

**Федеральное государственное автономное образовательное  
учреждение высшего образования  
«Московский физико-технический институт  
(национальный исследовательский университет)»**

**УТВЕРЖДЕНО**

**Директор по цифровизации  
образования**

**Д.И. Гриц**

	<b>Рабочая программа дисциплины (модуля)</b>
<b>по дисциплине:</b>	Продвинутые методы машинного обучения
<b>по направлению:</b>	Информатика и вычислительная техника
<b>профиль подготовки:</b>	Технологическое лидерство Физтех-школа Прикладной Математики и Информатики центр дополнительного, дополнительного профессионального и онлайн-образования "Пуск"
<b>курс:</b>	1
<b>квалификация:</b>	магистр

Семестры, формы промежуточной аттестации:

- 1 (осенний) - Зачет
- 2 (весенний) - Зачет

Аудиторных часов: 150 всего, в том числе:

- лекции: 60 час.
- семинары: 90 час.
- лабораторные занятия: 0 час.

Самостоятельная работа: 120 час.

Всего часов: 270, всего зач. ед.: 6

Программу составили:

А.О. Бугрий, старший методист  
Ж.И. Зубцова, канд. физ.-мат. наук, доцент  
Р.Г. Нейчев, старший преподаватель

Программа обсуждена на заседании центра дополнительного, дополнительного профессионального и онлайн-образования "Пуск" 23.03.2023

## Аннотация

Дисциплина состоит из двух модулей:

Модуль 1. Анализ данных в Python и введение в машинное обучение

Модуль 2. Машинное обучение

По итогам обучения обучающийся будет способен формализовать и алгоритмизировать поставленную задачу, написать программный код с использованием языков программирования, оформить код в соответствии с установленными требованиями.

### 1. Цели и задачи

#### Цель дисциплины

Целью реализации дисциплины является формирование/совершенствование компетенций слушателей в области решения профессиональных задач по анализу данных с применением Python и машинному обучению.

#### Задачи дисциплины

- сформировать умение использовать базовые типы и конструкции языка программирования Python;
- сформировать умение работать со стандартными структурами данных в Python, писать функции на Python, применять функциональные особенности языка, работать с файлами с помощью языка Python;
- сформировать умение применять механизмы наследования, создавать классы и работать с ними, обрабатывать исключения;
- сформировать умение искать и исправлять ошибки в программе на Python, тестировать программы на Python;
- сформировать умение писать многопоточный код на Python, писать асинхронный код на Python, работать с сетью, создать свое серверное сетевое приложение;
- сформировать умение пользоваться библиотеками Python для работы с данными;
- сформировать умение решать оптимизационные задачи с помощью Python;
- сформировать умение использовать математический аппарат для работы с данными;
- сформировать навыки построения предсказывающих моделей;
- сформировать умение оценивать качество построенных моделей;
- сформировать умение применять инструменты Python для решения задач машинного обучения.

### 2. Перечень формируемых компетенций

Освоение дисциплины направлено на формирование следующих компетенций:

Код и наименование компетенции	Индикаторы достижения компетенции
УК-1 Способен осуществлять критический анализ проблемных ситуаций на основе системного подхода, вырабатывать стратегию действий	УК-1.1 Анализирует проблемную ситуацию как систему, выявляя ее составляющие и связи между ними

### 3. Перечень планируемых результатов обучения по дисциплине (модулю)

В результате освоения дисциплины обучающиеся должны знать:

- особенности комбинирования диаграмм внутри дашбордов и презентаций
- условия применения линейных моделей
- основные подходы градиентной оптимизации
- основные виды диаграмм и элементы
- формальную постановку задачи машинного обучения с учителем
- формальную постановку задачи линейной классификации
- основные архитектуры
- основные особенности восприятия зрительной информации человеком
- инструменты для визуализации данных, используемые в работе аналитика
- инструменты описательной статистики
- основные понятия линейной алгебры
- основные понятия теории вероятностей
- основные нелинейные функции активации
- специфические свойства деревьев
- формальную постановку задачи обработки естественного языка
- интерфейс SK-learn
- основные техники регуляризации в глубоком обучении
- понятия Padding, Strides, Pooling
- основные возможности библиотек и основные типы задач, которые можно решить с помощью библиотек
- функциональные возможности Yandex DataLens
- основные понятия машинного обучения
- наивный Байесовский классификатор
- линейные модели машинного обучения
- какие функции потерь применимы для задачи классификации
- теорему Байеса
- теорему Гаусса-Маркова
- теорему Экарта-Янга

уметь:

- ставить исследовательскую задачу
- строить дашборды с различным уровнем детализации
- решать простые задачи регрессии (обучение с учителем)
- решать простые задачи кластеризации (обучение без учителя)
- решать задачи классификации и регрессии методом ближайших соседей
- находить производную функции, экстремумы и выпуклости функции, градиент
- производить оптимизацию гладких функций и условную оптимизацию
- улучшать качество предсказаний с помощью основных и продвинутых техник ансамблирования
- выбрать и обосновать выбор вида графика или диаграммы, наилучшим образом визуализирующую заданные метрики
- находить вероятность и условную вероятность
- оценивать качество модели для решения задачи классификации
- Выполнять практические задачи и проекты в команде
- представлять полученные результаты
- строить диаграмму распределения плотности
- решать базовые задачи из теории вероятности с помощью Python
- описывать характеристики и возможности технологии машинного обучения
- выполнять базовые операции с векторами и матрицами
- решать задачи линейной регрессии
- производить матричное разложение SVD
- решать задачи бинарной и мультиклассовой классификации
- строить модель с регуляризацией в PyTorch
- проводить разведочный анализ данных и выявлять недостаточность данных
- использовать Pandas для работы с датафреймами
- использовать библиотеку Numpy для работы с numpy массивами
- использовать библиотеки Matplotlib, Seaborn, Plotly, Bokeh, Altair, SciKit Learn для визуализации данных
- строить инфопанели в Yandex Datalens
- проводить предварительную обработку данных
- проводить корреляционный анализ
- строить графики и диаграммы с помощью библиотек altair и seaborn
- визуализировать результаты обработки и анализа данных
- проводить предиктивный анализ
- работать с векторами и матрицами с помощью NumPy
- снижать размерность признакового пространства
- оценивать подвыборку с помощью критерия мисклассификации, энтропийного критерия и критерия Джинни
- преобразовывать текстовые данные и извлекать из них признаки
- загружать данные с помощью Pandas
- реализовывать ML-pipeline
- строить дерево решений
- оценивать значимость признаков с помощью подходов Information gain, LIME, Shap
- решать практические задачи с помощью наивного Байесовского классификатора
- производить научные вычисления с помощью SciPy
- строить графики с Matplotlib
- реализовывать логистическую регрессию с помощью PyTorch
- решать задачи классификации и регрессии с помощью решающих деревьев
- реализовывать градиентный бустинг с помощью CatBoost
- описывать линейные и нелинейные зависимости с помощью нейросетей
- собирать и отбирать данные
- базово описывать характеристики и возможности технологии нейросети
- решать простые задачи с использованием технологий нейросети
- строить простейшую нейросеть на PyTorch
- строить рекуррентную нейронную сеть
- бороться с проблемой затухающих градиентов и проблемой взрывающихся градиентов
- применять сверточные слои для обработки изображений
- решать задачи классификации и регрессии с помощью градиентного бустинга
- строить модели с помощью LSTM и GRU
- бороться с переобучением нейросети с помощью регуляризации
- строить информативные векторные представления слов

