Аналитический отчет по результатам курсовой работы.

В рамках курсовой работы была поставлена задача по созданию моделей для

1. Регрессия для IC50
2. Регрессия для CC50
3. Регрессия для SI
4. Классификация: превышает ли значение IC50 медианное значение выборки
5. Классификация: превышает ли значение CC50 медианное значение выборки
6. Классификация: превышает ли значение SI медианное значение выборки
7. Классификация: превышает ли значение SI значение 8

Код решения соответствующих подзадач представлен по ссылке.

Задачи решались поэтапно и соответсвенно можно посмотреть эволюцию кода. Сперва решались задачи регресси, а уже потом задачи классификации.

Датасет представленный для курсовой работы представляет собой набор данных из 1001 записей, содержащих 213 физико-химических признаков (индекс сразу не учитываем в признаках), **3 из которых являются целевыми и не используются при решении поставленных задач**.

Первым этапом работы над курсовой было проведение EDA. ссылка

EDA разделен на 4 подэтапа

1. Общая информация.

В рамках общей информации рассмотрена общая информация о представленных данных, выявленно, что данные содержат пропуски в ряде колонок:

'MaxPartialCharge', 'MinPartialCharge', 'MaxAbsPartialCharge', 'MinAbsPartialCharge', 'BCUT2D_MWHI', 'BCUT2D_MWLOW', 'BCUT2D_CHGHI', 'BCUT2D_CHGLO', 'BCUT2D_LOGPHI', 'BCUT2D_LOGPLOW', 'BCUT2D_MRHI', 'BCUT2D_MRLOW'

Вцелом, данные пропуски соответсвуют 3 записям, что составляет 0.3% от общего числа данных. На этом этапе было принято решение, что данными записями можно спокойно пожертвовать и просто удалить их при решении задач классиифкации и регрессии, а не искать способы их заполнить.

2. Анализ IC50

IC50, mM — концентрация вещества, при которой наблюдается 50% ингибирование активности (мера эффективности лекарства).

Основные характеристики:

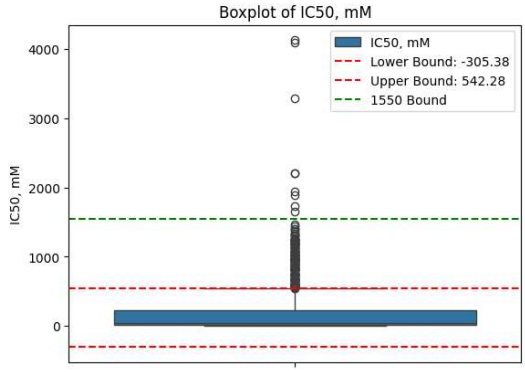
Среднее значение = 221.118757

Стандартное отклонение = 400.510657

Минимальное значение = 0.003517

Максимальное значение = 4128.529377

Из представленнх данных видно, что у нас существует очень большой разброс значений. Можно сделать вывод, что данные содержа выбросы. Границей выбросов можно считать значения выше 542.28.



При этом на графике можно наблюдать достаточно плотную группу значений в диапазоне 542.28 - 1550. Количество значений выше 542.28 = 145 или ~14.53%. Исходя из существенного объема выбросов было принято решение, что выбросами будем считать значения выше 1550. Значение 1550 было подобрано вручную с помощью анализа графиков распределения данных.

Был проведен корреляционный анализ данных и выделены параметры, у которых уровень корреляции по модулю превышает 0.2. Самая существенная корреляция выявлена с CC50 и составляет 0.52. Все остальные параметры коррелируют с целевым значением ниже 0.3. Можно сделать вывод, что уровень корреляции между целевым значением и параметрами весьма слабый. Таким образом линейные модели для решения поставленных задач не подходят. Кроме того, была выявлена мультиколлениарность между параметрами.

Дополнительно были построены графики распределения по самым коррелирующим параметрам, которые также подтвердили отсутствие линейных зависимостей.

3.Анализ CC50

CC50, mM — концентрация, вызывающая гибель 50% клеток (мера токсичности)

Основные характеристики:

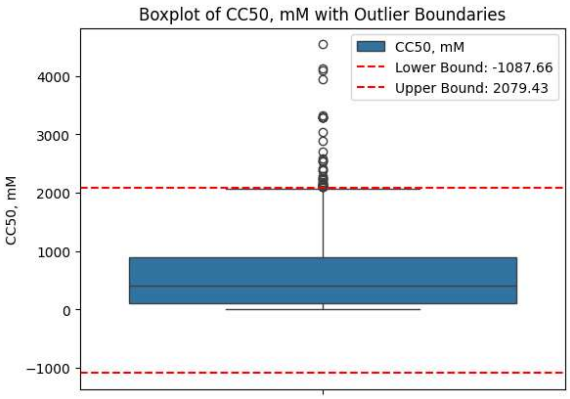
Среднее значение = 586.668414

Стандартное отклонение = 642.016454

Минимальное значение = 0.700808

Максимальное значение = 4538.976189

Также, как и в предыдущем случае, мы видем, что у нас существует очень большой разброс значений. Можно сделать вывод, что данные содержа выбросы. Границей выбросов можно считать значения выше 2079.43. Количество значений выше верхней границы: 39 или 3.9%



По результатам корреляционного анализа выявленны параметры, котороые коррелируют с целевым значением по модулю > 0.2. Самая сильная корреляция выявлена с параметром IC50, mM и составляет ~ 0.5368. Все остальные параметры коррелируют в диапазоне > 0.2 и < 0.3 , что говорит о слабой корреляциия. Линейные модели также не применимы.

4.Анализ SI

SI - селективный индекс (SI = CC50 / IC50), отражает терапевтическое окно.

Основные характеристики:

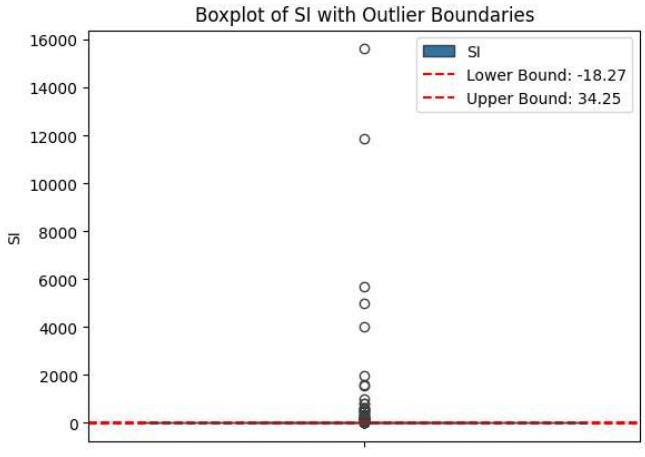
Среднее значение = 72.132600

Стандартное отклонение = 699.829004

Минимальное значение = 0.011489

Максимальное значение = 15620.600000

Опять мы видим колосальный разброс значений. Границей выбросов будем считать значение > 34.25



По результатам корреляционного анализа выявленно всего лишь 2 параметра, которые по модулю превышают 0.2

Коэффициент корреляции между SI и IC50, mM: -0.378204293062784

Коэффициент корреляции между SI и fr_Imine: 0.2353267774166921

И снова мы видим, что линейные модели нам не подходят.

ЗАДАЧИ РЕГРЕССИИ

Метрики, используемые для оценки моделей и выбора лучших параметров в задачах регрессии:

Для выбора лучших параметров с помощью GridSearchCV использовалась метрика MSE - среднеквадратическая ошибка. MSE штрафует модели за большие ошибки.

Для сравнения моделий используются следующие метрики:

1. MSE
2. MAE - смотрим среднее абсолютное отклонение предсказанных данных от фактических.
3. R2 - оценивает, насколько хорошо модель регрессии объясняет вариацию зависимой переменной от признаков.

При решении задачи регрессии IC50 были рассмотренны следюущие модели:

1. KNeighborsRegressor

Параметры:

-n_neighbors - кол-во ближайших соседий

-weights - определяет, как взвешиваются соседние точки при голосовании

-algorithm - алгоритм, который используется для поиска ближайших соседей

-p - 1:расстояние Манхэттена (L1-норма) 2: евклидово расстояние (L2-норма)

2. RandomForestRegressor

-n_estimators - кол-во деревьев

-max_depth - максимальная глубина дерева

-min_samples_split - минимальное количество образцов, необходимое для разделения узла

-min_samples_leaf - минимальное количество образцов в листе дерева

-max_features - количество признаков, используемых при построении каждого дерева

3. SVR

-kernel - ядро (использованы ядра rbf, poly, linear)

-C - коэффициент регуляризации

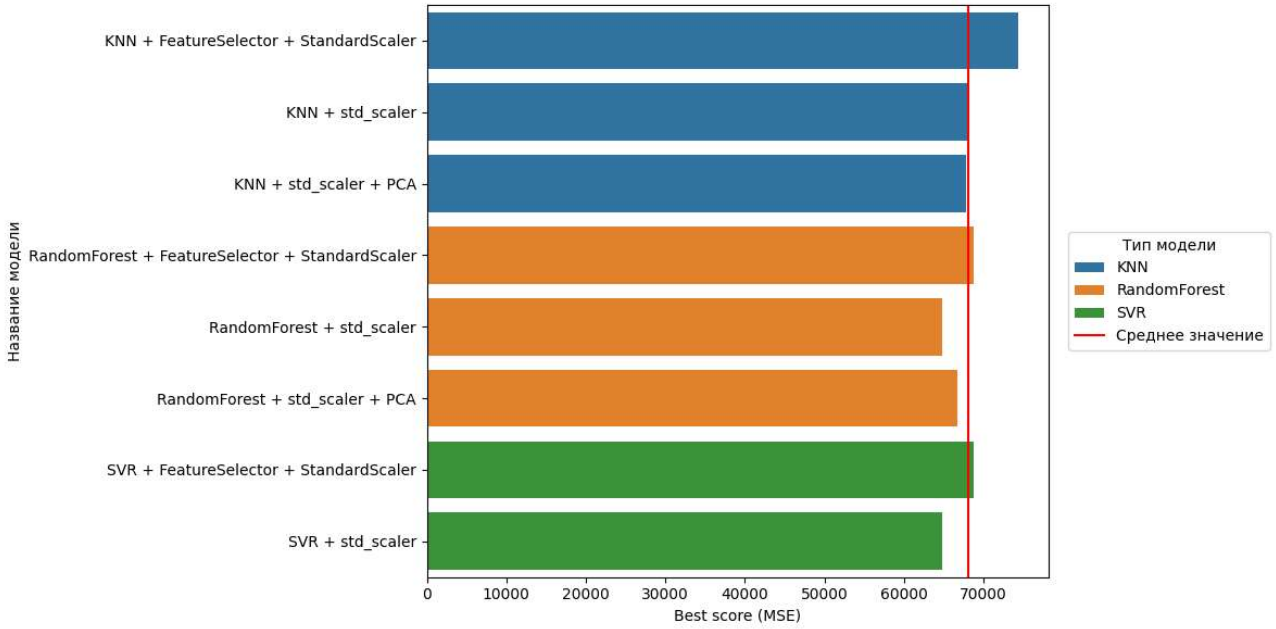
-gamma - гиперпараметр ядра RBF, определяет "радиус влияния" одной точки

-epsilon - размер эпсилон-трубки - зона, внутри которой ошибки не штрафуются.

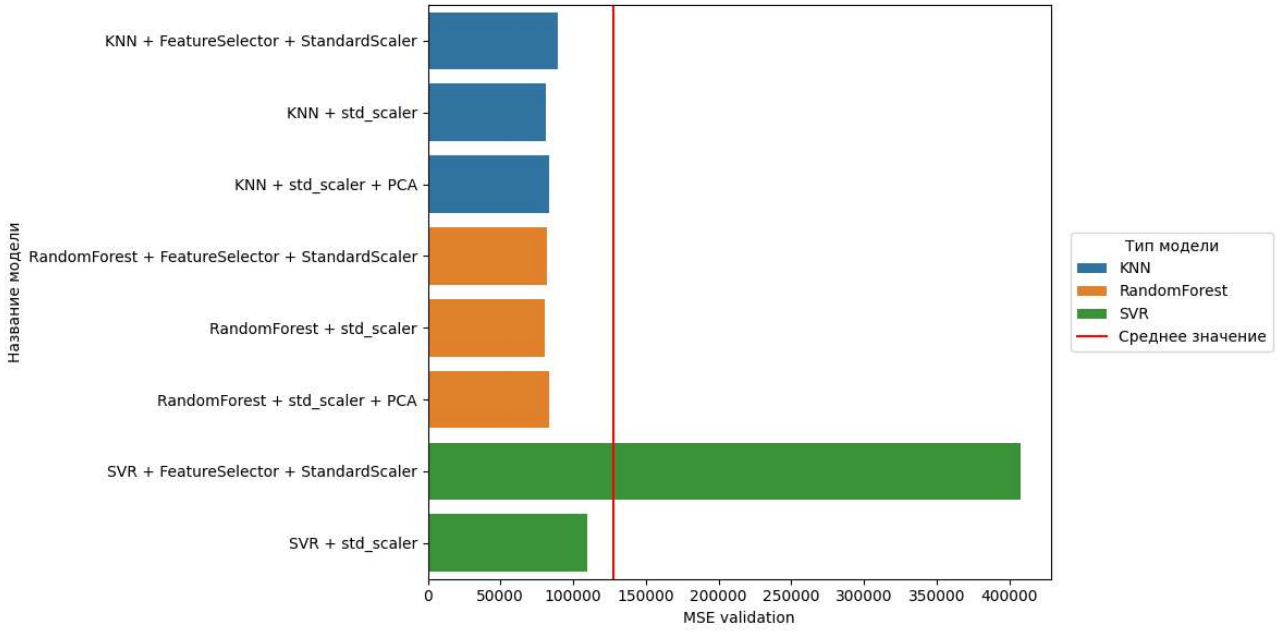
Были использованы дополнительные шаги

- •Стандартизация
- •Выбор параметров + стандартизация
- •Стандартизация + PCA (метод главных компонент)

