УДК 004.622
МРНТИ 28.23.25

ОПТИМИЗАЦИЯ АЛГОРИТМОВ ПОДГОТОВКИ ДАННЫХ ДЛЯ ОБУЧЕНИЯ И ПРИМЕНЕНИЯ МОДЕЛЕЙ МАШИННОГО ОБУЧЕНИЯ НА ЯЗЫКЕ PYTHON

ДЖИЛИКБАЕВ М., АКЖАЛОВА А.

Казахстанско-Британский технический университет

Абстракт: *В данной работе рассмотрена целесообразность применения различных алгоритмов подготовки данных для более качественного обучения модели на языке программирования python3. Рассмотрены способы взаимодействия с отсутствующими значениями в наборе данных и способы их устранения в зависимости от различных факторов. Рассмотрены алгоритмы преобразования номинальных переменных в вид, пригодный для обучения моделей библиотеки Scikit-Learn. Также рассмотрен способ комбинирования алгоритмов преобразования данных для достижения наивысшей предиктивной способности по F1 мере на примере модели бинарной классификации.*

Ключевые слова: *подготовка данных, python3, one-hot-encoding, предиктивная способность*

PYTHON МАШИНАЛЫҚ ОҚУ МОДЕЛЬДЕРІН ОҚЫТУ ЖӘНЕ ҚОЛДАНУ ҮШІН МӘЛІМЕТТЕР ДАЙЫНДАУ АЛГОРИТМДЕРІН ОҢТАЙЛАНДЫРУ

Аңдатпа: *Бұл мақалада, python3 бағдарламалау тіліндегі моделін неғұрлым сапалы үйрету үшін әртүрлі деректер дайындау алгоритмдерін пайдаланудың техникалық-экономикалық негіздемесі саналады. Деректер жиынтығында жетіспейтін шамалармен өзара әрекеттесу әдістері және әрқилы факторларға байланысты оларды жою әдістері қарастырылған. Номиналды айнымалыларды Scikit-Learn код жиынтығындағы модельдерін оқытуға қолайлы формаға түрлендірудің алгоритмдері ұсынылған. Сондай-ақ бинарлық классификация моделінің мысалында F1 өлшеміндегі ең жоғары болжамды қабілетке жету үшін мәліметтерді түрлендіру алгоритмдерін біріктіру әдісі беріледі.*

Түйінді сөздер: *деректерді дайындау, python3, one-hot-encoding, болжау қабілеті*

OPTIMIZATION OF DATA PREPARATION ALGORITHMS FOR TRAINING AND APPLICATION OF MACHINE LEARNING MODELS IN PYTHON

Abstract: *In this paper, the feasibility of using various data preparation algorithms for better training of the model in the python3 programming language is considered. We describe how to interact with missing values in the data set and how to eliminate them, depending on various factors. Algorithms for converting nominal variables into a form suitable for teaching models of the Scikit-Learn library are considered. Also, a method of combining data conversion algorithms to achieve the highest predictive ability in F1 measure was applied using the example of a binary classification model.*

Key words: *data preparation, python3, one-hot-encoding, predictive ability*

Введение

Подготовка данных для обучения моделей и их использования является настолько же важным шагом машинного обучения, как и построение самой модели [3-5]. Существует множество моделей и способов их улучшения, в то время как действенных алгоритмов преобразования данных в десятки раз меньше, хотя, стоит заметить, что, имея качественно подготовленный набор данных часто для получения относительно хорошего результата, оказывается достаточным построение даже самой простейшей модели. Большинство открытых к бесплатному использованию python библиотек с моделями машинного обучения, такие как Scikit-Learn, Keras, Theano и др., требуют для работы только числовые входные данные, в которых не должны содержаться отсутствующие значения. Однако наборы данных задач реального мира в подавляющем большинстве содержат как столбцы с отсутствующими значениями, так и столбцы с номинальными или численно зашифрованными данными. В данной статье рассмотрены способы приведения данных к необходимому для обучения модели виду и влияние различных алгоритмов и допущений при подготовке данных на конечный результат (предсказательную способность) модели.

