Министерство науки и высшего образования Российской Федерации
Федеральное государственное автономное образовательное учреждение высшего образования
«Национальный исследовательский ядерный университет «МИФИ» (НИЯУ МИФИ)

«Институт интеллектуальных кибернетических систем»

кафедра 42

группа М24-525

Козырев Дмитрий Анатольевич

Курсовая работа

«Прогнозирование биологической активности химических соединений с использованием методов машинного обучения»

2025 г.

1. Цель работы

Первым этапом разработки лекарственных средств является выявление потенциально активных и безопасных молекул на ранней стадии, до проведения дорогостоящих in vitro и in vivo испытаний. В этой связи особую роль играют методы хемоинформатики и машинного обучения (in silico) , в частности методы количественного прогноза «структура–активность» (QSAR, Quantitative Structure–Activity Relationship), позволяющие прогнозировать фармакологические свойства соединений на основе их молекулярной структуры.

В данной курсовой работе рассматривается задача прогнозирования трёх ключевых биологических показателей, характеризующих действие химического соединения:

- **IC_{50} (полумаксимальная ингибирующая концентрация)** — концентрация вещества, необходимая для ингибирования 50% активности определённой биомишени. Используется для оценки **эффективности** соединения как потенциального лекарственного средства.
- **CC_{50} (полумаксимальная цитотоксическая концентрация)** — концентрация, при которой соединение вызывает гибель 50% клеток. Отражает **токсичность** вещества.
- **SI (Selectivity Index, индекс селективности)** — рассчитывается как отношение CC_{50} к IC_{50} и используется для оценки **селективности** действия соединения, то есть способности воздействовать на мишень без выраженного вреда для здоровых клеток. Чем выше значение SI, тем более селективным и потенциально безопасным считается соединение.

В рамках работы решаются как задачи **регрессии** — для количественного прогнозирования значений IC_{50}, CC_{50} и SI, так и задачи **классификации** — для определения принадлежности соединения к классу высокоэффективных, токсичных или селективных. Использование обеих постановок позволяет комплексно оценивать свойства соединений и применять модели в различных сценариях доклинических исследований.

Целью курсовой работы является разработка и оценка моделей машинного обучения (регрессионных и классификационных), способных предсказывать значения указанных биологических показателей на основе структурных молекулярных дескрипторов.

2. **Описание данных**

Для анализа представлен обширный набор признаков, включающий*:

1. Общие молекулярные дескрипторы
MolWt: молекулярная масса.

HeavyAtomCount: количество тяжёлых атомов (не включая водород).

NumValenceElectrons: общее количество валентных электронов.

NumRadicalElectrons: количество неспаренных (радикальных) электронов.

FractionCSP3: доля sp^3-гибридизованных атомов углерода.

TPSA: топологическая полярная поверхность, оценивает проницаемость через мембраны.

LabuteASA: аппроксимация доступной поверхности (ASA) по методу Labute. Этот метод используется для оценки площади молекулярной поверхности, доступной для взаимодействия с растворителем или другими молекулами.

QED: комплексная числовая оценка «лекарственности» молекулы, основанная на совокупности её химических и физических свойств. Помогает предсказать, насколько соединение подходит для разработки лекарственных препаратов. а практике QED рассчитывают с помощью специализированных программ и библиотек для химической информатики (например, RDKit)

SPS: предполагаемая сложность/стоимость синтеза (если доступна). Она помогает предсказать, насколько трудоёмким и дорогим будет получение молекулы в лаборатории или на производстве. Вычисляют с помощью специализированных программных инструментов и алгоритмов, которые анализируют молекулярную структуру и на основе статистических моделей, данных о реакциях и стоимости реагентов прогнозируют сложность и стоимость синтеза.

MolLogP: логарифм коэффициента распределения (гидрофобность).

MolMR: молекулярная рефрактивность (показатель поляризуемости).

2. Электронные дескрипторы

MaxPartialCharge / MinPartialCharge / MaxAbsPartialCharge / MinAbsPartialCharge: экстремальные значения частичных зарядов.

PEOE_VSA: группа дескрипторов, отражающих распределение зарядов , вычисленных методом уравнивания орбитальной электроотрицательности (PEOE).

EState_VSA: объединение информации о зарядовом состоянии и их топологическом расположении в молекуле..

MaxEStateIndex / MinEStateIndex / MaxAbsEStateIndex / MinAbsEStateIndex: экстремальные значения индекса электротопологического состояния.

3. Топологические дескрипторы

Chi0, Chi1, Chi2, ..., Chi4v: индексы связности Чи. отражающие молекулярную топологию и связанные с числом связей, типом атомов и степенью разветвления.

Kappa1, Kappa2, Kappa3: индексы Кьера. которые характеризуют форму молекулы, её компактность и степень разветвлённости.

HallKierAlpha: эмпирический дескриптор стерической насыщенности т.е. насколько молекула "заполнена" в пространстве.

BalabanJ: индекс связности, учитывающий длину путей и цикличность в молекуле. Позволяет оценить степень разветвленности и топологическую сложность структуры.

Ipc, AvgIpc, BertzCT: информационные и сложностные индексы, характеризующие структурную сложность молекулы на основе анализа её графа. Используются для количественной оценки разнообразия и запутанности структуры.

4. BCUT-дескрипторы

способ представить структуру молекулы в виде числового вектора, который учитывает как физико-химические свойства, так и структурные особенности. Считаются с использованием **взвешенной матрицы смежности**, построенной на основе структуры молекулы, и последующего извлечения **собственных значений** этой матрицы.

BCUT2D_MWHI / MWLOW: с учётом молекулярной массы.

BCUT2D_CHGHI / CHGLOW: по заряду.

BCUT2D_LOGPHI / LOGPLOW: по logP.

BCUT2D_MRHI / MRLOW: по молекулярной рефрактивности.

5. VSA-дескрипторы
VSA-дескрипторы (Van der Waals Surface Area) — это характеристики, описывающие, **как физико-химические свойства распределены по поверхности молекулы**, основанные на площади ван-дер-ваальсовых поверхностей атомов:

SMR_VSA1–10: связаны с молекулярной рефрактивностью.

SlogP_VSA1–12: связь с гидрофобностью (logP).

EState_VSA1–10: электротопология по поверхности.

PEOE_VSA1–14: связь с частичными зарядами.

6. Отпечатки (Morgan fingerprints)

Morgan fingerprints — это векторные представления молекул, которые кодируют их **структурные фрагменты** (окружения атомов) с помощью алгоритма, аналогичного **распространению по графу**. Часто используются для сравнения и поиска похожих соединений в химических базах данных.

FpDensityMorgan1, 2, 3: плотность битов при радиусах 1, 2 и 3 (нормализовано на число атомов).

7. Фрагментные дескрипторы

это числовые признаки, которые отражают **наличие или количество определённых химических групп или структурных фрагментов** в молекуле. Бывают как бинарные (есть/нет), так и счетные (количество повторений)

Фенолы: fr_phenol, fr_Ar_OH.

Амины: fr_NH2, fr_amine, fr_aniline.

Азосоединения: fr_azide, fr_azo, fr_diazo.

Галогены: fr_halogen, fr_alkyl_halide.

Барбитураты: fr_barbitur.

Нитро-соединения: fr_nitro, fr_nitro_arom.

Лактон/лактам: fr_lactone, fr_lactam.

Кольца: fr_benzene, fr_pyridine, fr_furan, fr_thiazole.

8. Структурные количественные дескрипторы

Описывают важные элементы молекулярной структуры, отражают потенциальные свойства молекул, влияющие на их фармакокинетику, реакционную способность и «лекарственность».

NumHAcceptors / NumHDonors: акцепторы/доноры водородных связей.

NumRotatableBonds: количество вращающихся связей.

NumAromaticRings / NumAliphaticRings / NumSaturatedRings: кольцевые структуры.

NumHeteroatoms: количество гетероатомов.

RingCount: общее количество колец.

Такой разнообразный набор признаков позволяет модели машинного обучения захватывать различные уровни молекулярной информации и выявлять сложные взаимосвязи между структурой соединения и его биологической активностью.

*Ссылки на источники информации:
- Маджидов Т.И. Введение в хемоинформатику: Компьютерное представление химических структур: учеб. пособие / Т.И. Маджидов, И.И. Баскин, И.С. Антипин, А.А. Варнек. – Казань: Казан. ун-т, 2013. – 174 с.

- Chemoinformatics: Basic Concepts and Methods
Share Icon, Thomas Engel (Editor), Johann Gasteiger (Editor)
ISBN: 978-3-527-33109-3 December 2018 608 pages

Выводы по обзору признаков и целевых переменных:

- Колонку **Unnamed: 0** (порядковый номер записи) необходимо удалить, даже если ее оставление приводит к улучшению метрик т.к. модель может запомнить конкретные номера записей и использовать их как "ключ", а не учиться на реальных закономерностях
- Значения **IC_{50}, CC_{50} и SI** и производные от них не могут использоваться для обучения моделей т.к. в реальных данных ни один из этих признаков не известен. Модель должна базироваться только на доступных и измеримых которые реально есть в момент прогнозирования.
- При классификации на основе **медианного значения IC_{50}, CC_{50} или SI** нельзя заранее вычислять медиану по всему датасету и использовать её в качестве порога для формирования целевой переменной. Такой подход приводит к утечке данных, так как информация о всей выборке становится известна на этапе обучения, что искажает оценку качества модели и приводит к завышенным метрикам.
- Предсказанные значения IC_{50} и CC_{50}, полученные с помощью регрессионных моделей, могут использоваться в качестве инженерных признаков. Эти предсказанные значения представляют собой аппроксимации, основанные на доступных независимых признаках, и при корректном построении не создают утечки данных.
- Однако важно, чтобы такие инженерные признаки, как $\log(IC_{50})$ и $\log(CC_{50})$, **рассчитывались строго на обучающей выборке, а затем применялись к валидационной и тестовой.** В противном случае существует риск косвенной утечки, так как предсказания могут частично включать информацию из тестовых данных, что приводит к завышению метрик модели.

- Дескриптор **SPS** можно исключить, поскольку он не соответствует поставленной задаче и не несёт дополнительной информативности в рамках целевой постановки.
- Дескрипторы **FpDensityMorganX** лучше удалить если неизвестно их происхождение. В некоторых базах и публикациях эти дескрипторы могут быть предрасчитаны с использованием или фильтрацией на основе активности. В нашем датасете уже присутствуют значения биологических показателей — IC50, CC50 и SI. С высокой вероятностью дескрипторы FpDensityMorganX могли быть рассчитаны либо отобраны с использованием информации об этих целевых переменных.

2. Цель построения моделей:

Целью является построение и применение моделей машинного обучения для решения задач высокопроизводительного скрининга (HTS) химических соединений, направленного на предварительное in silico предсказание их биологической активности (IC_{50}), токсичности (CC_{50}) и селективности (SI).

Высокопроизводительный скрининг (High-Throughput Screening, HTS) это метод, применяемый на ранних стадиях доклинических исследований для быстрой оценки большого числа соединений с целью выявления потенциально активных, нетоксичных и селективных кандидатов. Применение регрессионных и классификационных моделей позволяет существенно оптимизировать этот процесс.

Полученные модели могут быть использоваться для:

- **количественного предсказания** IC_{50}, CC_{50} и SI на основе структурных дескрипторов соединений (регрессия);
- **классификации соединений** на активные/неактивные, токсичные/нетоксичные и селективные/неселективные **по заданным порогам**.
  В данной работе в качестве порогов используются медианные значения показателей, однако пайплайн предусматривает и альтернативные пороги, например: $IC_{50} < 10$ μM, $CC_{50} > 50$ μM, $SI > 8$;
- **выделения ключевых структурных признаков**, оказывающих наибольшее влияние на активность и безопасность;
- **приоритизации химических соединений для дальнейшего синтеза и in vitro/in vivo тестирования** на основании предсказанных значений и принадлежности к положительным классам.

3. Тестируемые модели

3.1. Задачи регрессии

В исследовании были использованы следующие алгоритмы машинного обучения:

- **KNeighborsRegressor (KNN)** — ленивый алгоритм, прогнозирует по среднему ближайших соседей. Прост, но чувствителен к масштабу признаков и шуму.
- **Random Forest Regressor** — ансамбль деревьев решений с бэггингом. Устойчив к переобучению, хорошо работает с разнородными данными.
- **Gradient Boosting Regressor** — последовательное обучение деревьев на ошибках предыдущих. Высокая точность, но чувствителен к переобучению.
- **HistGradientBoosting Regressor** — оптимизированная реализация бустинга с гистограммами. Быстрее на больших данных.
- **AdaBoost Regressor** — адаптивный бустинг слабых моделей. Лучше на простых задачах, чувствителен к шуму.
- **XGBoost Regressor** — эффективная и масштабируемая реализация градиентного бустинга с регуляризацией.
- **LightGBM Regressor** — бустинг на основе листов дерева и гистограмм. Очень быстрый, хорошо работает с большим числом признаков.
- **CatBoost Regressor** — бустинг с автоматической обработкой категориальных признаков. Устойчив к переобучению
- **Составной регрессор (Stacking)** построен на основе моделей Random Forest, XGBoost, Gradient Boosting и CatBoost, с Linear Regression в качестве мета-модели. Это ансамбль деревьев с различными алгоритмами построения: от бэггинга (Random Forest) до различных реализаций бустинга. Такие модели существенно различаются по механизму обучения и, как правило, дают разнообразные ошибки — что критически важно для стекинга, поскольку позволяет избежать дублирования информации между базовыми

моделями.
Linear Regression выступает в роли мета-модели — простого и интерпретируемого метода, который линейно комбинирует выходы базовых моделей, извлекая из них синергетический эффект.

- **MLPRegressor (scikit-learn) -** полносвязная нейросеть (многослойный перцептрон) для регрессии. Способна аппроксимировать сложные нелинейные зависимости, но требует настройки архитектуры и чувствительна к масштабированию данных
- **Модель на PyTorch:**Реализована собственная полносвязная регрессионная сеть.

3.2. Задачи классификации

- **Logistic Regression** — линейный классификатор, хорошо работает при разделимости классов. Быстрый, интерпретируемый, устойчив к переобучению.
- **KNeighborsClassifier** — классификация по большинству ближайших соседей. Прост в реализации, но чувствителен к масштабу и размеру выборки.
- **Random Forest** — ансамбль деревьев с бэггингом. Устойчив к переобучению, хорошо работает с разнородными данными и признаками.
- **Gradient Boosting** — последовательное обучение слабых моделей на ошибках. Высокая точность, но требует настройки и аккуратного контроля переобучения.
- **HistGradientBoostingClassifier** — ускоренная реализация градиентного бустинга с гистограммами. Эффективен на больших и разреженных данных.
- **AdaBoost** — бустинг с адаптацией весов ошибочно классифицированных объектов. Прост и эффективен на небольших наборах данных.
- **XGBoost** — мощный бустинг с регуляризацией, встроенной обработкой пропусков и высокой скоростью. Один из лидеров в соревнованиях.
- **LightGBM** — бустинг по листам с использованием гистограмм. Очень быстрый и масштабируемый, особенно при большом количестве признаков.
- **CatBoost** — бустинг с поддержкой категориальных признаков без явного кодирования. Высокая точность, работает «из коробки».
- **MLPClassifier (Многослойный персептрон)** — нейронная сеть с несколькими слоями. Хорошо аппроксимирует сложные зависимости, чувствителен к параметрам и масштабу данных.
- **GaussianNB (Наивный байесовский классификатор)** — простая и быстрая модель, основанная на предположении о нормальном распределении признаков. Хорош для текстов и базовых задач.
- **SVM (Support Vector Machine)** — эффективен в задачах с высокой размерностью, особенно на малых выборках. Хорошо работает с нелинейными границами (при использовании ядра).
- **StackingClassifier** — ансамбль, объединяющий предсказания нескольких базовых моделей с помощью мета-классификатора. В данной реализации использует Random Forest, Gradient Boosting, XGBoost и CatBoost в качестве базовых моделей, а Logistic Regression — в роли мета-модели. Комбинирует преимущества разных алгоритмов, повышая точность и устойчивость к переобучению.

4. Метрики и критерии подбора параметров моделей

- **Для задач регрессии в качестве основной метрики использовался коэффициент детерминации (R^2),** так как он отражает долю объясненной дисперсии зависимой переменной моделью и позволяет оценить, насколько хорошо модель описывает данные в целом.
- **В качестве дополнительной метрики использовалась MAE (средняя абсолютная ошибка),** которая позволяет оценить среднюю величину ошибки в тех же единицах, что и предсказываемая переменная. MAE применялась исключительно для сравнения моделей **на одном и том же тестовом наборе данных** и не использовалась при отборе модели.
- **Для задач классификации по медианному значению в качестве основной метрики выбрана accuracy**, поскольку классы сбалансированы и доля правильно классифицированных объектов дает объективную оценку качества модели. В качестве дополнительных метрик использовались ROC AUC и F1-score, поскольку они позволяют учесть возможные погрешности при незначительном смещении баланса классов, а также лучше характеризуют поведение модели в случае ошибок I и II рода.
- Для задачи классификации по значению **SI>8** предположительно основной метрикой является **precission**, т.е. мы стремимся к минимизации ложных срабатываний (то есть случаев, когда модель ошибочно предсказывает SI > 8 для неподходящих образцов).

Recall применяют когда важно найти как можно больше активных объектов и пропуски опасны — предположительно менее актуально в данной задаче.

- Некоторые методы оптимизации моделей, особенно на этапе финальной доработки, могут приводить к улучшению одних метрик, но одновременно — к ухудшению других. Для комплексной оценки качества модели важно сохранять баланс между метриками.

