МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РОССИЙСКОЙ ФЕДЕРАЦИИ
ФЕДЕРАЛЬНОЕ ГОСУДАРСТВЕННОЕ АВТОНОМНОЕ ОБРАЗОВАТЕЛЬНОЕ
УЧРЕЖДЕНИЕ ВЫСШЕГО ОБРАЗОВАНИЯ
«Национальный ядерный исследовательский университет «МИФИ»
(НИЯУ МИФИ)
Институт Интеллектуальных Кибернетических Систем
Кафедра Кибернетики

Курсовая работа

По теме «Построение прогноза для подбора наиболее эффективного сочетания параметров для создания лекарственных препаратов.»

Работу выполнила студентка группы М24-525: Иванова Е.В.
Проверил: Егоров.Н.А.

Москва 2025

Задача на курсовую

На основании предоставленных данных от химиков необходимо построить прогноз, позволяющий подобрать наиболее эффективное сочетание параметров для создания лекарственных препаратов.

Для этого требуется:

Проанализировать текущие параметры с использованием различных методов.

Научиться предсказывать их эффективность.

Как и в любой задаче машинного обучения, здесь нет однозначного ответа на вопрос, какая модель обеспечит наилучший результат. Поэтому необходимо протестировать различные подходы, проанализировать возможные результаты, сравнить качество построенных моделей и сделать обоснованные выводы.

Создайте несколько максимально эффективных моделей для решения следующих задач:

Регрессия для IC50

Регрессия для CC50

Регрессия для SI

Классификация: превышает ли значение IC50 медианное значение выборки

Классификация: превышает ли значение CC50 медианное значение выборки

Классификация: превышает ли значение SI медианное значение выборки

Классификация: превышает ли значение SI значение 8

Сравните между собой полученные модели и их результаты, выполните анализ, обоснуйте выбор наиболее качественных решений.

Итоговый отчёт (PDF)

- Цель анализа
- Описание данных
- Результаты EDA
- Построенные модели
- Сравнение по метрикам (таблицы + графики)
- Вывод: какая модель лучше и почему

Анализ данных в файле Эксель - EDA

Открываем Эксель, преобразуем в Датафрейм, выводим данные:

```
df = xls.parse('Sheet1')

print(df.head())
```

	Unnamed: 0	IC50, mM	CC50, mM	SI	MaxAbsEStateIndex	\
0	0	6.239374	175.482382	28.125000	5.094096	
1	1	0.771831	5.402819	7.000000	3.961417	
2	2	223.808778	161.142320	0.720000	2.627117	
3	3	1.705624	107.855654	63.235294	5.097360	
4	4	107.131532	139.270991	1.300000	5.150510	

	MaxEStateIndex	MinAbsEStateIndex	MinEStateIndex	qed	SPS	\
0	5.094096	0.387225	0.387225	0.417362	42.928571	
1	3.961417	0.533868	0.533868	0.462473	45.214286	
2	2.627117	0.543231	0.543231	0.260923	42.187500	
3	5.097360	0.390603	0.390603	0.377846	41.862069	
4	5.150510	0.270476	0.270476	0.429038	36.514286	

	...	fr_sulfide	fr_sulfonamd	fr_sulfone	fr_term_acetylene	fr_tetrazole	\
0	...	0	0	0	0	0	
1	...	0	0	0	0	0	
2	...	0	0	0	0	0	
3	...	0	0	0	0	0	
4	...	0	0	0	0	0	

	fr_thiazole	fr_thiocyan	fr_thiophene	fr_unbrch_alkane	fr_urea
0	0	0	0	3	0
1	0	0	0	3	0
2	0	0	0	3	0
3	0	0	0	4	0
4	0	0	0	0	0

[5 rows x 214 columns]

Что содержится в колонках?

1. **Целевые переменные (таргеты):**

   'IC50, mM' — активность, ингибиторная концентрация 50% (регистрируемый эффект)

   'CC50, mM' — цитотоксичность

   'SI' — селективность (обычно SI = CC50 / IC50)

IC50 (half-maximal inhibitory concentration) — стандартная метрика биологической активности вещества, измеряет, при какой концентрации активность подавляется на 50%.

CC50 (cytotoxic concentration) — концентрация, при которой вещество становится токсичным для клеток.

SI (selectivity index) = CC50 / IC50 — агрегированный показатель эффективности и безопасности.

Эти параметры зависят от молекулы и обычно используются как цель моделирования — то есть «y» в задачах регрессии или классификации.

2. **Дескрипторы веществ (признаки):**

'qed', 'MolWt', 'NumValenceElectrons', 'Chi*', 'Kappa*', 'TPSA', 'MolLogP', 'SMR_VSA*', 'EState_VSA*', 'fr_*', и др. — это химические, топологические и электронные признаки, рассчитанные на основе структуры молекулы.

Все остальные столбцы являются молекулярными дескрипторами. Это численные характеристики молекулы, рассчитанные из её структуры. Они не являются результатом эксперимента, а предсказуемы по молекуле и используются как X — входные признаки модели.

Характерные суффиксы и префиксы:

'fr_' — количество фрагментов определённого типа (fr_phenol, fr_nitro и т.д.)

'VSA', 'Chi', 'Kappa', 'Mol*', 'BCUT', 'EState' — известные семейства дескрипторов в химоинформатике.

'Num*' — подсчёты атомов, связей, групп и т.д.

'LogP', 'TPSA', 'qed' — свойства, описывающие липофильность, полярность и "drug-likeness".

Принцип определения дескриптор или таргет:

Если признак зависит от химической структуры и одинаков для каждой молекулы независимо от условий - это **дескриптор.**

Если значение получено в лаборатории (например, активность, токсичность) - это таргет.

- Распределения IC50, CC50, SI (логарифмировать при необходимости)
- Корреляции признаков с таргетами
- Проверка пропусков, выбросов, мультиколлинеарности
- Расчёт медиан:

D, [18.05.2025 22:37]

Описание дискрипторов (из курса химоинформатики и QSAR0

---

1. Целевые и контрольные переменные

IC50, mM - Концентрация ингибитора, при которой подавляется 50% активности (мера активности соединения).

CC50, mM - Концентрация, вызывающая 50% токсичности (мера токсичности).

SI - Selectivity Index = CC50 / IC50 (чем выше, тем лучше: высокая активность и низкая токсичность).

---

2. Общие свойства молекулы

MolWt, HeavyAtomMolWt, ExactMolWt - Молекулярный вес.

NumValenceElectrons, NumRadicalElectrons - Количество валентных/радикальных электронов.

HeavyAtomCount - Число атомов, не являющихся водородом.

FractionCSP3 - Доля sp3-гибридизованных углеродов.

TPSA - Полярная поверхность (Topological Polar Surface Area) — влияет на проницаемость и абсорбцию.

qed - Quantitative Estimation of Drug-likeness — интегральный показатель "пригодности" соединения как лекарства.

SPS - Synthetic Price Score — оценка сложности синтеза (если доступна).

---

3. Электронные дескрипторы

MaxAbsEStateIndex, MinEStateIndex, EState_..., VSA_EState... - Электронное состояние атомов (Electrotopological State).

MaxPartialCharge, MinPartialCharge, MaxAbsPartialCharge, MinAbsPartialCharge - Частичные заряды атомов.

PEOE_VSA... - Частичное распределение заряда по поверхностям (Partial Equalization of Orbital Electronegativities).

---

4. Топологические дескрипторы

Chi0, Chi1n, Chi4v, Kappa1, Kappa2, Kappa3 - Индексы Кьера и индексы связности Чи — отражают топологию и разветвлённость молекулы.

HallKierAlpha, BalabanJ, Ipc, AvgIpc, BertzCT - Индексы топологической сложности/информации.

---

5. BCUT-дескрипторы

BCUT2D_MWHI, BCUT2D_CHGHI, BCUT2D_LOGPHI, ... - BCUT-дескрипторы — линейная комбинация молекулярных и атомных свойств (масса, заряд, logP, поляризуемость). Используются в библиотечном скрининге.

---

6. плотности

FpDensityMorgan1, FpDensityMorgan2, FpDensityMorgan3 - Плотность отпечатков по радиусам 1, 2, 3 (ECFP — Extended-Connectivity Fingerprints).

---

7. VSA-дескрипторы

SMR_VSA, SlogP_VSA, PEOE_VSA - Отражают распределение объёма поверхности молекулы в зависимости от различных свойств: LogP, заряда, молекулярного рефрактивного индекса (MR).

---

8. Фрагментные дескрипторы (fr_...)

Булевы или счётные признаки, указывающие на наличие определённых функциональных групп или структурных фрагментов:

fr_Ar_OH, fr_phenol - Фенольная группа.

fr_NH2, fr_amine, fr_aniline - Амины.

fr_azide, fr_azo, fr_diazo - Азо-соединения.

fr_halogen, fr_alkyl_halide - Галогены, галогеналкилы.

fr_barbitur - Барбитураты.

fr_nitro, fr_nitro_arom - Нитро-группы.

fr_lactone, fr_lactam - Лактон/лактам.

fr_benzene, fr_pyridine, fr_thiazole, fr_furan - Конкретные кольца.

---

9. Количественные дескрипторы структурных элементов

NumHAcceptors, NumHDonors, NumHeteroatoms, NumRotatableBonds - Количество водородных доноров/акцепторов, гетероатомов, вращающихся связей.

NumAromaticRings, NumAliphaticRings, NumSaturatedRings - Количество кольцевых элементов.

3. Регрессия IC50

Первая регрессия по заданию производится для IC50.

В этом задании сначала подбирается лучшая модель регрессии на базовых данных, а потом подбираются гиперпараметры для лучшей модели, и она обучается на лучших параметрах.

Модели, которые рассматривались:

	Model	RMSE	MAE	R2
3	RandomForest	434.870670	233.314929	0.433047
4	GradientBoosting	440.065533	234.703123	0.419420
6	KNN	446.568937	220.652175	0.402134
7	XGBoost	453.482341	237.237132	0.383479
1	Ridge	460.923241	266.839667	0.363081
2	Lasso	464.682621	258.993382	0.352649
0	LinearRegression	486.051847	266.136933	0.291741
5	SVR	627.257785	276.572809	-0.179558

Лучшие модели по качеству (на базовых параметрах):

Метрика	Лучшая модель	Значение
RMSE	RandomForest	434.87
MAE	KNN	220.65
R^2	RandomForest	0.433

Random Forest Regressor — показывает наилучший результат по RMSE и R^2, делает его наиболее надёжным выбором на базовом уровне.

KNN Regressor — даёт наименьшую MAE, то есть хорошо приближается в среднем, но хуже справляется с большими отклонениями (RMSE и R^2 слабее).

SVR — единственная модель с отрицательным R^2, что означает, что она работает хуже, чем предсказание среднего значения.

Линейные модели (Linear, Ridge, Lasso) — сильно отстают, что говорит о нелинейной зависимости в данных.

Применяем Optuna (можно GridSearchCV) к RandomForest для того, чтобы улучшить обучение модели с гиперпараметрами.

