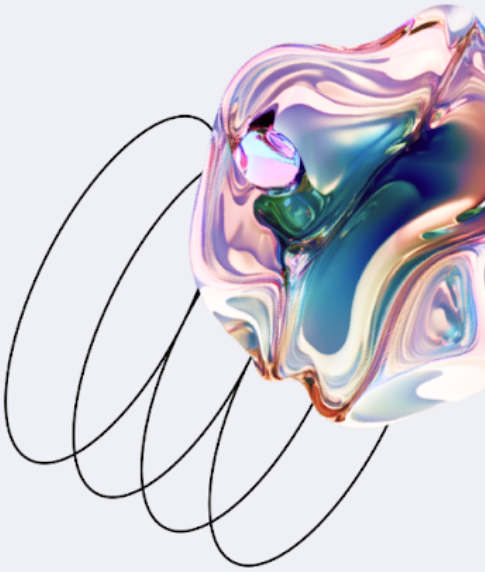
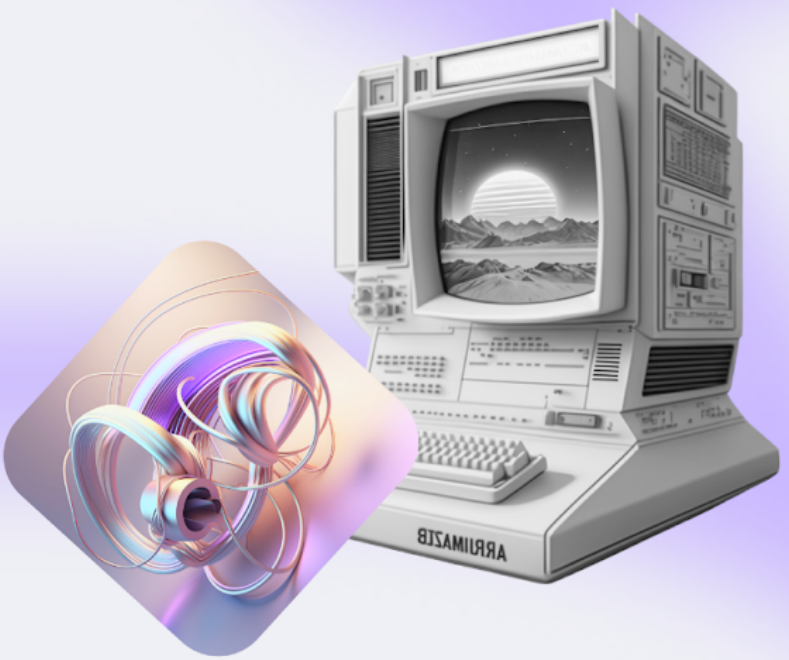
GeekBrains



Инструменты разметки наборов данных

Сбор и разметка данных



Оглавление

Введение

Добро пожаловать на заключительную лекцию курса «Сбор и разметка данных»!

На предыдущих занятиях мы рассмотрели целый ряд тем, включая основы клиент-серверного взаимодействия, парсинг HTML, веб-скрейпинг и работу с базами данных (MongoDB и Clickhouse). Сегодня обсудим инструменты разметки данных.

Разметка данных — важный шаг в построении моделей машинного обучения. Включает ручную аннотацию данных метками, которые указывают на правильную классификацию или вывод. Разметка может быть трудоемкой и сложной, но она нужна для создания точных и надежных моделей.

В этой лекции мы разберемся в процессе разметки, рассмотрим методы, которые позволят сделать ее более эффективной, и познакомимся с инструментами.

Определение разметки данных и ее значение в машинном обучении

Разметка или аннотация данных — это процесс присвоения меток точкам данных. Эти метки используют, чтобы указать правильную классификацию или вывод для каждой точки данных, что позволяет обучать модели машинного обучения распознавать закономерности и делать прогнозы.

Чтобы проиллюстрировать важность разметки данных, рассмотрим пример. Компания создает модель машинного обучения, которая сможет определять тип объекта на изображении: людей, автомобили и здания. Чтобы ее обучить, нужно разметить большое количество изображений: метки должны указывать на тип объекта на каждом из них. Но разные люди могут размечать объекты по-разному.