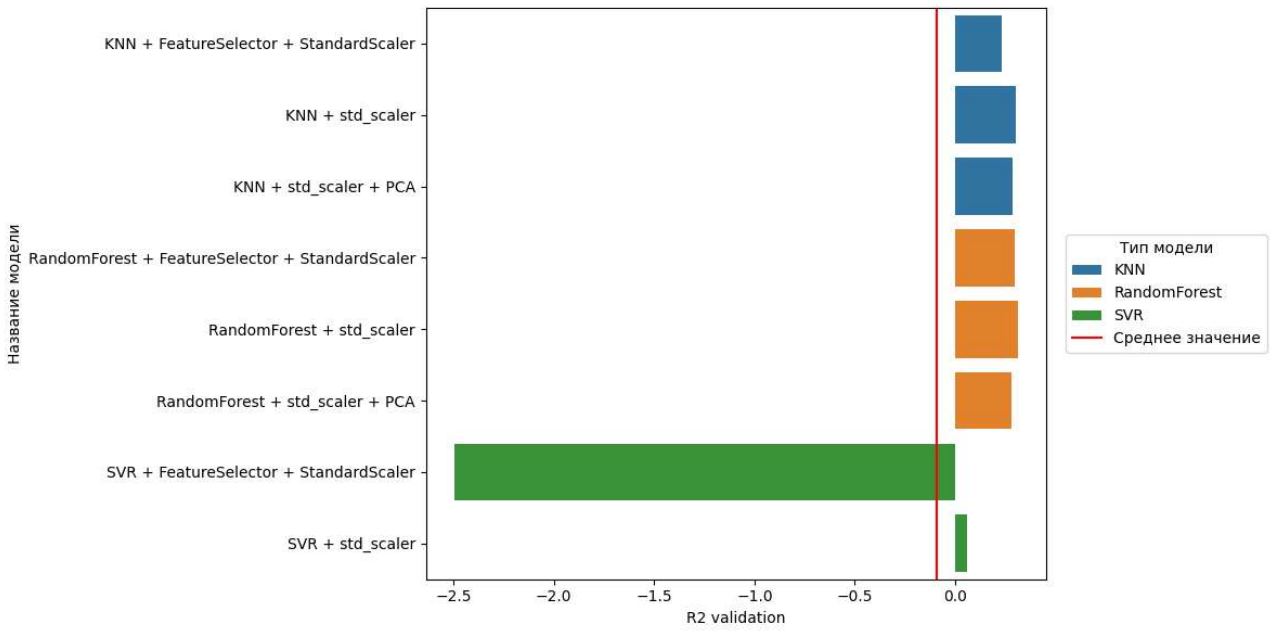
Сравнительные результаты представленны на графиках ниже



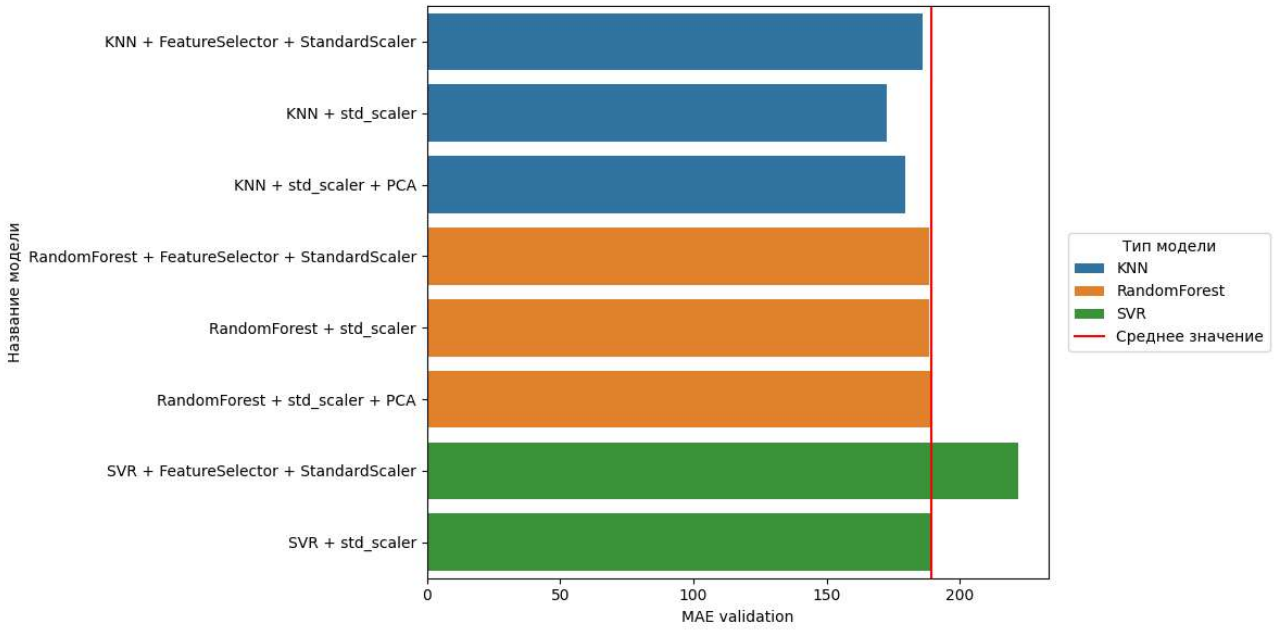
На представленном графике видно, что на тренировочных данных лучше всего себя показала модель RandomForest + std_scaler. Общее наблюдение для всех моделей - дополнительные шаги кроме стандартизации не дают улучшение метрики.



На тестовой выборке лучший параметр MSE был у модели KNN + std_scaler. На втором месте с несущественны остованием оказалась RandomForest + std_scaler.



По метрике R2 лучшее значение у модели RandomForest + std_scaler. На втором месте KNN + std_scaler Тут же можно отметить сильно отрицательное значение для модели SVR + FeatureSelector + standardScaler.



По метрике MAE первые места занимают модели KNN, что вцелом весьма предсказуемо исходя из логики работы алгоритма.

С учетом 3еч параметров лучшей моделью считаем **KNN + std_scaler.**

При решении задачи регрессии CC50 были рассмотренны следюущие модели:

1. KNeighborsRegressor

Параметры (описаны ранее)

2. RandomForestRegressor

Параметры (описаны ранее)

3. AdaBoostRegressor

Параметры:

-n_estimators - кол-во базовых моделей

-learning_rate - коэффициент скорости обучения

-loss - функция потерь

-estimator - базовые классификаторы (Дерево решений с глубиной - 1, 2 или 3)

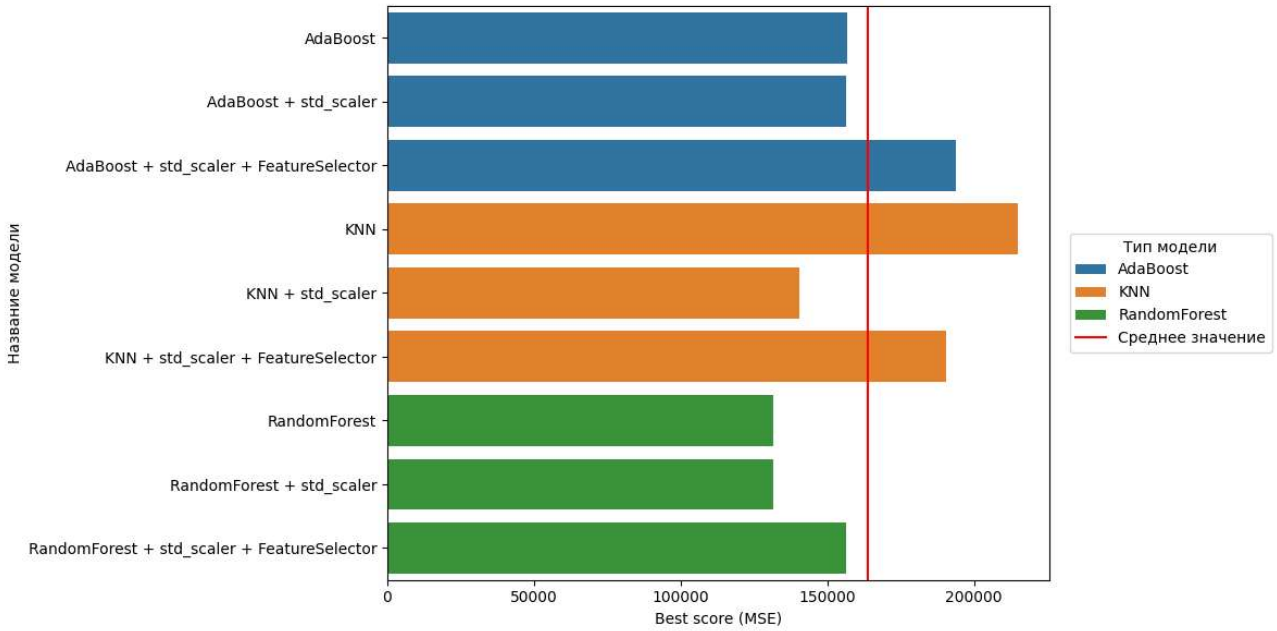
Были использованы дополнительные шаги

- Чистые данные

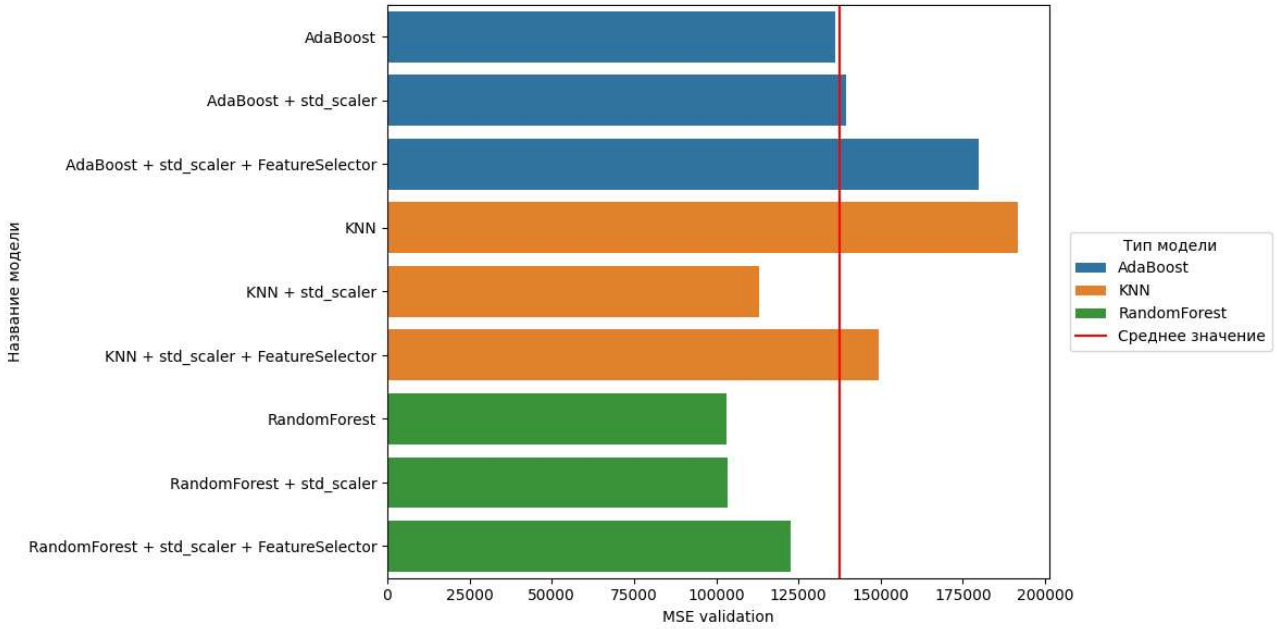
- Стандартизация

- Стандартизация + Выбор параметров

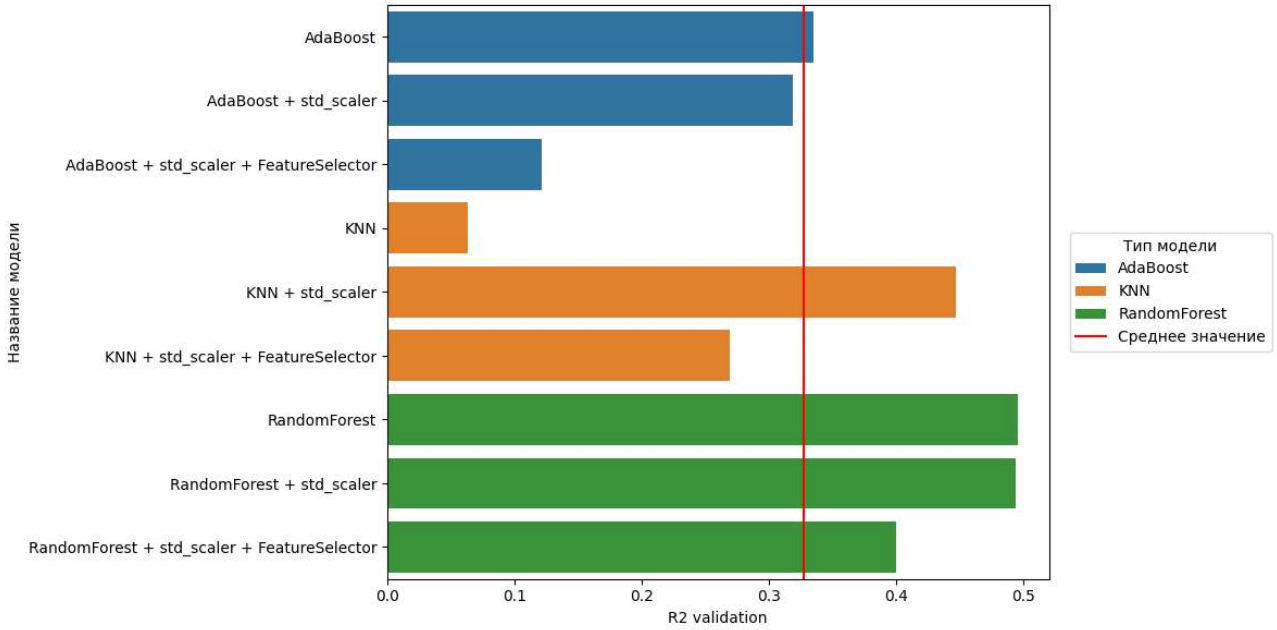
Сравнительные результаты представлены на графиках ниже



