Федеральное государственное образовательное бюджетное

учреждение высшего образования

**«Финансовый университет**

**при Правительстве Российской Федерации»**

**(Финансовый университет)**

Департамент анализа данных и машинного обучения

Пояснительная записка к курсовой работе по дисциплине

«Технологии анализа данных и машинного обучения»

на тему:

**«Машинное обучение в задачах классификации текстов»**

Выполнил:

Студент группы ПИ19-1

Содиков Фарход Фирдавсович

Научный руководитель:

Доцент, Кандидат педагогических наук

Никитин Пётр Владимирович

**Москва**

**2022**

Содержание

[ВВЕДЕНИЕ 3](#_Toc101134391)

[Инструменты реализации 4](#_Toc101134392)

[Среда разработки Jupyter Notebook 4](#_Toc101134393)

[Язык программирования Python 4](#_Toc101134394)

[Библиотека NumPy 4](#_Toc101134395)

[Библиотека Pandas 4](#_Toc101134396)

[Библиотека Scikit Learn 4](#_Toc101134397)

[ЗАКЛЮЧЕНИЕ 6](#_Toc101134398)

[ИСТОЧНИКИ 6](#_Toc101134399)

# **ВВЕДЕНИЕ**

Основная идея курсового проекта, в целом машинного обучения, состоит в том, чтобы машина (компьютерная) не просто использовала заранее написанный алгоритмический код, а сама обучалась решению поставленной задачи. Главной и единственной средой разработки в создании машинного обучения является Jupyter Notebook, с помощью которого сразу можно видеть результат выполнения кода или одного из его фрагментов. В данном проекте единственным языком программирования является Python, потому что он является одним из тех языков, который легко и просто решает проблемы, связанные с машинным обучением и анализом больших данных. Также этот язык имеет множество различных библиотек, некоторые из которых используется в поставленной задаче.

В разработку машинного обучения входит:

* Классификация данных
* Кластеризация данных
* Выявление аномалий (в нашей задаче не является обязательным пунктом)
* Снижение размерности (в нашей задаче не является обязательным пунктом)
* Предсказание событий (регрессия)

В наше время искусственный интеллект значительно развивается и всё больше IT-компаний нуждаются в создании машин, которые обучают самих себя для автоматизации работ не только в IT-сфере, но и в других отраслях.

К концу работы должен быть готовый машинный код, который проанализирует небольшой набор данных (датасет) и обучит себя эмпирическим методом с использованием различных библиотек.

# **Инструменты реализации**

## **Среда разработки Jupyter Notebook**

Jupyter Notebook – это среда разработки, где пользователь сразу может видеть выполнение кода или его отдельные фрагменты. Одной из главных отличий этой среды от других является возможность разбиения программного кода на куски с любым порядком выполнения. То есть вы можете создать класс или функцию и сразу же её проверить без запуска и выполнения всего программного кода. Также можно отдельно смотреть содержимое загруженных файлов и отдельно обработать данные файла. Этот функционал помогает достичь поставленных целей, так как, если мне захочется построить график прямо в середине кода я смогу просто запустить функцию и увидеть результат. Команда jupyter notebook создала свой ноутбук в облаке. Теперь пользователи могут пользоваться им через браузеры.

## **Язык программирования Python**

Язык программирования Python – высокоуровневый ЯП общего назначения, направленный на повышение производительности разработчика и читаемости кода. Этот язык имеет множество парадигм программирования, такие как объектно-ориентированное (ООП), императивное, структурное, функциональное и другие. У языка есть свободная лицензия, позволяющая использовать ЯП без ограничений в любых разработках.

Python хорошо подходит под выполнения математических задач, научных расчётов, а также для машинного обучения и анализа больших объёмов данных. Следующей причиной, по которой мы используем этот язык, является наличие большого количества различных библиотек для разработки приложений, а также анализа, обработки, визуализации данных, но основными пакетами в разработке машинного обучения и анализа данных являются следующие библиотеки: NumPy, Pandas, Sklearn, Matplotlib, Seaborn, а также time, collections, itertools, warnings.

