

МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ рОССИЙСКОЙ ФЕДЕРАЦИИ

Федеральное государственное автономное образовательное учреждение высшего образования

**«Дальневосточный федеральный университет»**

(ДВФУ)

|  |  |  |
| --- | --- | --- |
| **ИНСТИТУТ МАТЕМАТИКИ И КОМПЬЮТЕРНЫХ ТЕХНОЛОГИЙ (ШКОЛА)**  **Департамент математического и компьютерного моделирования** | | |
|  | |
|  |
|  |

**КУРСОВОЙ ПРОЕКТ**

по дисциплине «Машинное обучение.

Системы искусственного интеллекта»

на тему «Линейная регрессия»

направление подготовки 01.04.02 «Прикладная математика и информатика»

магистерская программа «Математические и компьютерные технологии»

|  |  |
| --- | --- |
| Оценка  Регистрационный номер \_\_\_\_\_\_\_  «\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023г. | Выполнил студент группы  № М9122-01.04.02мкт  \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_Миролюбова А.А.  (подпись)  «\_\_\_\_\_» \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ 2024г.  Руководитель \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_  (должность, ученое звание)  \_\_\_\_\_\_\_\_\_\_ \_\_\_\_\_\_\_\_\_\_\_\_\_  (подпись) (ФИО)  «\_\_\_\_\_\_»\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_2023г. |

г. Владивосток

2024

|  |
| --- |
| **Оглавление**  [Введение](file:///C:\Users\Алёна\Desktop\мага%202\кп\Реферат,%20Миролюбова%20Алёна,%20%20М9122-01.04.02мкт.docx#_Toc61081863) 3  [Основная часть](file:///C:\Users\Алёна\Desktop\мага%202\кп\Реферат,%20Миролюбова%20Алёна,%20%20М9122-01.04.02мкт.docx#_Toc61081864) 4  [Заключение](file:///C:\Users\Алёна\Desktop\мага%202\кп\Реферат,%20Миролюбова%20Алёна,%20%20М9122-01.04.02мкт.docx#_Toc61081865) 23  [Список использованных источников](file:///C:\Users\Алёна\Desktop\мага%202\кп\Реферат,%20Миролюбова%20Алёна,%20%20М9122-01.04.02мкт.docx#_Toc61081865) 24 |

## **Введение**

Машинное обучение является одним из самых актуальных и быстро развивающихся направлений в области искусственного интеллекта. Оно представляет собой набор методов и алгоритмов, позволяющих компьютерным системам обучаться на основе предоставленных данных и делать предсказания или принимать решения без явно заданной программы.

Одним из наиболее распространенных и широко применяемых методов машинного обучения является линейная регрессия. Она применяется для анализа и предсказания зависимости между двумя переменными, в которой одна переменная (независимая) является предиктором, а другая переменная (зависимая) является целевой.

Цель данного курсового проекта состоит в изучении принципов и применении линейной регрессии для решения задачи прогнозирования на основе предоставленных данных. В процессе работы будет рассмотрено основное понятие линейной регрессии, ее математические модели и алгоритмы, а также сформулированы необходимые метрики для оценки качества модели.

## **Основная часть**

В качестве данных для обучения линейной регрессии возьмём стоимость домов: <https://www.kaggle.com/c/house-prices-advanced-regression-techniques/data>.

Цель — спрогнозировать цену продажи каждого дома. Для каждого идентификатора в тестовом наборе необходимо спрогнозировать значение переменной SalePrice.

В этом наборе данных, состоящем из 79 объясняющих переменных, описывающих (почти) каждый аспект жилых домов в Эймсе, штат Айова, предстоит предсказать окончательную цену каждого дома. Целевой переменной будет `SalePrice`. То есть мы будем предсказывать стоимость жилья на основе его характеристик.

