Подпишитесь на DeepL Pro и переводите документы большего объема.
Подробнее на www.DeepL.com/pro.

Введение в машинное обучение
Задание 1

Бакалавры

Весна

2024

1 Общие инструкции

Формат представления. Вы должны представить свои решения через Moodle в виде одного zip-файла. zip-архив должен содержать файл Ipynb, все файлы данных в формате csv и модель из задания 1 с именем **poly_ optimized model.sav**. Пожалуйста, укажите ваше имя и электронную почту на in- nopolis.university в качестве первой строки в блокноте. Исходный код должен быть чистым, легко читаемым и хорошо документированным - **плохой стиль кода будет наказываться**, подробности смотрите в головной ячейке тетради. Все строки, которые вы должны заполнить, помечены комментариями, начинающимися с *# TODO Напишите свой код здесь*.

Баллы. Процент рядом с разделами и подразделами задания обозначает максимальный процент баллов за задание, который вы можете набрать. За элегантные решения могут быть начислены бонусные баллы, однако эти баллы смогут лишь отменить эффект штрафов. Вам предстоит решить 2 задачи, которые в лучшем случае принесут вам максимальный балл. Мы также предоставляем бонусную задачу (+10 баллов) в качестве подушки безопасности, если вы не уверены в своих основных решениях. Обратите внимание, что вы не можете получить более 100 баллов.

Политика в отношении плагиата. Плагиатные решения будут строго наказываться для всех участников. Также не следует просто копировать и вставлять решения из Интернета. Вам разрешается работать над общими идеями с другими студентами, а также обращаться к книгам и интернет-ресурсам. Однако не забывайте указывать источники, которые вы используете, и набирать весь код и документацию самостоятельно.

2 Отказ от ответственности

В этом задании вы должны продемонстрировать понимание и освоение материалов, изученных за первые 4 недели. Задание включает в себя очистку и предварительную обработку данных, обучение классических алгоритмов машинного обучения, регрессию, классификацию, настройку гиперпараметров, визуализацию функций и т. д.
Возможно, описания заданий покажутся вам слишком четкими и подробными, но это сделано специально: вы должны продемонстрировать, что можете следовать конкретным инструкциям

для решения задачи машинного обучения. Будьте готовы к тому, что следующее задание будет более сложным и открытым.

Поскольку здесь много клеток и инструкций, которые вам нужно выполнить, легко пропустить некоторые подзадания, например, объяснение своих наблюдений. Пожалуйста, будьте внимательны, когда читаете описания, пропущенные ответы будут считаться нулевыми.

3 Задание 1 (50 %)

В задании 1 вы должны решить задачу многофакторной регрессии путем обучения модели линейной регрессии из sklearn. Предполагается, что вы знакомы с основными понятиями алгоритма ML, такими как обучение, оценка производительности, перекрестная валидация, недоучет и переучет, настройка гиперпараметров и визуализация признаков.

3.1 Линейная регрессия (10 %)

Набор данных хранится в файле *train 1.csv* и содержит 4 столбца признаков (X_{1-4}) и одну метку (y). Природа столбцов неизвестна, считайте, что это нелинейная функция. Этот набор данных не требует предварительной обработки, однако вы должны разделить его на обучающую и проверочную части.

1. Загрузите набор данных из файла train 1.csv в виде Pandas DataFrame.
2. Разделите и перемешайте на обучающие (80% всего набора данных) и проверочные (20% всего набора данных) кадры данных.
3. Используя **модель линейной регрессии** из sklearn, подгоните ее к обучающему набору данных.
4. Проведите оценку на валидационном наборе данных, выведя значения MSE, RMSE, MAE и R2.

3.2 Настройка полиномиальной регрессии - (25%)

Давайте попробуем применить модель полиномиальной регрессии к тому же набору данных и сравним производительность. В полиномиальной регрессии вы будете иметь дело с 1 гиперпараметром на признак - степенью полинома. Поскольку мы не знаем, какая степень является правильной, нам придется подстраивать это значение с помощью алгоритма Grid Search.
В этой подзадаче ваш максимальный балл зависит от оценки **закрытого** набора тестовых данных, который вам не выдается. Мы гарантируем, что тестовые данные выбраны случайным образом из того же распределения, что и обучающие. Распределение баллов:

- RMSE > 65 : 2 балла;
- 65 ≥ RMSE > 20 : 5 баллов;

- 20 ≥ RMSE > 5 : 7 баллов;

- 5 ≥ RMSE > 1,4 : 9 баллов;
- RMSE ≤ 1,4 : 10 баллов.

Вы должны сделать следующее:

1. Используя конвейер sklearn, постройте конвейер полиномиальной регрессии, состоящий из (1) класса полиномиальных признаков и (2) класса линейной регрессии. Используйте "степень=2" в полиноме.
2. Используйте GridSearch для поиска лучшей модели полиномиальной регрессии и выведите лучшие параметры. Установите *degrees = range(2, 6), cross-validation=8, scoring ='negative mean squared error'*, используя лучшие параметры, предскажите на "X test" и оцените с помощью MSE, RMSE, MAE и R2 score.
3. **Сохраните модель,** не меняя имени файла (poly optimized model.sav). Эта модель будет использоваться для расчета баллов в этом задании.

3.3 Определите линейные зависимые признаки (15%)

Алгоритмы регрессии основаны на предположении о независимости признаков. Поскольку у нас есть несколько признаков, интересно определить, *коррелируют* ли *некоторые из них*. Набор данных содержит как минимум одну такую линейно зависимую пару, и в этом задании вам предлагается определить ее и объяснить свой выбор. Мы рекомендуем использовать визуализацию признаков (график PairGrid).

