Вопрос № 1. Рассмотрите основные особенности обучения моделей машинного обучения на размеченных данных.

Вопрос № 2. Проанализируйте виды признаков, используемых в машинном обучении.

Вопрос № 3. Рассмотрите возможности использования линейных моделей в задачах регрессии.

Вопрос № 4. Рассмотрите возможности использования линейных моделей в задачах классификации.

Вопрос № 5. Задача регрессии - предсказание значений непрерывной целевой переменной

Вопрос № 6. Стохастический градиентный спуск

Вопрос № 7. K ближайших соседей

Вопрос № 8. Простая линейная регрессия

Вопрос № 9. Логистическая регрессия.

Вопрос № 10. Деревья принятия решений.

Вопрос № 11. Нейронные сети прямого распростанения

Вопрос № 12. Нейронные сети обратного распространения.

Password @ 159357

**Вопрос №1.**

1.1 Задайте функцию, которая не принимает никаких аргументов и просто выводит на экран строку: It's my first function

В конце программы вызовите эту функцию.

*def my\_function():*

*print("It's my first function")*

*# вызов функции*

*my\_function()*

1.2 Имеется список с оценками студента:

m = [2, 3, 5, 5, 2, 2, 3, 3, 4, 5, 4, 4]

Необходимо с помощью срезов выбрать элементы с 3-го по 7-й (включительно) и вывести их на экран в обратном порядке.

*m = [2, 3, 5, 5, 2, 2, 3, 3, 4, 5, 4, 4]*

*# выбираем элементы с 3-го по 7-й (включительно) с помощью среза*

*selected\_elements = m[2:7]*

*# выводим выбранные элементы на экран в обратном порядке*

*print(selected\_elements[::-1])*

*В результате выполнения этого кода на экран будет выведен список* ***[3, 2, 5, 5, 3]****, который содержит элементы с 3-го по 7-й (включительно) в обратном порядке.*

**Вопрос №2.**

2.1 Вводятся целые числа в одну строчку через пробел. Необходимо преобразовать их в список lst , затем, удалить последнее значение и если оно нечетное, то в список (в конец) добавить True, иначе - False. Отобразить полученный список на экране командой: print(\*lst)

Реализовать программу без использования условного оператора.

*# вводим числа через пробел и преобразуем их в список*

*lst = list(map(int, input().split()))*

*# определяем четность последнего значения*

*last\_element\_is\_odd = lst[-1] % 2 == 1*

*# удаляем последнее значение из списка*

*lst.pop()*

*# добавляем в список значение, соответствующее четности последнего элемента*

*lst.append(last\_element\_is\_odd)*

*# выводим список на экран*

*print(\*lst)*

*4 5 3 2*

*4 5 3 False*

*В этом примере мы сначала проверяем длину списка с помощью функции* ***len()****. Если в списке есть как минимум два элемента, мы выполняем операцию* ***pop()*** *и добавляем новый элемент в конец списка. Если же в списке менее двух элементов, мы не выполняем никаких операций.*

2.2 На вход программы поступают данные в виде набора строк в формате:

ключ1=значение1  
ключ2=значение2  
...  
ключN=значениеN

Ключами здесь выступают целые числа (см. пример ниже). Необходимо их преобразовать в словарь d (без использования функции dict()) и вывести его на экран командой:

print(\*sorted(d.items()))

P. S. Для считывания списка целиком в программе уже записаны начальные строчки.

*# начальные строки программы*

*input\_data = ['1=apple', '2=banana', '3=cherry']*

*# создаем пустой словарь*

*d = {}*

*# проходим по каждой строке в списке input\_data*

*for line in input\_data:*

*# разделяем строку на ключ и значение*

*key, value = line.split('=')*

*# преобразуем ключ в целое число*

*key = int(key)*

*# добавляем ключ и значение в словарь*

*d[key] = value*

*# выводим отсортированный словарь на экран*

*print(\*sorted(d.items()))*

*(1, 'apple') (2, 'banana') (3, 'cherry')*

*Здесь мы сначала создаем пустой словарь* ***d****. Затем мы проходим по каждой строке в списке* ***input\_data****. В каждой строке мы разделяем ключ и значение, используя разделитель* ***'='****. Затем мы преобразуем ключ в целое число с помощью функции* ***int()****. Наконец, мы добавляем ключ и значение в словарь* ***d*** *с помощью операции присваивания.*