Характеристики:

* SalePrice — цена продажи недвижимости в долларах. Это целевая переменная, которую мы пытаемся предсказать.
* MSSubClass: класс здания.
* MSZoning: Общая классификация зонирования.
* LotFrontage: погонные футы улицы, соединенной с участком.
* LotArea: размер участка в квадратных футах.
* Street: Тип подъезда к дороге
* Alley: Тип доступа к переулку
* LotShape: Общий вид объекта недвижимости
* LandContour: Ровность участка
* Utilities: Типы доступных утилит.
* LotConfig: конфигурация лота
* LandSlope: Уклон участка
* Neighborhood: физические местоположения в пределах города Эймс.
* Condition1: Близость к главной дороге или железной дороге.
* Condition2: Близость к главной дороге или железной дороге (при наличии второй)
* BldgType: Тип жилья
* HouseStyle: Стиль жилища
* TotalQual: общее качество материала и отделки.
* TotalCond: общая оценка состояния.
* YearBuilt: дата первоначального строительства.
* YearRemodAdd: дата ремоделирования.
* RoofStyle: Тип крыши.
* RoofMatl: Материал крыши
* Exterior1st: Наружное покрытие дома
* Exterior2nd: Внешнее покрытие дома (если более одного материала)
* MasVnrType: тип каменной облицовки.
* MasVnrArea: площадь каменного шпона в квадратных футах.
* ExterQual: Качество материалов экстерьера
* ExterCond: Текущее состояние материала снаружи.
* Foundation: Тип фундамента
* BsmtQual: Высота подвала.
* BsmtCond: Общее состояние подвала.
* BsmtExposure: стены подвала на уровне сада или выходного этажа.
* BsmtFinType1: Качество готовой площади подвала.
* BsmtFinSF1: готовые квадратные футы типа 1.
* BsmtFinType2: качество второй готовой площади (если имеется).
* BsmtFinSF2: готовые квадратные футы типа 2.
* BsmtUnfSF: Недостроенные квадратные футы подвала.
* TotalBsmtSF: Общая площадь подвала в квадратных футах.
* Heating: Тип отопления
* HeatingQC: качество и состояние отопления.
* CentralAir: Центральное кондиционирование.
* Electrical: Электрическая система
* 1stFlrSF: квадратные футы первого этажа.
* 2ndFlrSF: квадратные футы второго этажа.
* LowQualFinSF: квадратные футы с отделкой низкого качества (все этажи)
* GrLivArea: жилая площадь над землей, квадратные футы.
* BsmtFullBath: Полностью оборудованные ванные комнаты в подвале.
* BsmtHalfBath: Ванные комнаты в подвале.
* FullBath: полностью оборудованные ванные комнаты над уровнем моря.
* HalfBath: полуванны выше уровня.
* Bedroom: Количество спален над цокольным этажом.
* Kitchen: Количество кухонь
* KitchenQual: Качество кухни
* TotRmsAbvGrd: общее количество комнат выше уровня (без ванных комнат).
* Functional: Рейтинг функциональности дома
* Fireplaces: Количество каминов
* FireplaceQu: Качество камина
* GarageType: Расположение гаража
* GarageYrBlt: Год постройки гаража.
* GarageFinish: Внутренняя отделка гаража.
* GarageCars: Размер гаража по вместимости автомобиля.
* GarageArea: Размер гаража в квадратных футах.
* GarageQual: гаражное качество.
* GarageCond: состояние гаража.
* PavedDrive: Асфальтированная подъездная дорога
* WoodDeckSF: Площадь деревянного настила в квадратных футах.
* OpenPorchSF: площадь открытой веранды в квадратных футах.
* EnclosedPorch: площадь крытой веранды в квадратных футах.
* 3SsnPorch: Трехсезонная веранда в квадратных футах
* ScreenPorch: площадь веранды в квадратных футах.
* PoolArea: площадь бассейна в квадратных футах.
* PoolQC: качество бассейна
* Fence: Качество забора
* MiscFeature: Разные функции, не вошедшие в другие категории.
* MiscVal: $Значение различных функций.
* MoSold: месяц продажи
* YrSold: Год продажи
* SaleType: Тип продажи
* SaleCondition: Состояние продажи

**Ход работы**

Загрузка данных:

df = pd.read\_csv('../data/house\_prices.csv.gz', index\_col='Id')

Изображение выглядит как текст, снимок экрана, черный, Шрифт

Автоматически созданное описаниеdf.head()

Анализ целевого признака:

Удостоверимся, что распределение целевого признака хорошее: без выбросов. Мы хотим видеть нормальное распределение или достаточно близкое к нему. Так как с ростом объема выборки, оценка максимального правдоподобия лучше описывает нормальное распределение.

