# **Институт интеллектуальных кибернетических систем НИЯУ МИФИ**

# **Группа: М24-525**

# **Студент: Колесников Владислав Вячеславович**

# **Классическое машинное обучение**

# **Курсовая работа.**

**2025**

## Анализ данных (EDA)

**1. Общие сведения о данных**

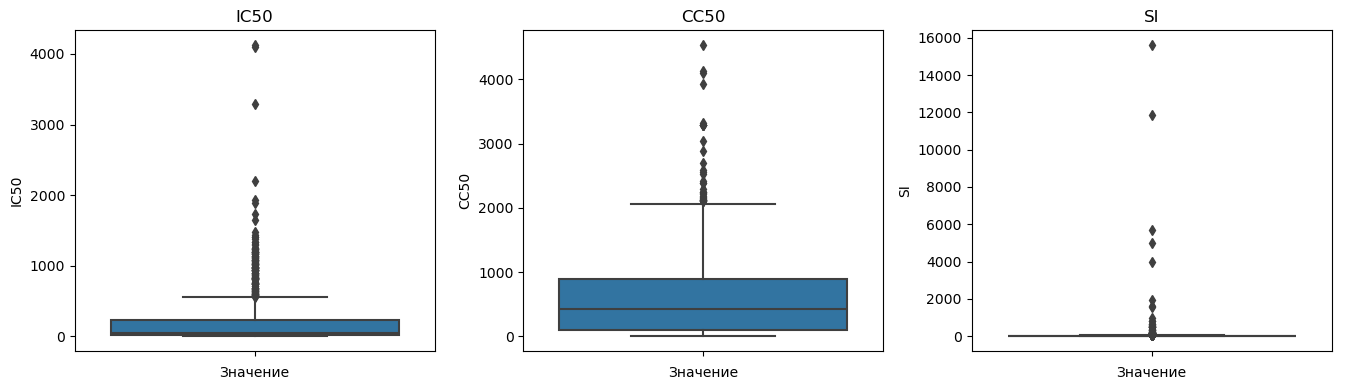
* **Объем данных:**
  + 1001 наблюдение.
  + 213 числовых признаков (107 типа float64, 106 типа int64).
* **Целевые переменные:**
  + IC50, mM (концентрация ингибирования).
  + CC50, mM (цитотоксическая концентрация).
  + SI (индекс селективности).

**2. Пропущенные значения**

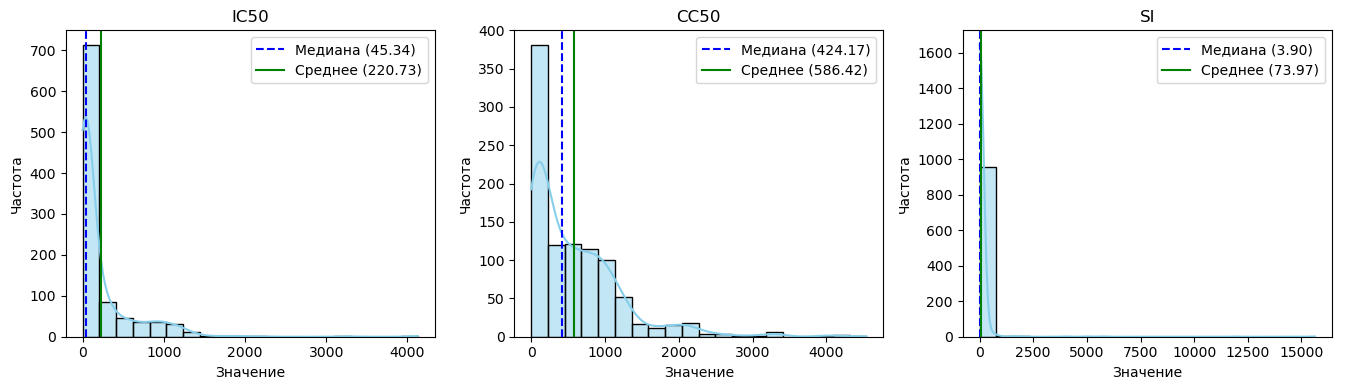
* **Обнаружены в 8 столбцах** (по 3 пропуска в каждом, 0.3% от данных):

**3. Аномалии и уникальные значения**

* **Константные признаки:**
  + SMR\_VSA8, SlogP\_VSA9, NumRadicalElectrons (всезначения = 0)
* **Выбросы:**
  + Максимальные значения IC50 (4128.53), CC50 (4538.98)и SI (15620.6) значительно превышают 75-й перцентиль.



**4. Распределения ключевых переменных**

****

1. Распределение IC50 характеризуется:

* Сильной правосторонней асимметрией (среднее 222.81 vs медиана 46.59)
* Широким диапазоном значений от 0.0035 до 4128.53
* 75% значений сосредоточены ниже 224.98
* Наличие экстремальных выбросов (>4000)

2. Распределение CC50 показывает:

* Среднее значение 589.11 значительно превышает медиану 411.04
* 25% соединений имеют CC50 < 100
* Максимальные значения достигают 4538.98
* Стандартное отклонение 642.87 свидетельствует о высокой вариативности

3. Распределение SI показывает:

* Крайне неравномерное распределение (среднее 72.51 vs медиана 3.85)
* Диапазон значений от 0.0115 до 15620.6
* 75% значений ниже 16.57
* Коэффициент вариации >900%

4. Сравнительный анализ:

* Все три целевые переменные демонстрируют правостороннюю асимметрию
* Наибольший относительный разброс характерен для SI
* Распределения IC50 и CC50 имеют схожий характер, но разные масштабы
* Наличие экстремальных выбросов характерно для всех трех параметров

**5. Корреляции и мультиколлинеарность**

A blue and white grid with black text

AI-generated content may be incorrect.

* **Высокая корреляция (>0.8) между:**

HeavyAtomMolWt, Chi4n, MolWt, fr\_C\_O, NumHDonors, fr\_NH0, fr\_ether, fr\_phenol\_noOrthoHbond, Chi1v, Chi1, Chi0n, BCUT2D\_LOGPLOW, fr\_Al\_OH, Chi0v, MinAbsPartialCharge, fr\_COO2, Kappa3, BCUT2D\_LOGPHI, fr\_C\_O\_noCOO, Chi0, SlogP\_VSA4, Kappa1, MinPartialCharge, VSA\_EState10, fr\_alkyl\_halide, NumHAcceptors, SlogP\_VSA6, Chi4v, NumHeteroatoms, ExactMolWt, fr\_COO, BCUT2D\_MWHI, SlogP\_VSA11, NumAromaticRings, Kappa2, SMR\_VSA4, r\_halogen, Chi3n, NumAromaticCarbocycles, SPS, fr\_benzene, SMR\_VSA7, MolMR, State\_VSA10, Chi1n, NumSaturatedCarbocycles, fr\_Al\_OH\_noTert, fr\_phenol, NumSaturatedRings, VSA\_EState3, FpDensityMorgan3, HallKierAlpha, SlogP\_VSA5, fr\_Ar\_N, HeavyAtomCount, Chi2n, NumValenceElectrons, FpDensityMorgan2, TPSA, SlogP\_VSA10, fr\_ketone\_Topliss, Chi2v, VSA\_EState6, LabuteASA, SMR\_VSA1, NumRotatableBonds, Chi3v

**6. В процессе предобработки были выполнены следующие шаги**

* Полные дубликаты строк удалены
* Пропуски заполнить медианами
* Константные признаки удалены
* Признаки с высокой мультиколлинеарностью удалены
* Произведена нормализация (StandardScaler).

