

Добро пожаловать в Colab!

(Новое) Попробуйте Gemini API

- [Generate a Gemini API key](#)
- [Talk to Gemini with the Speech-to-Text API](#)
- [Gemini API: Quickstart with Python](#)
- [Gemini API code sample](#)
- [Compare Gemini with ChatGPT](#)
- [More notebooks](#)

```
import pandas as pd
import matplotlib.pyplot as plt
import seaborn as sns
```

```
data = pd.read_csv("Coding/Project-статистика - Project - Лист1.csv")
```

```
print("Первые 5 строк данных:")
print(data.head())
```

↪ Первые 5 строк данных:

	Отметка времени	Пол	Возраст	Номер телефона для связи \
0	17.11.2024 15:13:48	Мужской	19	Мегаком
1	17.11.2024 15:43:28	Мужской	20	Beeline
2	17.11.2024 15:54:36	Женский	21	Ошка
3	17.11.2024 16:00:00	Женский	20	Ошка
4	17.11.2024 16:07:10	Женский	20	Мегаком

В какие дни недели обычно свободен? Это потребуется чтобы организовать ив

0	Суббота
1	понедельник
2	Понедельник, Пятница, Воскресенье
3	Вторник, Среда, Четверг, Пятница
4	Понедельник, Суббота, Воскресенье

	Работаешь?	Количество часов	GPA	Будешь оканчивать универ? \
0	Стажировка	20	3.12	ДА
1	Стажировка	16	2.79	ДА
2	НЕТ	0	3.90	ДА
3	НЕТ	0	2.67	ДА
4	Стажировка	12	2.45	ДА

Нужна ли помощь по некоторым предметам? Если да то пишите. \

0	НЕТ
1	НЕТ
2	НЕТ
3	НЕТ
4	НЕТ

Пишите свои идеи насчет ивенты после финалов этого семестра(коньки, наци

0	коньки
1	без разницы
2	Национальный парк
3	коньки
4	коньки

```
print("\nОсновные статистические показатели:")
print(data.describe())
```

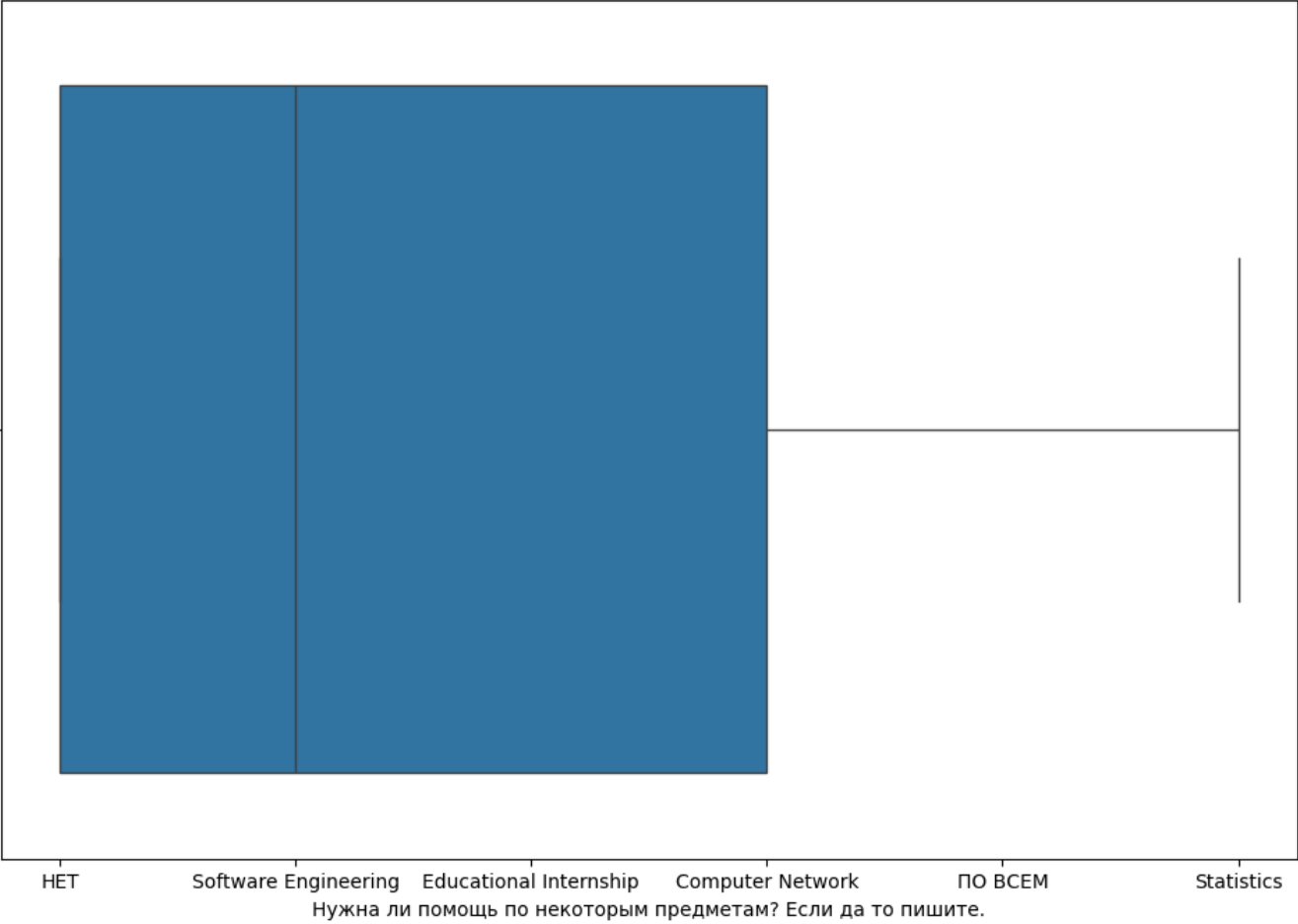
```
# Построим коробчатую диаграмму (Box Plot)
plt.figure(figsize=(12, 8))
sns.boxplot(x=data['Нужна ли помощь по некоторым предметам? Если да то пишите.'])
plt.title('Коробчатая диаграмма')
plt.show()
```