Коэффициент эксцесса показывает степень остроты пика распределения случайной величины. Для нормального распределения он равен `0`. Если хвосты распределения «тяжелее», а пик более «приплюснутый», чем у нормального распределения, то коэффициент меньше 0. А если хвосты «легкие», то больше 0.

График распределения SalePrice, с предполагаемым нормальным распределением.

ns.distplot(df['SalePrice'], fit=stats.norm)

plt.xlabel('SalePrice')

plt.ylabel('Density')

Изображение выглядит как диаграмма, График, снимок экрана, линия

Автоматически созданное описание

Коэффициент эксцесса для SalePrice.

kurtosis = stats.kurtosis(df['SalePrice'])

kurtosis

kurtosis = 6.509812011089439

График показывающий реальные квантили выборки и теоретические. По нему можно видеть, на сколько много выбросов и на сколько выборка предсказуема.

stats.probplot(df['SalePrice'], plot=plt)

plt.title('Probability Plot of SalePrice')

По графику probplot можно судить о том, насколько выборка отклоняется от нормального распределения. Есть значительные отклонения от прямой, это может указывать на наличие выбросов или других аномалий в данных.

Изображение выглядит как текст, снимок экрана, линия, График

Автоматически созданное описание

Анализ зависимостей:

Графики, отражающие зависимости признаков:

Дата постройки дома и его стоимость

sns.jointplot( x=df["YearBuilt"], y=df['SalePrice'])

plt.xlabel("Year Built")

plt.ylabel("Sale Price")

plt.show()

Изображение выглядит как текст, снимок экрана, диаграмма, карта

Автоматически созданное описание

"Качество дома и материалов" и его стоимость.

sns.boxplot( x=df["OverallQual"], y=df["SalePrice"])

plt.xlabel("Overall Quality")

plt.ylabel("Sale Price")

plt.title("Dependence of Sale Price on Overall Quality")

plt.show()

Изображение выглядит как текст, диаграмма, снимок экрана, План

Автоматически созданное описание

Площадь дома и стоимость.

sns.lmplot(data=df, x="GrLivArea", y="SalePrice")

plt.xlabel("Living Area")

plt.ylabel("Sale Price")

plt.title("Dependence of Sale Price on Living Area")

plt.show()

Изображение выглядит как текст, снимок экрана, карта, диаграмма

Автоматически созданное описание

Судя по графикам, есть некоторые линейные зависимости. Но строить по графику на каждый признак из 80 штук долго и не наглядно.

График корреляции всех признаков:

encoded\_df = df.apply(lambda x: pd.factorize(x)[0])

corr = encoded\_df.corr()

mask = np.triu(np.ones\_like(corr, dtype=bool))

plt.figure(figsize=(15, 15))

sns.heatmap(corr, mask=mask, cmap='RdYlGn', square=True, linewidths=.5)

plt.title('Correlation Matrix')

Изображение выглядит как текст, снимок экрана, линия, диаграмма

Автоматически созданное описаниеplt.show()

Очевидно, здесь есть сильно коррелируемые объекты с целевой переменной.

Визуализация этих сильно коррелируемых объектов (больше 0.5 по абсолютному значению):

corr\_df = encoded\_df.corr().abs()

high\_corr\_var = np.where(corr\_df > 0.5)

high\_corr\_pairs = [(corr\_df.index[x], corr\_df.columns[y]) for x, y in zip(\*high\_corr\_var) if x != y and x < y]

high\_corr\_df = encoded\_df[list(set(x for pair in high\_corr\_pairs for x in pair))]

corr = high\_corr\_df.corr()

mask = np.triu(np.ones\_like(corr, dtype=bool))

plt.figure(figsize=(15,15))

sns.heatmap(high\_corr\_df.corr(), mask = mask, annot=True, cmap='RdYlGn', square=True, fmt='.2f',

            xticklabels=True, yticklabels=True)

plt.title('Heatmap of Highly Correlated Features')

Изображение выглядит как текст, снимок экрана, диаграмма, Красочность

Автоматически созданное описаниеplt.show()

high\_corr\_features = list(set(x for pair in high\_corr\_pairs for x in pair))

print("Highly Correlated Features:", high\_corr\_features)

