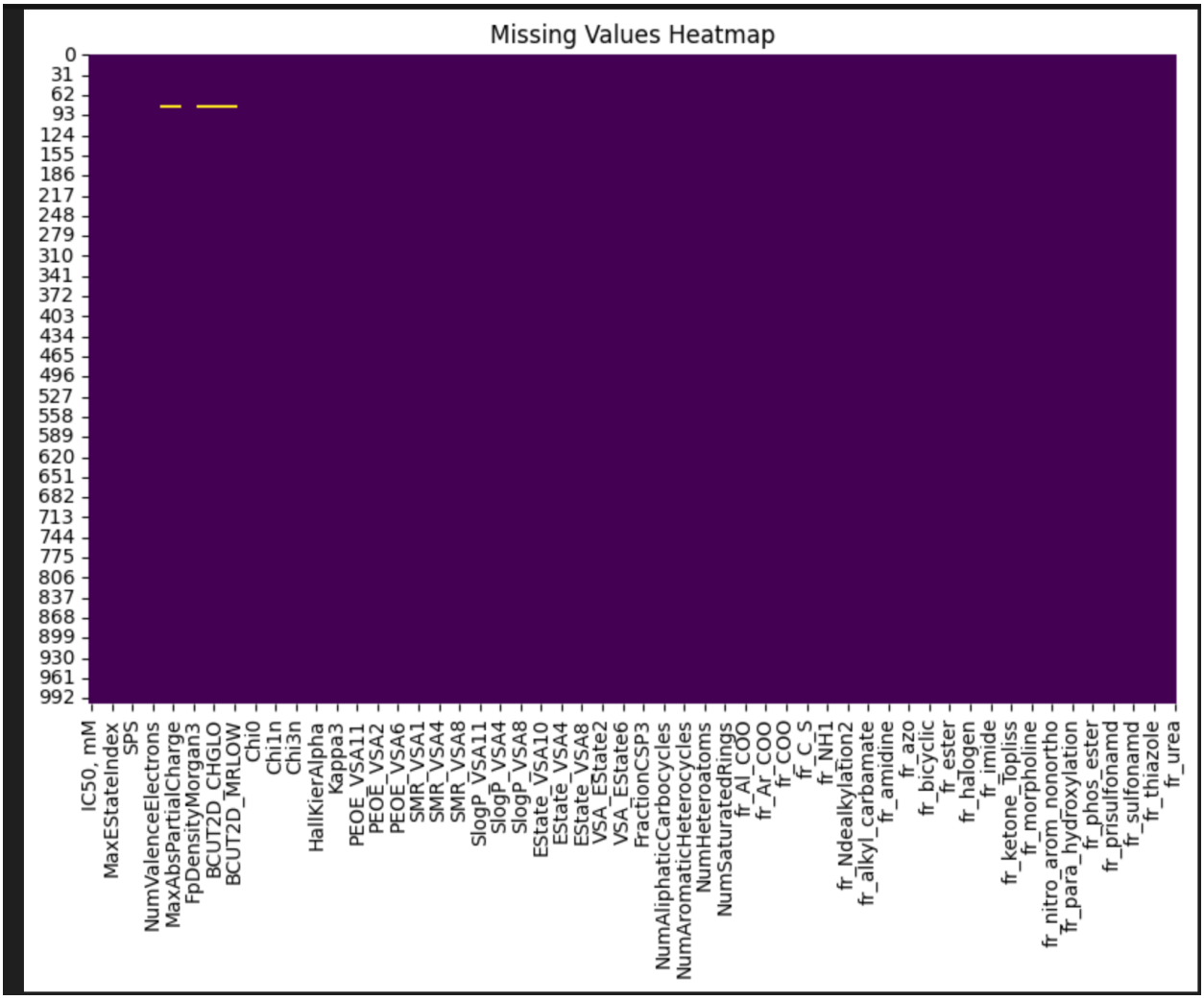
Арсеничев В О

Отчет по курсовой работе.

