МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ  
РОССИЙСКОЙ ФЕДЕРАЦИИ

ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ  
«ВЯТСКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт математики и информационных систем

Факультет автоматики и вычислительной техники

Кафедра систем автоматизации управления

**Линейная регрессия: прогноз оклада по описанию вакансии**

Отчет по лабораторной работе №1  
по дисциплине

«Большие данные»

Выполнил студент гр. ИТб-4302-02-00 \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Вершинин П.А./

(Подпись)

Руководитель ст. преподаватель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ /Родионов К.В./

(Подпись)

Работа защищена с оценкой «\_\_\_\_\_\_\_\_\_\_\_» «\_\_\_» \_\_\_\_\_\_\_\_\_\_ 2023 г.

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

В ходе работы необходимо выполнить следующее задание:

* загрузить данные;
* предварительно обработать загруженные данные;
* обучить модель;
* построить прогнозы.

Для использования линейной регрессий необходимо загрузить данные. Чтобы загрузить данные необходимо выполнить следующий код:

|  |
| --- |
| # Загружаем данные из файла salary-train.csv  data\_train = Refactoring\_csv\_file('salary-train.csv') |

После загрузки данных происходит преобразование текстовых описаний в нижний регистр и замена всех символов, кроме букв и цифр, на пробелы. Чтобы провести все преобразования данных необходимо выполнить следующий код:

|  |
| --- |
| def Refactoring\_csv\_file(fileName, dtTest=False):  dt = pd.read\_csv(fileName)  # Привести тексты к нижнему регистру и заменить всё, кроме букв и цифр, на пробелы  dt['FullDescription'] = (dt['FullDescription']  .apply(lambda x:  re.sub('[^a-zA-Z0-9]', ' ', x.lower())))  if not dtTest:  # Замена пропусков на 'nan' в столбце 'LocationNormalized'  dt['LocationNormalized'].fillna('nan', inplace=True)  # Замена пропусков на 'nan' в столбце 'ContractTime'  dt['ContractTime'].fillna('nan', inplace=True)  return dt |

Также заполняем пропуски в столбцах «LocationNormalized» и «ContractTime» на специальную строку «nan».

Полученный результат подвергается предобработке. Необходимо преобразовать в векторы признаков, и оставить только те слова, которые встречаются хотя бы в 5 объектах. Реализующий данное преобразование код представлен ниже:

|  |
| --- |
| # Инициализация векторов  tfidf\_vectorizer = TfidfVectorizer(min\_df=5)  enc = DictVectorizer()  # Преобразование текстовых данных в векторы признаков на обучающей выборке  tfidf\_features = tfidf\_vectorizer.fit\_transform(data\_train['FullDescription'])  cat\_features = enc.fit\_transform(data\_train[['LocationNormalized', 'ContractTime']].to\_dict('records'))  # Объединить векторы признаков  df\_train\_features = hstack([tfidf\_features, cat\_features]) |

С помощью DictVectorizer преобразовываем категориальные данные в числовые признаки. Это включает в себя столбцы «LocationNormalized» и «ContractTime». Следующим шагом является использование «hstack» для объединения всех полученных признаков в одну матрицу "объекты-признаки".

Для обучения гребневой регрессии необходимо определить целевую переменную и задать значение параметра alpha. Программный код реализации обучения представлен ниже:

|  |
| --- |
| def Regression(target, features):  # Обучить гребневую регрессию  clf = Ridge(alpha=1.0)  clf.fit(features, target)  return clf |

Для построения прогнозов для двух примеров из файла «salary-test-mini.csv» необходимо провести те же действия, что и с файлом «salary-train.csv». Для построения прогноза оклада по описанию вакансии необходимо выполнить код:

|  |
| --- |
| # Загрузить данные из файла salary-test-mini.csv  data\_test = Refactoring\_csv\_file('salary-test-mini.csv', True)  # Преобразование текстовых данных в векторы признаков на тестовых данных  tfidf\_test\_features = tfidf\_vectorizer.transform(data\_test['FullDescription'])  cat\_test\_features = enc.transform(data\_test[['LocationNormalized', 'ContractTime']].to\_dict('records'))  # Объединить векторы признаков  df\_test\_features = hstack([tfidf\_test\_features, cat\_test\_features])  # Построить прогнозы  predictions = ridge\_regression.predict(df\_test\_features)  print(predictions) |

При работе с тестовыми данными добавляется параметр «dtTest», равный True, и поэтому пропущенные значения не заменяются на «nan».

Результат прогнозирования оклада по описанию вакансии представлен на рисунке 1.



Рисунок 1 – Результат прогнозирования

В ходе лабораторной работы были приобретены навыки работы с линейной регрессии и её применения к текстовым данным.