владеть:

- стандартными структурами данных в Python, умением писать функции на Python, применять функциональные особенности языка, работать с файлами с помощью языка Python;
- механизмами наследования, создавать классы и работать с ними, обрабатывать исключения;
- навыками выбора подходящего метода оптимизации для конкретной задачи;
- навыками применения библиотеки Python для построения модели линейной регрессии, решающих деревьев и композиций алгоритмов, для обучения метрических алгоритмов, SVM, байесовских моделей.

#### 4. Содержание дисциплины (модуля), структурированное по темам (разделам) с указанием отведенного на них количества академических часов и видов учебных занятий

##### 4.1. Разделы дисциплины (модуля) и трудоемкости по видам учебных занятий

№	Тема (раздел) дисциплины	Трудоемкость по видам учебных занятий, включая самостоятельную работу, час.			
		Лекции	Семинары	Лаборат. работы	Самост. работа
1	Анализ данных в Python и введение в машинное обучение	30	30		75
2	Машинное обучение	30	60		45
Итого часов		60	90		120
Подготовка к экзамену		0 час.			
Общая трудоёмкость		270 час., 6 зач.ед.			

##### 4.2. Содержание дисциплины (модуля), структурированное по темам (разделам)

Семестр: 1 (Осенний)

#### 1. Анализ данных в Python и введение в машинное обучение

##### 1. Python для анализа данных

##### 1.1. Что необходимо знать о данных

Лекция

Виды данных

Предобработка данных

Инструменты работы с данными

Практическая работа

Выполнение задачи на программирование по теме урока, тестирование

Самостоятельная работа

Самостоятельное выполнение заданий по теме урока

##### 1.2. Линейная алгебра. Библиотека NumPy

Лекция

Массивы

Векторы

Практическая работа

Выполнение задачи на программирование по теме урока, тестирование

Самостоятельная работа

Самостоятельное выполнение заданий по теме урока

##### 1.3. Библиотека Pandas. Описательная статистика

Лекция  
Библиотека Pandas  
Описательные статистики  
Практическая работа  
Выполнение задачи на программирование по теме урока, тестирование  
Самостоятельная работа  
Самостоятельное выполнение заданий по теме урока

#### 1.4. Статистика вывода

Лекция  
Выборка и генеральная совокупность  
Распределения  
Оценки  
Тестирование гипотез  
Практическая работа  
Выполнение задачи на программирование по теме урока, тестирование  
Самостоятельная работа  
Самостоятельное выполнение заданий по теме урока

### 2. Визуализация данных

#### 2.1. Разбор прикладных инструментов визуализации данных

Лекция  
Yandex DataLens: построение диаграмм без программирования  
Использование библиотек Python для визуализации. Библиотека Altair  
Использование библиотек Python для визуализации. Библиотека Seaborn  
Практическая работа  
Тест и задание для самопроверки  
Самостоятельная работа  
Самостоятельное выполнение заданий по теме урока, изучение дополнительных материалов

#### 2.2. Диаграммы в контексте: инфопанели и презентации

Лекция  
Интерактивные средства и связанные представления  
Презентации на основе диаграмм. Общие практики  
Подготовка инфопанелей и презентаций с помощью Yandex DataLens  
Подготовка инфопанелей и презентаций с помощью библиотеки Altair  
Практическая работа  
Тест и задание для самопроверки  
Самостоятельная работа  
Самостоятельное выполнение заданий по теме урока, изучение дополнительных материалов

#### 2.3. Методы визуализации. Библиотека Matplotlib

Лекция  
Методы визуализации  
Библиотека Matplotlib  
Практическая работа  
Выполнение задачи на программирование по теме урока, тестирование  
Самостоятельная работа  
Самостоятельное выполнение заданий по теме урока

#### 2.4. Библиотека Seaborn и возможности дополнительных библиотек для визуализации данных.

Основы теории вероятности  
Лекция  
Библиотека Seaborn

Основы теории вероятности

Практическая работа

Выполнение задачи на программирование по теме урока, тестирование

Самостоятельная работа

Самостоятельное выполнение заданий по теме урока

## 2.5. Примеры использования библиотек Python в работе аналитика данных

Лекция

Пример анализа данных с применением Pandas

Пример анализа данных из нескольких источников. Визуализация. Пример совместного использования нескольких библиотек

Работа с матрицей корреляции в Pandas

Введение в A/B-тестирование на Python

Создание интерактивных графиков с animation(), FuncAnimation(), camera.animate()

Практическая работа

Тест и задание для самопроверки

Самостоятельная работа

Самостоятельное выполнение заданий по теме урока, изучение дополнительных материалов

## 2.6. Проект

Лекция

Задание проекта

Рубрикатор оценивания

Практическая работа

Выполнение проекта

Самостоятельная работа

Изучение дополнительных материалов

## 3. Введение в машинное обучение

3.1. Машинное обучение с учителем: линейные модели, измерение качества модели, ансамблевые модели

Лекция

Линейные модели

Измерение качества модели

Ансамблевые модели

Практическая работа

Выполнение задачи на программирование по теме урока, тестирование

Самостоятельная работа

Самостоятельное выполнение заданий по теме урока

3.2. Машинное обучение без учителя: методы кластеризации, методы понижения размерности, рекомендательные системы

