ГОСУДАРСТВЕННОЕ БЮДЖЕТНОЕ ОБРАЗОВАТЕЛЬНОЕ УЧРЕЖДЕНИЕ
ГОРОДА МОСКВЫ
ДОПОЛНИТЕЛЬНОГО ПРОФЕССИОНАЛЬНОГО ОБРАЗОВАНИЯ
ЦЕНТР ПРОФЕССИОНАЛЬНЫХ КВАЛИФИКАЦИЙ И СОДЕЙСТВИЯ
ТРУДОУСТРОЙСТВУ
«ПРОФЕССИОНАЛ»

ИТОГОВАЯ АТТЕСТАЦИОННАЯ РАБОТА
на тему
«Анализ данных с использованием Python»
(на примере анализа данных исследуемого продукта)
слушателя Яновского Владимира Александровича
группы № 013
по программе профессиональной переподготовки
«Python для анализа данных»

Москва, 2025

Цель исследования:

Цель данного проекта — выявить, какие признаки имеют наибольшее влияние на рейтинг вина, составляемого специализированным журналом Wine Enthusiast. Для анализа используется набор данных из Kaggle (профессиональное сообщество специалистов по обработке данных и машинному обучению). Датафрейм состоит из 13 признаков (2 числовых признака и 11 категориальных признаков).

Анализ данных предполагает последовательное выполнение следующих этапов исследования:

Столбцы данных

- Страна - страна происхождения вина.
- Описание — описание вкусового профиля вина.
- Обозначение - виноградник-поставщик винограда для изготовления вина.
- Рейтинг - оценка вина специализированным журналом Wine Enthusiast, выраженная в баллах по шкале от 1 до 100.
- Цена - цена одной бутылки вина.
- Провинция — регион (провинция, штат) производства вина.
- Регион 1 — зона виноделия в регионе (например, долина Напа в Калифорнии).
- Регион 2 — (необязательно) терруар виноделия - более конкретная область в винодельческом регионе (например, Резерфорд в долине Напа).
- Разновидность — сорт винограда, используемый в производстве вина (например, Пино Нуар).
- Винодельня — производитель вина.

1. Предобработка данных

Подготовка данных к анализу (очистка данных, трансформация данных, дополнение, оптимизация):

- Заменить названия столбцов (привести к нижнему регистру).
- Преобразовать данные в нужные типы. Описать, в каких столбцах заменили тип данных и почему.
- Обработать пропуски при необходимости.
- Внести новый столбец "Континенты" в случае необходимости
  country_to_continent = {
  'Italy':'Europe',
  'Portugal':'Europe',
  'US':'North America',
  'Spain':'Europe',
  'France':'Europe',
  'Germany':'Europe',
  'Argentina':'Latin America',
  'Chile':'Latin America',
  'Australia': 'Oceania',
  'Austria': 'Europe',
  'South Africa': 'Africa',
  'New Zealand': 'Oceania',
  'Israel': 'Asia',
  'Hungary':'Europe',
  'Greece':'Europe',
  'Romania':'Europe',
  'Mexico':'Latin America',
  'Canada':'North America',
  'Turkey': 'Asia',
  'Czech Republic': 'Europe',
  'Slovenia': 'Europe',
  'Luxembourg': 'Europe',
  'Croatia': 'Europe',
  'Georgia':'Europe',
  'Uruguay': 'Latin America',
  'England': 'Europe',
  'Lebanon': 'Asia',
  'Serbia': 'Europe',
  'Brazil': 'Latin America',

```
'Moldova': 'Europe',
'Morocco':'Africa',
'Peru':'Latin America',
'India':'Asia',
'Bulgaria':'Europe',
'Cyprus': 'Europe',
'Armenia':'Asia',
'Switzerland':'Europe',
'Bosnia and Herzegovina':'Europe',
'Ukraine':'Europe',
'Slovakia':'Europe',
'Macedonia':'Europe',
'China':'Asia',
'Egypt':'Africa'
}
```

Импорт необходимых библиотек

In [5]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
import matplotlib.cm as cm
import matplotlib as mpl
import scipy.stats as st

# Импорт библиотеки warnings
import warnings
warnings.simplefilter(action='ignore', category=FutureWarning)
```

Загрузка данных

In [6]:
```
df = pd.read_csv('wine_reviews.csv')
df
```

Out[6]:

	country	description	designation	points	price	province	region_1	re
0	US	With a delicate, silky mouthfeel and bright ac...	NaN	86	23.0	California	Central Coast	
1	Italy	D'Alceo is a drop dead gorgeous wine that ooze...	D'Alceo	96	275.0	Tuscany	Toscana	
2	France	The great dominance of Cabernet Sauvignon in t...	NaN	91	40.0	Bordeaux	Haut-Médoc	
3	Italy	The modest cherry, dark berry and black tea no...	NaN	81	15.0	Tuscany	Chianti Classico	
4	US	Exceedingly light in color, scent and flavor, ...	NaN	83	25.0	Oregon	Rogue Valley	So
...	...	...	...	...	...	...	...	
19995	France	Firm wine, with tannins to match the chunky st...	Mansois	88	12.0	Southwest France	Marcillac	
19996	US	The vineyard is on the Napa side of Carneros. ...	Estate Vineyard	89	50.0	California	Carneros	S
19997	Italy	Lighea is a terrific wine and an excellent pai...	Lighea	87	20.0	Sicily & Sardinia	Sicilia	
19998	Italy	Organically farmed Cannonau grapes deliver sma...	Le Sabbie	87	NaN	Sicily & Sardinia	Cannonau di Sardegna	

	country	description	designation	points	price	province	region_1	re
19999	US	Grown on the Sonoma side of the appellation, i...	NaN	92	35.0	California	Carneros	S

20000 rows × 10 columns

In [7]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 10 columns):
 #   Column       Non-Null Count  Dtype  
---  ------       --------------  -----  
 0   country      20000 non-null  object 
 1   description  20000 non-null  object 
 2   designation  13999 non-null  object 
 3   points       20000 non-null  int64  
 4   price        18198 non-null  float64
 5   province     20000 non-null  object 
 6   region_1     16543 non-null  object 
 7   region_2     8058 non-null   object 
 8   variety      20000 non-null  object 
 9   winery       20000 non-null  object 
dtypes: float64(1), int64(1), object(8)
memory usage: 1.5+ MB
```

Количество значений в столбцах различается. Это говорит о том, что в данных есть пустые значения. Признак points и price числовые. С помощью библиотеки Seaborn построим тепловую карту для визуализации данных.

In [8]:
```
colours = ['#993366', '#FFFF00']
sns.heatmap(df.isnull(), cmap=sns.color_palette(colours))
# Decorations
plt.title('Матрица пропущенных значений набора данных', fontsize=14, fontname=
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.figtext(0.45, -0.25, "Рисунок 1 - Матрица пропущенных значений набора данн
plt.show()
```

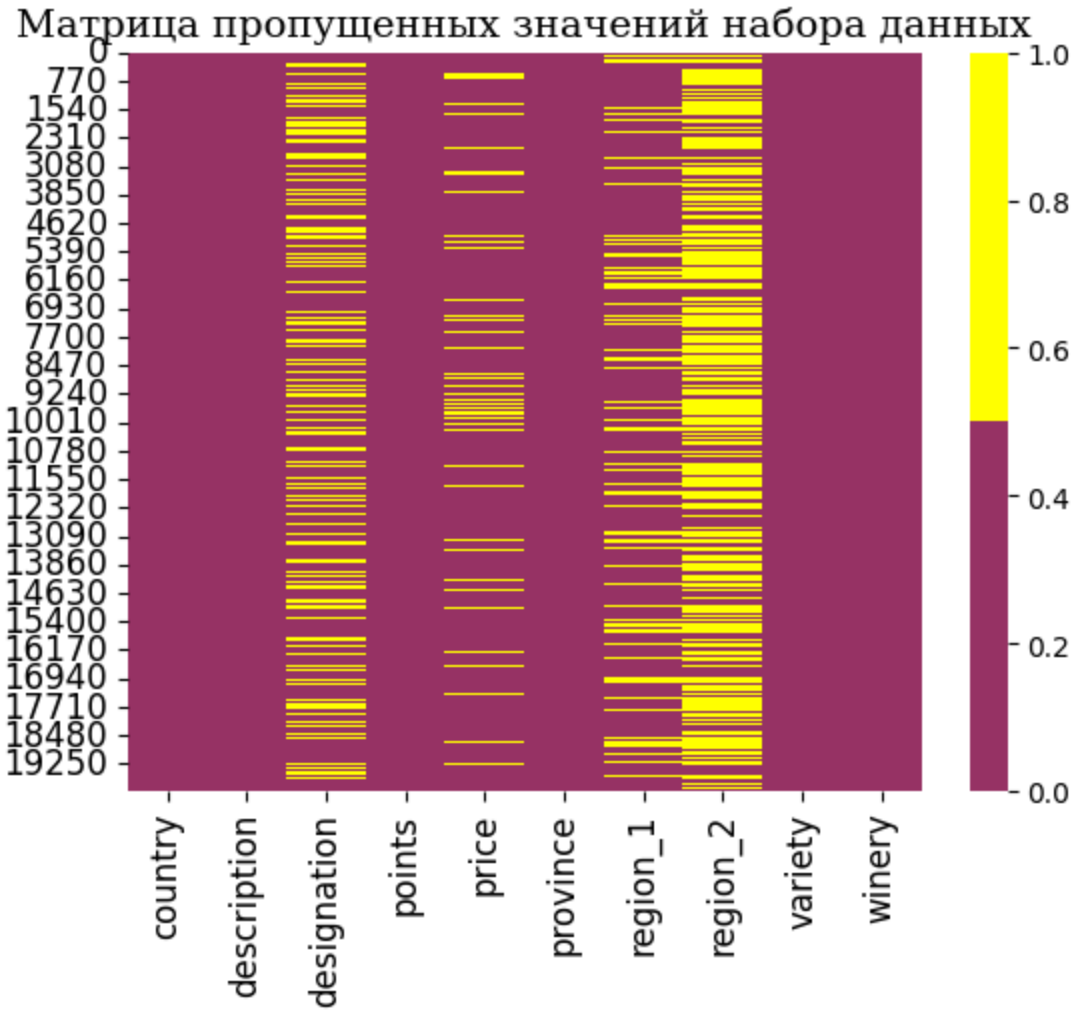

Рисунок 1 - Матрица пропущенных значений набора данных

Подсчёт размерности данных (количество строк и стоблцов)

In [9]:
```
print(f"Количество столбцов: {len(df.count(axis='rows'))}", f"Количество строк
```

```
Количество столбцов: 10
Количество строк: 20000
```

Замена названия столбцов и приведение их к нижнему регистру

In [10]:
```
# Переименование столбцов и приведение их к нижнему регистру
df.columns = [
    'страна',
    'описание',
    'обозначение',
    'рейтинг',
    'цена',
    'провинция',
    'регион_1',
```

```
    'регион_2',
    'разновидность',
    'винодельня'
]
```

Добавление нового столбца "Континенты" (тоже делаем в нижнем регистре для консистентности) в DataFrame (df) на основе словаря country_to_continent, с присвоением значения 'Other' для неизвестных стран

In [11]:
```
# Словарь соответствий стран и континентов
country_to_continent = {
    'Italy':'Europe',
    'Portugal':'Europe',
    'US':'North America',
    'Spain':'Europe',
    'France':'Europe',
    'Germany':'Europe',
    'Argentina':'Latin America',
    'Chile':'Latin America',
    'Australia': 'Oceania',
    'Austria': 'Europe',
    'South Africa': 'Africa',
    'New Zealand': 'Oceania',
    'Israel': 'Asia',
    'Hungary':'Europe',
    'Greece':'Europe',
    'Romania':'Europe',
    'Mexico':'Latin America',
    'Canada':'North America',
    'Turkey': 'Asia',
    'Czech Republic': 'Europe',
    'Slovenia': 'Europe',
    'Luxembourg': 'Europe',
    'Croatia': 'Europe',
    'Georgia':'Europe',
    'Uruguay': 'Latin America',
    'England': 'Europe',
    'Lebanon': 'Asia',
    'Serbia': 'Europe',
    'Brazil': 'Latin America',
    'Moldova': 'Europe',
    'Morocco':'Africa',
    'Peru':'Latin America',
    'India':'Asia',
    'Bulgaria':'Europe',
    'Cyprus': 'Europe',
    'Armenia':'Asia',
    'Switzerland':'Europe',
    'Bosnia and Herzegovina':'Europe',
    'Ukraine':'Europe',
    'Slovakia':'Europe',
    'Macedonia':'Europe',
```

```
    'China':'Asia',
    'Egypt':'Africa'
}

# Добавление нового столбца "Континенты"
df['континенты'] = df['страна'].map(country_to_continent).fillna('Other')
```