Поэтому донастройка моделей осуществлялась с учётом следующего принципа: если улучшение основной метрики сопровождается более значительным снижением второстепенных показателей, такие изменения признавались нецелесообразными и отклонялись.

Например, калибровка вероятностей методом Platt может повышать accuracy за счёт улучшения пороговых предсказаний, но при этом, если калибровка ухудшает способность модели правильно ранжировать объекты по вероятности принадлежности к классу, может снижать ROC AUC.

- Все метрики на конечном этапе рассчитывались на отложенной тестовой выборке. Поскольку состав тестовой выборки влияет на итоговые значения (иногда существенно), возможны колебания метрик. Однако все методы настройки (в том числе fine-tuning и доработка признаков) направлены на улучшение показателей **на тех же тестовых данных** при стабильных условиях. При этом важно, чтобы метрики кросс-валидации на обучающих данных не сильно расходились с тестовыми — это служит критерием устойчивости модели.

5. Методика построения моделей (основные этапы)

Построение моделей осуществлялось в несколько последовательных этапов:

а. Анализ и первичная предобработка данных:
Удаление признаков с единственным значением, заполнение пропущенных значений, исключение нерелевантных или избыточных признаков.

б. Создание инженерных признаков (feature engineering)
На данном этапе формировались новые признаки, основанные на доменной специфике и статистических преобразованиях. Результирующий датасет был сохранён в файл df.csv **(основной датасет).**

в. Расширенная предобработка данных:
Для df.csv были применены методы бинаризации, обрезки выбросов и трансформации Yeo-Johnson. Полученные данные сохранены в файл df_bin.csv.

г. Удаление признаков на основе корреляционного и дисперсионного анализа для основного датасета:
Из df.csv удалялись признаки на основе корреляционного и дисперсионного анализа. Полученные данные сохранены в файл df_cut.csv.

д. Удаление признаков на основе корреляционного и дисперсионного анализа для предобработанного датасета:
Аналогичная фильтрация признаков была проведена для df_bin.csv. Результат сохранён в файл df_bin_cut.csv.

Таким образом были сформированы четыре датасета (df.csv, df_bin.csv, df_cut.csv, df_bin_cut.csv) **с различной степенью и типом предобработки.**

е. Базовое обучение моделей:
На всех четырёх датасетах (с разным типом предобработки данных) обучались различные модели (регрессии и классификации) с базовыми параметрами.

ж. Выбор оптимальной модели и предобработки для каждой задачи:
Для каждой из задач (регрессия IC_{50}, CC_{50}, SI; классификация IC_{50}, CC_{50}, SI) выбиралась модель с наилучшими метриками и подходящим вариантом предобработанных данных.

з. Подбор гиперпараметров (fine tuning):
Для каждой выбранной модели и соответствующего датасета проводился двухэтапный подбор гиперпараметров:

1. **RandomizedSearchCV** — первичный широкий поиск.
2. **GridSearchCV** — уточняющий поиск в пределах оптимальных областей.

и. Финальная доработка признаков:
На завершающем этапе для повышения метрик качества осуществлялась индивидуальная работа с признаками и структурой данных без изменения модели. Методы подбирались вручную в зависимости от задачи. Например:

- Применение **ранговой трансформации**, **перцентильного ранжирования** и **биннинга**, если они давали прирост в метриках.
- Итеративный отбор **наиболее значимых признаков** (по feature importance) с отсечением остальных, если это давало прирост в метриках.
- Для задач **бинарной классификации** — **калибровка вероятностей** с помощью метода **Platt (sigmoid)**.

Основной целью этапа fine tuning являлось повышение качества предсказаний именно на независимой тестовой выборке, а не на внутренней кросс-валидации.

к. Автоматизация предобработки и обучения:

Для каждой из моделей предусмотрена возможность объединения этапов предобработки и обучения в единый пайплайн с использованием sklearn.Pipeline. Структура проекта организована с разнесением кода предобработки, отбора признаков и моделей по отдельным модулям, что позволяет легко интегрировать их в пайплайны.

Реализация пайплайнов в рамках данной работы не представлена, однако архитектура проекта адаптирована под автоматизацию и может быть реализована при необходимости (например, для переноса в продакшн или подключения AutoML-инструментов).

Оценка предложенного подхода к построению моделей:

Данная методика построения моделей была выбрана по причине того. что на одних данных необходимо построить несколько различных моделей для различных целевых переменных.

У такого подхода есть как преимуществами, так и недостатками.

Преимущества:

- **Гибкость и адаптивность к задаче:**
  Каждая модель строится индивидуально под конкретную целевую переменную с подбором подходящих признаков, типа предобработки и архитектуры. Это позволяет учесть специфику данных и добиться максимально возможной точности.
- **Вариативность и сравнимость подходов:**
  Использование нескольких версий датасетов с разной степенью предобработки позволяет провести всесторонний анализ и выбрать наилучшее сочетание данных и модели. Это повышает обоснованность выбора финальных решений.
- **Точная настройка (fine-tuning) как гиперпараметров модели, так и представления данных под конкретную задачу**.
- **Контроль на каждом этапе:**
  Раздельное хранение данных и моделей после каждого этапа обработки обеспечивает прозрачность, удобство анализа и возможность быстро воссоздать или изменить конкретный этап без повторного запуска всего процесса.

Недостатки:

- **Риск overfitting и влияние data drift:**

Чрезмерная оптимизация моделей под конкретную выборку может ухудшить их обобщающую способность. Особенно это критично при дрейфе данных (data drift) — когда новые данные отличаются по распределениям или характеристикам, например, из-за изменений в экспериментальных условиях.

- **Слабая универсальность и переносимость:**
  Каждая модель строится на уникальном наборе признаков и типе предобработки, что затрудняет применение единой схемы или шаблона для новых задач. Это снижает повторное использование кода и мешает тиражированию решений.
- **Ограниченная автоматизация:**
  Из-за высокой доли ручной настройки (от отбора признаков до постобработки) затруднено создание универсального автоматического пайплайна. Это усложняет масштабирование проекта, интеграцию с AutoML-системами и эксплуатацию в продакшн.

3. EDA:

3.1. **Предварительная обработка данных**

а. Удаление признаков с единственным уникальным значением

- Такие признаки не несут информативности и не участвуют в различении объектов, поэтому подлежат исключению. Из оригинальных данных удалено 18 признаков

б. **Заполнение пропущенных значений (импутация)**

Доля пропусков в данном датасете не велика (3 пропуска), но для построения пайплайна необходимо определиться с методикой заполнения пропусков:

- Не желательно заполнять пропуски в числовых дескрипторах медианой соответствующего всего столбца — это утечка данных
- Не желательно заполнять пропуски пропуски в числовых дескрипторах медианой соответствующего всего столбца — это утечка данных.
- Импутация происходит **только на тренировочных данных** (например, вычисляется медиана на тренировочной выборке).
- Затем эта же медиана применяется для заполнения пропусков в тестовых данных (без повторного вычисления медианы на тесте).

в. **Проверка и фильтрация отрицательных значений:**

- Значительная часть дискрипторов по своей природе **не могут принимать отрицательные значения**, поскольку они отражают количественные, структурные или физико-химические характеристики, которые либо ноль, либо положительны:

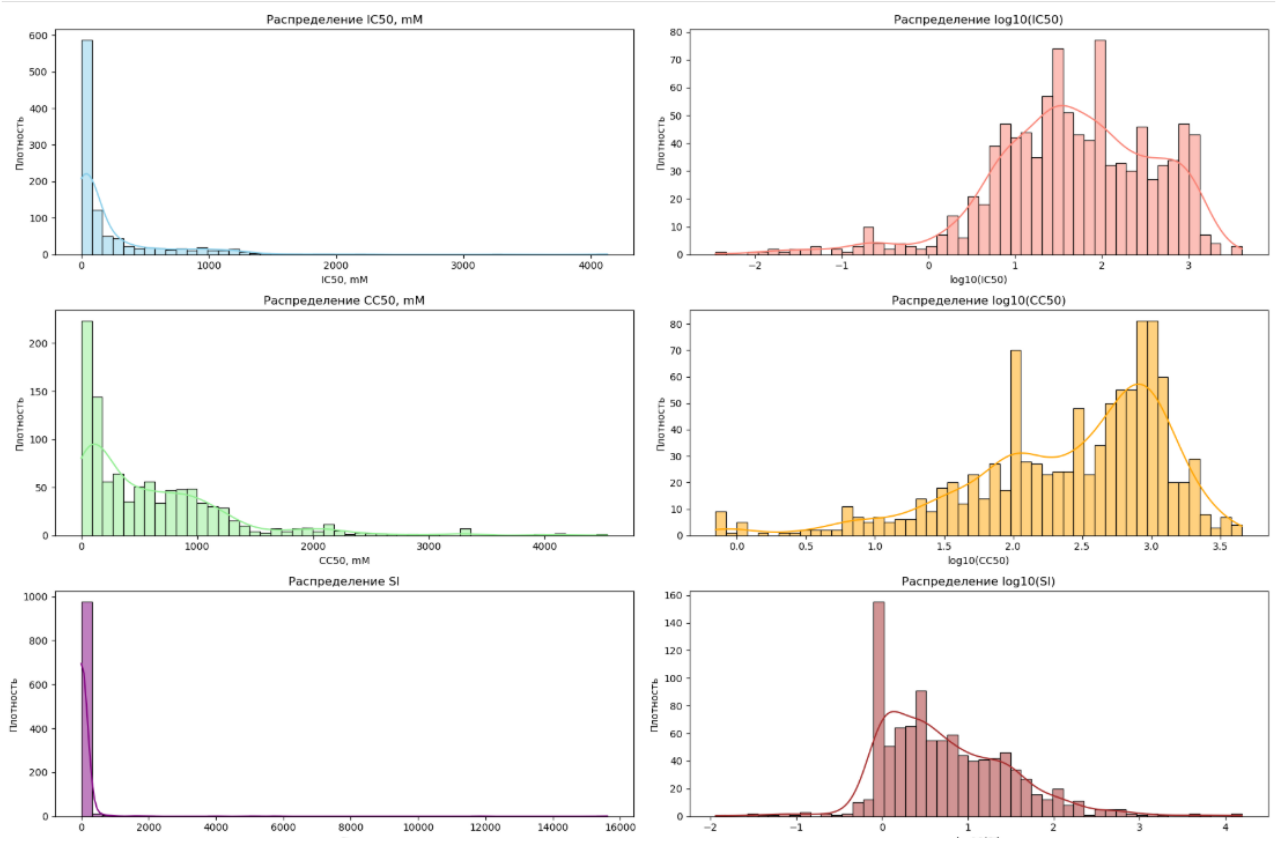
Дескрипторы, не принимающие отрицательных значений:

- Общие: MolWt, MolMR, TPSA, LabuteASA, QED, FractionCSP3, HeavyAtomCount, NumValenceElectrons, NumRadicalElectrons
- Электронные: MaxAbsPartialCharge, MinAbsPartialCharge, MaxAbsEStateIndex, MinAbsEStateIndex, PEOE_VSA1–14, EState_VSA1–10
- Топологические: Chi0–Chi4, Chi1v–Chi4v, Kappa1–3, HallKierAlpha, BalabanJ, Ipc, AvgIpc, BertzCT
- VSA-дескрипторы: SMR_VSA1–10, SlogP_VSA1–12, PEOE_VSA1–14, EState_VSA1–10
- Morgan fingerprints: FpDensityMorgan1–3
- Фрагментные: все дескрипторы вида fr_* (например, fr_phenol, fr_NH2, fr_nitro и т.д.)
- Структурные: NumHAcceptors, NumHDonors, NumRotatableBonds, NumAromaticRings, NumAliphaticRings, NumSaturatedRings, NumHeteroatoms, RingCount
- Целевые переменные IC_{50}, CC_{50}, SI также не могут быть отрицательными

Поэтому необходимо произвести анализ и удалить или скорректировать наблюдения с отрицательными значениями в перечисленных признаках и целевых переменных, поскольку они физически и биологически не имеют смысла.

Вышеописанную предобработку (п.п. а, б, в) необходимо включить в пайплайн, даже если в обучающем датасете таких данных нет, поскольку в новых они могут появиться из-за ошибок измерений, преобразований или внешних источников, и система должна быть устойчива к таким ситуациям.

3.2. **Изучение распределений целевых переменных и значимых признаков**



Анализ распределения целевых переменных:

Изучение распределения значений целевых переменных (например, IC_{50}) показало выраженную левостороннюю асимметрию:

- Основная масса значений сосредоточена в нижней части диапазона (например, для IC_{50} — ниже 200 мкМ), в то время как распределение имеет длинный правый хвост, доходящий до 4000 мкМ.
- Такое распределение затрудняет применение алгоритмов, чувствительных к форме распределения, особенно линейных моделей, которые предполагают симметричное или нормальное распределение ошибок.

После применения логарифмического преобразования:

- Распределения стали более симметричным и приближённым к нормальному.
- Выбросы и редкие экстремальные значения утратили доминирующее влияние.
- Признаки стали более подходящим для линейного моделирования, а также для моделей, чувствительных к масштабу и распределению (например, регрессии, SVM, kNN).

Выводы:

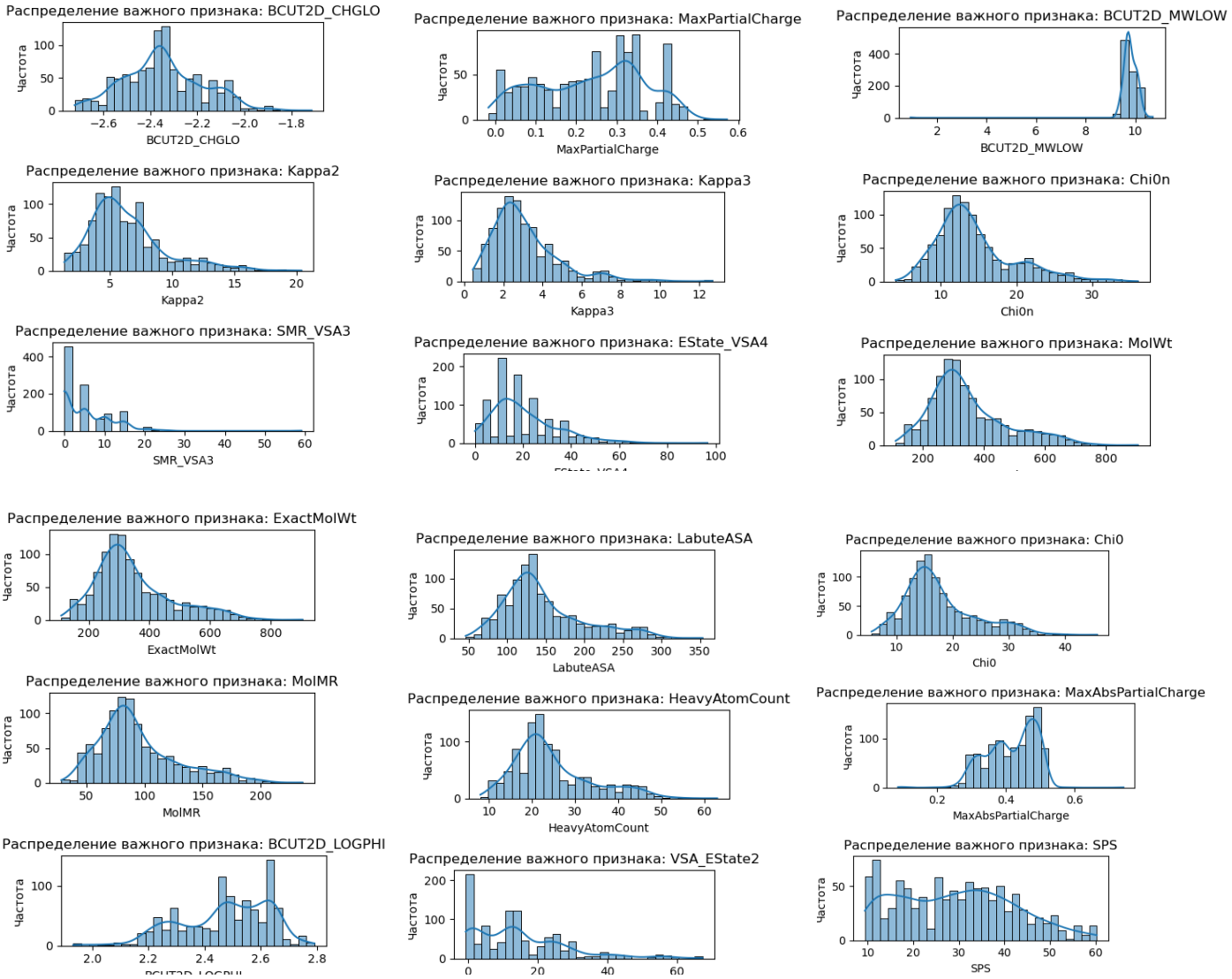
- Для задач регрессии предпочтительно использовать логарифм целевой переменной — это улучшает стабильность и предсказательную способность моделей.
- Для задач классификации все еще необходимо использовать оригинальные (непреобразованные) значения целевых переменных, но логарифмы признаков (не целевых переменных) можно включать в качестве инженерных признаков для повышения информативности моделей.

