SQL

Задача — проанализировать базу данных крупного сервиса для чтения книг по подписке. База данных содержит информацию о книгах, издательствах, авторах, а также пользовательские обзоры книг. Эти данные помогут сформулировать ценностное предложение для нового продукта.

Описание данных

Таблица `books`

Содержит данные о книгах:

- `book_id` — идентификатор книги;
- `author_id` — идентификатор автора;
- `title` — название книги;
- `num_pages` — количество страниц;
- `publication_date` — дата публикации книги;
- `publisher_id` — идентификатор издателя.

Таблица `authors`

Содержит данные об авторах:

- `author_id` — идентификатор автора;
- `author` — имя автора.

Таблица `publishers`

Содержит данные об издательствах:

- `publisher_id` — идентификатор издательства;
- `publisher` — название издательства;

Таблица `ratings`

Содержит данные о пользовательских оценках книг:

- `rating_id` — идентификатор оценки;
- `book_id` — идентификатор книги;
- `username` — имя пользователя, оставившего оценку;
- `rating` — оценка книги.

Таблица `reviews`

Содержит данные о пользовательских обзорах на книги:

- `review_id` — идентификатор обзора;
- `book_id` — идентификатор книги;
- `username` — имя пользователя, написавшего обзор;
- `text` — текст обзора.

Оглавление

- Шаг 0. Исследуем таблицы — выведем первые строки
- Шаг 1. Посчитаем, сколько книг вышло после 1 января 2000 года
- Шаг 2. Посчитаем количество обзоров и среднюю оценку для каждой книги
- Шаг 3. Определим издательство, которое выпустило наибольшее число книг толще 50 страниц, тем самым исключив из анализа брошюры
- Шаг 4. Определим автора с самой высокой средней оценкой книг — учитываем только книги с 50 и более оценками
- Шаг 5. Посчитаем среднее количество обзоров от пользователей, которые поставили больше 50 оценок
- Выводы

In [1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import plotly.express as px
import seaborn as sns
import math
import datetime
from IPython.display import display
from plotly import graph_objects as go
from scipy import stats as st
import warnings
warnings.filterwarnings('ignore')
from sklearn.preprocessing import StandardScaler
from sklearn.model_selection import train_test_split
from sklearn.linear_model import LogisticRegression
from sklearn.linear_model import Lasso, Ridge
from sklearn.tree import DecisionTreeRegressor
from sklearn.ensemble import RandomForestRegressor, GradientBoostingRegressor
from sklearn.ensemble import RandomForestClassifier
from sklearn.metrics import accuracy_score, precision_score, recall_score
from scipy.cluster.hierarchy import dendrogram, linkage
from sklearn.cluster import KMeans
```

In [2]:
```
# импортируем библиотеки
import pandas as pd
from sqlalchemy import create_engine
# устанавливаем параметры
db_config = {'user': 'praktikum_student', # имя пользователя
 'pwd': 'Sdf4$2;d-d30pp', # пароль
 'host': 'rc1b-wcoijxj3yxfsf3fs.mdb.yandexcloud.net',
 'port': 6432, # порт подключения
 'db': 'data-analyst-final-project-db'} # название базы данных
connection_string = 'postgresql://{}:{}@{}:{}/{}'.format(db_config['user'],
 db_config['pwd'],
 db_config['host'],
 db_config['port'],
 db_config['db'])
# сохраняем коннектор
engine = create_engine(connection_string, connect_args={'sslmode':'require'})
```

Шаг 0. Исследуем таблицы — выведем первые строки

Выведем в цикле первые пять строк каждой таблицы.

In [3]:
```
for i in ['books', 'authors', 'publishers', 'ratings', 'reviews']:
    query = 'SELECT * FROM ' + i + ' LIMIT 5;'
    display(pd.io.sql.read_sql(query, con = engine))
```

	book_id	author_id	title	num_pages	publication_date	publisher_id
0	1	546	'Salem's Lot	594	2005-11-01	93
1	2	465	1 000 Places to See Before You Die	992	2003-05-22	336
2	3	407	13 Little Blue Envelopes (Little Blue Envelope...	322	2010-12-21	135
3	4	82	1491: New Revelations of the Americas Before C...	541	2006-10-10	309
4	5	125	1776	386	2006-07-04	268

	author_id	author
0	1	A.S. Byatt
1	2	Aesop/Laura Harris/Laura Gibbs
2	3	Agatha Christie
3	4	Alan Brennert
4	5	Alan Moore/David Lloyd

	publisher_id	publisher
0	1	Ace
1	2	Ace Book
2	3	Ace Books
3	4	Ace Hardcover
4	5	Addison Wesley Publishing Company

	rating_id	book_id	username	rating
0	1	1	ryanfranco	4
1	2	1	grantpatricia	2
2	3	1	brandtandrea	5
3	4	2	lorichen	3
4	5	2	mariokeller	2

	review_id	book_id	username	text
0	1	1	brandtandrea	Mention society tell send professor analysis. ...
1	2	1	ryanfranco	Foot glass pretty audience hit themselves. Amo...
2	3	2	lorichen	Listen treat keep worry. Miss husband tax but ...
3	4	3	johnsonamanda	Finally month interesting blue could nature cu...
4	5	3	scotttamara	Nation purpose heavy give wait song will. List...

И общую информацию по каждой из них.

In [4]:
```
for i in ['books', 'authors', 'publishers', 'ratings', 'reviews']:
    query = 'SELECT * FROM ' + i + ';'
    display(pd.io.sql.read_sql(query, con = engine).info())
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 1000 entries, 0 to 999
Data columns (total 6 columns):
book_id             1000 non-null int64
author_id           1000 non-null int64
title               1000 non-null object
num_pages           1000 non-null int64
publication_date    1000 non-null object
publisher_id        1000 non-null int64
dtypes: int64(4), object(2)
memory usage: 47.0+ KB

