**Лекция 13**

**Корреляция и регрессия**

*(Как находить связи между переменными и предсказывать будущее)*

**1. Корреляция: мера связи**

**Корреляция** показывает, насколько две переменные связаны линейно.

**1.1. Коэффициент корреляции Пирсона**

Формула:

r = ∑(xi−xˉ)(yi−yˉ) / ∑(xi−xˉ)^2⋅∑(yi−yˉ)^2​

* r∈[−1,1]:
  + r=1 — идеальная прямая связь.
  + r=−1 — идеальная обратная связь.
  + r=0 — связи нет.

**Пример на Python:**

import numpy as np

x = np.array([1, 2, 3, 4, 5])

y = np.array([2, 4, 5, 4, 5])

r = np.corrcoef(x, y)[0, 1]

print(f"Корреляция Пирсона: {r:.2f}") # ~0.79

**1.2. Другие виды корреляции**

1. **Спирмена (ранговая):**  
   Оценивает монотонную связь. Подходит для нелинейных данных.

from scipy.stats import spearmanr

rho, \_ = spearmanr(x, y)

print(f"Корреляция Спирмена: {rho:.2f}") # ~0.82

1. **Кендалла:**  
   Используется для порядковых данных.

from scipy.stats import kendalltau

tau, \_ = kendalltau(x, y)

print(f"Корреляция Кендалла: {tau:.2f}") # ~0.73

**1.3. Визуализация корреляции**

**График рассеяния (scatter plot):**

import matplotlib.pyplot as plt

plt.scatter(x, y, color='blue')

plt.title("Зависимость y от x")

plt.xlabel("x")

plt.ylabel("y")

plt.grid(True)

plt.show()

**2. Регрессия: предсказание значений**

**Регрессия** позволяет моделировать зависимость одной переменной от другой.

**2.1. Линейная регрессия**

Уравнение:

y=b0+b1⋅x+ϵ

* b0​ — интерсепт (сдвиг).
* b1​ — коэффициент наклона.
* ϵ — ошибка.

**Метод наименьших квадратов (МНК):**

b1=∑(xi−xˉ)(yi−yˉ) / ∑(xi−xˉ)^2, b0=yˉ−b1⋅xˉ

**Пример на Python:**

from sklearn.linear\_model import LinearRegression

X = x.reshape(-1, 1) # Преобразуем в 2D-массив

model = LinearRegression().fit(X, y)

print(f"Уравнение: y = {model.coef\_[0]:.2f}x + {model.intercept\_:.2f}")

# Пример вывода: y = 0.70x + 1.90

# Предсказание

y\_pred = model.predict(X)

**2.2. Оценка качества модели**

1. **Коэффициент детерминации (R²):**

R2 = 1−(∑(yi−y^i)^2 / ∑(yi−yˉ)^2​)

* + R2=1 — идеальное предсказание.
  + R2=0 — модель не лучше среднего.

1. **Среднеквадратичная ошибка (MSE):**

MSE=1/n \* ∑(yi−y^i)^2

**Расчёт на Python:**

from sklearn.metrics import r2\_score, mean\_squared\_error

r2 = r2\_score(y, y\_pred)

mse = mean\_squared\_error(y, y\_pred)

print(f"R²: {r2:.2f}, MSE: {mse:.2f}") # R²: ~0.62, MSE: ~0.46

**2.3. Визуализация регрессии**

plt.scatter(x, y, color='blue', label='Данные')

plt.plot(x, y\_pred, color='red', linewidth=2, label='Линия регрессии')

plt.title("Линейная регрессия")

plt.legend()

plt.show()

**3. Множественная регрессия**

Моделирует зависимость одной переменной от **нескольких** предикторов.

**Пример:**

# Данные: площадь дома (X1) и число комнат (X2) → цена (y)

X = np.array([[100, 2], [150, 3], [200, 4], [250, 4]])

y = np.array([300, 450, 600, 750])

model = LinearRegression().fit(X, y)

print(f"Уравнение: y = {model.coef\_[0]:.2f}x1 + {model.coef\_[1]:.2f}x2 + {model.intercept\_:.2f}")

**4. Распространённые ошибки**

1. **Путаница между корреляцией и причинностью:**
   * Корреляция не означает, что одна переменная вызывает другую!
2. **Игнорирование нелинейных связей:**
   * Если связь не линейна, используйте полиномиальную регрессию.
3. **Переобучение (overfitting):**
   * Не добавляйте слишком много предикторов без необходимости.

**5. Практическая задача**

**Условие:** Предсказать стоимость автомобиля на основе пробега.

**Данные:**

import pandas as pd

data = pd.DataFrame({

'Пробег (тыс. км)': [50, 100, 150, 200, 250],

'Цена (тыс. руб)': [800, 600, 500, 400, 300]

})

**Решение:**

python

Copy

Download

X = data[['Пробег (тыс. км)']]

y = data['Цена (тыс. руб)']

model = LinearRegression().fit(X, y)

print(f"Уравнение: Цена = {model.coef\_[0]:.2f} \* Пробег + {model.intercept\_:.2f}")

# Пример: Цена = -2.00 \* Пробег + 900.00

**6. Применение в Data Science**

* **Прогнозирование:** Цены, спрос, курс акций.
* **Анализ влияния:** Как реклама влияет на продажи.
* **A/B-тесты:** Оценка эффективности изменений.