Highly Correlated Features: ['GarageQual', 'PoolArea', 'PoolQC', 'BsmtFinType2', 'BsmtFinSF2', 'ExterQual', 'GarageCond', 'LandSlope', 'TotalBsmtSF', 'RoofStyle', 'MasVnrType', 'MasVnrArea', 'BldgType', 'MiscFeature', 'RoofMatl', 'MiscVal', 'GarageYrBlt', 'KitchenQual', '1stFlrSF', 'Exterior2nd', 'HouseStyle', 'MSSubClass', 'LandContour', 'Exterior1st', 'Fireplaces', 'GrLivArea', 'FireplaceQu', 'YearBuilt']

Пропущенные данные:

Подсчёт общего количества пропущенных данных по каждому признаку, доли пропущенных данных выборки.

Удаление из выборки признаков, доля пропущенных которых больше 10%.

Заполнение пропущенных данных.

missing\_values\_count = df.isnull().sum()

print("Общее количество пропущенных данных по каждому признаку:\n", missing\_values\_count)

total\_cells = np.product(df.shape)

total\_missing = missing\_values\_count.sum()

missing\_ratio = (total\_missing / total\_cells) \* 100

print("Доля пропущенных данных выборки: {:.2f} %".format(missing\_ratio))

missing\_percentages = (df.isnull().sum() / df.shape[0]) \* 100

missing\_features = list(missing\_percentages[missing\_percentages > 10].index)

df = df.drop(columns=missing\_features)

print("Признаки, доля пропущенных которых больше 10%:", missing\_features)

categorical\_cols = df.select\_dtypes(include=['object']).columns

df[categorical\_cols] = df[categorical\_cols].fillna('unknown')

numeric\_columns = df.select\_dtypes(include=['float']).columns

df[numeric\_columns] = df[numeric\_columns].fillna(df[numeric\_columns].mean())

df

Общее количество пропущенных данных по каждому признаку:

MSSubClass 0

MSZoning 0

LotFrontage 259

LotArea 0

Street 0

...

MoSold 0

YrSold 0

SaleType 0

SaleCondition 0

SalePrice 0

Доля пропущенных данных выборки: 6.70 %

Признаки, доля пропущенных которых больше 10%: ['LotFrontage', 'Alley', 'MasVnrType', 'FireplaceQu', 'PoolQC', 'Fence', 'MiscFeature']

Изображение выглядит как текст, снимок экрана, черный, клавиатура

Автоматически созданное описание

Категориальные переменные:

линейная регрессия не умеет обрабатывать категориальные признаки.

Конвертация всех категориальных признаков в вектора числовых.

df = pd.get\_dummies(df, columns=df.select\_dtypes(include=['object']).columns)

print("Number of Features:", len(df.columns))

Разделение данных на данные и целевую переменную.

X = df.drop(columns=['SalePrice'])

y = df['SalePrice']

Переобучение:

Обучим обычную модель без регуляризации.

Выведем все коэффициенты для каждого признака, отсортировав по возрастанию по модулю значения коэффициента.

Вывод среднего ошибки предсказания на 5 фолдах (cv=5) для этой линейной регресии.

from sklearn.linear\_model import LinearRegression

from sklearn.model\_selection import cross\_val\_score

model = LinearRegression()

model.fit(X, y)

print("Коэффициенты признаков отсортированные по возрастанию модуля")

coef = model.coef\_

sorted\_coef\_index = sorted(range(len(coef)), key=lambda k: abs(coef[k]))

for i in sorted\_coef\_index:

    print(f"{X.columns[i]}: {coef[i]:.2f}")

print("Cреднее значение ошибки предсказания на 5 фолдах")

mse\_scores = cross\_val\_score(model, X, y, cv=5, scoring='neg\_mean\_squared\_error')

mse\_scores = -mse\_scores

print("Mean MSE:", mse\_scores.mean())

Коэффициенты признаков отсортированные по возрастанию модуля

MiscVal: 0.12

LotArea: 0.76

BsmtUnfSF: -1.92

OpenPorchSF: 2.53

EnclosedPorch: 4.41

BsmtFinSF2: 7.02

WoodDeckSF: 13.41

MasVnrArea: 13.43

1stFlrSF: 15.37

BsmtFinSF1: 17.42

GarageArea: 18.38

LowQualFinSF: -19.53

MSSubClass: -19.79

TotalBsmtSF: 22.52

ScreenPorch: 28.59

GarageYrBlt: -30.18

GrLivArea: 31.15

CentralAir\_N: -31.17

CentralAir\_Y: 31.17

3SsnPorch: 31.41

2ndFlrSF: 35.32

Functional\_Mod: 84.63

PoolArea: 85.11

YearRemodAdd: 100.63

...