## Исследование моделей для решения задач регрессии.

## Общая часть

**Для исследования задач регрессии были использованы следующие модели:**

* **LinearRegression** - Базовая линейная модель, быстрая но чувствительная к выбросам и мультиколлинеарности
* **Ridge** - Линейная модель с L2-регуляризацией, устойчива к переобучению
* **Lasso** - Линейная модель с L1-регуляризацией, выполняет отбор признаков
* **ElasticNet** - Комбинация L1+L2 регуляризации, баланс между Ridge и Lasso
* **BayesianRidge** - Байесовский подход с оценкой неопределённости параметров
* **DecisionTreeRegressor** - Простое нелинейное дерево решений, склонно к переобучению
* **RandomForestRegressor** - Ансамбль деревьев с бэггингом, устойчив к шуму
* **GradientBoostingRegressor** - Бустинговый ансамбль, высокая точность но требует настройки
* **SVR** - Метод опорных векторов, эффективен для сложных нелинейных зависимостей
* **KNeighborsRegressor** - Метод ближайших соседей, требует масштабирования признаков
* **CatBoostRegressor** - Продвинутый бустинг с автоматической обработкой категорий
* **MLPRegressor** - Нейросетевая модель, требует больших вычислительных ресурсов

**Описание использованных гиперпараметров моделей регрессии**

**1. Линейные модели с регуляризацией**

**Ridge (гребневая регрессия):**

* alpha (0.1, 1.0, 10.0) - сила регуляризации L2
* solver ('auto', 'svd', 'cholesky', 'lsqr', 'sparse\_cg', 'sag', 'saga') - алгоритм оптимизации

**Lasso:**

* alpha (0.1, 1.0, 10.0) - коэффициент регуляризации L1
* max\_iter (1000, 5000, 10000) - максимальное число итераций
* tol (1e-4, 1e-3, 1e-2) - допуск сходимости

**ElasticNet:**

* alpha (0.1, 1.0, 10.0) - общая сила регуляризации
* l1\_ratio (0.1, 0.5, 0.9) - соотношение L1/L2 регуляризации
* max\_iter (1000, 5000, 10000) - максимальное число итераций
* tol (1e-4, 1e-3, 1e-2) - допуск сходимости

**BayesianRidge:**

* Параметры априорных распределений:
  + alpha\_1 (1e-6, 1e-5, 1e-4)
  + alpha\_2 (1e-6, 1e-5, 1e-4)
  + lambda\_1 (1e-6, 1e-5, 1e-4)
  + lambda\_2 (1e-6, 1e-5, 1e-4)
* max\_iter (100, 300, 500) - ограничение итераций

**2. Древовидные модели и ансамбли**

**GradientBoostingRegressor:**

* n\_estimators (50, 100, 200) - число деревьев
* learning\_rate (0.01, 0.1, 0.2) - темп обучения
* max\_depth (3, 5, 10) - глубина деревьев
* Параметрывыборки
  + Subsample (0.8, 0.9, 1.0)
  + min\_samples\_split (2, 5, 10)
  + min\_samples\_leaf (1, 2, 4)

**RandomForestRegressor:**

* n\_estimators (50, 100) - количество деревьев
* max\_depth (5, 10, None) - максимальная глубина

**DecisionTreeRegressor:**

* max\_depth (3, 5, 7, 10, None) - контроль сложности дерева

**CatBoostRegressor:**

* depth (4, 6, 8) - глубина деревьев
* learning\_rate (0.01, 0.1) - скорость обучения

**3. Нейросетевые и метрические методы**

**MLPRegressor (нейросеть):**

* hidden\_layer\_sizes [(100,), (100,50)]- архитектурасети
* max\_iter (500-1000) - эпохи обучения

**SVR (метод опорных векторов):**

* C (0.1, 1, 10) - параметр регуляризации
* kernel ('rbf', 'linear')- типядра

**KNeighborsRegressor:**

* n\_neighbors (3, 5, 7, 10) - число соседей
* leaf\_size (10, 30, 50) - размер листьев для ускорения поиска
* p (1-2) - метрика (1 - Манхэттен, 2 - Евклид)
* weights ('uniform', 'distance')- взвешиваниесоседей
* algorithm ('ball\_tree', 'kd\_tree', 'brute') - методпоискасоседей

**Основные метрики**

* Accuracy : доля правильно предсказанных классов.
* Recall (Полнота) : способность модели находить все положительные примеры.
* ROC AUC (Area Under Curve) : мера качества бинарной классификации.
* R² (Коэффициент детерминации) : объясняет, насколько хорошо модель воспроизводит данные.

**Методы редуцирования признаков используемые в исследовании**

* No Reduction
* PCA
* UMAP

## Исследование моделей для решения задач регрессии **IC50**

A screenshot of a computer

AI-generated content may be incorrect.

A close-up of a graph

AI-generated content may be incorrect.

A graph of a graph

AI-generated content may be incorrect.

## Результат исследования

**Модели с лучшими показателями метрик**

1. ROC AUC

* CatBoost с PCA : 0.7708
* KNN без редукции : 0.7673
* Gradient Boosting с PCA : 0.7642

1. Accuracy

* MLP без редукции : 0.6407
* KNN без редукции : 0.6287
* SVR с PCA : 0.6287

1. Recall

* У нескольких моделей (Decision Tree, Random Forest, Gradient Boosting и др.) значение Recall = 1.0

1. R²

* CatBoost без редукции : 0.1830
* MLP без редукции : -0.1970 (отрицательное значение говорит о плохом соответствии модели данным)

**Вывод:**

Лучшим выбором будет**CatBoostRegressor** с применением PCA

**Параметры:**depth: 6; learning\_rate: 0.01;

Обоснование:

* Высокий ROC AUC = 0.7708 — лучший результат среди всех моделей.
* Recall = 1.0 — хорошо находит все положительные примеры.
* R² = 0.1775 — *стабильнее, чем у многих других*объяснение дисперсии данных
* Использование PCA снижает размерность признаков и помогает бороться с переобучением, сохраняя высокую эффективность.

## Исследование моделей для решения задач регрессии **CC50**

A black and white table with numbers

AI-generated content may be incorrect.

A close-up of a graph

AI-generated content may be incorrect.

A graph of a number of colored lines

AI-generated content may be incorrect.

