

# **Институт интеллектуальных кибернетических систем НИЯУ МИФИ**

**Группа: М24-525  
Студент: Колесников  
Владислав Вячеславович**

**Классическое машинное обучение**

**Курсовая работа.**

# Анализ данных (EDA)

## 1. Общие сведения о данных

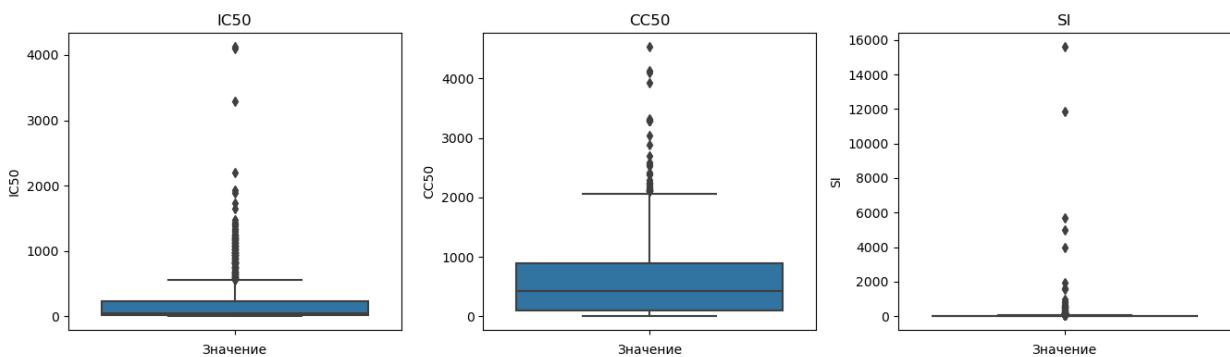
- **Объем данных:**
  - 1001 наблюдение.
  - 213 числовых признаков (107 типа float64, 106 типа int64).
- **Целевые переменные:**
  - IC50, mM (концентрация ингибирования).
  - CC50, mM (цитотоксическая концентрация).
  - SI (индекс селективности).

## 2. Пропущенные значения

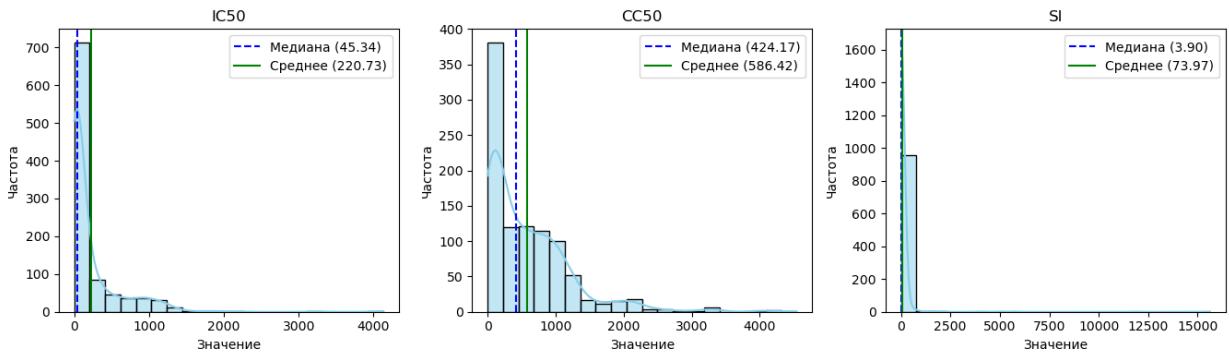
- **Обнаружены в 8 столбцах** (по 3 пропуска в каждом, 0.3% от данных):

## 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

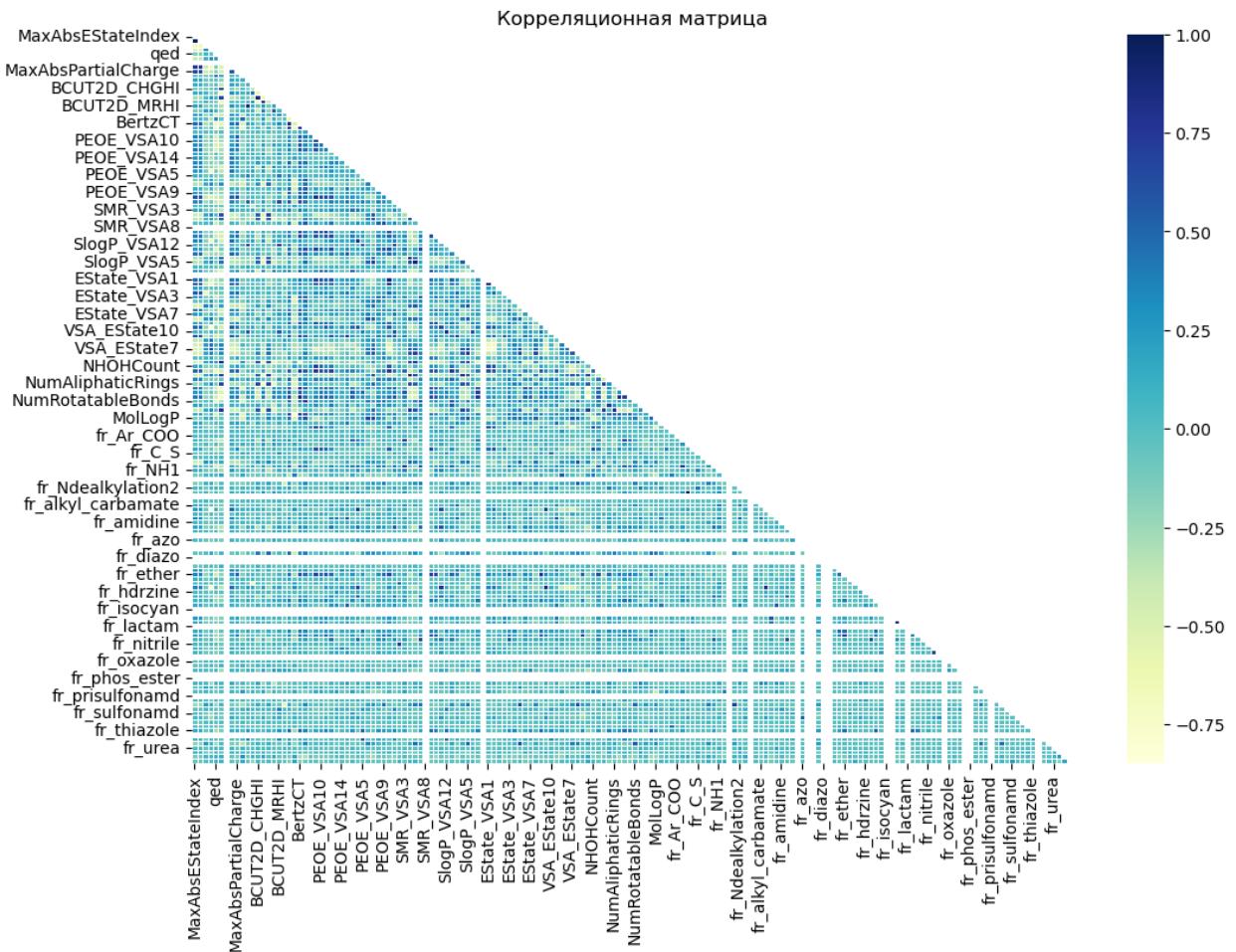
3. Распределение SI показывает:

- 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



- 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 неопределенности 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 (ребневая 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
- Параметры выборки
  - 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)]- архитектура сети
- max\_iter (500-1000) - эпохи обучения

### **SVR (метод опорных векторов):**

- C (0.1, 1, 10) - параметр регуляризации
- kernel ('rbf', 'linear')- тип ядра

### **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')- взвешивание соседей
- algorithm ('ball\_tree', 'kd\_tree', 'brute') - метод поиска соседей

## **Основные метрики**

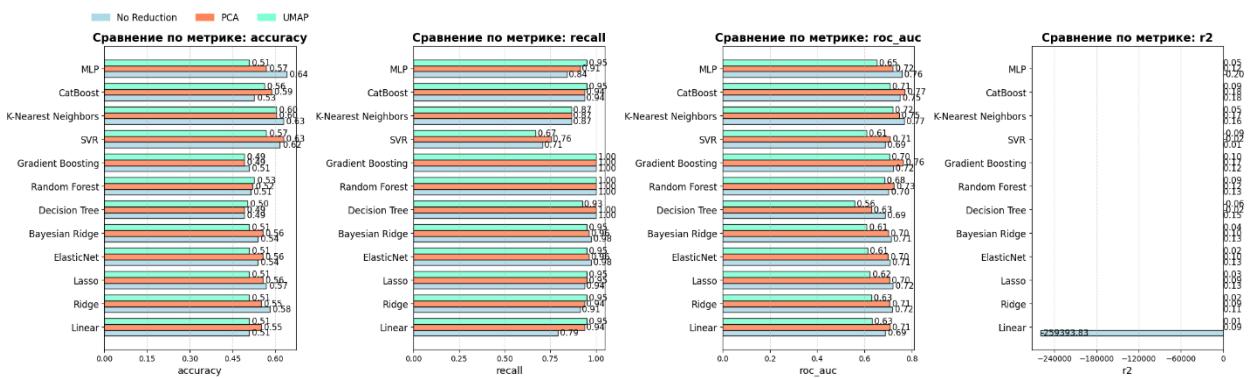
- Accuracy : доля правильно предсказанных классов.
- Recall (Полнота) : способность модели находить все положительные примеры.
- ROC AUC (Area Under Curve) : мера качества бинарной классификации.
- R<sup>2</sup> (Коэффициент детерминации) : объясняет, насколько хорошо модель воспроизводит данные.

