ГЛАВА 1

Выбор наборов гиперпараметров

В данной главе будет показан процесс выбора набора возможных значений для каждого гиперпараметра алгоритмов машинного обучения, которые используются в данный программе.

Для того чтобы программа находила более оптимальные значения гиперпараметров как можно быстрее требуется взять для каждого гиперпараметра такие наборы возможных значений, которые будут давать наилучшие результаты. Но некоторые алгоритмы имеют слишком много комбинаций значений, поэтому нужно каким-то образом их сократить.

Для этого будет проделан следующий порядок действий:

1. Берётся набор датасетов определённого класса задач, на которых будут проверяться разные комбинации гиперпараметров.

2. Выбираем алгоритм машинного обучения для данного класса задач.

3. После чего берётся набор гиперпараметров, которые наиболее значимы для хорошего результата этого алгоритма.

4. Для каждого гиперпараметра из этого набора, берётся множество возможных значений, которое может принимать этот гиперпараметр, назовём его начальным набором значений.

5. После чего мы получаем множество всех возможных комбинаций значений гиперпараметров для выбранного алгоритма.

6. Через функцию **BayesSearchCV()** из библиотеки **scikit-optimize**, которая работает на принципах Байсовской оптимизации, на каждом датасеты будет испробовано 50 различных комбинаций гиперпараметров, через которые выше названная функция на каждой итерации будут стремиться улучшить точность алгоритма.

7. После того как результаты работы функции **BayesSearchCV()** на каждом датасеты будут получены, для каждой комбинации воз-

можных значений гиперпараметров будет взято среднее арифметическое от их точности на каждом датасете.

8. Получив среднею точность для испробованных комбинаций строится график, где по оси X будет номер комбинации, а по оси Y точность.

Получив такой график нужно снизить количество возможных комбинаций и при этом чтобы этих точек на графике осталось как можно больше и у них была хорошая точность, потому что эти точки, которые обозначают точность комбинации гиперпараметров, были получены на основании Байсовской оптимизации из-за чего можно предположить что эти комбинации гиперпараметров являются наиболее выгодными для получения высокой точности.

Данную задачу можно решить следующим образом, посмотреть для каждого гиперпараметра какие его значения встречаются реже всего или они показывают плохую точность, и после чего их убрать. Таким образом получается итоговый набор значений гиперпараметров. После этого будет построен график с точками, которые остались(они будут обозначены синим), и точками,которые были потеряны(они будут обозначены красным).

1.1 Алгоритмы регрессии

Для вычисления оптимальных возможных значений гиперпараметров алгоритмов регрессии используется 20 датасетов, таблица с ними будет приведена ниже.

ТАБЛИЦА

Метрикой качества здесь будет являться коэффициент детерминации, или коэффициент R^2: $R^2 = 1 - \frac{\sum_{i=1}^{l}(a(x_i)-y_i)^2}{\sum_{i=1}^{l}(y_i-\overline{y})^2}$

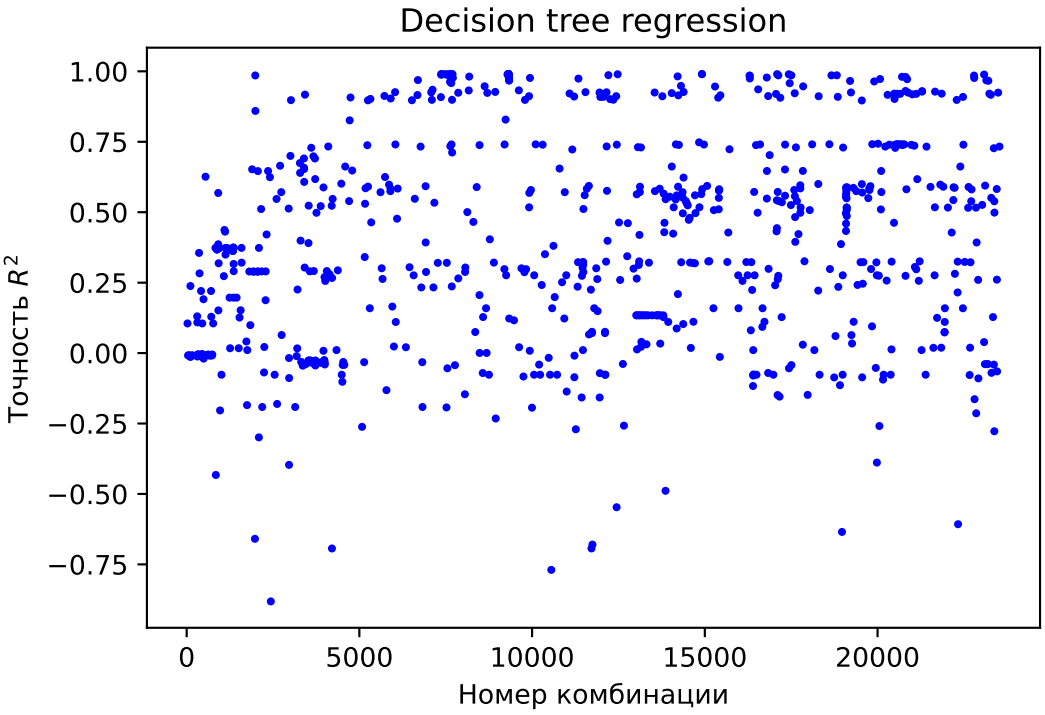

Рис. 1.1: Начальный набор значений гиперпараметров для алгоритма Decision tree regressor.

1.1.1 Дерево решений (Decision Tree)

Рассмотрим для данного алгоритма следующий начальный набор значений:

- *max_depth:* [1, 29]
- *min_samples_split:* [2, 29]
- *min_samples_leaf:* [1, 29]

Соответственно получая 23 548 возможных комбинаций значений гиперпараметров.