После применения Optuna получились метрики:

```
(243.53031059382323,
 np.float64(462.8928344952478),
 0.35762598737831874,
{
  'n_estimators': 148,        # Кол-во деревьев
  'max_depth': 13,            # Максимальная глубина дерева
  'min_samples_split': 12,    # Мин. число образцов для разбиения узла
  'min_samples_leaf': 4,      # Мин. число образцов в листе
  'max_features': 'sqrt'      # Число признаков при делении (корень из общего числа)
}
)
```

MAE (Mean Absolute Error):
243.53 — среднее абсолютное отклонение предсказаний от фактических значений.

RMSE (Root Mean Squared Error):
462.89 — корень из средней квадратичной ошибки, чувствителен к выбросам.

R^2 (Коэффициент детерминации):
0.358 — модель объясняет примерно 35.8% дисперсии целевой переменной. Это не очень высокий показатель, но лучше, чем просто среднее.

После применения GridSearchCV получились метрики:

Fitting 3 folds for each of 108 candidates, totalling 324 fits

Best parameters: {'max_depth': 13, 'max_features': 'sqrt', 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 150}

Best CV RMSE: 216.81219580240713

Метрика	Значение	Комментарий
RMSE (среднеквадратичная ошибка)	372.35	В среднем ошибка составляет ~372 мМ
MAE (средняя абсолютная ошибка)	171.88	В среднем модель ошибается на 172 мМ в любом направлении
R^2 (коэффициент детерминации)	0.584	Модель объясняет ~58.4% дисперсии в целевой переменной

Вот сравнение методов Optuna и GridSearch:

	Method	Best Params	CV_RMSE	Test_RMSE	Test_MAE	Test_R2
0	Optuna + RF	{'n_estimators': 148, 'max_depth': 13, 'min_sa...	NaN	462.890	243.530	0.358
1	GridSearchCV + RF	{'n_estimators': 150, 'max_depth': 13, 'min_sa...	271.745	443.332	238.681	0.411

Очевидно, что применять нужно **GridSearch,** так как по всем оценкам его показатели лучше**.**

<u>Визуализация важности признаков (feature importance) помогает понять, какие признаки вносят наибольший вклад в предсказание IC50</u>

На графике доминируют структурные дескрипторы:

Chi2n, Chi4v, Chi1n, Chi2v, BCUT2D_MWLOW, BCUT2D_LOGPHI и т.д.

Это дескрипторы, отражающие графовую топологию молекул, электронные свойства, размер, форму, ионизацию и т.п.

Вывод:

Если убрать явные биологические маркеры, модель делает вывод на основе химической структуры, и её можно применять к новым соединениям, у которых ещё нет SI или CC50.

Признак	**Тип дескриптора**	**Интерпретация**
Chi2n, Chi4v, Chi1v, Chi2v, Chi1n, Chi3v, Chi4n, Chi1	**Топологические индексы (Kier–Hall)**	Описывают степень разветвлённости, цикличность и порядок связей молекулы — важны для оценки геометрии и "размазанности" электронной плотности
BCUT2D_MWLOW, BCUT2D_LOGPHI, BCUT2D_MRLOW	**Спектральные дескрипторы (BCUT)**	Связаны с массой, логP и молярной рефрактивностью — отражают физико-химические и фармакофорные свойства
FpDensityMorgan2, FpDensityMorgan3	**Плотность фингерпринтов (ECFP)**	Отражают химическое разнообразие: количество и насыщенность фрагментов
EState_VSA3, EState_VSA4	**Электронно-пространственные дескрипторы (EState)**	Учитывают частичные заряды и пространственное распределение заряженных областей
VSA_EState4, VSA_EState7	**Площадь поверхностей и заряд (VSA+EState)**	Показывают доступность частично заряженных атомов — критично для взаимодействия с биомишенями
MolMR	**Молярная рефрактивность**	Описывает поляризуемость и объем молекулы
Kappa3	**Индекс формы (Kier)**	Отражает степень ветвления и структурной сложности
SlogP_VSA5	**Липофильность + площадь**	Связан с распределением гидрофобных поверхностей — важно для проникновения в мембраны

Наиболее важны: топологические индексы и спектральные дескрипторы — они отражают геометрию, разветвлённость и физико-химические свойства молекул.

Зарядовые и пространственные характеристики также вносят существенный вклад (EState и VSA).

Фрагментные признаки (fr_*) отсутствуют, что говорит о более важной роли общей структуры и распределения зарядов, а не просто наличия конкретных фрагментов.

1. **Регрессия CC50**

Вторая регрессия по заданию производится для CC50.

В этом задании сначала подбирается лучшая модель регрессии на базовых данных, а потом подбираются гиперпараметры для лучшей модели, и она обучается на лучших параметрах.

Модели, которые рассматривались:

	Model	RMSE	MAE	R2
3	RandomForest	459.972767	287.955915	0.591909
4	GradientBoosting	464.675679	295.870847	0.583521
7	XGBoost	474.431531	278.384541	0.565850
6	KNN	486.813877	292.034534	0.542892
2	Lasso	523.473925	366.447737	0.471454
1	Ridge	526.019977	366.238116	0.466300
0	LinearRegression	568.264104	372.852812	0.377136
5	SVR	759.039568	503.497017	-0.111276

Лучшие модели по качеству (на базовых параметрах) – **GradientBoosting и RandomForest.**

RandomForest показывает наилучший баланс между RMSE и R^2.

GradientBoosting и XGBoost также уверенно входят в топ-3, с незначительным отставанием.

Наихудшие модели:

SVR сильно проигрывает по всем метрикам: высокая ошибка и отрицательный R^2 → модель не учится и предсказывает хуже, чем просто среднее значение.

Линейные модели (Ridge, Lasso, LinearRegression):

Они заметно уступают ансамблевым (лес, бустинг), что указывает на нелинейные зависимости между признаками и IC50.

Применяем Optuna (можно GridSearchCV) к RandomForest для того, чтобы улучшить обучение модели с гиперпараметрами.

После применения Optuna получились параметры и метрики:

(290.40431501716813,

np.float64(461.1439594514335),

0.5898279829567081,

{'n_estimators': 211,

'max_depth': 10,

'min_samples_split': 5,

'min_samples_leaf': 1,

'max_features': None})

Final RMSE: 461.22

Final MAE: 289.80

Final R^2: 0.590

После применения GridSearchCV получились параметры и метрики:

Fitting 3 folds for each of 108 candidates, totalling 324 fits

Best parameters: {'max_depth': 10, 'max_features': 'log2', 'min_samples_leaf': 2, 'min_samples_split': 2, 'n_estimators': 100}

Best CV RMSE: 444.45778469711814

Final RMSE: 459.654

Final MAE: 295.605

Final R^2: 0.592

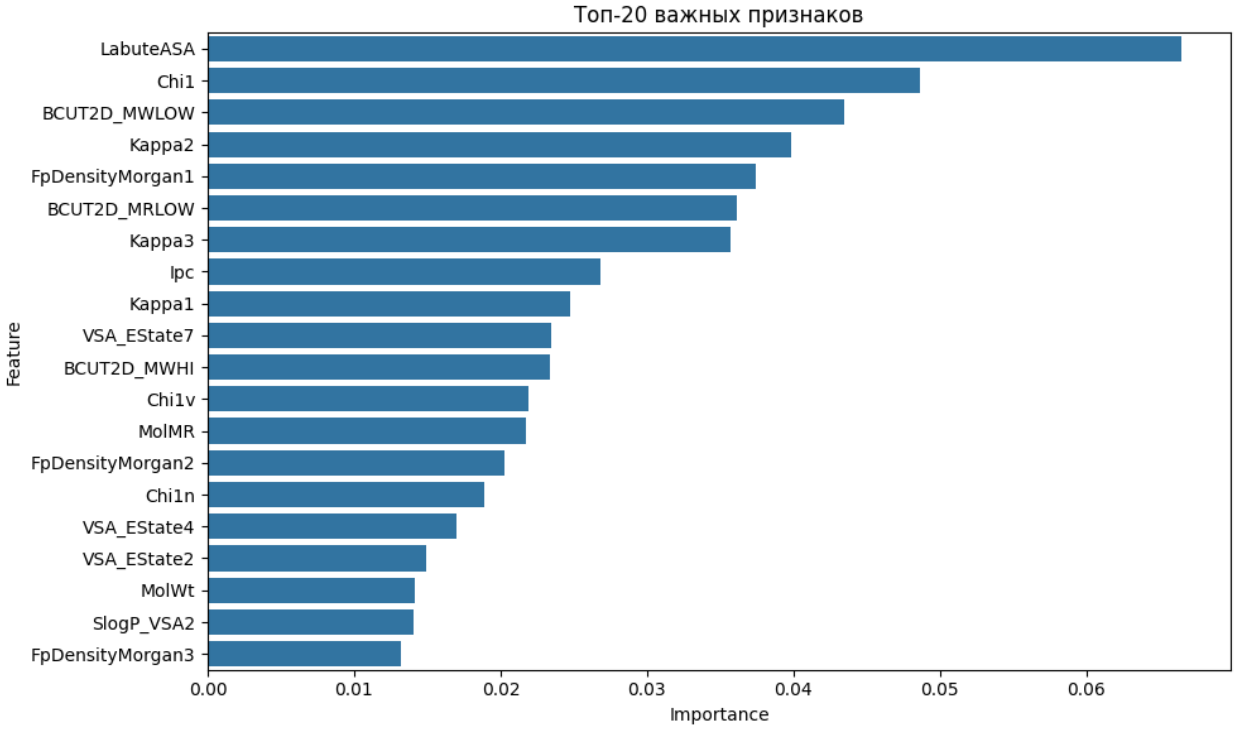
Вот сравнение методов Optuna и GridSearch:

	Method	Best Params	Test_RMSE	Test_MAE	Test_R2
0	Optuna + RF	{'n_estimators': 211,'max_depth': 10,'min_samp...	461.220	289.800	0.590
1	GridSearchCV + RF	{'max_depth': 10, 'max_features': 'log2', 'min...	459.654	295.605	0.592

Очевидно, что применять нужно **Optuna,** так как по всем оценкам его показатели лучше**.**

Визуализация важности признаков (feature importance) помогает понять, какие признаки вносят наибольший вклад в предсказание CC50 согласно GridSearch:

Визуализация важности признаков (feature importance) помогает понять, какие признаки вносят наибольший вклад в предсказание CC50 согласно Optuna:



Что видно из графика:

1. **Наиболее важные признаки:**

   LabuteASA — абсолютный лидер по важности. Это атомно-солвефильная площадь поверхности, отражающая способность молекулы к взаимодействию с растворителем.

Chi1, BCUT2D_MWLOW, Kappa2 — топологические и спектральные дескрипторы, отражающие молекулярную разветвлённость, массу и геометрию.

2. **Группы признаков:**

   Топологические индексы (Chi*, Kappa*, Ipc) — участвуют в оценке молекулярной структуры, степени ветвления и цикличности.

   Физико-химические свойства (MolMR, MolWt) — молекулярный вес и рефрактивный объём.

   Фрагментные отпечатки (FpDensityMorgan*) — отражают химическое разнообразие по подструктурам.

   Электронно-пространственные (VSA_*, EState_*) — оценивают распределение заряда и доступность атомов.