Лекция

Обучение без учителя. Методы кластеризации

Методы понижения размерности

Рекомендательные системы

Практическая работа

Выполнение задачи на программирование по теме урока, тестирование

Самостоятельная работа

Самостоятельное выполнение заданий по теме урока

## 3.3. Нейросети: основы нейронных сетей, архитектуры нейронных сетей

Лекция

Основы нейронных сетей

Архитектуры нейронных сетей

Практическая работа  
Выполнение задачи на программирование по теме урока, тестирование  
Самостоятельная работа  
Самостоятельное выполнение заданий по теме урока

#### 4. Зачет

### Семестр: 2 (Весенний)

#### 2. Машинное обучение

##### 1. Математические основы машинного обучения

###### 1.1. Метод ближайших соседей

Лекция

Цели и план занятия

Основные понятия машинного обучения

Формальная задача

Метод k ближайших соседей

Реализация kNN в Python

Практическая работа

Выполнение задания на программирование

Самостоятельная работа

Самостоятельное выполнение заданий по теме лекции, изучение дополнительных материалов.

###### 1.2. Случайность. Наивный Байесовский классификатор

Лекция

Цели и план занятия

Вероятность. Свойства вероятности

Условная вероятность. Теорема Байеса

Наивный Байесовский классификатор

Реализация наивного байесовского классификатора

Эмпирические функции распределения

Практическая работа

Выполнение задачи на программирование по теме урока, тестирование

Самостоятельная работа

Самостоятельное выполнение заданий по теме урока, изучение дополнительных материалов

###### 1.3. Оптимизация

Лекция

Цели и план занятия

Производная и ее применения

Градиентная оптимизация

Условная оптимизация

Решение задачи оптимизации градиентными методами

Практическая работа

Выполнение заданий по теме лекции

Самостоятельная работа

Самостоятельное выполнение заданий по теме лекции, изучение дополнительных материалов

###### 1.4. Задача регрессии. Линейная регрессия

Лекция

Цели и план занятия

Линейные модели машинного обучения

Линейная регрессия



Теорема Гаусса-Маркова

L1 и L2 регуляризация

Решение линейной регрессии и анализ устойчивости решения

Практическая работа

Выполнение задачи на программирование по теме урока, тестирование

Самостоятельная работа

Самостоятельное выполнение заданий по теме урока, изучение дополнительных материалов

### 1.5. Обработка и визуализация данных. Матричные разложения

Лекция

Цели и план занятия

Задача снижения размерности

Метод главных компонент

Визуализация в Python на примере векторных представлений слов

Практическая работа

Выполнение задачи на программирование по теме урока, тестирование

Самостоятельная работа

Самостоятельное выполнение заданий по теме урока, изучение дополнительных материалов

### 1.6. Задача классификации. Логистическая регрессия

Лекция

Цели и план занятия

Задача линейной классификации

Правдоподобие

Логистическая регрессия

Мультиклассовая классификация

Метрики классификации

Базовое введение в PyTorch

Практическая работа

Выполнение задачи на программирование по теме урока

Самостоятельная работа

Самостоятельное выполнение заданий по теме урока, изучение дополнительных материалов

## 2. Машинное обучение

### 2.1. Метод опорных векторов. Оценка качества классификации. Методы кросс-валидации

Лекция

Цели и план занятия

Метод опорных векторов

Нелинейный метод опорных векторов

Оценка качества классификации

Методы кросс-валидации

Cross-validation riddle

Практическая работа

Выполнение задачи на программирование по теме урока, тестирование

Самостоятельная работа

Самостоятельное выполнение заданий по теме урока, изучение дополнительных материалов

### 2.2. Решающие деревья и техники ансамблирования

Лекция

Цели и план занятия

Решающие деревья

Процедура построения дерева решений

Критерии информативности: Энтропия

Критерий Джини

Критерии в задаче регрессии. Усечение деревьев

Специфические свойства деревьев

Практическая работа

Выполнение заданий по теме лекции

Самостоятельная работа

Самостоятельное выполнение заданий по теме лекции, изучение дополнительных материалов

### 2.3. Seaborn и практика по деревьям

Лекция

Цели и план занятия

Seaborn и практика по деревьям

Практическая работа

Выполнение задачи на программирование по теме урока

Самостоятельная работа

Самостоятельное выполнение заданий по теме урока, изучение дополнительных материалов

### 2.4. Случайный лес. Продвинутое техники ансамблирования. Дилемма смещения-дисперсии.

Лекция

Цели и план занятия

Техника ансамблирования

RSM

Дилемма смещения

Смешивание

Стекинг

Практическая работа

Выполнение задачи на программирование по теме урока, тестирование

Самостоятельная работа

Самостоятельное выполнение заданий по теме урока, изучение дополнительных материалов

### 2.5. Градиентный бустинг

Лекция

Цели и план занятия

Бустинг

Градиентный бустинг.

Визуализация градиентного бустинга

CatBoost

Практическая работа

Выполнение заданий по теме лекции

Самостоятельная работа

Самостоятельное выполнение заданий по теме лекции, изучение дополнительных материалов

### 2.6. Градиентный бустинг на практике

Лекция

Цели и план занятия

Градиентный бустинг. Примеры

Градиентный бустинг. Применение в Python

Практическая работа

Выполнение задачи на программирование по теме урока

Самостоятельная работа

Самостоятельное выполнение заданий по теме лекции, изучение дополнительных материалов

### 2.7. Оценка значимости признаков

Лекция

Цели и план занятия

Information gain и прочие подходы для деревьев

LIME

Shar

Практическая работа

Выполнение задачи на программирование по теме урока, тестирование

Самостоятельная работа

Самостоятельное выполнение заданий по теме лекции, изучение дополнительных материалов

## 2.8. Введение в глубокое обучение

Лекция

Цели и план занятия

История искусственных нейронных сетей

Нейронные сети

Механизм обратного распространения ошибки

Функции активации

Интерактивное демо

Нейронные сети. Итоги

Простейшая нейросеть на PyTorch

Практическая работа

Выполнение заданий по теме лекции

Самостоятельная работа

Самостоятельное выполнение заданий по теме лекции, изучение дополнительных материалов