После того как мы избавимся от значений гиперпараметров, которые встречаются реже всего или имеют низкий показатель точности, получим следующие:

- *max_depth:* [6, 19]

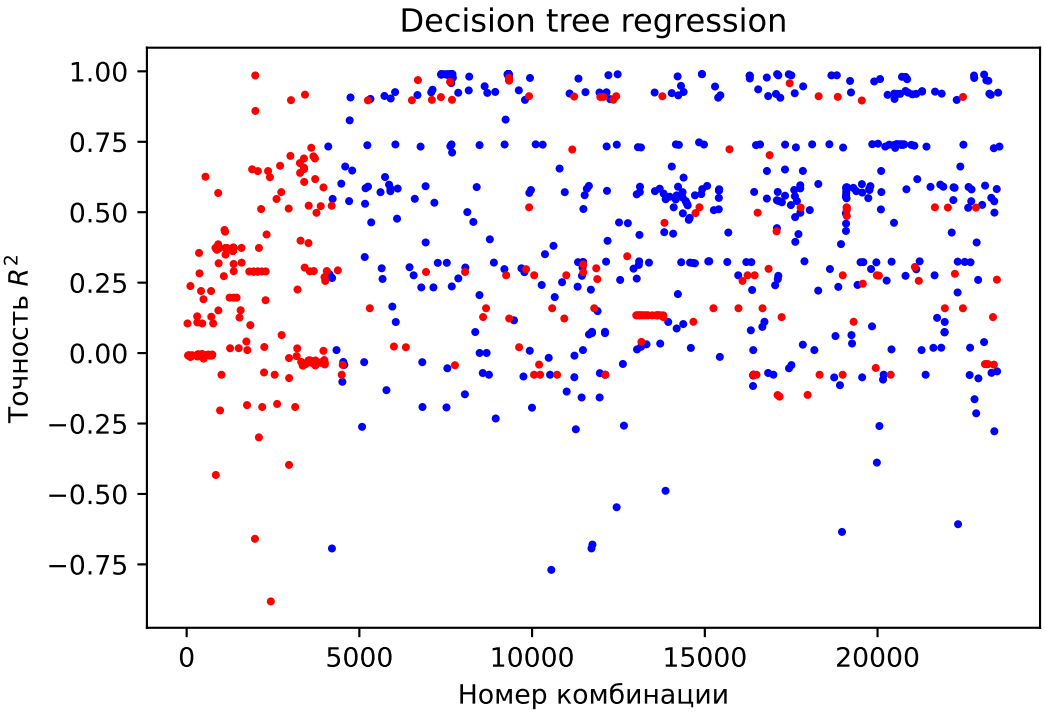

Рис. 1.2: Итоговый набор значений гиперпараметров для алгоритма Decision tree regressor.

- *min_samples_split:* [2, 29]
- *min_samples_leaf:* [1, 23]

Теперь данный алгоритм имеет 15 456 возможных комбинации значений гиперпараметров. 60% точек осталось.

1.1.2 Случайные и сверхслучайные деревья решения (Random Forest and Extra Trees)

Здесь будет рассмотрено сразу два алгоритма Random Forest и Extra Trees, так как это схожие алгоритмы и начальный набор значений гиперпараметров для них был один тот же. Для данных алгоритмов следующий начальный набор значений:

- *max_features:* [0.10, 1.0], с шагом 0.05
- *min_samples_split:* [2, 29]

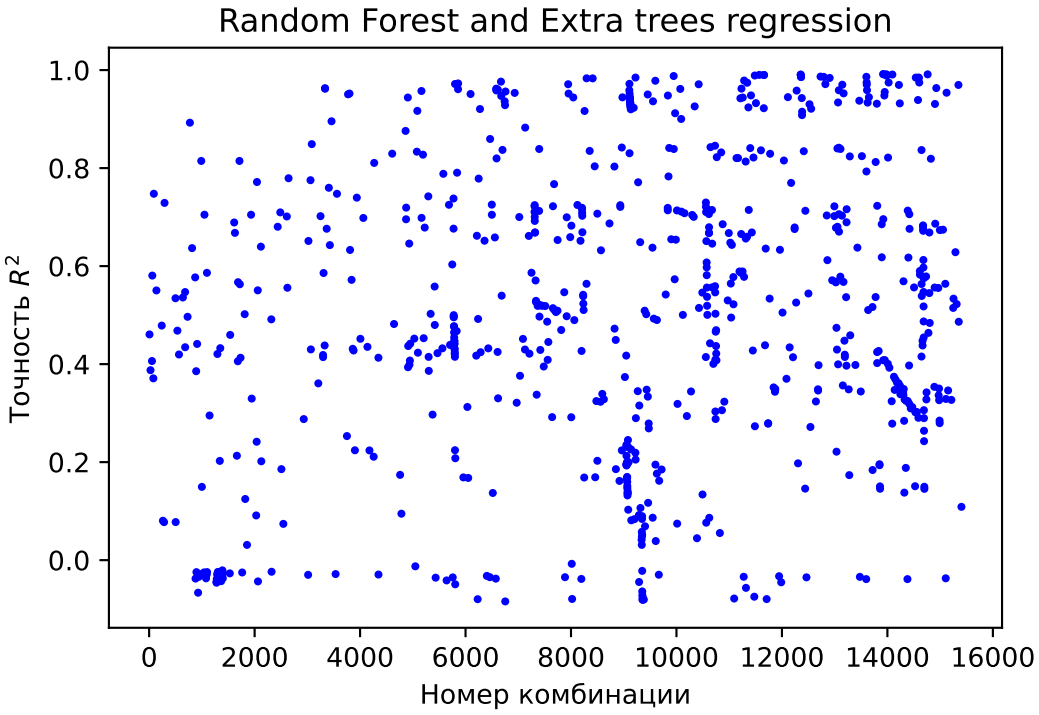

Рис. 1.3: Начальный набор значений гиперпараметров для алгоритмов Random forest regressor и Extra trees regressor.

- *min_samples_leaf:* [1, 29]

Соответственно получая 15 428 возможных комбинаций значений гиперпараметров.

После того как мы избавимся от значений гиперпараметров, которые встречаются реже всего или имеют низкий показатель точности, получим следующие возможные значения:

- *max_features:* [0.4, 1.0], с шагом 0.05
- *min_samples_split:* [2, 25]
- *min_samples_leaf:* [1, 22]

Теперь данные алгоритмы имеют 6 864 возможных комбинации значений гиперпараметров. 60% точек осталось.

