Министерство образования и науки Российской Федерации

Российский государственный университет нефти и газа

(национальный исследовательский университет)

имени И.М. Губкина

Кафедра информатики

Отчет по Домашнему заданию №1 («Задача регрессии»)

Дисциплины ***Средства интеллектуального анализа данных и машинное обучение***

Выполнил:

Студент группы АА-22-08

Сафуанов Артур

Проверила:

Доцент кафедры информатики, к.ф.м.н.

Вишневская Елена Александровна

Москва, 2025 г.

Оглавление

Введение

Современный этап развития науки и технологий характеризуется стремительным ростом объёмов данных, что делает актуальным применение методов анализа информации для решения прикладных задач. Регрессионный анализ, как один из ключевых инструментов машинного обучения и статистики, позволяет выявлять зависимости между переменными, прогнозировать значения целевых показателей и принимать обоснованные решения на основе данных. В контексте Data Science умение работать с данными, проводить их подготовку, исследование и строить предсказательные модели становится критически важным навыком.

Целью данной работы является освоение основных этапов исследовательского анализа данных (EDA) и применение методов регрессионного моделирования на практике.

Ход работы

Задание 1. Подобрать из открытых источников или смоделировать набор данных для анализа.

Решение:

Kaggle — это одна из самых популярных платформ для специалистов в области анализа данных, машинного обучения и искусственного интеллекта. Она предоставляет доступ к огромному количеству открытых наборов данных, а также позволяет участвовать в соревнования, изучать готовые решения и делиться своими наработками. Kaggle является отличным ресурсом для поиска данных, которые можно использовать в учебных и исследовательских целях.

Для выполнения задания 1 был выбран датасет possum.csv, доступный на платформе Kaggle. Этот набор данных содержит информацию о морфологических характеристиках опоссумов и включает 14 признаков, таких как длина головы, хвоста, возраст и пол. Датасет идеально подходит для задач регрессии и классификации, а также для отработки навыков исследовательского анализа данных.

Признаки (переменные):

1. **case** — идентификатор образца (уникальный номер).
2. **site** — место, где был пойман опоссум (1-7).
3. **Pop** — популяция (Vic или other).
4. **sex** — пол опоссума (m — мужской, f — женский).
5. **age** — возраст опоссума (в годах).
6. **hdlngth** — длина головы (в мм).
7. **skullw** — ширина черепа (в мм).
8. **totlngth** — общая длина тела (в см).

9. **taill** — длина хвоста (в см).
10. **footlgth** — длина ступни (в мм).
11. **earconch** — длина ушной раковины (в мм).
12. **eye** — расстояние между глазами (в мм).
13. **chest** — обхват груди (в см).
14. **belly** — обхват живота (в см).

Задание 2. Ознакомиться с возможностями получения и загрузки данных из различных источников.

Решение:

Основные источники данных:

1. **Локальные файлы:**
    - **CSV (Comma-Separated Values):** Текстовый формат, где данные разделены запятыми или другими разделителями.
    - **Excel (XLSX):** Табличный формат, поддерживающий несколько листов и сложные структуры.
    - **JSON (JavaScript Object Notation):** Формат для хранения и передачи структурированных данных, часто используемый в веб-приложениях.
    - **Текстовые файлы (TXT):** Простые файлы, содержащие неструктурированные данные.
2. **Базы данных:**
    - Данные могут храниться в реляционных базах данных (например, MySQL, PostgreSQL) или NoSQL-базах (например, MongoDB).
    - Для работы с базами данных используются специализированные библиотеки и языки запросов (SQL).

3. **Веб-источники:**
    - Данные можно загружать с веб-сайтов с помощью **веб-скрапинга** (например, с использованием библиотек BeautifulSoup или Scrapy в Python).
    - Многие сервисы предоставляют доступ к данным через **API** (Application Programming Interface), например, Twitter API, Google Maps API и др.
4. **Специализированные платформы:**
    - Платформы, такие как **Kaggle**, **UCI Machine Learning Repository** или **Google Dataset Search**, предоставляют доступ к открытым наборам данных для анализа и машинного обучения.

Задание 3. Изучить методы импорта данных из файлов различных форматов, способы обработки некорректных значений.

Решение:

```
data_csv = pd.read_csv('possum.csv')

data_excel = pd.read_excel('possum.xls')

data_csv, data_excel = data_csv.drop('case', axis=1), data_excel.drop('case', axis=1)

data_csv.info()
```

Сначала импортируем в csv формате:

```
1 data_csv
```

	site	Pop	sex	age	hdlngth	skullw	totlngth	taill	footlgth	earconch	eye	chest	belly
0	1	Vic	m	8.0	94.1	60.4	89.0	36.0	74.5	54.5	15.2	28.0	36.0
1	1	Vic	f	6.0	92.5	57.6	91.5	36.5	72.5	51.2	16.0	28.5	33.0
2	1	Vic	f	6.0	94.0	60.0	95.5	39.0	75.4	51.9	15.5	30.0	34.0
3	1	Vic	f	6.0	93.2	57.1	92.0	38.0	76.1	52.2	15.2	28.0	34.0
4	1	Vic	f	2.0	91.5	56.3	85.5	36.0	71.0	53.2	15.1	28.5	33.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
99	7	other	m	1.0	89.5	56.0	81.5	36.5	66.0	46.8	14.8	23.0	27.0
100	7	other	m	1.0	88.6	54.7	82.5	39.0	64.4	48.0	14.0	25.0	33.0
101	7	other	f	6.0	92.4	55.0	89.0	38.0	63.5	45.4	13.0	25.0	30.0
102	7	other	m	4.0	91.5	55.2	82.5	36.5	62.9	45.9	15.4	25.0	29.0
103	7	other	f	3.0	93.6	59.9	89.0	40.0	67.6	46.0	14.8	28.5	33.5

