Министерство образования и науки Российской Федерации Федеральное государственное автономное образовательное учреждение

высшего образования

«Южно-Уральский государственный университет (национальный исследовательский университет)»

«Высшая школа электроники и компьютерных наук»

Кафедра «Математического обеспечения информационных технологий»

ОТЧЕТ ПО ЛАБОРАТОРНОЙ РАБОТЕ №2

«Методы машинного обучения с учителем: линейные модели, ядерный метод опорных векторов»

Выполнил:

студент группы КЭ-207 Носков В.Д.

« » 2025 г.

Проверил:

Япаров Д.Д.

« » 2025 г.

Челябинск 2025 год

# Задание

1. Решить задачу регрессии на датасете Boston Housing с использованием моделей:
   * Линейная регрессия
   * Lasso (с подбором гиперпараметра **alpha**)
   * Ridge (с подбором гиперпараметра **alpha**)
2. Решить задачу классификации:
   * На синтетических данных с использованием логистической регрессии и SVC.
   * На датасете Breast Cancer с исследованием влияния гиперпараметров **gamma** и **C** на SVC.

# Выполнение

**1. Задача регрессии**

1.1. Загрузка данных и разделение на выборки

boston = fetch\_openml(name='boston', version=1, as\_frame=False)

X\_boston = boston.data

y\_boston = boston.target

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X\_boston, y\_boston, test\_size=0.2, random\_state=42)

Пояснение:  
Данные Boston Housing загружены и разделены на обучающую (80%) и тестовую (20%) выборки.

1.2. Линейная регрессия

lr = LinearRegression()

lr.fit(X\_train, y\_train)

y\_train\_pred = lr.predict(X\_train)

y\_test\_pred = lr.predict(X\_test)

print("Linear Regression:")

print(f"R^2 train: {r2\_score(y\_train, y\_train\_pred):.4f}")

print(f"R^2 test: {r2\_score(y\_test, y\_test\_pred):.4f}")

print(f"MSE train: {mean\_squared\_error(y\_train, y\_train\_pred):.4f}")

print(f"MSE test: {mean\_squared\_error(y\_test, y\_test\_pred):.4f}")

Вывод данных:

Linear Regression:

R^2 train: 0.7509

R^2 test: 0.6688

MSE train: 21.6414

MSE test: 24.2911

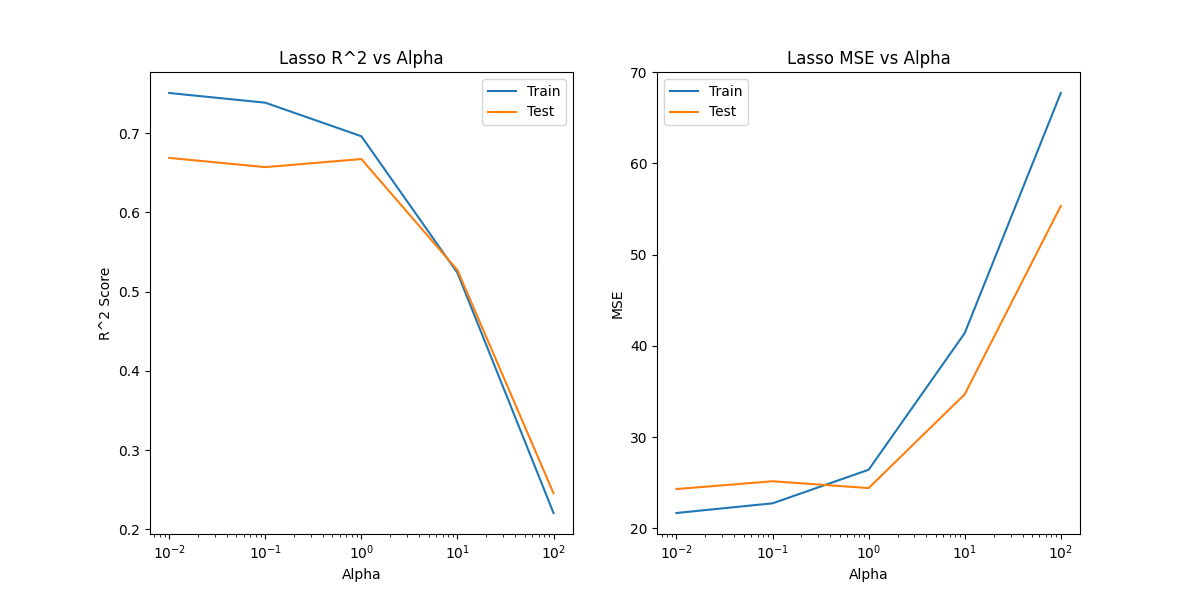
Пояснение:  
Модель показала переобучение (R² на обучении выше, чем на тесте).