1) EDA

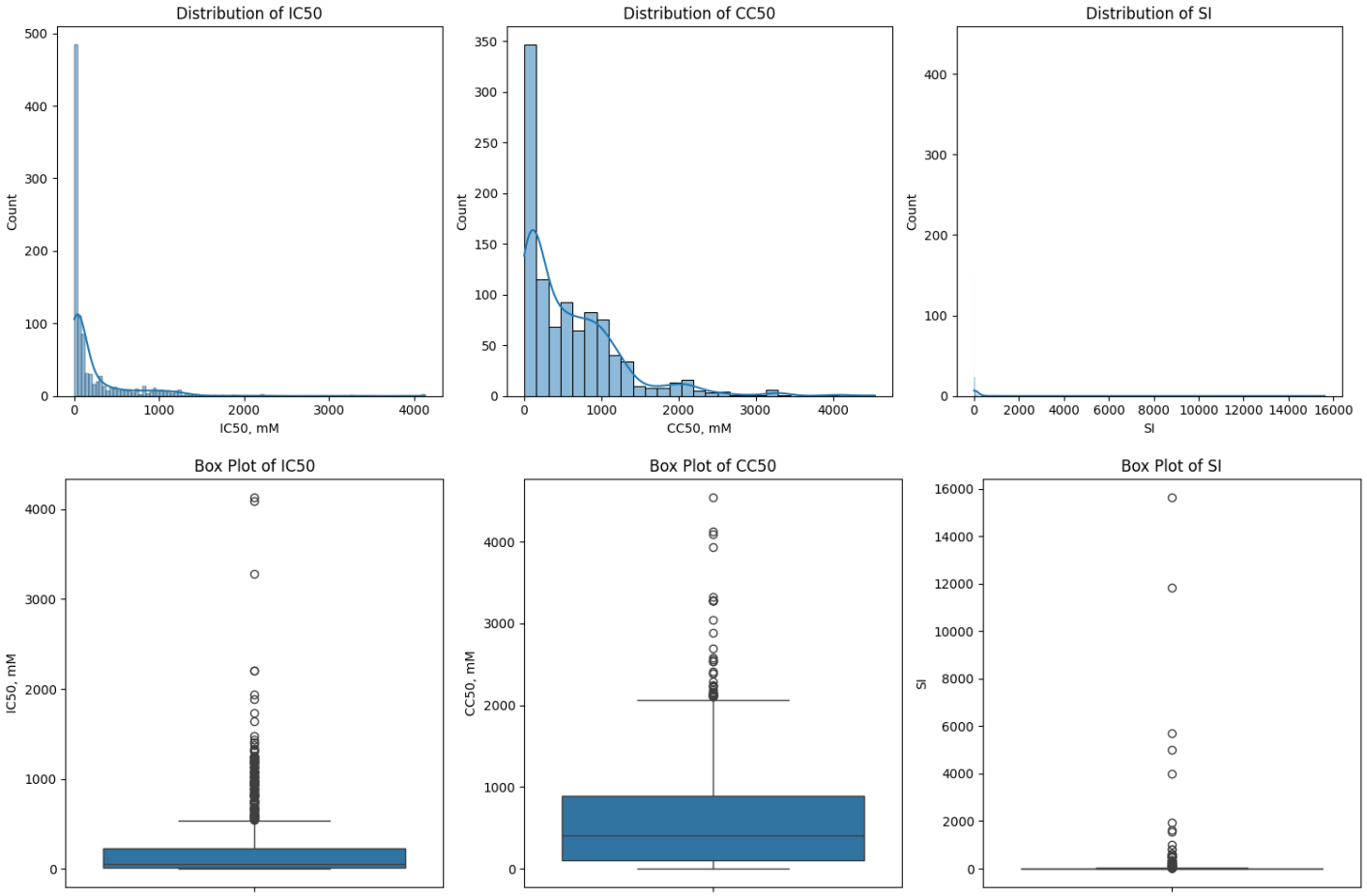
Наши данные насчитывают 1001 семплов из 210 признаков и 3 целевых переменных

В первую очередь проверим наши данные на пропуски



Видно, что в наших данных наблюдаются пропуски. Так как они в небольшом кол-ве, то можно их заполнить каким-то средним значением, например медианой.