Например, один человек может обозначить изображение, как содержащее автомобиль, если видна только небольшая его часть, а другой — если автомобиль виден целиком. Кто-то может обозначить объект как человека, если видно его лицо, кто-то — если видно все тело. Из-за таких несоответствий данные могут быть шумными или неточными. Точность и надежность модели снизится.

Дата-инженерам важно понимать инструменты и методы маркировки данных, даже если они за нее не отвечают. Понимая процесс маркировки, дата-инженеры могут выбрать правильные инструменты и технологии, оптимизировать обработку данных, обеспечить их качество и согласованность.

Есть несколько типов инструментов, включая маркировку на основе правил, маркировку с участием человека и маркировку на основе активного обучения. У каждого свои преимущества и недостатки. Их важно понимать, чтобы правильно выбрать инструмент для конкретной задачи разметки.

Типы инструментов разметки данных

В этом разделе сделаем обзор существующих типов разметки данных и обсудим их сильные стороны и ограничения.

Разметка на основе правил подразумевает создание набора правил или критериев для присвоения меток данным. Подходит для простых задач маркировки, которые можно определить с помощью набора правил. Например, для автоматического присвоения меток текстовым данным на основе ключевых слов или фраз.

Преимущества — скорость, низкая стоимость, способность обрабатывать большие объемы данных. Но такая разметка не подходит для более сложных задач, требующих человеческого суждения или интерпретации.

Разметка по принципу «человек в цикле» (HITL, human-in-the-Loop) — комбинация ручной и автоматической разметки: человек предоставляет начальные метки, а модель машинного обучения учится на них, чтобы со временем повысить свою точность.

Инструменты HITL-разметки часто используют для более сложных задач, однако они могут быть более дорогими и трудоемкими, чем инструменты разметки на основе правил.

Разметка на основе активного обучения — использование алгоритмов машинного обучения для выбора наиболее информативных точек данных для маркировки. Идея в том, чтобы сосредоточить усилия по разметке на наиболее сложных точках данных, а не маркировать все точки данных без разбора. Для оптимизации процесса разметки и сокращения объема ручной разметки часто используют инструменты с активным обучением.

Преимущества — сокращается необходимость ручных разметок и повышается эффективность. Но для настройки инструментов может потребоваться больше опыта, и они подходят не для всех типов задач.

Гибридные инструменты разметки сочетают в себе два подхода или более. Обеспечивают более гибкий и адаптируемый процесс. Например, для широкого спектра задач подходит сочетание разметки на основе правил с HITL-разметкой.

Далее мы разберем несколько примеров, когда каждый тип инструмента может быть лучшим выбором.

Разметка на основе правил

Разметка на основе правил подразумевает создание набора правил или критериев для присвоения меток данным. Эти правила могут быть основаны на ключевых словах, фразах или шаблонах, которые ассоциируются с меткой.

Один из примеров такой разметки — анализ настроений: текстовые данные маркируются как положительные, отрицательные или нейтральные на основе слов или фраз, связанных с каждым настроением.

Например, пост в соцсети можно разметить как положительный, если он содержит слова «счастливый», «отличный» или «потрясающий», и отрицательный, если в нем есть слова «разочарование», «расстроенный» или «ужасный».

Другой пример разметки на основе правил — фильтрация спама: электронные письма помечаются как спам или не спам на основе критериев — ключевых слов или email-адреса отправителя.

Иногда такую разметку используют в качестве отправной точки для решения более сложных задач. Например, мы можем присвоить начальные метки текстовым данным, которые затем проверит и скорректирует человек, чтобы обеспечить точность.

Посмотрим пример простой системы разметки на основе правил для анализа настроений с использованием Python и библиотеки Natural Language Toolkit (NLTK).

```
import pandas as pd
import nltk
nltk.download('punkt')
```

nltk.download('punkt') — это команда библиотеки NLTK, которая загружает пакет данных punkt. Пакет punkt содержит предварительно обученные модели для токенизации текста.

Токенизация текста подразумевает разбиение фрагмента на мелкие единицы — отдельные слова или предложения. Это обычный этап предварительной обработки в задачах обработки естественного языка. Его цель — извлечение значимой информации из необработанных текстовых данных.

