

# Шакиров Тимур Маратович

ИУ5-65Б

19 вариант

```
In [27]: import pandas as pd
import numpy as np
from sklearn.model_selection import train_test_split
from sklearn.preprocessing import LabelEncoder, StandardScaler
from sklearn.svm import SVC
from sklearn.ensemble import GradientBoostingClassifier
from sklearn.metrics import accuracy_score, f1_score, classification_report

In [7]: df = pd.read_csv('investments_VC.csv', encoding='latin1', sep=None, engine='python', on_bad_lines='skip')

In [11]: df = df[['status', ' funding_total_usd ', 'country_code', 'funding_rounds', 'founded_year']]
df = df.dropna(subset=['status'])

In [9]: print(df.columns.tolist())

['permalink', 'name', 'homepage_url', 'category_list', ' market ', ' funding_total_usd ', 'status', 'country_co
de', 'state_code', 'region', 'city', 'funding_rounds', 'founded_at', 'founded_month', 'founded_quarter', 'found
ed_year', 'first_funding_at', 'last_funding_at', 'seed', 'venture', 'equity_crowdfunding', 'undisclosed', 'conv
ertible_note', 'debt_financing', 'angel', 'grant', 'private_equity', 'post_ipo_equity', 'post_ipo_debt', 'secon
dary_market', 'product_crowdfunding', 'round_A', 'round_B', 'round_C', 'round_D', 'round_E', 'round_F', 'round_
G', 'round_H']

In [12]: df['status'] = df['status'].apply(lambda x: 'acquired' if x == 'acquired' else 'other')

In [15]: df[' funding_total_usd '] = df[' funding_total_usd '].replace(['$'], '', regex=True).replace('None', np.nan)
df[' funding_total_usd '] = pd.to_numeric(df[' funding_total_usd '], errors='coerce')
median_value = df[' funding_total_usd '].median()
df[' funding_total_usd '] = df[' funding_total_usd '].fillna(median_value)

In [17]: df['founded_year'] = df['founded_year'].fillna(df['founded_year'].median())
df['funding_rounds'] = df['funding_rounds'].fillna(df['funding_rounds'].median())
df['country_code'] = df['country_code'].fillna('UNKNOWN')

In [18]: df['country_code'] = LabelEncoder().fit_transform(df['country_code'])
df['status'] = LabelEncoder().fit_transform(df['status'])

In [19]: X = df.drop(columns='status')
y = df['status']

In [20]: scaler = StandardScaler()
X_scaled = scaler.fit_transform(X)

In [21]: X_train, X_test, y_train, y_test = train_test_split(X_scaled, y, test_size=0.2, random_state=42)

In [22]: svm = SVC()
svm.fit(X_train, y_train)
y_pred_svm = svm.predict(X_test)

In [23]: gb = GradientBoostingClassifier()
gb.fit(X_train, y_train)
y_pred_gb = gb.predict(X_test)

In [29]: print("SVM:")
print("Accuracy:", accuracy_score(y_test, y_pred_svm))
print("F1 Score:", f1_score(y_test, y_pred_svm))
print(classification_report(y_test, y_pred_svm, zero_division=0))

SVM:
Accuracy: 0.9243557772236076
F1 Score: 0.9606911447084233

```

	precision	recall	f1-score	support
0	0.00	0.00	0.00	728
1	0.92	1.00	0.96	8896
accuracy			0.92	9624
macro avg	0.46	0.50	0.48	9624
weighted avg	0.85	0.92	0.89	9624

```
In [30]: print("Gradient Boosting:")
print("Accuracy:", accuracy_score(y_test, y_pred_gb))
```

```
print("F1 Score:", f1_score(y_test, y_pred_gb))
print(classification_report(y_test, y_pred_gb, zero_division=0))
```

Gradient Boosting:

Accuracy: 0.9240440565253533

F1 Score: 0.960522762866555

	precision	recall	f1-score	support
0	0.00	0.00	0.00	728
1	0.92	1.00	0.96	8896
accuracy			0.92	9624
macro avg	0.46	0.50	0.48	9624
weighted avg	0.85	0.92	0.89	9624

## Классификация или регрессия?

В данной работе решалась задача классификации, а не регрессии. Это определяется по следующим признакам:

- Целевая переменная `status` была преобразована в двоичный классификационный признак:
  - 0 — компании со статусом "acquired"
  - 1 — компании с любым другим статусом ("other")
- Были использованы модели классификации:
  - Метод опорных векторов (SVM) — SVC
  - Градиентный бустинг — GradientBoostingClassifier
- Для оценки качества моделей применялись метрики классификации:
  - Accuracy
  - F1-score
  - А также precision и recall из отчёта `classification_report`

Таким образом, несмотря на наличие числовых признаков в данных, задача направлена на предсказание категории (принадлежит ли компания к классу "acquired" или нет), а не на предсказание непрерывной величины, что однозначно указывает на тип задачи — классификация.

## Какие метрики качества Вы использовали и почему?

В данной задаче были использованы следующие метрики:

1. Accuracy (доля правильных предсказаний) — показывает, какая часть объектов была классифицирована правильно. Это базовая метрика, но она может быть обманчивой при несбалансированных классах.
2. F1-Score (гармоническое среднее между precision и recall) — особенно полезна в задачах с несбалансированными классами. В данном случае класс "acquired" встречается редко, и F1-score лучше отражает реальное качество модели.

Также был выведен `classification report`, содержащий:

- precision — точность (сколько из предсказанных как "acquired" реально были такими),
- recall — полнота (сколько из всех настоящих "acquired" модель нашла),
- f1-score — итоговая мера качества.

## Какие выводы можно сделать о качестве построенных моделей?

1. Высокое значение Accuracy (~92%) и F1-Score (~0.96) для класса 1 ("other") говорит о том, что модель хорошо распознаёт основной (мажоритарный) класс.
2. Класс "acquired" (метка 0):
  - precision = 0, recall = 0, f1-score = 0 — модель вообще не распознала ни одного объекта этого класса. Это говорит о сильном дисбалансе классов.
  - Подтверждается тем, что из 9624 объектов — только 728 (~7.6%) относятся к классу 0.

1. SVM и Gradient Boosting показывают почти одинаковые результаты, но обе модели склонны игнорировать редкий класс.