None

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 636 entries, 0 to 635
Data columns (total 2 columns):
author_id    636 non-null int64
author       636 non-null object
dtypes: int64(1), object(1)
memory usage: 10.1+ KB

None

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 340 entries, 0 to 339
Data columns (total 2 columns):
publisher_id    340 non-null int64
publisher       340 non-null object
dtypes: int64(1), object(1)
memory usage: 5.4+ KB

None

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 6456 entries, 0 to 6455
Data columns (total 4 columns):
rating_id    6456 non-null int64
book_id      6456 non-null int64
username     6456 non-null object
rating       6456 non-null int64
dtypes: int64(3), object(1)
memory usage: 201.9+ KB

None

<class 'pandas.core.frame.DataFrame'>
RangeIndex: 2793 entries, 0 to 2792
Data columns (total 4 columns):
review_id    2793 non-null int64
book_id      2793 non-null int64
username     2793 non-null object
text         2793 non-null object
dtypes: int64(2), object(2)
memory usage: 87.4+ KB

None
```

К оглавлению

Шаг 1. Посчитаем, сколько книг вышло после 1 января 2000 года

In [5]:
```
query = '''
SELECT COUNT(book_id) AS book_number
FROM books
WHERE publication_date::date > '2000-01-01';
'''
```

In [6]:
```
pd.io.sql.read_sql(query, con = engine)
```

Out[6]:

	book_number
0	819

Выводы: большинство книг из таблицы `books` опубликовано в XXI веке.

К оглавлению

Шаг 2. Посчитаем количество обзоров и среднюю оценку для каждой книги

In [11]:
```
query = '''
SELECT title AS title,
COUNT(review_id) AS review_number
FROM books
LEFT JOIN reviews ON books.book_id = reviews.book_id
GROUP BY title
'''
```

In [12]:
```
pd.io.sql.read_sql(query, con = engine)
```

Out[12]:

	title	review_number
0	The Count of Monte Cristo	5
1	Count Zero (Sprawl #2)	2
2	The Botany of Desire: A Plant's-Eye View of th...	2
3	The Poisonwood Bible	5
4	The Canterbury Tales	3
...	...	...
994	Of Love and Other Demons	2
995	In the Heart of the Sea: The Tragedy of the Wh...	3
996	Welcome to Temptation (Dempseys #1)	2
997	World's End (The Sandman #8)	2
998	Holes (Holes #1)	5

999 rows × 2 columns

In [13]:
```
query = '''
SELECT title AS title,
AVG(rating) AS rating
FROM books
LEFT JOIN ratings ON books.book_id = ratings.book_id
GROUP BY title
'''
```

In [14]:
```
pd.io.sql.read_sql(query, con = engine)
```

Out[14]:

	title	rating
0	The Count of Monte Cristo	4.217391
1	Count Zero (Sprawl #2)	2.500000
2	The Botany of Desire: A Plant's-Eye View of th...	3.500000
3	The Poisonwood Bible	4.363636
4	The Canterbury Tales	3.333333
...	...	...
994	Of Love and Other Demons	4.500000
995	In the Heart of the Sea: The Tragedy of the Wh...	3.333333
996	Welcome to Temptation (Dempseys #1)	5.000000
997	World's End (The Sandman #8)	4.500000
998	Holes (Holes #1)	3.967742

999 rows × 2 columns

Объединим запросы.

In [15]:
```
query = '''
SELECT review_query.title,
review_query.review_number,
rating_query.rating
FROM
(
SELECT title AS title,
COUNT(review_id) AS review_number
FROM books
LEFT JOIN reviews ON books.book_id = reviews.book_id
GROUP BY title
) AS review_query
INNER JOIN
(
SELECT title AS title,
AVG(rating) AS rating
FROM books
LEFT JOIN ratings ON books.book_id = ratings.book_id
GROUP BY title
) AS rating_query