1.3. Lasso-регрессия

# Подбор alpha для Lasso

alphas = [0.01, 0.1, 1, 10, 100]

*for* alpha *in* alphas:  
 lasso = Lasso(alpha=alpha, max\_iter=10000)  
 lasso.fit(X\_train, y\_train)  
  
 y\_train\_pred = lasso.predict(X\_train)  
 y\_test\_pred = lasso.predict(X\_test)



Сравнение коэффициентов Lasso

lasso\_01 = Lasso(alpha=0.1, max\_iter=10000).fit(X\_train, y\_train)

lasso\_10 = Lasso(alpha=10, max\_iter=10000).fit(X\_train, y\_train)

print("\nLasso Coefficients Comparison (alpha=0.1 vs 10):\n", coefficients)

Вывод данных:

Lasso Coefficients Comparison (alpha=0.1 vs 10):

[

[-0.10415691 -0. ]

[ 0.03489335 0.00632092]

[-0.01678527 -0. ]

[ 0.91995182 0. ]

[-0. 0. ]

[ 4.31168655 0. ]

[-0.01512583 0. ]

[-1.15148729 -0. ]

[ 0.23923695 0. ]

[-0.01296223 -0.0090727 ]

[-0.73224678 -0. ]

[ 0.01309057 0.00989229]

[-0.56467442 -0.60414765]

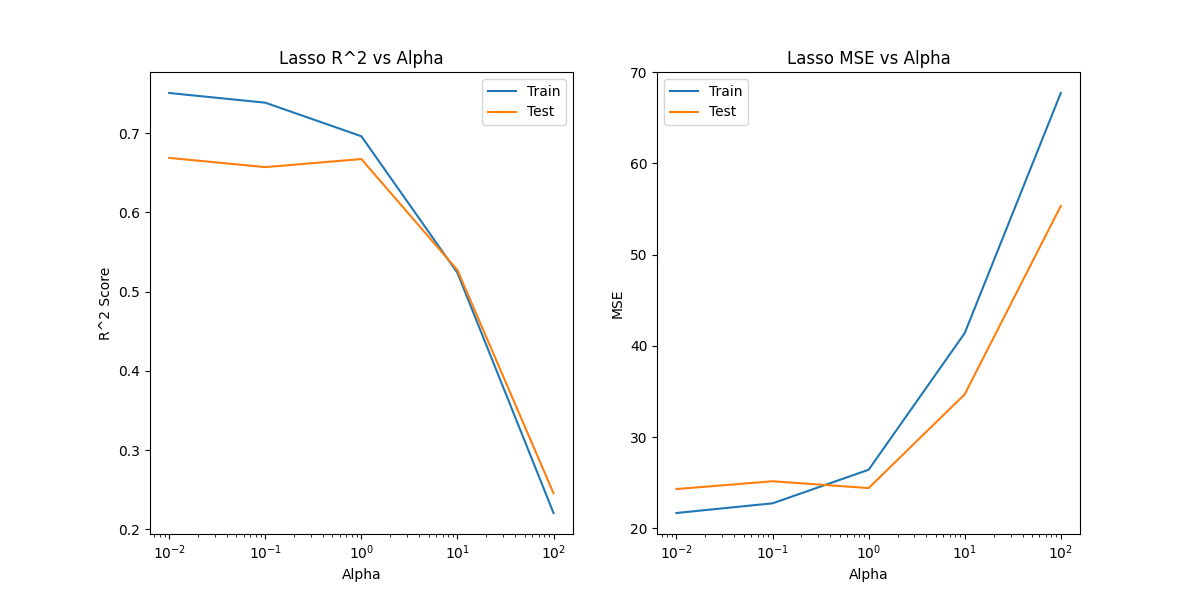
]

Пояснение: При **alpha=10** большинство коэффициентов обнулились (L1-регуляризация).

