Министерство науки и высшего образования Российской Федерации

Федеральное государственное автономное образовательное учреждение

высшего образования

«Сибирский федеральный университет»

**ИТОГОВАЯ АТТЕСТАЦИОННАЯ РАБОТА**   
по программе профессиональной переподготовки   
«Аналитика данных»

на тему: «Система для автоматического определения музыкального жанра»

|  |  |
| --- | --- |
| Слушатель: Симанов Иван Николаевич |  |
| Группа: КИ22-06Б |  |
| Руководитель итоговой аттестационной работы: Михалев А.С.,  старший преподаватель  кафедры Вычислительной техники  ИКИТ СФУ |  |

Красноярск

2024

СОДЕРЖАНИЕ

[Введение 4](#_Toc169629949)

[1 Анализ предметной области 5](#_Toc169629950)

[1.1 Постановка задачи 5](#_Toc169629951)

[1.2 Извлечение признаков 5](#_Toc169629952)

[1.3 Выбор средств разработки 6](#_Toc169629953)

[2 Проектирование приложения 7](#_Toc169629954)

[2.1 Описание данных 7](#_Toc169629955)

[2.2 Описание функционала приложения 8](#_Toc169629956)

[2.3 Требования к программе или программному изделию 8](#_Toc169629957)

[2.4 Стадии и этапы разработки 8](#_Toc169629958)

[3 Реализация приложения 10](#_Toc169629959)

[3.1 Предварительная обработка данных 10](#_Toc169629960)

[3.2 Сбор данных и предобработка 11](#_Toc169629961)

[3.3 Извлечение признаков 12](#_Toc169629962)

[3.4 Построение и обучение модели 12](#_Toc169629963)

[3.5 Оценка модели 13](#_Toc169629964)

[3.6 Классификация и распределение аудиофайлов 13](#_Toc169629965)

[4 Точность и производительность модели 14](#_Toc169629966)

[Заключение 14](#_Toc169629967)

[Список использованных источников 15](#_Toc169629968)

[ПРИЛОЖЕНИЕ А 16](#_Toc169629969)

# Введение

Музыка играет важную роль в жизни людей, являясь неотъемлемой частью культуры и повседневной жизни. Определение музыкального жанра может быть полезным для различных приложений, таких как музыкальные рекомендации, организация медиабиблиотек и анализ музыкальных тенденций. Современные технологии машинного обучения и обработки аудио позволяют автоматизировать процесс определения жанра, делая его быстрым и точным.

Цель данной работы - разработка системы для автоматического определения музыкального жанра на основе аудиофайлов с использованием методов машинного обучения. Система должна быть способна классифицировать музыкальные треки по жанрам с высокой точностью и производительностью.

Для достижения поставленной цели необходимо решить следующие задачи:

1. Изучение существующих методов и подходов к определению музыкального жанра.

2. Сбор и подготовка данных для обучения модели, в том числе:

* Загрузка и предобработка датасета GTZAN.
* Извлечение аудиопризнаков (мел-кепстральных коэффициентов (MFCCs) и других признаков) из аудиофайлов.

3. Разработка и обучение нейронной сети для классификации музыкальных жанров.

4. Оценка точности и производительности разработанной модели.

5. Реализация системы для автоматической классификации новых аудиофайлов и распределения их по жанровым папкам.

6. Тестирование и валидация системы на реальных данных.

7. Разработка документации и руководства пользователя для системы.

1 Анализ предметной области

## 1.1 Постановка задачи

Классификация музыкальных жанров может быть выполнена с использованием различных методов машинного обучения, включая традиционные алгоритмы, такие как K-ближайших соседей (KNN), метод опорных векторов (SVM), а также современные методы глубокого обучения, такие как свёрточные нейронные сети (CNN) и рекуррентные нейронные сети (RNN).

## Извлечение признаков

Для классификации музыкальных жанров используются различные аудиопризнаки, включая мел-кепстральные коэффициенты (MFCCs), спектральную контрастность, хрома-фичи и другие. В данной работе основное внимание будет уделено MFCCs, которые являются стандартным методом для представления аудиосигналов в задачах классификации.

Формула для вычисления MFCC:

, n = 1, 2, …, N (1)

где – амплитуда сигнала в k-й полосе частот, K – количество полос, N – количество коэффициентов.

## 1.3 Выбор средств разработки

Выбор средств разработки осуществляется на первых стадиях работы над проектом, после определения требований к создаваемому программному продукту.

Существующие на сегодняшний день языки программирования можно выделить в следующие группы:

* универсальные языки высокого уровня;
* специализированные языки разработчика ПО;
* специализированные языки пользователя;
* языки низкого уровня.