104 rows × 13 columns

Затем выводится такая же таблица с эксель файла:

```
1 data_excel
```

	site	Pop	sex	age	hdlngth	skullw	totlngth	taill	footlgth	earconch	eye	chest	belly
0	1	Vic	m	8.0	94.1	60.4	89.0	36.0	74.5	54.5	15.2	28.0	36.0
1	1	Vic	f	6.0	92.5	57.6	91.5	36.5	72.5	51.2	16.0	28.5	33.0
2	1	Vic	f	6.0	94.0	60.0	95.5	39.0	75.4	51.9	15.5	30.0	34.0
3	1	Vic	f	6.0	93.2	57.1	92.0	38.0	76.1	52.2	15.2	28.0	34.0
4	1	Vic	f	2.0	91.5	56.3	85.5	36.0	71.0	53.2	15.1	28.5	33.0
...	...	...	...	...	...	...	...	...	...	...	...	...	...
99	7	other	m	1.0	89.5	56.0	81.5	36.5	66.0	46.8	14.8	23.0	27.0
100	7	other	m	1.0	88.6	54.7	82.5	39.0	64.4	48.0	14.0	25.0	33.0
101	7	other	f	6.0	92.4	55.0	89.0	38.0	63.5	45.4	13.0	25.0	30.0
102	7	other	m	4.0	91.5	55.2	82.5	36.5	62.9	45.9	15.4	25.0	29.0
103	7	other	f	3.0	93.6	59.9	89.0	40.0	67.6	46.0	14.8	28.5	33.5

104 rows × 13 columns

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 104 entries, 0 to 103
Data columns (total 13 columns):
 #   Column    Non-Null Count  Dtype
---  ------    --------------  -----
 0   site      104 non-null    int64
 1   Pop       104 non-null    object
 2   sex       104 non-null    object
 3   age       102 non-null    float64
 4   hdlngth   104 non-null    float64
 5   skullw    104 non-null    float64
 6   totlngth  104 non-null    float64
 7   taill     104 non-null    float64
 8   footlgth  103 non-null    float64
 9   earconch  104 non-null    float64
 10  eye       104 non-null    float64
 11  chest     104 non-null    float64
 12  belly     104 non-null    float64
dtypes: float64(10), int64(1), object(2)
memory usage: 10.7+ KB
```

Методы считывания данных с файлов различных форматов изучены.

Теперь приступим к обработке некорректных значений с помощью встроенных методов pandas:

data_csv.dropna(inplace=True)

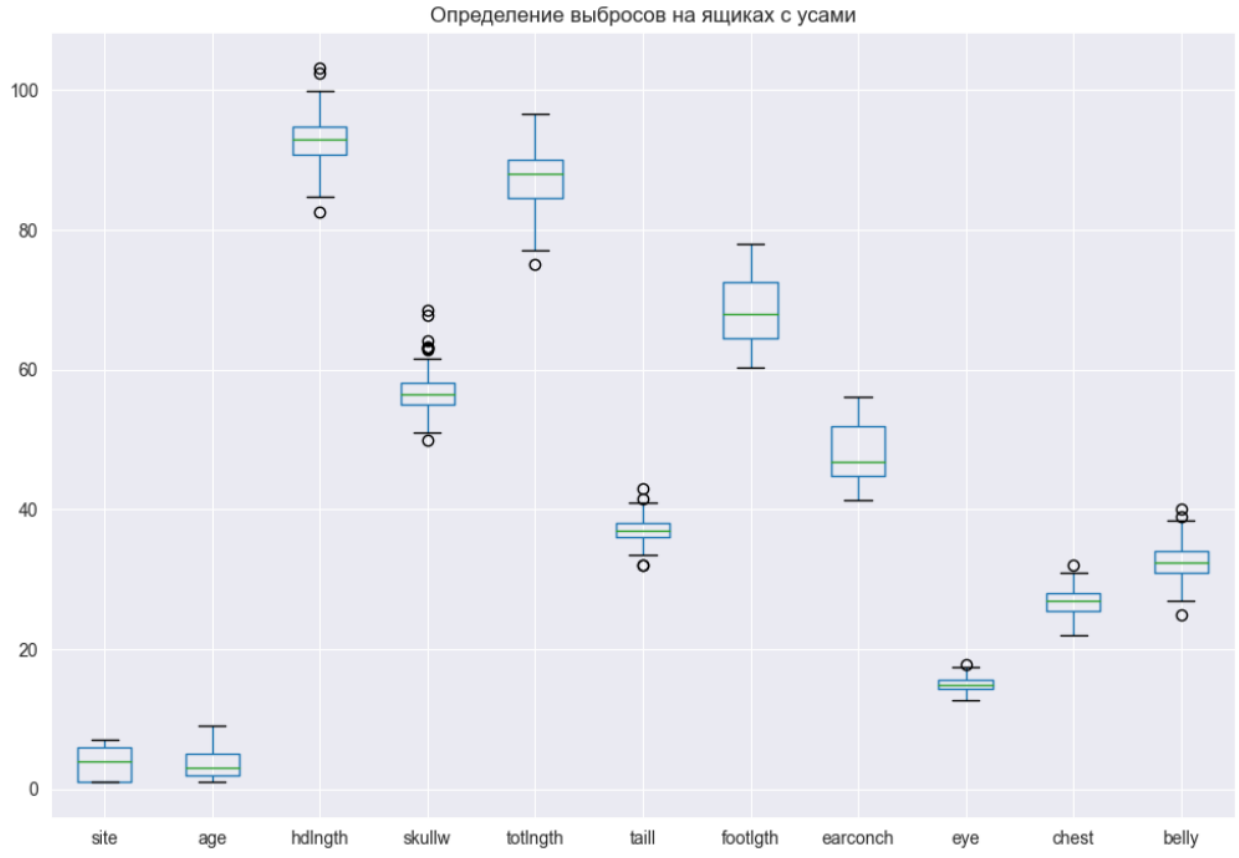
data_csv.isnull().sum()