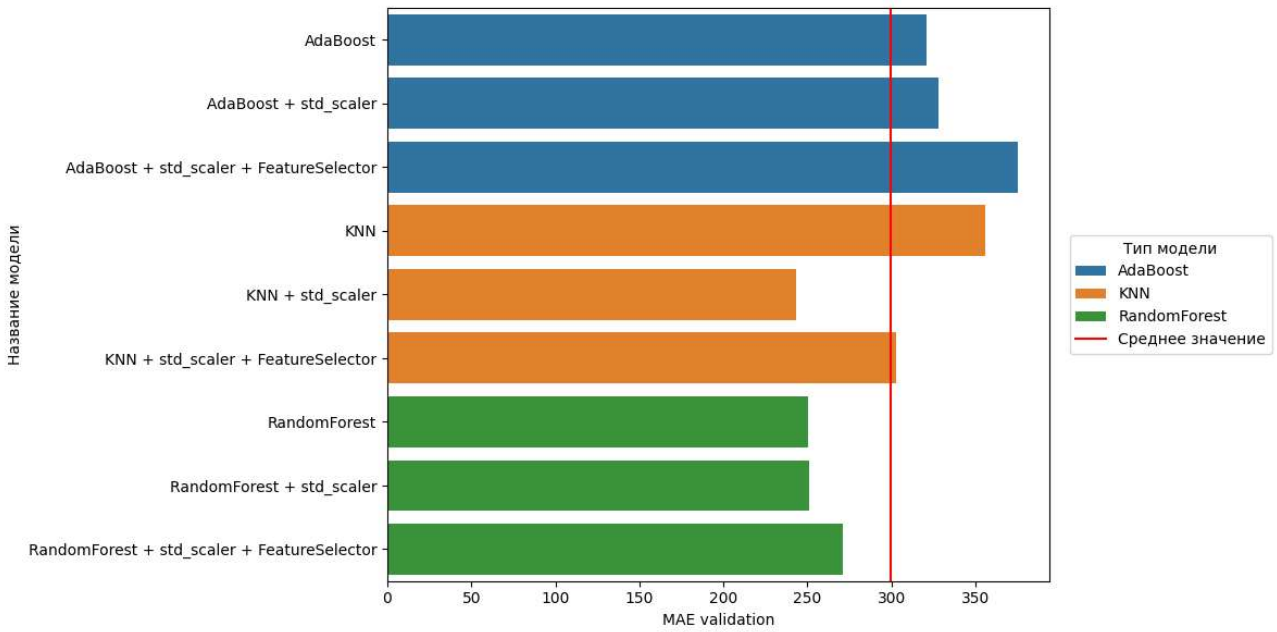
На тренировочных данных лучшие показатель у моделей RandomForest и RandomForest + std_scaler. На втором месте KNN + std_scaler.



На валидационной выборке расположение моделей такое же, как и на тестовой.



Лучшее значение метрики R2 у тех же самых моделей.



По метрике MAE лучшей моделью стала KNN + std_scaler, а вторые места занимают RandomForest и RandomForest + std_scaler.

По совокупности метрик лучшей моделью становится RandomForest.

При решении задачи регрессии SI были рассмотренны следюущие модели:

1. KNeighborsRegressor

Параметры (описаны ранее)

2. RandomForestRegressor

Параметры (описаны ранее)

3. AdaBoostRegressor

Параметры: (описаны ранее)

Были использованы дополнительные шаги

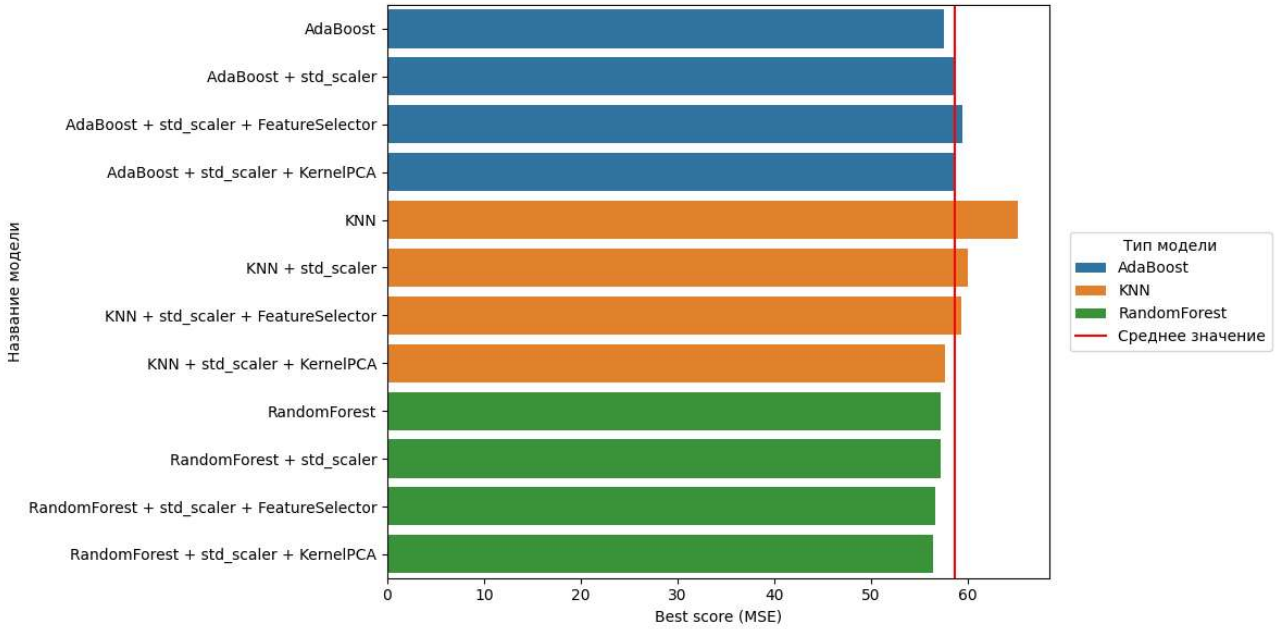
- Чистые данные

- Стандартизация

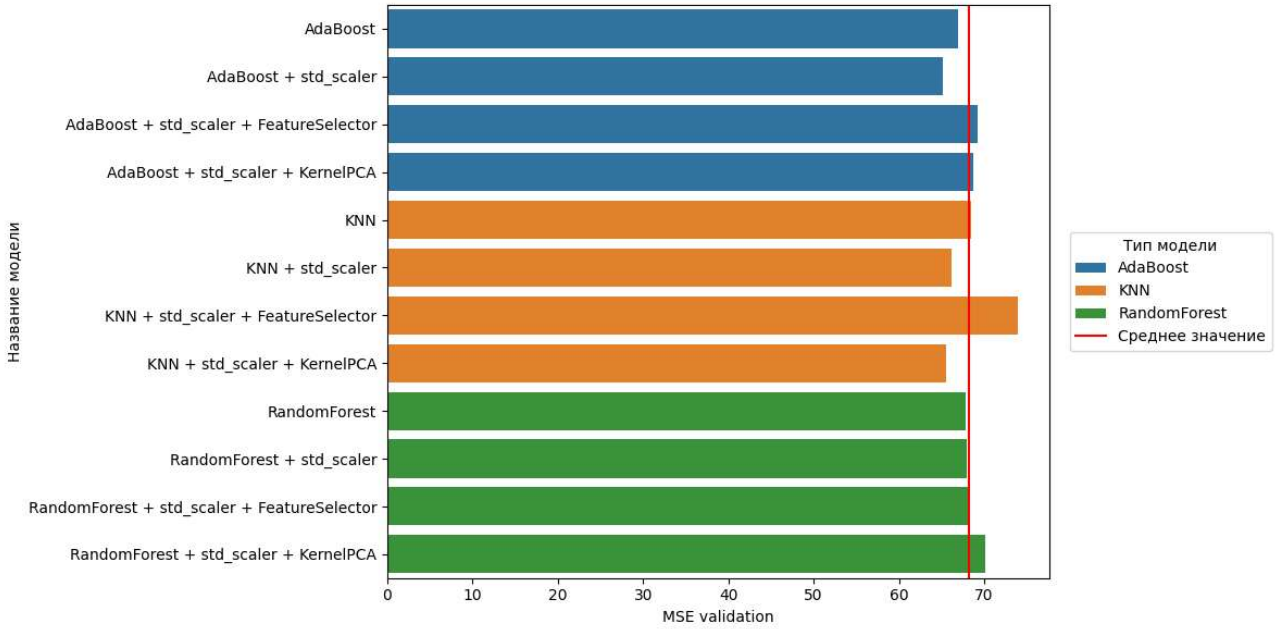
- Стандартизация + Выбор параметров

- Стандартизация + понижение размерности данных с помощью ядерных функций (KernelPCA).

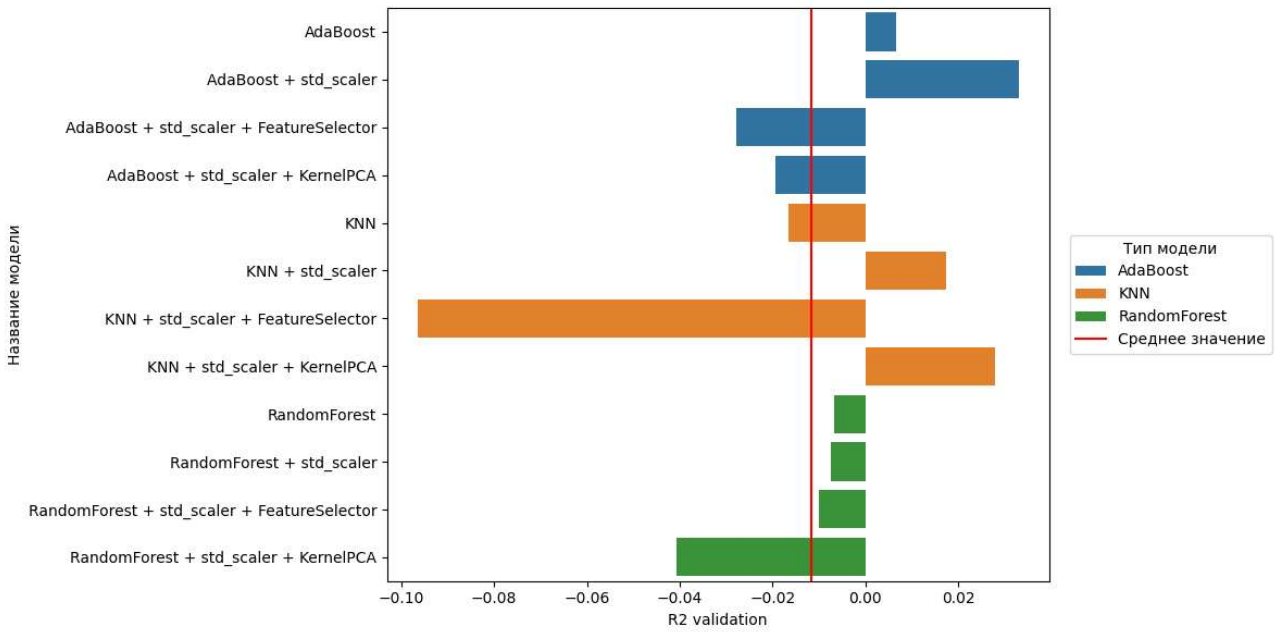
Важный момент лучшие параметры для KernelPCA подбирались на основании KNN регрессора и использовались для всех остальных моделей.



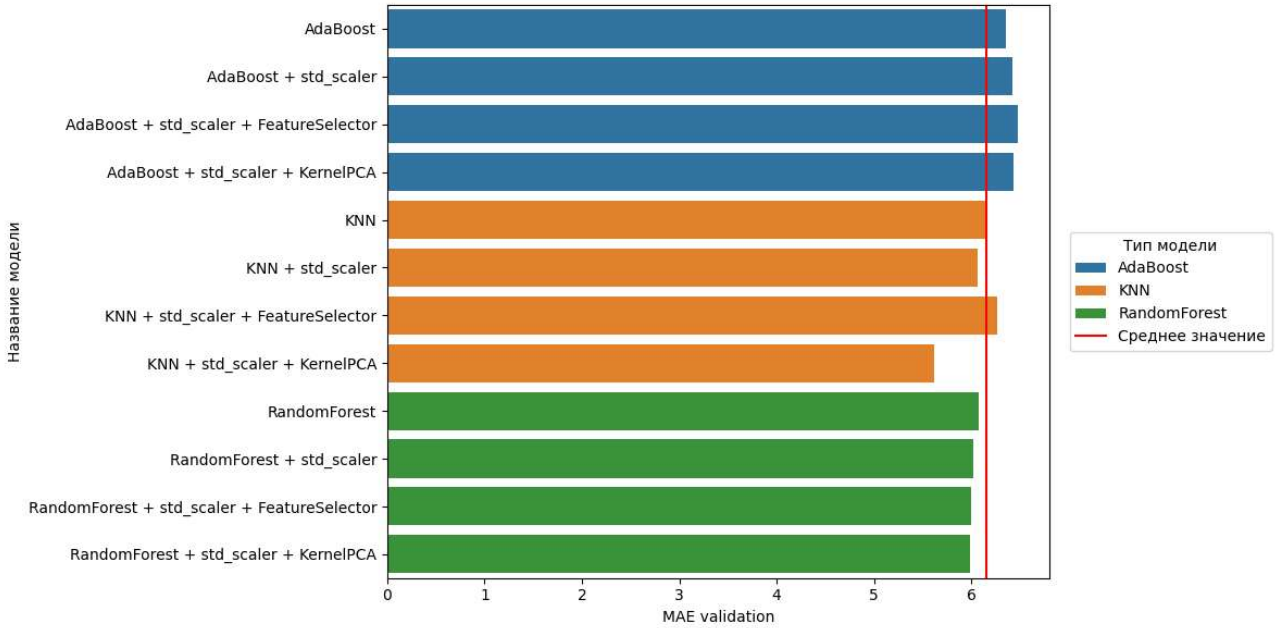
Лучшее значение метрики MSE на тестовых данных у модели RandomForest + std_scaler + Kernel PCA. Второе место у модели RandomForest + std_scaler, а третье у AdaBoost.