Для разработки приложения было решено рассматривать универсальные высокоуровневые языки общего назначения. В данном проекте выбор пал на язык Python.

Python – высокоуровневый язык программирования общего назначения с динамической строгой типизацией и автоматическим управлением памятью, ориентированный на повышение производительности разработчика, читаемости кода и его качества, а также на обеспечение переносимости написанных на нём программ.

Python является мультипарадигмальным языком программирования, поддерживающим императивное, процедурное, структурное, объектно-ориентированное программирование, мета-программирование и функциональное программирование. Задачи обобщённого программирования решаются за счёт динамической типизации. Основные архитектурные черты – динамическая типизация, автоматическое управление памятью, полная интроспекция, механизм обработки исключений, поддержка многопоточных вычислений с глобальной блокировкой интерпретатора (GIL), высокоуровневые структуры данных. Поддерживается разбиение программ на модули, которые, в свою очередь, могут объединяться в пакеты.

# 2 Проектирование приложения

## 2.1 Описание данных

В качестве исходных данных были использованы поинтервальные наборы данных геофизических исследований скважин с месторождений в формате .xlsx (Microsoft Excel).

Объем и перечень данных ГИС приведен в таблице 1. Целевая переменная – классификация пород по Данхему.

Таблица 1 – Перечень ГИС

|  |  |  |  |
| --- | --- | --- | --- |
| № | Метод/свойство | Мнемоника в las | Кол-во скважин |
| 1 | Глубина | MD | 31 |
| 2 | ГК | GK | 31 |
| 3 | СГК (калий) | SGKK | 28 |
| 4 | СГК (торий) | SGKTh | 28 |
| 5 | СГК (уран) | SGKU | 28 |
| 6 | НГК | NGK | 31 |
| 7 | ННКт (большой зонд) | NKTB | 24 |
| 8 | ННКт (малый зонд) | NKTM | 24 |
| 9 | dTp | DTP | 31 |
| 10 | dTs | DTS | 23 |
| 11 | ГГКп | PL\_GG | 31 |
| 12 | БК | BK | 31 |
| 13 | МБК | MBK | 27 |
| 14 | Пористость | POR\_QE | 31 |
| 15 | Водородосодержвание | W | 31 |
| 16 | Данхем | Danhem | 31 |

## 2.2 Описание функционала приложения

Для пользователей приложение должно предоставлять следующие возможности:

1. Загрузка данных о скважине – загрузка данных ГИС в формате .xlsx. Функциональные требования: уведомление пользователя при загрузке некорректного файла или при ошибках обработки файла.
2. Чтение и просмотр данных в графическом виде из файлов. Функциональные требования: графическое представление всей информации, содержащейся в файле; уведомление пользователя при ошибках чтения файла.
3. Прогнозирование типа породы по данным геологических исследований скважин.

## 2.3 Требования к программе или программному изделию

Точность прогноза модели должна составлять не менее 80%. Время определения результатов – менее 5 минут.

Сохранность данных системы должна обеспечиваться при всех аварийных ситуациях.

Вероятность безотказной работы системы должна составлять не менее 99.99%.

## 2.4 Стадии и этапы разработки

Разработка должна быть проведена в три стадии:

1) определение функциональных требований к приложению;

2) рабочее проектирование: анализ обучающей выборки, разработка

3) тестирование и передача в эксплуатацию.

**2.5 Архитектура приложения**

Для классификации музыкальных жанров будет использована многослойная перцептронная сеть (MLP), состоящая из нескольких полносвязных слоев с функцией активации ReLU и выходным слоем с функцией активации softmax.

1. Система управления файлами – производит чтение данных (в формате mp3).
2. Система обработки данных – система выполняет предобработку исходных данных.
3. Система анализа данных – включает в себя обученную модель машинного обучения с настроенными параметрами, принимающую на вход mp3 файлы.
4. Система выгрузки данных – сохранение обработанных файлов в отдельные папки.

В качестве модели прогнозирования жанров музыки используется датасет GTZAN.

# 3 Реализация приложения

## Предварительная обработка данных

Сперва необходимо указать путь к датасету.

