МИНИСТЕРСТВО НАУКИ И ВЫСШЕГО ОБРАЗОВАНИЯ РФ
ВОЛГОГРАДСКИЙ ГОСУДАРСТВЕННЫЙ ТЕХНИЧЕСКИЙ УНИВЕРСИТЕТ

Т.А. Яновский, Д.А. Астахов

# Машинное обучение и нейросетевые модели

*Учебно-методическое пособие*


Волгоград
2021

УДК 004.8. Искусственный интеллект

Р е ц е н з е н т ы :

кафедра информационных систем и математического моделирования Волгоградского института управления – филиала Российской академии народного хозяйства и государственной службы при Президенте РФ, зав. кафедрой, канд. техн. наук, доцент *О. А. Астафурова*;

профессор кафедры методики преподавания математики и физики, ИКТ Волгоградского государственного социально-педагогического университета, д-р пед. наук *Т. М. Петрова*

Печатается по решению редакционно-издательского совета Волгоградского государственного технического университета

**Яновский, Т. А.**

Машинное обучение и нейросетевые модели: учебно-методическое пособие / Т.А. Яновский, Д.А. Астахов; ВолгГТУ. – Волгоград, 2021. – 109 с.

На правах рукописи.

В учебном пособии представлены основные положения теории машинного обучения и рассмотрены вопросы, связанные с разработкой моделей машинного обучения, включая нейросетевые. Приведена классификация задач машинного обучения и примеры построения машинно обученных моделей. Пособие предназначено для студентов факультета электроники и вычислительной техники очной и заочной форм обучения.

На правах рукописи

ОГЛАВЛЕНИЕ

# ВВЕДЕНИЕ

Машинное обучение – это автоматизированный процесс, который извлекает шаблоны из данных. При создании моделей для приложений аналитического прогнозирования используется обучение с учителем. При этом методы машинного обучения с учителем автоматически строят модель взаимосвязей между набором описательных признаков и целевым признаком на основе набора статистических примеров, или прецедентов. Затем эту модель можно использовать для прогнозирования.

Алгоритмы машинного обучения работают путем поиска в наборе возможных моделей прогнозирования такой модели, которая наилучшим образом отражает взаимосвязь между описательными признаками и целевым признаком в обучающей выборке. Также значительное место в практической деятельности аналитика занимают задачи без учителя – в них отсутствует размеченные записи и на первый план выходят проблемы анализа сходств и различий.

Современный специалист в области анализа данных должен сформировать навыки корректной постановки задач машинного обучения, уметь сопоставлять задаче наиболее адекватные модели машинного обучения, выполнять настройку их параметров и оценивать качество. Также необходимы навыки предваряющего эти действия разведочного анализа данных.

Особое место среди методов машинного обучения занимают искусственные нейронные сети – основанная на аналогиях функционирования головного мозга технология моделирования, нашедшая особенно широкое применение в задачах искусственного интеллекта с неструктурированными данными (видео и текст).

# 1. Курс лекций по дисциплине

## 1.1 Машинное обучение

### 1.1.1 Основы машинного обучения.

Введение в задачи, методы и инструменты машинного обучения.

Базовые статистические инструменты анализа данных.

Методы математической статистики являются базовым инструментарием академического исследователя.

Культура работы с данными, проверки гипотез, необходима в рамках разведочного анализа данных, построения конвейера предобработки данных.

### 1.1.2 Разведочный анализ данных.

Протокол разведочного анализа данных.

Распределения данных, статистические эксперименты и проверка значимости.

### 1.1.3 Регрессия в машинном обучении.

Простая и множественная линейная регрессия, нелинейная регрессия.

Описывается концепция регрессионного моделирования, способы подготовки данных, подгонки и валидации моделей.

### 1.1.4 Введение в задачи классификации.

Наивный байесовский алгоритм, дискриминантный анализ и логистическая регрессия, метод главных компонент.

Описываются ключевые подходы и методы решения задачи классификации, классические задачи классификации – спам, кредитный скоринг.

### 1.1.5 Качество классификации.

Оценивание моделей классификации и стратегии в отношении несбалансированных данных.

Статистическое машинное обучение.

Деревья решений в задаче классификации.

### 1.1.6 Информационно-центричные методы классификации.

Метод k ближайших соседей, древовидные модели, бэггинг и случайный лес, бустинговый подход

### 1.1.7 Ансамблевый подход.

Описывается сущность ансамблевого подхода, его сильные и слабые стороны.

## 1.2 Искусственные нейронные сети

### 1.2.1 Основы программирования нейронных сетей.

Введение в тематику искусственных нейронных сетей. Модель искусственного нейрона. Общее представление об искусственной нейронной сети. Библиотеки для обучения нейронных сетей. Распознавание предметов одежды. Обзор набора данных и выбор архитектуры нейронной сети. Распознавание животных. Построение архитектуры нейронной сети и ее обучение. Анализ качества обучения нейронной сети

### 1.2.2 Обучение искусственной нейронной сети.

Обучение искусственного нейрона. Обучение искусственной нейронной сети. Метод обратного распространения ошибки.

### 1.2.3 Нейронные сети для анализа табличных данных.

Применение нейронных сетей для решения задачи регрессии

### 1.2.4 Нейронные сети для задачи анализа изображений.

Сверточные нейронные сети. Распознавание объектов на изображении. Предобученные нейронные сети. Перенос обучения в нейронных сетях

## 1.3 Обучение без учителя

Анализ главных компонент, кластеризация на основе k средних и иерархическая. Модельно-ориентированная кластеризация

## 1.4 Вопросы и задания

1. Базовые статистические инструменты анализа данных:
2. Протокол разведочного анализа данных.
3. Выявление выбросов
4. Проверка однородности групповых дисперсий
5. Проверка на нормальность распределений
6. Выявление коллинеарности и формы связи между переменными
7. Влияние пространственно-временных факторов на анализируемую переменную.
8. Прямоугольные данные
9. Оценки центрального положения и вариабельности
10. Распределение данных и анализ двоичных и категориальных данных
11. Корреляция
12. Случайный отбор и смещенная выборка
13. Выборочное распределение статистики
14. Бутстрап
15. Доверительные интервалы распределения
16. Основные теоретические распределения, их применимость и свойства
17. A/B тестирование
18. Проверка статистических гипотез и перестановочные тесты
19. Статистическая значимость и p-значения
20. Мощность и размер выборки
21. Простая линейная регрессия

22. Множественная линейная регрессия
23. Факторные переменные в регрессии
24. Нелинейная регрессия
25. Наивный байесовский алгоритм классификации
26. Дискриминантный анализ
27. Логистическая регрессия
28. Оценивание моделей классификации
29. Стратегии в случае несбалансированных данных
30. Метод к ближайших соседей
31. Древовидные модели
32. Бэггинг и случайный лес
33. Бустинг
34. Анализ главных компонент
35. Кластеризация на основе к средних
36. Иерархическая кластеризация
37. Модельно-ориентированная кластеризация
38. Шкалирование и категориальные переменные
39. Искусственные нейронные сети
40. Концепция и технология глубокого обучения
41. Обучение с подкреплением

## 2. Методические указания к курсовой работе

### 2.1. Задание на курсовую работу и методические указания к ее выполнению

Целью обучения студентов является формирование у них знаний и навыков выполнения проектов в области анализа данных и моделирования с использованием моделей машинного обучения, включая разработку искусственных нейронных сетей.

#### 2.1.1 Цели и задачи курсового проекта

Курсовая работа является самостоятельной работой студента, завершающей изучение курса «Машинное обучение и нейросетевые модели». Цель курсовой работы – научить студента применять теоретические знания для решения практических задач в составе проекта анализа данных. Курсовая работа включает программные решения задач математико-статистического моделирования и машинного обучения моделей на языке R/Python.

#### 2.1.2 Задания на курсовой проект

Выбор темы курсовой работы осуществляется в соответствии с перечнем заданий и утверждается преподавателем.

Курсовая работа состоит из 5 разделов/заданий по разделам.

1. изучение предметной области и постановка задачи (регрессии, классификации, кластеризации);
2. сбор и обработка данных;
3. разведочный анализ данных (качественная визуализация данных и осмысление графиков, вычисление описательных статистик, и проч.);
4. построение статистических моделей и алгоритмов их машинного обучения, включая разработку искусственных нейронных сетей;

5. интерпретация и использование модели машинного обучения (концепция продукта данных или принятие решения на основе обобщения данных)

В результате формируется пояснительная записка, содержащая результаты выполнения по разделам. Курсовая работа оформляется в соответствии с шаблоном и оценивается в процессе защиты работы. Оформляется задание на курсовую работу и пояснительная записка по шаблонам, приведённым на сайте ВолгГТУ http://umu.vstu.ru/umu-docs/uo/forms.

Объем пояснительной записки курсовой работы – 30-40 страниц, включая инфографику.

Работа оформляется с учётом требований ГОСТ 2.105-79 и ГОСТ 7-32-81.

Рукописный текст представляется на одной стороне листа писчей бумаги формата А4. Размеры полей: левого – 35 мм; правого – 10 мм; верхнего и нижнего – 20 мм. Он должен иметь сквозную нумерацию страниц. Буквенные обозначения должны быть расшифрованы, указаны единицы измерения используемых в процессе вычисления величин.

Каждый раздел оценивается максимум на 5 баллов. По сумме балов оценка работы может достигать 25.

### 2.1.3 Темы курсовых работ

1.Прогнозирование возврата персональных ссуд в инвестиционно-кредитной компании Lending Club (датафрейм loan200 из пакета FNN языка R)

2.Задача кредитного скоринга по данным о 1000 клиентов одного из немецких банков (датафрейм GermanCredit из пакета caret языка R)

3.Оценивание стоимости домов на основе данных о жилом округе Кинг (Сэтл, шт. Вашингтон) (датафрейм data.frame house в базовом оснащении языка R)

4.Задача об обнаружении спама по коллекции данных Центра машинного обучения и интеллектуальных систем Калифорнийского университета http://archive.ics.uci.edu/ml/

5.Использование результатов гидробиологических исследований обилия водорослей в различных реках (датафрейм algae из пакета DMwR языка R)

6. Изучение семейного статуса 2000 швейцарцев на протяжении 16 лет их жизни между 15 и 30 годами (датафрейм biofam из пакета TraMineR языка R)

7. Определение способа изготовления стекла по его химическому составу (датафрейм fgl из пакета MASS языка R)

8.Определение возраста морского ушка на данных о 4177 особях этого вида. Данные берутся из коллекции данных Центра машинного обучения и интеллектуальных систем Калифорнийского университета http://archive.ics.uci.edu/ml/machine-learning-databases/abalone/abalone.data

Геоботаника:

9.Анализ зависимости между показателями химического состава и обилием 44 видов травянистых растений на 24 пробных площадках (датафреймы varechem и varecpec из пакета vegan языка R)

10.Задача анализа и моделирования данных в различных предметных областях. На данных, найденных студентами и одобренных преподавателем.