Лучшее значение MSE на валидационной выборке у AdaBoost + std_scaler. Второе место у KNN + std_scaler + KernelPCA. Третье занимает KNN + std_scaler.



Метрика R2 показывает, что все модели весьма плохо объясняют вариацию целевого значения. Лучшее значение метрики у AdaBoost + std_scaler. За ней идет KNN + std_scaler + KernelPCA.



По метрике MAE лучшее значение у модели KNN + std_scaler + KernelPCA.

По совокупности 3ех метрик лучшая модель - KNN + std_scaler + KernelPCA.

ЗАДАЧИ КЛАССИФИКАЦИИ

Метрики, используемые для оценки моделей и выбора лучших параметров в задачах классификации:

Для выбора лучших параметров с помощью GridSearchCV использовалась метрика ROC_AUC. Метрика ROC AUC оценивает, насколько хорошо модель различает два класса — положительный и отрицательный

Для сравнения моделий используются следующие метрики:

1. ROC_AUC.
2. Accuracy – доля правильно предсказанных ответов модели от общего количества предсказаний.
3. Precession - доля объектов, правильно классифицированных как положительные, среди всех объектов, которые модель предсказала как положительные.
4. Recall - метрика, которая показывает, какую долю объектов положительного класса модель правильно определила из всех реально существующих объектов этого класса.
5. F1 - метрика, которая показывает сбалансированное среднее между точностью (precision) и полнотой (recall).

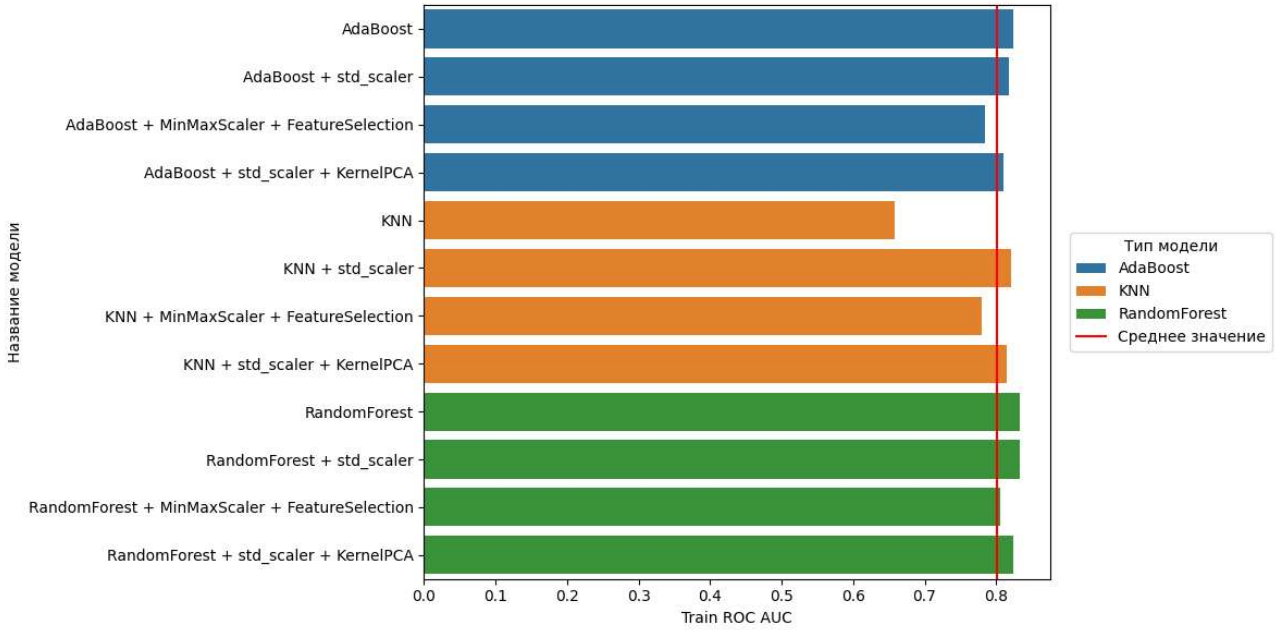
Для всех задач классификаций использовались следующие модели:
1. KneighborsClassifier
2. RandomForestClassifier
3. AdaBoostClassifier

Также выполнялись следующие шаги:
1. Чистые данные
2. Стандартизация
3. Масштабирование данных + выбор признаков

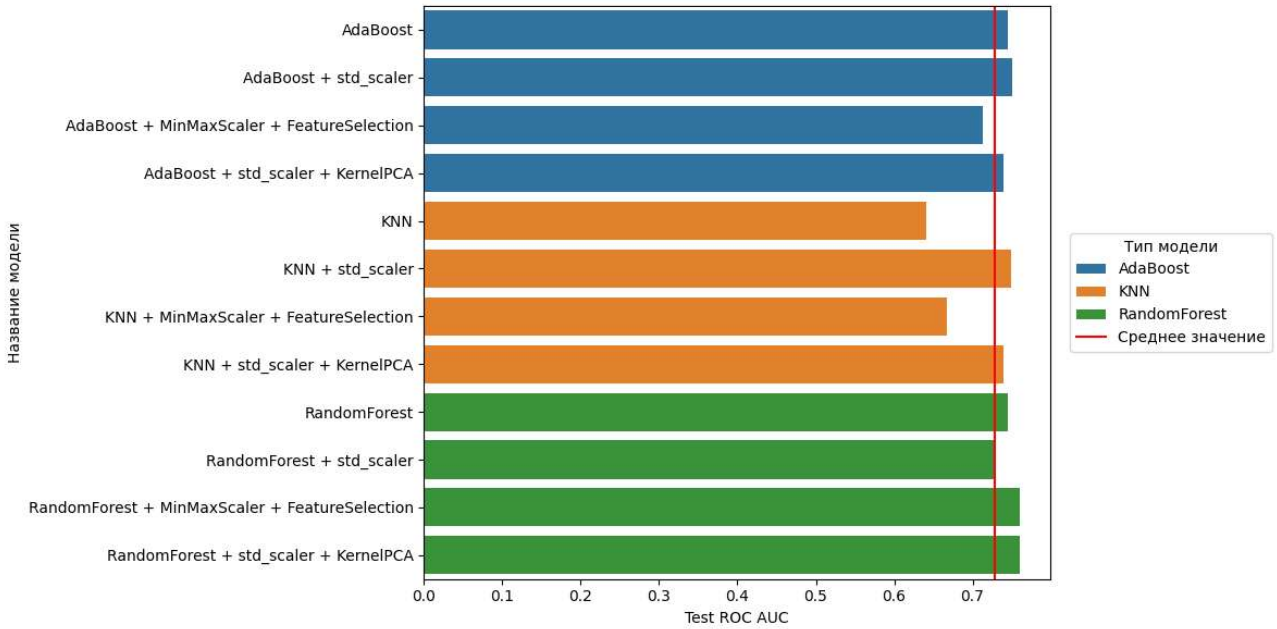
Масштабирование применялось, т. к. есть признаки с отрицательными значениями, что делает не возможным выбор признаков с помощью функции chi2.

4. Стандартизация + понижение размерности данных с помощью ядерных функций.

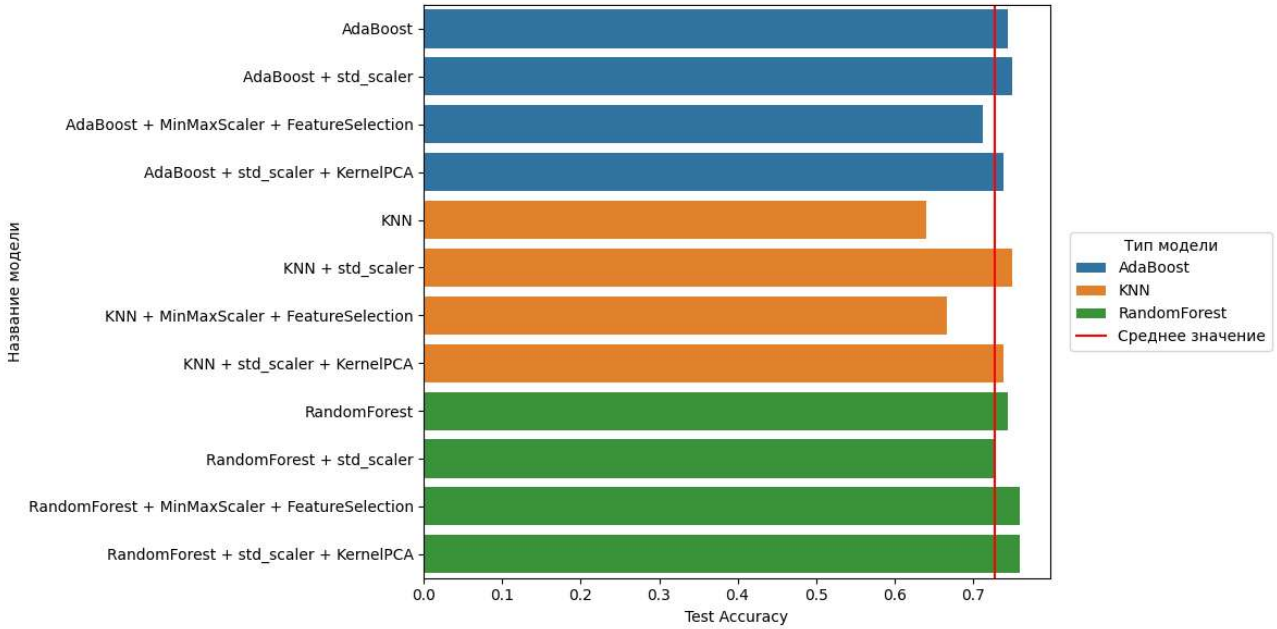
Классификация CC50 > медианы



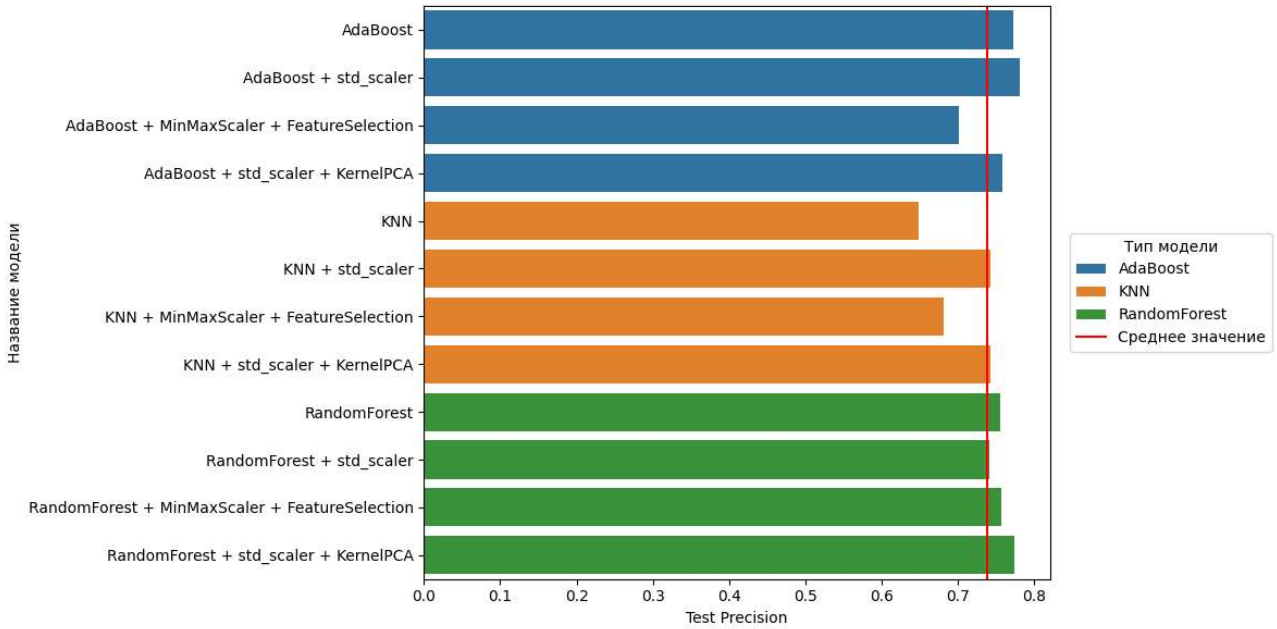
На тренировочных данных лучшее значение у моделей RandomForest и RandomForest + std_scaler. На третьем месте KNN + std_scaler.



