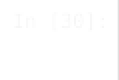
Ходырев Роман Владиславович

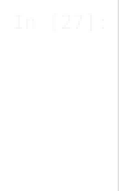


print("Gradient Boosting:")

print("Accuracy:", accuracy\_score(y\_test, y\_pred\_gb))

ИУ5-65Б

# 18 вариант



**import** pandas **as** pd

**import** numpy **as** np

**from** sklearn.model\_selection **import** train\_test\_split

**from** sklearn.preprocessing **import** LabelEncoder, StandardScaler

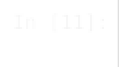
**from** sklearn.svm **import** SVC

**from** sklearn.ensemble **import** GradientBoostingClassifier

**from** sklearn.metrics **import** accuracy\_score, f1\_score, classification\_report



df **=** pd**.**read\_csv('investments\_VC.csv', encoding**=**'latin1', sep**=None**, engine**=**'python', on\_bad\_lines**=**'skip')



df **=** df[['status', ' funding\_total\_usd ', 'country\_code', 'funding\_rounds', 'founded\_year']] df **=** df**.**dropna(subset**=**['status'])



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']



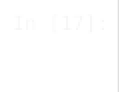
df['status'] **=** df['status']**.**apply(**lambda** x: 'acquired' **if** x **==** 'acquired' **else** 'other')



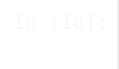
df[' funding\_total\_usd '] **=** df[' funding\_total\_usd ']**.**replace('[\$,]', '', regex**=True**)**.**replace('None', np**.**nan) df[' funding\_total\_usd '] **=** pd**.**to\_numeric(df[' funding\_total\_usd '], errors**=**'coerce')

median\_value **=** df[' funding\_total\_usd ']**.**median()

df[' funding\_total\_usd '] **=** df[' funding\_total\_usd ']**.**fillna(median\_value)



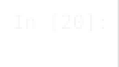
df['founded\_year'] **=** df['founded\_year']**.**fillna(df['founded\_year']**.**median()) df['funding\_rounds'] **=** df['funding\_rounds']**.**fillna(df['funding\_rounds']**.**median()) df['country\_code'] **=** df['country\_code']**.**fillna('UNKNOWN')



df['country\_code'] **=** LabelEncoder()**.**fit\_transform(df['country\_code']) df['status'] **=** LabelEncoder()**.**fit\_transform(df['status'])



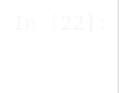
X **=** df**.**drop(columns**=**'status') y **=** df['status']



scaler **=** StandardScaler() X\_scaled **=** scaler**.**fit\_transform(X)

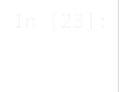


X\_train, X\_test, y\_train, y\_test **=** train\_test\_split(X\_scaled, y, test\_size**=**0.2, random\_state**=**42)



svm **=** SVC() svm**.**fit(X\_train, y\_train)

y\_pred\_svm **=** svm**.**predict(X\_test)



gb **=** GradientBoostingClassifier() gb**.**fit(X\_train, y\_train) y\_pred\_gb **=** gb**.**predict(X\_test)



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 |

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:

1. — компании со статусом "acquired"
2. — 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.