**Модели с лучшими показателями метрик**

1. ROC AUC

* KNN (No Reduction) — 0.8535
* CatBoost (PCA) — 0.8372
* Random Forest (No Reduction) — 0.8283
* CatBoost (No Reduction) — 0.8254

2. Accuracy

* KNN (No Reduction) — 0.8021
* CatBoost (PCA) — 0.7380
* Gradient Boosting (No Reduction) — 0.7273
* CatBoost (No Reduction) — 0.7219
* Random Forest (No Reduction) — 0.7219

3. Recall

* Decision Tree (UMAP) — 1.0
* CatBoost (No Reduction) — 0.8876
* KNN (No Reduction) — 0.8652
* Random Forest (No Reduction) — 0.8427
* Gradient Boosting (No Reduction) — 0.8539

DecisionTree с UMAP имеет полноту 1.0, но это может указывать на переобучение или смещение в данных. Более стабильные варианты: CatBoost , GradientBoosting , KNN и RandomForest без редукции.

4. R²

* CatBoost (No Reduction) — 0.5108
* CatBoost (PCA) — 0.5076
* Random Forest (No Reduction) — 0.4881
* MLP (PCA) — 0.4556
* KNN (No Reduction) — 0.4811

**Вывод:**

Лучшим выбором будет**CatBoostRegressor** с применением No Reduction

**Параметры:**depth: 6; learning\_rate: 0.01;

Обоснование:

* ROC AUC = 0.825
* Recall = 0.888
* R² = 0.511
* Accuracy =0.722

Модель демонстрирует наивысшую объясняющую способность (R²) , хорошие значения Recall и ROC AUC , а также не требует дополнительных преобразований признаков.

## Исследование моделей для решения задач регрессии **SI**

A black and white screen with white text

AI-generated content may be incorrect.

A close-up of a graph

AI-generated content may be incorrect.

A graph of a graph

AI-generated content may be incorrect.

**Модели с лучшими показателями метрик**

1. ROCAUC

* CatBoost (безредукции) : 0.6759
* Gradient Boosting (безредукции) : 0.6720
* RandomForest (без редукции) : 0.6894

2. Accuracy

* K-Nearest Neighbors (UMAP) : 0.6118
* SVR (PCA) : 0.6471
* MLP (PCA) : 0.6412

3. Recall

* Все моделикроме SVRпоказали максимальное значение полноты, равное или приближающееся к 1.0.

4. R²

* CatBoost (без редукции) : 0.0907
* Random Forest (безредукции) : 0.0671
* Gradient Boosting (безредукции) : 0.0610

**Вывод:**

Лучшим выбором будет **CatBoostRegressor** с применением No Reduction

**Параметры:**depth: 4; learning\_rate: 0.01;

Обоснование:

* ROC AUC = 0.6759
* Recall = 1
* R² = 0.0907
* Accuracy =0.5941

Модель обеспечивает оптимальный баланс между всеми метриками. Хотя ее точность не самая высокая, она компенсируется отличным значением ROC AUC и стабильностью (R²). Также стоит отметить, что высокое значение полноты (Recall = 1.0) гарантирует, что модель не пропустит положительные случаи.

## Исследование моделей для решения задач классификации.

## Общая часть

**Для исследования задач классификации были использованы следующие модели:**

* LogisticRegression : Линейная модель для бинарной и мультиномиальной классификации, основанная на логистической функции.
* DecisionTreeClassifier : Модель, строящая дерево решений для разделения данных на классы на основе простых правил.
* RandomForestClassifier : Ансамбль деревьев решений, где каждое дерево голосует за класс, а итоговый класс определяется большинством голосов.
* CatBoostClassifier : Градиентный бустинг, оптимизированный для работы с категориальными признаками и устойчивый к переобучению.
* MLPClassifier : Многослойный перцептрон для классификации с использованием скрытых слоев и нелинейных активаций.
* SVC : Метод опорных векторов, разделяющий данные гиперплоскостью с максимальным зазором для классификации.
* SGDClassifier : Линейная модель, обучаемая методом стохастического градиентного спуска для эффективной классификации больших данных.
* GradientBoostingClassifier : Ансамблевая модель, последовательно улучшающая ошибки предыдущих деревьев для повышения точности классификации.
* AdaBoostClassifier : Адаптивный бустинг, увеличивающий вес ошибочно классифицированных объектов для улучшения качества модели.
* XGBClassifier : Эффективная реализация градиентного бустинга с поддержкой регуляризации и быстрым обучением для задач классификации.

**Описание использованных гиперпараметров моделей регрессии**

1. LogisticRegression

* C (0.1, 1, 10) - Обратная величина коэффициента регуляризации
* solver ('liblinear', 'lbfgs') - Алгоритм оптимизации для поиска параметров модели.

2. DecisionTreeClassifier

* max\_depth (3, 5, 7, 10, None) - Максимальная глубина дерева.
* criterion ('gini', 'entropy') - Критерий разделения в узлах дерева.

3. RandomForestClassifier

* n\_estimators (50, 100) - Количество деревьев в ансамбле
* max\_depth (5, 10, None) - Максимальная глубина каждого дерева.
* max\_features ('sqrt', 'log2', None) - Количество признаков, рассматриваемых для разделения в каждом узле..

4. CatBoostClassifier

* depth (4, 6, 8) - Глубина деревьев в ансамбле.
* learning\_rate (0.01, 0.1) - Скорость обучения.
* iterations (100, 500) - Количество итераций (деревьев) в ансамбле.

5. MLPClassifier

* hidden\_layer\_sizes ((100,), (100, 50)) - Размеры скрытых слоев нейронной сети.
* activation ('relu', 'tanh') - Функция активации нейронов.
* max\_iter (500, 1000) - Максимальное количество итераций для сходимости алгоритма.

6. SVC (Support Vector Classifier)

* C (0.1, 1, 10) - Параметр регуляризации.
* kernel ('rbf', 'linear') - Ядро для преобразования данных в пространство более высокой размерности.

7. SGDClassifier

* loss ('hinge', 'log\_loss', 'modified\_huber') - Функция потерь.
* penalty ('l2', 'l1', 'elasticnet') - Тип регуляризации.
* alpha (0.0001, 0.001, 0.01) - Коэффициент регуляризации.
* max\_iter (1000, 5000) - Максимальное количество итераций для сходимости.

8. GradientBoostingClassifier

* n\_estimators (50, 100, 200) - Количество деревьев в ансамбле.
* learning\_rate (0.01, 0.1, 0.2) - Скорость обучения.
* max\_depth (3, 5, 10) - Максимальная глубина каждого дерева.
* subsample (0.8, 0.9, 1.0) - Доля выборки, используемая для обучения каждого дерева.

9. AdaBoostClassifier

* n\_estimators (50, 100, 200) - Количество базовых моделей (деревьев) в ансамбле.
* learning\_rate (0.01, 0.1, 1.0) - Скорость обучения.

10. XGBClassifier

* n\_estimators (50, 100, 200) - Количество деревьев в ансамбле.
* learning\_rate (0.01, 0.1, 0.2) - Скорость обучения.
* max\_depth (3, 5, 10) - Максимальная глубина каждого дерева.
* subsample (0.8, 0.9, 1.0) - Доля выборки, используемая для обучения каждого дерева.
* colsample\_bytree (0.8, 0.9, 1.0) - Доля признаков, используемых для обучения каждого дерева.