## **Методы редуцирования признаков используемые в исследовании**

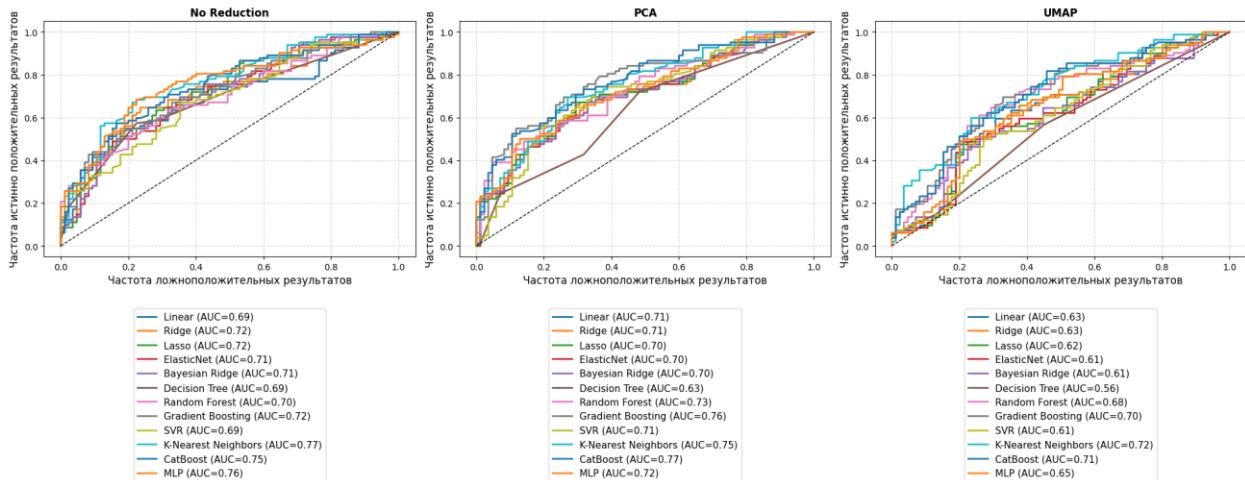
- No Reduction
- PCA
- UMAP

# Исследование моделей для решения задач регрессии IC50

Модель	Метод редукции	Лучшие параметры	Accuracy	Recall	ROC AUC	R2
Bayesian Ridge	No Reduction	'n': 0.0001, 'max_iter': 100}	0.5389221556886228	0.975609756097561	0.7116929698708753	0.13264240922512982
Bayesian Ridge	PCA	'n': 0.0001, 'max_iter': 100}	0.5568862275449101	0.9634146341463414	0.7005021520803444	0.09646951184937791
Bayesian Ridge	UMAP	'n': 0.0001, 'max_iter': 100}	0.5089820359281437	0.9512195121951219	0.6093256814921091	0.03811940635858124
CatBoost	No Reduction	'n': 4, 'learning_rate': 0.01}	0.5269461077844312	0.9390243902439024	0.7505738880918222	0.18303075863630502
CatBoost	PCA	'n': 6, 'learning_rate': 0.01}	0.5868263473053892	0.9390243902439024	0.770803443328551	0.17751565623476973
CatBoost	UMAP	'n': 4, 'learning_rate': 0.01}	0.562874251497006	0.9512195121951219	0.7071018651362984	0.09212090550640695
Decision Tree	No Reduction	{'max_depth': 3}	0.49101796407185627		1.0	0.6866571018651363
Decision Tree	PCA	{'max_depth': 3}	0.49101796407185627		1.0	0.6298421807747488
Decision Tree	UMAP	{'max_depth': 3}	0.5029940119760479	0.926829268292683	0.5576040172166428	-0.06415617582163446
ElasticNet	No Reduction	'max_iter': 1000, 'tol': 0.01}	0.5389221556886228	0.975609756097561	0.7071736011477762	0.13074235231370557
ElasticNet	PCA	'max_iter': 1000, 'tol': 0.01}	0.5568862275449101	0.9634146341463414	0.6996413199426111	0.09651952987571766
ElasticNet	UMAP	'max_iter': 1000, 'tol': 0.01}	0.5089820359281437	0.9512195121951219	0.6137733142037302	0.02190994045283068
Gradient Boosting	No Reduction	'rs': 200, 'subsample': 0.8}	0.5089820359281437		1.0	0.7217360114777618
Gradient Boosting	PCA	'rs': 100, 'subsample': 0.8}	0.49101796407185627		1.0	0.7642037302725969
Gradient Boosting	UMAP	'rs': 100, 'subsample': 0.9}	0.49101796407185627		1.0	0.7047345767575323
K-Nearest Neighbors	No Reduction	'p': 1, 'weights': 'uniform'}	0.6287425149700598	0.8658536585365854	0.7672883787661406	0.15538240517137947
K-Nearest Neighbors	PCA	'p': 2, 'weights': 'uniform'}	0.6047904191616766	0.8658536585365854	0.7476327116212338	0.17194993345043597
K-Nearest Neighbors	UMAP	'p': 2, 'weights': 'uniform'}	0.6047904191616766	0.8658536585365854	0.71987087517934	0.05262499620551775
Lasso	No Reduction	'ax_iter': 1000, 'tol': 0.001}	0.5688622754491018	0.9390243902439024	0.7207317073170731	0.13109566841710651
Lasso	PCA	'ax_iter': 1000, 'tol': 0.01}	0.5568862275449101	0.9512195121951219	0.7048063127690101	0.09297856403804106
Lasso	UMAP	'ax_iter': 1000, 'tol': 0.01}	0.5089820359281437	0.9512195121951219	0.6235294117647059	0.03277523861292342
Linear	No Reduction	default	0.5089820359281437	0.7926829268292683	0.6878048780487804	-259393.83007211224
Linear	PCA	default	0.5508982035928144	0.9390243902439024	0.7071018651362985	0.09140954749267916
Linear	UMAP	default	0.5089820359281437	0.9512195121951219	0.6321377331420374	0.013447161128861351
MLP	No Reduction	's': (100,), 'max_iter': 500}	0.6407185628742516	0.8414634146341463	0.7591104734576758	-0.1969871484091025
MLP	PCA	's': (100,), 'max_iter': 500}	0.5688622754491018	0.9146341463414634	0.7204447632711621	0.12111915522304462
MLP	UMAP	(100, 50), 'max_iter': 500}	0.5089820359281437	0.9512195121951219	0.6520803443328551	0.04528041023040896
Random Forest	No Reduction	'pth': 5, 'n_estimators': 50}	0.5149700598802395		1.0	0.7019368723098995
Random Forest	PCA	'th': 5, 'n_estimators': 100}	0.5209580838323353		1.0	0.7250358680057389
Random Forest	UMAP	'th': 5, 'n_estimators': 100}	0.5269461077844312		1.0	0.6840746054519369
Ridge	No Reduction	'pha': 10.0, 'solver': 'saga'}	0.5808383233532934	0.9146341463414634	0.7178622668579627	0.10719210618001884
Ridge	PCA	10.0, 'solver': 'sparse_cg'}	0.5508982035928144	0.9390243902439024	0.7065279770444763	0.09172897957789017
Ridge	UMAP	: 1.0, 'solver': 'sparse_cg'}	0.5089820359281437	0.9512195121951219	0.6294117647058824	0.015037313667567442
SVR	No Reduction	{'C': 1, 'kernel': 'linear'}	0.6167664670658682	0.7073170731707317	0.6865853658536584	0.006257693258119312
SVR	PCA	{'C': 10, 'kernel': 'linear'}	0.6287425149700598	0.7560975609756098	0.7089670014347202	-0.015774013496727024
SVR	UMAP	{'C': 10, 'kernel': 'linear'}	0.5688622754491018	0.6707317073170732	0.6081779053084649	-0.0912015660936254



Сравнение ROC-кривых для различных режимов



## 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

#### 2. Accuracy

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

#### 3. Recall

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

#### 4. R<sup>2</sup>

- 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

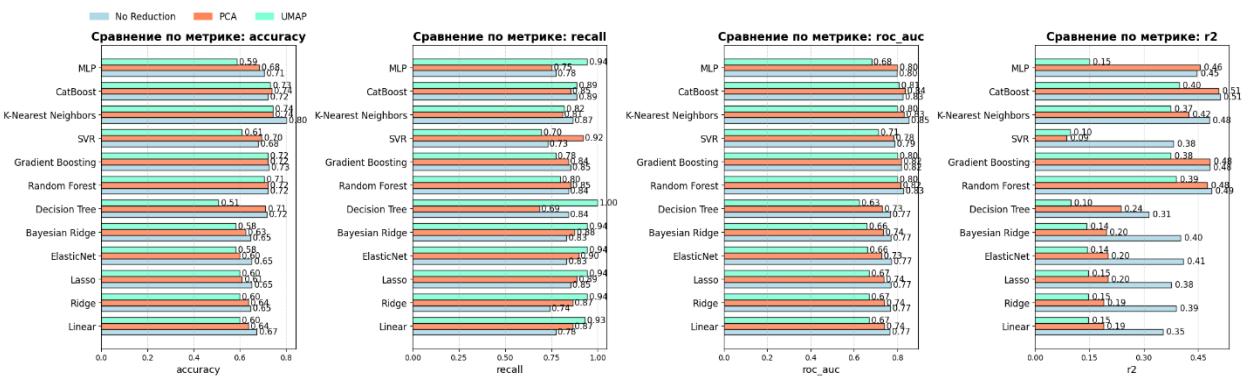
**Параметры:** depth: 6; learning\_rate: 0.01;