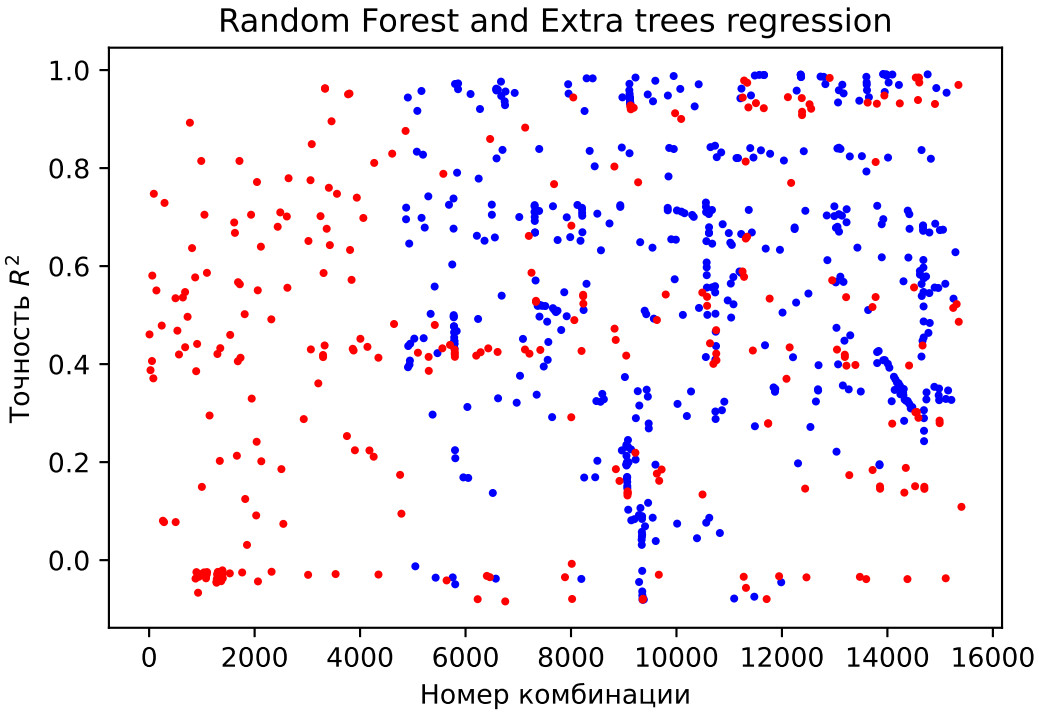

Рис. 1.4: Итоговый набор значений гиперпараметров для алгоритмов Random forest regressor и Extra trees regressor.

1.1.3 Градиентный бустинг (Gradient boosting)

Рассмотрим для данного алгоритма следующий начальный набор значений гиперпараметров:

- *learning_rate:* {1e-4, 1e-3, 1e-2, 1e-1, 0.5, 1}
- *max_depth:* [1, 19]
- *min_samples_split:* [2, 19]
- *min_samples_leaf:* [1, 24]
- *subsample:* [0.2, 1.00], с шагом 0.05
- *max_features:* [0.20, 1.0], с шагом 0.05

Соответственно получая 13 395 456 возможных комбинаций значений гиперпараметров. На Рис.1.5 показаны не все точки, это связанно с тем,

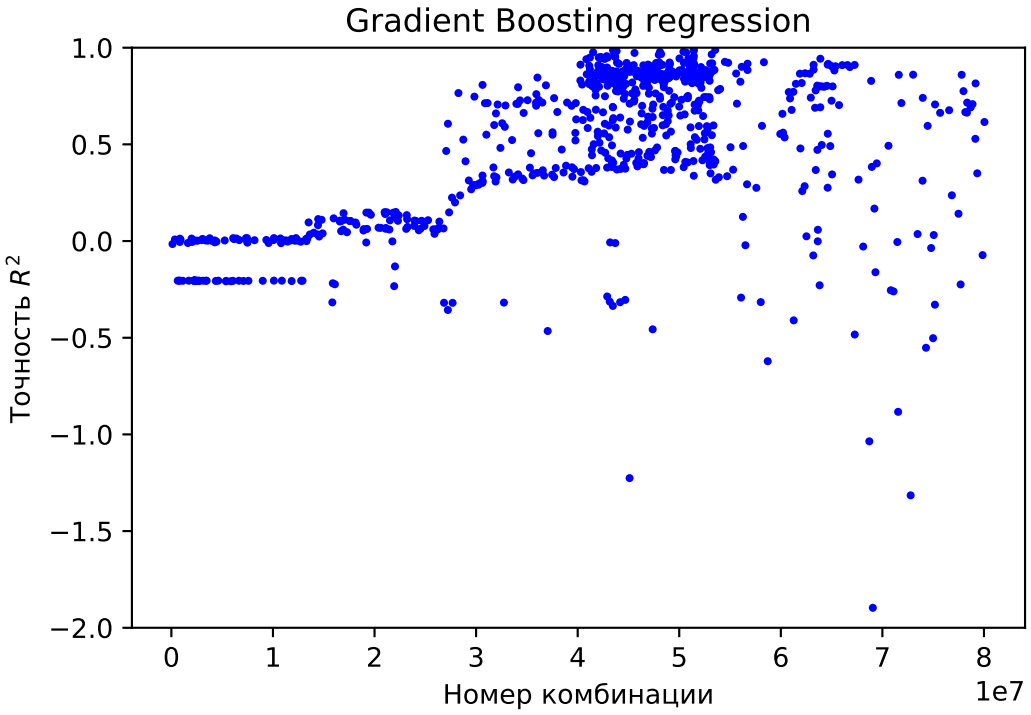

Рис. 1.5: Начальный набор значений гиперпараметров для алгоритма Gradient Boosting regression.

что имеются выбросы с большими отрицательными значениями.

После того как мы избавимся от значений гиперпараметров, которые встречаются реже всего или имеют низкий показатель точности, получим следующие возможные значения:

- *learning_rate:* {1e-2, 1e-1, 0.5}
- *max_depth:* [1, 19]
- *min_samples_split:* [2, 19]
- *min_samples_leaf:* [1, 24]
- *subsample:* [0.2, 1.00], с шагом 0.05
- *max_features:* [0.20, 1.0], с шагом 0.05

Теперь данный алгоритм имеет 6 697 728 возможных комбинации значений гиперпараметров. 73% точек осталось.

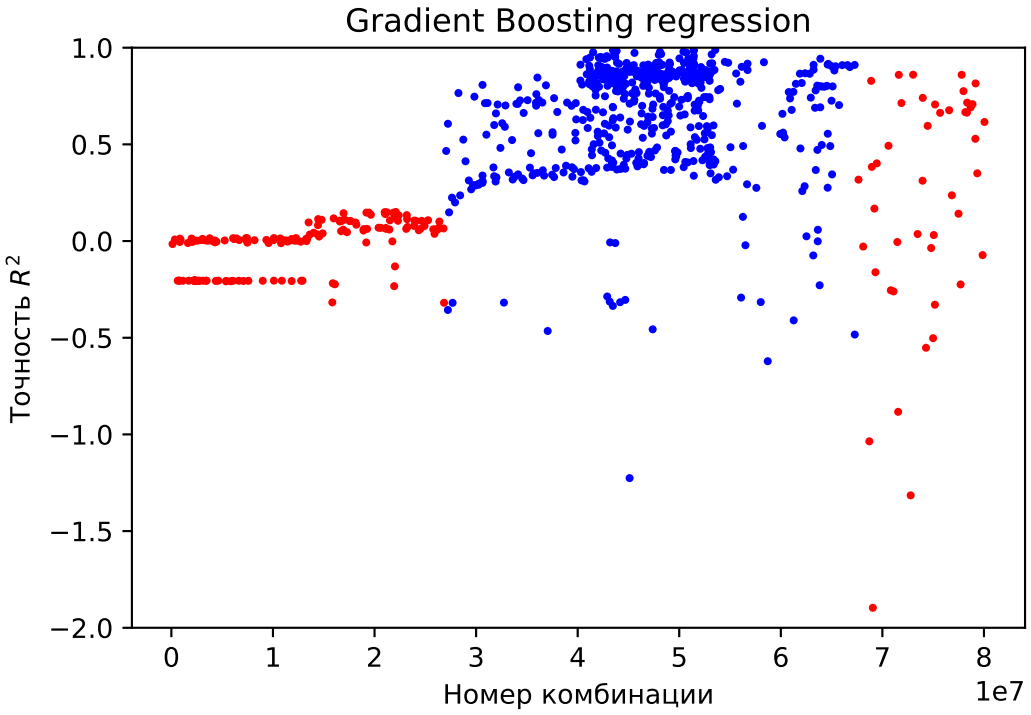

Рис. 1.6: Итоговый набор значений гиперпараметров для алгоритма для алгоритма Gradient Boosting regression.