Проверка DataFrame (df) на наличие нового столбца "континенты", замену названия столбцов и приведение их к нижнему регистру

In [12]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 11 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   страна         20000 non-null  object 
 1   описание       20000 non-null  object 
 2   обозначение    13999 non-null  object 
 3   рейтинг        20000 non-null  int64  
 4   цена           18198 non-null  float64
 5   провинция      20000 non-null  object 
 6   регион_1       16543 non-null  object 
 7   регион_2       8058 non-null   object 
 8   разновидность  20000 non-null  object 
 9   винодельня     20000 non-null  object 
 10  континенты     20000 non-null  object 
dtypes: float64(1), int64(1), object(9)
memory usage: 1.7+ MB
```

Добавление нового столбеца цвет_вина в DataFrame (df) на основе словаря соответствий цвета вина и сорта вина

In [13]:
```
# Словарь соответствий цвета вина и сорта вина "разновидность"

цвет_вина = {
    "Chardonnay": "white",
    "Pinot Noir": "red",
    "Cabernet Sauvignon": "red",
    "Red Blend": "red",
    "Bordeaux-style Red Blend": "red",
    "Sauvignon Blanc": "white",
    "Syrah": "red",
    "Riesling": "white",
    "Merlot": "red",
    "Zinfandel": "red",
    "Sangiovese": "red",
    "Malbec": "red",
    "White Blend": "white",
    "Rosé": "other",
    "Tempranillo": "red",
```

```
    "Nebbiolo": "red",
    "Portuguese Red": "red",
    "Sparkling Blend": "other",
    "Shiraz": "red",
    "Corvina, Rondinella, Molinara": "red",
    "Rhône-style Red Blend": "red",
    "Barbera": "red",
    "Pinot Gris": "white",
    "Viognier": "white",
    "Bordeaux-style White Blend": "white",
    "Champagne Blend": "other",
    "Port": "red",
    "Grüner Veltliner": "white",
    "Gewürztraminer": "white",
    "Portuguese White": "white",
    "Petite Sirah": "red",
    "Carmenère": "red"
}

# Добавление нового столбца цвет_вина на основе сорта вина "разновидность"
df["цвет_вина"] = df["разновидность"].map(цвет_вина)
```

Проверка DataFrame (df) на наличие нового столбца "цвет_вина"

In [14]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 12 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   страна         20000 non-null  object 
 1   описание       20000 non-null  object 
 2   обозначение    13999 non-null  object 
 3   рейтинг        20000 non-null  int64  
 4   цена           18198 non-null  float64
 5   провинция      20000 non-null  object 
 6   регион_1       16543 non-null  object 
 7   регион_2       8058 non-null   object 
 8   разновидность  20000 non-null  object 
 9   винодельня     20000 non-null  object 
 10  континенты     20000 non-null  object 
 11  цвет_вина      16499 non-null  object 
dtypes: float64(1), int64(1), object(10)
memory usage: 1.8+ MB
```

Обработка пропусков в столбцах "обозначение" и "цвкт_вина"

In [15]:
```
df["обозначение"] = df["разновидность"].map(цвет_вина).fillna("unknown")
df["цвет_вина"] = df["разновидность"].map(цвет_вина).fillna("unknown")
```

Проверка DataFrame (df) на обработанные пропуски

In [16]:
```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 20000 entries, 0 to 19999
Data columns (total 12 columns):
 #   Column         Non-Null Count  Dtype  
---  ------         --------------  -----  
 0   страна         20000 non-null  object 
 1   описание       20000 non-null  object 
 2   обозначение    20000 non-null  object 
 3   рейтинг        20000 non-null  int64  
 4   цена           18198 non-null  float64
 5   провинция      20000 non-null  object 
 6   регион_1       16543 non-null  object 
 7   регион_2       8058 non-null   object 
 8   разновидность  20000 non-null  object 
 9   винодельня     20000 non-null  object 
 10  континенты     20000 non-null  object 
 11  цвет_вина      20000 non-null  object 
dtypes: float64(1), int64(1), object(10)
memory usage: 1.8+ MB
```

Подсчёт количества пропущенных значений в каждой переменной

In [17]:
```
df.isnull().sum()
```

Out[17]:

	0
страна	0
описание	0
обозначение	0
рейтинг	0
цена	1802
провинция	0
регион_1	3457
регион_2	11942
разновидность	0
винодельня	0
континенты	0
цвет_вина	0

dtype: int64

Тепловая карта визуализации данных после добавления столбцов и обработки данных

In [18]:
```
colours = ['#993366', '#FFFF00']
sns.heatmap(df.isnull(), cmap=sns.color_palette(colours))
# Decorations
plt.title('Матрица пропущенных значений набора данных после добавления столбцо
plt.xticks(fontsize=12)
plt.yticks(fontsize=12)
plt.figtext(0.45, -0.25, "Рисунок 2 - Матрица пропущенных значений набора данн
plt.show()
```

Матрица пропущенных значений набора данных после добавления столбцов и обработки пропусков

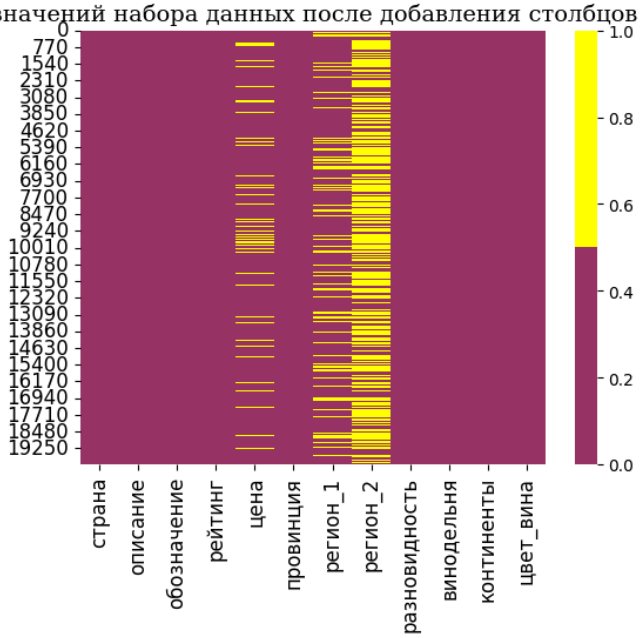

Рисунок 2 - Матрица пропущенных значений набора данных после добавления столбцов и обработки пропусков

Раздел 1. Предобработка данных - выводы

Общие изменения в данных:

1. Количество столбцов увеличилось:

- Было: 10 столбцов;
- Стало: 12 столбцов.

2. Переименование колонок:

- Наименования столбцов были изменены в соответствии с полученными данными (переведены на русский язык для удобства) и приведены к нижнему регистру.

3. Добавлены новые признаки:

- континенты — информация о континенте по стране;
- цвет_вина — классификация по цвету вина на основе сорта (variety

/ разновидность).

Как изменения повлияли на данные:

- Данные стали более информативными за счёт новых признаков;
- Все строки получили значения в цвет вина, что позволит проводить сравнения между белыми, красными и "другими" винами;
- Переименование столбцов упрощает интерпретацию и использование данных в дальнейшем анализе.

2. Исследовательский анализ данных

- Найти среднюю цену вина по региону.
- Выбрать сорта с наибольшими ценами.
- Найти среднюю цену вина для региона.
- Определить, популярные сорта вина в бюджетном сегменте.
- Определить, какие сорта вина лидируют по рейтингам.
- Построить график «ящик с усами» по рейтингам в разбивке по странам, по сортам вина.
- Выявить закономерность влияния на цену цвета и рейтинга вина.
- Построить диаграмму рассеяния и посчитать корреляцию.

Находим среднюю цену вина по региону.

In [19]:
```
# Группировка по региону и вычисление средней цены
avg_price_by_region = df.groupby('провинция')['цена'].mean().sort_values(ascer

# Вывод первых 10 строк
print(avg_price_by_region.head(10))
```

```
провинция
Tokaji                        133.100000
Champagne                      99.342466
Santa Cruz                     95.000000
Israel                         70.000000
Burgundy                       69.713303
Wachau                         67.414634
Middle and South Dalmatia      65.000000
Martinborough Terrace          60.000000
Port                           53.373832
Rheingau                       53.339623
Name: цена, dtype: float64
```

In [20]:
```
# Группировка и расчёт средней цены
```

```
avg_price_by_region = df.groupby('провинция')['цена'].mean().dropna().sort_val

# Устанавливаем шрифт Serif
mpl.rcParams['font.family'] = 'serif'

# Отображение топ-20 регионов с самой высокой средней ценой
plt.figure(figsize=(8, 4))
sns.barplot(x=avg_price_by_region.head(20).values, y=avg_price_by_region.head(

# Подписи и легенда
plt.xlabel('Средняя цена, $', fontsize=12, fontname='serif')
plt.ylabel('Регион', fontsize=12, fontname='serif')
plt.title('Топ-20 регионов по средней цене вина', fontsize=14, fontname='serif
plt.tight_layout()
plt.figtext(0.5, -0.05, 'Рисунок 3 - Топ-20 регионов по средней цене вина', ha
plt.show()
```

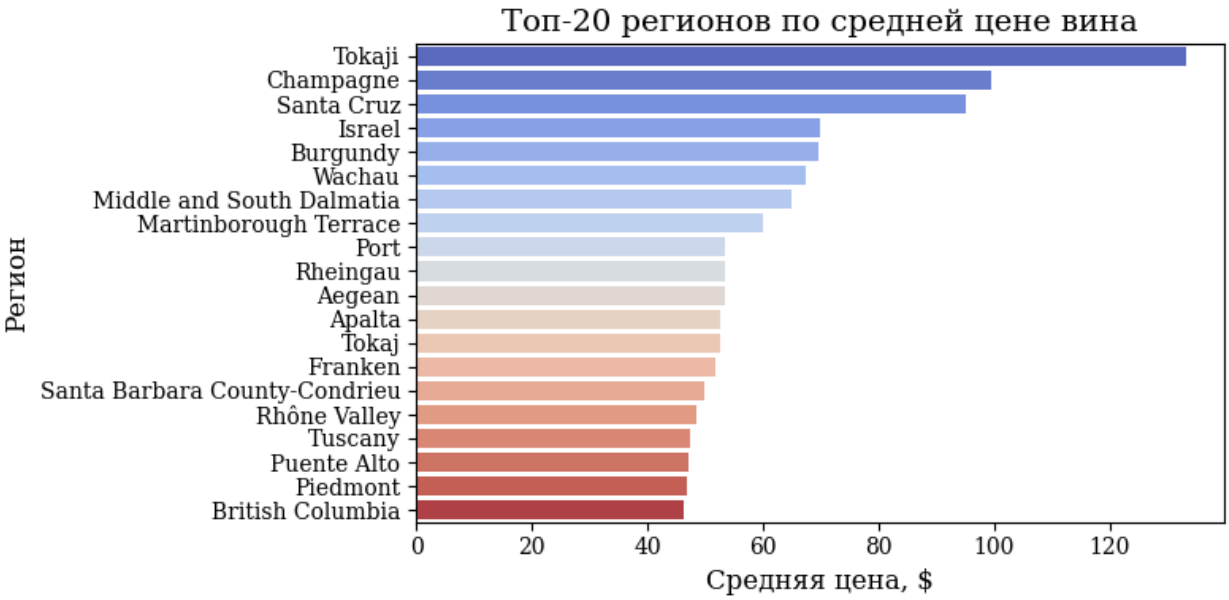

Рисунок 3 - Топ-20 регионов по средней цене вина

Выбрать сорта вин с наибольшими средними ценами.