11. Свободная тема, связанная с инициативным исследованием анонимных данных студентов. Предполагает первичную постановку задачи (например, изучить влияние занятий спортом на успеваемость в учебе) включая набор исследовательских гипотез, самостоятельное определение предикторов и отклика, формирование анкеты для опроса студентов и сбора прямых или косвенных данных для оценки предикторов и отклика, анкетирование (анонимное), обработка данных и формирование датафрейма,

разведочный анализ данных, использование датафрейма для обучения и тестирования моделей, интерпретация результатов разведочного анализа данных и полученных моделей в контексте проверяемых гипотез.

Выбор этой темы является предпочтительным в связи с необходимостью реализации всего конвейера работ исследовательского проекта данных.

Все темы подразумевают широкую вариабельность поставленных и решаемых задач. В процессе выполнения курсовой работы студентам необходимо продемонстрировать различные подходы к решению поставленных задач на основе широкого класса методов и моделей машинного обучения. Особенно высоко оценивается своеобразие постановки задачи и оригинальность ее решения, состоящее в уникальной последовательности операций преобразования и моделирования данных

## 2.2. Пример выполнения курсового проекта

**Минобрнауки России**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Волгоградский государственный технический университет»**

Факультет «Электроники и вычислительной техники»_____

Направление (специальность) _____

Кафедра «Системы автоматизированного проектирования и поискового конструирования»

Дисциплина_____

Утверждаю

Зав. кафедрой Щербаков М.В.

«_____» _____20 ___ г.

**Задание**

**на курсовую работу**

Студенты

1. Тема: Анализ данных отражающих рейтинг самых счастливых стран

Утверждена приказом от «_____» _____ 20___ г. № _____

2. Срок представления работы (проекта) к защите «26» июня 2020 г.

3. Содержание расчетно-пояснительной записки: _Подготовка данных для исследования, разведочный анализ, регрессионный анализ, кластеризация и дерево решений, создание и обучение RNN модели_____

4. Перечень графического материала: ___30 рисунков_____

5. Дата выдачи задания «  » _____2021 г.

Руководитель работы (проекта)_____ __Яновский Т.А._

подпись, дата инициалы и фамилия

Задание принял к исполнению_____

подпись, дата инициалы и фамилия

Задание принял к исполнению_____

подпись, дата инициалы и фамилия

Задание принял к исполнению_____

подпись, дата инициалы и фамилия

**Минобрнауки России**

**Федеральное государственное бюджетное образовательное учреждение высшего образования**

**«Волгоградский государственный технический университет»**

Факультет «Электроники и вычислительной техники»_____

Кафедра «Системы автоматизированного проектирования и поискового конструирования»

# ПОЯСНИТЕЛЬНАЯ ЗАПИСКА

## к курсовой работе

по дисциплине _____

на тему Анализ данных, отражающих рейтинг самых счастливых стран

Студенты_____

Руководитель работы (проекта) _____ _Яновский Т.А._

(подпись и дата подписания) (инициалы и фамилия)

Члены комиссии:

_____ _____

(подпись и дата подписания) (инициалы и фамилия)

_____ _____

(подпись и дата подписания) (инициалы и фамилия)

_____ _____

(подпись и дата подписания) (инициалы и фамилия)

Нормоконтролер _____

_____

(подпись, дата подписания) (инициалы и фамилия)

Волгоград 2021 г.

Содержание

…

## Введение

Достижение благосостояния и возможность построения счастливой жизни волнуют человечество на протяжении всей его истории.

При этом одним из главных факторов "счастья" отдельного человека является возможность удовлетворить свои потребности. Материальное положение человека является инструментом удовлетворения этих самых потребностей и непосредственно влияет на уровень счастья человека. Так как для того, чтобы уровень счастья был высоким, нужно, чтобы большее количество население имело достаточный уровень дохода, что достигается путём развития экономической сферы, а именно: увеличение ВВП, снижение уровня коррупции, увеличение продолжительности жизни, свобода выбора. Мы решили разобраться в этой теме и провели исследование.

## Подготовка данных для исследования

Для того, чтобы изучить и проанализировать данную тему, мы использовали пять датасетов за 2015-2019 года, которые были представлены в формате .csv.

```
df2015 = pd.read_csv("2015.csv")
df2016 = pd.read_csv("2016.csv")
df2017 = pd.read_csv("2017.csv")
df2018 = pd.read_csv("2018.csv")
df2019 = pd.read_csv("2019.csv")
```

Прежде, чем начать исследование, необходимо было подготовить данные для дальнейшей работы с ними. Для этого были выполнены следующие действия/операции:

### Удаление ненужных столбцов в датасетах

```
df2015_figures = df2015.drop(['Country','Region','Standard Error','Dystopia Residual','Happiness Rank'], axis=1)
df2016_figures = df2016.drop(['Country','Region','Lower Confidence Interval','Upper Confidence Interval','Dystopia Residual','Happiness Rank'], axis=1)
df2017_figures = df2017.drop(['Country','Happiness.Rank','Whisker.high','Whisker.low','Dystopia.Residual'], axis=1)
df2018_figures = df2018.drop(['Country or region', 'Overall rank'], axis=1)
df2019_figures = df2019.drop(['Country or region', 'Overall rank'], axis=1)
```

### Приведение названия столбцов к одному виду

```
df2015_figures = df2015_figures[['Happiness Score','Economy (GDP per Capita)', 'Family', 'Health (Life Expectancy)', 'Freedom', 'Generosity', 'Trust (Government Corruption)']]
df2016_figures = df2016_figures[['Happiness Score','Economy (GDP per Capi-
```

```
ta)', 'Family', 'Health (Life Expectancy)', 'Freedom', 'Generosity', 'T
rust (Government Corruption)']]
   df2015_figures.rename(columns={'Happiness Score': 'Score', 'Economy
(GDP per Capita)': 'GDP per capita', 'Family': 'Social support', 'Healt
h (Life Expectancy)': 'Healthy life expectancy', 'Freedom': 'Freedom to
 make life choices','Trust (Government Corruption)': 'Perceptions of co
rruption'}, inplace=True)
   df2016_figures.rename(columns={'Happiness Score': 'Score', 'Economy
(GDP per Capita)': 'GDP per capita', 'Family': 'Social support', 'Healt
h (Life Expectancy)': 'Healthy life expectancy', 'Freedom': 'Freedom to
 make life choices','Trust (Government Corruption)': 'Perceptions of co
rruption'}, inplace=True)
   df2017_figures.rename(columns={'Happiness.Score': 'Score', 'Economy.
.GDP.per.Capita.': 'GDP per capita', 'Family': 'Social support', 'Healt
h..Life.Expectancy.': 'Healthy life expectancy', 'Freedom': 'Freedom to
 make life choices','Trust..Government.Corruption.': 'Perceptions of co
rruption'}, inplace=True)
```

Объединение всех датасетов

```
   df_all = df2019_figures.append(df2018_figures, ignore_index=True)
   df_all = df_all.append(df2017_figures, ignore_index = True)
   df_all = df_all.append(df2016_figures, ignore_index = True)
   df_all = df_all.append(df2015_figures, ignore_index = True)
```

Сортировка по столбцу «Score»

```
   df_all = df_all.sort_values(by=['Score'], ascending=False,ignore_ind
ex = True)
```

Далее на рисунке 1 представлен готовый для исследования датасет с данными за 2015-2019 года

| | Score | GDP per capita | Social support | Healthy life expectancy | Freedom to make life choices | Generosity | Perceptions of corruption |
|---|---|---|---|---|---|---|---|
| **0** | 7.769 | 1.340000 | 1.587000 | 0.986000 | 0.596000 | 0.153000 | 0.393000 |
| **1** | 7.632 | 1.305000 | 1.592000 | 0.874000 | 0.681000 | 0.202000 | 0.393000 |
| **2** | 7.600 | 1.383000 | 1.573000 | 0.996000 | 0.592000 | 0.252000 | 0.410000 |
| **3** | 7.594 | 1.456000 | 1.582000 | 0.861000 | 0.686000 | 0.286000 | 0.340000 |
| **4** | 7.587 | 1.396510 | 1.349510 | 0.941430 | 0.665570 | 0.296780 | 0.419780 |
| **...** | ... | ... | ... | ... | ... | ... | ... |
| **777** | 2.905 | 0.015300 | 0.415870 | 0.223960 | 0.118500 | 0.197270 | 0.100620 |
| **778** | 2.905 | 0.091623 | 0.629794 | 0.151611 | 0.059901 | 0.204435 | 0.084148 |
| **779** | 2.853 | 0.306000 | 0.575000 | 0.295000 | 0.010000 | 0.202000 | 0.091000 |
| **780** | 2.839 | 0.208680 | 0.139950 | 0.284430 | 0.364530 | 0.166810 | 0.107310 |
| **781** | 2.693 | 0.000000 | 0.000000 | 0.018773 | 0.270842 | 0.280876 | 0.056565 |

782 rows × 7 columns

Рисунок 1 – Данные 2015-2019 года

## Разведочный анализ

Разведочный анализ - это предварительный анализ данных с целью выявления наиболее общих зависимостей, закономерностей и тенденций, характера и свойств анализируемых данных, законов распределения анализируемых величин. Применяется для нахождения связей между переменными в ситуациях, когда отсутствуют (или недостаточны) априорные представления о природе этих связей.

Для проведении анализа были выполнены следующие действия:

Сравнение уровня счастья с 2015 года по 2019 год

Нашли средние значения Score за каждый год

Создали датасет с колонками «Year» и «value»

```
ScoreYears = pd.DataFrame([['2015', df2015_figures['Score'].mean()],
 ['2016', df2016_figures['Score'].mean()], ['2017', df2017_figures['Sco
re'].mean()],  ['2018', df2018_figures['Score'].mean()],  ['2019', df20
19_figures['Score'].mean()]], columns=['Years', 'value'])
ScoreYears.set_index('Years', inplace=True)
```

Созданный датасет:

```
Years    value
2015   5.375734
2016   5.382185
```

| | |
|---|---|
| 2017 | 5.354019 |
| 2018 | 5.375917 |
| 2019 | 5.407096 |

На рисунке 2 представлен график сравнения уровня счастья за 2015-2019 года.


Рисунок 2 – График уровней счастья за 2015-2019г

Как мы видим на данном графике, изменение среднего значения Score незначительно. В то же время мы можем сделать вывод, что уровень счастья в 2017-ом году был наименьшим, а в 2019-ом году достиг максимального значения.

Корреляционная матрица.

Одним из основных методов разведочного анализа является корреляционный анализ с целью поиска коэффициентов.

Корреляционная матрица — это квадратная (или прямоугольная) таблица, в которой на пересечении соответствующих строки и столбца находится коэффициент корреляции между соответствующими параметрами.