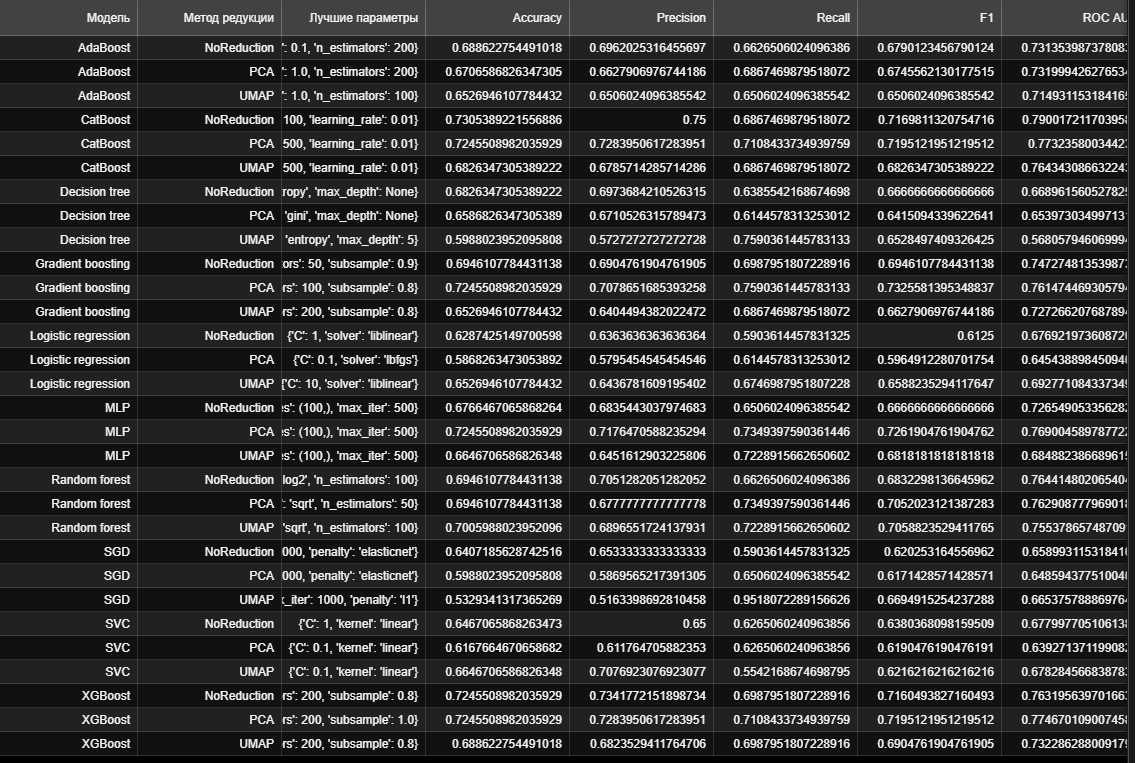
**Основные метрики**

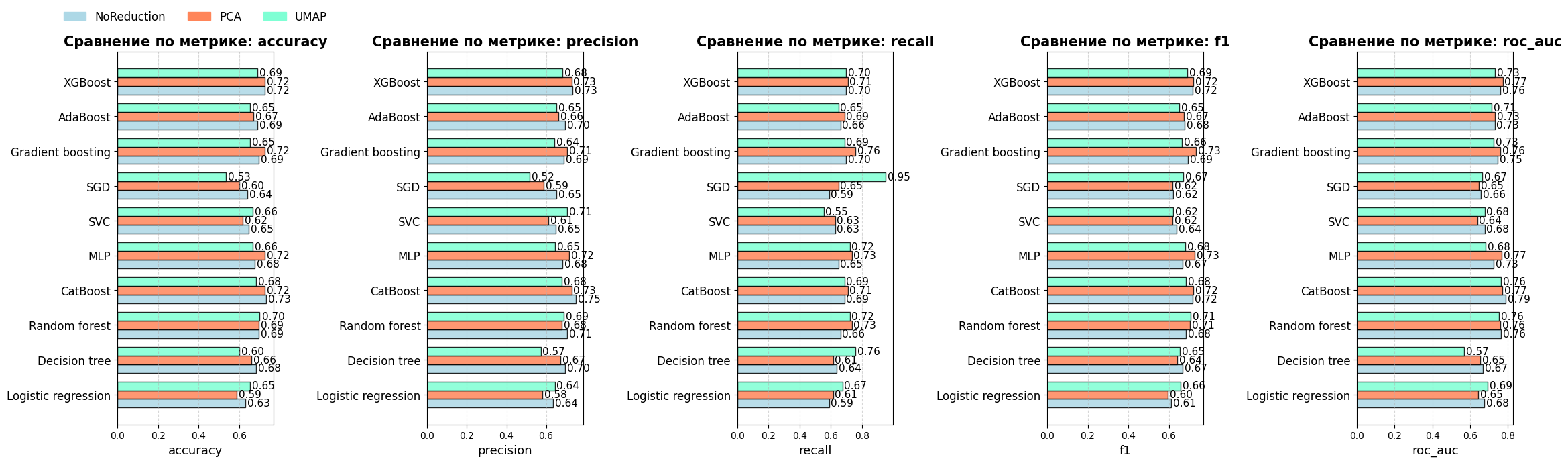
* Accuracy : доля правильно предсказанных классов.
* Recall (Полнота) : способность модели находить все положительные примеры.
* ROC AUC (Area Under Curve) : мера качества бинарной классификации.
* R² (Коэффициент детерминации) : объясняет, насколько хорошо модель воспроизводит данные.

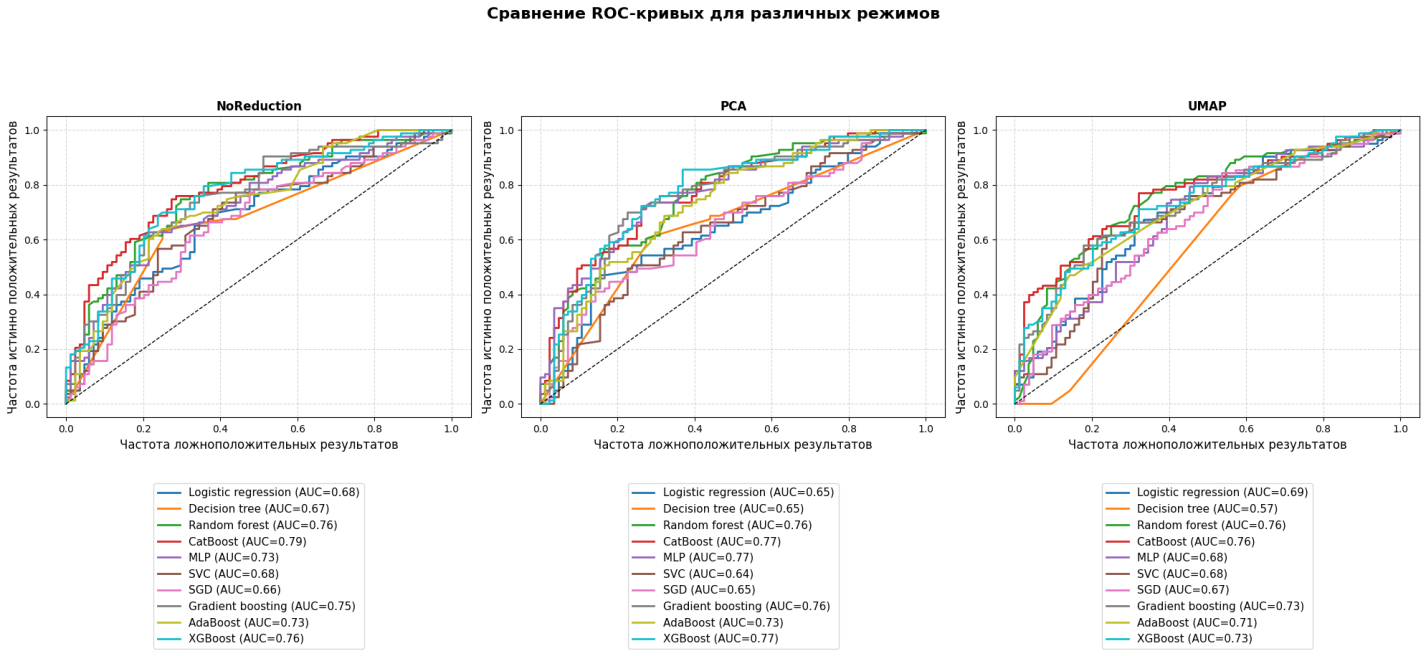
**Методы редуцирования признаков используемые в исследовании**