Анализ распределения значимых признаков:

- Проанализируем распределения наиболее значимых признаков (как наиболее релевантных), чтобы выявить ключевые закономерности в данных.
- Значимые признаки для целевых переменных определим с помощью модели RandomForestRegressor
- Например для целевой переменной log(IC50):

BCUT2D_CHGLO 0.007335 ; Kappa2 0.007319; SMR_VSA3 0.003027; MaxPartialCharge 0.002835;

Kappa3 0.002777; Estate_VSA4 0.002709; BCUT2D_MWLOW 0.002632; Chi0n 0.002466; MolWt 0.002089; ExactMolWt 0.001769; MolMR 0.001608; BCUT2D_LOGPHI 0.001192; LabuteASA 0.001102; HeavyAtomCount 0.001057; VSA_Estate2 0.000993; Chi0 0.000955; MaxAbsPartialCharge 0.000801; SPS 0.000724; fr_sulfide 0.000716; Chi1n 0.000684; HeavyAtomMolWt 0.000682; VSA_Estate9 0.000671; AvgIpc 0.000664; Chi1 0.000653; BertzCT 0.000645; Chi1v 0.000606; Kappa1 0.000592; Chi2n 0.000537



Вывод:

- Большинство признаков оказались информативными для прогнозирования $\log10(IC_{50})$.
- Однако некоторые из них, такие как BCUT2D_MWLOW (в приведённом примере), обладают выраженной асимметрией распределения или наличием выбросов, что может снижать эффективность отдельных моделей.
- Для таких признаков требуется статистическая трансформация признаков, включающая, например:
    - логарифмическое преобразование,
    - ранговую нормализацию (rank transformation),
    - трансформацию Yeo–Johnson.

  Эти методы позволяют привести распределения к более "нормальной" форме и снизить влияние выбросов.
- Подобные преобразования особенно важны для моделей, чувствительных к распределению признаков и масштабу данных (например, линейной регрессии, SVM, KNN), так как они предполагают определённые статистические свойства входных данных.
- В то же время для моделей градиентного бустинга и случайного леса такие трансформации не являются критичными, поскольку эти алгоритмы устойчивы к масштабу и распределению признаков.

4. Feature Engineering

На этапе инженерии признаков (feature engineering) были применены как статистическая трансформация признаков, так и создание новых агрегированных и производных признаков для извлечения более глубоких закономерностей из исходных молекулярных дескрипторов.

а. **Создание признаков с доменной интерпретацией**

Созданы новые инженерные признаки как отношения или комбинации исходных дескрипторов, обладающие химико-биологической интерпретацией (Feature engineering на основе доменных знаний, т.е. на основе знаний предметной области):

- **NumHAcceptors / NumHDonors**
  Отражает баланс между возможностью молекулы принимать и отдавать водородные связи.
  Значимость:
  Влияет на растворимость в водной и липидной среде.
  Связан с проникающей способностью через клеточные мембраны.
  Важен для связывания с белками — может влиять на аффинность и специфичность взаимодействий.
- **MolWt / TPSA**
  Соотношение массы молекулы к её топологической полярной поверхности.
  Значимость:
  Используется как показатель способности проникать через липидные мембраны.
  Высокое значение — может указывать на большую гидрофобность и высокую мембранную проницаемость.
  Низкое значение — свидетельствует о высокой полярности, что может улучшать растворимость, но снижать проницаемость.
- **FractionCSP3 / Kappa1**
  Индикатор насыщенности (sp^3-гибридизация) относительно структурной разветвлённости.
  Значимость:
  Высокая доля sp^3-гибридизации связана с трёхмерной структурой и может улучшать селективность к белкам-мишеням.
  Связь с формой молекулы — важна для предсказания фармакокинетических свойств.
- **NumRotatableBonds / HeavyAtomCount**
  Нормированная оценка гибкости молекулы.
  Значимость:
  Высокая гибкость может ухудшать связывание с целевыми сайтами (из-за потерь энтропии).
  Однако умеренная гибкость может быть полезной для адаптивности в активных центрах.
- **FractionCSP3 × MolWt**
  Комплексный показатель насыщенности и массы.
  Значимость:
  Высокое значение может указывать на объёмную, насыщенную структуру.
  Низкое значение — на плоские, ароматические молекулы.
  Помогает дифференцировать потенциально «лекарственные» структуры от менее перспективных.
- **NumHAcceptors + NumHDonors**
  Общее число возможных водородных связей.
  Значимость:
  Прямая связь с водорастворимостью и потенциалом связывания с белками.
  Один из ключевых факторов в правилах Липински для оценки «лекарственности».

б. **Агрегация признаков по группам дескрипторов**

Для групп дескрипторов были рассчитаны статистики: mean, max, sum, std:

- Chi (Chi0–Chi4v) — топологические индексы связности.
- Kappa1–3 — индексы формы и разветвлённости молекулы.
- PEOE_VSA* — заряды по поверхности молекулы (электронная плотность).
- Estate_VSA* — отражают зарядовое состояние и топологию.
- SMR_VSA* — связь с молекулярной рефрактивностью.

- SlogP_VSA* — распределение гидрофобности на поверхности молекулы.
- - для MaxAbsPartialCharge, MinAbsPartialCharge, MaxPartialCharge, MinPartialCharge ввели среднее и разброс: Среднее этих показателей по группе молекул помогает понять общий уровень и тенденцию зарядов в датасете — например, средний максимальный заряд по всем молекулам.Разброс (дисперсия, стандартное отклонение) показывает вариативность: насколько сильно отличаются молекулы по этим характеристикам.

Такая агрегация позволяет извлечь обобщённую информацию из группы взаимосвязанных дескрипторов, а также при необходимости уменьшить размерность.

в. **Добавление полиномиальных признаков**

Для тор 5 признакам (по результатам SHAP анализа) были добавлены полиномиальные признаки:

BCUT2D_MWLOW x Kappa2 Усиливает совместное влияние молекулярной массы и формы молекулы (Kappa2 — топологический индекс гибкости/извилистости).

BCUT2D_MWLOW x VSA_EState4 Усиливает влияние массы вместе с электронным состоянием (EState4) и объемом поверхности (VSA) определённых участков молекулы.

BCUT2D_MWLOW_x Chi1n Усиливает совместный эффект массы и первого хирального индекса Chi1n, отражая влияние стереохимии.

Kappa2 x BCUT2D_MRLOW Усиливает влияние гибкости вместе с распределением атомов с низкой молекулярной радикальной величиной.

Kappa2 x VSA_EState4 Усиливает взаимодействие гибкости с электронной структурой и объемом поверхности.

BCUT2D_MRLOW x VSA_EState4 Усиливает взаимодействие между молекулярной радикальной величиной и электронной поверхностью.

BCUT2D_MRLOW x Chi1n Усиливает совместное влияние молекулярной радикальной величины и хиральности.

VSA_EState4 x Chi1n Усиливает взаимодействие электронной структуры с хиральностью.

- **По результатам предобработки и Feature Engineering данные были сохранены в файл df.csv для дальнейшей работы с данными**.

в. **Размерность датафрейма после первичной предобработки и создания инженерных признаков:**

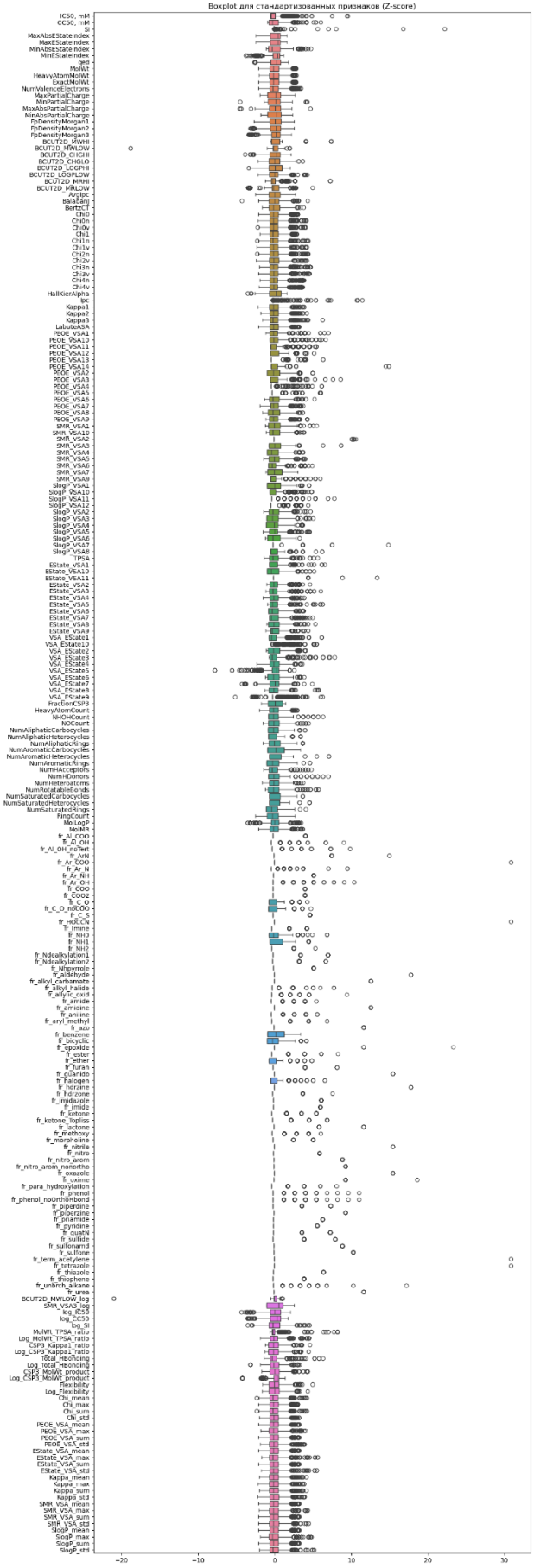
- 1001 строк, 235 признаков, добавлено 33 инженерных признака
- датафрейм сохранен в файл **df.csv**

- **Предварительные выводы для выбора дальнейшей стратегии моделирования**:
  При относительно небольшом объёме данных (1001 наблюдение) и высокой размерности признакового пространства (235 признаков) применение глубоких нейронных сетей нецелесообразно из-за высокого риска переобучения и недостатка обучающих данных, что подтверждается дальнейшими экспериментами.
- В таких задачах предпочтительно использовать классические методы машинного обучения, такие как Random Forest, градиентный бустинг, логистическую регрессию, SVM и другие алгоритмы, которые менее чувствительны к ограниченному количеству данных и большому числу признаков.
- Для табличных данных такого объема использование GPU зачастую неэффективно, особенно для моделей, оптимизированных для CPU (например, CatBoost). В нашей работе это подтвердилось: время передачи данных между CPU и GPU превышало время обучения на CPU, что связано с дополнительными накладными расходами при копировании данных.
- Наличие значительного количества признаков (включая целевые переменные) с выраженной асимметрией распределения и выбросами может снижать эффективность моделей, чувствительных к статистическим характеристикам входных данных (таких как линейные модели, SVM, KNN), даже несмотря на применение трансформаций. Возможно более предпочтительными будут модели основанные на деревьях решений (tree-based ensembles).

г. Бинаризация и обработка выбросов

- **Анализ выбросов по Z-оценке**

- Анализ выбросов в признаках проводился с использованием **Z-оценки (стандартизованных значений)** и сохранен в файле zscore_outliers_report.txt , а также zscore_boxplot.png
Z-оценка интуитивно более понятна: она показывает, насколько далеко значение отклоняется от среднего в терминах стандартных отклонений и не требует предварительного знания распределения.



- **Бинаризация**

- Некоторые признаки были преобразованы в двоичный (бинарный) формат, при котором все ненулевые значения заменялись на 1, а нулевые — на 0.

- Критерий бинаризации**:**
Признак подвергался бинаризации, если более 90% его значений равны нулю (значение 90% является настраиваемым гиперпараметром). Это соответствует признакам с крайне редкими ненулевыми значениями, что характерно, например, для индикаторов структурных фрагментов молекул.
- Цель бинаризации**:**
  - Повысить интерпретируемость редких признаков.
  - Упростить структуру данных.
  - Снизить влияние выбросов.
  - Повысить эффективность некоторых моделей машинного обучения, чувствительных к распределению признаков.

- **Обработка выбросов методом межквартильного размаха (IQR)**

Для числовых признаков в датафрейме была выполнена процедура замены выбросов, основанная на методе межквартильного размаха (IQR).

- Для каждого числового признака были рассчитаны первый (Q1) и третий (Q3) квартили.
- Вычислено значение межквартильного размаха:
  IQR = Q3 – Q1.
- Определены границы допустимых значений:
  • Нижняя граница: Q1 – 1.5 × IQR (гиперпараметр)
  • Верхняя граница: Q3 + 2,5 × IQR (гиперпараметр, с увеличенным множителем))
- Значения за пределами этих границ классифицировались как выбросы.
- Обнаруженные выбросы заменялись:
  • Значения ниже нижней границы — на нижнюю границу,
  • Значения выше верхней границы — на верхнюю границу.

Цель замены выбросов:
• Снижение влияния экстремальных значений на обучение моделей.
• Сглаживание распределения признаков при сохранении структуры данных.
• Снижение риска переобучения и повышение стабильности предсказаний.

Особенность подхода:

- В отличие от классического симметричного метода (где обе границы устанавливаются на уровне 1.5 × IQR), была использована асимметричная настройка:
• Нижняя граница оставалась стандартной: Q1 – 1.5 × IQR,
• Пробовлись различные варианты верхней границы, больше стандартной:
(Q3 + 2,5 × IQR); (Q3 + 15 × IQR)
Это позволило уменьшить чувствительность к большим значениям (например, в случае редких признаков или логарифмированных метрик), но не искажать информацию о возможных редких, но валидных наблюдениях
- Обработка выбросов не была включена в пайплайн предварительной обработкит(п.3.1), поскольку для ряда моделей (например, деревья решений и их ансамбли) не требуется принудительная фильтрация выбросов — они устойчивы к экстремальным значениям за счёт своей структуры. Кроме того, включение обработки выбросов в автоматический pipeline может привести к потере потенциально значимой информации и искажению распределений при обработке новых данных, особенно в случае редких, но валидных наблюдений
После выполнения описанных преобразований полученный датафрейм был сохранён в файл **df_bin.csv** для дальнейших этапов построения моделей.

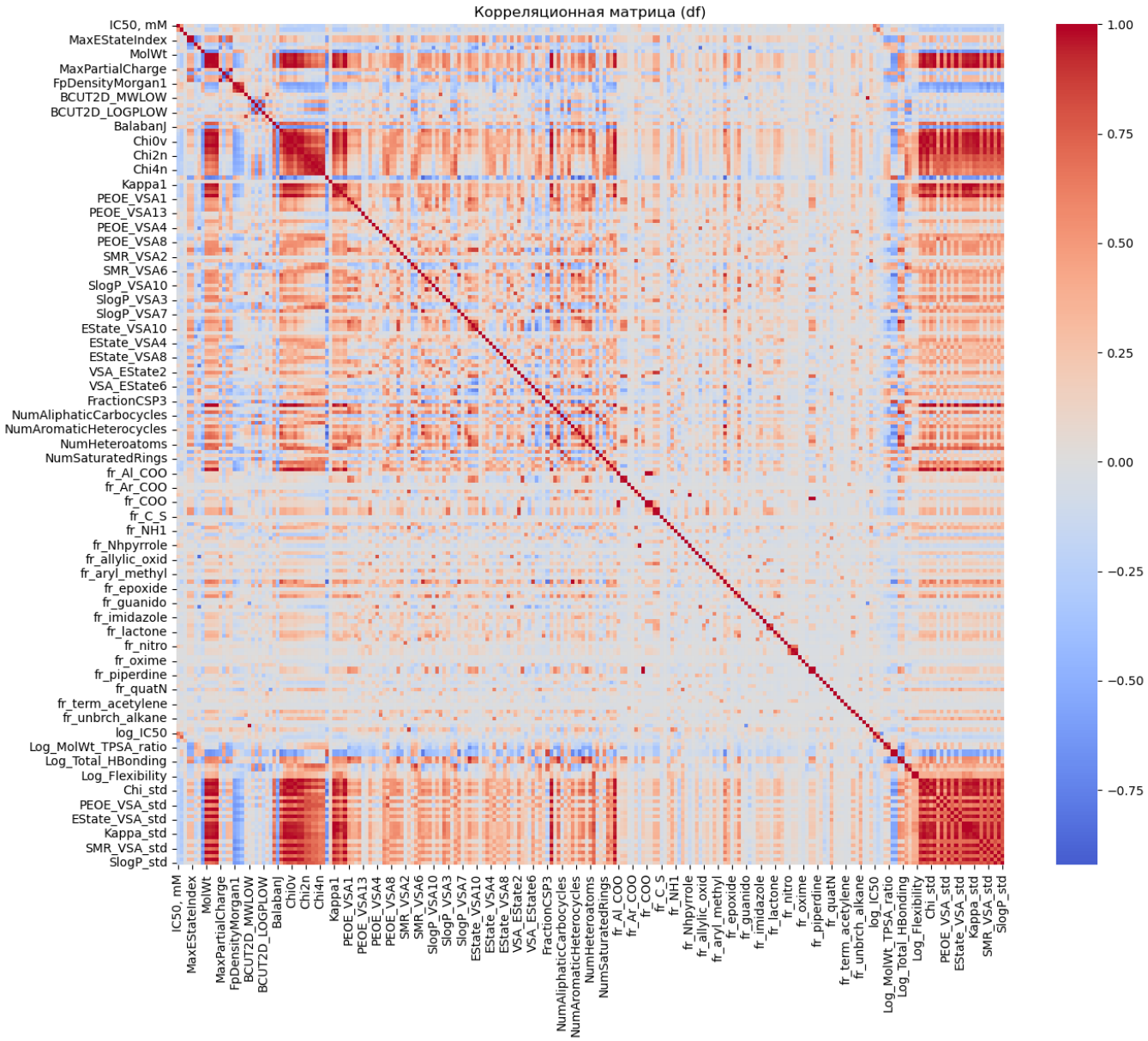
д. Корреляционный и дисперсионный анализ

- **Основные цели и задачи анализа**

- **Корреляционный анализ** позволил определить степень линейной взаимосвязи между числовыми признаками, а также между самими признаками. Это помогло выявить пары сильно коррелированных признаков (корреляция > 0.9 — гиперпараметр), которые могут создавать проблему мультиколлинеарности и влиять на стабильность моделей.

