МИНОБРНАУКИ РОССИИ

Федеральное государственное бюджетное

образовательное учреждение высшего образования

«ЧЕРЕПОВЕЦКИЙ ГОСУДАРСТВЕННЫЙ УНИВЕРСИТЕТ»

Институт Информационных Технологий

Кафедра МПО ЭВМ

Дисциплина «Математические методы решения задач искусственного интеллекта»

Лабораторная работа №4

«Анализ данных»

Исполнитель:

студент группы 1ПИб-02-3оп-22

Беляков Артемий Александрович

Руководитель:

Юдина Ольга Вадимовна

2024 год

Вам предоставлен набор данных социологического опроса, проведенного кадровой службой предприятия. Работающие отвечали на вопросы о своих ценностях (применительно к работе на предприятии) и оценивали свою готовность остаться работать на предприятии, если оно сменит место дислокации (город на пригород)

Предприятию важно понять, сможет ли оно сохранить работоспособный коллектив в случае такого переезда, и какую программу поддержки следует разработать

Задание

Для поддержки аналитической работы постройте:

* Регрессию, которая содержит как количественные, так и качественные переменные
* Кластеризацию (на основе метода ближайших соседей)
* Классификацию (на основе метода ближайших соседей)

Точность и адекватность всех методов должны быть оценены

В отчете – пояснения к выбору признаков и методу (методам) выбора расстояний.

Для начала изучим данные в таблице. Она содержит:

Категориальные переменные:

* Пол (м/ж)
* Должность (рабочий, специалист, руководитель)
* Значимые ценности (высокая з/п, коллектив, содержание)
* Образование (среднее/высшее)
* Готовность остаться (да/нет/возможно)

Количественные переменные:

* Возраст
* Длительность работы
* Средняя з/п

1. Регрессия

Регрессия – это метод анализа зависимости одной переменной (называемой зависимой или целевой) от одной или нескольких других переменных (называемых независимыми или объясняющими). Основная задача регрессии – построить модель, которая описывает эту зависимость и позволяет делать прогнозы.

Средствами пакета анализа Excel построим регрессию, где:

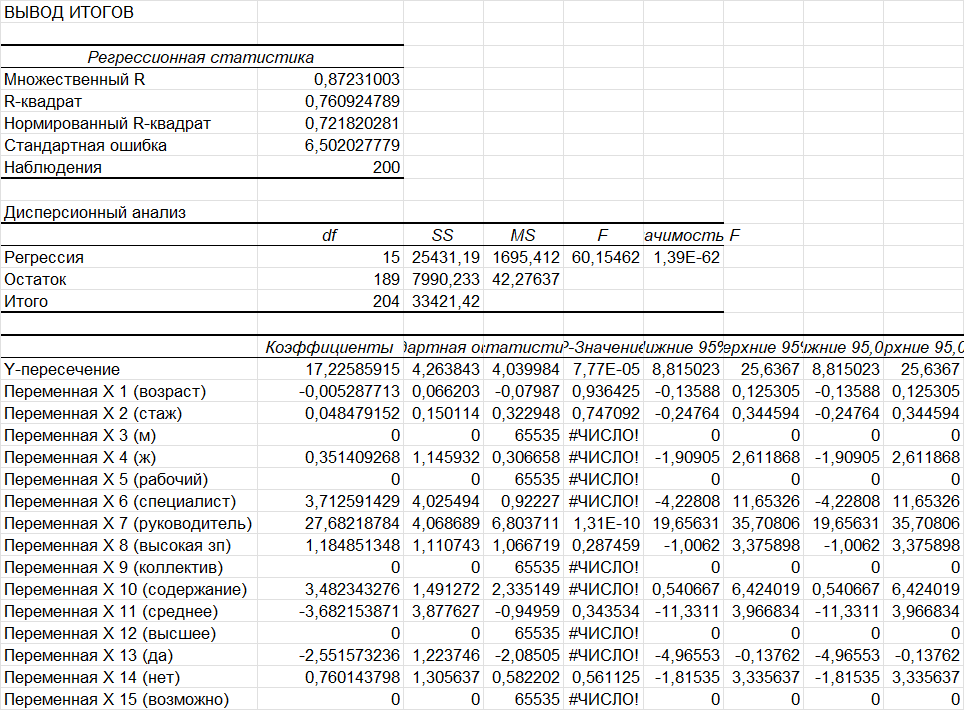
Зарплата – зависимая переменная

Все остальные переменные – независимые:

* Пол (мужской, женский)
* Должность (рабочий, специалист, руководитель)
* Значимые ценности (высокая з/п, коллектив, содержание)
* Образование (среднее, высшее)
* Возраст
* Длительность работы

Категориальные переменные разбиваем на несколько столбцов, для каждой категории отдельный столбец. Если человек принадлежит к данной категории, то ставится 1, в остальных категориях ставится 0.

Получаем следующий результат:



Проанализируем результаты.

Х1 – возраст. Коэффициент -0,005287713 – отрицательный. Это значит, что возраст имеет слабое влияние на размер зарплаты – с увеличением возраста зарплата уменьшается 0,005 тыс. руб.

Х2 – стаж. Коэффициент 0,048479152 – положительный. Это значит, что стаж имеет слабое влияние на размер зарплаты – с увеличением стажа зарплата увеличивается на 0,048 тыс. руб.

Х3 и Х4 пол. За базовую переменную возьмем мужской пол. Коэффициент для женского пола 0,351409268 – положительный. Женский пол оказывает положительное влияние на зарплату, увеличивая ее на 0,351 тыс. руб.

Х5, Х6, Х7 – должности. За базовую переменную возьмем должность «Рабочий». Специалисты зарабатывают больше рабочих на 3,713 тыс. руб. Руководители зарабатывают больше рабочих на 27,682 тыс. руб. Тем самым, влияние должности на з/п высокое

Х8, Х9, Х10 – ценности. За базовую переменную возьмем ценность «Коллектив». Люди с ценностью «Высокая з/п» зарабатывают на 1,185 тыс. руб. больше тех, у кого ценность «Коллектив». Люди с ценностью «Содержание» зарабатывают на 3,482 тыс. руб. больше тех, у кого ценность «Коллектив». Тем самым, ценности влияют на з/п.

Х11, Х12 – образование. За базовую переменную возьмем высшее образование. Люди со средним образованием зарабатывают на 3,682 тыс. руб. меньше людей с высшим. Тем самым, образование оказывает влияние на з/п.

Х13, Х14, Х15 – готовность остаться. За базовую категорию возьмем ответ «Возможно». Люди, желающие остаться, зарабатывают на 2,552 тыс. руб. меньше тех, кто не определился. Люди, не желающие остаться, зарабатывают на 0,76 тыс. руб. больше тех, кто не определился. Тем самым, желание остаться влияет на з/п.

Итоги:

* Наибольшее влияние на размер з/п оказывают должность, ценности и готовность остаться
* Влияние пола на зарплату среднее
* Возраст и стаж почти не влияют на зарплату

Оценим точность и адекватность метода.

R2 – коэффициент детерминации. Он показывает, какая доля изменчивости зависимой переменной объясняется моделью. Чем ближе R2 к 1, тем лучше модель объясняет данные.

В нашем случае R2 = 0,760924789. Следовательно, модель хорошо описывает данные.

1. Кластеризация

KNN (K-Nearest Neighbors) – это один из простых и популярных методов классификации и регрессии в машинном обучении. Основная идея метода заключается в том, чтобы для нового объекта (который нужно классифицировать или предсказать) найти его ближайших соседей в обучающей выборке и использовать их метки (для классификации) или значения (для регрессии) для получения прогноза.

Кластеризация – это задача, в которой модель пытается группировать объекты, которые похожи друг на друга, в кластеры, без заранее известных меток.

Кластеризацию будем проводить по возрасту и зарплате.