* No Reduction
* PCA
* UMAP

## Исследование моделей для решения задач классификации: превышает ли значение IC50 медианное значение выборки







**Модели с лучшими показателями метрик**

1. ROC AUC
   * CatBoost NoReduction 0.7900
   * XGBoost PCA 0.7747
   * MLP PCA 0.7690
   * RandomForest NoReduction 0.7644
   * GradientBoosting PCA 0.7615
2. Accuracy
   * CatBoost (NoReduction) : 0.7305
   * XGBoost (NoReduction) : 0.7246
   * RandomForest UMAP 0.7006
   * AdaBoost NoReduction 0.6886
3. Recall
   * SGD UMAP 0.9518
   * MLP UMAP 0.7229
   * GradientBoosting PCA 0.7590
   * CatBoost PCA 0.7108
   * RandomForest UMAP 0.7229 R²
4. F1-score
   * MLP PCA 0.7262
   * GradientBoosting PCA 0.7326
   * CatBoost PCA 0.7195
   * XGBoost NoReduction 0.7160
   * CatBoost NoReduction 0.7170
5. Precision
   * CatBoost NoReduction 0.75
   * XGBoost NoReduction 0.7342
   * MLP PCA 0.7176
   * RandomForest NoReduction 0.7051
   * GradientBoosting PCA 0.7079

**Вывод:**

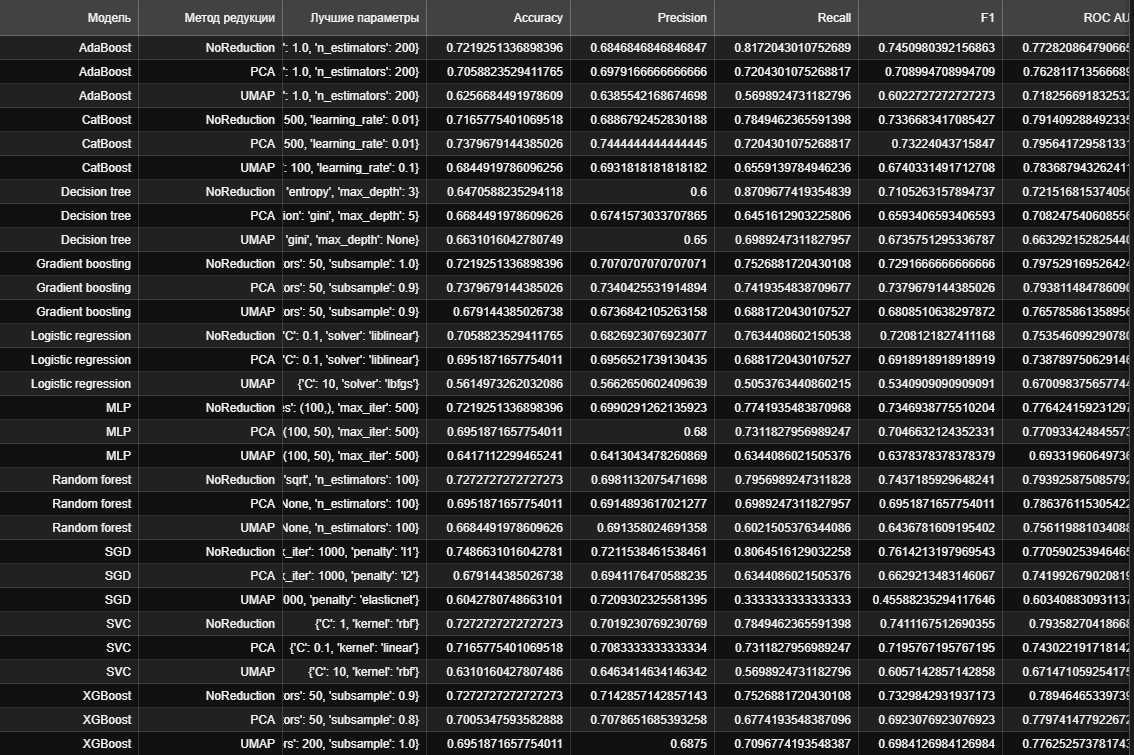
Лучшим выбором будет CatBoostClassifier с NoReduction

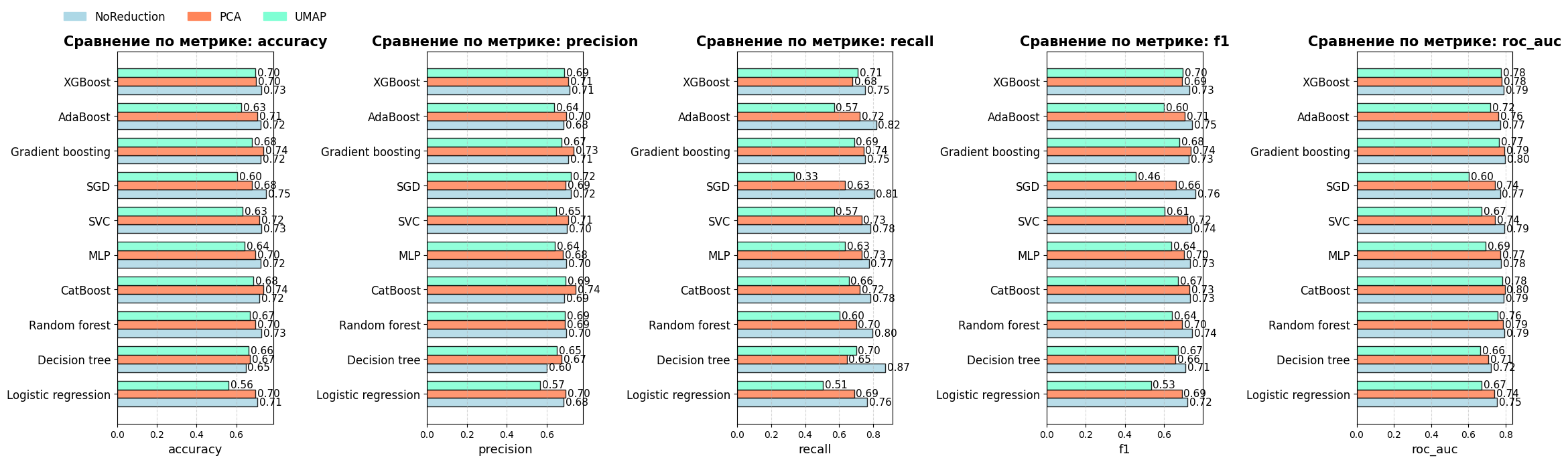
**Параметры:** depth: 8; iterations: 100; learning\_rate: 0.01;