```
site        0
Pop         0
sex         0
age         0
hdlngth     0
skullw      0
totlngth    0
taill       0
footlgth    0
earconch    0
eye         0
chest       0
belly       0
dtype: int64
```

Таким образом все пропущенные значения были исключены из датасета.

Сейчас воспользуемся методом IQR, чтобы избавиться от выбросов в данных, но для начала мы рассмотрим распределения каждой величины с помощью boxplots:

```
data_csv.boxplot(figsize=(12,8))

plt.title('Определение выбросов на ящиках с усами')

plt.show()
```



Далее пропишем код для удаления выбросов:

```
for col in data_csv.columns:
    if col not in ['Pop', 'sex']:
        print(f'Обработали выбросы в {col}')
        Q1 = np.percentile(data_csv[col], 25)
        Q3 = np.percentile(data_csv[col], 75)
        IQR = Q3 - Q1
        high = Q3 + 1.5 * IQR
        low = Q1 - 1.5 * IQR
        data_csv = data_csv[(data_csv[col] >= low) & (data_csv[col] <= high)]
print()
print(f'Текущая форма данных: {data_csv.shape}')
```

```
Обработали выбросы в site
Обработали выбросы в age
Обработали выбросы в hdlngth
Обработали выбросы в skullw
Обработали выбросы в totlngth
Обработали выбросы в taill
Обработали выбросы в footlgth
Обработали выбросы в earconch
Обработали выбросы в eye
Обработали выбросы в chest
Обработали выбросы в belly

Текущая форма данных: (85, 13)
```

Как видим выбросы были обработаны, так как датасет уменьшился в размере.

Задание 4 – 5. Провести одномерный и двумерный описательный анализ. Исследовать данные, провести одномерный описательный анализ.

Решение:

Начнём с одномерного анализа данных и приведём соответствующие описательные характеристики и графики:

```
print('Для каждого столбца указывается среднее арифметическое, стандартное отклонение, ' \
      'минимум и максимум, процентили:')