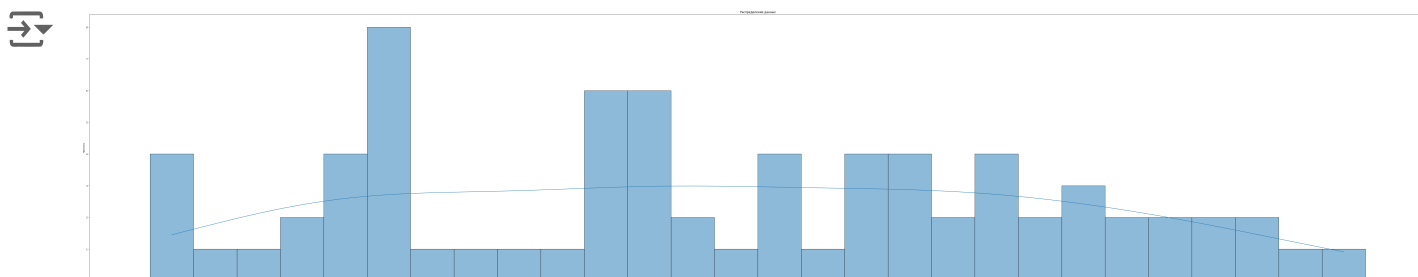
Основные статистические показатели:

	Возраст	Количество часов	GPA
count	73.000000	73.000000	73.000000
mean	20.041096	9.438356	3.148082
std	0.675737	9.613339	0.469582
min	18.000000	0.000000	2.450000
25%	20.000000	0.000000	2.750000
50%	20.000000	5.000000	2.990000
75%	20.000000	20.000000	3.650000
max	22.000000	30.000000	3.960000

Коробчатая диаграмма



Напишите программный код или сгенерируйте его с помощью искусственного интеллекта.



```
import pandas as pd
import seaborn as sns
import matplotlib.pyplot as plt

# Фильтрация только работающих студентов (часы работы больше 0)
working_students = data[data['Количество часов'] > 0]

# Укажите столбец, где указаны дни недели, когда студенты свободны
column_name = 'В какие дни недели обычно свободен? Это потребуется чтобы органи

# Построим гистограмму только для работающих студентов
plt.figure(figsize=(10, 6))
sns.histplot(working_students[column_name], kde=True, color='skyblue')

# Настроим заголовки и подписи осей
plt.title(f'Распределение свободного времени работающих студентов по дням недел
plt.xlabel(column_name, fontsize=12)
plt.ylabel('Частота', fontsize=12)
plt.xticks(rotation=45) # Повернем подписи оси X, если они длинные
plt.show()
```