Обоснование:

* Наивысший ROC AUC (0.7900) — говорит о лучшем разделении классов.
* Высокая точность (Precision = 0.75) — минимальное количество ложноположительных срабатываний.
* Хороший F1-score (0.717) — умеренный баланс между полнотой и точностью.
* Без метода редукции — сохраняет все исходные признаки, что может быть важно для интерпретируемости.

Исследование моделей для решения задач классификации: превышает ли значение CC50 медианное значение выборки







**Модели с лучшими показателями метрик**

ROC AUC

* + CatBoost PCA 0.7956
  + CatBoost NoReduction 0.7914
  + XGBoost UMAP 0.7763
  + SVC NoReduction 0.7936
  + SGD NoReduction 0.7706

Accuracy

* + RandomForest NoReduction 0.7273
  + CatBoost PCA 0.7380
  + GradientBoosting PCA 0.7380
  + AdaBoost NoReduction 0.7219
  + MLP NoReduction 0.7219

Recall

* + SGD NoReduction 0.8065
  + DecisionTree NoReduction 0.8710
  + RandomForest NoReduction 0.7957
  + LogisticRegression NoReduction 0.7634
  + CatBoost NoReduction 0.7849

F1-score

* + SVC NoReduction 0.7411
  + RandomForest NoReduction 0.7437
  + GradientBoosting PCA 0.7380
  + CatBoost PCA 0.7322
  + SGD NoReduction 0.7614

Precision

* + SVC NoReduction 0.7019
  + SGD NoReduction 0.7212
  + CatBoost PCA 0.7444
  + MLP NoReduction 0.6990
  + RandomForest NoReduction 0.6981

**Вывод:**

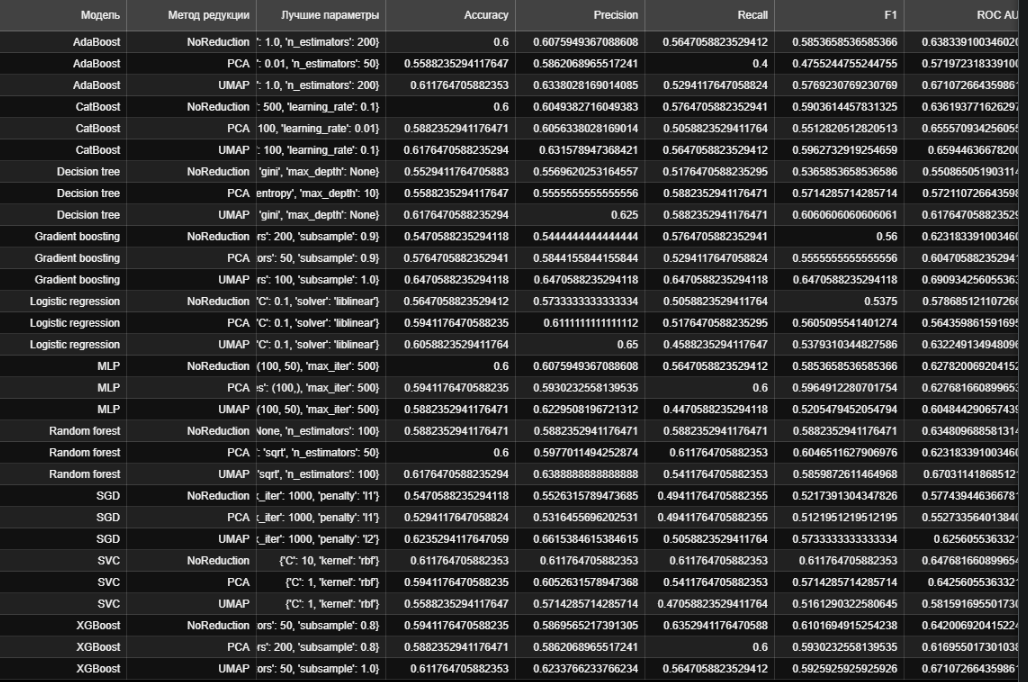
Лучшим выбором будет CatBoostClassifier с PCA

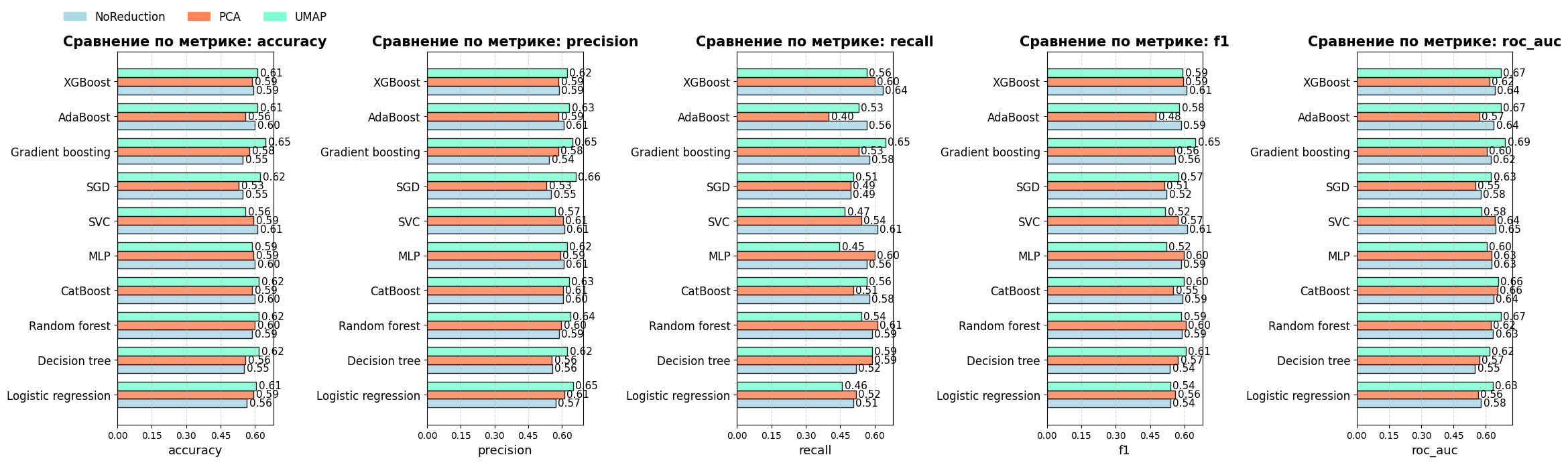
**Параметры:** depth = 6; iterations = 100; learning\_rate = 0.1;