## 2.9. SGD доработки

Лекция

Цели и план занятия

SGD доработки

Регуляризация в DL

Проблема переобучения

Аугментация и итоги

PyTorch: модель с регуляризацией

Практическая работа

Выполнение задачи на программирование по теме урока

Самостоятельная работа

Самостоятельное выполнение заданий по теме лекции, изучение дополнительных материалов

## 3. Глубокое обучение

### 3.1. Векторные представления слов

Лекция

Цели и план занятия

Введение в NLP

Предварительная обработка текста

Извлечение признаков

Векторное представление слов (Word Embeddings)

Векторные представления слов. Визуализация. (Word embeddings visualization)

Визуализация в Python

Практическая работа

Выполнение задачи на программирование по теме урока

Самостоятельная работа

Самостоятельное выполнение заданий по теме лекции, изучение дополнительных материалов

### 3.2. Рекуррентные нейронные сети. Проблема затухающего градиента

Лекция

Цели и план занятия

Языковое моделирование

Рекуррентные нейронные сети

RNN

LSTM

Проблема затухающих градиентов

Проблема взрывающихся градиентов

Языковое моделирование: реализация в Python

Рекуррентные нейронные сети: реализация в Python

Практическая работа

Выполнение задачи на программирование по теме урока

Самостоятельная работа

Самостоятельное выполнение заданий по теме лекции, изучение дополнительных материалов

### 3.3. Обработка изображений. Сверточные нейронные сети

Лекция

Цели и план занятия

Сверточные слои

Интерактивная демонстрация

Padding, Strides, Pooling

Обзор архитектур

Свертки для изображений. Базовый обзор, примеры

Практическая работа

Выполнение заданий по теме лекции

Самостоятельная работа

Самостоятельное выполнение заданий по теме лекции, изучение дополнительных материалов

### 3.4. Механизм внимания

Лекция

Цели и план занятия

Механизм внимания Attention. Обзор

Механизм внимания в математической форме

Механизм внимания Self Attention

Механизм внимания Multi-Head Attention

Реализация мв на питоне

Практическая работа

Выполнение задачи на программирование по теме урока

Самостоятельная работа

Самостоятельное выполнение заданий по теме лекции, изучение дополнительных материалов

### 3.5. Transformer

Лекция

Цели и план занятия

Обзор трансформера

Позиционное кодирование

Нормализация слоев

Преобразователь декодер

ELMO и завершение

Практическая работа

Выполнение задачи на программирование по теме урока

Самостоятельная работа

Самостоятельное выполнение заданий по теме лекции, изучение дополнительных материалов

## 4. Зачет

## **5. Описание материально-технической базы, необходимой для осуществления образовательного процесса по дисциплине (модулю)**

Система дистанционного обучения:

Обучающемуся необходимо наличие доступа в сеть интернет, компьютер.

Преподавателю курса необходимо наличие доступа администратора курса и оборудование для проведения дистанционных семинаров (вебинаров), качественный отказоустойчивый доступ в сеть интернет.

## **6.Перечень рекомендуемой литературы**

### Основная литература

1. Python и анализ данных, Электрон. версия печ. публикации / У. Маккини. — Москва, ДМК Пресс, 2020
2. Python и анализ данных, Первичная обработка данных с применением pandas, NumPy и IPython / У. Маккини. — Москва, ДМК Пресс, 2020.— URL: <https://e.lanbook.com/book/131721> (дата обращения: 26.01.2021). - Полный текст (Режим доступа : из сети МФТИ / Удаленный доступ)
3. Курс дифференциального и интегрального исчисления : в 3 т. Т. 1 : учебник для вузов : рек. М-вом образования Рос. Федерации / Г. М. Фихтенгольц ; пред. и прим. А. А. Флоринского .— 8-е изд. / .— М. : Физматлит, 2001, 2003, 2006, 2007 .— 680 с.
4. Курс аналитической геометрии и линейной алгебры [Текст], учебник для вузов /Д. В. Беклемишев. -СПб., Лань, 2019
5. Лекции по математическому анализу. В 3 частях, Часть 1, Функции одной переменной, учебник для вузов/Я. М. Дымарский , -Москва, МФТИ, 2020
6. Python и машинное обучение [Текст], крайне необходимое издание по новейшей предсказательной аналитике для более глубокого понимания методологии машинного обучения/С. Рашка, -М., ДМК Пресс, 2017

### Дополнительная литература

1. Введение в теорию вероятностей и ее приложения [Текст] : в 2 т : учеб. пособие для вузов. Т. 1 / В.Феллер ; пер. с пересмотр. 3-го англ. изд. Ю. В. Прохорова ; [придесл. А. Н. Колмогорова] .— М. : Мир, 1984 .— 528 с.
2. Аналитическая геометрия и линейная алгебра [Текст] : в 2 ч. : учеб. пособие для вузов. Ч. 1 / А. Е. Умнов ; М-во образования и науки Рос. Федерации, Моск. физико-техн. ин-т (гос. ун-т .— 2-е изд., испр. и доп. — М. : Изд-во МФТИ, 2006 .— 272 с.

## **7. Перечень ресурсов информационно-телекоммуникационной сети "Интернет", необходимых для освоения дисциплины (модуля)**

Think Python [Электронный ресурс] – Режим доступа -

<https://greenteapress.com/wp/think-python-2e/>

Automate the Boring Stuff with Python [Электронный ресурс] – Режим доступа -

<https://automatetheboringstuff.com/>

Dive Into Python 3 [Электронный ресурс] – Режим доступа

-<http://diveintopython3.problemsolving.io/>

Problem Solving with Algorithms and Data Structures using Python [Электронный ресурс] – Режим доступа -<https://runestone.academy/runestone/static/pythonds/index.html>

Swaroop Chitlur. A Byte of Python [Электронный ресурс] – Режим доступа -  
<https://wombat.org.ua/AByteOfPython/AByteofPythonRussian-2.02.pdf> – 2020.