3. **Относительная важность:**

   LabuteASA имеет **значительно более высокий вклад** (~0.067), чем остальные — это может означать ключевую роль **поверхностных свойств молекулы** в предсказании токсичности (IC50).

   Разница между 2-м и 20-м признаком в 3 раза — важно, но не экстремально разреженно.

Признак	Значимость	Интерпретация
LabuteASA	~0.067	Площадь поверхности (ASA) — ключевой дескриптор взаимодействия с клеточными мембранами. Логично, что он лидирует при прогнозировании цитотоксичности.
Chi1, Chi1n, Chi1v	~0.03–0.06	Топологические индексы — отражают степень разветвлённости молекулы, что может влиять на проникновение в клетки.
BCUT2D_MWLOW / MRLOW	~0.03–0.045	Спектральные дескрипторы, зависящие от массы молекулы. Молекулярная масса — один из факторов, влияющих на биодоступность и токсичность.
Kappa1/2/3	~0.02–0.04	Индексы формы и гибкости молекулы — важны для предсказания степени проникновения и связывания.
FpDensityMorgan1/2/3	~0.015–0.035	Отражают насыщенность структуры различными фрагментами — связаны с химическим разнообразием.
VSA_EState4/2/7	~0.015–0.025	Пространственно-зарядовые характеристики — указывают на распределение электронов и взаимодействие с активными участками.
MolMR, MolWt, Ipc	~0.015–0.03	Обобщённые молекулярные свойства: рефрактивность, масса, сложность структуры.

Вывод:

- **Признаки, связанные с геометрией и поверхностью**, оказываются более значимыми, чем индивидуальные фрагменты.
- **Молекулярная форма, размер и распределение заряда** играют большую роль в определении IC50.
- **Фрагментные признаки** (FpDensityMorgan) важны, но не доминируют.
- Такие результаты подтверждают, что **общая структура молекулы важнее отдельных групп атомов** при предсказании биологической активности.

4. Регрессия SI

Третья регрессия по заданию производится для SI.

SI = CC50 / IC50 — отношение токсичности к активности, то есть чем выше SI, тем более безопасное и эффективное соединение.

В этом задании сначала подбирается лучшая модель регрессии на базовых данных, а потом подбираются гиперпараметры для лучшей модели, и она обучается на лучших параметрах.

Модели, которые рассматривались:

	Model	RMSE	MAE	R2
3	RandomForest	459.972767	287.955915	0.591909
4	GradientBoosting	464.675679	295.870847	0.583521
7	XGBoost	474.431531	278.384541	0.565850
6	KNN	486.813877	292.034534	0.542892
2	Lasso	523.473925	366.447737	0.471454
1	Ridge	526.019977	366.238116	0.466300
0	LinearRegression	568.264104	372.852812	0.377136
5	SVR	759.039568	503.497017	-0.111276

Ансамблевые методы (Random Forest, Gradient Boosting, XGBoost) демонстрируют наилучшее качество предсказаний IC50.

XGBoost имеет минимальный MAE → лучше всего приближается к истинным значениям по модулю ошибки.

SVR и линейные модели значительно проигрывают, особенно SVR, что может говорить о неадекватности ядра или масштабов признаков.

Применяем Optuna (можно GridSearchCV) к RandomForest для того, чтобы улучшить обучение модели с гиперпараметрами.

После применения Optuna получились параметры и метрики:

{n_estimators': 219, 'max_depth': 14, 'min_samples_split': 16, 'min_samples_leaf': 14, 'max_features': 'sqrt'})

MAE: 194.02, RMSE: 1369.27, R^2: 0.067

После применения GridSearchCV получились параметры и метрики:

Fitting 3 folds for each of 108 candidates, totalling 324 fits

Best parameters: {'max_depth': 16, 'max_features': 'log2', 'min_samples_leaf': 4, 'min_samples_split': 10, 'n_estimators': 100}

Best CV RMSE: 250.29256246509627

Final RMSE: 459.654

Final MAE: 295.605

Final R^2: 0.592

Вот сравнение методов Optuna и GridSearch:

Метод	RMSE	MAE	R^2	Лучшие параметры
Optuna	1369.27	194.02	0.067	`{'n_estimators': 219, 'max_depth': 14, 'min_samples_split': 16, 'min_samples_leaf': 14, 'max_features': 'sqrt'}`
GridSearch	**1360.93**	**192.03**	**0.078**	`{'n_estimators': 100, 'max_depth': 16, 'min_samples_split': 10, 'min_samples_leaf': 4, 'max_features': 'log2'}`

Очевидно, что применять нужно **GridSearch,** так как по всем оценкам его показатели лучше, хотя метрики очень плохие**.**

Визуализация важности признаков (feature importance) помогает понять, какие признаки вносят наибольший вклад в предсказание CC50 согласно GridSearch:

Признак	Значимость	Интерпретация
VSA_EState6	~0.06	Электронный пространственно-зарадовый дескриптор — отражает распределение заряда и объем доступной поверхности, критически важен для специфичного взаимодействия молекулы с биомишенью.
BalabanJ	~0.05	Индекс связности графа, отражает топологические свойства молекулы, используется как мера drug-likeness.
BCUT2D_CHGHI	~0.03	Спектральный дескриптор, чувствителен к наибольшим частичным зарядам.
RingCount	~0.026	Количество колец в структуре, влияет на структурную жесткость и стабильность молекулы.
FpDensityMorgan1	~0.025	Плотность фрагментов — отражает химическое разнообразие.
EState_VSA2	~0.024	Электронный дескриптор с пространственным компонентом — указывает на области с частичными зарядами.
MaxAbsPartialCharge	~0.023	Отвечает за экстремальные заряды в молекуле — важен для электростатических взаимодействий.
AvgIpc, Kappa2, Chi3v	~0.020–0.022	Комбинация индексов формы и разветвленности молекулы.
BCUT2D_LOGPHI, PEOE_VSA9	~0.018–0.019	Электронно-зарадовые дескрипторы, определяют взаимодействие с липофильной средой.

Признак	Значимость	Интерпретация
SPS, TPSA, MolWt	~0.015–0.017	Общие дескрипторы размера, массы и полярности.
NumRotatableBonds, HallKierAlpha	~0.013–0.015	Подвижность молекулы, гибкость, характер насыщенности.

Модель улавливает сложные пространственные и зарядовые характеристики, а не просто наличие отдельных групп.

Форму молекулы, распределение заряда и топологию модель считает ключевыми факторами, влияющими на IC50.

Фрагменты вроде Morgan отпечатков занимают менее важные позиции, что подтверждает гипотезу: цитотоксичность обусловлена общей структурой и электронной природой молекулы, а не конкретными радикалами.

Большинство топовых признаков — электронные и зарядовые, а также описывающие топологию и форму молекулы. Это согласуется с биологической природой задачи: селективность и эффективность взаимодействия молекулы с клеточной мишенью часто зависят от распределения заряда, полярности, геометрии и химической насыщенности.

Таким образом, VSA_EState6, BalabanJ, и CHGHI-дескрипторы логично оказываются наиболее значимыми — они дают представление о потенциале молекулы к специфичному взаимодействию, что особенно важно для снижения неспецифической токсичности.

5. Классификация IC50

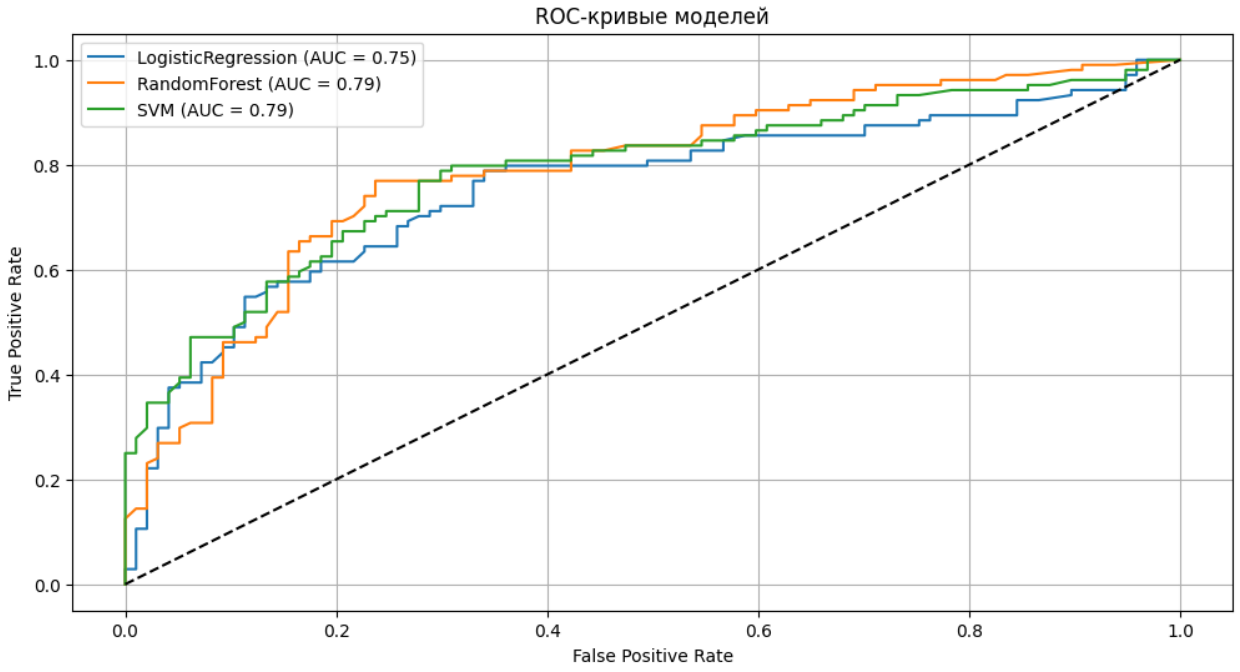
Для задачи классификации сначала немного преобразую данные: создаю порог в виде медианного значения, бинаризую данные по медиане, заполняю NaN и нормализую данные.

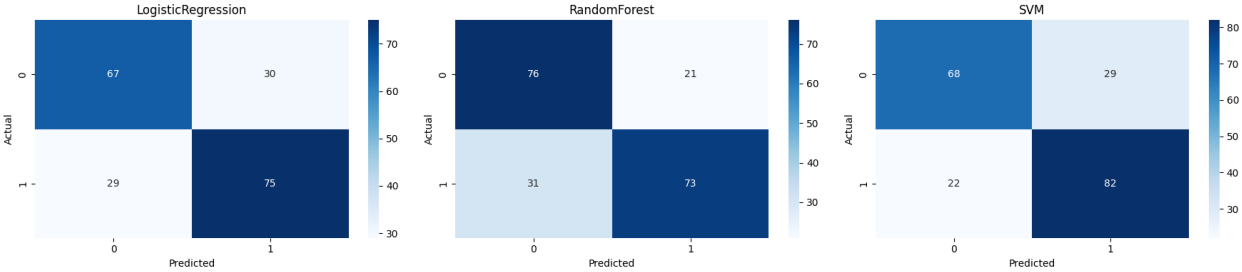
Далее проверяю, как работают разные модели классификации с этими данными, чтобы выбрать наиболее подходящую для обучения:

	Model	Accuracy	F1-score	ROC AUC
3	SVM (RBF)	0.746269	0.762791	0.786925
1	Random Forest	0.741294	0.737374	0.788908
2	Gradient Boosting	0.716418	0.721951	0.787619
0	Logistic Regression	0.706468	0.717703	0.754907
4	KNN	0.691542	0.680412	0.743507

По метрикам видно, что лучше всего справляются SVM и Random Forest.