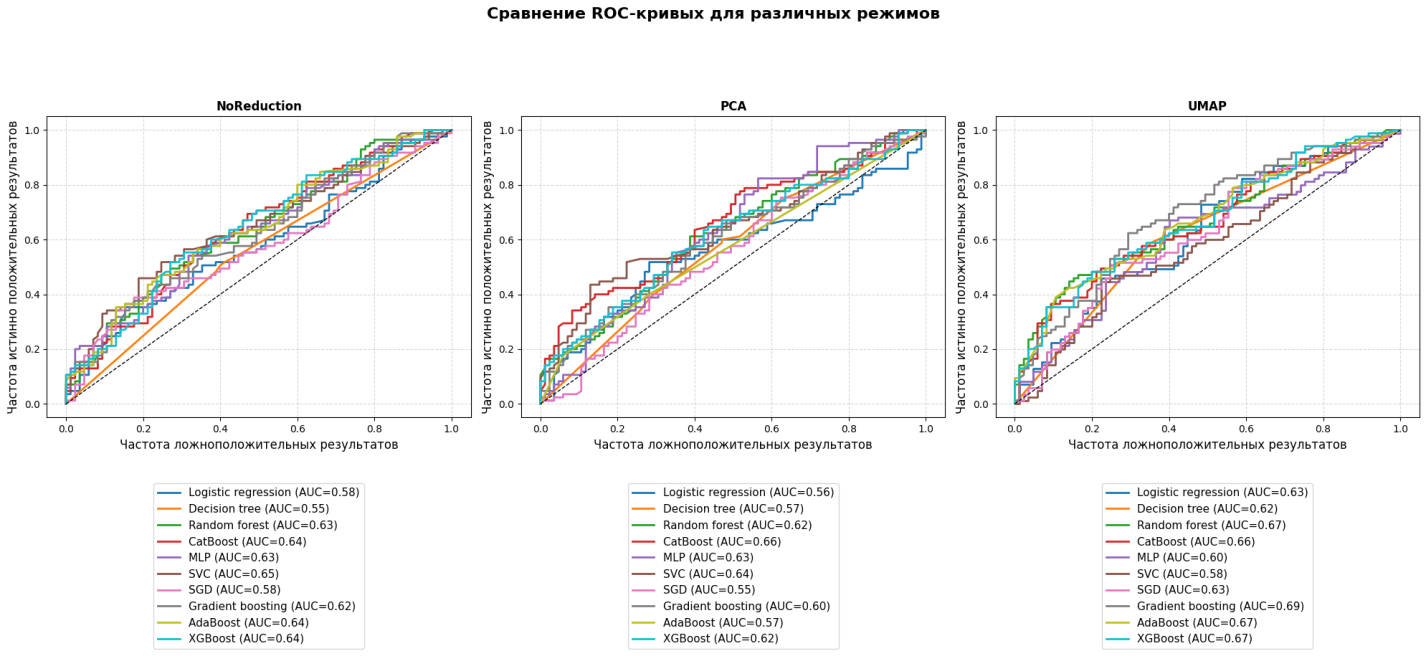
Обоснование:

* ROC AUC: 0.7956 (лучший)
* Precision: 0.7444 (лучший)
* Accuracy: 0.7380 (высокий)
* F1-score: 0.7322 (хороший)
* Recall: 0.7204 (умеренный)

Исследование моделей для решения задач классификации: превышает ли значение SI медианное значение выборки







**Модели с лучшими показателями метрик**

1. ROC AUC
   * GradientBoosting UMAP 0.6909
   * CatBoost UMAP 0.6594
   * RandomForest UMAP 0.6703
   * SGD UMAP 0.6256
2. Accuracy
   * SGD UMAP 0.6235
   * GradientBoosting UMAP 0.6470
   * RandomForest UMAP 0.6176
   * XGBoost UMAP 0.6117
3. Recall
   * GradientBoosting UMAP 0.6470
   * RandomForest UMAP 0.5411
   * SGD UMAP 0.5058
   * CatBoost UMAP 0.5647
4. F1-score
   * GradientBoosting UMAP 0.6470
   * SGD UMAP 0.5733
   * RandomForest UMAP 0.5859
   * CatBoost UMAP 0.5962
5. Precision
   * SVC NoReduction 0.6117
   * SGD UMAP 0.6615
   * GradientBoosting UMAP 0.6470
   * RandomForest UMAP 0.6388

**Вывод:**

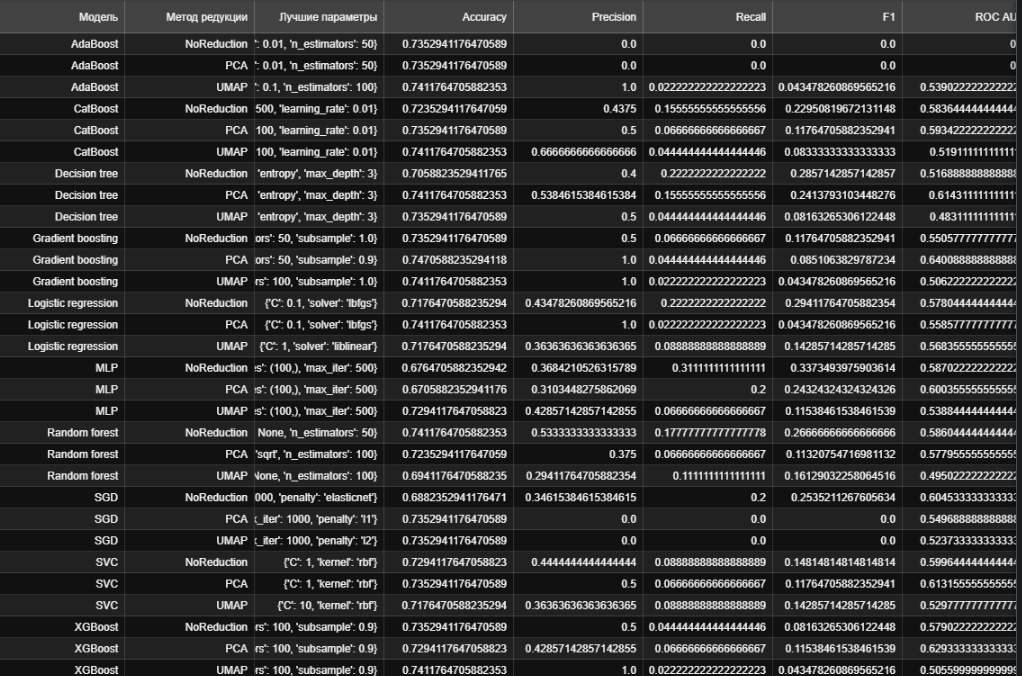
Лучшим выбором будет GradientBoosting c UMAP