*После того, как мы обработали все строки в списке* ***input\_data****, мы выводим отсортированный словарь на экран, используя метод* ***items()*** *для получения пар ключ-значение и функцию* ***sorted()*** *для их сортировки. Функция* ***print()*** *с аргументом* ***\**** *развертывает пары ключ-значение в отдельные аргументы, разделенные пробелами.*

**Вопрос №3.**

3.1 Вводятся оценки студента (числа от 2 до 5) в одну строку через пробел. Необходимо определить количество двоек и вывести это значение на экран.

*# вводим оценки студента через пробел*

*grades = list(map(int, input().split()))*

*# определяем количество двоек*

*count\_of\_twos = grades.count(2)*

*# выводим количество двоек на экран*

*print(count\_of\_twos)*

*1 2 2 3*

*2*

*В этом примере мы вводим оценки студента через пробел и преобразуем их в список* ***grades*** *с помощью функции* ***map()*** *и* ***list()****. Затем мы используем метод* ***count()*** *для подсчета количества вхождений числа 2 в списке* ***grades****. Полученное значение сохраняем в переменную* ***count\_of\_twos****. Наконец, мы выводим количество двоек на экран с помощью функции* ***print()****.*

3.2 Вводятся данные в формате ключ=значение в одну строчку через пробел. Необходимо на их основе создать словарь, затем проверить, существуют ли в нем ключи со значениями: 'house', 'True' и '5' (все ключи - строки). Если все они существуют, то вывести на экран ДА, иначе - НЕТ.

*# вводим данные в формате ключ=значение через пробел*

*data = input().split()*

*# создаем пустой словарь*

*dictionary = {}*

*# заполняем словарь значениями из введенных данных*

*for item in data:*

*key, value = item.split('=')*

*dictionary[key] = value*

*# проверяем наличие требуемых ключей в словаре*

*key1\_exists = 'house' in dictionary*

*key2\_exists = 'True' in dictionary*

*key3\_exists = '5' in dictionary*

*# выводим результат на экран*

*if key1\_exists and key2\_exists and key3\_exists:*

*print('ДА')*

*else:*

*print('НЕТ')*

house=01 True=02 5=03

ДА

*В этом примере мы сначала вводим данные в формате "ключ=значение" через пробел и разделяем их с помощью метода split(), сохраняя результат в список data. Затем мы создаем пустой словарь dictionary.*

*Затем мы проходим по каждому элементу списка data и разделяем его на ключ и значение с помощью метода split('='). Затем мы добавляем полученные пары ключ-значение в словарь dictionary.*

*Далее мы проверяем наличие требуемых ключей 'house', 'True' и '5' в словаре с помощью оператора in. Результат проверки сохраняем в отдельные переменные key1\_exists, key2\_exists и key3\_exists.*

*Наконец, мы выводим результат на экран в зависимости от наличия всех требуемых ключей в словаре с помощью условного оператора if-else. Если все ключи существуют, выводим 'ДА', иначе выводим 'НЕТ'.*

**Вопрос №4.**

4.1 Вводятся данные в формате ключ=значение в одну строчку через пробел. Необходимо на их основе создать словарь d, затем удалить из этого словаря ключи 'False' и '3', если они существуют. Ключами и значениями словаря являются строки. Вывести полученный словарь на экран командой: print(\*sorted(d.items()))

*# вводим данные в формате ключ=значение через пробел*

*data = input().split()*

*# создаем пустой словарь*

*d = {}*

*# заполняем словарь значениями из введенных данных*

*for item in data:*

*key, value = item.split('=')*

*d[key] = value*

*# удаляем ключи 'False' и '3', если они существуют*

*if 'False' in d:*

*del d['False']*

*if '3' in d:*

*del d['3']*

*# выводим полученный словарь на экран*

*print(\*sorted(d.items()))*

*a=01 False=02 3=03 4=04*

*('4', '04') ('a', '01')*

*В этом примере мы сначала вводим данные в формате "ключ=значение" через пробел и разделяем их с помощью метода* ***split()****, сохраняя результат в список* ***data****. Затем мы создаем пустой словарь* ***d****.*

