Практическая работа 11. Линейная регрессия: прогноз оклада по описанию вакансии

Цель работы: научиться использовать линейную регрессию и применять её к текстовым данным.

Линейные методы хорошо подходят для работы с разреженными данными, например, текстами. Это объясняется высокой скоростью обучения и небольшим количеством параметров, благодаря чему удаётся избежать переобучения.

Линейная регрессия имеет несколько разновидностей в зависимости от того, какой регуляризатор используется. В настоящей работе используется гребневая регрессия, в которой применяется квадратичный, или L2-регуляризатор.

Для извлечения TF-IDF-признаков из текстов воспользуйтесь классом sklearn.feature_extraction.text.TfidfVectorizer.

Для предсказания целевой переменной будем использовать гребневую регрессию, которая реализована в классе sklearn.linear_model.Ridge.

Обратите внимание, что признаки LocationNormalized и ContractTime являются строковыми, и поэтому с ними нельзя работать напрямую. Такие нечисловые признаки с неупорядоченными значениями называют категориальными или номинальными. Типичный подход к их обработке – кодирование категориального признака с m возможными значениями с помощью m бинарных признаков. Каждый бинарный признак соответствует одному из возможных значений категориального признака и является индикатором того, что на данном объекте он принимает данное значение. Данный подход иногда называют one-hot-кодированием. Воспользуйтесь им, чтобы перекодировать признаки LocationNormalized и ContractTime. Он уже реализован в классе sklearn.feature_extraction.DictVectorizer.

Пример использования:

```
from sklearn.feature_extraction import DictVectorizer
enc = DictVectorizer()
X_train_categ = enc.fit_transform(
     data_train[['LocationNormalized','ContractTime']]
          .to_dict('records'))
X_test_categ = enc.transform(
     data_test[['LocationNormalized','ContractTime']]
          .to_dict('records'))
```

Вам понадобится производить замену пропущенных значений на специальные строковые величины (например, ’nan’). Для этого подходит следующий код:

```
data_train['LocationNormalized'].fillna('nan', inplace = True)
data_train['ContractTime'].fillna('nan', inplace = True)
```

Для выполнения работы необходимо:

1) Загрузить данные об описаниях вакансий и соответствующих годовых зарплатах из файла salary-train.csv.
2) Провести предобработку: а) привести тексты к нижнему регистру; б) заменить всё, кроме букв и цифр, на пробелы – это облегчит дальнейшее разделение текста на слова. Для такой замены в строке text подходит следующий вызов:

```
re.sub('[^a-zA-Z0-9]', ' ', text.lower())
```

в) применить TfidfVectorizer для преобразования текстов в векторы признаков. Оставьте только те слова, которые встречаются хотя бы в 5 объектах (параметр min_df у TfidfVectorizer); г) заменить пропуски в столбцах LocationNormalized и ContractTime на специальную строку ’nan’. Код для этого был приведен выше; д) применить DictVectorizer для получения one-hot-кодирования признаков LocationNormalized и ContractTime; е) объединить все полученные признаки в одну матрицу "объекты- признаки". Обратите внимание, что матрицы для текстов и категориальных признаков являются разреженными. Для объединения их столбцов нужно воспользоваться функцией scipy.sparse.hstack.

3) Обучить гребневую регрессию с параметром alpha=1. Целевая переменная записана в столбце SalaryNormalized.

4) Построить прогнозы для двух примеров из файла salary-test-mini.csv. Значения полученных прогнозов являются ответом на задание. Укажите их через пробел.