Оптимизация методов подготовки данных для максимизации F1_score

Одним из действенных способов исключения отсутствующих значений из набора данных является удаление строк, в которых встречается хотя бы одно отсутствующее значение. Если строку данных представлять как вектор X = [x1, x2, … xi], то удалению подлежат все строки X, удовлетворяющие условию $\forall x \in X, x = \emptyset$. Этот метод хорошо показывает себя на небольших наборах данных, в которых отсутствующие переменные являются следствием человеческой ошибки или кратковременных и незначительных сбоев в логировании данных. Напротив, применяя этот метод к витрине данных с большим количеством столбцов или со значениями, в которых отсутствие значения как таковое несёт некий смысл, высока вероятность потерять значимую часть данных и существенно исказить предсказательную способность модели. Следовательно, использовать этот метод необходимо с учётом доли потери данных как в общем так и в разрезе целевой переменной:

$$\min\left(\frac{X_{\text{удалённых}}}{X_{\text{всего}}}, \frac{X_{\text{удалённых}}}{X_{\text{таргетных}}}, \frac{X_{\text{удалённых}}}{X_{\text{нетаргетных}}}\right) \leq M,$$

где M – максимально допустимая доля потери в данных.

В случае, когда удаление строк с отсутствующими значениями невозможно, принято производить замену пустых значений нулями, средним значением по столбцу или изначально согласованным фиксированным значением для каждой переменной [7-8]. Используя этот подход в большинстве случаев удаётся избавиться от пропусков с незначительными потерями информативности. Недостатком данного метода является необходимость изначально указать способ замены пустых значений для каждой переменной. Многие выбирают простой подход и заменяют все пропуски нулями, но это не всегда корректно. Так отсутствующий возраст, замененный нулём, автоматически запишет всех людей с отсутствующим возрастом в наборе данных в разряд новорожденных в понимании модели и результат расчетов будет иррелевантен данным. Логичнее заменить пропуски возраста средним значением, что в большинстве случаев даст намного меньшее отклонение при обучении модели. Другим используемым методом является создание отдельного столбца, в котором отсутствие значения помечается единицей, а присутствие нулём. Это целесообразно делать в случае, когда пропуск означает что-то конкретное, к примеру отсутствие даты открытия определённого продукта свидетельствует о том, что у клиента отсутствует данный продукт.

Помимо устранения отсутствующих значений набору данных может потребоваться преобразование номинальных переменных. Для этого чаще всего используется one-hot-encoding. Смысл данного метода заключает-

ся в замене признака, принимающего N различных значений, на N признаков, один из которых принимает значение True, тогда как остальные принимают значение False в зависимости от значения преобразуемого признака. Использование данного подхода решает проблему отсутствующих значений т.к. пропуски можно представить, как одно из значений признака. Таким образом признак, имеющий N различных значений и пропуски будет заменён на N+1 признак, имеющие значения True или False. Недостатком данного метода является чрезмерное увеличение размерности изначального набора данных при наличии номинальных признаков с большим количеством различных значений N, которые не являются значимыми.

Для описанных методов нет единого эффективного сценария использования. Максимальную эффективность будет показывать определённая комбинация методов, которая будет каждый раз изменяться в зависимости от полноты набора данных, специфики логирования, плотности распределения признаков и других особенностей поставленной задачи. В данной статье рассмотрен алгоритм подготовки данных для задачи бинарной классификации. Эффективность модели оценивается мерой F1, которая вычисляется по формуле [1]

$$precision = \frac{TP}{TP + FP}$$

$$recall = \frac{TP}{TP + FN}$$

$$F1 = \frac{2 * precision * recall}{precision + recall}$$

где TP (True Positive) – правильно распознанные моделью целевые строки;

FP (False Positive) – ошибочно названные целевыми нецелевые строки;

FN (False Negative) – это ошибочно названные нецелевыми целевые строки;

precision (точность) – доля верно определённых целевых строк из всех определённых строк [2];

recall (охват, полнота) – доля верно определённых моделью целевых строк из всех целевых строк в наборе данных.