Справочник по функциям DAX [Электронный ресурс] – Режим доступа -  
<https://docs.microsoft.com/ru-ru/dax/dax-function-reference>

## **8. Перечень информационных технологий, используемых при осуществлении образовательного процесса по дисциплине (модулю), включая перечень необходимого программного обеспечения и информационных справочных систем (при необходимости)**

Документация Postgres про сравнение строк -

<https://postgrespro.ru/docs/postgrespro/9.5/functions-matching>

Документация Postgres про другие функции работы со строками -

<https://postgrespro.ru/docs/postgrespro/9.5/functions-string>

Тестер регулярных выражений - <https://www.regextester.com>

Интерактивный учебник по SQL -<http://www.sql-tutorial.ru/ru/content.html>

Введение в анализ данных с помощью Pandas - <https://habr.com/ru/post/196980/>

Начало работы с Power BI -

<https://docs.microsoft.com/ru-ru/power-bi/fundamentals/desktop-getting-started>

Профессиональный информационно-аналитический ресурс, посвященный машинному обучению, распознаванию образов и интеллектуальному анализу данных – <http://www.machinelearning.ru/>

Информационная система «Единое окно доступа к образовательным ресурсам» (ИС «Единое окно») – <http://window.edu.ru/>

Платформа открытое образование – <https://openedu.ru/>

## **9. Методические указания для обучающихся по освоению дисциплины (модуля)**

Самостоятельная работа подразделяется на аудиторную и внеаудиторную. Аудиторную самостоятельную работу составляют практические задания, которые выполняются слушателями во время учебных занятий, результаты ее выполнения проверяются и оцениваются преподавателем в учебном процессе.

Внеаудиторная самостоятельная работа включает формы: изучение дополнительной литературы, подготовка итоговых проектов по модулям, подготовка проекта.

Основными критериями качества организации самостоятельной работы служит наличие контроля результатов самостоятельной работы.

Основными современными формами организации самостоятельной работы являются творческие работы и работа с информационными компьютерными технологиями.

**ОЦЕНОЧНЫЕ МАТЕРИАЛЫ ПО ДИСЦИПЛИНЕ (МОДУЛЮ)**

<b>по направлению:</b>	Информатика и вычислительная техника
<b>профиль подготовки:</b>	Технологическое лидерство Физтех-школа Прикладной Математики и Информатики центр дополнительного, дополнительного профессионального и онлайн-образования "Пуск"
<b>курс:</b>	1
<b>квалификация:</b>	магистр

Семестры, формы промежуточной аттестации:

- 1 (осенний) - Зачет
- 2 (весенний) - Зачет

**Разработчики:**

А.О. Бугрий, старший методист  
Ж.И. Зубцова, канд. физ.-мат. наук, доцент  
Р.Г. Нейчев, старший преподаватель

## 1. Компетенции, формируемые в процессе изучения дисциплины

Код и наименование компетенции	Индикаторы достижения компетенции
УК-1 Способен осуществлять критический анализ проблемных ситуаций на основе системного подхода, вырабатывать стратегию действий	УК-1.1 Анализирует проблемную ситуацию как систему, выявляя ее составляющие и связи между ними

## 2. Показатели оценивания компетенций

В результате изучения дисциплины «Продвинутые методы машинного обучения» обучающийся должен:

### знать:

- особенности комбинирования диаграмм внутри дашбордов и презентаций
- условия применения линейных моделей
- основные подходы градиентной оптимизации
- основные виды диаграмм и элементы
- формальную постановку задачи машинного обучения с учителем
- формальную постановку задачи линейной классификации
- основные архитектуры
- основные особенности восприятия зрительной информации человеком
- инструменты для визуализации данных, используемые в работе аналитика
- инструменты описательной статистики
- основные понятия линейной алгебры
- основные понятия теории вероятностей
- основные нелинейные функции активации
- специфические свойства деревьев
- формальную постановку задачи обработки естественного языка
- интерфейс SK-learn
- основные техники регуляризации в глубоком обучении
- понятия Padding, Strides, Pooling
- основные возможности библиотек и основные типы задач, которые можно решить с помощью библиотек
- функциональные возможности Yandex DataLens
- основные понятия машинного обучения
- наивный Байесовский классификатор
- линейные модели машинного обучения
- какие функции потерь применимы для задачи классификации
- теорему Байеса
- теорему Гаусса-Маркова
- теорему Экарта-Янга

### уметь:



- ставить исследовательскую задачу
- строить дашборды с различным уровнем детализации
- решать простые задачи регрессии (обучение с учителем)
- решать простые задачи кластеризации (обучение без учителя)
- решать задачи классификации и регрессии методом ближайших соседей
- находить производную функции, экстремумы и выпуклости функции, градиент
- производить оптимизацию гладких функций и условную оптимизацию
- улучшать качество предсказаний с помощью основных и продвинутых техник ансамблирования
- выбрать и обосновать выбор вида графика или диаграммы, наилучшим образом визуализирующую заданные метрики
- находить вероятность и условную вероятность
- оценивать качество модели для решения задачи классификации
- Выполнять практические задачи и проекты в команде
- представлять полученные результаты
- строить диаграмму распределения плотности
- решать базовые задачи из теории вероятности с помощью Python
- описывать характеристики и возможности технологии машинного обучения
- выполнять базовые операции с векторами и матрицами
- решать задачи линейной регрессии
- производить матричное разложение SVD
- решать задачи бинарной и мультиклассовой классификации
- строить модель с регуляризацией в PyTorch
- проводить разведочный анализ данных и выявлять недостаточность данных
- использовать Pandas для работы с датафреймами
- использовать библиотеку Numpy для работы с numpy массивами
- использовать библиотеки Matplotlib, Seaborn, Plotly, Bokeh, Altair, SciKit Learn для визуализации данных
- строить инфопанели в Yandex Datalens
- проводить предварительную обработку данных
- проводить корреляционный анализ
- строить графики и диаграммы с помощью библиотек altair и seaborn
- визуализировать результаты обработки и анализа данных
- проводить предиктивный анализ
- работать с векторами и матрицами с помощью NumPy
- снижать размерность признакового пространства
- оценивать подвыборку с помощью критерия мисклассификации, энтропийного критерия и критерия Джинни
- преобразовывать текстовые данные и извлекать из них признаки
- загружать данные с помощью Pandas
- реализовывать ML-pipeline
- строить дерево решений
- оценивать значимость признаков с помощью подходов Information gain, LIME, Shap
- решать практические задачи с помощью наивного Байесовского классификатора
- производить научные вычисления с помощью SciPy
- строить графики с Matplotlib
- реализовывать логистическую регрессию с помощью PyTorch
- решать задачи классификации и регрессии с помощью решающих деревьев
- реализовывать градиентный бустинг с помощью CatBoost
- описывать линейные и нелинейные зависимости с помощью нейросетей
- собирать и отбирать данные
- базово описывать характеристики и возможности технологии нейросети
- решать простые задачи с использованием технологий нейросети
- строить простейшую нейросеть на PyTorch
- строить рекуррентную нейронную сеть
- бороться с проблемой затухающих градиентов и проблемой взрывающихся градиентов
- применять сверточные слои для обработки изображений
- решать задачи классификации и регрессии с помощью градиентного бустинга
- строить модели с помощью LSTM и GRU
- бороться с переобучением нейросети с помощью регуляризации
- строить информативные векторные представления слов