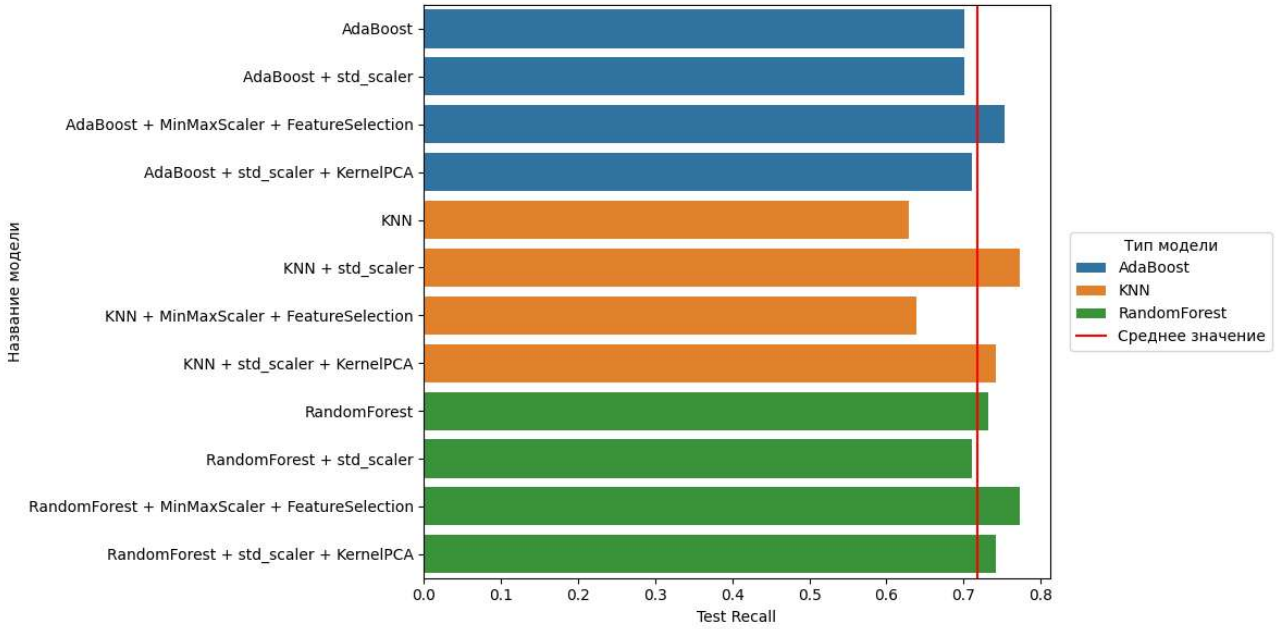
На валидационных данных лучшее значение у моделей RandomForest + MinMaxScaler + FeatureSelection и RandomForest + std_scaler + KernelPCA. На втором месте AdaBoost + std_scaler и KNN + std_scaler.



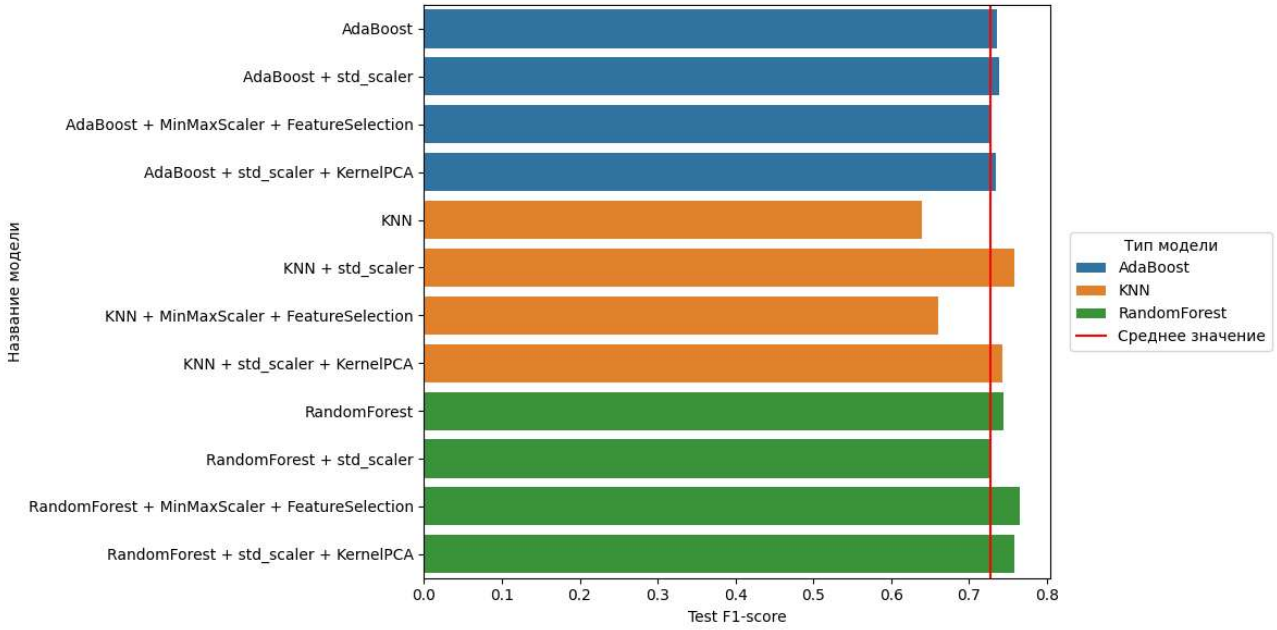
Лучшее значение метрики Accuracy у моделей RandomForest + MinMaxScaler + FeatureSelection и RandomForest + std_scaler + KernelPCA. На втором месте AdaBoost + std_scaler и KNN + std_scaler.



Лучшее значение по метрике Precession у модели AdaBoost + std_scaler. Второе место у RandomForest + std_scaler + KernelPCA и AdaBoost.



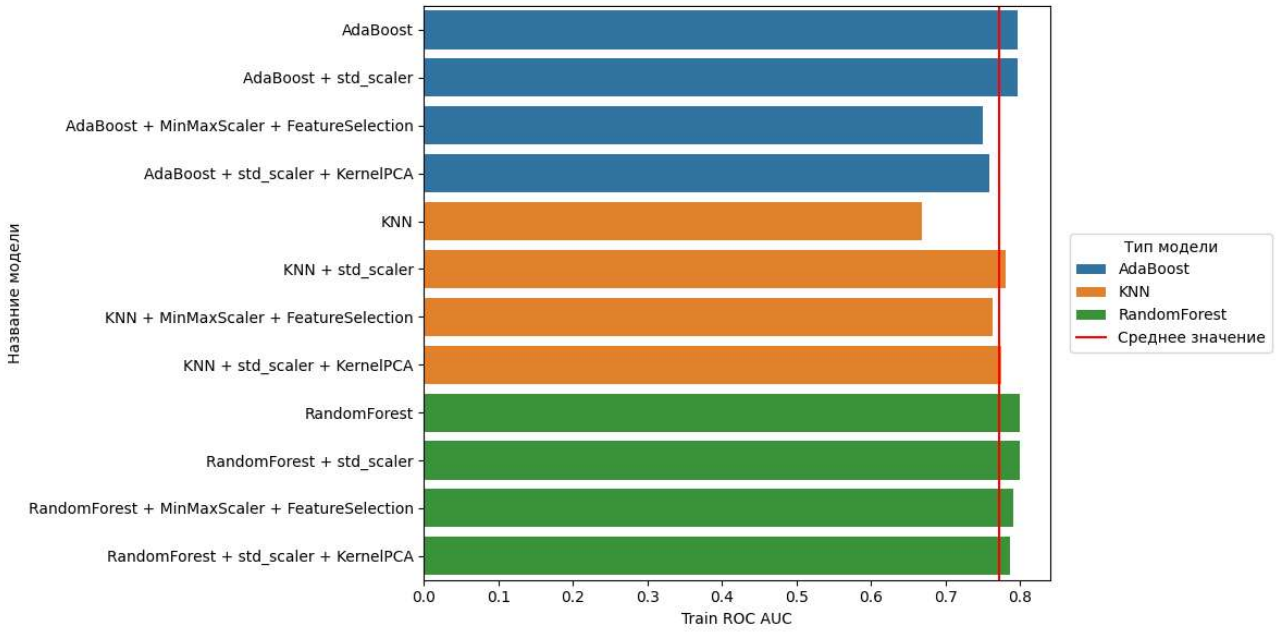
Первое место по метрике Recall занимают KNN + std_scaler и RandomForest + MinMaxScaler + FeatureSelector. Следом идет AdaBoost + MinMaxScaler + FeatureSelector.



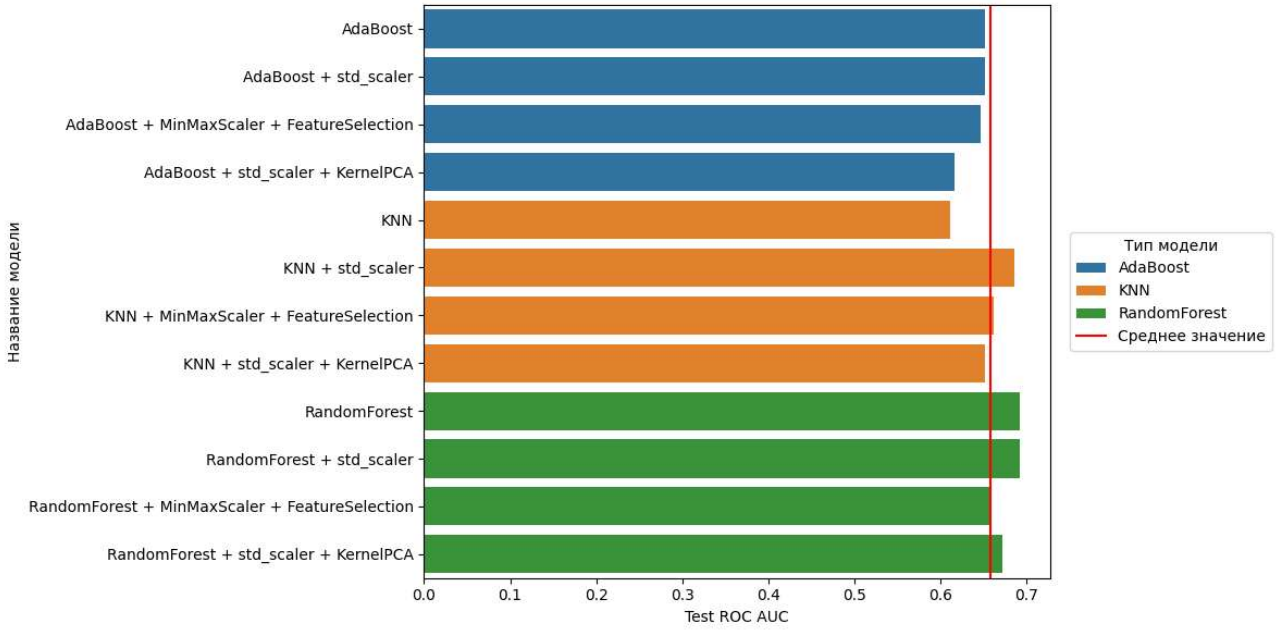
По метрике f1 первое место у RandomForest + MinMaxScaler + FeatureSelector, а второе место делят KNN + std_scaler и RandomForest + std_scaler + KernelPCA.

Исходя из представленных данных можно сделать вывод, что лучше моделью является RandomForest + MinMaxScaler + FeatureSelector.

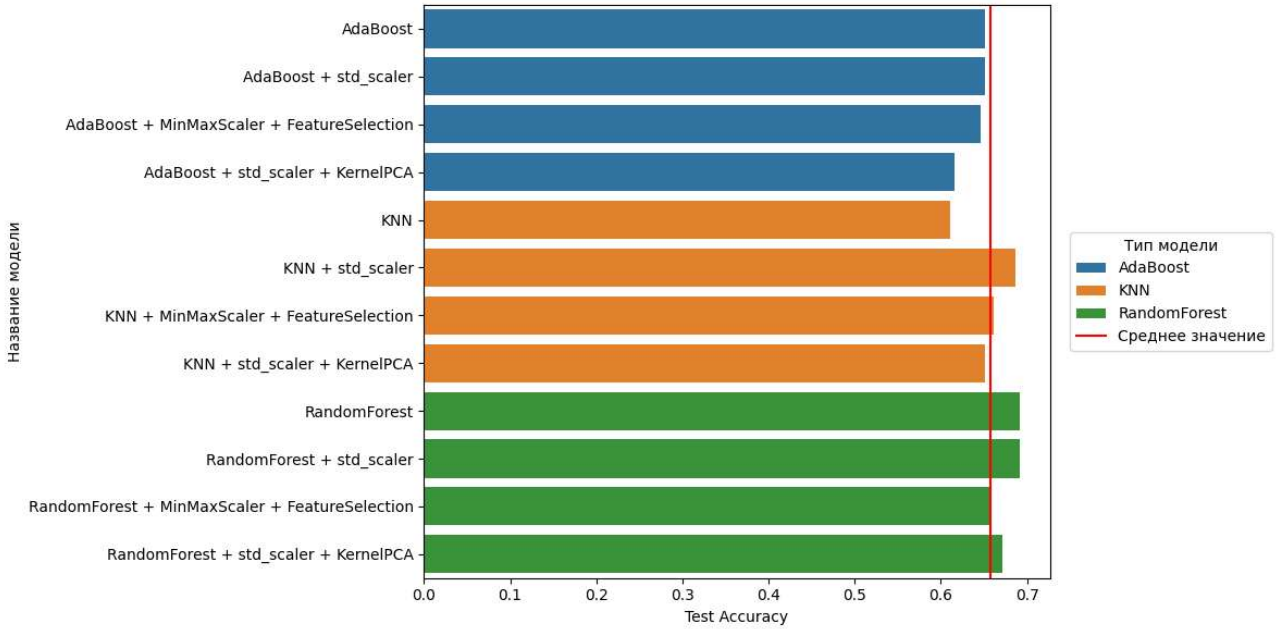
Классификация IC50 > медианы



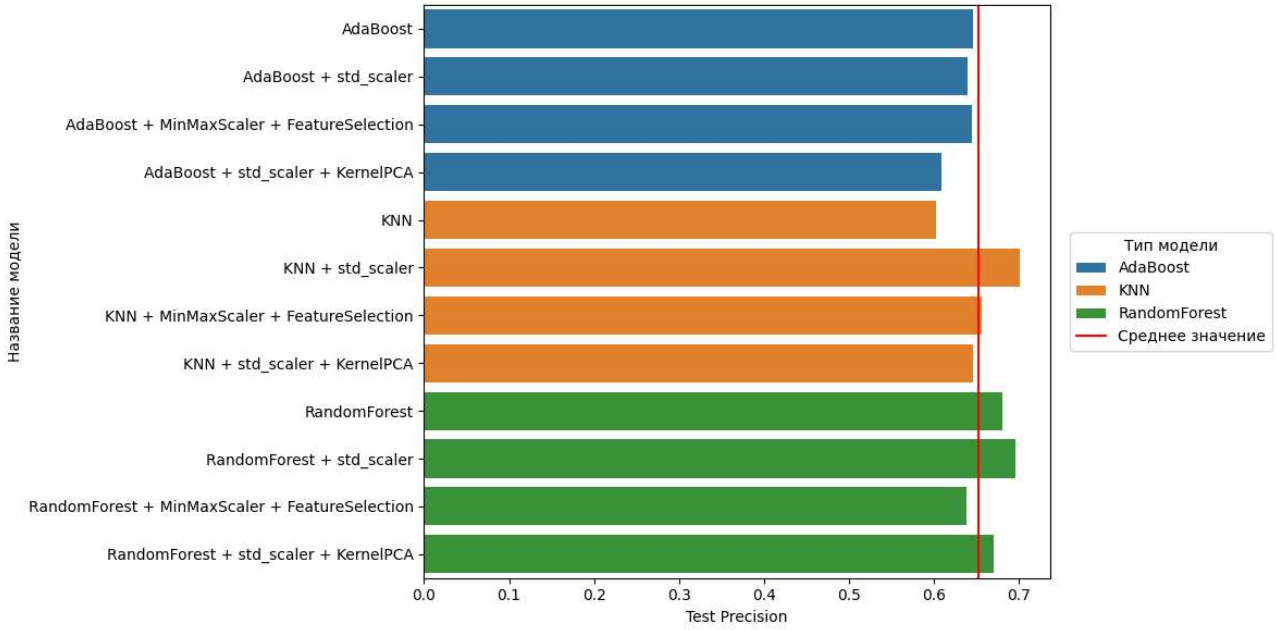
На тренировочных данных лучшей моделью является RandomForest + std_scaler. На самом деле 5 моделей показывают хороший показатель RandomForest, RandomForest + MinMaxScaler, AdaBoost, AdaBoost + std_scaler.