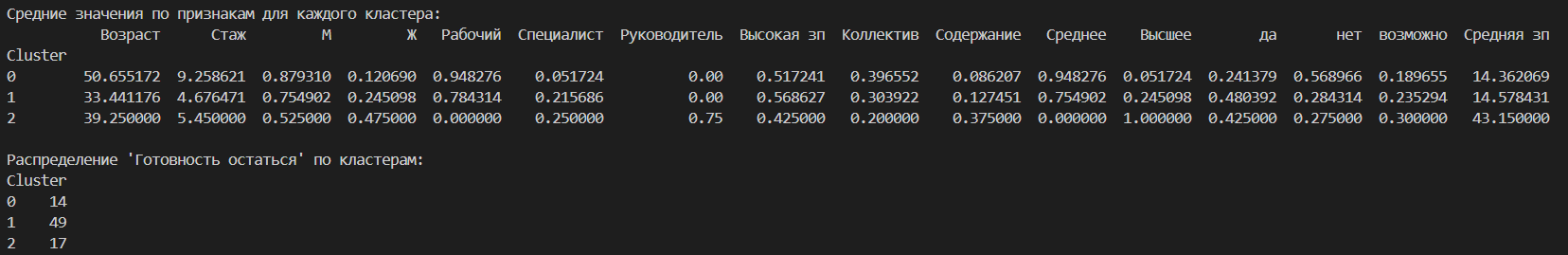
Для проведения кластеризации методом K-Means воспользуемся средствами языка программирования Python. Напишем программу, разделяющую данные на кластеры.

**import** pandas **as** pd  
**from** sklearn.model\_selection **import** train\_test\_split  
**from** sklearn.preprocessing **import** StandardScaler  
**from** sklearn.neighbors **import** KNeighborsClassifier  
**from** sklearn.metrics **import** accuracy\_score, classification\_report, confusion\_matrix  
  
*# Загрузка данных*data = pd.read\_excel(**'mm5.xlsx'**)  
  
*# Разделение данных на независимые и зависимые переменные*X = data[[**'Возраст'**, **'Средняя зп'**]] *# Независимые переменные*y = data[**'Готовность'**] *# Зависимая переменная  
  
# Нормализация данных*scaler = StandardScaler()  
X\_scaled = scaler.fit\_transform(X)  
  
*# Разделение на тренировочную и тестовую выборки*X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.3, random\_state=42)  
  
*# Создание и обучение модели KNN*knn = KNeighborsClassifier(n\_neighbors=5)  
knn.fit(X\_train, y\_train)  
  
*# Прогнозирование на тестовых данных*y\_pred = knn.predict(X\_test)  
  
*# Оценка точности модели*accuracy = accuracy\_score(y\_test, y\_pred)  
print(**f"Точность модели: {**accuracy**:.4f}"**)  
  
*# Отчет по классификации*print(**"Матрица ошибок:"**)  
print(confusion\_matrix(y\_test, y\_pred))  
  
print(**"Отчет по классификации:"**)  
print(classification\_report(y\_test, y\_pred))

Алгоритм работы программы:

1. Загружаются исходные данные из файла
2. Извлекаем числовые признаки, по которым будем проводить кластеризацию (в данном случае это Возраст и Средняя з/п).
3. Применяем алгоритм кластеризации KMeans с 3 кластерами.
4. Сохраняем результаты кластеризации в новый столбец Cluster, который будет содержать номер кластера для каждого наблюдения.
5. Вычисляем средние значения признаков для каждого кластера с помощью groupby('Cluster').mean().
6. Анализируем распределение значения Готовность остаться (например, столбец 'да') по каждому кластеру с помощью groupby('Cluster')['да'].sum(). Это покажет количество людей в каждом кластере, которые выразили готовность остаться.

Результат программы:



Всех сотрудников разбили на 3 кластера: 0, 1 и 2.