Страница 6 из 20

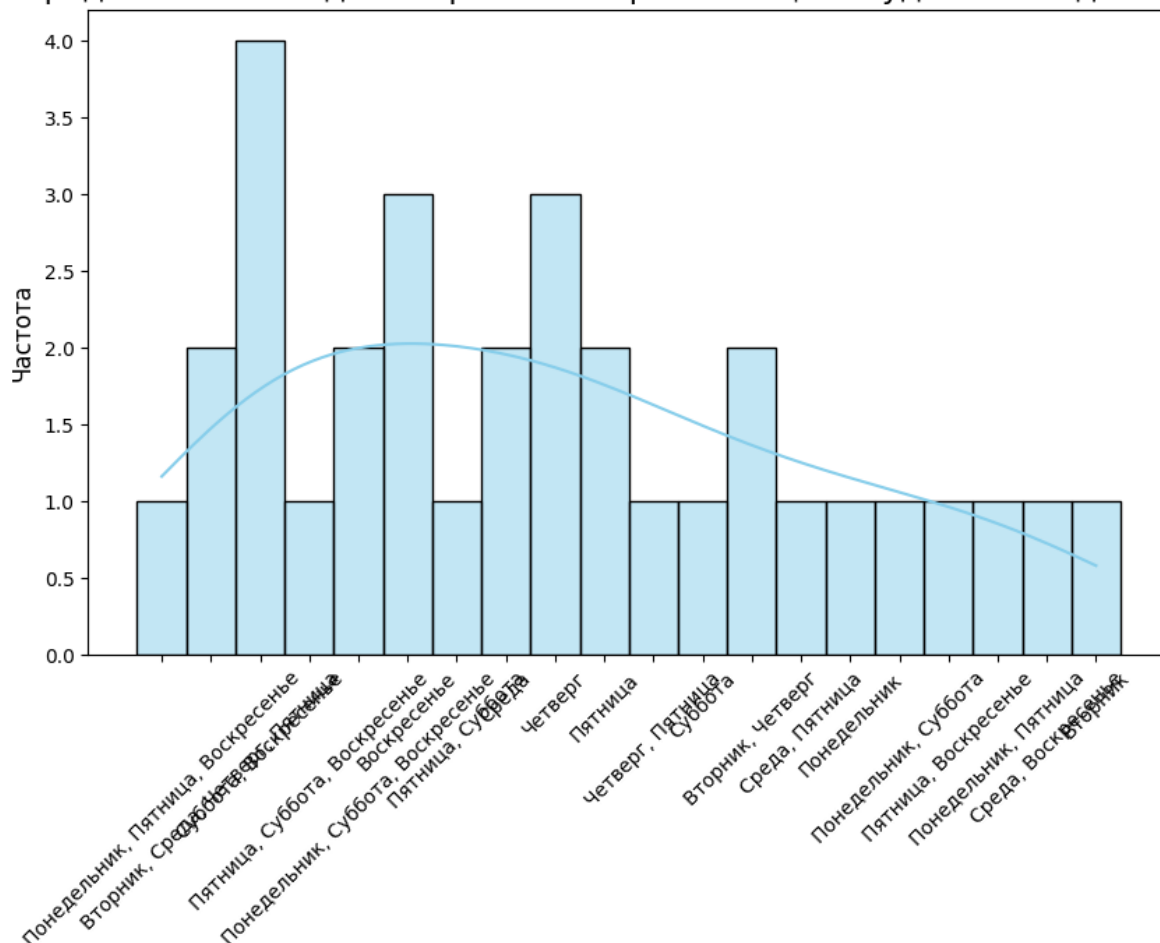
```
# Фильтрация только работающих студентов (часы работы больше 0)
working_students = data[data['Количество часов'] > 0]

# Укажите столбец, где указаны дни недели, когда студенты свободны
column_name = 'В какие дни недели обычно свободен? Это потребуется чтобы органи

# Построим гистограмму только для работающих студентов
plt.figure(figsize=(10, 6))