На рисунке 3 изображена корреляционная матрица исследуемых параметров

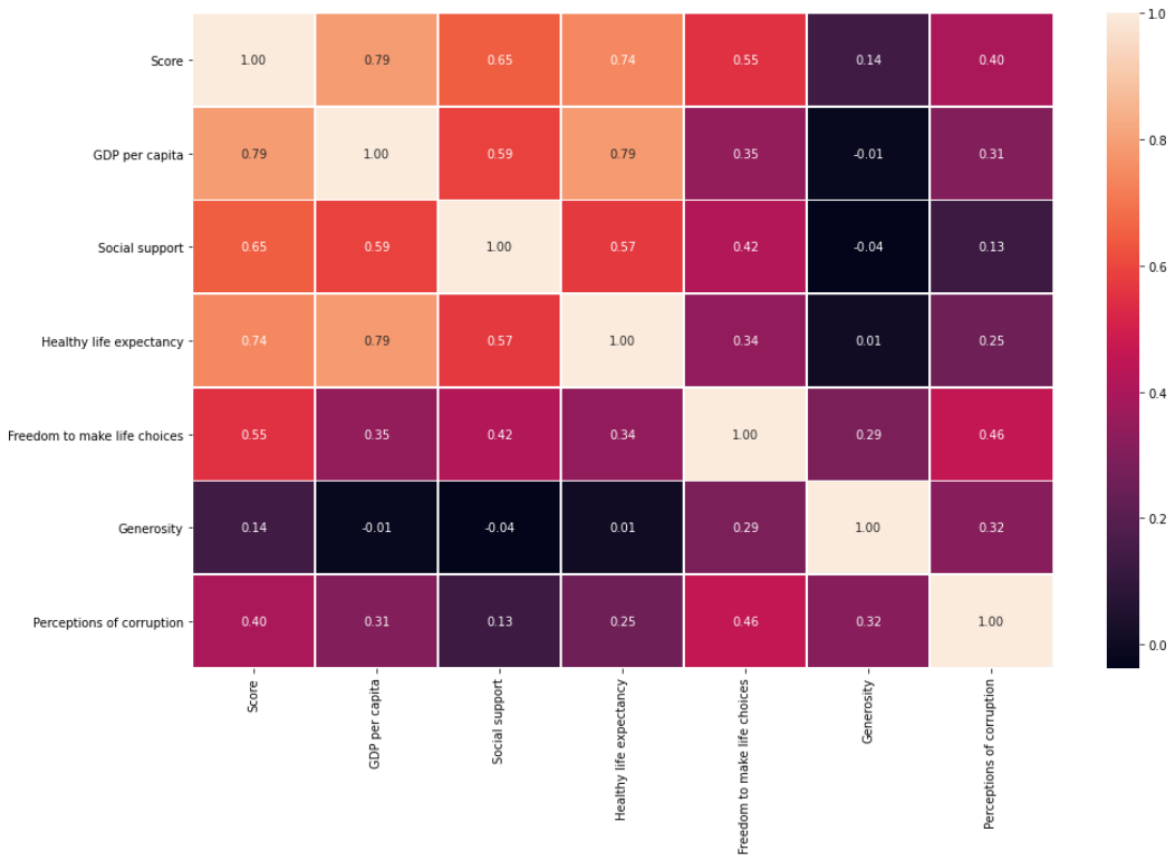



Рисунок 3 – Корреляционная матрица

Соотношение исследуемых параметров

Для того, чтобы проанализировать, как влияют некоторые критерии на число Score, мы построили графики соотношений.

График соотношения процента счастья к ВВП

(Score vs GDP per capita)

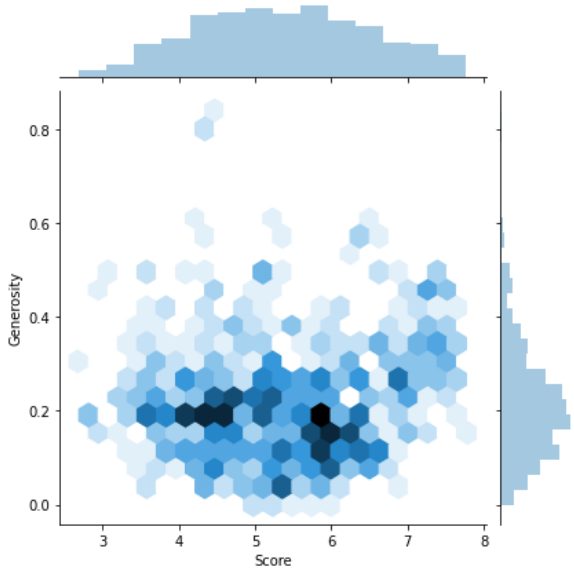
```
sns.jointplot(x="Score", y="GDP per capita", data=df_all,  kind="hex
")
```

График соотношения процента счастья к щедрости

(Score vs Generosity)

```
sns.jointplot(x="Score", y="Generosity", data=df_all,  kind="hex")
```

На рисунках 4.1 и 4.2 представлены график соотношений Score vs Generosity и Score vs GDP per capita



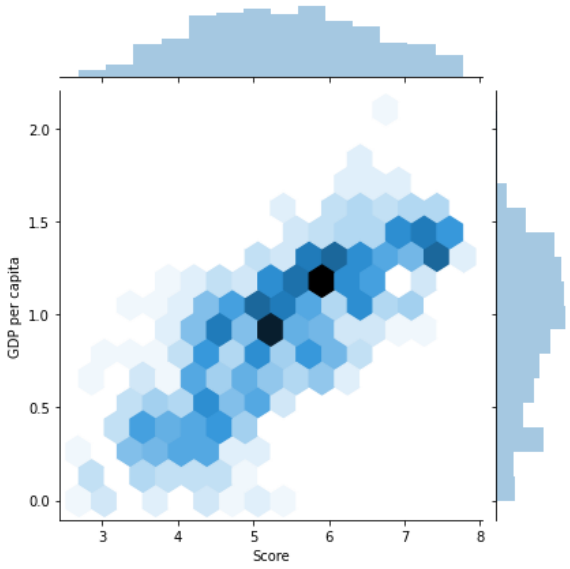




Рисунок 4.1 – График соотношения Score vs Generosity

Рисунок 4.2 – График соотношения Score vs GDP per capita

График соотношения процента счастья к социальной поддержке (Score vs Social support)

```
sns.jointplot(x="Score", y="Social support", data=df_all,  kind="hex
")
```

Гра-

фик соотношения процента счастья к продолжительности жизни (Score vs Healthy life expectancy)

```
sns.jointplot(x="Score", y="Healthy life expectancy", data=df_all, kind="hex")
```

На рисунках 4.3 и 4.4 представлены график соотношений Score vs Social support и Score vs Healthy life expectancy

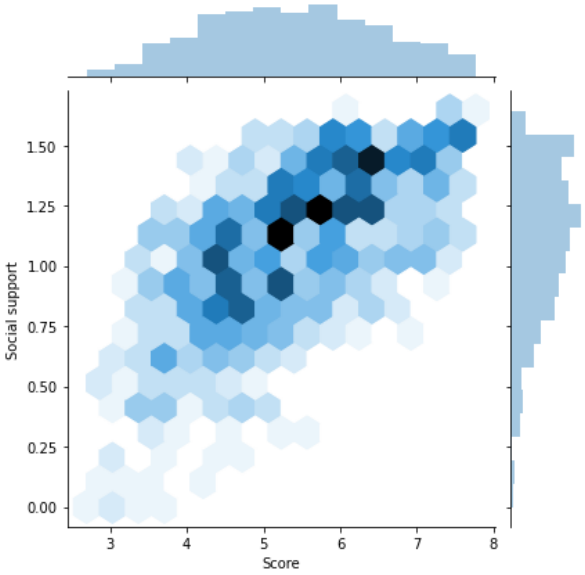



Рисунок 4.3 – График соотношения Score vs Social support

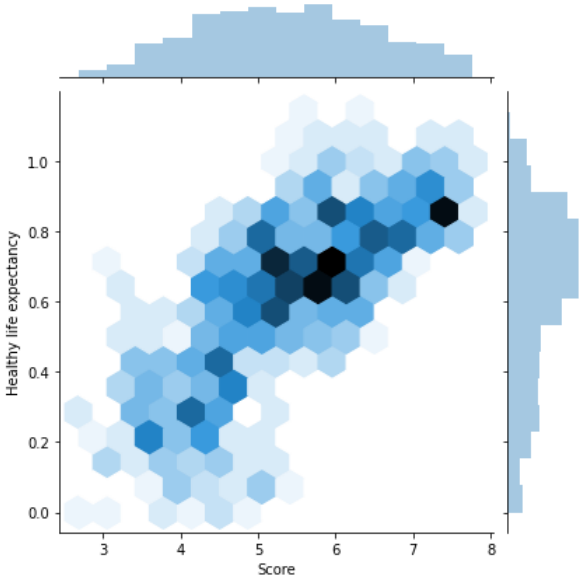



Рисунок 4.4 – График соотношения Score vs Healthy life expectancy

График соотношения процента счастья к продолжительности жизни (Score vs Freedom to make life choices)

```
sns.jointplot(x="Score", y="Freedom to make life choices", data=df_all,  kind="hex")
```

График соотношения процента счастья к продолжительности жизни (Score vs Perceptions of corruption)

```
sns.jointplot(x="Score", y="Perceptions of corruption", data=df_all, kind="hex")
```

На рисунках 4.5 и 4.6 представлены график соотношений Score vs Freedom to make life choices и Score vs Perceptions of corruption

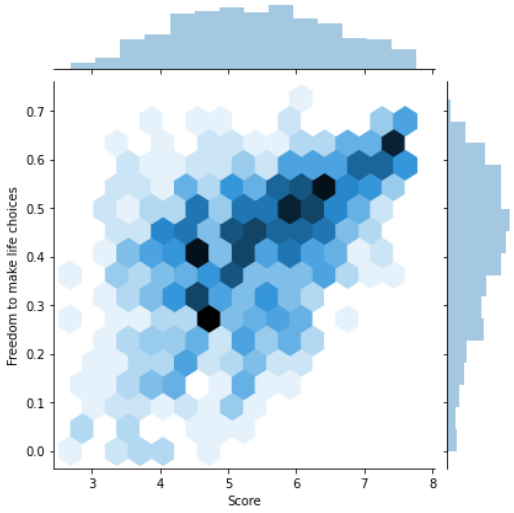



Рисунок 4.5 – График соотношения Score vs Freedom to make life choices

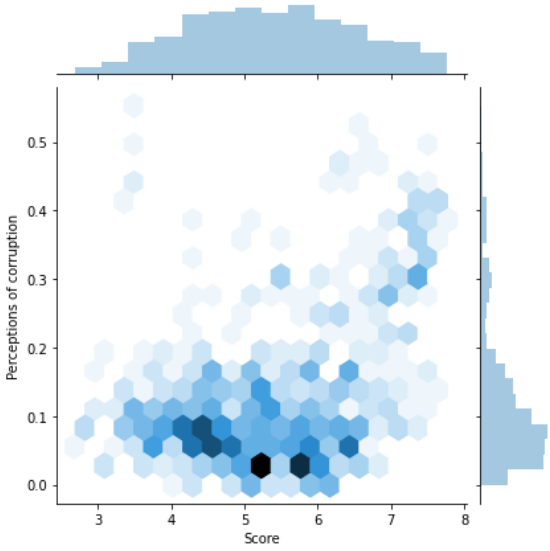



Рисунок 4.6 – График соотношения Score vs Perceptions of corruption

Выявление наиболее часто появляющихся значений Score

Для того, чтобы исследовать частоту появления значений Score, мы построили графики, которые показывают, какое значение «Score» чаще всего встречалось.

