

ML job interview questions

Algis Dumbris

Version 1.0

Table of Contents

According to openml_course	1
Pandas.....	1
Visualisation (https://habrahabr.ru/company/ods/blog/323210/)	1
Decidions tree (https://habrahabr.ru/company/ods/blog/322534/)	1
k Nearest Neighbors, или kNN	3
Линейная регрессия. Метод наименьших квадратов.....	4
Разложение ошибки на смещение и разброс (Bias-variance decomposition)	4
Логистическая регрессия	5
Линейный классификатор.....	5
RandomForrest	6
Questions and Answers.....	7

According to openml_course

Pandas

- Basics
 - `pd.info()`, `pd.describe()`
- To see how Churn related with 'International plan'
 - `pd.crosstab(df['Churn'], df['International plan'], margins=True)`

Visualisation (<https://habrahabr.ru/company/ods/blog/323210/>)

- Plot
 - `sales_df.groupby('Year_of_Release').sum().plot()`
- (Seaborn) How features related with each other
 - `sns.pairplot(df[cols])`
- joint plot — это гибрид scatter plot и histogram.
 - `sns.distplot(df.Critic_Score)`
- box plot
 - Коробка показывает интерквартильный размах распределения, то есть соответственно 25% (Q1) и 75% (Q3) перцентили. Черта внутри коробки обозначает медиану распределения. С коробкой разобрались, перейдем к усам. Усы отображают весь разброс точек кроме выбросов, то есть минимальные и максимальные значения, которые попадают в промежуток $(Q1 - 1.5 \cdot IQR, Q3 + 1.5 \cdot IQR)$, где $IQR = Q3 - Q1$ — интерквартильный размах. Точками на графике обозначаются выбросы (outliers) — те значения, которые не вписываются в промежуток значений, заданный усами графика.
- heat map
 - `sns.heatmap(platform_genre_sales, annot=True, fmt=".1f", linewidths=.5)`

Decidions tree (<https://habrahabr.ru/company/ods/blog/322534/>)

- *говорят, что компьютерная программа обучается при решении какой-то задачи из класса T , если ее производительность, согласно метрике P , улучшается при накоплении опыта E .*
- Interpretable for human
- C4.5 - Top 10 algorithms in data mining
- Shanon entropy $S = - \sum_{i=1}^N p_i \log_2 p_i$

- Формально прирост информации (information gain, IG) при разбиении выборки по признаку Q $IG(Q) = S_0 - \sum_{i=1}^q \frac{N_i}{N} S_i$,
 - q - количество получившихся групп
 - N_i - кол-во элементов в группе
 - S_i - энтропия группы
- Очевидно, энтропия группы с шариками одного цвета равна 0 ($\log_{21} = 0$), что соответствует представлению, что группа шариков одного цвета – упорядоченная.
- **Алгоритм**
 - В основе популярных алгоритмов построения дерева решений, таких как ID3 и C4.5, лежит принцип жадной максимизации прироста информации – на каждом шаге выбирается тот признак, при разделении по которому прирост информации оказывается наибольшим.
 - Далее процедура повторяется рекурсивно, пока энтропия не окажется равной нулю или какой-то малой величине

```
def build(L):
    create node t
    if the stopping criterion is True:
        assign a predictive model to t
    else:
        Find the best binary split L = L_left + L_right
        t.left = build(L_left)
        t.right = build(L_right)
    return t
```

- Другие критерии качества разбиения в задаче классификации
 - Неопределенность Джини (Gini impurity): $G = 1 - \sum_k (p_k)^2$ Максимизацию этого критерия можно интерпретировать как максимизацию числа пар объектов одного класса, оказавшихся в одном поддереве.
- самая простая эвристика для обработки количественных признаков в дереве решений: количественный признак сортируется по возрастанию, и в дереве проверяются только те пороги, при которых целевой признак меняет значение.
- Основные способы борьбы с переобучением в случае деревьев решений:
 - искусственное ограничение глубины или минимального числа объектов в листе: построение дерева просто в какой-то момент прекращается;
 - Стрижка дерева (pruning). При таком подходе дерево сначала строится до максимальной глубины, потом постепенно, снизу вверх, некоторые вершины дерева убираются за счет сравнения по качеству дерева с данным разбиением и без него (cross validation).
- **Дерево решений в задаче регрессии:**
 - меняется критерий качества - Дисперсия вокруг среднего $D = \frac{1}{l} \sum_{i=1}^l (y_i - \frac{1}{l} \sum_{i=1}^l y_i)^2$
 - где l – число объектов в листе, y_i – значения целевого признака. Попросту говоря,

минимизируя дисперсию вокруг среднего, мы ищем признаки, разбивающие выборку таким образом, что значения целевого признака в каждом листе примерно равны.

- **Pros**

- Порождение четких правил классификации, понятных человеку
- Поддержка и числовых, и категориальных признаков.