Теперь проведем анализ целевых переменных IC50, CC50, SI

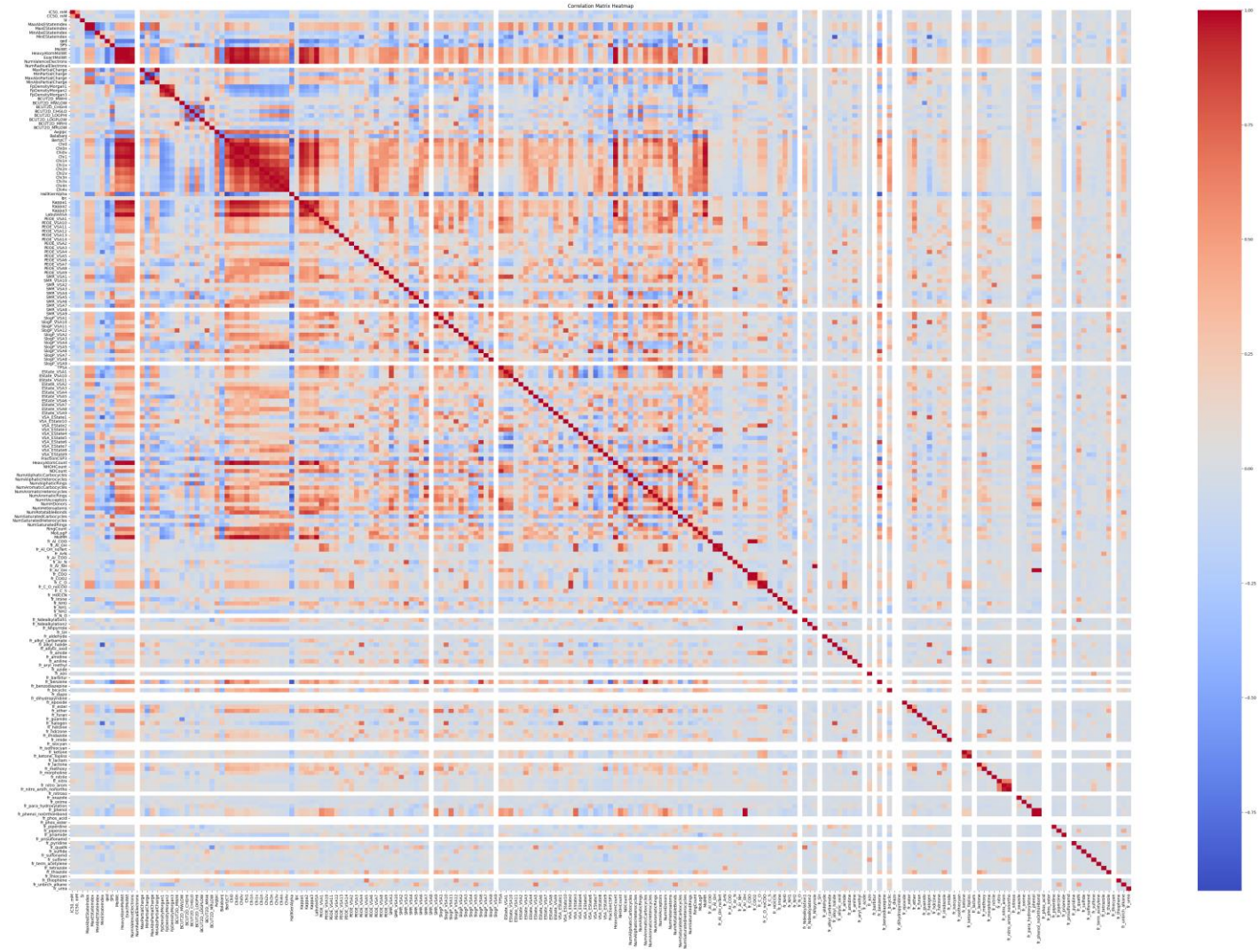


Видим, что целевые данные имеют выбросы и в целом имеют отличное от нормального распределения.