1.1.4 XGBoost

Рассмотрим для данного алгоритма следующий начальный набор значений гиперпараметров:

- *max_depth:* [1, 20]
- *learning_rate:* {0.00001, 0.0001, 0.001, 0.01, 0.1, 0.5, 1}
- *subsample:* [0.05, 1.00], с шагом 0.05
- *min_child_weight:* [1, 20]

Соответственно получая 56 000 возможных комбинаций значений гиперпараметров.

После того как мы избавимся от значений гиперпараметров, которые встречаются реже всего или имеют низкий показатель точности, получим следующие возможные значения:

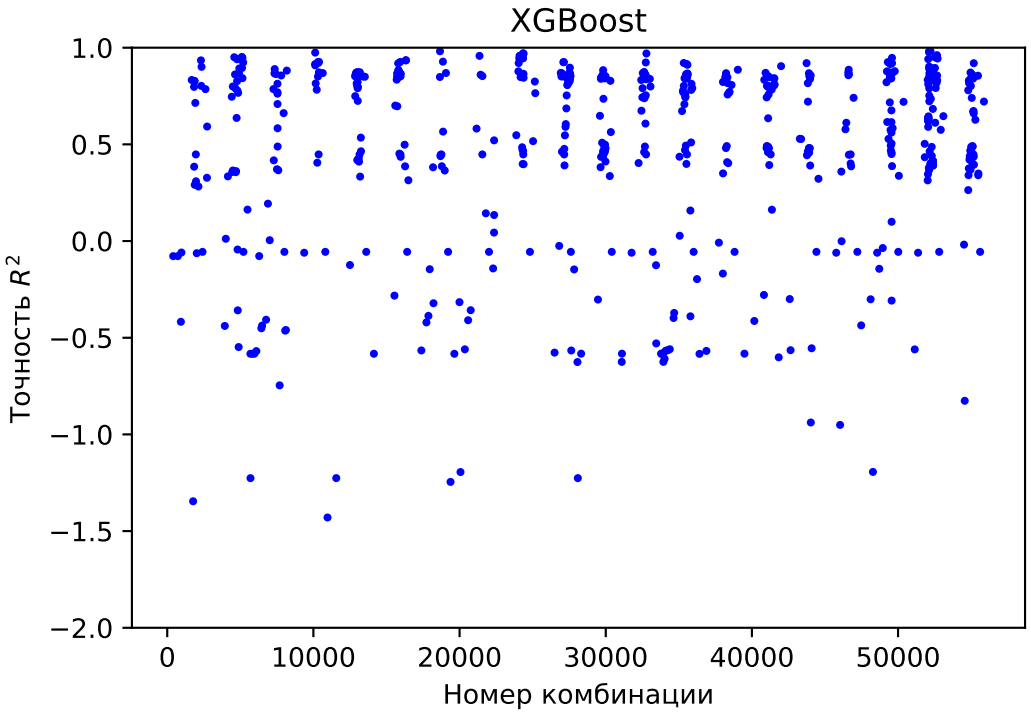

Рис. 1.7: Начальный набор значений гиперпараметров для алгоритма XGBoost regression.

- *max_depth:* [1, 20]
- *learning_rate:* {0.01, 0.1, 0.5, 1}
- *subsample:* [0.2, 1.00], с шагом 0.05
- *min_child_weight:* [1, 20]

Теперь данный алгоритм имеет 27 200 возможных комбинации значений гиперпараметров. 74% точек осталось.

Следующие алгоритмы имеют относительно маленькое число возможных значений гиперпараметров, поэтому для них не будет выполняться поиск наиболее оптимальных возможных значений гиперепараметров.

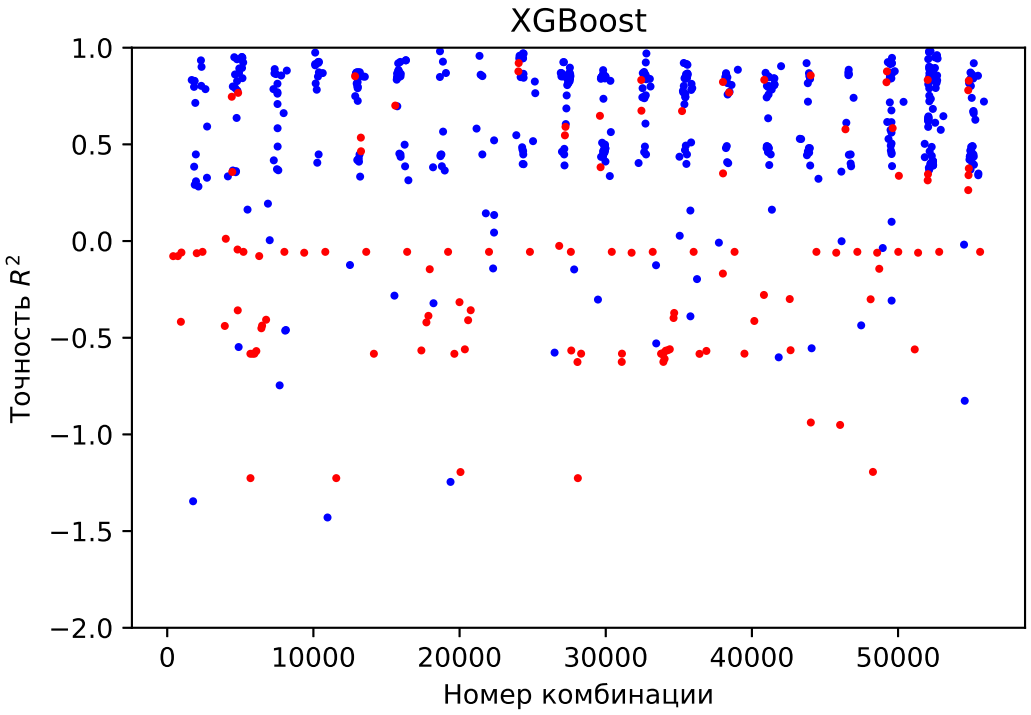

Рис. 1.8: Итоговый набор значений гиперпараметров для алгоритма для алгоритма XGBoost regression.