## **Библиотека NumPy**

NumPy – это одна из популярных библиотек с открытым исходным кодом языка Python, которая помогает добавить поддержку огромных многомерных массивов и матриц, а также высокоуровневых и математических функций. Одним из основных объектов NumPy является одномерный массив.

Если реализовывать математические алгоритмы на чистом Python, то зачастую ваш программный код работает медленнее, чем на других компилируемых языках, поэтому NumPy помогает оптимизировать работу с многомерными массивами. NumPy написан на языке C.

## **Библиотека Pandas**

Pandas – это высокоуровневая (построена поверх более низкоуровневой библиотеки NumPy) библиотека языка Python для анализа и обработки данных. Данная библиотека является самой продвинутой и перспективной в работе с данными. Самыми основными структурами данного пакета: DataFrame и Series. Эти специальные структуры данных и операции для манипулирования временными рядами и числовыми таблицами. Название Pandas происходит от эконометрического термина «панельные данные». Pandas находится под лицензией BSD, которая позволяет свободно пользоваться данной библиотекой. Возможности библиотеки:

* Манипуляция индексированными массивами двумерных массивов.
* Совмещение данных и обработка информации
* Объединение и слияние наборов данных

## **Библиотека Scikit Learn**

Scikit-learn – это также один из популярнейших и широко использующих пакетов для машинного обучения и Data Science. С помощью этого ПО можно выполнять большое количество операций. Scikit-learn имеет большое количество встроенных готовых алгоритмов. Эта библиотека имеет одну из лучших документаций о своих встроенных алгоритмах классах, методах и функциях.

Scikit-learn поддерживает:

* Выбор моделей
* Классификации
* Регрессии
* Предварительную обработку данных
* Кластерный анализ
* Уменьшение размерности

Scikit-learn не поддерживает:

* Нейронные сети
* Обучение с ассоциативными правилами
* Самоорганизующиеся карты (Кохонена сети)
* Обучения с подкреплением (Reinforcement learning)

У пакета есть свои наборы данных, с помощью которых можно тестировать свои модели. Так же как и вышеперечисленные библиотеки, Scikit-learn имеет открытый исходный код. Он бесплатный и лицензирован под BSD, как и Pandas-пакет.

Алгоритмы классификации:

* *LogisticRegression* (LogReg) – это один из главных алгоритмов данной библиотеки. Используется для отнесения исследований к дискретному набору классов. Эта регрессия преобразует свой вывод с помощью Сигмоида для возращения значения вероятности, которое в следствие может округлиться в сторону одного из дискретных классов.
* *KNeighborsClassifier* – это одна из контролируемых моделей машинного обучения. Модель учится на наборе помоченных данных. Сначала модель получает набор входных объектов и выходных значений. После приёма данных модель обучается на них, чтобы узнать, как сопоставить выходные данные с желаемыми выходными данными. Это, в свою очередь, нужно для того, чтоб модель могла делать прогнозы на невидимых данных. Работает она просто. В первую очередь модель берет ближайшие помеченные точки, глядя на «k». Затем происходит присвоение меток для большинства точек рядом с «k». Например, если k = 20, а пятнадцать из точек жёлтые, а пять остальных голубые, то рассматриваемая точка превратится в жёлтый, так как желтый цвет является большинством.
* *DecisionTreeClassifier* – это древовидная структура, похожая на блок-схему, в которой внутренний узел представляет функцию (или атрибут), ветвь представляет правило принятия решения, а каждый конечный узел представляет результат. Самый верхний узел в дереве решений известен как корневой узел. Он учится разделять на основе значения атрибута. Он рекурсивно разбивает дерево, вызывая рекурсивное разбиение. Эта структура, похожая на блок-схему, поможет вам в принятии решений. Это визуализация, похожая на блок-схему, которая легко имитирует мышление на человеческом уровне. Вот почему деревья решений легко понять и интерпретировать.
* *SVM* (машина опорных векторов) – это один из алгоритмов классификации с контролируемым машинным обучением. Её результаты хороши, поэтому этот алгоритм является одним из популярных. SVM отличается от других алгоритмов классификации тем, что он выбирает границу принятия решения, которая максимизирует расстояние от ближайших точек данных всех классов. SVM не просто находит границу принятия решения; он находит наиболее оптимальную границу принятия решения. Наиболее оптимальной границей принятия решения является та, которая имеет максимальный запас от ближайших точек всех классов. Ближайшие точки от границы принятия решения, которые максимизируют расстояние между границей принятия решения и точками, называются опорными векторами.
* *MLP* (Многослойный персептронный классификатор) – это модель искусственной нейронной сети с прямой связью, которая сопоставляет входные наборы данных с набором соответствующих выходных данных. MLP состоит из нескольких слоев, и каждый слой полностью соединен со следующим. Узлами слоев являются нейроны с нелинейными функциями активации, за исключением узлов входного слоя. Между входным и выходным слоями может быть один или несколько нелинейных скрытых слоев.
* *Наивные байесовские оценки* — это вероятностные оценки, основанные на теореме Байеса с предположениями о том, что между объектами существует сильная независимость. Теорема Байеса помогает нам определить вероятность наступления событий на основе некоторого предварительного знания условий, которые могут быть связаны с событием. Наивные байесовские классификаторы довольно хорошо работают для приложений классификации документов и фильтрации нежелательной почты. Он требует небольшого объема обучающих данных для настройки с учетом вероятностей для теоремы Байеса и поэтому работает довольно быстро. Scikit-Learn предоставляет список из 4 наивных байесовских оценок: 1) BernoulliNB - представляет собой классификатор, основанный на данных, представляющих собой многомерные распределения Бернулли. Распределение Бернулли подразумевает, что данные могут иметь несколько функций, но предполагается, что каждая из них является двоичной переменной.  
  2) GaussianNB - представляет собой классификатор, основанный на предположении, что вероятность признаков является гауссовым распределением.  
  3) ComplementNB - представляет собой классификатор, который использует дополнение каждого класса для вычисления весов модели. Это стандартный вариант многочленного наивного Байеса, который хорошо подходит для несбалансированных задач классификации классов.  
  4) MultinomialNB - представляет собой классификатор, который подходит для многомерно распределенных данных.
* *RandomForestClassifier* – это один из алгоритмов работы с учителем. Зачастую его используют для регрессии и для классификации. Этот алгоритм является одним из простых в использовании. Random Forest создаёт деревья решений для случайных sample-data. Затем делает прогноз от каждого дерева и в итоге выбирает лучший с помощью голосования. Бонусом является то, что он предоставляет мерку важности признаков. Случайный лес основан на алгоритме Борута, который определяет наиболее важные и значимые признаки датасета.

Этапы работы случайного леса:

1. Создание случайной выборки из датасета.
2. Построение деревьев для каждой выборки и получение результатов.
3. Голосование за каждый полученный прогноз.
4. Окончательным результатом будет являться ветвь с наибольшим количеством голосов.

## **Библиотека Matplotlib**

Matplotlib – это обширная библиотека для создания статических, анимированных и интерактивных визуализаций на Python. Matplotlib делает простые вещи простыми, а сложные возможными.

Возможности Matplotlib:

* Создание [интерактивных фигур](https://mybinder.org/v2/gh/matplotlib/mpl-brochure-binder/main?labpath=MatplotlibExample.ipynb), которые можно масштабировать, панорамировать и обновлять.
* Экспорт во [многие форматы файлов](https://matplotlib.org/stable/api/figure_api.html#matplotlib.figure.Figure.savefig).
* Богатый набор [сторонних пакетов](https://matplotlib.org/mpl-third-party/), созданных на основе Matplotlib.
* Создание [графиков качества публикации](https://ieeexplore.ieee.org/document/4160265/citations?tabFilter=papers).
* Настройка [визуального стиля](https://matplotlib.org/stable/gallery/style_sheets/style_sheets_reference.html) и [макета](https://matplotlib.org/stable/tutorials/provisional/mosaic.html).
* Встраивание в [JupyterLab и графические пользовательские интерфейсы](https://matplotlib.org/stable/gallery/#embedding-matplotlib-in-graphical-user-interfaces).

# **ЗАКЛЮЧЕНИЕ**

Заключение Заключение Заключение Заключение Заключение Заключение Заключение.

# **ИСТОЧНИКИ**

Источники Источники Источники Источники Источники Источники Источники Источники.