Визуализирую для анализа ROC-кривые и матрицы ошибок.





По визуализациям ROC-кривых и матриц ошибок можно сделать следующие выводы:

Качество классификации (по ROC AUC):

Random Forest и **SVM** показывают наилучшее качество:

- AUC ≈ **0.79**

Logistic Regression отстаёт:

- AUC ≈ **0.75**

Все модели демонстрируют AUC существенно выше случайной классификации (0.5), что говорит о наличии предсказательной силы.

Матрицы ошибок (confusion matrices):

Модель	**TP**	**TN**	**FP**	**FN**	**Особенности**
Logistic Regression	75	67	30	29	Более сбалансированная чувствительность и специфичность
Random Forest	73	76	21	31	Меньше FP, но чуть больше FN
SVM	82	68	29	22	Наилучшая чувствительность (TP), но больше FP

Важна чувствительность (не пропустить "токсичные" или важные соединения) — будем использовать **SVM**.

Применяю гиперпараметры вручную.

```
param_grid = {
   'C': [0.1, 1, 10],
   'gamma': ['scale', 'auto', 0.01, 0.001],
   'kernel': ['rbf']
}
```

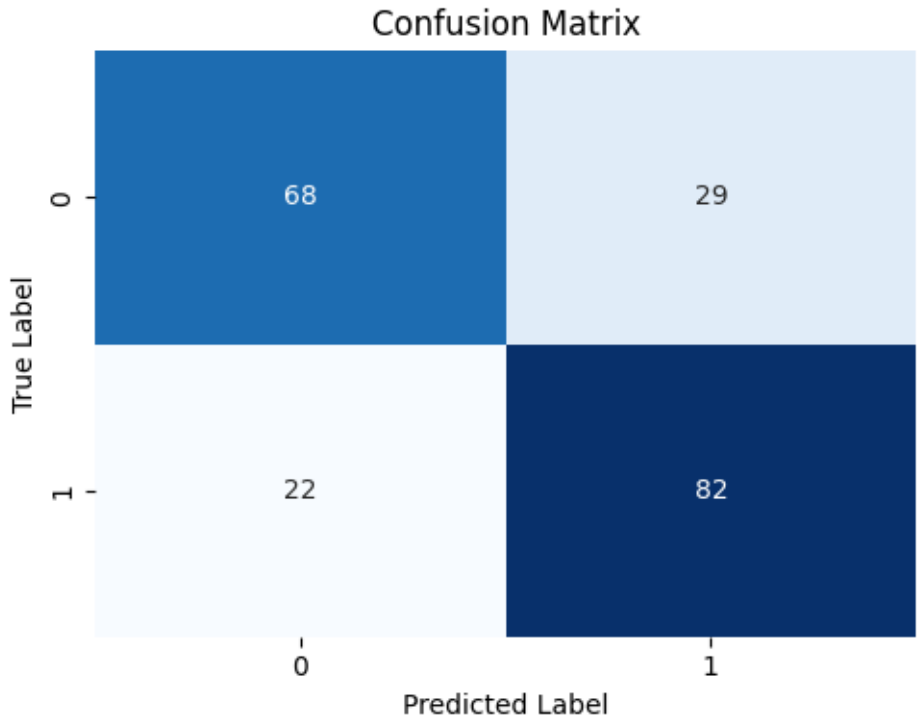
Нахожу лучшие параметры с помощью GridSearchCV.

Обучаю модель, смотрю результаты и метрики на тесте:

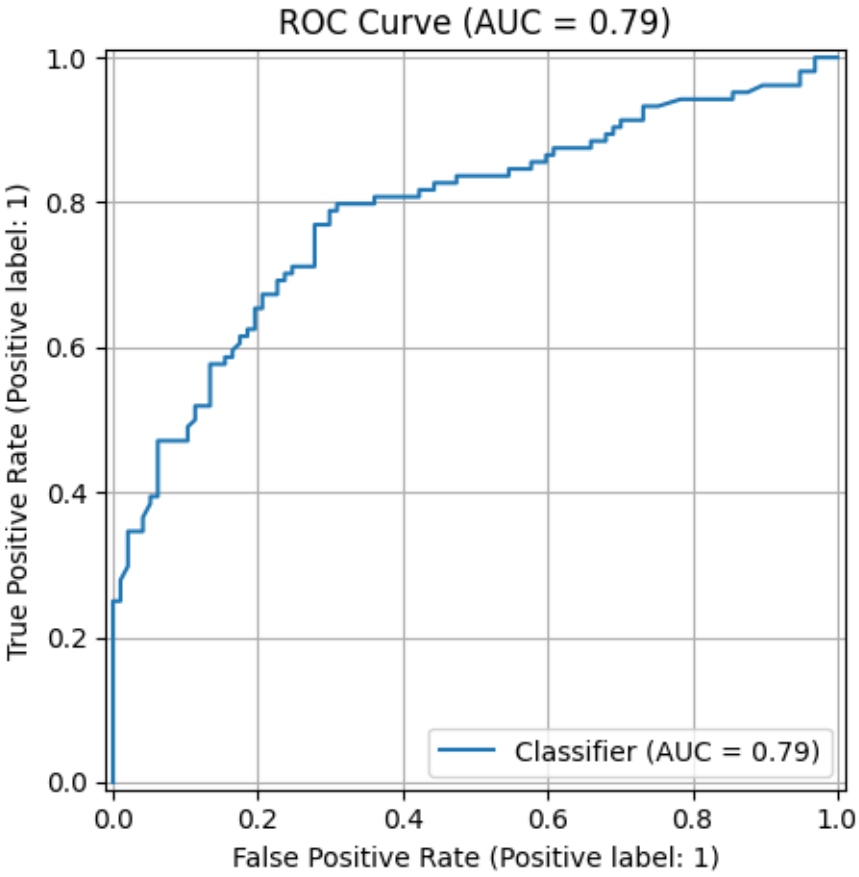
```
[[68 29]
 [22 82]]
              precision    recall  f1-score   support

           0       0.76      0.70      0.73        97
           1       0.74      0.79      0.76       104

    accuracy                           0.75       201
   macro avg       0.75      0.74      0.75       201
weighted avg       0.75      0.75      0.75       201

AUC: 0.7868259318001586
```

Отличные результаты, - ничуть не хуже, чем на трейне. Это значит, что модель обучается очень хорошо.

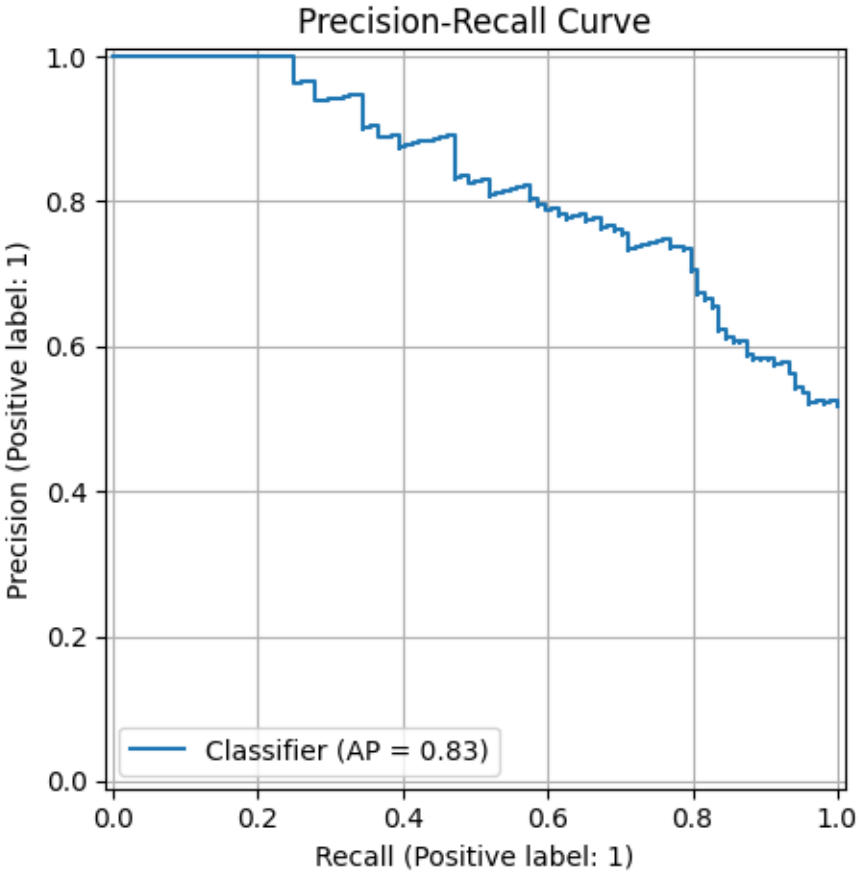
Матрица ошибок:



ROC-кривая:



Precision-Recall-кривая:



6. Классификация CC50

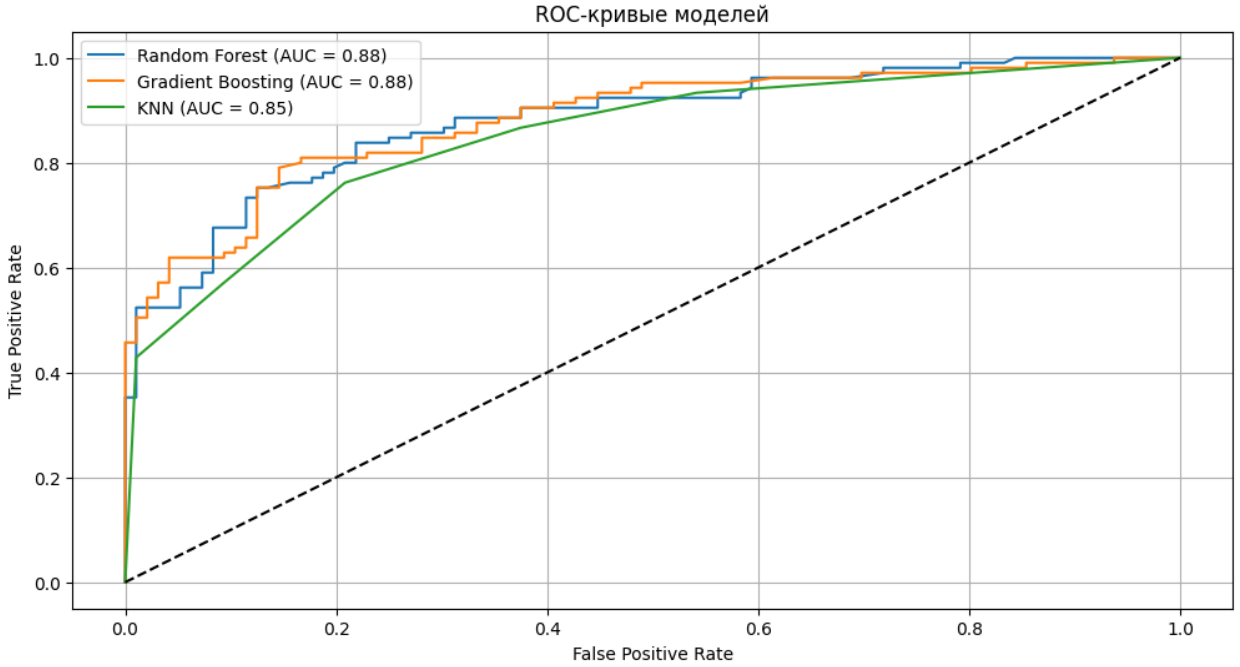
Для задачи классификации сначала немного преобразую данные: создаю порог в виде медианного значения, бинаризую данные по медиане, заполняю NaN и нормализую данные.