**владеть:**

- стандартными структурами данных в Python, умением писать функции на Python, применять функциональные особенности языка, работать с файлами с помощью языка Python;
- механизмами наследования, создавать классы и работать с ними, обрабатывать исключения;
- навыками выбора подходящего метода оптимизации для конкретной задачи;
- навыками применения библиотеки Python для построения модели линейной регрессии, решающих деревьев и композиций алгоритмов, для обучения метрических алгоритмов, SVM, байесовских моделей.

### 3. Перечень типовых (примерных) вопросов, заданий, тем для подготовки к текущему контролю

#### Практические задания

##### Примеры тестовых вопросов:

1. Сколько примерно тысячелетий насчитывает визуализация данных (не считая глиняных таблиц и совсем редких примеров)?
2. Что демонстрирует пример Энскомба?
3. Какие два из трёх перечисленных значений слова «визуализация» релевантны цели визуализации данных?
4. К какой из областей визуализации данных ближе всего визуальная бизнес-аналитика?
5. В чём принципиальное отличие визуализации информации (infoviz) от традиционной научной визуализации данных?
6. Объект описывается 10 числовыми признаками. Сколько параметров у линейной регрессионной модели (целевая переменная – скаляр)?
7. В каком случае предпочтительно выбирать MAE в качестве функции потерь в задаче регрессии?
8. Для каких случаев доступно аналитическое решение задачи регрессии?
9. Что общего имеют PCA и kNN?
10. Какой способ регуляризации в линейной регрессии имеет тенденцию к "отбору признаков"?

##### Пример задания на программирование

Вам дан датафрейм:

```
import numpy as np
import pandas as pd
```

```
df = pd.DataFrame(np.nan, index=[0, 1, 2, 3], columns=['A', 'B'])
```

```
df.loc[0,"A"] = 1
```

```
df.loc[1,"A"] = 2
```

```
df.loc[2,"A"] = 3
```

```
df.loc[3,"A"] = 4
```

```
df.loc[0,"B"] = 5
```

```
df.loc[1,"B"] = 6
```

```
df.loc[2,"B"] = 7
```

```
df
```

```
  A  B
```

```
0  1.0  5.0
```

```
1  2.0  6.0
```

```
2  3.0  7.0
```

```
3  4.0 NaN
```

Выполните задание:

Переименуйте названия строк в последовательность чисел от 1 до 4, используя соответствующий метод pandas.

Стандартный вывод:

Нет входных данных

Ожидаемый результат:

Датафрейм с переименованными строками

```
df.set_axis([1, 2, 3, 4], axis='index')
A  B
1  1.0  5.0
2  2.0  6.0
3  3.0  7.0
4  4.0  NaN
df.set_axis(list(range(1,5)), axis='index')
```

```
A  B
1  1.0  5.0
2  2.0  6.0
3  3.0  7.0
4  4.0  NaN
```

Контрольный пример `df.index[3] == 3`

Стандартный ввод

Ожидаемый результат

True

Пример задания на программирование

В данном задании вам необходимо реализовать функции ошибки для линейной регрессии и их производные по параметрам, не используя автоматическое дифференцирование. Все методы должны быть реализованы только с использованием библиотеки `numpy`.

Ваша основная задача: вывести формулы для производных MSE, MAE, L1 и L2 регуляризационных членов в векторном случае (т.е. когда и объект `_`, и целевое значение `_` являются векторами).

Для работы вновь обратимся к Boston housing prices dataset. Он был предобработан для вашего удобства и будет загружен ниже.

# Run some setup code for this notebook.

```
import random
import numpy as np
import matplotlib.pyplot as plt
import json
with open('boston_subset.json', 'r') as iofile:
    dataset = json.load(iofile)
feature_matrix = np.array(dataset['data'])
targets = np.array(dataset['target'])
```

Имплементация функций потерь и методов регуляризации.

Для того, чтобы решить задание, вам необходимо реализовать все методы в файле `loss_and_derivatives.py`. Внимание, в данном задании не требуется использовать свободный член (bias term), т.е. линейная модель примет простой вид

$\hat{y} =$

Единичный столбец также не добавляется к матрице .

Реализуйте методы для MSE, MAE, L1 и L2 регуляризации, а также вычисления их производных (опциональное задание) по параметрам линейной модели.

Для вашего удобства данные уже предобработаны, и использование линейной модели без свободного члена не является ошибкой. В данном задании он не должен быть использован.

```
import numpy as np
```

```
class LossAndDerivatives:
```

```
    @staticmethod
```

```
    def mse(X, Y, w):
```

```
        """
```

```
        X : numpy array of shape ('n_observations', 'n_features')
```

```
        Y : numpy array of shape ('n_observations', 'target_dimensionality') or ('n_observations',)
```

```
        w : numpy array of shape ('n_features', 'target_dimensionality') or ('n_features',)
```

Return : float

single number with MSE value of linear model ( $X \cdot w$ ) with no bias term on the selected dataset.

Comment: If Y is two-dimensional, average the error over both dimensions.

"""

return np.mean((X.dot(w) - Y)\*\*2)

@staticmethod

def mae(X, Y, w):

"""

X : numpy array of shape ('n\_observations', 'n\_features')

Y : numpy array of shape ('n\_observations', 'target\_dimensionality') or ('n\_observations',)

w : numpy array of shape ('n\_features', 'target\_dimensionality') or ('n\_features',)

Return: float

single number with MAE value of linear model ( $X \cdot w$ ) with no bias term on the selected dataset.

Comment: If Y is two-dimensional, average the error over both dimensions.