- **Отбор признаков с низкой дисперсией** выявил признаки с практически постоянными значениями, которые не вносят полезной информации для модели и были рекомендованы к удалению.
- На основании проведённого анализа был сформирован новый датасет, из которого удалены:
  - признаки с низкой дисперсией (σ^2<1e-4 – гиперпараметр),
  - избыточные признаки с высокой корреляцией.

- Цель - улучшить качество и стабильность моделей машинного обучения за счёт:
  - снижения избыточности данных,
  - устранения влияния мультиколлинеарности,
  - исключения малоинформативных признаков,
  - повышения интерпретируемости и эффективности последующего анализа.



Преобразования на основе корреляционного и дисперсионного анализа были проведены **отдельно**:
- д**ля основного датасета df.csv** (оригинальный предобработанный датасет с добавленными инженерными признаками). Измененные данные сохранены в файл **df_cut.csv**
- **для датасета df_bin.csv** (с обработанными выбросами и бинаризацией). Измененные данные сохранены в файл **df_cut_bin.csv**

Также для возможности последующего анализа были сохранены следующие файлы:

df_analysis_report.txt — подробный текстовый отчёт**;**

df_correlation_matrix.png — визуализация матриц корреляции;

report_df.txt, report_df_bin.txt, report_df_cut.txt и report_df_cut_bin.txt содержат сведения об изменениях, внесённых в данные по сравнению с исходным датасетом

В частности, информация о добавленных и удаленных признаках:

===== df и df.bin =====
Удалённые признаки: ['NumRadicalElectrons', 'SMR_VSA8', 'SPS', 'SlogP_VSA9', 'Unnamed: 0', 'fr_N_O', 'fr_SH', 'fr_azide', 'fr_barbitur', 'fr_benzodiazepine', 'fr_diazo', 'fr_dihydropyridine', 'fr_isocyan', 'fr_isothiocyan', 'fr_lactam', 'fr_nitroso', 'fr_phos_acid', 'fr_phos_ester', 'fr_prisulfonamd', 'fr_thiocyan']

Добавленные признаки: ['BCUT2D_MWLOW_log', 'CSP3_Kappa1_ratio', 'CSP3_MolWt_product', 'Chi_max', 'Chi_mean', 'Chi_std', 'Chi_sum', 'EState_VSA_max', 'EState_VSA_mean', 'EState_VSA_std', 'EState_VSA_sum', 'Flexibility', 'Kappa_max', 'Kappa_mean', 'Kappa_std', 'Kappa_sum', 'Log_CSP3_Kappa1_ratio', 'Log_CSP3_MolWt_product', 'Log_Flexibility', 'Log_MolWt_TPSA_ratio', 'Log_Total_HBonding', 'MolWt_TPSA_ratio', 'PEOE_VSA_max', 'PEOE_VSA_mean', 'PEOE_VSA_std', 'PEOE_VSA_sum', 'SMR_VSA3_log', 'SMR_VSA_max', 'SMR_VSA_mean', 'SMR_VSA_std', 'SMR_VSA_sum', 'SlogP_max', 'SlogP_mean', 'SlogP_std', 'SlogP_sum', 'Total_HBonding', 'log_CC50', 'log_IC50', 'log_SI']

===== df_cut =====
Удалённые признаки:['BertzCT', 'Chi0', 'Chi0n', 'Chi0v', 'Chi1', 'Chi1n', 'Chi1v', 'Chi2n', 'Chi2v', 'Chi3n', 'Chi3v', 'Chi4n', 'Chi4v', 'ExactM olWt', 'FpDensityMorgan2', 'FpDensityMorgan3', 'HallKierAlpha', 'HeavyAtomCount', 'HeavyAtomMolWt', 'Kappa1', 'Kappa2', 'Kappa3', 'LabuteASA', 'MaxAbsPartialCharge', 'MaxEStateIndex', 'MinAbsPartialCharge', 'MolMR', 'NOCount', 'NumAromaticCarbocycles', 'NumHAcceptors', 'NumHDonors', 'NumHeteroatoms', 'NumRadicalElectrons', 'NumSaturatedCarbocycles', 'NumValenceElectrons', 'SMR_VSA8', 'SPS', 'SlogP_VSA11', 'SlogP_VSA6', 'SlogP_VSA9', 'Unnamed: 0', 'VSA_EState6', 'fr_Al_OH_noTert', 'fr_COO', 'fr_COO2', 'fr_C_O', 'fr_C_O_noCOO', 'fr_N_O', 'fr_Nhpyrrole', 'fr_SH', 'fr_azide', 'fr_barbitur', 'fr_benzene', 'fr_benzodiazepine', 'fr_diazo', 'fr_dihydropyridine', 'fr_isocyan', 'fr_isothiocyan', 'fr_lactam', 'fr_nitro_arom_nonortho', 'fr_nitroso', 'fr_phenol', 'fr_phenol_noOrthoHbond', 'fr_phos_acid', 'fr_phos_ester', 'fr_prisulfonamd', 'fr_thiocyan']

Добавленные признаки:['CSP3_Kappa1_ratio', 'CSP3_MolWt_product', 'EState_VSA_max', 'Flexibility', 'Log_CSP3_MolWt_product', 'Log_MolWt_TPSA_ratio', 'MolWt_TPSA_ratio', 'PEOE_VSA_max', 'SMR_VSA3_log', 'SMR_VSA_max', 'SlogP_max', 'log_CC50', 'log_IC50', 'log_SI']

===== df_cut_bin =====

Удалённые признаки: ['BertzCT', 'Chi0', 'Chi0n', 'Chi0v', 'Chi1', 'Chi1n', 'Chi1v', 'Chi2n', 'Chi2v', 'Chi3n', 'Chi3v', 'Chi4n', 'Chi4v', 'EState_VSA11', 'ExactMolWt', 'FpDensityMorgan2', 'FpDensityMorgan3', 'HallKierAlpha', 'HeavyAtomCount', 'HeavyAtomMolWt', 'Kappa1', 'Kappa2', 'Kappa3', 'LabuteASA', 'MaxAbsPartialCharge', 'MaxEStateIndex', 'MinAbsPartialCharge', 'MolMR', 'NOCount', 'NumAromaticCarbocycles', 'NumHAcceptors', 'NumHDonors', 'NumHeteroatoms', 'NumRadicalElectrons', 'NumSaturatedCarbocycles', 'NumValenceElectrons', 'PEOE_VSA13', 'PEOE_VSA4', 'PEOE_VSA5', 'SMR_VSA2', 'SMR_VSA8', 'SPS', 'SlogP_VSA6', 'SlogP_VSA7', 'SlogP_VSA9', 'Unnamed: 0', 'VSA_EState10', 'VSA_EState6', 'VSA_EState9', 'fr_Al_COO', 'fr_Al_OH', 'fr_Al_OH_noTert', 'fr_ArN', 'fr_Ar_COO', 'fr_Ar_NH', 'fr_Ar_OH', 'fr_COO', 'fr_COO2', 'fr_C_O_noCOO', 'fr_C_S', 'fr_HOCCN', 'fr_Imine', 'fr_NH2', 'fr_N_O', 'fr_Ndealkylation1', 'fr_Ndealkylation2', 'fr_Nhpyrrole', 'fr_SH', 'fr_aldehyde', 'fr_alkyl_carbamate', 'fr_alkyl_halide', 'fr_allylic_oxid', 'fr_amide', 'fr_amidine', 'fr_aniline', 'fr_aryl_methyl', 'fr_azide', 'fr_azo', 'fr_barbitur', 'fr_benzene', 'fr_benzodiazepine', 'fr_diazo', 'fr_dihydropyridine', 'fr_epoxide', 'fr_ester', 'fr_furan', 'fr_guanido', 'fr_hdrzine', 'fr_hdrzone', 'fr_imidazole', 'fr_imide', 'fr_isocyan', 'fr_isothiocyan', 'fr_ketone', 'fr_ketone_Topliss', 'fr_lactam', 'fr_lactone', 'fr_methoxy', 'fr_morpholine', 'fr_nitrile', 'fr_nitro', 'fr_nitro_arom', 'fr_nitro_arom_nonortho', 'fr_nitroso', 'fr_oxazole', 'fr_oxime', 'fr_para_hydroxylation', 'fr_phenol', 'fr_phenol_noOrthoHbond', 'fr_phos_acid', 'fr_phos_ester', 'fr_piperdine', 'fr_piperzine', 'fr_priamide', 'fr_prisulfonamd', 'fr_pyridine', 'fr_quatN', 'fr_sulfide', 'fr_sulfonamd', 'fr_sulfone', 'fr_term_acetylene', 'fr_tetrazole', 'fr_thiazole', 'fr_thiocyan', 'fr_thiophene', 'fr_unbrch_alkane', 'fr_urea']

Добавленные признаки:['CSP3_Kappa1_ratio', 'CSP3_MolWt_product', 'EState_VSA_max', 'Flexibility', 'MolWt_TPSA_ratio', 'PEOE_VSA_max', 'SMR_VSA_max', 'SlogP_max', 'log_CC50', 'log_IC50', 'log_SI']

5. **Выбор базовой модели (baseline)**

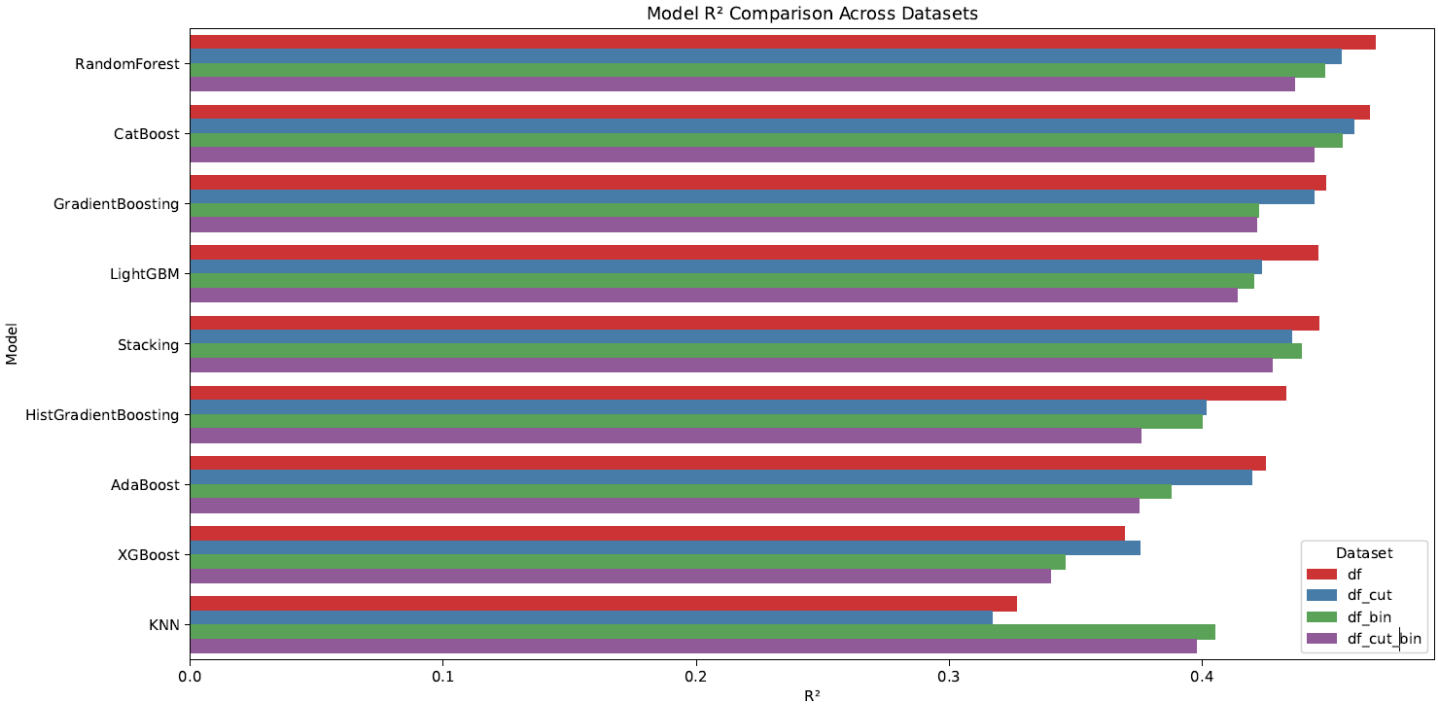
Каждая из моделей, указанных в пункте 3, была обучена на всех наборах данных, перечисленных в пункте 4. По результатам оценки качества на основе соответствующих метрик для каждой задачи была выбрана наилучшая модель в сочетании с наиболее подходящим датафреймом.

Следует отметить, что масштабирование (для тех моделей, где это необходимо), должно проводиться исключительно на обучающей выборке (train), а затем применяться к валидационной и тестовой выборкам — чтобы избежать утечки данных.

5.1. Задачи регрессии

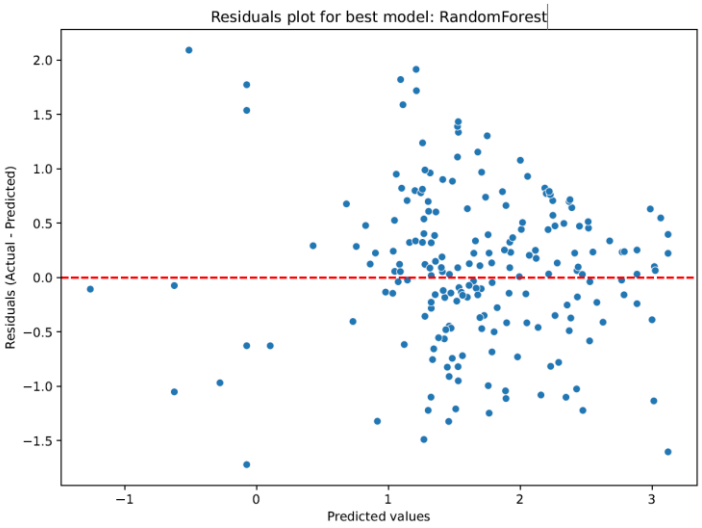
а. Для целевой переменной log10(IC50)

на этапе baseline модели обучались без добавления предсказанного CC50 (как инженерного признака)



Model	MAE	MSE	RMSE	MedAE	R²	Dataset
RandomForest	0,528345	0,461976	0,679688	0,427182	0,468511	Df
CatBoost	0,534156	0,465033	0,681933	0,427833	0,466121	Df
GradientBoosting	0,544193	0,479603	0,692534	0,445678	0,448804	Df
LightGBM	0,535067	0,481322	0,693774	0,416118	0,445765	Df
Stacking	0,545526	0,482832	0,694861	0,445636	0,446211	Df
HistGradientBoosting	0,53109	0,492504	0,701786	0,424208	0,432847	Df
AdaBoost	0,588497	0,501979	0,708505	0,541894	0,424742	Df
XGBoost	0,560157	0,545143	0,738338	0,446888	0,369199	Df
KNN	0,591624	0,587416	0,766431	0,4881	0,326658	Df
CatBoost	0,540027	0,470547	0,685964	0,436512	0,459752	df_cut
RandomForest	0,532973	0,474703	0,688987	0,411723	0,454882	df_cut
GradientBoosting	0,550917	0,485353	0,696673	0,458169	0,444263	df_cut
Stacking	0,550602	0,492588	0,701846	0,456379	0,435417	df_cut
LightGBM	0,546548	0,502101	0,708591	0,438311	0,423318	df_cut
AdaBoost	0,585463	0,50592	0,711281	0,547963	0,419409	df_cut
HistGradientBoosting	0,550477	0,519803	0,720973	0,433873	0,401434	df_cut
XGBoost	0,554804	0,539116	0,734245	0,445979	0,375383	df_cut
KNN	0,596337	0,598234	0,773456	0,489113	0,316844	df_cut
CatBoost	0,522788	0,438063	0,661863	0,427271	0,455239	df_bin

RandomForest	0,520595	0,443108	0,665663	0,415116	0,448219	df_bin
Stacking	0,533409	0,451888	0,672226	0,445541	0,43899	df_bin
GradientBoosting	0,541087	0,464944	0,681868	0,451435	0,422418	df_bin
LightGBM	0,52918	0,465528	0,682296	0,411865	0,420313	df_bin
KNN	0,540041	0,476342	0,690176	0,445288	0,404775	df_bin
HistGradientBoosting	0,525772	0,480487	0,693171	0,401331	0,400097	df_bin
AdaBoost	0,585641	0,493859	0,702751	0,537288	0,387686	df_bin
XGBoost	0,548498	0,521944	0,722457	0,42918	0,345758	df_bin
CatBoost	0,527677	0,446744	0,668389	0,437328	0,444036	df_cut_bin
RandomForest	0,524699	0,453241	0,673232	0,422742	0,436454	df_cut_bin
Stacking	0,542826	0,461109	0,67905	0,473289	0,427552	df_cut_bin
GradientBoosting	0,545064	0,466103	0,682718	0,452216	0,421282	df_cut_bin
LightGBM	0,534581	0,471416	0,686598	0,419682	0,413788	df_cut_bin
KNN	0,540453	0,481429	0,693851	0,438644	0,397625	df_cut_bin
HistGradientBoosting	0,542441	0,500828	0,707692	0,42514	0,3757	df_cut_bin
AdaBoost	0,593046	0,50429	0,710134	0,549485	0,374822	df_cut_bin
XGBoost	0,551497	0,527222	0,7261	0,434506	0,340127	df_cut_bin
Model	**MAE**	**MSE**	**RMSE**	**MedAE**	**R²**	**Dataset**