Подход полезен для анализа настроения, когда мы хотим проанализировать настроение отдельных слов в тексте. Например, мы разделим предложение «Я люблю этот фильм!» на отдельные слова и получим их список: [«я», «люблю», «этот», «фильм»].

```
# Определение набора положительных и отрицательных слов
positive_words = ['happy', 'excited', 'awesome']
negative_words = ['disappointed', 'frustrated', 'terrible']

# Функция для присвоения метки настроения на основе наличия
положительных или отрицательных слов
def get_sentiment(text):
    # Токенизация текста на отдельные слова
    words = nltk.word_tokenize(text.lower())

    # Подсчет количества положительных и отрицательных слов
    num_positive = sum([1 for word in words if word in
positive_words])
    num_negative = sum([1 for word in words if word in
negative_words])

    # Присвоение метки на основе чистой оценки настроения
    if num_positive > num_negative:
        return 'positive'
    elif num_negative > num_positive:
        return 'negative'
    else:
        return 'neutral'

# Тестирование функции
text1 = "I'm so happy today!"
text2 = "I'm so frustrated with this service."
text3 = "This movie was okay, I guess."
print(get_sentiment(text1))
print(get_sentiment(text2))
print(get_sentiment(text3))
```

В примере мы определяем набор положительных и отрицательных слов и используем функцию для присвоения метки настроения тексту на их основе.

Функция get_sentiment() разбивает текст на отдельные слова с помощью функции word_tokenize() в NLTK, подсчитывает количество положительных и отрицательных слов и присваивает метку.

Это простой пример, но он иллюстрирует, как можно использовать разметку на основе правил для присвоения меток данным по определенным критериям или шаблонам. На практике более сложные системы, основанные на правилах, могут

использовать регулярные выражения или другие методы для извлечения определенных шаблонов из текстовых данных для маркировки.

Давайте усложним задачу и выполним разметку датасета твитов. Импортируем датасет и посмотрим на данные.

```
tweets = pd.read_csv('tweets.csv')
tweets.head()
```

Наша задача — выполнить разметку твитов. Как и в предыдущем примере, разделим их на 3 категории. Импортируем класс TextBlob из модуля textblob.

```
from textblob import TextBlob
```

TextBlob — это библиотека Python для обработки естественного языка. Она предоставляет простой API для распространенных задач NLP: анализа настроений, тегирования части речи и классификации текста.

```
def get_sentiment(tweet):
    blob = TextBlob(tweet)
    return blob.sentiment.polarity
```

В этой функции мы создаем объект TextBlob из текстовой строки (твита) и используем метод sentiment.polarity, чтобы получить оценку полярности настроения текста. Результирующая оценка настроения представляет собой плавающее значение между -1 и 1, где значения ближе к 1 указывают на более позитивное настроение, а значения ближе к -1 — на более негативное настроение.

```
tweets['sentiment'] = tweets['text'].apply(get_sentiment)
```

Затем применяем функцию к столбцу 'text' датасета, чтобы получить полярность настроения каждого твита.

```
def get_sentiment_label(score):
    if score > 0:
        return 'positive'
    elif score < 0:
        return 'negative'
    else:
        return 'neutral'
```

Функция присваивает метку настроения на основе оценки полярности настроения.

```
tweets['sentiment_label'] =
tweets['sentiment'].apply(get_sentiment_label)
```

Наконец, применяем функцию к столбцу 'sentiment' датасета для присвоения меток настроения.

Мы также можем применить метки к данным на основе числового порога. Например, можем разделить пользователей по количеству подписчиков на 3 категории в зависимости от количества подписчиков.

```
def get_category(value):
    if value > 100000:
        return 'high'
    elif value < 5000:
        return 'low'
    else:
        return 'medium'
```

Хотя разметка на основе правил может быть полезна для простых задач, у нее есть проблемы. В первую очередь — ограниченная гибкость. Разметка на основе правил опирается на конкретные критерии или шаблоны, она может не справиться со сложными задачами.

Кроме того, системы разметки на основе правил могут давать менее точные результаты, чем другие методы, особенно при решении задач, требующих более тонкого понимания данных. Наконец, эта система разметки может требовать постоянного обслуживания и обновления по мере появления новых данных или критериев, что может отнимать много времени и средств.

В целом, разметка на основе правил полезна для простых задач, но важно тщательно изучить конкретный случай использования и потенциальные проблемы перед внедрением.