"""

# YOUR CODE HERE

return

@staticmethod

def l2\_reg(w):

"""

w : numpy array of shape ('n\_features', 'target\_dimensionality') or ('n\_features',)

Return: float

single number with sum of squared elements of the weight matrix ( $\sum_{ij} w_{ij}^2$ )

Computes the L2 regularization term for the weight matrix w.

"""

# YOUR CODE HERE

return

@staticmethod

def l1\_reg(w):

"""

w : numpy array of shape ('n\_features', 'target\_dimensionality')

Return : float

single number with sum of the absolute values of the weight matrix ( $\sum_{ij} |w_{ij}|$ )

Computes the L1 regularization term for the weight matrix w.

"""

# YOUR CODE HERE

return

```
@staticmethod
def no_reg(w):
    """
    Simply ignores the regularization
    """
    return 0.
```

```
@staticmethod
def mse_derivative(X, Y, w):
    """
    X : numpy array of shape (`n_observations`, `n_features`)
    Y : numpy array of shape (`n_observations`, `target_dimensionality`) or (`n_observations`,)
    w : numpy array of shape (`n_features`, `target_dimensionality`) or (`n_features`,)

    Return : numpy array of same shape as `w`

    Computes the MSE derivative for linear regression (X.dot(w)) with no bias term
    w.r.t. w weight matrix.

    Please mention, that in case `target_dimensionality` > 1 the error is averaged along this
    dimension as well, so you need to consider that fact in derivative implementation.
    """

    # YOUR CODE HERE
    return
```

```
@staticmethod
def mae_derivative(X, Y, w):
    """
    X : numpy array of shape (`n_observations`, `n_features`)
    Y : numpy array of shape (`n_observations`, `target_dimensionality`) or (`n_observations`,)
    w : numpy array of shape (`n_features`, `target_dimensionality`) or (`n_features`,)

    Return : numpy array of same shape as `w`

    Computes the MAE derivative for linear regression (X.dot(w)) with no bias term
    w.r.t. w weight matrix.

    Please mention, that in case `target_dimensionality` > 1 the error is averaged along this
    dimension as well, so you need to consider that fact in derivative implementation.
    """

    # YOUR CODE HERE
    return
```

```
@staticmethod
def l2_reg_derivative(w):
    """
    w : numpy array of shape (`n_features`, `target_dimensionality`) or (`n_features`,)

    Return : numpy array of same shape as `w`

    Computes the L2 regularization term derivative w.r.t. the weight matrix w.
```

```
"""
```

```
# YOUR CODE HERE
```

```
return
```

```
@staticmethod
```

```
def l1_reg_derivative(w):
```

```
"""
```

```
Y : numpy array of shape (`n_observations`, `target_dimensionality`) or (`n_observations`,)
```

```
w : numpy array of shape (`n_features`, `target_dimensionality`) or (`n_features`,)
```

```
Return : numpy array of same shape as `w`
```

```
Computes the L1 regularization term derivative w.r.t. the weight matrix w.
```

```
"""
```

```
# YOUR CODE HERE
```

```
return
```

```
@staticmethod
```

```
def no_reg_derivative(w):
```

```
"""
```

```
Simply ignores the derivative
```

```
"""
```

```
return np.zeros_like(w)
```

Обращаем ваше внимание, требуется реализовать решение в векторном виде (т.е. для каждого объекта предсказание  $\hat{y}$  является вектором с размерностью  $\geq 1$ . При подсчете ошибки она усредняется как по объектам, так и по размерности  $y$ .

Например, для вектора ошибок на одном объекте  $[1., 1., 1., 1.]$  значение функции ошибки будет равно  $\frac{1}{4} (1.+1.+1.+1.)$

Для вашего удобства метод `.mse` уже реализован и вы можете обращаться к нему за примером.

Для проверки своего кода вам доступно несколько assert'ов:

```
w = np.array([1., 1.])
```

```
x_n, y_n = feature_matrix, targets
```

```
# Repeating data to make everything multi-dimensional
```

```
w = np.vstack([w[None, :] + 0.27, w[None, :] + 0.22, w[None, :] + 0.45, w[None, :] + 0.1]).T
```

```
y_n = np.hstack([y_n[:, None], 2*y_n[:, None], 3*y_n[:, None], 4*y_n[:, None]])
```

```
reference_mse_derivative = np.array([
```

```
    [ 7.32890068, 12.88731311, 18.82128365, 23.97731238],
```

```
    [ 9.55674399, 17.05397661, 24.98807528, 32.01723714]
```

```
])
```

```
reference_l2_reg_derivative = np.array([
```

```
    [2.54, 2.44, 2.9, 2.2 ],
```

```
    [2.54, 2.44, 2.9, 2.2 ]
```

```
])
```

```
assert np.allclose(
```

```
    reference_mse_derivative,
```

```
    LossAndDerivatives.mse_derivative(x_n, y_n, w), rtol=1e-3
```

```
), 'Something wrong with MSE derivative'
```

```
assert np.allclose(
```

```

reference_l2_reg_derivative,
LossAndDerivatives.l2_reg_derivative(w), rtol=1e-3
), 'Something wrong with L2 reg derivative'

print(
'MSE derivative:\n{} \n\nL2 reg derivative:\n{}'.format(
    LossAndDerivatives.mse_derivative(x_n, y_n, w),
    LossAndDerivatives.l2_reg_derivative(w))
)

reference_mae_derivative = np.array([
    [0.19708867, 0.19621798, 0.19621798, 0.19572906],
    [0.25574138, 0.25524507, 0.25524507, 0.25406404]
])
reference_l1_reg_derivative = np.array([
    [1., 1., 1., 1.],
    [1., 1., 1., 1.]
])

assert np.allclose(
    reference_mae_derivative,
    LossAndDerivatives.mae_derivative(x_n, y_n, w), rtol=1e-3
), 'Something wrong with MAE derivative'

assert np.allclose(
    reference_l1_reg_derivative,
    LossAndDerivatives.l1_reg_derivative(w), rtol=1e-3
), 'Something wrong with L1 reg derivative'

print(
'MAE derivative:\n{} \n\nL1 reg derivative:\n{}'.format(
    LossAndDerivatives.mae_derivative(x_n, y_n, w),
    LossAndDerivatives.l1_reg_derivative(w))
)