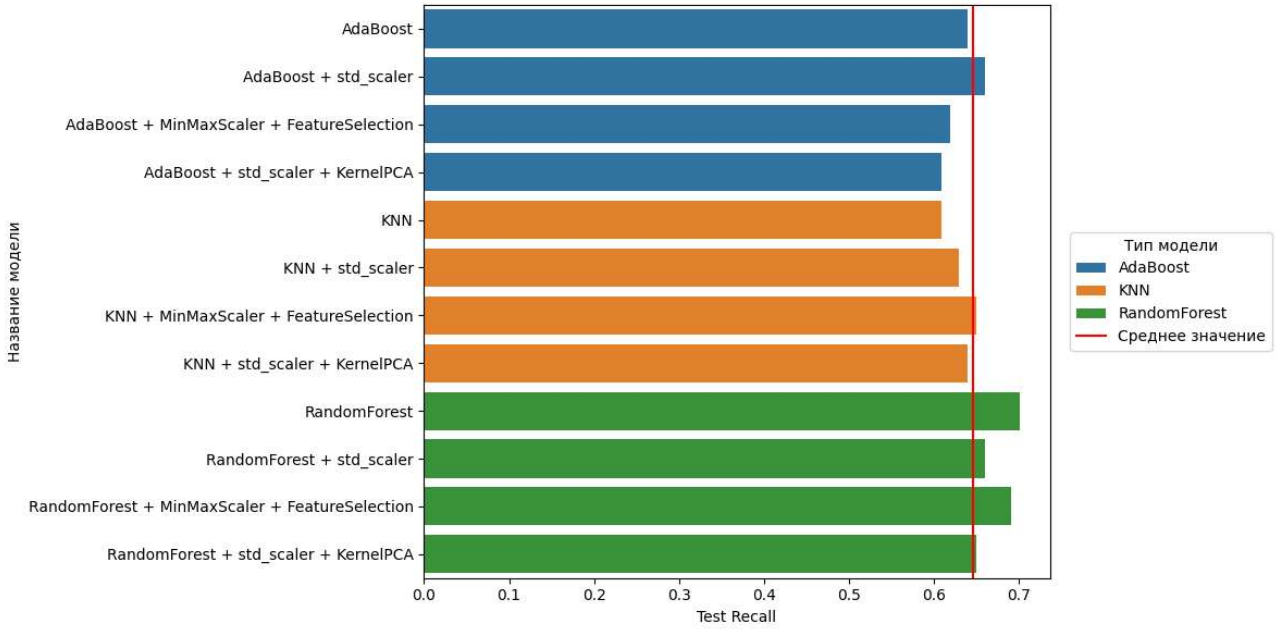
На валидационной выборке картина несколько меняется. Несмотря на то, что лидирующее место все еще за моделями RandomForest (в том числе с дополнительной предобработкой), на третьем месте появляется KNN + std_scaler. А модели AdaBoost начинают показывать значения ниже среднего.



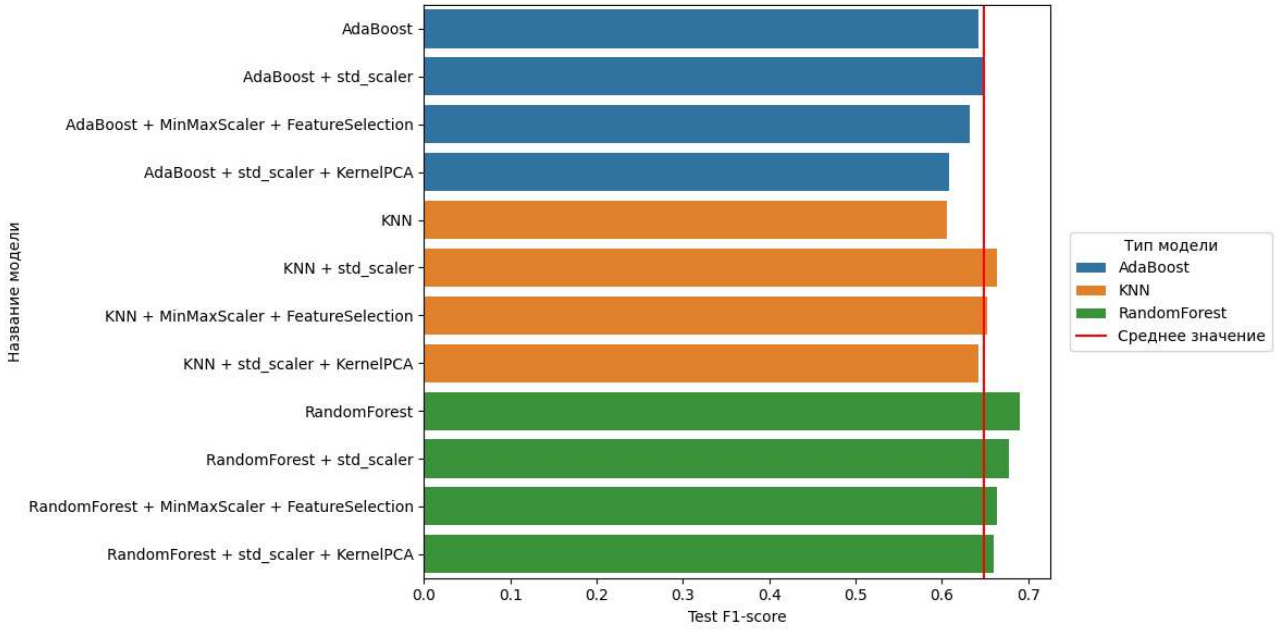
По метрике Accuracy картина такая же, как и по метрике ROC_AUC на валидационной выборке.



По метрике Precesion картина опять несколько меняется. Теперь лидирующее значение с небольшим значением за моделью KNN + std_scaler, а второе место за моделями RandomForest (исключение RandomForest + MinMaxSelector + FeatureSelector).



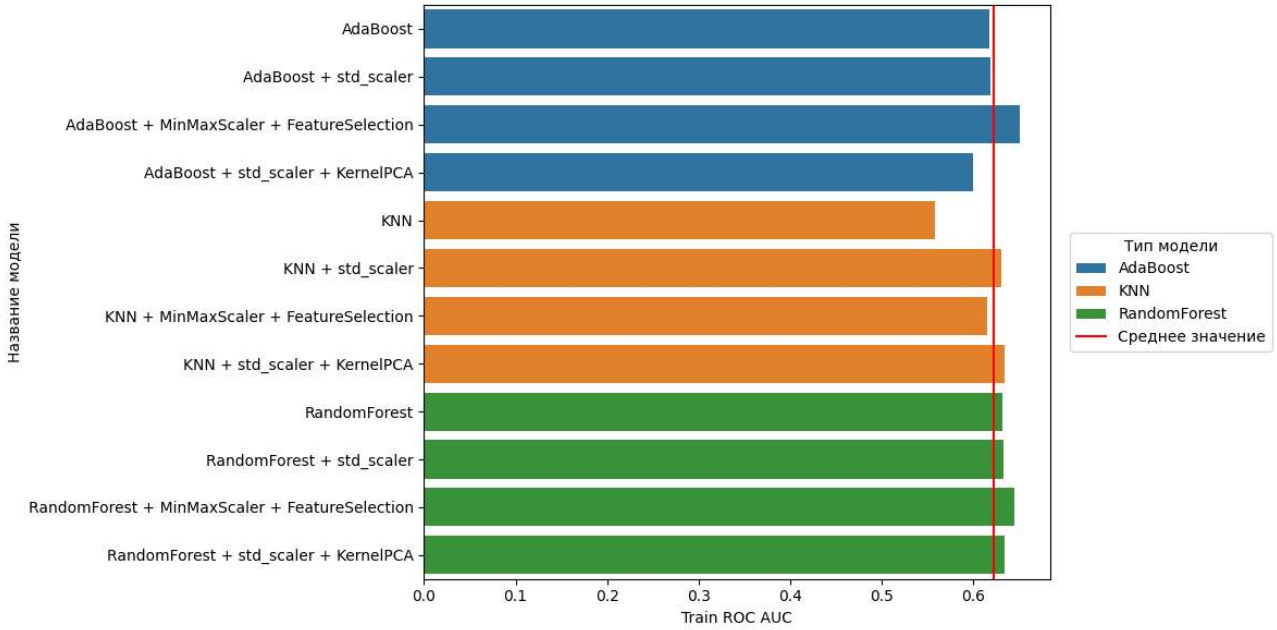
По метрике Recall абсолютное лидерство за моделями RandomForest. Следом идет AdaBoost + std_scaler.



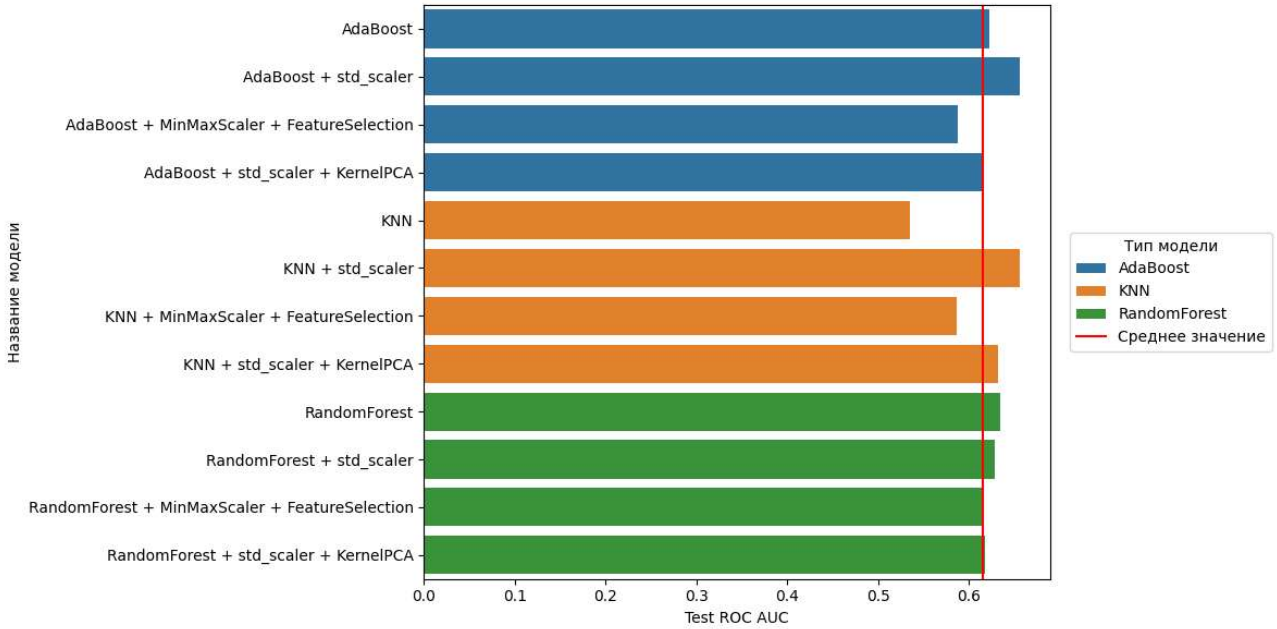
По метрике f1 все модели RandomForest показывают значение выше среднего. Следом за моделями RandomForest идет KNN + std_scaler.

Итого, лучшей моделью является RandomForest на чистых данных.