sns.histplot(working_students[column_name], kde=True, color='skyblue')

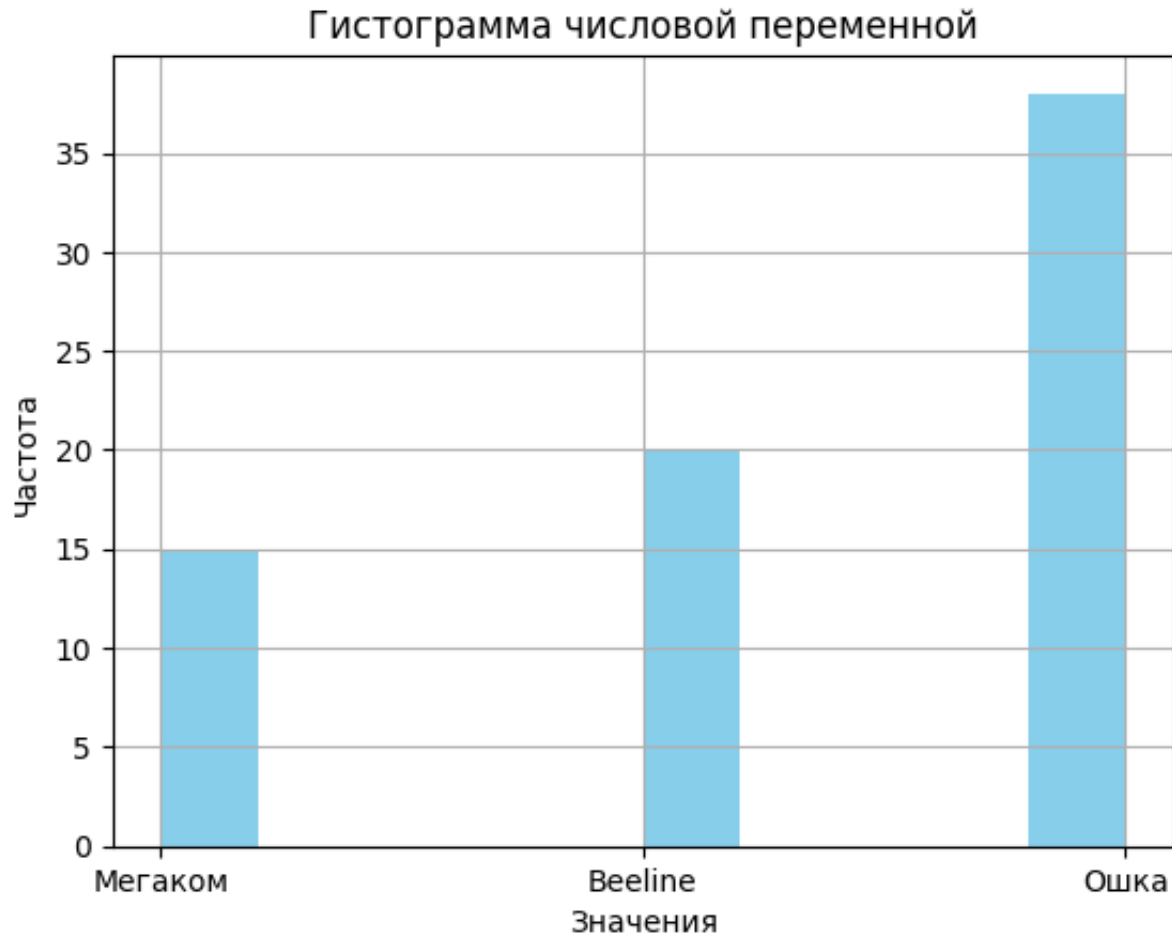
# Настроим заголовки и подписи осей
plt.title('Распределение свободного времени не работающих студентов по дням не
plt.xlabel(column_name, fontsize=12)
plt.ylabel('Частота', fontsize=12)
plt.xticks(rotation=45) # Повернем подписи оси X, если они длинные
plt.show()
```

