиране на **хиперпараметри**! Ако **„нагласим“ параметъра по тестовите данни, ще получим подвеждащи резултати**.

Разделяме тренировъчните данни на **train set** и **validation set**. Изпробваме различни стойности на **хиперпараметъра** и избираме

най-добрата по validation set. След това обучаваме **финалния модел** с тази стойност върху **цялото обучение**.

Nested Cross-Validation (Вложена крос-валидация) - Когато **тренировъчните данни са малко**, използваме **вътрешна** крос-валидация за избор на параметър.

Сравняване на алгоритми за машинно обучение:

Как да разберем кой алгоритъм е по-добър? Емпирично: чрез **10-fold cross-validation**; **Статистически**: чрез **тестове за значимост**.

Paired t-test (Сдвоен t-тест) -Използва се за **сравняване на два модела** върху **едни и същи** тестови данни. **Стъпки:**

1.Изчисляваме **средната грешка** на модел **A** и модел **B** върху няколко разцепвания на данните.

2.Проверяваме дали разликата между средните стойности е **статистически значима**.

3.Ако разликата е **значима**, можем да твърдим, че единият модел е **по-добър**.

Тестът е базиран на Student's t-distribution, открит от **William Gosset**, докато работел за **пивоварната Guinness**.

Unpaired observations (Неподредени наблюдения) - Ако оценките от **Cross-Validation (CV)** идват от различни **datasets (набори от данни)**, те вече не са **paired (сдвоени)**. (Или може би имаме **k** оценки за една схема и **j** оценки за друга схема.) В такъв случай трябва да използваме **unpaired t-test (t-тест за независими извадки)** със **min(k, j) – 1** степени на свобода. Статистиката на **t-test** се изчислява по съответна формула.

Dependent estimates (Зависими оценки) - Приемаме, че имаме достатъчно данни, за да създадем няколко **datasets** с желания размер. Ако това не е така, трябва да **преизползваме данни**, например чрез **Cross-Validation (CV)** с различни разпределения. Това води до **зависими извадки**, което може да направи **незначителни разлики значими**. За справяне с този проблем се използва **corrected resampled t-test (коригиран повторен t-тест)**. Приемаме, че използваме **repeated hold-out method (повторен hold-out метод)** с **k** изпълнения, където: **n1** – брой примери за **training (обучение); n2** – брой примери за **testing (тестване)**

Predicting probabilities (Предсказване на вероятности) - Досега използваната метрика за оценка на моделите е **success rate (успех)**, наричан още **0-1 loss function (0-1 загуба)**. Повечето **classifiers (класфикатори)** генерират вероятности за класове. Ако искаме да оценим **точността на тези вероятности**, **0-1 loss** не е най-подходящата метрика.

Quadratic loss function (Квадратична функция на загубите) - Нека **p1 … pk** са вероятностите за даден пример, а **i** е индексът на реалния клас. Тогава **Quadratic loss** се дефинира така:

$\mathbf{L=\sum j(p_j-a_j)^2}$, $\mathbf{a_j = 1}$ само за реалния клас, а за останалите $\mathbf{a_j= 0}$ Целта е да **минимизираме** тази загуба. Можем да покажем, че **очакваната стойност** на загубата е минимална, когато $\mathbf{p_j = p_{j*}}$ (истинските вероятности).

Informational loss function (Информационна загуба) - $\log(p_i)$, където **i** е реалният клас. Това представлява броя битове, нужни за предаване на истинския клас. Тук също целта е да **минимизираме загубата**, като $p_j \rightarrow p_{j^*}$. Проблем: zero-frequency problem (проблем с нулеви вероятности) Ако някоя вероятност е **0**, логаритъмът става неопределен.

Коя функция на загубите да изберем?

- И двете **поощряват честни вероятностни оценки**.
- **Quadratic loss** отчита всички вероятности за даден пример.
- **Informational loss** разглежда само вероятността на истинския клас.
- **Quadratic loss** е **ограничена (≤ 2)**.
- **Informational loss** може да бъде **безкрайна**. **Informational loss** е свързана с **Minimum Description Length (MDL) принципа**.

Counting the cost (Отчитане на цената на грешките) - Различните видове грешки в класификацията имат **различни разходи/цена**.

Confusion matrix (Матрица на объркване)- да дефинираме **различни разходи** за **false positives (FP)** и **false negatives (FN)**. Има и други видове разходи –**разходи за събиране данни**.

Aside: Kappa statistic (Капа статистика) - измерва **подобрението спрямо случаен предсказател**.

κ=Success rate на реалния модел−Success rate на случаен модел1−Success rate на случаен модел Стойностите са: 1 – перфектна точност; 0 – същото като случайно предсказване.

Cost-sensitive classification (Класификация, чувствителна към разходи) - Обикновено избираме класа с най-висока вероятност, но тук **вземаме предвид разходите**. Минимизираме **очакваните разходи**, като използваме **кост матрица** и изчисляваме:

$E=P_1 \cdot C_1+P_2 \cdot C_2+...+P_k \cdot C_k$

Избираме класа, който **минимизира очаквания разход**.

ROC Curves & Lift Charts (ROC криви и Lift диаграми)

ROC(Receiver Operating Characteristic) крива – показва съотношението **TPR (True Positive Rate)** спрямо **FPR (False Positive Rate)**. **Lift диаграма** – визуализира подобрението на модела спрямо случайно предсказване.

AUC (Area Under Curve) – площта под ROC кривата. По-голяма стойност → по-добър модел.

Model Selection & MDL (Избор на модел и MDL принцип)

MDL (Minimum Description Length) – стремим се към **най-простия модел**, който описва данните.

DL=Размер на модела+Грешки на модела

Това е **алтернатива** на **Maximum A Posteriori (MAP)** метода. Целта е да избегнем **overfitting (пренасищане)**.

Validation Set & Model Selection (Валидационен набор и избор на модел) Класически проблем:

- Избор на **брой атрибути** за **линейна регресия**.
- Определяне на **размера на decision tree (решаващо дърво)**.

Един подход е **cross-validation** или **валидационен сет**. Целта е да изберем модела с **най-добро представяне върху валидационния сет**.