- **Cons**

- деревья очень чувствительны к шумам во входных данных, вся модель может кардинально измениться, если немного изменится обучающая выборка
- Разделяющая граница, построенная деревом решений, имеет свои ограничения (состоит из гиперплоскостей, перпендикулярных какой-то из координатной оси)
- Необходимость отсекаать ветви дерева (pruning)
- Сложно поддерживаются пропуски в данных.

k Nearest Neighbors, или kNN

- Формально основой метода является гипотеза компактности: если метрика расстояния между примерами введена достаточно удачно, то схожие примеры гораздо чаще лежат в одном классе, чем в разных.

- **Алгоритм**

- Вычислить расстояние до каждого из объектов обучающей выборки
- Отобрать объектов обучающей выборки, расстояние до которых минимально
- Класс классифицируемого объекта — это класс, наиболее часто встречающийся среди ближайших соседей
- Под задачу регрессии метод адаптируется довольно легко – на 3 шаге возвращается не метка, а число – среднее (или медианное) значение целевого признака среди соседей.
- метрика расстояния между объектами (часто используются метрика Хэмминга, евклидово расстояние, косинусное расстояние и расстояние Минковского).
 - Отметим, что при использовании большинства метрик значения признаков надо масштабировать. Условно говоря, чтобы признак "Зарплата" с диапазоном значений до 100 тысяч не вносил больший вклад в расстояние, чем "Возраст" со значениями до 100.

- **Pros**

- Простая реализация;
- Можно адаптировать под нужную задачу выбором метрики или ядра (в двух словах: ядро может задавать операцию сходства для сложных объектов типа графов, а сам подход kNN остается тем же).
- Неплохая интерпретация, можно объяснить, почему тестовый пример был классифицирован именно так.

- **Cons**

- в реальных задачах, как правило, число соседей, используемых для классификации, будет большим (100-150), и в таком случае алгоритм будет работать не так быстро, как дерево решений;
- Если в наборе данных много признаков, то трудно подобрать подходящие веса и определить, какие признаки не важны для классификации/регрессии;
- Нет теоретических оснований выбора определенного числа соседей — только перебор
- Как правило, плохо работает, когда признаков много, из-за "проклятия размерности" (the curse of dimensionality).

Линейная регрессия. Метод наименьших квадратов

- Модель $\vec{y} = X\vec{w} + \epsilon$
 - на модель накладываются следующие ограничения
 - матожидание случайных ошибок равно нулю
 - дисперсия случайных ошибок одинакова и конечна, это свойство называется гомоскедастичностью
 - случайные ошибки не скоррелированы
 - Один из способов вычислить значения параметров модели является метод наименьших квадратов (МНК), который минимизирует среднеквадратичную ошибку между реальным значением зависимой переменной и прогнозом, выданным моделью
 - $\vec{w} = (X^T X)^{-1} X^T \vec{y}$ (see Deep Learning Book)
 - Для решения данной оптимизационной задачи необходимо вычислить производные по параметрам модели, приравнять их к нулю и решить полученные уравнения относительно
 - можем утверждать, опираясь на теорему Маркова-Гаусса, что оценка МНК является лучшей оценкой параметров модели, среди всех линейных и несмещенных оценок, то есть обладающей наименьшей дисперсией.

Разложение ошибки на смещение и разброс (Bias-variance decomposition)

- Смещение – это то, насколько далеки предсказания модели от правды
- дисперсия – степень, в которой эти предсказания различаются между итерациями модели.
- Линейные модели (<https://www.coursera.org/learn/supervised-learning/lecture/Ctw7C/smieshchieniie-i-razbros>)

- Большое смещение (Bias)
- Низкий разброс (Variance)
- Tree-based модели
 - Низкое смещение
 - Большой разброс
- Усреднение алгоритмов
 - Не меняет смещение
 - Разброс $1/N$ (разброс базового алгоритма) + (корреляция между базовыми алгоритмами)
 - Если алгоритмы независимы - разброс уменьшается в N раз
- Как увеличить независимость алгоритмов
 - Бэггинг (Bagging от Bootstrap aggregation): обучение на случайной подвыборке (выбираем случайные строки)
 - Метод случайных подпространств: обучаем на случайном подмножестве признаков (выбираем случайные столбцы)
- Регуляризация
 - $\vec{w} = (X^T X + \lambda E)^{-1} X^T \vec{y}$ Такая регрессия называется гребневой регрессией (ridge regression). А гребнем является как раз диагональная матрица, которую мы прибавляем к матрице $(X^T X)$, в результате получается гарантированно регулярная матрица. Такое решение уменьшает дисперсию, но становится смещенным, т.к. минимизируется также и норма вектора параметров, что заставляет решение сдвигаться в сторону нуля.