ON review_query.title = rating_query.title
ORDER BY review_query.review_number DESC;
'''
```

In [16]:
```
pd.io.sql.read_sql(query, con = engine)
```

Out[16]:

	title	review_number	rating
0	Memoirs of a Geisha	8	4.107143
1	Twilight (Twilight #1)	7	3.662500
2	Outlander (Outlander #1)	6	4.125000
3	The Da Vinci Code (Robert Langdon #2)	6	3.830508
4	The Glass Castle	6	4.206897
...	...	...	...
994	The Cat in the Hat and Other Dr. Seuss Favorites	0	5.000000
995	Essential Tales and Poems	0	4.000000
996	Anne Rice's The Vampire Lestat: A Graphic Novel	0	3.666667
997	The Natural Way to Draw	0	3.000000
998	Leonardo's Notebooks	0	4.000000

999 rows × 3 columns

Выводы: у книг с наибольшим числом рецензий не всегда высокий рейтинг. Также можно заметить, что название одной книги повторяется.

К оглавлению

Шаг 3. Определим издательство, которое выпустило наибольшее число книг толще 50 страниц, тем самым исключив из анализа брошюры

In [17]:
```
query = '''
SELECT publisher AS publisher,
COUNT(publisher) AS publisher_number
FROM publishers
INNER JOIN books ON publishers.publisher_id = books.publisher_id
WHERE num_pages > 50
GROUP BY publisher
ORDER BY publisher_number DESC;
'''
```

In [18]:
```
pd.io.sql.read_sql(query, con = engine)
```

Out[18]:

	publisher	publisher_number
0	Penguin Books	42
1	Vintage	31
2	Grand Central Publishing	25
3	Penguin Classics	24
4	Ballantine Books	19
...	...	...
329	Turtleback	1
330	Atheneum Books for Young Readers: Richard Jack...	1
331	Penguin Signet	1
332	Victor Gollancz	1
333	Harvard Business Review Press	1

334 rows × 2 columns

Выводы: больше всего книг толще 50 страниц выпустило издательство Penguin Books.

К оглавлению

Шаг 4. Определим автора с самой высокой средней оценкой книг — учитываем только книги с 50 и более оценками

In [21]:
```
query = '''
SELECT book_id AS book_id,
AVG(rating) AS rating
FROM ratings
GROUP BY book_id
HAVING COUNT(rating_id) >= 50;
'''
```

In [22]:
```
pd.io.sql.read_sql(query, con = engine)
```

Out[22]:

	book_id	rating
0	75	3.678571
1	750	4.125000
2	545	3.787879
3	948	3.662500
4	488	3.622951
5	696	3.830508
6	722	4.391892
7	627	3.789474
8	733	3.750000
9	779	4.080645
10	405	3.901408
11	302	4.414634
12	673	3.825581
13	300	4.246575
14	299	4.287500
15	301	4.186667
16	399	4.192308
17	79	3.729730
18	656	4.264151

In [23]:
```
query = '''
SELECT author AS author,
book_id AS book_id
FROM books
INNER JOIN authors ON books.author_id = authors.author_id;
'''
```

In [24]:
```
pd.io.sql.read_sql(query, con = engine)
```

Out[24]:

	author	book_id
0	Stephen King/Jerry N. Uelsmann	1
1	Patricia Schultz	2
2	Maureen Johnson	3
3	Charles C. Mann	4
4	David McCullough	5
...	...	...
995	Terry Pratchett	996
996	Orson Scott Card/Piotr W. Cholewa	997
997	Geraldine Brooks	998
998	Christopher Moore	999
999	Robert M. Pirsig	1000

1000 rows × 2 columns

Объединим запросы.

In [25]:
```
query = '''
SELECT authors_books.author AS author,
ratings_over50.rating AS rating
FROM
(
SELECT book_id AS book_id,
AVG(rating) AS rating
FROM ratings
GROUP BY book_id
HAVING COUNT(rating_id) >= 50
) AS ratings_over50
INNER JOIN
(
SELECT author AS author,
book_id AS book_id
FROM books
INNER JOIN authors ON books.author_id = authors.author_id
) AS authors_books
ON ratings_over50.book_id = authors_books.book_id
ORDER BY rating DESC;
'''
```

In [26]:
```
pd.io.sql.read_sql(query, con = engine)
```

Out[26]:

	author	rating
0	J.K. Rowling/Mary GrandPré	4.414634
1	J.R.R. Tolkien	4.391892
2	J.K. Rowling/Mary GrandPré	4.287500
3	Markus Zusak/Cao Xuân Việt Khương	4.264151
4	J.K. Rowling/Mary GrandPré	4.246575
5	Louisa May Alcott	4.192308
6	J.K. Rowling/Mary GrandPré	4.186667
7	J.R.R. Tolkien	4.125000
8	Rick Riordan	4.080645
9	William Golding	3.901408
10	Dan Brown	3.830508
11	J.D. Salinger	3.825581
12	Paulo Coelho/Alan R. Clarke/Özdemir İnce	3.789474
13	William Shakespeare/Paul Werstine/Barbara A. M...	3.787879
14	Lois Lowry	3.750000
15	George Orwell/Boris Grabnar/Peter Škerl	3.729730
16	Dan Brown	3.678571
17	Stephenie Meyer	3.662500
18	John Steinbeck	3.622951

Выводы: автор с самой высокой средней оценкой книг (для 50 и более оценок) — Джоан Роулинг.

К оглавлению

Шаг 5. Посчитаем среднее количество обзоров от пользователей, которые поставили больше 50 оценок

Посчитаем пользователей, которые поставили больше 50 оценок.

In [31]:
```
query = '''
SELECT username AS username,
COUNT(rating_id) AS rating_number
FROM ratings
GROUP BY username
HAVING COUNT(rating_id) > 50;
'''
```

In [32]:
```
pd.io.sql.read_sql(query, con = engine)
```

Out[32]:

	username	rating_number
0	sfitzgerald	55
1	jennifermiller	53
2	xdavis	51
3	paul88	56
4	martinadam	56
5	richard89	55

Посчитаем общее количество обзоров для всех пользователей.

In [33]:
```
query = '''
SELECT username AS username,
COUNT(review_id) AS review_number
FROM reviews
GROUP BY username;
'''
```

In [34]:
```
pd.io.sql.read_sql(query, con = engine)
```

Out[34]:

	username	review_number
0	zjohnston	14
1	jnelson	21
2	lewisdesiree	19
3	patrickhudson	15
4	ulowe	16
...	...	...
155	shermannatalie	24
156	dmiller	18
157	charlesraymond	16
158	isaiahreyes	15
159	masonsara	14

160 rows × 2 columns

Выделим нужных пользователей.

In [35]:
```
query = '''
SELECT username AS username,
COUNT(review_id) AS review_number
FROM reviews
WHERE username IN
(
SELECT username AS username
FROM ratings
GROUP BY username
HAVING COUNT(rating_id) > 50
)
GROUP BY username;
'''
```

In [36]:
```
pd.io.sql.read_sql(query, con = engine)
```

Out[36]:

	username	review_number
0	sfitzgerald	28
1	jennifermiller	25
2	xdavis	18
3	paul88	22
4	martinadam	27
5	richard89	26

Найдём среднее количество обзоров.

In [37]:
```
query = '''
SELECT AVG(review_number.review_number) AS avg_review_number
FROM
(
SELECT username AS username,
COUNT(review_id) AS review_number
FROM reviews
WHERE username IN
(
SELECT username AS username
FROM ratings
GROUP BY username
HAVING COUNT(rating_id) > 50
)
GROUP BY username
) AS review_number;
'''
```

In [38]:
```
pd.io.sql.read_sql(query, con = engine)
```

Out[38]:

	avg_review_number
0	24.333333

Выводы: среднее количество обзоров от пользователей, которые поставили больше 50 оценок, — 24.

К оглавлению

Выводы

Большинство книг из таблицы `books` опубликовано в XXI веке.

У книг с наибольшим числом рецензий не всегда высокий рейтинг. Также можно заметить, что название одной книги повторяется.

Больше всего книг толще 50 страниц выпустило издательство Penguin Books.

Автор с самой высокой средней оценкой книг (для 50 и более оценок) — Джоан Роулинг.

Среднее количество обзоров от пользователей, которые поставили больше 50 оценок, — 24.333333.

К оглавлению