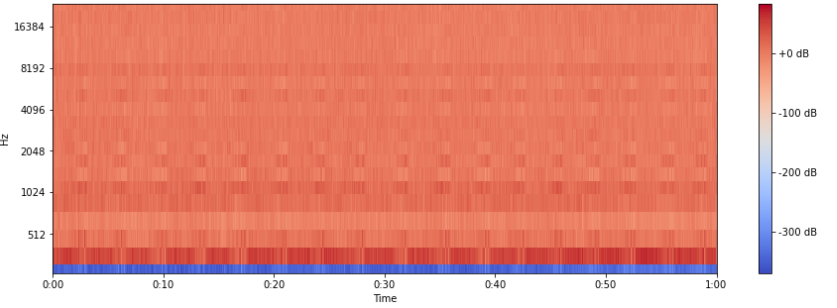
DATASET\_PATH = 'genres\_original'

GENRES = 'classical country disco hiphop jazz metal pop rock'.split()

SAMPLES\_TO\_CONSIDER = 22050 \* 30

Из датасета было выбрано 8 жанров для распознавания. В каждом жанре используется множество записей формата .wav длительностью 30 секунд.

MFCC — это матрица значений, которая захватывает тембральные аспекты. Мел - это единица высоты звука, это та же частотная ось, только выражается в Мелах, вместо Герц. Кепстр - это акустическая волна (параметры голосового тракта, тоновые сигналы, шумы и фильтры). MFCC - Представляют собой набор признаков, которые описывают общую форму спектральной огибающей.

Рисунок 2 – Мел-кепстральный коэффициент (MFCC)

## 

## 3.2 Сбор данных и предобработка

import os

import librosa

import numpy as np

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

from sklearn.preprocessing import LabelEncoder

DATASET\_PATH = 'path\_to\_gtzan\_dataset'

GENRES = 'blues classical country disco hiphop jazz metal pop reggae rock'.split()

SAMPLES\_TO\_CONSIDER = 22050 \* 30 # 30 секунд аудио

def load\_data(dataset\_path, genres):

data = []

labels = []

for genre in genres:

genre\_path = os.path.join(dataset\_path, genre)

for file\_name in os.listdir(genre\_path):

if file\_name.endswith(".wav"):

file\_path = os.path.join(genre\_path, file\_name)

y, sr = librosa.load(file\_path, sr=22050)

if len(y) >= SAMPLES\_TO\_CONSIDER:

y = y[:SAMPLES\_TO\_CONSIDER]

data.append(y)

labels.append(genre)

return np.array(data), np.array(labels)

data, labels = load\_data(DATASET\_PATH, GENRES)

## Извлечение признаков

def extract\_features(y, sr):

mfccs = librosa.feature.mfcc(y=y, sr=sr, n\_mfcc=13)

mfccs\_mean = np.mean(mfccs.T, axis=0)

return mfccs\_mean

features = np.array([extract\_features(y, 22050) for y in data])

label\_encoder = LabelEncoder()

encoded\_labels = label\_encoder.fit\_transform(labels)

X\_train, X\_test, y\_train, y\_test = train\_test\_split(features, encoded\_labels, test\_size=0.2, random\_state=42)

## Построение и обучение модели

import tensorflow as tf

from tensorflow.keras.models import Sequential

from tensorflow.keras.layers import Dense, Dropout

model = Sequential([

Dense(256, activation='relu', input\_shape=(X\_train.shape[1],)),

Dropout(0.5),

Dense(128, activation='relu'),

Dropout(0.5),

Dense(len(GENRES), activation='softmax')

])

model.compile(optimizer='adam',loss='sparse\_categorical\_crossentropy',metrics=['accuracy'])

model.summary()

model.fit(X\_train, y\_train, epochs=30, batch\_size=32, validation\_data=(X\_test, y\_test))

model.save('music\_genre\_classification\_model.h5')

## Оценка модели

loss, accuracy = model.evaluate(X\_test, y\_test)

print(f"Test Accuracy: {accuracy \* 100:.2f}%")

## Классификация и распределение аудиофайлов

import shutil

from tensorflow.keras.models import load\_model

def classify\_and\_move\_files(model, input\_folder, output\_folder, label\_encoder):

for file\_name in os.listdir(input\_folder):

if file\_name.endswith(".wav"):

file\_path = os.path.join(input\_folder, file\_name)

y, sr = librosa.load(file\_path, sr=22050)

if len(y) >= SAMPLES\_TO\_CONSIDER:

y = y[:SAMPLES\_TO\_CONSIDER]

features = extract\_features(y, sr).reshape(1, -1)

genre\_index = model.predict(features).argmax(axis=1)[0]

genre = label\_encoder.inverse\_transform([genre\_index])[0]

genre\_folder = os.path.join(output\_folder, genre)

os.makedirs(genre\_folder, exist\_ok=True)

shutil.move(file\_path, os.path.join(genre\_folder, file\_name))

model = load\_model('music\_genre\_classification\_model.h5')

classify\_and\_move\_files(model, 'path\_to\_new\_music', 'path\_to\_sorted\_music', label\_encoder

## Точность и производительность модели

Оценка точности модели на тестовом наборе данных показала, что точность составляет 78.79%. Это демонстрирует, что предложенная модель способна эффективно классифицировать музыкальные жанры. Дополнительные эксперименты с изменением архитектуры модели и гиперпараметров могут привести к дальнейшему улучшению результатов.