В 2015 году

```
plt.figure(figsize=(8,5))
plt.title("Уровень счастья 2015",fontsize=30)
sns.distplot(data15['Score'], bins=30, color = "g")
```

В 2016 году

```
plt.figure(figsize=(8,5))
plt.title("Уровень счастья 2016",fontsize=30)
sns.distplot(data16['Score'], bins=30, color = "b")
```

На рисунках 5.1 и 5.2 представлены графики уровней счастья с наиболее частыми значениями за 2015 и 2016 года

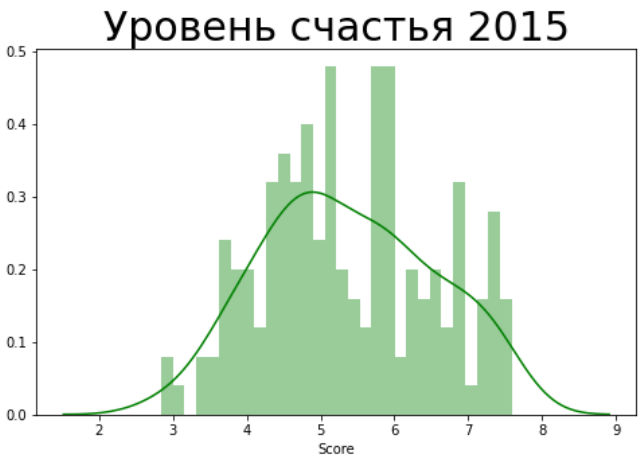



Рисунок 5.1 – График с наиболее частыми значениями Score за 2015 год

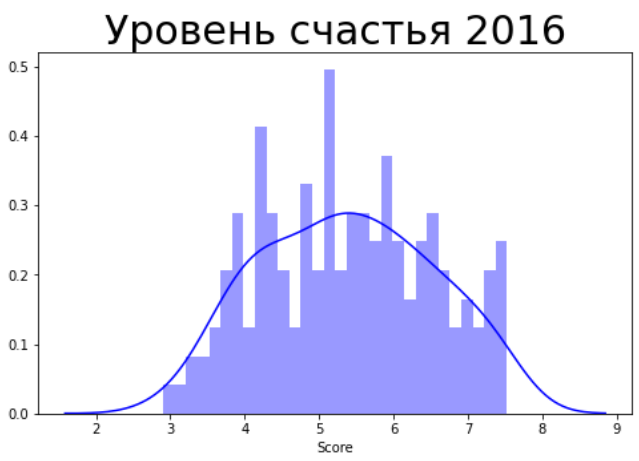



Рисунок 5.2 – График с наиболее частыми значения Score за 2016 год

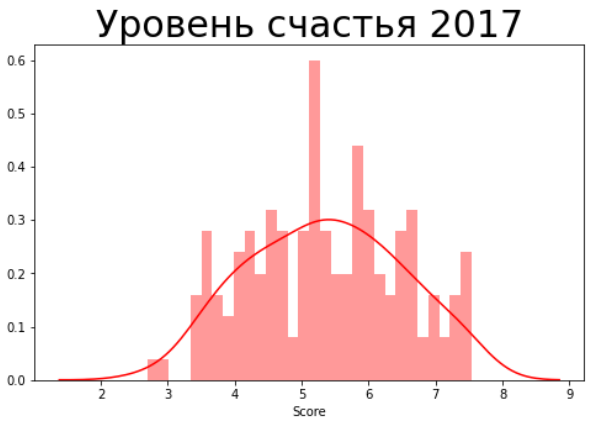
В 2017 году

```
plt.figure(figsize=(8,5))
plt.title("Уровень счастья 2017",fontsize=30)
sns.distplot(data17['Score'], bins=30, color = "r")
```

В 2018 году

```
plt.figure(figsize=(8,5))
plt.title("Уровень счастья 2018",fontsize=30)
sns.distplot(data18['Score'], bins=30, color = "y")
```

На рисунках 5.3 и 5.4 представлены графики уровней счастья с наиболее частыми значениями за 2017 и 2018 года



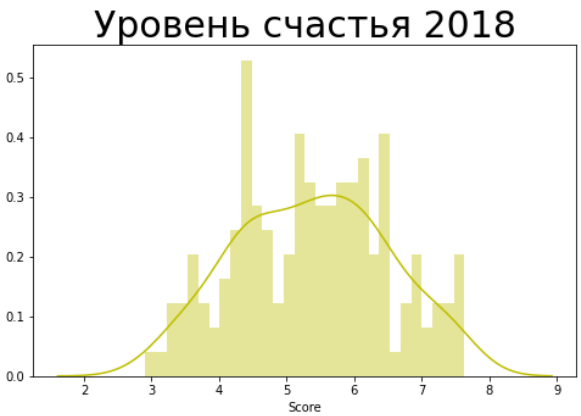



Рисунок 5.3 – График с наиболее частыми значениями Score за 2017 год

Рисунок 5.4 – График с наиболее частыми значения Score за 2018 год

В 2019 году

```
plt.figure(figsize=(8,5))
plt.title("Уровень счастья 2019",fontsize=30)
sns.distplot(data19['Score'], bins=30, color = "#00fff2")
```

На рисунках 5.5 представлен график уровня счастья с наиболее частыми значениями за 2019 год

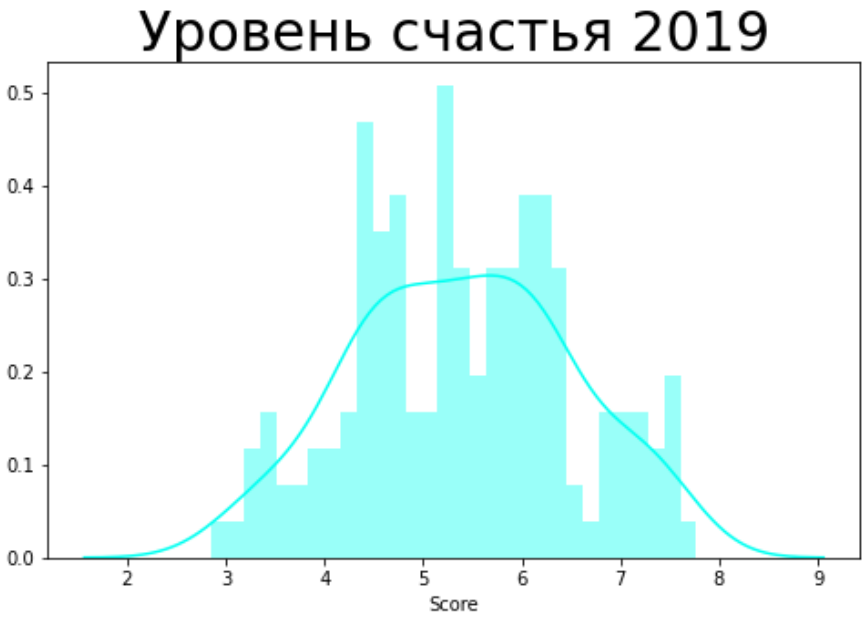



Рисунок 5.5 – График с наиболее частыми значениями Score за 2019 год

Диаграмма размаха

Для того чтобы более точно проанализировать как влияет ВВП на уровень счастья, был построена диаграмма размаха. Такой вид диаграммы в удобной форме показывает медиану, нижний и верхний квартили, минимальное и максимальное значение выборки и выбросы.

```
    data2 = df_all.copy()

    da-
ta2['Score_1'] = pd.cut(data2['Score'], [2, 4, 5, 6, 7, 10], labels=['2
-4_score','4-5_score', '5-6_score', '6-7_score', '7+_score'])
    plt.figure(figsize=(15,8))
    sns.boxplot(x="Score_1", y="GDP per capita", data=data2)
    plt.show()
```

На рисунке 6 представлена диаграмма соотношения уровня Score vs GDP per capita

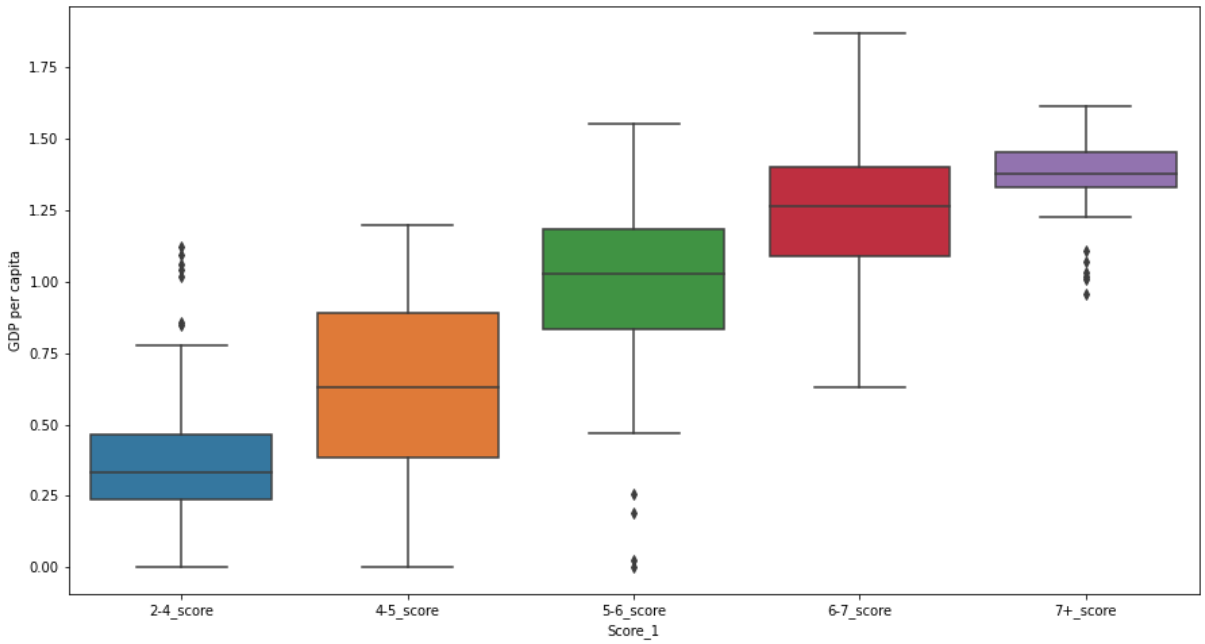



Рисунок 6 - Диаграмма соотношения уровня Score vs GDP per capita

## Регрессионный анализ

Регрессионный анализ - метод моделирования измеряемых данных и исследования их свойств. Данные состоят из пар значений зависимой переменной (переменной отклика) и независимой переменной (объясняющей переменной). Регрессионная модель есть функция независимой переменной и параметров с добавленной случайной переменной. Параметры модели настраиваются таким образом, что модель наилучшим образом приближает данные.