Инструменты разметки human-in-the-loop (HITL)

Инструменты human-in-the-loop (HITL) или «человек в цикле» сочетают в себе автоматизацию машинного обучения с точностью и суждениями экспертов-людей.

Система машинного обучения сначала автоматически размечает подмножество данных, а затем эксперты-люди проверяют и уточняют эти метки, чтобы обеспечить

точность и полноту. Уточненные метки затем используют для системы машинного обучения, которую, в свою очередь, можно использовать для более точной разметки новых данных.

У инструментов human-in-the-loop есть ряд преимуществ:

1. Первоначальный автоматизированный процесс разметки позволяет сэкономить время и усилия по сравнению с ручной разметкой, поскольку система машинного обучения может быстро размечать большие объемы данных.

2. Процесс обеспечивает более высокий уровень точности и полноты разметки данных, поскольку эксперты могут уловить ошибки или нюансы, которые система машинного обучения может пропустить.

3. Уточненные метки могут использоваться для повышения точности системы машинного обучения с течением времени, поскольку система учится на основе исправлений и отзывов экспертов-людей.

Примеры инструментов разметки с участием человека:

- **Активное обучение** — метод машинного обучения, который выбирает наиболее информативные точки данных для маркировки человеком, основываясь на неопределенности или энтропии модели машинного обучения. Благодаря выбору наиболее информативных точек данных, усилия человека по маркировке могут быть сосредоточены на самых важных областях данных.

- **Полуавтоматическое обучение** (semi-supervised learning) — метод машинного обучения, который сочетает в себе маркированные и немаркированные данные. Используя как меченые, так и немеченые данные, система машинного обучения может использовать преимущества как автоматической, так и ручной разметки, при этом минимизируя количество необходимых ручных маркировок.

- **Краудсорсинг** — это метод разметки с участием человека, который предполагает передачу задач разметки большой группе работников, часто через онлайн-платформу. Краудсорсинг может быть экономически эффективным и масштабируемым способом выполнения разметки человеком, но может потребовать дополнительных мер контроля качества, чтобы обеспечить точность и последовательность.

У разметки с участием человека также есть ограничения и проблемы. Например, она может занимать много времени и требовать значительных ресурсов для реализации, особенно для больших наборов данных или сложных задач. Кроме того, качество разметки зависит от опыта людей-экспертов. Чтобы обеспечить точность, может потребоваться контроль качества.

Перейдем к примеру активного обучения. У нас есть датасет, состоящий из 25 000 рецензий на фильмы для обучения и 25 000 для тестирования. Этот датасет включает бинарную классификацию настроений, то есть либо положительную, либо отрицательную. Предположим, что 20% нашего датасета размечены человеком. Наша задача — разметить остальную часть.

Импортируем необходимые библиотеки Python.

```
from sklearn.feature_extraction.text import TfidfVectorizer
from sklearn.linear_model import LogisticRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score
from sklearn.utils import shuffle
```

Этот код импортирует несколько пакетов Python, которые будут использоваться для предварительной обработки данных, извлечения признаков, обучения модели и оценки.

Класс TfidfVectorizer — это метод извлечения текстовых признаков, который преобразует коллекцию необработанных текстовых документов в матрицу TF-IDF признаков. Полученная матрица может быть использована в качестве входных данных для алгоритма машинного обучения.

Класс LogisticRegression — это алгоритм классификации, который можно использовать для предсказания категориальных меток на основе числовых признаков. В контексте анализа настроений LogisticRegression можно использовать для предсказания положительного или отрицательного настроения рецензии на фильм на основе признаков TF-IDF, извлеченных из текста рецензии.

train_test_split — это служебная функция, которая может использоваться для разделения набора данных на обучающий и тестовый наборы для машинного обучения. Мы будем использовать эту функцию для разделения датасета на небольшую размеченную выборку и большую неразмеченную.

f1_score — это метрика для оценки эффективности алгоритма классификации. f1_score измеряет среднее гармоническое значение точности и полноты (precision и

recall) и находится в диапазоне от 0 до 1, при этом более высокие показатели свидетельствуют о более высокой эффективности.

Функция shuffle служит для перемешивания строк датасета.