1.1.5 Метод k-ближайших соседей (k-nearest neighbors)

Набор возможных значений для данного алгоритма является следующим:

- *n_neighbors:* [1, 100]
- *weights:* {uniform, distance}
- *p:* {1, 2}

1.1.6 Метод Лассо с использованием метода наименьших углов (LassoLarsCV)

Набор возможных значений для данного алгоритма является следующим:

- *normalize:* {True, False}

1.1.7 Эластичная сеть (ElasticNet)

Набор возможных значений для данного алгоритма является следующим:

- *l1_ratio:* [0, 1], с шагом 0.5
- *tol:* {1e-5, 1e-4, 1e-3, 1e-2, 1e-1}

1.1.8 AdaBoost

Набор возможных значений для данного алгоритма является следующим:

- *learning_rate:* {1e-3, 1e-2, 1e-1, 0.5, 1}
- *loss:* {linear, square, exponential}

1.1.9 Метод опорных векторов для регрессии (Support Vector Regression)

Набор возможных значений для данного алгоритма является следующим:

- *loss:* {epsilon_insensitive, squared_epsilon_insensitive}
- *tol:* {1e-5, 1e-4, 1e-3, 1e-2, 1e-1}
- *C:* {1e-4, 1e-3, 1e-2, 1e-1, 0.5, 1, 5, 10, 15, 20, 25, 50, 70}
- *epsilon:* {1e-4, 1e-3, 1e-2, 1e-1, 1}

1.1.10 Стохастический градиентный спуск (Stochastic Gradient Descent)

Набор возможных значений для данного алгоритма является следующим:

- *loss:* {squared_loss, huber, epsilon_insensitive}
- *penalty:* {elasticnet}
- *alpha:* {0.0, 0.01, 0.001}
- *learning_rate:* {invscaling, constant}
- *fit_intercept:* {True, False}
- *l1_ratio:* {0.25, 0.0, 1.0, 0.75, 0.5}
- *eta0:* {0.1, 1.0, 0.01}
- *power_t:* {0.5, 0.0, 1.0, 0.1, 100, 10, 50}

1.1.11 Ридж-регрессия (Rigde-regression)

Набор возможных значений для данного алгоритма является следующим:

- *alphas:* {1e-4, 1e-3, 1e-2, 1e-1, 0.5, 1, 5, 10, 15, 20, 25, 50, 70}

1.2 Алгоритмы классификации

Для вычисления оптимальных возможных значений гиперпараметров алгоритмов классификации используется 20 датасетов, таблица с ними будет приведена ниже.

ТАБЛИЦА

Метрикой качества здесь будет являться F-мера: $F = \frac{2*precision*recall}{precision+recall}$

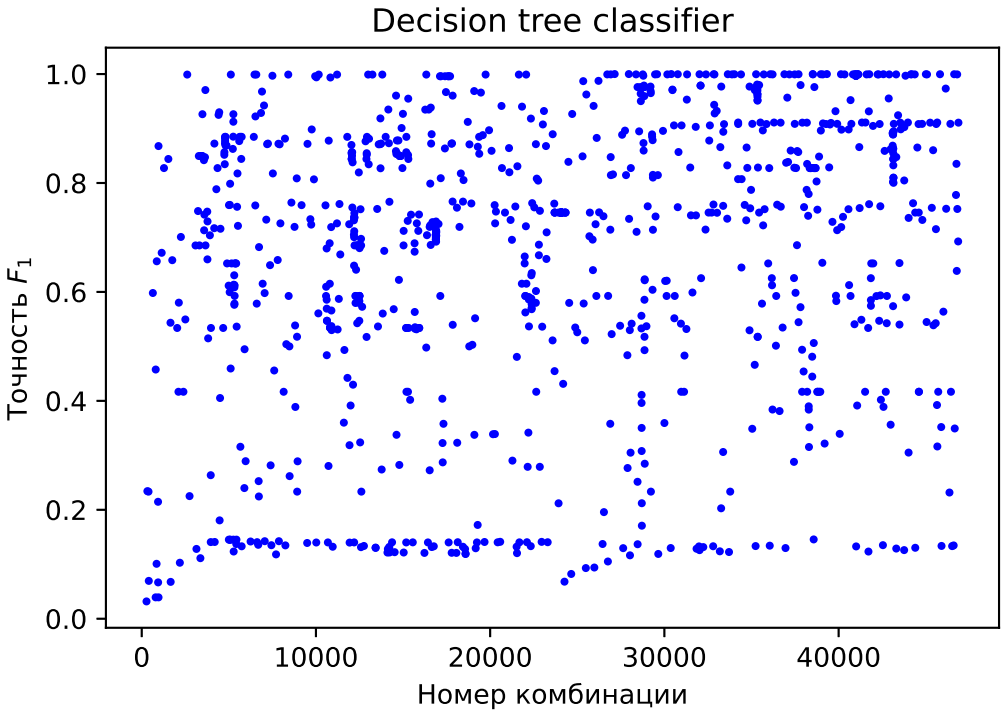

Рис. 1.9: Начальный набор значений гиперпараметров для алгоритма Decision tree classifier.

1.2.1 Дерево решений (Decision Tree)

Рассмотрим для данного алгоритма следующий начальный набор значений:

- *criterion:* {gini, entropy}
- *max_depth:* [1, 29]
- *min_samples_split:* [2, 29]
- *min_samples_leaf:* [1, 29]

Соответственно получая 47 096 возможных комбинаций значений гиперпараметров.

После того как мы избавимся от значений гиперпараметров, которые встречаются реже всего или имеют низкий показатель точности, получим следующие:

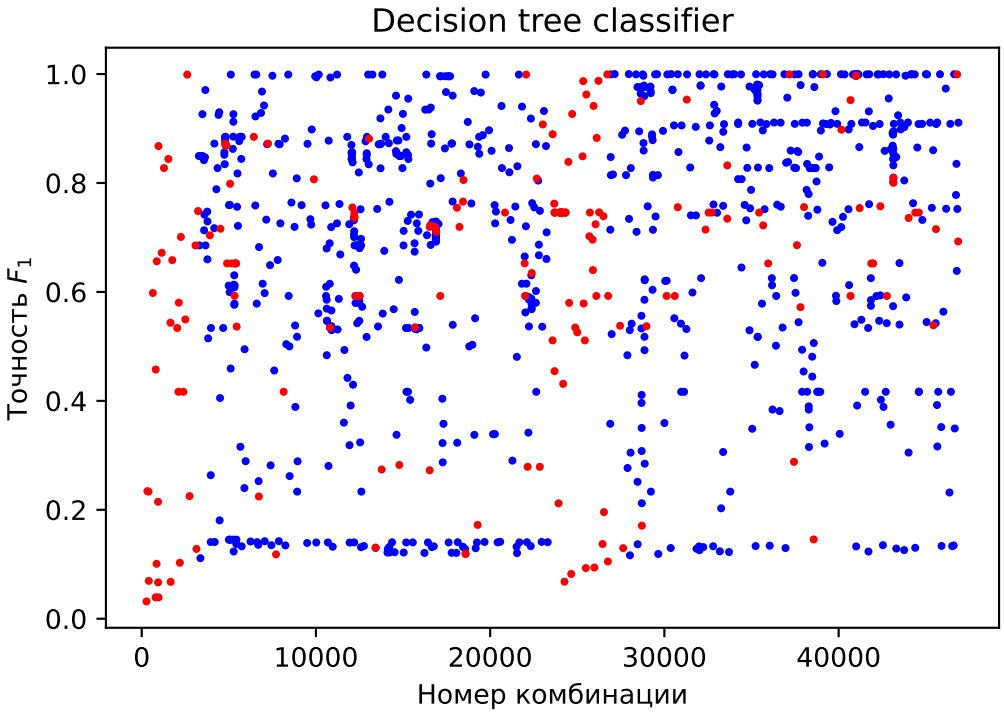

Рис. 1.10: Итоговый набор значений гиперпараметров для алгоритма Decision tree classifier.

- *criterion:* {gini, entropy}
- *max_depth:* [5, 29]
- *min_samples_split:* [1, 29]
- *min_samples_leaf:* [1, 25]

Теперь данный алгоритм имеет 35000 возможных комбинации значений гиперпараметров. 79% точек осталось.

1.2.2 Случайные и сверхслучайные деревья решения (Random Forest and Extra Trees)

Здесь будет рассмотрено сразу два алгоритма Random Forest и Extra Trees, так как это схожие алгоритмы и начальный набор значений гиперпараметров для них был один тот же. Для данных алгоритмов следующий начальный набор значений:

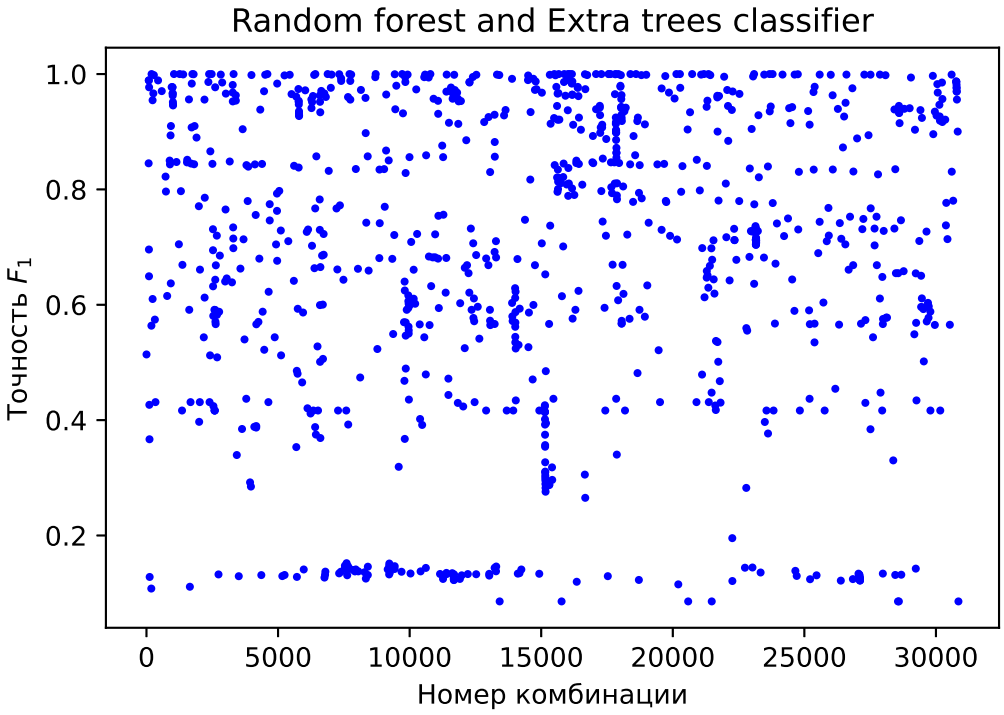

Рис. 1.11: Начальный набор значений гиперпараметров для алгоритмов Random forest classifier и Extra trees classifier.

- *criterion:* {gini, entropy}
- *max_features:* [0.1, 1.0], с шагом 0.05
- *min_samples_split:* [2, 29]
- *min_samples_leaf:* [1, 29]

Соответственно получая 30 856 возможных комбинаций значений гиперпараметров.

После того как мы избавимся от значений гиперпараметров, которые встречаются реже всего или имеют низкий показатель точности, получим следующие возможные значения:

- *criterion:* {gini, entropy}
- *max_features:* [0.1, 1.0], с шагом 0.05
- *min_samples_split:* [1, 29]

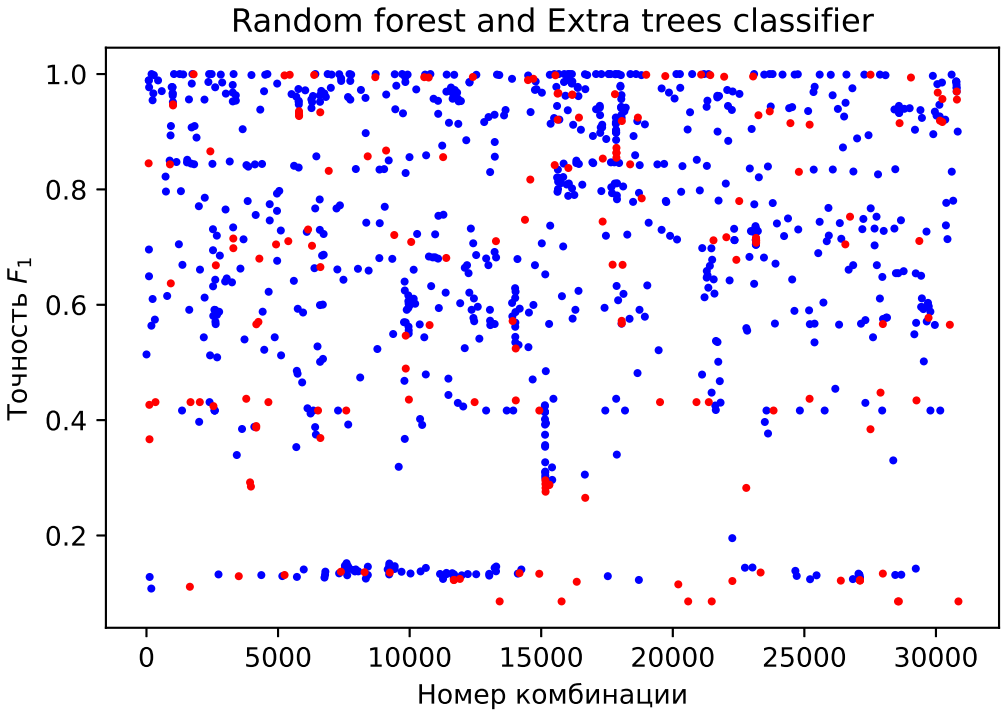

Рис. 1.12: Итоговый набор значений гиперпараметров для алгоритмов Random forest classifier и Extra trees classifier.