Для проведения регрессионного анализа были выполнены следующие действия:

Выявление величины с наибольшим влиянием на Score путем построения графиков линейной регрессии

Построение графиков Score vs GDP per capita и Score vs Social support

```
   plt.figure(figsize=(8,5))
sns.regplot(x="Score", y="GDP per capita", data=df_all)
   plt.figure(figsize=(8,5))
   sns.regplot(x="Score", y="Social support", data=df_all)
```

На рисунках 7.1 и 7.2 представлены графики линейной регрессии Score vs GDP per capita и Score vs Social support

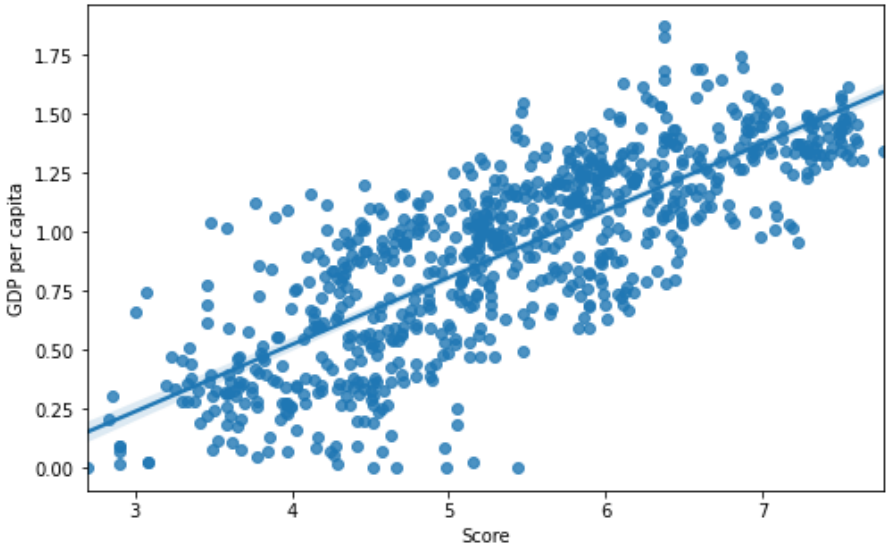



Рисунок 7.1 – График линейной регрессии Score vs GDP per capita

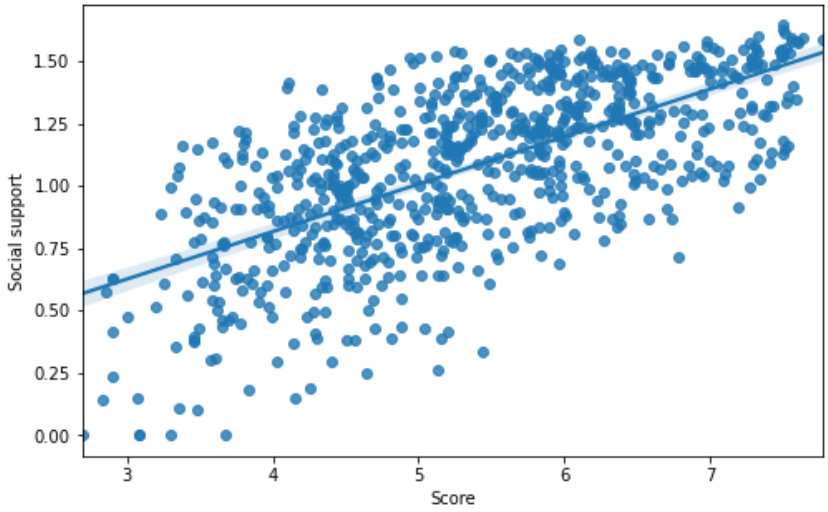



Рисунок 7.2 – График линейной регрессии Score vs Social support

Построение графиков Score vs Healthy life expectancy и Score vs Freedom to make life choices

```
    plt.figure(figsize=(8,5))
    sns.regplot(x="Score", y="Healthy life expectancy", data=df_all)
    plt.figure(figsize=(8,5))
    sns.regplot(x="Score", y="Freedom to make life choices", data=df_all
)
```

На рисунках 7.3 и 7.4 представлены графики линейной регрессии Score vs Healthy life expectancy и Score vs Freedom to make life choices

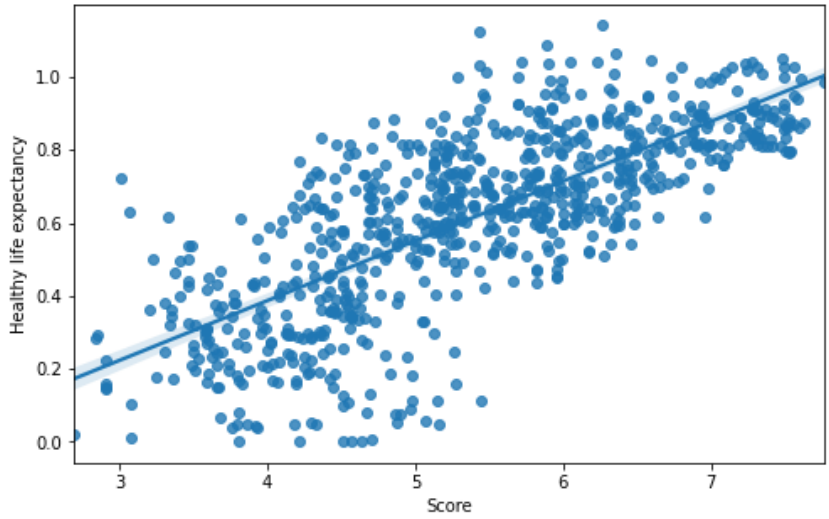



Рисунок 7.3 – График линейной регрессии Score vs Healthy life expectancy

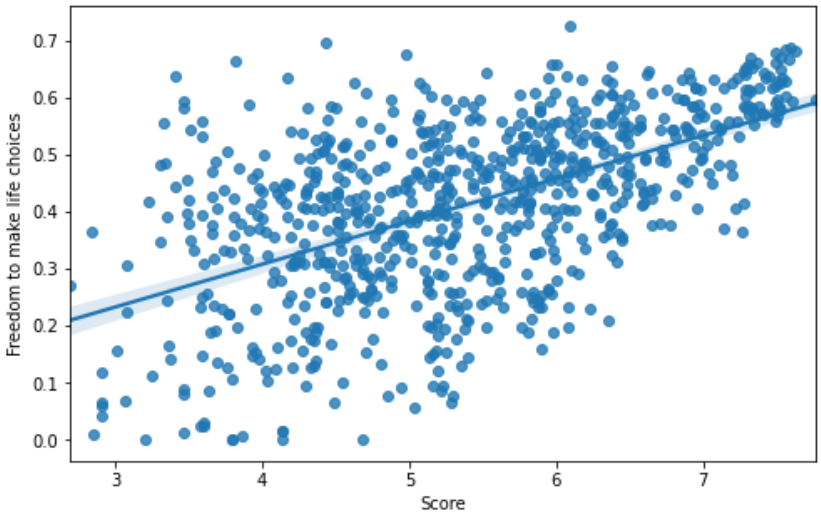



Рисунок 7.4 – График линейной регрессии Score vs Freedom to make life choices

Построение графиков Score vs Generosity и Score vs Perceptions of corruption

```
plt.figure(figsize=(8,5))
```

```
sns.regplot(x="Score", y="Generosity", data=df_all)
plt.figure(figsize=(8,5))
sns.regplot(x="Score", y="Perceptions of corruption", data=df_all)
```

На рисунках 7.5 и 7.6 представлены графики линейной регрессии Score vs Generosity и Score vs Perceptions of corruption

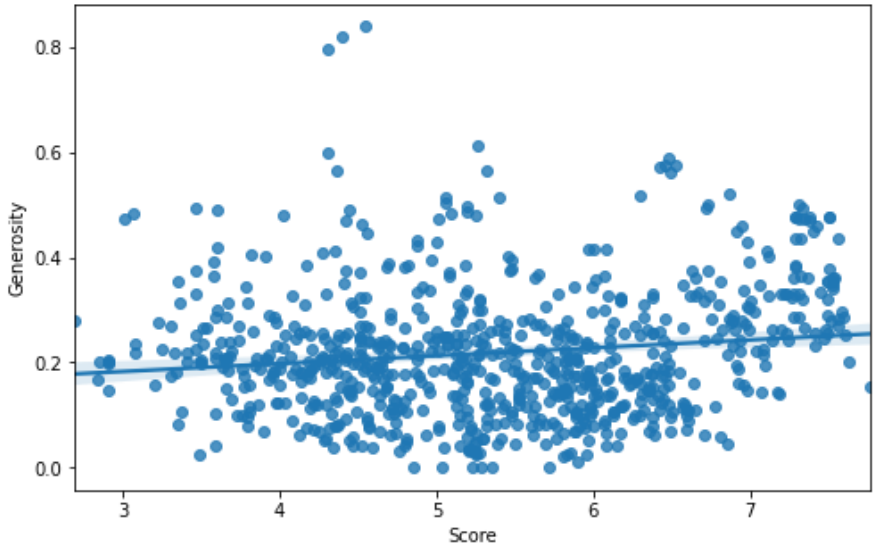



Рисунок 7.5 – График линейной регрессии Score vs Generosity

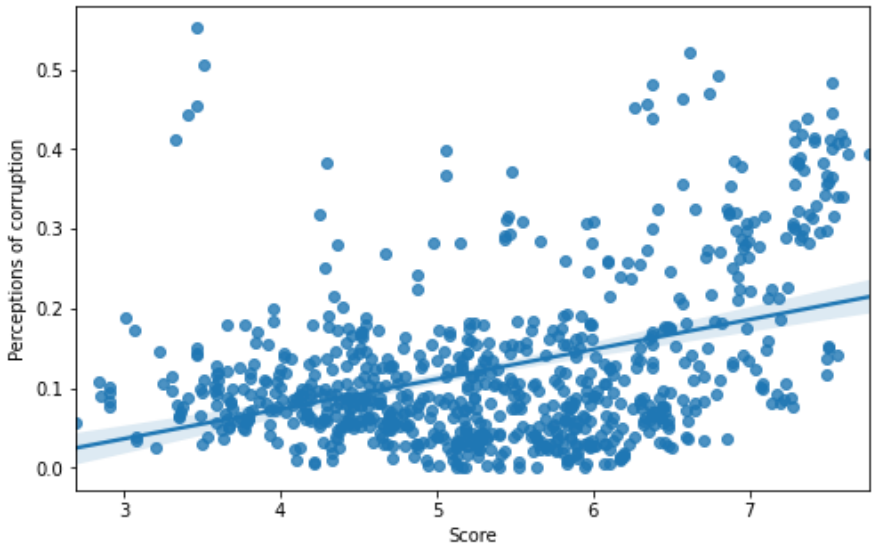



Рисунок 7.6 – График линейной регрессии Score vs Perceptions of corruption

Нахождение лучшей регрессионной модели, которая описывает массив данных

Для выявления лучшей регрессионной модели необходимо рассчитать коэффициент R^2 путем следующих действий:

Получим Y

```
trg = df_all[['Score']]
```

Получим зависимые значения

```
trn = df_all.drop(['Score'], axis=1)
```

Объявим все модели для дальнейшего их обучения

```
models = [LinearRegression(), # метод наименьших квадратов
            RandomForestRegres-
sor(n_estimators=100, max_features ='sqrt'), # случайный лес
            KNeighborsRegressor(n_neighbors=6), # метод ближайших со
седей
            SVR(kernel='linear'), # метод опорных векторов с линейны
м ядром
          ]
```

Разобьем данные на тестовую и обучающую выборки

```
Xtrain, Xtest, Ytrain, Ytest = train_test_split(trn, trg, test_size=
0.4)
```