data_csv.describe()
```

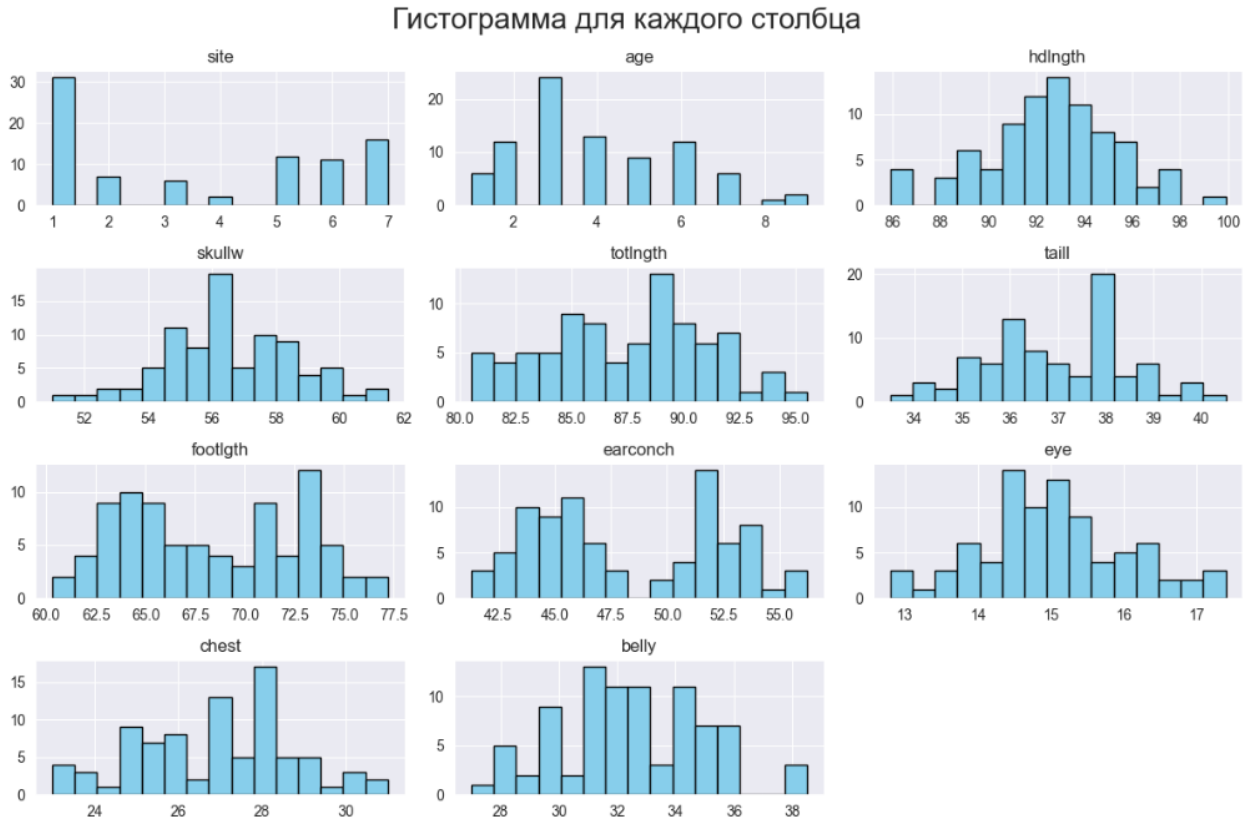
Для каждого столбца указывается среднее арифметическое, стандартное отклонение, минимум и максимум, процентили:

	site	age	hdlngth	skullw	totlngth	taill	footlgth	earconch	eye	chest	belly
count	85.000000	85.000000	85.000000	85.000000	85.000000	85.000000	85.000000	85.000000	85.000000	85.000000	85.000000
mean	3.635294	3.988235	92.612941	56.492941	87.261176	36.970588	68.303529	48.284706	15.040000	26.905882	32.511765
std	2.443880	1.899211	2.773205	2.072656	3.600334	1.572367	4.445609	4.157967	0.984547	1.785347	2.416432
min	1.000000	1.000000	85.900000	51.000000	80.500000	33.500000	60.300000	41.300000	12.800000	23.000000	27.000000
25%	1.000000	3.000000	91.000000	55.000000	85.000000	36.000000	64.400000	44.800000	14.400000	25.500000	31.000000
50%	3.000000	4.000000	92.800000	56.300000	88.000000	37.000000	67.900000	47.000000	15.000000	27.000000	32.000000
75%	6.000000	5.000000	94.400000	57.900000	89.500000	38.000000	72.500000	52.000000	15.700000	28.000000	34.000000
max	7.000000	9.000000	99.900000	61.500000	95.500000	40.500000	77.200000	56.200000	17.400000	31.000000	38.500000

В таблице указаны основные показатели, учитывающиеся при одномерном анализе данных.

Перейдём к визуализации столбцов:

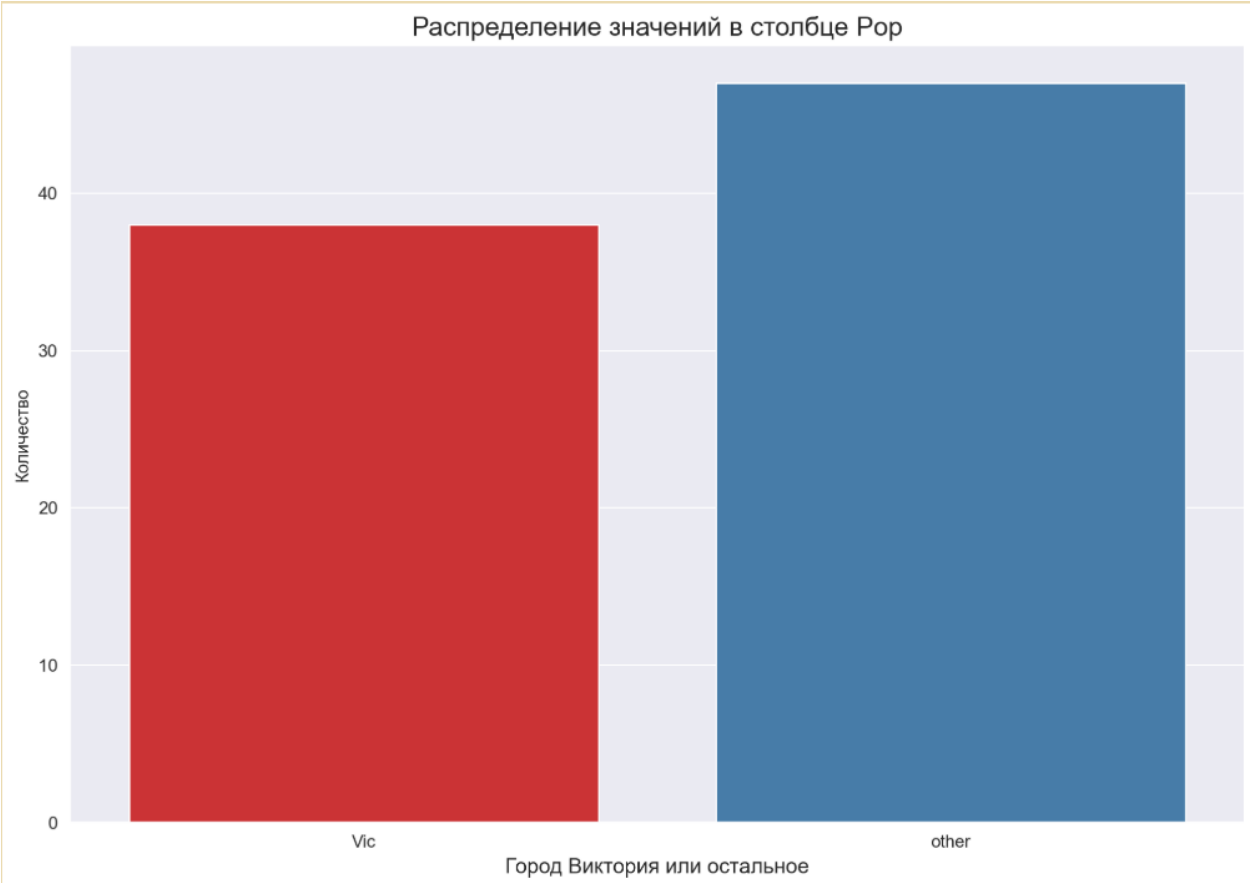
```
data_csv.hist(figsize=(12, 8), edgecolor='black', bins=15, color='skyblue')

plt.suptitle('Гистограмма для каждого столбца', fontsize=20)

plt.tight_layout()