Загружаем данные в pandas датафрейм.

```
reviews = pd.read_csv('train.csv', engine='python',
error_bad_lines=False)
```

engine='python' — аргумент функции read_csv(), определяющий используемый механизм синтаксического анализа. По умолчанию функция read_csv() использует механизм C, который быстрее, но может не справиться с некоторыми типами данных или кодировками. В этом случае мы указываем механизм Python, чтобы гарантировать, что парсинг CSV-файла будет выполнен правильно.

error_bad_lines=False определяет, как обрабатывать ошибки парсинга. По умолчанию функция read_csv() выдает ошибку и прекращает парсинг, когда встречает в CSV-файле неправильно оформленную строку. В этом случае мы устанавливаем параметр error_bad_lines в False, чтобы игнорировать неправильно сформированные строки и продолжить парсинг файла.

```
reviews = shuffle(reviews)
```

Этот код перемешивает строки датасета случайным образом. Это важно для обеспечения того, чтобы обучающие и тестовые данные представляли общее распределение набора данных.

```
labeled_reviews, unlabeled_reviews = train_test_split(reviews,
train_size=0.2, random_state=42)
```

Разбиваем датасет на две выборки: небольшой размеченный набор, содержащий 20% данных, и большой «неразмеченный» набор, содержащий остальные 80% данных. На самом деле он тоже размечен, но для учебных целей и решения нашей задачи предположим, что разметки нет.

```
def train_model(labeled_reviews):
    # Векторизация текстовых данных с помощью TF-IDF
    vectorizer = TfidfVectorizer()
    X = vectorizer.fit_transform(labeled_reviews['text'])
    y = labeled_reviews['sentiment']
```

```
    # Обучение модели логистической регрессии на размеченных
данных
    model = LogisticRegression()
    model.fit(X, y)

    return model, vectorizer
```

Функция train_model() принимает на вход размеченный набор данных рецензий на фильмы, векторизует текстовые данные с помощью метода TF-IDF и обучает модель логистической регрессии. Обученная модель и векторизатор возвращаются в качестве выходных данных.

Когда мы говорим о «векторизации текстовых данных с помощью метода TF-IDF», мы, по сути, преобразуем коллекцию необработанных текстовых документов в матрицу числовых характеристик, которые могут быть использованы в качестве входных данных для алгоритма машинного обучения.

Расшифровка аббревиатуры TF-IDF — Term Frequency-Inverse Document Frequency. Это числовое представление важности каждого слова в документе или коллекции документов:

- **Частота термина (TF)** слова в документе рассчитывается как количество раз, когда слово встречается в документе, деленное на общее количество слов в документе. Дает представление о том, как часто слово встречается в документе.
- **Обратная частота документа (IDF)** слова рассчитывается как логарифм общего количества документов в коллекции, деленный на количество документов, содержащих данное слово. Дает представление о том, насколько часто встречается слово во всех документах коллекции.

Показатель TF-IDF слова в документе рассчитывается как произведение частоты термина и обратной частоты документа. Он отражает важность слова в контексте документа и коллекции документов.

Таким образом, когда мы векторизуем текстовые данные с помощью метода TF-IDF, мы, по сути, преобразуем каждый документ коллекции в числовой вектор, где каждый элемент вектора представляет собой оценку TF-IDF определенного слова в документе. Это позволяет нам представить текстовые данные в виде матрицы числовых характеристик, которые можно использовать в качестве входных данных для алгоритма машинного обучения.

```
model, vectorizer = train_model(labeled_reviews)
```

Код вызывает функцию train_model() на небольшом наборе данных с метками и сохраняет обученную модель и векторизатор в переменных model и vectorizer соответственно.

```
X_unlabeled = vectorizer.transform(unlabeled_reviews['text'])
y_unlabeled_predicted = model.predict(X_unlabeled)
```

Код использует обученную модель и векторизатор для прогнозирования настроения неразмеченных данных. Текстовые данные преобразуются в вектор TF-IDF с помощью обученного векторизатора, а модель логистической регрессии используется для прогнозирования меток настроения.

```
y_unlabeled_proba = model.predict_proba(X_unlabeled)
uncertainty = -(y_unlabeled_proba * np.log2(y_unlabeled_proba)).sum(axis=1)
```

Неопределенность или энтропия прогноза модели для точки данных — это мера того, насколько модель уверена в своем прогнозе.