Создаем временные структуры

```
TestModels = DataFrame()
tmp = {}
```

Получаем имя модели

```
for model in models:

  m = str(model)
  tmp['Model'] = m[:m.index('(')]
```

Обучаем модель

```
model.fit(Xtrain, Ytrain)
```

Вычисляем коэффициент детерминации

```
tmp['R2_Y'] = r2_score(Ytest, model.predict(Xtest))
```

Записываем данные и итоговый DataFrame

```
TestModels = TestModels.append([tmp])
```

Делаем индекс по названию модели

```
TestModels.set_index('Model', inplace=True)
print(TestModels)
```

Созданный датасет с рассчитанным коэффициентом R^2

```
Model                   R2_Y
LinearRegression        0.788655
RandomForestRegressor   0.840712
KNeighborsRegressor     0.836159
SVR                     0.789132
```

Построим графики R^2 коэффициентов

```
TestModels.R2_Y.plot(kind='bar', title='R2_Y')
```

Далее на рисунке 8 представлен график R^2 коэффициентов

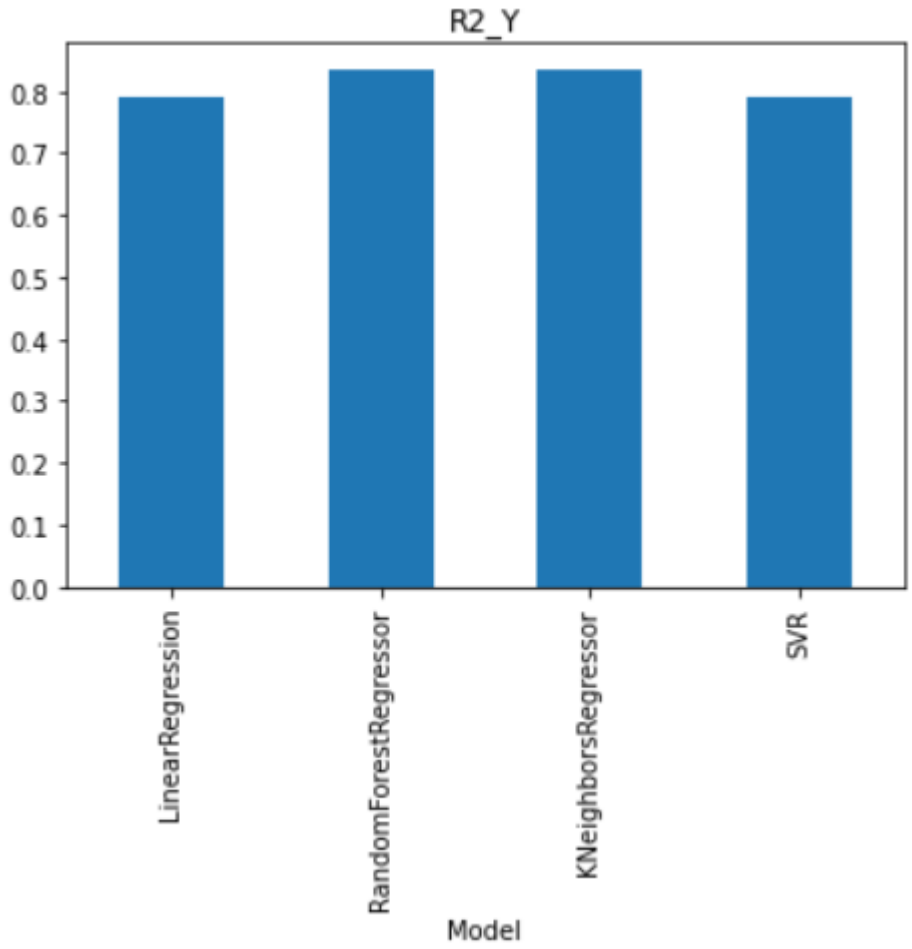



Рисунок 8 – График R^2 коэффициентов

Исходя из графика, можно сделать вывод, что модель, построенная методом случайных деревьев, самая точная (R^2 = 0.840712)

По новой обучим модель

```
model = models[1]
model.fit(Xtrain, Ytrain)
```

Посмотрим, как каждый фактор влияет на прогнозное значение Y

```
model.feature_importances_
```

вывод:

```
array([0.33196456,  0.1460898  ,  0.28349085,  0.11946502,  0.04653737,
0.07245241])
```

Как мы можем увидеть ВВП на душу населения и продолжительность жизни наиболее сильно влияют на оценку счастья (около 60%), социальная поддержка и свобода жизненного выбора почти равнозначны (около 14 %),
а самое незначительное влияние у щедрости и коррупции (около 5% и 9% с ответственно)

## Кластеризация и дерево решений

Кластеризация (кластерный анализ) — задача группировки множества объектов на подмножества (кластеры) таким образом, чтобы объекты из одного кластера были более похожи друг на друга, чем на объекты из других кластеров по какому-либо критерию.

Кластеризация проводилась методами: k-means, Mean shift, иерархическая кластеризация.

### Метод k-mean

Алгоритм разбивает множество элементов векторного пространства на заранее известное число кластеров k. Основная идея заключается в том, что на каждой итерации перевычисляется центр масс для каждого кластера, полученного на предыдущем шаге, затем векторы разбиваются на кластеры вновь в соответствии с тем, какой из новых центров оказался ближе по выбранной метрике.

```
X = df_all.values[:,1:]
X = np.nan_to_num(X)
Clus_dataSet = StandardScaler().fit_transform(X)
k_means = KMeans(init = "k-means++", n_clusters = 3, n_init = 12)
k_means.fit(X)
labels = k_means.labels_
```

```
df_all["Class"] = labels
df_all.groupby("Class").mean()
```

На 9.1 изображен результат выполнения метода k-mean

| Class | Score | GDP per capita | Social support | Healthy life expectancy | Freedom to make life choices | Generosity | Perceptions of corruption |
|---|---|---|---|---|---|---|---|
| 0 | 6.488727 | 1.329645 | 1.348938 | 0.836728 | 0.498882 | 0.238157 | 0.183698 |
| 1 | 5.304936 | 0.931416 | 1.083568 | 0.632216 | 0.388437 | 0.184902 | 0.085470 |
| 2 | 4.141788 | 0.388352 | 0.745482 | 0.312042 | 0.339124 | 0.244801 | 0.114086 |

Рисунок 9 – результат выполнения метода k-mean

Как мы можем видеть алгоритм K-means разделил наши данные на 3 кластера. Попробуем разобраться по каким критериям производилось разделение

К классу 0 присвоены страны, в которых наивысшие показатели ВВП на душу населения, социальная поддержка, предполагаемая продолжительность жизни, свобода жизненного выбора, но в то же время средний показатель щедрости и наивысший показатель коррумпированности

К классу 1 присвоены страны в которых средние показатели ВВП на душу населения, социальная поддержка, предполагаемая продолжительность жизни, свобода жизненного выбора, но в то же время низший показатель щедрости и низший показатель коррумпированности

К классу 2 присвоены страны в которых низшие показатели ВВП на душу населения, социальная поддержка, предполагаемая продолжительность жизни, свобода жизненного выбора, но в то же время высший показатель щедрости и средний показатель коррумпированности

Из чего можно сделать вывод, что для 2019 года ключевыми показателями являются:

- ВВП на душу населения
- социальная поддержка
- предполагаемая продолжительность жизни
- свобода жизненного выбора

Метод Mean shift

Алгоритм Mean shift назначает точки данных кластерам итеративно, смещая точки в направлении наивысшей плотности точек данных, то есть центроида кластера. В отличие от кластеризации k-means, он не делает никаких предположений; следовательно, это непараметрический алгоритм.

```
X = df_all.values[:,1:]
X = np.nan_to_num(X)
Clus_dataSet = StandardScaler().fit_transform(X)

ms = MeanShift()
ms.fit(Clus_dataSet)
labels = ms.labels_
cluster_centers = ms.cluster_centers_
print(cluster_centers)
n_clusters_ = len(np.unique(labels))
print("Estimated clusters:", n_clusters_)
colors = 10*['r.','g.','b.','c.','k.','y.','m.']

for i in range(len(X)):
   plt.plot(X[i][0], X[i][1], colors[labels[i]], markersize = 5)
plt.scatter(cluster_centers[:,0],cluster_centers[:,1])
plt.show()
```

На 9.2 изображен результат выполнения метода Mean shift

```
[[ 0.10726012  0.17384108  0.14108375 -0.03910207 -0.34458138 -0.34705862]]
Estimated clusters: 1
```

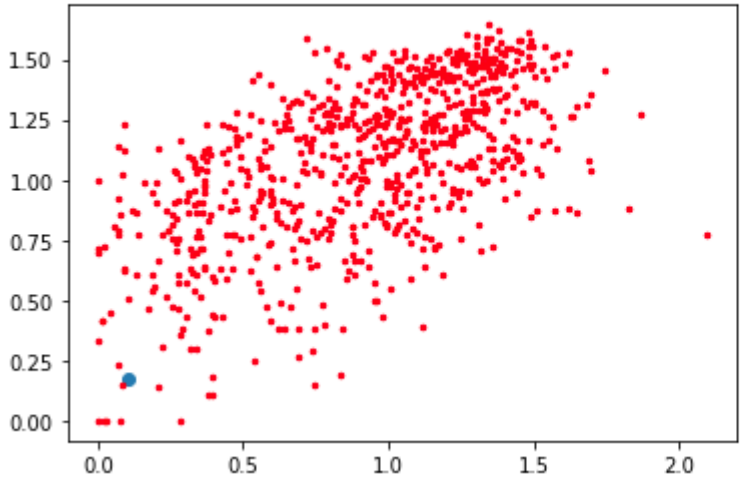



Рисунок 10 – результат выполнения метода k-mean

Иерархическая кластеризация

Суть иерархической кластеризации состоит в последовательном объединении меньших кластеров в большие или разделении больших кластеров на меньшие.

```
seeds_df = df_all
varieties = list(seeds_df.pop('Score'))
samples = seeds_df.values

mergings = linkage(samples, method='complete')
plt.figure(figsize=(80, 30))
dendrogram(mergings,
           labels=varieties,
           leaf_rotation=90,
           leaf_font_size=6,
           )
plt.show()
```