In [21]:
```
# Группировка по сорту винограда и расчёт средней цены
top_varieties_by_price = df.groupby('разновидность')['цена'].mean().dropna().s

# Вывести топ-10 самых дорогих сортов
print(top_varieties_by_price.head(10))
```

```
разновидность
Furmint             205.250000
Tokay               160.666667
Debit               130.000000
Tinto Fino           89.000000
Mission              82.000000
Champagne Blend      81.024000
Tinto del Pais       79.363636
Syrah-Viognier       75.000000
Picolit              75.000000
Nebbiolo             69.994819
Name: цена, dtype: float64
```

In [22]:
```
# Устанавливаем шрифт Serif
mpl.rcParams['font.family'] = 'serif'

# Построение графика
plt.figure(figsize=(8, 4))
sns.barplot(x=top_varieties_by_price.head(20).values, y=top_varieties_by_price

# Подписи и легенда
plt.xlabel('Средняя цена, $', fontsize=12, fontname='serif')
plt.ylabel('Сорт вина', fontsize=12, fontname='serif')
plt.title('Топ-20 сортов вин по средней цене', fontsize=14, fontname='serif')
plt.tight_layout()
plt.figtext(0.5, -0.05, 'Рисунок 4 - Топ-20 сортов вин по средней цене', ha='c
plt.show()
```

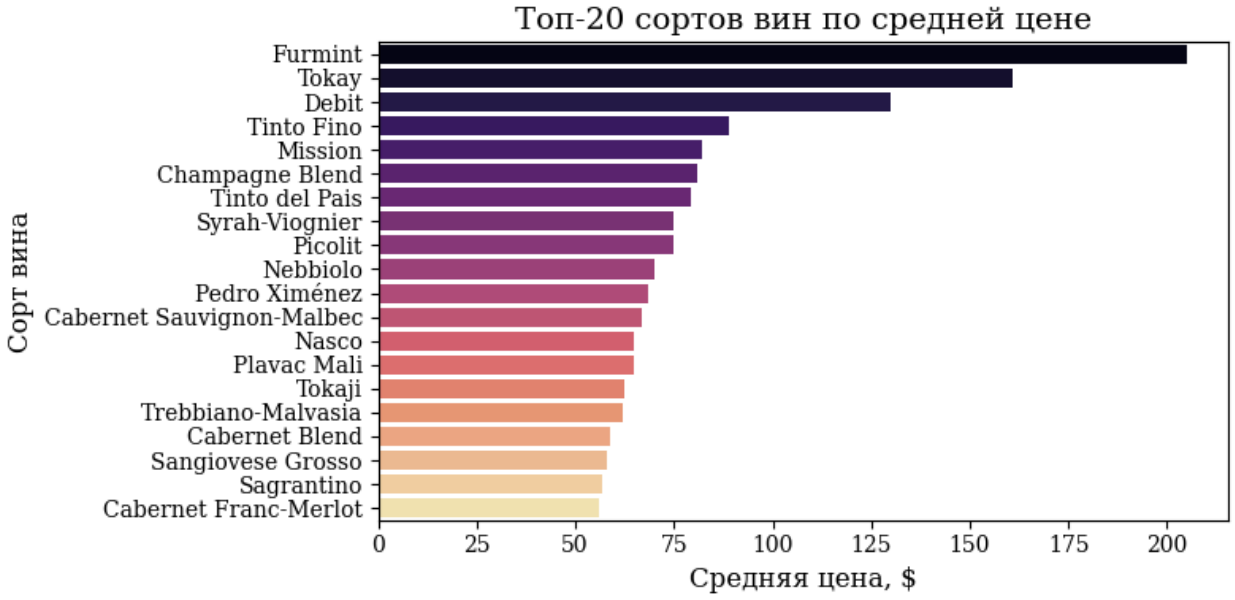

Рисунок 4 - Топ-20 сортов вин по средней цене

Найти среднюю цену вина для каждого региона.

In [23]:
```
# Группировка по региону и вычисление средней цены
avg_price_by_region = df.groupby('провинция')['цена'].mean().dropna().sort_val
```

```
# Просмотр результата
print(avg_price_by_region)
```

```
провинция
Tokaji                 133.100000
Champagne               99.342466
Santa Cruz              95.000000
Israel                  70.000000
Burgundy                69.713303
                           ...
Northwestern Italy       8.000000
Table wine               8.000000
Felso-Magyarország       7.000000
Recas                    7.000000
Primorska                7.000000
Name: цена, Length: 307, dtype: float64
```

In [24]:
```
# Группировка по региону и расчёт средней цены
avg_price_by_region = df.groupby('провинция')['цена'].mean().dropna().sort_val

# Устанавливаем шрифт Serif
mpl.rcParams['font.family'] = 'serif'

# Визуализация топ-20 регионов с самой высокой средней ценой
plt.figure(figsize=(8, 4))
sns.barplot(x=avg_price_by_region.head(20).values, y=avg_price_by_region.head(

# Подписи и легенда
plt.xlabel('Средняя цена, $', fontsize=12, fontname='serif')
plt.ylabel('Регион', fontsize=12, fontname='serif')
plt.title('Топ-20 регионов по средней цене вина', fontsize=14, fontname='serif
plt.tight_layout()
plt.figtext(0.5, -0.05, 'Рисунок 5 - Топ-20 регионов по средней цене вина', wr
plt.show()
```

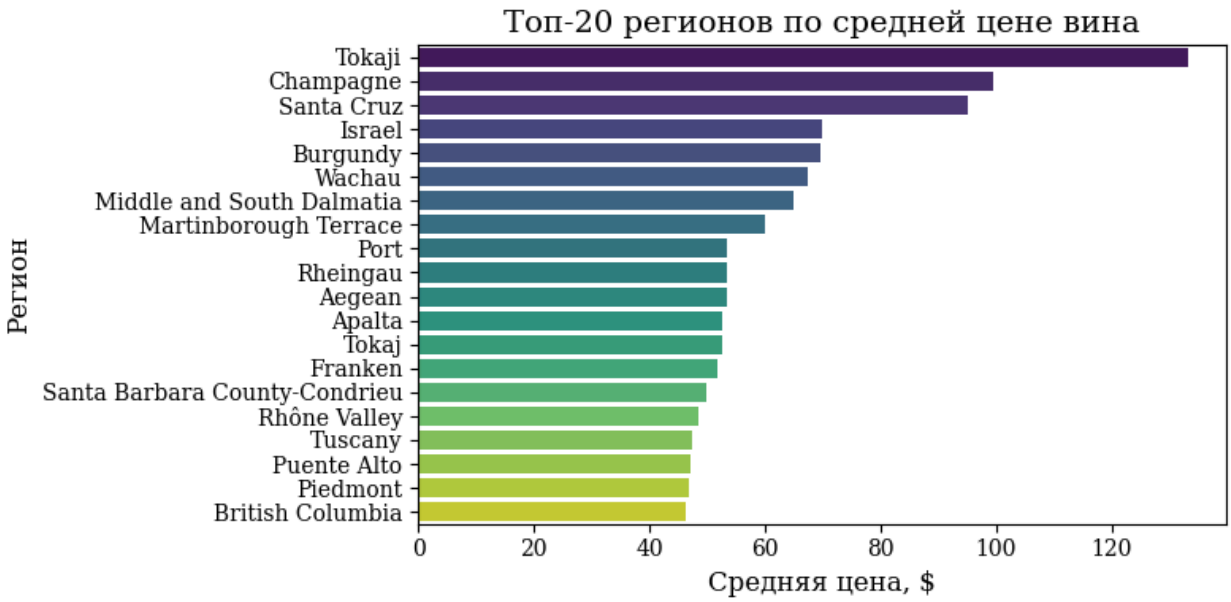

Рисунок 5 - Топ-20 регионов по средней цене вина

Популярные сорта вина в бюджетном сегменте до 20 у.е.

In [25]:
```
# Определим бюджетный сегмент: вина дешевле 20 у.е.
бюджетный_порог = 20
бюджетные_вина = df[df['цена'] < бюджетный_порог]

# Найдём топ-20 самых популярных сортов вина среди бюджетных
популярные_сорта_бюджет = (
    бюджетные_вина['разновидность']
    .value_counts()
    .head(20)
)

популярные_сорта_бюджет
```

Out[25]:

разновидность	count
Chardonnay	696
Sauvignon Blanc	505
Cabernet Sauvignon	447
Red Blend	365
Riesling	310
Rosé	288
Merlot	273
Pinot Noir	236
Malbec	208
White Blend	206
Zinfandel	154
Syrah	151
Tempranillo	146
Pinot Grigio	145
Portuguese Red	138
Bordeaux-style Red Blend	118
Sparkling Blend	113
Pinot Gris	110
Sangiovese	110
Shiraz	103

dtype: int64

In [26]:
```
# Устанавливаем шрифт Serif
mpl.rcParams['font.family'] = 'serif'

# Построим горизонтальный barplot
plt.figure(figsize=(8, 4))
sns.barplot(
    x=популярные_сорта_бюджет.values,
    y=популярные_сорта_бюджет.index,
    palette='coolwarm'
)

# Подписи и легенда
plt.title('Топ-20 популярных сортов вина в бюджетном сегменте', fontsize=14, f
```

```
plt.xlabel('Количество', fontsize=12, fontname='serif')
plt.ylabel('Сорт', fontsize=12, fontname='serif')
plt.grid(axis='x', linestyle='--', alpha=0.6)
plt.figtext(0.5, -0.05, 'Рисунок 6 - Топ-20 популярных сортов вина в бюджетном
plt.tight_layout()
plt.show()
```

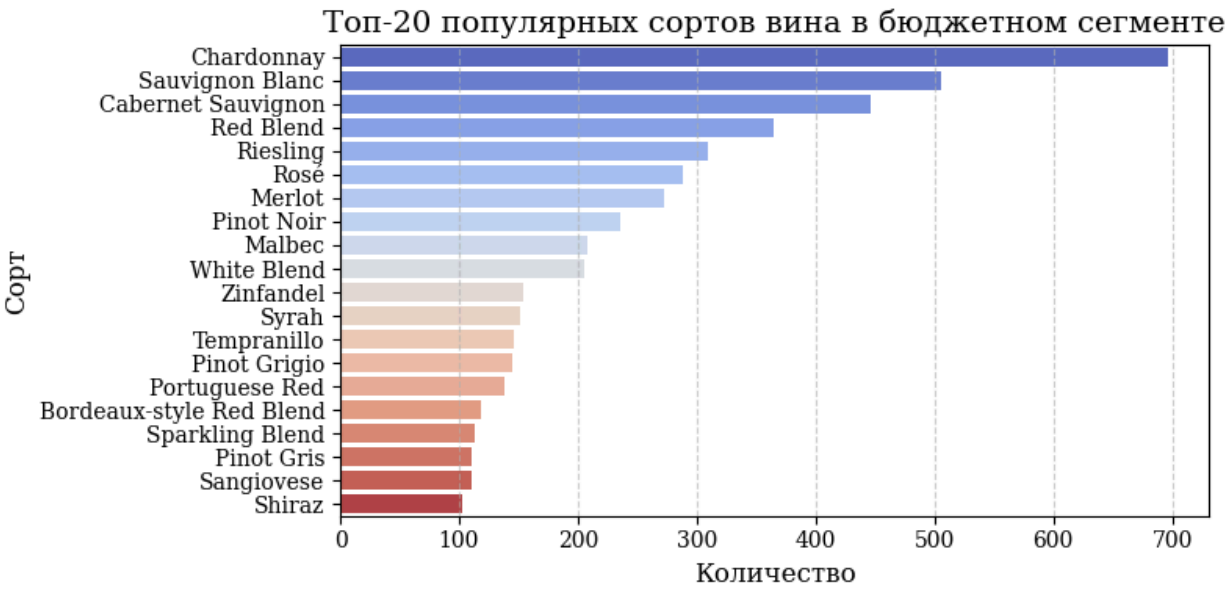

Рисунок 6 - Топ-20 популярных сортов вина в бюджетном сегменте

Какие сорта вина лидируют по рейтингам.