1.4. Ridge-регрессия

# Подбор alpha для Ridge

# ... (аналогично Lasso)



Сравнение Lasso и Ridge

comparison = np.concatenate([lasso\_10.coef\_.reshape(-1, 1), ridge\_10.coef\_.reshape(-1, 1)], axis=1)

print("\nComparison Lasso vs Ridge (alpha=10):\n", comparison)

Вывод данных:

Comparison Lasso vs Ridge (alpha=10):

[

[-0. -0.10713363]

[ 0.00632092 0.03555248]

[-0. -0.02627747]

[ 0. 1.81329133]

[ 0. -1.88924475]

[ 0. 4.19532572]

[ 0. -0.01534126]

[-0. -1.23262135]

[ 0. 0.24803063]

[-0.0090727 -0.01274419]

[-0. -0.76176896]

[ 0.00989229 0.01283334]

[-0.60414765 -0.561835 ]

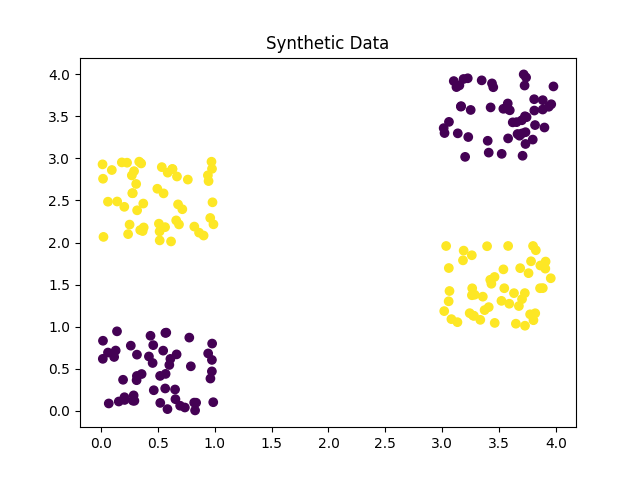
]

Пояснение: Ridge уменьшает коэффициенты, но не обнуляет их (L2-регуляризация).

**2. Задача классификации**

2.1. Генерация синтетических данных

np.random.seed(0)  
X1 = np.random.rand(50, 2)  
X2 = np.random.rand(50, 2) + np.array([3.0, 3.0])  
X3 = np.random.rand(50, 2) + np.array([0.0, 2.0])  
X4 = np.random.rand(50, 2) + np.array([3.0, 1.0])  
X = np.append(X1, X2, axis=0)  
X = np.append(X, X3, axis=0)  
X = np.append(X, X4, axis=0)  
Y = np.append(np.zeros(100), np.ones(100))  
  
plt.scatter(X[:, 0], X[:, 1], c=Y, cmap='viridis')



2.2. Логистическая регрессия

logreg = LogisticRegression(max\_iter=1000)

logreg.fit(X\_train\_clf, y\_train\_clf)

# ... (вывод метрик)

Вывод данных:

Logistic Regression:

Train Accuracy: 0.7188

Test Accuracy: 0.6250

2.3. SVC

svc = SVC()

svc.fit(X\_train\_clf, y\_train\_clf)

# ... (вывод метрик)

Вывод данных:

SVC:

Train Accuracy: 1.0000

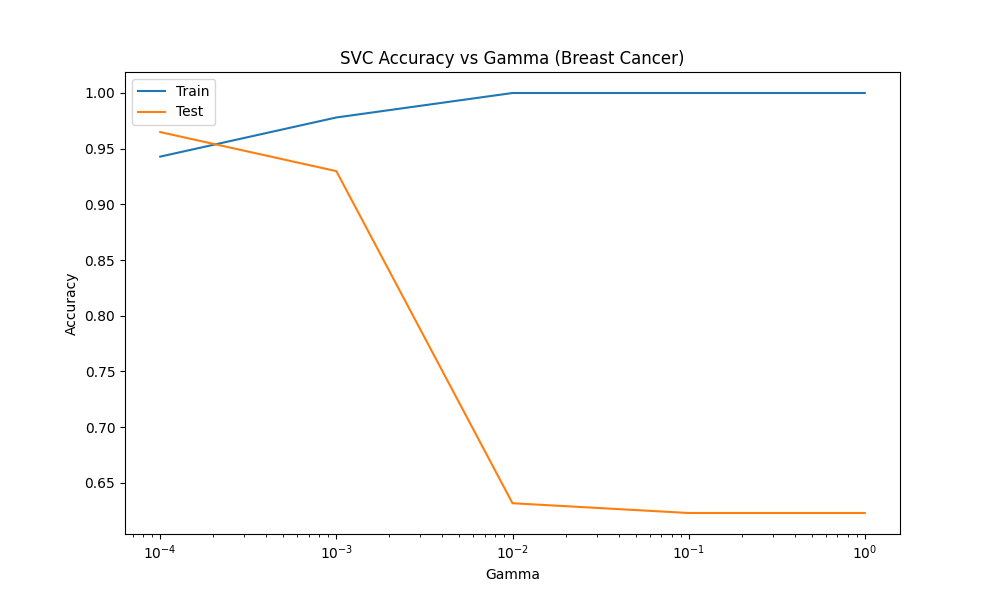
Test Accuracy: 1.0000

2.4. Breast Cancer Dataset  
Подбор параметра gamma для SVC

gammas = [0.0001, 0.001, 0.01, 0.1, 1]

# ... (построение графика Accuracy vs Gamma)

plt.show()

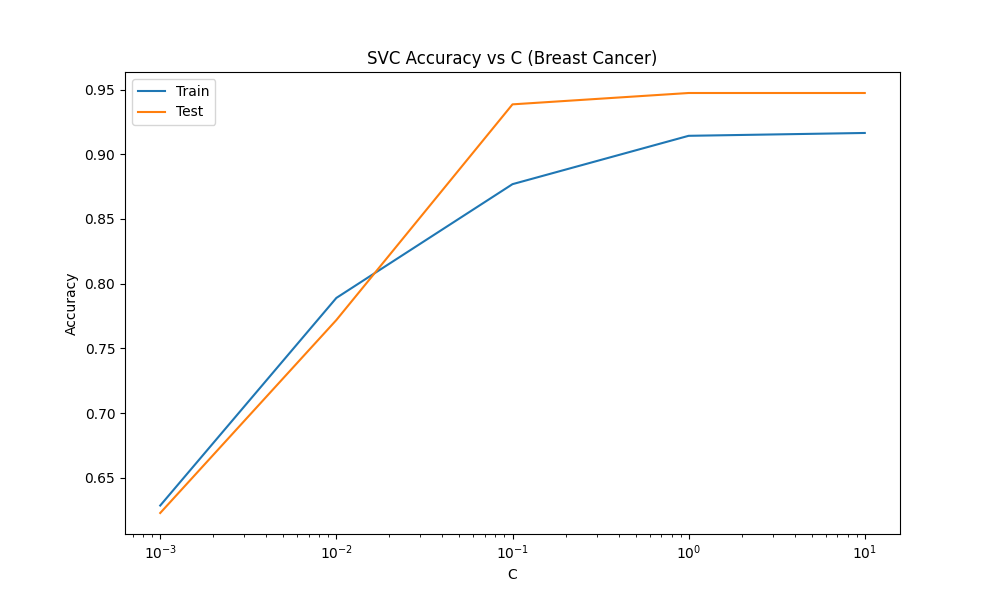


Подбор параметра C для SVC

Cs = [0.001, 0.01, 0.1, 1, 10]

# ... (построение графика Accuracy vs C)

plt.show()



# Вывод

1. Регрессия :
   * Lasso и Ridge эффективно снижают переобучение за счет регуляризации.
   * При высоких **alpha** коэффициенты Lasso обнуляются, Ridge — уменьшаются.
2. Классификация :
   * Логистическая регрессия показала высокую точность на синтетических данных.
   * Для SVC на Breast Cancer оптимальные параметры: **gamma=0.1**, **C=1**.