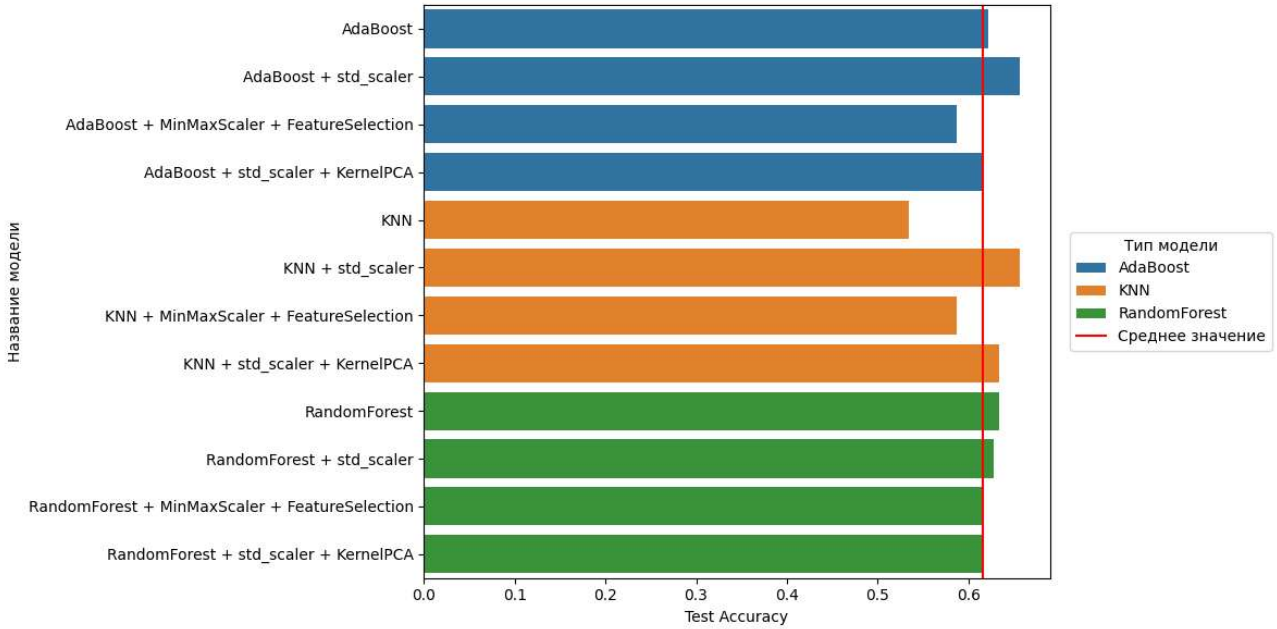
Классификация SI > медианы



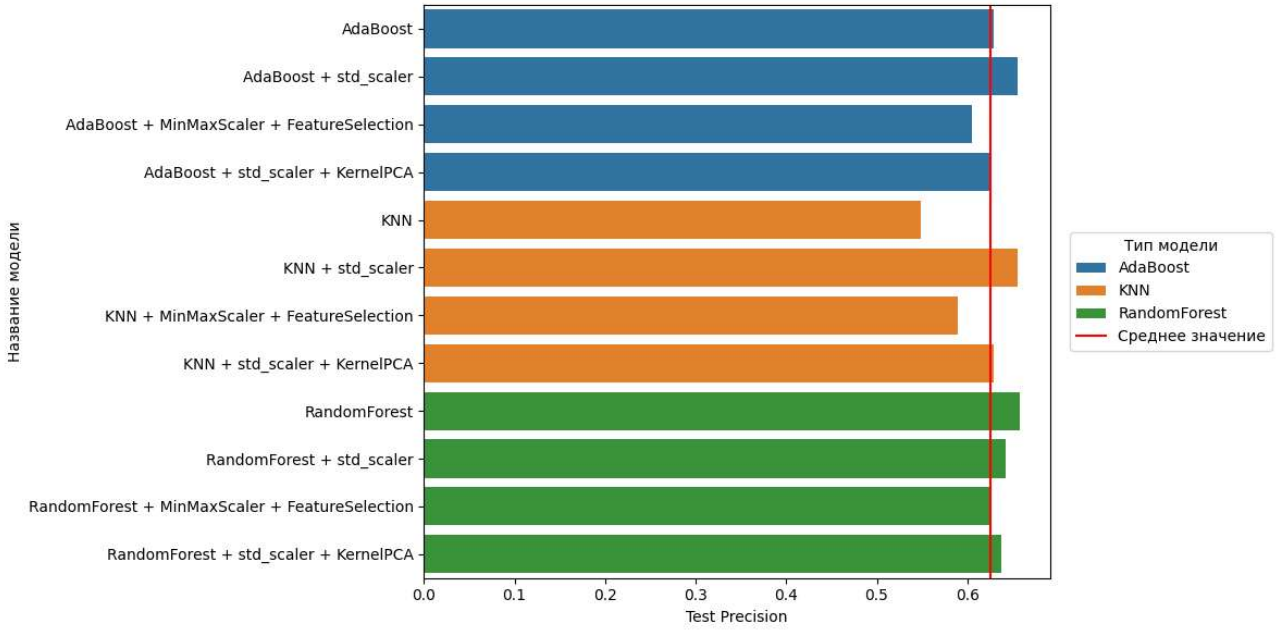
На тренировочных данных лучшая метрика ROC_AUC у моделей AdaBoost + MinMaxScaler + FeatureSelection и RandomForest + MinMaxScaler + FeatureSelection.



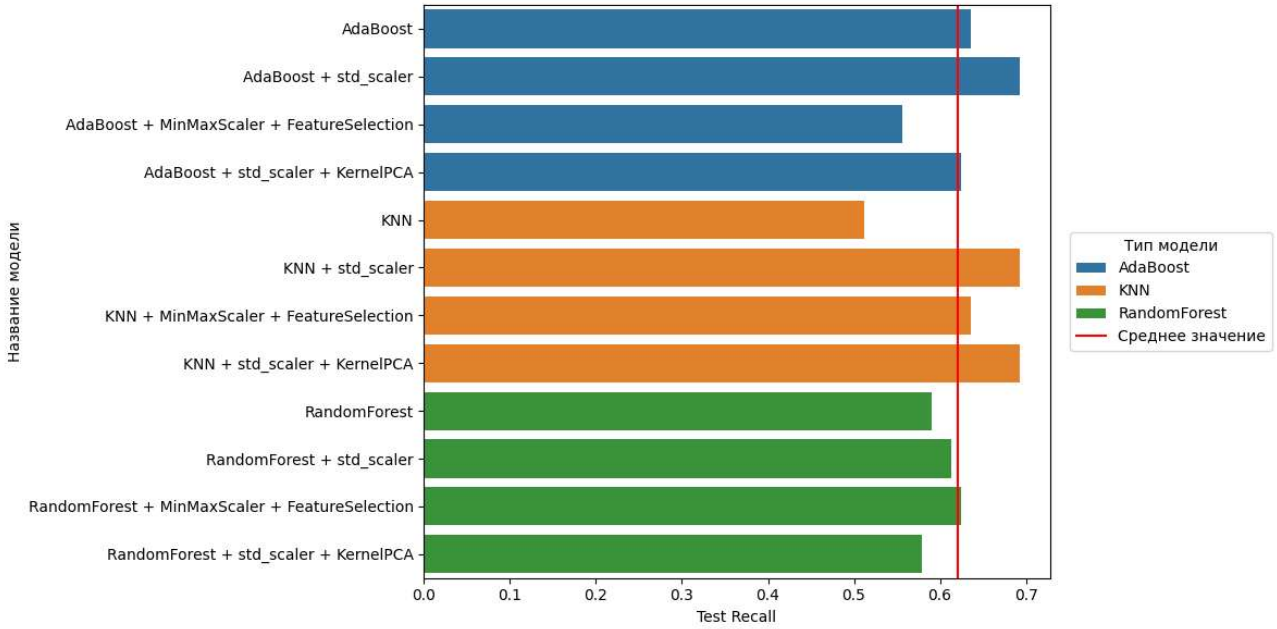
На валидационных данных картина существенно меняется — лидировавшая модель начинает показывать значения ниже среднего. Лучшая метрика у моделей AdaBoost + std_scaler и KNN + std_scaler (стоит отметить, что эта и остальне метрики у этих моделей абсолютно одинаковые). Второе место за моделью RandomForest.



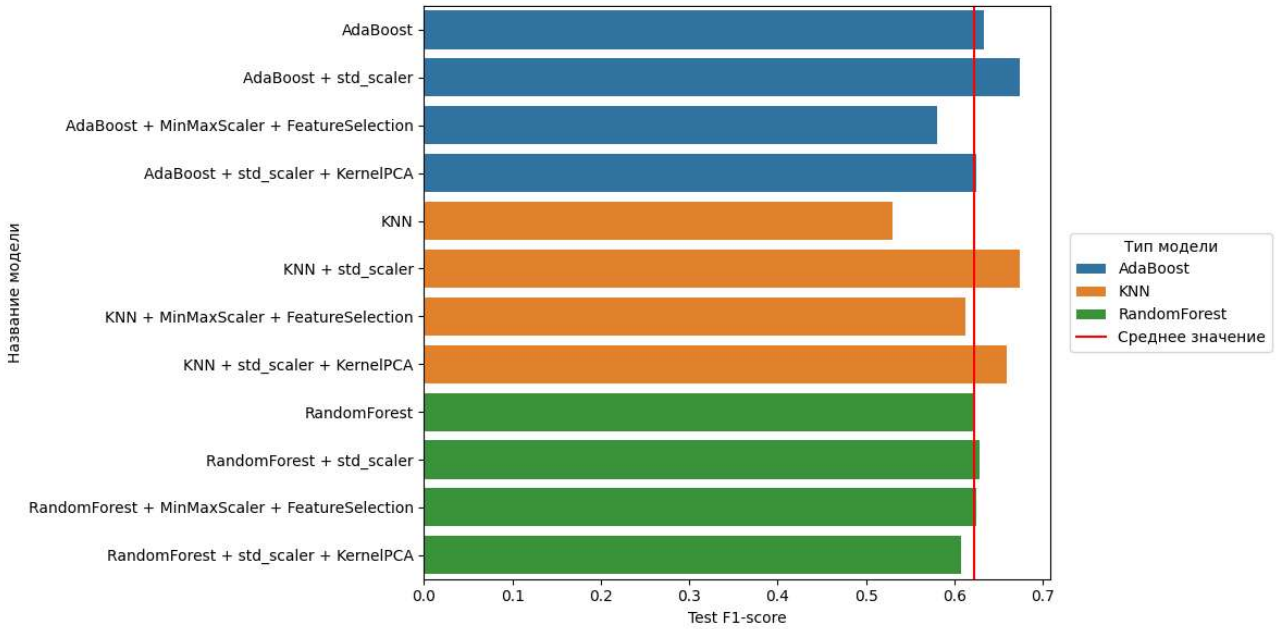
По метрике Accuracy ситуация идентичная.



По метрике Precision незначительно впереди моделль RandomForest, а следом идут модели AdaBoost + std_scaler и KNN + std_scaler.



На метрике Recall три модели показывают одинаково высокое значение - AdaBoost + std_scaler, KNN + std_scaler и KNN + std_scaler + KernelPCA.

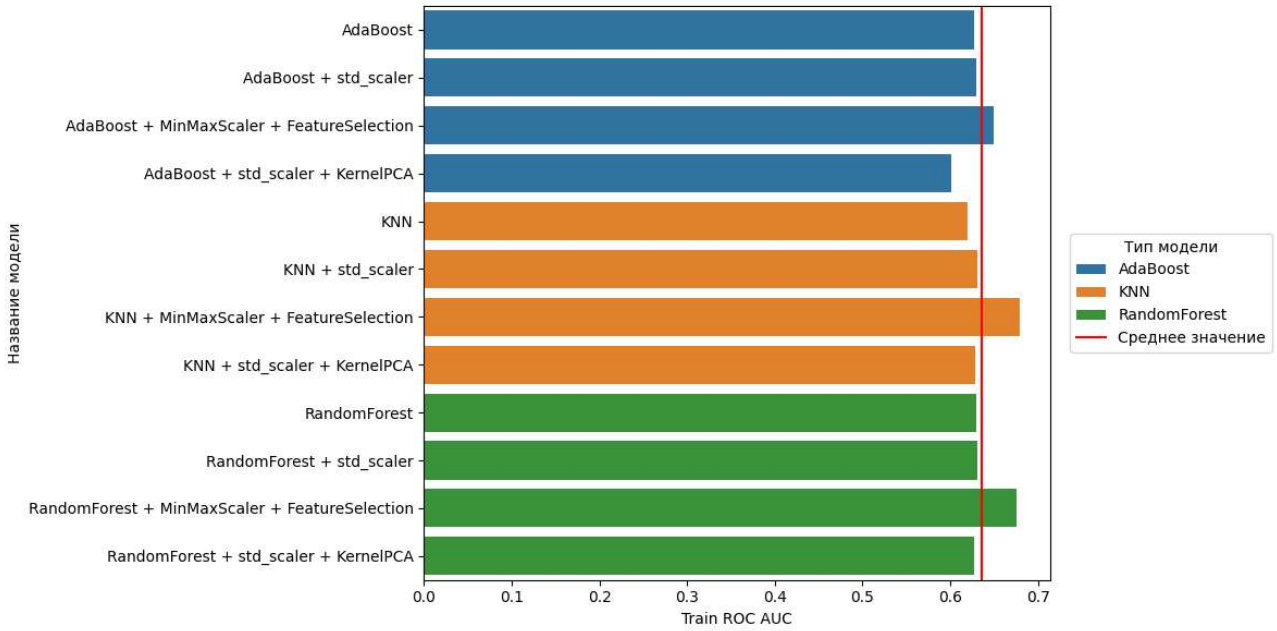


Лучшее значение метрики f1 остается за AdaBoost + std_scaler и KNN + std_scaler, а второе место за KNN + std_scaler + KernelPCA.