* Кластер 0: Люди старшего возраста с низкой зарплатой
* Кластер 1: Молодые люди с низкой зарплатой
* Кластер 2: Люди среднего возраста с высокой зарплатой

Рассмотрим получившиеся кластеры по готовности остаться:

* Кластер 0: 24,1% готовы, 56,9% не готовы, 19% не определились
* Кластер 1: 48% готовы, 28,4% не готовы, 23,5% не определились
* Кластер 2: 42,5% готовы, 27,5% не готовы, 30% не определились

Выводы:

* Большинство людей старшего возраста с низкой зарплатой не готовы оставаться на предприятии
* Большинство людей молодого возраста с низкой зарплатой готовы оставаться на предприятии
* Большинство часть людей среднего возраста с высокой зарплатой готовы остаться на предприятии
* С возрастом люди хуже относятся к переменам, поэтому будут не готовы к переезду. Возраст влияет на готовность остаться
* Готовность остаться выразили как молодые люди с низкой зарплатой, так и люди среднего возраста с высокой зарплатой. Поэтому зарплата не является ключевым фактором при принятии решения.

Оценим точность и адекватность модели.

Силуэтный коэффициент – это мера качества кластеризации. Его значение варьируется от -1 до 1. Чем ближе к 1, тем лучше кластеризация.

В нашем случае силуэтный коэффициент: 0.44841838944100937. Это значение говорит о том, что кластеры выделены с умеренной чёткостью. Внутри кластеров элементы имеют некоторую схожесть, но в то же время кластеры не идеально разделены.

1. Классификация

Классификация – это задача, в которой у нас есть набор данных с известными метками классов, и мы обучаем модель, чтобы она могла предсказать эти метки для новых, неизвестных данных.

Для проведения классификации методом KNN также воспользуемся средствами языка программирования Python. Напишем программу, производящую классификацию данных.

import pandas as pd

from sklearn.model\_selection import train\_test\_split

from sklearn.preprocessing import StandardScaler

from sklearn.neighbors import KNeighborsClassifier

from sklearn.metrics import accuracy\_score, classification\_report, confusion\_matrix

# Загрузка данных

data = pd.read\_excel('Data2.xlsx')

# Разделение данных на независимые и зависимые переменные

X = data.drop(['Возраст', 'Стаж'], axis=1)  # Независимые переменные

y = data['Готовность']  # Зависимая переменная

# Нормализация данных

scaler = StandardScaler()

X\_scaled = scaler.fit\_transform(X)

# Разделение на тренировочную и тестовую выборки

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_scaled, y, test\_size=0.3, random\_state=42)

# Создание и обучение модели KNN

knn = KNeighborsClassifier(n\_neighbors=5)

knn.fit(X\_train, y\_train)

# Прогнозирование на тестовых данных

y\_pred = knn.predict(X\_test)

# Оценка точности модели

accuracy = accuracy\_score(y\_test, y\_pred)

print(f"Точность модели: {accuracy:.4f}")

# Отчет по классификации

print("Матрица ошибок:")

print(confusion\_matrix(y\_test, y\_pred))

print("Отчет по классификации:")

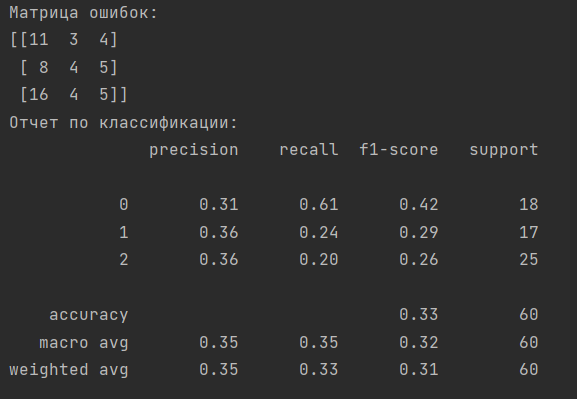
print(classification\_report(y\_test, y\_pred))

Алгоритм программы:

1. Из файла загружаются исходные данные
2. Столбец «Готовность остаться» выделяется как зависимая переменная (y). Столбцы «Возраст» и «Стаж» выделяются как независимые переменные (X).
3. Данные в X масштабируются с помощью StandardScaler для приведения признаков к одному масштабу.
4. Данные делятся на тренировочную (70%) и тестовую (30%) выборки с помощью train\_test\_split, с фиксированным значением random\_state для воспроизводимости.
5. Создается модель KNN (K-Nearest Neighbors) с числом соседей n\_neighbors=5.
6. Модель обучается на тренировочных данных (X\_train, y\_train).
7. Модель предсказывает значения зависимой переменной (y\_pred) для тестовой выборки (X\_test).
8. Вычисляется точность модели (accuracy\_score), которая показывает долю правильных предсказаний.
9. Выводится матрица ошибок для анализа ошибок классификации.
10. Генерируется отчет по классификации (classification report), содержащий метрики:

* Precision (точность),
* Recall (полнота),
* F1-score (сбалансированная метрика),
* Support (количество истинных объектов каждого класса).

Результат программы:



Здесь классы означают:

* 0 класс – не готовы остаться
* 1 класс – возможно готовы остаться
* 2 класс – готовы остаться

Матрица ошибок – показывает, сколько объектов каждого класса было правильно или ошибочно отнесено к другим классам:

* К 0 классу правильно отнесли 11 человек, 3 человек из 0 класса ошибочно был отнесен в 1 класс и 4 человека из 0 класса ошибочно отнесены к 2 классу
* К 1 классу правильно отнесли 4 человек, 8 человек из 1 класса ошибочно были отнесены во 0 класс, 5 человека из 1 класса ошибочно были отнесены во 2 класс
* Ко 2 классу правильно отнесли 5 человек, 16 человек из 2 класса ошибочно были отнесены в 0 класс и 4 человека из 2 класса ошибочно отнесены были ко 2 классу

Precision – точность – показывает, какой процент объектов, предсказанных как принадлежащие этому классу, действительно относятся к этому классу.

* 31% человек, отнесенных к 0 классу, действительно ему принадлежат
* 36% человек, отнесенных к 1 классу, действительно ему принадлежат
* 36% человек, отнесенных ко 2 классу, действительно ему принадлежат

Recall – полнота – показывает, какой процент объектов этого класса модель смогла правильно классифицировать из всех объектов этого класса.

* 61% человек, принадлежащих к 0 классу, были отнесены в этот класс
* 24% человек, принадлежащих к 1 классу, были отнесены в этот класс
* 20% человек, принадлежащих ко 2 классу, были отнесены в этот класс

f1-score – f1-оценка – среднее гармоническое точности и полноты, которое учитывает как ложные положительные, так и ложные отрицательные классификации

Класс 0: F1 = 0.42 – это неплохой результат. Это значит, что модель достаточно хорошо находит объекты этого класса.

Класс 1: F1 = 0.29 – для этого класса результат хуже, что может указывать на проблемы с точностью или полнотой (возможно, есть недооцененные или переоцененные объекты).

Класс 2: F1 = 0.26 – это результат похож на пердыдущий, все так же возможно, есть недооцененные или переоцененные объекты.

Macro avg – средняя точность по классам – усреднение метрик по всем классам без учета их пропорций в данных.

* Средняя точность – 0,35
* Средняя полнота – 0,35
* Средняя f1-оценка – 0,32

Weighted average – это усреднение с учетом количества объектов в каждом классе

* Средняя точность – 0,35
* Средняя полнота – 0,33
* Средняя f1-оценка – 0,31

Точность модели – это доля правильных предсказаний на тестовой выборке. Точность данной модели – 0,33, т. е. 33% человек были классифицированы правильно. Это посредственный результат.

Вывод

В ходе выполнения лабораторной работы был выполнен анализ данных различными методами – с помощью построения регрессии, а также кластеризации и классификации с использованием метода KNN.