plt.show()
```



Здесь можем видеть, что в датасете присутствуют как дискретные данные, так и непрерывные.

Посмотрим на графики категориальных признаков:

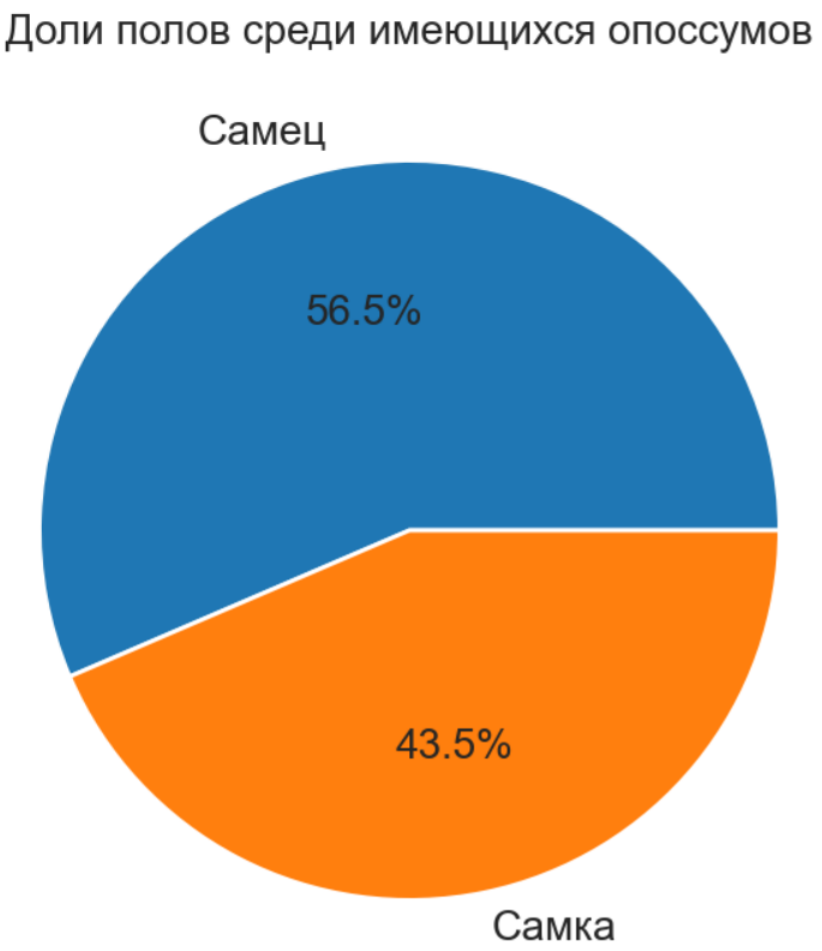
```
plt.figure(figsize=(12,8),dpi=200)

sns.countplot(data=data_csv, x='Pop', palette='Set1')

plt.title('Распределение значений в столбце Pop', fontsize=15)

plt.ylabel('Количество')

plt.xlabel('Город Виктория или остальное', fontsize=12)

plt.show()
```



Видим, что опоссумов с других территорий больше, чем с города Виктория.

Далее построим pieplot по полу опоссумов:

```
pie_data = data_csv['sex'].value_counts()

plt.figure(figsize=(8,4),dpi=200)

plt.pie(pie_data.values, labels=['Самец', 'Самка'], autopct='%1.1f%%');