```

Градиентный спуск для решения реальной задачи

Следующая функция позволяет найти оптимальные значения параметров с помощью градиентного спуска:

```

def get_w_by_grad(X, Y, w_0, loss_mode='mse', reg_mode=None, lr=0.05, n_steps=100,
reg_coeff=0.05):
    if loss_mode == 'mse':
        loss_function = LossAndDerivatives.mse
        loss_derivative = LossAndDerivatives.mse_derivative
    elif loss_mode == 'mae':
        loss_function = LossAndDerivatives.mae
        loss_derivative = LossAndDerivatives.mae_derivative
    else:
        raise ValueError('Unknown loss function. Available loss functions: `mse`, `mae`')

    if reg_mode is None:
        reg_function = LossAndDerivatives.no_reg
        reg_derivative = LossAndDerivatives.no_reg_derivative # lambda w: np.zeros_like(w)
    elif reg_mode == 'l2':
        reg_function = LossAndDerivatives.l2_reg

```

```

    reg_derivative = LossAndDerivatives.l2_reg_derivative
elif reg_mode == 'l1':
    reg_function = LossAndDerivatives.l1_reg
    reg_derivative = LossAndDerivatives.l1_reg_derivative
else:
    raise ValueError('Unknown regularization mode. Available modes: `l1`, `l2`, None')

w = w_0.copy()

for i in range(n_steps):
    empirical_risk = loss_function(X, Y, w) + reg_coeff * reg_function(w)
    gradient = loss_derivative(X, Y, w) + reg_coeff * reg_derivative(w)
    gradient_norm = np.linalg.norm(gradient)
    if gradient_norm > 5.:
        gradient = gradient / gradient_norm * 5.
    w -= lr * gradient

    if i % 25 == 0:
        print('Step={}, loss={}, \ngradient values={}\n'.format(i, empirical_risk, gradient))
return w

```

Рассмотрим простой пример:

```

# Initial weight matrix
w = np.ones((2,1), dtype=float)
y_n = targets[:, None]
w_grad = get_w_by_grad(x_n, y_n, w, loss_mode='mse', reg_mode='l2', n_steps=250)

```

Сравнение с sklearn

Сравним реализованную модель с версией из sklearn:

```

from sklearn.linear_model import Ridge
lr = Ridge(alpha=0.05)
lr.fit(x_n, y_n)
print('sklearn linear regression implementation delivers MSE = {}'.format(np.mean((lr.predict(x_n) -
y_n)**2)))
plt.scatter(x_n[:, -1], y_n[:, -1])
plt.scatter(x_n[:, -1], x_n.dot(w_grad)[:, -1], color='orange', label='Handwritten linear regression',
linewidth=5)
plt.scatter(x_n[:, -1], lr.predict(x_n), color='cyan', label='sklearn Ridge')
plt.legend()
plt.show()

```

Сдача задания

Сдайте в чекер реализованный класс LossAndDerivatives. Для этого можете скопировать всю ячейку с кодом (в том числе и импортирование numpy) в файл derivatives.py.

На этом задание завершено. Поздравляем!

#### 4. Перечень типовых (примерных) вопросов и тем для проведения промежуточной аттестации обучающихся

Вопрос 1.1. Продолжите предложение.

Предположение i.i.d(независимые одинаково распределенные) обычно применяется к...  
наблюдениям

признакам

значениям целевой переменной



Вопрос 1.2. Продолжите предложение.

В задаче обучения с учителем (supervised) "учителем" выступает(ют)...

Эксперт, который настраивает модели

Ваш научный руководитель

Данные с некоторой разметкой (например, исторические данные или размеченные ассессорами)

Вопрос 1.3. Выберите верный ответ/ответы.

Какие из следующих задач являются задачами обучения с учителем?

Регрессия

Классификация

Уменьшение размерности

Биометрическая идентификация (например, FaceID)

Вопрос 1.4. Представьте, что вы используете kNN для задачи регрессии, и все объекты имеют целевые значения, равные 1 и -1. Может ли какой-либо тестовый пример получить прогнозируемое целевое значение -2?

Да

Нет

Затрудняюсь ответить

Вопрос 1.5. Выберите все правильные названия представления табличных данных без значений целевой переменной:

Матрица плана

Матрица объект-признак

Датасет

Матрица целевых переменных

Вопрос 1.6. Продолжите предложение.

В задаче классификации...

Целевая переменная принимает только целочисленные значения

Целевая переменная является вектором из действительных чисел

Целевая переменная принимает значение из конечного множества ответов

Вопрос 1.7. Продолжите предложение (возможны несколько вариантов).

Число соседей в методе kNN является...

Параметром

Гиперпараметром

Метрикой

Функцией потерь

Вопрос 1.8. Продолжите предложение (возможны несколько вариантов).

Метрика в kNN задает...

Число соседей, до которых считается расстояние

Способ подсчета расстояний между объектами

Функцию потерь

Вопрос 1.9. Выберите верный ответ/ответы.

Какая из предложенных функций может использоваться в качестве метрики?

Вопрос 1.10. Среди признаков, в которых описывается объект, есть цены в рублях, и расстояние в километрах. Затем цены (признаки) перевели в евро. Изменятся ли предсказания алгоритма kNN?

Да

Нет

Вопрос 1.11. Выберите верный ответ/ответы.

Результаты обучения каких моделей не изменятся (игнорируя машинную точность) при домножении всех значений одного из признака на 2?

Линейная регрессия без регуляризации

SVM

Решающее дерево

Градиентный бустинг над решающими деревьями

Логистическая регрессия с L2 регуляризацией

#### Критерии оценивания

Форма аттестации предусматривает зачет в форме тестирования.

Оценка «зачтено» выставляется студенту, если он показал всесторонние, систематизированные, глубокие знания учебной программы дисциплины и умение уверенно применять их на практике при решении конкретных задач, свободное и правильное обоснование принятых решений.

Оценка «не зачтено» выставляется студенту, который не знает большей части основного содержания учебной программы дисциплины, допускает грубые ошибки в формулировках основных понятий дисциплины и не умеет использовать полученные знания при решении типовых практических задач.

Максимальная сумма, которую можно набрать, успешно выполнив все контрольные мероприятия, составляет 100 баллов. Для получения положительной оценки «зачтено» необходимо набрать не менее 30 баллов.

#### **5. Методические материалы, определяющие процедуры оценивания знаний, умений, навыков и (или) опыта деятельности**

Во время проведения зачета обучающиеся могут пользоваться программой дисциплины.