В контексте активного обучения нас интересует вычисление неопределенности или энтропии предсказаний модели для каждой точки данных в неразмеченном наборе. Это позволяет нам определить, в отношении каких точек данных модель наиболее неопределенна, и выбрать эти точки данных для маркировки человеком.

Если модель очень уверена в своем предсказании, предсказанные вероятности будут очень высокими для одной метки класса и очень низкими для другой метки класса, что приведет к низкому значению энтропии. Если модель не уверена в своем прогнозе, предсказанные вероятности будут более равномерно распределены между двумя метками класса, что приведет к более высокому значению энтропии.

```
labeled_reviews_new = unlabeled_reviews.iloc[uncertainty.argsort()[:100]]
unlabeled_reviews_new = unlabeled_reviews.iloc[uncertainty.argsort()[100:]]
```

Выбираем 100 самых неопределенных точек данных, разметку которых нужно будет выполнить вручную. Функция argsort() из NumPy используется для сортировки точек данных по неопределенности, а функция iloc[] из Pandas — для выбора 100 лучших и остальных точек данных.

```
labeled_reviews = pd.concat([labeled_reviews, labeled_reviews_new])
```

После того как человек-эксперт выполнил ручную разметку данных, мы можем объединить новые размеченные точки данных с существующим размеченным набором и сохранить объединенный набор в переменной labeled_reviews.

```
model, vectorizer = train_model(labeled_reviews)
```

Переобучаем модель на обновленном датасете.

```
reviews_test = pd.read_csv('test.csv', engine='python', error_bad_lines=False)
reviews_test['sentiment'] = reviews_test['sentiment'].replace({'neg': 0, 'pos':
1})
X_test = vectorizer.transform(reviews_test['text'])
y_test_predicted = model.predict(X_test)
f1 = f1_score(reviews_test['sentiment'], y_test_predicted)

print(f1)
```

Наконец, проверим производительность модели на тестовом датасете.

Используем обученную модель и векторизатор для предсказания меток настроения для тестового датасета рецензий на фильмы и вычислим F1-score для предсказаний модели. Показатель F1 является мерой точности и полноты (precision и recall) модели и варьируется от 0 до 1, при этом более высокие показатели свидетельствуют о более высокой эффективности.

Мы получили значение 0.85. Это значит, что модель анализа настроения достигла относительно высокого уровня точности в предсказании настроения рецензий на фильмы. Однако важно понимать, что производительность модели анализа настроений может варьироваться в зависимости от качества и разнообразия датасета, количества размеченных вручную данных, а также конкретного алгоритма машинного обучения и выбранных гиперпараметров. Поэтому важно тщательно оценить производительность модели с помощью нескольких метрик и методов, а также протестировать ее на множестве различных наборов данных и сценариев.

Мы можем итеративно продолжать весь процесс, выбирая наиболее неопределенные точки данных для маркировки человеком и переобучая модель на расширенном маркированном наборе. Таким образом можно постепенно улучшить производительность модели, при этом минимизируя количество человеческих трудозатрат. Этот подход может быть особенно полезен для больших наборов данных, где ручная маркировка отнимает много времени и стоит дорого.

Популярные инструменты разметки данных

Существует несколько инструментов разметки данных. Дата-инженеры и специалисты по анализу данных могут использовать их, чтобы размечать большие датасеты для машинного обучения. Эти инструменты могут помочь автоматизировать разметку, уменьшить количество ошибок и повысить эффективность процесса. Перечислим некоторые из них.

Label Studio — инструмент разметки данных с открытым исходным кодом. Поддерживает широкий спектр форматов данных, включая текст, изображения и аудио. Предлагает удобный интерфейс для создания задач маркировки. Поддерживает разные типы разметки, такие как классификация текста, распознавание сущностей и сегментация изображений. Предлагает мощную систему плагинов, которая обеспечивает легкую настройку и интеграцию с другими инструментами.

Amazon SageMaker Ground Truth — сервис разметки данных. Предоставляет платформу для создания высококачественных размеченных датасетов для машинного обучения. Поддерживает разные форматы данных и типы аннотаций, а также предлагает встроенные функции контроля качества для точности разметки.