На рисунке 9.3 изображен результат выполнения иерархической кластеризации

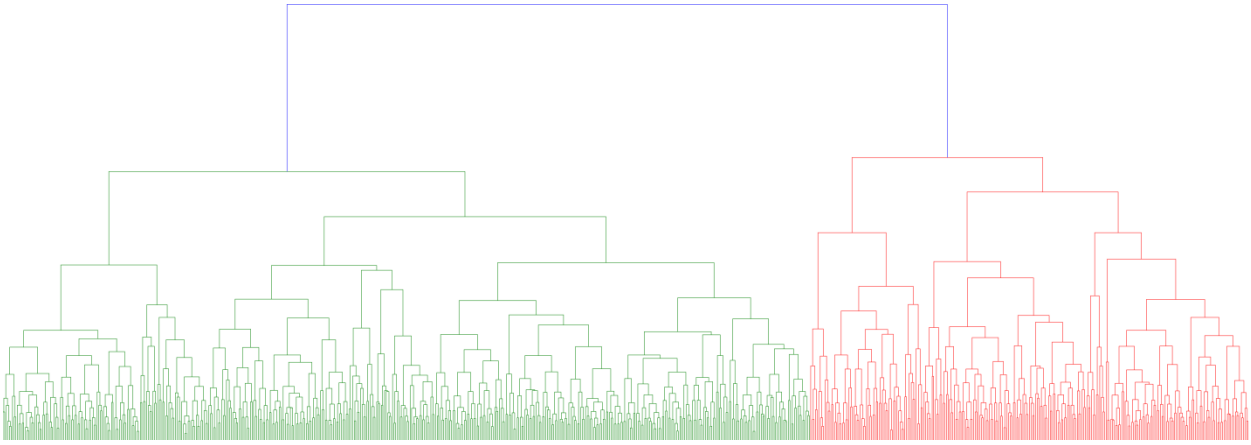


Рисунок 11 – Иерархическая кластеризация

Было построено дерево решений для классов кластеризованных методом k-means

```
df_all.head()
age_sal_tree = DecisionTreeClassifier()
age_sal_tree.fit(df_all[['Score', 'GDP per capita','Social support',
'Healthy life expectancy','Freedom to make life choices','Generosity','
Perceptions of corruption']].values, df_all['Class'].values);

dot_data = StringIO()
ex-
port_graphviz(age_sal_tree , out_file=dot_data, feature_names=['Score',
 'GDP per capita','Social support','Healthy life expectancy','Freedom t
o make life choices','Generosity','Perceptions of corruption'])
(graph, ) = graph_from_dot_data(dot_data.getvalue())
Image(graph.create_png())
```

На рисунке 12 изображено дерево решений

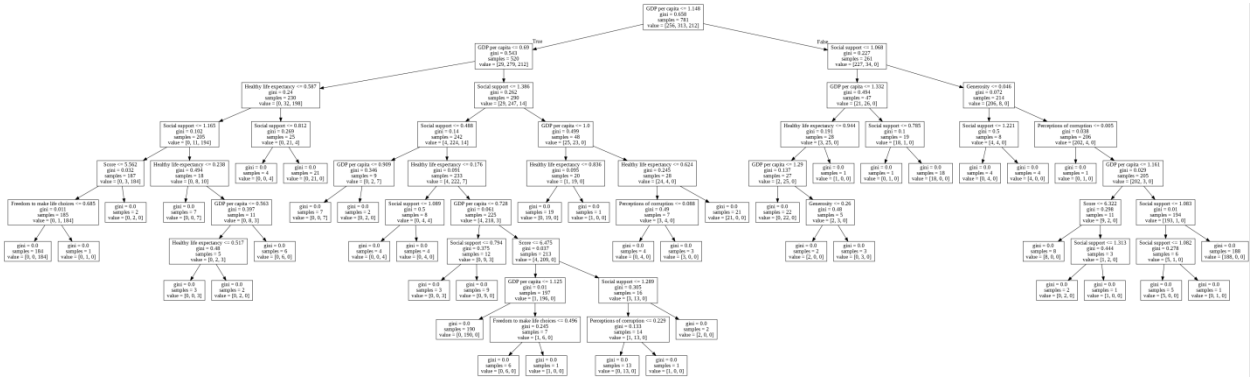


Рисунок 12 – Дерево решений

## Создание и обучение RNN модели

Рекуррентные нейронные сети (RNN) — вид нейронных сетей, где связи между элементами образуют направленную последовательность. Благодаря этому появляется возможность обрабатывать серии событий во времени или последовательные пространственные цепочки.

При реализации нейронной сети мы использовали следующие настройки:

- метод оптимизации – adam,
- метрика расчёта потерь – MAE
- обучение производится на 20 эпохах

На рисунке 13 представлен график истории обучения нейронной сети на каждой эпохе

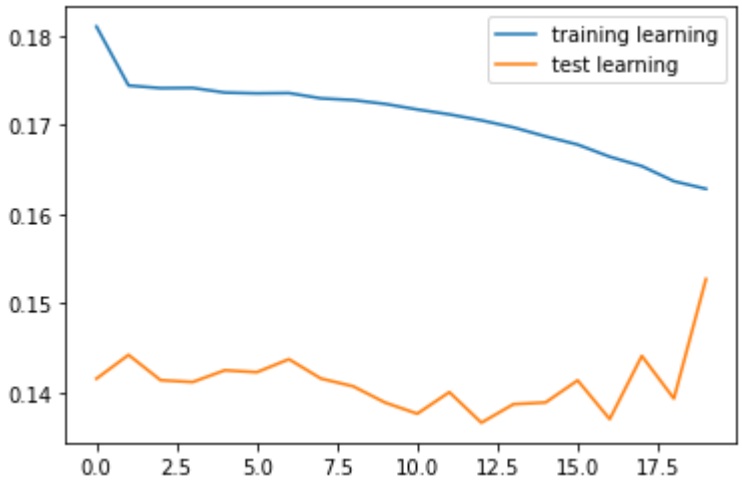
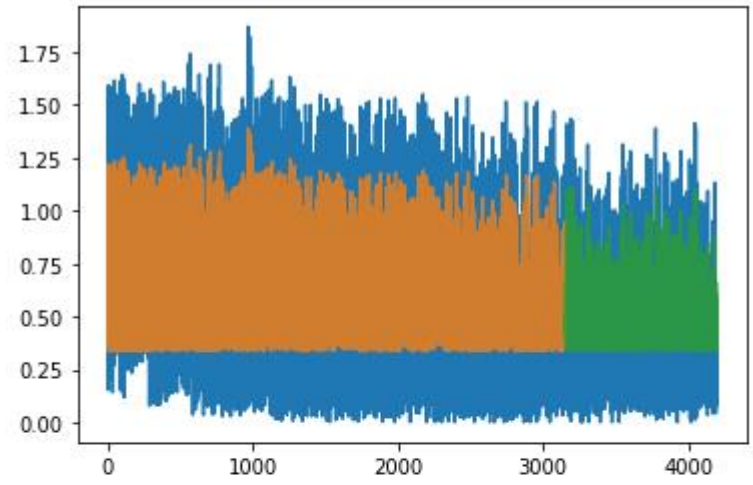
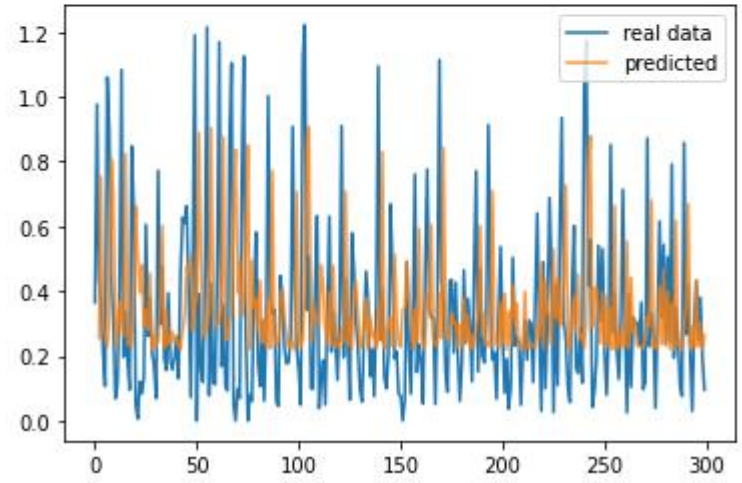



Рисунок 13 – График истории обучения

На рисунке 14 представлены графики предсказанных и реальных критериальных значений




```
train MSE = 0.26
test MSE = 0.11
```




```
MSE = 0.10
```

Рисунок 14 - Графики предсказанных и реальных критериальных значений

Как можно увидеть, нейросеть хоть и имеет низкий объем потерь, но ее точность не самая высокая, объясняется это тем, что наш исходный массив данных имеет небольшой объем.

Код построения нейросети:

```
def rearrange(ds, lb=2):
    X, Y = [], []
    for i in range(len(ds) - lb - 1):
        a = ds[i:(i + lb), 0]
        X.append(a)
        Y.append(ds[i + lb, 0])
    X = np.array(X)
    X = np.reshape(X, (X.shape[0], 1, 1))
    return X, np.array(Y)

def create_rnn():
    # создаём модель Sequential
    model = Sequential()
    # добавляем входной слой из LSTM
    model.add(LSTM(8, input_shape=(1, 1)))
    # и на выходе Dense слой
    model.add(Dense(1))
    # метрика расчёта потерь -- MAE (Mean Absolute Error)
    # метод оптимизации -- adam
    model.compile(loss='mae', optimizer='adam')
    return model



# путь до файла с моделью
model_file = Path('model_BTC.h5')

# загружаем наши данные
df = df_all.copy()
df = df.drop(['Score'], axis=1)
take_n = 700
# делаем выборку данных для обучения и проверки
ds = np.array(df[:take_n]).reshape(-1, 1)
# в определенном виде
print(ds[:10])

# перенормируем данные
scl = MinMaxScaler(feature_range=(0, 1))
ds_scl = scl.fit_transform(ds)
```

```
    # разделяем на две выборки: обучающую и для проверки
    d_train, d_test = train_test_split(ds_scl, shuffle=False)
    print(f'train_shape = {d_train.shape}, test_shape = {d_test.shape}')

    # количество элементов по которым делается прогноз
    lb = 1
    # количество элементов для пропуска
    skip_count = lb * 2
    # приводим данные к виду который понимает нейронка
    X_train, Y_train = rearrange(d_train, lb)
    X_test, Y_test = rearrange(d_test, lb)
    print(f'X_train_size = {X_train.shape}, Y_train_shape = {Y_train.sha
pe}')

    # проверка на существование файла
    if not model_file.exists():
        # создаём
        model = create_rnn()
        # обучаем
        fit-
ted = model.fit(X_train, Y_train, epochs=20, batch_size=1, validation_d
ata=(X_test, Y_test), verbose=2)
        # сохраняем чтобы каждый раз не обучать
        model.save(model_file)
        # смотрим как она обучалась на каждой эпохе
        plt.plot(fitted.history['loss'], label='training learning')
        plt.plot(fitted.history['val_loss'], label='test learning')
        plt.legend()
        plt.show()
    else:
        # загружаем если есть обученная модель
        model = load_model(model_file)

    # посмотрим как она справляется с прогнозом на тех же данных
    x_train = X_train.reshape(X_train.shape[0])
    x_test = X_test.reshape(X_test.shape[0])
    y_pred_train = model.predict(X_train)
    y_pred_test = model.predict(X_test)
```

```
    # возвращаем данным исходные границы изменения
    y_pred_train = scl.inverse_transform(y_pred_train)
    y_pred_test = scl.inverse_transform(y_pred_test)

    # реальные значения x
    x_real = np.arange(0, ds.shape[0], 1)
    x_v_train, x_v_test = train_test_split(x_real, shuffle=False)
    # и графики
    plt.plot(x_real, ds, label='real data')
    plt.plot(x_v_train[skip_count:], y_pred_train, label='train predict'
, alpha=0.8)
    plt.plot(x_v_test[skip_count:], y_pred_test, label='test predict', a
lpha=0.8)
    plt.show()

    # делаем выборку исключая первые элементы
    y_real_first = ds[skip_count:y_pred_train.size + skip_count]
    y_real_second = ds[y_real_first.shape[0] + 2 * skip_count:]
    # считаем ошибку прогнозирования используя MSE
    print('train MSE = {:.2f}'.format(mean_squared_error(y_real_first, y
_pred_train)))
    print('test MSE = {:.2f}'.format(mean_squared_error(y_real_second, y
_pred_test)))

    take_m = 50
    # подготовим новые данные
    ds = np.array(df[take_n:take_n + take_m]).reshape(-1, 1)
    ds_scl = scl.fit_transform(ds)
    X_val, _ = rearrange(ds_scl, lb)
    # произведём прогноз на новом блоке данных
    Y_pred = model.predict(X_val)
    # вернём нормальным вид данным
    y_pred = scl.inverse_transform(Y_pred)
    x_real = np.arange(0, ds.shape[0], 1)

    # выводим графики
    plt.plot(x_real, ds, label='real data')
    plt.plot(x_real[skip_count:], y_pred, label='predicted', alpha=0.8)
    plt.legend()
```

```
    plt.show()

    print('MSE = {:.2f}'.format(mean_squared_error(ds[skip_count:], y_pr
ed)))
```

## Заключение

Проведя анализ датасета, команде удалось установить корреляционный зависимости, построить регрессивную модель, кластеризировать данные несколькими способами, на основе проведенной кластеризации построить деревья решений, а также создать реккурентную нейронную сеть, которая предсказывает величину оценочных критериев.