Логистическая регрессия

Линейный классификатор

- Основная идея линейного классификатора заключается в том, что признаковое пространство может быть разделено гиперплоскостью на два полупространства, в каждом из которых прогнозируется одно из двух значений целевого класса.
- Логистическая регрессия является частным случаем линейного классификатора, но она обладает хорошим "умением" – прогнозировать вероятность отнесения примера к классу "+":
 - $p_+ = P(y_i = 1 | \vec{x}_i, \vec{w})$
 - Прогнозирование не просто ответа ("1" или "-1"), а именно вероятности отнесения к классу "1" во многих задачах является очень важным бизнес-требованием.
 - Итак, мы хотим прогнозировать вероятность p_+ , а пока умеем строить линейный прогноз с помощью МНК: . Каким образом преобразовать полученное значение в вероятность, пределы которой – $[0, 1]$? Очевидно, для этого нужна некоторая функция. В модели логистической регрессии для этого берется конкретная функция

$$\sigma(x) = \frac{1}{1 + \exp^{-x}} \text{ (sigmoid function)}$$

- the XOR problem
 - Очевидно, нельзя провести прямую так, чтобы без ошибок разделить один класс от другого. Поэтому логистическая регрессия плохо справляется с такой задачей.
 - А вот если на вход подать полиномиальные признаки, в данном случае до 2 степени, то проблема решается.
 - На практике полиномиальные признаки действительно помогают, но строить их явно – вычислительно неэффективно. Гораздо быстрее работает SVM с ядровым трюком. При таком подходе в пространстве высокой размерности считается только расстояние между объектами (задаваемое функцией-ядром), а явно плодить комбинаторно большое число признаков не приходится.
- pros
 - Практически вне конкуренции, когда признаков очень много (от сотен тысяч и более), и они разреженные (хотя есть еще факторизационные машины)
 - Коэффициенты перед признаками могут интерпретироваться
 - Модель может строить и нелинейную границу, если на вход подать полиномиальные признаки
- cons
 - Плохо работают в задачах, в которых зависимость ответов от признаков сложная, нелинейная
 - чаще линейные методы работают хуже, чем, например, SVM и ансамбли

RandomForrest

- Ансамбль моделей, использующих метод случайного подпространства
- Итоговый классификатор - для задачи классификации мы выбираем решение голосованием по большинству, а в задаче регрессии — средним.
- Основное различие случайного леса и бэггинга на деревьях решений заключается в том, что в случайном лесе выбирается случайное подмножество признаков, и лучший признак для разделения узла определяется из подвыборки признаков, в отличие от бэггинга, где все функции рассматриваются для разделения в узле.
- В сверхслучайных деревьях (Extremely Randomized Trees) больше случайности в том, как вычисляются разделения в узлах.
 - Их используют если RandomForrest переобучается
- Метод случайного леса схож с методом ближайших соседей. Случайные леса, по сути, осуществляют предсказания для объектов на основе меток похожих объектов из обучения. Схожесть объектов при этом тем выше, чем чаще эти объекты оказываются в одном и том же листе дерева.
- Pros
 - имеет высокую точность предсказания, на большинстве задач будет лучше

линейных алгоритмов; точность сравнима с точностью бустинга

- практически не чувствителен к выбросам в данных из-за случайного сэмлирования
- не чувствителен к масштабированию
- одинаково хорошо обрабатывает как непрерывные, так и дискретные признаки
- хорошо работает с пропущенными данными
- высокая параллелизуемость и масштабируемость.

- **Cons**

- результаты случайного леса сложнее интерпретировать
- алгоритм работает хуже многих линейных методов, когда в выборке очень много разреженных признаков

Questions and Answers

- Feature selection

- Отбор признаков используют для устранения избыточных признаков (например, которые дублируются) и нерелевантных (например, которые не имеют отношения к решению задачи) признаков, что может
 - повысить надёжность обучения (уменьшить эффект переобучения)
 - повысить скорость работы алгоритмов (чем меньше признаков, тем быстрее)

- Overfitting

- переобучение возникает при использовании избыточно сложных моделей.

- Регуляризация

- в задаче оптимизации к целевой функции добавляют регуляризационное слагаемое (т.н. штраф).
- любой способ борьбы с переобучением, который касается настройки модели, в машинном обучении называют регуляризацией (начиная от dropout и заканчивая pruning)

- Unbalanced data

- StratifiedKFold- при разбиении на фолды надо сохранять пропорцию классов
- SMOTE: Synthetic Minority Over-sampling Technique

- Outliers

- удаление выбросов на этапе подготовки данных (в том числе, детектирование аномальных значений, винзоризация, стат. критерии, преобразование признаков и т.п.),
- применение т.н. робастных моделей (например, линейных с настройкой не на сумму квадратов ошибки, а на сумму модулей),
- удаление выбросов и переобучение моделей (например, удаляя объекты, на которых модель ошибается сильнее).

- Anomaly Detection
 - детектирование выбросов (Outlier Detection) и «новизны» (Novelty Detection).
 - Statistics test
 - Z-value или Kurtosis measure.
 - ML methods
 - Метод опорных векторов для одного класса (OneClassSVM)
 - Изолирующий лес (IsolationForest)
- Feature selection
 - VarianceThreshold
 - SelectKBest, f_classif
 - RandomForest, Lasso
 - Exhaustive Feature Selection.
 - Sequential Feature Selector
 - Boruta
- Мат ожидание (expectation, mathematical expectation, EV) -
 - the expected value of a random variable, intuitively, is the long-run average value of repetitions of the experiment it represents.
- Non-parametric method
- Generative vs
- maximum likelihood estimation метод максимального правдоподобия
- Как строить рекомендательные системы