В наших данных 210 признаков, немалое кол-во. Возможно, некоторые зависимы между собой

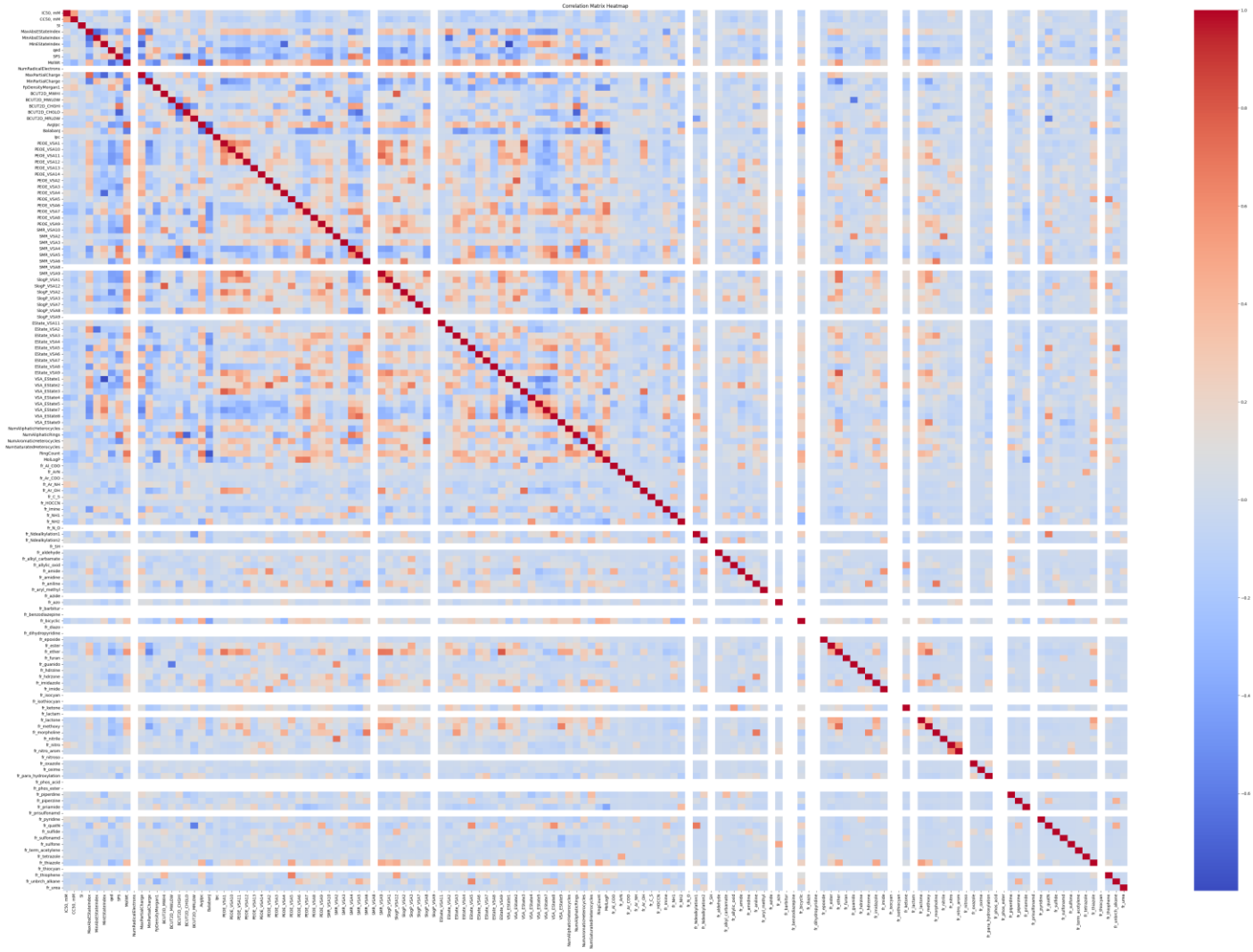
Проанализируем наши признаки и отберем лучшие для наших задач.

Построим тепловую карту корреляций наших признаков:



Мы можем видеть, что часть признаков сильно коррелируют между собой. Удали эти признаки. Обозначим порог для сильной корреляции в **0.8**

После удаления признаков получаем такую карту корреляции:



Наши признаки сократились до 139, но зато остались самые информативные.

Перейдем к решению задач.

2) Решение задач

2.0) Предобработка

В предобработке будем использовать масштабирование данных через MinMaxScale и будем делить их на тестовую и тренировочную выборку в отношении 3:7, получаем формы ((700, 139), (301, 139)) для тренировочной и тестовой соответственно.

2.1) **Регрессия для IC50**

Для регрессии проверим качество на трех алгоритмах: Линейная регрессия, Случайный лес, Градиентный бустинг.