- Гетероскедастичность: Наблюдается некоторое увеличение разброса остатков при больших предсказанных значениях (справа на графике), что может указывать на неоднородность дисперсии

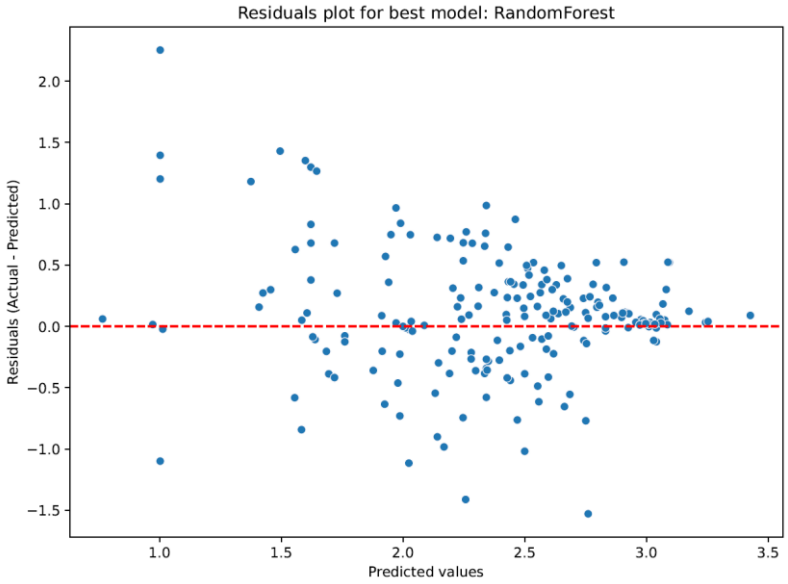
- **RandomForest** показал лучшие метрики R2=0,469 для целевой переменной **log10(IC50)** на датасете **df.csv**

б. Для целевой переменной log10(CC50)

на этапе baseline модели обучались без добавления предсказанного IC50 (как инженерного признака)

Model	**MAE**	**MSE**	**RMSE**	**MedAE**	**R²**	**Dataset**
RandomForest	0,384111	0,302102	0,549638	0,264784	0,406903	df
CatBoost	0,389858	0,305876	0,553061	0,263592	0,399656	df
LightGBM	0,375841	0,306317	0,553459	0,2387	0,397845	df
GradientBoosting	0,391753	0,30942	0,556256	0,276546	0,392351	df
HistGradientBoosting	0,385067	0,31561	0,561792	0,252776	0,379354	df
XGBoost	0,38642	0,327404	0,572192	0,253422	0,356596	df
Stacking	0,442892	0,337644	0,581071	0,353154	0,338854	df
AdaBoost	0,484042	0,370452	0,608648	0,406059	0,272349	df
KNN	0,455899	0,447115	0,668666	0,311285	0,113424	df

RandomForest	0,384322	0,299678	0,547429	0,259434	0,412591	df_cut
CatBoost	0,387521	0,303102	0,550547	0,259123	0,404452	df_cut
LightGBM	0,376051	0,304655	0,551956	0,253951	0,401236	df_cut
GradientBoosting	0,392456	0,309566	0,556387	0,266149	0,392548	df_cut
HistGradientBoosting	0,38233	0,313123	0,559574	0,256526	0,38483	df_cut
XGBoost	0,384443	0,322692	0,56806	0,240094	0,365381	df_cut
Stacking	0,436246	0,33056	0,574944	0,343007	0,352453	df_cut
AdaBoost	0,489424	0,382044	0,618097	0,38874	0,250809	df_cut
KNN	0,453262	0,442617	0,665294	0,304517	0,124464	df_cut
RandomForest	0,372146	0,266583	0,516317	0,265191	0,427748	df_bin
LightGBM	0,36252	0,268815	0,518473	0,247959	0,421701	df_bin
GradientBoosting	0,379847	0,270765	0,520351	0,278324	0,418286	df_bin
CatBoost	0,377281	0,272983	0,522478	0,2599	0,413145	df_bin
HistGradientBoosting	0,371359	0,277616	0,526893	0,255693	0,402994	df_bin
XGBoost	0,376639	0,298739	0,54657	0,235712	0,357917	df_bin
Stacking	0,429384	0,301577	0,54916	0,336728	0,352966	df_bin
AdaBoost	0,469455	0,32823	0,572914	0,409896	0,295032	df_bin
KNN	0,398114	0,334272	0,578163	0,273273	0,279562	df_bin
RandomForest	0,37264	0,26599	0,515742	0,26112	0,429105	df_cut_bin
CatBoost	0,374758	0,26882	0,518479	0,263397	0,422509	df_cut_bin
LightGBM	0,36474	0,27338	0,522858	0,242897	0,412757	df_cut_bin
GradientBoosting	0,384657	0,276474	0,525808	0,2797	0,406429	df_cut_bin
HistGradientBoosting	0,370969	0,276616	0,525943	0,254002	0,405941	df_cut_bin
Stacking	0,421696	0,291776	0,540163	0,327434	0,373546	df_cut_bin
XGBoost	0,37993	0,301062	0,548691	0,247582	0,353297	df_cut_bin
AdaBoost	0,467566	0,326496	0,571398	0,407923	0,299457	df_cut_bin
KNN	0,396338	0,331598	0,575846	0,271752	0,284713	df_cut_bin
Model	**MAE**	**MSE**	**RMSE**	**MedAE**	**R²**	**Dataset**

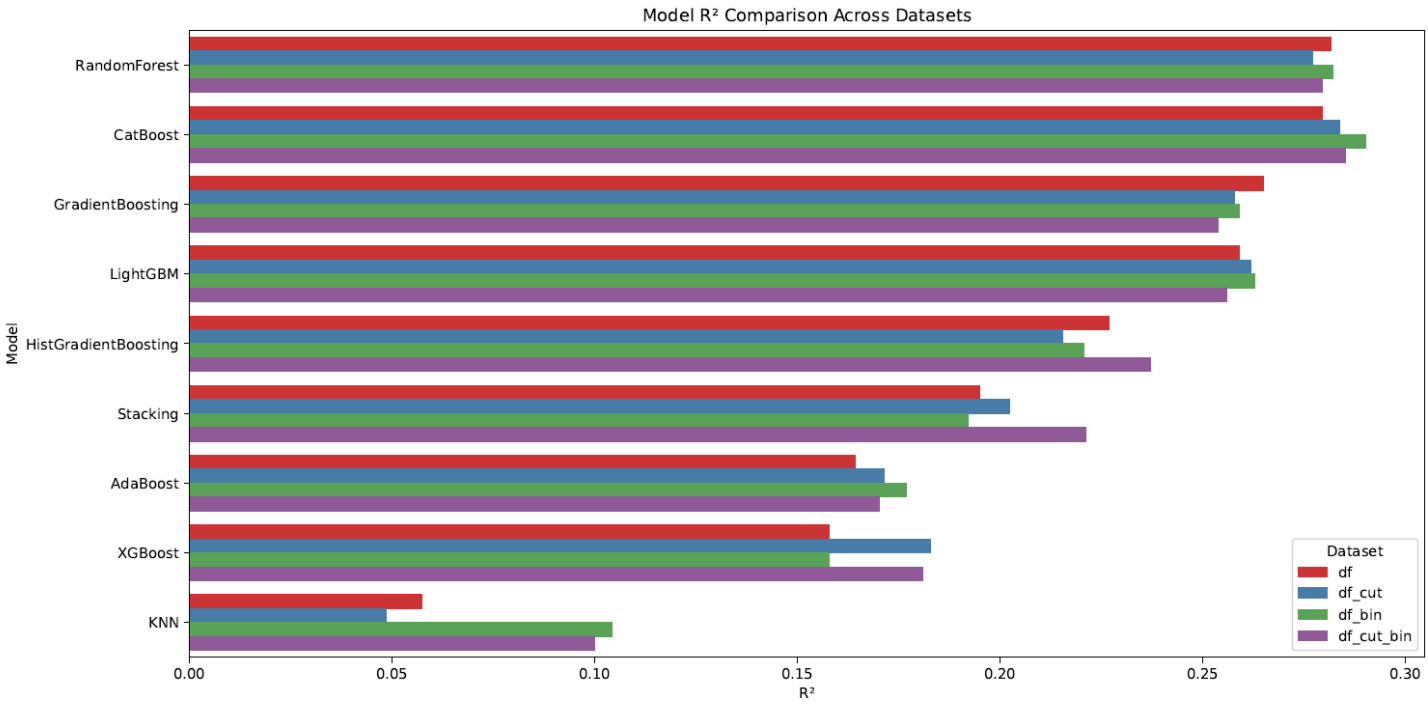


- Небольшая асимметрия: Наблюдается легкий дисбаланс - положительные остатки достигают больших значений, чем отрицательные

- RandomForest показал лучшие метрики R2=0,43 для целевой переменной **log10(IC50)** на датасете **df_cut_bin.csv**

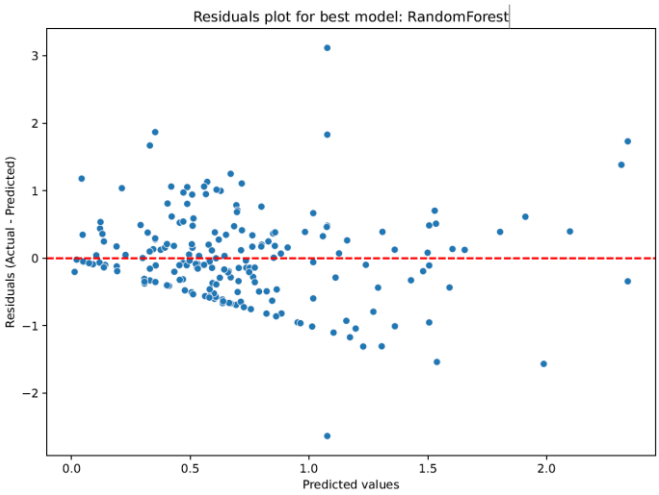
в. Для целевой переменной log10(SI)

.на этапе baseline модели обучались без добавления предсказанных IC50 и CC50 (как инженерных признаков)



Model	MAE	MSE	RMSE	MedAE	R²	Dataset
RandomForest	0,474145	0,40672	0,637746	0,374305	0,281727	df
CatBoost	0,479574	0,407809	0,6386	0,378908	0,279382	df
GradientBoosting	0,486482	0,414599	0,643894	0,382141	0,264841	df
LightGBM	0,477396	0,419027	0,647323	0,366072	0,258991	df
HistGradientBoosting	0,486262	0,437171	0,661189	0,375208	0,226906	df
Stacking	0,532163	0,459186	0,677633	0,472514	0,195009	df
AdaBoost	0,545477	0,472109	0,687102	0,485851	0,164212	df
XGBoost	0,502014	0,476039	0,689956	0,378898	0,157742	df
KNN	0,533304	0,53655	0,732496	0,420664	0,057371	df
CatBoost	0,477256	0,404922	0,636335	0,389141	0,283843	df_cut
RandomForest	0,474789	0,409132	0,639634	0,382577	0,277038	df_cut
LightGBM	0,479918	0,416497	0,645366	0,378333	0,261866	df_cut
GradientBoosting	0,487973	0,41926	0,647503	0,3829	0,257909	df_cut
HistGradientBoosting	0,493894	0,442222	0,664997	0,37036	0,215484	df_cut
Stacking	0,52788	0,453831	0,67367	0,472002	0,202235	df_cut
XGBoost	0,492585	0,46027	0,678432	0,372657	0,182879	df_cut
AdaBoost	0,536109	0,46821	0,684259	0,479795	0,171386	df_cut
KNN	0,53393	0,541989	0,736199	0,42167	0,048452	df_cut
CatBoost	0,473557	0,396746	0,629878	0,373714	0,290288	df_bin
RandomForest	0,472113	0,401847	0,633914	0,364834	0,282149	df_bin
LightGBM	0,477279	0,411566	0,641534	0,369983	0,262855	df_bin
GradientBoosting	0,486357	0,413396	0,642959	0,380702	0,259092	df_bin
HistGradientBoosting	0,486339	0,434816	0,659406	0,364573	0,220651	df_bin
Stacking	0,532283	0,455222	0,674702	0,474286	0,192001	df_bin
AdaBoost	0,536828	0,460089	0,678299	0,486264	0,176922	df_bin
XGBoost	0,502111	0,470408	0,685863	0,375874	0,157846	df_bin
KNN	0,51465	0,503497	0,709575	0,396261	0,10427	df_bin
CatBoost	0,474272	0,399054	0,631707	0,370389	0,285124	df_cut_bin
RandomForest	0,472413	0,403021	0,63484	0,371653	0,279577	df_cut_bin

LightGBM	0,479633	0,414875	0,644108	0,366723	0,255934	df_cut_bin
GradientBoosting	0,489461	0,416477	0,64535	0,38858	0,253691	df_cut_bin
HistGradientBoosting	0,483576	0,425461	0,652274	0,377367	0,237021	df_cut_bin
Stacking	0,520102	0,437516	0,66145	0,454294	0,221137	df_cut_bin
XGBoost	0,49174	0,454503	0,674169	0,367728	0,180843	df_cut_bin
AdaBoost	0,534613	0,464521	0,681557	0,478056	0,170295	df_cut_bin
KNN	0,510471	0,506477	0,711672	0,383547	0,100014	df_cut_bin
Model	**MAE**	**MSE**	**RMSE**	**MedAE**	**R²**	**Dataset**



- - Гетероскедастичность - заметно увеличение разброса остатков при больших предсказанных значениях (справа на графике)

- CatBoost показал лучшие метрики R2=0,29 для целевой переменной **log10(SI)** на датасете **df_bin.csv**

Выводы по baseline для задач регрессии:

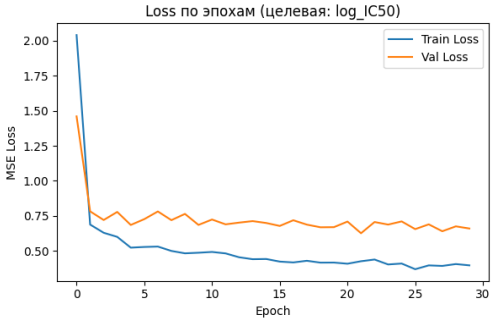
- Для различных целевых переменных наилучшие результаты демонстрировали разные модели и различные варианты предобработки данных.
- Если задача состоит в достижении максимальной точности для каждой конкретной задачи, необходимо подбирать модель и подход к предобработке индивидуально.
- Однако при анализе результатов можно заметить, что существует возможность подобрать такую комбинацию модели и предобработки, при которой снижение метрик по сравнению с оптимальными значениями будет минимальным для всех задач. В этом случае можно построить универсальный пайплайн для задач регрессии (если такая цель стоит).
- В нашем случае наилучшим компромиссным решением оказалась модель RandomForest, обученная на датасете df.csv.

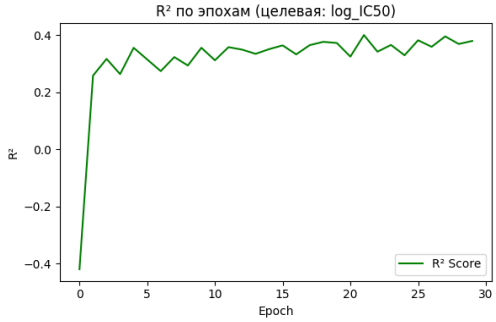
г. Отдельно была проведена оценка регрессионных моделей на базе **глубокой нейронной сети (DNN)**.

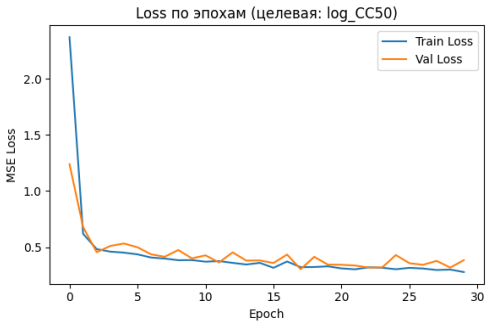
- MLPRegressor (scikit-learn):
Использована модель с семью скрытыми слоями: (1024, 512, 256, 128, 64, 32, 16) и функцией активации ReLU. Обучение проводилось с оптимизатором Adam, адаптивным learning rate и механизмом early stopping.

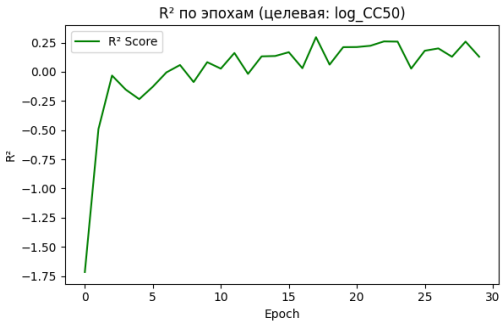
целевая переменная log10(IC50), для тестовой выборки: $R^2 = 0.4479$
целевая переменная log10(CC50), для тестовой выборки: $R^2 = 0.3739$
целевая переменная log10(SI), для тестовой выборки: $R^2 = 0.2008$

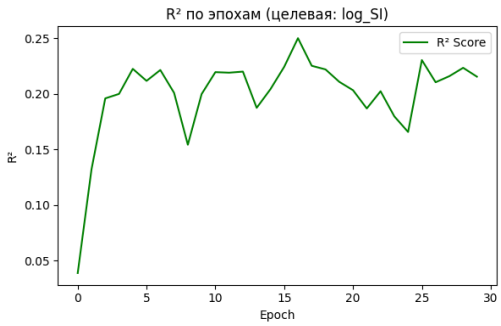
- Модель на PyTorch:
Реализована собственная полносвязная регрессионная сеть. Обучение выполнялось с использованием функции потерь MSELoss и оптимизатора Adam











Вывод по DNN:
ожидаемо метрики для DNN получились хуже, чем на лучших классических моделях, предположительно по следующим причинам:

- Небольшой объём данных и большое количество признаков:

Глубокие нейронные сети (особенно многослойные) требуют большого объёма данных для эффективного обучения. При недостатке данных они склонны к переобучению или не способны выявить устойчивые закономерности. Несмотря на регуляризацию и dropout, DNN могут подстраиваться под шум в условиях ограниченного обучающего множества.