- *min_samples_leaf:* [1, 20]

Теперь данные алгоритмы имеют 21 280 возможных комбинации значений гиперпараметров. 82% точек осталось.

1.2.3 Градиентный бустинг (Gradient boosting)

Рассмотрим для данного алгоритма следующий начальный набор значений гиперпараметров:

- *learning_rate:* {1e-4, 1e-3, 1e-2, 1e-1, 0.5, 1}
- *max_depth:* [1, 19]
- *min_samples_split:* [2, 30]
- *min_samples_leaf:* [1, 30]
- *subsample:* [0.1, 1.00], с шагом 0.05

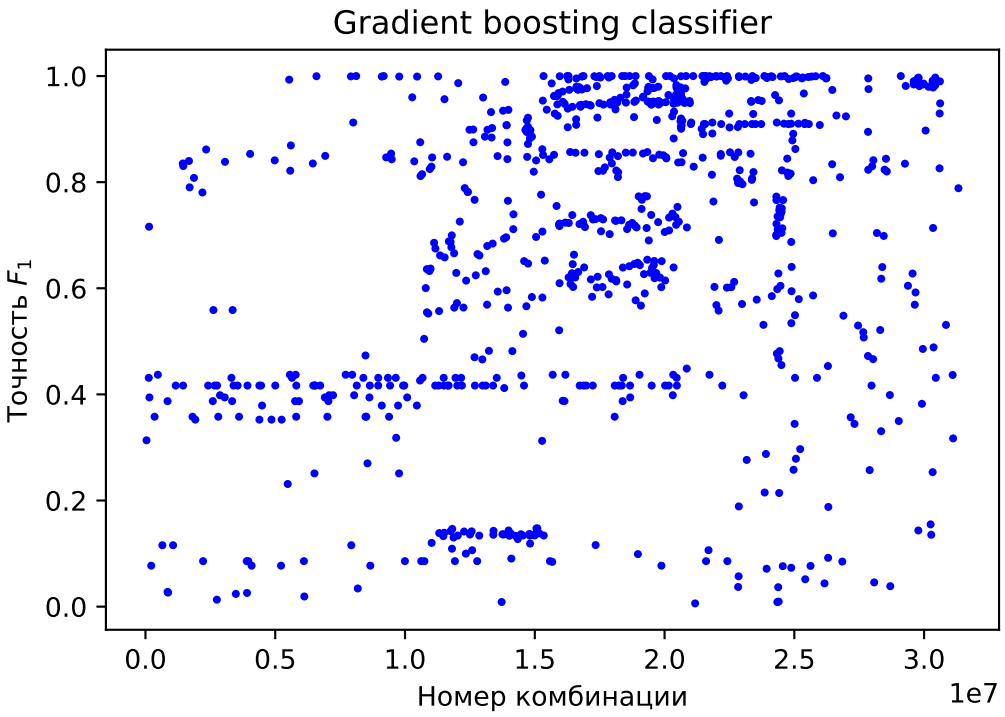

Рис. 1.13: Начальный набор значений гиперпараметров для алгоритма Gradient Boosting classifier.

- *max_features:* [0.05, 1.0], с шагом 0.05

Соответственно получая 35 175 840 возможных комбинаций значений гиперпараметров.

После того как мы избавимся от значений гиперпараметров, которые встречаются реже всего или имеют низкий показатель точности, получим следующие возможные значения:

- *learning_rate:* {1e-3, 1e-2, 1e-1, 0.5}
- *max_depth:* [1, 19]
- *min_samples_split:* [5, 30]
- *min_samples_leaf:* [1, 30]
- *subsample:* [0.1, 1.00], с шагом 0.05
- *max_features:* [0.05, 1.0], с шагом 0.05

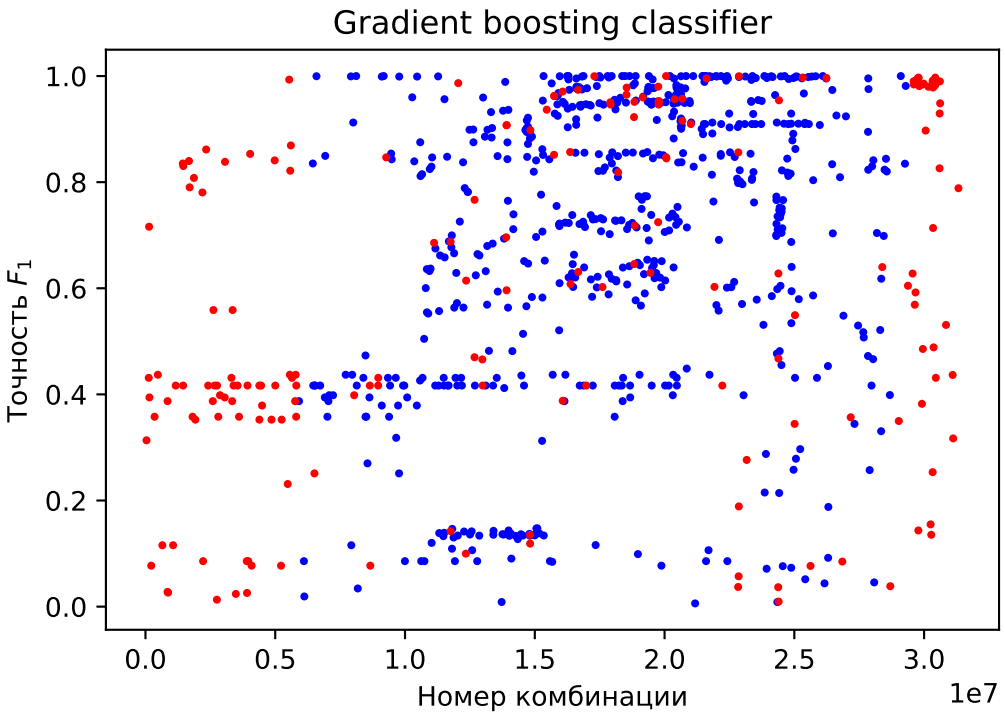

Рис. 1.14: Итоговый набор значений гиперпараметров для алгоритма для алгоритма Gradient Boosting classifier.

Теперь данный алгоритм имеет 20 938 000 возможных комбинации значений гиперпараметров. 77% точек осталось.