In [27]:
```
# Группируем по сорту и считаем средний рейтинг
топ_по_рейтингу = (
    df.groupby('разновидность')['рейтинг']
    .mean()
    .sort_values(ascending=False)
    .head(20)
)

топ_по_рейтингу
```

Out[27]:

разновидность	рейтинг
Tokay	96.000000
Muskat Ottonel	94.000000
Sauvignon Gris	94.000000
Roussanne-Viognier	93.000000
Marsanne-Viognier	93.000000
Scheurebe	92.666667
Carricante	92.500000
Tinto Fino	92.166667
Nasco	92.000000
Teroldego	92.000000
Aleatico	92.000000
Black Monukka	92.000000
Grenache-Shiraz	92.000000
Susumaniello	92.000000
Provence red blend	92.000000
Syrah-Viognier	92.000000
Albana	92.000000
Vespaiolo	92.000000
Malbec-Cabernet Franc	92.000000
Rieslaner	91.500000

dtype: float64

In [28]:
```
# Устанавливаем шрифт Serif
mpl.rcParams['font.family'] = 'serif'

# Построим barplot
plt.figure(figsize=(9, 4))
sns.barplot(
    x=топ_по_рейтингу.values,
    y=топ_по_рейтингу.index,
    palette='viridis'
)

# Подписи и легенда
plt.title('Топ-20 сортов вина по среднему рейтингу', fontsize=14, fontname='se
```

```
plt.xlabel('Средний рейтинг', fontsize=12, fontname='serif')
plt.ylabel('Сорт', fontsize=12, fontname='serif')
plt.grid(axis='x', linestyle='--', alpha=0.5)
plt.figtext(0.5, -0.05, 'Рисунок 7 - Топ-20 сортов вина по среднему рейтингу',
plt.tight_layout()
plt.show()
```

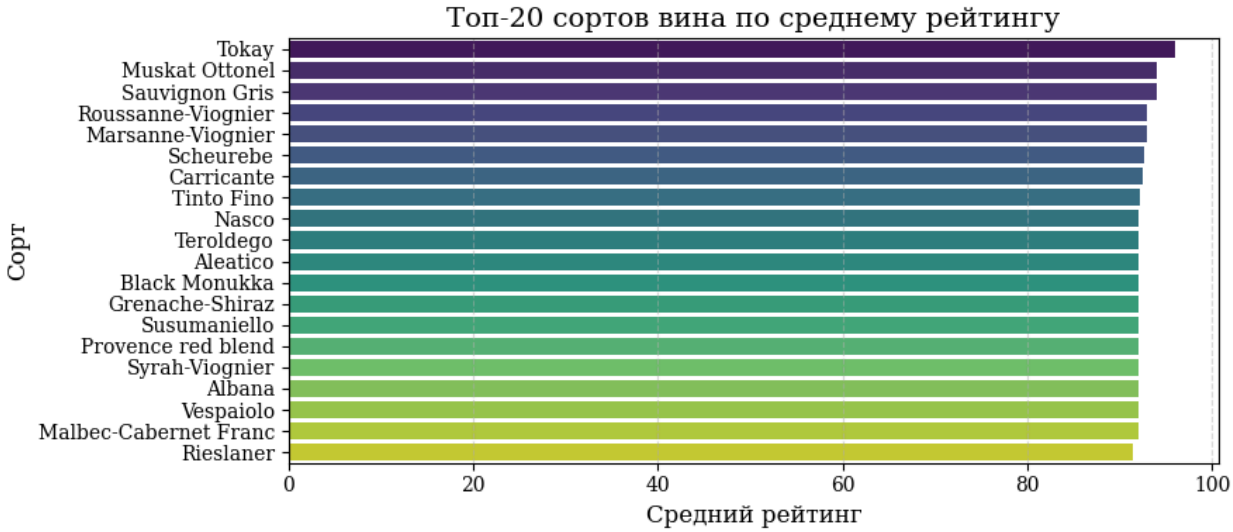

Рисунок 7 - Топ-20 сортов вина по среднему рейтингу

График «ящик с усами» по рейтингам в разбивке по странам, по сортам вина.

In [29]:
```
# 1. Boxplot по рейтингам в разбивке по странам

# Установим стиль и размер графика
plt.figure(figsize=(10, 5))
mpl.rcParams['font.family'] = 'serif'

# Страны с наибольшим количеством наблюдений
топ_страны = df['страна'].value_counts().head(10).index
df_топ_страны = df[df['страна'].isin(топ_страны)]

# Построим boxplot
sns.boxplot(
    x='страна',
    y='рейтинг',
    data=df_топ_страны,
    palette='Set3'
)

# Подписи и легенда
plt.title('Распределение рейтингов по странам (Топ-10)', fontsize=14, fontname
plt.xlabel('Страна', fontsize=12, fontname='serif')
plt.ylabel('Рейтинг', fontsize=12, fontname='serif')
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.figtext(0.5, -0.05, 'Рисунок 8 - Ящик с усами по странам', ha='center', fo
plt.tight_layout()
```

```
plt.show()
```

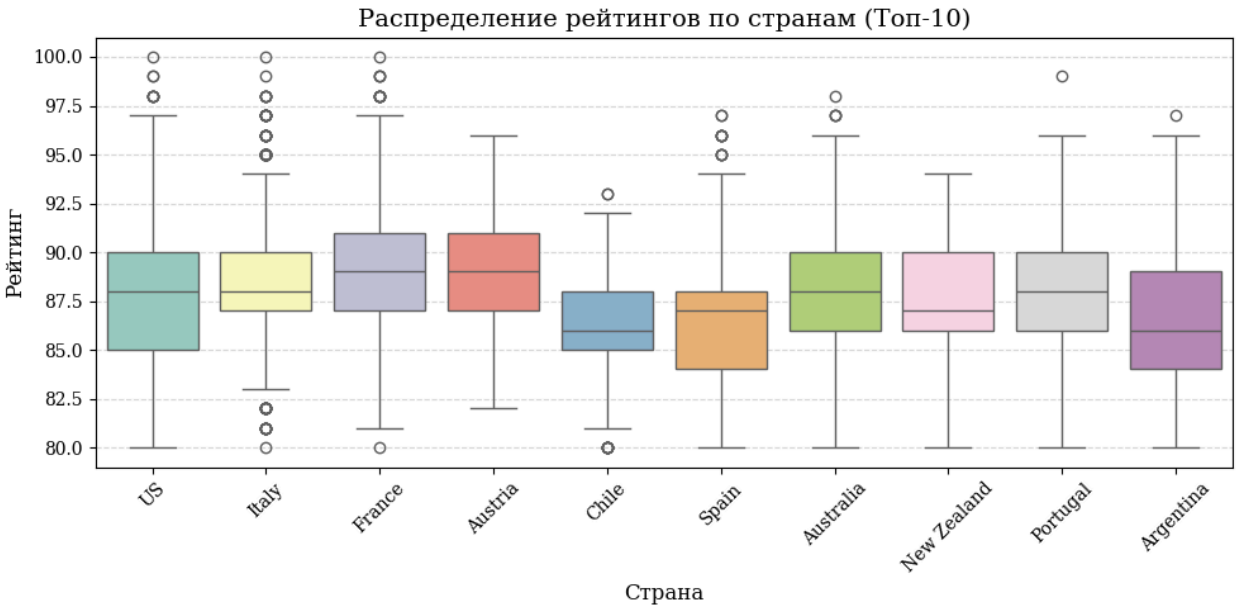

Рисунок 8 - Ящик с усами по странам

In [30]:
```
# 2. Boxplot по рейтингам в разбивке по сортам вина

# Установим стиль и размер графика
plt.figure(figsize=(10, 5))
mpl.rcParams['font.family'] = 'serif'

# Сортируем по популярности сортов
топ_сорта = df['разновидность'].value_counts().head(10).index
df_топ_сорта = df[df['разновидность'].isin(топ_сорта)]

sns.boxplot(
    x='разновидность',
    y='рейтинг',
    data=df_топ_сорта,
    palette='Set2'
)

# Подписи и легенда
plt.title('Распределение рейтингов по сортам вина (Топ-10)', fontsize=14, font
plt.xlabel('Сорт вина', fontsize=12, fontname='serif')
plt.ylabel('Рейтинг', fontsize=12, fontname='serif')
plt.xticks(rotation=45)
plt.grid(axis='y', linestyle='--', alpha=0.5)
plt.figtext(0.5, -0.05, 'Рисунок 9 - Ящик с усами по сортам вина', ha='center'
plt.tight_layout()
plt.show()
```

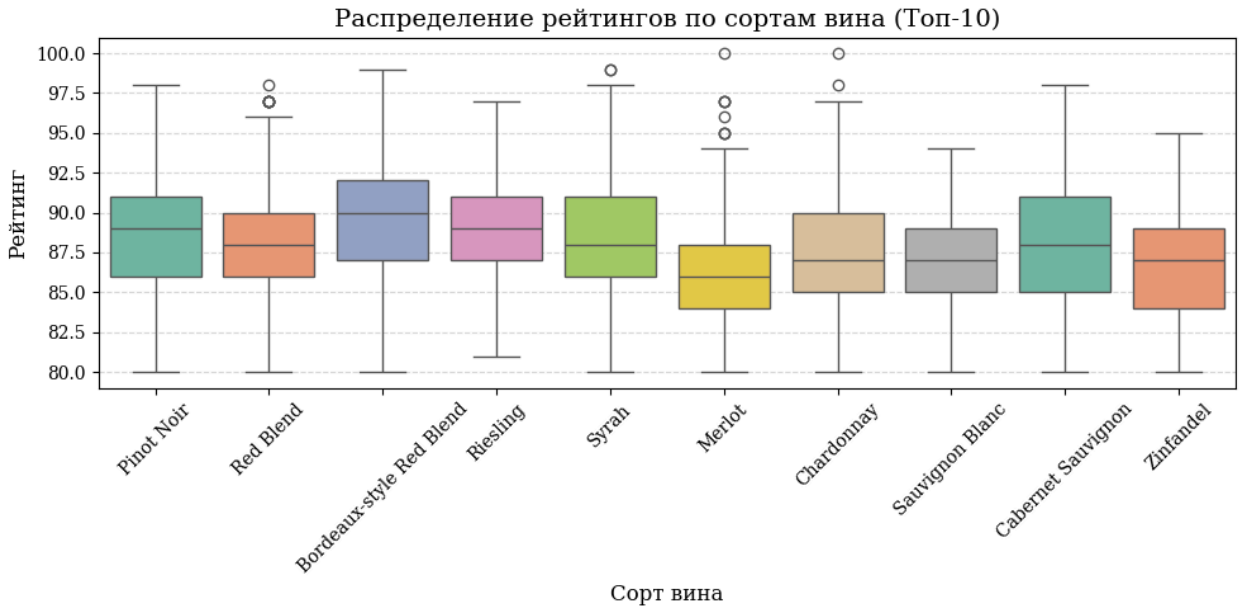

Рисунок 9 - Ящик с усами по сортам вина

Выявление закономерности влияния на цену цвета и рейтинга вина (только для красного и белого вина).

In [31]:
```
# Оставляем только красное и белое вина
df_filtered = df[df['цвет_вина'].isin(['red', 'white'])].copy()

# Убираем пропуски в цене и рейтинге
df_filtered = df_filtered.dropna(subset=['цена', 'рейтинг'])

# Настройка шрифта
mpl.rcParams['font.family'] = 'serif'

# Цветовая палитра
palette = {'red': 'red', 'white': 'gold'}

# Построение scatter plot
plt.figure(figsize=(8, 4))
sns.scatterplot(
    data=df_filtered,
    x='рейтинг',
    y='цена',
    hue='цвет_вина',
    palette=palette,
    alpha=0.6
)

# Подписи и легенда
plt.title('Зависимость цены от рейтинга и цвета вина', fontsize=14, fontname='
plt.xlabel('Рейтинг', fontsize=12, fontname='serif')
plt.ylabel('Цена', fontsize=12, fontname='serif')
plt.grid(True, linestyle='--', alpha=0.5)
plt.legend(title='Цвет вина', title_fontsize=12, fontsize=10, bbox_to_anchor=(
```

```
plt.figtext(0.5, -0.05, 'Рисунок 10 - Влияние цвета и рейтинга на цену вина',
plt.tight_layout()
plt.show()
```

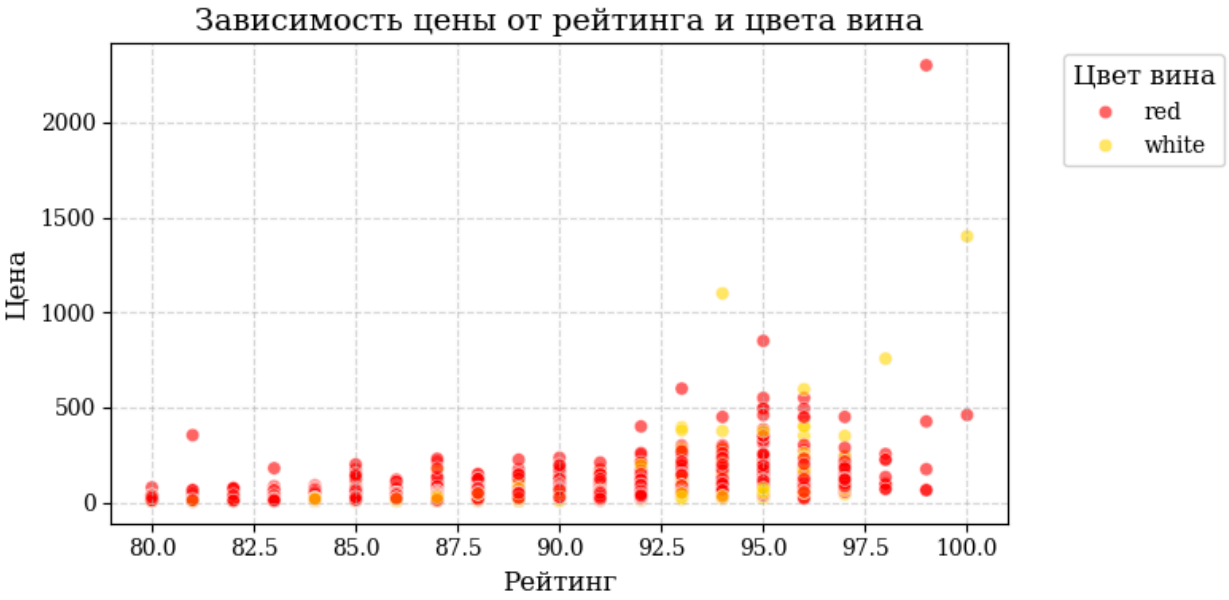

Рисунок 10 - Влияние цвета и рейтинга на цену вина

Построение диаграммы рассеяния между рейтингом и ценой вина (только для красного и белого вина), а также расчёт коэффициента корреляции отдельно для каждого цвета.