Hive Data Annotations — веб-инструмент для разметки данных. Поддерживает текстовые, графические и видеоданные. Предлагает простой и интуитивно понятный интерфейс для создания задач разметки и поддерживает разные типы разметки, такие как анализ настроения, классификация изображений и обнаружение объектов. Hive Data Annotations также предлагает функции совместной работы, такие как назначение и проверка заданий.

Supervisely — платформа разметки данных на основе ИИ. Поддерживает разные форматы данных, включая изображения, видео и облака точек. Предлагает ряд инструментов разметки, таких как детекция, сегментация и отслеживание объектов. Поддерживает как ручную, так и полуавтоматическую разметку. Предлагает интеграцию с другими инструментами машинного обучения, такими как TensorFlow и PyTorch.

Prodigy — это инструмент разметки данных с удобным интерфейсом. Предлагает ряд типов разметки, включая классификацию текста, распознавание сущностей и разметку изображений. Поддерживает как ручную, так и полуавтоматическую разметку. Предлагает интеграцию с популярными фреймворками машинного обучения, такими как scikit-learn.

Мы назвали далеко не все инструменты. У каждого из них есть свои сильные и слабые стороны. Выбор может быть непростым, потому что зависит от потребностей, требований задачи разметки данных и других факторов. Вот несколько советов, которые помогут вам выбрать подходящий инструмент:

1. **Определите требования к разметке.** Учитывайте тип данных (текст, изображение, видео и так далее), типы аннотаций, которые вам нужны (классификация, сегментация и так далее), размер датасета и уровень автоматизации, который требуется.
2. **Оцените возможности инструмента.** Ищите функции, которые имеют отношение к вашему проекту, такие как поддержка нескольких типов аннотаций, интеграция с системами машинного обучения и инструменты для совместной работы. Учитывайте простоту инструмента, качество пользовательского интерфейса и масштабируемость для больших датасетов.
3. **Рассмотрите стоимость и модель лицензирования.** Некоторые инструменты могут требовать абонентскую плату, другие предлагают модель оплаты за использование. Учитывайте бюджет и экономическую эффективность каждого инструмента, принимайте во внимание размер датасета и уровень автоматизации, который требуется.
4. **Обратите внимание на поддержку сообщества и документацию.** Ищите инструменты с активным сообществом пользователей, онлайн-форумами и документацией, которую легко понять. Они будут полезны, если возникнут проблемы или вопросы по инструменту.
5. **Попробуйте несколько инструментов, прежде чем покупать.** Многие предлагают бесплатные пробные версии, чтобы протестировать возможности и пользовательский интерфейс перед покупкой. Вы сможете получить представление об инструменте и убедиться, что он отвечает вашим требованиям.

Стратегии, которые помогут повысить точность и эффективность разметки:

1. **Определите четкие правила разметки.** В них должны быть указаны типы аннотаций, категории и критерии для каждой задачи. Их нужно передать всем специалистам по разметке и убедиться, что все работают на основе одного и того же набора рекомендаций. Такой подход поможет сократить количество ошибок и несоответствий.
2. **Используйте несколько специалистов**, чтобы повысить точность и надежность разметки. Можно использовать голосование или межгрупповое

соглашение для точности окончательной маркировки. Подход особенно актуален в медицинских задачах, когда разные специалисты могут по-разному оценить один и тот же рентгеновский снимок.

3. **Используйте методы предварительной разметки**, такие как разметка на основе правил или автоматическая разметка. Они могут помочь сократить объем ручной разметки.

4. **Мониторинг и оценка производительности специалиста по разметке.** Наконец, важно отслеживать и оценивать работу маркировщиков, чтобы убедиться, что они точно и последовательно размечают данные. Контролируя работу маркировщиков, давая обратную связь и обучая их по мере необходимости, вы сможете обеспечить точность и эффективность разметки.

Работа в Label Studio

Перейдем к практике в одном из сервисов разметки данных — Label Studio.