*Затем мы проходим по каждому элементу списка* ***data*** *и разделяем его на ключ и значение с помощью метода* ***split('=')****. Затем мы добавляем полученные пары ключ-значение в словарь* ***d****.*

*Далее мы проверяем наличие ключей 'False' и '3' в словаре* ***d*** *с помощью оператора* ***in****. Если они существуют, мы используем оператор* ***del*** *для удаления соответствующих ключей из словаря.*

*Наконец, мы выводим полученный словарь на экран, сортируя его пары ключ-значение с помощью функции* ***sorted()*** *и разворачивая их с помощью оператора* ***\**** *в отдельные аргументы для функции* ***print()****.*

4.2 Вводятся номера телефонов в одну строчку через пробел с разными кодами стран: +7, +6, +2, +4 и т.д. Необходимо составить словарь d, где ключи - это коды +7, +6, +2 и т.п., а значения - список номеров (следующих в том же порядке, что и во входной строке) с соответствующими кодами. Полученный словарь вывести командой: print(\*sorted(d.items()))

*# вводим номера телефонов в одну строку через пробел*

*phone\_numbers = input().split()*

*# создаем пустой словарь*

*d = {}*

*# проходим по каждому номеру телефона*

*for number in phone\_numbers:*

*# извлекаем код страны из номера*

*country\_code = number[:2]*

*# если код страны уже есть в словаре, добавляем номер в соответствующий список*

*if country\_code in d:*

*d[country\_code].append(number)*

*# иначе создаем новую запись в словаре с пустым списком номеров*

*else:*

*d[country\_code] = [number]*

*# выводим полученный словарь на экран*

*print(\*sorted(d.items()))*

*+77712110729*

*('+7', ['+77712110729'])*

*В этом примере мы сначала вводим номера телефонов в одну строку через пробел и разделяем их с помощью метода* ***split()****, сохраняя результат в список* ***phone\_numbers****. Затем мы создаем пустой словарь* ***d****.*

*Затем мы проходим по каждому номеру телефона в списке* ***phone\_numbers****. Для каждого номера мы извлекаем первые два символа, которые представляют код страны, с помощью среза* ***number[:2]****. Затем мы проверяем, есть ли уже такой код страны в словаре* ***d****. Если код страны уже есть, мы добавляем текущий номер в соответствующий список номеров в словаре. Если же код страны отсутствует в словаре, мы создаем новую запись с ключом в виде кода страны и значением в виде пустого списка, и добавляем текущий номер в этот список.*

*Наконец, мы выводим полученный словарь на экран, сортируя его пары ключ-значение с помощью функции* ***sorted()*** *и разворачивая их с помощью оператора* ***\**** *в отдельные аргументы для функции* ***print()****.*

**Вопрос №5.**

5.1 Пользователь вводит в цикле целые положительные числа, пока не введет число 0. Для каждого числа вычисляется квадратный корень (с точностью до сотых) и значение выводится на экран (в столбик). С помощью словаря выполните кэширование данных так, чтобы при повторном вводе того же самого числа результат не вычислялся, а бралось ранее вычисленное значение из словаря. При этом на экране должно выводиться: значение из кэша: <число>

*import math*

*# создаем пустой словарь для кэширования результатов*

*cache = {}*

*while True:*

*# вводим число*

*number = int(input("Введите число (0 для выхода): "))*

*# проверяем, является ли число нулем*

*if number == 0:*

*break*

*# проверяем, есть ли значение в кэше*

*if number in cache:*

*print("Значение из кэша:", cache[number])*

*else:*

*# вычисляем квадратный корень с точностью до сотых*

*sqrt\_value = round(math.sqrt(number), 2)*

*# сохраняем результат в кэше*

*cache[number] = sqrt\_value*

*print(sqrt\_value)*

*Введите число (0 для выхода): 1*

*1.0*

*Введите число (0 для выхода): 2*

*1.41*

*Введите число (0 для выхода): 2*

*Значение из кэша: 1.41*

*Введите число (0 для выхода): 9*

*3.0*

*Введите число (0 для выхода): 0*

*В этом примере мы создаем пустой словарь* ***cache*** *для кэширования результатов. Затем мы входим в цикл, который будет выполняться до тех пор, пока пользователь не введет число 0.*