Оценочное количество данных пригодных для DNN (зависит от архитектуры сети): количество строк должно превышать количество признаков в 10-100 раз, т.е. в нашем случае >10000 строк.

- Отсутствие структуры в признаках:

Нейросети лучше работают, когда признаки имеют пространственную или временную структуру (например, изображения или последовательности). В табличных данных, где признаки независимы и могут быть разного масштаба и связи между ними не выражены явно или линейно, классические модели (особенно ансамблевые деревья) часто работают лучше и стабильнее, поскольку они не требуют структурированной информации.

- Классические модели оптимизированы под табличные данные:

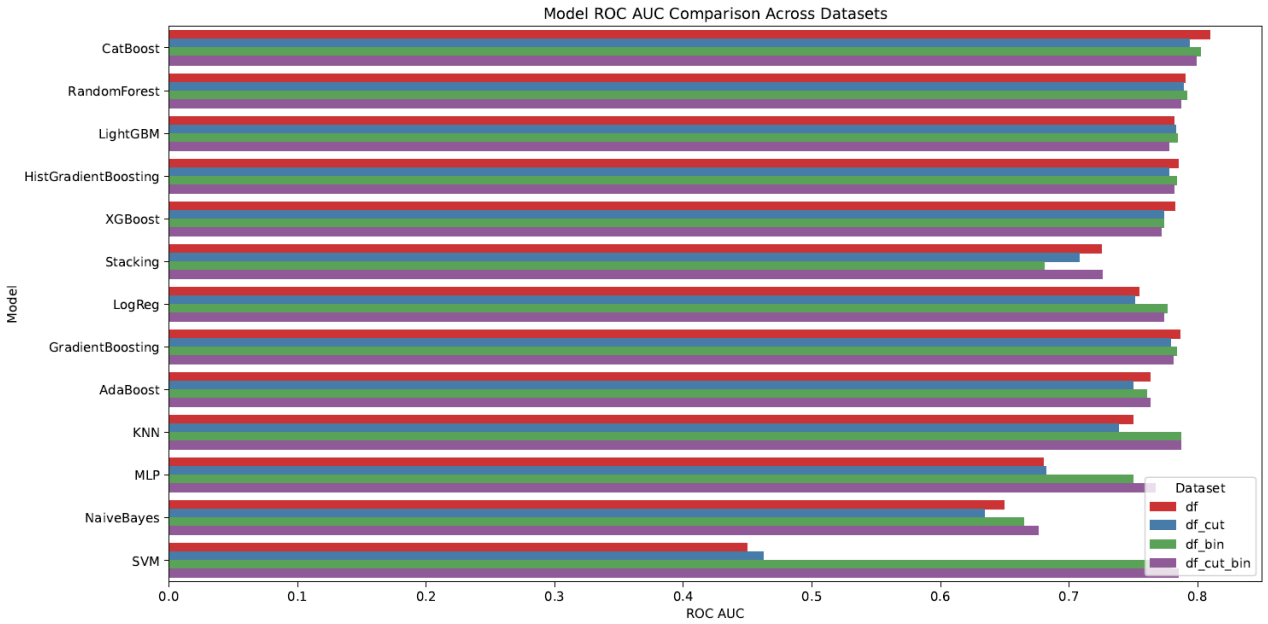
Алгоритмы типа Random Forest, Gradient Boosting, XGBoost и CatBoost изначально создавались и дорабатывались для задач на табличных данных — в отличие от нейросетей, которые эффективнее на других типах данных (изображения, звук, текст, видео, временные ряды.).

- Однако при значительном увеличении объёма данных (>10000 строк для рассматриваемой задачи), глубокие нейронные сети способны выявлять сложные и скрытые нелинейные зависимости, которые могут быть недоступны для классических моделей. Поэтому при повторном обучении модели на расширенном или обновлённом датасете (например, в условиях продакшн-среды) целесообразно рассматривать и DNN как одну из альтернатив, особенно в сочетании с тщательной настройкой архитектуры и регуляризации.

5.2. Задачи классификации

a. Бинарная классификация для медианного значения IC50

на этапе baseline модели обучались без добавления предсказанного CC50 (как инженерного признака)



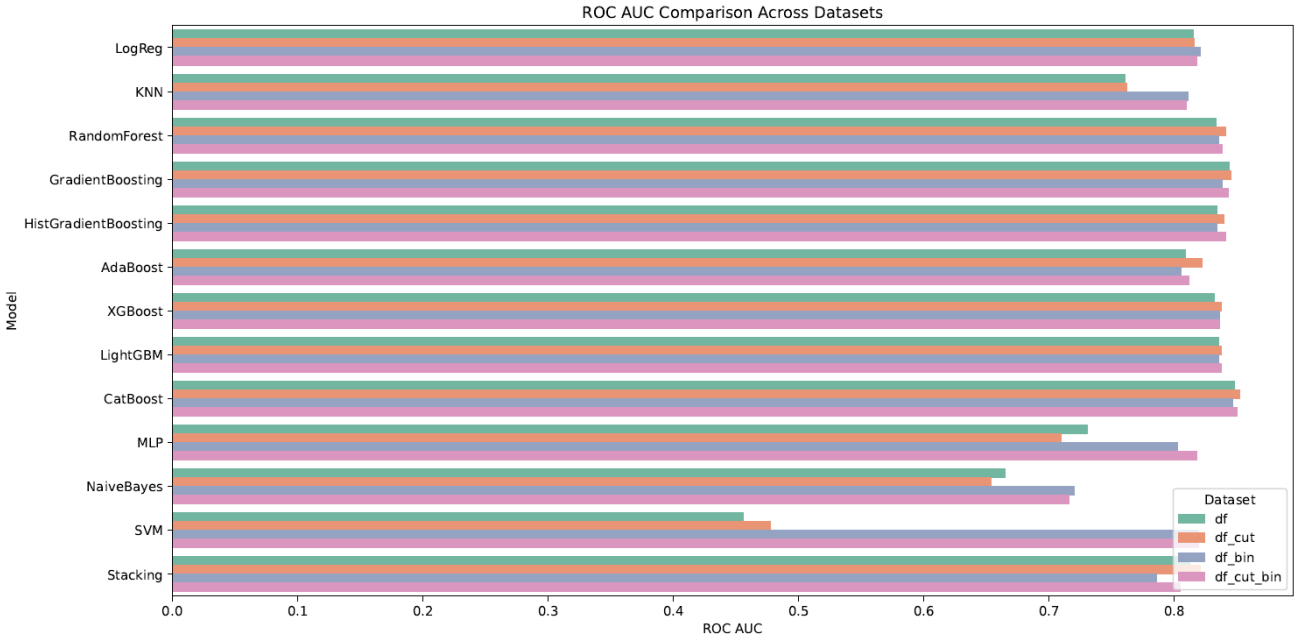
Model	Accuracy	Precision	Recall	F1	ROC AUC	Dataset
CatBoost	0,722269	0,727233	0,710977	0,716444	0,809784	df
RandomForest	0,721274	0,724998	0,710882	0,71631	0,790475	df
LightGBM	0,709279	0,708599	0,703241	0,704841	0,781313	df
HistGradientBoosting	0,708274	0,708752	0,700793	0,703084	0,78501	df
XGBoost	0,706289	0,706874	0,703034	0,702804	0,781948	df
Stacking	0,701289	0,708889	0,685566	0,693017	0,725164	df
LogReg	0,698259	0,703908	0,686009	0,691711	0,754485	df
GradientBoosting	0,702284	0,712695	0,67404	0,691278	0,78599	df
AdaBoost	0,691299	0,695469	0,683725	0,684478	0,762797	df
KNN	0,681308	0,677563	0,682439	0,678055	0,749468	df
MLP	0,643308	0,651166	0,637815	0,639451	0,679928	df
NaiveBayes	0,559512	0,575969	0,17017	0,17055	0,649261	df
SVM	0,535517	0,11746	0,193043	0,146053	0,449529	df
RandomForest	0,713274	0,711929	0,713885	0,711076	0,788722	df_cut
CatBoost	0,713259	0,710815	0,715115	0,710712	0,793359	df_cut
LightGBM	0,713269	0,711918	0,71328	0,710264	0,783239	df_cut
XGBoost	0,705274	0,707389	0,702263	0,702252	0,77362	df_cut
GradientBoosting	0,700294	0,699481	0,701547	0,698797	0,779161	df_cut
HistGradientBoosting	0,703274	0,706045	0,694989	0,69825	0,777458	df_cut
AdaBoost	0,689299	0,689205	0,689083	0,686394	0,74943	df_cut
LogReg	0,682289	0,682702	0,68227	0,679675	0,750804	df_cut
KNN	0,676303	0,665882	0,690175	0,675655	0,738559	df_cut
Stacking	0,667303	0,671278	0,651431	0,658244	0,707576	df_cut
MLP	0,658313	0,665695	0,659498	0,655002	0,68214	df_cut
NaiveBayes	0,547512	0,574286	0,138866	0,154276	0,63436	df_cut
SVM	0,535517	0,11746	0,193043	0,146053	0,461948	df_cut
RandomForest	0,717289	0,71609	0,716916	0,714664	0,791719	df_bin
CatBoost	0,716274	0,718044	0,707069	0,710533	0,801979	df_bin

SVM	0,707279	0,701347	0,721131	0,707085	0,779312	df_bin
XGBoost	0,704289	0,700939	0,712357	0,704206	0,773799	df_bin
GradientBoosting	0,708274	0,7117	0,695007	0,70137	0,78327	df_bin
LightGBM	0,705264	0,704859	0,700946	0,700784	0,784249	df_bin
HistGradientBoosting	0,703279	0,706604	0,697449	0,699181	0,783643	df_bin
KNN	0,706299	0,714449	0,687974	0,697674	0,78691	df_bin
LogReg	0,693294	0,688658	0,708992	0,695087	0,776333	df_bin
AdaBoost	0,693289	0,702802	0,681031	0,685013	0,760601	df_bin
MLP	0,678264	0,678199	0,699111	0,678234	0,749402	df_bin
Stacking	0,661303	0,668207	0,667628	0,662864	0,680825	df_bin
NaiveBayes	0,632378	0,614414	0,709283	0,656204	0,664808	df_bin
RandomForest	0,720284	0,719705	0,716131	0,7166	0,786653	df_cut_bin
XGBoost	0,712294	0,710251	0,718206	0,712127	0,771859	df_cut_bin
CatBoost	0,717244	0,723927	0,701252	0,709981	0,798892	df_cut_bin
SVM	0,710254	0,707033	0,721682	0,708449	0,784824	df_cut_bin
KNN	0,712303	0,715957	0,699193	0,705127	0,787164	df_cut_bin
GradientBoosting	0,701294	0,698242	0,70464	0,699536	0,78115	df_cut_bin
HistGradientBoosting	0,703284	0,706861	0,691945	0,697093	0,781289	df_cut_bin
LightGBM	0,694264	0,694595	0,692986	0,690604	0,777387	df_cut_bin
AdaBoost	0,691303	0,692312	0,694313	0,689311	0,762814	df_cut_bin
LogReg	0,687303	0,687629	0,686792	0,684084	0,773466	df_cut_bin
NaiveBayes	0,666353	0,652893	0,706899	0,676569	0,675681	df_cut_bin
MLP	0,686294	0,707229	0,655038	0,672343	0,767299	df_cut_bin
Stacking	0,672284	0,671644	0,67563	0,670711	0,72596	df_cut_bin
Model	**Accuracy**	**Precision**	**Recall**	**F1**	**ROC AUC**	**Dataset**

- CatBoost показал лучшие метрики accuracy=0,722 для бинарной классификации по медиане **IC50** датасете **df.csv**

б. Бинарная классификация для медианного значения CC50

на этапе baseline модели обучались без добавления предсказанного IC50 (как инженерного признака)

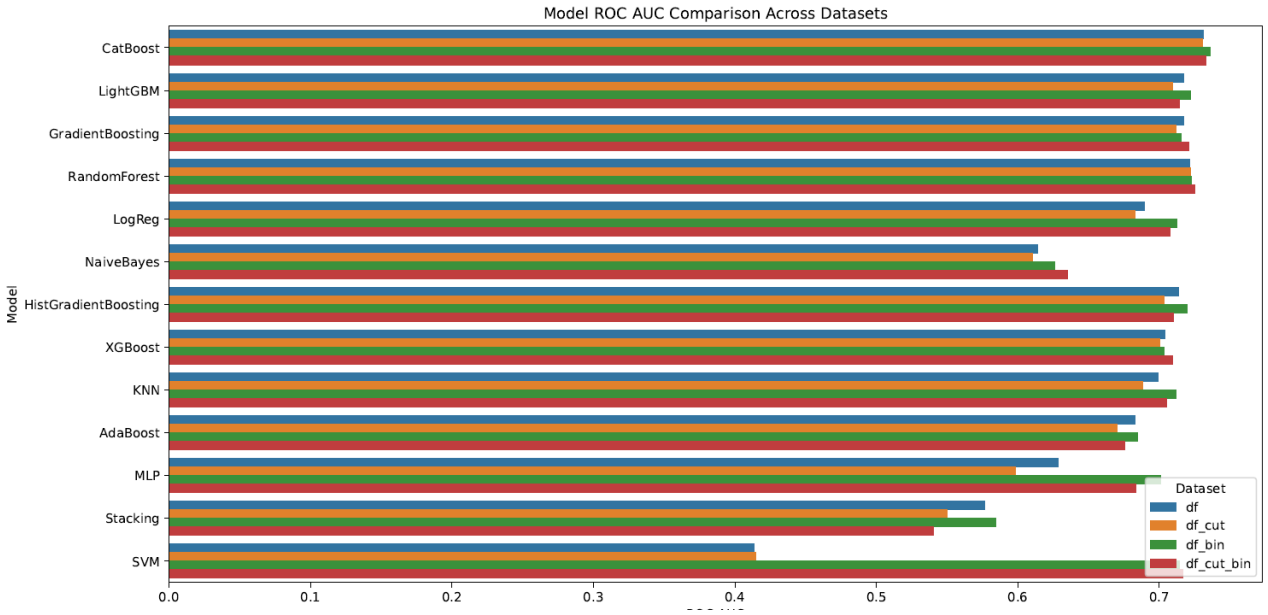


Model	**Accuracy**	**Precision**	**Recall**	**F1**	**ROC AUC**	**Dataset**
LogReg	0,726244	0,713062	0,76047	0,735237	0,81495	Df
KNN	0,674328	0,669944	0,690655	0,679795	0,760546	Df

RandomForest	0,743199	0,738246	0,758665	0,747634	0,833382	Df
GradientBoosting	0,754209	0,747265	0,77082	0,75817	0,84385	Df
HistGradientBoosting	0,745174	0,747224	0,744166	0,745289	0,834267	Df
AdaBoost	0,722234	0,713129	0,746634	0,728835	0,808935	Df
XGBoost	0,740209	0,742	0,740702	0,740769	0,832057	Df
LightGBM	0,748194	0,750777	0,74633	0,748038	0,835035	Df
CatBoost	0,743204	0,737205	0,760691	0,748021	0,84826	Df
MLP	0,700274	0,694055	0,718819	0,705803	0,730177	Df
NaiveBayes	0,539478	0,527274	0,969552	0,680614	0,664703	Df
SVM	0,504473	0,303473	0,6	0,40294	0,456044	Df
Stacking	0,749234	0,758313	0,737016	0,746272	0,812316	Df
LogReg	0,729259	0,714975	0,766813	0,739205	0,815885	df_cut
KNN	0,676303	0,666401	0,708038	0,686473	0,761705	df_cut
RandomForest	0,750209	0,752008	0,750649	0,750575	0,841058	df_cut
GradientBoosting	0,754194	0,749499	0,766193	0,757125	0,844823	df_cut
HistGradientBoosting	0,758189	0,763005	0,752883	0,756952	0,839718	df_cut
AdaBoost	0,729204	0,719796	0,751382	0,734736	0,822205	df_cut
XGBoost	0,757209	0,763015	0,748594	0,755144	0,837523	df_cut
LightGBM	0,751199	0,750379	0,754232	0,751943	0,83771	df_cut
CatBoost	0,759194	0,75821	0,763991	0,760264	0,852437	df_cut
MLP	0,673368	0,677484	0,685548	0,677945	0,70919	df_cut
NaiveBayes	0,536478	0,528681	0,946871	0,673354	0,653384	df_cut
SVM	0,504473	0,303473	0,6	0,40294	0,477292	df_cut
Stacking	0,751179	0,763718	0,729933	0,745906	0,82035	df_cut
LogReg	0,731229	0,724862	0,75335	0,737482	0,820576	df_bin
KNN	0,724254	0,721828	0,734599	0,727212	0,811075	df_bin
RandomForest	0,740219	0,732761	0,75835	0,744999	0,835046	df_bin
GradientBoosting	0,751199	0,750477	0,75439	0,752105	0,837973	df_bin
HistGradientBoosting	0,750189	0,751672	0,750665	0,750563	0,833681	df_bin
AdaBoost	0,706244	0,694111	0,73786	0,714635	0,805378	df_bin
XGBoost	0,744199	0,74073	0,752232	0,746234	0,83609	df_bin
LightGBM	0,745209	0,74313	0,752716	0,747343	0,83533	df_bin
CatBoost	0,744189	0,739368	0,757833	0,747697	0,846342	df_bin
MLP	0,736239	0,72911	0,759787	0,742156	0,802781	df_bin
NaiveBayes	0,663348	0,633695	0,778493	0,698229	0,719735	df_bin
SVM	0,743269	0,724453	0,788469	0,754978	0,81896	df_bin
Stacking	0,738194	0,749764	0,715546	0,731374	0,785451	df_bin
LogReg	0,729259	0,720928	0,755866	0,736468	0,817663	df_cut_bin
KNN	0,743229	0,738295	0,758607	0,747655	0,809716	df_cut_bin
RandomForest	0,757194	0,755018	0,764908	0,759009	0,837867	df_cut_bin
GradientBoosting	0,755189	0,750865	0,766413	0,757473	0,843177	df_cut_bin
HistGradientBoosting	0,758194	0,764755	0,74839	0,755896	0,840728	df_cut_bin
AdaBoost	0,719234	0,7158	0,727539	0,72116	0,811639	df_cut_bin
XGBoost	0,759199	0,761712	0,759421	0,75931	0,835953	df_cut_bin
LightGBM	0,753194	0,755942	0,75052	0,752686	0,837336	df_cut_bin
CatBoost	0,748199	0,747141	0,752046	0,749075	0,849844	df_cut_bin
MLP	0,746239	0,741002	0,761031	0,74972	0,817998	df_cut_bin
NaiveBayes	0,658358	0,637407	0,73616	0,683107	0,716114	df_cut_bin
SVM	0,742264	0,724766	0,785877	0,753351	0,819479	df_cut_bin
Stacking	0,739209	0,753669	0,713667	0,732022	0,80477	df_cut_bin
Model	**Accuracy**	**Precision**	**Recall**	**F1**	**ROC AUC**	**Dataset**