Распределение свободного времени не работающих студентов по дням недели

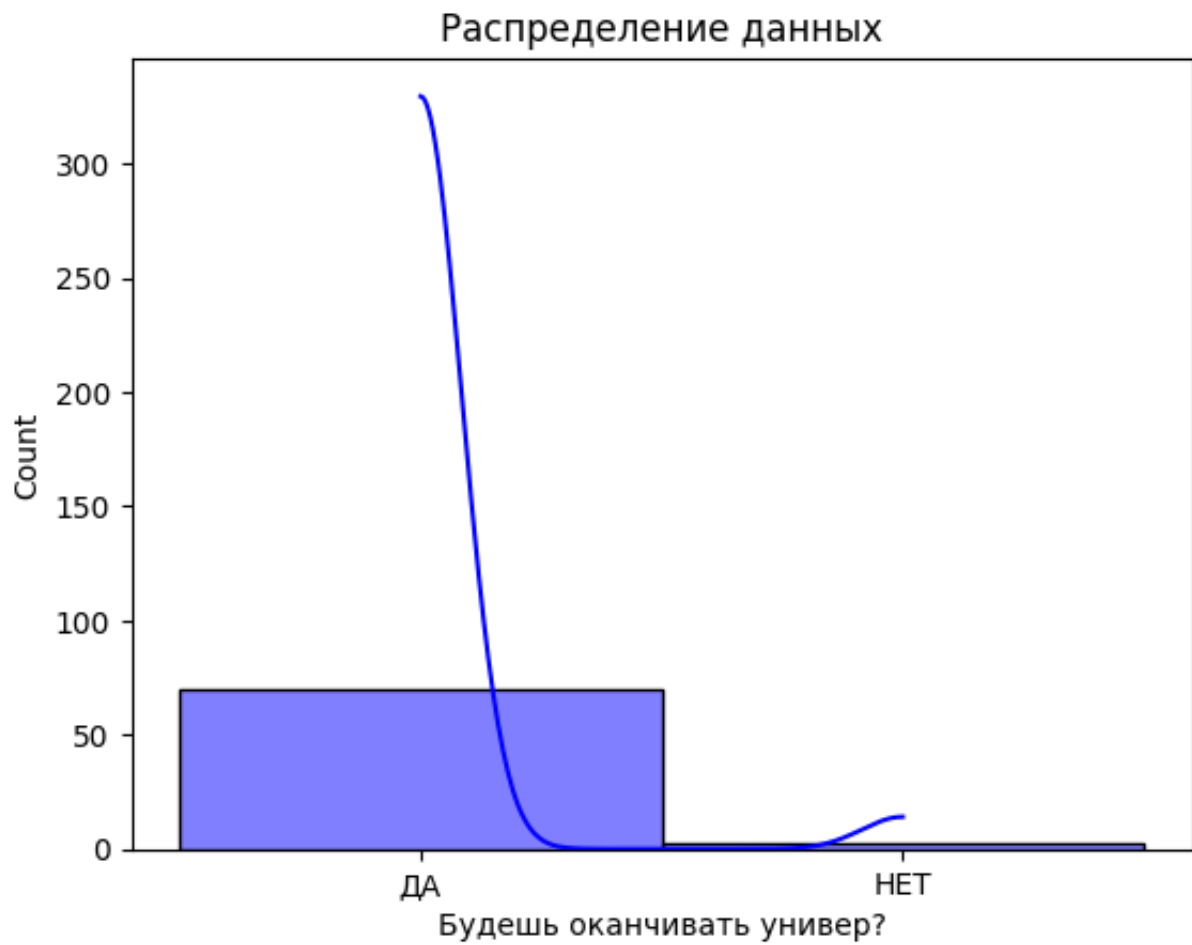


В какие дни недели обычно свободен? Это потребуется чтобы организовать ивенты.

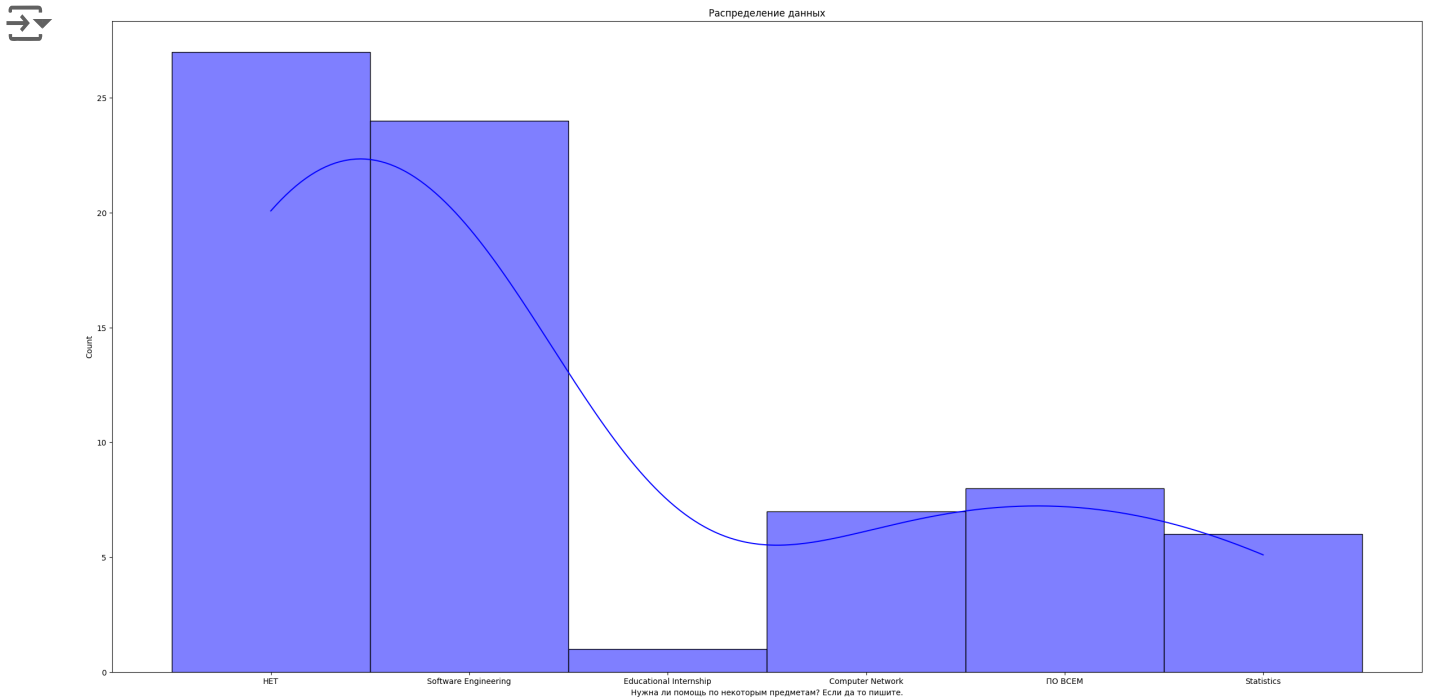

```
data['Номер телефона для связи'].hist(bins=10, color='skyblue')  
plt.title("Гистограмма числовой переменной")  
plt.xlabel("Значения")  
plt.ylabel("Частота")  
plt.show()
```



```
sns.histplot(data['Будешь оканчивать универ?'], kde=True, color='blue')  
plt.title("Распределение данных")  
plt.show()
```



```
plt.figure(figsize=(30, 15))
sns.histplot(data['Нужна ли помощь по некоторым предметам? Если да то пишите.'])
plt.title("Распределение данных")
plt.show()
```



```
from sklearn.linear_model import LinearRegression

# Выбор данных для регрессии
X = data[['Возраст']]
y = data['GPA']

# Модель линейной регрессии
model = LinearRegression()
model.fit(X, y)

# Коэффициенты
print("\nКоэффициенты регрессии:")
print(f"Intercept: {model.intercept_}, Coefficient: {model.coef_}")
```



```
Коэффициенты регрессии:
Intercept: 1.5291791666666667, Coefficient: [0.08077917]
```

```
print("Пропущенные значения по столбцам:")
print(data.isnull().sum())
```



```
Пропущенные значения по столбцам:
Отметка времени
Пол
Возраст
Номер телефона для связи
В какие дни недели обычно свободен? Это потребуется чтобы организовать ивен
Работаешь?
Количество часов
GPA
Будешь оканчивать универ?
Нужна ли помощь по некоторым предметам? Если да то пишите.
Пишите свои идеи насчет ивенты после финалов этого семестра(коньки, национ
dtype: int64
```

```
# Группировка данных по какому-то признаку (например, по полу)
grouped_data = data.groupby('Пол')['GPA'].mean()
print(grouped_data)
```

```
# Сравнение средних значений между группами
from scipy.stats import ttest_ind