Condition2\_PosN: -192203.73

RoofMatl\_ClyTile: -608415.24

Cреднее значение ошибки предсказания на 5 фолдах

Mean MSE: 1221254100.9779387

Ridge и Lasso:

Обучение Lasso и Ridge для разных значений регуляризации alpha.

alphas = np.arange(1, 501, 10)

lasso\_coefs = []

ridge\_coefs = []

for alpha in alphas:

    lasso = Lasso(alpha=alpha)

    lasso.fit(X, y)

    lasso\_coefs.append(lasso.coef\_)

    ridge = Ridge(alpha=alpha)

    ridge.fit(X, y)

    ridge\_coefs.append(ridge.coef\_)

График значений коэффициентов модели для каждого признака. По оси X значение alpha, по Y - значение коэффициента. на графике.

colors = plt.cm.rainbow(np.linspace(0, 1, len(lasso\_coefs)))

for i, color in enumerate(colors):

    plt.plot(alphas, [coef[i] for coef in lasso\_coefs], color=color)

plt.xlabel('Alpha')

plt.ylabel('Coefficient Value Lasso')

plt.title('Coefficients vs Alpha')

plt.legend(loc='upper right')

plt.show()

for i, color in enumerate(colors):

    plt.plot(alphas, [coef[i] for coef in ridge\_coefs], color=color)

plt.xlabel('Alpha')

plt.ylabel('Coefficient Value Ridge')

plt.title('Coefficients vs Alpha')

plt.legend(loc='upper right')

plt.show()

Изображение выглядит как снимок экрана, линия, диаграмма, График

Автоматически созданное описание

Изображение выглядит как снимок экрана, текст, линия, диаграмма

Автоматически созданное описание

Гистограмма, по оси Y количество коэффициентов, равных 0 (<0.0001) для Loss регресиии, а по оси X - значение коэффициента альфа.

num\_zero\_coefs = [np.sum(np.abs(coef) < 0.0001) for coef in lasso\_coefs]

plt.figure(figsize=(10, 6))

plt.bar(alphas, num\_zero\_coefs, width=15, align='center')

plt.xlabel('Alpha')

plt.ylabel('Number of zero coefficients')

plt.title('Number of Zero Coefficients for Different Alpha Values in Lasso Regression')

plt.show()

Изображение выглядит как текст, снимок экрана, диаграмма, График

Автоматически созданное описание

Поиск оптимального alpha:

С помощью метода LassoCV поиск оптимальное значение alpha.

from sklearn.linear\_model import LassoCV

lasso = LassoCV(cv=5)

lasso.fit(X, y)

optimal\_alpha = lasso.alpha\_

print("Оптимальное значение alpha:", optimal\_alpha)

Оптимальное значение alpha: 209067.7746829921

График зависимости MSE от alpha.

mse\_path = lasso.mse\_path\_

plt.plot(lasso.alphas\_, mse\_path.mean(axis=1))

plt.xlabel('alpha')

plt.ylabel('MSE')

plt.title('Зависимость MSE от alpha')

plt.show()

Изображение выглядит как текст, снимок экрана, линия, диаграмма

Автоматически созданное описание-

Гистограмма магнитуды признаков, на которой показано значение коэффициента для каждого признака. График отсортирован по величине коэффициентов. Признаки с нулевыми коэффициентами удалены.

coef = lasso.coef\_

df\_coef = pd.DataFrame({'Признак': X.columns, 'Коэффициент': coef})

df\_coef = df\_coef[df\_coef['Коэффициент'] != 0]  # Удаление признаков с нулевыми коэффициентами

df\_coef = df\_coef.sort\_values(by='Коэффициент')  # Сортировка по величине коэффициентов

plt.barh(df\_coef['Признак'], df\_coef['Коэффициент'])

plt.xlabel('Коэффициент')

plt.ylabel('Признак')

plt.title('Гистограмма магнитуда признаков')

plt.show()

num\_nonzero\_features = len(df\_coef)

print("Количество ненулевых признаков:", num\_nonzero\_features)