4 Задание 2 (50 %)

В этом задании вам предстоит решить задачу бинарной классификации. Вам предстоит классифицировать покемонов, являются ли они легендарными покемонами или нет. Помимо классификации, вы должны быть знакомы с понятиями предварительной обработки данных, метриками классификации, регуляризацией и т. д.
Информация о наборе данных (pokemon_modified.csv):

- В 36 колонках представлены характеристики покемонов.
- В наборе данных 801 строка, каждая строка кодирует покемона.
- В колонке "Легендарный" указано, является ли покемон легендарным или нет.

- В 3 столбцах характеристик отсутствуют некоторые значения.
- Существует 2 бесполезных для классификации столбца признаков.
- Существует 1 колонка признаков, которые должны быть категорически закодированы.

4.1 Предварительная обработка данных (20 %)

В этом разделе вы должны загрузить данные, проанализировать признаки, решить, какие признаки следует удалить, какие закодировать, а какие вменить.

1. Загрузите набор данных pokemon с помощью фрейма данных pandas.
2. Исследуйте набор данных с помощью pd.head(), pd.info(). Удалите 2 лишних признака и объясните, почему они должны быть удалены.
3. Разделите набор данных на train/test с соотношением 0,8/0,2. Сбалансирован ли набор данных по классам?
4. Проверьте недостающие значения и впишите их с помощью SimpleImputer. Вы можете использовать стратегии *среднего* или *наиболее частого значения*.
5. Дважды проверьте, нет ли пропущенных значений.
6. Определите и закодируйте категориальный признак с помощью кодировщика OneHot.
7. Масштабируйте данные с помощью нормализации MinMax или StandardScaler.
8. Постройте корреляционную матрицу. Ответьте на следующие вопросы: Есть ли в наборе данных сильно коррелированные признаки? Является ли это проблемой? При необходимости выполните предварительную обработку данных.

4.2 Подгонка и сравнение моделей - (30 %)

Вам нужно обучить и оценить несколько моделей классификации из sklearn:

- Классификатор логистической регрессии;
- Классификатор KNN;
- Гауссовский классификатор Naive-Bayes.

Для логистической регрессии и KNN существуют гиперпараметры, которые нам нужно настроить с помощью Grid-Search: обратная сила регуляризации *C*, техника ре-гуляризации и решатель.
Гиперпараметрами KNN являются значение K - количество соседей, метрика - функция вычисления расстояния и вес - присваивать ли всем соседним точкам одинаковый вес или определять вес в

зависимости от расстояния.

Классификатор Naive-Bayes не требует настройки гиперпараметров. Следуйте инструкциям:

1. Используя GridSearchCV, найдите наилучшие гиперпараметры для модели логистической регрессии. Попробуйте различные варианты со *штрафом: ['l1','l2']; C: np.logspace(-3,3,7); solver: ['newton-cg', 'lbfgs', 'liblinear'], scoring: 'f1'*.
2. Оцените модель логистической регрессии с наилучшими параметрами на тестовых данных с помощью точности, прецизионности, запоминания и F1 score.
3. Выведите коэффициенты регрессии и найдите названия 5 наиболее влиятельных признаков и 5 наиболее игнорируемых признаков.
4. Используя GridSearchCV, найдите наилучший гиперпараметр для модели KNN. Попробуйте различные варианты с *k :list(range(1, 15)), весами:['uniform', 'distance'], метрикой:['euclidean', 'manhattan', 'chebyshev', 'cosine']*.
5. Оцените модель KNN с наилучшими параметрами на тестовых данных, используя точность, прецизионность, отзыв и F1 score.
6. Подгоните Gaussian Naive-Bayes к данным и оцените их на тестовом наборе данных, выводя метрики.
7. Какая метрика наиболее подходит для этой задачи и почему? Объясните, какой вид ошибки (ложноотрицательная или ложноположительная) более критичен для этого набора данных.
8. Сравните 3 классификатора с точки зрения точности, прецизионности, запоминания и F1-score. Какая модель лучше всего подходит для этой задачи? Объясните, почему.

5 Бонусное задание (10 %)

Задание "Бонус" преследует две цели: (а) облегчить дальнейшее обучение и (б) компенсировать возможные ошибки, допущенные при выполнении предыдущих заданий. Мы ожидаем, что вы будете искать информацию по этой теме самостоятельно. Ранее вы реализовали бинарный классификатор для определения того, является ли покемон легендарным или нет. В этом задании вам предстоит сделать еще один шаг вперед и решить задачу многоклассовой классификации. Есть две техники, с помощью которых можно решить эту задачу: One-vs.-Rest и Multinomial classifier.

Набор данных содержит 3 признака и 1 переменную результата с 3 классами. Набор данных является чистым и предварительно обработанным, в нем нет пропущенных значений.

1. Загрузите данные из файлов bonus train.csv и bonus test.csv с помощью фрейма данных pandas, затем разделите обучающие данные на X train и y train, тестовые данные разделите на X test и y test.

2. Используя библиотеку seaborn, постройте график обучающих данных с помощью функции pairplot.
   kind = "scatter" и *hue = "target"*.
3. Используя модель логистической регрессии из sklearn, подгоните модель к обучающим данным, сначала с помощью мультиномиального метода, а затем с помощью метода one-vs.-rest.
4. Оцените модели логистической регрессии с помощью средней точности на тестовом наборе данных.
5. Используйте GridSearch для настройки значения C логистической регрессии и мультикласса. Используйте *C: np.logspace(-10, 10,7), multi class : ['multino- mial', 'ovr']*.
6. Прокомментируйте, почему один из них лучше другого.
7. И наконец, визуализируйте границу принятия решения модели, показавшей наилучшие результаты на обучающем наборе данных, в 2D. Подсказка: подгоните модель только по двум признакам и постройте график.