- CatBoost показал лучшие метрики accuracy=0,759 для бинарной классификации по медиане **IC50** датасете **df_cut.csv**

в. Бинарная классификация для медианного значения SI



Model	Accuracy	Precision	Recall	F1	ROC AUC	Dataset
CatBoost	0,677318	0,686196	0,657716	0,669717	0,731108	df
LightGBM	0,673318	0,68019	0,657946	0,667484	0,716986	df
GradientBoosting	0,666363	0,665422	0,673702	0,667209	0,717401	df
RandomForest	0,667318	0,671177	0,660582	0,663388	0,721524	df
LogReg	0,655363	0,651557	0,666471	0,657468	0,689605	df
NaiveBayes	0,526537	0,523556	0,917476	0,656637	0,613892	df
HistGradientBoosting	0,660343	0,663959	0,652473	0,656191	0,713766	df
XGBoost	0,640338	0,644995	0,632292	0,636408	0,704185	df
KNN	0,645343	0,653502	0,621244	0,635815	0,699173	df
AdaBoost	0,644353	0,653928	0,613587	0,631497	0,683027	df
MLP	0,619348	0,621419	0,623334	0,620259	0,628646	df
Stacking	0,569343	0,563961	0,576185	0,567624	0,576529	df
SVM	0,503542	0,534383	0,80612	0,548596	0,413365	df
CatBoost	0,669313	0,678935	0,649827	0,661287	0,730262	df_cut
NaiveBayes	0,519537	0,515895	0,917476	0,652525	0,610171	df_cut
GradientBoosting	0,656368	0,657785	0,653292	0,652138	0,711482	df_cut
LightGBM	0,654333	0,657187	0,649593	0,651429	0,709294	df_cut
HistGradientBoosting	0,654323	0,657371	0,648248	0,650914	0,703568	df_cut
RandomForest	0,655323	0,661951	0,642526	0,64933	0,721745	df_cut
LogReg	0,649368	0,646217	0,65695	0,649298	0,683073	df_cut
XGBoost	0,647363	0,649248	0,644117	0,644395	0,700571	df_cut
KNN	0,630328	0,639266	0,609734	0,622879	0,688416	df_cut
AdaBoost	0,633343	0,643136	0,603671	0,62076	0,670088	df_cut
MLP	0,609368	0,609234	0,616515	0,611206	0,598431	df_cut
SVM	0,503542	0,534383	0,80612	0,548596	0,414879	df_cut
Stacking	0,538403	0,532204	0,554621	0,54135	0,549953	df_cut
CatBoost	0,678323	0,686355	0,659995	0,670773	0,736188	df_bin
HistGradientBoosting	0,662323	0,665365	0,656276	0,658655	0,719635	df_bin
SVM	0,669333	0,683766	0,643191	0,657962	0,714035	df_bin
LightGBM	0,660333	0,666334	0,648188	0,654977	0,722342	df_bin

KNN	0,660323	0,664913	0,646342	0,654927	0,711657	df_bin
LogReg	0,655383	0,652354	0,662994	0,654695	0,712407	df_bin
RandomForest	0,652333	0,655012	0,649929	0,650137	0,722802	df_bin
GradientBoosting	0,654353	0,657952	0,64736	0,649366	0,715484	df_bin
XGBoost	0,642343	0,637085	0,661994	0,647862	0,703268	df_bin
AdaBoost	0,646343	0,654929	0,622649	0,637367	0,684805	df_bin
MLP	0,652343	0,668635	0,615286	0,63353	0,70087	df_bin
NaiveBayes	0,596423	0,594769	0,614013	0,603256	0,62607	df_bin
Stacking	0,571343	0,565001	0,584301	0,572626	0,584441	df_bin
CatBoost	0,686313	0,699228	0,65745	0,675299	0,733051	df_cut_bin
LogReg	0,670338	0,672555	0,668198	0,668089	0,707699	df_cut_bin
GradientBoosting	0,674328	0,680832	0,660223	0,665786	0,720882	df_cut_bin
RandomForest	0,667313	0,670577	0,664227	0,664901	0,724907	df_cut_bin
SVM	0,673333	0,689442	0,640705	0,659293	0,71691	df_cut_bin
LightGBM	0,661328	0,664785	0,654158	0,657052	0,714373	df_cut_bin
KNN	0,658333	0,660386	0,654087	0,656852	0,704903	df_cut_bin
HistGradientBoosting	0,659313	0,663643	0,652473	0,655429	0,710156	df_cut_bin
MLP	0,642328	0,634679	0,675942	0,653688	0,683656	df_cut_bin
XGBoost	0,648348	0,649344	0,650023	0,647417	0,709447	df_cut_bin
AdaBoost	0,639338	0,649118	0,606555	0,625196	0,675322	df_cut_bin
NaiveBayes	0,605388	0,60628	0,602754	0,604434	0,635102	df_cut_bin
Stacking	0,551368	0,548423	0,571464	0,559155	0,54033	df_cut_bin
Model	**Accuracy**	**Precision**	**Recall**	**F1**	**ROC AUC**	**Dataset**

CatBoost показал лучшие метрики accuracy=0,673 для бинарной классификации по медиане **IC50** датасете **df.csv**

Выводы по baseline для задач классификации по медиане:

- Для различных целевых переменных наилучшие результаты демонстрировали разные модели и различные варианты предобработки данных.
- Если задача состоит в достижении максимальной точности для каждой конкретной задачи, необходимо подбирать модель и подход к предобработке индивидуально.
- Однако при анализе результатов можно заметить, что существует возможность подобрать такую комбинацию модели и предобработки, при которой снижение метрик по сравнению с оптимальными значениями будет минимальным для всех задач. В этом случае можно построить универсальный пайплайн для задач регрессии (если такая цель стоит).
- В нашем случае наилучшим компромиссным решением оказалась модель CatBoost, обученная на датасете df_cut.csv.

6. Тонкая настройка (fine tuning)

В тонкую настройку включены следующие этапы:

- двухэтапный подбор гиперпараметров для лучшей модели и датасета:

RandomizedSearchCV использовался для предварительного поиска перспективной области гиперпараметров. GridSearchCV применялся для более точной настройки в найденной области.

- Для повышения качества моделей предсказывались логарифмы IC50 и CC50 с помощью отдельных регрессионных моделей, обученных исключительно на тренировочной выборке. Полученные предсказания затем

использовались как новые признаки и для тренировочной, и для тестовой части, чтобы исключить утечку данных.

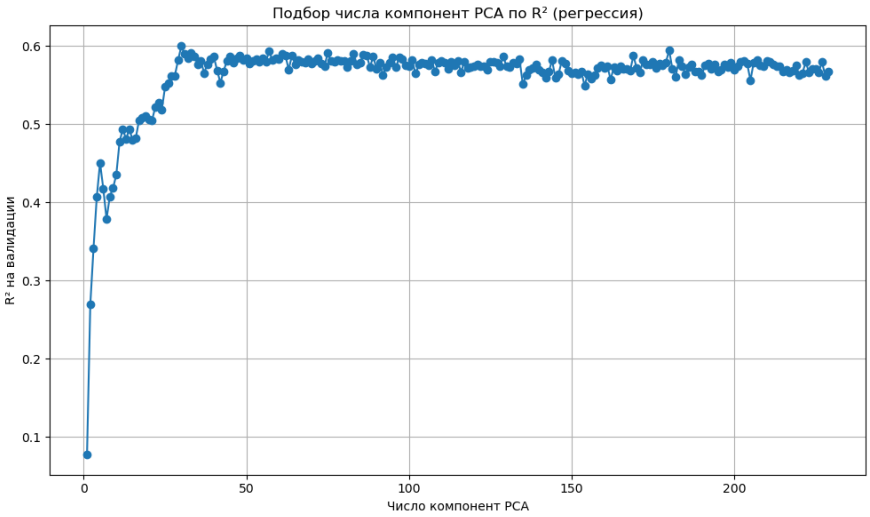
Алгоритм следующий:

Модели регрессии обучаются только на train**.**

Потом они делают предсказания для:

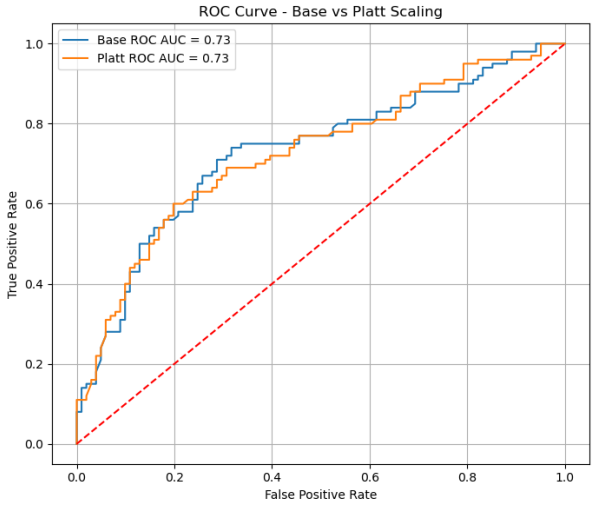
- обучающей выборки (чтобы вставить эти признаки в train);
- тестовой выборки (чтобы вставить признаки в test — но эти данные не использовались при обучении**).**

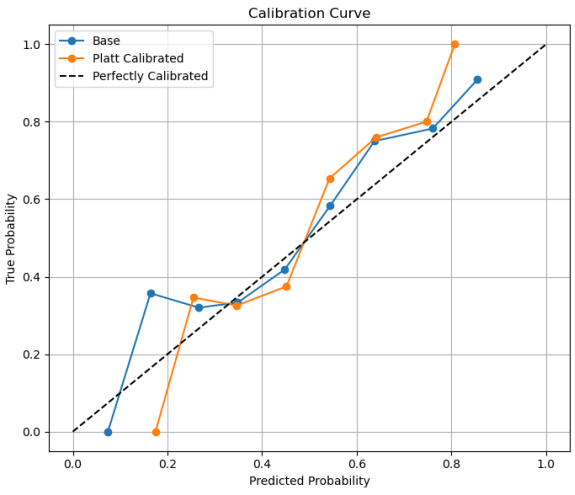
- Отбор количества важных признаков в цикле. Оставлялось такое количество важных признаков, которое давало наилучшие метрики (при условии сохранения баланса между метриками, о котором говорилось выше). Итеративный отбор позволяет избежать включения нерелевантных или шумовых признаков, которые могут снижать обобщающую способность модели.



- Для задач бинарной классификации — калибровка вероятностей с помощью метода Платта (Platt scaling, сигмоидальная калибровка) для улучшения согласованности вероятностных предсказаний, там где это приводило к улучшению метрик и тоже с учетом необходимости баланса между метриками.

Например калибровка для классификации по медиане SI привела к улучшению accuracy без ухудшения ROC_AUC:





7. Лучшие значения метрик

- Регрессия для log10(IC50):

на тестовой выборке MAE 0.4623, MSE 0.3507, RMSE 0.5922, MedAE 0.3795, **R² 0.6480**

на кросс валидации MAE: 0.4658 ± 0.0199, RMSE: 0.5929 ± 0.0306, R²: 0.5932 ± 0.0782

- Регрессия для log10(CC50):

на тестовой выборке MAE 0.2833, MSE 0.1743, RMSE 0.4175, MedAE 0.1789, **R² 0.5971**

на кросс валидации MAE: 0.3158 ± 0.0192, RMSE: 0.4509 ± 0.0262, R²: 0.5628 ± 0.0399

- Регрессия для log10(SI):

на тестовой выборке MAE 0.3494, MSE: 0.2788, RMSE: 0.5280, MedAE: 0.2318, **R²: 0.5344**

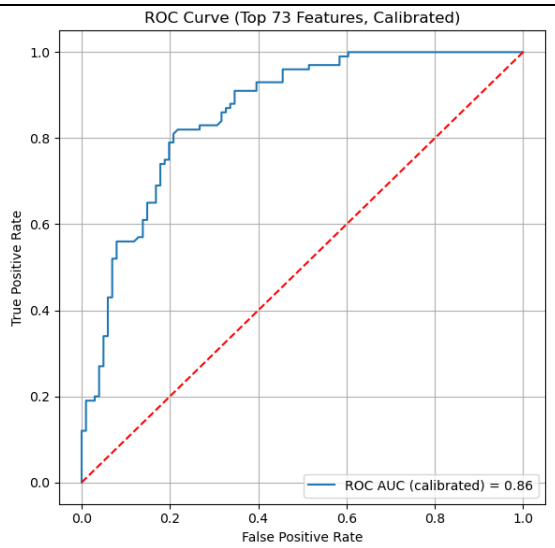
на кросс валидации MAE 0.3294 ± 0.0202, RMSE: 0.4945 ± 0.0317, R²: 0.5618 ± 0.0385

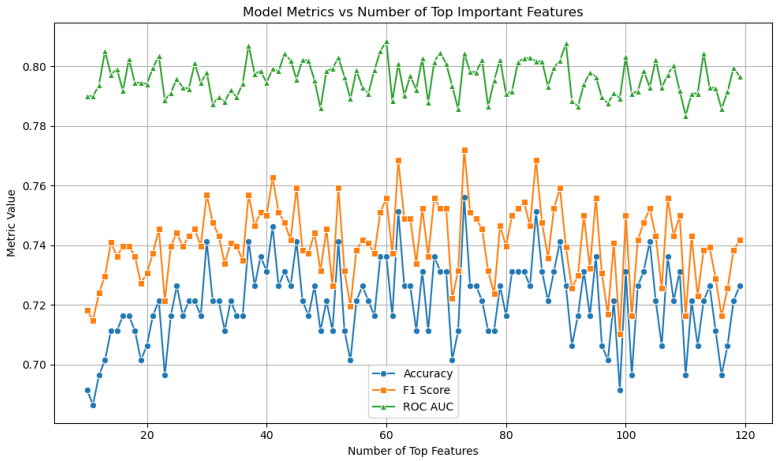
- Бинарная классификация по медиане IC50:

Classification report (calibrated) на 20% отложенной выборке:

	precision	recall	f1-score	support
0	0.81	0.70	0.75	101
1	0.73	0.83	0.78	100
accuracy			0.77	201
macro avg	0.77	0.77	0.77	201
weighted avg	0.77	0.77	0.77	201

Test Accuracy (calibrated): 0.7662
Test ROC AUC (calibrated): 0.8600



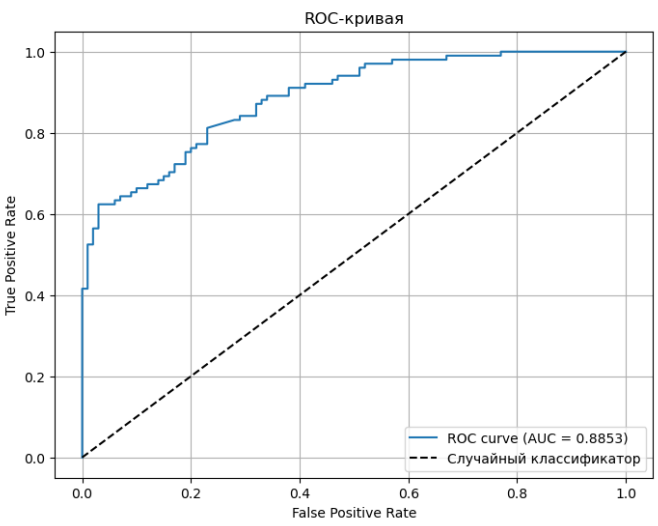


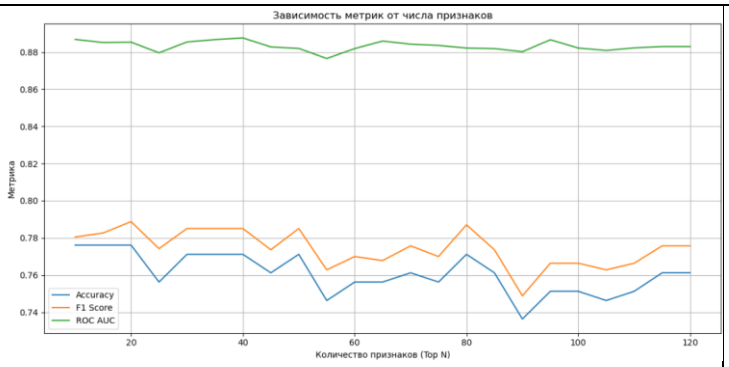
- Бинарная классификация по медиане CC50:

=== Classification Report на на 20% отложенной выборке (только топ-20 признаков) ===

	precision	recall	f1-score	support
0	0.81	0.72	0.76	100
1	0.75	0.83	0.79	101
accuracy			0.78	201
macro avg	0.78	0.78	0.78	201
weighted avg	0.78	0.78	0.78	201

Accuracy: 0.7761
Precision: 0.7500
Recall: 0.8317
F1 Score: 0.7887
ROC AUC: 0.8853



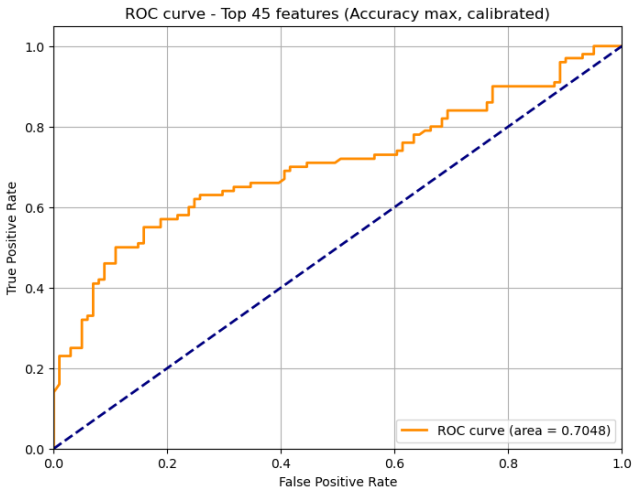


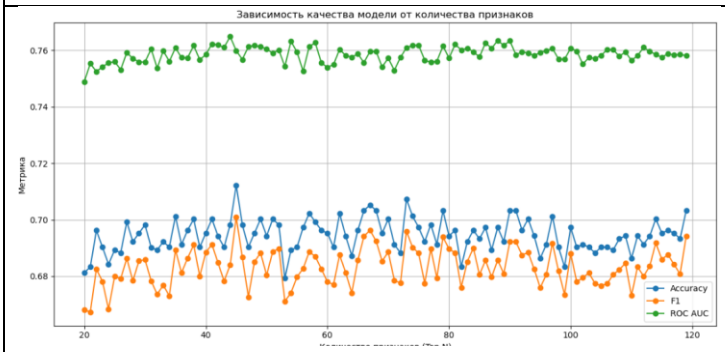
- Бинарная классификация по медиане SI:

=== Метрика: F1 - Top N признаков: 45 на 20% отложенной выборке ===
Accuracy: 0.6766
Precision: 0.7215
Recall: 0.5700
F1 Score: 0.6369
ROC AUC: 0.7048

Classification Report:

	precision	recall	f1-score	support
0	0.65	0.78	0.71	101
1	0.72	0.57	0.64	100
accuracy			0.68	201
macro avg	0.68	0.68	0.67	201
weighted avg	0.68	0.68	0.67	201





Метрики модели классификации по медиане SI оказались ниже по сравнению с baseline-результатами, что объясняется различием в подходах к оценке качества: в базовых моделях метрики рассчитывались с использованием кросс-валидации, тогда как здесь — на отложенной тестовой выборке.

Снижение метрик на отложенной выборке по сравнению с baseline-моделями (оцененными по кросс-валидации) не является ошибкой, а отражает реальное качество обобщения.

Поскольку тестовая выборка полностью исключена из процесса обучения и валидации, она представляет собой более строгую и реалистичную проверку обобщающей способности модели. В таких условиях значения метрик зачастую оказываются ниже, чем при кросс-валидации, что является нормальной практикой.

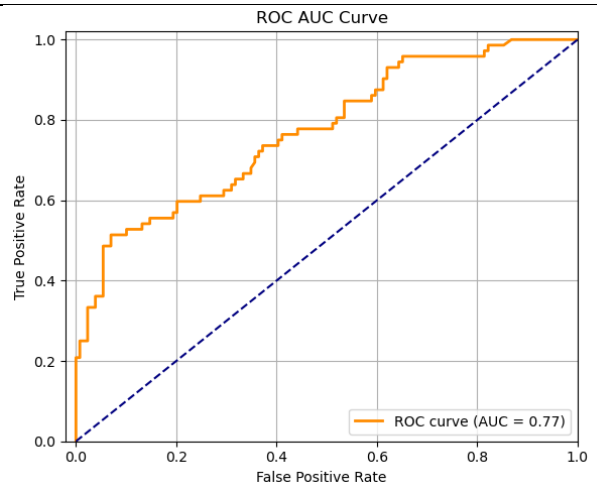
- Бинарная классификация по порогу SI>8:

=== Финальная оценка на 20% отложенной выборке ===

Accuracy: 0.7512
Precision: 0.7037
Recall: 0.5278
F1 Score: 0.6032
ROC AUC: 0.7712

=== Classification Report ===

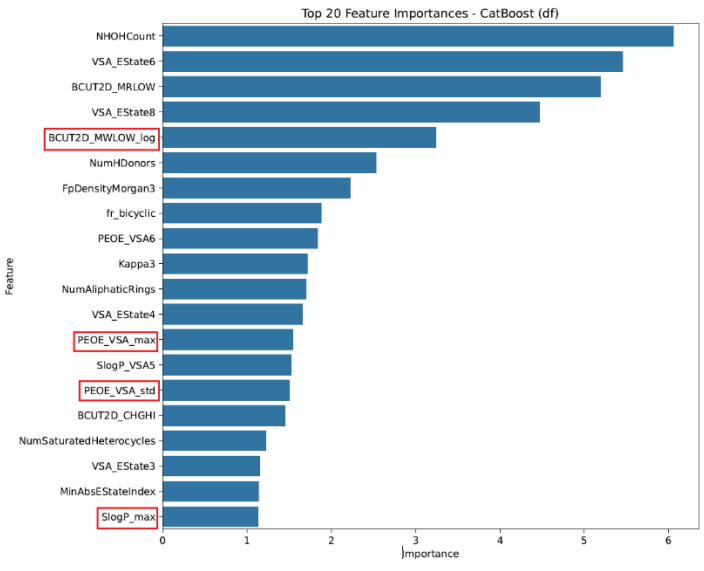
	precision	recall	f1-score	support
SI ≤ 8	0.77	0.88	0.82	129
SI > 8	0.70	0.53	0.60	72
accuracy			0.75	201
macro avg	0.74	0.70	0.71	201
weighted avg	0.75	0.75	0.74	201

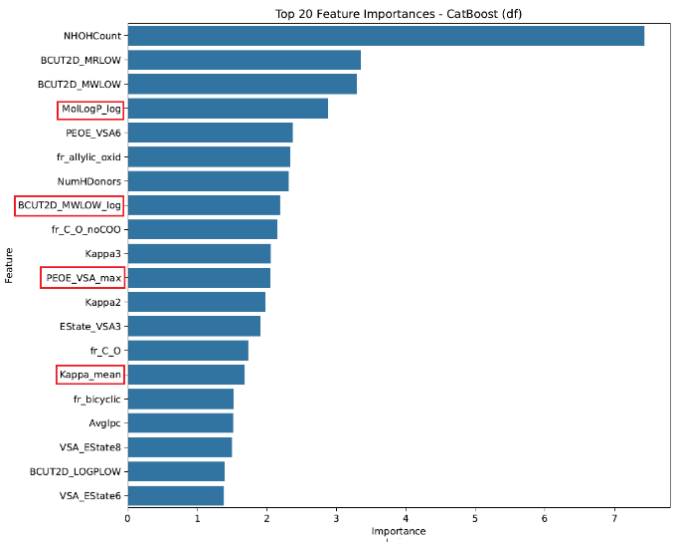


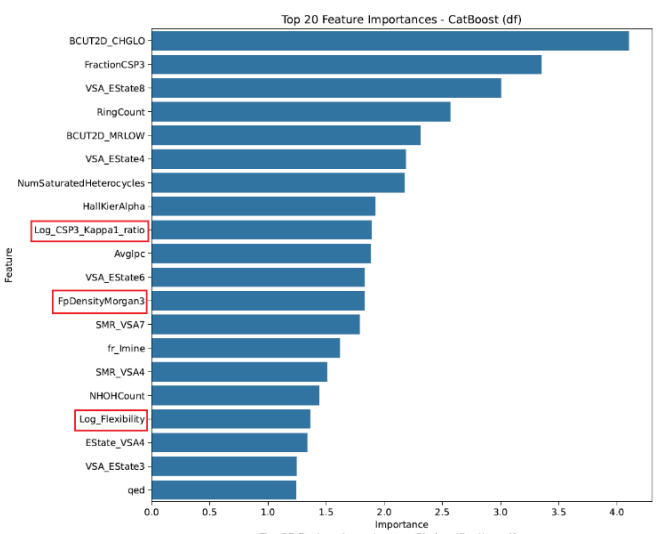
Вывод по бинарной классификации SI>8:

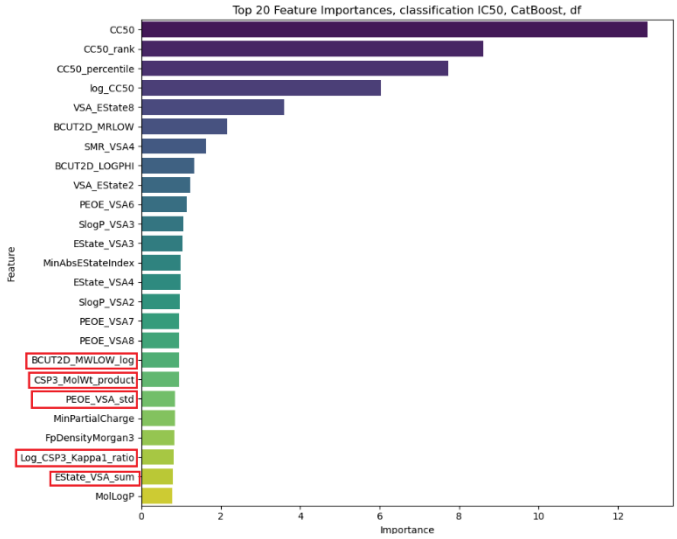
Presicion = 0,7 , т.е. из всех предсказанных "SI > 8", 70% верны — достаточно хороший результат
Recall = 0,53 т.е. лишь 53% реально позитивных случаев найдены — низкий результат
ROC AUC = 0,77, т.е. модель достаточн хорошо разделяет классы
Classification report показывает, что модель лучше справляется с большим классом (SI ≤ 8), чем с меньшинством (SI > 8), следствие дисбаланса
В нашем случае важно минимизировать ложные срабатывания, поэтому метрика Precision является приоритетной

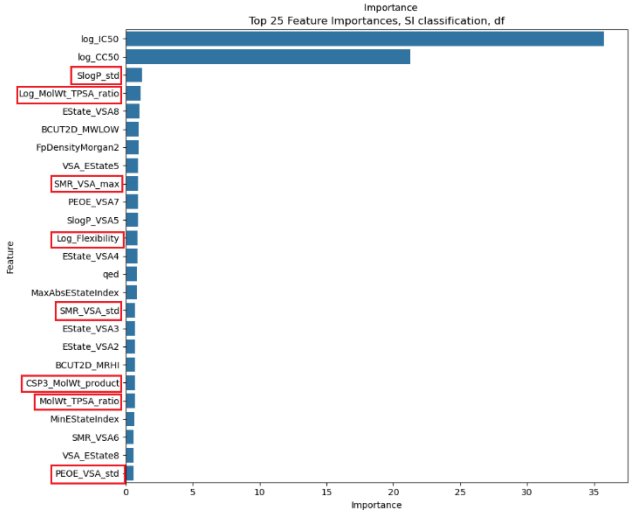
8. **Оценка эффективности внедрения инженерных признаков**

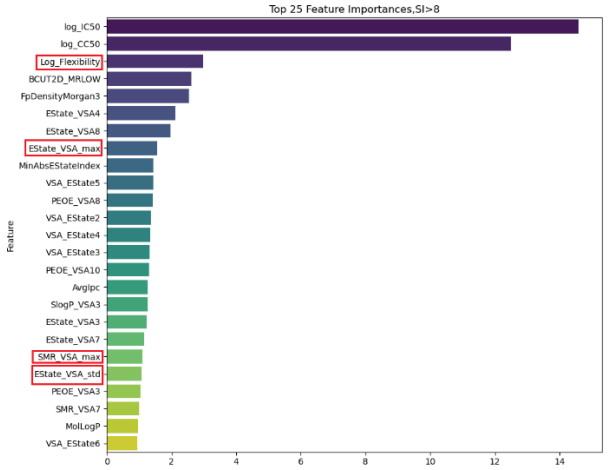












Вывод по инженерным признакам:

- Около 15-25% признаков из top 20 features являются инженерными. То, что такие признаки входят в топ по важности, говорит о том, что они обогащают модель новой информацией
- В основном это агрегатные признаки и логарифмирование.
- Инженерные признаки с доменной интерпретацией также встречаются среди топ 20 ('Log_Flexibility')
- Создание полиномиальных признаков из топ 5 не внесло никакого вклада в увеличение метрик

8. **Оценка полученных результатов**

Для QSAR-моделирования результаты, конечно, сильно зависят от используемых данных, но в целом их можно о ценивать по следующим критериям (для открытых химико-биологических баз ChEMBL и PubChem):

1. Регрессия (IC50, CC50, SI)

Качество	**R^2** (PubChem)	**Мои результаты R^2 ,**
Отлично	> 0.70	
Очень хорошо	0.60–0.70	R^2 0.6480 для log10(IC50)
Хорошо	0.40–0.60	R^2 0.5971 для log10(CC50) R^2: 0.5344 для log10(SI)
Удовлетворительно	0.20–0.40	
Плохо	< 0.20	

2. Классификация (IC50 > медианы, CC50 > медианы, SI > медианы, SI > 8)

Качество	**Precision / Recall / F1** (PubChem)	**ROC AUC** (PubChem)	**Мои результаты F1,**	**Мои результаты ROC AUC**
Отлично	> 0.85	> 0.90		
Очень хорошо	0.75–0.85	0.85–0.90	0,79 для CC50	0,86 для IC50 0,885 для CC50
Хорошо	0.65–0.75	0.75–0.85	0,77 для IC50	0,77 для SI>8
Удовлетворительно	0.55–0.65	0.65–0.75	0,637 для SI	0,705 для SI
Плохо	< 0.55	< 0.65	0,60 для SI>8	

- SI (Selectivity Index) как производная величина предсказывается хуже потому что:

- SI = CC50 / IC50 — это производная величина, формула которой усиливает шум, ошибки и выбросы.
- Маленькие ошибки в IC50 и CC50 могут приводить к большим ошибкам в SI (особенно если IC50 → 0).
- Если IC50 и CC50 коррелируют, то SI может быть нестабильным и сложным для обучения модели.
- производные показатели часто требуют специальных методов стабилизации данных и дополнительной предобработки для улучшения качества моделирования.

Падение метрик для SI примерно на 20-25% по F1 и 10% по ROC AUC **—** это нормальный «уровень сложности» для производной переменной.

- При перекосе классов (например, SI > 8 = 36%) precision может быть более важным, если нужно избегать ложных срабатываний (false positives).

Другие выводы (не отмеченные ранее)**:**

- Для задач регрессии**,** в целом наилучшие результаты были получены на исходном датасете без обработки выбросов и без удаления признаков на основе корреляционного анализа.
- Для задач классификации**,** в целом лучшие показатели наблюдались при использовании датасета без обработки выбросов**,** но с предварительным удалением сильно коррелированных признаков**.**
- В обоих случаях отсутствие необходимости в обработке выбросов объясняется тем, что большинство использованных моделей относятся к семейству деревоподобных алгоритмов **(**градиентный бустинг**,** случайный лес и т.д.**),** которые устойчивы к выбросам
- Предположение почему на деревьях корреляция менее критична для регрессии, чем для классификации:

- **В регрессии целевая переменная непрерывная.**

Здесь цель — минимизировать среднеквадратичную ошибку (MSE) и коррелированные признаки не мешают, а порой даже дополняют друг друга
Даже если два признака дают один и тот же «тип» сигнала, модель будет использовать их в той степени, в какой они улучшают точность предсказания, и не более того.
Бустинг в регрессии аппроксимирует остатки — он обучается на ошибках предыдущих деревьев. Даже если в модели есть коррелированные признаки, они редко вызывают дублирование, потому что каждый следующий шаг уточняет только то, что не объяснено ранее.
В **регрессии** коррелированные признаки могут играть роль «страховки» — даже если один из них шумный, другой может «подтянуть» ошибку вниз..

- В классификации целевая переменная дискретная

Границы решений деревьев часто зависят от небольшого количества объектов, особенно в глубине дерева. При наличии коррелированных признаков дерево может строить несколько очень похожих ветвей, «думая», что находит полезный сигнал, но на самом деле дублируя один и тот же шаблон, что ведёт к переобучению.
В задачах классификации коррелированные признаки могут усиливать переобучение: дерево воспринимает их как отдельные важные сигналы и многократно использует схожие разбиения, из-за чего усложняется структура и снижается обобщающая способность
В классификации критерии (Gini, Entropy) не всегда устойчивы к дублирующим признакам, особенно при небольшом числе примеров в узлах дерева.