Изображение выглядит как текст, снимок экрана, дисплей, число

Автоматически созданное описание

Количество ненулевых признаков: 11.

mse\_scores = cross\_val\_score(lasso, X, y, scoring='neg\_mean\_squared\_error', cv=5)

mean\_mse = -mse\_scores.mean()

print("Среднее значение MSE на 5 фолдах:", mean\_mse)

Среднее значение MSE на 5 фолдах: 1885748973.7168381.

Предсказание на отложенной выборке:

Разбиение датасета на 2 части - тренировочную и тестовую в соотношении 0.33.

from sklearn.model\_selection import train\_test\_split

X\_train, X\_test, y\_train, y\_test = train\_test\_split(X, y, test\_size=0.33, shuffle=True, random\_state=42)

Теперь, возьмём модель. Запустим её тренировку по тренировочной выборке X\_train, y\_train . Построем предсказания.

model = Lasso()

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

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

Визуализация сравнения предсказанных значений и настоящих.

indices = np.arange(len(y\_pred))

plt.scatter(y\_pred, indices, label='Predictions')

plt.scatter(y\_test, indices,  label='Actual Values')

plt.legend()

plt.show()

**Изображение выглядит как снимок экрана, текст

Автоматически созданное описание**

plt.scatter(y\_test, y\_pred)

plt.plot([y\_test.min(), y\_test.max()], [y\_test.min(), y\_test.max()], 'r--')

plt.xlabel('Настоящие значения')

plt.ylabel('Предсказанные значения')

plt.title('Сравнение предсказаний с точными значениями')

plt.show()

Изображение выглядит как текст, снимок экрана, линия, График

Автоматически созданное описание

Как видно из графиков, модель имеет достаточно большую точность предсказаний.

## **Заключение**

В ходе выполнения курсового проекта было проведено детальное исследование линейной регрессии - одного из основных методов машинного обучения, используемого для прогнозирования числовых значений.

Были изучены основные принципы линейной регрессии, включая формулу линейной функции, методы поиска оптимальных параметров модели, а также способы оценки качества модели.

В задании был предложен набор данных, на основе которого проводился анализ и построение модели линейной регрессии. Был выполнен предварительный анализ данных, включающий их визуализацию, обработку пропущенных значений и выбросов, а также выбор признаков, наиболее коррелирующих с целевой переменной.

Затем была построена модель линейной регрессии с использованием выбранных признаков, проведена ее оценка на тренировочной выборке и оценка качества предсказаний на тестовой выборке. В ходе оценки были использованы различные метрики, такие как MSE (среднеквадратичная ошибка) и R^2 (коэффициент детерминации).

На основе полученных результатов можно сделать вывод о качестве модели: ее предсказания достаточно точные, что свидетельствует о ее пригодности для данной задачи. Однако стоит отметить, что в дальнейшем возможно улучшение модели путем добавления новых признаков или использования более сложных моделей, таких как полиномиальная регрессия или регрессия с регуляризацией.

Таким образом, выполнение курсового проекта по линейной регрессии позволяет получить навыки работы с одним из основных инструментов машинного обучения и развить умения анализировать, строить и оценивать модели для прогнозирования числовых значений. Результаты данного проекта могут быть полезными в решении практических задач прогнозирования в различных областях, таких как финансы, маркетинг, медицина и др.

[**Список использованных источников**](file:///C:\Users\Алёна\Desktop\мага%202\кп\Реферат,%20Миролюбова%20Алёна,%20%20М9122-01.04.02мкт.docx#_Toc61081865)

1. [Документация Scikit Learn](https://scikit-learn.org/stable/)
2. Флах П. Машинное обучение. — М.: ДМК Пресс, 2015. — 400 с.
3. Ян Лекун. Как учится машина. Революция в области нейронных сетей и глубокого обучения. (Библиотека Сбера: Искусственный интеллект). — М.: Альпина нон-фикшн, 2021.
4. Е.З. Демиденко. Линейная и нелинейная регрессия. — М.: Финансы и статистика, 1981. — 302 с.