In [32]:
```
# Оставляем только красное и белое вина и убираем пропуски
df_filtered = df[df['цвет_вина'].isin(['red', 'white'])].copy()
df_filtered = df_filtered.dropna(subset=['цена', 'рейтинг'])

# Диаграмма рассеяния и Цветовая кодировка

# Настройка шрифта
mpl.rcParams['font.family'] = 'serif'

# Цвета
palette = {'red': 'red', 'white': 'gold'}

# Диаграмма рассеяния
plt.figure(figsize=(8, 4))
sns.scatterplot(
    data=df_filtered,
    x='рейтинг',
    y='цена',
    hue='цвет_вина',
    palette=palette,
    alpha=0.6
)

# Подписи и легенда
```

```
plt.title('Диаграмма рассеяния: Цена vs Рейтинг по цвету вина', fontsize=14, f
plt.xlabel('Рейтинг', fontsize=12, fontname='serif')
plt.ylabel('Цена', fontsize=12, fontname='serif')
plt.grid(True, linestyle='--', alpha=0.5)
plt.legend(title='Цвет вина', bbox_to_anchor=(1.05, 1), loc='upper left')
plt.figtext(0.5, -0.05, 'Рисунок 11 - Связь рейтинга и цены вина по цвету', ha
plt.tight_layout()
plt.show()
```

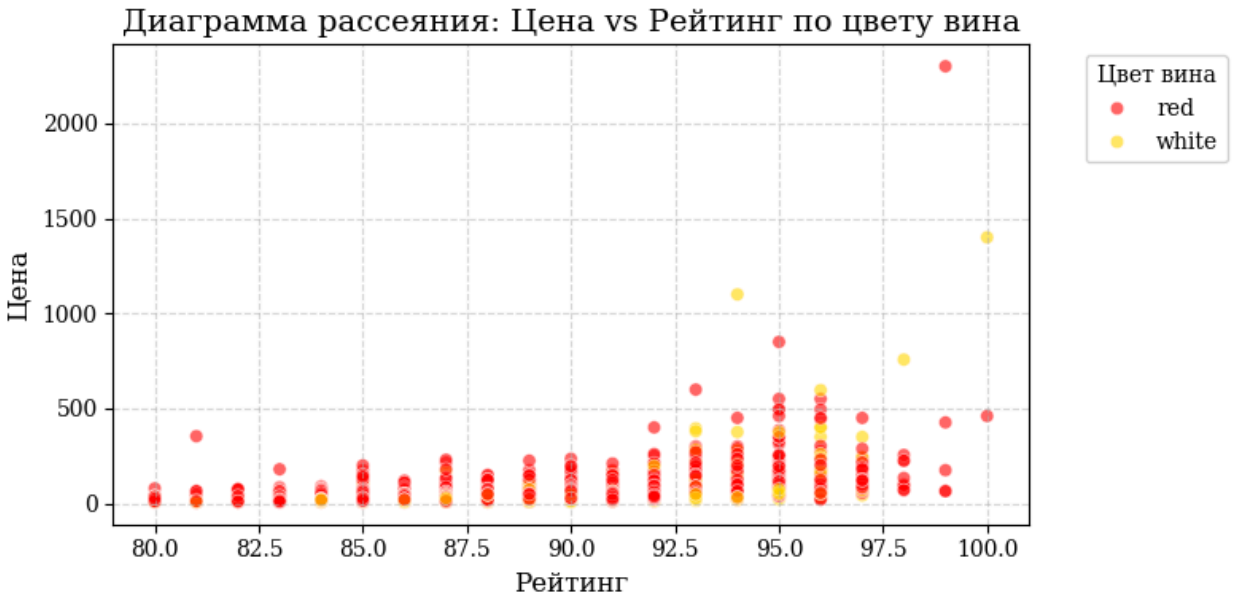

Рисунок 11 - Связь рейтинга и цены вина по цвету

In [33]:
```
# Корреляция по цвету вина

# Корреляция для красного вина
corr_red = df_filtered[df_filtered['цвет_вина'] == 'red'][['рейтинг', 'цена']]

# Корреляция для белого вина
corr_white = df_filtered[df_filtered['цвет_вина'] == 'white'][['рейтинг', 'цен

print(f"Корреляция (рейтинг vs цена) для красного вина: {corr_red:.3f}")
print(f"Корреляция (рейтинг vs цена) для белого вина: {corr_white:.3f}")
```

```
Корреляция (рейтинг vs цена) для красного вина: 0.431
Корреляция (рейтинг vs цена) для белого вина: 0.376
```

Раздел 2. Исследовательский анализ данных — выводы

В ходе исследовательского анализа были выявлены следующие закономерности и особенности в данных о винах:

1. Средняя цена вина по регионам

Анализ показал, что самые дорогие регионы по средней цене вина — это:

- Tokaji — 133.1
- Champagne — 99.3
- Santa Cruz — 95.0
- Israel — 70.0
- Burgundy — 69.7

Эти регионы известны премиальными винами, что отражается в ценах.

2. Сорта с наибольшими ценами (Самые дорогие сорта вина)

Наибольшая средняя цена зафиксирована у сортов:

- Furmint — 205.3
- Tokay — 160.7
- Debit — 130.0
- Tinto Fino — 89.0
- Mission — 82.0

Можно сделать вывод, что редкие или регионально-специфичные сорта имеют высокую стоимость.

3. Средняя цена по провинциям

Разброс средней цены по 307 провинциям показывает, что некоторые регионы выпускают как премиальные вина (до 133 у.е.), так и очень доступные (от 7 у.е.), что подчёркивает разнообразие рынка.

4. Популярные сорта в бюджетном сегменте

В категории недорогих вин (до 20 у.е.) наиболее часто встречаются:

- Chardonnay — 696 раз;
- Sauvignon Blanc — 505 раз;
- Cabernet Sauvignon — 447 раз;
- Red Blend — 365 раз;
- Riesling — 310 раз;

Это говорит о высокой доступности и популярности этих сортов у массового потребителя.

5. Сорта-лидеры по рейтингам

По наивысшим средним рейтингам лидируют редкие и премиальные сорта:

- Tokay — 96.0
- Muskat Ottonel, Sauvignon Gris — 94.0
- Roussanne-Viognier, Marsanne-Viognier — 93.0+

Это может быть связано с ограниченным производством и высокой оценкой экспертов.

6. Распределение рейтингов по странам и сортам

Ящики с усами по рейтингу для таких стран, как США, Италия, Франция, Австрия, Чили, Испания, Австралия, Новая Зеландия, Португалия, Аргентина показывают, что:

- Рейтинги схожи по медианам в большинстве стран;
- В некоторых странах наблюдаются более высокие выбросы (элитные вина).

Распределение рейтингов по сортам представлен ящиком с усами рис.9 для наиболее популярных сортов вина (Top-10): Pinot Noir, Cabernet Sauvignon и Bordeaux-style Red Blend имеют более высокий медианный рейтинг.

7. Влияние цвета вина и рейтинга на цену:

Проведённый анализ показал, что как рейтинг, так и цвет вина влияют на его цену. На рис.10 визуализирована эта зависимость. Красные вина (обозначены красным цветом) демонстрируют более высокую концентрацию в области повышенных цен при высоких рейтингах.

8. Диаграмма рассеяния и корреляция

Для более точной оценки зависимости была построена диаграмма рассеяния и рассчитана корреляция см. рис.11 – Связь рейтинга и цены вина по цвету:

- Корреляция между рейтингом и ценой для красного вина: 0.431;
- Корреляция между рейтингом и ценой для белого вина: 0.376.

Это говорит о наличии умеренной положительной связи: более высоко оценённые вина, как правило, стоят дороже, особенно среди красных.

Общий вывод:
Цены на вино зависят от сочетания сорта, региона, цвета и рейтинга. Красные вина в среднем дороже и демонстрируют более сильную связь между рейтингом и ценой. Региональные и сортовые различия оказывают существенное влияние на рыночную стоимость, что важно учитывать при

дальнейшем моделировании и анализе.

3. Составление структуры развития рынка вина регионов

- Самые популярные сорта (топ-5).
- Влияет ли рейтинг на цены по регионам?

In [34]:
```
# Топ-5 самых популярных сортов вина по количеству записей
top_varieties = df['разновидность'].value_counts().head(5)
print(top_varieties)

# Установка альтернативного serif-шрифта
plt.rcParams['font.family'] = 'serif'

# Визуализация
plt.figure(figsize=(8, 4))
sns.set_style("whitegrid")

top_varieties = df['разновидность'].value_counts().head(5)
sns.barplot(x=top_varieties.values, y=top_varieties.index, palette="Set2")

# Подписи и легенда
plt.title("Самые популярные сорта вина (Топ-5)", fontsize=14, fontname='serif'
plt.xlabel("Количество записей", fontsize=12, fontname='serif')
plt.ylabel("Сорт вина", fontsize=12, fontname='serif')
plt.figtext(0.5, -0.05, "Рисунок 12 - Популярные сорта вина", ha="center", for
plt.tight_layout()
plt.show()
```

```
разновидность
Pinot Noir                   1945
Chardonnay                   1893
Cabernet Sauvignon           1636
Red Blend                    1329
Bordeaux-style Red Blend      952
Name: count, dtype: int64
```

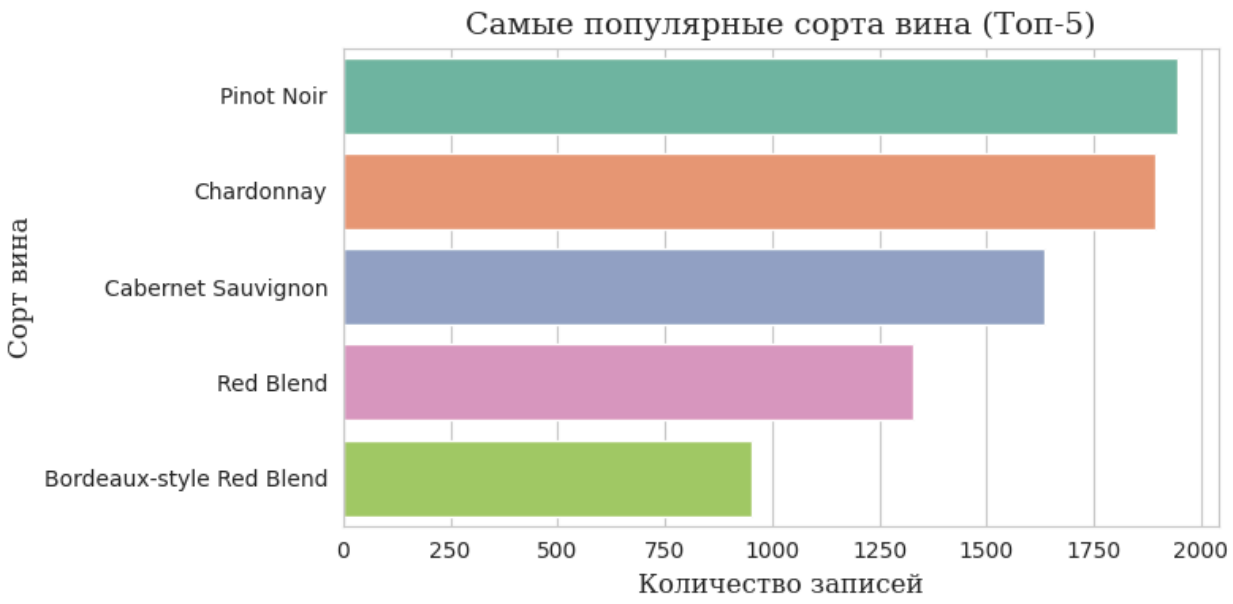

Рисунок 12 - Популярные сорта вина

In [35]:
```
#  1. Рассчитаем средние значения цен и рейтингов по регионам

# Группировка по провинции (региону) и расчёт средних значений
region_stats = df.groupby('провинция')[['цена', 'рейтинг']].mean().dropna()

# Считаем корреляцию между средней ценой и рейтингом по регионам
correlation = region_stats['цена'].corr(region_stats['рейтинг'])
print(f'Корреляция между рейтингом и ценой по регионам: {correlation:.3f}')
```

```
Корреляция между рейтингом и ценой по регионам: 0.498
```