Набор данных и способ построения модели остаётся неизменным чтобы исключить факт влияния этих составляющих на предсказательную силу модели. Изначальный датасет (ДС) состоит из 200 000 строк и 505 столбцов: 504 признаков из которых 372 численных, 25 бинарных, 107 номинальных и 1 целевая переменная, представленная бинарным значением. Количество строк целевой переменной = 33 456, концентрация целевой переменной 16.7%. Из набора данных выделяется 70% целевых и нецелевых строк для обучения модели. С оставшимися данными алгоритм производит сверку и получает значение F1 метрики.

Для определения эффективности различных комбинаций методов был выбран подход Grid Search [6]. Данный подход позволяет построить дискретное n-мерное пространство по степеням свободы каждого используемого метода (из n методов) и получить значение F1 score модели для каждой комбинации.

Метод замены отсутствующих значений имеет одну бинарную степень свободы – использовать умную замену или нет (*advanced_missing_values_method*). Если было решено не использовать умную замену пропусков, то алгоритм просто заменит все отсутствующие значения нулями. в обратном случае произойдёт преобразование в соответствии со списком со значениями преобразования для каждой переменной:

1. Заменить отсутствие на 0.
2. Заменить на среднее значение по столбцу.
3. Заменить на 0 и создать отдельный столбец.

Метод преобразования номинальных переменных в численные, использующий алгоритм one-hot-encoding, имеет три степени свободы:

1. Минимальная доля неотсутствующих значений в переменной (not_missing_values_conditions).

2. Максимальное количество состояний номинальной переменной (max_value_ counts_ conditions).

3. Минимальная учитываемая концентрация состояния (min_concentration_per_value_ conditions).

Если процент отсутствия значений в переменной превышает порог «минимальной доли неотсутствующих значений», то алгоритм не будет преобразовывать эту номинальную переменную. Это необходимо делать для исключения номинальных строк с большим содержанием отсутствующих значений. Следующая степень свободы метода «максимальное количество состояний номинальной переменной» указывает какое количество значений будет учитываться как отдельное состояние (по убыванию концентрации). Таким образом в переменной останутся только N самых часто встречающихся признаков, а остальные будут преобразованы в состояние «others». Это сделано для того, чтобы из одной номинальной переменной путём преобразования образовывалось максимум N переменных состояния. Если указать слишком большое значение для N, есть риск образования большого количества переменных и превышения вычислительных мощностей компьютера при обучении модели. Заключительная степень свободы «минимальная учитываемая концентрация состояния» данного метода обозначает порог минимальной концентрация определённого состояния номинальной переменной относительно других состояний, не учитывая состояние «пустое значение». Редко встречающиеся переменные будут преобразованы в состояние «others». Такое преобразование целесообразно делать, чтобы модель не переобучалась на частных случаях.

Для симуляции метода grid search использовались следующие состояния для каждой степени свободы:

advanced_missing_values_method_conditions = [True, False]
not_missing_values_conditions = [0.01, 0.05, 0.1, 0.2, 0.3, 0.5]
max_value_counts_conditions = [1, 5, 10, 15, 20, 25]
min_concentration_per_value_conditions = [0.01, 0.05, 0.1, 0.3, 0.5]

Сам метод Grid Search реализован вложенными функциями for:

```
for advanced_missing_values_method in advanced_missing_values_method_conditions:
  for not_missing_values in not_missing_values_conditions:
    for max_value_counts in max_value_counts_conditions:
      for min_concentration_per_value in min_concentration_per_value_conditions:
```

В каждую отработку происходит преобразование набора данных, обучение модели, сравнение предсказанных моделью данных с фактическими данными и запись используемых состояний с предсказательной способностью модели. Как видно из рисунка 1, подавляющее большинство комбинаций состояний дало результат $F1 \approx 0.52$