Обоснование:

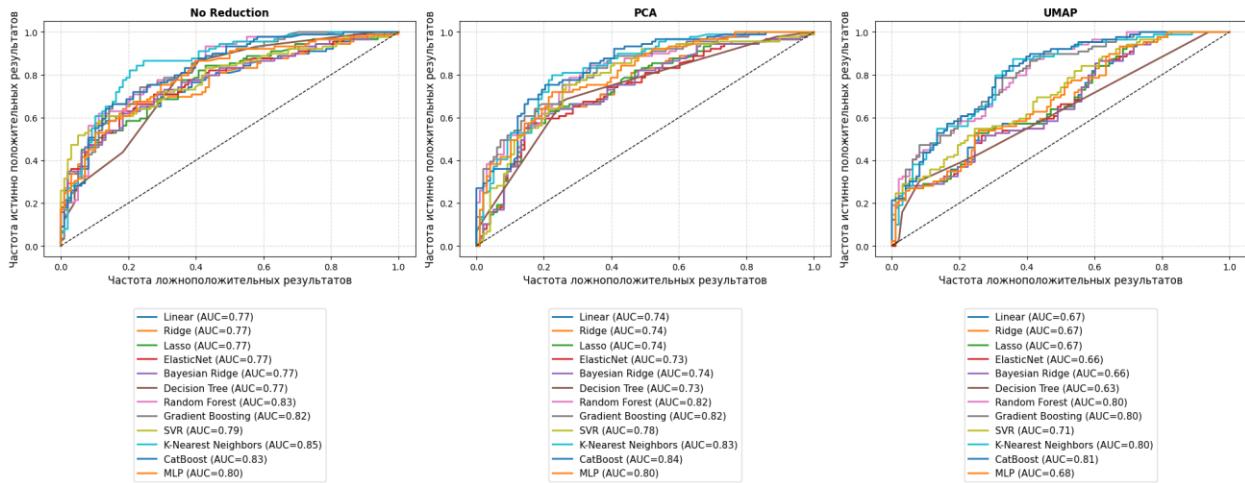
- Высокий ROC AUC = 0.7708 — лучший результат среди всех моделей.
- Recall = 1.0 — хорошо находит все положительные примеры.
- $R^2 = 0.1775$  — *стабильнее, чем у многих других* объяснение дисперсии данных
- Использование PCA снижает размерность признаков и помогает бороться с переобучением, сохраняя высокую эффективность.

# Исследование моделей для решения задач регрессии CC50

Модель	Метод редукции	Лучшие параметры	Accuracy	Recall	ROC AUC	R2
Bayesian Ridge	No Reduction	{'max_iter': 100}	0.6470588235294118	0.8314606741573034	0.7709240999770696	0.4019158419116301
Bayesian Ridge	PCA	{'max_iter': 100}	0.6256684491978609	0.8764044943820225	0.7364136665902317	0.1965243389184217
Bayesian Ridge	UMAP	{'max_iter': 100}	0.5828877005347594	0.9438202247191011	0.6600550332492547	0.1424842632169664
CatBoost	No Reduction	{'learning_rate': 0.01}	0.7219251336898396	0.8876404494382022	0.8253840862187571	0.5108075526691062
CatBoost	PCA	{'learning_rate': 0.01}	0.7379679144385026	0.8539325842696629	0.8371933042880073	0.507647254206928
CatBoost	UMAP	{'learning_rate': 0.01}	0.732620320855615	0.8876404494382022	0.8085301536344875	0.3981808019730818
Decision Tree	No Reduction	{'max_depth': 5}	0.7165775401069518	0.8426966292134831	0.7680004586104104	0.3143411133910718
Decision Tree	PCA	{'max_depth': 3}	0.7112299465240641	0.6853932584269663	0.731368952075212	0.23710823743362142
Decision Tree	UMAP	{'max_depth': 3}	0.5080213903743316	1.0	0.6252579683558817	0.09831717299181075
ElasticNet	No Reduction	{'max_iter': 1000, 'tol': 0.01}	0.6524064171122995	0.8314606741573034	0.7729878468241229	0.40979227197100043
ElasticNet	PCA	{'max_iter': 1000, 'tol': 0.01}	0.5989304812834224	0.898876404494382	0.7274707635863334	0.20101874423073873
ElasticNet	UMAP	{'max_iter': 1000, 'tol': 0.01}	0.5828877005347594	0.9438202247191011	0.6626920431093786	0.14381556317552102
Gradient Boosting	No Reduction	{'n_estimators': 50, 'subsample': 0.8}	0.7272727272727273	0.8539325842696629	0.8188488878697546	0.4824099950179084
Gradient Boosting	PCA	{'n_estimators': 200, 'subsample': 0.8}	0.7219251336898396	0.8426966292134831	0.8196514560880532	0.4829583342351048
Gradient Boosting	UMAP	{'n_estimators': 200, 'subsample': 0.9}	0.7219251336898396	0.7752808988764045	0.8001605136436597	0.37516530415437976
K-Nearest Neighbors	No Reduction	{'p': 1, 'weights': 'distance'}	0.8021390374331551	0.8651685393258427	0.8534739738592066	0.4811011653304029
K-Nearest Neighbors	PCA	{'p': 2, 'weights': 'distance'}	0.7433155080213903	0.8089887640449438	0.8305434533363907	0.4245345595594166
K-Nearest Neighbors	UMAP	{'p': 1, 'weights': 'distance'}	0.7433155080213903	0.8202247191011236	0.7998165558358175	0.3746207993380365
Lasso	No Reduction	{'max_iter': 1000, 'tol': 0.01}	0.6524064171122995	0.8539325842696629	0.7714973629901399	0.37593710090128674
Lasso	PCA	{'max_iter': 1000, 'tol': 0.01}	0.6096256684491979	0.8876404494382022	0.738362760834671	0.20082974044618895
Lasso	UMAP	{'max_iter': 1000, 'tol': 0.01}	0.5989304812834224	0.9438202247191011	0.6692272414583811	0.14699525088373555
Linear	No Reduction	default	0.6737967914438503	0.7752808988764045	0.7665673010777344	0.3528113720419671
Linear	PCA	default	0.6363636363636364	0.8651685393258427	0.7407704654895666	0.18873088594417087
Linear	UMAP	default	0.5989304812834224	0.9325842696629213	0.6710616831002063	0.14718100453078486
MLP	No Reduction	{'hidden_layer_sizes': (100,), 'max_iter': 500}	0.7058823529411765	0.7752808988764045	0.7976381563861499	0.4472648749661433
MLP	PCA	{'hidden_layer_sizes': (100, 50), 'max_iter': 500}	0.6844919786096256	0.7528089887640449	0.8017656500802568	0.4556477481359231
MLP	UMAP	{'hidden_layer_sizes': (100, 50), 'max_iter': 1000}	0.5882352941176471	0.9438202247191011	0.6813804173354735	0.1498171535590076
Random Forest	No Reduction	{'n_estimators': 50}	0.7219251336898396	0.8426966292134831	0.8282504012841091	0.4880608886595433
Random Forest	PCA	{'n_estimators': 100}	0.7219251336898396	0.8539325842696629	0.8154093097913322	0.47592258026006895
Random Forest	UMAP	{'n_estimators': 100}	0.7058823529411765	0.797752808988764	0.8010777344645723	0.38992104504042965
Ridge	No Reduction	{'alpha': 10.0, 'solver': 'saga'}	0.6470588235294118	0.7415730337078652	0.7686310479247879	0.38938210926279404
Ridge	PCA	{'alpha': 10.0, 'solver': 'sag'}	0.6363636363636364	0.8651685393258427	0.7406558128869525	0.18939340737837185
Ridge	UMAP	{'alpha': 10.0, 'solver': 'lsqr'}	0.5989304812834224	0.9438202247191011	0.6707177252923641	0.14735634741531856
SVR	No Reduction	{'C': 10, 'kernel': 'linear'}	0.679144385026738	0.7303370786516854	0.7872047695482688	0.38220080888243035
SVR	PCA	{'C': 10, 'kernel': 'linear'}	0.6951871657754011	0.9213483146067416	0.782733180463196	0.08733675859439616
SVR	UMAP	{'C': 1, 'kernel': 'linear'}	0.6096256684491979	0.6966292134831461	0.7118780096308187	0.09726441350124115



### Сравнение ROC-кривых для различных режимов



## 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. R<sup>2</sup>

- 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;