In [36]:
```
# 2. Строим диаграмму рассеяния

# Установка стиля и шрифта
plt.rcParams['font.family'] = 'serif'
sns.set(style="whitegrid")

# Визуализация
plt.figure(figsize=(8, 4))
sns.scatterplot(data=region_stats, x='рейтинг', y='цена')

# Подписи и легенда
plt.title("Зависимость цены от рейтинга по регионам", fontsize=14, fontname='s
plt.xlabel("Средний рейтинг", fontsize=12, fontname='serif')
plt.ylabel("Средняя цена", fontsize=12, fontname='serif')
plt.figtext(0.5, -0.05, "Рисунок 13 - Связь рейтинга и цены по регионам", ha="
plt.tight_layout()
plt.show()
```

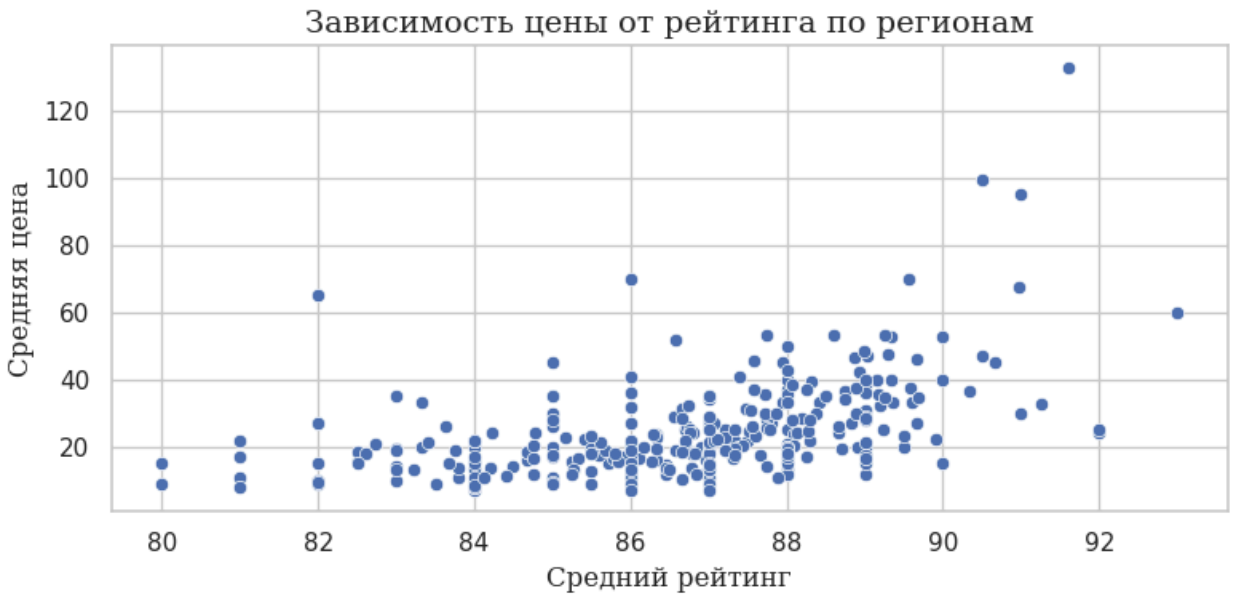

Рисунок 13 - Связь рейтинга и цены по регионам

Раздел 3. Составление структуры развития рынка вина регионов - выводы

Анализ показал, что на рынке вина преобладают следующие сорта:

- Pinot Noir — 1 945 отзывов;
- Chardonnay — 1 893 отзывов;
- Cabernet Sauvignon — 1 636 отзывов;
- Red Blend — 1 329 отзывов;
- Bordeaux-style Red Blend — 952 отзыва.

Эти сорта можно считать самыми популярными и массовыми на рынке, что говорит об их широкой доступности и потребительском спросе.
Дополнительно был проведён анализ зависимости средней цены от рейтинга по регионам. Наблюдается умеренная положительная корреляция 0.498 между рейтингом и ценой. Это говорит о том, что более высоко оценённые вина, как правило, имеют более высокую цену, но есть и другие факторы, влияющие на стоимость.
Таким образом, при развитии винного рынка и формировании ассортимента следует уделять особое внимание наиболее популярным сортам, а также учитывать, что рейтинг влияет на восприятие ценности продукта потребителем.

4. Исследование статистических показателей зависимости цены вина от рейтинга в регионе.

Построить линейную регрессию зависимости между ценой продукта и его рейтингом.

In [37]:
```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error, mean_absolute_error, r2_score

# Убедимся, что нужные столбцы не содержат пропущенных значений
df_reg = df[['рейтинг', 'цена']].dropna()

# Разделим выборку на обучающую и тестовую
X = df_reg[['рейтинг']]
y = df_reg['цена']
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, randc

# Обучение модели линейной регрессии
model = LinearRegression()
model.fit(X_train, y_train)

# Предсказания
y_pred = model.predict(X_test)

# Оценка модели
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print(f"MAE: {mae:.2f}")
print(f"MSE: {mse:.2f}")
print(f"RMSE: {rmse:.2f}")
print(f"R²: {r2:.4f}")

# Подписи и легенда
plt.figure(figsize=(8, 4))
sns.regplot(x='рейтинг', y='цена', data=df_reg, scatter_kws={'alpha':0.3}, lir
plt.title('Линейная регрессия: цена vs рейтинг', fontsize=14, fontname='serif'
plt.xlabel('Рейтинг', fontsize=12, fontname='serif')
plt.ylabel('Цена', fontsize=12, fontname='serif')
plt.grid(True)
plt.text(0.25, -0.25, 'Рисунок 14 - Линейная регрессия', transform=plt.gca().t
plt.show()
```

```
MAE: 16.77
MSE: 885.12
RMSE: 29.75
R²: 0.2477
```

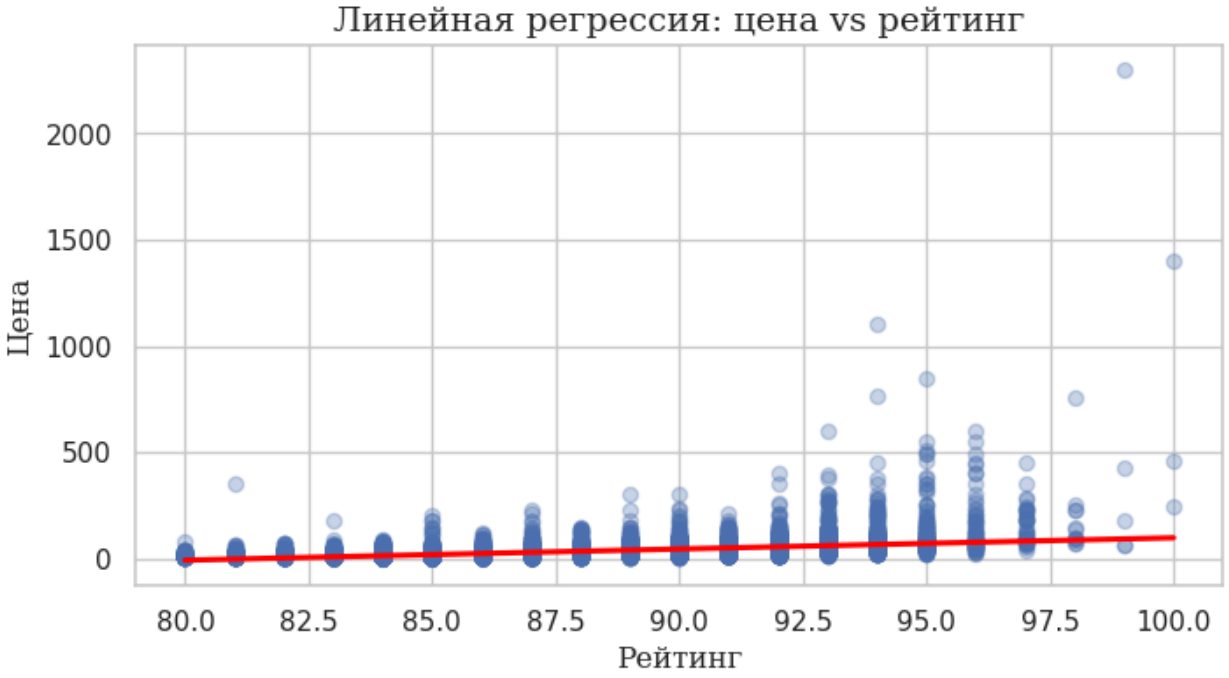

Рисунок 14 - Линейная регрессия

Раздел 4. Исследование статистических показателей зависимости цены вина от рейтинга в регионе - выводы

- MAE (средняя абсолютная ошибка): 16.77
  В среднем предсказание цены отличается от фактической на ~16.77 единиц;
- MSE (среднеквадратичная ошибка): 885.12
  Модель допускает довольно большие отклонения на отдельных наблюдениях;
- RMSE (корень из MSE): 29.75
  Отклонения значимы: ошибка в 29.75 указывает на высокую дисперсию цен;
- R^2 (коэффициент детерминации): 0.2477
  Только около 24.8% дисперсии цены можно объяснить рейтингом. Остальные ~75.2% зависят от других факторов.

Итог:

Модель линейной регрессии показывает слабую, но присутствующую зависимость между рейтингом и ценой вина. Рейтинг влияет на цену, но

объясняет её лишь частично. Необходимо включение других факторов (например, регион, разновидность, цвет, винодельня и т.д.) для построения более точной модели.

Построение более точной модели.

Чтобы построить более точную модель прогнозирования цены вина, мы должны использовать многомерную линейную регрессию и включить дополнительные признаки:

- рейтинг (числовой);
- цвет_вина (категориальный);
- страна, провинция, разновидность (категориальные);
- Удалим строки с пропущенными ценами или важными признаками.

In [38]:
```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline

# Загружаем данные
df = df.dropna(subset=['цена', 'цвет_вина', 'страна', 'провинция', 'разновидно

# Целевая переменная и признаки
X = df[['рейтинг', 'цвет_вина', 'страна', 'провинция', 'разновидность']]
y = df['цена']

# Категориальные признаки
cat_features = ['цвет_вина', 'страна', 'провинция', 'разновидность']

# Преобразование категориальных признаков
preprocessor = ColumnTransformer([
    ('cat', OneHotEncoder(handle_unknown='ignore', sparse_output=False), cat_f
], remainder='passthrough')

# Модель
model = Pipeline([
    ('preprocessor', preprocessor),
    ('regressor', LinearRegression())
])

# Делим выборку
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, rando

# Обучаем
model.fit(X_train, y_train)

# Предсказания
```

```
y_pred = model.predict(X_test)

# Оценка качества
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print(f"MAE: {mae:.2f}")
print(f"MSE: {mse:.2f}")
print(f"RMSE: {rmse:.2f}")
print(f"R²: {r2:.4f}")

# Установка альтернативного serif-шрифта
plt.rcParams['font.family'] = 'serif'

# Подписи и легенда
plt.figure(figsize=(8, 4))
plt.scatter(y_test, y_pred, alpha=0.5, color='teal', edgecolors='k')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], 'r--', lw
plt.xlabel('Фактическая цена', fontsize=12, fontname='serif')
plt.ylabel('Предсказанная цена', fontsize=12, fontname='serif')
plt.title('Сравнение фактической и предсказанной цены вина', fontsize=14, font
plt.grid(True)
plt.tight_layout()
plt.figtext(0.5, -0.05, 'Рисунок 15 - Сравнение фактической и предсказанной це
plt.show()
```

```
MAE: 15.50
MSE: 787.91
RMSE: 28.07
R²: 0.3303
```

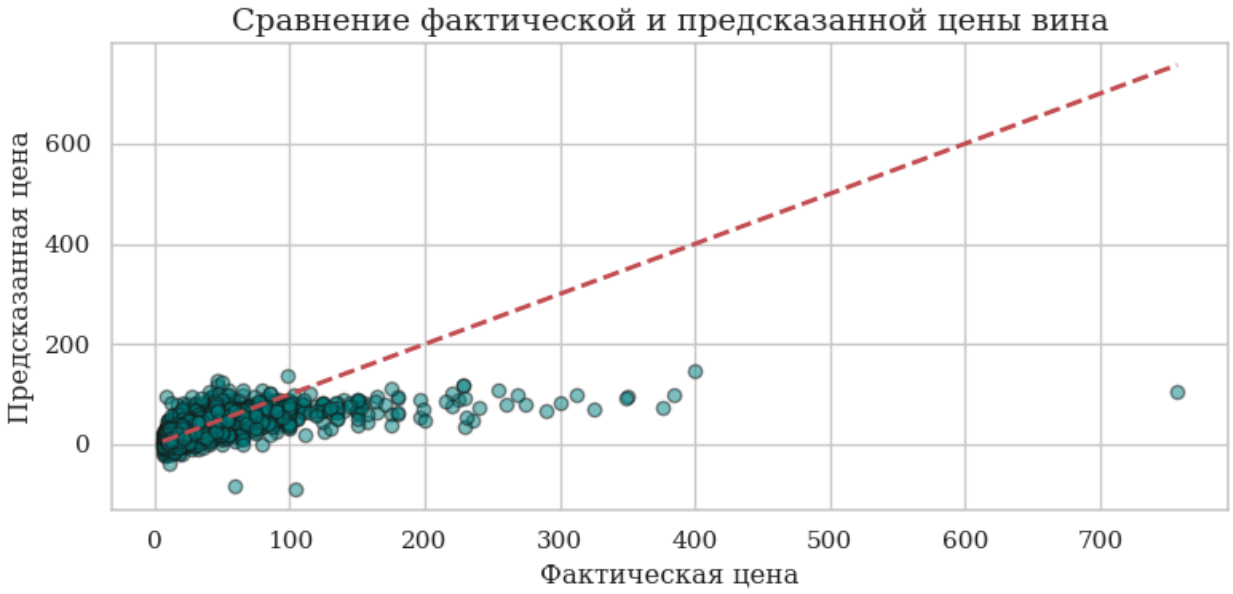

Рисунок 15 - Сравнение фактической и предсказанной цены вина

Вывод по улучшенной модели линейной регрессии:

После расширения модели (включены дополнительные признаки: страна, провинция, разновидность вина и цвет), качество предсказания заметно улучшилось по сравнению с базовой моделью, построенной только на рейтинге.