# Заключение

В ходе выполнения итоговой работы было произведено изучение предметной области интерпретации геологических исследований скважин, выполнен обзор подходов и существующих решений. На основании обзора было принято решение о целесообразности применения методов машинного обучения для решения этой задачи.

В процессе изучения данных была выполнена предварительная обработка данных, включающая в себя удаление выбросов и восстановление пропусков.

Был произведен обзор основных методов машинного обучения и их особенностей и отобраны методы для проверки. Для выбранных методов было произведено обучение на имеющихся данных ГИС, произведено предсказание на тестовой выборке и оценена точность.

Было произведено сравнение методов машинного обучения на основании рассчитанных при оценке качества предсказания метрик. Наилучший результат, согласно полученным метрикам, оказался у модели *HistGradientBoosting*.

В ходе исследования был создан прототип программной системы. Для решения задачи был использован язык программирования Python.

# Список использованных источников

1. [Бурков](https://www.chitai-gorod.ru/books/authors/burkov_a/) А. Машинное обучение без лишних слов. [Библиотека программиста](https://www.chitai-gorod.ru/books/publishers/piter/biblioteka_programmista/). – М., 2020. – 192 с.

2. Грас Д. Data Science. Наука о данных с нуля: пер. с англ. – СПб.:  
 БХВ-Петербург, 2017. – 336 с.

7. Мюллер А., Гвидо С. Введение в машинное обучение с помощью Python. Руководство для специалистов по работе с данными. – М., 2017. – 480 с.

8. Плас Дж. В.Python для сложных задач: наука о данных и машинное обучение. – СПб.: Петербург, 2018 – 576 с.

9. Силен Д., Мейсман А., Али М. Основы Data Science и Big Data, Python и наука о данных. – М., 2017. – 336 с.

10. Скиена С. Наука о данных: учебный курс. Пер. с англ. – СПб.: ООО «Диалектика», 2020. – 544 с.

11. Флах П. Машинное обучение. Наука и искусство построения алгоритмов, которые извлекают знания из данных: пер. с англ. А.А. Слинкина. – М.: ДМК Пресс, 2015. – 400 с.

12. Хасти Т., Тибришани Р. Основы статистического обучения: интеллектуальный анализ данных, логический вывод и прогнозирование. – М., 2020. – 768 с.

13. Элбон К. Машинное обучение с использованием Python. Сборник рецептов. – СПб.: БХВ-Петербург, 2020. – 384 с.

# ПРИЛОЖЕНИЕ А

1. **Функция загрузки данных**

def load\_data(dataset\_path, genres):

data = []

labels = []

for genre in genres:

genre\_path = os.path.join(dataset\_path, genre)

for file\_name in os.listdir(genre\_path):

if file\_name.endswith(".wav"):

file\_path = os.path.join(genre\_path, file\_name)

y, sr = librosa.load(file\_path, sr=22050)

if len(y) >= SAMPLES\_TO\_CONSIDER:

y = y[:SAMPLES\_TO\_CONSIDER]

data.append(y)

labels.append(genre)

return np.array(data), np.array(labels)

1. **Функция извлечения данных**

def extract\_features(y, sr):

mfccs = librosa.feature.mfcc(y=y, sr=sr, n\_mfcc=13)

mfccs\_mean = np.mean(mfccs.T, axis=0)

return mfccs\_mean

1. **Функция распределения файлов**

def classify\_and\_move\_files(model, input\_folder, output\_folder, label\_encoder):

for file\_name in os.listdir(input\_folder):

if file\_name.endswith(".mp3"):

file\_path = os.path.join(input\_folder, file\_name)

y, sr = librosa.load(file\_path, sr=22050)

if len(y) >= SAMPLES\_TO\_CONSIDER:

y = y[:SAMPLES\_TO\_CONSIDER]

features = extract\_features(y, sr).reshape(1, -1)

genre\_index = model.predict(features).argmax(axis=1)[0]

genre = label\_encoder.inverse\_transform([genre\_index])[0]

genre\_folder = os.path.join(output\_folder, genre)

os.makedirs(genre\_folder, exist\_ok=True)

shutil.move(file\_path, os.path.join(genre\_folder, file\_name))