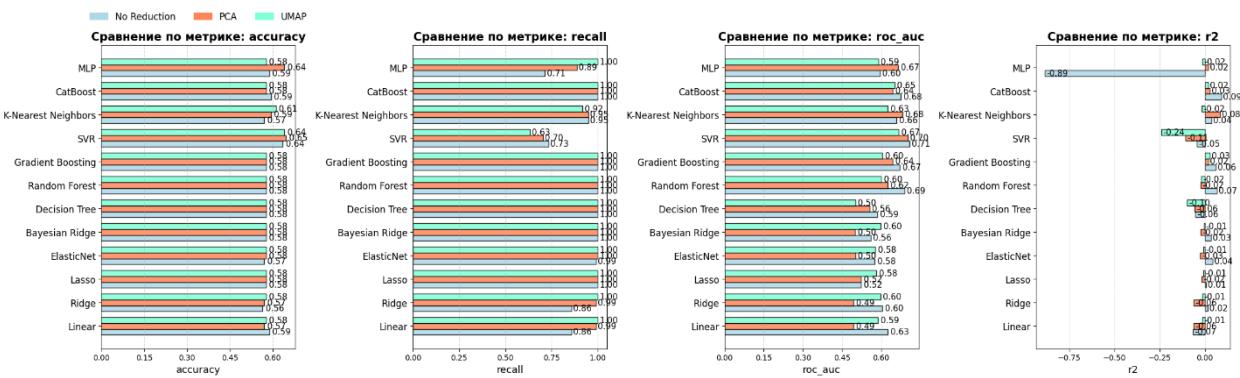
Обоснование:

- ROC AUC = 0.825
- Recall = 0.888
- $R^2$  = 0.511
- Accuracy = 0.722

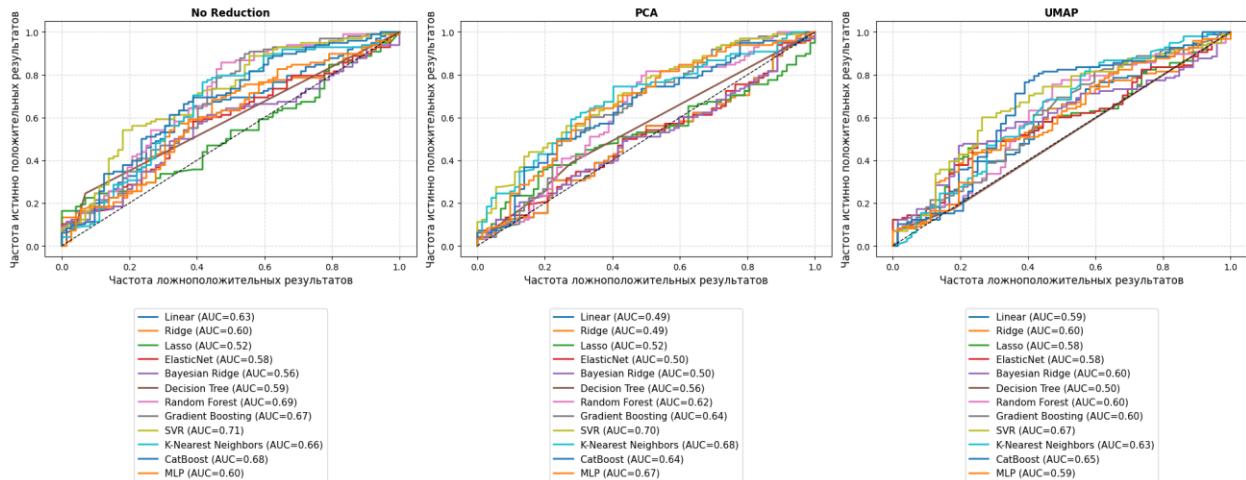
Модель демонстрирует наивысшую объясняющую способность ( $R^2$ ) , хорошие значения Recall и ROC AUC , а также не требует дополнительных преобразований признаков.

# Исследование моделей для решения задач регрессии SI

Модель	Метод редукции	Лучшие параметры	Accuracy	Recall	ROC AUC	R2
Bayesian Ridge	No Reduction	{'max_iter': 100}	0.5764705882352941	1.0	0.5600198412698413	0.033637735189812945
Bayesian Ridge	PCA	{'max_iter': 100}	0.5764705882352941	1.0	0.5007794784580499	-0.024120291183947984
Bayesian Ridge	UMAP	{'max_iter': 100}	0.5764705882352941	1.0	0.5980725623582767	-0.005180727657822848
CatBoost	No Reduction	{'learning_rate': 0.01}	0.5941176470588235	1.0	0.6759495464852607	0.09068968313304548
CatBoost	PCA	{'learning_rate': 0.01}	0.5764705882352941	1.0	0.6436366213151928	0.0261604721534483
CatBoost	UMAP	{'learning_rate': 0.01}	0.5764705882352941	1.0	0.6515022675736962	0.017993321123503825
Decision Tree	No Reduction	{'max_depth': 3}	0.5764705882352941	1.0	0.5858843537414967	-0.05539244896283435
Decision Tree	PCA	{'max_depth': 3}	0.5764705882352941	1.0	0.5555555555555556	-0.05799334346264562
Decision Tree	UMAP	{'max_depth': 3}	0.5764705882352941	1.0	0.5013463718820861	-0.09855266942386876
ElasticNet	No Reduction	{'max_iter': 1000, 'tol': 0.001}	0.5705882352941176	0.9897959183673469	0.5750425170068028	0.04200142917749472
ElasticNet	PCA	{'max_iter': 1000, 'tol': 0.0001}	0.5764705882352941	1.0	0.5013463718820862	-0.02814673847744504
ElasticNet	UMAP	{'max_iter': 1000, 'tol': 0.01}	0.5764705882352941	1.0	0.5783730158730158	-0.007290126032342359
Gradient Boosting	No Reduction	{'n_estimators': 100, 'subsample': 0.8}	0.5764705882352941	1.0	0.671981295170069	0.06104916738855992
Gradient Boosting	PCA	{'n_estimators': 50, 'subsample': 0.8}	0.5764705882352941	1.0	0.644203514739229	0.017059379652651585
Gradient Boosting	UMAP	{'n_estimators': 50, 'subsample': 0.9}	0.5764705882352941	1.0	0.6043083900226758	0.02757390704911311
K-Nearest Neighbors	No Reduction	{'p': 2, 'weights': 'uniform'}	0.5705882352941176	0.9489795918367347	0.6597222222222222	0.035313668484219685
K-Nearest Neighbors	PCA	{'p': 1, 'weights': 'uniform'}	0.5941176470588235	0.9489795918367347	0.6820436507936508	0.08361974348468593
K-Nearest Neighbors	UMAP	{'p': 2, 'weights': 'uniform'}	0.611764705882353	0.9183673469387755	0.6254960317460317	-0.01679781503629596
Lasso	No Reduction	{'max_iter': 1000, 'tol': 0.01}	0.5764705882352941	1.0	0.5227465986394558	0.006047098128565409
Lasso	PCA	{'max_iter': 1000, 'tol': 0.01}	0.5764705882352941	1.0	0.523030045351474	-0.017282289777845294
Lasso	UMAP	{'max_iter': 1000, 'tol': 0.01}	0.5764705882352941	1.0	0.5816326530612245	-0.00824589022355493
Linear	No Reduction	default	0.5882352941176471	0.8571428571428571	0.626204648526077	-0.06569176325343462
Linear	PCA	default	0.5705882352941176	0.9897959183673469	0.4942602040816327	-0.06296574457063508
Linear	UMAP	default	0.5764705882352941	1.0	0.5874433106575964	-0.013549076658365466
MLP	No Reduction	{'hidden_layer_sizes': (100,), 'max_iter': 500}	0.5882352941176471	0.7142857142857143	0.5954506802721089	-0.8851119148026632
MLP	PCA	{'hidden_layer_sizes': (100,), 'max_iter': 500}	0.6411764705882353	0.8877551020408163	0.6660289115646257	0.017512444747463385
MLP	UMAP	{'hidden_layer_sizes': (100, 50), 'max_iter': 500}	0.5764705882352941	1.0	0.5890022675736961	-0.01574252152194755
Random Forest	No Reduction	{'n_estimators': 100}	0.5764705882352941	1.0	0.6894132653061225	0.06713159814331005
Random Forest	PCA	{'n_estimators': 50}	0.5764705882352941	1.0	0.6247874149659864	-0.02388671636594686
Random Forest	UMAP	{'n_estimators': 100}	0.5764705882352941	1.0	0.6001275510204083	-0.02205433938088719
Ridge	No Reduction	{'alpha': 10.0, 'solver': 'saga'}	0.5647058823529412	0.8571428571428571	0.6049461451247166	0.01502874891311401
Ridge	PCA	{'alpha': 10.0, 'solver': 'sag'}	0.5705882352941176	0.9897959183673469	0.4941184807256236	-0.06221633090769885
Ridge	UMAP	{'alpha': 10.0, 'solver': 'sparse_cg'}	0.5764705882352941	1.0	0.5975056689342404	-0.01349107175653722
SVR	No Reduction	{'C': 10, 'kernel': 'rbf'}	0.6352941176470588	0.7346938775510204	0.7081207482993197	-0.04513979539301549
SVR	PCA	{'C': 10, 'kernel': 'rbf'}	0.6470588235294118	0.7040816326530612	0.7018849206349206	-0.10866627522012329
SVR	UMAP	{'C': 10, 'kernel': 'rbf'}	0.6411764705882353	0.6326530612244898	0.6678004535147393	-0.24194786740147345



### Сравнение ROC-кривых для различных режимов



## 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 модели кроме SVR показали максимальное значение полноты, равное или приближающееся к 1.0.