1.2.4 XGBoost

Рассмотрим для данного алгоритма следующий начальный набор значений гиперпараметров:

- *learning_rate:* {0.0001, 0.001, 0.01,, 0.1, 0.5, 1}
- *max_depth:* [1, 20]
- *subsample:* [0.05, 1.00], с шагом 0.05
- *min_child_weight:* [1, 29]

Соответственно получая 66 120 возможных комбинаций значений гиперпараметров.

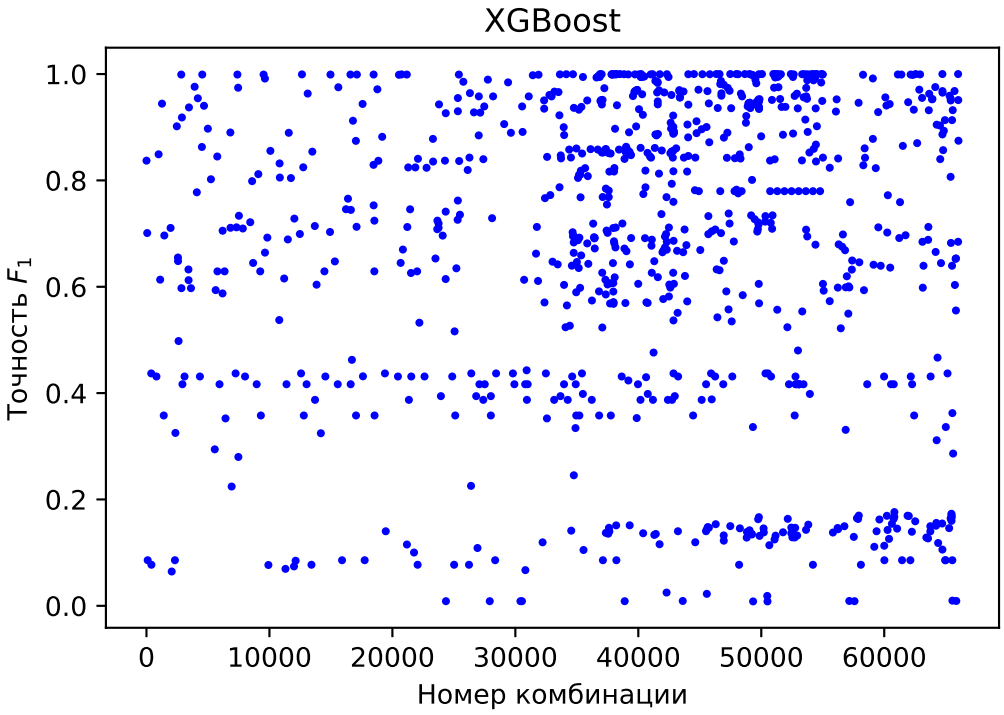

Рис. 1.15: Начальный набор значений гиперпараметров для алгоритма XGBoost classifier.

После того как мы избавимся от значений гиперпараметров, которые встречаются реже всего или имеют низкий показатель точности, получим следующие возможные значения:

- *learning_rate:* {0.01, 0.1, 0.5, 1}
- *max_depth:* [1, 20]
- *subsample:* [0.05, 1.00], с шагом 0.05
- *min_child_weight:* [1, 20]

Теперь данный алгоритм имеет 30 400 возможных комбинации значений гиперпараметров. 70% точек осталось.

Следующие алгоритмы имеют относительно маленькое число возможных значений гиперпараметров, поэтому для них не будет выполняться поиск наиболее оптимальных возможных значений гиперепараметров.

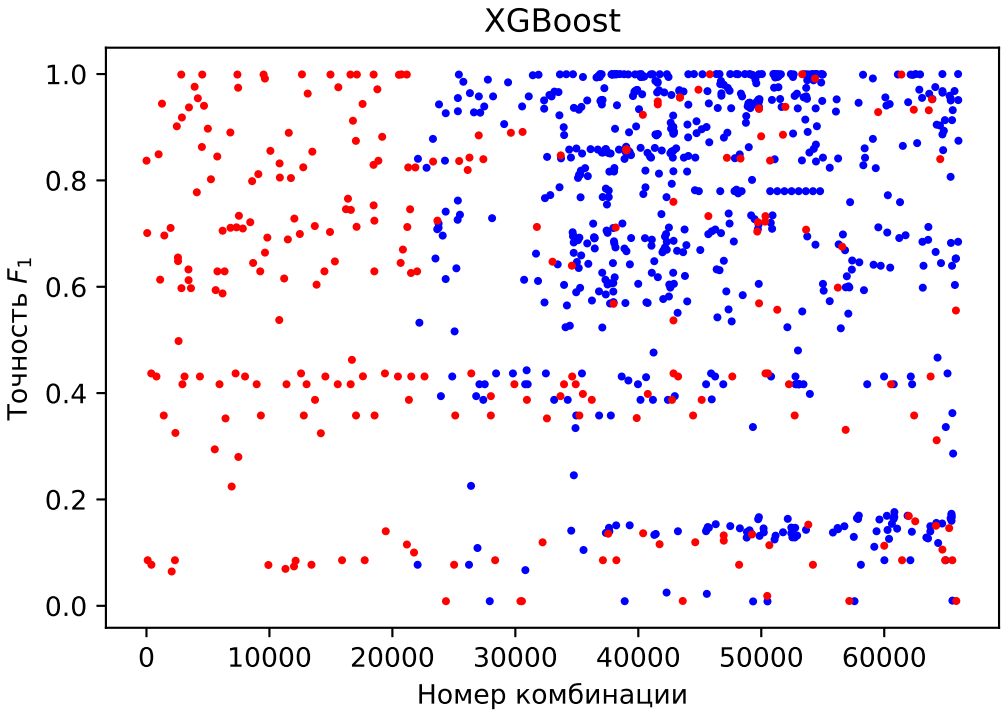

Рис. 1.16: Итоговый набор значений гиперпараметров для алгоритма для алгоритма XGBoost classifier.

1.2.5 Метод k-ближайших соседей (k-nearest neighbors)

Набор возможных значений для данного алгоритма является следующим:

- *n_neighbors:* [1, 100]
- *weights:* {uniform, distance}
- *p:* {1, 2}

1.2.6 Логистическая регрессия (Logistic Regression)

Набор возможных значений для данного алгоритма является следующим:

- *penalty:* {elasticnet}

- *l1_ratio:* {0.25, 0.0, 1.0, 0.75, 0.5}
- *C:* {1e-4, 1e-3, 1e-2, 1e-1, 0.5, 1, 5, 10, 15, 20, 25, 50, 70}

1.2.7 AdaBoost

Набор возможных значений для данного алгоритма является следующим:

- *learning_rate:* {1e-3, 1e-2, 1e-1, 0.5, 1}