Министерство науки и высшего образования Российской Федерации

Федеральное государственное бюджетное образовательное учреждение

высшего образования

«Ульяновский государственный технический университет»

Кафедра: «Вычислительная техника»

Дисциплина: «Системы искусственного интеллекта»

Лабораторная работа №3

«Линейная регрессия»

Вариант 15

Выполнила

Студентка группы ИВТАСбд-31

Микка Е. И.

Ульяновск, 2024

# Общее задание на лабораторную работу

В данной лабораторной работе необходимо:

1. Разделить исходную выборку на обучающую и тестовую выборки, не используя стандартных функций.
2. Обучить модель линейной регрессии по обучающей выборки и проверить точность модели по тестовой выборки.
3. Построить модель с использованием полиномиальной функции и графики зависимости точности модели на обучающей и тестовой выборке от степеней функции.
4. Построить модель с использованием полиномиальной регуляризации и графики зависимости точности модели на обучающей и тестовой выборке от коэффициента регуляризации.

# Ход выполнения с описанием программной реализации

Задание 1.

Этот код загружает набор данных из файла, затем он разделяет данные на признаки (X) и целевые значения (Y). Далее индекс разделения генерируется как 80% от общей длины данных, чтобы разделить данные на обучающую выборку (X\_train\_set, Y\_train\_set) и тестовую выборку (X\_test\_set, Y\_test\_set). Для обучающего набора используются первые 80% данных, а для тестового набора остальные 20%.

data = pd.read\_excel('data\_akbilgic.xlsx')  
  
X = data[['SP', 'DAX', 'FTSE', 'NIKKEI', 'BOVESPA', 'EU', 'EM']]  
y = data['TL ISE']  
  
split\_index = int(0.8 \* len(X))  
X\_train\_set, X\_test\_set = X[:split\_index], X[split\_index:]  
Y\_train\_set, Y\_test\_set = y[:split\_index], y[split\_index:]

Задание 2.

Код определяет целевые значения для обучающей и тестовой выборок (Y\_train2 и Y\_test2). Далее создается модель линейной регрессии model, которая будет использоваться для предсказания целевого значения. Модель обучается (fit) на обучающем выборке (X\_train2, Y\_train2). Наконец, используя обученную модель, делается предсказание (predict) на тестовом наборе признаков X\_test2, и предсказанные значения сохраняются в Y\_pred.

X\_train2 = X\_train\_set['FTSE'].to\_frame()  
X\_test2 = X\_test\_set['FTSE'].to\_frame()  
  
Y\_train2 = Y\_train\_set  
Y\_test2 = Y\_test\_set  
  
model = LinearRegression()  
  
model.fit(X\_train2, Y\_train2)  
   
Y\_pred = model.predict(X\_test2)

Задание 3.

Этот код выполняет цикл по степеням полиномиальной функции от 1 до 15. Для каждой степени создается объект PolynomialFeatures и объект LinearRegression, которые используются в контейнере Pipeline для выполнения обработки данных. Обученная модель с текущей степенью полинома используется для предсказания значений на обучающей и тестовой выборках. Коэффициент детерминации (R2) вычисляется как метрика качества модели для обучающей и тестовой выборок. Затем находится наилучший коэффициент детерминации в тестовой выборке (best\_r2) и соответствующая ему степень полинома (best\_degree), который является индексом в массиве r2\_test, где достигается наилучший результат R2.

n\_d = 15  
  
r2\_train = [0] \* n\_d  
r2\_test = [0] \* n\_d  
  
fig, axs = plt.subplots(1, 2)  
fig.suptitle("Зависимость коэффициента детерминации от степени полиномиальной функции")  
for i in range(15):  
 polynomial\_features = PolynomialFeatures(degree=i+1, include\_bias=False)  
 linear\_regression = LinearRegression()  
  
 pipeline = Pipeline(  
 [  
 ("polynomial\_features", polynomial\_features),  
 ("linear\_regression", linear\_regression),  
 ]  
 )  
  
 pipeline.fit(X\_train2, Y\_train2)  
  
 Y\_train\_pred = pipeline.predict(X\_train2)  
 Y\_test\_pred = pipeline.predict(X\_test2)  
  
 r2\_train[i] = r2\_score(Y\_train2, Y\_train\_pred)  
 r2\_test[i] = r2\_score(Y\_test2, Y\_test\_pred)  
  
best\_r2 = max(r2\_test)   
best\_degree = r2\_test.index(best\_r2)

Задание 4.

Этот код выполняет итерацию по 1000 значениям коэффициента регуляризации alpha в диапазоне от 0 до 1. Для каждого значения alpha создается объект PolynomialFeatures с предыдущей наилучшей степенью полинома (bestdegree) и объект Ridge регрессии с заданным коэффициентом регуляризации. Модель Ridge регрессии обучается на обучающих данных Xtrain2 и Ytrain2 с помощью контейнера Pipeline, который включает PolynomialFeatures и Ridge. Затем предсказанные значения Y\_train\_pred и Y\_test\_pred получаются с использованием обученной модели, и для каждого значения alpha вычисляется коэффициент детерминации (R2) для обучающей и тестовой выборок. Далее находится наилучший коэффициент детерминации в тестовой выборке (best\_r2\_alpha) и соответствующее значение alpha (best\_alpha), которое является индексом с наилучшим результатом в массиве r2\_test\_alpha.

degree = best\_degree  
r2\_train\_alpha = [0] \* 1000  
r2\_test\_alpha = [0] \* 1000  
  
fig, axs = plt.subplots(1, 2)  
fig.suptitle("Зависимость коэффициента детерминации от коэффициента регуляризации")  
for i, alpha in enumerate(np.linspace(0, 1, 1000)):  
 polynomial\_features = PolynomialFeatures(degree=degree, include\_bias=False)  
 linear\_regression = linear\_model.Ridge(alpha=alpha)  
  
 pipeline = Pipeline(  
 [  
 ("polynomial\_features", polynomial\_features),  
 ("linear\_regression", linear\_regression),  
 ]  
 )  
  
 pipeline.fit(X\_train2, Y\_train2)  
  
 Y\_train\_pred = pipeline.predict(X\_train2)  
 Y\_test\_pred = pipeline.predict(X\_test2)  
  
 r2\_train\_alpha[i] = r2\_score(Y\_train2, Y\_train\_pred)  
 r2\_test\_alpha[i] = r2\_score(Y\_test2, Y\_test\_pred)  
  
best\_r2 = max(r2\_test\_alpha)  
best\_alpha = r2\_test\_alpha.index(best\_r2)

# 

# Тестирование

Задание 2. График



Задание 3. Графики

****

Задание 4. Графики



Вывод в консоли