*В каждой итерации цикла мы сначала вводим число с помощью функции* ***input()****, преобразуем его в целое число с помощью* ***int()****, и сохраняем результат в переменную* ***number****.*

*Затем мы проверяем, есть ли значение в словаре* ***cache*** *для данного числа. Если значение уже есть в кэше, мы выводим сообщение "Значение из кэша:" и выводим соответствующее значение из словаря* ***cache****. Если значение отсутствует в кэше, мы вычисляем квадратный корень с помощью функции* ***math.sqrt()*** *и округляем его до двух знаков после запятой с помощью функции* ***round()****. Затем сохраняем результат в кэше, используя число как ключ и вычисленное значение как значение. После этого мы выводим вычисленный квадратный корень на экран.*

*Таким образом, при повторном вводе того же числа результат будет браться из словаря* ***cache****, что обеспечивает кэширование данных.*

5.2 Имеется закодированная строка с помощью азбуки Морзе. Коды разделены между собой пробелом. Необходимо ее раскодировать, используя азбуку Морзе из предыдущего занятия. Полученное сообщение (строку) вывести на экран.

*# Словарь с кодами Морзе*

*morse\_code = {*

*'.-': 'A', '-...': 'B', '-.-.': 'C', '-..': 'D', '.': 'E',*

*'..-.': 'F', '--.': 'G', '....': 'H', '..': 'I', '.---': 'J',*

*'-.-': 'K', '.-..': 'L', '--': 'M', '-.': 'N', '---': 'O',*

*'.--.': 'P', '--.-': 'Q', '.-.': 'R', '...': 'S', '-': 'T',*

*'..-': 'U', '...-': 'V', '.--': 'W', '-..-': 'X', '-.--': 'Y',*

*'--..': 'Z', '.----': '1', '..---': '2', '...--': '3', '....-': '4',*

*'.....': '5', '-....': '6', '--...': '7', '---..': '8', '----.': '9',*

*'-----': '0', '--..--': ',', '.-.-.-': '.', '..--..': '?', '-..-.': '/',*

*'-....-': '-', '-.--.': '(', '-.--.-': ')', '.-.-.': '+', '-...-': '=',*

*'.-..-.': '"', '.--.-.': '@', '...---...': 'SOS'*

*}*

*# Закодированная строка*

*encoded\_string = input("Введите закодированную строку: ")*

*# Разделяем закодированную строку на коды символов*

*codes = encoded\_string.split(' ')*

*# Раскодированная строка*

*decoded\_string = ''*

*# Проходим по каждому коду символа и добавляем соответствующий символ в раскодированную строку*

*for code in codes:*

*if code in morse\_code:*

*decoded\_string += morse\_code[code]*

*else:*

*# Если код не найден в словаре, добавляем пробел*

*decoded\_string += ' '*

*# Выводим раскодированную строку на экран*

*print(decoded\_string)*

*Введите закодированную строку: .-. .. ... .--.*

*RISP*

*Программа предлагает ввести закодированную строку и сохраняет ее в переменной* ***encoded\_string****. Затем мы разделяем закодированную строку на отдельные коды символов с помощью метода* ***split(' ')****, сохраняя результат в списке* ***codes****.*

*Затем мы создаем пустую строку* ***decoded\_string****, которая будет содержать раскодированное сообщение.*

*Затем мы проходим по каждому коду символа в списке* ***codes*** *и проверяем, есть ли этот код в словаре* ***morse\_code****. Если код найден, мы добавляем соответствующий символ в строку* ***decoded\_string****. Если код не найден в словаре, мы добавляем пробел*