Будем перебирать по сеткам на кросс-валидации.
Сетки для линейной регрессии нет
Сетка для случайного леса

```
'n_estimators': [50, 100, 200],
'max_depth': [None, 10, 20],
'min_samples_split': [2, 5, 10],
'min_samples_leaf': [1, 2, 4]
```

Сетка для градиентного бустинга

```
'n_estimators': [50, 100, 200],
'learning_rate': [0.01, 0.1, 0.2],
'max_depth': [3, 5, 7]
```

По результатам получаем такие результаты выбранных лучших моделей

```
Best parameters for rf_ic50: {'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 50}
Best score for rf_ic50: -1.812397544170821
Best parameters for lr_ic50: {}
Best score for lr_ic50: -4.028063257813157
Best parameters for gb_ic50: {'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 100}
Best score for gb_ic50: -1.7892461315859411
```

Теперь проверим наши модели на тестовых данных на метриках r_2, MSE, RMSE

```
Model: lr_ic50
R-squared: -9214484699140618977728.0000
MSE: 2726897721737168289792.0000
RMSE: 52219706258.6259

Model: rf_ic50
R-squared: -0.0457
MSE: 3.0946
RMSE: 1.7591

Model: gb_ic50
R-squared: -0.3283
MSE: 3.9310
RMSE: 1.9827
```

По результатам для регрессии IC50 лучше подойдет модель **случайны лес.**

2.2) **Регрессия для CC50**

Для этой задачи регрессии будем использовать такие же модели и такие же сетки перебора, как и в задаче с переменной IC50.

По результатам получаем такие результаты выбранных лучших моделей

```
Best parameters for rf_cc50: {'max_depth': 20, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 100}
Best score for rf_cc50: -1.4472490685190373
Best parameters for lr_cc50: {}
Best score for lr_cc50: -4.148788449950536
Best parameters for gb_cc50: {'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 100}
Best score for gb_cc50: -1.4841260722369887
```

Теперь проверим наши модели на тестовых данных на метриках r_2, MSE, RMSE

```
Model: lr_cc50
R-squared: -14589412886887407616.0000
MSE: 32381882702777688064.0000
RMSE: 5690508123.4260

Model: rf_cc50
R-squared: -0.9856
MSE: 4.4071
RMSE: 2.0993

Model: gb_cc50
R-squared: -1.9747
MSE: 6.6026
RMSE: 2.5695
```

По результатам для регрессии CC50 лучше подойдет модель **случайны лес.**

2.3) **Регрессия для SI**

В условиях задачи сказано, что переменная SI рассчитана на основе IC50 и CC50. Так что можно использовать только эти две переменные для нашей задачи с переменной SI. В дальнейшем алгоритм такой же, как в предыдущих регрессиях (модели, перебор).

По результатам получаем такие результаты выбранных лучших моделей

```
Best parameters for rf_si: {'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 100}
Best score for rf_si: -0.02108409659768976
Best parameters for lr_si: {}
Best score for lr_si: -0.09579339015431731
Best parameters for gb_si: {'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 200}
Best score for gb_si: -0.020920787923755468
```

Теперь проверим наши модели на тестовых данных на метриках r_2, MSE, RMSE

```
Model: lr_si
R-squared: -62.1445
MSE: 76.6747
RMSE: 8.7564

Model: rf_si
R-squared: -0.1461
MSE: 1.3917
RMSE: 1.1797

Model: gb_si
R-squared: -0.1410
MSE: 1.3854
RMSE: 1.1770
```

По результатам для регрессии SI лучше подойдет модель **Градиентный бустинг.**

2.4) **Классификация: превышает ли значение IC50 медианное значение выборки**

В этой задаче нам нужно реализовать целевые переменные в виде двух классов, большей медианы и меньшей медианы.

Для классификации проверим качество на четырех алгоритмах: Логистическая регрессия, Случайны лес, Градиентный бустинг, SVC.