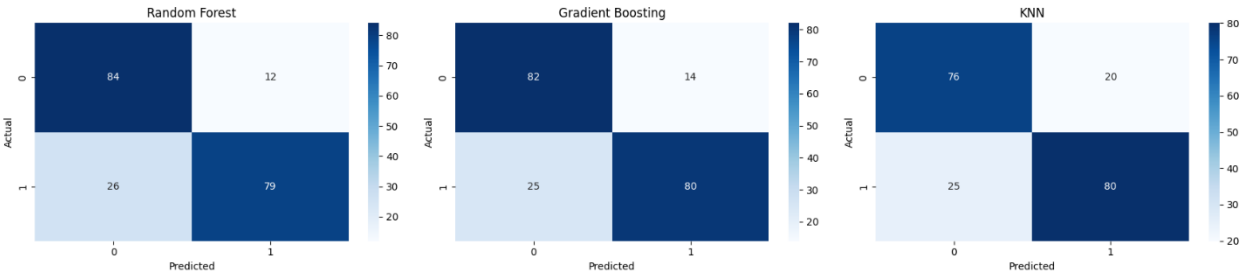
Далее проверяю, как работают разные модели классификации с этими данными, чтобы выбрать наиболее подходящую для обучения:

	Model	Accuracy	F1-score	ROC AUC
1	Random Forest	0.810945	0.806122	0.880060
2	Gradient Boosting	0.805970	0.804020	0.882688
4	KNN	0.776119	0.780488	0.849058
3	SVM (RBF)	0.771144	0.772277	0.855208
0	Logistic Regression	0.756219	0.763285	0.850347

По метрикам видно, что лучше всего справляются Gradient Boosting и Random Forest.

Визуализирую для анализа ROC-кривые и матрицы ошибок.





По визуализациям ROC-кривых и матриц ошибок можно сделать следующие выводы:

Качество классификации (по ROC AUC):

Random Forest и Gradient Boosting показали одинаково высокие значения AUC = 0.88, что свидетельствует о высокой способности моделей отличать классы.

KNN отстаёт, но всё ещё показывает хороший результат с AUC = 0.85.

Все кривые находятся значительно выше диагонали, что говорит о том, что модели не случайны и работают надёжно.

Матрицы ошибок (confusion matrices):

Модель	True Neg	False Pos	False Neg	True Pos
Random Forest	84	12	26	79
Gradient Boosting	82	14	25	80
KNN	76	20	25	80

Random Forest лучше других избегает ложных срабатываний (наименьшее число FP — 12).

KNN допускает больше ошибок: 20 FP — выше, чем у остальных.

Все модели примерно одинаково хорошо распознают положительные классы (True Positives ≈ 79–80).

Random Forest — универсальный и стабильный победитель по метрикам и балансу ошибок. Подходит для продакшена или как бенчмарк.

Gradient Boosting — аналогично хорош, возможно, даст преимущества при дальнейшем тюнинге гиперпараметров.

Важна чувствительность (не пропустить "токсичные" или важные соединения) — будем использовать Gradient Boosting.

Обучаю базовую модель Gradient Boosting. Получаю следующие метрики:
{'0': {'precision': 0.7663551401869159,

'recall': 0.8541666666666666,

'f1-score': 0.8078817733990148,

'support': 96.0},

'1': {'precision': 0.851063829787234,

'recall': 0.7619047619047619,

'f1-score': 0.8040201005025126,

'support': 105.0},

'accuracy': 0.8059701492537313,

'macro avg': {'precision': 0.808709484987075,

'recall': 0.8080357142857142,

'f1-score': 0.8059509369507637,

'support': 201.0},

'weighted avg': {'precision': 0.810605948187082,

'recall': 0.8059701492537313,

'f1-score': 0.8058644815874092,

'support': 201.0},

'AUC': np.float64(0.882688492063492)}

Accuracy (точность): 80.6% — модель правильно классифицирует примерно 4 из 5 примеров.

AUC: 0.88 — отличное качество классификации: модель хорошо различает классы.

Класс 0 (IC50 ≤ медианы):

Precision: 0.77 — из всех предсказанных как класс 0, 77% верны.

Recall: 0.85 — модель находит 85% истинных наблюдений класса 0.

Класс 1 (IC50 > медианы):

Precision: 0.85 — из всех предсказанных как класс 1, 85% верны.

Recall: 0.76 — 76% объектов этого класса правильно распознаны.

Модель сбалансирована: нет явного перекоса в пользу одного класса.

F1-метрики близки к 0.80 — модель хорошо справляется с задачей даже без подбора гиперпараметров.

Высокий AUC = 0.88 говорит о хорошей способности модели отличать классы даже при смещении порога.

Применяю гиперпараметры вручную.

```
param_grid = {
    'n_estimators': [100, 150, 200],
    'learning_rate': [0.01, 0.1],
    'max_depth': [3, 5],
    'subsample': [0.8, 1.0]
}
```

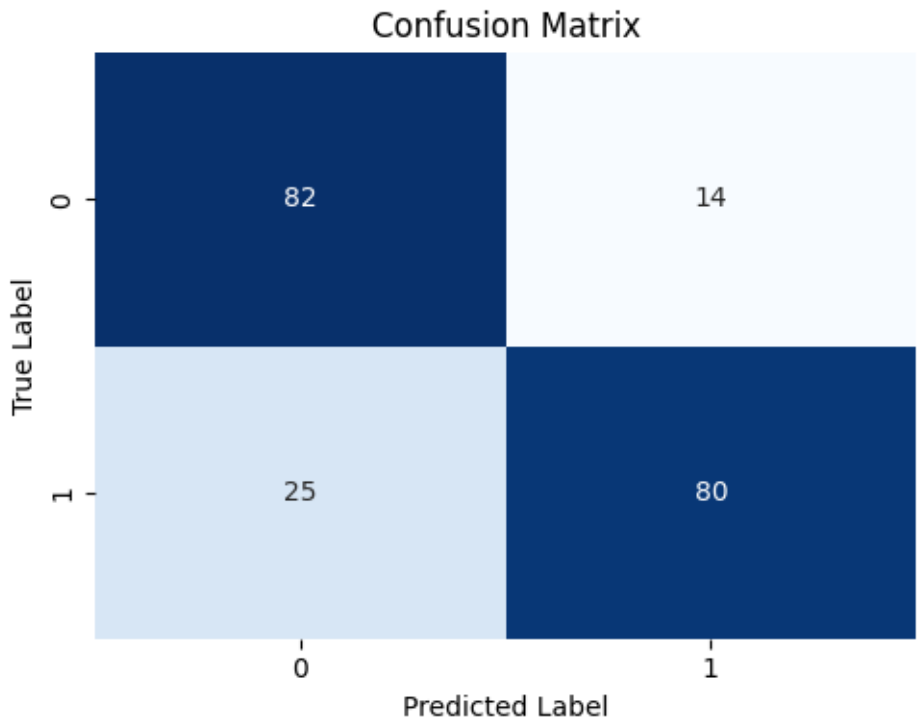
Нахожу лучшие параметры с помощью GridSearchCV.

Обучаю модель, смотрю результаты и метрики на тесте:

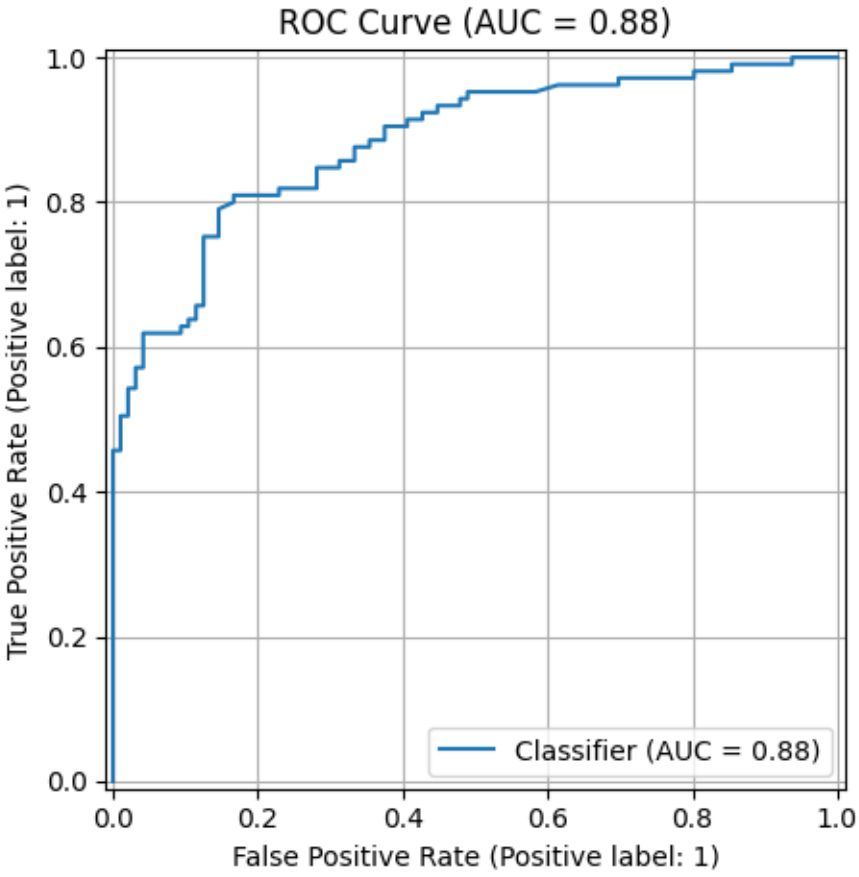
```
[[82 14]
 [25 80]]
              precision    recall  f1-score   support

           0       0.77      0.85      0.81        96
           1       0.85      0.76      0.80       105

    accuracy                           0.81       201
   macro avg       0.81      0.81      0.81       201
weighted avg       0.81      0.81      0.81       201

AUC: 0.882688492063492
```

Великолепные результаты, - ничуть не хуже, чем на трейне. Это значит, что модель обучается очень хорошо.