## 3. Методические указания к лабораторным работам

### 3.1 Лабораторная работа № 1. Базовые статистические инструменты анализа данных, изучение библиотечных инструментов R/Python

**Задание**

Создайте фрейм данных (таблицу) из N записей со следующими полями: Nrow - номер записи, Name – имя сотрудника, BirthYear – год рождения, EmployYear – год приема на работу, Salary –зарплата. Заполните данный фрейм данными так, что Nrow изменяется от 1 до N. Name задается произвольно, BirthYear распределено равномерно (случайно) на отрезке [1965, 1990], EmployYear распределен равномерно на отрезке [BirthYear+18, 2006], Salary задается произвольнов интервале от 10000 до 30000. Подсчитайте число сотрудников с зарплатой, больше 15000. Добавьте в данные поле, соответствующее суммарному подоходному налогу (ставка 13%), выплаченному сотрудником за время работы в организации.

### 3.2 Лабораторная работа № 2. Разведочный анализ данных

**Задание**

Для выбранного или произвольного набора данных из репозитория UC Irvine Machine Learning Repository необходимо выполнить следующие вычисления:

1. найти среднее значение, медиану, экстремумы любого параметра;
2. произвести предобработку данных (удалить шумы, дубликаты)
3. найти дисперсию и стандартное отклонение любого параметра;
4. найти линейную зависимость между двумя любыми параметрами;
5. построить диаграммы рассеяния;
6. построить гистограмму, определить тип распределения одного из параметров;
7. подвести итог (summary).

### 3.3 Лабораторная работа № 3. Регрессионный анализ

**Задание**

Решите задачу множественной линейной регрессии для выбранного набора данных или произвольного набора данных из репозитория UC Irvine Machine Learning Repository. Проанализируйте полученные результаты и сделайте выводы

### 3.4 Лабораторная работа № 4. метод главных компонент

**Задание**

1. Выполните анализ главных компонент для выбранного или произвольного набора данных из репозитория UC Irvine Machine Learning Repository. Выделить главные факторы, приведите интерпретацию результатов (или покажите, что этого сделать невозможно).

2. Выберите два коррелирующих параметра и проведите для них анализ главных компонент, выделите главные факторы, постарайтесь дать интепретацию.

3. Сравните результаты по двум пунктам

### 3.5 Лабораторная работа № 5. деревья решений

**Задание**

Для выбранного или произвольного набора данных из репозитория UC Irvine Machine Learning Repository необходимо выполнить классификацию с помощью метода дерева решений, построить само дерево, произвести интерпретацию полученного результата.

### 3.6 Лабораторная работа № 6. ансамблевый подход

**Задание**

Для выбранного или произвольного набора данных из репозитория UC Irvine Machine Learning Repository необходимо выполнить классификацию с использованием ансамблевого подхода – бэггинга или бустинга на выбор, произвести интерпретацию полученного результата.

### 3.7 Лабораторная работа № 7. Кластеризация

**Задание** (одно на выбор)

1. Выполните кластеризацию данных с использованием k-means для своего или произвольного набора данных из репозитория UC Irvine Machine Learning Repository. Оцените качество кластеризации в зависимости от числа итераций и значения k. Дайте рекомендации по выбору числа кластеров.

2. Выполните кластеризацию данных с использованием MeanShift для своего или произвольного набора данных из репозитория UC Irvine Machine Learning Repository. Проанализируйте как изменения параметра h при реализации MeanShift влияют на результат?

### 3.8 Лабораторная работа № 8. искусственные нейронные сети

**Задание**

Для выбранного или произвольного набора данных из репозитория UC Irvine Machine Learning Repository решить задачу классификации (кластеризации или распознавания образов) с использованием ИНС. Разработать собственную ИНС. Создать топологию ИНС. Обучить ИНС. Выполнить эмуляцию сети.

### 3.9 Требования и состав отчёта

1. Отчёт должен быть выполнен на листе размером А4.

2. Отчёт должен начинаться с титульного листа с названием вуза и факультета, номером и названием лабораторной работы, вариантом, ФИО студента, № группы, ФИО преподавателя, городом и годом.

3. В отчёте нужно кратко описать задание, показать основные этапы вычисления при выполнении всех операций, сформулировать выводы.

4. Отчёт предоставить в бумажном или электронном виде (записать на флэш-накопитель и продублировать себе на электронную почту).

## 4. ЗАКЛЮЧЕНИЕ

Технология машинного обучения и нейросетевых моделей является необычайно мощным инструментом решения научно-технических задач в современном мире данных. Владение данной технологией требует знания не только инструментальных средств ее реализации, но и теоретических основ: математической статистики и линейной алгебры, анализа временных рядов; математико-статистических основ и алгоритмов машинного обучения; принципов валидации моделей, внедрения моделей в бизнес-продукты. Этим вопросам и посвящен курс машинного обучения и нейросетевых моделей.

## Рекомендуемая литература по курсу


1 С.Хайкин. Нейронные сети: полный курс. 2-е изд. М., "Вильямс", 2006.

2 Д.А.Тархов. Нейронные сети. Модели и алгоритмы. М., Радиотехника, 2005. (Научная серия "Нейрокомпьютеры и их применение", ред. А.И.Галушкин. Кн.18.)

3 Л.Г.Комарцова, А.В.Максимов. Нейрокомпьютеры. М., Изд-во МГТУ им.Баумана, 2004.

4 А.И.Галушкин. Нейронные сети. Основы теории. М., Горячая линия - Телеком, 2010.

5 В.А.Головко. Нейронные сети: обучение, организация и применение. М., ИПРЖР, 2001.

6 C.C.Aggarwal. Neural Networks and Deep Learning. A Textbook. Springer International Publishing AG, 2018. DOI 10.1007/978-3-319-94463-0 ISBN 978-3-319-94462-3

7 К.В. Воронцов. Машинное обучение. Курс лекций.

8 С.А. Шумский. Машинный интеллект. Очерки по теории машинного обучения и искусственного интеллекта. М., РИОР, 2019. DOI: 10.29039/02011-1

9 I.Goodfellow, Y.Bengio, A.Courville. Deep Learning. MIT Press, 2016. http://www.deeplearningbook.org/

10 D.Foster. Generative Deep Learning: Teaching Machines to Paint, Write, Compose, and Play. O'Reilly Media, 2019. ISBN 1492041947.

11 Дж.Вандер Плас. Python для сложных задач. Наука о данных и машинное обучение. Питер, 2018. ISBN 978-5-446-10914-2.

12 М.Доусон. Программируем на Python. Питер, 2012. ISBN 978-5-459-00314-7.

13 М.Лутц. Программирование на Python. Том 1. Символ-Плюс , 2011. ISBN 978-5-93286-210-0.

14 У.Маккинни. Python и анализ данных. ДМК Пресс , 2015, 482 с. ISBN 978-5-97060-315-4.

15 А.Мюллер, С.Гвидо. Введение в машинное обучение с помощью Python. Руководство для специалистов по работе с данными. Вильямс, 2017, 480 с. ISBN 978-5-99089-108-1.

16 Б.Любанович. Простой Python. Современный стиль программирования. Питер, 2019, 480 с. ISBN 978-5-446-11054-4.

17 О.Жерон. Прикладное машинное обучение с помощью Scikit-Learn и TensorFlow. Концепции, инструменты и техники для создания интеллектуальных систем. Вильямс, 2018, 688 с. ISBN 978-5-950-02962-2.

18 Х.Уикем, Г.Гроулмунд. Язык R в задачах науки о данных. Импорт, подготовка, обработка, визуализация и моделирование данных. Вильямс, 2017. ISBN 978-5-9909446-8-8.

19 Р.Кабаков. R в действии. Анализ и визуализация данных на языке R. ДМК Пресс, 2014, 588 с. ISBN 978-5-97060-241-6.

20 P.Dalgaard. Introductory Statistics with R. Springer, 2008, XVI+364 pp. ISBN 978-0-387-79054-1.

21 Т.Рашид. Создаём нейронную сеть. Математические идеи, лежащие в основе нейронных сетей, и поэтапное создание собственной нейронной сети на языке Python. Вильямс, 2018. ISBN 978-1530826605.

22 Ф.Шолле. Глубокое обучение на Python. Питер, 2018. ISBN 978-5-4461-0770-4.

23 С.Рашка, В.Мирджалили. Python и машинное обучение. Машинное и глубокое обучение с использованием Python, scikit-learn и TensorFlow. 2-е издание. Вильямс, 2019. ISBN 978-5-907114-52-4.

24 Б.Бенгфорт, Р.Билбро, Т.Охеда. Прикладной анализ текстовых данных на Python. Машинное обучение и создание приложений обработки естественного языка. Питер, 2019. ISBN 978-5-4461-1153-4.Я

25 R.Sutton, A.Barto. Reinforcement Learning: An Introduction. MIT Press, 1998.

26 L.Graesser, W.L.Keng. Foundations of Deep Reinforcement Learning: Theory and Practice in Python. Addison-Wesley Professional, 2019.

27 T.Beysolow II. Applied Reinforcement Learning with Python: With OpenAI Gym, Tensorflow, and Keras. Apress, 2019. ISBN 1484251261.

28 M.Lapan. Deep Reinforcement Learning Hands-On: Apply modern RL methods, with deep Q-networks, value iteration, policy gradients, TRPO, AlphaGo Zero and more. Packt Publishing, 2018. ISBN 1788834240.

Учебное издание

Тимур Александрович Яновский
Дмитрий Андреевич Астахов

**МАШИННОЕ ОБУЧЕНИЕ И НЕЙРОСЕТЕВЫЕ МОДЕЛИ**

*Учебно-методическое пособие*

Редактор *Л. Н. Рыжих*

На правах рукописи

Волгоградский государственный технический университет.
400005, г. Волгоград, просп. В. И. Ленина, 28, корп. 1.