### 4. R<sup>2</sup>

- 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
- R<sup>2</sup> = 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 (R<sup>2</sup>). 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**

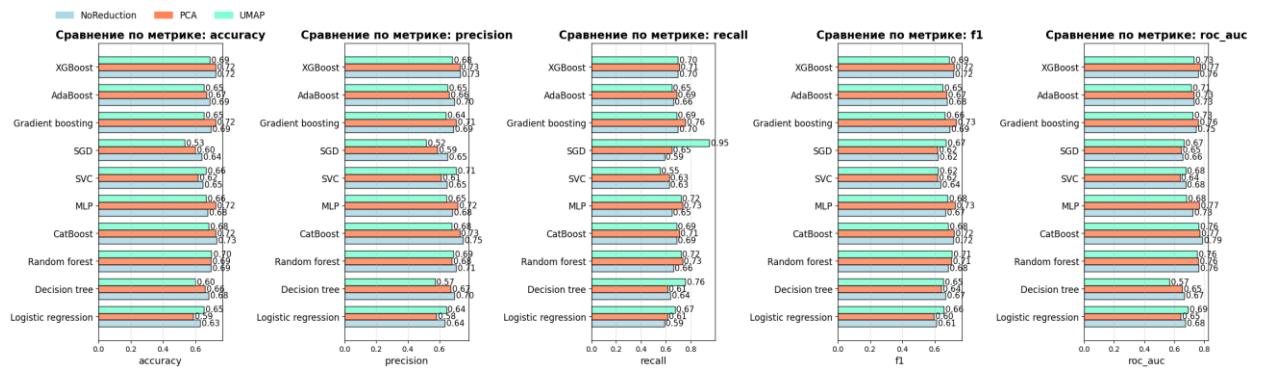
- Accuracy : доля правильно предсказанных классов.
- Recall (Полнота) : способность модели находить все положительные примеры.
- ROC AUC (Area Under Curve) : мера качества бинарной классификации.
- R<sup>2</sup> (Коэффициент детерминации) : объясняет, насколько хорошо модель воспроизводит данные.

## **Методы редуцирования признаков используемые в исследовании**

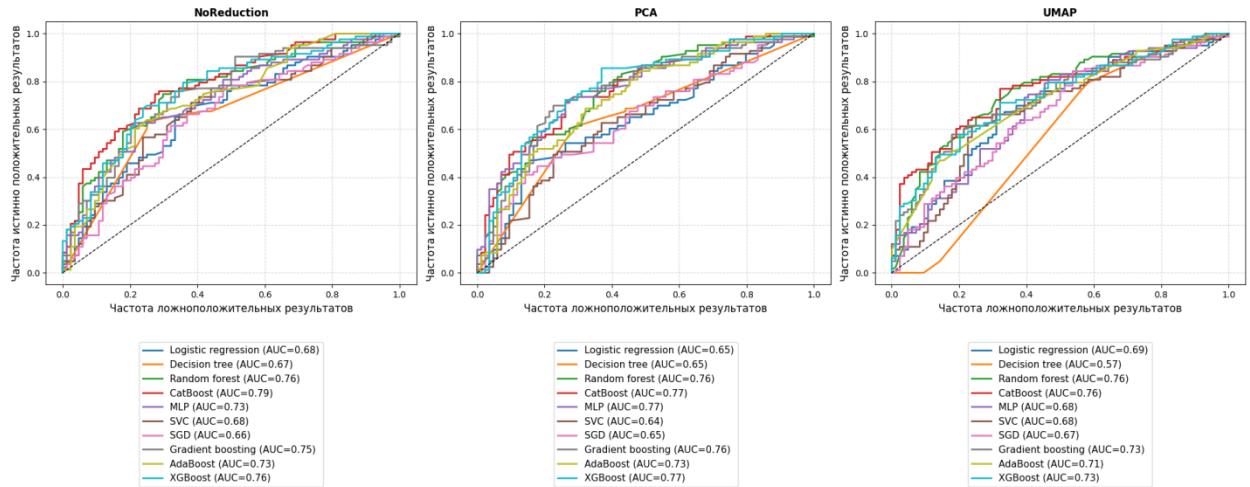
- No Reduction
- PCA
- UMAP

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

Модель	Метод редукции	Лучшие параметры	Accuracy	Precision	Recall	F1	ROC AU
AdaBoost	NoReduction	`n_estimators': 200}	0.688622754491018	0.6962025316455697	0.6626506024096386	0.6790123456790124	0.73135398737808
AdaBoost	PCA	`n_estimators': 200}	0.6706586826347305	0.6627906976744186	0.6867469879518072	0.6745562130177515	0.73199942627653
AdaBoost	UMAP	`n_estimators': 100}	0.6526946107784432	0.6506024096385542	0.6506024096385542	0.6506024096385542	0.71493115318416
CatBoost	NoReduction	100, 'learning_rate': 0.01}	0.7305389221556886	0.75	0.6867469879518072	0.7169811320754716	0.79001721170395
CatBoost	PCA	500, 'learning_rate': 0.01}	0.7245508982035929	0.7283950617283951	0.7108433734939759	0.7195121951219512	0.7732358003442
CatBoost	UMAP	500, 'learning_rate': 0.01}	0.6826347305389222	0.6785714285714286	0.6867469879518072	0.6826347305389222	0.76434308663224
Decision tree	NoReduction	copy, 'max_depth': None}	0.6826347305389222	0.6973684210526315	0.6385542168674698	0.6666666666666666	0.66896156052782
Decision tree	PCA	'gini', 'max_depth': None}	0.65868262347305389	0.6710526315789473	0.6144578313253012	0.6415094339622641	0.65397303499713
Decision tree	UMAP	'entropy', 'max_depth': 5}	0.5988023952095808	0.5727272727272728	0.7590361445783133	0.6528497409326425	0.56805794606999
Gradient boosting	NoReduction	rs': 50, 'subsample': 0.9}	0.6946107784431138	0.6904761904761905	0.6987951807228916	0.6946107784431138	0.74727481353987
Gradient boosting	PCA	rs': 100, 'subsample': 0.8}	0.7245508982035929	0.7078651685393258	0.7590361445783133	0.7325581395348837	0.76147446930579
Gradient boosting	UMAP	rs': 200, 'subsample': 0.8}	0.6526946107784432	0.6404494382022472	0.6867469879518072	0.6627906976744186	0.72726620768789
Logistic regression	NoReduction	{'C': 1, 'solver': 'lbfgs'}	0.6287425149700598	0.6363636363636364	0.5903614457831325	0.6125	0.67692197360872
Logistic regression	PCA	{'C': 0.1, 'solver': 'lbfgs'}	0.5868263473053892	0.5795454545454546	0.6144578313253012	0.5964912280701754	0.64543889845094
Logistic regression	UMAP	{'C': 10, 'solver': 'lbfgs'}	0.6526946107784432	0.6436781609195402	0.6746987951807228	0.6588235294117647	0.69277108433734
MLP	NoReduction	is': (100,), 'max_iter': 500}	0.6766467065868264	0.6835443037974683	0.6506024096385542	0.6666666666666666	0.72654905335628
MLP	PCA	is': (100,), 'max_iter': 500}	0.7245508982035929	0.7176470588235294	0.7349397590361446	0.7261904761904762	0.76900458978772
MLP	UMAP	is': (100,), 'max_iter': 500}	0.6646706586826348	0.6451612903225806	0.7228915662650602	0.6818181818181818	0.68488238668961
Random forest	NoReduction	log2', 'n_estimators': 100}	0.6946107784431138	0.70512805128052	0.6626506024096386	0.6832298136645962	0.76441480206540
Random forest	PCA	'sqrt', 'n_estimators': 50}	0.6946107784431138	0.6777777777777778	0.7349397590361446	0.7052023121387283	0.76290877796901
Random forest	UMAP	'sqrt', 'n_estimators': 100}	0.7005988023952096	0.6896551724137931	0.7228915662650602	0.7058823529411765	0.75537865748709
SGD	NoReduction	000, 'penalty': 'elasticnet'}	0.6407185628742516	0.6533333333333333	0.5903614457831325	0.620253164556962	0.65899311531841
SGD	PCA	000, 'penalty': 'elasticnet'}	0.5988023952095808	0.5869565217391305	0.6506024096385542	0.6171428571428571	0.64859437751004
SGD	UMAP	iter: 1000, 'penalty': 'l1'	0.5329341317365269	0.5163398692810458	0.9518072289156626	0.6694915254237288	0.66537578886976
SVC	NoReduction	{'C': 1, 'kernel': 'linear'}	0.6467065868263473	0.65	0.6265060240963856	0.638036808159509	0.67799770510613
SVC	PCA	{'C': 0.1, 'kernel': 'linear'}	0.6167664670658682	0.6117647058823523	0.6265060240963856	0.6190476190476191	0.63927137119908
SVC	UMAP	{'C': 0.1, 'kernel': 'linear'}	0.6646706586826348	0.7076923076923077	0.5542168674698795	0.6216216216216216	0.67828456683878
XGBoost	NoReduction	rs': 200, 'subsample': 0.8}	0.7245508982035929	0.7341772151898734	0.6987951807228916	0.7160493827160493	0.76319563970166
XGBoost	PCA	rs': 200, 'subsample': 1.0}	0.7245508982035929	0.7283950617283951	0.7108433734939759	0.7195121951219512	0.77467010900745
XGBoost	UMAP	rs': 200, 'subsample': 0.8}	0.688622754491018	0.6823529411764706	0.6987951807228916	0.6904761904761905	0.73228628800917