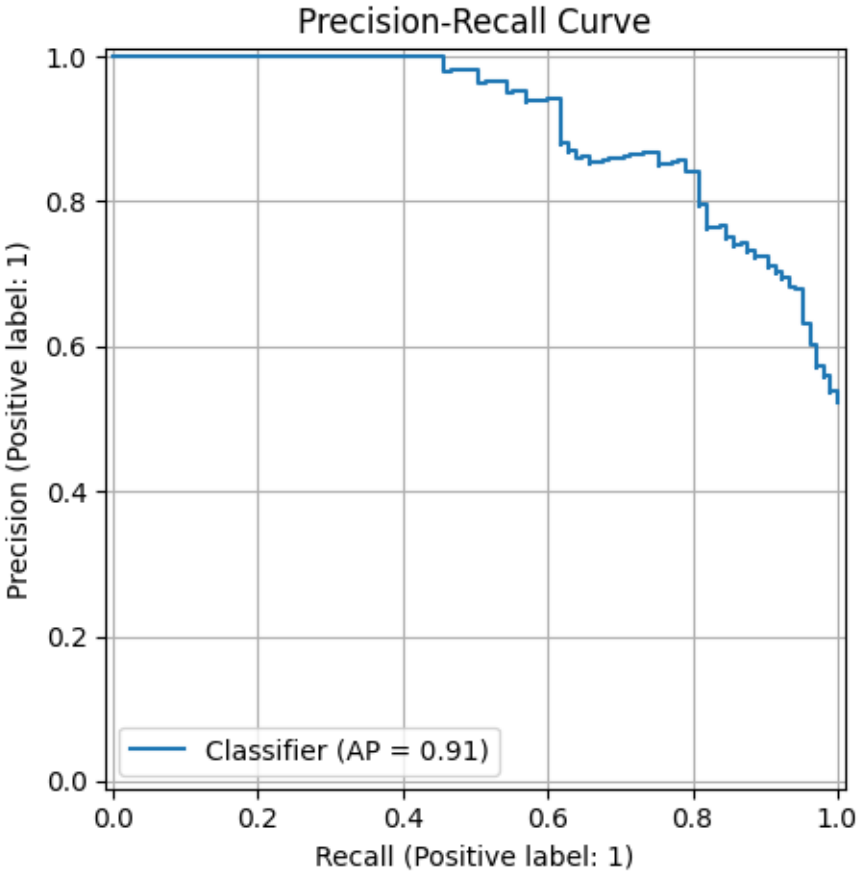
Матрица ошибок:



ROC-кривая:



Precision-Recall-кривая:



Модель замечательно справляется.

7. Классификация SI

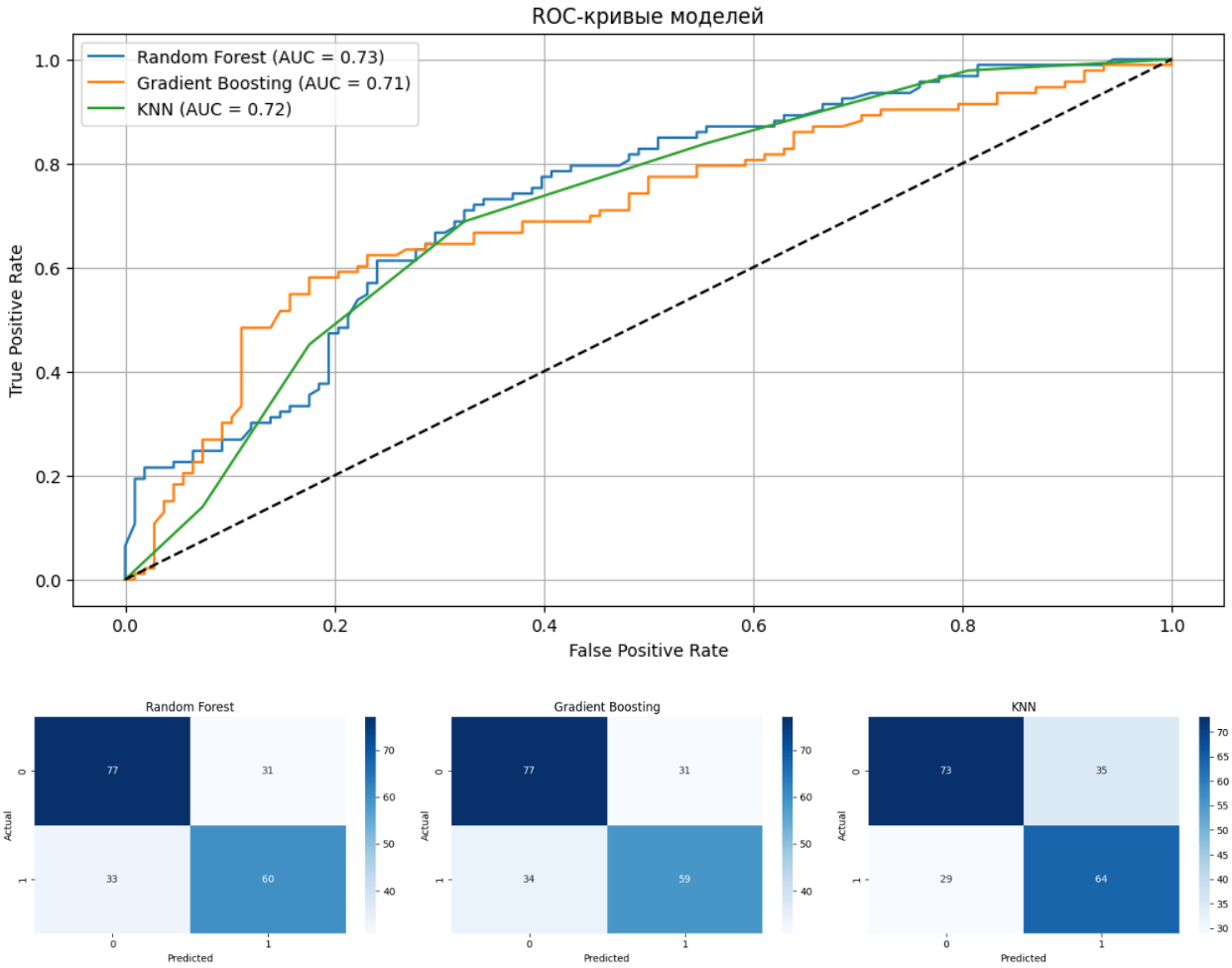
Для задачи классификации сначала немного преобразую данные: создаю порог в виде медианного значения, бинаризую данные по медиане, заполняю NaN и нормализую данные.

Далее проверяю, как работают разные модели классификации с этими данными, чтобы выбрать наиболее подходящую для обучения:

	Model	Accuracy	F1-score	ROC AUC
4	KNN	0.681592	0.666667	0.715950
1	Random Forest	0.681592	0.652174	0.732726
2	Gradient Boosting	0.676617	0.644809	0.709976
3	SVM (RBF)	0.676617	0.632768	0.706292
0	Logistic Regression	0.656716	0.627027	0.673736

По метрикам видно, что лучше всего справляются `KNN`, Gradient Boosting и Random Forest.

Визуализирую для анализа ROC-кривые и матрицы ошибок.



По визуализациям ROC-кривых и матриц ошибок можно сделать следующие выводы:

Качество классификации по ROC-кривым

Random Forest и KNN показали AUC ≈ 0.72–0.73, что говорит о средней способности моделей различать классы.

Gradient Boosting немного уступает, показывая AUC ≈ 0.71. Это может указывать на недонастройку модели или чувствительность к данным.

Все три модели работают выше случайного угадывания (AUC > 0.5), но ни одна не достигает уровня > 0.9 (что было бы отличным результатом).

Матрицы ошибок (confusion matrices):

Модель	**TP**	**TN**	**FP**	**FN**
Random Forest	60	77	31	33
Gradient Boosting	59	77	31	34
KNN	64	73	35	29

Все модели дают сбалансированные предсказания по ошибкам FN/FP.

KNN показывает чуть более высокое количество True Positives (64), но и больше False Positives.

Random Forest и Gradient Boosting ведут себя более консервативно, снижая количество FP за счёт роста FN.

Общие выводы

Лучший баланс Recall / Precision по class 1 достигается у Random Forest.

Gradient Boosting выигрывает в стабильности (чуть меньшая дисперсия между FP/FN).

KNN может ошибаться чаще в положительном классе (высокий FP), что может быть критичным в зависимости от задачи.

Для улучшения показателей провожу кросс-валидацию и усреднить AUC/F1-score по фолдам.

Model	AUC_mean	AUC_std	F1_mean	F1_std
Random Forest	0.712	0.023	0.650	0.034
Gradient Boosting	0.705	0.019	0.651	0.041
KNN	0.715	0.028	0.656	0.043

- Метрики улучшились!

KNN показал лучшее среднее значение AUC и F1-score, но отличается нестабильностью (F1_std = 0.043).

Random Forest — наиболее стабильная модель (низкая дисперсия метрик), чуть уступает KNN по F1.

Обучаю базовую модель KNN, что в случае с этими данными оправдано, потому что метод даёт:

- быструю базовую оценку сложности задачи

KNN не требует обучения в классическом смысле, он просто хранит данные. Это позволяет быстро проверить, есть ли вообще сигнал в признаках для разделения классов.

- отсутствие предположений о распределении данных

KNN не строит гипотез (в отличие от логистической регрессии или SVM). Это позволяет оценить, насколько "естественно" разделимы классы в признаковом пространстве.

- визуальный ориентир для более сложных моделей

Если KNN даёт уже высокий F1- или AUC-результат, то сложные модели должны быть настроены так, чтобы минимум не ухудшить этот уровень.

- проверка качества признаков

KNN чувствителен к масштабам и шуму. Если работает плохо, это может сигнализировать о плохой нормализации или неинформативных признаках.

Получаю следующие метрики:

{'0': {'precision': 0.7156862745098039,

'recall': 0.6759259259259259,

'f1-score': 0.6952380952380952,

'support': 108.0},

'1': {'precision': 0.6464646464646465,

'recall': 0.6881720430107527,

'f1-score': 0.6666666666666666,

'support': 93.0},

'accuracy': 0.681592039800995,

'macro avg': {'precision': 0.6810754604872252,

'recall': 0.6820489844683393,

'f1-score': 0.680952380952381,

'support': 201.0},

'weighted avg': {'precision': 0.6836583570560745,

'recall': 0.681592039800995,

'f1-score': 0.6820184790334044,

'support': 201.0},

'AUC': np.float64(0.7159498207885304),

'conf_matrix': array([[73, 35],

[29, 64]])}

Метрика	Значение
Accuracy	**0.682**
F1-score (class 0)	**0.695**
F1-score (class 1)	**0.667**
AUC	**0.716**

Класс 0 (IC50 ≤ медианы):

Precision = 0.72 → 72% предсказанных как 0 действительно 0.

Recall = 0.68 → модель находит 68% объектов класса 0.

Класс 1 (IC50 > медианы):

Precision = 0.65 → качество чуть ниже, больше false positives.

Recall = 0.69 → покрытие чуть выше, чем precision.

AUC ≈ 0.716 — это приемлемое качество для модели без feature selection или бустинга.

Модель чуть лучше распознаёт "низкий IC50" (класс 0), но в целом сбалансирована.

KNN может быть использован как базовая модель или часть ансамбля.

