Требования и ограничения регрессионных моделей

- Между входными признаками и целевым должна быть линейная взаимосвязь.
- Из-за мультиколлинеарности коэффициенты модели станут неустойчивыми, и её будет невозможно интерпретировать.
- Перед обучением линейных моделей нужно масштабировать данные: признаки разного масштаба приводят к систематическим ошибкам.

Метрики классификации

Матрица ошибок — таблица, в которую занесено количество предсказаний модели разного типа. Позволяет визуально оценить результаты работы модели.

	Предсказание = 0	Предсказание = 1
Реальность = 0	True Negative (TN)	False Positive (FP)
Реальность = 1	False Negative (FN)	True Positive (TP)

- TN (True Negative), **истинно отрицательный ответ** — количество объектов класса 0, которые были верно классифицированы, то есть им был присвоен класс 0.

- FN (False Negative), **ложноотрицательный ответ** — количество объектов класса 1, которые были неверно классифицированы, то есть им был присвоен класс 0.

- TP (True Positive), **истинно положительный ответ** — количество объектов класса 1, которые были верно классифицированы, то есть им был присвоен класс 1.

- FP (False Positive), **ложноположительный ответ** — количество объектов класса 0, которые были неверно классифицированы, то есть им был присвоен класс 1.

Метрики в задаче классификации

```
from sklearn.metrics import confusion_matrix
import seaborn as sns

# подсчёт матрицы ошибок
cm = confusion_matrix(
    y_test,  # предсказанные классы
    y_pred  # истинные классы
)

# визуализации матрицы ошибок через тепловую карту
sns.heatmap(cm, annot=True, fmt='d', cmap='Blues_r')
plt.ylabel('True label')
plt.xlabel('Predicted');
```

Accuracy — эта метрика показывает долю верных ответов модели.

$$accuracy = \frac{T}{N},$$

где:

- T — количество верных ответов;
- N — общее число ответов.

```
from sklearn.metrics import accuracy_score


accuracy = accuracy_score(
    y_test,  # предсказанные классы
    y_pred  # истинные классы
)
```

Метрики в задаче классификации

Precision, точность — эта метрика показывает точность, с которой модель присваивает объектам класс 1. Иными словами, precision определяет, не слишком ли часто модель выставляет класс 1 объектам класса 0.

$$precision = \frac{TP}{TP + FP}$$

```
from sklearn.metrics import precision_score


precision = precision_score(
    y_test,  # предсказанные классы
    y_pred  # истинные классы
)
```

Recall, полнота — эта метрика измеряет, смогла ли модель классификации присвоить класс 1 всем объектам этого класса.

$$recall = \frac{TP}{TP + FN}$$

```
from sklearn.metrics import recall_score


recall = recall_score(
    y_test,  # предсказанные классы
    y_pred  # истинные классы
)
```

Настройка порога классификации

```
from sklearn.linear_model import LogisticRegression

# создание и обучение модели
clf = LogisticRegression()
clf = clf.fit(X_train_scalled, y_train)

# предсказание вероятностей
y_proba = clf.predict_proba(X_test_scalled)[:, 1]

# список с порогами
thresholds = [round(i, 2) for i in np.linspace(0.1, 1, num=4, endpoint=False)]

# добавить столбцы с новыми предсказаниями в таблицу
for i in thresholds:
  data['y_pred_' + str(i)] = data['y_proba'].apply(lambda x: 1 if x >= i else 0)
```