```

```
group_1 = data[data['Пол'] == 'Мужской']['GPA']
group_2 = data[data['Пол'] == 'Женский']['GPA']
```

```
t_stat, p_value = ttest_ind(group_1, group_2)
print(f"T-statistic: {t_stat}, P-value: {p_value}")
```

```
# Группировка данных по какому-то признаку (например, по полу)
grouped_data = data.groupby('Пол')['Количество часов'].mean()
print(grouped_data)
```

```
# Сравнение средних значений между группами
from scipy.stats import ttest_ind
```

```
group_1 = data[data['Пол'] == 'Мужской']['Количество часов']
group_2 = data[data['Пол'] == 'Женский']['Количество часов']
```

```
t_stat, p_value = ttest_ind(group_1, group_2)
print(f"T-statistic: {t_stat}, P-value: {p_value}")
```



```
Пол
Женский    3.100833
Мужской    3.194054
Name: GPA, dtype: float64
T-statistic: 0.8463190725172828, P-value: 0.4002182485021101
Пол
Женский    10.416667
Мужской     8.486486
Name: Количество часов, dtype: float64
T-statistic: -0.8560656440194393, P-value: 0.39484142672536027
```

```
from sklearn.linear_model import LinearRegression
from sklearn.model_selection import train_test_split
from sklearn.metrics import mean_squared_error

# Выбираем независимые переменные (например, возраст, часы работы, пол) и завис
X = data[['Возраст', 'Количество часов', 'Пол']]
y = data['GPA']

# Преобразуем категориальные данные (например, пол) в числовые значения
X = pd.get_dummies(X, drop_first=True)

# Разделим данные на тренировочную и тестовую выборки
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random

# Создадим модель линейной регрессии
model = LinearRegression()
model.fit(X_train, y_train)

# Сделаем предсказания
y_pred = model.predict(X_test)
print(y_pred)

# Оценка модели
mse = mean_squared_error(y_test, y_pred)
print(f"Mean Squared Error: {mse}")
```