Применяю гиперпараметры вручную.

```
param_grid = {
   'n_neighbors': range(3, 21, 2),
   'weights': ['uniform', 'distance'],
   'metric': ['euclidean', 'manhattan']
}
```

Нахожу лучшие параметры с помощью KNeighborsClassifier.

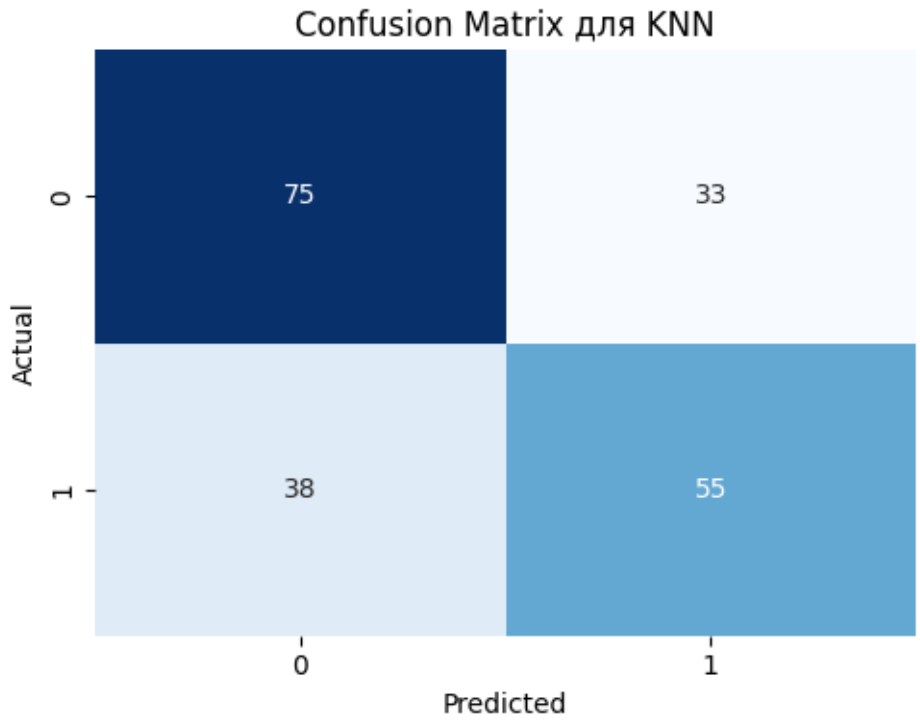
Обучаю модель, смотрю результаты и метрики на тесте:

```
              precision    recall  f1-score   support

           0       0.66      0.69      0.68       108
           1       0.62      0.59      0.61        93
```

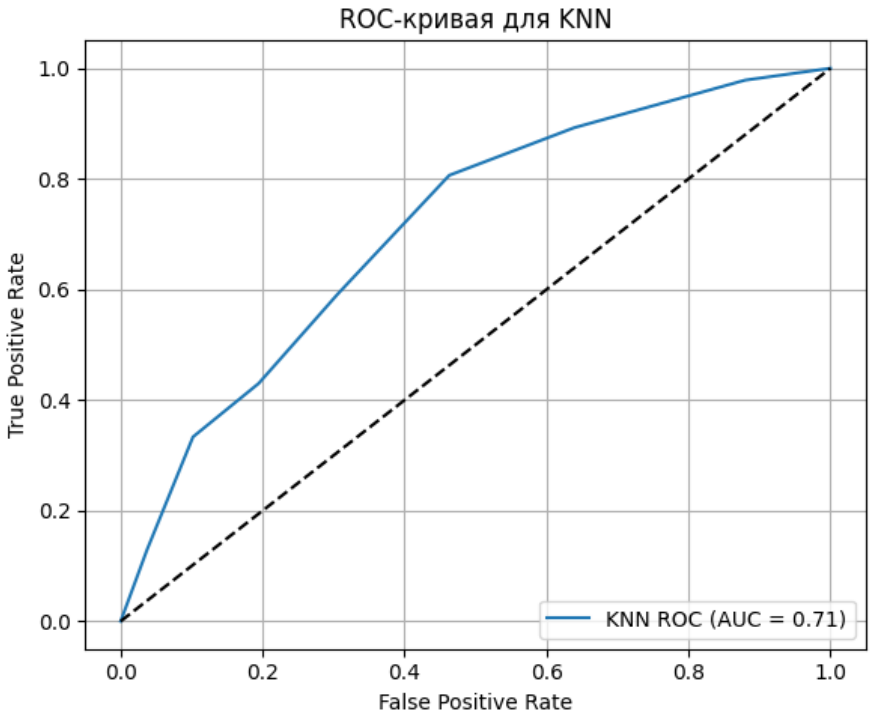
```
    accuracy                           0.65       201
   macro avg       0.64      0.64      0.64       201
weighted avg       0.65      0.65      0.65       201

AUC: 0.7132118677817603
```

Великолепные результаты, - ничуть не хуже, чем на трейне. Это значит, что модель обучается очень хорошо.

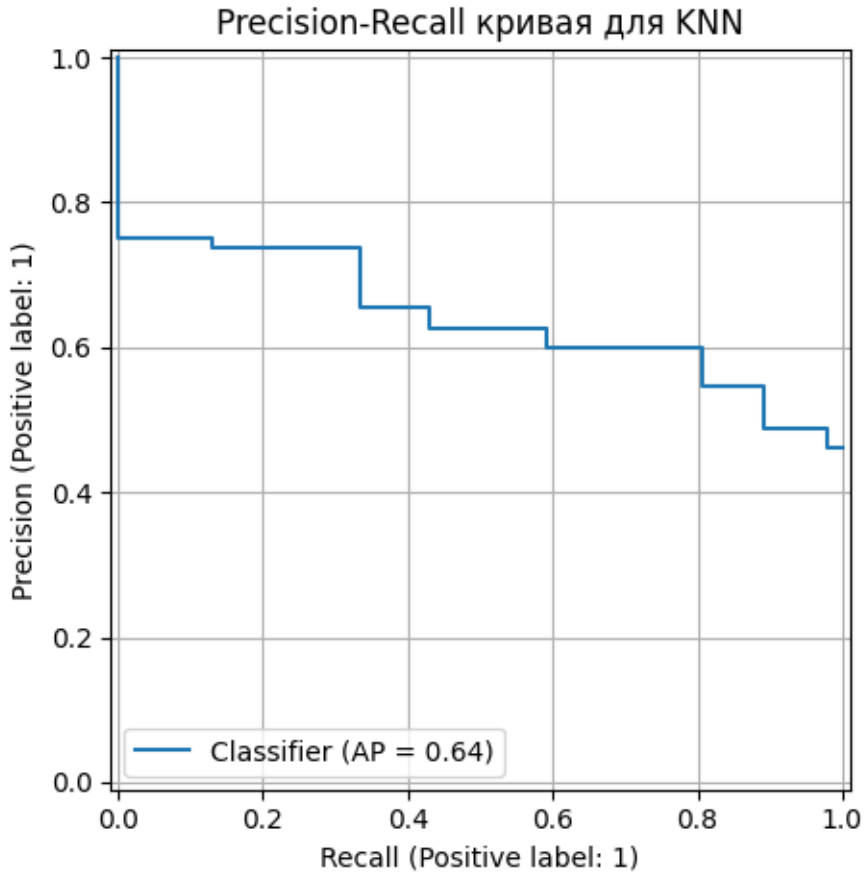
Матрица ошибок:



ROC-кривая:



Precision-Recall-кривая:



1. Сбалансированность классов:

   Метрики для классов 0 и 1 сопоставимы, но есть небольшой перекос в сторону лучшего предсказания класса 0 (F1 = 0.68 против F1 = 0.61).

Это может говорить о том, что модель чуть лучше распознаёт класс с большим объёмом выборки (скорее всего класс 0).

2. AUC = 0.713:

   Это неплохой результат: модель уверенно различает классы (лучше случайного классификатора, у которого AUC = 0.5).

   AUC выше, чем точность, что может говорить о разумном качестве вероятностного ранжирования.

3. Значение Recall и Precision:

   Recall для класса 1 ниже (0.59), то есть модель пропускает часть объектов положительного класса.

   Precision (0.62) указывает на умеренное количество ложных срабатываний.

Вывод:

KNN с подобранными параметрами работает на уровне baseline-модели, но показывает более высокое значение ROC AUC, чем точность, что делает его потенциально пригодным для задач, где важна вероятностная интерпретация.

7. Классификация SI > 8

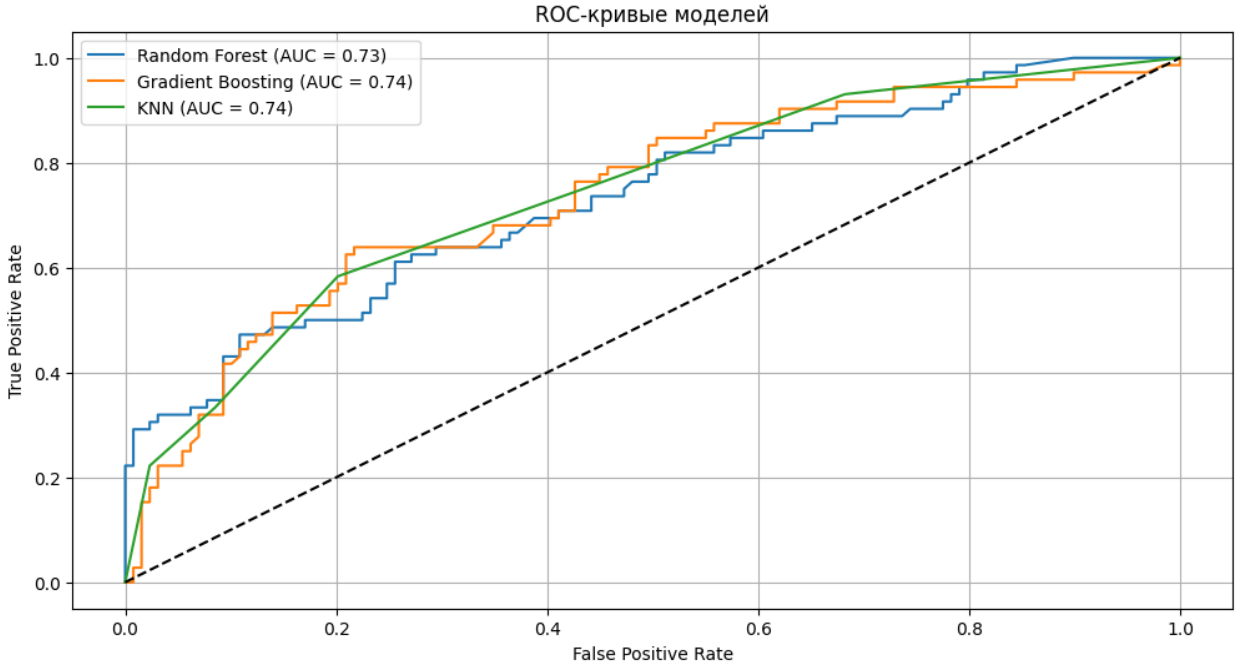
Для задачи классификации сначала немного преобразую данные: создаю порог в виде значения, равного 8, заполняю NaN и нормализую данные.