Будем перебирать по сеткам на кросс-валидации.
Сетки для логистической регрессии нет
Сетка для случайного леса

```
'n_estimators': [50, 100, 200],
'max_depth': [None, 10, 20],
'min_samples_split': [2, 5, 10],
'min_samples_leaf': [1, 2, 4]
```

Сетка для градиентного бустинга

```
'n_estimators': [50, 100, 200],
'learning_rate': [0.01, 0.1, 0.2],
'max_depth': [3, 5, 7]
```

Сетка для SVC

```
['svc_class_ic50'] = {'C': [0.1, 1, 10, 100],
 'gamma': [1, 0.1, 0.01, 0.001],
 'kernel': ['rbf']}
```

По результатам получаем такие результаты выбранных лучших моделей

```
Best parameters for lr_class_ic50: {}
Best score for lr_class_ic50: 0.6885714285714286
Best parameters for rf_class_ic50: {'max_depth': 10, 'min_samples_leaf': 2, 'min_samples_split': 5, 'n_estimators': 200}
Best score for rf_class_ic50: 0.7200000000000001
Best parameters for gb_class_ic50: {'learning_rate': 0.01, 'max_depth': 5, 'n_estimators': 100}
Best score for gb_class_ic50: 0.7014285714285714
Best parameters for svc_class_ic50: {'C': 10, 'gamma': 0.1, 'kernel': 'rbf'}
Best score for svc_class_ic50: 0.7185714285714285
```

Теперь проверим наши модели на тестовых данных на метриках Accuracy, Precision, Recall, F1-score, ROC AUC

```
Model: lr_class_ic50
Accuracy: 0.6844
Precision: 0.6647
Recall: 0.7400
F1-score: 0.7003
ROC AUC: 0.7394039735099337

Model: rf_class_ic50
Accuracy: 0.7243
Precision: 0.6851
Recall: 0.8267
F1-score: 0.7492
ROC AUC: 0.7989183222958057

Model: gb_class_ic50
Accuracy: 0.7076
Precision: 0.6722
Recall: 0.8067
F1-score: 0.7333
ROC AUC: 0.7775938189845475

Model: svc_class_ic50
Accuracy: 0.6844
Precision: 0.6627
Recall: 0.7467
F1-score: 0.7022
ROC AUC: N/A
```

По результатам для классификации лучше по метрике accuracy подойдет модель **случайны лес.**

2.5) **Классификация: превышает ли значение CC50 медианное значение выборки**

Для этой задачи классификации будем использовать такие же модели и такие же сетки перебора, как и в задаче с переменной IC50.

По результатам получаем такие результаты выбранных лучших моделей

```
Best parameters for lr_class_cc50: {}
Best score for lr_class_cc50: 0.7428571428571429
Best parameters for rf_class_cc50: {'max_depth': 10, 'min_samples_leaf': 1, 'min_samples_split': 10, 'n_estimators': 200}
Best score for rf_class_cc50: 0.7571428571428571
Best parameters for gb_class_cc50: {'learning_rate': 0.01, 'max_depth': 3, 'n_estimators': 200}
Best score for gb_class_cc50: 0.7485714285714286
Best parameters for svc_class_cc50: {'C': 10, 'gamma': 0.1, 'kernel': 'rbf'}
Best score for svc_class_cc50: 0.7514285714285714
```

Теперь проверим наши модели на тестовых данных на метриках Accuracy, Precision, Recall, F1-score, ROC AUC

```
Model: lr_class_cc50
Accuracy: 0.7409
Precision: 0.7118
Recall: 0.8067
F1-score: 0.7562
ROC AUC: 0.8096467991169978

Model: rf_class_cc50
Accuracy: 0.7276
Precision: 0.6932
Recall: 0.8133
F1-score: 0.7485
ROC AUC: 0.8232008830022075

Model: gb_class_cc50
Accuracy: 0.7475
Precision: 0.7202
Recall: 0.8067
F1-score: 0.7610
ROC AUC: 0.806644591611479

Model: svc_class_cc50
Accuracy: 0.7243
Precision: 0.6936
Recall: 0.8000
F1-score: 0.7430
ROC AUC: N/A
```

По результатам для классификации лучше по метрике accuracy подойдет модель **градиентный бустинг.**

2.6) **Классификация: превышает ли значение SI медианное значение выборки**

Для этой задачи классификации будем использовать такие же модели и такие же сетки перебора, как и в задаче с переменной IC50.