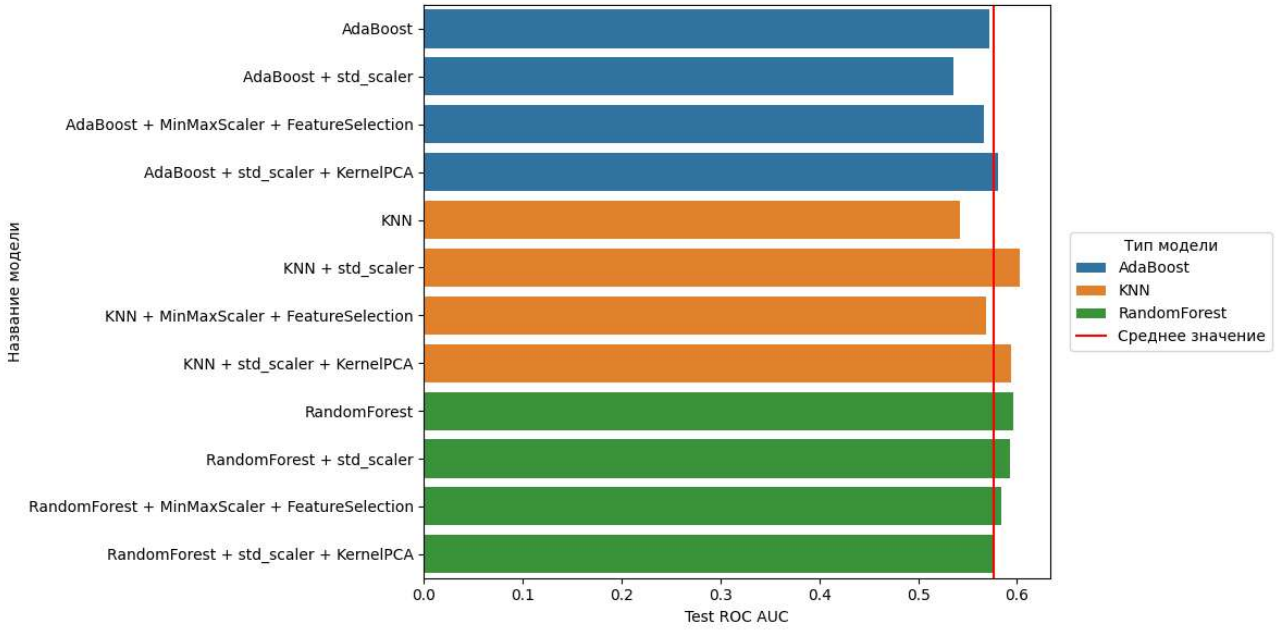
Итого: первое место за 2 моделями - AdaBoost + std_scaler и KNN + std_scaler. Но с учетом результата, полученного на тренировочной выброке предпотение стоит отдать KNN + std_scaler.

Классификация SI > 8

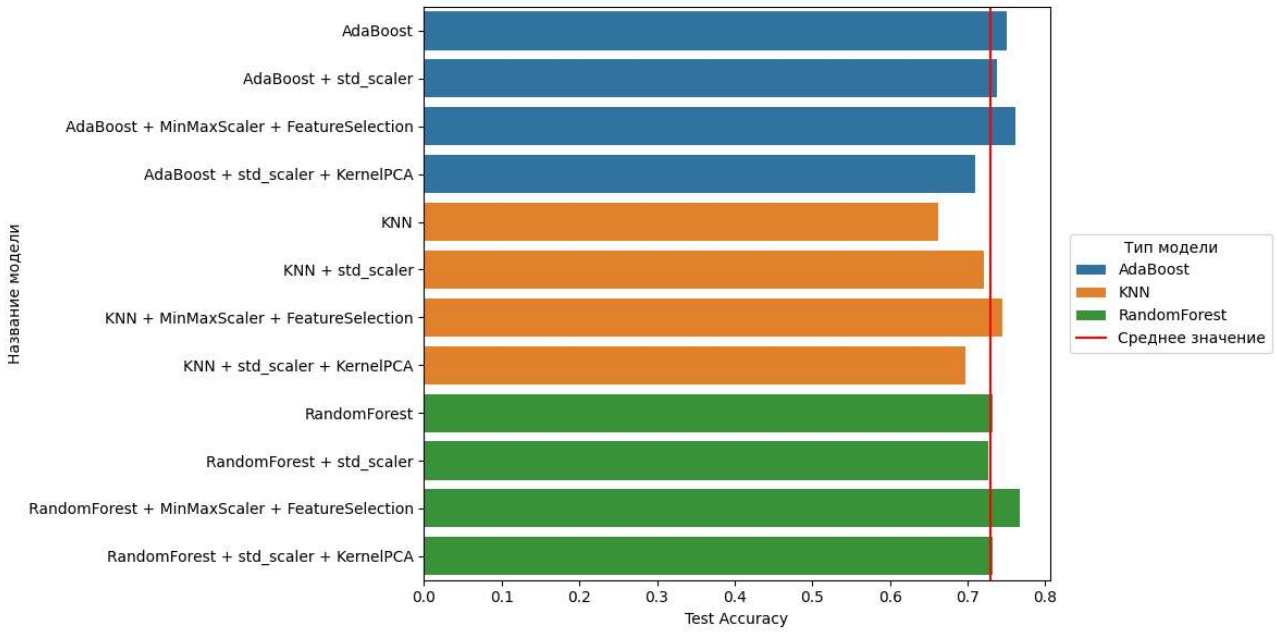
Особенность классификации SI > 8.
Все предыдущие задачи классификации делили выборку на 2 равные части, что следует исходя из смысла медианы. В данном случае мы делим выборку не по медиане, что приводит к тому, что классы у нас есть преобладание одного класса над другим. Большая часть данных находится в значениях до 8.



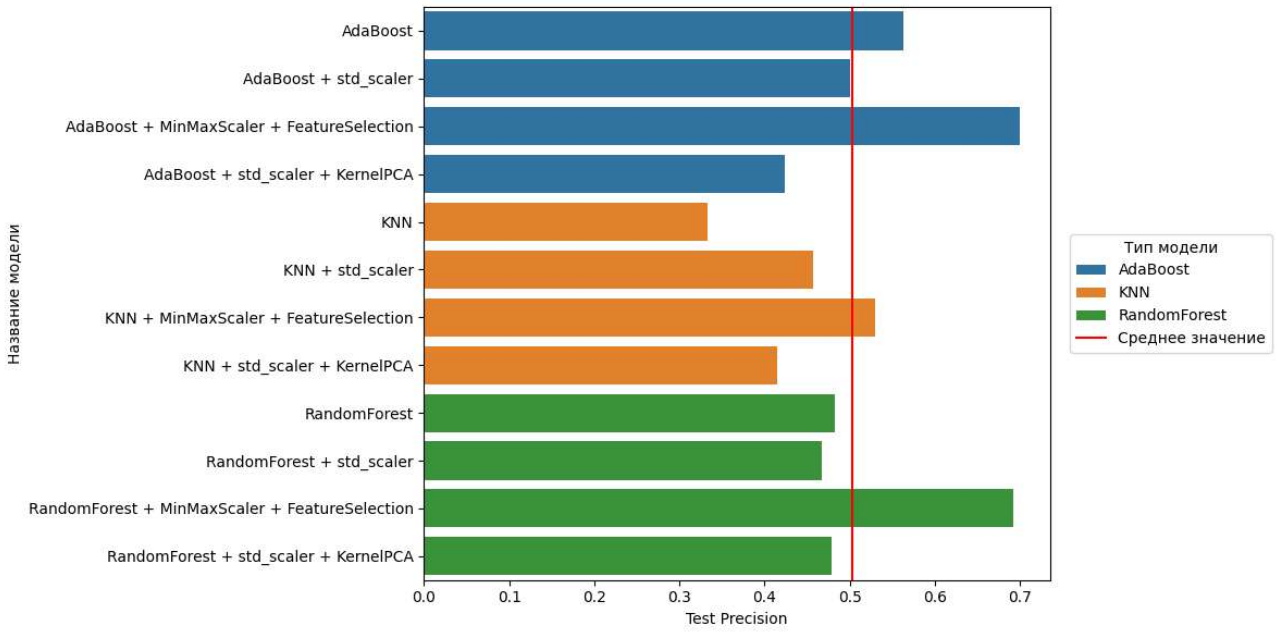
Лучшая метрика на тренировочных данных у модели KNN + MinMaxScaler + FeatureSelection. Следом с незначительным отрывом идет RandomForest + MinMaxScaler + FeatureSelection. На третьем месте AdaBoost + MinMaxScaler + FeatureSelection.



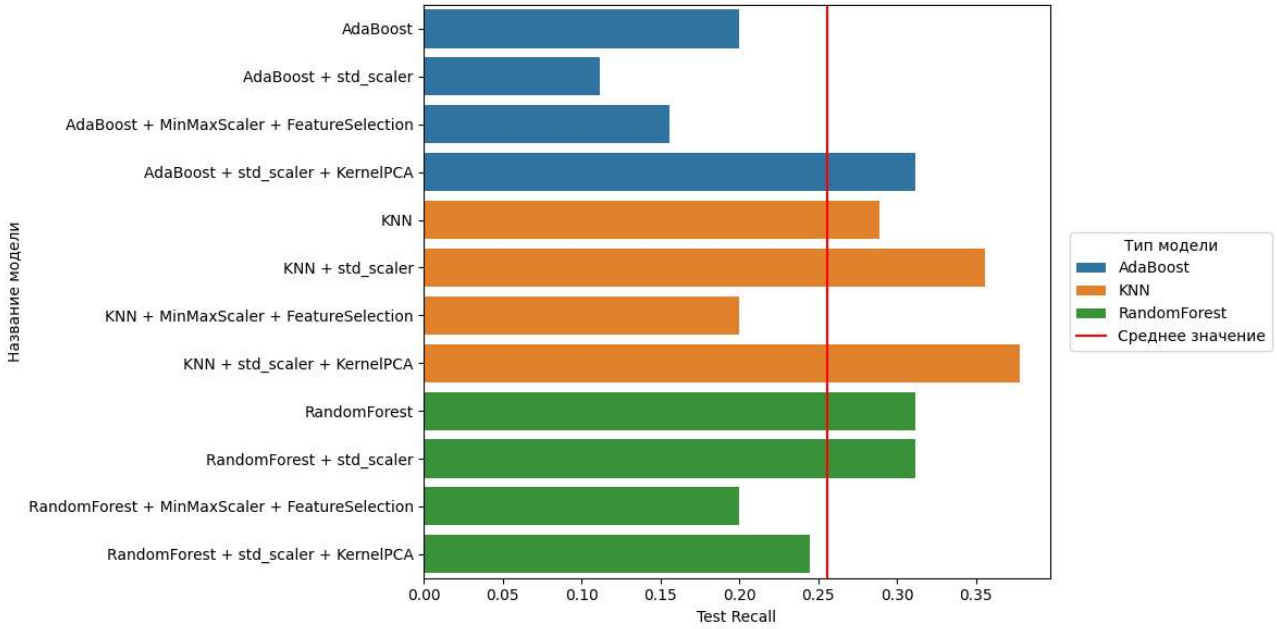
На валидационных данных лучшая метрика у KNN + std_scaler. Второе место у RandomForest, а третье у KNN + std_scaler + KernelPCA.



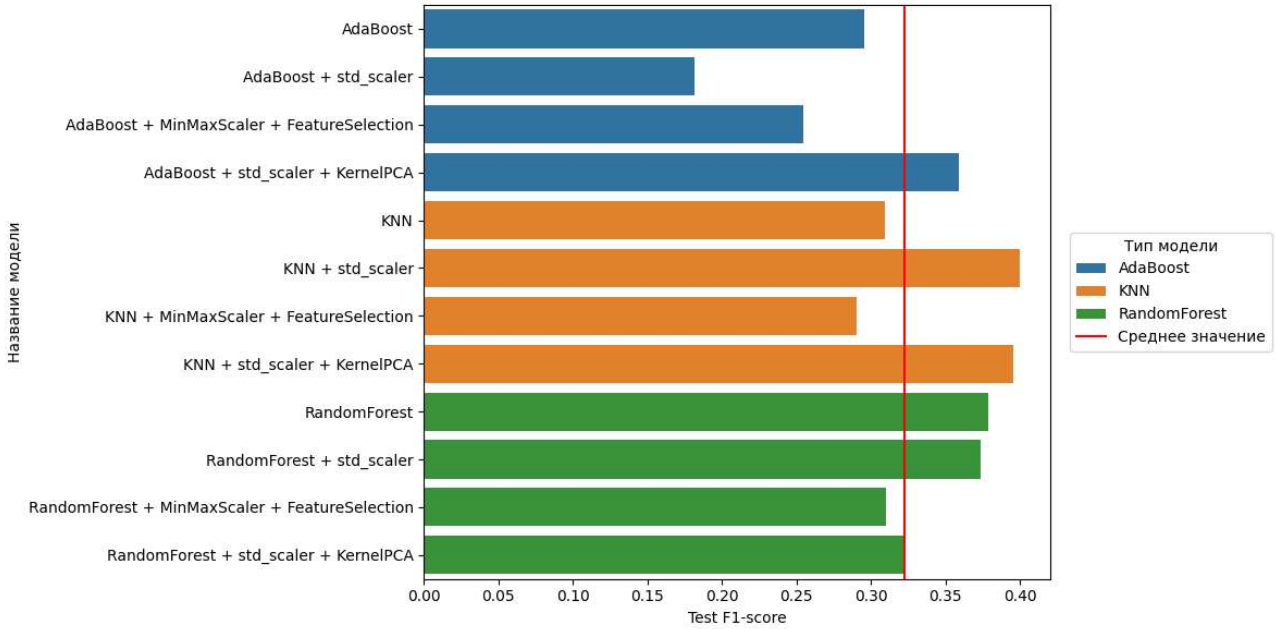
Метрика Accuracy имеет лучшее значение для RandomForest + MinMaxScaler + FeatureSelection и AdaBoost + MinMaxScaler + FeatureSelection. Далее идут модели AdaBoost и KNN + MinMaxScaler + FeatureSelection.



Собственно на метрике Precision мы начинаем видет влияние дисбаланса классов. Модели стремтся отнести все значения в больший класс. Хороший показатель у 2 моделей - RandomForest + MinMaxScaler + FeatureSelection и AdaBoost + MinMaxScaler + FeatureSelection.



Метрика Recall еще силнее показывает влияние дисбаланса классов. Лучшее значение у моделей KNN + std_scaler + KernelPCA и KNN + std_scaler, они лучше классифицируют класс, меньший по количеству записей.



Аналогичную историю мы видем на метрике f1.

В целом, для данной задачи не хватило этапа уравнивания класса (стоило использовать SMOTE Synthetic Minority Over-sampling Technique) , для того, чтобы можно было корректно выбирать модель.

Исходя из представленных результатов предпочтение все же стои отдать моделям RandomForest или RandomForest + std_scaler, как наиболее сбалансированным.