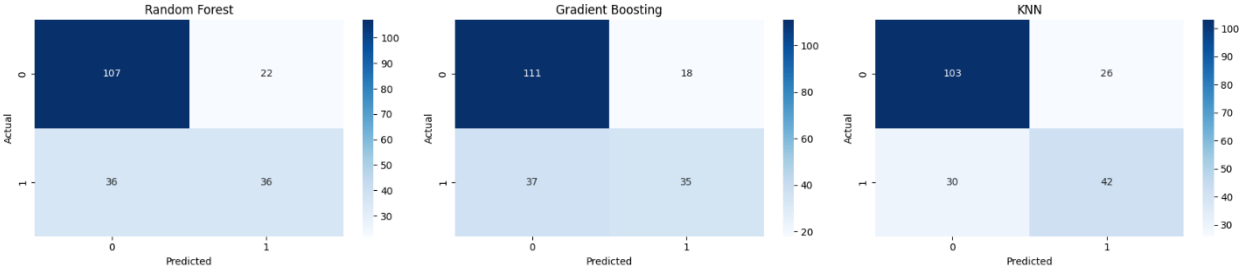
Далее проверяю, как работают разные модели классификации с этими данными, чтобы выбрать наиболее подходящую для обучения:

	Model	Accuracy	F1-score	ROC AUC
4	KNN	0.721393	0.600000	0.743379
2	Gradient Boosting	0.726368	0.560000	0.739987
1	Random Forest	0.711443	0.553846	0.733796
0	Logistic Regression	0.676617	0.532374	0.669089
3	SVM (RBF)	0.716418	0.528926	0.726906

По метрикам видно, что лучше всего справляются KNN, Gradient Boosting и Random Forest.

Визуализирую для анализа ROC-кривые и матрицы ошибок.





По визуализациям ROC-кривых и матриц ошибок можно сделать следующие выводы:

Качество классификации по ROC-кривым

Gradient Boosting и KNN показали наилучшее качество вероятностного предсказания (AUC ≈ 0.74).

Все три модели лучше случайного предсказания (AUC > 0.5), но остаются в умеренной зоне качества (0.7–0.8).

Матрицы ошибок (confusion matrices):

Метрика	**Random Forest**	**Gradient Boosting**	**KNN**
TP (1→1)	36	35	42
FN (1→0)	36	37	30
FP (0→1)	22	18	26
TN (0→0)	107	111	103

TP – правильно классифицированные положительные объекты
FN – пропущенные положительные объекты
FP – ложные срабатывания
TN – правильно классифицированные отрицательные объекты

Gradient Boosting дал наименьшее количество FP (18), что снижает вероятность ложных тревог.

KNN предсказал наибольшее количество настоящих положительных классов (TP=42), но ценой большего FP.

Random Forest демонстрирует сбалансированное поведение, но FN и TP равны (36).

Общие выводы

Gradient Boosting — осторожная модель: больше TN, меньше FP.

KNN — агрессивнее классифицирует 1-класс: больше TP, но и больше FP.

Random Forest — более симметричен по ошибкам (TP ≈ FN).

KNN показал лучшее среднее значение AUC и F1-score, но отличается нестабильностью (F1_std = 0.043).

Random Forest — наиболее стабильная модель (низкая дисперсия метрик), чуть уступает KNN по F1.

Обучаю базовую модель KNN, что в случае с этими данными оправдано, потому что метод даёт:

- быструю базовую оценку сложности задачи

KNN не требует обучения в классическом смысле, он просто хранит данные. Это позволяет быстро проверить, есть ли вообще сигнал в признаках для разделения классов.

- отсутствие предположений о распределении данных

KNN не строит гипотез (в отличие от логистической регрессии или SVM). Это позволяет оценить, насколько "естественно" разделимы классы в признаковом пространстве.

- визуальный ориентир для более сложных моделей

Если KNN даёт уже высокий F1- или AUC-результат, то сложные модели должны быть настроены так, чтобы минимум не ухудшить этот уровень.

- проверка качества признаков

KNN чувствителен к масштабам и шуму. Если работает плохо, это может сигнализировать о плохой нормализации или неинформативных признаках.

Получаю следующие метрики:

{'0': {'precision': 0.7744360902255639,

'recall': 0.7984496124031008,

'f1-score': 0.7862595419847328,

'support': 129.0},

'1': {'precision': 0.6176470588235294,
'recall': 0.5833333333333334,
'f1-score': 0.6,
'support': 72.0},
'accuracy': 0.7213930348258707,
'macro avg': {'precision': 0.6960415745245467,
'recall': 0.6908914728682171,
'f1-score': 0.6931297709923664,
'support': 201.0},
'weighted avg': {'precision': 0.7182728550964769,
'recall': 0.7213930348258707,
'f1-score': 0.7195397060499031,
'support': 201.0},
'AUC': np.float64(0.7433785529715763),
'conf_matrix': array([[103, 26],
[30, 42]])}

Метрика | Значение

Accuracy | 0.721
F1-score (class 0) | 0.786
F1-score (class 1) | 0.600
AUC | 0.743

Класс 0 (SI ≤ медианы):

Precision = 0.774 → 77.4% предсказаний класса 0 действительно соответствуют классу 0.
Recall = 0.798 → модель правильно находит 79.8% объектов класса 0.

Класс 1 (SI > медианы):

Precision = 0.618 → качество ниже, больше ложных срабатываний на класс 1.
Recall = 0.583 → покрытие слабее, модель пропускает часть объектов класса 1.

AUC ≈ 0.743

Это умеренное качество модели: она лучше случайного угадывания, но может быть улучшена за счёт:

- отбора признаков (feature selection),
- подбора гиперпараметров,
- использования ансамблей или бустинга

Применяю гиперпараметры вручную.

```
param_grid = {
  'n_neighbors': range(3, 21, 2),
  'weights': ['uniform', 'distance'],
  'metric': ['euclidean', 'manhattan']
}
```

Нахожу лучшие параметры с помощью KNeighborsClassifier.

Обучаю модель, смотрю результаты и метрики на тесте:

```
              precision    recall  f1-score   support

           0       0.75      0.78      0.76       129
           1       0.57      0.54      0.56        72

    accuracy                           0.69       201
   macro avg       0.66      0.66      0.66       201
weighted avg       0.69      0.69      0.69       201

AUC: 0.7287360034453058
```

Результат стал хуже. В данном случае нужно работать не с гиперпараметрами, а с фичами или ансамблями или бустингом.

В итоге, ансамбль:

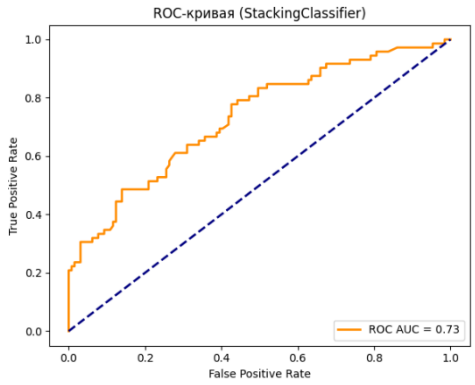
```
base_learners = [
  ('lr', LogisticRegression(max_iter=1000)),
  ('rf', RandomForestClassifier(random_state=42)),
  ('knn', KNeighborsClassifier(n_neighbors=9))
]
```

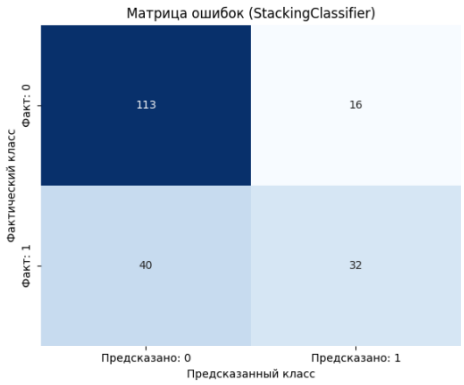
Дал такой же результат, как KNeighboursClassifier:

F1: 0.5333333333333333

AUC: 0.7314276485788115

Это неплохой результат, но, скорее всего нужно работать с признаками.





ROC-кривая и Матрица ошибок.

True Positive (TP) = 32 — модель правильно предсказала класс 1 (SI > 8).

False Negative (FN) = 40 — модель не распознала 1-й класс.

False Positive (FP) = 16 — модель ошибочно присвоила 1-й класс.

True Negative (TN) = 113 — модель корректно распознала класс 0.

Вывод:

Модель показывает умеренную способность распознавать высокий класс SI, но пока относительно слабый recall по классу 1 (много FN). Это можно улучшать через:

балансировку классов (например, class_weight),

дополнительный feature selection,

бустинг/бэггинг,

tuning порога вероятности.

Вот иерархия и взаимная вложенность понятий Data Science (DS), Deep Learning (DL), Machine Learning (ML), классического ML и других смежных областей:

---

1. Data Science (DS) – Наука о данных

- Самая широкая область, объединяющая методы работы с данными.

- Включает:

- Машинное обучение (ML)
- Глубокое обучение (DL)
- Статистику
- Визуализацию данных
- Обработку и инженерию данных (ETL, Feature Engineering)
- Big Data (Hadoop, Spark)
- Предметную экспертизу (Domain Knowledge)

---

2. Machine Learning (ML) – Машинное обучение (подмножество DS)

- Алгоритмы, которые учатся на данных без явного программирования.
- Классический ML (не-DL подходы):
  - Обучение с учителем (Supervised Learning)
    - Линейная регрессия
    - Деревья решений, Random Forest
    - SVM, XGBoost
  - Обучение без учителя (Unsupervised Learning)
    - Кластеризация (K-Means, DBSCAN)
    - PCA, t-SNE
  - Ансамблевые методы
  - Обучение с подкреплением (Reinforcement Learning)

- Глубокое обучение (DL) – подмножество ML

---

3. Deep Learning (DL) – Глубокое обучение (подмножество ML)

- Нейронные сети с множеством слоёв.
- Основные архитектуры:
  - CNN (свёрточные сети) – для изображений
  - RNN/LSTM/Transformer – для последовательностей (текст, временные ряды)

- GAN (генеративные сети)

- Autoencoders

- Требует больших данных и вычислительных ресурсов.

---

4. Сравнение ML и DL

Критерий	Классический ML	Deep Learning
Данные	Работает на малых данных	Требует больших данных
Фичи	Нужна feature engineering	Автоматическое извлечение фич
Интерпретация	Легче интерпретировать	Сложнее (чёрный ящик)
Вычисления	Меньше ресурсов	Требует GPU/TPU

---

5. Другие смежные области

- Computer Vision (CV) – использует DL (CNN) и классический ML.

- Natural Language Processing (NLP) – использует RNN, Transformer, BERT.

- Reinforcement Learning (RL) – может сочетать ML и DL (Deep Q-Networks).

- Big Data – фреймворки (Hadoop, Spark) для обработки данных, которые могут использоваться в DS.

Итоговая иерархия вложенности:

Data Science ⊃ Machine Learning ⊃ Deep Learning

Классический ML – это ML без DL.

DL – частный случай ML, который сам является частью DS.