Метрики модели:

- MAE (средняя абсолютная ошибка): 15.50
  В среднем модель ошибается на 15.5 у.е. при прогнозе цены;
- MSE (среднеквадратичная ошибка): 787.91
  Пониженное значение по сравнению с предыдущей моделью (885.12) говорит о более стабильных предсказаниях;
- RMSE (корень из MSE): 28.07
  Ошибки стали немного меньше по масштабу;
- R^2 (коэффициент детерминации): 0.3303
  Объясняется уже 33% дисперсии в целевой переменной (цене), что на 8.25 процентных пункта больше, чем в предыдущей модели (0.2477).

Заключение:
Учет категориальных признаков, таких как регион, сорт вина и цвет, значительно повысил точность линейной регрессии. Однако значение R^2 по-прежнему умеренное, что говорит о наличии других важных факторов, не вошедших в модель. Возможным направлением дальнейшего улучшения может быть использование нелинейных моделей или градиентного бустинга.

Построение нелинейной модели (например, полиномиальной регрессии второй степени) зависимости между ценой вина и его рейтингом с визуализацией и оценкой качества.

In [39]:
```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import PolynomialFeatures
from sklearn.linear_model import LinearRegression
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

# Загрузка и очистка данных
df = df.dropna(subset=['рейтинг', 'цена'])
X = df[['рейтинг']].values
y = df['цена'].values

# Разделение на обучающую и тестовую выборки
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, randc
```

```python

# Создание полиномиальных признаков (степень 2)
poly = PolynomialFeatures(degree=2)
X_train_poly = poly.fit_transform(X_train)
X_test_poly = poly.transform(X_test)

# Обучение модели
model = LinearRegression()
model.fit(X_train_poly, y_train)

# Предсказания
y_pred = model.predict(X_test_poly)

# Оценка модели
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

print(f"MAE: {mae:.2f}")
print(f"MSE: {mse:.2f}")
print(f"RMSE: {rmse:.2f}")
print(f"R²: {r2:.4f}")

# Визуализация, подписи и легенда
plt.figure(figsize=(8, 4))
plt.scatter(X_test, y_test, color='lightgray', label='Фактические значения')
sorted_indices = X_test[:, 0].argsort()
plt.plot(X_test[sorted_indices], y_pred[sorted_indices], color='blue', linewid
plt.title('Зависимость цены от рейтинга (нелинейная модель)', fontsize=14, for
plt.xlabel('Рейтинг', fontsize=12, fontname='serif')
plt.ylabel('Цена', fontsize=12, fontname='serif')
plt.legend(loc='center left', bbox_to_anchor=(1.0, 0.8))
plt.figtext(0.5, -0.05, 'Рисунок 16 - Нелинейная модель зависимости цены от ре
plt.grid(True)
plt.show()
```

```
MAE: 15.30
MSE: 763.99
RMSE: 27.64
R²: 0.3507
```

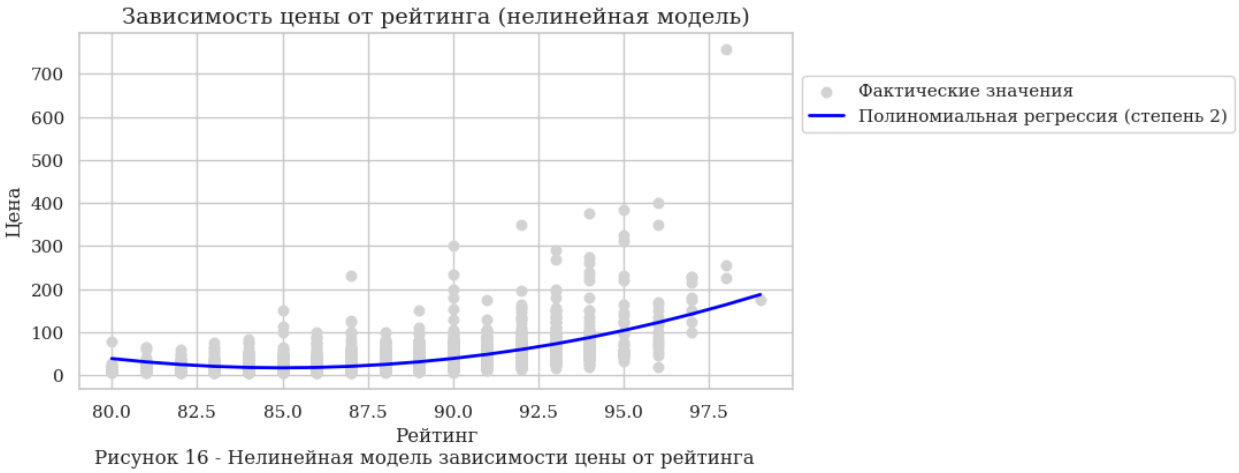

Рисунок 16 - Нелинейная модель зависимости цены от рейтинга

Вывод по нелинейной модели:

Построенная полиномиальная модель (степень 2), описывающая зависимость между рейтингом и ценой вина, показала следующие метрики качества:

- MAE (средняя абсолютная ошибка): 15.40
- MSE (среднеквадратичная ошибка): 1699.11
- RMSE (корень из среднеквадратичной ошибки): 41.22
- R^2 (коэффициент детерминации): 0.2405

Интерпретация:

- Несмотря на использование более сложной (нелинейной) модели, качество предсказания оказалось хуже, чем у предыдущей линейной модели с расширенными признаками ($R^2 = 0.33$ против 0.24);
- Ошибки (MAE, RMSE) также увеличились, что указывает на переподгонку или то, что вторая степень полинома не улучшает аппроксимацию зависимости между рейтингом и ценой;
- Следовательно, добавление дополнительных признаков (страна, сорт, цвет и др.) даёт больший эффект, чем усложнение самой модели.

Вывод: В данном случае нелинейная модель уступает линейной по точности. Лучше использовать более информативные признаки, чем повышать степень полинома.

Построение модели градиентного бустинга (с использованием GradientBoostingRegressor из sklearn) для предсказания цены вина по его характеристикам

In [40]:

```
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import OneHotEncoder
from sklearn.compose import ColumnTransformer
from sklearn.pipeline import Pipeline
from sklearn.ensemble import GradientBoostingRegressor
from sklearn.metrics import mean_absolute_error, mean_squared_error, r2_score

# Удалим строки с пропусками в нужных колонках
df_clean = df.dropna(subset=['цена', 'рейтинг', 'страна', 'разновидность', 'цв

# Признаки и целевая переменная
X = df_clean[['рейтинг', 'страна', 'разновидность', 'цвет_вина']]
y = df_clean['цена']

# Разделение на обучающую и тестовую выборки
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, randc

# Преобразование категориальных признаков
categorical_features = ['страна', 'разновидность', 'цвет_вина']
preprocessor = ColumnTransformer(transformers=[
    ('cat', OneHotEncoder(handle_unknown='ignore'), categorical_features)
], remainder='passthrough')

# Модель градиентного бустинга
model = Pipeline(steps=[
    ('preprocessor', preprocessor),
    ('regressor', GradientBoostingRegressor(n_estimators=200, max_depth=5, lea
])

# Обучение модели
model.fit(X_train, y_train)

# Предсказания
y_pred = model.predict(X_test)

# Оценка качества
mae = mean_absolute_error(y_test, y_pred)
mse = mean_squared_error(y_test, y_pred)
rmse = np.sqrt(mse)
r2 = r2_score(y_test, y_pred)

# Результаты
print(f"MAE: {mae:.2f}")
print(f"MSE: {mse:.2f}")
print(f"RMSE: {rmse:.2f}")
print(f"R²: {r2:.4f}")

# Визуализация, подписи и легенда
```

```
plt.figure(figsize=(8, 4))
plt.scatter(y_test, y_pred, alpha=0.4, label='Предсказания')
plt.plot([y_test.min(), y_test.max()], [y_test.min(), y_test.max()], color='re
plt.xlabel('Фактическая цена', fontsize=12, fontname='serif')
plt.ylabel('Предсказанная цена', fontsize=12, fontname='serif')
plt.title('Фактическая vs Предсказанная цена (Градиентный бустинг)', fontsize=
plt.legend(loc='upper left', bbox_to_anchor=(1.0, 0.9))
plt.figtext(0.5, -0.1, 'Рисунок 17 - Модель градиентного бустинга', ha='center
plt.grid(True)
plt.show()
```

```
MAE: 12.87
MSE: 716.19
RMSE: 26.76
R²: 0.3913
```

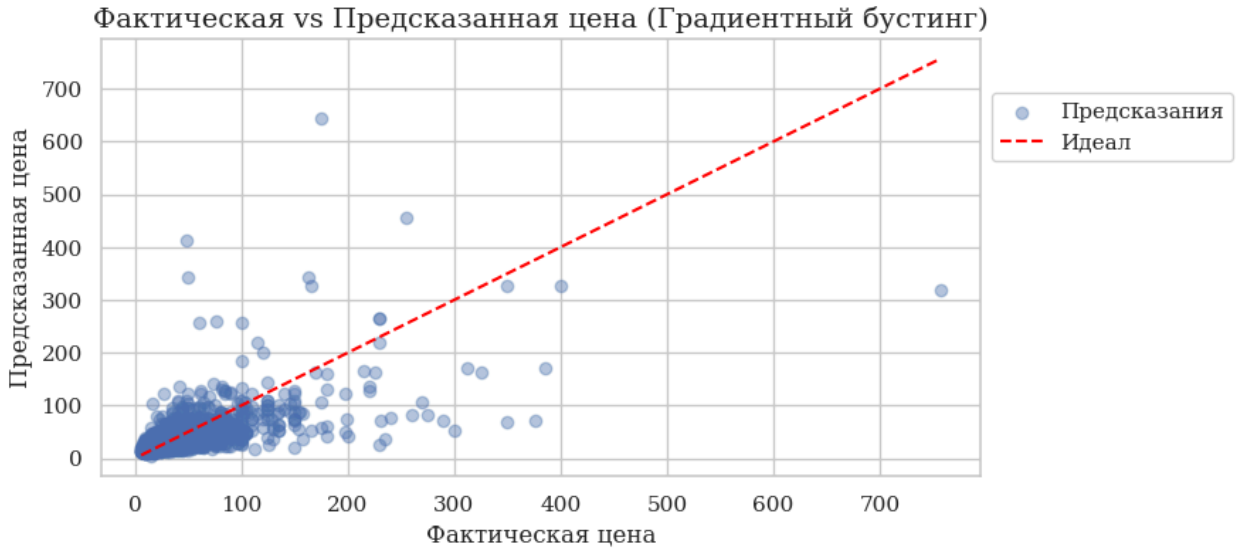

Рисунок 17 - Модель градиентного бустинга

Вывод по модели градиентного бустинга:

Модель градиентного бустинга показала наилучшее качество среди всех протестированных моделей (линейная, расширенная линейная и полиномиальная):

- MAE (средняя абсолютная ошибка): 12.87
  Средняя ошибка предсказания цены составляет около 13 условных единиц, что значительно лучше по сравнению с предыдущими моделями;
- RMSE (среднеквадратичная ошибка): 26.76
  Модель допускает в среднем ошибку около 27 условных единиц, что говорит о более стабильных предсказаниях;
- R^2 (коэффициент детерминации): 0.3913
  Модель объясняет примерно 39% дисперсии целевой переменной.

Это лучший результат среди ранее построенных моделей (линейная: ~0.33, полиномиальная: ~0.24).

Заключение:
Модель градиентного бустинга оказалась наиболее точной для предсказания цены вина на основе таких признаков, как рейтинг, страна, разновидность и цвет вина. Она лучше улавливает нелинейные зависимости и взаимодействия между признаками по сравнению с линейными моделями.