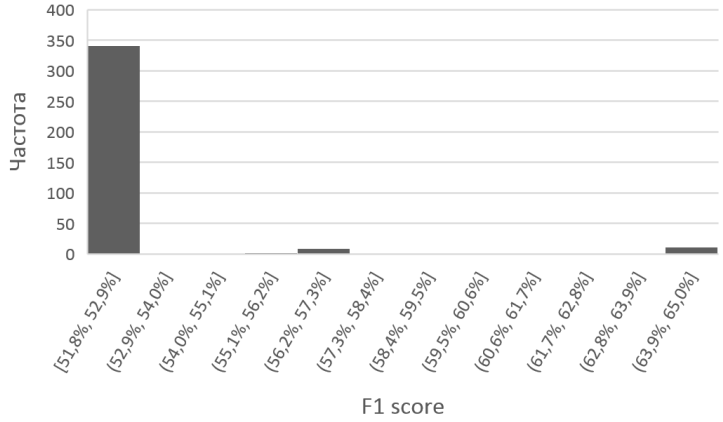

Рис. 1 – Распределение комбинаций состояний по значениям F1_score

Однако удалось найти такие комбинации состояний, у которых значение F1 заметно выше - 0.55, 0.57 и даже 0.64. Это говорит о том, что при определённых комбинациях состояний степеней свободы вышеназванных алгоритмов данные удаётся преобразовать более информативным для обучения модели образом.

Результаты и их обсуждение

Как видно из таблицы 1 для достижения прироста предсказательной силы модели необходимо было использовать умную замену отсутствующих значений (advanced_missing_values_method = True), не исключать из рассмотрения номинальные переменные с минимальной долей неотсутствующих значений не более чем в 20% (значения в столбце not_missing_values_min_prc), оставить достаточную вариативность значений при преобразовании номинальной переменной в множество числовых (не менее 20 значений, столбец max_value_counts) и при этом учитывать даже самые редко встречающиеся значения (min_concentration_per_value не более 5%).

Таблица 1 – Комбинация с максимальным F1_score

advanced_missing_values_method	not_missing_values_min_prc	max_value_counts	min_concentration_per_value	F1_score
ИСТИНА	0,01	20	0,05	0,649
ИСТИНА	0,01	25	0,05	0,649
ИСТИНА	0,05	20	0,05	0,649
ИСТИНА	0,1	25	0,05	0,649
ИСТИНА	0,2	20	0,01	0,649
ИСТИНА	0,2	20	0,05	0,649
ИСТИНА	0,2	25	0,01	0,649
ИСТИНА	0,2	25	0,001	0,649

Несмотря на достигнутый результат данная работа имеет потенциал к улучшению. К примеру, используя метод Grid Search, необходимо будет каждый раз проходить по каждой точке n-мерного пространства, исследуя даже заведомо неудачные комбинации состояний. Для сокращения количества итераций до обнаружения хорошей конфигурации существуют адаптивные байесовские методы. Эти методы выбирают следующее состояние для проверки, учитывая результаты на уже проверенных значениях. Мысль заключается в том, чтобы на каждой итерации найти компромисс между (а) исследованием регионов рядом с самыми удачными точками среди найденных и (б) исследованием регионов с большой неопределенностью, где могут находиться еще более удачные состояния.

ЛИТЕРАТУРА

1. D. Powers, "Evaluation: From Precision, Recall and F-Measure to ROC, Informedness, Markedness & Correlation," J. Mach. Learn. Res., 2, No.1, 37--63 (2011).
2. D. L. Olson and D. Delen, Advanced Data Mining Techniques, Springer, New York (2008).
3. S. E. Whang, "Goods: Organizing google’s datasets," SIGMOD, 795–-806 (2016).

4. L. Chen and A. Kumar, "Enabling and optimizing nonlinear feature interactions in factorized linear algebra," SIGMOD, 1571–-1588 (2019).
5. L. Chen, A. Kumar, J. F. Naughton and J. M. Patel, "Towards linear algebra over normalized data," PVLDB, 10, No.11, 1214–-1225 (2017).
6. I. Czogiel, K. Luebke and C. Weihs, "Response surface methodology for optimizing hyper parameters," Technical report, Universitat Dortmund Fachbereich Statistik (2005).
7. I. D. Erhan, Y. Bengio, A. Courville, P. Manzagol, P. Vincent and S. Bengio, ``Why does unsupervised pre-training help deep learning?" Journal of Machine Learning Research, 625–-660 (2010).
8. G. E. Hinton, "A practical guide to training restricted Boltzmann machines," Technical Report, University of Toronto 1 (2010).