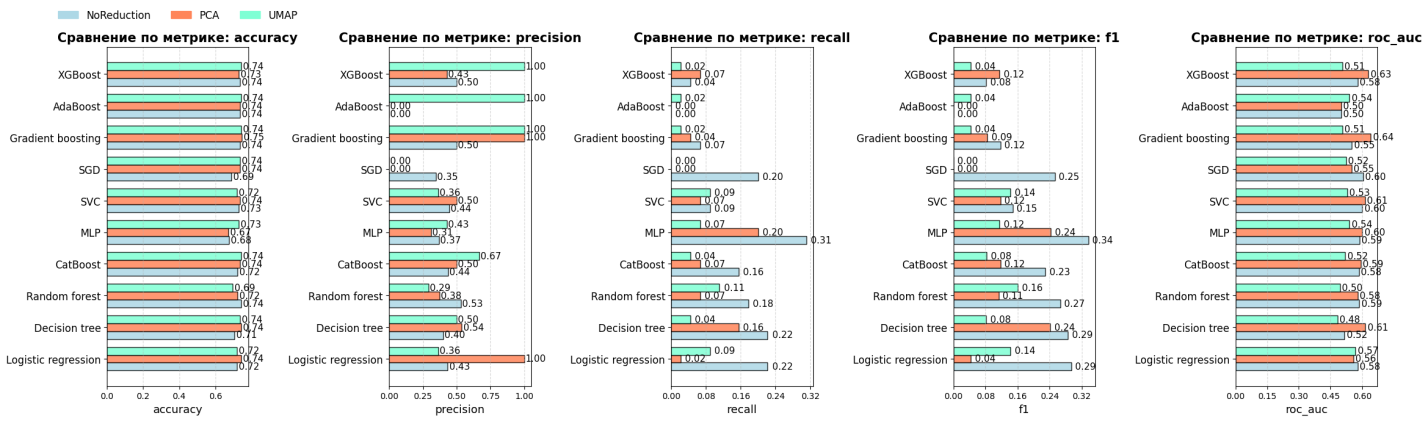
**Параметры:** learning\_rate = 0.2; max\_depth = 5; n\_estimators = 100; subsample = 1.0

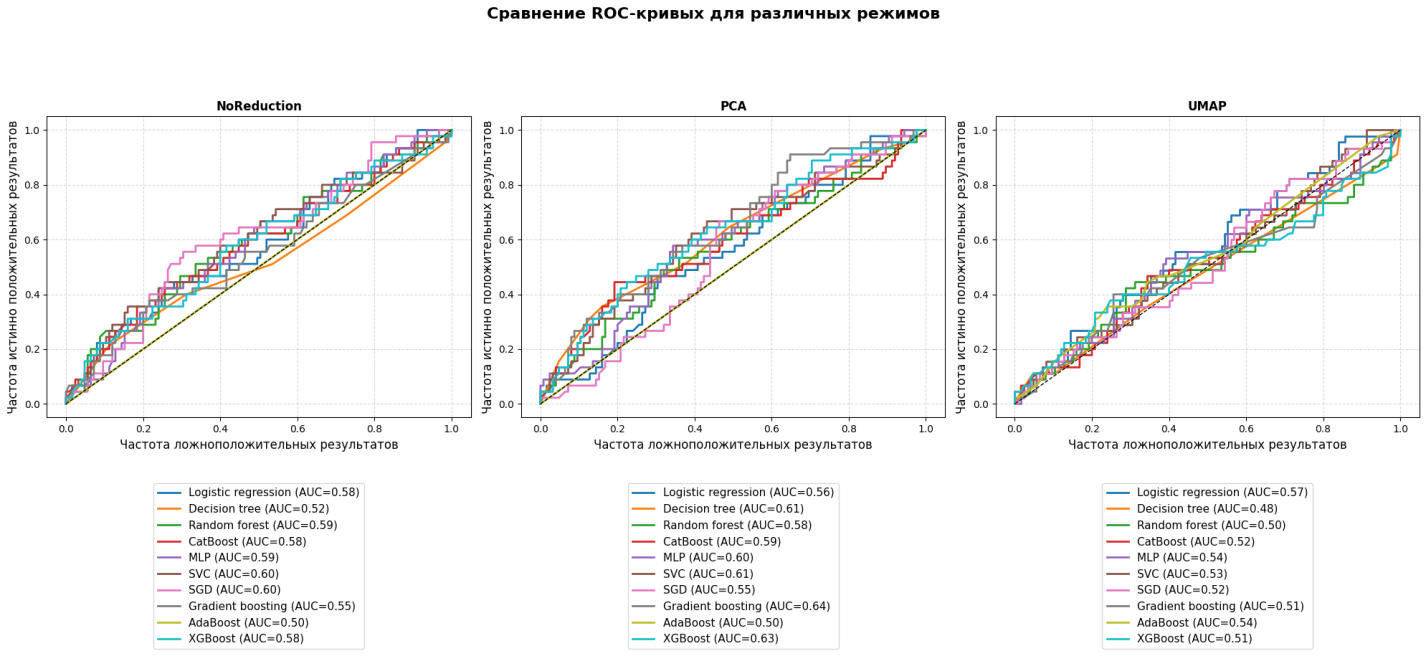
Обоснование:

* Высокое значение ROC AUC (0.6909) — хорошая способность различать классы.
* Лучший F1-score — баланс между precision и recall.
* Наивысший recall — минимизирует ложноотрицательные ошибки.
* Хорошая точность — меньше ложноположительных ошибок.

Исследование моделей для решения задач классификации: превышает ли значение SI значение 8







**Модели с лучшими показателями метрик**

1. ROC AUC
   * GradientBoosting PCA 0.6401
   * LogisticRegression PCA 0.5586
   * GradientBoosting UMAP 0.5062
   * CatBoost NoReduction 0.5836
2. Accuracy
   * GradientBoosting UMAP 0.7412
   * MLP NoReduction 0.6765
   * DecisionTree PCA 0.7412
   * LogisticRegression PCA 0.7412
3. Recall
   * AdaBoost UMAP 0.0222
   * SGD UMAP 0.0000
   * DecisionTree NoReduction 0.2222
   * LogisticRegression NoReduction 0.2222
4. F1-score
   * MLP NoReduction 0.3373
   * DecisionTree NoReduction 0.2857
   * LogisticRegression NoReduction 0.2941
   * CatBoost NoReduction 0.2295
5. Precision
   * GradientBoosting UMAP 1.0
   * AdaBoost UMAP 1.0
   * LogisticRegression PCA 1.0
   * XGBoost UMAP 1.0

**Вывод:**

Лучшим выбором будет MLP с NoReduction

**Параметры:** activation = 'relu'; hidden\_layer\_sizes = (100,); max\_iter = 500

Обоснование:

* Наивысший F1-score (0.3373) — лучший баланс между precision 0.3684 и recall 0.3111.
* Хороший уровень полноты Recall (0.3111) — относительно хорошо находит положительные случаи.
* Приемлемый ROC AUC (0.5870) — указывает на умеренную способность разделения классов.
* Не самая высокая точность и ROC AUC, но компенсируется хорошим F1.