### Сравнение ROC-кривых для различных режимов



## 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 R<sup>2</sup>

### 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

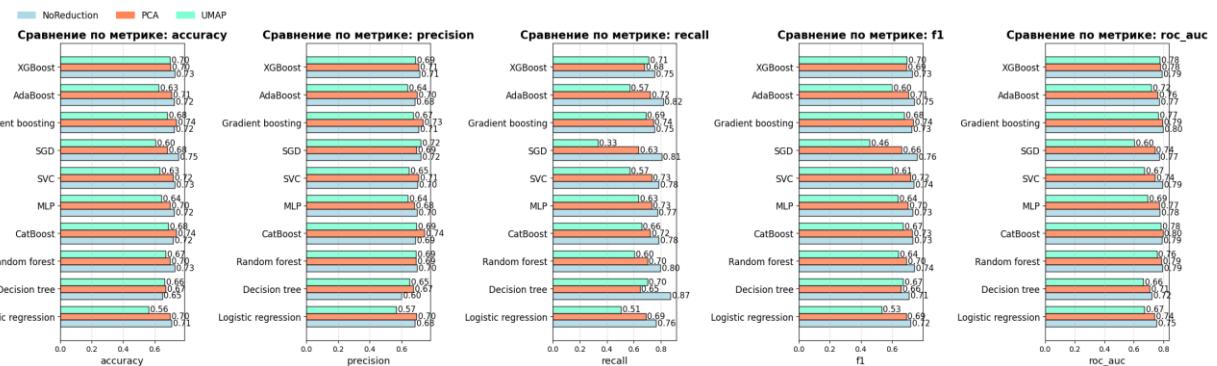
**Параметры:** depth: 8; iterations: 100; learning\_rate: 0.01;

Обоснование:

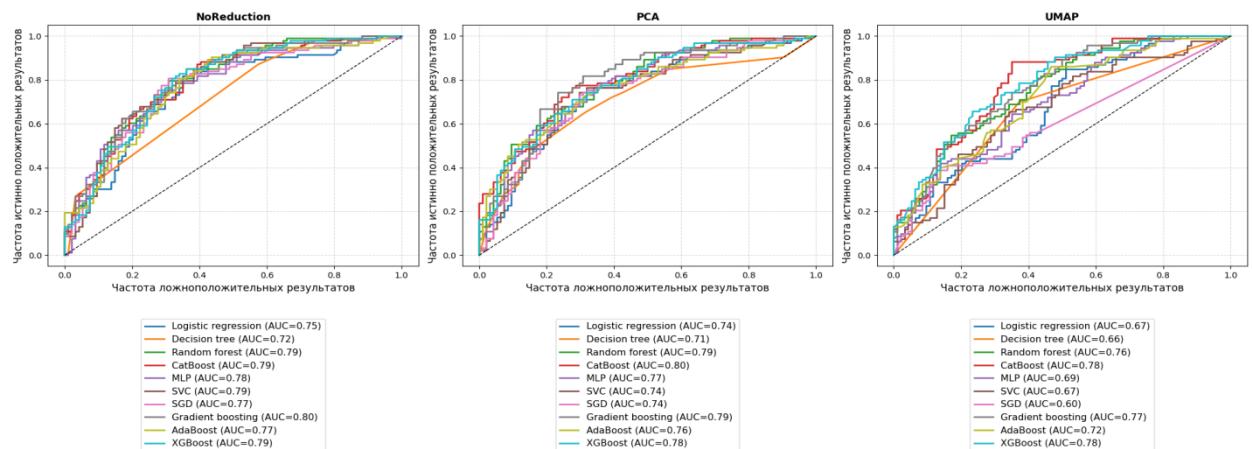
- Наивысший ROC AUC (0.7900) — говорит о лучшем разделении классов.
- Высокая точность (Precision = 0.75) — минимальное количество ложноположительных срабатываний.
- Хороший F1-score (0.717) — умеренный баланс между полнотой и точностью.
- Без метода редукции — сохраняет все исходные признаки, что может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

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

Модель	Метод редукции	Лучшие параметры	Accuracy	Precision	Recall	F1	ROC AU
AdaBoost	NoReduction	{'n_estimators': 200}	0.7219251336898396	0.6846846846846847	0.8172043010752689	0.7450980392156863	0.77282086479066
AdaBoost	PCA	{'n_estimators': 200}	0.7058823529411765	0.6979166666666666	0.720430107268817	0.708994708994709	0.76281171356668
AdaBoost	UMAP	{'n_estimators': 200}	0.6256684491978609	0.638554216874698	0.5698924731182796	0.6022727272727273	0.71825669183253
CatBoost	NoReduction	{'learning_rate': 0.01}	0.7165775401069518	0.6886792452830188	0.7849462365591398	0.7336683417085427	0.791499882849233
CatBoost	PCA	{'learning_rate': 0.01}	0.7379679144385026	0.7444444444444445	0.7204301075268817	0.73224043715847	0.79564172958133
CatBoost	UMAP	{'learning_rate': 0.01}	0.6844919786096256	0.6931818181818182	0.6559139784946236	0.6740331491712708	0.783668794326241
Decision tree	NoReduction	{'entropy', 'max_depth': 3}	0.6470588235294118	0.6	0.8709677419354839	0.7105263157894737	0.72151681537405
Decision tree	PCA	{'gini', 'max_depth': 5}	0.6684491978609626	0.6741573033707865	0.6451612903225806	0.65993406593406593	0.70824754060855
Decision tree	UMAP	{'gini', 'max_depth': None}	0.6631016042780749	0.65	0.6989247311827957	0.6735751295336787	0.66329215282544
Gradient boosting	NoReduction	{'n_estimators': 50, 'subsample': 1.0}	0.7219251336898396	0.7070707070707071	0.7526881720430108	0.7291666666666666	0.79752916952642
Gradient boosting	PCA	{'n_estimators': 50, 'subsample': 0.9}	0.7379679144385026	0.734025531914894	0.7419354838709677	0.7379679144385026	0.79381148478609
Gradient boosting	UMAP	{'n_estimators': 50, 'subsample': 0.9}	0.679144385026738	0.6736842105263158	0.6881720430107527	0.6808510638297872	0.76578586135895
Logistic regression	NoReduction	{'C': 0.1, 'solver': 'lbfgs'}	0.7058823529411765	0.6826923076923077	0.7634408602150538	0.7208121827411168	0.75354609929078
Logistic regression	PCA	{'C': 0.1, 'solver': 'lbfgs'}	0.6951871657754011	0.6956521739130435	0.6881720430107527	0.6918918918918919	0.73878975062914
Logistic regression	UMAP	{'C': 10, 'solver': 'lbfgs'}	0.5614973262032086	0.5662650602409639	0.5053763440860215	0.53409090909090901	0.67009837565774
MLP	NoReduction	{'activation': ('relu',), 'max_iter': 500}	0.7219251336898396	0.6990291262135923	0.7741935483870968	0.7346938775510204	0.7764215923129
MLP	PCA	{'activation': ('relu',), 'max_iter': 500}	0.6951871657754011	0.68	0.7311827956989247	0.7046632124352331	0.77093342484557
MLP	UMAP	{'activation': ('relu',), 'max_iter': 500}	0.6417112299465241	0.641304378260869	0.6344086021505376	0.6378378378378379	0.6933196064973
Random forest	NoReduction	{'criterion': 'sqrt', 'n_estimators': 100}	0.7272727272727273	0.6981132075471698	0.7956989247311828	0.743718592648241	0.7932587508579
Random forest	PCA	{'criterion': 'gini', 'n_estimators': 100}	0.6951871657754011	0.6914893617021277	0.6989247311827957	0.6951871657754011	0.78637611530542
Random forest	UMAP	{'criterion': 'gini', 'n_estimators': 100}	0.6684491978609626	0.691358024691358	0.6021505376344086	0.6436781609195402	0.75611988103408
SGD	NoReduction	{'alpha': 1000, 'penalty': 'l1'}	0.7486631016042781	0.7211538461538461	0.8064516129032258	0.7614213187969543	0.77059025394646
SGD	PCA	{'alpha': 1000, 'penalty': 'l2'}	0.679144385026738	0.6941176470588235	0.6344086021505376	0.6629213483146067	0.74199276902081
SGD	UMAP	{'alpha': 1000, 'penalty': 'elasticnet'}	0.6042780748663101	0.7209302325581395	0.3333333333333333	0.45588235294117646	0.60340883093113
SVC	NoReduction	{'C': 1, 'kernel': 'rbf'}	0.7272727272727273	0.7019230769230769	0.7849462365591398	0.7411167512690355	0.7935627041866
SVC	PCA	{'C': 0.1, 'kernel': 'linear'}	0.7165775401069518	0.7083333333333334	0.7311827956989247	0.7195767195767195	0.74302219171814
SVC	UMAP	{'C': 10, 'kernel': 'rbf'}	0.6310160427807486	0.6463414634146342	0.5698924731182796	0.6057142857142858	0.67147105925417
XGBoost	NoReduction	{'n_estimators': 50, 'subsample': 0.9}	0.7272727272727273	0.7142857142857143	0.7526881720430108	0.7329842931937173	0.7894646533973
XGBoost	PCA	{'n_estimators': 50, 'subsample': 0.8}	0.7005347593582888	0.7078651685393258	0.6774193548387096	0.6923076923076923	0.77974147792267
XGBoost	UMAP	{'n_estimators': 200, 'subsample': 1.0}	0.6951871657754011	0.6875	0.7096774193548387	0.6984126984126984	0.77625257378174