plt.title('Доли полов среди имеющихся опоссумов', fontsize=10);
```



Можем сделать вывод, что в датасете самцов больше, чем самок.

Приступим к двумерному описательному анализу и посмотрим на матрицу корреляций:

print('Матрица корреляций:')

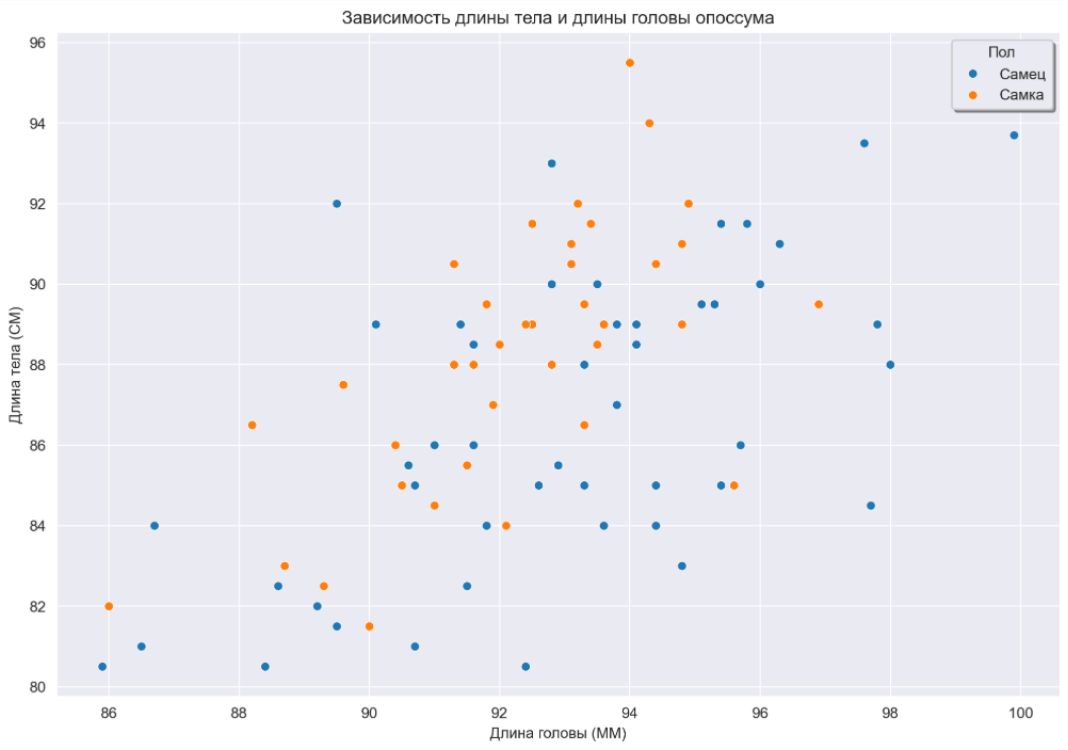
data_csv.corr()

Матрица корреляций:

	site	age	hdlngth	skullw	totlngth	taill	footlgth	earconch	eye	chest	belly
site	1.000000	-0.167653	-0.308798	-0.224023	-0.434859	0.415411	-0.834181	-0.824622	-0.129432	-0.436329	-0.250243
age	-0.167653	1.000000	0.401458	0.449385	0.298692	0.081606	0.183304	0.097062	0.167061	0.408695	0.377460
hdlngth	-0.308798	0.401458	1.000000	0.718643	0.562747	0.044726	0.389509	0.286855	0.369768	0.530310	0.409548
skullw	-0.224023	0.449385	0.718643	1.000000	0.529262	0.170161	0.328597	0.150461	0.424788	0.541907	0.439157
totlngth	-0.434859	0.298692	0.562747	0.529262	1.000000	0.390835	0.467469	0.301029	0.211221	0.492999	0.328804
taill	0.415411	0.081606	0.044726	0.170161	0.390835	1.000000	-0.292404	-0.453473	0.032683	0.007484	0.101922
footlgth	-0.834181	0.183304	0.389509	0.328597	0.467469	-0.292404	1.000000	0.849689	0.043975	0.467791	0.272944
earconch	-0.824622	0.097062	0.286855	0.150461	0.301029	-0.453473	0.849689	1.000000	-0.065891	0.335052	0.139949
eye	-0.129432	0.167061	0.369768	0.424788	0.211221	0.032683	0.043975	-0.065891	1.000000	0.183675	0.231731
chest	-0.436329	0.408695	0.530310	0.541907	0.492999	0.007484	0.467791	0.335052	0.183675	1.000000	0.561119
belly	-0.250243	0.377460	0.409548	0.439157	0.328804	0.101922	0.272944	0.139949	0.231731	0.561119	1.000000

Также рассмотрим диаграммы рассеяния для лучшего понимания зависимостей в данных:

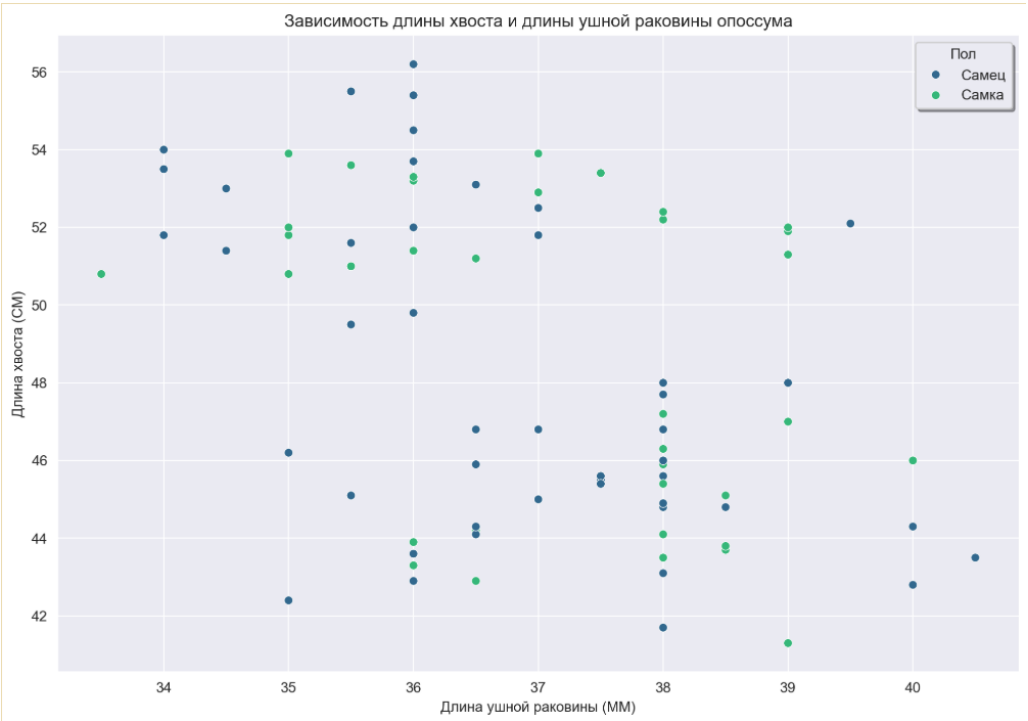
```
plt.figure(figsize=(12,8), dpi=200)
legend_map = {'m': 'Самец',
              'f': 'Самка'}
sns.scatterplot(data=data_csv, x='hdlngth', y='totlngth', hue=data_csv['sex'].map(legend_map))
plt.title('Зависимость длины тела и длины головы опоссума')
plt.xlabel('Длина головы (ММ)')
plt.ylabel('Длина тела (СМ)')
plt.legend(title='Пол', shadow=True, frameon=True)
plt.show()
```



Видим, что, например, длина головы и тела имеют положительную корреляцию. Другими словами, чем больше длина головы, тем больше длина тела в целом.

Рассмотрим ещё одну диаграмму:

```
plt.figure(figsize=(12,8), dpi=200)
legend_map = {'m': 'Самец',
              'f': 'Самка'}
sns.scatterplot(data=data_csv, x='taill', y='earconch', hue=data_csv['sex'].map(legend_map),
                palette='viridis')
plt.title('Зависимость длины хвоста и длины ушной раковины опоссума')
plt.xlabel('Длина ушной раковины (ММ)')
plt.ylabel('Длина хвоста (СМ)')
plt.legend(title='Пол', shadow=True, frameon=True)
plt.show()
```



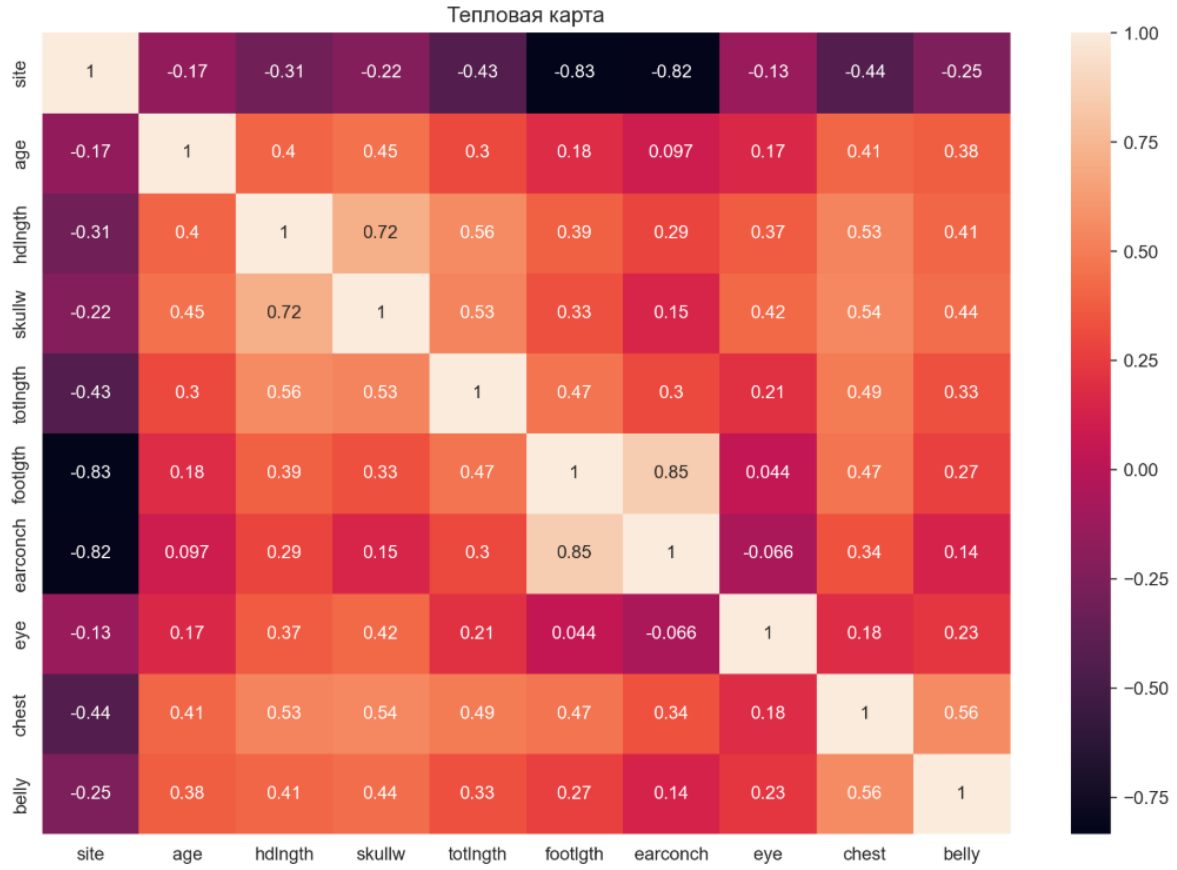
Здесь уже мы имеем дело с отрицательной корреляцией. И как это ни странно, с уменьшением длины ушной раковины, длина хвоста увеличивается.

Задание 6. Провести разведочный двумерный анализ данных с целью выявления зависимостей.

Решение:

Для определения зависимостей между признаками построим heatmap и удалим мультиколлинеарные признаки:

```
data_corr = data_csv.drop(['taill', 'Pop', 'sex'], axis=1).corr()

plt.figure(figsize=(12,8), dpi=200)

sns.heatmap(data=data_corr, annot=True, cmap='rocket')

plt.show()
```



Удалим столбец earconch, так как он сильно коррелирует со столбцом footlngth и site

data_csv = data_csv.drop('earconch', axis=1)

Задание 7. Подготовить данные и построить линейную регрессионную модель, оценить точность.

Решение:

Подготовка данных к регрессии и обучение модели

В качестве целевой переменной возьмём длину тела

Предобработаем категориальные признаки путём присваивания им числовых значений: 0 и 1.

X['Pop'] = X['Pop'].map({'Vic': 1, 'other': 0})

X['sex'] = X['sex'].map({'m': 1, 'f': 0})

```
1  X.head()
```

	site	Pop	sex	age	hdlngth	skullw	taill	footlgth	eye	chest	belly
0	1	1	1	8.0	94.1	60.4	36.0	74.5	15.2	28.0	36.0
1	1	1	0	6.0	92.5	57.6	36.5	72.5	16.0	28.5	33.0
2	1	1	0	6.0	94.0	60.0	39.0	75.4	15.5	30.0	34.0
3	1	1	0	6.0	93.2	57.1	38.0	76.1	15.2	28.0	34.0
4	1	1	0	2.0	91.5	56.3	36.0	71.0	15.1	28.5	33.0

Теперь распишем пайплайн, в который включены сразу масштабирование и линейная регрессия. Таким образом, мы не будем отдельно обрабатывать данные:

```
pipe = Pipeline([
    ('scale', StandardScaler()),
    ('model', LinearRegression())
])
```

Теперь разделим выборку на тренировочную и тестовую:

```
X = data_csv.drop('totlngth', axis=1)
y = data_csv[['totlngth']]
```

Так как данных для обучения достаточно мало, то необходимо выполнить кросс-валидацию:

```
mse = cross_val_score(pipe, X, y, cv=5, scoring='neg_mean_squared_error')
mae = cross_val_score(pipe, X, y, cv=5, scoring='neg_mean_absolute_error')
r2 = cross_val_score(pipe, X, y, cv=5, scoring='r2')
```

Выведем все регрессионные метрики в виде датасета:

```
pd.DataFrame(data=[-mse, -mae, r2, np.sqrt(-mse)], index=['MSE', 'MAE', 'R2', 'RMSE']).transpose()
```

	MSE	MAE	R2	RMSE
0	4.313335	1.715126	0.299886	2.076857
1	4.756613	1.814315	0.458901	2.180966
2	8.624832	2.379548	0.399037	2.936806
3	2.499957	1.273938	0.787379	1.581125
4	4.676967	1.856336	0.264612	2.162630

Кросс-валидация проводилась, используя 5 циклов, поэтому и получилось 5 значений. Наконец, усредним значения, чтобы получить истинные результаты:

```
pd.DataFrame(data=[(-mse).mean(), (-mae).mean(), r2.mean(), (np.sqrt(-mse)).mean()],
```

index=['MSE', 'MAE', 'R2', 'RMSE'],

columns=['Среднее значение'])

	Среднее значение
MSE	4.974341
MAE	1.807853
R2	0.441963
RMSE	2.187677

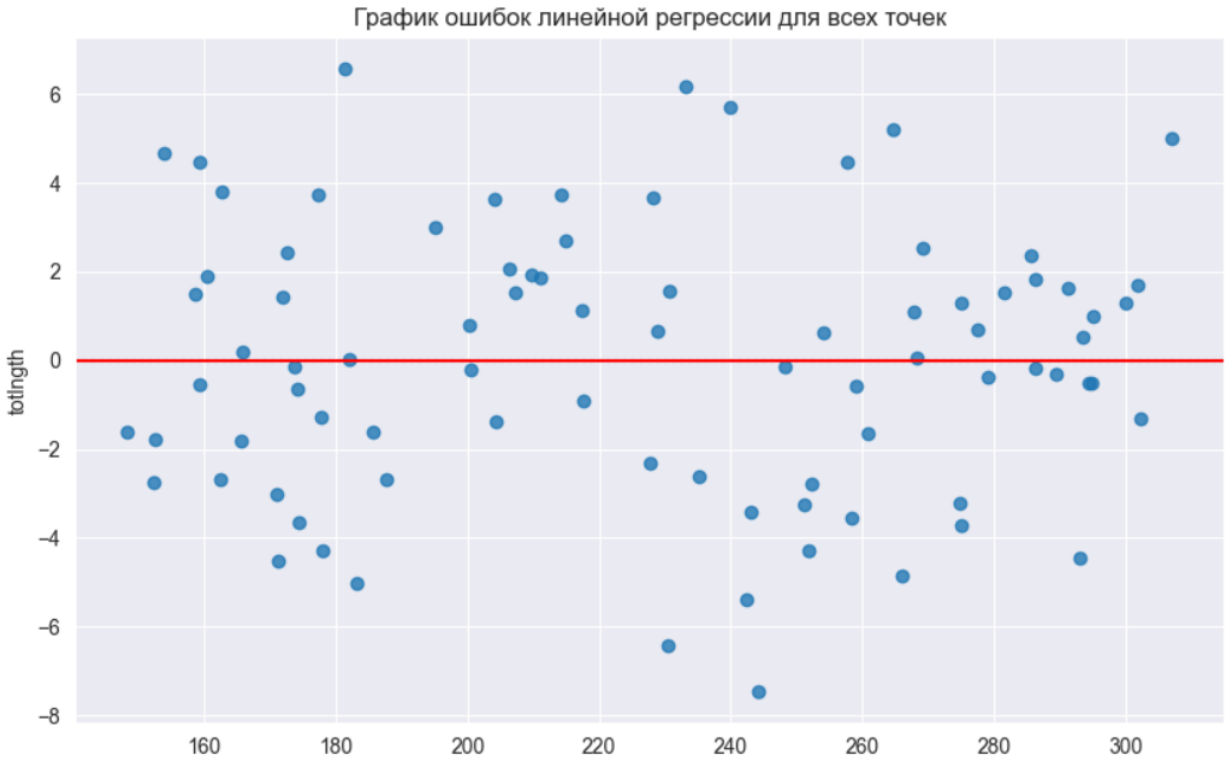
Таким образом, мы получили RMSE = 2.187, что означает небольшое отклонение относительно общей длины тела. В пересчёте на реальные цифры, модель ошибается в среднем на 2 см в предсказании, когда опоссумы имеют среднюю длину примерно 1 м.

Также выведем коэффициенты множественной регрессии и коэффициент пересечения в таблице:

	case	site	Pop	sex	age	hdlngth	skullw	taill	footlgth	eye	chest	belly
Коэффициент обучения	-1.426603	-0.505616	-0.279218	-0.331317	-0.113039	1.159616	0.284442	2.012064	0.352262	-0.075315	0.175323	-0.285199

```
Коэффициент пересечения: 87.26117647058824
```

И добавлю график ошибок, чтобы лучше интерпретировать обучение:



Список используемой литературы

1. **Pandas:**
    - Официальная документация Pandas.
    - McKinney, W. Python for Data Analysis: Data Wrangling with Pandas, NumPy, and IPython. — 2nd Edition. — O'Reilly Media, 2017. — 544 p.
2. **Scikit-learn:**
    - Официальная документация Scikit-learn.
    - Pedregosa, F., et al. Scikit-learn: Machine Learning in Python // Journal of Machine Learning Research. — 2011. — Vol. 12. — P. 2825–2830.
3. **Seaborn:**
    - Официальная документация Seaborn.
    - Waskom, M. L. seaborn: statistical data visualization // Journal of Open Source Software. — 2021. — Vol. 6, № 60. — P. 3021.
4. **Matplotlib:**
    - Официальная документация Matplotlib.
    - Hunter, J. D. Matplotlib: A 2D Graphics Environment // Computing in Science & Engineering. — 2007. — Vol. 9, № 3. — P. 90–95.
5. **Общие ресурсы по Python и анализу данных:**
    - VanderPlas, J. Python Data Science Handbook: Essential Tools for Working with Data. — O'Reilly Media, 2016. — 548 p.