**Вопрос №6.**

1.В список: a = [5.4, 6.7, 10.4]

добавить в конец вложенный список со значениями, вводимыми в программу (целые числа вводятся в строчку через пробел). Результирующий список lst вывести на экран командой: print(lst)

2. Имеется кортеж: t = (3.4, -56.7)

Вводится последовательность целых чисел в одну строчку через пробел. Необходимо их добавить в кортеж t. Результат вывести на экран командой: print(t)

**Вопрос №7.**

1. Вводятся целые числа в одну строку через пробел. На их основе формируется кортеж. Необходимо найти и вывести все индексы неуникальных (повторяющихся) значений в этом кортеже. Результат отобразите в виде строки чисел, записанных через пробел.
2. Вводятся три строчки стихотворения (каждая с новой строки). Сохранить его в виде вложенного списка с разбивкой по строкам и словам (слова разделяются пробелом). Результирующий список lst вывести на экран командой: print(lst)

**Вопрос №8.**

1. Вводится  матрица чисел из трех строк. В каждой строке числа разделяются пробелом. Необходимо вывести на экран последний столбец этой матрицы в виде строки из трех чисел через пробел.
2. Имеется двумерный кортеж, размером 5 x 5 элементов:

t = ((1, 0, 0, 0, 0),

 (0, 1, 0, 0, 0),

 (0, 0, 1, 0, 0),

 (0, 0, 0, 1, 0),

 (0, 0, 0, 0, 1))

Вводится натуральное число N (N < 5). Необходимо на основе кортежа t сформировать новый аналогичный кортеж t2 размером N x N элементов. Результат вывести на экран в виде таблицы чисел.

**Вопрос №9.**

1. Вводятся два вещественных числа в одну строку через пробел. Вывести на экран наибольшее из чисел. Задачу решить с помощью условного оператора.
2. Вводится текст в одну строку, слова разделены пробелом. Необходимо подсчитать число уникальных слов (без учета регистра) в этом тексте. Результат (число уникальных слов) вывести на экран.

**Вопрос №10.**

1. Вводится слово. Необходимо определить, является ли это слово палиндромом (одинаково читается вперед и назад, например, АННА). Регистр букв не учитывать. Если введенное слово палиндром, на экран вывести ДА, иначе - НЕТ.
2. Вводятся два списка целых чисел каждый с новой строки (в строке наборы чисел через пробел). Необходимо выбрать и отобразить на экране уникальные числа, присутствующие в первом списке, но отсутствующие во втором. Результат выведите на экран в виде строки чисел, записанных по возрастанию через пробел.

**Вопрос №11.**

1. Вводятся два целых положительных числа m и n в одну строку через пробел. Если число m делится нацело на число n, то вывести на экран частное от деления (результат деления) в виде целого числа. В противном случае вывести сообщение «m на n нацело не делится» (без кавычек) и вместо m и n подставить соответствующие числа, например: «13 на 2 нацело не делится».
2. Вводятся два списка целых чисел каждый с новой строки (в строке наборы чисел через пробел). Необходимо выбрать и отобразить на экране уникальные числа, присутствующие в первом или втором списках, но отсутствующие одновременно в обоих. Результат выведите на экран в виде строки чисел, записанных по возрастанию через пробел.

**Вопрос №12.**

1. Вводится список оценок студента - его ответов у доски по предмету "Информатика" в виде чисел от 2 до 5 в одну строку через пробел. Если студент имеет хотя бы одну двойку, то он не допускается до экзамена. Определить на основе введенного списка, допущен ли студент. Если допущен, то вывести слово ДОПУЩЕН, иначе - НЕ ДОПУЩЕН. При реализации задачи используйте множество для определения наличия двойки.
2. Вводится натуральное число, которое может содержать только простые множители 1, 2, 3, 5 и 7 (любые из них, не обязательно все). Необходимо разложить введенное число на простые множители и проверить, содержит ли оно множители 2, 3 и 5 (обязательно все их, хотя бы один раз). Если это так, то вывести ДА, иначе - НЕТ.