Сравнение ROC-кривых для различных режимов



## **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

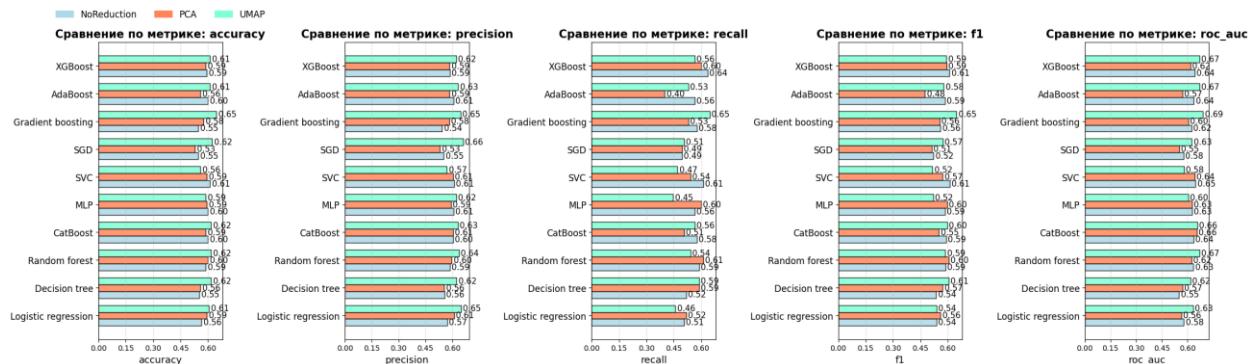
**Параметры:** depth = 6; iterations = 100; learning\_rate = 0.1;

### Обоснование:

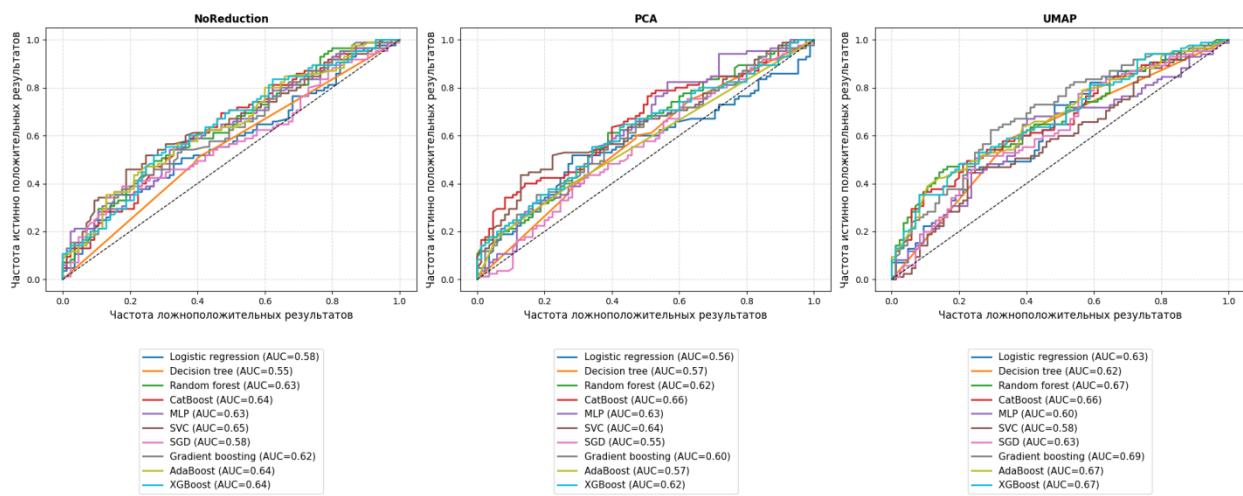
- ROC AUC: 0.7956 (лучший)
- Precision: 0.7444 (лучший)
- Accuracy: 0.7380 (высокий)
- F1-score: 0.7322 (хороший)
- Recall: 0.7204 (умеренный)

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

Модель	Метод редукции	Лучшие параметры	Accuracy	Precision	Recall	F1	ROC AU
AdaBoost	NoReduction	`n_estimators': 200}	0.6	0.6075949367088608	0.5647058823529412	0.5853658536585366	0.63833910034602
AdaBoost	PCA	`n_estimators': 50}	0.5588235294117647	0.5862068965517241	0.4	0.4755244755244755	0.57197231833910
AdaBoost	UMAP	`n_estimators': 200}	0.611764705882353	0.6338028169014085	0.5294117647058824	0.5769230769230769	0.67107266435986
CalBoost	NoReduction	'learning_rate': 0.1}	0.6	0.6049382716049383	0.5764705882352941	0.5903614457831325	0.63619377162629
CalBoost	PCA	100, 'learning_rate': 0.01}	0.5882352941176471	0.6056338028169014	0.505882352941176471	0.5512820512820513	0.65557093425605
CalBoost	UMAP	100, 'learning_rate': 0.1}	0.6176470588235294	0.631578947368421	0.5647058823529412	0.5962732919254659	0.65944636678200
Decision tree	NoReduction	'gini', 'max_depth': None}	0.5529411764705883	0.5569620253164557	0.5176470588235295	0.5365853658536586	0.55086505190311
Decision tree	PCA	'entropy', 'max_depth': 10}	0.5588235294117647	0.5555555555555556	0.5882352941176471	0.5714285714285714	0.5721107266435986
Decision tree	UMAP	'gini', 'max_depth': None}	0.6176470588235294	0.625	0.5882352941176471	0.6060606060606061	0.6176470588235295
Gradient boosting	NoReduction	'rs': 200, 'subsample': 0.9}	0.5470588235294118	0.5444444444444444	0.5764705882352941	0.56	0.623183391003460
Gradient boosting	PCA	'rs': 50, 'subsample': 0.9}	0.5764705882352941	0.5844155844155844	0.5294117647058824	0.5555555555555556	0.60470588235294
Gradient boosting	UMAP	'rs': 100, 'subsample': 1.0}	0.6470588235294118	0.6470588235294118	0.6470588235294118	0.6470588235294118	0.69093425605536
Logistic regression	NoReduction	'C': 0.1, 'solver': 'lbfgs'}	0.5647058823529412	0.5733333333333334	0.5058823529411764	0.5375	0.57868512110726
Logistic regression	PCA	'C': 0.1, 'solver': 'lbfgs'}	0.5941176470588235	0.6111111111111112	0.5176470588235295	0.5605095541401274	0.56435986159169
Logistic regression	UMAP	'C': 0.1, 'solver': 'lbfgs'}	0.6058823529411764	0.65	0.4588235294117647	0.5379310344827586	0.63224913494809
MLP	NoReduction	(100, 50), 'max_iter': 500}	0.6	0.6075949367088608	0.5647058823529412	0.5853658536585366	0.62782006920415
MLP	PCA	's': (100,), 'max_iter': 500}	0.5941176470588235	0.5930232558139535	0.6	0.5964912280701754	0.62768166089965
MLP	UMAP	(100, 50), 'max_iter': 500}	0.5882352941176471	0.6229508196721312	0.4470588235294118	0.5205479452054794	0.60484429065743
Random forest	NoReduction	None, 'n_estimators': 100}	0.5882352941176471	0.5882352941176471	0.5882352941176471	0.5882352941176471	0.63480968858131
Random forest	PCA	'sqrt', 'n_estimators': 50}	0.6	0.5977011494252874	0.611764705882353	0.6046511627906976	0.623183391003460
Random forest	UMAP	'sqrt', 'n_estimators': 100}	0.6176470588235294	0.6388888888888888	0.5411764705882353	0.5859872611464968	0.67031141868512
SGD	NoReduction	'l1': 1000, 'penalty': 'l1'}	0.5470588235294118	0.5526315789473685	0.4941176470588235	0.5217391304347826	0.57743944636678
SGD	PCA	'l1': 1000, 'penalty': 'l1'}	0.5294117647058824	0.5316455696202531	0.4941176470588235	0.5121951219512195	0.55273356401384
SGD	UMAP	'l1': 1000, 'penalty': 'l2'	0.6235294117647059	0.6615384615384615	0.5058823529411764	0.5733333333333334	0.625605536332
SVC	NoReduction	{'C': 10, 'kernel': 'rbf'}	0.611764705882353	0.611764705882353	0.611764705882353	0.611764705882353	0.64768166089965
SVC	PCA	{'C': 1, 'kernel': 'rbf'}	0.5941176470588235	0.6056231578947368	0.5411764705882353	0.5714285714285714	0.642560553632
SVC	UMAP	{'C': 1, 'kernel': 'rbf'}	0.5588235294117647	0.5714285714285714	0.47058823529411764	0.5161290322580645	0.58159169550173
XGBoost	NoReduction	'rs': 50, 'subsample': 0.8}	0.5941176470588235	0.5869565217391305	0.6352941176470588	0.6101694915254238	0.64200692041522
XGBoost	PCA	'rs': 200, 'subsample': 0.8}	0.5882352941176471	0.5862068965517241	0.6	0.5930232558139535	0.61695501730103
XGBoost	UMAP	'rs': 50, 'subsample': 1.0}	0.611764705882353	0.623766233766234	0.5647058823529412	0.5925925925925926	0.67107266435986