↔ [3.11038444 2.92811084 3.02730365 2.81099574 2.82440746 3.40336247
3.04151559 3.40336247 3.40336247 2.80749522 3.02730365 3.40336247
3.28995773 3.14791928 2.89947714]
Mean Squared Error: 0.15076880303932053

```
from sklearn.cluster import KMeans
```

```
# Применяем кластеризацию
```


```
X = data[['Возраст', 'Количество часов', 'GPA']]
```

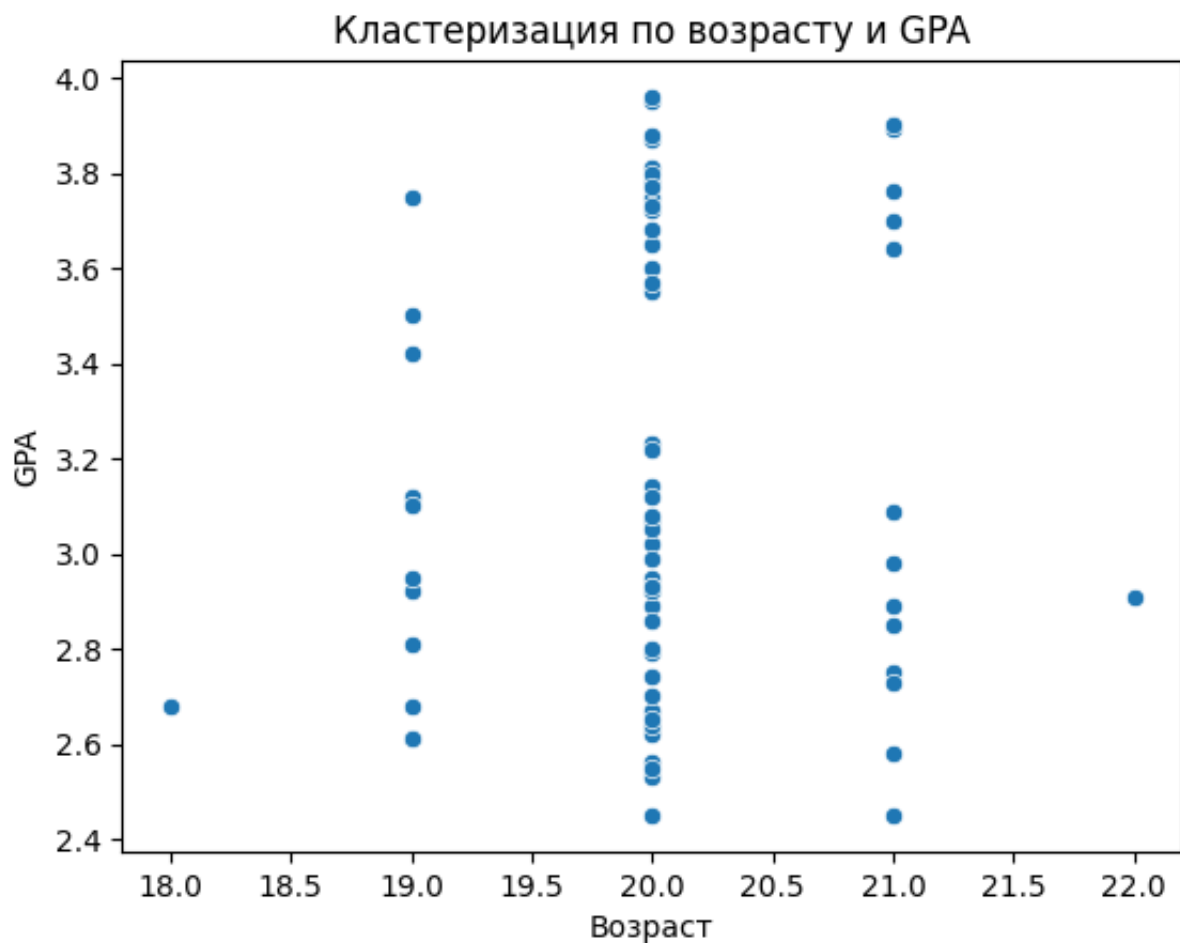
```
# Визуализация кластеров
```

```
sns.scatterplot(data=data, x='Возраст', y='GPA', palette='Set1')
```

```
plt.title('Кластеризация по возрасту и GPA')
```

```
plt.show()
```

 <ipython-input-80-5df64344252c>:8: UserWarning: Ignoring `palette` because
sns.scatterplot(data=data, x='Возраст', y='GPA', palette='Set1')



```
from sklearn.ensemble import RandomForestRegressor

# Создание модели случайного леса для предсказания GPA
X = data[['Возраст', 'Количество часов', 'Пол']]
y = data['GPA']
X = pd.get_dummies(X, drop_first=True)

# Разделение данных на тренировочную и тестовую выборки
X_train, X_test, y_train, y_test = train_test_split(X, y, test_size=0.2, random

# Модель случайного леса
rf_model = RandomForestRegressor(n_estimators=100, random_state=42)
rf_model.fit(X_train, y_train)

# Оценка модели
y_pred = rf_model.predict(X_test)
mse_rf = mean_squared_error(y_test, y_pred)
print(f"Random Forest Mean Squared Error: {mse_rf}")
```