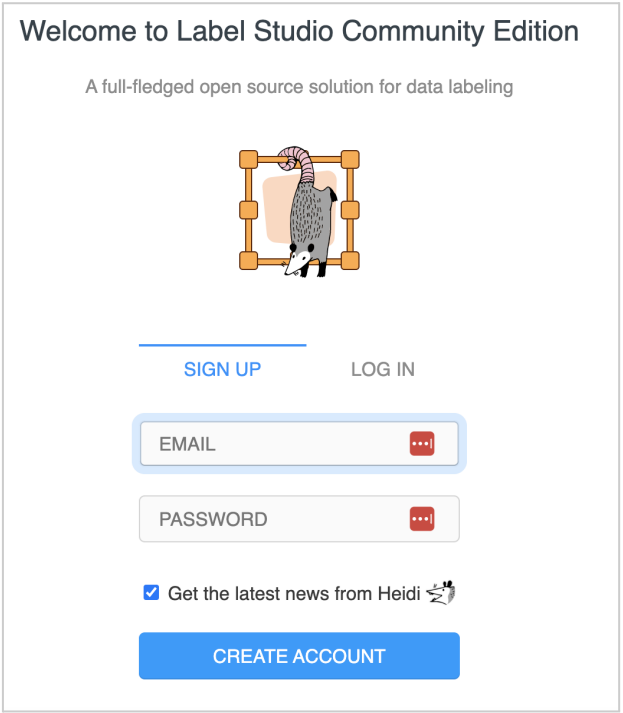
Сначала нужно установить Label Studio на компьютер. Для установки с помощью pip, нужен Python версии 3.7 или более поздней. В командной строке выполним команду:

```
pip install label-studio
```

После установки Label Studio запускаем сервер с помощью команды:

```
label-studio
```

Веб-браузер по умолчанию автоматически откроется на локальном хосте http://localhost:8080 вместе с Label Studio.



Далее нужно зарегистрироваться и войти в аккаунт.

После установки Label Studio мы можем создать новый проект разметки для анализа настроений. Для этого нажимаем кнопку Create project на главной странице Label Studio.

Вводим имя проекта. На вкладке Data import загружаем датасет tweets.csv и выбираем Treat CSV/TSV as List of tasks.

Затем переходим на вкладку Labeling setup и выбираем в левом меню Natural Language Processing, а затем Text classification. Для анализа настроения вам нужно будет отнести каждую точку текстовых данных к одной из предопределенных категорий, то есть Positive, Negative, Neutral.

После определения задачи разметки можно приступить к разметке текстовых данных. Для этого нажимаем первую задачу (первую строку в таблице) и начинаем классифицировать каждую точку текстовых данных в соответствии с заданными категориями.

После разметки текстовых данных можно просмотреть и экспортировать размеченные данные. Для этого нажимаем на кнопку Export и выбираем формат экспорта, например CSV, JSON или XML.

Таким образом, Label Studio позволяет упростить разметку данных, а также обеспечивает контроль качества размеченных данных и их пригодность для моделей машинного обучения.

Заключение

Инструменты разметки данных играют важную роль в области машинного обучения, позволяют эффективно и точно размечать данные для моделей.

Существует множество инструментов — от разметки, основанной на правилах, до разметки с участием человека и активного обучения. Выбор правильного инструмента и внедрение лучших практик — ключ к высококачественным результатам.

В будущем инструменты разметки данных, вероятно, будут развиваться и совершенствоваться. Появятся новые методы и технологии, которые позволят более эффективно и точно размечать данные. С ростом доступности крупномасштабных датасетов и передовых алгоритмов машинного обучения инструменты разметки будут играть важную роль в разработке более мощных и сложных моделей машинного обучения, способных решать широкий спектр реальных задач.

Мы подошли к завершению курса по сбору и разметке данных. В течение цикла лекций мы многому научились. Мы изучили методы и технологии работы с данными: от API до веб-скрейпинга. Проделали большую практическую работу. Полученные знания вы сможете применить в различных аспектах своей работы.

Сбор и разметка данных — это не просто техническая задача, а важнейшая часть создания этических и ответственных приложений искусственного интеллекта. Спасибо, что присоединились к нам в этом путешествии. Желаем вам всего наилучшего в будущих начинаниях!

Что можно почитать еще?

1. 5 подходов к разметке данных для проектов машинного обучения
2. Оптимизация разметки данных с помощью активного обучения
3. Как избавиться от проблем при разметке данных для обучения ML моделей?
4. Бросить #ВызовРадиологу: как мы организовали процесс разметки медицинских данных для обучения ИИ