Сравнение ROC-кривых для различных режимов



## **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 с UMAP

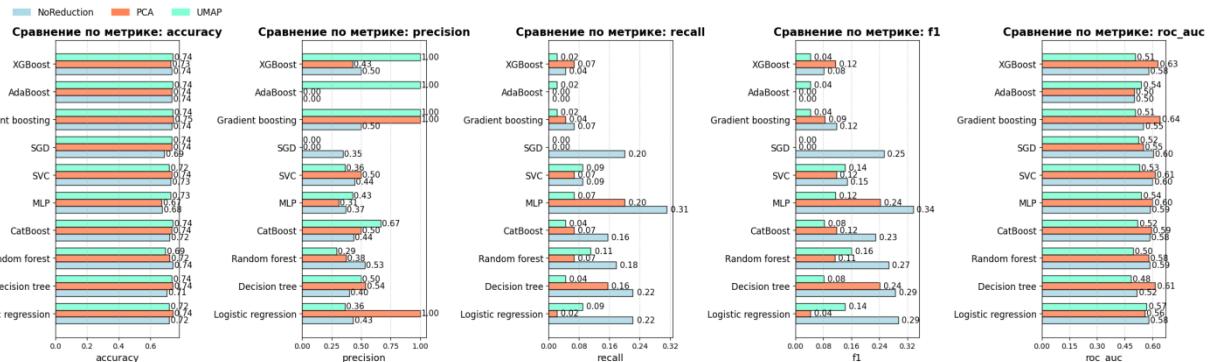
**Параметры:** learning\_rate = 0.2; max\_depth = 5; n\_estimators = 100; subsample = 1.0

### **Обоснование:**

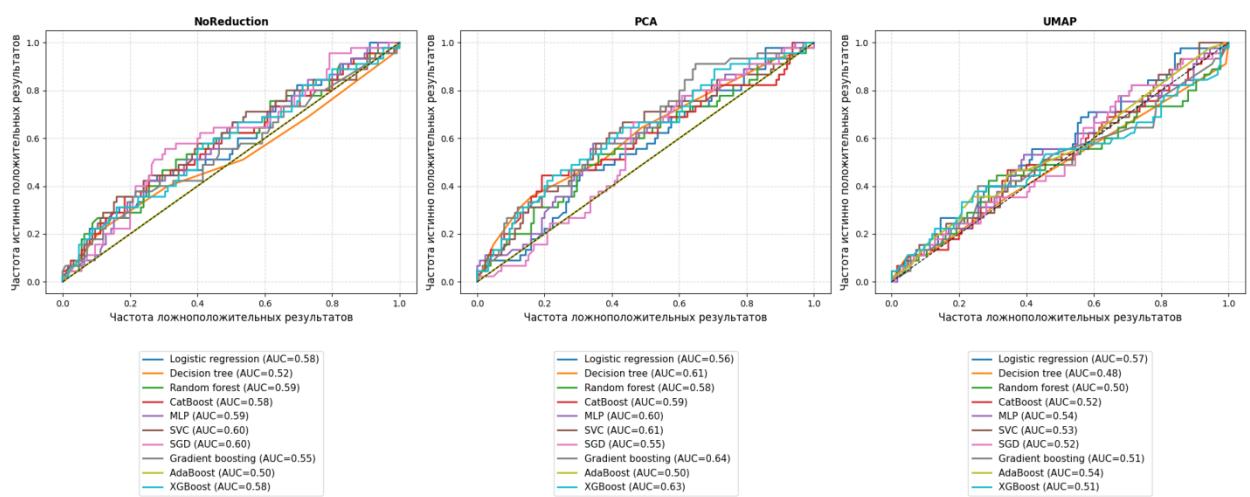
- Высокое значение ROC AUC (0.6909) — хорошая способность различать классы.
- Лучший F1-score — баланс между precision и recall.
- Наивысший recall — минимизирует ложноотрицательные ошибки.
- Хорошая точность — меньше ложноположительных ошибок.

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

Модель	Метод редукции	Лучшие параметры	Accuracy	Precision	Recall	F1	ROC AU
AdaBoost	NoReduction	{'n_estimators': 50}	0.7352941176470589	0.0	0.0	0.0	0
AdaBoost	PCA	{'n_estimators': 50}	0.7352941176470589	0.0	0.0	0.0	0
AdaBoost	UMAP	{'n_estimators': 100}	0.7411764705882353	1.0	0.02222222222222223	0.043478260869565216	0.5390222222222222
CatBoost	NoReduction	{'learning_rate': 0.01}	0.7235294117647059	0.4375	0.15555555555555556	0.22950819672131148	0.5836444444444444
CatBoost	PCA	{'learning_rate': 0.01}	0.7352941176470589	0.5	0.066666666666666667	0.11764705882352941	0.5934222222222222
CatBoost	UMAP	{'learning_rate': 0.01}	0.7411764705882353	0.666666666666666666	0.044444444444444446	0.08333333333333333	0.5191111111111111
Decision tree	NoReduction	{'entropy', 'max_depth': 3}	0.7058823529411765	0.4	0.2222222222222222	0.2857142857142857	0.5168888888888888
Decision tree	PCA	{'entropy', 'max_depth': 3}	0.7411764705882353	0.5384615384615384	0.15555555555555556	0.2413793103448276	0.6143111111111111
Decision tree	UMAP	{'entropy', 'max_depth': 3}	0.7352941176470589	0.5	0.044444444444444446	0.08163265306122448	0.4831111111111111
Gradient boosting	NoReduction	{'rs': 50, 'subsample': 1.0}	0.7352941176470589	0.5	0.066666666666666667	0.11764705882352941	0.5505777777777777
Gradient boosting	PCA	{'rs': 50, 'subsample': 0.9}	0.7470588235294118	1.0	0.044444444444444446	0.0851063829787234	0.6400888888888888
Gradient boosting	UMAP	{'rs': 100, 'subsample': 1.0}	0.7411764705882353	1.0	0.02222222222222223	0.043478260869565216	0.5062222222222222
Logistic regression	NoReduction	{'C': 0.1, 'solver': 'lbfgs'}	0.7176470588235294	0.43478260869565216	0.2222222222222222	0.29411764705882354	0.5780444444444444
Logistic regression	PCA	{'C': 0.1, 'solver': 'lbfgs'}	0.7411764705882353	1.0	0.02222222222222223	0.043478260869565216	0.5585777777777777
Logistic regression	UMAP	{'C': 1, 'solver': 'liblinear'}	0.7176470588235294	0.36363636363636365	0.088888888888888889	0.14285714285714285	0.5683555555555555
MLP	NoReduction	{'is': '(100,)', 'max_iter': 500}	0.6764705882352942	0.3684210526315789	0.3111111111111111	0.3373493975903614	0.5870222222222222
MLP	PCA	{'is': '(100,)', 'max_iter': 500}	0.6705882352941176	0.3103448275862069	0.2	0.24324324324324326	0.6003555555555555
MLP	UMAP	{'is': '(100,)', 'max_iter': 500}	0.7294117647058823	0.42857142857142855	0.066666666666666667	0.11538461538461539	0.5388444444444444
Random forest	NoReduction	{'None', 'n_estimators': 50}	0.7411764705882353	0.5333333333333333	0.17777777777777778	0.266666666666666666	0.5860444444444444
Random forest	PCA	{'sqrt', 'n_estimators': 100}	0.7235294117647059	0.375	0.066666666666666667	0.11320754716981132	0.5779555555555555
Random forest	UMAP	{'None', 'n_estimators': 100}	0.6941176470588235	0.29411764705882354	0.1111111111111111	0.16129032258064516	0.4950222222222222
SGD	NoReduction	{'000', 'penalty': 'elasticnet'}	0.6882352941176471	0.34615384615384615	0.2	0.2535211267605634	0.6045333333333333
SGD	PCA	{'1_iter': 1000, 'penalty': '11'}	0.7352941176470589	0.0	0.0	0.0	0.5496888888888888
SGD	UMAP	{'1_iter': 1000, 'penalty': '12'}	0.7352941176470589	0.0	0.0	0.0	0.5237333333333333
SVC	NoReduction	{'C': 1, 'kernel': 'rbf'}	0.7294117647058823	0.4444444444444444	0.088888888888888889	0.14814814814814814	0.5996444444444444
SVC	PCA	{'C': 1, 'kernel': 'rbf'}	0.7352941176470589	0.5	0.066666666666666667	0.11764705882352941	0.6131555555555555
SVC	UMAP	{'C': 10, 'kernel': 'rbf'}	0.7176470588235294	0.36363636363636365	0.088888888888888889	0.14285714285714285	0.5297777777777777
XGBoost	NoReduction	{'rs': 100, 'subsample': 0.9}	0.7352941176470589	0.5	0.044444444444444446	0.08163265306122448	0.5790222222222222
XGBoost	PCA	{'rs': 100, 'subsample': 0.9}	0.7294117647058823	0.42857142857142855	0.066666666666666667	0.11538461538461539	0.6293333333333333
XGBoost	UMAP	{'rs': 100, 'subsample': 0.9}	0.7411764705882353	1.0	0.02222222222222223	0.043478260869565216	0.5055999999999999



Сравнение ROC-кривых для различных режимов



## **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.