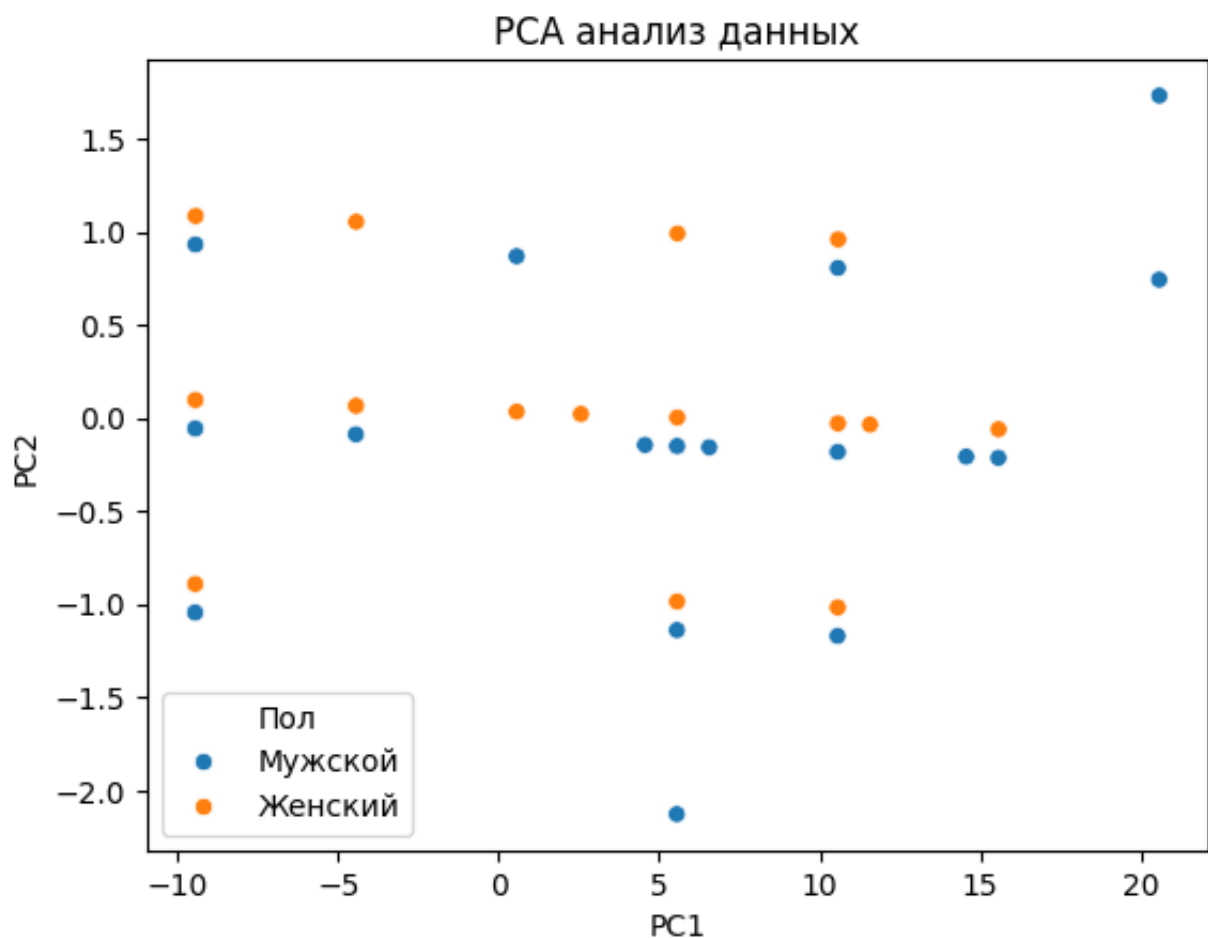
➡ Random Forest Mean Squared Error: 0.10680763911170708


```
from sklearn.decomposition import PCA

# Применяем PCA для уменьшения размерности
pca = PCA(n_components=2)
principal_components = pca.fit_transform(X)

# Создаем DataFrame с основными компонентами
pca_df = pd.DataFrame(data=principal_components, columns=['PC1', 'PC2'])

# Визуализация результатов PCA
sns.scatterplot(x='PC1', y='PC2', data=pca_df, hue=data['Пол'])
plt.title('PCA анализ данных')
plt.show()
```



Уже знакомы с Colab? В этом видео рассказывается о функциях, которые вы могли пропустить: интерактивных таблицах, истории выполненного кода и палитре команд.



Что такое Colab?

Colaboratory, или просто Colab, позволяет писать и выполнять код Python в браузере. При этом:

- не требуется никакой настройки;
- бесплатный доступ к графическим процессорам;
- предоставлять доступ к документам другим людям очень просто.

Это отличное решение для **студентов, специалистов по обработке данных и исследователей в области искусственного интеллекта**. Чтобы узнать больше, посмотрите [ознакомительное видео](#) или начните работу с инструментом ниже.

Анализ и обработка данных

Colab позволяет использовать для анализа и визуализации данных все возможности популярных библиотек Python. Например, в ячейке ниже используется библиотека **numpy** для генерации случайных данных, а также библиотека **matplotlib** для их визуализации. Чтобы изменить код, достаточно нажать на ячейку.

> Ресурсы по теме

Работа с блокнотами в Colab

- [Общие сведения о Colaboratory](#)
- [Руководство для Markdown](#)
- [Импорт библиотек и установка зависимостей](#)
- [Сохранение и загрузка блокнотов в GitHub](#)
- [Интерактивные формы](#)
- [Интерактивные виджеты](#)

Работа с данными

- [Загрузка данных: Диск, Таблицы и Google Cloud Storage](#)
- [Диаграмма: визуализация данных](#)
- [Начало работы с BigQuery](#)

Экспресс-курс по машинному обучению

Вот несколько блокнотов из онлайн-курса по машинному обучению от Google. Ещё больше информации доступно на [сайте курса](#).

- [Знакомство с Pandas DataFrame](#)
- [Линейная регрессия в tf.keras с использованием синтетических данных](#)

Использование ускорителей

- [TensorFlow с графическими процессорами](#)
- [TensorFlow с TPU](#)

↳ Скрыта 1 ячейка.