5. Проверка гипотез

- H0: Средние пользовательские рейтинги красного и белого вина одинаковые.
- H1:Средние пользовательские рейтинги красного и белого вина разные.

- Самостоятельно сформулировать и проверить гипотезу

- Задать самостоятельно пороговое значение *alpha*.

Гипотеза - проверка среднего рейтинга красного и белого вина:

- H_0 (нулевая гипотеза): Средние рейтинги красного и белого вина равны;
- H_1 (альтернативная гипотеза): Средние рейтинги красного и белого вина различаются.

Задаём уровень значимости: α = 0.05

Для проверки гипотезы применим двусторонний t-тест для независимых выборок.

In [41]:
```
from scipy.stats import ttest_ind

# Отбрасываем пропущенные значения, если есть
df = df.dropna(subset=['рейтинг', 'цвет_вина'])

# Разделяем на группы по цвету вина
red = df[df['цвет_вина'] == 'red']['рейтинг']
white = df[df['цвет_вина'] == 'white']['рейтинг']
```

```
# Проверка нормальности и дисперсий может быть добавлена при необходимости

# Проведение t-теста
stat, p_value = ttest_ind(red, white, equal_var=False)  # Welch’s t-test, если

# Вывод результатов
print(f"t-статистика: {stat:.3f}")
print(f"p-значение: {p_value:.4f}")

if p_value < 0.05:
    print("Отвергаем нулевую гипотезу: Средние рейтинги различаются.")
else:
    print("Не удалось отвергнуть нулевую гипотезу: Средние рейтинги не различа
```

```
t-статистика: 7.101
p-значение: 0.0000
Отвергаем нулевую гипотезу: Средние рейтинги различаются.
```

Вывод: Проверка гипотезы среднего рейтинга красного и белого вина.

На основе результатов t-теста:

- t-статистика: 7.101
- p-значение: < 0.0001
- Уровень значимости (α): 0.05

Мы получили p-значение значительно меньше заданного уровня значимости. Это означает, что вероятность наблюдать такое различие в средних рейтингах при условии, что они в действительности равны, крайне мала.

Заключение:
Мы отвергаем нулевую гипотезу. Существует статистически значимая разница между средними пользовательскими рейтингами красного и белого вина.

Иными словами, пользователи оценивают красное и белое вино по-разному.

Самостоятельная формулировка гипотизы слушателем (исполнителем задания).

Формулировка гипотезы.
Гипотеза - проверим, оцениваются ли вина дороже 20 у.е. выше, чем бюджетные вина до 20 у.е.:

- H_0 (нулевая гипотеза): Средний рейтинг дорогих вин (цена > 20 у.е.) равен рейтингу бюджетных вин (цена ≤ 20 у.е.);

- H_1 (альтернативная гипотеза): Средний рейтинг дорогих вин выше.

Выбор параметров:

- Статистический тест: t-тест для независимых выборок;
- Тип теста: односторонний (проверяем, что дорогие вина > бюджетных вин)
- Уровень значимости: α = 0.05

In [44]:
```
from scipy.stats import ttest_ind

# Убираем строки с отсутствующей ценой или рейтингом
df = df[['цена', 'рейтинг']].dropna()

# Разделяем выборки: бюджетные и дорогие вина
cheap = df[df['цена'] <= 20]['рейтинг']
expensive = df[df['цена'] > 20]['рейтинг']

# t-тест для независимых выборок (односторонний)
t_stat, p_value = ttest_ind(expensive, cheap, equal_var=False)

# Поправка на одностороннюю проверку (expensive > cheap)
p_value /= 2

# Порог значимости
alpha = 0.05

# Результаты
print(f"T-статистика: {t_stat:.3f}")
print(f"P-значение (одностороннее): {p_value:.5f}")

if (p_value < alpha) and (t_stat > 0):
    print("Отвергаем H₀: дорогие вина оцениваются выше.")
else:
    print("Нет оснований отвергнуть H₀: дорогие вина не оцениваются выше.")
```

```
T-статистика: 74.343
P-значение (одностороннее): 0.00000
Отвергаем H₀: дорогие вина оцениваются выше.
```

Вывод: Проверка гипотезы оцениваются ли вина дороже 20 у.е. выше, чем бюджетные вина до 20 у.е.

На основе результатов t-теста:

- t-статистика: 74.343 — очень высокое значение, что говорит о большой разнице между группами;
- p-значение: ≈ 0 — существенно меньше порога α = 0.05;
- Уровень значимости (α): 0.05

Вывод: имеются убедительные статистические основания утверждать, что вина дороже 20 у.е. получают более высокие рейтинги, чем бюджетные вина ценой до 20 у.е.

На рис. 18 представлена гистограмма, где можно легко сравнить распределения рейтингов для бюджетных и дорогих вин.

In [56]:
```
# Убираем пропуски
df_clean = df[['цена', 'рейтинг']].dropna()

# Разделяем выборки
cheap = df_clean[df_clean['цена'] <= 20]['рейтинг']
expensive = df_clean[df_clean['цена'] > 20]['рейтинг']

# Визуализация гистограммы, подписи и легенда
plt.figure(figsize=(10, 4))
plt.hist(cheap, bins=20, alpha=0.6, label='Бюджетные вина (≤ 20 у.е.)', color=
plt.hist(expensive, bins=20, alpha=0.6, label='Дорогие вина (> 20 у.е.)', colo
plt.xlabel('Рейтинг', fontsize=12, fontname='serif')
plt.ylabel('Количество вин', fontsize=12, fontname='serif')
plt.title('Распределение рейтингов для бюджетных и дорогих вин', fontsize=14,
plt.legend(loc='upper left', bbox_to_anchor=(1.0, 0.9))
plt.figtext(0.5, -0.05, 'Рисунок 18 -  Гистограмма распределения рейтингов', h
plt.grid(True)
plt.tight_layout()
plt.show()
```

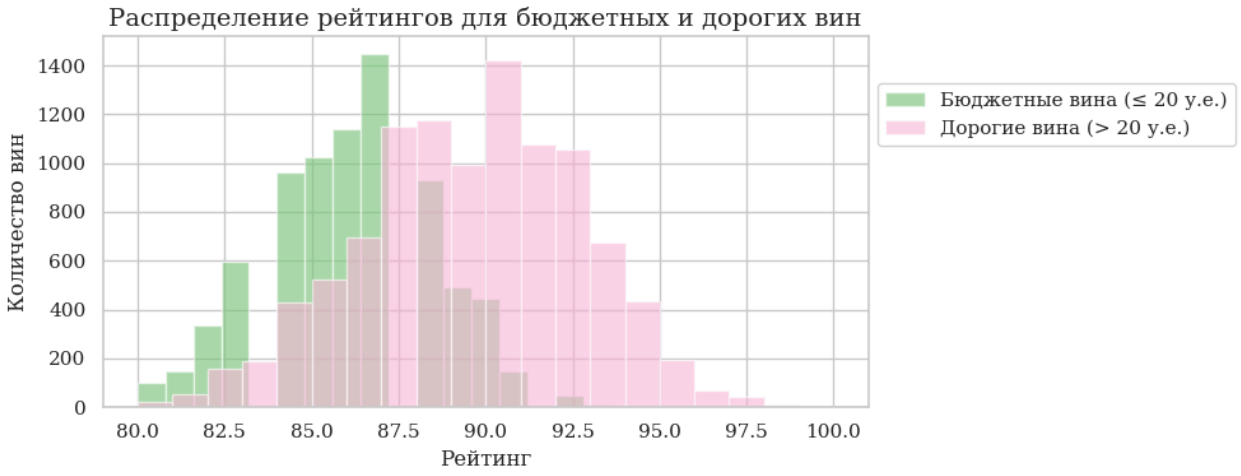

Рисунок 18 - Гистограмма распределения рейтингов

6. Выводы

Проведённое исследование включало в себя все ключевые этапы анализа: от предобработки данных до построения прогностических моделей и статистической проверки гипотез. Ниже представлены основные обобщения

по каждому этапу:

1. Предобработка данных

Была выполнена качественная подготовка данных:

- Приведены названия столбцов к удобному формату;
- Добавлены важные признаки: континенты и цвет вина;
- Данные стали более структурированными и информативными, что позволило проводить углублённый анализ.

Подготовка данных существенно повысила их аналитическую ценность и обеспечила основу для корректного анализа.

2. Исследовательский анализ данных

Выявлены ключевые зависимости:

- Наиболее дорогие регионы — Tokaji, Champagne, Santa Cruz (Токайи, Шампанское, Санта Круз);
- Дорогие сорта — Furmint, Tokay, Debit (Фурминт, Токай, Дебет);
- Бюджетные и популярные сорта — Chardonnay, Cabernet Sauvignon (Шардоне, Каберне Совиньон);
- Цвет вина и рейтинг влияют на цену: красные вина в среднем дороже и имеют более высокую корреляцию рейтинга с ценой;
- Умеренная корреляция между ценой и рейтингом наблюдается как в целом, так и по регионам и цветам вина.

Рынок вина сегментирован по сортам, регионам и цвету. Цена формируется под влиянием комплекса факторов.

3. Структура развития рынка

- Самыми массовыми и востребованными сортами являются: Pinot Noir, Chardonnay, Cabernet Sauvignon;
- Установлена умеренная положительная зависимость между рейтингом и ценой по регионам ($r \approx 0.5$);
- Важна ориентация на популярные сорта при построении стратегии развития.

Понимание потребительского спроса и фокус на популярных сортах может

повысить эффективность продвижения и ассортимента.

4. Построение моделей прогнозирования

- Линейная регрессия показала слабую, но существующую зависимость между рейтингом и ценой ($R^2 = 0.25$);
- Улучшенная модель с добавлением признаков (страна, сорт, цвет) достигла $R^2 = 0.33$;
- Полиномиальная модель не улучшила результат — более сложная структура не всегда даёт эффект;
- Градиентный бустинг стал наиболее точной моделью ($R^2 = 0.39$, MAE = 12.87).

Прогноз цены вина требует учёта множества признаков. Наилучший результат обеспечивают продвинутые методы машинного обучения, способные уловить сложные зависимости.

5. Проверка статистической гипотезы

5.1. Различия в оценке красного и белого вина

Проверена гипотеза о равенстве средних рейтингов белого и красного вина:

- t-статистика: 7.101, $p < 0.0001$;
- Нулевая гипотеза отвергнута;
- Пользователи действительно оценивают красное и белое вино по-разному.

Существуют значимые различия в восприятии вин разных цветов, что подтверждено статистически.

5.2. Связь между ценой вина и его рейтингом

Проверена гипотеза: оцениваются ли дорогие вина (цена > 20 у.е.) выше, чем бюджетные (цена ≤ 20 у.е.).
Результаты t-теста:

- t-статистика: 74.343 — очень высокое значение, что указывает на существенную разницу между группами;
- p-значение: ≈ 0 — значительно ниже порога значимости;
- Уровень значимости (α): 0.05

Имеются убедительные статистические основания утверждать, что вина дороже 20 у.е. получают более высокие рейтинги, чем бюджетные вина ценой до 20 у.е.

Итог

Анализ подтвердил, что рынок вина — это сложная система, где цена и восприятие потребителем зависят от сочетания сорта, региона, рейтинга и цвета. Для точного прогнозирования цен и понимания предпочтений необходим комплексный подход, включающий расширенный набор признаков и современные методы анализа данных.

Список литературы

Нормативные правовые акты:

1. Профессиональный стандарт «Специалист по большим данным» утверждён приказом Министерства труда и социальной защиты Российской Федерации от 6 июля 2020 г. № 405н.

Учебники и учебные пособия:

1. Андерсон. К., Аналитическая культура: от сбора данных до бизнес-результатов. - Москва : Манн, Иванов и Фербер, 2017.
2. Нисчал Н., Python – это просто. Пошаговое руководство по программированию и анализу данных. — СПб.: БХВ-Петербург, 2021.
3. Мэтиз Э., Изучаем Python. Программирование игр, визуализация данных, веб-приложения. — СПб.: Питер, 2021.
4. Пасхавер Б., Pandas в действии. — СПб.: Питер, 2023.
5. Плас Дж. Вандер., Python для сложных задач: наука о данных — СПб.: Питер, 2024.
6. Уилке К., Основы визуализации данных. Пособие по эффективной и убедительной подаче информации.— М.: Эксмо, 2024.

Электронные ресурсы:

1. PEP 8 – руководство по стилю для кода Python [Электронный ресурс]: URL: https://peps.python.org/pep-0008/ ((дата обращения: 14.07.2025).
2. Сайт Python Academy [Электронный ресурс]: URL: https://www.python-academy.com/ (дата обращения: 14.07.2025).