По результатам получаем такие результаты выбранных лучших моделей

```
Best parameters for lr_class_si_med: {}
Best score for lr_class_si_med: 0.9342857142857144
Best parameters for rf_class_si_med: {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 2, 'n_estimators': 50}
Best score for rf_class_si_med: 0.9685714285714286
Best parameters for gb_class_si_med: {'learning_rate': 0.1, 'max_depth': 5, 'n_estimators': 50}
Best score for gb_class_si_med: 0.9642857142857142
Best parameters for svc_class_si_med: {'C': 100, 'gamma': 1, 'kernel': 'rbf'}
Best score for svc_class_si_med: 0.9885714285714287
```

Теперь проверим наши модели на тестовых данных на метриках Accuracy, Precision, Recall, F1-score, ROC AUC

```
Model: lr_class_si_med
Accuracy: 0.9336
Precision: 0.9710
Recall: 0.8933
F1-score: 0.9306
ROC AUC: 0.9901545253863135

Model: rf_class_si_med
Accuracy: 0.9734
Precision: 0.9863
Recall: 0.9600
F1-score: 0.9730
ROC AUC: 0.9972406181015452

Model: gb_class_si_med
Accuracy: 0.9668
Precision: 0.9730
Recall: 0.9600
F1-score: 0.9664
ROC AUC: 0.9909492273730685

Model: svc_class_si_med
Accuracy: 0.9934
Precision: 1.0000
Recall: 0.9867
F1-score: 0.9933
ROC AUC: N/A
```

По результатам для классификации лучше по метрике accuracy подойдет модель **SVC.**

2.7) **Классификация: превышает ли значение SI значение 8 (значение 3)**

В этой задаче, к сожалению, нет данных, где SI больше 8, поэтому методом подбора возьмем значение больше 3, их 230. В остальном задача не поменялась.

По результатам получаем такие результаты выбранных лучших моделей.

```
Best parameters for lr_class_si8: {}
Best score for lr_class_si8: 0.9142857142857143
Best parameters for rf_class_si8: {'max_depth': None, 'min_samples_leaf': 1, 'min_samples_split': 5, 'n_estimators': 50}
Best score for rf_class_si8: 0.9757142857142856
Best parameters for gb_class_si8: {'learning_rate': 0.1, 'max_depth': 3, 'n_estimators': 50}
Best score for gb_class_si8: 0.9685714285714285
Best parameters for svc_class_si8: {'C': 10, 'gamma': 1, 'kernel': 'rbf'}
Best score for svc_class_si8: 0.9885714285714287
```

Теперь проверим наши модели на тестовых данных на метриках Accuracy, Precision, Recall, F1-score, ROC AUC.

```
Model: lr_class_si8
Accuracy: 0.9402
Precision: 1.0000
Recall: 0.7391
F1-score: 0.8500
ROC AUC: 0.9971264367816092

Model: rf_class_si8
Accuracy: 0.9934
Precision: 1.0000
Recall: 0.9710
F1-score: 0.9853
ROC AUC: 0.9999375312343828

Model: gb_class_si8
Accuracy: 0.9767
Precision: 1.0000
Recall: 0.8986
F1-score: 0.9466
ROC AUC: 0.9993128435782108

Model: svc_class_si8
Accuracy: 1.0000
Precision: 1.0000
Recall: 1.0000
F1-score: 1.0000
ROC AUC: N/A
```

По результатам для классификации лучше по метрике accuracy подойдет модель **SVC.**

3) Выводы

В рамках курсовой работы были проведены анализ данных и эксперименты по выбору лучшей модели для 7 задач. Результаты экспериментов представлены в сводной таблице:

Задача	Лучший алгоритм	Метрика MSE	Метрика Accuracy
Регрессия для IC50	Случайный лес	3.0946	N/A
Регрессия для CC50	Случайный лес	4.4071	N/A
Регрессия для SI	Градиентный бустинг	1.3854	N/A
Классификация: превышает ли значение IC50 медианное значение выборки	Случайны лес	N/A	0.7243
Классификация: превышает ли значение CC50 медианное значение выборки	Градиентный бустинг	N/A	0.7475
Классификация: превышает ли значение SI медианное значение выборки	SVC	N/A	0.9934
Классификация: превышает ли значение SI значение 3	SVC	N/A	1.0

Дальнейшее улучшение:

1) Можно провести дополнительный анализ для отдельных признаков на информативность
2) Сделать доп предобработку данных, преобразовать к нормальному распределению, обработать выбросы и тд
3) Добавить другие алгоритмы в экспериментах.